IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

José Riquelme
Alicia Troncoso
Salvador García

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 1:
Modelos Predictivos

Identifying ecosystem patterns from time series of anchovy (Engraulis ringens) and sardine (Sardinops Sagax) landings in northern Chile

1st Francisco Plaza
Unidad de Auditoría y Control
Instituto de Fomento Pesquero
Valparaiso, Chile
francisco.plaza.vega@ifop.cl

2nd Rodrigo Salas
Escuela de Ingeniería Biomédica
Universidad de Valparaíso
Valparaiso, Chile
rodrigo.salas@uv.cl

3rd Eleuterio Yáñez
Escuela de Ciencias del Mar
Pontificia Universidad Católica de Valparaíso
Valparaíso, Chile
eleuterio.yanez@pucv.cl

***Abstract*—This work presents a Knowledge Discovery from Data (KDD) approach in time series pattern identification for anchovy and sardine monthly fishery-biological data in northern Chile. Time series, multivariate analysis and data mining techniques, along with technical literature review for results validation are implemented. This approach, achieved an integration between variables, identifying relevant patterns, associated with fisheries abundance fluctuations and strong association with environmental changes such as El Niño and long-term cold-warm regimes between them, establishing predominant time-periods. The latter, establishes groundwork for studying underlying functional relationships that could reduce gaps in the national fisheries research and management policies.**

***Index Terms*—Anchovy, Sardine, Ecosystem, Patterns, Data Mining, Time Series, Knowledge Discovery from Data**

I. Introduction

The Chile-Peru Current System (CHPCS) is a very productive marine system due to nutrient transport by large-scale horizontal advection and persistent coastal upwelling. In Chile the average annual landing in the last 30 years is 4.8 million tons and the pelagic resources of the northern zone (18°21'S-24°S) represent 40% ([2]). In this area the fishery is successively based on anchovy (*Engraulis ringens*) and sardine (*Sardinops sagax*), with notable changes associated with fishing effort and fluctuations in the environment ([3]).

Ocean-climate changes alter marine ecosystems at various scales, in fact, [3] propose an integrative conceptual model of the main local to large-scale phenomena involved in northern Chile. On the intra-seasonal scale, coastal trapped waves are mainly responsible for most of the variability of sea surface temperature and currents on the continental shelf and slope of the CHPCS. On the interannual scale, changes in atmospheric and oceanographic conditions are mainly associated with the El Niño Southern Oscillation (ENSO) phenomenon. Interdecadal long-term regime shifts (warm or cold), would influence the reorganization of marine communities and trophodynamic relationships, inducing changes in dominant species. Thus, the link between the variation of anchovy abundance and environmental changes at different spatio-temporal scales opens the possibility for predicting fluctuations in landings in the short, medium, and long term, one of the main objectives of fisheries management ([4]; [5]). Therefore, the main objective for this work is to identify patterns of interaction between those variables in northern Chile.

II. Materials and methods

A. Data

The study analyzes monthly data considering local and global environmental variables and fishing variables from 1963-2011 in northern Chile (18°21'S-24°S and from the coast until 73°O). This area is where industrial purse seine fleet operates in northern Chile. Table I summarize all considered variables.

TABLE I
Summary of analyzed variables

Type	*Variable*	*Desctiption*
Local	SST	Sea Surface Temperature from Antofagasta coastal oceanographic station
	TI	Turbulence Index from Antofagasta coastal oceanographic station
	MSL	Mean Sea Level from Antofagasta coastal oceanographic station
Global	MEI	Multivariate ENSO Index
	PDO	Pacific Decadal Oscillation index
	N12	Climatic Index in the Niño12 area
	N34	Climatic Index in the Niño34 area
	SOI	Southern Oscillation Index
	CTI	Cold Tongue Index
Fisheries	VANC	% days per month of anchovy fishing prohibition (fisheries ban)
	VSAR	% days per month of sardine fishing prohibition (fisheries ban)
	LANC	Anchovy landings in northern Chile
	LSAR	Sardine landings in northern Chile

B. General procedure

For this study, a similar approach to [6] is considered, involving as a first step an inspection of the multivariate time series, using signal decomposition (trend, seasonality and noise), then Principal Components Analysis is performed to reduce dimensionality and to discover preliminary patterns (which are contrasted with field literature). Finally, as a third step these patterns are grouped, using k-means clustering, in order to obtain different ecosystem periods, which allows to identify changing trends (increases or decreases) of anchovy and sardine landings, associated to environmental conditions. Integrated graphical representations are then implemented to show discovered patterns.

III. Results and discussion

Figure 1 shows the additive decomposition performed on the 4 PCs, showing the trend, seasonality and irregular component.

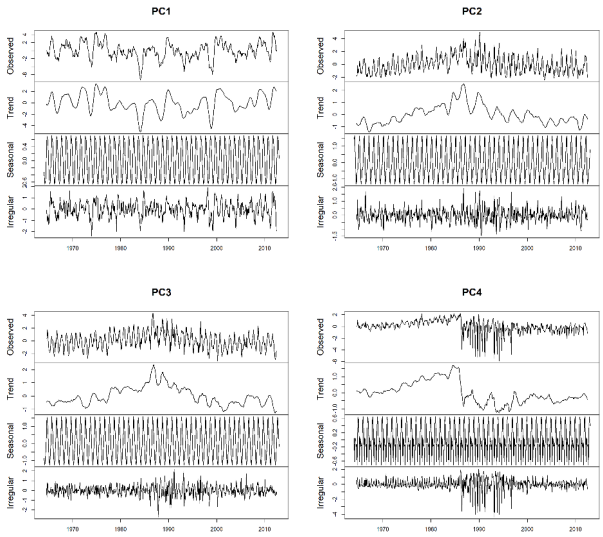


Fig. 1. Decomposition of the selected PCA components.

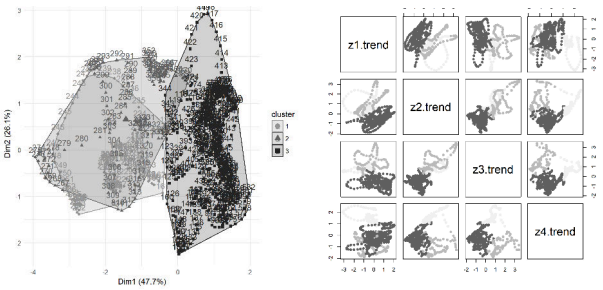


Fig. 2. Clustering performance and results. Representation of clusters centroids (left), classification of the principal components (right).

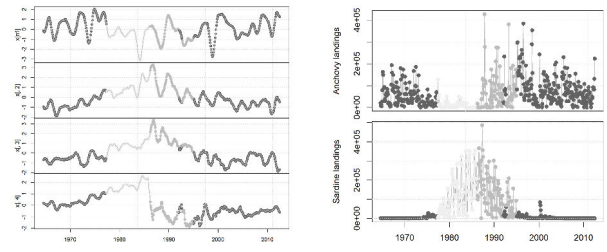


Fig. 3. Cluster classification applied to data. Cluster classification of PCA selected components (left), Cluster classification of anchovy and sardine landings time series (LANC and LSAR)(right).

In order to address the problem, a series of research questions were established to guide the analysis. To summarize the results obtained in this study, the answer for each research question is presented as follows:

1) *what is the relationship between the time series between sardine and anchovy?*. Results show an alternance between anchovy and sardine time series. The latter can be observed in both the time series of each fishery and in the results obtained from the clustering process (Fig. 3, having an anchovy period, then a transition period, followed by a sardine period, and finally an anchovy period, respectively.
2) *what is the relationship between environmental variables and the anchovy-sardine?*. As discussed before, the first 4 components from the PCA have higher correlation with environmental variables. Also, the clustering process identified 3 clusters which can be related to environmental long-term cold-warm-cold periods(Fig. 2). Furthermore, those long-term periods are related with anchovy and sardine landings presence or absence. The underlying processes that describe functional relationships between particular environmental variables and anchovy and sardine behavior are still being discussed among the scientific community. However, this study proposes a more integrated scope, considering the ecosystem as a whole.
3) *Are there any procedures that allow to identify patterns?*. A process of mixing three well-known techniques (time series decomposition, principal components analysis and clustering) was successfully implemented in order to identify ecosystemic patterns.
4) *Are there identifiable patterns in the time series of anchovy, sardine, and environmental variables?*. Four clear periods could be identified, from 1963-1977, 1977-1986, 1986-1995 and 1995-2011, having an anchovy period, then a transition period, followed by a sardine period, and finally an anchovy period, respectively (Fig. 3).

Acknowledgment

This paper is presented as an overview of the work done by [1].

References

References

[1] F. Plaza, R. Salas, and E. Yáñez, "Identifying ecosystem patterns from time series of anchovy (Engraulis ringens) and sardine (Sardinops sagax) landings in northern Chile," Journal of Statistical Computation and Simulation, vol.88(10), pp. 1863–1881.
[2] SERNAPESCA. Anuarios Estadísticos de Pesca. Servicio Nacional de Pesca y Acuicultura, Chile; 1950-2012.
[3] Yáñez E, Hormazábal S, Silva C, et al. Coupling between the environment and the pelagic resources exploited off northern Chile: ecosystem indicators and a conceptual model. Latin American journal of aquatic research. 2008 00; 36: 159–181.
[4] Zhou S, Smith ADM, Punt AE, et al. Ecosystem-based fisheries management requires a change to the selective fishing philosophy. Proceedings of the National Academy of Sciences. 2010;107(21):9485–9489.
[5] Chavez FP, Ryan J, Lluch-Cota SE, et al. From anchovies to sardines and back: multidecadal change in the pacific ocean. Science. 2003; 299(5604): 217–221.
[6] Fayyad U, Piatetsky-Shapiro G, Smyth P. From Data Mining to Knowledge Discovery in Databases. AI Magazine. 1996; 3(17): 37–54.

Análisis Big Data para la Respuesta a la Demanda en el Mercado Eléctrico

José Antonio Fábregas,
José María Luna-Romera
and José C. Riquelme Santos
Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Sevilla, España

Ángel Arcos-Vargas
Organización Industrial y Gestión de Empresas
Universidad de Sevilla
Sevilla, España

Javier Tejedor Aguilera
Endesa S.A.
Madrid, España

***Abstract*—El modelo de negocio tradicional de las compañías energéticas está cambiando los últimos años. La introducción de los contadores inteligentes ha conllevado un aumento exponencial del volumen de datos disponibles, y su análisis puede ayudar a encontrar patrones de consumo entre los clientes eléctricos para reducir costes y proteger el medioambiente. Las centrales generan energía eléctrica para poder cubrir los picos de consumo en momentos puntuales. Un conjunto de técnicas denominadas "demand response" intenta dar solución a este problema usando propuestas de inteligencia artificial. En este documento se propone una metodología para el procesado de los grandes volúmenes de datos como los que generan los contadores inteligentes. Tanto para el preprocesado como para la optimización y realización de este análisis se utilizan técnicas big data. En concreto, una versión distribuida del algoritmo k-means y de varios índices de validación interna de clustering para big data en Spark. Los datos de origen corresponden los consumos de clientes eléctricos de Cataluña durante el año 2016. El análisis de estos consumidores realizado en este trabajo ayuda a su caracterización. Este mayor conocimiento sobre los hábitos de consumos y tipos de clientes, puede facilitar a las compañías eléctricas la labor.**

***Index Terms*—Big data, respuesta a la demanda, clustering, contadores inteligentes, consumo eléctrico**

I. Introducción

Durante la mayor parte del siglo XX, la relación entre los usuarios de electricidad y las empresas de distribución se mantuvo sin cambios. No se eligieron proveedores y, por lo tanto, no había necesidad de tratar a los consumidores como clientes. Sin embargo, la desregulación, la agenda verde y los continuos saltos tecnológicos han cambiado esta relación. Nuevas limitaciones como la seguridad del suministro, la competitividad y la sostenibilidad son los tres ejes prioritarios para cambiar el modelo energético actual, que pueden lograrse mediante objetivos como la reducción de las emisiones y la mejora de la generación de energía renovable y la eficiencia energética.

Una herramienta esencial en este nuevo modelo son los llamados "contadores inteligentes", que no deben entenderse sólo como dispositivos que miden el consumo sino como verdaderos sensores para una red eléctrica. Estos sensores facilitan una red altamente flexible y adaptable que integra de forma inteligente las acciones de los usuarios que se conectan a ella para conseguir un suministro eficiente, seguro y sostenible.

Uno de los principales problemas del sector eléctrico es la necesidad de disponer de capacidad de generación y de una red sobredimensionada para cubrir los picos de alto consumo de los clientes en determinados momentos. Sin embargo, existen soluciones basadas en la adaptación de la demanda a la energía disponible en lugar de aumentar la oferta para satisfacer la demanda. Esto se denomina "Respuesta a la Demanda" y su objetivo es cambiar los hábitos de consumo de electricidad de los clientes en respuesta a los cambios en los precios del suministro. El principal inconveniente es el gran volumen de información disponible en estas redes, ya que sólo puede manejarse con técnicas big data.

Nuestra propuesta se basa en el procesado de estos grandes conjuntos de datos de manera distribuida y paralela. En particular, la aplicación de técnicas de minería de datos para comprender mejor los patrones de consumo de los clientes. Por un lado, haremos uso de HDFS [1] para el almacenamiento de datos distribuido. Mientras que el procesado lo realizaremos con Spark [2], una plataforma de computación distribuida y paralela. En concreto, utilizaremos la implementación del algoritmo k-means de la librería MLlib [3] de Spark, así como cuatro índices de validación de clustering para big data [4]. Este estudio podría ayudar a la planificación de las conexiones de las fuentes de energía renovables a la red, con un doble objetivo: la reducción de precios y la sostenibilidad medioambiental.

La estructura de este artículo es la siguiente. En la sección II se describen los trabajos relacionados. En la sección III se detallan las características del dataset. En la sección IV se muestran los resultados de los experimentos realizados para preprocesar los datos y aplicar técnicas de clustering. Para finalizar, en la sección V se presentan las conclusiones de la investigación realizada.

II. Trabajo relacionados

La irregularidad de la demanda de electricidad es uno de los principales problemas del sector. Esto se debe a que las compañías eléctricas deben tener tanto una capacidad de generación sobredimensionada, como redundancia de la red para hacer frente a grandes cantidades de demanda que sólo se requieren unas pocas horas al año. Normalmente, se establece un umbral del 20% para la generación de electricidad latente, que debe cubrir aproximadamente el 5% del tiempo de servicio de la red (pico de demanda) [5]. Algunos de los recursos para resolver este problema necesitan la implicación de los usuarios. Estas soluciones se estudian bajo el nombre de 'respuesta a la demanda' (Demand Response DR) [6]. En contraste con las ideas convencionales de aumentar la oferta para satisfacer la demanda, las soluciones apuntan a satisfacerla con la energía disponible.

El objetivo es cambiar las pautas de consumo de energía de los clientes en respuesta a los cambios en los precios ofrecidos. Esto permitirá a las compañías eléctricas gestionar mejor la demanda con un mejor ajuste de las predicciones y una reducción del coste de la energía para los clientes. Existen múltiples iniciativas de posibles esquemas de fijación de precios, que en algunos casos incluso mantienen los beneficios para las empresas proveedoras [7]. Una de las principales ventajas de la respuesta a la demanda es ofrecer una opción sostenible con una generación de energía más volátil. Sobre todo en España, donde existe una alta presencia de fuentes de generación renovables. Para implementar los mecanismos de respuesta a la demanda, las redes eléctricas deben evolucionar a una infraestructura que permita el flujo de información entre los diferentes participantes del sistema eléctrico. En este campo es donde los grandes datos se convierten en una tecnología esencial para analizar este flujo de información y convertirlo en conocimiento útil.

Estos datos de consumo de los clientes, obtenidos mediante contadores inteligentes, no son sino múltiples series temporales. El análisis de series temporales puede entenderse como una secuencia de valores observados a lo largo del tiempo y ordenados cronológicamente [8]. Como el tiempo es una variable continua, las muestras se registran en puntos sucesivos igualmente espaciados. Por lo tanto, las series temporales son una secuencia de datos de tiempo discreto.

En el contexto de la minería de datos de series temporales, el principal desafío es cómo representar los datos. El enfoque más común es transformar las series temporales en otro ámbito para la reducción de la dimensionalidad y desarrollar un mecanismo de indexación. La medida de la similitud entre las sub-secuencias de series temporales y la segmentación son las dos tareas principales en la minería de series temporales que corresponden con las tareas clásicas de la minería de datos. El uso cada vez mayor de datos de series cronológicas ha dado lugar a una gran cantidad de intentos de investigación y desarrollo en el campo de la minería de datos [9].

En este trabajo nos centraremos en el clustering, un método de minería de datos para agrupar instancias no etiquetadas de conjuntos de datos. La idea es que las instancias recogidas en un mismo grupo tendrán un comportamiento similar [10]. El clustering de series temporales surge como un enfoque útil para minar patrones comunes a partir de datos dependientes del tiempo [11] que se caracterizan por tener una alta dimensionalidad y un gran tamaño.

Centrándonos en el clustering a partir de datos de consumo de energía, son muchas las propuestas enmarcadas en este campo: En [12] se presenta el efecto de las medidas de similitud en la aplicación de la agrupación para descubrir los patrones energéticos de los edificios. Para obtener perfiles de carga típicos de los clientes, en [13] se propone un índice de estabilidad para elegir el algoritmo de agrupamiento que mejor se adapte a este problema de reconocimiento de patrones. Además, se propone otro índice de prioridad (basado en el índice de estabilidad) para determinar el rango de prioridad de los agrupamientos. En [14] desarrolla una técnica de clustering de particiones para extraer información útil de los precios de la electricidad. Mientras que en [15] se usan técnicas de clustering con el objetivo de agrupar y etiquetar las muestras de un conjunto de datos para pronosticar el comportamiento de las series temporales basadas en la similitud de las secuencias de patrones.

En relación a la gestión inteligente de la demanda de electricidad, en [16] los autores proponen un Virtual Power Player como gestor para satisfacer la demanda y reserva de energía eléctrica requerida. En [17] se presenta un análisis de los datos de los contadores inteligentes de los clientes para comprender mejor la demanda máxima y las principales fuentes de variabilidad en su comportamiento.

Además de los métodos clásicos de gestión de datos, el enfoque de big data ha surgido recientemente debido a la disponibilidad de una gran cantidad de datos, sistemas de ficheros distribuidos, y potentes motores de procesamiento distribuido. Esto ha propiciado que muchos de los algoritmos de minería de datos se hayan adaptado al entorno de big data, como por ejemplo los algoritmos de clustering. En cuanto al campo del consumo de energía, en la actualidad han surgido varias grandes soluciones de datos, como las optimizaciones de redes inteligentes en [18] y los patrones de consumo energético en [19].

III. Características del dataset

Los datos originales se encuentran almacenados en 34 tablas, divididas en cientos de archivos CSV. Estas tablas contienen una amplia información sobre consumos, tarifas, contadores, datos geográficos o personales de consumidores eléctricos de Endesa en Cataluña entre 2010 y 2016 para un tamaño total de 1,8 TB.

Los clientes en los que centramos este estudio son aquellos que poseen tarifas 2.0A y 20DHA. Ambas del mercado libre de Endesa para potencias contratadas menores a 10kW, donde se encuentran inmensa mayoría de hogares y pequeños locales. La tarifa 2.0A mantiene un precio fijo durante todo el año, mientras que la 20DHA tiene discriminación horaria de dos periodos. En esta última, los periodos punta y valle marcan dos precios distintos según la hora en la que se consuma la energía: punta de 12h a 22h en invierno y de 13h a 23h en verano; valle de 22h a 12h en invierno y de 23h a 13h en verano.

En la Tabla Ia se observa la distribución de los clientes según su tarifa. Debido a la gran variedad de potencias contratadas, La Tabla Ib muestra una distribución de los clientes por cada rango de potencia. Estos rangos están basados en los valores estándar de potencia que actualmente pueden contratarse al tener contadores inteligentes.

TABLE I: Distribución de clientes

(a) Por tarifa contratada

Tarifa	Clientes
2.0A	102,123
2DHA	6,614

(b) Por potencia contratada

Potencia(kW)	Clientes
[0.1-2.3)	8,972
[2.3-3.45)	21,969
[3.45-4.60)	38,578
[4.60-5.75)	22,328
[5.75-6.90)	8,772
[6.90-8.05)	3,134
[8.05-9.20)	4,946
[9.20-10.0)	82

IV. Análisis realizados

En esta sección se presentan los resultados de los análisis realizados. En el apartado IV-A se detalla el procesado de los datos En el apartado IV-B se analizan cuatro índices de validación de clustering obtener el k óptimo. En el apartado IV-C se muestran los clusters obtenidos con el algortimo k-means. Por último, en el apartado IV-D se realiza una evaluación de los resultados.

Para la realización de los experimentos se han utilizado los siguientes entornos:

- Cluster propio: 72 procesadores Intel Xeon E7-4820, 128 GB RAM y 8 TB de almacenamiento.
- Cluster de EMR (Elastic Map Reduce) de AWS: cinco instancias de m3.2xlarge con 16 procesadores Intel Xeon E5-2670 v2 (Ivy Bridge), 30 GB RAM y 2 SSDs DE 80 GB cada una.

A. Procesado de datos

El primer objetivo es conseguir un conjunto de datos que tenga sentido minar. Primero, almacenaremos la gran cantidad de datos de los que disponemos en un sistema de ficheros distribuido (HDFS) configurado en nuestro cluster. Para reducir el tamaño de los datos, los ficheros CSV se pasan a Parquet, un formato de datos orientado a columnas que los comprime y codifica. Una vez almacenados y formateados, todo el procesamiento de estos datos se realiza de forma distribuida y paralela con Spark. Debido a que estos datos también se procesarán con herramientas online de Amazon Web Service, se almacenan además en el S3, el sistema de almacenamiento en línea de Amazon. Posteriormente se estudian y seleccionan los atributos necesarios para construir un primer dataset. Las tablas procesadas se muestran a continuación (Tabla II):

TABLE II: Tablas procesadas

Tabla	Elementos (millones)	Tamaño (MB)
Clientes	20.6	716
Contratos	40.6	560
Maestro Contratos	33.1	666
Curvas de carga	2,094.44	340,992

Una vez construido este primer dataset, se seleccionan los usuarios con: una potencia contratada igual o inferior a 10kW, una tarifa 2.0A o 20DHA, y que tengan con todas las lecturas de consumo del año 2016. Por último, se descartan las instancias con valores nulos. Este dataset está compuesto por 47,829,235 instancias correspondientes a las 365 curvas de carga de 2016 de 131,039 clientes. De cada consumidor, tenemos 24 lecturas de consumo horario para cada una de sus 365 instancias.

Para construir las series temporales de 2016 es necesario transformar este dataset de instancias diarias en uno de instancias anuales .De esta forma, el nuevo dataset constaría de 131,039 instancias, una por cliente, con: 8760 (365x24) lecturas horarias de 2016, la cups (Código Universal del Punto de Suministro), la tarifa y la potencia contratadas.

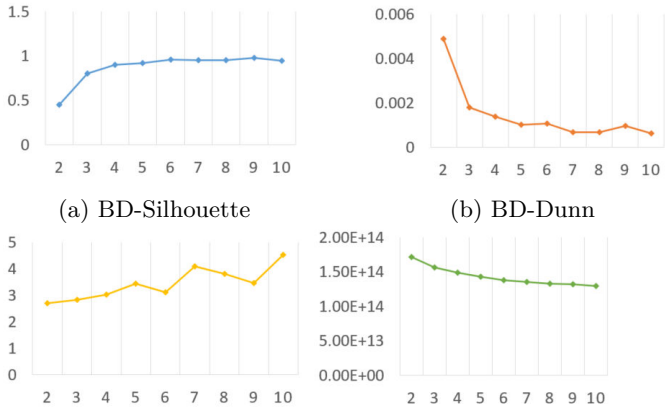
A continuación generamos un nuevo atributo con el que categorizamos a los consumidores en función a la potencia contratada. Por último, construimos un dataset alternativo con diferencias de consumo normalizadas: hallamos la diferencia entre cada par de valores de consumo consecutivo y la dividimos por la media de consumo de ese día. De esta forma tendremos un dataset con los consumos horarios de 2016 y otro con las diferencias normalizadas de consumo del mismo periodo. Estos datasets se utilizarán tanto de forma conjunta como individual en los siguientes apartados con el objetivo de encontrar patrones en los hábitos de consumo eléctrico de los clientes.

B. Determinación del número óptimo de clusters

Antes de aplicar algoritmos de clustering a nuestros datasets es necesario determinar cuál es el número óptimo de clusters (k) a obtener. Para ello, aplicamos a cada uno de los datasets cuatro índices de validación de clustering para big data (BD-CVIs) [4]: BD-Silhouette [4], BD-Dunn [4], Davies-Bouldin [20] y Within Set Sum of Square Errors (WSSSE) [21].

En la Figura 1a se muestra la representación gráfica resultados índice BD-Silhouette. Para este índice los valores óptimos de k son sus máximos, 6 y 9. Estos valores coincides con los máximos de la gráfica correspondiente al índice BD-Dunn (Figura 1b). En el caso del índice Davies-Bouldin los valores óptimos se encuentran en los mínimos,

que vuelven a coincidir en 6 y 9, tal y como se observa en la Figura 1c. Por último, los resultados del índice WSSSE representados en la Figura 1d no arrojan un valor claro. En este índice buscamos una estabilización de valores y, como podemos ver, no hay un valor concreto en lo que esto ocurra. Tras analizar estos resultados, hemos obtenido los valores 6 y 9 como óptimos para la realización del clustering.



(a) BD-Silhouette (b) BD-Dunn

(c) Davies-Bouldin (d) WSSSE

Fig. 1: Índices big data de validación de clustering

Al igual que para el dataset de comsumos, hemos vuelto a aplicar estos índices al dataset de diferencias normalizadas comentado al final del apartado IV-A. En este caso, los resultados para el valor óptimo de k fueron 5 y 7.

C. Clustering

Una vez calculado el número óptimo de clusters, aplicamos a cada dataset la versión implementada en Spark del algoritmo k-means. Esta implementación fue desarrollada para poder extraer patrones en sistemas paralelos y distribuidos. A la hora de ejecutar el algoritmo, le daremos como entrada el objeto RDD (Resilient Distributed Dataset) y el k obtenido anteriormente. Como resultado obtendremos una serie de clusters con elementos de cada uno de los datasets.

Para el dataset de consumos, dos de los clusters obtenidos con k=9 tenían menos de 5 elementos, por lo que se decidió trabajar con el otro valor óptimo obtenido de k=6. En el caso del dataset de diferencias normalizadas, dos de los clusters para k=5 contenían un único elemento. Por este motivo, se optó por tomar el valor k=7.

La distribución de los elementos en los 6 clusters del dataset de consumos se muestra en la Tabla IIIa. De la misma forma, en la Tabla IIIb podemos ver cómo se distribuyen los elementos correspondientes a los 7 clusters del dataset de diferencias normalizadas.

En la Figura 2 se representan las curvas de consumo horario formadas por los centroides de cada uno de los clusters durante una semana de enero de 2016. En ella destaca que la mayoría de los consumidores, agrupados

TABLE III: Clusters de los datasets para k óptimo

(a) Dataset de consumos para k=6

Cluster	Elementos
0	12,029
1	50,643
2	1002
3	138
4	1116
5	43,789

(b) Dataset de diferencias normalizadas para k=7

Cluster	Elementos
0	42,276
1	8241
2	2544
3	18,994
4	28,462
5	7191
6	1029

en los clusters 1 y 5, tienen consumos inferior a 1 kWh. También puede observarse como un grupo reducido de clientes, que conforma el cluster 3, tiene un consumo muy alto por la noche (de 19 a 8 horas) y prácticamente nulo durante el día.

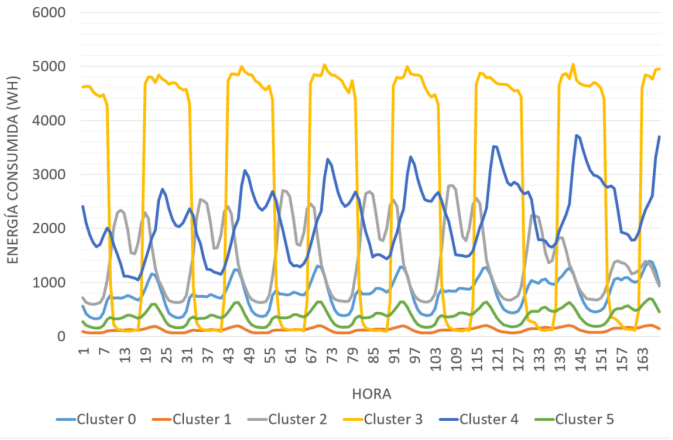


Fig. 2: Curvas de consumo horario de los centroides durante una semana de enero

En la Figura 3 se representan las las curvas de los mismos centroides durante una semana de julio. Se observa que los consumidores de los clusters 0, 1 y 5 mantienen un consumo prácticamente igual al de enero, aunque los picos máximo se acercan más a la media noche. Sin embargo, los del cluster descienden de forma radical hasta equipararse a los del 0. También destacan las curvas de consumo del cluster 3, donde las horas de alto consumo se reducen a 6 (00 a 6 horas).

Por su parte, la Figura 4 muestra las curvas de las diferencias horarias normalizadas en el mismo periodo de tiempo que la Figura 2. En ella podemos observar como los mayores picos de diferencias de consumo entre horas pertenecen a los elementos de los clusters 1,2,5 y 6. Estos clusters resultan ser los menos numerosos, representando un 17% del total de los consumidores. Lo que nos indica que el consumo de la mayoría de los usuarios a lo largo del día mantiene cierta uniformidad.

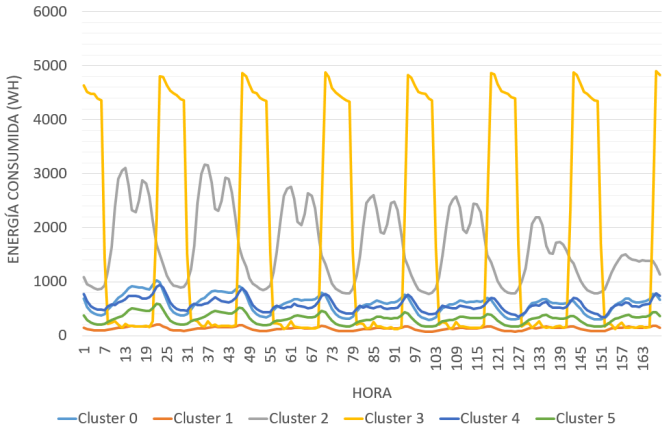


Fig. 3: Curvas de consumo horario de los centroides durante una semana de julio

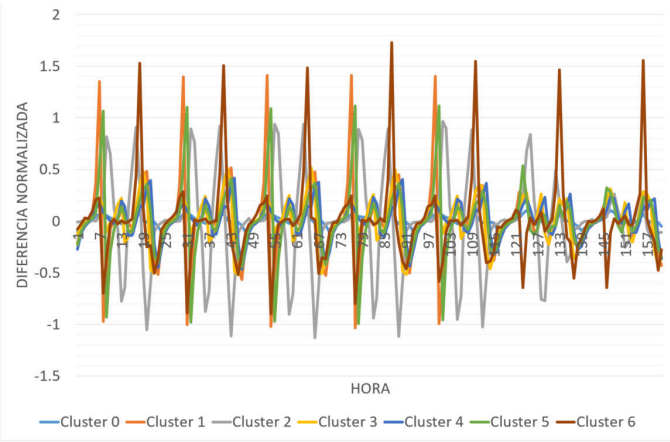


Fig. 4: Curvas de diferencias horarias normalizadas de los centroides

D. *Evaluación de resultados*

1) Evaluación no supervisada: En esta última fase evaluaremos los resultados obtenidos tras aplicar el algoritmo k-means a los datasets. Para ello, se generan las tablas de contingencia entre los clusters obtenidos para los distintos datasets, cruzando los resultados entre los mismos. Esto nos permite encontrar patrones que definan a los consumidores en relación a las características de los diferentes conjuntos de datos.

La tabla de la Figura 5 muestra por filas los 6 clusters del dataset de consumos (C0 a C5), y por columnas los 7 obtenidos del dataset de diferencias normalizadas (D0 a D6). Estos valores representan los porcentajes relativos al total de cada fila. Es decir, el porcentaje de consumidores de cada cluster de consumos presente en cada uno de los clusters de diferencias normalizadas.

En la Figura 5 destaca que casi la totalidad del cluster C3 está formado por consumidores del cluster D6. Si atendemos a la Figura 2 esto caracterizaría a un grupo de clientes con un consumo nocturno muy alto. Si además

		CLUSTER DIFERENCIAS NORMALIZADAS							
		D0	D1	D2	D3	D4	D5	D6	
CLUSTER CONSUMOS	C0	29.11%	10.20%	3.85%	19.03%	27.53%	9.85%	0.43%	*100.00%*
	C1	53.06%	5.25%	1.87%	13.03%	20.61%	4.74%	1.44%	*100.00%*
	C2	52.20%	1.60%	27.64%	7.98%	7.29%	3.19%	0.10%	*100.00%*
	C3	5.07%	0.00%	0.00%	0.00%	0.00%	0.72%	94.20%	*100.00%*
	C4	56.16%	7.83%	0.09%	7.13%	16.99%	8.36%	3.43%	*100.00%*
	C5	24.51%	9.70%	1.96%	22.71%	32.99%	7.94%	0.18%	*100.00%*

Fig. 5: Tabla de contingencia de valores relativos al total de cada fila

nos fijamos en la Figura 4, observamos que tienen un mínimo y un máximo en las diferencias horarias que se mantiene constante los 7 días de la semana. Esto podría indicar un perfil de consumidor muy concreto, con un consumo energético nocturno alto y uniforme. Por otro lado, podemos observar que la mitad cluster C1, el más numeroso, está formado por consumidores del cluster D0, también el más numeroso. Aquí se identifica un amplio grupo de usuarios de consumo bajo y con pocos cambios, cuyas mayores variaciones en el consumo se producen a las 8h.y 18h.

A continuación vamos a analizar el cluster C5, de un tamaño similar al C1 y con un consumo un poco más alto. En este caso, sólo se compone en un 24.51% de consumidores del cluster D0. Sin embargo, aumenta considerablemente el número de elementos pertenecientes a los clusters D3 y D4. En ellos, las diferencias de consumo horario (algo más altas que en el anterior) se concentran en las 19h. en los consumidores del cluster D3 y las 21 h. en los de D4. Por lo que podemos interpretar que los clientes con un consumo algo mayor que los del cluster C1 son más heterogéneos en cuanto a su comportamiento.

Ahora analizaremos los resultados desde la otra perspectiva. En la Figura 6 se representan los porcentajes relativos al total de cada columna. Es este caso, el porcentaje de consumidores de cada cluster de diferencias normalizadas presente en cada uno de los clusters de consumos.

		CLUSTER DIFERENCIAS NORMALIZADAS						
		D0	D1	D2	D3	D4	D5	D6
CLUSTER CONSUMOS	C0	8.28%	14.89%	18.20%	12.05%	11.63%	16.48%	5.05%
	C1	63.56%	32.29%	37.15%	34.74%	36.68%	33.39%	70.65%
	C2	1.24%	0.19%	10.89%	0.42%	0.26%	0.45%	0.10%
	C3	0.02%	0.00%	0.00%	0.00%	0.00%	0.01%	12.63%
	C4	1.51%	1.08%	0.04%	0.43%	0.68%	1.32%	3.79%
	C5	25.39%	51.55%	33.73%	52.36%	50.76%	48.35%	7.77%
		100.00%	*100.00%*	*100.00%*	*100.00%*	*100.00%*	*100.00%*	*100.00%*

Fig. 6: Tabla de contingencia de valores relativos al total de cada columna

Si nos fijamos en la Figura 6, lo primero que podemos destacar es que todos los clusters de diferencias normalizadas están compuestos mayoritariamente por los clusters C1 y C5. Algo normal debido entre los dos aglutinan el 86% del total de consumidores. Los clusters D0 y D1 tienen una mayoría de elementos de C1, mientras que los clusters D1, D3, D4 y D5 tienen una mayor presencia de elementos del C5. Por otro lado, en el D2 los clusters C1

y C5 no tienen una representación tan alta debido al 10% de elementos del cluster C2.

Centrándonos en el cluster D6, vemos que el 70.65% de sus clientes pertenecen al cluster C1. Mientras que observando la Figura 5 se denota que sólo el 1.44% de los del cluster C1 de consumos pertenecen al cluster D6.

2) Evaluación semi-supervisada: En este apartado tomaremos como referencia el rango de potencia contratada al que pertenece cada consumidor. Por lo que cruzamos elementos de los clusters del dataset de comsumo (Ver Tabla IIIa) con los de cada rango de potencia (ver Tabla Ib). El objetivo es encontrar relaciones entre los distintos tipos de consumidores y sus potencias contratadas. En la tabla de contingencia de la Figura 7 se representan los clusters del dataset de consumos por filas (C0 a C5) y los rangos de potencia por columnas (P1 a P8). Estos valores representan los porcentajes relativos al total de cada fila. Es decir, el porcentaje de consumidores de cada cluster de consumos presente en cada uno de los rangos de potencia.

		RANGO DE POTENCIA								
		P1	P2	P3	P4	P5	P6	P7	P8	
CLUSTER CONSUMOS	C0	2.65%	9.51%	27.46%	23.09%	17.67%	6.66%	12.80%	0.17%	*100.00%*
	C1	13.76%	26.37%	34.08%	17.74%	4.43%	1.50%	2.08%	0.03%	*100.00%*
	C2	2.69%	5.29%	12.28%	16.87%	14.87%	19.56%	26.85%	1.60%	*100.00%*
	C3	3.62%	0.72%	10.87%	21.74%	10.14%	47.10%	4.35%	1.45%	*100.00%*
	C4	4.15%	9.01%	12.54%	15.19%	26.50%	19.08%	12.90%	0.62%	*100.00%*
	C5	3.67%	16.70%	40.50%	23.28%	8.89%	2.50%	4.41%	0.05%	*100.00%*

Fig. 7: Tabla de contingencia de valores relativos al total de cada fila

Si observamos los clusters C1 y C5, los de menor consumo (ver Figura 2), comprobamos que la mayoría de sus elementos pertenece a los rangos P1 a P4, las potencias bajas. Mientras, en los clusters C2 y C4 aumentan de forma considerable los consumidores con potencias más altas (P5, P6 y P7). Como es lógico, la mayoría de los clientes con consumo bajo tienen contratadas potencias medias y bajas. Además, más de la mitad de los clientes con un consumo alto contrató una potencia alta. Un caso particular es el C3, donde casi la mitad de sus elementos pertenecen a P6. Esto indica que cerca del 50% de los clientes con alto consumo nocturno tienen contratados entre 6.90 y 8.05kW.

Ahora vamos a analizarlo desde el punto de vista de la potencia contratada. Si observamos la Figura 8, se representan los porcentajes relativos al total de cada columna. Es decir, el porcentaje de consumidores de cada rango de potencia presente en cada uno de los clusters de consumos.

Podemos ver como más del 85% de los consumidores de los rangos P1 a P4 pertenecen a los clusters C1 y C5. Por lo que la relación entre estos grupos de clusters y rangos de potencia existe en ambos sentidos: Clientes con bajo consumo contrató un nivel potencia baja o media y viceversa. En el análisis de la Figura 7 vimos como los clientes con alto consumo tenían potencias altas contratadas. Pero, si nos fijamos en los rangos P5 a P8,

		RANGO DE POTENCIA							
		P1	P2	P3	P4	P5	P6	P7	P8
CLUSTER CONSUMOS	C0	3.54%	5.20%	8.55%	12.42%	24.34%	25.53%	31.10%	24.39%
	C1	77.66%	60.80%	44.75%	40.25%	25.72%	24.31%	21.33%	15.85%
	C2	0.30%	0.24%	0.32%	0.76%	1.71%	6.25%	5.44%	19.51%
	C3	0.06%	0.00%	0.04%	0.13%	0.16%	2.07%	0.12%	2.44%
	C4	0.52%	0.46%	0.37%	0.77%	3.44%	6.89%	2.95%	8.54%
	C5	17.92%	33.28%	45.98%	45.66%	44.63%	34.94%	39.06%	29.27%
		100.00%	*100.00%*	*100.00%*	*100.00%*	*100.00%*	*100.00%*	*100.00%*	*100.00%*

Fig. 8: Tabla de contingencia de valores relativos al total de cada columna

observamos que éstos también están compuestos entre un 45% y un 70% por clientes con bajo consumo. Por lo que, aunque los clientes con alto consumo tienen potencias altas contratadas, la mayoría de consumidores con estas potencias tienen un bajo consumo. Al igual que antes, nos vamos a centrar el caso de P6 y C3. Mientras que el 47.1% de los consumidores de C3 pertenecían a P6, sólo el 2.07% de los de P6 se encuentran en C3. Esto refuerza la conclusión anterior, ya que más del 85% de los consumidores con potencia entre 6.9 y 8.05kW tienen consumos que rondan entre los 0.5 y 1.5kWh.

V. Conclusiones y trabajo futuro

En este trabajo se ha presentado la aplicación de técnicas big data para el análisis de datos de consumidores eléctricos. La caracterización de estos clientes obtenida como resultado da lugar a las siguientes conclusiones:

- Más del 85% de los clientes presentan curvas de carga de consumo donde los valores máximos no superan el kWh.
- Existe grupo de 138 clientes con un consumo nocturno muy alto. Y, aunque la cantidad de horas de consumo durante el verano es muy inferior al invierno, siempre se producen en periodos valle. El 97.8% de estos usuarios tienen contratada una tarifa adaptada a sus consumos, con discriminación horaria.
- En su inmensa mayoría, los clientes con un bajo consumo tenían contratada una baja potencia y viceversa.
- Los usuarios con un consumo medio-alto contrataron una potencia media-alta. Sin embargo, el 77.32% de los clientes que contrataron estos niveles de potencia, consumió valores de energía que no llegaron a 1kWh. Por lo que más de 3/4 partes de estos clientes tienen potencias contratadas muy por encima de lo que necesitan.
- Durante todo el año, los picos de consumo que se alcanzan por las mañanas y al mediodía se producen en horarios valle. Lo mismo ocurre con los picos nocturnos en verano, ya que estos aparecen entre las 23 y la 1. Esto indica que las tarifas de discriminación horaria podrían ser beneficiosas para los clientes con estos hábitos de consumo. Sin embargo, sólo el 5% (4,784 de 95,569 clientes) tienen contratada este tipo de tarifa.

- En trabajos futuros se caracterizará a los consumidores en función de sus consumos y tarifas. Además, se analizarán y recomendarán las tarifas y potencias óptimas a contratar de forma personalizada para cada tipo de cliente.

References

[1] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, May 2010, pp. 1–10.

[2] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," vol. 10, pp. 10–10, 07 2010.

[3] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar, "Mllib: Machine learning in apache spark," *Journal of Machine Learning Research*, vol. 17, no. 34, pp. 1–7, 2016.

[4] J. M. Luna-Romera, J. García-Gutiérrez, M. Martínez-Ballesteros, and J. C. Riquelme Santos, "An approach to validity indices for clustering techniques in big data," *Progress in Artificial Intelligence*, vol. 7, no. 2, pp. 81–94, Jun 2018.

[5] H. T. Haider, O. H. See, and W. Elmenreich, "A review of residential demand response of smart grid," *Renewable and Sustainable Energy Reviews*, vol. 59, pp. 166 – 178, 2016.

[6] I. Hussain, S. Mohsin, A. Basit, Z. A. Khan, U. Qasim, and N. Javaid, "A review on demand response: Pricing, optimization, and appliance scheduling," *Procedia Computer Science*, vol. 52, pp. 843 – 850, 2015.

[7] H. T. Haider, O. H. See, and W. Elmenreich, "Residential demand response scheme based on adaptive consumption level pricing," *Energy*, vol. 113, pp. 301 – 308, 2016.

[8] F. Martínez-Álvarez, A. Troncoso, G. Asencio-Cortés, and J. C. Riquelme, "A survey on data mining techniques applied to electricity-related time series forecasting," *Energies*, vol. 8, no. 11, pp. 13 162–13 193, 2015.

[9] T. chung Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164 – 181, 2011.

[10] S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah, "Time-series clustering – a decade review," *Information Systems*, vol. 53, pp. 16 – 38, 2015.

[11] H. Wang, W. Wang, J. Yang, and P. Yu, "Clustering by pattern similarity in large data sets," vol. 3, 10 2002.

[12] F. Iglesias and W. Kastner, "Analysis of similarity measures in times series clustering for the discovery of building energy patterns," *Energies*, vol. 6, no. 2, pp. 579–597, 2013.

[13] T. Zhang, G. Zhang, J. Lu, X. Feng, and W. Yang, "A new index and classification approach for load pattern analysis of large electricity customers," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 153–160, Feb 2012.

[14] F. Martínez-Álvarez, A. Troncoso, J. C. Riquelme, and J. M. Riquelme, "Partitioning-clustering techniques applied to the electricity price time series," in *Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning*, ser. IDEAL'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 990–999.

[15] F. M. Alvarez, A. Troncoso, J. C. Riquelme, and J. S. A. Ruiz, "Energy time series forecasting based on pattern sequence similarity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 8, pp. 1230–1243, Aug 2011.

[16] P. Faria, Z. Vale, and J. Baptista, "Demand response programs design and use considering intensive penetration of distributed generation," *Energies*, vol. 8, no. 6, pp. 6230–6246, 2015.

[17] S. Haben, C. Singleton, and P. Grindrod, "Analysis and clustering of residential customers energy behavioral demand using smart meter data," *IEEE Transactions on Smart Grid*, vol. 7, no. 1, pp. 136–144, Jan 2016.

[18] J.-S. Chou and N.-T. Ngo, "Smart grid data analytics framework for increasing energy savings in residential buildings," *Automation in Construction*, vol. 72, pp. 247 – 257, 2016.

[19] R. Perez-Chacon, R. L. Talavera-Llames, F. Martinez-Alvarez, and A. Troncoso, "Finding electric energy consumption patterns in big time series data," in *Distributed Computing and Artificial Intelligence, 13th International Conference*, S. Omatu, A. Semalat, G. Bocewicz, P. Sitek, I. E. Nielsen, J. A. García García, and J. Bajo, Eds. Cham: Springer International Publishing, 2016, pp. 231–238.

[20] D. L. Davies and D. Bouldin, "A cluster separation measure," vol. PAMI-1, pp. 224 – 227, 05 1979.

[21] "Spark clustering rdd based api documentation for spark 2.3.0. 2017." https://spark.apache.org/docs/2.3.0/mllib-clustering.html, accessed: 2018-06-11.

Un procedimiento efectivo para descomponer y modelar series temporales en agricultura

Francisco Aragón, Francisco Javier Baldán, Manuel Parra, José M. Benítez
Depto. Ciencias de la Computación e Inteligencia Artificial, DICITS, DaSCI
Universidad de Granada
Granada, España
{far, fjbaldan, manuelparra, J.M.Benitez}@decsai.ugr.es

***Resumen*—En este trabajo proponemos una forma innovadora de abordar casos reales de predicción de la producción de cultivos en una cooperativa. Nuestro enfoque consiste en la descomposición de la serie temporal original de los cultivos en sub-series temporales según una serie de factores, con el objetivo de generar un modelo predictivo del cultivo a partir de los modelos predictivos parciales de las sub-series. El ajuste de los modelos se realiza mediante un conjunto de técnicas estadísticas y de Aprendizaje Automático. Esta metodología se ha comparado con una metodología intuitiva que consiste en una predicción directa de las series temporales. Los resultados muestran que nuestro enfoque logra un mejor rendimiento de predicción que la manera directa, por lo que aplicar una metodología de descomposición es más adecuada para este problema que la no descomposición.**

***Palabras Clave*—agricultura, predicción, modelos predictivos, series temporales, descomposición**

I. Introducción

La agricultura es una actividad económica que resulta de vital importancia en prácticamente todos los países del mundo. En los últimos años la mejora del rendimiento agrícola (cantidad de producción obtenida por área cultivada) había sido propiciada por avances en la maquinaria empleada, nuevas técnicas de siembra, así como mediante la mejora de semillas y agroquímicos o un mejor control de plagas y enfermedades. Pero ahora, la agricultura comercial se ha vuelto una actividad de alta tecnología, en la que los avances tecnológicos informáticos también tienen su aplicación ya que permiten generar información de alta calidad sobre los procesos productivos.

Por lo tanto, la agricultura puede beneficiarse del auge de las técnicas englobadas dentro del ámbito de la Inteligencia Artificial y del Análisis de Datos, para crear modelos predictivos que predigan situaciones futuras que sean de ayuda para mejorar tanto la productividad de las cosechas como la toma de decisiones en todo lo relativo a ellas. Dentro del ámbito de la agricultura una gran cantidad de procesos tales como la producción o la aparición de plagas entre otras, pueden modelarse como una colección de observaciones habitualmente ordenadas a lo largo de un periodo de tiempo, es decir, mediante una serie temporal. Por medio de la aplicación sobre las series temporales de un conjunto de técnicas estadísticas y de Aprendizaje Automático se pueden generar modelos predictivos que sean capaces de extraer sus regularidades y de realizar predicciones.

Estos modelos predictivos resultantes de los análisis de los datos, pueden ser de gran importancia en las cooperativas y asociaciones de las que dependen los pequeños agricultores a la hora de la toma de decisiones en una gran cantidad de situaciones cotidianas relacionadas por ejemplo con la gestión de plagas, de los cultivos en el campo, o la gestión de la producción esperada ya sea para ofertarla en el mercado, para contratar el personal necesario para tratarla o para obtener el material necesario para su preparación y embalaje. La eficiencia en todas estas situaciones va a depender de la capacidad de transformar los datos brutos en información precisa que permita tomar la decisión acertada en cada una de ellas, siendo la gestión de la producción una de las actividades de mayor importancia. Por ello, disponer de información fiable sobre las expectativas de producción es crítico en el sector agrícola.

En este artículo se propone descomponer las series temporales del rendimiento de un cultivo en un conjunto de sub-series temporales en función de una serie de factores, siendo clave la semana de plantación, con la intención de que con la descomposición se obtengan sub-series con patrones más establecidos que la serie original general. El análisis y ajuste de los modelos predictivos de las sub-series temporales se realiza mediante una batería de métodos estadísticos como son los métodos ARIMA [3] y de Aprendizaje Automático como son las redes neuronales entre otras, para a partir de ellos, realizar una predicción general de los kilogramos del cultivo. La metodología propuesta se ha evaluado en varios casos de estudio reales, y se ha comparado con una metodología directa que consiste en una predicción de las series temporales sin descomponer. Los resultados muestran como se obtiene una mejora de la predicción al aplicar la metodología planteada con respecto a la serie sin descomponer.

El resto del artículo está organizado de la siguiente manera: la sección 2 describe el estado actual en que se encuentra la aplicación de modelos predictivos sobre problemas de predicción de cosechas modelados como series temporales. La sección 3 describe el problema, la forma en que ha llevado a cabo la descomposición, y la metodología empleada para resolverlo. En la sección 4 se detalla la experimentación realizada, se muestra el conjunto de datos y los resultados

obtenidos. En la sección 5 se discuten los resultados obtenidos. Por último, la sección 6 presenta las conclusiones obtenidas tras la realización del trabajo.

II. Estado del arte

En el ámbito de la agricultura, los investigadores han propuesto numerosos modelos y procedimientos para mejorar la predicción de cosechas. La mayoría de ellos han tenido un enfoque multivariante, en los que se trata de incorporar a la predicción el impacto que numerosas variables relativas a cultivos tales como la lluvia, la temperatura, los fertilizantes o la calidad del suelo tienen en la cosecha. Los métodos más ampliamente empleados para realizar este tipo de modelos multivariantes son las redes neuronales, más en concreto en su variante *feed-fordward*. En [10] y [12] se desarrollan estudios sobre si redes neuronales con variables relacionadas al cultivo son adecuadas para realizar la predicciones futuras, realizan una evaluación de la capacidad predictiva de diferentes parametrizaciones de la red, y comparan los resultados con otros modelos de regresión. En otros trabajos como en [14] se ha realizado un estudio concreto de las variables óptimas para la predicción de cosecha de maíz. Existe un déficit de trabajos en cuanto a la predicción de cosechas que tengan un enfoque multivariante y que además empleen métodos distintos a redes neuronales. Algunos de esos trabajos son [6] y [9], que demuestran la capacidad de métodos de *Random Forests (RF)* para la estimación de cultivos, y en [4] para máquinas de soporte vectorial. En [16] se realiza una comparación entre diferentes metodos regresivos como redes neuronales, funciones de base radial o árboles de regresión.

En cuanto a un enfoque univariante, los métodos ARIMA son los más ampliamente usados en la predicción de cosechas. En [15] se emplean los métodos ARIMA para predecir el área y la producción de trigo en los próximos años. También los alisados exponenciales los acompañan en algunos trabajos como en [1]. En [5] se realiza una comparación sobre la capacidad predictiva entre los métodos ARIMA y los alisados exponenciales.

La mayoría de los trabajos que se han realizado en este ámbito se han realizado desde un punto de vista multivariante, centrando el estudio en la importancia del impacto de los factores endógenos sobre la variable a predecir. Las redes neuronales y los métodos ARIMA son los más empleados, siendo también aplicados aunque en menor medida las máquinas de soporte vectorial o los métodos *Random Forest*. Son menos los trabajos en los que se ha empleado un enfoque univariante que trabaje directamente sobre la serie temporal. Aún así, los trabajos tratan de una manera muy superficial a la serie temporal ya que no tratan de adaptarla o modificarla para obtener una serie más fácil de predecir, sino que se centran directamente en aplicar un método concreto, con diferentes entradas o diferentes parametrizaciones para obtener una mejor predicción. Además, los trabajos solo emplean un método para el análisis de la serie, obviando la posibilidad de combinar varios métodos sobre la misma serie mediante una subdivisión de la misma.

III. Propuesta

Para hacer una declaración más precisa se formaliza el problema de predicción abordado, por lo que a continuación se detalla nuestra propuesta.

III-A. Descripción del problema y formulación

El problema se ubica en el ámbito de una cooperativa de frutas y hortalizas. La cooperativa está compuesta de un conjunto de parcelas, pertenecientes a los agricultores asociados, y la producción se divide en campañas, que representan los años agrícolas, de septiembre a julio. Cada parcela planta un producto específico durante una campaña. La superficie de cultivo de las parcelas se mide en metros cuadrados y cada parcela tiene su propia superficie. Los productos que se plantan en la cooperativa son diferentes variedades de frutas y verduras y su producción se mide en kilogramos. La producción total de un producto está formada por la suma de la producción de cada parcela que planta el producto. Algunos de los productos se plantan durante un cierto periodo de tiempo, siguiendo un patrón de siembra similar en cada campaña, mientras que otros se plantan de forma irregular a lo largo de la campaña. La producción durante el ciclo de vida de un cultivo suele ser similar para el mismo producto, por lo que para los productos plantados en periodos cercanos la producción sigue un patrón similar.

El principal interés de la cooperativa reside en conocer la producción en unidades de peso (p.e. kilogramos) que producirá cada producto para la próxima semana, de manera que la cooperativa pueda manejar adecuadamente el volumen de producción que se tendrá. Por lo tanto, la mejor manera de medir la producción es de forma semanal, y el problema a tratar consiste en la predicción de la producción en kilogramos de producto para la próxima semana. Este problema se abordará descomponiendo las series temporales de producción en sub-series temporales de semanas de siembra significativas, y empleando sobre ellas técnicas de Inteligencia Artificial y métodos estadísticos para generar modelos predictivos cuyas predicciones se irán agregando para obtener finalmente la producción global.

III-B. Descomposición de las series temporales

La idea principal del enfoque de descomposición en sub-series temporales es capturar el comportamiento similar que debe tener la producción de un producto que ha sido plantado en el mismo corto periodo de tiempo, ya que su ciclo de vida será muy parecido. Por lo tanto, el principal criterio de descomposición está tomado de uno de los factores más influyentes en la producción: la fecha de siembra. Las fechas de plantación se agrupan por semanas. La descomposición se realiza en semanas de siembra significativas y habrá tantas sub-series temporales como semanas de plantación significativas se establezcan. Las semanas significativas son aquellas que presentan un gran número de parcelas en plantación y por tanto la mayor parte de la producción del producto. Para determinar el conjunto de semanas significativas, se computan

las frecuencias de las semanas de siembra durante las campañas anteriores para analizar el número de parcelas que han plantado el producto en cada semana, y por lo tanto elegir las semanas con una mayor frecuencia de parcelas en plantación. Las semanas cuyas frecuencias de siembra son pequeñas se agrupan en una o dos series temporales adicionales.

Para los casos particulares bajo estudio, la descomposición se hizo como se expresa en la Tabla I. La Tabla I revela la elección de semanas significativas para los productos, y por lo tanto el número de sub-series temporales. El producto 1 tiene tres importantes semanas consecutivas de plantación; 37, 38 y 39 que cubren el 70 % de las parcelas. Para el producto 2 las semanas elegidas presentan cierta dispersión ya que tiene sus semanas significativas entre la 8 y la 11 y entre la 31 y la 38, representando el 67 % del total de parcelas. En general, para ambos productos predomina la plantación en las semanas correspondientes a los meses de agosto y septiembre.

Tabla I
Semanas de plantación seleccionadas para cada sub-serie

Nº de la subserie	**Semanas de plantación de la subserie**	
	Producto 1	**Producto 2**
Sub1	37	8-9
Sub2	38	10-11
Sub3	39	31-32
Sub4	-	33-34
Sub5	-	35-36
Sub6	-	37-38
Sub7	-	Entre la 12 y la 30
Sub8	-	No entre la 12 y la 30

III-C. Metodología

Para resolver el problema, proponemos desarrollar modelos que predigan la producción de un producto para la semana siguiente. Esta producción se modela como una serie temporal del rendimiento histórico semanal del producto. El rendimiento del cultivo se utiliza como valor de la serie temporal en lugar de los kilogramos, ya que elimina el efecto de la influencia que tiene la superficie sobre el número total de kilogramos producidos por una parcela en un periodo. El rendimiento global de cada producto se obtiene dividiendo la suma total de los kilogramos producidos por las parcelas en producción en cada periodo de tiempo por la suma de las superficies de las parcelas que contribuyen en ese periodo de tiempo.

$$\text{Rendimiento de cultivo} = \frac{\sum kg}{\sum \text{superficie}} \tag{1}$$

La metodología propuesta para desarrollar los modelos predictivos se basa en la descomposición de las series temporales del rendimiento de los cultivos de un producto en sub-series temporales. En cada sub-serie temporal solo se tratan los cultivos que han sido plantados en un periodo de tiempo muy corto, por lo que como paso esencial para nuestra metodología se deben analizar y agrupar las semanas de plantación para aprovechar la homogeneidad en la duración y producción del cultivo y además cubrir el mayor número de parcelas posible. Como resultado de esto, las sub-series temporales que se obtengan presentarán comportamientos más predecibles.

Una vez que se tienen las sub-series parciales de un producto, se ajusta un modelo predictivo para cada una de ellas. El proceso de construcción del modelo predictivo sigue los siguientes pasos:

1. Preprocesamiento: Se realiza una imputación de valores faltantes en las sub-series temporales en caso de que existan, para que todas las sub-series temporales tengan la misma longitud. Además, se realiza una normalización min-max entre el rango [0,1]:

$$\text{normalización } TS = \frac{TS - min_{TS}}{max_{TS} - min_{TS}} \tag{2}$$

   donde TS denota la sub-serie temporal original, y min_{TS} y max_{TS} los valores mínimo y máximo de la sub-serie temporal. Por último, las series parciales se dividen en datos de entrenamiento y datos de prueba.
2. Modelado: Se ajustan modelos predictivos de las sub-series temporales de entrenamiento utilizando un conjunto de técnicas estadísticas y de *machine learning* (ARIMA, alisados exponenciales, máquinas de soporte vectorial, *random forest*, redes neuronales, redes neuronales parcialmente recurrentes y modelos aditivos). Para cada una de las técnicas anteriores se generan un conjunto de modelos en función de las diferentes parametrizaciones empleadas.
3. Validación: El paso de validación determina en primer lugar cuál es el modelo más adecuado para cada técnica y, a partir de él, cuál es el mejor modelo global. El mejor modelo será aquel que obtenga una menor medida de error y por tanto una mayor capacidad predictiva. Para evaluar el error y la capacidad predictiva de cada uno de los modelos generados se aplica una validación cruzada *leave-one-out* [7] sobre las sub-series temporales de prueba. Para esta variante de la validación cruzada, existe un único valor de prueba para cada iteración, con la particularidad de que el conjunto de entrenamiento está formado por los valores que se producen temporalmente antes del valor de prueba. Así pues, no se usan valores futuros al valor de prueba en el entrenamiento del modelo. En cada iteración, el valor de prueba de la iteración anterior se incorpora al conjunto de entrenamiento, y el siguiente valor temporal que antes no ha sido usado se incorpora como valor de prueba.
4. Predicción: Después de completar los pasos anteriores, se tiene el modelo predictivo más adecuado para cada sub-serie temporal. A partir de estos modelos se genera una predicción para la semana siguiente del rendimiento de cultivo en cada sub-serie temporal. La predicción en kilogramos se obtiene multiplicando el rendimiento previsto obtenido de la cosecha y la superficie estimada de cultivo de las parcelas:

$$kg = \text{rendimiento*superficie} \tag{3}$$

El área de cultivo se estima a partir de las parcelas que han plantado en la presente campaña. Para ello se añade la superficie de todas las parcelas plantadas, menos la superficie de aquellas parcelas que han plantado y que no han podido producir al llevar plantadas menos días del tiempo que el producto necesita para empezar a producir, y aquellas parcelas que ya han terminado de producir.

5. Agregación: Para obtener la predicción global de kilogramos, se suman los valores pronosticados de kilogramos de cada sub-serie temporal.

$$KG = \sum_{i=1}^{n} kg_i \tag{4}$$

Dónde $i = \{1, 2, ...n\}$ es el número de sub-series temporales.

IV. Estudio empírico

En esta sección, se describe el estudio empírico que hemos diseñado para evaluar la validez y el desempeño de nuestra propuesta de predicción para el problema abordado. La metodología se ha aplicado a casos reales. Se detallan los conjuntos de datos, las medidas de rendimiento y los enfoques competitivos. Después, se presentan los resultados empíricos obtenidos tras la realización de los experimentos.

IV-A. Procedimiento

Para comprobar la validez, el enfoque propuesto en este trabajo se ha comparado con un enfoque de predicción directo del problema. Esta estrategia directa consiste en aplicar algunos métodos de predicción a la serie temporal completa sin ningún tipo de descomposición. Las diferentes técnicas de predicción empleadas son ampliamente utilizadas en el estado del arte actual. En particular, los modelos considerados son: ARIMA, redes neuronales, máquinas de soporte vectorial, alisados exponenciales, redes neuronales parcialmente recurrentes, *Random Forest* y el modelo aditivo *Facebook Prophet* [17]. Argumentamos que este enfoque directo obtiene una menor capacidad predictiva, y por lo tanto, un enfoque más efectivo es considerar la descomposición basada en la semana de siembra.

IV-B. Configuración

Las series temporales de los productos y las sub-series temporales son univariantes. El periodo de las series temporales y las sub-series temporales se ha establecido en 52. Cada observación de la serie temporal y las sub-series temporales representa un valor semanal. Los valores de las series temporales se dividen en conjunto de entrenamiento y conjunto de pruebas. El porcentaje de valores en entrenamiento para todas las series temporales es el primer 65 % de los valores totales, dejando el 35 % restante como conjunto de pruebas. La longitud de la serie temporal del producto 1 es de 248, 161 valores para entrenamiento y 87 para la prueba. Para el segundo producto la longitud es de 222, 144 valores para el entrenamiento y 78 para la prueba. Se ha utilizado el error cuadrático medio (RMSE) y el error absoluto medio (MAE) como medida del rendimiento:

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(o_t - \bar{pe}_t)^2} \tag{5}$$

$$MAE = \frac{1}{T}\sum_{t=1}^{T}|o_t - \bar{pe}_t| \tag{6}$$

donde o_t denota el valor de rendimiento original de la sub-serie temporal, pe_t el valor de rendimiento predicho y T el número de valores de prueba. Por otro lado, todo el procesamiento se ha implementado en el lenguaje de programación R, utilizando gran cantidad de los paquetes disponibles en el repositorio CRAN [2], [8], [11], [13], [17].

IV-C. Conjunto de datos

Los datos de producción se han recogido de una cooperativa situada en el sur de España. Para este trabajo se han seleccionado dos cultivos:

- Producto 1. Vegetal de temporada. La producción de rendimiento para este producto está disponible desde septiembre de 2013 hasta abril de 2018, cubriendo las campañas 2013, 2014, 2015, 2016 y 2017. La plantación principal tiene lugar en el mes de septiembre, pero también se realizan pequeñas plantaciones en los meses de agosto y febrero.
- Producto 2. Fruta de temporada. La producción de rendimiento para este producto está disponible desde enero de 2014 hasta abril de 2018, cubriendo la mitad de la campaña de 2013 y las campañas completas de 2014, 2015 y 2016 y gran parte de la de 2017. La plantación principal ocurre en los meses de agosto-septiembre y en los meses de febrero-marzo.

IV-D. Resultados

La Tabla II muestra el RMSE en prueba obtenido con nuestra metodología propuesta y con el enfoque directo. La Tabla III muestra el MAE. En las filas de la primera columna de ambas tablas se muestra la metodología empleada, ya sea la metodología propuesta en este trabajo o la metodología directa. La segunda columna especifica el método de modelado empleado por la metodología, en el caso de la metodología propuesta se han empleado una combinación de todos los métodos para ajustar la serie, pero en el caso de la metodología directa se ha empleado solo un método en cada caso. La columna 3 se subdivide en dos columnas siendo cada una referente al error cometido en cada producto. Todos los valores RMSE y MAE están normalizados en el rango [0-1].

V. Análisis y discusión de los resultados

Los resultados de la Tabla II y Tabla III muestran como nuestro enfoque logra una mejor capacidad predictiva que el enfoque directo para cada uno de los casos reales al obtener un error menor tanto en RMSE como en MAE.

Tabla II
RMSE normalizado obtenido por nuestra propuesta y de manera directa sobre el subconjunto de prueba

Metodología	Modelo usado	Error cometido (RMSE)	
		Producto 1	Producto 2
Propuesta	Todos	**0.0371**	**0.0368**
Directa	ARIMA	0.0536	0.0458
Directa	Red Neuronal	0.0502	0.0446
Directa	SVM	0.0643	0.0685
Directa	Alisado Exponencial	0.0565	0.0475
Directa	Red Neuronal Parcialmente Recurrente	0.0457	0.0445
Directa	Random Forest	0.0658	0.0487
Directa	Modelo Aditivo	0.0600	0.0579

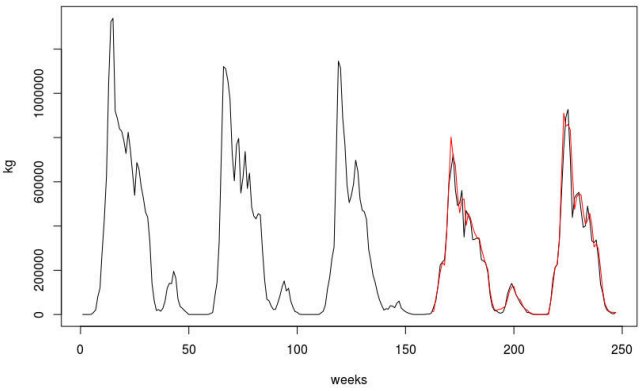
Tabla III
MAE normalizado obtenido por nuestra propuesta y de manera directa sobre el subconjunto de prueba

Metodología	Modelo usado	Error cometido (MAE)	
		Producto 1	Producto 2
Propuesta	Todos	**0.0214**	**0.0247**
Directa	ARIMA	0.0346	0.0400
Directa	Red Neuronal	0.0331	0.0292
Directa	SVM	0.0485	0.0403
Directa	Alisado Exponencial	0.0384	0.0350
Directa	Red Neuronal Parcialmente Recurrente	0.0346	0.0302
Directa	Random Forest	0.0447	0.0343
Directa	Modelo Aditivo	0.0488	0.0488

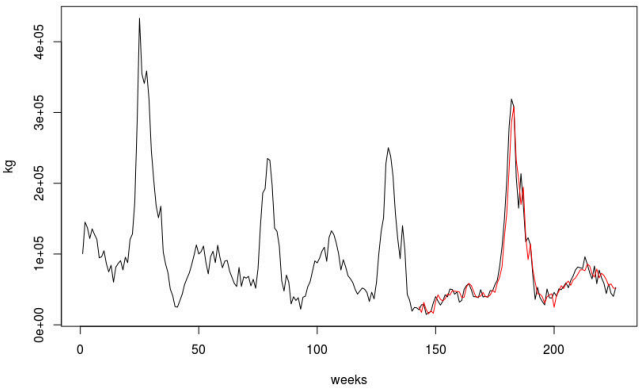
La mejora al emplear el enfoque con descomposición es considerable para el producto 1. El RMSE obtenido con nuestra metodología es de 0.0371, mientras que el menor RMSE obtenido con la metodología directa en cualquiera de sus variantes es de 0.0457 conseguido aplicando una red neuronal parcialmente recurrente. Para los ajustes con el resto de métodos en la metodología directa el RMSE es superior a 0.05 e incluso a 0.06. Por otro lado nuestra metodología obtiene un MAE más bajo de 0.0214, mientras que el menor MAE obtenido con la metodología directa es de 0.0331.

Mientras que para el producto 2 nuestra metodología ofrece también una mejor capacidad predictiva que la metodología directa. El RMSE más bajo obtenido con nuestro enfoque es del 0.0368 mientras que el menor de los RMSE obtenidos por el enfoque directo directo es de 0.0445, conseguido también con una red neuronal parcialmente recurrente. El resto de ajustes con el enfoque directo ofrecen un RMSE similar al 0.0445 conseguido por la red neuronal parcialmente recurrente, salvo varias excepciones. Con respeto al MAE el menor conseguido por nuestra metodología es de 0.0247, y el menor del enfoque directo es de 0.0292. El resto de variantes del enfoque directo obtienen un MAE superior al 0.03.

Así pues estos resultados muestran que tanto para el caso del producto 1 como el producto 2 los errores cometidos por nuestro enfoque son siempre menores que los errores obtenidos por el enfoque directo, siendo la mejora obtenida considerable.



(a) Producto 1



(b) Producto 2

Figura 1. Serie temporal de los kilogramos de producción real y predicción realizada por nuestra metodología (línea roja) en los casos de prueba para los dos productos analizados

VI. Conclusiones

En este trabajo se ha presentado una forma innovadora de realizar el análisis de una serie temporal del rendimiento de la producción de productos reales de una cooperativa hortofrutícula mediante la descomposición de la serie temporal general en sub-series temporales según la semana de plantación, siendo el modelo predictivo final el resultado de la mezcla entre los mejores modelos ajustados para cada sub-serie a partir de un conjunto de métodos estadísticos y de Aprendizaje Automático. Además, esta metodología ha sido comparada con otra implementación posible del mismo problema.

Los resultados de los experimentos han mostrado que la metodología propuesta mejora la capacidad predictiva de la metodología comparada, ya que obtiene un error menor para ambos productos analizados. Además, siempre mejora el error obtenido en cada una de las comparaciones. Por lo que se puede concluir que una descomposición de la serie temporal según las semanas significativas de plantación se da lugar a

que se obtenga una mejora significativa en la predicción con respecto a la serie sin descomponer.

Agradecimientos

Esta investigación ha sido parcialmente financiada por los Proyectos de Investigación Nacional TIN2013-47210-P, y TIN2016-81113-R y por el Proyecto de Investigación de Excelencia de la Junta de Andalucía P12-TIC-2958.

Referencias

[1] Akram, M., Bhatti, I., Ashfaq, M., Khan, A.A. *Hierarchical Forecasts of Agronomy-Based Data*, American Journal of Mathematical and Management Sciences, 36(1), 49-65, 2017

[2] Bergmeir C. and Benítez J.M. *Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS*. Journal of Statistical Software, 46(7), 1-26, 2012

[3] Box G.E.P. and Jenkins G.M. *Time Series Analysis, Forecasting and Control*, 1970

[4] Brdar S., Culibrk D., Marinkovic B., Crnobarac J., Crnojevic V. *Support Vector Machines with Features Contribution Analysis for Agricultural Yield Prediction*, Second International Workshop on Sensing Technologies in Agriculture, Forestry and Environment, 43-47, 2011

[5] Choudhury, A. and Jones, J. *Crop yield prediction using time series models*, Journal of Economics and Economic Education Research., 15, 53-68, 2014.

[6] Fukuda S., Spreer W., Yasunaga E., Yuge K., Sardsud V. and Müller J. *Random Forests modelling for the estimation of mango (Mangifera indica L. cv. Chok Anan) fruit yields under different irrigation regimes*, Agricultural Water Management, 116(1), 142-150,2013

[7] Hastie T., Tibshirani R., Friedman, J. *The Elements of Statistical Learning*, Springer, New York, 2009.

[8] Hyndman R.J. and Khandakar Y. *Automatic time series forecasting: the forecast package for R*. Journal of Statistical Software, 26(3), 1-22, 2008

[9] Jeong J.H., Resop J., Mueller N., Fleisher D.H., Yun K.,Butler E., Timlin D., Shim K., Gerber J., Reddy V., and Kim S.H. *Random Forests for Global and Regional Crop Yield Predictions*, PLoS One. 2016; 11(6): e0156571.

[10] Ji, B., Sun Y., Yang S. and Wan J. *Artificial neural networks for rice yield prediction in mountainous regions*, Journal of Agricultural Science, 145: 249-26, 2007.

[11] Karatzoglou A., Smola A., Hornik K. and Zeileis A. *kernlab - An S4 Package for Kernel Methods in R*. Journal of Statistical Software, 11(9), 1-20, 2004

[12] Kaul M., Hill R.L., and Walthall C. *Artificial neural networks for corn and soybean yield prediction*, Agricultural Systems, 85(1): 1-18, 2005.

[13] Liaw A. and Wiener M. *Classification and Regression by randomForest*. R News, 2(3), 18-22, 2002

[14] Liu J., Goering C.E., and Tian L. *A neural network for setting target corn yields*, Transactions of the ASAE. Vol. 44(3): 705–713, 2007.

[15] Najeeb I., Khuda B., Asif M. *Use of the ARIMA Model for Forecasting Wheat Area and Production in Pakistan*, Journal of Agriculture ans Social Sciences, 1(2), 120-122, 2015

[16] Ruß G. *Data Mining of Agricultural Yield Data: A Comparison of Regression Models*, In: Perner P. (eds) Advances in Data Mining. Applications and Theoretical Aspects, ICDM 2009. Lecture Notes in Computer Science, vol 5633.

[17] Taylor S. and Letham B. *prophet: Automatic Forecasting Procedure. R package version 0.1*. 2017

[18] Wuo W., Xue H. *An incorporative statistic and neural approach for crop yield modelling and forecasting*, Neural Computing and Applications, 21(1): 109–117, 2012.

¿Requiere la clasificación de series temporales métodos específicos?

Amaia Abanda[1,2]
[1] *Basque Center for Applied Mathematics (BCAM)*
[2] *Intelligent Systems Group (ISG)*
Department of Computer Science and Artificial Intelligence,
University of the Basque Country UPV/EHU
Bilbao, Spain
aabanda@bcamath.org

Usue Mori[2,3]
[2] *Intelligent Systems Group (ISG)*
Department of Computer Science and Artificial Intelligence
[3] *Department of Applied Mathematics*
Statistics and Operational Research
University of the Basque Country UPV/EHU
Bilbao, Spain
usue.mori@ehu.es

Jose A. Lozano[1,2]
[1] *Basque Center for Applied Mathematics (BCAM)*
[2] *Intelligent Systems Group (ISG)*
Department of Computer Science and Artificial Intelligence,
University of the Basque Country UPV/EHU
Bilbao, Spain
ja.lozano@ehu.eus

***Resumen*—La clasificación de series temporales tiene la peculiaridad, respecto a otros problemas de clasificación supervisada, de que las observaciones de las series o variables predictoras tienen un orden específico. La mayoría de las soluciones propuestas en la literatura consideran que este orden es discriminatorio para la clasificación y, por tanto, emplean métodos específicos que tienen en cuenta el orden. El objetivo de esta investigación es explorar, de una manera preliminar, si realmente es siempre necesario el uso de métodos específicos para series temporales o si hay algunos casos en los que los métodos de clasificación no específicos, habitualmente utilizados para datos de tipo convencional, obtienen mejores resultados que los específicos. La experimentación llevada a cabo en 40 bases de datos del repositorio UCR muestra como en los casos en los que el orden de las observaciones no es relevante para la clasificación, los clasificadores no específicos consiguen mejorar la precisión, mientras que en los casos en los que el orden es un factor clave no lo consiguen.**

***Index Terms*—Clasificación, series temporales, orden, vector, información temporal, métodos específicos.**

I. Introduction

Una serie temporal es una secuencia de datos u observaciones que viene ordenada respecto al tiempo -la mayoría de las veces-, o respecto a otros aspectos como el espacio, por ejemplo. Es un tipo particular de datos precisamente porque tiene una naturaleza ordinal que la mayoría de datos no tienen. Las series temporales aparecen naturalmente en diferentes áreas como la bio-informática o economía [1], y están adquiriendo una gran relevancia en el ámbito del aprendizaje automático debido al gran reto que supone trabajar con este tipo especial de datos. De esta manera, continuamente están surgiendo nuevos métodos para representar, indexar, agrupar y clasificar series temporales [2]. El presente trabajo se sitúa dentro del área de la clasificación de series temporales -Time Series Classification (TSC)-, donde cada serie del conjunto de entrenamiento tiene asociada una clase, y el objetivo es encontrar una función tal que, dada una serie nueva, sea capaz de predecir cuál es su clase. La diferencia principal con el problema clásico de clasificación es que, mientras en éstos las instancias vienen descritas por atributos sin orden específico, en la clasificación de series temporales las instancias vienes definidas por las propias series temporales completas [3][4]. Además de que las variables predictoras están ordenadas en este caso, las series pueden tener un número muy alto de observaciones, las longitudes pueden ser variables dentro de una misma base de datos y, dependiendo del contexto, pueden ser observaciones con mucho ruido [5].

La peculiaridad de las series temporales, su posible interpretabilidad y su significado semántico han llevado a la comunidad científica a asumir que éstas tienen que ser tratadas con métodos específicamente diseñados para series [4][6][7][8]. Estos métodos tienen en cuenta la información temporal que intrínsecamente contienen las series para extraer atributos, construir modelos, medidas de similitud, etc. útiles para la clasificación. El orden de las observaciones ha sido hasta ahora una de las características más comúnmente asumidas como información temporal discriminatoria; en los métodos basados en distancias elásticas [3][9][10], en los métodos basados en modelos autoregresivos [11][12] en los modelos ocultos de Márkov [13], o en los métodos extraen atributos temporales de las series [5].

Esta asunción implica que los métodos no específicos, es decir, los métodos basados en clasificadores tradicionales

que no tienen en cuenta la información temporal, no deberían ser competitivos con los específicos en cuanto a precisión se refiere. En este trabajo se pretende explorar hasta qué punto esta afirmación es cierta y, sobre todo, en qué casos se cumple o no se cumple.

Este trabajo parte de la clasificación de series temporales basada en distancias [14], donde se utiliza la distancia/similitud entre series como criterio para la clasificación; dos series que sean similares pertenecerán a la misma clase. En particular, el método que hasta ahora más se ha utilizado para en la clasificación de series temporales basada en distancias es el 1-Nearest Neighbour (1NN), clasificador que aunque haya demostrado buenos resultados en el ámbito [2][9][15][16], no es muy robusto. En cuanto a las distancias utilizadas, en el ámbito de la clasificación de series temporales, podemos distinguir entre distancias que tienen en cuenta el orden de las observaciones y distancias que no lo tienen, pero muchas de las distancias situadas en el segundo grupo son muy costosas computacionalmente. En este contexto, sería interesante saber en qué casos no es necesario el uso de métodos específicos -casos en los que se podrían utilizar clasificadores estándares de una manera eficiente y con mejores resultados-.

El trabajo se organiza de la siguiente forma: en la Sección II se introducen algunos conceptos básicos para contextualizar la clasificación de series temporales basada en distancias, en la Sección III se presenta la propuesta y la experimentación realizada y, finalmente, en la Sección IV presentamos las principales conclusiones obtenidas.

II. Conceptos básicos y motivación

En esta sección se presenta un breve resumen de los conceptos básicos necesarios para abordar la clasificación de series temporales basada en distancias y se introducen algunas de las distancias más conocidas para series.

II-A. Clasificación de series temporales

Los métodos de clasificación de series temporales se pueden dividir entre tres principales categorías [15]: métodos basados en atributos, basados en modelos y basados en distancias. Dentro de la primera línea, los investigadores tratan de extraer características fundamentales de las series para obtener una nueva representación no-ordenada y de menor dimensión que, además, contenga la información más relevante de las series. Esta transformación evita tener que aprender directamente con las series originales y las traslada a un nuevo espacio donde las características discriminatorias pueden ser más detectables [17]. Algunos ejemplos de estas representaciones incluyen Discrete Fourier Transformation (DFT) [5], Discrete Wavelet Transformation (DWT) [18] o Piecewise Aggregate Approximation (PAA) [19]. Por otro lado, en los métodos basados en modelos, se asume que todas las series de una clase han sido generadas por un modelo subyacente y a una serie nueva se le asigna la clase del modelo al que mejor se ajuste. Algunos de los modelos más utilizados son los modelos auto-regresivos [11] [12] o los modelos ocultos de Markov [13]. Por último, en los métodos basados en distancias los investigadores tratan de definir una similitud, o disimilitud, entre series que tenga en cuenta diferentes características semánticas o temporales. La definición de una distancia adecuada es una cuestión crucial, ya que cada distancia refleja diferentes características de las series. Una vez elegida la distancia adecuada, la clasificación se lleva a cabo empleando métodos basados en distancias, en la mayoría de los casos clasificadores basados en el vecino más cercano (1NN). Entre las distancias para series temporales más conocidas están la distancia Euclídea, Dynamic Time Warping (DTW) [20] o Edit Distance with RealPenalty (ERP) [21].

II-B. Distancias de series temporales

Las distancias entre series temporales suelen categorizarse habitualmente en dos grupos principales mostradas en la Figura 1 [22]; las *medidas rígidas*, que se refieren a aquellas medidas que comparan el punto i-ésimo de una serie con el punto i-ésimo de la otra, y las *medidas elásticas*, que tratan de crear un mapeo no lineal entre las series para alinearlas, permitiendo una comparación de uno-a-varios puntos. La mayor diferencia entre estos dos tipos de medidas es que las medidas rígidas, al no considerar los puntos de alrededor, tratan la serie como si fuera un vector y por tanto, no consideran la ordenación de las mediciones. De esta forma, aún desordenando la serie, obtendríamos los mismos resultados de clasificación. Las medidas elásticas, por el contrario, consideran que el orden de las observaciones es determinante y hacen uso de la misma en su cálculo. Por tanto, otra ordenación de las observaciones podría cambiar por completo los resultados de la clasificación.

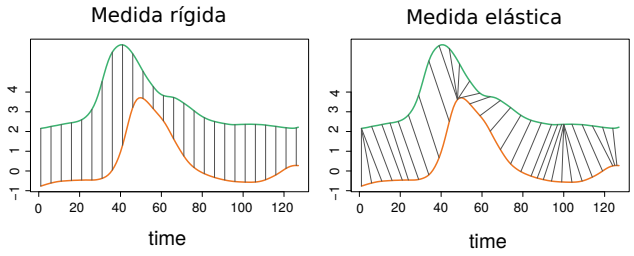


Figura 1: Alineamientos de medidas rígidas y elásticas

La distancia rígida más habitual es la denominada distancia Euclídea. Se define de la siguiente forma: dadas dos series temporales T=(t_1,...,t_n) y S=(s_1, ..., s_n),

$$ED(T,S) = \sqrt{\sum_{i=1}^{n}(t_i - s_i)^2}$$

Esta distancia es ampliamente conocida debido que cumple buenas propiedades, como ser métrica, así como su bajo coste computacional y simplicidad, razones por las

que es habitualmente utilizada en numerosos problemas de aprendizaje automático. Sin embargo, tiene también algunos inconvenientes como que sólo acepta dos series que tengan la misma longitud, o como se ha comentado anteriormente, que no considera el orden de las observaciones.

En vista de las carencias de la distancia Euclídea, en los últimos años se han presentado numerosas distancias elásticas específicamente diseñadas para medir la disimilitud entre series temporales. La más habitual entre ellas es la Dynamic Time Warping (DTW). Esta distancia alinea las dos series con tal de minimizar la distancia entre ellas y, por lo tanto, es robusta frente a desfases y distorsiones en el tiempo. Además, a diferencia de la distancia Euclídea, permite calcular la distancia entre series de diferentes longitudes. Dadas dos series temporales T y S, el objetivo de la DTW es buscar la alineación óptima que minimice la distancia entre ellas. Para ello, el primer paso es construir una matriz de distancias de n×m donde cada posición (i, j) contiene la distancia $(t_i - s_j)^2$, que representa el coste de alinear la observación i-ésima de T con la observación j-ésima de S. Así, un alineamiento entre dos series se define como un camino π en la matriz de distancias, que tiene que cumplir ciertas restricciones (continua, monótona creciente y puntos de inicio y final fijos) (ver Figura 2). El alineamiento óptimo entre dos series es aquel que minimiza la distancia acumulativa. Finalmente, la distancia DTW(T,S) se define como la distancia acumulativa del camino óptimo entre T y S. En la práctica, este camino óptimo se encuentra mediante métodos de programación dinámica [23] y uno de las mayores desventajas de ésta distancia es que este proceso tiene un coste computacional de $\mathcal{O}(N^2)$, siendo N el número de series.

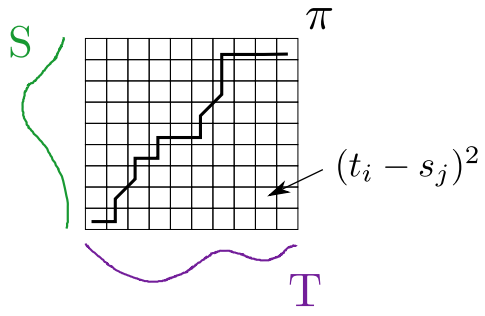


Figura 2: Matriz de distancias y camino óptimo de DTW

La mayoría de los métodos de clasificación de series temporales asumen que el orden de las observaciones es fundamental para discriminar entre clases y emplean medidas elásticas como la DTW [24][16][10]. El objetivo de este trabajo es realizar un estudio preliminar para explorar si realmente es, o en qué casos, la información temporal -en particular, el orden de las observaciones- fundamental para la clasificación. Esta cuestión proporcionaría información importante sobre la utilidad de clasificadores específicos para series temporales, ya que si el orden de las observaciones no es relevante, no habría necesidad de usar métodos específicos -clasificadores basados en la temporalidad-, sino que las series podrían entenderse como simples vectores y podrían utilizarse también otros clasificadores clásicos, obteniendo quizás resultados mejores.

III. Propuesta y experimentación

Como hemos expuesto anteriormente, el objetivo de este trabajo es explorar si realmente la información temporal y, en particular, el orden de las observaciones, juega siempre un papel fundamental en la clasificación de series temporales. Además, la hipótesis principal es que en aquellos casos donde el orden no es discriminatorio, los clasificadores no específicos podrían funcionar mejor, debido a que son más complejos y robustos que el 1-NN. Es decir, en los casos donde el orden no es relevante, las series pueden ser interpretadas como vectores (*feature vectors*) y, por tanto, las observaciones puede entenderse como dimensiones que no tienen un orden pre-definido. En los casos donde el orden es un aspecto discriminatorio para la clasificación, el hecho de que cada observación esté en una posición concreta es relevante y por tanto no se pueden interpretar las series como vectores. Los clasificadores no específicos, o tradicionales, esperan como entrada un vector sin orden, por lo que en el primer caso podrían funcionar bien mientras que en el segundo, al no ser capaces de tener en cuenta el orden, deberían obtener peores resultados.

El estudio parte de la clasificación basada en distancias y, más particularmente, tomamos el clasificador 1-NN como referencia. Como distancia que no considera el orden de las observaciones se utiliza la distancia Euclídea, mientras que como distancia que sí considera el orden consideraremos la DTW.

Asumiremos que en los casos en los que la distancia Euclídea obtiene mayor precisión que la DTW, el orden de las observaciones no es relevante para la discriminación, mientras que en las que la distancia DTW obtiene mayor precisión sí lo es. El objetivo es explorar si en los casos en los que 1-NN-EUC (método no específico) tiene mayor precisión que el 1-NN-DTW (método específico), los clasificadores estándares (métodos no específicos), como el Support Vector Machine o naive Bayes, mejoran más (veces) los resultados que en los casos en los que el 1-NN-DTW tiene mayor precisión que el 1-NN-EUC. En particular, se quiere comprobar si, en el caso de que el 1-NN-EUC obtiene mejores resultados que el 1-NN-DTW, los clasificadores estándares pueden llegar a mejorar los resultados obtenidos con el 1-NN.

Con el objetivo de poder contrastar las hipótesis planteadas, la clasificación se ha llevado a cabo utilizando, por un lado, 1-NN-Euclídea y 1-NN-DTW y, por otro, 3 clasificadores estándares (no específicos): Support Vector Machine (SVM), naive Bayes (NB) y Random Forest (RF). No se han ajustado los parámetros de ninguno para poder establecer una base de referencia. Los experimentos se han

realizado con 40 bases de datos del repositorio de series temporales UCR [25], un repositorio de bases de datos de series temporales que es habitualmente utilizada como referencia para evaluar nuevos métodos de clasificación de series temporales.

En el Cuadro I se muestran los resultados de la clasificación de las bases de datos donde el 1-NN-EUC obtiene mayor precisión que el 1-NN-DTW (15/40), mientras que en el Cuadro II la de aquellas en las que el 1-NN-DTW obtiene mayor precisión que 1-NN-EUC (25/40). Estas proporciones eran de esperar debido a que el 1-NN-DTW se considera uno de los métodos que mejores resultados obtienen en la clasificación de series temporales [2][9][16] [15]. Se han separado las bases de datos en estas dos tablas para poder visualizar mejor si hay una tendencia diferente en cada uno de los casos.

En el primer cuadro, donde el método no específico obtiene mejores resultados que el específico, se puede observar que los clasificadores estándares superan en algunos casos (12/15) al 1-NN. En el segundo cuadro, en cambio, donde el método específico funciona mejor que el no específico, los clasificadores estándares no superan al 1-NN en casi ningún caso (6/25).

BBDD	EUC	DTW	SVM	NB	RF
Adiac	0.61	0.52	0.40	0.57	**0.64**
Beef	0.53	0.50	0.50	0.47	**0.57**
Chlothe	0.65	0.62	0.58	0.35	**0.71**
CinC	0.90	0.69	0.73	0.88	0.76
ECG	0.88	0.80	0.87	0.77	0.83
ECGF5	0.80	0.75	**0.84**	0.78	**0.80**
GunPoint	0.91	0.89	0.81	0.78	**0.92**
Haptics	0.37	0.35	**0.45**	**0.44**	**0.46**
SonyAI	0.70	0.66	**0.78**	**0.94**	0.68
SonyAI II	0.86	0.82	0.82	0.79	0.80
Swedish	0.79	0.75	**0.88**	**0.86**	**0.88**
WaveX	0.74	0.73	**0.76**	0.66	**0.76**
WaveY	0.66	0.64	**0.67**	0.56	**0.69**
Wafer	0.99	0.98	**0.99**	0.70	**0.99**
FetalECG2	0.88	0.83	**0.90**	0.83	**0.91**

Cuadro I: Resultados de clasificación de las bases de datos donde 1-NN-Euc obtiene mayor precisión que 1-NN-DTW.

Con fin de entender mejor los resultados, el Cuadro III sintetiza y compara los dos cuadros anteriores. En él se refleja para cada una de las tablas (representadas por las filas EUC y DTW) el número de veces en el que al menos uno -y al menos dos- de los clasificadores no-específicos empleados supera la precisión del 1-NN. Se puede apreciar que en los casos del Cuadro I (fila EUC), los métodos no-específicos tienden a superarlo al menos una vez en el 80 % de los casos, mientras que los casos del Cuadro II (fila DTW), este porcentaje baja hasta el 24 %. Lo mismo sucede si contamos las veces en las que los métodos no-específicos superan al menos 2 veces al 1NN: el valor es 53 % para las bases de datos donde la Euclídea funciona mejor y 4 % para las del DTW.

Por lo tanto, se puede apreciar una tendencia a que los

BBDD	EUC	DTW	SVM	NB	RF
50Words	0.63	0.71	0.62	0.58	0.66
CBF	0.85	0.97	0.87	0.90	0.91
Coffee	0.75	0.79	0.71	0.68	0.68
Cricket_X	0.57	0.73	0.54	0.44	0.62
Cricket_Y	0.64	0.75	0.63	0.53	0.74
Cricket_Z	0.62	0.73	0.58	0.42	0.66
Diatom	0.93	0.96	0.90	0.87	0.91
Face_all	0.71	0.75	0.74	0.69	**0.81**
Face_four	0.78	0.84	0.64	**0.90**	0.77
FacesUCR	0.77	0.92	0.73	0.74	0.79
Fish	0.78	0.85	0.78	0.66	0.79
InlineSkate	0.34	0.38	0.25	0.22	0.35
Lightning2	0.75	0.80	0.70	0.67	0.75
Lightning7	0.58	0.77	0.63	0.71	0.75
MALLAT	0.91	0.92	0.77	0.87	**0.93**
MedicalImg	0.68	0.73	0.57	0.44	**0.73**
MoteStrain	0.88	0.90	0.85	0.84	0.89
OSU Leaf	0.52	0.63	0.52	0.38	0.51
Symbols	0.90	0.95	0.76	0.64	0.87
Synthetic	0.88	0.95	0.98	**0.96**	**0.96**
Trace	0.76	0.99	0.71	0.80	0.86
Two Lead	0.75	0.93	0.72	0.69	0.73
Two Patterns	0.91	0.99	0.89	0.46	0.85
WaveY	0.65	0.65	0.70	0.56	**0.71**
WordSyn	0.62	0.67	0.53	0.48	0.57

Cuadro II: Resultados de clasificación de las bases de datos donde 1-NN-Dtw obtiene mayor precisión que 1-NN-Euc.

métodos no-específicos funcionen mejor en los casos en los que la distancia Euclídea funciona mejor que la DTW con el 1-NN. Es decir, en las bases de datos en las que el orden de las observaciones no es relevante para clasificar y los métodos no-específicos -que no tienen en cuenta el orden- funcionan mejor que en las bases de datos que en las que el orden si es relevante.

	$\geq$ 1 vez	$\geq$ 2 veces
EUC	12/15 = 80 %	8/15 = 53 %
DTW	6/25 = 24 %	1/25 = 4 %

Cuadro III: Comparativa de las veces que los clasificadores estándares superan el 1-NN en el método no específico (EUC) y específico (DTW).

Para evaluar estadísticamente los Cuadros I y II, se han realizado mútliples test estadísticos emparejados de Wilcoxon utilizando el paquete 'scmamp' [26] para R. En el primer caso, se ha realizado 3 test estadísticos para comprobar si hay diferencias significativas entre los métodos del Cuadro I. En particular, s han comparado las siguientes columnas: 1-NN-EUC con SVM, 1-NN-EUC con NB y 1-NN-EUC con RF. Después de corregir los p-valores obtenidos de cada uno de ellos (0.34,0.09 y 0.28, correspondientemente), no podemos rechazar la hipótesis nula y, por tanto, no podemos afirmar que haya diferencias significativas entre los métodos. Es decir, no hay evidencias estadísticas de que los métodos no-específicos y el 1-NN tengan una comportamiento diferente. En el segundo caso, se han comparado los resultados de los métodos del Cuadro II; en particular, comparando el 1-NN-DTW con SVM,

1-NN-DTW con NB y 1-NN-DTW con RF. Después de corregir los p-valores obtenidos (0.00004, 0.00004 y 0.0008) podemos rechazar la hipótesis nula y afirmar que hay diferencias significativas entre los métodos. En particular, podemos afirmar que los métodos no-específicos tienen peores resultados de clasificación que el 1-NN-DTW. Resumiendo, en los casos donde la distancia Euclídea funciona mejor que la DTW (el orden no tiene importancia), los métodos no-específicos se comportan parecido al 1-NN, mientras que en los casos donde la DTW funciona mejor (el orden sí tiene importancia), funcionan peor que el 1-NN.

Para terminar, se ha realizado otros 2 test estadísticos para comparar si hay diferencias significativas entre los resultados del 1-NN y el mejor de los 3 métodos no-específicos en cada caso (Cuadro IV). En ambos casos, los test afirman que hay diferencias significativas con p-valores 0.01 y 0.002. En los casos donde el 1-NN-EUC obtiene mejores resultados que el 1-NN-DTW, el test indica que al menos uno de los 3 métodos no-específicos obtiene mejores resultados que el 1-NN. En los casos donde el 1-NN-DTW obtiene mejores resultados que el 1-NN-EUC, por el contrario, el test muestra que los 3 métodos específicos obtiene peores resultados que el 1-NN.

BBDD	EUC	Mejor
Adiac	0.61	**0.64**
Beef	0.53	**0.57**
Chlothe	0.65	**0.71**
CinC	**0.90**	0.88
ECG	**0.88**	0.87
ECGF5	0.80	**0.84**
GunPoint	0.91	**0.92**
Haptics	0.37	**0.46**
SonyAI	0.70	**0.94**
SonyAI II	**0.86**	0.82
Swedish	0.79	**0.88**
WaveX	0.74	**0.76**
WaveY	0.66	**0.69**
Wafer	**0.99**	**0.99**
FetalECG2	0.88	**0.91**

BBDD	DTW	Mejor
50Words	0.71	0.66
CBF	**0.97**	0.91
Coffee	**0.79**	0.71
Cricket_X	**0.73**	0.62
Cricket_Y	**0.75**	0.74
Cricket_Z	**0.73**	0.66
Diatom	**0.96**	0.91
Face_all	0.75	**0.81**
Face_four	0.84	**0.90**
FacesUCR	**0.92**	0.79
Fish	**0.85**	0.79
InlineSkate	**0.38**	0.35
Lightning2	**0.80**	0.75
Lightning7	**0.77**	0.75
MALLAT	**0.92**	0.93
MedicalImg	**0.73**	**0.73**
MoteStrain	**0.90**	0.89
OSU Leaf	**0.63**	0.52
Symbols	**0.95**	0.87
Synthetic	0.95	**0.98**
Trace	**0.99**	0.86
Two Lead	**0.93**	0.73
Two Patterns	**0.99**	0.89
WaveY	0.65	**0.71**
WordSyn	**0.67**	0.57

Cuadro IV: Resultados de clasificación del 1-NN y el mejor de los 3 clasificadores estándares en cada base de datos.

IV. Conclusiones

En este trabajo el objetivo era explorar de una manera preliminar si realmente es necesario siempre utilizar los métodos específicos que tienen en cuenta la información temporal, en particular, el orden de las observaciones, para la clasificación de series temporales. La mayoría de los métodos habitualmente utilizados en clasificación de series temporales, especialmente aquellos basados en distancias, asumen, por defecto, que el orden es un factor discriminatorio para la clasificación y que, por tanto, interpretar las series como vectores sin orden no aportaría beneficios. Está cuestión limita el uso de clasificadores estándares que no están basados en la temporalidad (por ejemplo SVM, naive Bayes o Random Forest), ya que si el orden de las series es crucial, utilizar estos clasificadores que no lo tienen en cuenta no tendría sentido. Sin embargo, la hipótesis de este estudio es que no hay que asumir siempre que el orden es discriminatorio y hay casos en los que los clasificadores estándares pueden funcionar igual o mejor que el tan valorado 1-NN.

Para evaluar está hipótesis se han tomado como métodos de referencia el 1-NN-EUC (que no tiene en cuenta el orden de las observaciones) y el 1-NN-DTW (que sí tiene en cuenta el orden de las observaciones). De esta manera, en la experimentación se observa que en los casos donde el 1-NN-EUC obtiene mejores resultado que el 1-NN-DTW, los clasificadores estándares pueden mejorar la precisión de clasificación (en el 80 % de los casos), mientras que en los otros casos los clasificadores estándares no consiguen mejorarlo (solo en el 24 % de los casos lo hacen).

Este resultado era esperado ya que si en una base de datos el 1-NN-EUC obtiene mayor precisión que el 1-NN-DTW se puede suponer que el orden de las observaciones no es un factor determinante para la clasificación. En ese caso, los clasificadores estándares, que son más complejos y robustos que el 1-NN, y no tienen en cuenta en orden, deberían funcionar bien. Por otro lado, si en una base de datos el 1-NN-DTW obtiene mayor precisión que el 1-NN-EUC se puede interpretar que el orden de las observaciones sí es un factor determinante para la clasificación, y tiene sentido que los clasificadores estándares, que no tienen en cuenta el orden de las observaciones, rindan peor. De hecho, los test estadísticos realizados a los resultados confirman las hipótesis. En otras palabras, cuando el orden de las observaciones no es un factor discriminatorio, los clasificadores estándares mejoran el resultado del 1-NN, mientras que cuando el orden es sí es relevante no lo consiguen.

En conclusión, no es que los clasificadores estándares, y métodos no específicos en general, sean inadecuados para la clasificación de series temporales, si no que dependiendo de las características discriminatorias de cada base de datos (en particular, la relevancia del orden de las observaciones para la clasificación), pueden llegar a ser beneficiosos. Por ello, sería interesante entender más en profundidad cuáles son las características temporales discriminatorias de las series, seguir estudiando la importancia del orden para la clasificación y buscar otras maneras de evaluar la influencia de este factor. Además, sería también enriquecedor seguir explorando el uso de clasificadores estándares, o métodos no específicos, para series temporales, ya que son mucho más robustos que el

1-NN y dependiendo de la distancia empleada en el 1-NN, mucho más rápidos.

Agradecimientos

Esta investigación esta subvencionada por el Gobierno Vasco a través del programa BERC 2018-2021, así como del Ministerio de Economía y Competitividad (MINECO) mediante la acreditación de excelencia BCAM Severo Ochoa SEV-2013-0323. También del proyecto TIN2017-82626-R, financiado por AEI/FEDER (UE) y acrónimo "GECECPAST". Además, gracias al Programa de Grupos de Investigación 2013-2018 (IT-609-13) (Gobierno Vasco) y TIN2016-78365-R (MINECO). A. Abanda está subvencionada con la beca BES-2016-076890.

Referencias

[1] E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks," *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, p. 102, 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=775047.775062

[2] P. Esling and C. Agon, "Time-series data mining," *ACM Computing Surveys*, vol. 45, no. 1, pp. 1–34, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2379776.2379788

[3] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, 2017.

[4] T. C. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2010.09.007

[5] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," *ACM SIGMOD International Conference on Management of Data*, pp. 419–429, 1994.

[6] M. Cuturi and J. Vert, "A kernel for time series based on global alignments," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 1, pp. 413–416, 2007. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4217433

[7] R. Povinelli, "Time series classification using Gaussian mixture models of reconstructed phase spaces," *Knowledge and Data ...*, vol. 16, no. 6, pp. 779–783, 2004. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1294898

[8] D. Eads, D. Hill, S. David, S. Perkings, J. Ma, R. Porter, and J. Theiler, "Genetic Algorithms and Support Vector Machines for Time Series Classi cation," *Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation*, vol. 4787, 2002.

[9] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Mining and Knowledge Discovery*, vol. 26, no. 2, pp. 275–309, 2013.

[10] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 565–592, 2015.

[11] A. Bagnall and Gareth Janacek, "A run length transformation for discriminating between auto regressive time series," *Journal of Classification*, vol. 31, no. October, pp. 274–295, 2014.

[12] M. Corduas and D. Piccolo, "Time series clustering and classification by the autoregressive metric," *Computational Statistics and Data Analysis*, vol. 52, no. 4, pp. 1860–1872, 2008.

[13] P. Smyth, "Clustering sequences with hidden Markov models," *Advances in Neural Information Processing Systems*, vol. 9, pp. 648–654, 1997. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.44.3648

[14] A. Abanda, U. Mori, and J. A. Lozano, "A review on distance based time series classification," *https://arxiv.org/abs/1806.04509*, pp. 1–28, 2018. [Online]. Available: http://arxiv.org/abs/1806.04509

[15] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 1, p. 40, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1882471.1882478

[16] Y. Chen, B. Hu, E. Keogh, and G. E. Batista, "DTW-D: Time Series Semi-Supervised Learning from a Single Example," *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, p. 383, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2487575.2487633

[17] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, "Time-series classification with COTE: The collective of transformation-based ensembles," *Proceedings of 32nd ICDE International Conference on Data Engineering*, vol. 27, no. 9, pp. 1548–1549, 2016.

[18] I. Popivanov and R. J. Miller, "Similarity Search Over Time-Series Data Using Wavelets," *Proceedings 18th International Conference on Data Engineering (ICDE)*, pp. 212–221, 2002.

[19] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," *Knowledge and Information Systems*, vol. 3, no. 3, pp. 263–286, 2001. [Online]. Available: http://link.springer.com/10.1007/PL00011669

[20] D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *Workshop on Knowledge Knowledge Discovery in Databases*, vol. 398, pp. 359–370, 1994. [Online]. Available: http://www.aaai.org/Papers/Workshops/1994/WS-94-03/WS94-03-031.pdf

[21] L. Chen and R. Ng, "On The Marriage of Lp-norms and Edit Distance," in *International conference on Very large data bases*, 2004, pp. 792–803.

[22] H. Kaya and . Gündüz-Ö üdücü, "A distance based time series classification framework," *Information Systems*, vol. 51, pp. 27–42, 2015.

[23] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[24] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction," *Proceedings of the 23rd ICML International Conference on Machine learning*, pp. 1033—-1040, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1143974

[25] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. E. Batista, "The UCR Time Series Classification Archive," 2015. [Online]. Available: www.timeseriesclassification.com

[26] B. Calvo and G. Santafe, "scmamp: Statistical comparison of multiple algorithms in multiple problems," *The R Journal*, vol. Accepted for publication, 2015.

Reglas de Asociación en Flujos de Datos para Monitorizar Actividad de Teléfonos Móviles

Elena Ruiz, Jorge Casillas
DaSCI (Centro de Investigación en Ciencia
de Datos e Inteligencia Computacional)
Universidad de Granada, Granada, España
Email: {eruiz, casillas}@decsai.ugr.es

***Abstract*—Los algoritmos de minería de flujo de datos trabajan sobre datos con altas tasas de llegada, que evolucionan a lo largo del tiempo y requieren respuesta en tiempo real. Este tipo de técnicas, que procesan los datos al vuelo, han captado la atención tanto del ámbito científico como del industrial. El aprendizaje descriptivo en flujos de datos nos permite tener un modelo que se adapta a la evolución de los datos para explicar qué está pasando en tiempo real. En este trabajo, mostramos el potencial de este campo usando información real registrada a través de teléfonos móviles durante meses (por el MIT Human Dynamics Lab). El objetivo es evolucionar de forma dinámica reglas de asociación que expliquen la actividad del usuario en cualquier momento de forma muy eficiente para que pueda incorporarse en un dispositivo móvil. Para conseguir este objetivo empleamos un algoritmo evolutivo que aprende y mantiene de forma incremental una población de reglas de asociación.**

***Keywords*—reglas de asociación; flujo de datos; algoritmo genético; aprendizaje automático; aprendizaje online**

I. Introducción

Vivimos en la era de los datos, donde todos nuestros movimientos y actividades son registrados (o podrían serlo) y, a veces, almacenados y procesados. Obviamente, una gran parte de la información generada carece de interés. Elegir qué información es relevante, sintetizarla y extraer conocimiento de ella es, cada vez, un aspecto más crítico en la sociedad actual. En ocasiones es posible utilizar esta información para obtener modelos (data mining) que simplifican la compleja realidad que contiene dicha información.

La necesidad de extraer información relevante de fuentes de datos ordenados cronológicamente en forma de un flujo continuo, veloz y que cambia con el tiempo, excediendo las capacidades habituales de almacenamiento y procesamiento son cada vez más comunes tanto en el entorno industrial como en el científico [1]. Para solucionar este tipo de problemas es posible gestionar flujos de datos, secuencias infinitas de registros estructurados que se reciben de forma continua [1]. La característica clave de estos sistemas es que los datos producidos por estos flujos no se almacenan de forma permanente, sino que se procesan sobre la marcha. Cada dato es analizado, procesado y olvidado, haciendo posible la gestión de grandes cantidades de datos en tiempo real, incluso con capacidades de almacenamiento y procesamiento reducidas.

Los principales problemas de aprendizaje estudiados en la minería de flujo de datos [1], [2] son: (1) clasificación [3], (2) clustering [4] y (3) patrones frecuentes. En los últimos años, la mayor parte de la literatura especializada en este área se ha centrado en clasificación (y *concept-drift*); a pesar de su falta de aplicabilidad en casos reales, lo que ha derivado en experimentaciones basadas únicamente en benchmarks y datos sintéticos. Sería más realista generar modelos descriptivos e interpretables que permitan monitorear sistemas.

En general, el aprendizaje no supervisado es más directamente aplicable a problemas reales de flujo de datos, por lo que el clustering incremental ha experimentado un desarrollo significativo. Sin embargo, el conocimiento que se descubre (segmentación) resulta a menudo insuficiente para ayudar en la toma de decisiones. Por lo tanto, el descubrimiento de patrones frecuentes y reglas de asociación se considera una muy buena manera de abordar muchos problemas de flujo de datos cuyo propósito consiste en supervisar o monitorear (no predecir) en tiempo real usando modelos independientes, significativos, legibles y simples. Más concretamente, el descubrimiento de asociaciones en flujos de datos mediante la producción de reglas de asociación en un proceso completamente on-line, es particularmente interesante debido a: (1) la demanda de interpretabilidad de los patrones descubiertos en los datos, (2) la necesidad de descubrir patrones a medida que suceden, y (3) los altos y continuos volúmenes de datos a procesar, que exigen algoritmos escalables. Un caso real que supone un buen ejemplo de la utilidad de este campo es la detección de amenazas potenciales para los sitios web y las infraestructuras de red [5]. Existen otras estrategias de detección de anomalías (estadísticas o basadas en densidad), pero típicamente se basan en datos etiquetados y, por lo tanto, no se adaptan a nuevos conceptos. Otro posible caso de uso es el analizado en este documento, en el que se trabaja sobre distintos tipos de información relacionada con el uso del teléfono móvil.

El objetivo de este trabajo es mostrar el potencial de la minería de reglas de asociación en flujo de datos al tratar con varios meses de datos reales de uso de teléfonos móviles (tasa de muestreo de un minuto) proporcionados por el MIT Media Lab. El objetivo final es mantener dinámicamente un conjunto de reglas de asociación que expliquen la actividad del usuario

Este trabajo ha sido financiado por los fondos MINECO/FEDER (TIN2017-89517-P), y por el Proyecto BigDaP-TOOLS - Ayudas Fundación BBVA a Equipos de Investigación Científica 2016. E. Ruiz disfruta de un contrato vinculado al proyecto TIN2014-57251-P del MINECO.

en cualquier momento de forma muy eficiente.

El resto de este documento está organizado de la siguiente manera: La sección II presenta las principales características de Fuzzy-CSar (el algoritmo utilizado en este estudio). La sección III proporciona una descripción del estudio realizado por los investigadores del MIT y explica el proceso de preparación que hemos aplicado a los datos. La sección IV presenta los resultados obtenidos. Finalmente, la Sección V resume y concluye el trabajo.

II. Fuzzy-CSar

Fuzzy-CSar [6] está diseñado para extraer reglas de asociación difusas de flujos de datos mediante la combinación de un algoritmo genético (GA) y mecanismos de aportación de crédito de forma on-line. Es uno de los pocos algoritmos capaces de generar directamente reglas de asociación (no solo *itemsets* frecuentes) con atributos tanto cuantitativos como cualitativos de forma puramente on-line, sin emplear ventana deslizante ni ninguna otra técnica para almacenar datos. Gracias a estas propiedades, el algoritmo encaja perfectamente con el propósito de este trabajo.

Fuzzy-CSar mantiene una población de individuos, donde cada uno está representado por una regla de asociación difusa y un grupo de parámetros que evalúan la calidad de la regla. La regla de asociación difusa consiste en un antecedente y un consecuente. Se permite que el antecedente tenga un número arbitrario de atributos mientras que el consecuente consiste en un solo atributo que no debe estar presente en el antecedente de la misma regla. Cada variable puede estar representada en la regla por una disyunción de términos lingüísticos (etiquetas) para facilitar una mayor generalización. Cada individuo tiene un total de ocho parámetros de calidad.

En cada iteración del proceso de aprendizaje de Fuzzy-CSar se recibe un nuevo ejemplo y el algoritmo lleva a cabo una serie de pasos para actualizar los parámetros de los individuos de la población y descubrir nuevas reglas relevantes. Para descubrir estas nuevas reglas prometedoras, se aplica un algoritmo genético estacionario basado en nichos [7]. Además, se aplican operadores de cruce, distintos tipos de mutación y *covering* con ciertas probabilidades. Podemos ver un esquema de la fase de aprendizaje de Fuzzy-CSar en el algoritmo 1. Complementariamente, explicamos brevemente algunos componentes de la fase de aprendizaje, una explicación más detallada se puede encontrar en [6].

Algoritmo 1: Esquema de la fase de aprendizaje de Fuzzy-CSar [6]

```
proceso TrainFuzzy-CSar( ejemploEntrenamiento e_t,
 Población [P] en el instante t )
Data: e_t tiene la forma {x_i}_{i=1}^{ℓ}
Result: Población [P] en el instante t + 1
begin
    e'_t ← granulación(e_t);
    genera [M] a partir de [P] usando e'_t;
    if |[M]| < θ_mna then
        genera θ_mna − |[M]| individuos que concuerdan usando e'_t y
          actualizando [P];
    end
    agrupa individuos en [M] por su antecedente formando distintos [A]_i;
    selecciona [A] según probabilidad;
    subsume individuos en [A];
    actualiza individuos en [M] ; // Por lo tanto, todos los
      [A]_i se actualizan.
    if el tiempo medio en [A] desde la última vez de GA > θ_GA then
        se lleva a cabo un evento genético en [A] considerando e'_t y
          actualizando [P];
    end
end
```

A. Parámetros de Calidad: Soporte y Confianza

En Fuzzy-CSar tenemos ocho parámetros de calidad para cada individuo de la población. Vamos a explicar cómo se calculan dos de ellos, los dos más utilizados y los más significativos para este trabajo: soporte y confianza (información sobre el cálculo de otros parámetros en [6]).

Si definimos formalmente el campo de minería de reglas de asociación como sigue [8]: Siendo $I = i_1, i_2, ..., i_l$ un conjunto de características binarias (ítems) de l elementos. Siendo $Tr = tr_1, tr_2, ..., tr_N$ un conjunto de N transacciones donde cada transacción tr_j contiene un vector binario que indica en cada posición si un ítem en particular está presente o no. Entonces un ítem X ($X \subset I$) va a tener asociado un soporte que es una medida de su importancia en T y se calcula como $supp(X) = |X(T)|/|T|$, donde $X(T)$ es el conjunto de variables en el antecedente de la regla. Si el soporte de un determinado *itemset* (conjunto de ítems) supera un umbral definido por el usuario (*minsupp*) este *itemset* se considera como un *conjunto frecuente de ítems*. Si X e Y son ambos conjuntos frecuentes de ítems y $X \cap Y = \emptyset$, podemos definir una regla de asociación como una implicación del tipo $X \rightarrow Y$. El soporte y la confianza de una regla de asociación son las medidas cualitativas tradicionalmente más usadas:

$$supp(X \rightarrow Y) = \frac{supp(X \cup Y)}{|T|}, conf(X \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}. \quad (1)$$

Donde el soporte indica la frecuencia con la que se cumplen los patrones y la confianza evalúa la fuerza de la implicación indicada en la regla de asociación.

Sea $I = \{i_1, i_2, \ldots i_\ell\}$ un conjunto de ℓ características, $A \subset I$, $C \subset I$ y $A \cap C = \emptyset$. Una regla de asociación difusa es una implicación de la forma $X \rightarrow Y$ en la cual:

$$X = \bigwedge_{i_i \in A} \mu_{\widetilde{A}}(i_i) \quad \text{e} \quad Y = \bigwedge_{i_j \in C} \mu_{\widetilde{C}}(i_j), \quad (2)$$

donde $\mu_{\widetilde{C}}(i_j)$ es el grado de pertenencia de la variable en el consecuente y $\mu_{\widetilde{A}}(i_i)$ es el grado de pertenencia de las variables del antecedente. En esta situación, el soporte se extiende usando el producto T-norm y la confianza se extiende usando la *implicación de Dienes* [9]:

$$supp(X \rightarrow Y) = \frac{1}{|T|} \sum \mu_{\widetilde{A}}(X) \cdot \mu_{\widetilde{C}}(Y) \quad (3)$$

$$conf(X \rightarrow Y) = \frac{\sum(\mu_{\widetilde{A}}(X) \cdot \max\{1 - \mu_{\widetilde{A}}(X), \mu_{\widetilde{C}}(Y)\})}{\sum \mu_{\widetilde{A}}(X)}. \quad (4)$$

donde $\mu_{\widetilde{A}}(X)$ es el grado de pertenencia de la parte del antecedente de la regla y $\mu_{\widetilde{C}}(Y)$ es el grado de pertenencia de la parte del consecuente de la regla.

B. Operador de Covering

Este operador genera nuevas reglas de asociación difusas cuando hay menos de θ_{mna} (siendo θ_{mna} un parámetro de configuración) individuos en el *match set [M]*, individuos de la población actual que concuerdan con el ejemplo recibido e.

El operador de covering construye un nuevo individuo que concuerda con e en el máximo grado posible: para cada variable de entrada de e, e_i, el operador decide aleatoriamente si e_i va a formar parte del antecedente de la regla. Después, elige la variable que estará en el consecuente de entre aquellas que no han sido seleccionadas para formar parte del antecedente.

C. Subsunción de Reglas

Para cada regla de $[A]$ se comprueba su posible subsunción con cada una de las otras reglas del conjunto. Una regla r_i es una candidata para subsumir r_j si: (1) r_i es más general que r_j, y (2) ambas reglas tienen confianzas similares y r_i tiene la suficiente experiencia. Una regla r_i es considerada más general que r_j si todas las variables de r_i están también definidas en r_j y, para cada una de estas variables, r_i tiene, al menos, los mismos términos lingüísticos que r_j.

D. Descubrimiento de Nuevas Reglas

Fuzzy-CSar usa un algoritmo genético incremental estacionario basado en nichos para descubrir nuevas reglas. Este algoritmo genético que se aplica al *association set* seleccionado, cuenta con tres tipos diferentes de mutación: (1) mutación del antecedente; (2) mutación del consecuente, y (3) mutación de términos lingüísticos.

En Fuzzy-CSar, como es común entre los miembros de la familia Michigan-style LCS [10], el coste del algoritmo se incrementa linealmente con el tamaño máximo de la población de reglas, el máximo número de variables por regla, y semilogarítmicamente con el coste de ordenar el *match set*. Es importante resaltar que Fuzzy-CSar no depende directamente del número de transacciones, lo que lo hace muy adecuado para trabajar con grandes bases de datos. En el estudio aquí presentado, el algoritmo se aplica en un problema real (el cual se explica a continuación). La eficiencia es un requisito muy importante en minería de flujo de datos. En el problema aquí abordado, hemos estimado que el tiempo medio que Fuzzy-CSar tarda en procesar cada dato es de unos 15 milisegundos, siendo así factible procesar más de 60 muestras por segundo.

III. Estudio 'Friends and Family'

A. Datos Originales

El *estudio Friends and Family*[1] es una investigación llevada a cabo por el MIT Media Lab, durante los años 2010 y 2011 [11]. Este estudio transforma una comunidad residencial cercana a una conocida universidad norteamericana en un *laboratorio viviente* durante 15 meses. Durante estos meses, los investigadores del Media Lab presentaron su sistema de registro de interacciones sociales y comportamiento basado en teléfonos móviles. Durante cerca de un año, toda actividad, comunicación y detalle social de las vidas de un gran número de miembros de la mencionada comunidad fue registrado mientras ellos realizaban sus tareas cotidianas con normalidad. Un total de 130 sujetos formaron parte del estudio. Durante el período del estudio, se recogió una gran cantidad de datos que resultó en un conjunto de datos muy completo y longitudinal, bautizado como *Friends and Family dataset*. Dicho conjunto de datos incluye una gran colección de señales basadas en la actividad del teléfono móvil incluyendo comunicación (llamadas y mensajes), aplicaciones instaladas, ejecución de aplicaciones, acelerómetro, dispositivos bluetooth próximos...

El estudio se dividió en dos fases. Una fase piloto de 6 meses de duración que comenzó en marzo de 2010, y una segunda fase iniciada en septiembre de 2010. Hasta 130 sujetos participaron en esta segunda fase.

Se proporcionaron *smartphones* a los participantes del estudio con la condición de que estos debían ser sus teléfonos principales durante su participación en el estudio. Estos dispositivos harían el papel de sensores sociales *in-situ* para registrar las características de la actividad de los sujetos.

Parte de la colección de datos obtenida del "Estudio Friends and Family" fue publicada y ha servido como punto de partida para nuestro estudio. El tamaño de esta parte publicada de la colección supera los 7GB. Además, esta gran cantidad de datos está distribuida en distintos archivos cuyo origen, formato y estructura varían.

Los datos recogidos de los teléfonos móviles son el núcleo principal de la colección de datos construida a partir del estudio. La tabla I enumera algunos tipos de información incluidos en esta colección y utilizados en nuestro estudio, junto con sus frecuencias de muestreo originales.

Tabla I: Principales tipos de información incluidos en los datos originales y usados en nuestro estudio

Información	Frecuencia muestreo
Dispositivos bluetooth próximos	cada 5 minutos
Registro de llamadas	cuando una llamada es enviada/recibida
Registro de SMS	cuando un SMS es enviado/recibido
Aplicaciones en el dispositivo	cada 10 minutos
Aplicaciones en ejecución	cada 30 segundos

Dado el origen y las peculiaridades de la información recogida, el estudio se llevó a cabo bajo estrictos protocolos que aseguran la privacidad de todos los participantes.

B. Preparación del Flujo de Datos

La estructura original de los datos no era la más adecuada para el proceso de extracción de conocimiento ni para obtener resultados de calidad. Por lo tanto, los datos tuvieron que ser tratados antes de aplicar el algoritmo. Los datos se encontraban distribuidos en diferentes archivos con diferente formato, estructura y frecuencia de muestreo dependiendo del tipo de información y del método de recolección utilizado. Cada tipo de información había sido registrado con sus propias peculiaridades, los datos de todos los participantes estaban mezclados y no todos los sujetos estuvieron implicados al mismo nivel en el estudio. Fue necesario aplicar un proceso de preparación sobre los datos en bruto para obtener un conjunto de datos completo, unificado y específico para cada sujeto.

[1] http://realitycommons.media.mit.edu/friendsdataset.html

En primer lugar, se realizó un estudio exploratorio de los datos para conocer y comprender mejor el problema. No toda la información original resulta útil para nuestros objetivos. Es necesario decidir qué información es relevante y cuál no.

Un algoritmo evolutivo completamente en línea presenta algunas peculiaridades dado que cada dato va a ser procesado una sola vez y el algoritmo no va a tratar con el conjunto de datos completo en ningún momento. Esto puede hacer que cierta información aparezca solo por un momento para acabar desapareciendo para el algoritmo a pesar de que puedan proporcionar información relevante. Tratando de minimizar este riesgo, se generaron variables del tipo "cuántas llamadas se han registrado en los últimos X minutos". Finalmente, se eligió una frecuencia de muestreo unificada de un minuto para los conjuntos de datos finales tratando de no perder demasiado detalle pero, al mismo tiempo, intentando evitar que el nivel de granularidad de los datos se vuelva innecesariamente bajo.

Para clarificar y resumir, describimos cada paso de la preparación de datos.

1) **Filtrado de datos por fecha**: se seleccionan los datos a partir del 01/10/2010, excluyendo los de la fase piloto.
2) **Agrupamiento de datos por participante**: se agrupa todos los datos de cada participante.
3) **Duplicados y orden cronológico**: eliminamos registros duplicados y ordenamos cronológicamente.
4) **Integración de datos**: los datos libres de duplicados y ordenados cronológicamente recogidos durante la segunda fase del estudio para cada participante se integran en conjuntos de datos individuales.

La figura 1 representa la evolución en el tiempo para los principales atributos después del proceso de preparación de los datos. Los datos presentados en dicha figura corresponden a un sujeto elegido como ejemplo para el análisis de resultados (Sección IV). El gráfico de la figura 1 tiene una resolución diaria, por lo que debemos tener en cuenta que los valores mostrados para las variables acumulativas (llamadas y contadores de mensajes) en los gráficos para un día dado son diez veces el número real de llamadas/mensajes. Esta figura nos ayuda a entender lo difícil que sería extraer información útil y descubrir asociaciones interesantes sin la asistencia proporcionada por el algoritmo de minería de flujo de datos.

IV. Experimentación y Resultados

A. Diseño de Experimentos

Los atributos de entrada utilizados en la experimentación realizada para obtener los resultados presentados en este trabajo son los siguientes: día de la semana; minuto del día [0, 1439]; porcentaje de batería; tres contadores de SMS en los últimos 10 minutos (entrantes, salientes y global); cuatro contadores de llamadas durante los últimos 10 minutos (entrantes, salientes, perdidas y global); tres variables referentes a acelerómetro; alguna aplicación desinstalada y alguna aplicación en ejecución.

La tabla II muestra los valores asignados a los principales parámetros de configuración de Fuzzy-CSar para los experimentos (cuyos resultados se discuten a continuación).

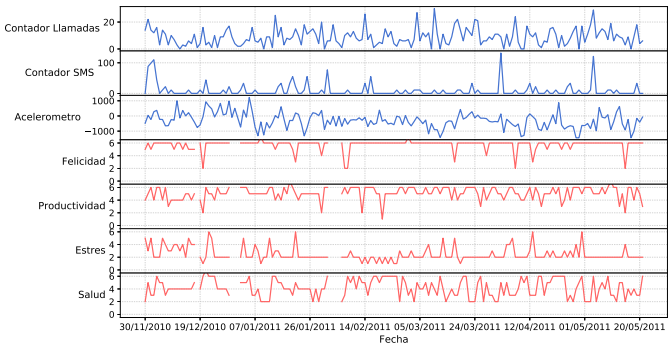


Figura 1: Representación gráfica de la evolución de algunos atributos que forman el conjunto de datos de uno de los participantes (Identificador del participante: sp10-01-24)

Tabla II: Principales parámetros de configuración de Fuzzy-CSar junto con los valores asignados en los experimentos

Parámetro	**Valor**
Tamaño máximo población	10.000
Comprobar subsunción	Sí
Tipo de *association sets*	Antecedente
Número máximo de conjuntos difusos	Dependiente del rango de la variable
Forzar etiquetas adyacentes	Sí
Número máximo de término lingüísticos	60% de los conjuntos difusos
Mutación	Sí
Comprobar subsunción en el GA	Sí

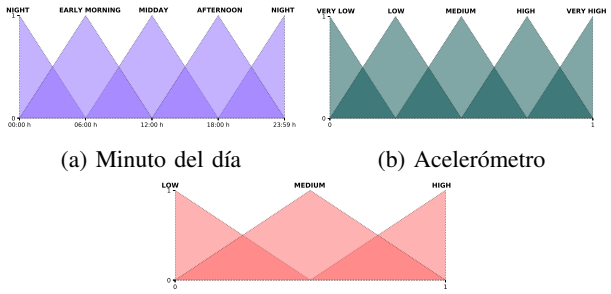
B. Análisis de Resultados

Como se ha explicado, Fuzzy-CSar mantiene continuamente una población de reglas. Es complicado encontrar una manera de trazar la evolución completa de toda esta población. Para poder representar y analizar esta evolución, nos centramos en ciertas reglas completas, consecuentes y antecedentes.

La figura 2 puede ayudar a entender mejor la descripción de algunas reglas de asociación que se muestran en esta sección. En ella es posible observar un esquema de la nomenclatura empleada para referirnos a los conjuntos difusos usados por Fuzzy-CSar para atributos numéricos.

Según estas nomenclaturas, la tabla III muestra tres reglas obtenidas por Fuzzy-CSar. Estos ejemplos ayudan a entender y visualizar mejor la estructura de las reglas de asociación obtenidas por Fuzzy-CSar. En la figura 3 podemos ver la evolución del número de copias almacenadas en la población (numerosidad), lo que representa la importancia relativa de la regla en cada momento, para cada una de las reglas representadas en la tabla III. Podemos observar cómo esta evolución es completamente diferente para cada regla. R_1 (rojo) aparece en un cierto momento, luego la numerosidad de la regla aumenta, para posteriormente disminuir hasta que la regla desaparece. Pero después de eso, la regla aparece de nuevo y repite el mismo proceso. Sin embargo, la regla R_2 (azul) aparece cerca del mes de abril para continuar aumentando su numerosidad hasta el final del experimento. Finalmente, la regla R_3 (verde) existe en la población desde el principio y su numerosidad es siempre creciente. Si analizamos la asociación representada por R_3 esta evolución parece muy lógica ya que normalmente

la gente duerme por la noche y no hace ninguna llamada por lo que esta condición es casi siempre cierta.



(a) Minuto del día (b) Acelerómetro

(c) Llamadas y SMS

Figura 2: Nomenclatura empleada para los conjuntos difusos en las variables referentes a: (a) minutos del día, (b) acelerómetro, y (c) contadores de llamadas y mensajes

Tabla III: Ejemplos de reglas generadas por Fuzzy-CSar durante los experimentos (figura 3)

ID	Rule	Soporte*	Confianza*
R_1	Si hora es MIDDAY o AFTERNOON entonces acelerómetro es MEDIUM, HIGH, VERY HIGH	0.124	0.802
R_2	Si número de SMS salientes es LOW entonces acelerómetro es HIGH, VERY HIGH	0.606	0.709
R_3	Si hora es NIGHT o EARLY MORNING y número de SMS entrantes es LOW entonces número de llamadas salientes es LOW	0.496	0.997

* El soporte y confianza se refieren al momento con máximo número de copias

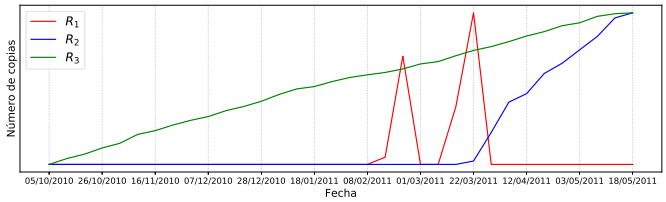


Figura 3: Evolución del número de copias en la población (numerosidad) de: R_1 (máximo 21 copias), R_2 (máximo 90 copias) y R_3 (máximo 537 copias) (tabla III)

Generalizando un poco nuestro análisis podemos centrarnos en un consecuente determinado en lugar de en una regla determinada. Siguiendo esta idea presentamos una serie de gráficos en los que analizamos la evolución en numerosidad de un conjunto específico de reglas que contienen la misma variable en su consecuente o incluso la misma etiqueta. Como prueba de concepto, a continuación se analiza la evolución de la cantidad de actividad física practicada por el participante.

La variable *AccelAccum*, incluida en el conjunto de datos usado, presenta valores relacionados con la información de acelerometría recogida de los *smartphones* de los participantes. Los investigadores del MIT relacionan dicha información de acelerometría con la actividad física, como también lo hacen otros investigadores [12]. Siguiendo esta interpretación,

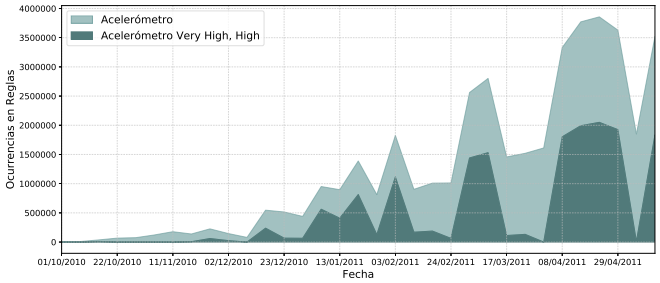


Figura 4: Comparativa entre la evolución en el nº total de reglas con *AccelAccum* en el consecuente y el número de ellas con la etiqueta VERY HIGH, HIGH ($supp \geq 0.1$ y $conf \geq 0.7$). Datos del participante *sp10-01-24*

entendemos que un mayor número de reglas cuyos consecuentes apuntan a valores *HIGH, VERY-HIGH* de la variable *AccelAccum* puede ser interpretado como un indicador de un aumento de la actividad física practicada por el sujeto. La figura 4 representa la evolución en el número de reglas, que superan los umbrales de soporte y confianza, cuyos consecuentes contienen la variable *AccelAccum* con cualquier etiqueta (área color claro) y las reglas cuyos consecuentes contienen valores altos de la variable *AccelAccum* (área color oscuro). Teniendo esto en cuenta, el gráfico representado en la figura 4 muestra cómo el participante *sp10-01-24* aumenta la cantidad de actividad física durante ciertos períodos de tiempo.

La figura 5 nos ayuda a entender la distribución de los valores de *AccelAccum* durante un cierto periodo de tiempo. Como se observa, no hay concentraciones especiales de valores altos o muy altos. El algoritmo comienza a encontrar relaciones que explican los altos valores de *AccelAccum* y, como consecuencia, aumenta el número de reglas con este consecuente específico y suficiente confianza.

Dado que se ha utilizado un algoritmo de reglas de asociación, y no uno de patrones frecuentes, podemos utilizar las relaciones establecidas por las reglas de asociación entre antecedentes y consecuentes como parte de nuestro análisis.

Así, profundizamos en los resultados analizando la composición del antecedente de las reglas con altos valores de *AccelAccum* en el consecuente. La figura 6 representa la evolución de estas reglas de calidad distinguiendo entre variables en el antecedente. En esta figura se muestra cómo la mayoría de los antecedentes se relacionan con la no utilización de varias funciones del teléfono móvil, por ejemplo, llamadas, mensajes, aplicaciones.... Los antecedentes agrupados bajo la etiqueta "Otro" se relacionan principalmente con la hora y el día de la semana. Este hecho refuerza la idea de que los valores altos del acelerómetro están relacionados con la actividad física y no con otros tipos de uso del *smartphone*.

V. Conclusiones

En este trabajo hemos mostrado una aplicación real de un algoritmo evolutivo (Fuzzy-CSar) para minería de reglas de asociación en flujo de datos, cuyo objetivo es descubrir de forma on-line y en tiempo real las relaciones entre los atributos

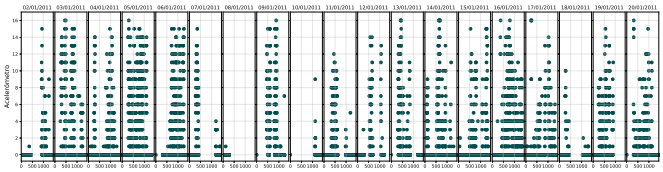


Figura 5: Distribución de los valores de la variable *AccelAccum* a lo largo de los minutos de cada día del 2 al 21 de enero de 2011. Datos del participante *sp10-01-24*

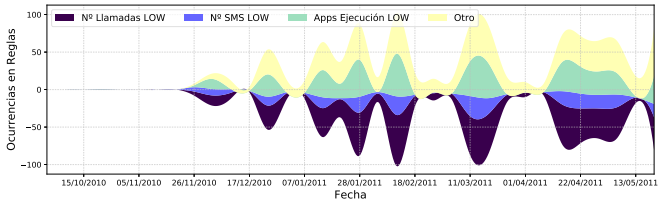


Figura 6: Comparación entre la evolución del nº total de reglas ($supp \geq 0.1$ y $conf \geq 0.7$) cuyo consecuente contiene *AccelAccum* con etiqueta VERY HIGH, HIGH y el nº de ellas con antecedentes referidos a la inactividad del teléfono. Datos del participante *sp10-01-24*

de un flujo de datos. Esta aplicación simula un sistema de monitorización en el que los datos de entrada consisten en muestras reales basadas en información sobre llamadas, SMS, acelerómetros, aplicaciones, etc. recogida desde *smartphones*. Estos datos constituyen un claro ejemplo de problema real en el que la información se genera como un flujo continuo e infinito y puede ser explotada de forma muy productiva sin necesidad de ser almacenada en grandes conjuntos de datos que requieran altas capacidades de procesamiento.

En este caso, el algoritmo evolutivo descubre reglas en tiempo real para explicar lo que está sucediendo en todo momento. Los resultados muestran la evolución que está experimentando la población de reglas de asociación a medida que aumenta la cantidad de datos procesada. Estos son solo algunos ejemplos que muestran cómo el algoritmo Fuzzy-CSar y las reglas de asociación descubiertas por él, hacen posible descubrir nuevas relaciones que no habrían sido descubiertas directamente a partir de datos en bruto. Además, esto se consigue de una manera muy eficiente, ya que Fuzzy-CSar tarda solo 15 ms en procesar cada dato, es decir, en el caso específico de datos con esta estructura es capaz de lidiar con una frecuencia de entrada de unos 67 Hz. En un entorno donde los datos llegan en tiempo real en forma de flujo infinito, como en este trabajo, es posible generar y actualizar un modelo en tiempo real, permitiendo un proceso de monitoreo que puede ayudar en un sistema de toma de decisiones. Otro posible caso de uso podría ser integrar este algoritmo en una aplicación móvil que utilice directamente el conocimiento descubierto por el algoritmo para tomar decisiones (por ejemplo, recomendaciones musicales o sugerencia de aplicaciones) basadas en reglas específicas. Dado que el algoritmo no asume ninguna estructura de problema a priori, es capaz de adaptarse a las características de los datos de cada sujeto. Cabe destacar la importancia de utilizar un algoritmo con capacidad para adaptarse a los cambios conceptuales, ya que a menudo estos cambios proporcionan la información más relevante.

En conclusión, en este trabajo se muestra el potencial de la minería de reglas de asociación en flujo de datos para la monitorización de problemas reales. El trabajo realizado y los resultados obtenidos revelaron que el desarrollo de un sistema capaz de monitorizar el uso que una persona está haciendo de su teléfono a través de un algoritmo de minería de asociaciones en flujos de datos (utilizando técnicas on-line e incrementales), descubriendo información útil, es una opción factible. En este momento, se están considerando varias líneas para continuar con este trabajo, entre las que se incluyen las siguientes: (1) continuar con las mejoras de Fuzzy-CSar y pulir las adaptaciones del algoritmo a los datos de actividad de teléfonos móviles, (2) estudiar más profundamente los resultados obtenidos en este conjunto de datos, (3) aprovechar la buena eficiencia de Fuzzy-CSar para la integración en dispositivos móviles, y (4) desarrollar aplicaciones para obtener más datos de este tipo utilizando luego la información relevante descubierta a través de las asociaciones aprendidas.

References

[1] J. Gama, *Knowledge discovery from data streams*. Chapman & Hall/CRC, 2010.

[2] M. Sayed-Mouchaweh and E. Lughofer, *Learning in non-stationary environments : methods and applications*. Springer, 2012.

[3] A. Orriols-Puig and J. Casillas, "Fuzzy knowledge representation study for incremental learning in data streams and classification problems," *Soft Computing*, vol. 15, no. 12, pp. 2389–2414, dec 2011.

[4] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: theory and practice," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 515–528, may 2003.

[5] G. Corral, A. Garcia-Piquer, A. Orriols-Puig, A. Fornells, and E. Golobardes, "Analysis of vulnerability assessment results based on CAOS," *Applied Soft Computing*, vol. 11, no. 7, pp. 4321–4331, oct 2011.

[6] A. Sancho-Asensio, A. Orriols-Puig, and J. Casillas, "Evolving association streams," *Information Sciences*, vol. 334-335, pp. 250–272, mar 2016.

[7] S. W. Wilson, "Classifier Fitness Based on Accuracy," *Evolutionary Computation*, vol. 3, no. 2, pp. 149–175, jun 1995.

[8] R. Agrawal, T. Imieliński, A. Swami, R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93*, vol. 22, no. 2. New York, New York, USA: ACM Press, 1993, pp. 207–216.

[9] D. Dubois, E. Hüllermeier, and H. Prade, "A systematic approach to the assessment of fuzzy association rules," *Data Mining and Knowledge Discovery*, vol. 13, no. 2, pp. 167–192, sep 2006.

[10] A. Orriols-Puig, J. Casillas, and F. J. Martínez-López, "Unsupervised Learning of Fuzzy Association Rules for Consumer Behavior Modeling," *Mathware & Soft Computing*, vol. 16, pp. 29–43, 2009.

[11] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland, "Social fMRI: Investigating and shaping social mechanisms in the real world," *Pervasive and Mobile Computing*, vol. 7, no. 6, pp. 643–659, dec 2011.

[12] J. Bort-Roig, N. D. Gilson, A. Puig-Ribera, R. S. Contreras, and S. G. Trost, "Measuring and Influencing Physical Activity with Smartphone Technology: A Systematic Review," *Sports Medicine*, vol. 44, no. 5, pp. 671–686, may 2014.

Un Sistema de Recomendación de Asignaturas Multi-Criterio con Optimización Genética

Aurora Esteban, Amelia Zafra, Cristóbal Romero
Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Córdoba, España

***Resumen*—Este artículo propone un Sistema de recomendación (SR) híbrido multi-criterio para resolver el problema de recomendación de asignaturas en los estudios universitarios. Por un lado, utiliza Filtrado Colaborativo (FC) con la información del estudiante y por otro lado, Filtrado Basado en Contenido (FBC) con la información de las asignaturas. Para determinar los factores que resultan más relevantes en la recomendación, así como para optimizar la configuración del SR híbrido en lo referente a las medidas de distancia a considerar y el tamaño del vecindario, se ha diseñado un Algoritmo Genético (AG) que optimiza de forma automática el ajuste de todos los parámetros. Un estudio experimental con 2500 valoraciones reales de los estudiantes del Grado en Ingeniería Informática de la Universidad de Córdoba, demuestra los excelentes resultados obtenidos por el SR propuesto y la importancia de utilizar un modelo híbrido donde los diferentes criterios utilizados sean ponderados de acuerdo a su relevancia.**

***Index Terms*—Sistema de recomendación, Filtrado colaborativo, Filtrado basado en contenido, Algoritmo genético**

I. Introducción

En la era de la información que recientemente se vive, los Sistemas de Recomendación (SR) se han consolidado en diversos campos, entre los que se encuentran los entornos educativos. Específicamente, la recomendación de asignaturas o cursos se ha afianzado como una interesante y creciente línea de investigación dentro de la minería de datos aplicada a la educación [1]. En el caso concreto de la recomendación de asignaturas en los estudios universitarios, su importancia viene dada por el hecho de que los estudios universitarios tienen un número variable de asignaturas optativas entre las que los estudiantes deben elegir para completar los créditos de la titulación que están cursando. Normalmente, los estudiantes no cuentan con información suficiente para hacer esta elección, y recurren a compañeros que la hayan cursado para conocer su opinión. En este contexto, los SR se presentan como una herramienta esencial, capaz de ofrecerles asignaturas relevantes en base a sus preferencias individuales, sus intereses, sus necesidades o su rendimiento [2].

Aunque existen algunos estudios que trabajan con enfoques de SR híbridos [3], [4] o con múltiples criterios [5], [6], no incluyen todos los criterios aquí considerados, ni se centran en estudiar la influencia que los diferentes factores tienen en el proceso de recomendación.

En este trabajo se presenta un SR híbrido multi-criterio, que combina una técnica de Filtrado Colaborativo (FC) utilizando información del estudiante (considerando como criterios, sus valoraciones, sus calificaciones y la especialidad cursada dentro de la carrera), con una técnica de Filtrado basado en Contenido (FBC) utilizando información de las asignaturas (considerando como criterios, sus competencias, profesores, contenidos teóricos o prácticos y área de conocimiento). Para determinar la importancia de cada criterio se propone un Algoritmo Genético (AG) que optimice los pesos asignados a cada uno de estos criterios en el SR. Así mismo, otros parámetros del SR como las métricas de similitud utilizadas, o el número de vecinos también serán optimizadas simultáneamente. En la metodología propuesta, el SR es configurado con los parámetros optimizados por el AG, para finalmente, realizar las recomendaciones finales a los estudiantes.

Con el fin de garantizar una evaluación rigurosa del SR, se va a utilizar un conjunto de datos proveniente del Grado en Ingeniería Informática de la Universidad de Córdoba (España), que incluye valoraciones y calificaciones de estudiantes reales. Así mismo, en el proceso de evaluación se utilizará validación cruzada manteniendo el equilibrio entre las particiones de datos.

El resto del trabajo se organiza como sigue. En la sección II se hace un repaso de los trabajos previos. En la sección III se especifica la metodología propuesta, describiendo tanto la información, como el SR híbrido diseñado. En la sección IV se explica el AG utilizado para la optimización. En la sección V se describe el estudio experimental realizado, tanto el estudio de la relevancia de los criterios, como la comparativa con otros modelos. Finalmente, la sección VI presenta las conclusiones obtenidas y el trabajo futuro.

II. Trabajos relacionados

En los últimos años los SR multi-criterio han sido ampliamente aplicados a la recomendación de asignaturas o cursos. S. Spiegel [7] explora una de las primeras aplicaciones de factorización de matriz multi-criterio para la predicción de valoraciones de asignaturas. Más adelante, Vialardi et al. [5] propusieron técnicas multi-criterio para la predicción de calificaciones de estudiantes abordadas

como un problema de clasificación, y Parameswaran et al. [8] exploraron la aplicación de restricciones a la recomendación multi-criterio. Así mismo, se ha explorado la aplicación de otras técnicas, como propuestas basadas en ontologías [9], [10], redes neuronales [11], y algoritmos bio-inspirados como colonias de hormigas [12] o sistemas inmunes artificiales [3]. La mayoría de estas propuestas sólo están basados en las valoraciones de los estudiantes. Desde otra perspectiva, la importancia del momento en el que se cursan las asignaturas ha sido estudiada usando Cadenas de Markov en base a las calificaciones [13], así como aplicando multi-criterio [14]. Recientemente, se ha estudiado la relevancia de las competencias que las asignaturas ofrecen [6] o la aplicación de análisis semántico [15].

Se puede resumir que la mayoría de los enfoques se centran principalmente en el rendimiento de los estudiantes, y no usan criterios adicionales. Además, los sistemas que utilizan más criterios, no hacen ningún estudio para determinar la influencia de cada uno de ellos en la calidad de las recomendaciones, mostrando así los beneficios que aporta nuestra propuesta.

III. Propuesta de Sistema de Recomendación Híbrido

En esta sección se describe la metodología propuesta, especificando desde la recolección y preparación de los datos, hasta el desarrollo de la propuesta planteada.

A. Descripción y preparación de los datos

Este trabajo se ha desarrollado utilizando los datos recogidos del Grado en Ingeniería Informática de la Universidad de Córdoba (España). Estos datos son relativos tanto a los estudiantes, como a las asignaturas.

1) Información de estudiante: la información de los estudiantes (figura 1) se ha obtenido por medio de encuestas realizadas durante tres cursos académicos (de 2016 a 2018) a los alumnos de los últimos cursos, recogiendo un total de 2500 valoraciones de 63 asignaturas incluidas en el plan de estudios. La información que se ha considerado es:

- Una valoración de la satisfacción general del estudiante en cada asignatura, representada como un valor numérico entero entre 1 y 5. El valor máximo correspondería a un 5, y el mínimo a un 1.
- La calificación obtenida por el estudiante para cada asignatura, representada con un valor numérico real, en el rango $[0, 10]$.
- La especialidad seleccionada por el estudiante. Concretamente, el grado en Ingeniería Informática que se estudia, ofrece tres especialidades: computación, computadores e ingeniería del software, representados con un identificador numérico de 1 a 3.

2) Información de la asignatura: la información de las 63 asignaturas consideradas en el plan de estudios (figura 2), se ha obtenido de la página oficial del grado dentro de la Universidad[1]. La información que se ha considerado es:

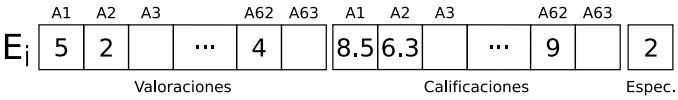


Figura 1. Información del estudiante

- Los profesores que imparten docencia en cada asignatura, representados con un vector binario, indicando con el valor 1 si el profesor ha impartido la asignatura y con el valor 0 si no la ha impartido.
- Las competencias que se deben adquirir al realizar la asignatura, representadas con un vector binario, indicando con el valor 1 que la asignatura proporciona la competencia, y con el valor 0 que no la proporciona.
- El área de conocimiento al que la asignatura pertenece, representada con un identificador numérico de 1 a 8, ya que en el grado estudiado se contemplan ocho áreas de conocimiento diferentes.
- Los contenidos teóricos y prácticos de la asignatura, representados con un vector de frecuencias de las palabras claves. Las palabras claves son obtenidas a partir del procesamiento automático de textos aplicado a la guía docente de cada asignatura.

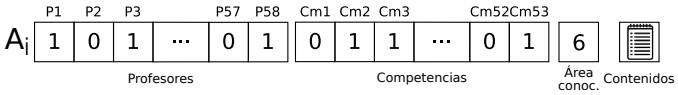


Figura 2. Información de la asignatura

B. Sistema de Recomendación

El SR híbrido multi-criterio propuesto en este trabajo combina las valoraciones obtenidas por un sistema basado en FC (especificado en la sección III-B1 y las obtenidas por un sistema de FBC (especificado en la sección III-B2). De este modo, la estimación de preferencia p de un estudiante i sobre una asignatura j, se calcula como la combinación lineal de las preferencias dadas por el FC ($FC_{i,j}$) y el FBC ($FBC_{i,j}$):

$$p_{i,j} = \alpha \cdot FC_{i,j} + \beta \cdot FBC_{i,j} \quad (1)$$
$$Con\ \alpha + \beta = 1$$

Tanto en el sistema FC, como en el FBC se representan las valoraciones estimadas de cada asignatura en un rango de $[1, 5]$, por lo que la estimación final también estará en dicho rango.

Se puede apreciar en la ecuación 1, que los parámetros que determinan la relevancia que se le da a la valoración obtenida por cada uno de los sistemas considerados son los pesos α y β, que deben ser configurados.

[1] http://www.uco.es/eps/node/619

En los siguientes apartados se especifican las características del sistema de FC y del sistema de FBC que se han diseñado.

1) Filtrado Colaborativo utilizando la información del estudiante: en el sistema de FC desarrollado, la valoración estimada de una asignatura para un estudiante se obtiene a partir de las valoraciones que ésta ha recibido por parte del resto de estudiantes con un perfil similar.

El aspecto multi-criterio es introducido en el cálculo de la similitud entre estudiantes: para cada par de estudiantes i y j, la similitud, s, agrega tres medidas de similitud que están en el rango $[0, 1]$: por un lado, calcula la similitud según sus valoraciones $V_{i,j}$ y sus calificaciones $C_{i,j}$, por otro lado, comprueba si la especialidad que están cursando ambos estudiantes coincide ($E_{i,j}$). Finalmente, los tres criterios se agregan en una combinación lineal:

$$s_{i,j} = \alpha \cdot V_{i,j} + \beta \cdot C_{i,j} + \gamma \cdot E_{i,j} \quad (2)$$
$$Con\ \alpha + \beta + \gamma = 1$$

Por tanto, será necesario configurar las métricas utilizadas para valoraciones y calificaciones, el vecindario, y los pesos α, β y γ de los criterios considerados.

2) Filtrado basado en Contenido utilizando la información de la asignatura: en el sistema de FBC desarrollado, la valoración estimada de una asignatura para un estudiante se obtiene a partir de la valoración que él mismo ha hecho de las asignaturas más similares a ésta.

El aspecto multi-criterio es introducido en el cálculo de la similitud entre asignaturas: para cada par de asignaturas i y j, la similitud (s) es una combinación de cuatro medidas que están en el rango $[0, 1]$: por un lado se calcula cuántos profesores y competencias tienen en común, $P_{i,j}$ y $Cm_{i,j}$, respectivamente. Por otro lado, se comprueba si el área de conocimiento coincide $A_{i,j}$, y finalmente, se obtiene la similitud según los contenidos aplicando minería de textos $Cn_{i,j}$, siguiendo el procedimiento especificado más adelante. Finalmente, se calcula la similitud final mediante una combinación lineal:

$$s_{i,j} = \alpha \cdot P_{i,j} + \beta \cdot Cm_{i,j} + \gamma \cdot A_{i,j} + \delta \cdot Cn_{i,j} \quad (3)$$
$$Con\ \alpha + \beta + \gamma + \delta = 1$$

Por tanto, para aplicar FBC será necesario determinar las métricas usadas para $P_{i,j}$ y $Cm_{i,j}$ y la relevancia de cada criterio por medio de los pesos α, β, γ y δ.

La similitud según los contenidos de la asignatura es obtenida a partir de la información descrita en su guía docente mediante minería de textos. El proceso compara los contenidos de dos asignaturas, para finalmente dar un valor de similitud entre 0 y 1:

1) Indexación del apartado *Contenidos* de las guías docentes de las asignaturas: este proceso se lleva a cabo con un analizador sintáctico y con un conjunto de *stop-words* específicas al dominio. Como resultado se obtiene para cada documento un conjunto de *tokens* junto a sus frecuencias de aparición.
2) Para cada par de asignaturas i y j, se crea un conjunto B que es la unión de sus *tokens*, y para cada asignatura, se crea un vector $\vec{i}$ y $\vec{j}$ que tendrá una posición por cada *token* en B indicando la frecuencia de aparición del mismo. Finalmente, se aplica la norma l_1 sobre cada vector para obtener las frecuencias relativas.
3) Finalmente, a los 2 vectores de frecuencias se les aplica la similitud del coseno, obteniendo el valor $Cn_{i,j}$ que se integra en la ecuación parametrizable descrita en 3:

$$cos(\theta) = \frac{\vec{i} \cdot \vec{j}}{\|\vec{i}\| \cdot \|\vec{j}\|} = \frac{\sum_{k=1}^{n} i_k j_k}{\sqrt{\sum_{k=1}^{n} i_k^2}\sqrt{\sum_{k=1}^{n} j_k^2}} \quad (4)$$

IV. Optimización del Sistema de Recomendación mediante Algoritmos Genéticos

El SR que aquí se presenta, descrito en la sección III-B, cuenta con múltiples criterios que deben ser ponderados con un peso que indique la relevancia que va a tener cada uno de ellos, así como con la configuración de diferentes métricas de similitud para cada uno de ellos, y del tamaño del vecindario a utilizar. Por ello, se ha decidido diseñar un AG que automáticamente descubra una configuración óptima de todos los parámetros del SR y permita obtener unas recomendaciones lo más satisfactorias posibles para los estudiantes.

A continuación, se especifica las características del AG que se ha diseñado y que sigue un esquema generacional con elitismo.

A. Representación de los individuos

Los individuos de la población se codifican mediante un cromosoma entero compuesto de 14 genes:

- Los dos primeros genes son los pesos que se van a utilizar para combinar las recomendaciones obtenidas por el FC utilizando la información del estudiante y por el FBC utilizando la información de la asignatura. Concretamente, serían los valores α y β definidos en la ecuación 1.
- Los tres siguientes genes se corresponden con los pesos que se asignan a los tres criterios que se han considerado en el FC utilizando la información del estudiante. Concretamente, serían los valores de α, β y γ definidos en la ecuación 2.
- Los cuatro siguientes genes codifican los pesos que son asignados a los cuatro criterios utilizados en el FBC que emplea la información de la asignatura. Concretamente, serían los valores de α, β, γ y δ definidos en la ecuación 3.
- El siguiente gen es el número de vecinos tenidos en cuenta para realizar las estimaciones.
- Los dos siguientes genes son las métricas utilizadas para calcular la similitud según las valoraciones y las

calificaciones respectivamente. Sus valores son tratados como categóricos, y se corresponderán con las métricas contempladas para obtener estas similitudes.
- Los dos últimos genes se corresponden con las métricas utilizadas para calcular la similitud según los profesores y las competencias. Al igual que el grupo anterior, tomarán valores categóricos que se corresponderán con las métricas contempladas para obtener estas similitudes.

B. Operadores genéticos

1) El operador de cruce: es un cruce uniforme que selecciona gen a gen de qué padre toma la información que pasará al hijo. La particularidad de este operador es que el conjunto de genes: (1,2), (3,4,5) y (6,7,8,9) son tratados en bloque, de forma que cada bloque completo es seleccionado del mismo padre para que lo herede uno de los hijos.

2) El operador de mutación: es una mutación uniforme, donde para cada individuo que va a ser mutado, se selecciona cada gen con una cierta probabilidad y se cambia su valor por otro valor aleatorio dentro del rango de valores definidos para ese gen.

3) Optimizador local: este operador, similar a los empleados en los algoritmos meméticos [16], consiste en realizar una pequeña modificación en los genes que representan las medidas de similitud y el número de vecinos de un mismo individuo. Si el cambio realizado hace que el individuo tenga una mejor función de aptitud, se continúa realizando cambios, hasta un máximo de cinco variaciones. Finalmente, si el individuo obtenido mejora al individuo inicial, lo sustituye en la nueva población, en caso contrario, el nuevo individuo obtenido será eliminado. Debido, al coste computacional de este operador, solamente es aplicado al mejor individuo de la población.

C. La función de aptitud

La función de aptitud o *fitness* que se emplea es la raíz del error cuadrático medio (RMSE) entre las valoraciones reales y las estimadas. Para optimizar los tiempos de cómputo, el fitness se obtendrá utilizando el 80% de los datos como valores conocidos y el restante 20% como datos a estimar para obtener el valor de RMSE.

V. Estudio experimental

Esta sección describe la experimentación realizada. El SR se ha desarrollado dentro del *framework* Apache Mahout[2] y el AG utilizando la *librería* JCLEC[3]. La máquina sobre la que se han ejecutado las pruebas es un ordenador personal con SO Ubuntu 16.04 64-bits, un procesador Intel Core i5-3317U y 12 GB de RAM.

[2]https://mahout.apache.org/
[3]http://jclec.sourceforge.net/

A. Configuración del Algoritmo Genético

Los principales parámetros utilizados en el AG, tras un estudio de diferentes configuraciones, son: tamaño de la población: 200, número de generaciones: 100, y probabilidad de mutación y cruce: 0.5 y 0.9 respectivamente.

Con respecto al rango de valores de cada gen: los 9 primeros genes que representan los pesos de cada criterio considerado, se han definido en el rango $[0, 10]$. Este rango podría variarse, ya que estos valores son normalizados antes de ser utilizados en las ecuaciones 1, 2 y 3. El gen 10, que representa el número de vecinos, se ha definido en el rango $[1, 50]$. Los genes 11 y 12 que representan las medidas de similitud de las valoraciones y las calificaciones, se definen en el rango $[0, 4]$ representando las 5 métricas consideradas, la distancia euclidiana, la distancia Manhattan, el coeficiente de correlación de Pearson, el coeficiente de correlación de Spearman y la similitud del coseno. Finalmente, los genes 13 y 14 que representan las medidas de similitud de los profesores y las competencias, se definen en el rango $[0, 1]$ representando las 2 métricas consideradas, el índice Jaccard y la función de verosimilitud logarítmica.

B. Influencia de los diferentes criterios en el Sistema de Recomendación

En esta sección, se evalúa la relevancia de cada uno de los criterios considerados en el SR híbrido presentado, para determinar los factores que pueden resultar más significativos para mejorar las recomendaciones.

En la tabla I se muestra la mejor configuración de pesos encontrada por el AG. En el caso de la configuración del FC que emplea información del estudiante, se puede apreciar que el factor más relevante es la calificación obtenida por los estudiantes (peso de 0.53), seguido por las valoraciones dadas a las asignaturas (peso de 0.25) y la especialidad cursada (peso de 0.22). Estos resultados muestran la relevancia de las calificaciones en la búsqueda

Cuadro I
Mejor configuración del SR obtenida por el AG

SR Híbrido	
Peso del FC	0.58
Peso del FBC	0.42
FC (Basado en información del estudiante)	
Valoraciones (métrica sim.)	Distancia Manhattan
Calificaciones (métrica sim.)	Distancia Manhattan
Valoraciones (peso)	0.25
Calificaciones (peso)	0.53
Especialidad (peso)	0.22
Tamaño vecindario	12
FBC (Basado en información de la asignatura)	
Profesores (métrica sim.)	Verosimilitud logarítmica
Competencias (métrica sim.)	Índice de Jaccard
Profesores (peso)	0.54
Competencias (peso)	0.00
Área conoc. (peso)	0.00
Contenido (peso)	0.46

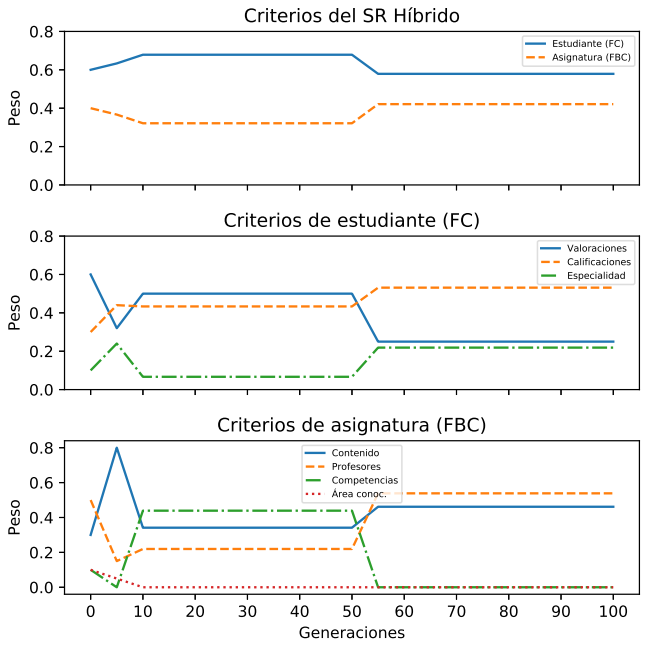


Figura 3. Evolución de los pesos de los criterios en el AG

de estudiantes similares. Así, una asignatura será recomendada a un cierto estudiante si, principalmente, ha sido valorada positivamente por otros estudiantes con unas calificaciones similares a la suya.

Analizando los criterios más relevantes relativos a la información de la asignatura (FBC), se obtiene como factores muy relevantes los profesores que imparten las asignaturas (0.54) y el contenido teórico y práctico de las mismas (0.46); los otros dos criterios (competencias que cubren las asignaturas y el área de conocimiento a la que pertenecen), aparecen como irrelevantes (0.00). Estos resultados, ponen de manifiesto la importancia de los profesores, siendo un factor tan importante como el contenido propio de la asignatura para determinar cómo de similares se perciben dos asignaturas. De esta manera, una asignatura será recomendada a un cierto estudiante si tiene un contenido similar y/o ha sido impartida por el mismo profesorado que otra asignatura que le haya interesado en el pasado.

Finalmente, los pesos asignadas por el AG a las estimaciones realizadas por los sistemas FC y FBC son ponderados con 0.58 y 0.42 respectivamente, mostrando la importancia de considerar tanto información del estudiante, como de la asignatura en la recomendación final.

Para estudiar cómo los pesos han ido evolucionando en las sucesivas generaciones del AG, se analiza la evolución del mejor individuo de la población (Figura 3). Se aprecia que los pesos asignados a las recomendaciones realizadas por el FC y por el FBC, comienzan bastante desequilibradas, dando más importancia al FC. Sin embargo, acaban convergiendo en una importancia similar. En el caso de los pesos asignados a los diferentes criterios considerados en la información del estudiante, se puede apreciar como en las soluciones iniciales el factor de las valoraciones es el más relevante. No obstante, conforme el AG va convergiendo y optimizando el ajuste de parámetros, las calificaciones toman más importancia, pasando valoraciones y especialidad a tener una relevancia similar y menor a las calificaciones. Finalmente, si se estudia la evolución de los pesos asignados por el AG a los diferentes criterios relativos a la información de las asignaturas, se puede apreciar que desde el principio, el área de conocimiento aparece como poco significativa, y es algo que se mantiene hasta el final, mientras que el contenido y los profesores va tomando más relevancia cada vez.

Se puede apreciar que a partir de la generación 60 el peso asignado a los diferentes criterios se mantiene, no siendo mejorados por ninguna otra solución de la población durante el resto de las generaciones. El motivo de que el AG continúe realizando generaciones y optimizando soluciones se debe a que continúa obteniendo mejores soluciones optimizando el resto de parámetros considerados en el AG y configurables en el SR híbrido diseñado, que son las métricas de similitud utilizadas y el tamaño del vecindario.

En la tabla I se muestra las métricas de similaridad y el tamaño de vecindario seleccionadas por el AG. En la figura 4, se muestra la evolución del mejor individuo de la población a lo largo de las generaciones para estos valores. En este caso, es interesante resaltar como estos valores van cambiando, a partir de la generación 60, para optimizar las valoraciones proporcionadas por el SR y en consecuencia reducir el RMSE. Se puede apreciar que existe cierta tendencia a que los criterios de valoraciones y calificaciones utilicen la misma métrica. Además, si esta métrica es la medida de similitud del coseno, se necesita

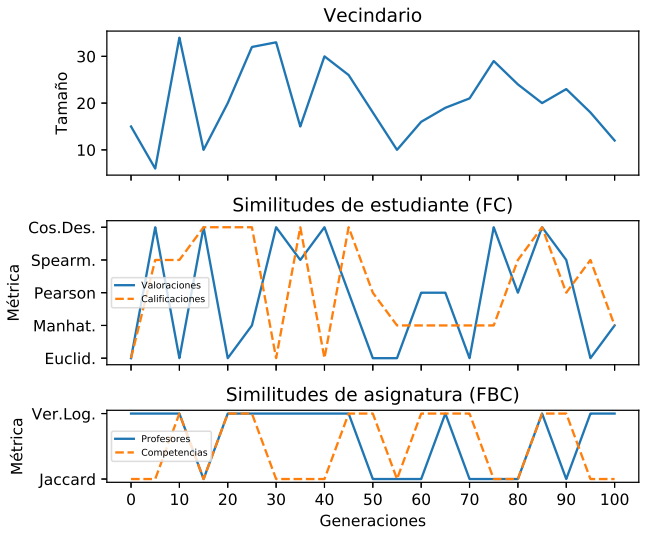


Figura 4. Evolución de las métricas y el vecindario en el AG

un mayor número de vecinos, mientras que para el resto de casos tiende a mantenerse entre 10 y 20.

C. Comparativa con otros modelos

Para concluir el estudio experimental, se compara el rendimiento del SR híbrido frente al uso de los modelos: FC multi-criterio y FBC multi-criterio con la misma configuración de pesos establecida por el AG, pero sin combinar sus resultados. También se incluye en la comparativa el uso de FC mono-criterio empleando solamente el criterio de las valoraciones (el criterio más ampliamente utilizado en los SR) y FBC mono-criterio empleando solamente la información del criterio del profesorado (el más representativo, de acuerdo a los pesos asignados).

Para llevar a cabo este estudio, se ha ejecutado una evaluación del SR utilizando validación cruzada con 10 *fold* con muestreo estratificado. Las métricas estudiadas son el RMSE, relacionado con cuánto difiere una estimación de la valoración real; la ganancia nDCG (*normalized Discounted Cumulative Gain*), relacionada con la capacidad del SR de ofrecer recomendaciones relevantes; el alcance, que determina el porcentaje de usuarios para los que se pueden obtener recomendaciones; y el tiempo de ejecución que necesita el SR para entrenar el modelo y ofrecer una recomendación.

Los resultados finales pueden verse en la tabla II. El enfoque híbrido obtiene mejores resultados en todas las métricas estudiadas. Aunque al usar más información, tarda algo más que los otros modelos en ejecutarse. De este modo, queda demostrada la importancia de utilizar múltiples criterios en el SR, asignándoles además pesos concretos, así como la de realizar una hibridación de diferentes técnicas.

Cuadro II
Comparativa entre modelos de SR

Método	**RMSE**	**nDCG**	**Alcance(%)**	**Tiempo(s)**
SR Híbrido	1.056	0.811	100.00	8.23
FC[1]	1.167	0.806	96.48	7.53
FBC[1]	1.201	0.214	99.36	6.86
FC[2]	1.233	0.798	96.48	5.98
FBC[2]	2.530	0.284	99.36	5.77

[1] multi-criterio, [2] mono-criterio

VI. Conclusiones

En este trabajo se ha desarrollado un SR aplicado a la recomendación de asignaturas híbrido y multi-criterio. Así mismo, se ha implementado un AG que realiza un ajuste de todos los parámetros utilizados en el SR propuesto para determinar las configuraciones óptimas para el SR. Las pruebas realizadas muestran que considerar varios criterios proporciona mejores resultados, pero que la relevancia de cada uno de ellos debe ser estudiada, ya que no todos los factores resultan igual de relevantes. Además, la consideración de un sistema híbrido, que combina tanto filtrado colaborativo, como basado en contenido, también optimiza los resultados alcanzados.

En un futuro se pretende ampliar las pruebas a otras titulaciones, de forma que se pueda comprobar si se obtienen los mismos resultados, y bajo qué circunstancias se pueden generalizar las conclusiones.

VII. Agradecimiento

Este trabajo está financiado por el proyecto de investigación TIN2017-83445-P del Ministerio de Economía y Competitividad y el Fondo Europeo de Desarrollo Regional.

Referencias

[1] C. Romero and S. Ventura. Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(1):12–27, 2013.

[2] T.N. Huynh-Lv, N. Huu-Hoa, and T.N. Nguyen. Methods for building course recommendation systems. In *Proceedings of the 8th International Conference on Knowledge and Systems Engineering*, pages 163–168, 2016.

[3] P.-C. Chang, C.-H. Lin, and M.-H. Chen. A hybrid course recommendation system by integrating collaborative filtering and artificial immune systems. *Algorithms*, 9(3):1–18, 2016.

[4] C. Kim, N. Choi, Y. Heo, and J. Sin. On the development of a course recommender system: A hybrid filtering approach. *Entrue Journal of Information Technology*, 14(2):71–82, 2015.

[5] C. Vialardi, J. Chue, J.P. Peche, G. Alvarado, B. Vinatea, J. Estrella, and A. Ortigosa. A data mining approach to guide students through the enrollment process based on academic performance. *User Modeling and User-Adapted Interaction*, 21(1-2):217–248, 2011.

[6] J.W. Han, J.C Jo, H.S. Ji, and H.S. Lim. A collaborative recommender system for learning courses considering the relevance of a learner's learning skills. *Networks Software Tools and Applications*, 19(4):2273–2284, 2016.

[7] S. Spiegel. *A Hybrid Approach to Recommender Systems based on Matrix Factorization*. Thesis, Tech. University Berlin, 2009.

[8] A. Parameswaran, P. Venetis, and H. Garcia-Molina. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Transactions on Information Systems*, 29(4):1–33, 2011.

[9] C.Y. Huang, R.C. Chen, and L.S. Chen. Course-recommender system based on ontology. In *Proceedings of 12th International Conference on Machine Learning and Cybernetics*, pages 1168–1173, 2013.

[10] L. Zhuhadar, O. Nasraoui, R. Wyatt, and E. Romero. Multi-model ontology-based hybrid recommender system in e-learning domain. In *Proceedings of the International Joint Conferences on Web Intelligence and Intelligent Agent Technology*, pages 91–95, 2009.

[11] A.A. Kardan, H. Sadeghi, S.S. Ghidary, and M.R.F. Sani. Prediction of student course selection in online higher education institutes using neural network. 65:1–11, 2013.

[12] J. Sobecki and J.M. Tomczak. Student courses recommendation using ant colony optimization. In *Proceedings of the 2sd Intelligent Information and Database Systems*, volume 5991 of *Lecture Notes in Artificial Intelligence*, pages 124–133, 2010.

[13] E. Khorasani, Z. Zhenge, and J. Champaign. A markov chain collaborative filtering model for course enrollment recommendations. In *Proceedings of the 4th International Conference on Big Data*, pages 3484–3490. IEEE, 2016.

[14] R. Wang. *Sequence-based Approaches to Course Recommender Systems*. Thesis, University of Alberta, 2017.

[15] H. Ma, X. Wang, J. Hou, and Y. Lu. Course recommendation based on semantic similarity analysis. In *Proceedings of 3rd Ieee International Conference on Control Science and Systems Engineering*, pages 638–641, 2017.

[16] E. Özcan and C. Başaran. A case study of memetic algorithms for constraint optimization. *Soft Computing*, 13(8-9):871–882, 2009.

Reconocimiento de genes en secuencias de ADN por medio de imágenes.

Luis A. Santamaría C. *Fac. Cs Computación. BUAP* Puebla, México ALULOC_KAPA@hotmail.com

Sarahí Zuñiga H. *Fac. Cs Computación. BUAP* Puebla, México sarahi.zuhe@gmail.com

Ivo H. Pineda T. *Fac. Cs Computación. BUAP* Puebla, México

María J. Somodevilla *Fac. Cs Computación. BUAP* Puebla, México mariasg@cs.buap.mx

Mario Rossainz L. *Fac. Cs Computación. BUAP* Puebla, México

***Resumen*—En los últimos años, el campo del aprendizaje automático ha progresado enormemente al abordar problemas difíciles de clasificación. El problema planteado en este artículo es reconocer secuencias de ADN, reconocer los límites entre exones e intrones utilizando una representación gráfica de secuencias de ADN y métodos recientes de aprendizaje profundo. El objetivo de este trabajo es clasificar secuencias de ADN utilizando una red neuronal convolucional (Convolutional Neural Network CNN). El conjunto de secuencias de ADN utilizado es la base de datos Molecular (Splice-junction Gene Sequences) Data Set que cuenta con 3190 secuencias, disponible en la página de la UCI, con tres clases de secuencias: límite exón-intrón, límite intrón-exón y ninguna. Se utilizó el conjunto de secuencias de ADN utilizado para el reconocimiento fueron 1847 secuencias de una base de datos con 4 tipos de virus de hepatitis C (tipo 1, 2, 3 y 6) tomada del repositorio disponible en la página de ViPR. Para la utilización de las secuencias de ADN se diseñó un método de representación donde cada base nitrogenada es representada en escala de grises para formar una imagen. Las imágenes generadas se utilizaron para entrenar la red neuronal convolucional. Los resultados muestran que una CNN puede hacer la clasificación de secuencias de ADN con un porcentaje de precisión de entrenamiento del 82 %, una precisión de validación del 75 % y una precisión de evaluación del 80.8 %. Se llega a la conclusión de que es posible clasificar las imágenes de secuencias de ADN de la base de datos empleada.**

***Palabras clave*—Reconocimiento de genes, Aprendizaje profundo, Redes neuronales convolucionales, Codificación de secuencias de ADN .**

I. Introducción

Los métodos de aprendizaje automático permiten identificar características que propician la clasificación, análisis y reconocimiento de patrones. En el área de la biología, el uso de métodos de aprendizaje automático, facilitan el reconocimiento de secuencias de ADN. Este trabajo reconoce los genes de ADN previamente procesados para ser representados por una imagen. Este artículo es divido en secciones, la primera sección es el estado del arte que es el conocimiento previo necesario para el reconocimiento de genes. La segunda sección detalla la metodología que fue utilizada para el análisis de las secuencias de ADN. La tercera y cuarta sección muestran los resultados y las conclusiones obtenidas.

II. Estado del Arte

Los mecanismos o procesos de predicción de genes son aquellos que, dentro del área de la biología computacional, se utilizan para la identificación algorítmica de trozos de secuencias, usualmente ADN genómico [8], y que son biológicamente funcionales. Esto, especialmente incluye los genes codificantes de proteínas y secuencias reguladoras. La identificación degenes es uno de los primeros y más importantes pasos para entender el genoma de una especie una vez ha sido secuenciado [11].

El ácido desoxirribonucleico (ADN) está compuesto por cuatro moléculas llamadas nucleótidos o bases nitrogenadas: adenina, timina, guanina y citosina [9]. Una molécula completa de ADN o, dicho de otro modo, una secuencia de ADN está compuesta por un alfabeto que contiene las letras de las cuatro bases nitrogenadas.

$$\begin{aligned} & \Sigma\{ATGC\} \\ \phi_i = (V_1, V_2, V_3, \ldots, V_n) & \\ & V_i \in \Sigma \end{aligned} \tag{1}$$

Donde una cadena ϕ es una secuencia de ADN formada elementos del alfabeto Σ y puede definir las características de un organismo vivo, conteniendo toda la información genética en unidades de herencia llamadas genes. Los mecanismos o procesos de predicción de genes son aquellos que, dentro del área de la biología computacional, se utilizan para la identificación algorítmica de trozos de secuencias, usualmente ADN genómico [1], y que son biológicamente funcionales. Esto, especialmente incluye los genes codificantes de proteínas y secuencias reguladoras. La identificación de genes es uno de los primeros y más importantes pasos para entender el genoma de una especie una vez ha sido secuenciado [2].

Las uniones de empalme son puntos en una secuencia de ADN en la que se elimina ADN ‘’inútil” durante el proceso de creación de proteínas en organismos superiores. El problema planteado en este conjunto de datos es reconocer, dada una secuencia de ADN, los límites entre los exones (las partes de la secuencia de ADN retenidas después del corte y empalme) y los intrones (las partes de la secuencia de ADN que se cortan). Este problema consiste en dos subtareas: reconocimiento de límites de exón / intrón (denominados sitios EI) y reconocimiento de límites de intrón / exón (sitios IE). (En la comunidad biológica, los límites de IE se refieren a los ‘’aceptantes” mientras que los límites de EI se conocen

como ‘’donantes”) [7]. Ambas tareas son complicadas ya que no existe una secuencia estándar para reconocer intrones y exones, razón por la cual es interesante diseñar herramientas que nos ayuden a identificarlos y clasificarlos.

El número de proyectos de investigación sobre genomas actualmente vigentes aumenta a un ritmo acelerado, y proporcionar un catálogo de genes para estos nuevos genomas es un desafío clave. La obtención de un conjunto de genes bien caracterizados, es un requisito básico en los pasos iniciales de cualquier proceso de creación de un genoma. Los métodos de búsqueda de genes computacionales se pueden categorizar libremente como basados en la alineación y en la composición de secuencias o una combinación de ambos. Los métodos basados en la alineación de secuencias se pueden usar cuando se intenta predecir un gen que codifica una proteína para la cual existe un homólogo estrechamente relacionado, este es el enfoque en GeneWise [5] y PROCRUSTES [4].

Los algoritmos basados en composición de secuencias (también conocidos como métodos de búsqueda de genes) contienen un modelo probabilístico de estructura génica basado en señales biológicas (sitios de empalme y sitios de inicio / detención de traducción) y propiedades de composición de secuencias funcionales (exones como secuencias codificantes e intrones como secuencias intermedias entre exones e intrones). A diferencia de los métodos basados en la alineación, estos algoritmos se basan sólo en las propiedades intrínsecas de los genes para construir estructuras genéticas predichas. Genscan [10] y Geneid [3] son los dos ejemplos de este enfoque y pueden encontrar genes conocidos y genes nuevos siempre que los genes se ajusten al modelo probabilístico subyacente. Una cadena de ADN es una molécula caracterizada por cuatro bases nitrogenadas Adenina, Timina, Guanina y Citosina [12]. Para mejorar la representación de una cadena de ADN se utilizan secuencias que pueden ser transformadas a representación con valores numéricos o alfabéticos: A (adenina), T (timina), G (guanina) y C (citosina). [10]. Sin embargo, la representación de grandes cantidades de información como secuencias de ADN no hacen sencillo su análisis matemático, esto crea la necesidad de encontrar nuevas formas de representar la información.

En 1988 Lapedes [6] y su equipo de trabajo entrenaron una red neuronal para reconocer genes en secuencias de ADN, lograron una precisión del 91.2 % en las uniones de corte y empalme de intrón / exón y del 92.8 % en las uniones de empalme de exón / intrón. Lo que dio origen a plantear el uso de redes neuronales convolucionales para resolver este mismo problema de clasificación. Este trabajo consistió en buscar una nueva forma de representar secuencias de ADN para su análisis, como ya se ha hecho referencia, existen actualmente diferentes métodos para reconocer genes, pero estas representaciones complican su análisis. La propuesta que presentamos es generar imágenes a partir secuencias de ADN y someterlas a análisis por técnicas de aprendizaje profundo, en específico a redes neuronales convolucionales; utilizadas para la clasificación de imágenes. Actualmente no se ha encontrado un modelo matemático que resuelva el proceso de clasificación por redes neuronales, pero sus resultados llegan a ser tan altos que superan el 99 % en algunos casos [1].

II-A. Red Neuronal Convolucional (CNN)

En los últimos años, el campo del aprendizaje automático ha progresado enormemente al abordar problemas de clasificación, identificación y reconocimiento de patrones. En particular, se ha encontrado que un tipo de modelo llamado red neuronal convolucional CNN (Convolutional Neural Network) por sus siglas en inglés, que logra un rendimiento razonable en tareas de reconocimiento visual de hardware, igualando o superando el rendimiento humano en algunos dominios [11]. Una CNN es un algoritmo para el aprendizaje automático en el que un modelo aprende a realizar tareas de clasificación directamente a partir de imágenes, videos o sonidos. Las CNNs son especialmente útiles para localizar patrones en imágenes con el objetivo de reconocer objetos, caras y escenas. Aprenden directamente a partir de los datos de imágenes, utilizando patrones para clasificar las imágenes y eliminar la necesidad de una extracción manual de características.

Inception-v3 está diseñado para el desafío de Reconocimiento Visual, ésta es una tarea estándar en visión artificial, donde los modelos intentan clasificar imágenes completas en 1000 clases de ImageNet. TensorFlow es una herramienta para el aprendizaje automático. Si bien contiene una amplia gama de funcionalidades, TensorFlow está diseñado principalmente para modelos de redes neuronales profundas. Los modelos modernos de reconocimiento de imágenes tienen millones de parámetros; entrenarlos desde cero requiere una gran cantidad de datos de entrenamiento etiquetados y una gran cantidad de potencia de cálculo (cientos de horas de GPU o más). El aprendizaje de transferencia es una técnica que ataja mucho de esto tomando una pieza de un modelo que ya ha sido entrenado en una tarea relacionada y reutilizándola en un nuevo modelo, en la figura 1 se muestra un ejemplo de una CNN, los filtros se aplican a cada imagen de entrenamiento con diferentes resoluciones, y la salida de cada imagen convolucionada se usa como entrada para la capa siguiente [2]. Aunque no es igual de preciso en comparación a la capacitación del modelo completo, es sorprendentemente eficaz para muchas aplicaciones, funciona con cantidades moderadas de datos de capacitación (miles, no millones de imágenes etiquetadas) y se puede ejecutar en tan solo treinta minutos en una computadora portátil sin una GPU [11].

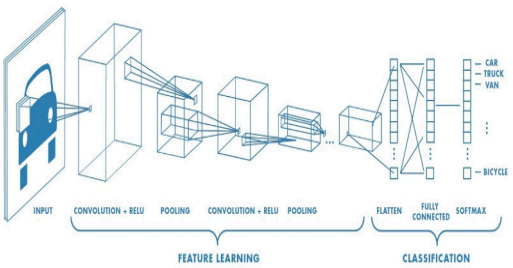


Figura 1. Ejemplo de una Red Neuronal Convolucional

III. Metodología

En esta sección se describe detalladamente como se generaron imágenes a partir de secuencias de ADN y su posterior uso en el entrenamiento de una red neuronal convolucional para clasificación de tres clases de secuencias. En esta etapa del proyecto consistió en convertir las secuencias de ADN a representaciones gráficas para entrenar una CNN. Un aspecto importante que se ha considerado este trabajo es que las CNN son utilizadas para el reconocimiento de patrones y clasificación de imágenes. Las secuencias de ADN de manera general son representadas por letras: A usada para la adenina, G para la guanina, C para la citosina y T para la timina, sin embargo, una CNN no está establecida para procesar información bajo este formato, por esta razón se diseñó una representación gráfica de las secuencias. El primer paso fue asignar un color en escala de grises a cada una de las letras como se muestra en el Cuadro I. Las escalas de grises va de 0 que representa negro, a 1 que represa el blanco, de tal manera que los colores intermedios resultantes son tonalidades de gris para mostrar un mejor contraste. Lo segundo fue hacer

Cuadro I
Representación por color de las bases nitrogenadas.

Base Nitrogenada	Valor de gris
A	0
C	0.3
G	0.7
T	1

que las secuencias pudieran ser representadas por una imagen específica a cada una. Para lograr esto se utilizó una matriz de dimensión 60 X 60, donde el valor 60 coincide con el número de bases nitrogenadas de todas las secuencias de la base de datos. Cada secuencia fue colocada en la primera fila y copiada en el resto de las filas hasta tener 60 en total, así el resultado final es una imagen con barras en la escala de grises como la que se muestra en la Figura 3, cada una de las imágenes obtenidas es específica para cada instancia de la base de datos como se observa en la Figura 2. En total se obtuvieron 3190 imágenes.

```
1  CCAGCTGCATCACAGGAGGCCAGCGAGCAGGTCTGTTCCAAGGGCCTTCGAGCCAGTCTG
2  CCAGCTGCATCACAGGAGGCCAGCGAGCAGGTCTGTTCCAAGGGCCTTCGAGCCAGTCTG
3  CCAGCTGCATCACAGGAGGCCAGCGAGCAGGTCTGTTCCAAGGGCCTTCGAGCCAGTCTG
.
.
.
60 CCAGCTGCATCACAGGAGGCCAGCGAGCAGGTCTGTTCCAAGGGCCTTCGAGCCAGTCTG
```

Figura 2. Secuencia ADN a codificar

III-A. Utilización de CNN a secuencias de ADN

En esta subsección se describe cómo se entrenó una CNN con las imágenes representativas de cada secuencia. Se utilizó

Figura 3. Imagen asociada

una CNN InceptionV3 a la que se le aplicó la transferencia de aprendizaje profundo para categorizar el reconocimiento de tres clases se secuencias de ADN: reconocimiento de límites de exón / intrón (denominados sitios EI), reconocimiento de límites de intrón / exón (sitios IE) y reconocimiento de ninguno de los dos anteriores (N).
Una vez que se logró representar a las secuencias de ADN como imágenes se utilizó una CNN y con la librería de software TensorFlow se construyó un modelo de clasificación basado en una red neuronal convolucional pre-entrenada. Se utilizaron CNNs InceptionV3 a las que se les aplicó la transferencia de aprendizaje profundo para categorizar el reconocimiento de una base de datos con cuatro clases de secuencias de ADN: virus de Hepatitis C tipo 1, 2, 3 y 6 y el reconocimiento de otra base de datos con tres clases de límites de exón / intrón (denominados sitios EI), reconocimiento de límites de intrón / exón (sitios IE) y reconocimiento de ninguno de los dos anteriores (N). Para ajustar el modelo a nuestro problema se entrenaron las últimas capas de las redes con instancias obtenidas de las bases de datos, ambas redes fueron entrenadas en 4000 pasos.

Primero se entrenó la CNN para hacer la clasificación de los 4 tipos de virus de Hepatitis, posteriormente se entrenó una CNN con solamente 2 clases: EI e IE y por último se entrenó otra CNN con todas las clases de la base de datos: EI, IE y N para comparar los resultados de las ultimas dos neuronas.

IV. Resultados

Los resultados de clasificación para la CNN entrenada con la base de datos de los cuatro tipos de virus de Hepatitis C muestran una precisión de evaluación 95 % con 145 imágenes probadas y al terminar el paso (k) 4000 la precisión de entrenamiento fue del 94.5 % y la precisión de validación del 95 % como se observa en la figura 4. El comportamiento decreciente de la entropia durante el entrenamiento, se aprecia en la figura 5 .
Al usar una CNN con las clases EI e IE se obtiene una precisión de evaluación del 80.8 % con 177 imágenes de prueba y al terminar el paso (k) 4000 la precisión de entrenamiento es del 82 % y la precisión de validación del 75 %. En la Figura 6 se muestra como la exactitud de entrenamiento (naranja) y validación (azul) va cambiando en cada paso y en la Figura 7 se muestra como la entropía disminuye con el incremento de los pasos durante el entrenamiento. Por otro lado, los resultados de la segunda CNN donde se utilizaron las tres clases de la base de datos muestran una precisión de evaluación

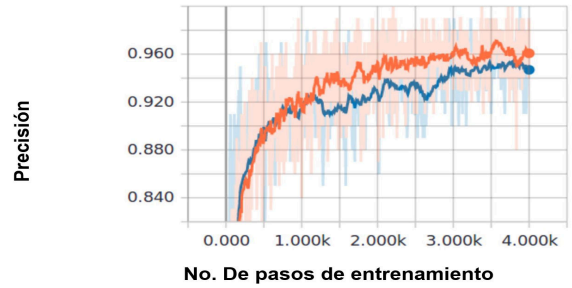


Figura 4. CNN con las clases de virus de Hepatitis C tipo 1, 2, 3 y 6. Naranja: precisión de entrenamiento. Azul: precisión de validación después de 4000 pasos (k).

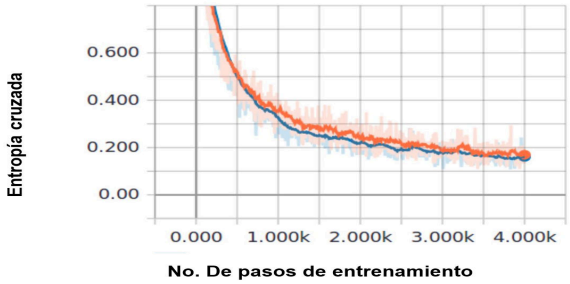


Figura 5. Entropía cruzada de la CNN con las clases de virus de Hepatitis C tipo 1, 2, 3 y 6 después de 4000 pasos (k). Trazo superior: entrenamiento. Trazo inferio: validación.

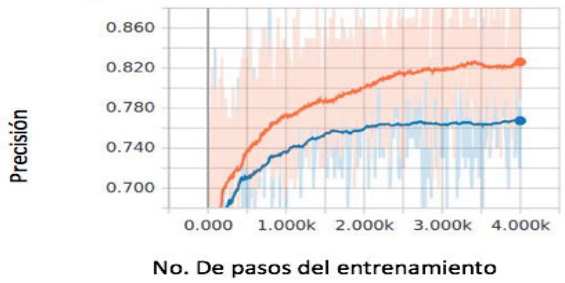


Figura 6. CNN con las clases IE y EI. Naranja: precisión de entrenamiento. Azul: precisión de validación después de 4000 pasos (k).

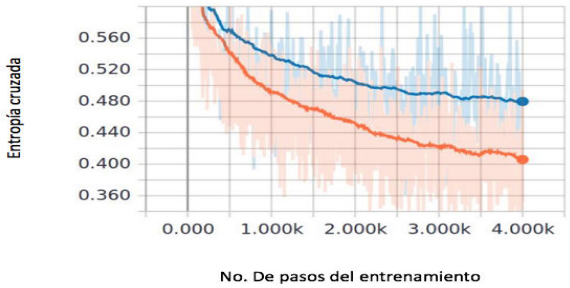


Figura 7. Entropía cruzada de la CNN con las clases IE y EI después de 4000 pasos (k). Naranja: entrenamiento. Azul: validación.

de 57.5 % con 301 imágenes y al terminar el paso 4000 la precisión de entrenamiento 69 % y la precisión de validación con un 56 % como se aprecia en la Figura 8. En la Figura 9 se muestra los cambios de la entropía en cada etapa del entrenamiento.

V. Conclusiones

Los resultados obtenidos de la CNN entrenada con la base de datos de virus de Hepatitis C sugieren que la metodología de aprendizaje automático empleada en este trabajo es adecuada para la clasificación de las imágenes generadas a partir de las secuencias de ADN, mostrando importantes y altos porcentajes de precisión de evaluación, precisión de entrenamiento y la precisión de validación. Estos resultados nos llevaron a realizar los siguientes experimentos para el reconocimiento de exones e introns en la siguiente base de datos. Para este caso las CNN muestran que los porcentajes de precisión de validación son menores en comparación a los de una red neuronal tomando como referencia el trabajo de Lapedes [6]. La importancia del trabajo es que se presentan resultados favorables para seguir explorando el uso de las redes neuronales convolucionales utilizando la representación de las secuencias de ADN como imágenes, un método de codificación sencillo y práctico.

En este trabajo se ha logrado realizar clasificación de secuencias de ADN usando una CNN y los resultados demuestran que las CNN son capaces de realizar esta clasificación hasta con un 80.8 % de precisión de evaluación para el experimento con las clases IE e EI y el 57.5 % para el experimento con las clases IE, EI y N. Resultados similares se pueden observar en la precisión de entrenamiento y validación de las Figuras 6 y 8. En el caso de los cuatro tipos de hepatitis se logran resultados de hasta 94.5 % de precisión de evaluación .

La diferencia entre los resultados obtenidos para los experimentos con dos y tres clases se puede justificar que al incrementar el número de clases se incrementa la entropía Figuras 7 y 9. La entropía cruzada es una métrica que puede utilizarse para reflejar la precisión de los pronósticos probabilísticos y está estrechamente vinculada con la estimación por máxima verosimilitud. La entropía cruzada es una función que permite evaluar el resultado de la clasificación en vez de utilizar la métrica del error cuadrático medio, el valor de la entropia cruzada permite evaluar el progreso del proceso de aprendizaje de la información [1].

Por otro lado, se habla de que la transferencia de aprendizaje es buena cuando se disponen de pocas imágenes para entrenar la red y que permite llegar a resultados aceptables en la mayoría de los casos, sin embargo, todavía es posible mejorar aún más la precisión de validación y entrenamiento y disminuir la entropía si se entrena una red neuronal desde cero, es decir se debe contar con una base de datos de millones de instancias y un equipo de cómputo con GPU para entrenar esta red pero seguramente ofrecerá mejores resultados que la CNN preentrenada que utilizamos para este trabajo.

En conclusión, se puede afirmar que una red neuronal convolucional del modelo InceptionV3 es capaz de clasificar secuencias de ADN si la secuencia es procesada y transformada a una imagen, sin embargo, los porcentajes de exactitud se pueden mejorar si se entrena una CNN con una base de

Figura 8. CNN con las clases IE, EI y N. Naranja: precisión de entrenamiento. Azul: precisión de validación después de 4000 pasos (k).

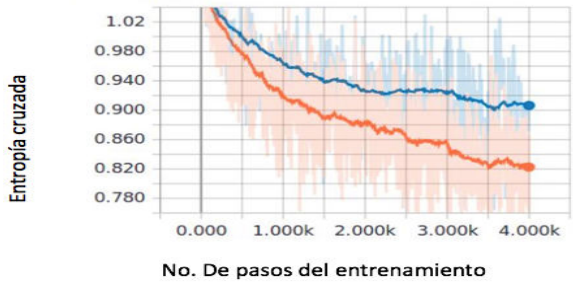


Figura 9. Entropía cruzada de la CNN con las clases IE, EI y N después de 4000 pasos (k). Naranja: entrenamiento. Azul: validación.

secuencias más grande.

Reconocimientos

Los autores agradecen al Consejo Nacional de Ciencia y Tecnología (CONACyT) de México , a la Benemérita Universidad Autónoma de Puebla la cual a través de la Facultad de Ciencias de la Computación han brindado el apoyo necesario para la realización y presentación del presente trabajo.

Referencias

[1] How to retrain an image classifier for new categories. https://www.tensorflow.org/tutorials/image_retraining. Accessed: 2018-05-28.

[2] Mathworks (2018). deep learning. https://la.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. Accessed: 2018-05-27.

[3] Jonas S Almeida and Susana Vinga. Universal sequence map (usm) of arbitrary discrete sequences. *BMC bioinformatics*, 3(1):6, 2002.

[4] Chris Burge and Samuel Karlin. Prediction of complete gene structures in human genomic dna1. *Journal of molecular biology*, 268(1):78–94, 1997.

[5] Mikhail S Gelfand, Andrey A Mironov, and Pavel A Pevzner. Gene recognition via spliced sequence alignment. *Proceedings of the National Academy of Sciences*, 93(17):9061–9066, 1996.

[6] A Lapedes, Christopher Barnes, Christian Burks, R Farber, and K Sirotkin. Application of neural networks and other machine learning algorithms to dna sequence analysis. Technical report, Los Alamos National Lab., NM (USA), 1988.

[7] Michiel O Noordewier, Geoffrey G Towell, and Jude W Shavlik. Training knowledge-based neural networks to recognize genes in dna sequences. In *Advances in neural information processing systems*, pages 530–536, 1991.

[8] Christos A Ouzounis. Rise and demise of bioinformatics? promise and progress. *PLoS computational biology*, 8(4):e1002487, 2012.

[9] Arturo Panduro. *BiologÃa molecular en la clÃnica*. McGraw-Hill Interamericana, 2009.

[10] Genís Parra, Enrique Blanco, and Roderic Guigó. Geneid in drosophila. *Genome research*, 10(4):511–515, 2000.

[11] SL Salzberg, DB Searls, and S Kasif. Computational gene prediction using neural networks and similarity search. *Computational Methods in Molecular Biology*, 32:109, 1998.

[12] Zhu-Jin Zhang. Dv-curve: a novel intuitive tool for visualizing and analyzing dna sequences. *Bioinformatics*, 25(9):1112–1117, 2009.

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 2:
Modelos Descriptivos

Reglas de Asociación en Datos Multi-Instancia mediante Programación Genética Gramatical

José María Luna
Depto. Informática y Análisis Numérico,
Universidad de Córdoba
Email: jmluna@uco.es

Oscar Reyes
Depto. Informática y Análisis Numérico,
Universidad de Córdoba
Email: ogreyesp@gmail.com

María José del Jesus
Depto. Informática,
Universidad de Jaén
Email: mjjesus@ujaen.es

Sebastián Ventura
Depto. Informática y Análisis Numérico,
Universidad de Córdoba
Email: sventura@uco.es

***Abstract*—El estado del arte actual en minería de reglas de asociación es bastante prometedor en cuanto a la extracción automática de relaciones de interés entre patrones o elementos de grandes bases de datos. Por lo general, los datos se encuentran representados en forma tabular, donde cada fila define inequívocamente un registro de datos u objeto en el dominio de la aplicación. En ocasiones, estos registros de datos no pueden o deben ser considerados de forma aislada (por ejemplo, datos que comprenden hábitos de compra de clientes de manera que diferentes registros pueden describir un mismo cliente) y, por tanto, el problema no puede ser abordado como un problema de minería de reglas de asociación tradicional. Un primer intento de resolver este problema consideró un enfoque determinista sobre dominios discretos y requiriendo analizar todo el espacio de búsqueda (el tiempo de computo es exponencial respecto al número de elementos). Sin embargo, cuando nos enfrentamos a un problema real, la información se encuentra generalmente definida en dominios continuos, se requieren resultados en el menor tiempo posible, y se necesita satisfacer las expectativas de los usuarios (los resultados deben coincidir con la forma deseada). En este sentido, el objetivo de este trabajo es solucionar estos desafíos mediante un algoritmo de programación genética gramatical. Tanto sus fortalezas como defectos se analizan sobre un conjunto variado de datos en el estudio experimental.**

I. Introducción

Desde que se produjo la revolución digital, la cantidad de información que se almacena en cada dominio de aplicación ha crecido a un ritmo exponencial. En este sentido, se requiere que las diferentes técnicas de minería de datos mejoren su eficiencia para poder tratar con tales conjuntos masivos de datos en un tiempo considerable. Una de las tareas de minería de datos que se ha visto claramente afectada por la dimensionalidad de los datos es la minería de reglas de asociación (ARM por su término en inglés) [1]. Considerando un conjunto de datos con k elementos, el espacio de búsqueda en el que ARM debe buscar soluciones incluye un total de $3^k - 2^{k+1} + 1$ reglas de asociación diferentes obtenidas sobre un total de $2^k - 1$ patrones o combinaciones de elementos.

Incluso cuando se trata de un problema realmente complejo, especialmente para grandes conjuntos de datos, diferentes investigadores han propuesto algoritmos de ARM realmente eficientes. Dichas propuestas fueron especialmente relevantes gracias a las arquitecturas paralelas actuales y *frameworks* [**?**]. En este sentido, el problema de ARM se puede considerar como resuelto desde el punto de vista del rendimiento [**?**]. Sin embargo, el creciente interés en el almacenamiento de datos ha causado no solo un crecimiento en la dimensionalidad de los datos, sino también en la representaciones de dichos datos. A modo de ejemplo, consideremos una cadena de supermercados que quiere extraer conocimiento útil sobre los hábitos de compra de sus clientes para tomar decisiones correctas. Supongamos que la información se almacena en una tabla tradicional (cada fila indica una transacción diferente). Aquí, cada cliente tendrá un peso específico en la descripción de datos con respecto al número de transacciones que representa y, por lo tanto, esta representación de datos desvía el conocimiento extraído según el tipo de cliente (los clientes frecuentes son más importantes que los clientes esporádicos). Consideremos ahora que cada cliente se almacena como un ejemplo que comprende un número indefinido de transacciones, y los datos se describen sobre conjunto de ejemplos en lugar de sobre el conjunto de transacciones. De esta forma, el conocimiento extraído revela información explícita sobre los hábitos de los clientes independientemente de la cantidad de transacciones que realiza cada cliente. Esto implica que la forma en que se estructuran los datos es esencial para proporcionar el conocimiento correcto.

Este almacenamiento de información está por tanto relacionado con *multiple-instance learning* (MIL) [4], una generalización del aprendizaje supervisado tradicional. En MIL, los ejemplos son ambiguos y un solo ejemplo puede tener muchas instancias alternativas que lo describen. Esta ambigüedad se analizó recientemente [5] desde una perspectiva de descripción de datos mediante ARM sobre estructuras de datos de múltiples instancias. Sin embargo, este primer intento se basó en una metodología de búsqueda exhaustiva para establecer las bases en esta nueva representación para análisis descriptivo de datos y no se prestó atención al tiempo de ejecución, la metodología, el dominio de la aplicación y las

preferencias de los usuarios.

El objetivo de este trabajo es proponer un nuevo algoritmo de ARM para trabajar sobre datos de multi-instancia. El enfoque propuesto, conocido como MIAR-GePro, se basa en una metodología de programación genética gramatical y tiene como objetivo mejorar el estado del arte en este campo. MIAR-GePro puede trabajar tanto en dominios discretos como continuos, representando una importante ventaja respecto a la propuesta existente (requiere una discretización previa al proceso de extracción). Además, el modelo evolutivo propuesto es capaz de extraer reglas de asociación en un solo paso (la propuesta existente requiere extraer primero todos los patrones frecuentes y, a posteriori, obtener reglas de asociación sobre dichos patrones). Finalmente, los resultados obtenidos mediante MIAR-GePro están restringidos de acuerdo a una gramática predefinida por el usuario, por lo que es éste usuario final quien tiene la capacidad de definir la forma de las reglas que desea obtener. Todas estas características, así como algunos inconvenientes de esta propuesta se analizan en la etapa experimental al considerar un conjunto variado de datos.

El presente documento está organizado de la siguiente manera. La sección II presenta las definiciones más relevantes y el trabajo relacionado. La sección III describe el algoritmo MIAR-GePro propuesto. La sección IV presenta los conjuntos de datos utilizados en los experimentos, la configuración seguida, así como los resultados obtenidos. Finalmente, algunas observaciones finales se describen en la sección V.

II. Preliminares

En esta sección se describe por primera vez el problema de datos multi-instancia. A continuación, se define la tarea de minería de regla de asociación (ARM por su término en inglés) y se analiza el primer intento de extraer reglas de asociación en datos multi-instancia.

A. Datos Multi-Instancia

El problema de aprendizaje sobre datos multi-instancia fue introducido por primera vez por *Dietterich et al.* [4] en 1997 en el contexto de la predicción del comportamiento de diferentes fármacos. En este dominio de aplicación, los ejemplos son ambiguos y pueden incluir múltiples instancias alternativas que lo describen. De manera formal, supongamos que cada ejemplo e_i puede tener v_i variantes $e_{i,1}$, $e_{i,2}$, ..., e_{i,v_i}. Cada una de estas variantes estará representada por un vector de características distintas $V(e_{i,j})$, por lo que un ejemplo completo e_i se representa por lo tanto como un conjunto de vectores de características en la forma $\{V(e_{i,1}), V(e_{i,2}), ..., V(e_{i,v_i})\}$. Incluso cuando este problema se ha aplicado generalmente a las tareas de aprendizaje supervisado [6], realizar un análisis descriptivo de ese tipo de datos es de gran interés en muchos dominios. Considerando el conocido problema de análisis de la cesta de la compra, un único cliente puede describir diferentes compras, por ejemplo, en días diferentes de un mes. Como resultado, cada ejemplo (un cliente específico) incluye un número indefinido de registros (diferentes compras). En este escenario, el problema se puede definir como el análisis de un conjunto de datos en el que los datos se representan como multi-instancias, y se pueden aplicar diferentes hipótesis de trabajo.

La hipótesis de trabajo más simple fue propuesta por *Dietterich* [4], y considera que un determinado ejemplo se cumple si al menos una de las instancias incluidas en dicho ejemplo es satisfecha. En la minería de patrones, el problema se modela como encontrar un patrón P en el que $\exists j \ : \ P \subset V(e_{i,j})$ para una gran cantidad de ejemplos e_i. Teniendo en cuenta esta hipótesis en el análisis de la cesta de mercado, su objetivo es encontrar artículos (o conjunto de artículos) comprados al menos una vez por un gran número de clientes. Aquí, el objetivo es descubrir cuáles son los productos más populares o qué productos generalmente compran los clientes a la vez.

Múltiples autores han propuesto diferentes hipótesis de trabajo para tratar el problema de multi-instancia. *Weidmann et al.* [7] definió dos tipos adicionales de hipótesis que son casos especiales del propuesto por *Dietterich*. En la primera de estas hipótesis adicionales, el problema se modela como el de encontrar un patrón P en el que $\sum_{j=1}^{v_i} P \subset V(e_{i,j}) \geq m$ para un número alto de ejemplos e_i. Centrándose en el análisis de la cesta de la compra, esta hipótesis es realmente útil para determinar qué artículos (o conjunto de artículos) generalmente se compran más de m veces por un gran número de clientes. Aquí, no solo es importante conocer productos populares, sino que también se requiere que tales productos sean generalmente comprados por cada uno de los clientes. Finalmente, el segundo tipo de hipótesis definido por *Weidmann et al.* [7] establece que un patrón P se satisface con un número determinado de instancias en un ejemplo, y este número está entre un valor mínimo m y un valor máximo o. En otras palabras, $o \geq \sum_{j=1}^{v_i} P \subset V(e_{i,j}) \geq m$ en una gran cantidad de ejemplos e_i. Volviendo al análisis de la cesta de la compra, esta hipótesis es esencial para conocer el conjunto de clientes que consumen un producto específico un número predefinido de veces (dando un valor umbral mínimo y máximo).

B. Minería de Reglas de Asociación

La tarea de ARM fue definida formalmente por *Agrawal et al.* [8] en los años 90. Esta tarea tiene como objetivo extraer implicaciones de la forma $Antecedente \rightarrow Consecuente$ que sean de especial interés y desconocidas. Tanto $Antecedente$ (A) and $Consecuente$ (C) representan conjuntos disjuntos de elementos. De manera formal, se define $I = \{i_1, i_2, ..., i_n\}$ como el conjunto de n elementos incluidos en un conjunto de datos Ω, y $T = \{t_1, t_2, ..., t_m\}$ como el conjunto de transacciones o registros que cumplen $\forall t \in T : t \subseteq I \in \Omega$. Una regla de asociación es una implicación de la forma $A \rightarrow C$ donde $A \subset I$, $C \subset I$, y $A \cap C = \emptyset$. La frecuencia (también conocida como soporte) de una regla $A \rightarrow C$ es definida como el porcentaje de transacciones en las que se cumple $A \cup C \subseteq t : t \in T$. Este hecho se ha definido también en términos absolutos como el número de transacciones en las que aparece $A \cup C$, es decir, $|\{\forall t \in T : A \cup C \subseteq t, t \subseteq I \in \Omega\}|$. El significado de una regla de asociación es que si el antecedente

G = $(\Sigma_N, \Sigma_T, P, S)$ con:

S = Regla
Σ_N = {Regla, Antecedente, Consecuente, Condiciones, Condición, Condición_Nominal, Condición_Numérica}
Σ_T = {'Y', 'Atributo', 'valor', '=', '$\geq$','$\leq$'}
P = {Regla = Antecedente, Consecuente ;
Antecedente = Condiciones;
Consecuente = Condiciones;
Condiciones = 'Y', Condiciones, Condición | Condición ;
Condición = Condition_Nominal | Condición_Numérica ;
Condición_Nominal = 'Atributo', '=', 'valor' ;
Condición_Numérica = 'Atributo', '$\geq$', 'valor' | 'Atributo', '$\leq$', 'valor';}

Fig. 1. Gramática de contexto libre que representa reglas de asociación expresadas en notación BNF

de la regla se cumple, entonces es bastante probable que el consecuente de dicha regla también se cumpla.

El uso de ARM sobre representaciones de datos no estándar, por ejemplo datos multi-instancia, han sido recientemente estudiados debido a su relevancia. En un conjunto de datos multi-instancia Ω, cada bolsa de instancias o ejemplos e_i $in\Omega$ comprende un número variado l de transacciones o instancias, es decir, $e_i = \{t_{i,1}, t_{i,2}, ..., t_{i,l}\}$. Cada instancia única $t_{i,j}$ se define como un subconjunto de elementos tales que $t_{i,j} \in e_i$: j $leql, t_{i,j} \subseteq I$. La principal entre ARM clásica y la basada en multi-instancias radica en las medidas de calidad utilizadas para determinar el interés de las soluciones. En la forma más simple, una regla de asociación $A \rightarrow C$ cumplirá un ejemplo e_i si y sólo si al menos una de sus instancias $t_{i,j} \in e_i$ es satisfecha por la regla, es decir, $A \cup C \subseteq t_{i,j} : t_{i,j} \in e_i$.

La métrica conocida como soporte de una regla $A \rightarrow C$ se define como el número de ejemplos que la cumplen, es decir, $|\{\forall e_i \in \Omega \ : \ A \cup C \subseteq t_{i,j} \in e_i\}|$. El soporte puede definirse también en su forma relativa al tamaño del conjunto de datos tal que $|\{\forall e_i \in \Omega \ : \ A \cup C \subseteq t_{i,j} \in e_i\}|/|\Omega|$. Una característica importante del soporte en datos multi-instancia es la posibilidad de que tanto A como C satisfagan todas las transacciones, pero si se analizan juntas no satisfacen ninguna transacción. Esta afirmación no es posible en ARM clásico ya que considera cada transacción como una sola instancia. Centrándonos en otra medida de la calidad de las reglas en problemas multi-instancia, la confianza se define como la proporción del número de ejemplos que incluyen A y C entre todos los ejemplos que comprenden A. Otra métrica ampliamente utilizada entre las muchas existentes en ARM [10] es conocida como *lift* y permite establecer cuántas veces ocurren A y C juntas más a menudo de lo que se esperaría si fueran estadísticamente independientes, es decir $Lift(A \cup C) = Soporte(A \cup C)/(Soporte(A) \times Soporte(C))$.

III. MIAR-GePro

El algoritmo propuesto, conocido como MIAR-GePro (por sus siglas en inglés de *Multiple-Instance Association Rule Genetic Programming*), incorpora una gramática libre de contexto que se adapta tanto al problema en cuestión como al conocimiento subjetivo del usuario. Esta gramática permite predefinir la forma de las soluciones y, por lo tanto, el espacio de búsqueda se reduce a aquellas soluciones que cumplen con la gramática.

a) ***Criterio de codificación****:* en programación genética gramatical [11] cada solución al problema bajo estudio está representada por un genotipo (árbol de derivación basado en el lenguaje definido por una gramática G) y un fenotipo (significado de la estructura del árbol). Una gramática libre de contexto se define como $(\Sigma_N, \Sigma_T, P, S)$ donde Σ_T representa el alfabeto de los símbolos terminales y Σ_N el alfabeto de los símbolos no terminales, siendo ambos conjuntos disjuntos, es decir, $\Sigma_N \cap \Sigma_T = \emptyset$. Además, P establece el conjunto de reglas de producción que se utilizan para codificar una solución, y estas reglas de producción deben comenzar desde el símbolo inicial S. Una regla de producción se define como una regla $\alpha \rightarrow \beta$, donde $\alpha \in \Sigma_N$, y $\beta \in \{\Sigma_T \cup \Sigma_N\}^*$. Como se describió anteriormente, se aplica una serie de reglas de producción del conjunto P para obtener una solución que se representa como un árbol donde los nodos internos contienen solo símbolos no terminales y las hojas contienen solo terminales.

La Figura 1 muestra la gramática de contexto libre propuesta en el algoritmo MIAR-GePro. En esta gramática, el símbolo inicial S se representa con el término Rule (raíz del árbol de derivación), el cual siempre tiene como nodos hijos el antecedente y el consecuente de la regla. Estos dos nodos secundarios (antecedente y consecuente) pueden ser expandidos en una sola condición o un conjunto de condiciones que incluyen características tanto discretas como continuas. Como resultado y considerando la gramática propuesta (ver Figura 1), se obtiene el siguiente lenguaje de la gramática $L(G) = \{(Y \ Condicion)^n \ Condicion \ \rightarrow (Y \ Condicion)^m \ Condicion, \ n \geq 0, m \geq 0\}$ que cualquier solución válida debe satisfacer.

b) ***Proceso de evaluación****:* El proceso clave de cualquier algoritmo evolutivo es la evaluación de las soluciones, el cual guía el proceso de búsqueda a través de soluciones prometedoras dentro del espacio de búsqueda. Aunque se pueden utilizar muchas hipótesis de trabajo para evaluar las soluciones de MIAR-GePro (varias hipótesis se describieron en la sección II-A), en este trabajo hemos considerado la hipótesis tradicional de *Dietterich* donde un ejemplo específico se satisface si y sólo si se cumple al menos una instancia dentro de dicho ejemplo. Así pues, el número máximo de registros

que puede satisfacer una regla (solución al problema) vendrá dado por el número de ejemplos y no por el número total de instancias (considerando todos los ejemplos).

c) ***Operadores genéticos****:* MIAR-GePro incluye dos operadores genéticos para generar nuevas soluciones en cada generación del proceso evolutivo. Estos dos operadores se basan en el cruce y la mutación selectiva propuestos por *Whigham* [12]. El operador de cruce permite crear nuevas soluciones intercambiando subárboles aleatorios a partir de dos padres de forma que se garantizan soluciones correctas de acuerdo al lenguaje de la gramática (ver Figure 1). Para evitar el crecimiento incontrolado, se utiliza como parámetro una profundidad máxima de árbol. Con respecto al operador de mutación, éste selecciona de forma aleatoria un nodo no terminal del árbol y genera un nuevo subárbol a partir de él.

d) ***Esquema evolutivo****:* MIAR-GePro se basa en un algoritmo evolutivo generacional clásico con elitismo (ver Algoritmo 1). La élite está formada por un número máximo predefinido de soluciones, que son las soluciones que deben devolverse al finalizar la ejecución del algoritmo (ver línea 16, Algoritmo 1). Inicialmente, se genera aleatoriamente una población de individuos (soluciones en forma de árboles) (ver línea 3, Algoritmo 1) al considerar la gramática propuesta de contexto libre G y los atributos dentro del conjunto de datos Ω. Una vez que los individuos son evaluados con respecto a algunas medidas de calidad (ver *Evaluation Procedure*, líneas 17 a 44, Algoritmo 1) se ejecuta el ciclo principal, que comprende las siguientes operaciones (ver líneas 7 a 15, Algoritmo 1): (a) se selecciona un conjunto de individuos de la población pop para actuar como padres por medio de un selector por torneo de tamaño 2; (b) el conjunto de padres se cruzan y mutan para formar nuevos individuos (incluidos en el conjunto $newPop$); (c) la población general pop se actualiza mediante un reemplazo directo con elitismo, tomando las mejores soluciones de la élite $\mathcal{S}$ y del conjunto $newPop$. El tamaño máximo de $\mathcal{S}$ siempre es igual o menor que el tamaño de población principal (representado como $popSize$).

Algorithm 1 Pseudocódigo del algoritmo MIAR-GePro

Require: $\Omega, popSize, maxGen, maxSol, \alpha, G$
Ensure: $\mathcal{S}$
1: $\mathcal{S} \leftarrow \emptyset$
2: $pop \leftarrow \emptyset$ ▷ Conjunto de soluciones de cada iteración
3: $pop \leftarrow$ createSolutions $(G, \Omega, popSize)$ ▷ Gramática G usada para generar soluciones
4: $pop \leftarrow$ evaluationProcedure $(\Omega, nExamples, pop, \alpha)$
5: $\mathcal{S} \leftarrow$ takeBestSolutions$(pop, maxSol)$ ▷ Selecciona los $maxSol$ mejores soluciones encontradas en pop
6: $generation \leftarrow 1$
7: **while** $generation \leq maxGen$ **do** ▷ El algoritmo itera un número específico de generaciones
8: $parents \leftarrow$ selectParents(pop)
9: $newPop \leftarrow$ applyGeneticOperators $(parents)$
10: $newPop \leftarrow$ evaluationProcedure $(\Omega, newPop, \alpha)$
11: $\mathcal{S} \leftarrow \mathcal{S} \cup newPop$
12: $pop \leftarrow$ takeBestSolutions$(\mathcal{S}, popSize)$ ▷ Selecciona las $popSize$ mejores soluciones encontradas en $\mathcal{S}$
13: $\mathcal{S} \leftarrow$ takeBestSolutions$(pop, maxSol)$ ▷ Selecciona las $maxSol$ mejores soluciones encontradas en pop
14: $generation \leftarrow generation + 1$
15: **end while**
16: **return** $\mathcal{S}$

17: **procedure** Evaluation Procedure(Ω, pop, α)
18: $nExamples \leftarrow$ getNumberExamples(Ω)
19: **for all** solutions $s \in pop$ **do** ▷ Soluciones en pop
20: $countS \leftarrow 0$ ▷ Ejemplos satisfechos por la regla
21: $countA \leftarrow 0$ ▷ Ejemplos satisfechos por el antecedente
22: **for all** examples $e_i \in \Omega$ **do**
23: **for all** instance $t_{i,j} \in e_i$ **do**
24: **if** $s \subseteq t_{i,j}$ **then**
25: $countS \leftarrow countS + 1$
26: **break**
27: **end if**
28: **if** antecedent$(s) \subseteq t_{i,j}$ **then**
29: $countA \leftarrow countA + 1$
30: **break**
31: **end if**
32: **end for**
33: **end for**
34: $support \leftarrow countS/nExamples$
35: $supportA \leftarrow countA/nExamples$
36: $confidence \leftarrow countS/countA$
37: **if** support $\geq \alpha$ **then**
38: assignFitness$(s, support \times confidence)$
39: **else**
40: assignFitness$(s, 0)$
41: **end if**
42: **end for**
43: **return** pop
44: **end procedure**

IV. Estudio experimental

En esta sección, analizamos el rendimiento del algoritmo MIAR-GePro considerando una serie de conjuntos de datos y teniendo en cuenta hipótesis tradicional propuesta por *Dietterich*. En primer lugar, se realizará una comparativa del rendimiento de la propuesta respecto a un algoritmo de búsqueda exhaustivo (única propuesta existente en este campo hasta la fecha). En segundo lugar, se estudiarán una serie de métricas para medir la calidad media de las soluciones. Todos estos experimentos se realizarán considerando los siguientes parámetros en MIAR-GePro: 50 individuos; 100 generaciones; 0.8 probabilidad de cruce; 0.2 probabilidad de mutación; elitismo con un número máximo de soluciones de 20; y 0.1 como umbral de soporte mínimo.

En este estudio experimental se consideran 10 conjuntos de datos multi-instancia generados artificialmente (ver descripción en la Tabla I). A pesar de que existen multitud de conjuntos de datos reales multi-instancia en la literatura, éstos han sido especialmente diseñados para tareas de clasificación y, por tanto, no proporcionan resultados interesantes para una tarea específica como ARM. Con el fin de poder ejecutar algoritmos de búsqueda exhaustiva como Apriori-MI, los conjuntos de datos que comprenden atributos numéricos se han discretizado previamente. No obstante, cabe destacar que MIAR-GePro se puede aplicar directamente tanto a atributos discretos como continuos, no requiriendo ningún paso previo de procesamiento de datos.

TABLE I
CONJUNTO DE DATOS UTILIZADOS EN EL ESTUDIO EXPERIMENTAL

Conjunto de datos	#Ejemplos	#Atributos	#Instancias	Tamaño medio ejemplos
artificial1	200	6	705	3.53
artificial2	200	8	690	3.45
artificial3	500	8	1,742	3.48
artificial4	500	8	7,406	14.81
artificial5	1,000	10	6,958	6.96
artificial6	1,000	10	12,522	12.52
artificial7	2,000	12	19,986	9.99
artificial8	2,000	16	20,213	10.11
artificial9	5,000	6	17,482	3.50
artificial10	5,000	8	37,748	7.55

A. Análisis del rendimiento

El análisis de la escalabilidad de MIAR-GePro con respecto al algoritmo Apriori-MI [5] se ha llevado a cabo por medio de dos formas diferentes (ver Figuras 2 y 3). En primer lugar, es importante destacar que la escalabilidad de los algoritmos de ARM ha sido ampliamente estudiada por diferentes investigadores [3], lo que demuestra que el tiempo de ejecución de los algoritmos de búsqueda exhaustiva aumenta exponencialmente con el número de atributos. En este estudio, queremos analizar el rendimiento de los algoritmos de búsqueda tanto evolutivos como exhaustivos cuando se aplican sobre conjuntos de datos multi-instancia, considerando diferentes números de ejemplos (Figura 2) y atributos (Figura 3). Como se muestra, el algoritmo Apriori-MI requiere un tiempo computacional más alto con el incremento del número de ejemplos (también conocidos como bolsas), obteniendo todas las posibles reglas de asociación que satisfacen los umbrales establecidos (valor mínimo de soporte de 0.1. Destacar que este valor tan bajo permite obtener casi todas las reglas de asociación existentes). En cuanto a la escalabilidad de los algoritmos para diferentes números de atributos (ver Figura 3), se obtiene que algoritmos de búsqueda exhaustivos como

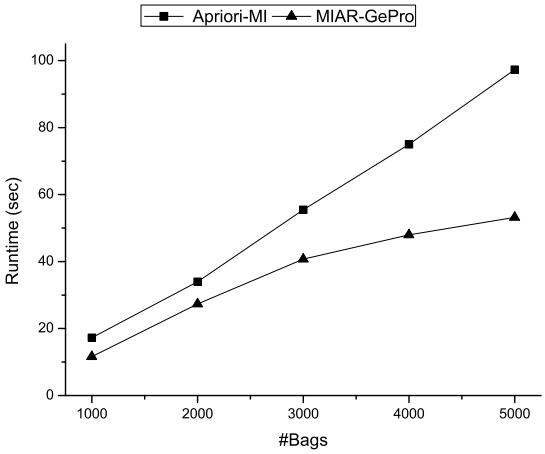


Fig. 2. Tiempo de ejecución en segundos para diferentes número de ejemplos (bolsas de instancias)

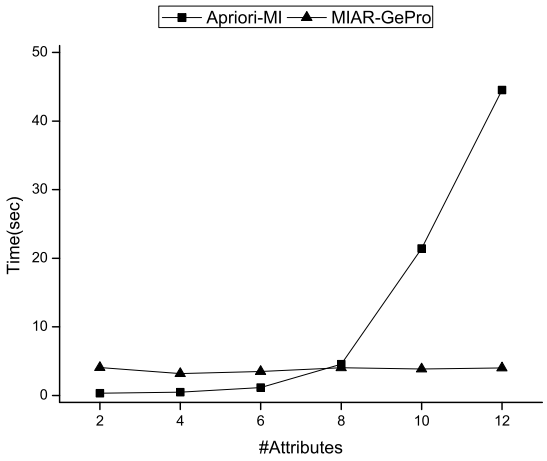


Fig. 3. Tiempo de ejecución en segundos para diferentes números de atributos

Apriori-MI aumentan exponencialmente su tiempo de cálculo con el incremento del número de atributos. Por el contrario, las propuestas evolutivas como MIAR-GePro no se ven afectadas por la cantidad de atributos.

B. Análisis de las medidas de calidad

La tabla II muestra los resultados obtenidos para diez conjuntos de datos diferentes discretizados en 5, 10 y 15 intervalos de igual anchura (la discretización es un requisito previo para poder ejecutar el enfoque de búsqueda exhaustiva existente). El algoritmo Apriori-MI se ejecuta utilizando un Soporte mínimo de 0.1, por lo que el algoritmo no tiene en cuenta ninguna regla que tenga un valor de Soporte inferior y calcula tres medidas de calidad diferentes (Soporte, Confianza y Lift). Analizando los resultados promedio representados en la Tabla II, el número de reglas descubiertas es mayor cuando el número de intervalos en los que se discretiza es menor. Esto se debe a la mayor cantidad de ejemplos que podrían satisfacerse cuando los atributos están discretizados en un pequeño número de intervalos (mayor rango de valores). Por el contrario, el mero hecho de considerar un número mayor de valores discretos (rango inferior de valores) implica que los atributos o patrones apenas se satisfacen, por lo que el número de ejemplos que comprenden patrones específicos es menor.

El mismo análisis anterior se ha realizado utilizando un enfoque evolutivo (ver Tabla III). En este caso específico, MIAR-GePro no requiere discretizar los conjuntos de datos en un número de intervalos, por lo que MIAR-GePro se ejecuta utilizando únicamente un umbral de soporte mínimo de 0.1. Además, MIAR-GePro permite extraer un número específico de reglas, por lo que hay que indicar dicho número que será el de las mejores reglas encontradas (se fija en 20 debido a que es el número de reglas generalmente utilizadas por los expertos en ARM con enfoques evolutivos [3]). El análisis de los resultados para la métrica de Confianza en el algoritmo evolutivo (no requiere preprocesado) determina mejores resultados que Apriori-MI. Es bastante interesante cómo cuanto mayor es el número de intervalos, mejores son

TABLE II
NÚMERO DE REGLAS OBTENIDAS Y VALORES MEDIOS PARA TRES MEDIDAS DE CALIDAD PARA EL ALGORITMO APRIORI-MI.

Conjunto de datos	#Reglas	Soporte	Confianza	Lift
artificial1-5	686	0.155	0.319	0.653
artificial1-10	192	0.156	0.408	1.101
artificial1-15	160	0.125	0.404	1.115
artificial2-5	1,480	0.147	0.324	0.702
artificial2-10	262	0.154	0.407	1.014
artificial2-15	208	0.128	0.406	1.112
artificial3-5	2,978	0.127	0.307	0.626
artificial3-10	140	0.240	0.548	0.994
artificial3-15	210	0.167	0.502	0.994
artificial4-5	36,788	0.147	0.244	0.365
artificial4-10	13,966	0.129	0.250	0.424
artificial4-15	430	0.313	0.560	0.987
artificial5-5	7,392	0.213	0.359	0.567
artificial5-10	294	0.294	0.542	0.993
artificial5-15	276	0.299	0.586	1.008
artificial6-5	26,670	0.172	0.263	0.372
artificial6-10	12,964	0.120	0.221	0.369
artificial6-15	538	0.289	0.538	0.998
artificial7-5	19,998	0.191	0.334	0.586
artificial7-10	35,979	0.118	0.342	0.993
artificial7-15	422	0.176	0.359	1.002
artificial8-5	15,523	0.164	0.366	0.898
artificial8-10	12,033	0.126	0.322	0.921
artificial8-15	528	0.194	0.542	1.001
artificial9-5	1,916	0.122	0.304	0.623
artificial9-10	100	0.243	0.552	1.002
artificial9-15	150	0.169	0.507	1.002
artificial10-5	4,498	0.226	0.372	0.581
artificial10-10	276	0.277	0.518	0.982
artificial10-15	224	0.309	0.592	1.024

TABLE III
NÚMERO DE REGLAS Y VALORES MEDIOS PARA TRES MÉTRICAS DE CALIDAD OBTENIDOS CON MIAR-GEPRO.

Conjunto de datis	#Reglas	Soporte	Confianza	Lift
artificial1	20	0.748	0.998	1.005
artificial2	20	0.895	1.000	1.005
artificial3	20	0.909	0.914	1.002
artificial4	20	0.975	0.999	0.999
artificial5	20	0.989	1.000	1.000
artificial6	20	0.987	0.999	0.999
artificial7	20	0.991	0.999	0.999
artificial8	20	0.995	1.000	1.000
artificial9	20	0.981	0.999	1.000
artificial10	20	0.997	0.999	0.999

los resultados obtenidos para esta medida de calidad en el enfoque de búsqueda exhaustiva. Esto explica por qué MIAR-GePro obtiene los mejores resultados, ya que no requiere un número fijo de intervalos (funciona directamente sobre atributos continuos).

V. CONCLUSIONES

El creciente interés en el almacenamiento de datos ha causado no solo un crecimiento en la dimensionalidad de los datos, sino también en la variedad de sus representaciones. Estudios recientes han propuesto soluciones para extraer reglas de asociación en conjuntos de datos donde múltiples registros describen un único objeto. Sin embargo, las soluciones existentes para la minería de reglas de asociación en este tipo de datos están restringidas a dominios discretos y requieren analizar todo el espacio de búsqueda (el tiempo de ejecución aumenta exponencialmente con el número de atributos). Así pues, el objetivo de este documento es proponer un nuevo algoritmo que permita trabajar en dominios continuos, obtener resultados lo antes posible y satisfacer las expectativas de los usuarios (los resultados deben coincidir con la forma deseada).

La propuesta MIAR-GePro utiliza programación genética gramatical para ARM sobre datos multi-instancia. MIAR-GePro puede trabajar en dominios discretos y continuos, representando una ventaja importante con respecto a la propuesta existente (Apriori-MI). Además, el modelo evolutivo propuesto es capaz de extraer las reglas de asociación en un solo paso (la propuesta existente requiere extraer primero todos los patrones frecuentes y, luego, obtener reglas de asociación de dichos patrones). El análisis experimental llevado a cabo en 10 diferentes conjuntos de datos ha demostrado que la propuesta es altamente eficiente, requiriendo un tiempo de ejecución que es casi constante para cualquier cantidad de atributos.

AGRADECIMIENTOS

Este trabajo ha sido financiado por el proyecto TIN2017-83445-P del Ministerio de Economía y Competitividad y Fondos FEDER.

REFERENCES

[1] C. C. Aggarwal and J. Han, *Frequent Pattern Mining*. Springer International Publishing, 2014.
[2] F. Padillo, J. M. Luna, F. Herrera, and S. Ventura, "Mining association rules on big data through mapreduce genetic programming," *Integrated Computer-Aided Engineering*, vol. 25, no. 1, pp. 31–48, 2018.
[3] S. Ventura and J. M. Luna, *Pattern Mining with Evolutionary Algorithms*. Springer International Publishing, 2016.
[4] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, no. 1–2, pp. 31 – 71, 1997.
[5] J. M. Luna, A. Cano, V. Sakalauskas, and S. Ventura, "Discovering useful patterns from multiple instance data," *Information Sciences*, vol. 357, pp. 23–38, 2016.
[6] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Sánchez Tarragó, and S. Vluymans, *Multiple Instance Learning - Foundations and Algorithms*. Springer, 2016. [Online]. Available: https://doi.org/10.1007/978-3-319-47759-6
[7] N. Weidmann, E. Frank, and B. Pfahringer, "A Two-level Learning Method for Generalized Multi-instance Problems," in *Proceedings of the 14th European Conference on Machine Learning*, 2003, pp. 468–479.
[8] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD Conference '93, Washington, DC, USA, 1993, pp. 207–216.
[9] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," *Data Mining and Knowledge Discovery*, vol. 8, pp. 53–87, 2004.
[10] F. Berzal, I. Blanco, D. Sánchez, and M. A. Vila, "Measuring the accuracy and interest of association rules: A new framework," *Intelligent Data Analysis*, vol. 6, no. 3, pp. 221–235, 2002.
[11] R. McKay, N. Hoai, P. Whigham, Y. Shan, and M. O'Neill, "Grammar-based Genetic Programming: a Survey," *Genetic Programming and Evolvable Machines*, vol. 11, pp. 365–396, 2010.
[12] P. Whigham and D. O. C. Science, "Grammatically-based genetic programming," in *Proceedings of the Workshop on Genetic Programming*, Tahoe City, California, USA, 1995, pp. 33–41.

Aproximación al índice externo de validación de clustering basado en chi cuadrado

José María Luna-Romera, Jorge García-Gutiérrez, María Martínez-Ballesteros and José C. Riquelme Santos
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Sevilla, España

***Abstract*—El clustering es una de las técnicas más utilizadas en minería de datos. Tiene como objetivo principal agrupar datos en clusters de manera que los objetos que pertenecen al mismo clúster sean más similares que los que pertenecen a diferentes clusters. La validación de un clustering es una tarea que se realiza aplicando los llamados índices de validación. Estos índices miden la calidad de la solución del clustering y se podrían clasificar como índices internos, los cuales calculan la calidad del clustering en función de los propios clusters; e índices externos, que miden la calidad mediante información externa de los datos, como puede ser la clase. Los índices externos que nos encontramos en la literatura están sujetos a una interpretación que puede dar lugar a error, por ello, el objetivo de este artículo es presentar un nuevo índice de validación externa basado en el test estadístico de chi cuadrado que mide la calidad del clustering de forma exacta, sin necesidad de tener que ser interpretado. Se ha realizado una experimentación usando 6 datasets que podrían ser considerados big data y los resultados obtenidos son prometedores ya que mejoran la tasa de aciertos y porcentaje de error respecto a los índices de la literatura.**

***Index Terms*—Análisis de clustering, validación de clustering, índices de validación externa, Big Data**

I. Introducción

El clustering es una técnica de minería de datos que agrupa datos no supervisados en clusters de manera que las instancias que pertenecen al mismo clúster son similares. El clustering se ha usado en diferentes áreas de conocimiento como las ciencias sociales [1], la biología [2], la electricidad [3] o la agricultura [4].

Existen numerosos métodos de clustering en la literatura, y por lo general, cada uno genera una solución de clustering diferente. En algunos casos, se pueden obtener diferentes soluciones con el mismo método con tan solo cambiar alguno de parámetros de entrada. Una de las principales tareas del clustering es el análisis y evaluación de las distintas soluciones. Para medir la calidad de la solución de clustering, existen los llamados índices de validación de clustering (CVI).

Los CVI se podrían dividir en dos categorías: índices internos e índices externos. Los índices internos miden la calidad de la solución en función de la distribución de las instancias por los clusters, es decir, evalúan la separación que existe entre los clusters y la compacidad que hay entre las instancias que pertenecen al mismo clúster. Este tipo de índice es el único que se puede aplicar cuando el dataset no aporta ningún dato adicional. Por otra parte, los índices externos son aquellos que evalúan los clusters en función de algún atributo externo como puede ser la clase. Los índices de este tipo comparan el resultado del clustering con el de una solución global denominada *ground truth*. De esta forma los índices saben a priori la solución óptima así como el número óptimo de clusters del dataset ya que el *ground truth* contiene esta información. Por lo general, los índices de la literatura indican la solución óptima a través de una representación gráfica, y los resultados podrían ser interpretados de manera imprecisa. Los CVI se han usado en [5]–[8]. En este trabajo vamos a centrarnos en los índices de validación externos.

El objetivo de este artículo es presentar un nuevo CVI de clustering externo basado en el test estadístico de chi cuadrado cuyo resultado no necesite ser interpretado. La efectividad de este nuevo índice se ha comparado con 3 índices de la literatura en 6 datasets que podrían ser consideradas big data. La implementación del índice se ha realizado haciendo uso de las librerías MLlib de Spark [9] por lo que el índice estaría preparado para ejecutarse con datasets que podrían considerarse big data.

Este artículo se organiza de la siguiente forma: Sección II trata la literatura sobre los índices externos de validación. En la Sección III se introduce el nuevo índice propuesto en este trabajo. Sección IV presenta los experimentos, la metodología y los resultados obtenidos. Y por último, las conclusiones y trabajos futuros en la Sección V.

II. Trabajos relacionados

Los índices externos evalúan los resultados de un clustering comparándolo con el *ground truth*. Los índices externos se podrían clasificar dependiendo del criterio de comparación del clustering con el *ground truth* [10]. Los índices podrían clasificarse en: *set matching*, *pair-counting* y *information theory*.

- *Set matching* es la categoría que establece que la etiqueta de cada instancia se corresponde con un clúster. Alguno de los índices de la literatura son *purity* [11], *F-measure* [12] y *Goodman-Kruskal* [13].
- Los índices *pair-counting* se basan en la comparación entre el número de instancias con la misma etiqueta y el resultado del clúster. Esta categoría incluye índices como: *rand index* [14], *adjusted rand index* [15], *Jaccard* [16], *Fowlkes-Mallows* [17], *Hubert Statistic* [18] y *Minkowski score* [19].

- Los índices basados en *information theory* como la *entropy* [11], *variation of information* [20] and *mutual information* [21] también se han aplicado en la literatura.

En los últimos años se han publicado en la literatura numerosos estudios que proponen nuevos índices externos para la validación de clusters. En [7] encontramos un nuevo índice *pair-counting* basado en un enfoque probabilístico intuitivo, que se utiliza para comparar soluciones que pueden tener un cierto grado de solape. Este índice fue probado usando 4 datasets artificiales con 6 clases y 4 datasets reales del repositorio UCI [22].

También se presentó un nuevo CVI en [23], pero en este caso, el índice se basa en la distancia Max-Min usando lo que denominan información previa. Este índice externo podría clasificarse en la categoría de *Set matching*. El rendimiento se comparó con índices de tipo *Set matching* y *Counting pairs* utilizando 6 datasets artificiales y 2 datasets reales también del repositorio UCI.

Los autores del trabajo presentado en [24] propusieron un nuevo índice basado en un conjunto de clasificadores supervisados. Podemos clasificar este índice como índice *pair-counting*. Para los experimentos se utilizaron 50 datasets reales del repositorio de la UCI y los resultados se compararon con algunos índices internos.

En [25] se presentó un nuevo CVI *pair-counting* para comparaciones analíticas. Aplica una corrección por azar y normaliza para cada grupo por separado. Los experimentos se llevaron a cabo con datasets artificiales con 3 clases y 6000 instancias cada una. Este nuevo índice obtuvo mejores resultados que otros CVI externos, tales como *purity*, *adjusted rand index* o *mutual information*.

Otros autores sugirieron en [26] un nuevo CVI de concordancia de texto basado en una concepción del grado de libertad que mide el intervalo de decisión entre dos clases. Este índice mide la calidad del clustering comparándolo con índices internos y externos. Se utilizaron 14 datasets reales para probar el rendimiento del índice.

La mayoría de estos CVI se comprueban comparando los resultados de los clusters con algunos de los CVI de la literatura y utilizando datasets sintéticos. Sin embargo, ninguno de estos índices ha sido probado en entornos paralelos y distribuidos utilizando grandes datasets. Este trabajo pretende proporcionar un CVI que permita trabajar con datasets que podrían considerarse big data y basado el test estadístico de chi cuadrado.

A. Chi cuadrado

El test estadístico de chi cuadrado es un método que mide la diferencia entre los valores esperados y los valores observados en una distribución entre dos variables [27]. La siguiente ecuación se utiliza para verificar esta correlación:

$$\chi^2 = \sum_{i}^{r}\sum_{j}^{c}\frac{(n_{ij} - E_{ij})^2}{E_{ij}} \quad (1)$$

donde r es el número de filas, c es el número de columnas, n_{ij} es el valor observado y E_{ij} es el valor esperado. E_{ij} viene dado por

$$E_{ij} = \frac{n_{ij}}{n} \quad (2)$$

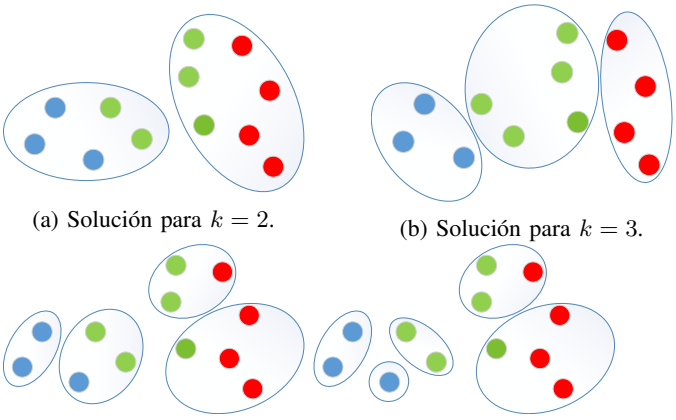
donde n es el número total de instancias. De manera que el valor de χ^2 estará más cerca de cero cuanto más se asemeje el valor observado al valor esperado.

III. Índice de validación de clustering externo basado en chi cuadrado

Supongamos una distribución de 12 instancias y 3 clases tal y como muestra la Figura 1, en la que cada punto representa una instancia y su color define la clase a la que pertenece.

Fig. 1: Representación de un dataset con 12 instancias y 3 clases donde los puntos representan a las instancias y los colores a las clases a las que pertenecen.

Antes de aplicar un método de clustering a este dataset tenemos que decidir el número de clusters (k) en el que lo vamos a dividir. Nuestro objetivo es encontrar el número óptimo de clusters de manera que las instancias que pertenecen a la misma clase queden agrupadas en los mismos clusters, y que los clusters tengan mayoritariamente instancias de una sola clase. La Figura 2 muestra las soluciones de clustering con $k = 2$ hasta $k = 5$, donde k es el número de clusters.



(a) Solución para $k = 2$.

(b) Solución para $k = 3$.

(c) Solución para $k = 4$.

(d) Solución para $k = 5$.

Fig. 2: Resultados de clustering para $k = 2$ hasta $k = 5$.

Para medir objetivamente la calidad de cada solución de clustering necesitaríamos un índice de validación externo. El

denominado *chi index* mide la calidad del clustering basandose en el test estadístico de chi cuadrado. Chi index aplica el test estadístico a la tabla de contingencia generada por la solución de clustering. Una tabla de contingencia muestra la distribución de frecuencias de las instancias, teniendo en cuenta las clases, en forma de matriz.

Siguiendo el ejemplo de la Figura 1, la Tabla I presenta la tabla de contingencia para de la solución de clustering para $k = 2$. Aquí podemos ver que el clúster 1 tiene 3 instancias de la clase azul y 2 instancias verdes, mientras que el clúster 2 tiene 4 instancias rojas y 3 verdes. Esta tabla puede ser analizada desde dos puntos de vista diferentes, teniendo en cuenta que queremos que cada clúster tenga instancias de la misma clase y que las instancias de una misma clase queden agrupadas en mismos clusters:

- Si la analizamos por filas, podemos decir que el clúster 1 está compuesto solo por instancias azules y verdes, sin instancias rojas. Y el clúster 2 está formado por instancias rojas y verdes.
- Si analizamos la tabla por columnas, podemos concluir que las instancias azules están sólo en el clúster 1, las instancias rojas están sólo en el clúster 2, y las verdes están repartidas en ambos clusters.

TABLE I: Tabla de contingencia para la solución de clustering con $k = 2$.

Clúster	Azul	Rojo	Verde	Total
1	3	0	2	5
2	0	4	3	7
Total	3	4	5	12

Estas lecturas se representan en las Tablas II y IIb, que son las tablas de contingencia expresadas con las frecuencias relativas al total de filas (Tabla IIa) y de columnas (Tabla IIb).

TABLE II: Tablas de contingencia relativas para $k = 2$.

(a) Frecuencias relativas al total de cada fila.

Clúster	Azul	Rojo	Verde	Total
1	60%	0%	40%	100%
2	0%	57%	43%	100%

(b) Frecuencias relativas al total de cada columna.

Clúster	Azul	Rojo	Verde
1	100%	0%	40%
2	0%	100%	60%
Total	100%	100%	100%

El siguiente paso sería obtener el valor de chi cuadrado para estas tablas, y realizar el mismo procedimiento en cada iteración.

En este ejemplo de juguete, hemos calculado el índice de chi para las soluciones de clustering desde $k = 2$ hasta $k = 6$. El objetivo es maximizar los valores del índice de chi en ambas tablas y minimizar la diferencia entre ellas. De esta manera, el resultado del índice de chi intentará que los valores observados y esperados sean lo más diferentes posible. Esto obligará a mantener la solución con el porcentaje más alto de clases en cada grupo.

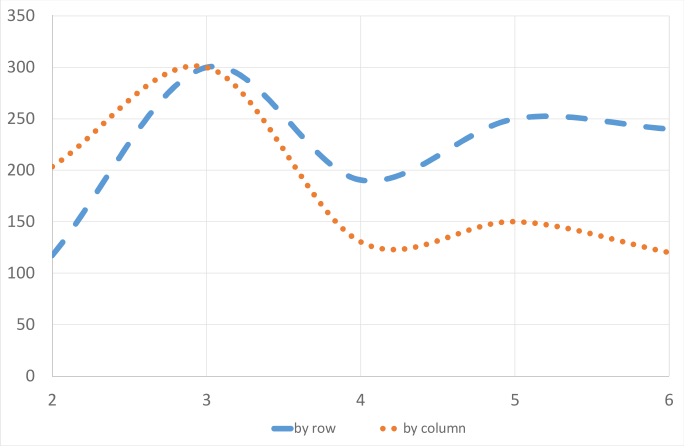


Fig. 3: Chi index result for $k = 2$ to 6.

La figura 3 muestra los resultados de nuestro ejemplo de juguete desde $k = 2$ hasta $k = 6$. Como se puede ver, en $k = 3$ las curvas alcanzaron un máximo, y también ambas curvas tienen el mismo valor. Por lo tanto, la mejor solución de clustering para este conjunto de datos es $k = 3$. Cabe destacar que este índice señala la solución cuando ambas curvas están en la distancia mínima o cruzadas. La interpretación de la solución es bastante simple porque sólo hay que mirar donde se cruzan ambas curvas.

Las tablas III muestran las tablas de contingencia relativas para la solución $k = 3$. Como se puede observar, cada clúster solo tiene instancias de una misma clase (Tabla IIIa), y cada clase está distribuida en un mismo clúster (Tabla IIIb).

TABLE III: Tablas de contingencia relativas para la solución de clustering $k = 3$.

(a) Frecuencias relativas al total de cada fila.

Clúster	Azul	Rojo	Verde	Total
1	**100%**	0%	0%	100%
2	0%	0%	**100%**	100%
3	0%	**100%**	0%	100%

(b) Frecuencias relativas al total de cada columna.

Clúster	Azul	Rojo	Verde
1	**100%**	0%	0%
2	0%	0%	**100%**
3	0%	**100%**	0%
Total	100%	100%	100%

Por lo tanto, *chi index* podría definirse como:

$$chi\,index = \min(\chi^2_{fila} - \chi^2_{columna}) \quad (3)$$

, donde

$$\chi^2_{fila} = \sum_{i}^{r}\sum_{j}^{c} \frac{(\frac{n_{ij}}{n_{i\cdot}} - \frac{n_{ij}}{n})}{\frac{n_{ij}}{n}} \quad (4)$$

$$\chi^2_{columna} = \sum_{i}^{r}\sum_{j}^{c} \frac{(\frac{n_{ij}}{n_{\cdot j}} - \frac{n_{ij}}{n})}{\frac{n_{ij}}{n}} \quad (5)$$

IV. Resultados

Esta sección describe los experimentos llevados a cabo para testear el CVI propuesto. Para realizar la comparativa se han usado 6 datasets públicos que podrían considerarse big data y se han comparado los resultados con 3 CVI de la literatura. Esta sección se compone de la Sección IV-A donde se detallan los datasets que se han utilizado en los experimentos. La Sección IV-B presenta el diseño de los experimentos seguidos. La Sección IV-D muestra los resultados de los experimentos realizados. La sección IV-D1 incluye un análisis estadístico para probar la efectividad de nuestro índice propuesto para los conjuntos de datos públicos. Finalmente, se incluye una discusión sobre los resultados en la Sección IV-D2.

A. Datasets

La tabla IV muestra los datasets utilizados para los experimentos. La tabla muestra las siguientes características: el nombre del dataset, el número de clases que va a ser usado como el número óptimo de clusters, el número de características y el número de instancias. Todos estos conjuntos de datos fueron descargados del repositorio de machine learning de UCI [22]. Todos los datasets incluyen la etiqueta de la clase, pero ésta no se ha utilizado para el proceso de clustering, solo se ha usado en la etapa de análisis.

TABLE IV: Descripción de los datasets.

Datasets	Clases	Atributos	Instancias
airlines	2	7	539,383
convtype	7	54	581,012
higgs	2	28	11,000,000
kddcup99	2	41	494,020
poker	10	10	829,202
susy	2	12	5,000,000

B. Diseño de experimentos

Para generar las soluciones de clustering, se han aplicado 3 métodos de clustering de Spark incluidos en la librería MLlib [9]: k-means, bisecting k-means y Gaussian mixture.

Estos métodos de clustering necesitan el número de clusters (k) en los que se va a particionar el dataset. Los experimentos se han realizado tomando los valores de k en el intervalo $[D_k-10, D_k+10]$ donde D_k es el número correcto de clusters del dataset siendo $k > 1$. Cada dataset se ha ejecutado con cada uno de estos 3 métodos de clustering con los que se han obtenido un total de 360 soluciones de clustering para probar los CVI. La comparativa se ha realizado entre 3 CVI de la literatura descritos en la Sección II y nuestro *chi index* propuesto.

C. Efectividad del CVI

La efectividad se mide en función de la cercanía a una solución ya dada (*ground-truth*) y realizar una comparativa de los resultados. El primer paso consiste en aplicar el algoritmo de clustering al dataset y obtener las múltiples soluciones. En el segundo paso se evalúan las soluciones de clustering aplicando los CVI. En el tercer y último paso se comparan los resultados del CVI y se selecciona el que mejor puntuación haya obtenido siguiendo.

Para hacer una comparativa entre los diferentes CVI se van a tener en cuenta los siguientes valores:

- Media de aciertos: este valor viene dado por la media de veces que el índice acierta el número óptimo de clusters por el total de datasets.
- Error medio al cuadrado: se calcula como la media de las distancias entre la predicción del índice I_i y el número correcto n_i por el total de datasets:

$$Error = \frac{\sum_{i \in n} d(I_i, n_i)^2}{n} \quad (6)$$

, donde n es el número total de datasets.

1) Test estadísticos: Por último, se ha aplicado un marco estadístico para probar el rendimiento de los CVI. Se ha seleccionado el test no paramétrico de Quade y el procedimiento post-hoc de Holm para validar estadísticamente las diferencias en los rangos medios de los *p-values* correspondientes alcanzados. Este análisis estadístico se realizó utilizando la plataforma de código abierto StatService [28].

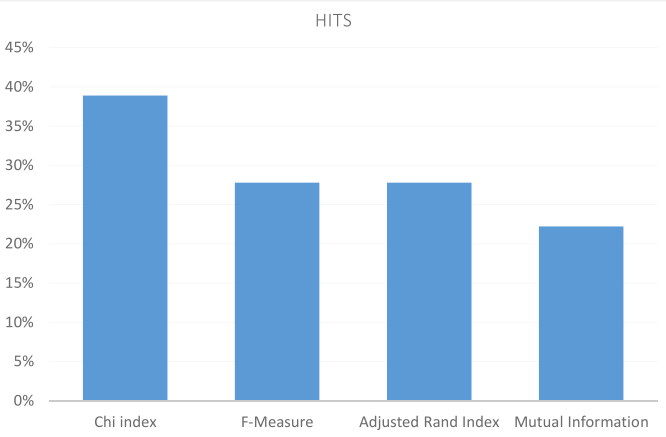
El test de Quade es una prueba estadística no paramétrica que evalúa las diferencias entre más de dos muestras estableciendo un ranking entre ellas. En nuestro caso, las muestras que vamos a evaluar son los CVI que vamos a comparar. Cuanto menor sea el *p-value*, mayor es la confianza de que un CVI funciona correctamente y, por lo tanto, se obtiene una mejor clasificación en el test de Quade.

Después de comprobar que los rankings medios son significativamente diferentes con un valor $\alpha = 0,05$, y siempre que el test de Quade rechace la hipótesis nula, se realizará el test post-hoc de Holm para evaluar el rendimiento relativo de las CVI estudiadas frente a un índice de control, en nuestro caso, tomaremos el que obtenga mejor puntuación en el ranking de Quade.

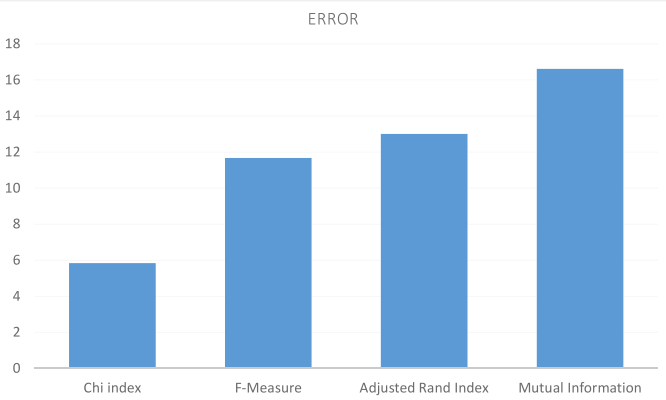
D. Resultados de los experimentos

Esta sección presenta los resultados realizados. La figura 4a muestra la media de aciertos para cada CVI en orden ascendente. *Chi index* ha alcanzado el valor más alto de aciertos (39%) con una diferencia significativa frente a sus competidores. Los índices de la literatura obtuvieron tasas similares de aciertos, que oscilaron entre el 28% en el caso de *F-Measure* y el 22% en el caso de *Mutual Information*.

Por otro lado, la Figura 4b presenta el error medio al cuadrado de cada CVI. Cabe señalar que *chi index* obtuvo el porcentaje de error más bajo (6%) quedando en segunda



(a) Media de aciertos para cada CVI.



(b) Error medio al cuadrado por CVI.

Fig. 4: Comparativa de resultados de los CVI.

posición *F-Measure* con casi el doble de puntos de error. Esto significa que *chi index* acierta el número óptimo de clusters la mayoría de las veces y en caso de error, el valor que indica no se aleja la solución.

1) Análisis estadístico: La Tabla V muestra el ranking del test de Quade para cada CVI. Como se muestra en el ranking, *chi index* ha quedado en primera posición con una puntuación de 1,807. El siguiente índice en el ranking fue *F-Measure* con una puntuación de 2,152, a 0,3 puntos del primer puesto. En última posición ha quedado *Mutual Information* con una puntuación en el ranking de 3,464.

TABLE V: Ranking del test de Quade.

CVI	Ranking
Chi index	1,807
F-Measure	2,152
Adjusted Rand Index	2,576
Mutual Information	3,464

El estadístico de Quade fue de 10,915, distribuida según una distribución F con 3 y 51 grados de libertad. El valor p de Quade fue 0,0 que fue inferior a 0.05. Por lo tanto, rechazó la hipótesis nula de que todos ellos se comportaron de manera similar con un nivel de significación de $\alpha = 0,05$.

Se ha realizado una prueba post-hoc por pares para verificar que nuestra propuesta es significativamente diferente al resto.

TABLE VI: Análisis post-hoc usando el procedimiento de Holm y tomando como índice de control a *chi index*.

CVI	p	z	α_{Holm}
Mutual Information	0,0058	2,760	0,0167
Adjusted Rand Index	0,2003	1,280	0,025
F-Measure	0,5530	0,574	0,050

La tabla VI muestra los *p-values*, el valor z y α_{Holm}, utilizando *chi index* como índice de control al ser el que mejor ranking obtuvo. Como puede verse, la hipótesis nula fue rechazada para todos los CVI. Por lo tanto, podemos concluir que *chi index* generó los mejores resultados (ya que obtuvo el mejor ranking) y fue estadísticamente diferente al resto de CVI.

2) Discusión: Los resultados del análisis de los datasets del repositorio de la UCI muestran que nuestro índice externo mejora la tasa de aciertos en 11% (Figura 4a). Además, en el caso de no poder alcanzar el número correcto de clusters, nuestro índice obtuvo una tasa de 6 puntos menos que los índices de la literatura (Figura 4b). Basado en la prueba de Quade (Tabla V), nuestro índice propuesto mejora los resultados en 2 puntos.

Es interesante resaltar que *chi index* indica la solución de clustering óptima de manera directa y concisa. A diferencia de los CVI de la literatura que necesitan ser interpretados siguiendo el método del codo y localizando máximos o mínimos locales, *chi index* indica la solución de clustering óptimo de manera directa y concisa como vimos en la Sección III.

V. Conclusiones

En este trabajo, se ha propuesto un nuevo índice de validación de clustering externo implementado en Spark para ser aplicado en datasets sin importar su tamaño. Hemos mostrado las diferencias entre nuestra propuesta y los índices de la literatura. El índice propuesto se basa en el test estadístico de chi cuadrado.

El estudio experimental indica que nuestro índice externo es muy competitivo. Hemos probado su efectividad en datasets públicos con un tamaño que podrían ser considerados big data en los que varían el número de clusters, sus características y el número de instancias. Los principales logros obtenidos son los siguientes:

- Un CVI externo basado en el test estadístico de chi cuadrado.
- Nuestro índice nos permitió estimar el número óptimo de clusters basado en la clase del dataset.
- Los resultados de *chi index* son directos y no requieren ser interpretados.
- El índice propuesto está listo para trabajar con conjuntos de datos que pueden ser considerados big data.
- El software de esta contribución se puede encontrar como un paquete de Spark en http://spark-packages.org/package/josemarialuna/externalValidity.

- El código fuente del índice de chi y los otros índices de la literatura se pueden encontrar en https://github.com/josemarialuna/ExternalValidity

Actualmente estamos aplicando este *chi index* en el análisis de datos de empleo y los resultados son prometedores. *Chi index* también se está aplicando en datos eléctricos en colaboración con una compañía eléctrica española. Como trabajo futuro, sería interesante ampliar la aplicación del índice en bases de datos que tengan multiclase.

Agradecimientos

Este trabajo ha sido apoyado por el Ministerio de Economía y Competitividad bajo el proyecto TIN2014-55894-C2-R. J.M. Luna-Romera es becario FPI del Ministerio de Economía y Competitividad.

References

[1] M. Ghane'i-Ostad, H. Vahdat-Nejad, and M. Abdolrazzagh-Nezhad, "Detecting overlapping communities in lbsns by fuzzy subtractive clustering," *Social Network Analysis and Mining*, vol. 8, no. 1, 2018, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85044313063&doi=10.1007%2fs13278-018-0502-5&partnerID=40&md5=96282433476a98b7bfc029f892772fc7

[2] F. Ginot, I. Theurkauff, F. Detcheverry, C. Ybert, and C. Cottin-Bizonne, "Aggregation-fragmentation and individual dynamics of active clusters," *Nature Communications*, vol. 9, no. 1, 2018, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85042217142&doi=10.1038%2fs41467-017-02625-7&partnerID=40&md5=0d3b0fa19ff161e129139ac5de367f18

[3] R. Perez-Chacon, R. L. Talavera-Llames, F. Martinez-Alvarez, and A. Troncoso, "Finding electric energy consumption patterns in big time series data," in *Distributed Computing and Artificial Intelligence, 13th International Conference*, S. Omatu, A. Semalat, G. Bocewicz, P. Sitek, I. E. Nielsen, J. A. García García, and J. Bajo, Eds. Cham: Springer International Publishing, 2016, pp. 231–238.

[4] X. Wu, J. Zhu, B. Wu, J. Sun, and C. Dai, "Discrimination of tea varieties using ftir spectroscopy and allied gustafson-kessel clustering," *Computers and Electronics in Agriculture*, vol. 147, pp. 64–69, 2018, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85042208353&doi=10.1016%2fj.compag.2018.02.014&partnerID=40&md5=bbb11cb2c0d295517af2c497192a4d43

[5] J. Rojas-Thomas, M. Santos, and M. Mora, "New internal index for clustering validation based on graphs," *Expert Systems with Applications*, vol. 86, pp. 334 – 349, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417417304104

[6] J. Hämäläinen, S. Jauhiainen, and T. Kärkkäinen, "Comparison of internal clustering validation indices for prototype-based clustering," *Algorithms*, vol. 10, no. 3, 2017, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85029738018&doi=10.3390%2fa10030105&partnerID=40&md5=15f131f750705ed3aad57f2d3dbba8b1

[7] D. Campo, G. Stegmayer, and D. Milone, "A new index for clustering validation with overlapped clusters," *Expert Systems with Applications*, vol. 64, pp. 549 – 556, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417416304158

[8] Z. Zhang, H. Fang, and H. Wang, "A new mi-based visualization aided validation index for mining big longitudinal web trial data," *IEEE Access*, vol. 4, pp. 2272–2280, 2016, cited By 7. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84979828536&doi=10.1109%2fACCESS.2016.2569074&partnerID=40&md5=cb527399d9c0c9990be218434656d657

[9] A. Spark, "Clustering - Spark 2.2.0 Documentation," https://spark.apache.org/docs/2.2.0/ml-clustering.html, 2018, [Online; accessed 6-april-2018].

[10] M. Rezaei and P. Fränti, "Set matching measures for external cluster validity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2173–2186, Aug 2016.

[11] Y. Zhao and G. Karypis, "Criterion functions for document clustering: Experiments and analysis," Tech. Rep., 2002.

[12] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering," in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '99. New York, NY, USA: ACM, 1999, pp. 16–22. [Online]. Available: http://doi.acm.org/10.1145/312129.312186

[13] L. A. Goodman and W. H. Kruskal, *Measures of Association for Cross Classifications*. New York, NY: Springer New York, 1979, pp. 2–34. [Online]. Available: https://doi.org/10.1007/978-1-4612-9995-0_1

[14] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971. [Online]. Available: http://www.jstor.org/stable/2284239

[15] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Is a correction for chance necessary?" in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 1073–1080. [Online]. Available: http://doi.acm.org/10.1145/1553374.1553511

[16] R. Sokal and P. Sneath, *Principles of Numerical Taxonomy*, ser. Books in biology. W. H. Freeman, 1963. [Online]. Available: https://books.google.es/books?id=3Y4aAAAAMAAJ

[17] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings," *Journal of the American Statistical Association*, vol. 78, no. 383, pp. 553–569, 1983.

[18] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, Dec 1985. [Online]. Available: https://doi.org/10.1007/BF01908075

[19] A. Ben-Hur and I. Guyon, *Detecting Stable Clusters Using Principal Component Analysis*. Totowa, NJ: Humana Press, 2003, pp. 159–182. [Online]. Available: https://doi.org/10.1385/1-59259-364-X:159

[20] M. Meilă, "Comparing clusterings by the variation of information," in *Learning Theory and Kernel Machines*, B. Schölkopf and M. K. Warmuth, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 173–187.

[21] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unit hypersphere using von mises-fisher distributions," *J. Mach. Learn. Res.*, vol. 6, pp. 1345–1382, Dec. 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1046920.1088718

[22] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[23] A. Alok, S. Saha, and A. Ekbal, "Development of an external cluster validity index using probabilistic approach and min-max distance," vol. 6, pp. 494–504, 06 2012.

[24] J. Rodríguez, M. Medina-Pérez, A. Gutierrez-Rodríguez, R. Monroy, and H. Terashima-Marín, "Cluster validation using an ensemble of supervised classifiers," *Knowledge-Based Systems*, vol. 145, pp. 1–14, 2018.

[25] M. Rezaei and P. Fränti, "Set matching measures for external cluster validity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2173–2186, Aug 2016.

[26] C. Liu, W. Wang, M. Konan, S. Wang, L. Huang, Y. Tang, and X. Zhang, "A new validity index of feature subset for evaluating the dimensionality reduction algorithms," *Knowledge-Based Systems*, vol. 121, pp. 83 – 98, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705117300291

[27] J. Antoch, "A Guide to Chi-Squared Testing : Greenwood, P.E. and Nikulin, M.S. New York: John Wiley & Sons, Inc.,.pp. 280 +XII, ISBN 0-471-55779-X. AMS 1991 Classification: 62-02, 62F03, 62H15," *Computational Statistics & Data Analysis*, vol. 23, no. 4, pp. 565–566, February 1997. [Online]. Available: https://ideas.repec.org/a/eee/csdana/v23y1997i4p565-566.html

[28] J. A. Parejo, J. Garcia, A. Ruiz-Cortes, and J. C. Riquelme, "Statservice: Herramienta de análisis estadístico como soporte para la investigación con metaheurísticas," in *Actas del VIII Congreso Expañol sobre Metaheurísticas, Algoritmos Evolutivos y Bio-inspirados*, 2012.

Atipicidad: Medida de calidad clave dentro del descubrimiento de reglas descriptivas supervisadas

C.J. Carmona[1], M.J. del Jesus[1], F. Herrera[2]
[1]*Departamento de Informática. Universidad de Jaén, España* ccarmona@ujaen.es, mjjesus@ujaen.es
[2]*Departamento de Ciencias de la Computación e Inteligencia Artificial. Universidad de Granada, España* herrera@decsai.ugr.es
[1,2]*Instituto Data Science and Computational Intelligence (DaSCI)*

***Resumen*—Esto es un resumen de nuestro trabajo publicado en la revista *Knowledge-Based Systems* [1] para su presentación en la Multiconferencia CAEPIA'18 Key Works.**

***Index Terms*—Descubrimiento de reglas descriptivas supervisadas, Descubrimiento de subgrupos, Conjuntos de contraste, Patrones emergentes, Atipicidad.**

I. Resumen

A lo largo de la literatura podemos encontrar un conjunto de técnicas que se encuentran a medio camino entre la predicción y la descripción, agrupadas bajo el nombre de descubrimiento de reglas descriptivas bajo aprendizaje supervisado (SDRD) [1], [2]. Este conjunto de técnicas intenta obtener reglas de una categoría o clase prefijada para describir información significativa y relevante del conjunto de datos.

El principal objetivo de las técnicas dentro de SDRD no es extraer un modelo para clasificar nuevas instancias, sino obtener un modelo que permita comprender, describir o encontrar fenómenos subyacentes de interés en los datos. Dentro de este área de investigación se agrupan todas aquellas técnicas que, mediante el uso de reglas y un modelo de aprendizaje supervisado, intentan obtener conocimiento descriptivo de los datos que sea significativo, inusual y de interés para el usuario, como el descubrimiento de subgrupos [3], la minería de patrones emergentes [4] y los conjuntos de contraste [5], entre otros. A continuación se describen brevemente estas técnicas.

Descubrimiento de subgrupos: Se define como el descubrimiento de subgrupos de la población estadísticamente interesantes, esto es, tan grandes como sea posible y con una distribución estadística de la propiedad de interés lo más atípica posible respecto al conjunto de la población. La medida de calidad para medir esta atipicidad [6] en una regla R se define como:

$$WRAcc(R) = \frac{p+n}{P+N}\left(\frac{p}{p+n} - \frac{P}{P+N}\right) \quad (1)$$

Conjuntos de contraste: Es una técnica de exploración para contrastar diferencias de grupos dentro de un conjunto de datos, es decir, descubrir conjuntos de ejemplos con amplias diferencias de soporte entre grupos del conjunto de datos y se mide mediante la diferencia de soporte [5]:

$$DS(R) = \left|\frac{p}{P} - \frac{n}{N}\right| \ \geq \ \delta \quad (2)$$

Patrones emergentes: Se centra en buscar conocimiento relacionado con los valores de una clase, donde el número de instancias cubiertas por un patrón sea muy elevado para un valor de la variable objetivo y muy bajo o nulo para el resto; es decir, que el mismo patrón tenga un soporte muy alto para una clase y muy bajo para las demás clases del problema, y viene dado por el índice de crecimiento [4] que se representa a continuación:

$$GR(R) = \frac{\frac{p}{P}}{\frac{n}{N}} > 1 \quad (3)$$

Centrándonos en la importancia de la medida de atipicidad hay que destacar que mide el equilibrio entre generalidad y precisión. En concreto, el segundo factor de la atipicidad es el factor que mide la ganancia de precisión, y tal y como se puede observar, es posible que la aticipidad de una regla sea inferior a cero cuando la regla tiene una baja calidad ya que el porcentaje de ejemplos negativos cubiertos es superior al de los positivos. En este sentido, una regla de interés para el experto debería tener siempre un valor positivo, es decir, se debe cumplir la siguiente desigualdad para obtener una regla de interés:

$$\frac{p}{p+n} > \frac{P}{P+N} \quad (4)$$

Si descomponemos esta desigualdad [1] podemos decir que para que una regla obtenga una atipicidad positiva y ser un subgrupo de interés, obtenemos que el producto escalar entre $p\overline{n}$ debe ser superior que $\overline{p}n$. En consecuencia, esta descomposición nos indica que el producto de ejemplos cubiertos y no cubiertos correctamente debería ser superior al producto de ejemplos cubiertos y no cubiertos incorrectamente:

$$p\ \overline{n} > \overline{p}\ n \quad (5)$$

En el trabajo publicado en [1] se presenta tras este primer análisis, la compatibilidad entre el descubrimiento de subgrupos, los conjuntos de contraste y los patrones emergentes gracias al uso de esta medida de calidad.

Por ejemplo, en el caso de los patrones emergentes (según definición) podemos afirmar que una regla es emergente cuando su índice de crecimiento es superior a 1, es decir:

$$GR(R) = \frac{\frac{p}{n}}{\frac{P}{N}} > 1 \quad (6)$$

Si aplicamos una descomposición a esta formulación descubrimos que:

$$\frac{\frac{p}{n}}{\frac{P}{N}} > 1$$
$$p\ \overline{n} > \overline{p}\ n \quad (7)$$

Según esta demostración podemos afirmar que la descomposición de la atipicidad positiva y del índice de crecimiento son similares, es decir, si un subgrupo es de interés también es un patrón emergente. Además, por otro lado se puede observar una relación directa entre la atipicidad y la diferencia de soporte en [2] que dice:

$$DS(R) = \frac{WRAcc(R)}{p(PIS) \cdot p(NIS)} \geq \delta \quad (8)$$

donde $p(PIS)$ es el porcentaje de ejemplos positivos para la clase del conjunto de datos a analizar y $p(NIS)$ es el porcentaje de ejemplos negativos para el conjunto de datos.

Al mismo tiempo, es importante tener en cuenta que el dominio de la atipicidad para un problema depende del porcentaje de los ejemplos para la clase o variable objetivo [7], y por tanto, el dominio se puede determinar mediante las siguientes ecuaciones para el límite inferior (LB_{WRAcc}) y para el superior (UB_{WRAcc}):

$$LB_{WRAcc} = (1 - p(PIS)) \cdot (0 - p(PIS)) \quad (9)$$

$$UB_{WRAcc} = p(PIS) \cdot (1 - p(PIS)) = p(PIS) \cdot p(NIS) \quad (10)$$

Es decir, el valor de DS depende de las propiedades del problema ya que está directamente relacionado con el porcentaje de ejemplos de la clase a analizar, y en resumen:

$$DS(R) = \frac{WRAcc(R)}{p(PIS) \cdot p(NIS)} = \frac{WRAcc(R)}{UB_{WRAcc}} \geq \delta \quad (11)$$

Una regla se considera conjunto de contraste con un $\delta = 0{,}10$ cuando:

$$WRAcc(R) \geq 0{,}10 \cdot UB_{WRAcc} \quad (12)$$

Sin embargo, tenemos la necesidad de estandarizar este valor de $WRAcc$ a $WRAcc$ normalizada ($WRAccN$) ya que debemos evitar la utilización de un valor δ que esté condicionado a este porcentaje de ejemplos de la clase a analizar. Esta mejora se consigue con respecto a la homogeneización de esta medida clave dentro del descubrimiento de reglas descriptivas supervisadas. La $WRAccN$ se normaliza en el intervalo [0,1] mediante la siguiente ecuación donde una regla es subgrupo de interés cuando su valor sea superior a 0.5.

$$WRAccN(R) = \frac{WRAcc(R) - LB_{WRAcc}}{UB_{WRAcc} - LB_{WRAcc}} \quad (13)$$

Considerando un δ igual a 0.10, y el intervalo positivo de $WRAccN$ igual a (0.5, 1.0], una regla se considera conjunto de contraste con un $\delta = 0{,}10$ cuando:

$$WRAccN(R) \geq 0{,}50 + 0{,}10 \cdot (1{,}00 - 0{,}50)$$
$$WRAccN(R) \geq 0{,}55 \quad (14)$$

, es decir, cuando el valor de la $WRAccN$ es superior o igual que el 10 % de la parte positiva de $WRAcc$.

En conclusión, la atipicidad es un factor clave a medir dentro del descubrimiento de reglas descriptivas supervisadas donde una regla con un valor de $WRAccN$ superior a 0.50 se considera de interés para el descubrimiento de subgrupos, y emergente para la extracción de patrones emergentes. Además, si la $WRAccN$ es superior o igual a 0.55 se considera también regla de contraste. Esta demostración muestra la relación directa entre todas estas técnicas que se han agrupado dentro del descubrimiento de reglas descriptivas supervisadas y muestra la importancia de medir la calidad de una regla con respecto a la $WRAcc$ en un problema que se desea analizar. Por otro lado, es interesante indicar que solo con el análisis de esta medida de calidad, los expertos serán capaces de determinar si la regla representa un subgrupo, una regla emergente y/o una regla de contraste. De la misma forma, esta medida debe ser clave para el diseño de nuevas propuestas dentro del descubrimiento de reglas descriptivas supervisadas.

Agradecimientos

Este trabajo ha sido subvencionado por el Ministerio de Economía y Competitividad de España bajo el proyecto TIN2014-916 57251-P y TIN2015-68454-R (Fondos FEDER).

Referencias

[1] C. J. Carmona, M. J. del Jesus, and F. Herrera, "A Unifying Analysis for the Supervised Descriptive Rule Discovery via the Weighted Relative Accuracy," *Knowledge-Based Systems*, vol. 139, pp. 89–100, 2018.

[2] P. Kralj-Novak, N. Lavrac, and G. I. Webb, "Supervised Descriptive Rule Discovery: A Unifying Survey of Constrast Set, Emerging Pateern and Subgroup Mining," *Journal of Machine Learning Research*, vol. 10, pp. 377–403, 2009.

[3] W. Kloesgen, "Explora: A Multipattern and Multistrategy Discovery Assistant," in *Advances in Knowledge Discovery and Data Mining*. American Association for Artificial Intelligence, 1996, pp. 249–271.

[4] G. Z. Dong and J. Y. Li, "Efficient Mining of Emerging Patterns: Discovering Trends and Differences," in *Proc. of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, 1999, pp. 43–52.

[5] S. D. Bay and M. J. Pazzani, "Detecting group differences: Mining contrast sets," *Data Mining and Knowledge Discovery*, vol. 5, no. 3, pp. 213–246, 2001.

[6] N. Lavrac, P. A. Flach, and B. Zupan, "Rule Evaluation Measures: A Unifying View," in *Proc. of the 9th International Workshop on Inductive Logic Programming*, ser. LNCS, vol. 1634. Springer, 1999, pp. 174–185.

[7] C. J. Carmona, V. Ruiz-Rodado, M. J. del Jesus, A. Weber, M. Grootveld, P. González, and D. Elizondo, "A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans," *Information Sciences*, vol. 298, pp. 180–197, 2015.

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 3:
APLICACIONES

Modeling the navigation on enrolment web information area of a university using machine learning techniques*

*Note: The full contents of this paper have been published in the volume *Lecture Notes in Artificial Intelligence 11160* (LNAI 11160)

Ainhoa Yera, Iñigo Perona, Olatz Arbelaitz, Javier Muguerza
Faculty of Informatics
University of the Basque Country
Donostia, Spain
{ainhoa.yera, inigo.perona, olatz.arbelaitz, j.muguerza}@ehu.eus

***Abstract*—This work analyses the navigation in the enrolment web information area of the University of the Basque Country. A complete data mining process shows that successful and failure navigation behaviors can be modeled using machine learning techniques. Unsupervised learning algorithms have been applied on two different domains: URLs visited by the users in each session (navigation sequence) and some interaction parameters extracted from the recorded click-stream (navigation style). Both domains have been used satisfactorily to model the behavior of success and failure navigation sessions achieving more than 78 % of accuracy predicting success or failure sessions. Furthermore, the clustering based on the navigation style was able to identify the main characteristics of each type of session and to build a subsystem that enables to detect failure type sessions with high precision.**

***Index Terms*—Web Usage Mining, Navigation Models, Web Interaction Characterization**

Risk factors for development of antibiotic resistance of *Enterococcus faecium* to Vancomycin. A subgroup discovery approach*

*Note: The full contents of this paper have been published in the volume *Lecture Notes in Artificial Intelligence 11160* (LNAI 11160)

A. Fajfar
Faculty of Health Sciences
University of Maribor
Maribor, Slovenia

M. Campos
Faculty of Computer Science
University of Murcia
Murcia, Spain
manuelcampos@um.es

F. Palacios
University Hospital of Getafe
Madrid, Spain

B. Canovas-Segura
Faculty of Computer Science
University of Murcia
Murcia, Spain

G. Stiglic
Faculty of Health Sciences
University of Maribor
Maribor, Slovenia

R. Marin
Faculty of Computer Science
University of Murcia
Murcia, Spain

***Abstract*—Health-care associated infections (HAI) are infections that are not present or incubated at the time of admission to hospital. HAI are one of the major causes of morbidity and mortality among immunocompromised patients and have an important economic impact. The bacteria isolated in microbiology cultures can be treated with a limited combination of antibiotics owing to their resistance to many groups of antibiotics, which represents a major challenge. This paper focuses on the problem of vancomycin resistant *Enterococcus faecium* (VREfm) due to its high prevalence in HAI, multidrug resistance and ability to survive under intense selective pressure. We use the subgroup discovery technique to identify target populations with high risk clinical factors for VREfm infections that we shall be able to incorporate into a clinical decision support system for antimicrobial stewardship program. The dataset used contained 201 susceptibility tests with *Enterococcus faecium* from a University Hospital in years 2014 and 2015. The clinician evaluated and discussed the knowledge reported by the most interesting subgroups based on their positive predictive value and sensitivity.**

***Index Terms*—**

Algoritmos de aprendizaje automático para predicción de niveles de niebla usando ventanas estáticas y dinámicas.

M. Díaz-Lozano[a], D. Guijo-Rubio[a], P. A. Gutiérrez[a], C. Casanova-Mateo[b,c], S. Salcedo-Sanz[d], C. Hervás-Martínez[a]

***Resumen*—Los eventos de muy baja visibilidad producidos por niebla son un problema recurrente en ciertas zonas cercanas a ríos y grandes montañas, que afectan fuertemente a la actividad humana en diferentes aspectos. Este tipo de eventos pueden llegar a suponer costes materiales e incluso humanos muy importantes. Uno de los sectores más influenciados por las condiciones de muy baja visibilidad son los medios de transporte, fundamentalmente el transporte aéreo, cuya actividad se ve seriamente mermada, provocando retrasos, cancelaciones y, en el peor de los casos, terribles accidentes. En el aeropuerto de Valladolid son muy frecuentes las situaciones de baja visibilidad por niebla, especialmente en los meses considerados de invierno (noviembre, diciembre, enero y febrero). Esto afecta de forma directa a la manera en la que operan los vuelos de este aeropuerto. De esta forma, es muy importante conocer las posibles condiciones de niebla a corto plazo para aplicar procedimientos de seguridad y organización dentro del aeropuerto. En el presente artículo se propone el uso de diferentes modelos de ventanas dinámicas y estáticas junto con clasificadores de aprendizaje automático, para la predicción de niveles de niebla. En lugar de abordar el problema como una tarea de regresión, la variable de interés para la caracterización del nivel de visibilidad en el aeropuerto (Rango Visual de Pista, RVR) se discretiza en 3 categorías, lo que aporta mayor robustez a los modelos de clasificación obtenidos. Los resultados indican que una combinación de ventana dinámica con ventana estática, junto con modelos de clasificación basados en *Gradient Boosted Trees* es la metodología que proporciona los mejores resultados.**

***Palabras clave*—Series temporales, Eventos de baja visibilidad, modelos autorregresivos, predicción.**

I. Introducción

La niebla es un fenómeno meteorológico que consiste en la aparición de gotas de agua en suspensión en forma de gotas, lo suficientemente pequeñas como para que la gravedad terrestre no las atraiga hacia la superficie. Este fenómeno se manifiesta a nivel de suelo, pudiendo considerarse una nube a muy baja altura. Su aparición puede deberse a diferentes causas, como la evaporación de la humedad del suelo o a la expedición de vapor por parte de vegetación o de grandes masas de agua [1]. En cualquier caso, su aparición está íntimamente ligada a la disminución de las condiciones de visibilidad en la superficie. Estos eventos de baja visibilidad suponen un riesgo particular para el tráfico aéreo, marítimo y terrestre, provocando interrupciones y problemas cuyos costes humanos se han llegado a comparar a los causados por tormentas y tornados [2]. En las operaciones llevadas a cabo en los aeropuertos, el trafico de vuelos se ve fuertemente afectado en estas condiciones [3], [4], debiendo ampliar el tiempo entre aterrizajes y despegues y pudiendo provocar retrasos y cancelaciones. Por esta razón, el personal de los aeropuertos necesita conocer con cierta precisión y antelación si en un futuro cercano tendrán que trabajar con condiciones de baja visibilidad por niebla, para activar los protocolos necesarios en su caso, y tratar de mitigar este tipo de situaciones problemáticas.

La predicción de eventos futuros tiene interés en la mayoría de campos de estudio, creándose multitud de modelos destinados para este cometido [5], [6], aplicados con éxito a problemas de predicción reales. Todos estos modelos suelen estar basados en análisis de series temporales, normalmente sobre codificación real. Los eventos de niebla que provocan baja visibilidad son un problema recurrente en multitud de aeropuertos en todo el mundo, y su predicción se ha enfocado mediante diversas técnicas: en Perth, el aeropuerto más grande de la costa sudoeste de Australia, se desarrolló un modelo que aplicaba lógica difusa para conseguir una predicción precisa [7]; en el aeropuerto de Calcuta, India, se abordó el problema utilizando árboles de decisión para identificar los parámetros más importantes que influyen en la visibilidad, realizando predicciones mediante una red neuronal [8]; en el aeropuerto de Valladolid, situado al noroeste de España, se afrontó este problema haciendo uso de diversos algoritmos de aprendizaje automático para regresión como máquinas de vectores soporte, preprocesando la serie antes de aplicarlos [9].

El objetivo de este artículo es proponer un modelo de

[a]: Dpto. de Informática y Análisis Numérico, Universidad de Córdoba, Córdoba, España. E-mail: {i42dilom, dguijo, pagutierrez, chervas}@uco.es

[b]: LATUV: Laboratorio de Teledetección, Universidad de Valladolid, Valladolid, España.

[c]: Dpto. de Ingeniería Civil: Construcción, Infraestructura y Transporte, Universidad Politécnica de Madrid, Madrid, España.

[d]: Dpto. de Teoría de la Señal y Comunicaciones, Universidad de Alcalá, Alcalá de Henares, Madrid, España. E-mail: sancho.salcedo@uah.es

Este trabajo ha sido desarrollado con la financiación de los proyectos TIN2017-85887-C2-1-P, TIN2017-85887-C2-2-P y TIN2017-90567-REDT del Ministerio de Economía y Competitividad de España (MINECO) y fondos FEDER. La investigación de David Guijo Rubio ha sido subvencionada por el proyecto PI15/01570 de la Fundación de Investigación Biomédica (FIBICO) y por el Programa Predoctoral FPU (Ministerio de Educación y Ciencia de España), referencia de beca FPU16/02128.

predicción horario basado en el preprocesamiento de series temporales categóricas, donde los eventos temporales son tres diferentes condiciones atmosféricas relacionadas con la aparición de eventos de baja visibilidad por niebla: no-niebla, neblina y niebla. Este preprocesamiento se lleva a cabo mediante el uso de diversos tipos de ventanas de valores pasados de las series temporales consideradas, de forma que la información obtenida a partir de cada serie pueda ser usada en el entrenamiento de cualquier modelo de aprendizaje automático.

En la siguiente sección (sección II), se presenta la base de datos considerada. En la sección III, se detallan los modelos de ventana propuestos para preprocesar los datos y para realizar las predicciones. En la sección IV, se detalla la configuración de la experimentación y los resultados obtenidos. Por último, la sección V incluye las conclusiones obtenidas a partir de los resultados conseguidos.

II. Base de datos

II-A. Origen de los datos

Los datos utilizados en este artículo han sido recopilados mediante un sistema localizado en el aeropuerto de Valladolid y perteneciente a la Agencia Estatal de Meteorología Española (AEMET). Dicho sistema provee información relevante sobre las condiciones meteorológicas al personal de los aeropuertos (pilotos, controladores y personal de tierra). De forma horaria, el sistema recoge información sobre diferentes factores meteorológicos, tales como la temperatura o la humedad. Dichos datos son recogidos por sensores y, por tanto, no existen datos perdidos, pudiéndose considerar cada variable meteorológica obtenida como una serie temporal. Las variables recogidas por este sistema son las siguientes: Temperatura (grados Celsius), Humedad relativa (%), Velocidad del viento (m/s), Dirección del viento (grados), Presión reducida al nivel del mar (QNH, hPa) y Rango Visual de Pista (*Runaway Visual Range, RVR*), que es la variable objetivo a predecir y se mide en metros.

El RVR se obtiene a partir de la media ponderada de tres sensores de visibilidad (visibilímetros), colocados a diferentes alturas de la pista (zona de toma de tierra, media pista y zona de parada). Además, las condiciones de niebla pueden modelarse mediante la combinación del resto de variables meteorológicas medidas, por estar ambas intrínsecamente relacionadas. Los experimentos llevados a cabo en este artículo se han realizado con datos horarios de 8 años completos, concretamente desde el 1 de noviembre de 2009 hasta el 31 de diciembre de 2016. De la totalidad de datos, el 70 % inicial ha sido usado para la fase de entrenamiento y el último 30 % para la fase de generalización.

II-B. Umbralización del RVR

Los visibilímetros utilizados en el aeropuerto de Valladolid solo obtienen medida hasta 2000m, considerando situaciones de visibilidad óptima por encima de esta medida (es decir marcan 2000m incluso cuando el valor real de visibilidad es más alto). Esto es debido a que los valores de visibilidad por encima de 2000m son considerados óptimos, y por tanto no son relevantes para la gestión de situaciones de baja visibilidad en el aeropuerto. En la experimentación diseñada en este artículo se propone un enfoque de predicción categórico mediante 3 clases, en el que a cada hora se le asigna una de las 3 posibles clases (alta, media o baja en relación al valor de RVR, asociado a las condiciones de niebla). Los umbrales utilizados para la discretización de los valores de RVR son los siguientes:

$$\text{Clase} = \begin{cases} \text{niebla}, & \text{si } RVR < 1000, \\ \text{neblina}, & \text{si } 1000 <= RVR < 1990, \\ \text{no niebla}, & \text{si } RVR >= 1990. \end{cases}$$

De esta forma, se obtiene un problema de clasificación con 3 clases, altamente desequilibrado, debido a que las condiciones de baja visibilidad son, afortunadamente, muy minoritarias respecto a situaciones de visibilidad óptima. La Tabla I muestra la proporción de las diferentes clases en los conjuntos de entrenamiento y generalización considerados.

Tabla I
Proporción de clases

Clase	Entrenamiento	Generalización
niebla	856 (6 %)	772 (12 %)
neblina	912 (6 %)	514 (8 %)
no niebla	12294 (88 %)	4740 (80 %)
Total	14062 (70 %)	6026 (30 %)

III. Modelos utilizados

Una serie temporal se define como un conjunto de datos cronológicamente ordenados y muestreados con una frecuencia constante. Formalmente, una serie temporal unidimensional se define como:

$$Y = \{y_0, y_1, y_2, ..., y_N\},$$

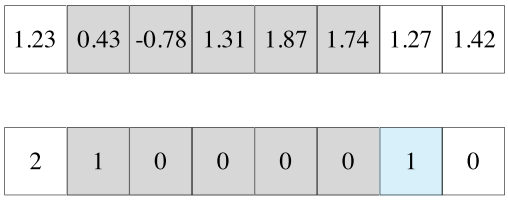
donde N es la longitud de la serie temporal.

La experimentación desarrollada en este artículo se ha llevado a cabo mediante el análisis de las series temporales descritas en la sección II-A, utilizando diferentes ventanas de valores pasados. Inicialmente, se crea un conjunto de patrones en los que la variable dependiente es el valor de la serie objetivo en el instante de tiempo a predecir y las variables independientes están formadas por la información extraída a partir de las ventanas. Con este conjunto de patrones, entrenamos cualquier modelo de aprendizaje automático, por lo que podríamos considerar que las ventanas actúan como un método de preprocesamiento.

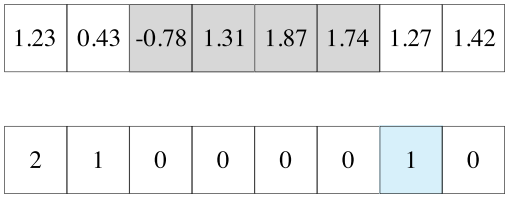
III-A. Extracción de características basada en ventanas

El análisis de series temporales mediante métodos autorregresivos permite modelar una variable en función de valores pasados, tanto de ella misma como de otras variables independientes relacionadas con el problema. En este artículo se propone el uso de 3 métodos, cada uno de los cuales limita de distinta forma la ventana de valores pasados utilizada para predecir el siguiente. Estos métodos pueden ser utilizados de forma individual, aplicando un único tipo de análisis,

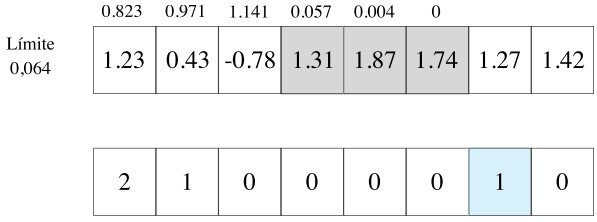
o combinada, de forma que se pueda aunar la información obtenida mediante varios tipos de ventanas. La Figura 1 muestra gráficamente el funcionamiento de estos 3 métodos en un problema sintético: una serie temporal categórica asociada a una variable dependiente a predecir y una serie temporal de valores reales asociada a una variable independiente, correlada con la primera. En dicha figura, el valor de la serie dependiente a predecir se encuentra sombreado en azul, mientras que las ventanas utilizadas para ello se encuentran sombreadas en gris.



(a) Ventana fija con *p=5*



(b) Ventana dinámica basada en cambio de clase



(c) Ventana dinámica basada en varianza con *r=0.1*

Figura 1. Distintos tipos de ventana propuestos.

III-A1. Modelo de ventana fija (VF): utiliza un número de instantes pasados constante para la predicción de cada valor de la serie temporal. Esta ventana es la que emplean los modelos autorregresivos (AR) clásicos, comúnmente utilizados en Estadística. El número de instantes es un parámetro del modelo, p (o orden del modelo AR), y los resultados obtenidos mediante este preprocesamiento son muy sensibles al valor de p. El modelo AR de orden p puede definirse como:

$$X_t = c + \sum_{i=1}^{p} \alpha_i X_{t-i} + \varepsilon_t \quad (1)$$

donde c representa una constante, α_i el coeficiente correspondiente al valor de la serie temporal en el instante $t-i$ y ε_t ruido blanco, esto es, una variable aleatoria Normal de media 0 y de varianza a determinar. En la Figura 1(a) se muestra un ejemplo de este tipo de preprocesamiento haciendo uso de una ventana fija de tamaño 5. Como se puede ver en dicha figura, para prededecir el valor sombreado en azul hace uso de los 5 valores anteriores de todas las series involucradas en el problema, tanto la dependiente como las independientes.

III-A2. Modelo de ventana dinámica basada en cambio de clase (VDCC): en series temporales categóricas, este modelo crea ventanas de tamaño variable limitándolas en base a los cambios de clase. De esta forma, para predecir un determinado valor, se identifica la clase inmediatamente anterior y se añaden valores anteriores mientras la clase no cambie, creándose ventanas más grandes cuando la clase es estable (existe una racha). En este tipo de preprocesamiento, los valores pasados de la serie dependiente son utilizados de forma indirecta dado que, aunque no se utilizan sus valores, son estos los que determinan el tamaño de las ventanas que se crean. En la Figura 1(b) se representa gráficamente la ventana creada para un determinado valor. En este ejemplo, la clase inmediatamente anterior al valor a analizar es 0, por lo que la ventana se extenderá hacia detrás, hasta que se encuentre una categoría distinta de 0, creando en este caso una ventana de 4 elementos. Tras ello, la información de las muestras de cada serie independiente que caigan dentro de la ventana se resume mediante las métricas detalladas en la sección IV-A.

III-A3. Modelo de ventana dinámica basada en varianza (VDV): en series temporales de valores reales, proponemos este modelo que crea ventanas de tamaño variable en función de la dinámica de la serie. Se analiza, de forma individual para cada serie temporal, la varianza de los valores incluidos en la ventana. Se añaden valores previos a la ventana hasta que se alcanza un determinado límite de varianza, que se establece como un porcentaje de la varianza total de la serie. Dicho porcentaje es un parámetro del modelo, r. La idea subyacente es que, cuando la varianza es demasiado alta, puede no tener sentido intentar resumir la información incluida en la misma. Además, dado que cada serie temporal se analiza de forma independiente, se pueden obtener ventanas de distinto tamaño para cada variable de entrada, cuando haya muchas variables independientes (como sucede en el problema considerado). La Figura 1(c) muestra un ejemplo de este procesamiento con un porcentaje de varianza total del 10 %. Este procesamiento no puede aplicarse sobre la serie categórica. En la serie de valores reales, se muestra, sobre cada valor, la varianza parcial, de forma tal que la ventana utilizada para predecir el valor sombreado en azul crecerá mientras que la varianza parcial sea menor que el límite (10 % de la total). Las ventanas serán resumidas mediante las métricas de la sección IV-A.

III-B. Clasificadores

La información obtenida tras el uso de las diferentes combinaciones de ventanas será utilizada para entrenar clasificadores que realicen la predicción de la categoría. En una primera aproximación, hemos considerado algunos de los modelos más clásicos de los incluidos en *scikit-learn* [15], una librería de código abierto implementada en *Python*. De esta forma, los clasificadores considerados son: Regresión logística (RL) [11],

Árboles de decisión (AD) [12] y conjuntos de clasificadores (concretamente *RandomForest (RF)* [13] y *GradientBoosting-Classifier (GB)* [14]).

IV. Experimentación y resultados

IV-A. Configuración

Como ya se especificó en la sección II-A, un 70 % de los datos se ha empleado como conjunto de entrenamiento, mientras que el 30 % formó el conjunto de *test*.

Para ambas ventanas dinámicas (tanto la basada en varianza como en el cambio de clase), se ha elegido resumir la información de las ventanas creadas utilizando dos estadísticos:

- La media aritmética ($\overline{W_s}$) de la ventana, definida como:

$$\overline{W_s} = \frac{1}{s}\sum_{y \in W_s} y$$

  donde s es el número de valores de la ventana y W_s la ventana creada.
- La varianza ($S^2_{W_s}$), utilizada para obtener una medida de dispersión de los valores contenidos en la ventana y definida como:

$$S^2_{W_s} = \frac{1}{s-1}\sum_{y \in W_s} (y - \overline{W_s})^2.$$

Los parámetros de los métodos de preprocesamiento y de los clasificadores han sido optimizados mediante una validación cruzada anidada de tipo 5-*fold*. A continuación se indican los valores explorados. Para el análisis de ventana fija, se ha utilizado un numero de muestras previas $p \in \{1, 2, 3, \ldots, 6\}$. Para el análisis de ventana dinámica basado en varianza, se ha optado por optimizar la proporción de varianza total utilizada para limitar las ventanas según $r \in \{0,1; 0{,}2; 0{,}3; \ldots; 0{,}6\}$.

Los parámetros de los modelos de clasificación también han sido optimizados mediante validación cruzada. Para RL, se ha optimizado la intensidad de la regularización mediante el parámetro $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$. Para los AD se ha optimizado la función para medir la calidad de una división, utilizando el coeficiente *Gini* y la entropía. En el caso de RF, se ha optimizado el número de árboles en cada bosque $n \in \{10, 50, 100\}$, y, para GB, el número de árboles utilizado en el algoritmo ha sido $n \in \{50, 100, 150, 200\}$ y el ratio de aprendizaje $l \in \{0{,}05; 0{,}1; 0{,}2; 0{,}3\}$.

IV-B. Métricas de rendimiento

Existen multitud de métricas para evaluar la calidad de una clasificación obtenida. Una de las más comunes es el porcentaje de patrones bien clasificados (*Correct Classified Ratio, CCR*) con la que se obtiene una medida de rendimiento global. No obstante, el problema a tratar en este artículo es altamente desequilibrado, haciendo que cualquier clasificador trivial que clasifique todos los patrones en la clase mayoritaria obtenga buena puntuación en *CCR*. Por ello, el rendimiento ha de medirse mediante otras métricas. Hemos elegido la Media Geométrica de Sensibilidades (*Geometric Mean of the Sensitivities, GMS*), una medida de rendimiento que tiene en cuenta la precisión de la clasificación en todas las clases. El GMS se define como:

$$GMS = \sqrt[n]{\prod_{i=1}^{n} S_i}$$

donde n es el número de clases y S_i la sensibilidad de la clase i-ésima, es decir:

$$S_i = \frac{n_i}{N_i}$$

siendo n_i el número de instancias de la clase *i* correctamente clasificadas y N_i el número total de instancias de la clase *i*. Es una medida a maximizar.

Por otro lado, la naturaleza ordinal de nuestro problema (las categorías están organizadas en una escala ordinal) hace que sea necesario obtener una medida de error que penalice más algunos tipos de errores (por ejemplo, la confusión de "no niebla" con "niebla alta"), para lo que se utilizará la Media de Errores Absolutos Medios (*Average Mean Absolute Error, AMAE*). El $AMAE$ se define como:

$$AMAE = \frac{1}{n}\sum_{i=1}^{n} MAE_i$$

donde n es el número de clases y MAE_i es la desviación media producida con respecto a la clase real considerando solo la clase *i*. Se define como:

$$MAE_i = \frac{1}{N_i}\sum_{i=1}^{N_i} |C(y_i) - C(\hat{y_i})|$$

siendo N_i el número de instancias de la clase *i*, $C(y_i)$ la clase real de la instancia i y $C(\hat{y_i})$ la clase predicha para la instancia *i*, representadas mediante valores numéricos (es decir, $C(y_i) = 0$ para no niebla, $C(y_i) = 1$ para no neblina y $C(y_i) = 2$ para niebla). Al tratarse de una medida de error, es una métrica a minimizar.

Tabla II
Resultados de *test* en GMS

Tipos de ventanas	Clasificadores			
	GB	RF	RL	AD
VF	*0.624*	*0.453*	*0.567*	*0.518*
VDCC	0.286	0.375	0.297	0.405
VDV	0.0	0.072	0.278	0.286
VF+VDCC	0.617	**0.492**	0.547	**0.540**
VF+VDV	**0.656**	0.355	0.533	0.467
VDCC+VDV	0.475	0.368	0.442	0.425
VF+VDCC+VDV	0.620	0.447	**0.591**	0.469

IV-C. Resultados

Las Tablas II y III incluyen todos los resultados obtenidos, para GMS y $AMAE$, respectivamente. Por filas, se muestran las combinaciones de ventanas (ventana fija, VF, ventana dinámica basada en cambio de clase, VCCC, y ventana dinámica basada en varianza, VDV, más todas las combinaciones posibles, VF+VDCC, VF+VDV, VDCC+VDV y VF+VDCC+VDV) y por columnas cada uno de los 4

Tabla III
Resultados de *test* en $AMAE$

Tipos de ventanas	Clasificadores			
	GB	RF	RL	AD
VF	0.317	**0.373**	*0.314*	*0.437*
VDCC	0.538	0.592	0.419	0.660
VDV	*0.312*	0.406	0.333	0.442
VF+VDCC	0.314	0.385	0.318	**0.426**
VF+VDV	**0.304**	0.396	0.323	0.457
VDCC+VDV	0.349	0.412	**0.306**	0.474
VF+VDCC+VDV	*0.312*	*0.384*	0.316	0.461

clasificadores considerados. Los mejores resultados por cada clasificador se muestran en negrita, los segundos mejores en cursiva. El mejor resultado global se encuentra doblemente subrayado, mientras que el segundo mejor está marcado con subrayado simple.

Atendiendo a dichas tablas, se pueden obtener diversas conclusiones:

- En términos de *GMS*, los mejores resultados para cada modelo se obtienen siempre mediante un preprocesamiento en el que intervienen dos o más ventanas. En términos de *AMAE*, a excepción del modelo *RandomForest*, ocurre lo mismo, lo que demuestra que el uso combinado de distintas ventanas mejora ampliamente los resultados obtenidos, en comparación con los obtenidos mediante el uso individual de los mismos.
- Al optimizar el *AMAE*, conviene utilizar la combinación de ventanas que produce el segundo mejor resultado, debido a que el tamaño del patrón que genera el uso de VDCC+VDV será mucho más pequeño que usar VF+VDV. Esto se debe a que el uso de ventanas dinámicas resume las muestras de las ventanas en dos métricas (media y varianza), haciendo que como máximo los patrones cuenten con 20 características. Atendiendo al *grid* de parámetros utilizados para ventana fija, los patrones generados con este método junto con la ventana dinámica pueden ser de hasta 40 características, ralentizando bastante el proceso de entrenamiento.
- La combinación VF+VDV consigue los mejores resultados, tanto en *GMS* como *AMAE*, mediante el uso del clasificador *GradientBoosting*. La ventana basada en varianza provee de una capacidad dinámica en el análisis de las series independientes, adaptándose a cada serie de forma individual, lo que combinado con una ventana fija que utilice muestras previas devuelva el mejor resultado.
- La VF siempre obtiene los segundos mejores resultados en *GMS*. Este aspecto era esperable, dado que existe una alta persistencia en la serie categórica dependiente a predecir. Así, en la optimización del tamaño de ventana fija, se incluye la posibilidad de crear ventanas de tamaño 1, haciendo que el modelo tienda a predecir la salida únicamente en función de un instante pasado. Aunque esto devuelva unos resultados aceptables, no es deseable, dado que este tipo de modelos nunca serán capaces de detectar un cambio de clase en la serie.
- El uso de un preprocesamiento VDV aislado devuelve unos resultados pobres. Esto es debido a que el uso de este tipo de ventanas está orientado a series reales, y al tratar con una serie dependiente de naturaleza categórica, la serie a predecir se modela únicamente en función de series independientes, perdiendo una parte importante de la información.
- Los resultados de *GMS* obtenidos con el uso de VDCC aislado, a pesar de utilizar solamente los valores de las series independientes, son mejores que los de VDV. La razón de este resultado es que, a pesar de que VDCC no usa los valores de la serie dependiente de forma directa, son éstos los que determinan el tamaño de ventana para cada muestra, por lo que de forma indirecta se está utilizando una información inherente a la serie dependiente.

Por último, la Figura 2 muestra una comparación de un fragmento de 96 horas de la serie real y la serie predicha por *GradientBoosting* con la mejor combinación de ventanas. Como puede observarse, el modelo funciona de forma aceptable, prediciendo correctamente los cambios producidos en las etiquetas reales. Tan solo en la última parte del gráfico se observan algunos artificios introducidos por el método.

V. Conclusiones

Este artículo evalúa el uso de diferentes clasificadores de aprendizaje automático para predecir eventos de baja visibilidad por niebla en el aeropuerto de Valladolid. Se realiza una predicción categórica basada en tres posibles tipos de situaciones (no niebla, neblina y niebla), utilizando como entrada valores pasados de un conjunto de variables meteorológicas medidas en el aeropuerto. Proponemos considerar distintos tipos de ventanas a la hora de analizar los valores pasados: ventanas fijas junto con dos tipos de ventanas dinámicas (una basada en cambios de la categoría de los días pasados analizados y otra basada en varianza de la variable independiente examinada). La ventaja fundamental de estos métodos dinámicos es que evitan prefijar el tamaño de la ventana, pudiéndose adaptar de forma dinámica a la serie temporal considerada.

Los resultados obtenidos indican que la combinación de ventanas fijas y dinámicas (especialmente aquella basada en varianza), junto con el conjunto de clasificadores *GradientBoosting* obtiene los mejores resultados, con valores de $AMAE$ cercanos a 0,3 (es decir, la predicción difiere en 0,3 categorías, en media, con respecto al valor real) y un GMS mayor que el $65\,\%$. Teniendo en cuenta estos resultados, se puede concluir que los modelos obtenidos pueden mejorar la seguridad y la eficiencia de las operaciones aeronáuticas que se llevan a cabo en aeropuertos bajo condiciones de baja visibilidad por niebla. Como trabajo futuro, planteamos el uso de clasificadores ordinales, dada la naturaleza ordinal de la clases consideradas.

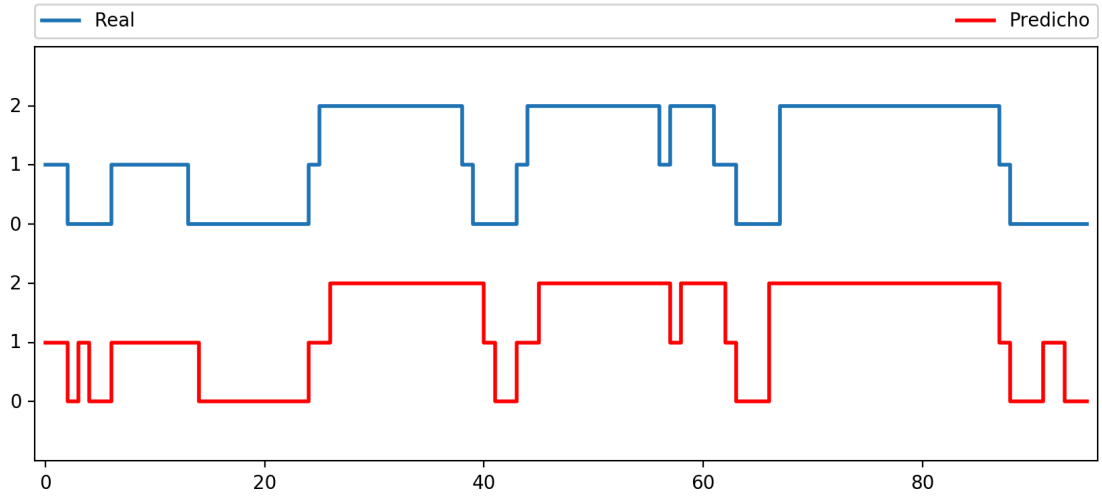


Figura 2. Comparación de etiquetas reales con las predichas por *GradientBoosting* con VF +VDV durante 96 horas. La clase 1 corresponde a no niebla, la clase 2 a neblina y la clase 3 a niebla.

Referencias

[1] D. Koracin, C. E. Dorman, J. M. Lewis, J. G. Hudson, E. M. Wilcox and A. Torregrosa, "Marine fog: A review," Atmospheric Research, vol. 143, pp. 142-175, 2014.

[2] I. Gultepe, et al., "Fog Research: A Review of Past Achievements and Future Perspectives," Pure and Applied Geophysics, vol. 164, pp. 1121-1159, 2007.

[3] H. Huang and C. Chen, "Climatological aspects of dense fog at Urumqi Diwopu International Airport and its impacts on flight on-time performance," Natural Hazards, vol. 81, pp. 1091-1106, 2016.

[4] N. Fedorova et al. "Fog Events at Maceio Airport on the Northern Coast of Brazil During 2002–2005 and 2007," Pure and Applied Geophysics, vol. 172, pp. 2727-2749, 2015.

[5] L. Zhenling et al., "Novel forecasting model based on improved wavelet transform, informative feature selection, and hybrid support vector machine on wind power forecasting," Journal of Ambient Intelligence and Humanized Computing, pp. 1-13, 2018.

[6] G. Noradin, A. Adel, S. Hossein and A. Oveis, "A new prediction model based on multi-block forecast engine in smart grid," Journal of Ambient Intelligence and Humanized Computing, pp. 1-16, 2017.

[7] Miao, Y. and Potts, R. and Huang, X., and Elliott, G. and Rivett, R., "A Fuzzy Logic Fog Forecasting Model for Perth Airport," Pure and Applied Geophysics, vol. 169, pp. 1107-1119, 2012.

[8] D. Duta and S. Chaudhuri, "Nowcasting visibility during wintertime fog over the airport of a metropolis of India: decision tree algorithm and artificial neural network approach," Natural Hazards, vol. 75, pp. 1349-1368, 2015.

[9] L. Cornejo-Bueno, C. Casanova-Mateo, J. Sanz-Justo, E. Cerro-Prada and S. Salcedo-Sanz, "Efficient Prediction of Low-Visibility Events at Airports Using Machine-Learning Regression," Boundary-Layer Meteorology, vol. 165, pp. 349-370, 2017.

[10] J. Yan-jie, G. Liang-peng, C. Xiao-shi and G. Wei-hong, "Strategies for multi-step-ahead available parking spaces forecasting based on wavelet transform," Journal of Central South University, vol. 24, pp. 1503-1512, 2017.

[11] J. P. Chao-Ying, L. L. Kuk and M. I. Gary, "An Introduction to Logistic Regression Analysis and Reporting," The Journal of Educational Research, vol. 96, pp. 3–14, 2002.

[12] L. Wei-Yin, "Fifty Years of Classification and Regression Trees," International Statistical Review, pp. 329-348, 2002.

[13] L. Breiman, "Random forests," Machine Learning, vol. 45(1), pp. 5-32, 2001.

[14] J. F. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," The Annals of Statistics, vol. 29, pp. 1189-1232, 2001.

[15] F. Pedregosa, "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

Intelligent Management of Measurement Units Equivalences in Food Databases*

*Note: The full contents of this paper have been published in the volume *Lecture Notes in Artificial Intelligence 11160* (LNAI 11160)

Beatriz Sevilla-Villanueva, Karina Gibert
Department of Statistics and Operations Research
Universitat Politècnica de Catalunya-BarcelonaTech (UPC)
Barcelona, Spain

Miquel Sànchez-Marrè
Department of Computer Science
Universitat Politècnica de Catalunya-BarcelonaTech (UPC)
Barcelona, Spain

***Abstract*—It is currently well-known that diet plays an important role in the promotion of healthy lifestyle and the prevention of chronic diseases. The Diet4You project is conceived to support the creation of an intelligent decision support system that provides personalized menus fitting a nutritional plan and taking into account the characteristics, needs and preferences of the person. The system involves a background food database, recording a collection of foods and prepared dishes with their standard portions as well as their nutritional decomposition in different food families.This DB is used to search the best combination of dishes approaching the total intake of different nutrients specified in the prescribed nutritional plan. The available background databases, specify the quantities of standard portions of several foods based on different measurement units which are not standardized, and it happens that the weight specified by one cup of melon is different from that of one cup of berries, among others. This arises the need of applying variable conversion factors to the dish description, before assessing whereas the total quantities of a certain menu fit well to the prescription. In this paper, a knowledge based approach is presented to the automatically management. An annotated reference food ontology is built on the basis of additional documentation. However the granularity of the information provided is heterogeneous and non exhaustive. The ontology-based missing values imputation is presented to overcome this limitations.**

***Index Terms*—Ontology, Missing Imputation, Database**

Predicción de delincuencia con datos públicos

1st Roberto Cuesta Calvo
Servicios Técnicos. Servicio Informática
Dirección General Guardia Civil
Madrid, España
rcuesta@guardiacivil.es

2nd Jesús Maudes Raedo
Departamento de Ingeniería Civil
Universidad de Burgos
Burgos, España
jmaudes@ubu.es

3rd José-Francisco, Díez-Pastor
Departamento de Ingeniería Civil
Universidad de Burgos
Burgos, España
jfdpastor@ubu.es

4th Ivan Arjona
Departamento de Ingeniería Civil
Universidad de Burgos
Burgos, España
email iaa0037@alu.ubu.es

***Resumen*—La concentración de recursos policiales en lugares considerados conflictivos contribuye a la reducción de la criminalidad en los mismos y a la optimización de esos recursos. En este artículo se presenta la utilización de técnicas de regresión para predecir el número de hechos delictivos en los municipios españoles. Para ello, se ha generado un conjunto de datos que fusiona los datos de la Guardia Civil con datos públicos sobre la estructura demográfica y las tendencias de voto en los municipios. El mejor regresor obtenido (i.e., Random Forests) alcanza con estos datos un RRSE (ráiz del error cuadrático relativo) del 41,23 %, y abre el camino para seguir incorporando datos públicos de otro tipo que tengan un mayor poder predictivo. Asimismo, se han utilizado reglas M5Rules para interpretar en lo posible los resultados.**

***Index Terms*—datos públicos, minería de datos, predicción de hechos**

I. Introducción

Para toda Fuerza y Cuerpo de Seguridad del Estado encargado de velar por los derechos y libertades de los ciudadanos se establece el concepto de territorialidad como un concepto clave en la búsqueda de maximizar la eficacia de sus recursos.

Es por ello que es constante el estudio estadístico de la criminalidad para optimizar la disposición de la fuerza sobre el terreno. Este estudio, constante en el tiempo, se acrecenta, aún mucho más, en épocas de crisis económica, pues es aquí donde existe una tendencia marcada en reducir recursos precisamente cuando, por diferentes y obvios motivos, existe riesgo de aumentar la criminalidad.

Si se deja a un lado la investigación en ciberdelincuencia y fraudes, la utilización de técnicas de aprendizaje automático en la predicción de delitos comunes ha sido muy escasa. Sólo en los últimos tiempos empiezan a aparecer algunos trabajos prometedores en esta línea. El enfoque del presente trabajo consiste en relacionar datos demográficos y de tendencia de voto en los municipios españoles, con los hechos delictivos cometidos durante un año; para así predecir la criminalidad de los municipios en función de dichas variables.

Trabajo parcialmente financiado por el proyecto TIN2015-67534-P del Ministerio de Economía Industria y Competitividad.

En este sentido, el estudio que se asemeja más al presentado en este artículo es quizás el de Alves y otros [1], que aplican técnicas de regresión con Random Forests [5] para predecir la cantidad de homicidios urbanos en Brasil a través de los datos sociológicos y demográficos de las ciudades. Las predicciones alcanzan un coeficiente de determinación de 0.97. Los autores apuntan al desempleo y el analfabetismo como las principales variables que utiliza su modelo predictivo.

Existen otros estudios en ámbitos geográficos más reducidos. En [17] se estudian cuales son las zonas conflictivas de un distrito policial al objeto de planificar la acción de las patrullas de calle. Se trabaja con los datos de Los Angeles (EEUU) y Kent (GB), y utilizan una aproximación basada en series temporales para estudiar la evolución de esas zonas.

En [15] se utilizan SVMs para predecir si una zona es conflictiva o no, en Columbus (Ohio) y St. Luis (Missouri).

En [7] se analizan las zonas de riesgo de crímenes sexuales en el campus de Charlottesville de la Universidad de Virginia. Se utilizan los clasificadores regresión logística y Random Forests [5] para clasificar un punto como conflictivo o no. Establece como variables mas importantes la proximidad de personas con antecedentes en violencia sexual y de residencias con fraternidades estudiantiles. También utilizan una aproximación de series temporales para analizar el intervalo horario, dia de la semana y época del año con mayor riesgo; así como la influencia de las condiciones meteorológicas, hallando la temperatura como factor climatológico más determinante. Utilizan *Kernel Density Estimation* (KDE) para comparar las probabilidades de riesgo en los distintos periodos de tiempo.

También se han utilizado técnicas de *Deep Learning* en la predicción de crímenes [14]. En este caso, las redes profundas predicen localizaciones y momentos probables de criminalidad en la ciudad de Chicago. Este estudio fusiona datos socioeconómicos con los datos policiales, además de datos climáticos. Otro trabajo en la misma línea para la ciudad de Manila es el de Báculo y otros [3]; en esta ocasión son las Redes Bayesianas el algoritmo de clasificación de entre los testados que mejores resultados ofrece.

El estudio que se presenta en el presente trabajo abarca todo tipo de delitos en la geografía Española. A diferencia de buena parte de los trabajos anteriormente revisados, no considera la evolución temporal de los hechos delictivos; ya que se centra en predicciones para todo el año; y además del uso de datos demográficos, presenta como novedad el uso de datos de preferencias políticas obtenidos a partir de los resultados electorales. Los resultados obtenidos son un primer paso dentro un proyecto que pretende integrar más datos públicos para mejorar las predicciones, pero que en el estado actual ya ofrece unos resultados interesantes.

El artículo se estructura como sigue; en la sección II se describen los datos y su procedencia, en la sección III se describen los experimentos con distintas técnicas de regresión para predecir los hechos delictivos, la sección IV trata de interpretar los resultados obtenidos, y finalmente en V se muestran las conclusiones y líneas futuras.

II. Obtención y descripción de los datos

Para este trabajo se han cruzado datos públicos de organismos oficiales con estadísticas de la Secretaría de Estado de seguridad a través de la Dirección General de la Guardia Civil.

Se entiende como datos públicos o abiertos aquellos que deben estar disponibles de manera libre, para acceder, utilizar, modificar y publicar sin restricciones de *copyright* [19].

Este trabajo se aprovecha de propuestas como la 'Iniciativa Aporta' [2] que promueve la apertura de información en el sector público en España. Esta iniciativa tiene el objetivo de favorecer el desarrollo de la reutilización de la información del sector público y ayudar a las administraciones para que publiquen sus datos de acuerdo al marco legislativo vigente.

Los gobiernos tienen la capacidad de obtener grandes cantidad de información sobre la población a través de varios organismos (como podría ser el *Instituto Nacional de Estadística*).

En este trabajo cada instancia del conjunto de datos representa un municipio. El número de registros del conjunto es 8.125, con un total de 124 atributos sin contar la clase (i.e., número de hechos delictivos). Se han utilizado dos fuentes públicas:

1. Instituto Nacional de Estadística (INE). Estadísticas del año 2016, correspondientes a lugar de nacimiento y rangos de edad. Un total de 114 atributos. Lugar de nacimiento (51 atributos), Rangos de edad (63 atributos).
2. Ministerio de Interior. Datos electorales, elecciones al Congreso (Junio 2016). Un total de 10 atributos.

Los atributos de lugar de nacimiento se distribuyen en 17 categorías, cada una de ellas desglosada en 3 sub-categorías (mujeres y hombres, solo mujeres, solo hombres), las categorías son: 1) Total, 2) Españoles, 3) Nacidos en España, 4) En la misma Comunidad Autónoma, 5) Misma Comunidad Autónoma. Misma Provincia, 6) Misma Comunidad Autónoma. Misma Provincia. Mismo Municipio, 7) Misma Comunidad Autónoma.Misma Provincia. Distinto Municipio, 8) Misma Comunidad Autónoma. Distinta Provincia, 9) En distinta Comunidad Autónoma, 10) Nacidos en el Extranjero, 11) Nacionalidad extranjera, 12) Europa, 13) Unión Europea, 14) África, 15) América, 16) Asia, 17) Oceanía, Apátridas y Resto.

Los atributos de rangos de edad se distribuyen en 21 categorías (0-4 años, 5-9 años, ..., 90-94 años, 95-99 años, más de 100 años) cada una de ellas desglosada en 3 sub-categorías (mujeres y hombres, solo mujeres, solo hombres).

El número de votos de cada formación se ha agrupado en 10 categorías (Extrema Izquierda, Izquierda, Centro Izquierda, Centro, Centro Derecha, Derecha, Extrema Derecha, Otros, En blanco y Nulos.). En búsqueda optimizar la objetividad de las conclusiones finales estas categorías fueron seleccionadas y categorizadas por fuentes abiertas y externas al personal de este estudio, con el objeto de no introducir subjetividades que pudiesen introducir errores o inducir a conclusiones erróneas.

En cuanto a la clase, en los 8.125 municipios evaluados se registraron un total de 36.806.873 hechos, de los cuales, los más comunes fueron: Delito de hurto (11,5 %), Infracción por el consumo o la tenencia de drogas en lugares públicos (9,4 %), Robo con fuerza (8,8 %), Infracciones al reglamento de vehículos (5,4 %), Infracciones al reglamento de circulación (3,9 %) y Alcoholemia (3,7 %).

Es importante recalcar la naturaleza pública y externa a la Guardia Civil de los datos utilizados, y que dicha institución está, por tanto, al margen de cómo se han categorizado los mismos tanto en general, como en particular en lo concerniente a cómo se han agrupado los partidos políticos y a cómo se han agrupado los inmigrantes por su procedencia.

III. Análisis de los datos con métodos de regresión

El conjunto de datos de la sección anterior sufrió dos transformaciones antes de ser utilizado. En primer lugar, se normalizaron todos los atributos, excepto la variable a predecir en el intervalo [0,1]. En segundo lugar, dado que la variable a predecir representa un recuento (i.e., *Nº de hechos delictivos*), se asume que sigue una distribución de Poisson, por lo que se ha aplicado la raíz cuadrada a dicha variable.

Una vez transformados los datos se procedió a experimentar en WEKA [12] mediante validación cruzada 10×10 diversas técnicas de regresión, para así conocer la más idónea. En principio, en todos los regresores se ha utilizado la configuración por defecto de WEKA, salvo en los casos en los que a continuación se indique lo contrario.

Los regresores utilizados en el experimento se agrupan en dos familias: por un lado regresores en solitario o *singletons*, y por otro multiregresores o *ensembles*.

Entre los *singletons* se probaron:

- Árbol de decisión M5P [20]. Los M5P son árboles de decisión de la familia de los *model trees*. Estos árboles contienen una regresión lineal en los nodos hoja.
- M5Rules [13], se trata de un método que obtiene reglas de decisión a partir de árboles M5P, por lo que no se espera que den unos resultados muy distintos que el propio M5P. La razón de incluirlos es justificar su fiabilidad cara a utilizarlos en la sección IV como herramienta para interpretar los resultados.

- Regresión Lineal, optimizando el parámetro *ridge* para cada una de las cuatro versiones que se probaron, y que surgieronn de activar/desactivar la selección de variables y la eliminación de atributos colineales. Se tomo como mejor versión la que no hacía selección de variables pero si eliminaba atributos colineales.
- SVM para regresión (SVM-Reg) [21] utilizando la implementación LIBLINEAR [9] con kernel lineal y el parámetro C optimizado.
- *k*-NN. Debido al elevado número de características en el conjunto de datos, se probaron dos versiones. La primera sin selección de atributos, la segunda con selección de atributos mediante *Correlation-based Feature Subset Selection* [11]. En ambas versiones se optimizó el número de vecinos. La mejor versión resultó ser la que no hace selección de atributos, y es la que se reporta en el artículo.

Los *ensembles* probados fueron los siguientes:

- Random Forest [5].
- AdaBoost.R2 [8], se han probado tres configuraciones con función de pérdida lineal, cuadrática y exponencial. Se seleccionó la configuración con mejores resultados en la validación cruzada 10×10 (i.e., pérdida cuadrática)
- Additive Regression, que es una implementación de *Stochastic Gradient Boosting* [10]
- Bagging [4]
- Iterated Bagging [6]. En este caso, para simular 100 árboles se utilizan 20 iteraciones Bagging de 5 árboles cada una.

Todos los *ensembles* utilizan 100 árboles M5P como regresores base, excepto Random Forest, que obviamente utiliza 100 Random Trees.

La métrica utilizada para evaluar los regresores es el RRSE o ráiz del error cuadrático relativo, que para θ_i el valor verdadero a estimar para la instancia $i-$ésima, $\hat{\theta}_i$ el valor resultado de la estimación de esa instancia, y $\bar{\theta}$ el valor medio de las θ_i, estimado a través de las instancias del conjunto de entrenamiento, se define como:

$$RRSE = \sqrt{\frac{\sum_{i=1}^{N}(\hat{\theta}_i - \theta_i)^2}{\sum_{i=1}^{N}(\bar{\theta} - \theta_i)^2}}$$

Los resultados se muestran en el Cuadro I. Como se puede apreciar, los *ensembles* están a la cabeza, mientras que los métodos lineales quedan en la parte inferior. Esto puede deberse a que, como se indica en [16] la relación entre el tamaño de las ciudades y su criminalidad obedece mas a una ley potencial que a una relación lineal. De hecho, los coeficientes de correlación en la tabla muestran una correlación muy discreta cuando se utilizan modelos lineales, que mejoran considerablemente en el caso de los árboles M5P y las reglas M5Rules, los cuales también usan modelos lineales en las hojas, y alcanzan un valor aproximado de 0.9 *ensembles* utilizados.

El mejor método, tanto por RRSE, como por coeficiente de correlación, es *Random Forests*. El RRSE alcanzado, 41.23,

Método	RRSE	Coef. Corr.
Random Forests	41.23	0.91
AdaBoost R2	41.63	0.91
Bagging	42.20	0.91
Iterated Bagging	43.42	0.90
Additive Regression	44.62	0.90
M5P	47.10	0.88
M5Rules	49.41	0.87
k-NN	52.27•	0.85•
SVM-Reg	74.59•	0.67•
Regresión Lineal	82.15•	0.62•

Cuadro I
RESULTADOS DE LOS DIFERENTES MÉTODOS ORDENADOS POR RRSE. LOS • INDICAN DIFERENCIAS SIGNIFICATIVAS CON EL MEJOR MÉTODO.

parece mostrar que los datos demográficos y de intención de voto sirven para explican aceptablemente la concentración de hechos delictivos, pero quizás aún hay margen de mejora incorporando en el futuro nuevas variables al modelo.

En la tabla se ha marcado con • aquellos valores que son estadísticamente peores, con un nivel de confianza del 95 %, al compararlos con el mejor método. El test estadístico utilizado es el *corrected resampled t*-test [18], debido a su idoneidad en el caso de utilizar validación cruzada. Se aprecia que no hay diferencias entre los métodos del grupo de los *ensembles*.

IV. Interpretación de los Resultados del Análisis y Líneas de Mejora

Para descubrir las posibles líneas de mejora del modelo actual se han seguido dos caminos:

- Investigar los municipios en los que peor se comporta el modelo.
- Generar reglas con el algoritmo *M5Rules* [13], para poder interpretar el conjunto de datos.

IV-A. Municipios que peor responden al modelo

Para tener una lista ordenada de los municipios que peor responden al modelo se halló el valor absoluto de la diferencia entre el número de hechos real y el número de hechos predichos por un *Random Forest* entrenado con todos los datos del conjunto. Cierto es que esta diferencia arroja unos valores muy optimistas, en tanto las diferencias se obtienen a partir de predicciones sobre los propios datos de entrenamiento, pero se asume que esa ventaja la van a tener todos los municipios. Una vez obtenida esa diferencia en valor absoluto, se divide por el número de habitantes del municipio, para evitar que el indicador únicamente señale a los municipios más grandes. Denotaremos a este indicador como Δ/hab.

El valor máximo de Δ/hab es del 29,99 %, y se alcanza en un municipio de 230 habitantes. Hay otro pequeño municipio de 20 habitantes que alcanza un 29,75 %, otro de 105 con un 24 %, y a partir de ahí una lista de 35 pequeños municipios, el mayor de ellos con 303 habitantes, hasta llegar a *Sant Josep de sa Talaia* con 25.849 habitantes y un Δ/hab =7,32 %.

En estas localidades tan pequeñas, cuando ocurren unos pocos hechos delictivos por encima de los previstos, el indicador Δ/hab se dispara.

Es llamativo que en esta lista ordenada por Δ/hab hay una serie de municipios de más de 10.000 habitantes intercalados con estos pequeños municipios. El Cuadro II muestra los que tienen un Δ/hab por encima del 2 %.

Municipio	Nº hab	Δ/hab	Hechos	Predicción	pos
S. Josep de sa Talaia	25.849	7,32 %	5.578	3.685	38
Calvià	49.580	4,79 %	6.281	3.907	80
Torrevieja	84.213	3,18 %	7.312	4.635	149
S. Antony de Portmany	24.478	2,96 %	3.260	2.536	177
Borriana	34.643	2,29 %	2.347	1.556	254
Las Rozas de Madrid	94.471	2,16 %	4.160	2.126	276
Benicasim	17.957	2,03 %	1.356	991	301
Guardamar del Segura	15.386	2,02 %	1.280	969	304

Cuadro II
MUNICIPIOS CON MÁS DE 10.000 HABITANTES Y $\Delta/hab > 2\,\%$.
POS=POSICIÓN EN EL RANKING.

Estas localidades podrían estar siendo predichas mal debido a que en su mayoría son conocidas plazas turísticas, y su tamaño real, teniendo en cuenta los turistas, seguramente difiera mucho del tamaño por habitantes empadronados. De hecho, todas las predicciones son siempre a la baja. No obstante, los valores de Δ/hab son bastantes moderados.

Por tanto, se aprecia que una línea de mejora a futuro podría venir por incorporar al conjunto de datos características nuevas que cuantifiquen el fenómeno turístico en los municipios.

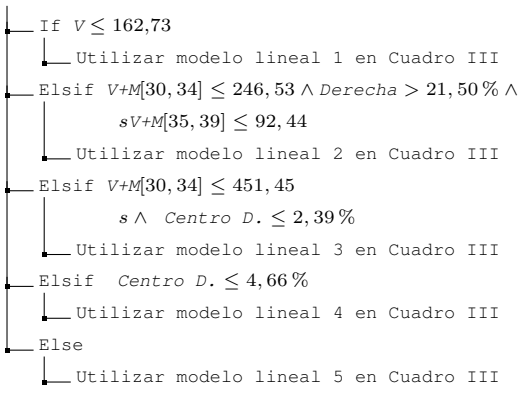
IV-B. Interpretación con M5Rules

Se ha utilizado para interpretar el conjunto de datos la generación de reglas mediante el algoritmo *M5Rules* [13]. *M5Rules* genera un árbol M5P, selecciona la mejor hoja, y haciendo AND con los nodos en el recorrido entre la raíz y dicha hoja, genera una primera regla. Después, descarta del árbol los datos que han caído en esa hoja, y vuelve a construir un nuevo M5P repitiendo todo el proceso para generar una segunda regla, y así sucesivamente va generando nuevos árboles por cada regla con los datos de entrenamiento que no son cubiertos por las reglas anteriores, hasta que finalmente a toda instancia la corresponda una regla.

La implementación WEKA de *M5Rules*, en la configuración por defecto del algoritmo, incluye un proceso de poda, de manera que elimina las reglas finales, que tienden a concentrarse en casos particulares, sustituyéndolas por un modelo lineal.

En el cuadro I de la sección I podía verse que no hay diferencias significativas en términos de *RRSE* entre el mejor método y *M5Rules*, por lo que parece una aproximación aceptable para tener una cierta interpretación de los datos desde la óptica de los árboles de decisión.

Para ello, se utilizó el conjunto de entrenamiento al completo y se generaron las siguientes cinco reglas:



V representa el número total de varones del municipio, *V*[x,y] representa el número de varones en el rango de edades [x,y], *V*+*M*[x,y] representa la suma de varones y mujeres en el rango de edades [x,y], *Derecha* y *Centro D* es el porcentaje de votos a formaciones de derecha y de centro derecha respectivamente. La última regla (i.e., *ELSE*) no contiene una condición lógica por que es la que proviene de la poda de las reglas menos importantes, ya comentada anteriormente.

Nótese que las reglas de la ilustración no son las originales generadas por el algoritmo, el cual, como ya se ha indicado trabaja con datos normalizados en el intervalo [0,1]. En su lugar, y para facilitar su comprensión, esas reglas se han traducido con los valores correspondientes a los datos sin normalizar.

Se aprecia que las expresiones lógicas en las cinco reglas toman como variables los rangos de edad, prestando mucha atención a los rangos en torno a 30-39 años para caracterizar los cinco grupos de municipios. Es probable que el modelo esté tomando esas edades para caracterizar el tamaño de los municipios. Como novedad importante frente a otros trabajos relacionados, la orientación del voto también ha sido tenida en cuenta. Por el contrario, el origen de la población (e.g., extranjera, nacida en el mismo municipio, etc...) no ha sido utilizada.

Los cinco grupos generados por las reglas, se describen en el Cuadro IV, mientras que sus respectivos modelos lineales están en el Cuadro III. En dicho cuadro se ha incluido una columna *Peso* que se calcula a partir de los valores absolutos de los coeficientes, de manera que representa el cociente entre el valor absoluto del coeficiente de ese atributo, dividido por la suma de los valores absolutos de todos los coeficientes. Una vez calculado el peso, los atributos se ordenan por el mismo descendentemente. El cuadro solo muestra los de mayor absoluto, concretamente los necesarios para que su suma supere el umbral del 33,34 % del peso total.

Los modelos lineales de este cuadro son confusos y no arrojan conclusiones sobre la influencia de un determinado colectivo en la aparición de hechos delictivos. Esto se debe principalmente a las relaciones de inclusión que existen entre gran parte de los atributos. Es decir, algunos atributos representan colectivos que están incluidos dentro de colectivos representados por otros atributos. Por ejemplo, en el modelo

lineal número 1, aparentemente la variable más importante es el número de extranjeros americanos, que contribuye a la aparición de hechos delictivos con signo positivo y un peso del $8,01\,\%$. Sin embargo, el coeficiente con tercer mayor peso son las mujeres de ese colectivo, que contribuyen negativamente con un peso del $4{,}70\,\%$, mientras que los varones de ese colectivo están en undécimo lugar con un peso, también negativo del $2,93\,\%4$, de manera que unos coeficientes están contrarrestando el peso de otros, y dado que el modelo está representando los coeficientes para predecir la raíz cuadrada de los hechos, no es directo establecer la contribución neta de los tres coeficientes. Hay más relaciones de inclusión, por ejemplo, en el modelo 2, los varones y mujeres de 30 a 34 años son un subconjunto de los varones y mujeres, y a la vez es superconjunto de los varones en ese rango de edad, etc ...

Modelo Lineal 1		
Coef.	Variable	Peso
+86.053,06	Americanos	8,01 %
-66.181,54	Varones+mujeres de 0 a 4 años	6,16 %
-50.466,39	Mujeres de América	4,70 %
-48.402,00	Varones+mujeres de 30 a 34 años	4,50 %
+40.735,51	Varones+mujeres de 40 a 44 años	3,79 %
+34.253,68	Varones+mujeres de 60 a 64 años	3,19 %
-33.294,81	Varones	3,10 %
+0,61	Término independiente	

Modelo Lineal 2		
Coef.	Variable	Peso
+31.480,21	Varones+mujeres de 30 a 34 años	16,76 %
-29.679,75	Varones+mujeres de 0 a 4 años	15,80 %
-15.372,52	Varones de 30 a 34 años	8,18 %
+2,07	Término independiente	

Modelo Lineal 3		
Coef.	Variable	Peso
-14.576,13	Extranjeros	21,68 %
+9.533,03	Mujeres extranjeras	14,18 %
+3.10	Término independiente	

Modelo Lineal 4		
Coef.	Variable	Peso
-295,28	Varones de 70 a 74 años	11,48 %
-250.56	Varones de 90 a 94 años	9,74 %
+247.15	Varones nacidos en el extranjero	9,61 %
-246.47	Mujeres asiáticas	9,58 %
+10,87	Término independiente	

Modelo Lineal 5		
Coef.	Variable	Peso
-120.03	Varones de 75 a 79 años	9,62 %
+118.35	Varones de 85 a 89 años	9,49 %
+110.85	Varones y mujeres de 80 a 84 años	8,88 %
-106.74	Mujeres nacidas en el extranjero	8,56 %
+0.126	Término independiente	

Cuadro III

MODELOS LINEALES OBTENIDOS PARA EL CONJUNTO DE REGLAS *M5Rules*. LA COLUMNA PESO REPRESENTA EL PESO DEL VALOR ABSOLUTO DE ESE COEFICIENTE EN EL MODELO LINEAL. SE MUESTRAN SOLO LOS DE MAYOR PESO.

Aunque los modelos lineales no parecen arrojar ninguna conclusión plausible, el análisis de los grupos que generan las reglas si que es algo más revelador. En el Cuadro IV se han incluido el mínimo, máximo, promedio y desviación típica de la población, número de hechos delictivos y Δ/hab para cada uno de los grupos de municipios definidos por las cinco reglas. Asimismo, la Figura 1 muestra también para los cinco grupos cómo se distribuyen los hechos delictivos frente al tamaño de los municipios. Los colores en la figura representan valores Δ/hab altos a medida que toman valores más claros.

Parece que los grupos correspondientes a las cuatro primeras reglas mantienen una cierta similitud en que simplemente constatan que a medida que aumenta el tamaño del municipio aumenta el número de hechos delictivos. En el grupo de la regla 1 llama la atención el máximo número de hechos (146 hechos en Escorca, Mallorca), que se corresponde con el municipio con el Δ/hab máximo que ya se comentó en la sección IV-A. Todos los demás municipios de R1 excepto éste están, sin embargo, por debajo de los 45 hechos.

El ratio Δ/hab es algo mas grande para la regla R1 (promedio de $0,74\,\%$), debido a que en los municipios pequeños se penaliza mucho una leve variación en unos pocos delitos, pero en los grupos R2 a R4 se estabiliza en torno a $0,30\,\%-0,38\,\%$.

El grupo correspondiente a la regla R5, sin embargo, es diferente a los demás. A pesar de englobar municipios grandes y muy grandes, mantiene un promedio de hechos delictivos muy bajo (3,22). Un análisis identificativo de dichos municipios nos revela que son todos municipios de Cataluña y País Vasco, donde hay transferidas muchas competencias a las policías autonómicas, y donde por tanto, el número de denuncias que tramita la Guardia Civil tiende a ser marginal.

Por tanto, podemos apuntar como debilidad del modelo predictivo que no predice los hechos delictivos en si, sino únicamente los denunciados a la Guardia Civil, como por otro lado es lógico, ya que son las denuncias que se han utilizado en el estudio. Por tanto, otra línea de mejora es la incorporación de datos de las policias autonómicas.

	R1	R2	R3	R4	R5
Municipios	3.118	1.846	1.437	830	894
Min Habs	5	258	299	4.027	291
Max Habs	341	2.036	8.483	3.165.541	1.608.746
Prom Habs	133,5	706,0	3.014,2	38.194,6	9.848,5
Dev Habs	79,2	342,0	1.574,4	127.095,5	58.499,7
Min Hechos	0	0	0	0	0
Max Hechos	146	85	459	10.025	792
Prom Hechos	2,55	14,32	70,06	542,83	3,22
Dev Hechos	4,31	11,22	55,25	775,15	27,86
Min Δ/hab	0,00 %	0,00 %	0,00 %	0,00 %	0,00 %
Max Δ/hab	29,99 %	10,50 %	5,90 %	7,30 %	0,70 %
Max Δ/hab	33,70 %	37,10 %	33,70 %	7,90 %	1,30 %
Prom Δ/hab	0,74 %	0,38 %	0,30 %	0,37 %	0,03 %
Dev Δ/hab	1,68 %	0,52 %	0,39 %	0,50 %	0,06 %

Cuadro IV

DESCRIPCIÓN ÁREAS CUBIERTAS POR CADA REGLA. R_i =Nº DE REGLA EN EL ORDEN DE INTERPRETACIÓN *M5rules*, MIN-MAX-PROM Y DEV = MÍNIMO, MÁXIMO, PROMEDIO Y DESVIACIÓN ESTÁNDAR RESPECTIVAMENTE. HABS=Nº DE HABITANTES, HECHOS=Nº HECHOS DELICTIVOS POR MUNICIPIO, Δ/HAB =PORCENTAJE DE ERROR POR HABITANTE.

V. Conclusiones y líneas futuras

En el presente trabajo se ha obtenido un modelo predictivo basado en *Random Forests* que permite predecir el número

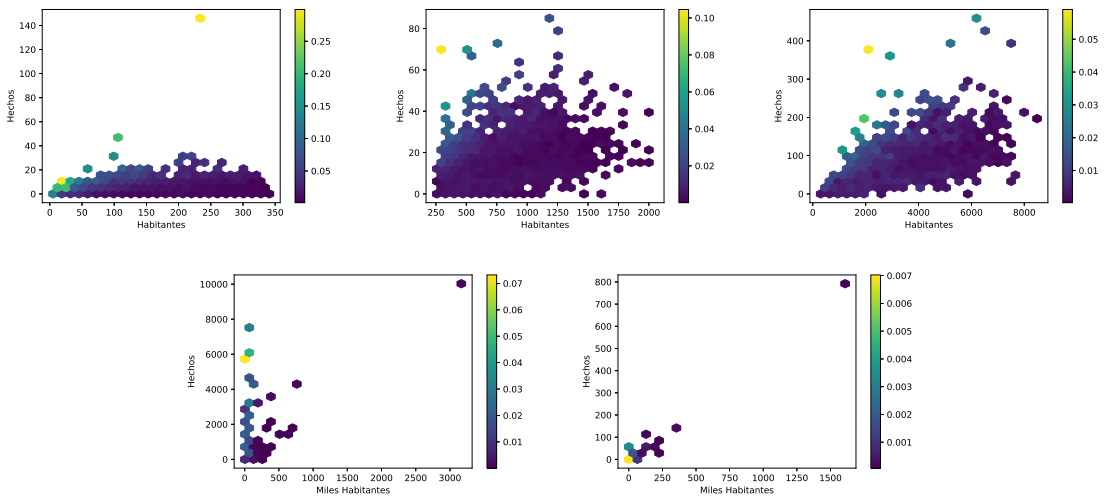


Figura 1. Un diagrama de *binning* hexagonal por cada uno de los subconjuntos resultantes de aplicar las reglas. En el eje y se muestran el número de habitantes, en el eje x el número de hechos y el color representa Δ/hab en la predicción en los municipios situados en esas coordenadas

de hechos delictivos que se denuncian a la Guardia Civil anualmente en cada municipio. El valor de RRSE alcanzado es de 41.23, pero en general todos los *ensembles* dan resultados que no son significativamente diferentes.

Los datos utilizados combinan la localización de las denuncias de la Guardia Civil con fuentes de datos públicas (i.e., INE y datos electorales de 2016). Una novedad que incorpora el presente trabajo es precisamente la incorporación de datos electorales.

Del análisis de los municipios en los que peor se comporta el modelo se deriva una posible mejora incorporando datos relativos a la ocupación turística, probablemente enfocados al turismo de playa.

Por otro lado, las reglas *M5Rules* han discriminado cinco grupos de municipios. Los cuatro primeros grupos parecen segmentar los municipios por su tamaño. El quinto grupo aglutina las denuncias en las comunidades autónomas en las que la policía autonómica ha sustituido a la Guardia Civil en muchas de sus competencias; por lo que otra línea de mejora cara identificar puntos calientes, es incorporar datos de denuncias de esos cuerpos policiales.

Referencias

[1] Luiz G.A. Alves, Haroldo V. Ribeiro, and Francisco A. Rodrigues. Crime prediction through urban metrics and statistical learning. *Physica A: Statistical Mechanics and its Applications*, 505:435 – 443, 2018.

[2] Iniciativa Aporta. Acerca de la iniciativa aporta. http://datos.gob.es/es/acerca-de-la-iniciativa-aporta. [Internet; descargado 16-mayo-2018].

[3] Maria Jeseca C. Baculo, Charlie S. Marzan, Remedios de Dios Bulos, and Conrado Ruiz. Geospatial-temporal analysis andclassification of criminal data in manila. In *Procs. of 2nd IEEE International Conference on Computational Intelligence and Applications*, pages 6–11. IEEE, 2017.

[4] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, Aug 1996.

[5] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[6] Leo Breiman. Using iterated bagging to debias regressions. *Machine Learning*, 45(3):261–277, Dec 2001.

[7] Elise Clougherty, John Clougherty, Xiaoqian Liu, and Donald Brown. Spatial and temporal analysis of sex crimes in charlottesville, virginia. In *Procs. of IEEE Systems and Infromation Engeneering Design Symposium*, pages 69–74. IEEE, 2015.

[8] Harris Drucker. Improving regressors using boosting techniques. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 107–115, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[9] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[10] Jerome H. Friedman. Stochastic gradient boosting. *Comput. Stat. Data Anal.*, 38(4):367–378, February 2002.

[11] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[12] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[13] Geoffrey Holmes, Mark Hall, and Eibe Frank. Generating rule sets from model trees. In *Proceedings of the 12th Australian Joint Conference on Artificial Intelligence: Advanced Topics in Artificial Intelligence*, AI '99, pages 1–12, London, UK, UK, 1999. Springer-Verlag.

[14] Hyeon-Woo Kang and Hang-Bong Kang. Prediction of crime occurrence from multimodal data using deep learning. *PLoS One*, 12(4):e0176244, 2017.

[15] Keivan Kianmehr and Reda Alhajj. Effectiveness of support vector machine for crime hot-spots prediction. *Applied Artificial Intelligence*, 22(5):433–458, 2008.

[16] J. C. Leitão, J. M. Miotto, M. Gerlach, and E. G. Altmann. Is this scaling nonlinear? *Royal Society Open Science*, 3(7), 2016.

[17] G. O. Mohler, M. B. Short, Sean Malinowski, Mark Johnson, G. E. Tita, Andrea L. Bertozzi, and P. J. Brantingham. Randomized controlled field trials of predictive policing. *Journal of the American Statistical Association*, 110(512):1399–1411, 2015.

[18] C. Nadeau and Y. Bengio. Inference for the generalization error. *Machine Learning*, 52(239–281), 2003.

[19] Comisión Económica para América Latina y el Caribe. ¿qué son los datos abiertos? https://biblioguias.cepal.org/EstadoAbierto/datospublicos. [Internet; descargado 16-mayo-2018].

[20] Ross J. Quinlan. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348, Singapore, 1992. World Scientific.

[21] Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, August 2004.

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 4:
METODOLOGÍAS

Diagnóstico de fallos mediante clasificadores: Análisis de robustez en ambientes de incertidumbre

JM. Bernal-de Lázaro
Dept. de Automática y Computación
Universidad Tecnológica de La Habana,
Habana, Cuba
jbernal@automatica.cujae.edu.cu

O. Llanes-Santiago
Dept. de Automática y Computación
Universidad Tecnológica de La Habana,
Habana, Cuba
orestes@tesla.cujae.edu.cu

A. Prieto-Moreno
Dept. de Automática y Computación
Universidad Tecnológica de La Habana,
Habana, Cuba
albprieto@automatica.cujae.edu.cu

A. Silva-Neto
Dept. de Ingeniería Mecánica
Instituto Politécnico do Rio de Janeiro
Rio de Janeiro, Brasil
ajsneto@iprj.uerj.br

C. Cruz Corona
Dept. de Ciencia de la Computación
Universidad de Granada
Granada, España
carloscruz@decsai.ugr.es

***Resumen*—La presencia de ruidos e interferencias es un fenómeno común en los procesos de transmisión y procesamiento de datos que se producen en ambientes industriales. Teóricamente, en ausencia de ruidos e interferencias, la información implícita en los datos trasmitidos puede ser mejor aprovechada. Sin embargo, la realidad es que el ruido resulta intrínseco a los sistemas eléctricos y entornos industriales, por lo que es recomendable considerar su efecto al trabajar con la información de los sensores en el proceso. En este contexto, el problema de la robustez para sistemas de diagnóstico de fallos puede ser definido como la capacidad de maximizar la detectabilidad y aislabilidad de los fallos, al mismo tiempo que se minimiza el efecto de perturbaciones, ruidos y cambios en los estados del sistema. El objetivo de este trabajo es estudiar los enfoques de diagnóstico de fallos, con énfasis en su robustez y aplicación en ambientes industriales ruidosos. Para ello, se propone un índice que permite evaluar el desempeño global de un diagnosticador en términos de su robustez durante su etapa de diseño. El índice propuesto complementa el error de clasificación mediante un factor de penalización que refleja la capacidad de rechazo al ruido por parte del diagnosticador. Para ejemplificar la utilidad del índice propuesto, se compara el desempeño de tres clasificadores: Árboles de Decisión (AD-ID3), Redes Neuronales Artificiales (RNA) y Máquinas de Soporte Vectorial (MSV), aplicados todos al diagnóstico de fallos en el Tanque Reactor Continuamente Agitado (CSTR).**

***Index Terms*—Diagnóstico de fallos; Clasificacin; Robustez**

I. Introducción

Durante los últimos 30 años, el diagnóstico de fallos (DF) como área de investigación, ha recibido una considerable atención [1], [2], [4], [5]. Estas publicaciones se centran en tres cuestiones primordiales dentro del diseño de los sistemas de diagnóstico de fallos, estas son: robustez, sensibilidad y rendimiento. De acuerdo con la definición dada por [9] y [10], el problema de la robustez en el diagnóstico de fallos puede entenderse como la capacidad de maximizar la detectabilidad y aislabilidad de los fallos, al mismo tiempo que se minimiza el efecto de perturbaciones, ruidos y cambios en las entradas/salidas, o estados del sistema. A partir de esta definición, se han desarrollado numerosos estudios relacionados con el análisis de la robustez en sistemas de diagnóstico que utilizan modelos matemáticos puros. Sin embargo, muy pocos estudios han investigado el impacto del ruido en los sistemas de diagnóstico basados en datos históricos. En este contexto, el tema de la robustez ha sido abordado mayormente desde el punto de vista de la insensibilidad del sistema de diagnóstico a datos fuera de rango (*outliers*), datos ausentes (*missing data*), y valores muy puntuales de ruido. Aunque se han hecho progresos considerables en este sentido, un problema persistente en el campo del diagnóstico de fallos basado en datos es que los estudios comparativos entre clasificadores no consideran el efecto del ruido o simplemente realizan un análisis de robustez local simulando valores constantes para el ruido en los datos. Por lo general, estas comparaciones se enfocan en una zona de trabajo donde se asume niveles ruido muy bajos que se consideran invariantes. En los procesos reales, sin embargo, es posible que la conexión/desconexión de equipos y diferentes fuentes de ruido externas modifiquen los datos obtenidos del proceso, introduciéndoles mayor variabilidad. Se requiere, por tanto, de un indicador que cuantifique el desempeño del sistema de diagnóstico en términos de insensibilidad ante este cambio y permita comparar el desempeño de diferentes herramientas de classificación empleadas en las tareas de diagnóstico de fallos. Una alternativa a este problema es el indicador de robustez aquí propuesto.

Para ello, la estructura de este trabajo es la siguiente. En la Sección 2 se discuten las consideraciones generales para el análisis de la robustez en sistemas de diagnóstico basados en datos. En la Sección 3 se realiza la propuesta de índicador de robustez, y se exponen sus aplicaciones potenciales. La aplicación del índice de robustez propuesto en el proceso de prueba Tanque Reactor Continuamente Agitado (CSTR) y el estudio comparativo de los clasificadores se realiza en la Sección 4. Por último, se emiten las conclusiones del trabajo.

II. Consideraciones generales

En el control de un proceso, una interferencia puede ser considerada como un tipo de perturbación externa generada por acoplamientos eléctricos y magnéticos (motores, equipos de alta potencia, etc), o debido a fenómenos naturales (tormentas, etc). Dado su origen conocido, el efecto de una interferencia periódica, intermitente, o aleatoria, puede ser minimizado por la sustitución de acoplamientos eléctricos y electromagnéticos; acciones que deben ir acompañadas además, del uso de conexiones apantalladas y protecciones eléctricas en las líneas de transmisión. Por otro lado, la contaminanción de señales debido al ruido es un concepto más general al considerar cualquier efecto aleatorio e impredecible (con acción temporal o constante), que distorsiona una señal original que es medida, transmitida y procesada. El incremento en la variabilidad de los datos transmitidos suele ser, por lo general, uno de los efectos más notables en las señales ruidosas. Cuando variabilidad incorporada por el ruido adiciona incertidumbre a la información contenida en una señal, es posible que la información original sea parcialmente enmascarada, modificada o imposible de identificar.

Un ruido puede afectar significativamente el desempeño de un diagnosticador incorporando variabilidad adicional en los datos, lo cual modifica negativamente las fronteras de decisión del clasificador [11]. Este fenómeno es fácil de entender si consideramos al clasificador como un algoritmo o función matemática que divide el espacio de características (síntomas), en tantas regiones como clases (fallos) existen [7]. Por ejemplo, considérese un espacio de características consistente en dos clases mutuamente excluyentes. La tarea del clasificador de diagnóstico es asignar una etiqueta de clase $\hat{y}_i$ a una nueva observación $x_i = \{\nu_1, \ldots, \nu_d\}$, dado $\hat{y}_i = f(x_i)$ con una etiqueta de clase predicha $\hat{y}_i \in \{c_1, c_2\}$ y ν_j variables medidas. El efecto del ruido en las mediciones puede hacer aparentemente similar el comportamiento de dos clases distintas, resultando en una mayor probabilidad de confundir los patrones de fallos diferentes.

Bajo esta filosofía, se pueden considerar varios escenarios cuando existe un aumento en la variabilidad de los datos. Una posible situación a considerar sería tener un diagnosticador robusto que mantiene un bajo error de clasificación, independientemente del nivel de variabilidad en los datos. Otro posible escenario es que, como resultado del efecto del ruido, el clasificador evaluado presente un deterioro importante en su desempeño mientras aumenta la variabilidad de los datos. A fin de comparar los clasificadores propuestos, en lo adelante, se establecerá que un diagnosticador robusto es aquel que mantiene altos indicadores de desempeño, independiente del efecto negativo de ruidos y/o perturbaciones externas. Desde el punto de vista de modelado se considerará, además, que se trata de un problema multivariable con $\boldsymbol{k}$ clases que representan los estados de operación del sistema, tal que el efecto de un ruido en las variables medidas puede ser modelado como:

$$X(t) = S(t) + \Gamma(t) \tag{1}$$

donde el comportamiento habitual en las mediciones del proceso es representado por $S(t) \sim \mathrm{N}(\mu_s, \Sigma_S)$ y $\Gamma(t) \sim \mathrm{N}(0, \sigma_\gamma^2 \mathrm{I})$ es la incertidumbre adicional incorporada por el ruido. En procesos reales, por lo general, no se cuenta con información sobre el tipo y cantidad de ruido implícito en las mediciones; pero esta información puede ser supuesta apriori. Además, con el objetivo de facilitar el estudio de la robustez de los clasificadores, se asume que el ruido (i.i.d.) es acotado y cada columna de $S(t)$ es de la forma $s_j(t) \sim N(\mu_{s_j}, \sigma_{s_j}^2)$, donde $j = \{1, \ldots, p\}$ denota la variable medida. A partir de esto, $\mu_{s_j} \pm 3\sigma_{s_j}$ determina el rango específico dentro del cual se tiene un 99,73 % de información válida para las distribuciones de cada una de las variables [6]. Por tanto, para una variabilidad en los datos que es acotada entre $\pm 3\sigma_{s_j}$ respecto al comportamiento nominal del proceso, $X(t)$ puede contener informaciń de $S(t)$, incluso si $\Gamma(t) \neq 0$; fuera de este intervalo es ms difícil obtener altos desempeños en las tareas de clasificación debido a la mezcla de clases.

III. Índicador de robustez propuesto

El rechazo de ruido por nivel es la capacidad de un clasificador de no ser afectado por la variabilidad en los datos, como resultado de un ruido acotado. Entonces, la sensibilidad al ruido por parte del clasificador, representada matemáticamente por el índice de robustez J_{RIL} se determina como:

$$J_{\mathrm{RIL}} = H\left[\left(I_{\mathrm{m}}^1 + V\right)\left(I_{\max}^0 - I_{\min}^0\right)\right] \tag{2}$$

donde I_{m}^1, $I_{\max}^0$ y $I_{\min}^0$ se calculan usando el método de los trapecios y brindan una medida del área bajo las curvas de tendencia del error de clasificación. Para un nivel de ruido, que varía con un incremento $\Delta\eta$ en un rango de $\eta_0 \leq \eta \leq \eta_{max}$, $I_{\min}^0$ y $I_{\max}^0$ son calculadas como sigue:

$$I_{\lim}^0 = \int_1^q \mathrm{E}_{\lim}\left(f(X)|\eta_i\right) d\eta \tag{3}$$

$$I_{\lim}^0 = \Delta\eta\left[\frac{\mathrm{E}_{\lim}|_{\eta=\eta_0}}{2} + \sum_{i=1}^{N_{i-1}} \mathrm{E}_{\lim}|_{\eta=\eta_i} + \frac{\mathrm{E}_{\lim}|_{\eta=\eta_{i+1}}}{2}\right] \tag{4}$$

Considerando que:

$$I_{\lim}^0 = \begin{cases} I_{\max}^0, & \text{si } \mathrm{E}_{\lim} = \mathrm{E}_{\max} \\ I_{\min}^0, & \text{si } \mathrm{E}_{\lim} = \mathrm{E}_{\min} \end{cases} \tag{5}$$

La diferencia entre $I_{\max}^0$ y $I_{\min}^0$ modela el comportamiento del clasificador teniendo en cuenta el intervalo de confianza para cada uno de los errores estimados. Por otra parte, I_{m}^1 está asociado con la tendencia del valor medio obtenido para el error de clasificación (E).

$$I_{\mathrm{m}}^1 = \int_1^q \eta_i^\ell \bar{\mathrm{E}}\left(f(X)|\eta_i\right) d\eta \tag{6}$$

$$I_{\mathrm{m}}^1 = \Delta\eta\left[\eta_0^\ell\left(\frac{\bar{\mathrm{E}}|_{\eta=\eta_0}}{2}\right) + \sum_{i=1}^{N_{i-1}} \eta_i^\ell(\bar{\mathrm{E}}|_{\eta=\eta_i}) + \frac{\eta_{i+1}^\ell(\bar{\mathrm{E}}|_{\eta=\eta_{i+1}})}{2}\right] \tag{7}$$

En las ecuaciones (4) y (7) los valores constantes η_0, η_{max} y $\Delta\eta$ se definen por el investigador. La relación entre estos

parámetros está dada por $H = \Delta\eta/(\eta_{max} - \eta_0)$. En tanto, la variabilidad σ_γ adicionada por un ruido $\Gamma(t)$ a una variable de $S(t)$ es:

$$\sigma_\gamma = \sqrt{(\eta - 1)}\sigma_s \quad \eta \in \mathbb{R}^+, \eta \geq 1 \tag{8}$$

donde el nivel de severidad η que caracteriza la relación entre $X(t)$ y $S(t)$, cuando los mismos proceden de una distribución normal, puede ser obtenido como:

$$\eta = \left(\frac{\Psi_X}{\Psi_S}\right)^2 = \left[\frac{(\sigma_x/\mu_s) \times 100}{(\sigma_s/\mu_s) \times 100}\right]^2 \tag{9}$$

En este contexto, el parámetro V considera la rápidez con que se deteriora el desempeño del clasificador a medida que se incrementa la variabilidad en los datos y está dado por:

$$V = \sum_{i=2}^{N} \left(\bar{E}_i - \bar{E}_{i-1}\right) / \left(\eta_i - \eta_{i-1}\right) \tag{10}$$

donde $\ell \in \mathbb{N}$ es un parámetro de magnificación usado en (7) y (10), a fin de penalizar aquellos clasificadores que ante elevados niveles de ruido, presentan un deterioro significativo en su desempeño. Diferentes valores de este parámetro, permiten dar mayor peso al comportamiento del clasificador según I^1_{m} o V. Para todos los casos analizados en el presente trabajo, el valor de este parámetro se fijó como $\ell = 2$.

IV. Robustez del diagnóstico en el CSTR

A continuación se evalúa la aplicación del índice de robustez J_{RIL} en el conocido proceso de prueba Tanque Reactor Continuamente Agitado (CSTR). Para ello, se emplean tres clasificadores diferentes y se incorpora variabilidad en el proceso según el esquema mostrado en la Figura 1.

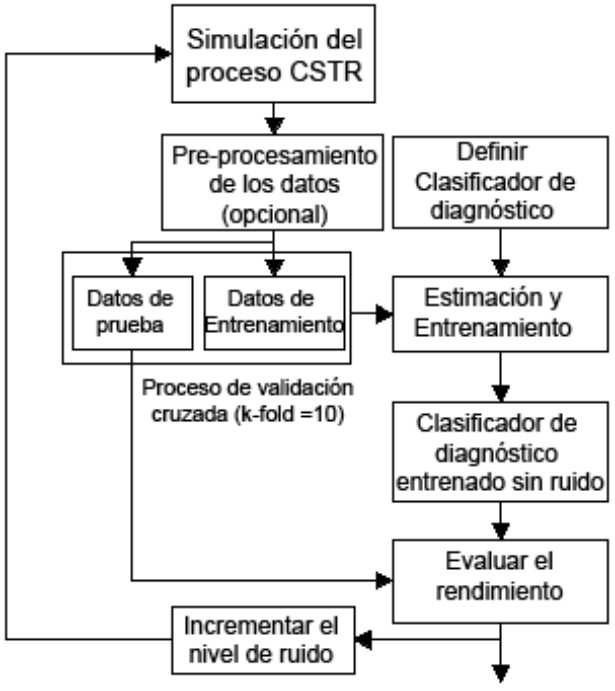


Figura 1. Flujograma empleado para los experimentos en el CSTR.

Para realizar la comparación de los clasificadores, se generan 91 conjuntos de datos históricos. El primer conjunto, describe la operación del CSTR considerando sólo la variabilidad típica en el proceso, y se utiliza para entrenar fuera de línea, cada uno de los clasificadores. Los conjuntos de datos históricos restantes, están asociados con la operación del proceso, a medida que se va incrementando el efecto del ruido. Todos los datos son recopilados fuera de línea, considerando la misma estructura y número de clases. Es decir, cada conjunto de datos históricos se forma a partir de nueve clases con 800 observaciones, que corresponden a cada uno de los fallos descritos en la Tabla I.

Cuadro I
Descripción de los fallos en el proceso CSTR.

No.	Descripción de los fallos	Valor
1	Variación abrupta en el flujo (Q_F)	10 L/min
2	Temperatura del reactor con una desviación	4 K
3	Incremento en la concentración (Aumento de C_{AF})	Pendiente $6 \cdot 10$ (mol/L)/min
4	Incremento en la concentración (Aumento de C_{AF})	Pendiente 0,1 K/min
5	Aumento de la temperatura (del flujo refrigerante T_{CF})	Pendiente 0,1 K/min
6	Variación abrupta en la presión en la línea de enfriamiento (P_U)	2,5 psi
7	Variación abrupta en la presión en la línea de salida del reactor(P_D)	5 psi
8	La válvula del refrigerante presenta juego.	20 % del span

En todos los casos, los datos generados son sometidos a un paso intermedio donde se estandarizan de acuerdo con el estado de operación normal. El error obtenido por los clasificadores, se almacena en cada iteración. Posteriormente, se determina el índice de robustez para $1 \leq \eta \leq 10$ y $\Delta\eta = 0{,}1$. El papel de los clasificadores consiste en identificar correctamente a cuál de los estados de fallos conocidos, corresponde una nueva observación del proceso.

IV-A. Clasificadores empleados.

En la literatura han sido abordadas múltiples herramientas discriminantes, con diferentes grados de éxito durante su aplicación a los problemas de diagnóstico de fallos. Seguidamente se presentan los aspectos básicos relacionados con las herramientas discriminantes utilizadas para probar el índice de robustez J_{RIL}.

Árboles de Decisión: Un Árbol de Decisión (AD) constituye un conjunto de condiciones organizadas en una estructura jerárquica para clasificar clases disjuntas. Cada rama, desde la raíz a las hojas, se puede interpretar como una regla, siendo los nodos hojas la clase asignada y los nodos internos los términos en conjunción (antecedente de la regla). Durante la clasificación con esta herramienta, cada elemento de su dominio es mapeado en un elemento de su rango, el cual es típicamente un identificador de clase o un valor numérico. En cada hoja del árbol se encuentra un elemento de rango;

mientras que en cada nodo interno se encuentra una prueba que tiene un conjunto de posibles resultados. Aunque este tipo de herramienta de clasificación no es la más competitiva en términos de predicción, en el presente trabajo se seleccionó para evaluar el índice de robustez propuesto por su simplicidad, fácil implementación e interpretación.

Hay varios enfoques para el diseño de estructuras de arboles e qupueden ser empleados. [8]. En este caso el AD fue implementado comenzando desde la raíz, donde se encuentra la variable medida que mayor información ofrece. Su construcción avanza a partir de las ramas del árbol, que son las variables ordenadas a partir de la información discriminante que ofrece cada una, hasta llegar a las hojas que corresponden con la clasificación de los fallos. Este procedimiento es conocido como Inducción de Arriba hacia Abajo (por su denominacin en idioma inglés: *Top-Down Induction of Decision Trees*) [13]. El clasificador basado en Árboles de Decisión [12], fue implementado usando el algoritmo de partición ID3. A partir de minimizar criterios de entropía, este algoritmo determina el árbol que genera menor cantidad de encuestas a los sensores y que dispone de la información más rápidamente.

***Redes Neuronales Artificiales**:* La siguiente herramienta discriminante que es probada, se basa en el uso de las Redes Neuronales Artificiales. Los clasificadores de este tipo, se caracterizan por su tolerancia al ruido y su capacidad para generalizar la información, por tanto, son ideales para evaluar el índice de robustez propuesto. En este caso, se utiliza una Red Neuronal del tipo perceptrón multicapa cuyos parámetros de entrada y salida estn asociados con la cantidad de variables del proceso y el número de fallos, respectivamente. En este sentido se desarrolla una arquitectura con una sola capa oculta con 9 neuronas, que posee 14 entradas y 9 salidas que permiten distinguir los fallos simulados en el CSTR. La red fue creada, entrenada e implementada utilizando el algoritmo de entrenamiento Levenberg−Marquardt. El proceso de entrenamiento se realizó de manera iterativa para minimizar el error cuadrático medio (MSE, *Mean Squared Error*) entre la salida de la red y el vector de entrenamiento. En cada iteración, el gradiente del desempeño de la función MSE fue utilizado para ajustar los pesos y umbrales de la red. En este estudio, se empleó un MSE= 10^{-6}, y un valor mínimo para el gradiente de 10^{-8}, así como un número máximo de épocas igual a 100. El proceso de entrenamiento de la red, se detiene si cualquiera de las condiciones anteriores se cumple. Los pesos iniciales de la red fueron generados de manera aleatoria.

***Máquinas de Soporte Vectorial.**:* Las Máquinas de Soporte Vectorial, representan un clasificador basado en funciones kernel que es relativamente nuevo. Su principio de operación se fundamenta en la idea de usar un hiperplano para crear un clasificador, cuyo margen de separación entre clases sea máximo. En un problema de clasificación binario, $\mathbf{w}$ es interpretada como la región entre los hiperplanos paralelos tal que $f(x) = \mathbf{w}^{\mathrm{T}}\mathbf{x_i} + \mathbf{w}_0 = \pm 1$.

A partir de esto, la distancia de cualquier punto localizado en uno de los dos hiperplanos a la funcin de clasificacin es igual a $\mathtt{d} = 1/\|\mathbf{w}\|$. La formulación del hiperplano queda definida entonces como:

$$J(\mathbf{w}, \mathbf{w}_0, \xi) = \min_{\mathrm{w}\neq 0} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{\ell}\xi_i \qquad (11)$$

$$sujeto\ a: \quad y_i(\mathbf{w}^{\mathrm{T}} + \mathbf{w}_0) \geq 1 - \xi_i \quad \xi_i \geq 0$$

donde $y_i \in \{-1, 1\}$ corresponde a la etiqueta de la clase asociada y C es el parámetro de regularización mediante el cual se logra un balance entre el error cometido y el ancho del margen. En caso de que x_i sea clasificada correctamente pero fuera del margen entonces, $\xi_i = 0$; si por el contrario, se encuentra dentro de éste, entonces $0 \leq \xi_i \leq 1$ y si ξ_i est mal clasificada $\xi_i \leq 1$. Utilizando la representación del problema en su forma dual, es posible entonces evaluar las condiciones de Karush-Kuhn-Tucker para obtener el correspondiente vector $\alpha = (\alpha_1, \ldots, \alpha_\ell)$ de multiplicadores de Lagrange positivos. Cuando el conjunto de entrenamiento no es linealmente separable, se adopta la filosofía kernel para mapear los vectores característicos hacia un espacio de mayor dimensión, donde las clases son linealmente separables. Como resultado es posible reescribir el problema dual, tal que:

$$W(\alpha) = \max_{\alpha\neq 0} \sum_{i=1}^{\ell}\alpha_i - \frac{1}{2}\sum_{i=1}^{\ell}\sum_{i=j}^{\ell}\alpha_i\alpha_j y_i y_j \mathbf{K}_{ij} \qquad (12)$$

Sustituyendo el producto punto por $\mathrm{K}_{ij} = \mathbf{k}(x_i, x_j)$ se obtiene la función de decisión generalizada mostrada anteriormente, tal que la misma queda expresada como:

$$f(x) = \mathbf{w}^{\mathrm{T}}\mathbf{x_i} + \mathbf{w}_0 = sign\left(\sum_{i=1}^{sv}\alpha_i y_i \mathbf{k}(x_i, x_j) + \mathbf{w}_0\right) \qquad (13)$$

donde sv corresponde a los vectores soportes obtenidos para α_i multiplicadores de Lagrange que son no nulos. Para el diseño del clasificador MSV, en este caso se optó por utilizar la estrategia binaria (1 vs.Todos). De esta manera se requieren solo $(c-1)$ máquinas soporte. Este enfoque de clasificación se implementó utilizando un kernel Gausiano, cuyo parámetro de ajuste fue estimado con la medida Alfa propuesta por [3].

IV-B. Análisis y discusión de los resultados

A fin de evaluar el desempeño de los clasificadores de diagnóstico en el CSTR, se realizan dos experimentos que consisten en incorporar ruido en las variables Flujo del refrigerante (C_{AF}) y Concentración de alimentación (Q_C) de dicho proceso. El primer paso de cada uno de los experimentos, se centra en determinar el error de clasificación cuando los datos históricos tienen el nivel de ruido que normalmente influye sobre el sistema. De esta manera, es posible conocer el valor medio del error, que se desea mantener para cada uno de los clasificadores a medida que se va incrementando la variabilidad de los datos durante los sucesivos pasos del procedimiento para calcular J_{RIL}.

Las Tablas II, IV y III, ilustran los resultados obtenidos para el primer paso del procedimiento. Nótese que los mejores

resultados se obtienen con las MSV, seguidas por las RNA y el clasificador AD. La diferencia entre el desempeño de los clasificadores es pequeña, y hace pensar que desde el punto de vista práctico es equivalente utilizar cualquiera de ellos como diagnosticador.

Cuadro II
MATRIZ DE CONFUSIÓN OBTENIDA PARA EL CLASIFICADOR AD.

	NOC	F1	F2	F3	F4	F5	F6	F7	F8	TA(%)
NOC	717	0	83	0	0	0	0	0	0	89.63
F1	0	800	0	0	0	0	0	0	0	100.0
F2	211	0	589	0	0	0	0	0	0	73.63
F3	11	0	1	788	0	0	0	0	0	98.50
F4	0	0	0	0	800	0	0	0	0	100.0
F5	0	0	0	0	0	800	0	0	0	100.0
F6	0	0	0	0	0	0	800	0	0	100.0
F7	0	0	0	0	0	0	0	800	0	100.0
F8	11	0	1	0	0	0	0	0	788	98.50
TA(%)	75.5	100	87.4	100	100	100	100	100	100	95.58
E (%)	24.5	0	12.6	0	0	0	0	0	0	4.42

Cuadro III
MATRIZ DE CONFUSIÓN OBTENIDA PARA EL CLASIFICADOR RNA.

	NOC	F1	F2	F3	F4	F5	F6	F7	F8	TA(%)
NOC	619	0	181	0	0	0	0	0	0	77.38
F1	0	800	0	0	0	0	0	0	0	100.0
F2	114	0	685	1	0	0	0	0	0	85.63
F3	6	0	2	792	0	0	0	0	0	99.00
F4	0	0	0	0	800	0	0	0	0	100.0
F5	0	0	0	0	0	800	0	0	0	100.0
F6	0	0	0	0	0	0	800	0	0	100.0
F7	0	0	0	0	0	0	0	800	0	100.0
F8	5	0	1	0	0	0	0	0	794	99.25
TA(%)	85.12	100	77	99.99	100	100	100	100	100	95.7
E (%)	14.88	0	23	0.001	0	0	0	0	0	4.30

Cuadro IV
MATRIZ DE CONFUSIÓN OBTENIDA PARA EL CLASIFICADOR MSV.

	NOC	F1	F2	F3	F4	F5	F6	F7	F8	TA(%)
NOC	711	0	89	0	0	0	0	0	0	88.88
F1	0	800	0	0	0	0	0	0	0	100.0
F2	182	0	618	1	0	0	0	0	0	77.25
F3	15	0	1	784	0	0	0	0	0	98.00
F4	0	0	0	0	800	0	0	0	0	100.0
F5	0	0	0	0	0	800	0	0	0	100.0
F6	0	0	0	0	0	0	800	0	0	100.0
F7	0	0	1	0	0	0	0	799	0	99.86
F8	16	0	3	0	0	0	0	0	781	97.63
TA(%)	73.37	100	88.25	99.99	100	100	100	100	100	95.74
E (%)	26.63	0	11.75	0.001	0	0	0	0	0	4.26

Al menos para este nivel de variabilidad en los datos, las Tablas II, IV y III muestran que la precisión se encuentra siempre por encima del 95% lo que denota una elevada certeza en las decisiones tomadas usando este criterio. A fin de complementar estos resultados, en la Figura 2 se muestra el comportamiento de cada clasificador, teniendo en cuenta el incremento en la variabilidad de los datos. Nótese que el análisis de robustez que mediante el índice J_{RIL} se muestra en la Figura 2 refleja que el desempeño anterior se deteriora significativamente a medida que aumentan los niveles de variabilidad en los datos como resultado de un ruido.

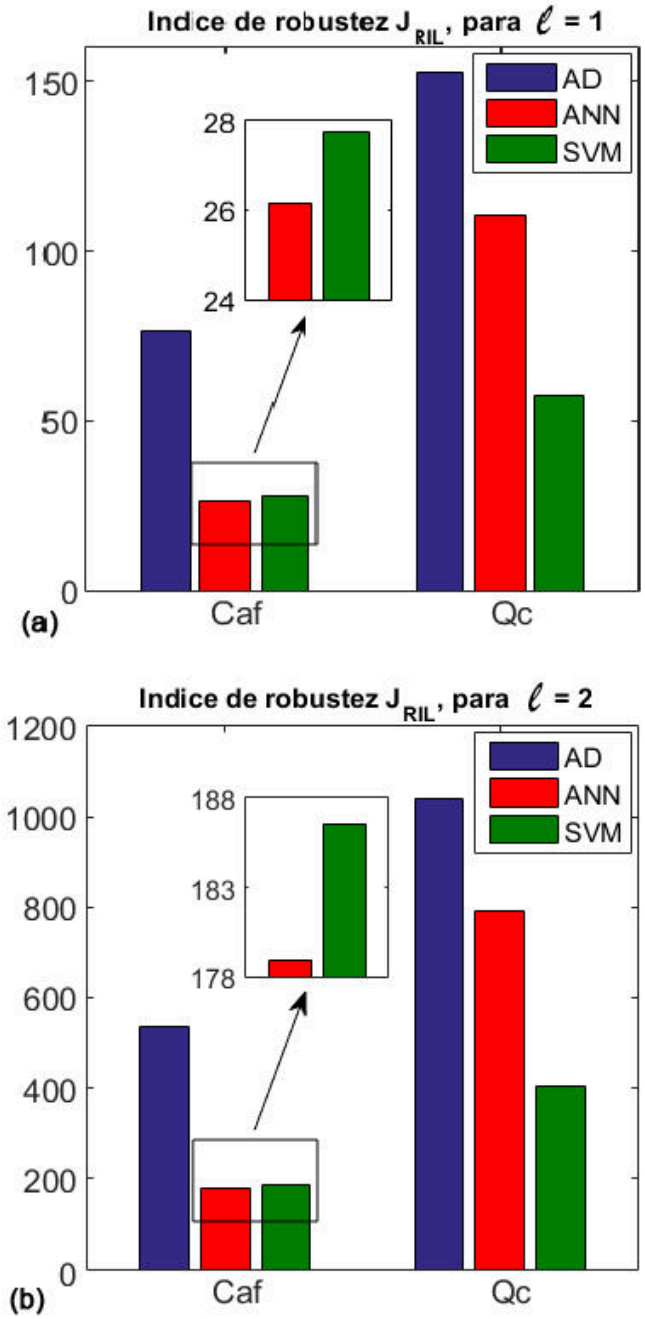


Figura 2. Comparando clasificadores mediante el índice de robustez J_{RIL}.

En este caso cuando la variable C_{AF} se encuentra afectada por diferentes niveles de ruido, el clasificador con mejor desempeño está basado en Redes Neuronales. En tanto, las Máquinas de Soporte Vectorial resultan el clasificador más robusto cuando la variable Q_C está expuesta a diferentes

niveles de ruido. Según se ilustra en la Figura 2, el parámetro de magnificación ℓ permite resaltar las diferencias entre valores de robustez similares. Al comparar las gráficas de barras anteriores, es notable que el sistema de diagnóstico diseado es más sensible al efecto del ruido en Q_C que en C_{AF} independientemente del clasificador empleado. En específico, el mayor deterioro se presenta para el clasificador basado en la herramienta de Árboles de Decisión. Teniendo en cuenta los resultados mostrados anteriormente, y considerando la simulitud en el desempeño de los clasificadores sin un incremento de la variabilidad típica del proceso, se podría decir que en este caso no es recomendable emplear un clasificador basado en Redes Neuronales o en Árboles de Decisión. Una alternativa viable sería, seleccionar las Máquinas de Soporte Vectorial como clasificador a utilizar dado el alto rendimiento y nivel de rechazo al ruido que esta herramienta obtuvo durante las pruebas realizadas.

V. Conclusiones

En el presente trabajo se discutió la influencia negativa del ruido en los procesos de clasificación que forman parte de las tareas de diagnóstico de fallos. Además, se resaltó la necesidad de contar con sistemas de diagnóstico que sean robustos ante ruidos y/o perturbaciones, manteniendo altos indicadores de rendimiento. A fin de identificar cuáles son las herramientas de clasificación que cumplen estos requisitos, se evalu el índice de robustez J_{RIL} usando el proceso CSTR. Los resultados obtenidos de estos experimentos mostraron que el índice propuesto permite, a partir de un único valor, establecer cuál es el clasificador más robusto, dado un ruido que afecta a las señales medidas con un rango de variabilidad conocido. Sin embargo, el indicador propuesto no brinda información específica sobre el deterioro que tiene el clasificador a medida que aumenta la variabilidad en los datos. En este sentido, es recomendable complementar el índice de robustez con una representación gráfica del deterioro del diagnosticador. De esta manera sería posible valorar la sensibilidad del clasificador analizado e identificar además, posibles alternativas de hibridación que permitan alcanzar rendimientos superiores en términos de robustez. La ventaja del índice propuesto, respecto a otros análisis de robustez local, radica en la capacidad de decisión que a partir de un único valor numérico brinda este indicador. Al utilizar el índice de robustez J_{RIL} no solo se reduce el número de análisis comparativos a realizar, sino que además, se evita considerar cualquier criterio subjetivo a la hora de seleccionar las herramientas de clasificación más adecuadas.

VI. Agradecimentos

Los autores agradecen el apoyo financiero brindado por la Asociación Universitaria Iberoamericana de Postgrado (AUIP). También agradece la colaboración del Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) en Brasil, y de la Universidad Tecnológica de la Habana, CUJAE. Así como los Proyectos TIN201786647-P y TIN2017-86647-P (MINECO/AEI/FEDER, UE).

Referencias

[1] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, Diagnosis and Fault-Tolerant Control, Springer, 2006.

[2] L. H. Chiang, R. D. Braatz, E. L. Russell, Fault detection and diagnosis in industrial systems, Springer, 2001.

[3] P. Chudzian, .Evaluation measures for kernel optimization", Pattern Recognition Letters. 33, 1108–1116, 2012.

[4] A. Das, J. Maiti, R. Banerjee, "Process monitoring and fault detection strategies: A review", International Journal of Quality and Reliability Management, 29(7), 720–752, 2012.

[5] S. X. Ding, Data-driven Design of Fault Diagnosis and Fault-tolerant Control Systems, Springer, 2014.

[6] D. C. Montgomery, Introduction to Statistical Quality Control, Wiley and Sons, 2005.

[7] J. Korbicz, Fault diagnosis: Models, Artificial Intelligence, Applications, Springer Science and Business Media, 2003.

[8] J. Hernández-Orallo, MJ. Ramírez-Quintana, C. Ferri, Introducción a la Minería de Datos, Prentice Hall and Addison-Wesley, 2004.

[9] R.J. Patton, J. Chen, Robust model-based fault diagnosis for dynamic systems, Kluwer Academic Publishers, London, 1999.

[10] K. Patan, M. Witczak, J. Korbicz, "Towards robustness in neural network based fault diagnosis", International Journal of Applied Mathematics and Computer Science. 18, 443–454, 2008.

[11] J. A. Sáez, M. Galar, J. Luengo, F. Herrera, .Analyzing the presence of noise in multi-class problems: Alleviating its influence with the One-vs-One decomposition", Knowledge and Information Systems. 38 (2014) 179–206.

[12] J. R. Quinlan, Ïnduction of decision trees", Machine Learning, 1(1):81–106, 1986.

[13] L. Rokach, O. Maimon, "Top-down induction of decision trees classifiers: A survey", IEEE Transactions on Systems, Man, and Cybernetics, 35(4):476–487, 2005.

shinytests: Una herramienta gráfica para la comparación estadística en minería de datos

Jacinto Carrasco*, Salvador García* and Francisco Herrera*
* Dpto. de Ciencias de la Computación e Inteligencia Artificial
Universidad de Granada
Granada, España
Email: {jacintocc, salvagl, herrera}@decsai.ugr.es

***Resumen*—Los test estadísticos constituyen el procedimiento más fiable para la validación de los resultados obtenidos en múltiples escenarios. En particular, debido a su robustez y aplicabilidad, los test no paramétrico son una herramienta habitual y útil en el proceso de diseño y evaluación de los algoritmos de aprendizaje automático para ámbitos tanto de clasificación como de optimización. Las nuevas tendencias como el uso de test bayesianos y la observación de la distribución del parámetro de interés representan un enfoque a tener en cuenta.**

En esta contribución se presenta la aplicación shiny de R *shinytests*, la cual integra test bayesianos y no paramétricos para facilitar la realización de test estadísticos en la comparación de algoritmos de aprendizaje automático y optimización.

***Index Terms*—Test estadísticos, test bayesianos, software, shinyapp, R**

I. Introducción

En el desarrollo de algoritmos de aprendizaje automático y de optimización existe una necesidad creciente de validar y examinar la incertidumbre presente en estos procesos. Los test estadísticos son la herramienta recomendada para asegurar que las conclusiones obtenidas de los correspondientes experimentos no están sesgadas por la intención del investigador o se han obtenido por una cuestión de azar [1].

Existen numerosos test que pueden usarse para este propósito, los cuales pueden ser clasificado en dos grandes categorías: Los test frecuentistas, principalmente los test de hipótesis nula [2] o NHST (*Null Hypothesis Statistical Tests*), y los test bayesianos [3]. El primer grupo está subdividido en test paramétricos, que no se tendrán en cuenta en este artículo por estar suficientemente extendidos, y los test no paramétricos [4], los cuales requieren unas condiciones de aplicabilidad menos estrictas que los test paramétricos aunque esto se traduzca en ocasiones en una menos habilidad para encontrar diferencias existentes entre los resultados de los algoritmos [5]. Estos prerrequisitos son habitualmente la normalidad de la muestra o la homocedasticidad, condiciones que pueden ser comprobadas mediante test no paramétricos como el test de Kolmogorov-Smirnov, así como otros test sobre ciertas propiedades de la muestra, como la aleatoriedad de una muestra o el ajuste a una distribución. Para la comparación del desempeño de algoritmos se usan habitualmente los test de Wilcoxon de rangos con signo o el test de Friedman para la comparación de múltiples algoritmos. Además de un resumen descriptivo de los test incluidos, se incluye un caso del uso de la aplicación *shiny* de `R` para la aplicación de los test bayesianos y la obtención de las gráficas correspondientes de la distribución del parámetro de interés, lo que ayuda a comprender estos test y sintetiza la información dada por éstos.

Esta contribución está organizada de la siguiente manera. En la Sección II se introducen los conceptos estadísticos necesarios y se describen los diferentes test estadísticos. En la Sección III se describen los principales métodos incluidos en la aplicación y se muestran varios ejemplos de su uso. En la Sección IV se concluye la contribución.

Este trabajo se ha sustentado por el proyecto de investigación TIN2017-89517-P. J. Carrasco disfruta de una beca FPU del Ministerio de Educación de España.

II. Antecedentes

En la Subsección II-A se introducen conceptos básicos de estadística que den soporte al resto del artículo. A continuación, en la Subsección II-B se describe el uso de los test frecuentistas clásicos para la comparación de algoritmos, con especial interés en los test no paramétricos. Los test bayesianos para la comparación de la eficacia de algoritmos se incluye en la Subsección II-C.

II-A. Conceptos preliminares

En la inferencia estadística estamos interesados en obtener una predicción fiable a partir de los datos, por lo que debemos evitar llegar a conclusiones erróneas producidas por efectos aleatorios. Los principales conceptos a tener en cuenta son [2]:

- Los resultados de los algoritmos implicados en la comparación constituyen una **muestra**. Esta representa el desempeño del algoritmo sobre uno o varios problemas, ya sea la medida de ajuste sobre un problema de optimización o bien el acierto sobre un conjunto de datos en un problema de clasificación. Desde el punto de vista estadístico, esta muestra proviene de una distribución de probabilidad desconocida y será usada para inferir información relevante.
- Al hablar del **parámetro** de interés, o de la distribución de un cierto parámetro, nos referimos a la medida usada para evaluar la diferencia entre los resultados de los algoritmos, o bien el ajuste de una muestra con respecto a una distribución.

- Un enfoque frecuentista para inferir información relevante consiste en el cálculo de un estadístico, es decir, un estimador de una característica de la distribución.
- La **distribución** de la cual obtenemos la muestra es desconocida, por lo que los estadísticos se usarán para estimar el parámetro de interés.

II-B. Test frecuentistas

Los test frecuentistas son la herramienta más común en la comparación del desempeño de algoritmos hasta ahora [6]. En ellos, se establece una hipótesis nula ($\mathcal{H}_0$) y una hipótesis alternativa ($\mathcal{H}_1$). Entonces, haciendo uso de una muestra se calcula la probabilidad de obtener una muestra tan alejada de la hipótesis nula como la que disponemos asumiendo que $\mathcal{H}_0$ es cierta. Esta probabilidad se conoce como p-valor [2]. Entonces, si la probabilidad obtenida es menor que un valor fijo α (normalmente 0,05), se rechaza $\mathcal{H}_0$, mientras que de otra manera no hay suficiente evidencias como para rechazar la hipótesis nula.

- Además, como estamos interesados en la comparación estadística, debemos prestar atención a las propiedades de los test estadísticos. Se define el **error de tipo I** como la probabilidad de rechazar la hipótesis nula $\mathbf{H}_0$ cuando es cierta y el **error de tipo II** cuando $\mathbf{H}_0$ no se rechaza y es falsa.
- La principal medida de para comparar la calidad de un test es la **potencia**, es decir, la probabilidad de rechazar $\mathcal{H}_0$. Estaremos interesados en obtener una mayor potencia manteniendo el error de tipo I, que se representa con el parámetro α.

II-B1. Test paramétricos: Estos test parten de la suposición de que la muestra proviene de una familia conocida de distribuciones, habitualmente la distribución normal. Cuando se cumple la hipótesis de normalidad se obtiene un test más potente. Los principales test que se corresponden con esta categoría son el t-test para la comparación de dos muestras pareadas y el test ANOVA para la comparación de múltiples algoritmos. En ambos test la hipótesis nula consiste en la equivalencia de la media del desempeño de los algoritmos involucrados.

II-B2. Test no paramétricos: Los test no paramétricos no asumen que la muestra provenga de una distribución de una familia conocida [7], lo que se traduce en que se tengan condiciones menos restrictivas sobre la muestra, como la simetría o la continuidad [8]. En consecuencia, los test no paramétricos son más robustos que los paramétricos, puesto que normalmente no se dan las condiciones necesarias para su uso.

Para asegurarnos de que estamos usando correctamente el test ANOVA o el t-test debemos comprobar la normalidad de la muestra, para lo que pueden usarse test sobre la bondad del ajuste para, al menos, no rechazar esta hipótesis. Sirven para ello test de bondad del ajuste como los test de Kolmogorov-Smirnov, Shapiro-Wilk y D'Agostino-Pearson [9].

El test no paramétrico recomendado para la comparación de algoritmos depende del número de algoritmos a comparar y distintas situaciones implicadas:

- **Test de signo y Test de Rangos con signo de Wilcoxon**: El test de signo es un análogo del t-test simple y el test de Wilcoxon es la versión análoga del t-test pareado.
- **Test de Friedman**: Este test cumple con la función análoga al test paramétrico ANOVA. Se realiza una comparación de k algoritmos en n problemas (conjuntos de datos o funciones *benchmark*). El estadístico se calcula en base al orden de los algoritmos para cada problema. El test de Iman-Davenport constituye una propuesta más potente basada en el test de Friedman.
- **Test de Friedman de rangos alineados**: Esta mejora del test de Friedman usa el orden de los resultados en todos los problemas, lo que se traduce en que es tenida en cuenta la dificultad de cada problema.

En el caso de que existan diferencias significativas en la realización de test para múltiples algoritmos y se rechace la hipótesis nula $\mathcal{H}_0$, nuestro propósito será discernir dónde se encuentran estas diferencias. Para este paso es necesario un ajuste en el p-valor obtenido para mantener el control sobre el *Family-wise Error Rate* (FWER). Algunos ejemplos de test post-hoc son los de Bonferroni-Dunn, Holm, Holland, Hochberg o Li [6], [10], [11].

II-C. Test bayesainos

Un enfoque distinto es el propuesto por Benavoli *et al.* [12]. La principal diferencia es que no se establece una hipótesis nula sobre el parámetro de interés para realizar un test de hipótesis nula, sino que se obtiene una distribución de probabilidad sobre el parámetro de interés.

II-C1. Comparación con los test frecuentistas: Según Benavoli [13], las principales diferencias que se podrían identificar son:

- En los test frecuentistas, las decisiones sobre la significatividad de un test son dicotómicas, basadas en el p-valor y el nivel α de significatividad. En la estadística bayesiana, no existe un umbral fijo para el rechazo de la hipótesis nula sino la distribución del parámetro, de donde obtenemos la probabilidad de que la hipótesis nula sea cierta.
- En la aplicación de los NHST existe una confusión habitual, y es que el p-valor no representa la probabilidad de que se dé la hipótesis nula, sino, asumiendo que la hipótesis nula es cierta, obtener una muestra tan alejada de $\mathcal{H}_0$ como la que disponemos. Normalmente querremos responder la primera pregunta, la cual obtenemos usando los test bayesianos.
- Una crítica común a los NHST es que el tamaño del efecto y el tamaño de la muestra no son distinguibles. Esto significa que un efecto tan pequeño como sea necesario puede ser considerado como significante si se añaden suficientes instancias a la muestra. Como el tamaño de la muestra depende del investigador, se

podría variar el número de observaciones hasta obtener el resultado esperado.
- Los NHST no ofrecen información cuando la hipótesis nula no se rechaza. En esta situación, no podríamos decir que no hay diferencia entre las muestras, sino que no disponemos suficientes evidencias para rechazar la hipótesis nula. En cambio, en los test bayesianos la distribución del parámetro es informativa aunque no indique una suficiente diferencia entre los algoritmos.
- El proceso para realizar un test bayesianos consiste en establecer un modelo probabilístico *a priori* (basándonos en la información que disponemos o con una distribución *a priori* poco informativa), calcular en interpretar la distribución *a posteriori* basándonos en los datos disponibles, y evaluar el modelo.

II-C2. t-test bayesiano correlado: Esta versión bayesiana del t-test se usa para comparar los resultados de dos algoritmos de clasificación en un escenario de validación cruzada con k *folds* partition [14]. Este test tiene en consideración la correlación entre los distintos folds y parte de la hipótesis de que los datos vienen de una distribución gaussiana multivariante cuya matriz de covarianza depende de la correlación ρ entre los *folds*. Debido a que ρ no puede estimarse a partir de los datos, se utiliza la heurística sugerida por Nadeau y Bengio [15] y $\rho = \frac{n_{test}}{n_{tot}}$, esto es, el número de instancias en la partición de evaluación partido por le número total de instancias. Se parte de una distribución Normal-Gamma como la distribución *a priori* de la diferencia entre los algoritmos, por lo que se obtiene *a posteriori* una distribución de Student sobre la diferencia entre las medias μ. Debemos además considerar la posibilidad de que no haya una diferencia significativa entre el desempeño de ambos algoritmos, por lo que se debe definir una región de equivalencia (a la que llamaremos *rope*, por *region of practical equivalence*), $[r_{min}, r_{max}]$, definida para μ, y las relaciones entre los algoritmos se considerarán en términos de la *rope*. Por ejemplo, para a_1, a_2 algoritmos involucrados en la comparación, $P(a_1 \gg a_2) = P(\mu > r_{max})$ o $P(a_1 = a_2) = P(\mu \in \text{rope})$, donde la relación entre los algoritmos se refiere a la comparación del desempeño de ambos algoritmos. La *rope* por tanto nos permite realizar decisiones automáticas, aunque volviendo de esta manera a la pérdida de información y las decisiones dicotómicas. Sin embargo, en esta ocasión la interpretación de las probabilidades son directas y los límites para las decisiones pueden variar en función del contexto.

II-C3. Test bayesiano de signo: La versión bayesiana del test no paramétrico de signo hace uso del Proceso de Dirchlet (DP, *Dirichlet Process*) [16]. Podemos entender este proceso como una distribución de probabilidad sobre una familia de distribuciones de probabilidad, de manera que la inferencia se realiza en dos pasos.

- En primer lugar se obtiene la función de densidad de la distribución *a posteriori* como una combinación lineal de deltas de Dirac centradas en las observaciones, cuyos pesos provienen de una distribución de Dirichlet.
- Entonces, aproximamos la anterior función de probabilidad *a posteriori* como una probabilidad *a posteriori* de la que podemos calcular la probabilidad del parámetro de pertenecer a cada región de interés.

II-C4. Test bayesiano de rangos con signo: La versión bayesiana de rangos con signo tiene el mismo *background* estadístico que el test bayesiano de signo. También hace uso del DP como el método para realizar la inferencia a partir del datos. La diferencia radica en el hecho de que el test de rangos con signo usa dos muestras y la comparación entre ellas en el cómputo de las probabilidades de las posibles relaciones entre algoritmos. En este test no obtenemos una fórmula para la distribución *a posteriori*, pero podemos obtenerla muestreando los pesos de la distribución de Dirichlet.

II-C5. Test bayesiano de Friedman: El test bayesiano de Friedman [17] realiza un procedimiento similar a los descritos previamente para comprobar si es factible que el parámetro con la media del orden de la clasificación de cada algoritmo se quede en una región cercana al punto medio que constituye la hipótesis nula en un test frecuentista ($[(m+1)/2, \ldots, (m+1)/2]$). Si el parámetro de interés μ no se encuentra en esta región, existirá una diferencia significativa.

III. Aplicación *shiny*

Esta sección contiene en la Subsección III-A una descripción de la aplicación *shiny* desarrollada y su base, el paquete `rNPBST` [18]. La Subsección III-B contiene una descripción de la utilización de la aplicación para la comparación de algoritmos, principalmente sobre el uso de los métodos bayesianos, debido a que éstos son menos conocidos y su uso no está extendido.

III-A. Paquete rNPBST

El paquete `rNPBST` ha sido desarrollado inicialmente como un wrapper de la biblioteca `JavaNPST` desarrollada por Derrac *et al.* [19]. Es una biblioteca en Java que integra un extensivo conjunto de test no paramétricos de diferentes familias y con diferentes propósitos.

En la biblioteca original en Java sólo se incluyen test no paramétricos aunque se han añadido en el paquete de `R` varios test bayesianos y métodos asociados de visualización a través de `ggplot` y `ggtern` [20]. Los test se clasifican 11 categorías atendiendo al propósito de los test o el tipo de dato usando Tabla I.

El paquete `rNPBST` está disponible en un repositorio de Github[1] y se puede instalar usando el paquete `devtools` y ejecutando en `R`:

```
devtools::install_github("JacintoCC/rNPBST")
```

III-B. Ejemplo de uso

Para la ejecución de la aplicación *shiny* será necesario ejecutar en `R` la siguiente función del paquete `shiny`:

```
shiny::runGitHub(repo = "shinytests",
                 username = "JacintoCC")
```

[1] http://www.github.com/JacintoCC/rNPBST

Tabla I: Test incluidos en la versión actual de `rNPBST`

Family	Test
Test de aleatoriedad	Número de rachas Rachas crecientes y decrecientes Rachas crecientes y decrecientes (Mediana) Von Neumann
Test de bondad del ajuste	Chi-Squared Kolmogorov-Smirnov Lilliefors Anderson-Darling
Una muestra y muestras pareadas	Cuantil de confianza Cuantil de la población Test de signo Test de Wilcoxon de rangos con signo
Procedimientos general de dos muestras	Wald-Wolfowitz Test de medias Control Median Kolmogorov-Smirnov
Problema de escala	David-Barton Freund-Ansari-Bradley Mood Klotz Siegel-Tukey Sukhatme
Problema de posición	Wilcoxon Rank-Sum van der Waerden
Independencia de muestras	Extended Median test Kruskal-Wallis Jonckheere-Terpstra Charkraborti-Desu
Muestras bivariadas	Kendall Daniel Trend
Múltiples clasificadores	Friedman Iman-Davenport Rangos alineados de Friedman Page Coeficiente de concordancia Concordanza incompleta Correlación parcial
Conteo de datos	Coeficiente de contingencia Test exacto de Fisher McNemar Test de igualdad multinomial Ordered Equality test
Bayesianos	t-test bayesiano correlado Test bayesian de signo Test bayesiano de rangos con signo Test bayesiano de Friedman

Entonces se descargan automáticamente los paquetes necesarios y se abre en el navegador la aplicación. Se incluye la posibilidad de subir un fichero `.csv` para realizar los test estadísticos, así como seleccionar distintos test. A medida que vamos realizando cambios en el conjunto de datos introducido, o seleccionando el test a realizar en el menú lateral, se actualizarán automáticamente los resultados. También podremos seleccionar la opción de incluir un gráfico en algunos de los test bayesianos, el cual podremos descargar.

Para ejemplificar el uso de algunos test, presentamos un un estudio comparativo entre cinco algoritmos clásicos para problemas de clasificación. Los algoritmos incluidos en la comparación están descritos en la Tabla II. Los resultados de cada algoritmo en los distintos conjuntos de datos se incluyen en el paquete `rNPBST` y como conjunto por defecto en la aplicación `shinytests` para poder ejemplificar su uso. La medida usada ha sido la *accuracy*. Se incluye para cada algoritmo descrito en la Tabla II una tabla con los resultados en las particiones 5-dob-cv [21] de algunos de los conjuntos de datos disponibles[2] para clasificación en el repositorio KEEL

[2]abalone, australian, automobile, balance, breast, bupa, car, cleveland, crx, dermatology, german, glass, hayes-roth, heart, ionosphere, led7digit, letter, lymphography, mushroom, optdigits, satimage, spambase, splice, tic-tac-toe, vehicle, vowel, wine, yeast and zoo

Tabla II: Algoritmos comparados en el conjunto de datos de ejemplo

Algoritmo	Descripción	Conjunto de datos
`multinom`	Regresión logística, del paquete `nnet`.	`results.lr`
`knn`	Biblioteca `class`. Parám. $k = 1$, $l = 0$.	`results.knn`
`randomForest`	Biblioteca `randomForest`. Parám. `mtray` $= \sqrt{p}$.	`results.rf`
`nnet`	Biblioteca `nnet` library.	`resuts.nnet`
`naiveBayes`	Clasificador Naïve Bayes del paquete `e1071`.	`results.nb`

Tabla III: Wilcoxon Rank Sum test

Wilcoxon Rank Sum test		
data.name		results[, 1:2]
statistic		665.00
p.value	Asymptotic Left Tail	0.001565
	Asymptotic Right Tail	0.998512
	Asymptotic Double Tail	0.003129

[22] y en las particiones creadas para ello en este repositorio. Los resultados de las diferentes particiones se resumen usando el promedio en el conjunto de datos `results`. Se han mantenido por separado los conjuntos para todas las particiones para usarlos en el t-test correlado bayesiano.

III-B1. Análisis de muestras pareadas: Para una comparación paramétrica entre dos algoritmos podemos usar el test Wilcoxon Rank-Sum. El resultado de aplicar dicho test al conjunto de datos por defecto, seleccionando las dos primeras columnas (correspondientes a la comparación de la regresión logística y el KNN) se incluye en la Tabla III, puesto que en la aplicación *shiny* se incluye tanto una tabla HTML mostrando los resultados, como el código TEXque produce dicha tabla.

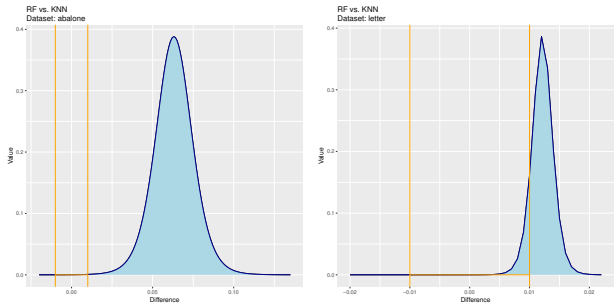
Vemos en la tabla Tabla III que la hipótesis nula $\mathcal{H}_0 : \mu_{LR} = \mu_{KNN}$ puede rechazarse debido a que el p-valor asintótico es menor que $0{,}05$, de manera que este test identifica una diferencia significativa entre estos dos algoritmos. Para determinar cuál obtiene mejores resultados, podemos mirar el p-valor para las hipótesis alternativas direccionales y concluimos quela regresión logística obtiene mejores resultados debido a que no podemos rechazar $\mathcal{H}_\prime$ cuando la hipótesis alternativa es $\mathcal{H}_1 : \mu_{LR} > \mu_{KNN}$.

III-B2. Test para comparaciones múltiples: Como se ha descrito en la Sección II, la hipótesis nula del test de Friedman es la equivalencia de las medianas de los diferentes algoritmos, por lo que un p-valor menor que un test significa que la hipótesis nula puede ser rechazada y existe una diferencia entre los algoritmos comparados. Se incluye en la Tabla IV los resultados del test de Friedman.

III-B3. t-test bayesiano correlado: En este test comparamos los resultados obtenidos por `random forest` y `knn` para un único dataset. Con el resultado de este test podemos obtener la Fig. 1 con la diferencia entre estos algoritmos.

Tabla IV: Friedman test

Friedman test		
data.name		df
statistic	s	2812.00
	q	39.06
p.value		6.789e-08



(a) Abalone (b) Letter

Figura 1: Distribución de RF vs KNN para dos conjuntos de datos

Tabla V: Bayesian correlated t-test

Bayesian correlated t-test		
probabilities for abalone dataset	left	4.962e-05
	rope	4.407e-04
	right	9.995e-01
probabilities for letter dataset	left	1.378e-07
	rope	1.105e-01
	right	8.895e-01
rope		-0.01
		0.01

La distribución *a posteriori* del parámetro de interés muestra cómo con un 99,9 % `random forest` tiene un mejor resultado que `knn` en este conjunto de datos. La distribución de la diferencia se muestra en Fig. 1 para los conjuntos de datos `abalone` y `letter`. En este segundo conjunto de datos, aunque `random forest` también obtiene un mejor resultado que `knn`, hay una mayor probabilidad de que ambos algoritmos obtengan el mismo resultado que en el primer conjunto de datos. Para los test bayesianos también obtenemos la Tabla V con los resultados, en este caso es la probabilidad de pertenencia a cada región de interés.

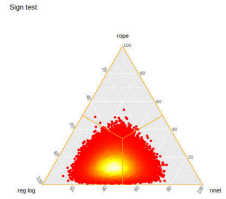
III-B4. Test bayesiano de signo: Para este test usamos los resultados promediados de dos algoritmos en todos los conjuntos de datos.

Hay una mayor probabilidad para la hipótesis de que la regresión logística obtenga un resultado mejor que la red neuronal, aunque podemos comprobar que las diferencias son pequeñas en la Tabla VI. En la Fig. 2 se observa una muestra de la distribución *a posteriori* y podemos comprobar cómo hay una mayor concentración de puntos en la región izquierda, que corresponde con la situación en la que la regresión logística obtiene un mejor resultado que la red neuronal. En la Fig. 2b la comparación se realiza entre `neural network` and `random forest`. Hay incluso una concentración mayor en la región izquierda, lo que nos dice que hay incluso una mayor probabilidad de que `random forest` obtenga un mejor resultado que `neural network`.

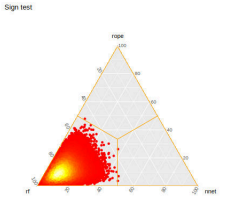
III-B5. Test bayesiano de rangos con signo: Repetimos la experimentación usando el test bayesiano de rangos con signo, mostrando la gráfica asociada en la Fig. 3 y los resultados numéricos en la Tabla VII.

Tabla VI: Bayesian Sign-test

test		
probabilities	left	0.4780
	rope	0.1444
	right	0.3777



(a) Red neuronal vs regresión logística



(b) Red neuronal vs random forest

Figura 2: Muestra de la distribución *a posteriori* del test bayesiano de signo

La probabilidad *a posteriori* para la región izquierda es ligeramente mayor que la probabilidad para el test bayesiano de signo, por lo que tenemos una mayor certeza para esta comparación usando el test bayesiano de rangos con signo. Como se puede ver en la Fig. 3, la distribución está desplazada hacia la izquierda, por lo que se espera una mayor potencia de este test con respecto al test bayesiano de signo.

III-B6. Test bayesiano de Friedman: En la Tabla VIII se incluye los resultados del test bayesiano de Friedman. En esta tabla se incluyen el orden medio de clasificación para cada algoritmo y la hipótesis seleccionada h, que en este caso $h = 1$, lo que significa que se rechaza que el parámetro pertenezca a la región de igual *ranking* para todos los algoritmos.

IV. Conclusiones

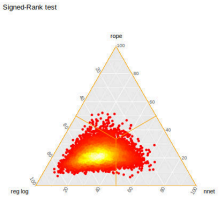
La experimentación inherente a la naturaleza del aprendizaje automático y el rápido crecimiento del número de algoritmos propuestos conlleva la necesidad de establecer un método claro de comparación del desempeño de estos algoritmos y

Tabla VII: Bayesian Signed-Rank test

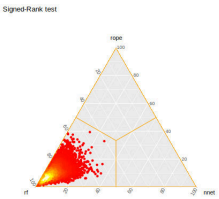
test		
probabilities	left	0.4921
	rope	0.2220
	right	0.2859

Tabla VIII: Bayesian Friedman test

Bayesian Friedman test					
h	1				
meanranks	3.1	2.433	4.467	2.233	2.767



(a) Red neuronal vs regresión logística



(b) Red neuronal vs random forest

Figura 3: Muestra de la distribución *a posteriori* del test bayesiano de rangos con signo.

una herramienta software que facilite este procedimiento. En esta contribución presentamos la aplicación *shiny* de `R`, cuyo principal objetivo es proporcionar una herramienta gráfica para los principales test no paramétricos y bayesianos existentes en el paquete `rNPBST`, de manera que se disponga de un software para investigadores interesados en comparar nuevo algoritmos. Como tareas futuras se trabajará en añadir nuevos test a esta aplicación.

Referencias

[1] N. Japkowicz and M. Shah, eds., *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011.

[2] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*. crc Press, 2003.

[3] J. M. Bernardo and A. F. Smith, *Bayesian Theory*. IOP Publishing, 2001.

[4] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[5] J. Luengo, S. García, and F. Herrera, "A study on the use of statistical tests for experimentation with neural networks: Analysis of parametric test conditions and non-parametric tests," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7798–7808, 2009.

[6] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.

[7] F. Pesarin and L. Salmaso, *Permutation Tests for Complex Data: Theory, Applications and Software*. John Wiley & Sons, 2010.

[8] E. Kasuya, "Wilcoxon signed-ranks test: Symmetry should be confirmed before the test," *Animal Behaviour*, vol. 79, pp. 765–767, Mar. 2010.

[9] J. Pizarro, E. Guerrero, and P. L. Galindo, "Multiple comparison procedures applied to model selection," *Neurocomputing*, vol. 48, no. 1, pp. 155–173, 2002.

[10] S. Garcia and F. Herrera, "An Extension on"Statistical Comparisons of Classifiers over Multiple Data Sets"for all Pairwise Comparisons," *Journal of Machine Learning Research*, vol. 9, no. Dec, pp. 2677–2694, 2008.

[11] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3 – 18, 2011.

[12] A. Benavoli and C. P. de Campos, "Statistical Tests for Joint Analysis of Performance Measures," in *Advanced Methodologies for Bayesian Networks - Second International Workshop, AMBN 2015, Yokohama, Japan, November 16-18, 2015. Proceedings*, pp. 76–92, 2015.

[13] A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a Change: A Tutorial for Comparing Multiple Classifiers Through Bayesian Analysis," *Journal of Machine Learning Research*, vol. 18, no. 77, pp. 1–36, 2017.

[14] G. Corani and A. Benavoli, "A Bayesian approach for comparing cross-validated algorithms on multiple data sets," *Machine Learning*, vol. 100, no. 2-3, pp. 285–304, 2015.

[15] C. Nadeau and Y. Bengio, "Inference for the generalization error," *Machine Learning*, vol. 52, no. 3, pp. 239–281, 2003.

[16] A. Benavoli, G. Corani, F. Mangili, M. Zaffalon, and F. Ruggeri, "A Bayesian Wilcoxon signed-rank test based on the Dirichlet process," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 1026–1034, 2014.

[17] A. Benavoli, G. Corani, F. Mangili, and M. Zaffalon, "A Bayesian nonparametric procedure for comparing algorithms," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1264–1272, 2015.

[18] J. Carrasco, S. García, M. del Mar Rueda, and F. Herrera, "rNPBST: An R Package Covering Non-parametric and Bayesian Statistical Tests," in *Hybrid Artificial Intelligent Systems: 12th International Conference, HAIS 2017, La Rioja, Spain, June 21-23, 2017, Proceedings* (F. J. Martínez de Pisón, R. Urraca, H. Quintián, and E. Corchado, eds.), pp. 281–292, Cham: Springer International Publishing, 2017.

[19] J. Derrac, S. García, and F. Herrera, "JavaNPST: Nonparametric Statistical Tests in Java," *ArXiv e-prints*, Jan. 2015.

[20] N. Hamilton, *Ggtern: An Extension to 'Ggplot2', for the Creation of Ternary Diagrams*. 2018. R package version 2.2.2.

[21] E. Alpaydin, "Combined 5 × 2 cv F Test for Comparing Supervised Classification Learning Algorithms," *Neural Computation*, vol. 11, pp. 1885–1892, Nov. 1999.

[22] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2010.

Extracción de factores relevantes en el análisis de datos biomédicos: una metodología basada en técnicas de aprendizaje supervisado

Oscar Reyes
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: ogreyes@uco.es

Jose M. Moyano
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: jmoyano@uco.es

Antonio Rivero-Juárez
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: arjvet@gmail.com

Raúl M. Luque
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: raul.luque@uco.es

Antonio Rivero
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: ariveror@gmail.com

Justo Castaño
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: justo@uco.es

Sebastián Ventura
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: sventura@uco.es

***Resumen*—La determinación del conjunto de variables que se diferencian significativamente entre los grupos de muestras presentes en un estudio biomédico es una tarea que comúnmente se realiza mediante el análisis de cada variable individualmente y/o utilizando técnicas no supervisadas que no tienen en cuenta directamente el criterio de los expertos. En este trabajo se presenta una metodología basada en técnicas de aprendizaje supervisado para guiar el análisis de datos biomédicos, que permite la extracción de subconjuntos de factores relevantes para una correcta clasificación de las muestras en los grupos definidos a priori por los expertos. La metodología propuesta consta de dos fases principales, en la primera se determina la importancia de los factores, mientras que la segunda fase se enfoca en la búsqueda de subconjuntos de factores relevantes mediante la construcción de modelos precisos que logran clasificar correctamente las muestras. La utilidad de la metodología propuesta se ilustra mediante dos casos de estudios reales, mostrando que mediante la aplicación de la misma se podrían detectar relaciones complejas entre los factores, y que favorece el análisis de datos biomédicos que tienen un elevado número de variables descriptoras.**

I. Introducción

Las técnicas de aprendizaje no supervisado, como el análisis de componentes principales y los algoritmos de clustering, son ampliamente utilizadas en el campo de la bioinformática [1]. Sin embargo, en sentido general este tipo de técnicas no toma en cuenta el criterio de los expertos, que previamente al análisis pudieron haber clasificado las muestras en grupos (cáncer vs sano, tumor maligno vs tumor benigno, etc.), lo que puede implicar una perdida significativa de información para la extracción del conocimiento en el análisis de datos biomédicos.

Las técnicas de aprendizaje supervisado, por otro lado, permiten que el conocimiento aportado por los expertos pueda guiar el análisis de los datos, mostrándole a los algoritmos cuáles son las conclusiones (salidas) a la cuales deben llegar. Por ejemplo, un algoritmo de clasificación de imágenes para el diagnóstico del melanoma tratará de aprender las relaciones que vinculan a los datos contenidos en las imágenes con las etiquetas asignadas [2]. De esta manera, los algoritmos de aprendizaje supervisado permiten, dado unos datos de entrada, encontrar una función que produce una salida lo más aproximada posible al conocimiento de los expertos.

Una de las tareas que comúnmente se realiza en el análisis de datos biomédicos es la determinación del conjunto de variables que se diferencian significativamente entre los grupos de muestras definidos por los expertos [3]. Por ejemplo, el *p-value* calculado por el t-test es ampliamente usado como indicador de la relevancia de un factor (en lo adelante se usa el termino "factores" para indicar el conjunto de variables que describe las muestras de un problema). Sin embargo, además de que los test paramétricos no deben ser usados en todas las situaciones (este tema se escapa del objetivo de este trabajo), se debe considerar que de esta manera el análisis que se realiza es univariante, desechándose así las relaciones estadísticas que normalmente existen entre los factores de un problema.

Por otro lado, es de destacar que muchos problemas de

biomedicina implican el análisis de un número considerable de factores [4], lo cual hace que la tarea anterior sea inviable de realizar si antes no se han filtrado los factores que son realmente relevantes para el estudio del problema. Ejemplo de esto se encuentra al realizar estudios que involucran el análisis de las expresiones de genes sobre un conjunto de muestras.

En este trabajo se presenta una metodología, la cual está basada en técnicas de aprendizaje supervisado, que permite la extracción de subconjuntos de factores relevantes para una correcta clasificación de las muestras en las clases definidas por los expertos (en lo adelante se usa el término "clase" para indicar la variable que describe la condición por la cual los expertos agrupan las muestras). Esta metodología consta de dos fases principales: (a) la determinación de la importancia de los factores, que permite determinar un ranking de importancia; y (b) la construcción de modelos de clasificación a partir de dicho ranking. El uso de esta metodología puede aportar varios beneficios al análisis de datos biomédicos, ya que no solo se pueden determinar subconjuntos de factores relevantes que influyen en la correcta clasificación de las muestras, sino que los métodos desarrollados también son capaces de detectar distribuciones conjuntas entre factores, e interacciones y dependencias complejas respecto a las clases.

El resto del este trabajo se organiza de la siguiente manera. En la Sección II se describe la metodología, explicando cada una de sus fases. La aplicación de la metodología propuesta se ilustra en la Sección III mediante dos casos de estudio reales, uno relacionado con el diagnóstico de tumores neuroendocrinos pulmonares y el otro con el aclaramiento espontáneo en Hepatitis C. Finalmente, en la Sección IV se presentan las conclusiones del presente trabajo.

II. Metodología

El esquema general de la metodología que se propone se muestra en la Figura 1. El preprocesamiento de los datos es un paso opcional, que no nos detendremos a analizar en profundidad en este trabajo. Sin embargo, hay que destacar que generalmente la calidad de los resultados en el análisis de datos biomédicos depende en gran medida de que se haya hecho un correcto preprocesamiento de los datos [5]. El preprocesamiento de datos abarca una amplia gama de métodos, que van desde la eliminación de outliers y la estimación de valores perdidos hasta el centrado, escalado y transformación de los datos. El uso de cada uno de los métodos de preprocesado debe tener una lógica y justificación correcta, ya que si bien es cierto que un correcto preprocesado de datos puede mejorar significativamente el análisis, también un preprocesamiento incorrecto puede conllevar a la obtención de conclusiones erróneas.

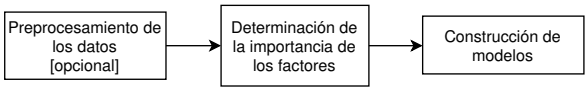


Figura 1. Esquema general de la metodología.

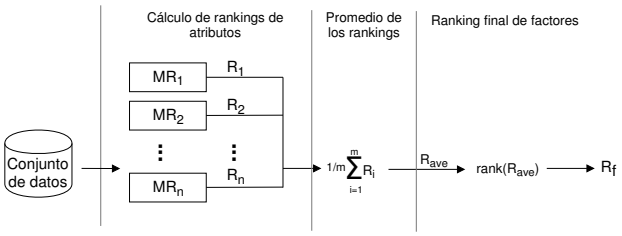


Figura 2. Cálculo del ranking final de factores.

II-A. *Determinación de la importancia de los factores*

La primera fase de la metodología se enfoca en la determinación de la importancia de cada uno de los factores del problema, y para ello se propone el uso de algoritmos supervisados de pesado de atributos o *feature weighting* (FW) [6]. La relevancia de un factor se determina mediante la asignación de un peso que representa la información que tiene este para la correcta separación de las muestras en las clases definidas por los expertos [7]. Un método de FW le asigna a cada factor un peso, siendo posible de esta manera obtener un ranking de factores directamente. El objetivo final de esta fase de la metodología es calcular un ranking donde están ordenados de mayor a menor importancia todos los factores.

Digamos que disponemos de m métodos de FW para lograr una mejor estimación del ranking final de factores. R_i representa el ranking calculado por el método i-ésimo, $R_i(f)$ representa el valor del factor f en el ranking R_i, y F es el conjunto de todos los factores existentes en el estudio. El ranking final de factores se calcula de la siguiente manera:

$$R_f{=}rank\left(\frac{1}{m}\sum_{i=1}^{m} R_i(f) : \forall f \in F\right), \qquad (1)$$

donde la función $rank(...)$ calcula el ranking final de factores a partir de los valores promedios de cada uno de los factores en los m ranking iniciales. La Figura 2 representa el cálculo del ranking final de factores.

Respecto a la cantidad de métodos de FW a utilizar en la estimación, cuanto mayor sea el número de métodos, más precisa será la estimación del ranking final. En este sentido, se recomienda el uso de métodos supervisados de FW que sean independientes de un clasificador para estimar la importancia de un factor, evitando de esta manera la introducción de sesgos y dependencias en el proceso de estimación. En su lugar se propone el uso de métodos de FW que calculen directamente medidas sobre los datos, como medidas de distancia, entropía o correlación. Estos métodos son conocidos en la literatura especializada como métodos filtros, y entre los más populares podemos encontrar a *Correlation Attribute Evaluation* [8] *Gain Ratio* [9], *Information Gain* [10] y *ReliefF* [11].

Es importante resaltar que para lograr una estimación precisa de la importancia de los factores, es necesario que cada uno de los m métodos de FW sean ejecutados mediante algún proceso de validación cruzada, el cual dependerá del tamaño del conjunto de datos analizado. Normalmente una validación

cruzada de 10 particiones repetidas varias veces es suficiente para lograr una buena estimación. Sin embargo, en el caso de que el conjunto de datos sea muy pequeño, se deberán considerar otras alternativas para la estimación, como una validación cruzada dejando uno fuera o *Leave One-out Cross Validation* (LOOC).

Por último, es importante destacar que en el ámbito de la biomedicina comúnmente la importancia de un factor se calcula agrupando las muestras por dos o más condiciones y se calcula la diferencia de este factor entre los diferentes grupos; por ejemplo el *p-value* calculado por el t-test es ampliamente usado como indicador de la relevancia de un factor. Sin embargo, además de que los test paramétricos no deben ser usados en todas las situaciones, se debe considerar que de esta manera el análisis que se realiza es univariante, desechándose así las relaciones estadísticas que normalmente existen entre varias variables descriptoras del problema. Esta característica principal es lo que distingue esta primera fase de la metodología propuesta en este trabajo. Los métodos filtros como *ReliefF*, son capaces de detectar distribuciones conjuntas entre variables, interacciones y dependencias complejas respecto a la clase, además de considerar como un todo el conjunto de factores F.

II-B. Construcción de modelos

Una vez estimado el ranking de factores, entonces se puede proceder a la determinación de los subconjuntos de factores que mejor logran predecir la clase añadida por los expertos. Sin embargo, esta no es una tarea fácil de realizar, ya que es complejo determinar un punto de corte a partir del cual los factores restantes se pueden considerar como irrelevantes para el análisis.

En lugar de realizar directamente un análisis sobre el ranking de factores R_f, en esta fase de la metodología se propone una búsqueda heurística guiada para encontrar el mejor subconjunto de factores; el método propuesto está inspirado en el algoritmo presentado por Reyes et al. [12]. En otras palabras, mediante esta fase se podrán determinar aquellos subconjuntos de factores a partir de los cuales se inducen modelos capaces de predecir efectivamente a qué clase pertenece cada muestra. La Figura 3 representa los pasos que sigue el algoritmo diseñado. Como puede observarse, es un proceso iterativo en el que, comenzando con el factor posicionado en el tope del ranking, en cada iteración se analiza si la inclusión del siguiente factor al subconjunto produce un mejor modelo. Finalmente el mejor subconjunto de factores será aquel sobre el cual se induce el mejor clasificador a lo largo de todas las iteraciones.

Para la comparación de la efectividad de los modelos se puede utilizar cualquier medida de evaluación, como el área bajo la curva ROC (AUC, por sus siglas en inglés), ampliamente usada en el análisis de datos biomédicos. Por otro lado, es de destacar que este procedimiento se puede realizar solamente considerando el ranking R_f o para cada sub-ranking R_f^g : $\forall g \in R_f$; el sub-ranking de factores R_f^g está compuesto por el factor g en el tope y todos los subsecuentes factores en R_f. El

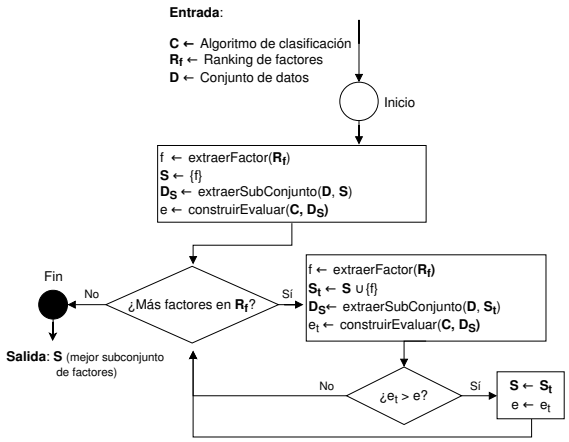


Figura 3. Búsqueda heurística del mejor subconjunto de factores.

primer caso claramente requeriría la construcción de un menor número de clasificadores ($O(|F|)$), mientras que la segunda opción requeriría la construcción de un número cuadrático de clasificadores ($O(|F|(|F|-1)/2)$); sin embargo, esta última opción es la que produce una mejor estimación. Además, al igual que en la fase anterior, es importante considerar que para lograr una estimación precisa del rendimiento de los clasificadores se debe utilizar un procedimiento de validación cruzada en el proceso de construcción de los mismos.

Respecto al criterio de parada del algoritmo, la Figura 3 ilustra un procedimiento que termina una vez que han sido evaluados todos los factores del ranking. Sin embargo, se pudieran definir otros criterios más flexibles que eviten evaluar completamente el ranking. Por ejemplo, los expertos pueden definir un umbral de aceptación de tal manera que el procedimiento se detenga a penas que se encuentre un modelo con un rendimiento superior a dicho umbral. Por otra parte, los expertos pueden estar interesados en no solo analizar el mejor modelo encontrado, sino los n mejores modelos construidos.

En esta fase de la metodología se puede utilizar cualquier algoritmo de clasificación para la construcción de los modelos, siempre y cuando este sea adecuado para el análisis. Por ejemplo, se puede usar cualquier algoritmo de clasificación binaria si solo se tienen dos posibles clases para las muestras, o cualquier algoritmo de clasificación multi-clase en caso de que se tengan más de dos clases. Por otra parte, es lógico que el rendimiento obtenido por los modelos dependerá de la efectividad y potencia que tenga el algoritmo de clasificación empleado; por ejemplo, es de esperar que un modelo de ensamblado como *Random Forest* [13] obtenga en promedio mejores resultados que otros modelos más sencillos como *KNN* [10] o *Naive Bayes* [10].

Por último, aclarar que también se pueden utilizar algoritmos de clasificación que tienen un proceso embebido de selección de atributos. En este último caso, es posible que el subconjunto de factores que finalmente utilice el modelo sea más pequeño que el subconjunto original sobre el cual se entrenó el algoritmo.

III. Casos de estudio

La metodología presentada en este trabajo es utilizada actualmente por varios laboratorios de investigación del Instituto Maimónides de Investigación Biomédica de Córdoba. A continuación se presentan dos casos de estudios reales que muestran la aplicación y utilidad de la propuesta.

III-A. Diagnóstico de tumores neuroendocrinos pulmonares

Los tumores neuroendocrinos pulmonares representan entre el 20 y el 30 % de todos los tumores neuroendocrinos [14]. La heterogeneidad, sus diferentes comportamientos clínicos, y la posibilidad de aparición recurrente y de hacer metástasis a largo plazo, enfatiza la importancia que tiene la identificación de nuevos marcadores de diagnósticos y terapéuticos, que pueden mejorar el diagnóstico, pronóstico y/o el tratamiento de los pacientes que sufren esta enfermedad [15].

Para este problema, los datos disponibles fueron de 26 muestras pareadas (muestras tumorales con su respectiva muestra de tejido normal adyacente), donde por cada muestra se tenía la expresión de 44 factores que regulan la maquinaria de *splicing*. El objetivo principal del estudio fue determinar subconjuntos de factores que caracterizaran claramente a las dos clases de muestras. En los datos originales no había datos perdidos, y en la etapa de preprocesamiento se eliminaron previamente aquellos factores que tenían varianza igual a cero, y además se centraron y escalaron los datos.

Mediante la primera fase de la metodología propuesta se obtuvo un ranking de factores que permitió determinar cuáles son en promedio los factores más relevantes para diferenciar las clases de muestras. La Figura 4 muestra las importancia de los 20 primeros factores del ranking.

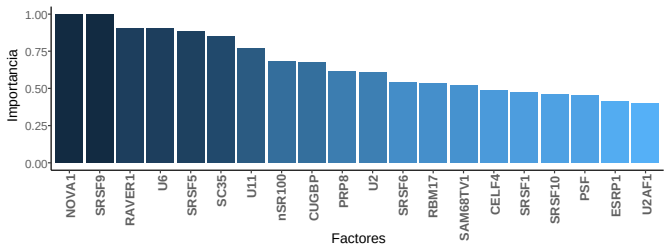


Figura 4. Ranking de factores para diferenciar entre muestras normales y tumorales.

Posteriormente, en la segunda fase de la metodología se utilizaron dos algoritmos de clasificación (*Logistic Regression* (LR) [16] y *Random Forest*) para la evaluación de subconjuntos de factores. Para la estimación de la precisión de los clasificadores se utilizó un procedimiento LOOC debido a que el número de muestras en el estudio era pequeño, y además se realizó una búsqueda *grid* de los mejores parámetros para el entrenamiento de los algoritmos. A partir de este análisis se encontraron 100 modelos con AUC mayor o igual a 0,85, arrojando subconjuntos de factores relevantes que aparecen generalmente en todos los modelos predictivos.

Por otra parte, aunque en la descripción de la metodología (véase Sección II) nos hemos limitado al uso de algoritmos de clasificación, es de destacar que en la segunda fase del procedimiento se podrían emplear algoritmos de clustering, siempre y cuando los resultados de agrupamiento sean analizados con medidas externas (como la pureza) que tienen en cuenta las clases definidas a priori por los expertos. De esta manera, a los efectos de la metodología el algoritmo de clustering empleado actuaría como si fuera un algoritmo supervisado. En este estudio, los resultados de la segunda fase de la metodología haciendo uso de un algoritmo de clustering jerárquico coinciden con los resultados obtenidos anteriormente por los algoritmos de clasificación, validando así la relevancia de los subconjuntos encontrados. La Figura 5 muestra un *heatmap* con uno de los subconjuntos de factores encontrados.

III-B. Aclaramiento espontáneo en Hepatitis C

Una vez que un paciente se infecta por el virus de Hepatitis C (VHC), se produce una hepatitis aguda que en la mayoría de los casos lleva a una infección crónica caracterizada por el avance gradual de fibrosis hepática, cirrosis y carcinoma hepatocelular [17]. Sin embargo, un porcentaje menor de pacientes resuelven su infección de manera espontánea. Por tanto, la identificación de factores o marcadores que ayuden a la predicción del aclaramiento espontáneo (AE) o infección crónica (IC) de VHC tendrían un alto impacto en la selección de la terapia que debería utilizarse para su tratamiento.

Para este problema, los datos disponibles fueron de 138 pacientes infectados con VHC, 81 de ellos con infección crónica y 57 en los que se produjo AE. Cada paciente estaba descrito por 43 marcadores distintos. En 43 muestras habían valores perdidos en algunos de sus marcadores, y se utilizó el algoritmo *knn-Imputation* [18] con $k = 3$ para estimar dichos valores.

A partir de la primer fase de la metodología, se obtuvo un ranking de factores, del cual en la Figura 6 se muestran los 20 primeros. De esta manera, se puede observar de manera simple la importancia de cada uno de los factores en el problema de VHC. El primer factor en el ranking tiene una importancia cercana a 1, lo que significa que en el proceso de estimación todos los métodos de FW le asignaron en promedio una alta importancia a dicho factor.

Posteriormente, en la segunda fase de la metodología se utilizaron varios clasificadores como *C4.5* [10] , *PART* [19], *Random Forest*, *Sparse Discriminant analysis* (*sparseLDA*) [20] y *Logistic Model Trees* (*LMT*) [21]. Para la ejecución de cada modelo se repitió 3 veces una validación cruzada en 10 particiones, evaluando en cada caso sobre el conjunto de *test* correspondiente, y promediando así los valores entre un total de 30 ejecuciones. Además, para la búsqueda de los parámetros de los algoritmos se realizó una búsqueda aleatoria de parámetros entre 30 combinaciones distintas.

Para este problema, se obtuvieron en total casi 400 modelos distintos con un AUC $> 0{,}8$; donde 126 tenían un AUC $> 0{,}85$; y 30 modelos con un AUC $> 0{,}87$. Posteriormente, para aquellos modelos con AUC $> 0{,}85$ se midió el número de veces que aparece cada uno de los atributos entre dichos

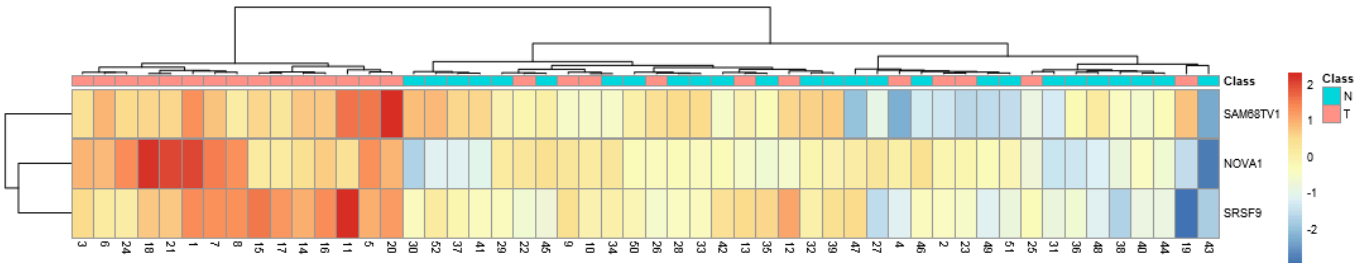


Figura 5. Heatmap generado a partir de un subconjunto de tres factores.

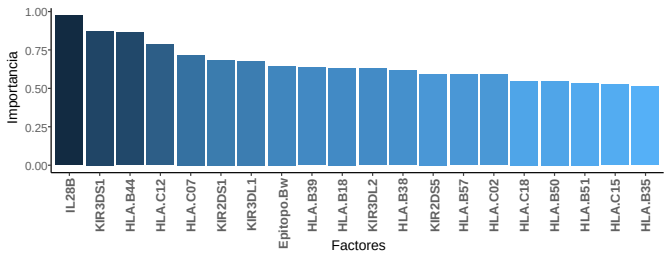


Figura 6. Ranking de factores para el problema de aclaramiento espontáneo en VHC.

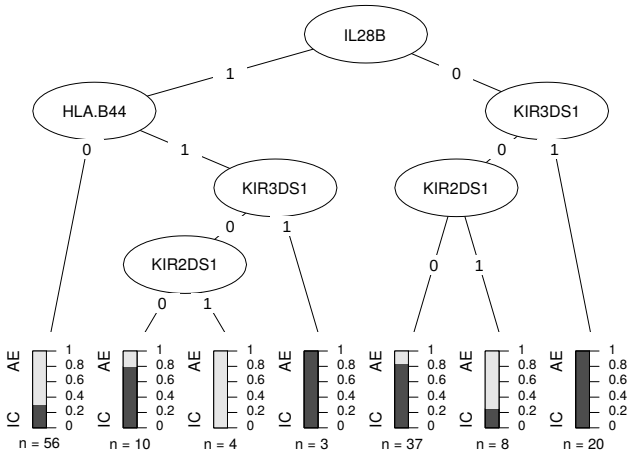


Figura 7. Árbol de decisión generado por uno de los modelos de C4.5 para el problema de VHC.

modelos. Esta medida proporciona otra aproximación de la importancia de cada uno de los factores en la predicción de la clase, pues un factor que aparezca en un gran número de modelos previsiblemente será más importante en la predicción de la clase que otro factor que aparezca con menor frecuencia.

Por otro lado, el hecho de utilizar modelos de árboles de decisión o reglas de asociación, como *C4.5* o *PART* respectivamente, hace que los modelos resultantes sean fácilmente interpretables por los expertos, pudiendo ver de manera sencilla cómo los factores discriminan para determinar si se predice una u otra clase. En la Figura 7 se muestra un ejemplo de los modelos de árbol obtenidos en el análisis. En la figura se observa como, para un nuevo paciente, dependiendo del valor de cada uno de los factores seleccionados, el modelo descenderá en el árbol hasta predecir la clase a la cual pertenece el paciente (nodo hoja). En estas hojas se puede observar cuál es el porcentaje de pacientes de cada una de las clases que cumplen las condiciones de los factores de los nodos superiores. Por ejemplo, para un paciente donde el factor *IL28B* valga 1 y el factor *HLA.B44* valga 0, el modelo asignará la clase AE, ya que en torno al 70 % de los pacientes observados con esa combinación de factores pertenecen a dicha clase. Cabe destacar también que, como se puede observar, los factores que aparecen en este modelo de árbol se encontraban en las primeras posiciones del ranking obtenido en la primera fase de la metodología, siendo la raíz del árbol precisamente el factor con mayor importancia en el ranking.

Por último, se comparan los resultados de los modelos generados mediante la metodología propuesta con modelos que son construidos considerando todos los factores del problema. La Tabla I muestra los resultados de esta comparación. Para cada

Tabla I
Mejora del rendimiento predictivo al ejecutar los distintos algoritmos utilizando todos los factores (AUC_0) o un subconjunto de los mismos determinados por la metodología propuesta (AUC_{sub}).

Algoritmo	AUC_0	AUC_{sub}	f_{sub}	% mejora
C4.5	0,766	0,842	10	9,92 %
PART	0,742	0,869	11	17,12 %
Random Forest	0,825	0,882	14	6,91 %
sparseLDA	0,839	0,880	8	4,89 %
LMT	0,803	0,872	7	8,59 %

uno de los algoritmos utilizados se muestra el valor de AUC obtenido utilizando los 43 factores del problema (AUC_0), y el mejor valor de AUC obtenido generando el modelo con subconjuntos de factores (AUC_{sub}); para este último caso, se indica además el número de factores utilizados para la construcción del modelo (f_{sub}). En la última columna de la tabla se incluye el porcentaje de mejora en rendimiento predictivo, calculado como $\frac{AUC_{sub}-AUC_0}{AUC_0}$. A partir de los resultados se puede observar como mediante la metodología propuesta se pueden obtener mejoras considerables en las predicciones de la clase; por ejemplo en el caso de *PART* se obtiene una mejora de un 17 % considerando solo 11 factores de los 43 existentes.

IV. Conclusiones

En este trabajo se ha propuesto una metodología para la extracción de factores relevantes en datos biomédicos mediante el uso de técnicas de aprendizaje supervisado. Dicha metodología se divide en dos partes principales, la creación de un ranking de factores que determine la importancia de cada uno de ellos, y la búsqueda de subconjuntos de factores que permitan construir modelos con una alta precisión para predecir el tipo de muestra. Mediante esta metodología es posible detectar relaciones complejas que existen entre los factores que describen a las muestras de un estudio, superando de esta manera el análisis de factores individuales que comúnmente se emplea en biomedicina.

La aplicación de la metodología se ilustró mediante dos casos de estudios reales, mostrando la utilidad y potencial de la misma. En estos problemas, gracias a la metodología propuesta se pudieron identificar subconjuntos de factores relevantes que permiten con una alta precisión clasificar las muestras en las clases definidas a priori por los expertos. Se espera que el uso de la presente metodología se pueda extender a otros grupos de investigación biomédica, facilitando el análisis de datos, así como la creación de biomarcadores para el tratamiento temprano de enfermedades patológicas.

Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2017-83445-P del Ministerio de Economía y Competitividad y Fondos FEDER. También ha sido financiado por la ayuda FPU del Ministerio de Educación FPU15/02948.

Referencias

[1] P. Baldi and S. Brunak, *Bioinformatics: the machine learning approach*. MIT press, 2001.

[2] I. Giotis, N. Molders, S. Land, M. Biehl, M. F. Jonkman, and N. Petkov, "MED-NODE: A computer-assisted melanoma diagnosis system using non-dermoscopic images," *Expert Systems with Applications*, vol. 42, no. 19, pp. 6578 – 6585, 2015.

[3] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[4] M. Gahete, M. del Rio-Moreno, E. Alors-Perez, O. Reyes, A. Camargo, J. Delgado-Lista, J. Lopez-Miranda, J. P. Castaño, and R. M. Luque, "Identification of an altered spliceosome-associated fingerprint as an early, predictive event for the development of type 2 diabetes in high-risk patients," in *100th Endocrine Society (ENDO) annual meeting*, 2018.

[5] R. A. van den Berg, H. C. J. Hoefsloot, J. A. Westerhuis, A. K. Smilde, and M. J. van der Werf, "Centering, scaling, and transformations: improving the biological information content of metabolomics data," *BMC genomics*, vol. 7, no. 1, p. 142, 2006.

[6] D. Wettschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 273–314, 1997.

[7] O. Reyes, C. Morell, and S. Ventura, "Evolutionary feature weighting to improve the performance of multi-label lazy algorithms," *Integrated Computer-Aided Engineering*, vol. 21, no. 4, pp. 339–354, 2014.

[8] M. A. Hall, "Correlation-based feature selection for machine learning," Tech. Rep., 1999.

[9] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2016.

[10] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[11] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Machine learning*, vol. 53, no. 1-2, pp. 23–69, 2003.

[12] O. Reyes, C. Morell, and S. Ventura, "Scalable extensions of the ReliefF algorithm for weighting and selecting features on the multi-label learning context," *Neurocomputing*, vol. 161, pp. 168–182, 2015.

[13] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[14] A. Fisseler-Eckhoff and M. Demes, "Neuroendocrine tumors of the lung," *Cancers*, vol. 4, no. 3, pp. 777–798, 2012.

[15] A. D. Herrera-Martínez, M. D. Gahete, R. Sánchez-Sánchez, R. O. Salas, R. Serrano-Blanch, A. Salvatierra, L. J. Hofland, R. M. Luque, M. A. Gálvez-Moreno, and J. P. Castaño, "The components of somatostatin and ghrelin systems are altered in neuroendocrine lung carcinoids and associated to clinical-histological features," *Lung Cancer*, vol. 109, pp. 128–136, 2017.

[16] S. K. Shevade and S. S. Keerthi, "A simple and efficient algorithm for gene selection using sparse logistic regression," *Bioinformatics*, vol. 19, no. 17, pp. 2246–2253, 2003.

[17] M. Frias, A. Rivero-Juárez, D. Rodriguez-Cano, A. Camacho, P. López-López, M. Risalde, B. Manzanares-Martín, T. Brieva, I. Machuca, and A. Rivero, "HLA-B, HLA-C and KIR improve the predictive value of IFNL3 for Hepatitis C spontaneous clearance," *Scientific Reports*, vol. 8, no. 1, p. 659, 2018.

[18] L. Torgo, *Data Mining with R, learning with case studies*. Chapman and Hall/CRC, 2010. [Online]. Available: http://www.dcc.fc.up.pt/ ltorgo/DataMiningWithR

[19] E. Frank and I. H. Witten, "Generating accurate rule sets without global optimization," in *Fifteenth International Conference on Machine Learning*, 1998, pp. 144–151.

[20] L. Clemmensen, T. Hastie, D. Witten, and B. Ersbøll, "Sparse discriminant analysis," *Technometrics*, vol. 53, no. 4, pp. 406–413, 2011.

[21] N. Landwehr, M. Hall, and E. Frank, "Logistic model trees," *Machine learning*, vol. 59, no. 1-2, pp. 161–205, 2005.

Aplicaciones de la técnica de *topic model* en repositorios software

Carlos López-Nozal
Área de Lenguajes y Sistemas Informáticos
Universidad de Burgos
Burgos, España
clopezno@ubu.es

César Ignacio García-Osorio
Área de Lenguajes y Sistemas Informáticos
Universidad de Burgos
Burgos, España
cgosorio@ubu.es

Álvar Arnaiz-González
Área de Lenguajes y Sistemas Informáticos
Universidad de Burgos
Burgos, España
alvarag@ubu.es

Mario Juez-Gil
Área de Lenguajes y Sistemas Informáticos
Universidad de Burgos
Burgos, España
mariojg@ubu.es

***Resumen*—Los equipos de desarrollo software usan repositorios de proyectos, accesibles mediante forjas como Github, donde aplican un proceso iterativo e incremental. En el proceso se incluyen actividades como: gestión de tareas, gestión de versiones, codificación, pruebas, documentación, revisiones de calidad y despliegue de aplicaciones. Durante estas actividades se generan múltiples productos a través de complejas interacciones entre sus participantes. La caracterización tanto de productos como de interacciones basadas en texto es un primer paso para comprender y hacer más eficiente la gestión del proyecto. La técnica de aprendizaje textual denominada modelo de temas (*topic model*), mejora la comprensión de grandes cantidades de datos textuales agrupando los documentos en temas. Con el objetivo de poder mejorar las actividades de un proceso de desarrollo software, en este trabajo se presenta una revisión bibliográfica de cómo se está aplicando la técnica de aprendizaje automático y cuáles son los resultados de su aplicación en el contexto del proceso de desarrollo software.**

***Index Terms*—Model Topic, Software Repository, Text Mining, Software Development, Machine Learning**

I. Introducción

En la última década han surgido forjas de proyectos software de fácil acceso tanto para proyectos empresariales como para proyectos *OpenSource* (SourceForge Github, GitLab, Bitbucket). Estas forjas suelen integrar múltiples sistemas para dar soporte a los flujos de trabajo y registrar las interacciones entre los miembros del equipo.

Debido al interés que ha surgido en este campo, la comunidad científica creó en 2004 la Conferencia *Internacional Mining Software Repositories* (MSR http://www.msrconf.org/) como un punto de encuentro para analizar los datos disponibles en los repositorios software para descubrir información interesante y procesable sobre sistemas de software y proyectos.

Una de las líneas de trabajo en el aprendizaje automático es el tratamiento de grandes cantidades de documentos basados en texto para poder extraer relaciones entre ellos. El modelado de temas (*topic model*) es una técnica de agrupamiento de documentos de texto (*clustering*). Cada tema se define siguiendo una distribución de probabilidades de un conjunto de palabras que ocurren con más frecuencia en los documentos de ese tema.

Muchas actividades de los repositorios software llevan implícita una comunicación textual entre los miembros del equipo. Por ejemplo, el análisis de la información textual de las revisiones utilizando la técnica de modelado de temas puede mejorar la comprensión del proceso de calidad.

El objetivo de este trabajo es estudiar en la literatura científica cómo se está aplicando la técnica de modelado de temas, en los conjuntos de datos extraídos desde repositorios software.

El resto del artículo se estructura de la siguiente forma. En la Sec. II, se definen los conceptos teóricos necesarios para la comprensión del artículo. En la Sec. III, se detalla el proceso utilizado para seleccionar la bibliografía a revisar. Posteriormente, en la Sec. IV se resumen los resultados de la revisión bibliográfica. Finalmente, en la Sec. V, se presentan las conclusiones obtenidas del presente trabajo.

II. Definición de conceptos teóricos

La sección se ha organizado en dos subsecciones, una para los conceptos relacionados con la ingeniería de software y otra para los conceptos relacionados con ciencias de la computación y procesamiento de lenguaje natural.

II-A. Sistemas utilizados en el desarrollo software

Uno de los resultados del proceso interno de un equipo de desarrollo de software, es crear un conjunto de ficheros de texto que contienen un código que se puede desplegar y ejecutar en algún dispositivo electrónico. El código puede estar escrito en múltiples lenguajes de programación. Para gestionar todo este proceso de creación se usan varios sistemas

de gestión integrados en lo que se denomina repositorio del proyecto. A continuación se da una definición de repositorio de software y se describen alguno de sus sistemas gestión.

- Repositorio software. Los repositorios software son espacios virtuales donde los equipos de desarrollo generan los artefactos colaborativos procedentes de las actividades de un proceso de desarrollo. Los repositorios se alojan en lo que se denominan forjas de proyectos software (*GitHub, GitLab, SourceForge, Bitbucket*) En los repositorios además de guardar los artefactos, versión final y versiones previas, se almacena la interacción de los miembros del equipo justificando el cambio de versión. Dependiendo del artefacto generado se utilizan distintos sistemas: foros de comunicación, sistemas de control de versiones, sistemas de gestión de tareas.
- Sistema de control de versiones. Los *commits* son una de las entidades clave de estos sistemas. Cada cambio de versión de un fichero, o conjunto de ficheros, generado en el repositorio se conoce como *commit*. La evolución del software puede representarse como una secuencia de *commits*. Cada commit es creado por un miembro del equipo de desarrollo para registrar un cambio en el software. Además de los detalles de los cambios efectuados en los ficheros, un elemento importante en los *commits* es la información textual que incluye el desarrollador para describir la naturaleza del cambio (mensaje asociado al *commit*). Los repositorios están compuestos por ramas de desarrollo y estas contienen *commits*. Las ramas pueden estar en repositorios centrales, distribuidos en múltiples repositorios locales o remotos. La rama de desarrollo principal es la que mantiene los ficheros actualizados para un versión del software que está en explotación.
- Sistema de gestión de tareas. Cualquier actividad que se va a realizar en un repositorio debería ser notificada con una tarea (el término tarea también es referido como *issue* o *ticket*). De manera general, las actividades en un proceso de desarrollo de software se pueden categorizar en: desarrollo (implementación de nueva funcionalidad), documentación, mantenimiento (pruebas, reestructuración de código, corrección de errores), explotación (despliegue de la aplicación) y revisiones de calidad. Los elementos fundamentales de una tarea son: el texto descriptivo de la tarea, un estado que indica si está abierta o cerrada, opcionalmente puede tener etiquetas que ayuden a su clasificación. Además cada tarea tiene asociado un responsable y también tiene asociado el conjunto de comentarios textuales de los distintos miembros del equipo, desde su apertura hasta su cierre. Cada tarea tiene un identificador que sirve como componente integrador con el resto de sistemas. Por ejemplo el texto descriptivo de un *commit* puede incluir el identificador de la tarea para recoger una trazabilidad de la descripción de los cambios ocasionados por la tarea. Las peticiones de integración (*pull request*) son un tipo de tareas especiales que se crean automáticamente en el sistema de gestión de tareas cuando un miembro del equipo solicita una integración de sus cambios en la rama de desarrollo principal. Las integraciones pueden ser aceptadas o rechazadas después de un proceso/diálogo de discusión y revisión entre los miembros del equipo.

II-B. Agrupamiento y procesamiento de documentos

- Modelado de temas o *topic modeling*, es una técnica avanzada de recuperación de información que automáticamente encuentra los temas generales de en un conjunto de documentos de texto, llamado corpus, sin la necesidad de etiquetas, datos de entrenamiento o taxonomías predefinidas. El modelado de temas solo usa la frecuencias de las palabras y la co-ocurrencia de frecuencias en los documentos para construir un modelo de palabras relacionadas. Utilizando este enfoque simple, el modelado de temas se ha utilizado con éxito en múltiples dominios para organizar y analizar automáticamente millones de documentos no estructurados.
  En la actualidad existen varios algoritmos que sirven para implementar esta técnica, siendo los más referenciados LDA (Latent Dirichlet Allocation) [4], LSI (Latent Semantic Indexing) y HDP (Hierarchical Dirichlet Process). A continuación formalizamos matemáticamente este concepto. Dado un corpus de documentos $D = \{d_1, ..., d_n\}$ donde cada documento $d_i, i = 1, ..., n$ es una secuencia de m palabras denotadas por $d_i = \{w_1, ..., w_m\}$, $w_j \in W, j = 1, ..., m$. W es el vocabulario del conjunto de documentos. Cada documento d_i se puede modelar como una distribución multinomial θ^{d_i} sobre t temas, y cada tema z_k, $k = 1, ..., t$ se modela como una distribución multinomial ϕ^k sobre el conjunto de palabras W.
- Caracterización de documentos de texto. La técnica de modelado de temas necesita caracterizar cada documento $\{d\}$ con conjunto de valores de entrada $\{x_1, ..., x_n\}$. La bolsa de palabras (*Bag of Words*) es la aproximación más utilizada para representar documentos de texto. En esta representación cada palabra del conjunto de documentos es considerada como una característica x_j. Existen varios tipos de transformaciones:
  - Boolean: es la más simple de las transformaciones. Si una palabra w_j está presente en el documento, la característica x_j toma el valor de 1 y 0 en caso contrario.
  - Raw TF (*Term Frequency*): en esta transformación, la característica x_j toma como valor el número de ocurrencias de la palabra, w_j, en el documento d_i y se referencia como $f_{i,j}$. Aunque parece una representación más sofisticada, tiene la desventaja que palabras con poco significado a menudo tienen valores de frecuencias muy altos.
  - Escala logarítmica TF: esta versión de TF se usa para suavizar la desventaja mencionada de la representación Raw TF. Existen varias transformaciones logarítmicas, una de las más usada se calcula aplicando

Cuadro I
Ejemplo de tareas de procesamiento de lenguaje natural.

.token	.lemma	.tag	.es_alfabético	.palabra vacía
Updates	update	NOUN	True	False
link	link	VERB	True	False
text	text	NOUN	True	False
and	and	CONJ	True	True
URLs	url	NOUN	True	False
in	in	ADP	True	True
other	other	ADJ	True	True
docs	doc	NOUN	True	False

$\log(1 + f_{i,j})$ para cada palabra w_j en el documento d_i.

- IDF (*Inverse Document Frequency*): IDF asigna el valor $f_{i,j} \log(N/N_j)$, donde N es el número total de documentos del corpus, y N_j es el número de documentos que contienen la palabra. Aplicando esta trasformación, una palabra tiene más peso si es infrecuente en el corpus de documentos.
- TF-IDF: las transformaciones TF y IDF se multiplican. De esta forma se incrementa el valor de la característica cuando una palabra es frecuente en el documento pero infrecuente en el corpus de documentos.

- Procesamiento del lenguaje natural. La extracción de palabras de los documentos se obtiene aplicando una secuencia variable de tareas de procesamiento entre las que se pueden incluir las siguientes: *tokenización*, eliminación de palabras sin significado, selección de términos gramaticales (sustantivo, verbo, adjetivo, adverbio), identificación de secuencias de palabras que se utilizan juntas y lematización.
  La *tokenización* es la tarea encargada de dividir un texto en *tokens*. Cada token es una palabra contenida en el texto y está separado con delimitadores, normalmente son espacios en blanco y signos de puntuación. En el lenguaje existen palabras vacías que no aportan significado, como puede ser las preposiciones, artículos, etc. Se suele disponer de una lista de palabras vacías asociadas a cada idioma, generalmente denominada *stop words*. El proceso de identificación sintáctica de las palabras en una frase del documento se denomina etiquetado gramatical (conocido también por su nombre en inglés, *part-of-speech tagging, POS tagging o POST*). El lematizado es una forma de normalización de la palabra, reduce una palabra a su forma base, raíz o lema.
  En el Cuadro I se muestra un ejemplo ilustrativo de los resultados de aplicar las tareas de procesamiento de texto: tokenización, identificación de palabras vacías, etiquetado gramatical y lematización. El ejemplo es aplicado para procesar la siguiente entrada de texto "*Updates link text and URLs in other docs"*. Las columnas se corresponden con las salidas de las tareas del preproceso (token, lema, tag, stop, es alfabético) y las filas son tokens de la frase de entrada.

III. Proceso de selección de artículos

En esta sección se describe cuál ha sido el proceso de búsqueda y selección de artículos que sirvan para comprender cómo se está utilizando la técnica de modelado de temas en los repositorios software.

Se utiliza la base de datos Scopus para seleccionar los documentos de interés a revisar. Inicialmente la búsqueda con la palabra clave "GitHub", se hace en el título, resumen y palabras clave de los artículos. El número de artículos encontrados con este criterio de búsqueda fue de 3,697. En la siguiente iteración de búsqueda se incluye en la cadena de búsqueda "LDA" (*Latent Dirichlet Allocation*), por ser las siglas del algoritmo de modelado de temas más utilizado. Con este refinamiento del criterio de búsqueda, el número de documentos fue de 18. Posteriormente se refina la búsqueda incluyendo otros repositorios de proyectos y otros algoritmos de modelado de tópicos. Finalmente se obtienen 30 artículos siendo las principales áreas de conocimiento *Computer Science* (24) y *Mathematics* (5).

Como última etapa del proceso de la búsqueda sistemática, se realizó una lectura de los resúmenes y palabras clave para poder verificar la validez del artículo respecto a los criterios de búsqueda. En esa inspección se encontraron varios falsos positivos, por un lado debido a la aparición de la palabra reservada Github como url utilizada para distribuir una determinada implementación de software en un repositorio. Por otro lado, por el uso del término "lda" en la sentencia "*Science and Technology Publications, Lda*". Como resultado se obtuvieron ocho artículos, dos de revistas y seis de actas de congresos internacionales.

Además de este proceso de búsqueda sistemática, se realizó una búsqueda no sistemática utilizando múltiples criterios con términos relacionados: *machine learning*, *text mining*, *text clustering*, *mining software repositories*, ... Como resultado de este proceso se incluyeron tres artículos más de revistas.

El Cuadro II contiene una descripción de las referencias bibliográficas de estudio ordenadas por método de búsqueda (sistemático vs. no sistemático) y después por año de publicación.

IV. Revisión bibliográfica

Para facilitar la comprensión de esta revisión, en las siguientes subsecciones caracterizamos todos los trabajos seleccionados desde tres perspectivas: descripción del conjunto de datos que utilizan, aplicación del modelo de temas, técnicas de procesamiento del lenguaje natural.

IV-A. Descripción de los conjuntos de datos

En el Cuadro III se muestran los resultados de la caracterización para los que se han considerado las siguientes cinco características:

- CD1 Tipo de entidades, es un medida nominal que identifica los elementos de análisis dentro del repositorio software. Puede tomar los siguientes valores: *pull request, commit, source code, fichero Readme, issue, post*. En la Subsec. II-A se han explicado *pull request,*

Cuadro II
Artículos seleccionados para la revisión

Título	Tipo	Año	Referencia
What are developers talking about? An analysis of topics and trends in Stack Overflow	Revista	2014	[3]
Open source is a continual bugfixing by a few	Actas congreso	2014	[5]
An insight into the pull requests of GitHub	Actas congreso	2014	[12]
Mining source code topics through topic model and words embedding	Actas congreso	2016	[17]
Topic-Based Integrator Matching for Pull Request	Actas congreso	2017	[8]
Cataloging GitHub repositories	Actas congreso	2017	[13]
Developer Identity Linkage and Behavior Mining Across GitHub and StackOverflow	Revista	2017	[15]
Mining developer behavior across git hub and stack overflow	Actas congreso	2017	[16]
Mining software repositories for defect categorization	Revista	2015	[7]
MSR4SM: Using topic models to effectively mining software repositories for software maintenance tasks	Revista	2015	[14]
Understanding Review Expertise of Developers: A Reviewer Recommendation Approach Based on Latent Dirichlet Allocation	Revista	2018	[6]

Cuadro III
Resumen de las características de los conjuntos de datos experimentales de la bibliografía.

BIB	CD1	CD2	CD3	CD4	CD5
[3]	post	3 447 987	7 meses actividad	SI	StackOverFlow
[5]	commits, issues	NO	43	NO	2014 MSR Challenge
[12]	pull request	9 421	78	NO	2014 MSR Challenge
[17]	source code	NO	100	SI	2013 MSR Challenge
[8]	pull request	4 364	3	NO	Github
[13]	fichero Readme	10 000	10 000	SI	Github
[16]	issue y post	16 000	No especificado	SI	Github StackOverFlow
[7]	Issue-bug	2 500	4	NO	OpenSource
[14]	post, commit, bugs	NO	3	NO	OpenSource
[6]	pull request	1 345	5	NO	Github

commit, source code e issue. Por *post* se entiende el mensaje y respuestas enviadas por los desarrolladores en un foro de tipo pregunta respuesta, como *StackOverFlow*. Los ficheros *Readme* de los repositorios de GitHub, son ficheros de texto que sirven para describir el contenido del repositorio, son una página de presentación del repositorio.

- CD2 Número de entidades o documentos de texto en la validación empírica del algoritmo de modelado de temas.
- CD3 Número de repositorios incluidos en el diseño experimental.
- CD4 Uso de entidades de múltiples repositorios es una medida booleana. En el caso de ser cierta, indica que en el diseño experimental utiliza las entidades de múltiples repositorios mezcladas entre sí. En [3] toma el valor de cierto, porque extrae los temas asociados a los post de la plataforma StackOverFlow durante un periodo de siete meses de actividad. En [17] se agrupan por temas ficheros fuentes de múltiples repositorios y en [13] se analizan los ficheros *Readme* de múltiples repositorios para extraer sus temas automáticamente.
- CD5 Disponibilidad de acceso a los conjuntos de datos utilizados en el validación empírica es una medida nominal. Github y StackoverFlow indican que en los trabajos se utilizada un API (*Application Program Interface*) pública para acceder a datos de estos repositorios. *MSR Challenge* (*Mining Software Repositories*) son los conjuntos de datos utilizados para una sesión de desafío del conferencia internacional MSR. *OpenSource* se ha utilizado para categorizar los trabajos con información de repositorios de tipo *OpenSource*.

IV-B. Aplicación del modelo de tópicos

En la Tabla IV se muestran los resultados de la caracterización para los que se han considerado las siguientes seis características:

- AMT1 Algoritmo es una medida nominal con el nombre del algoritmo de modelado de temas. En la bibliografía existe varias implementaciones distintas del algoritmo LDA, se han identificado con un número, 1 para la implementación conocida como MALLET (http://mallet.cs.umass.edu/) y 2 para la implementación conocida como JGIBBLDA (http://jgibblda.sourceforge.net/). LDA-GA indica una combinación del uso de LDA con algoritmos genéticos (*Genetic Algorithms*) para la optimización de sus parámetros en un conjunto de datos concreto. EmbTE *Embedded Topic Extraction*, se corresponde con la solución particular presentada [17], al igual que SDCL *Software Defect CLustering* es la propuesta de solución en [7].
- AMT2 Iteraciones es un parámetro del algoritmo que sirve para determinar la condición de parada del algoritmo y convergencia de los resultados. Es un parámetro opcional.
- AMT3 Número de tópicos/temas es un parámetro necesario por el algoritmo LDA.
- AMT4 Etiquetado manual es una medida booleana. En el caso de ser cierta indica que los temas, probabilidades de conjuntos de palabras, obtenidos como salida del algoritmo son verificados manualmente por una persona. El objetivo de la verificación es mejorar la comprensión humana asignando una única palabra clave al conjunto palabras obtenidas con el algoritmo. Por ejemplo en el etiquetado manual de [3] se asigna "*SQL*" como palabra clave del tema cuyas cuatro palabras más probables son "*quer*", "*table*", "*sql*" y "*row*". El proceso de etiquetado manual, implica determinar el número de palabras con las que establecer la palabra clave (por ejemplo en [12] usan 4). Además, hay que tomar decisiones sobre cuáles de los temas obtenidos como salida del algoritmo son coherentes. En [13] durante las inspecciones : *i*) asignan una etiqueta a un tema, *ii*) eliminan un tema por no ser coherente y *iii*) fusionan varios temas en uno.
- AMT5 Validación es una medida booleana que indica si en el diseño experimental se incluye alguna calibración

Cuadro IV
Resumen de las características de la aplicación del algoritmo de modelado de la bibliografía.

BIB	AMT1	AMT2	AMT3	AMT4	AMT5	AMT6
[3]	LDA1	500	40	SI	NO	NO
[5]	LDA1	1000	50	NO	NO	NO
[12]	LDA2	3000	100	SI	NO	NO
[17]	LDA,EmbTE		NO	NO	SI	IDF -TF y TF
[8]	LDA2	1000	15	NO		NO
[13]	LDAGA	500	49	SI	SI	IDF-TF
[16]	LDA	NO	NO	NO	NO	NO
[7]	SDCL			NO		IDF-TF
[14]	LDAGA	NO	NO	NO	SI	NO
[6]	LDA1	NO	20	NO	NO	NO

Cuadro V
Resumen de las características de la aplicación del algoritmo de modelado de la bibliografía.

BIB	TP1	TP2	TP3	TP4	TP5	TP6
[3]	ALTO	SI	SI	NO	NO	2-grams
[5]	MEDIO	SI	NO	SI	NO	NO
[12]	BAJO	SI	SI	NO	NO	NO
[17]	MEDIO	SI	SI	SI	SI	NO
[8]	BAJO	SI	SI	NO	NO	NO
[13]	ALTO	SI	SI	NO	NO	NO
[16]	BAJO	SI1	SI	SI	NO	NO
[7]	BAJO	SI	SI	NO	NO	NO
[14]	MEDIO	SI	NO	NO	NO	NO
[6]	MEDIO	SI	SI	NO	NO	NO

de los parámetros del algoritmo. Por ejemplo en [13] se utilizan medidas de coherencia de los temas obtenidos con LDA para determinar el parámetro de número de tópicos óptimo. La utilización de algoritmos genéticos para la optimización de parámetros de LDA es una solución presentada en varios trabajos [13], [14].

- AMT6 Transformación del corpus es una medida nominal que indica la representación numérica del conjunto de documentos. Es uno de los parámetros de entradas obligatorios del algoritmo ya que no puede configurarse con un valor por defecto. Los posibles valores son los presentados en Subsc. II-B: TF, IDF-TF.

IV-C. Técnicas de procesamiento del lenguaje natural

En la Tabla V se muestran los resultados de la caracterización para los que se han considerado las siguientes seis características:

- TP1 Tokenización es una medida ordinal que puede tomar los valores BAJO, MEDIO y ALTO. BAJO hace referencia cuando el proceso de división de textos en palabras se basa únicamente en utilizar espacios en blanco y signos de puntuación como separadores. MEDIO cuando utiliza un sistema de tokenización especial relacionado con el tipo de documento. AVANZADO cuando realiza múltiples sistemas de tokenización especiales. Para aclarar el concepto de sistema de tokenización especial se presentan algunos casos concretos. En [13] se elimina texto de cabeceras de los ficheros *Readme* para quedarse sólo con aspectos funcionales, se eliminan texto relacionados con licencias, instalación... Cuando los documentos a tratar contienen código de programación, como en [6] y [17], se realiza un tipo de procesamiento especial denominado *camel case splitting* para dividir identificadores de las entidades de código que están compuestos por varias palabras, por ejemplo "*nameToIndex*", "*loanInterest*" o "*hasDupdName*". En [14] se eliminan identificadores de usuario dentro del cuerpo de los documentos, típicamente precedidos por el símbolo @.
- TP2 Palabras vacías es una medida booleana para indicar que se ha incluido esta tarea en el procesamiento del lenguaje. En [17] y en [6] se añaden las palabras que forman parte del lenguaje de programación (*if*, *implements*, *class*...).
- TP3 Lematización es una medida booleana para indicar que se ha incluido la tarea en el procesamiento del lenguaje. El tipo de algoritmo empleado mayoritariamente es el de Porter.
- TP4 Filtrado de palabras en el corpus es una medida booleana para indicar que se ha incluido esta tarea en el procesamiento del lenguaje. Los valores umbrales de las frecuencia de palabras en el corpus añaden poca información relevante para categorizar. En [5] se eliminan cadenas de palabras muy frecuentes en el texto de las *issues*: "*good job*", "*thank you*". En [17] elimina palabras muy frecuentes en los identificadores de entidades de programación como por ejemplo "*set*". En [15] se aplica una estrategia más general eliminando todas las palabras que tienen una frecuencia inferior a 20 en el conjunto de documentos.
- TP5 Etiquetado sintáctico es una medida booleana para indicar que se ha incluido esta tarea en el procesamiento del lenguaje. Además del etiquetado de lenguaje natural explicado en Subsec. II-B se considera etiquetado sintáctico la identificación de entidades de código: clase, método, parámetro, etc. que se aplica en [17] sobre los documentos de texto basados en código fuente.
- TP6 Identificación de secuencia de palabras que se usan juntas es una medida booleana para indicar que se ha incluido esta tarea en el procesamiento del lenguaje. En [3] se muestra el siguiente ejemplo de aplicación de está técnica "*compile time error*", tiene tres uni-grams "*compile*", "*time*", "*error*" y 2-grams ("*compile_time*", "*time_error*"). Para aplicar está técnica configura un parámetro de la implementación MALLET del algoritmo LDA, llamado *gram-size*.

V. Conclusiones

En la actualidad, empieza a existir un interés en aplicar la técnica de modelado de temas en las actividades de los sistemas de los repositorios software. Su objetivo es mejorar las tareas de mantenimiento del software caracterizando las entidades de interacción textual. Esta observación está basada en que la búsqueda sistemática de bibliografía ha localizado diversos artículos comprendidos entre las fechas 2014 y 2018.

En la revisión bibliográfica, la aplicación del modelado de temas ha sido motivada por diferentes causas. Una de ellas ha sido la aplicación de manera directa sobre alguna entidad textual de los repositorios, con el objetivo de mejorar la comprensión. Otra motivación es utilizar las salidas del algoritmo de modelado de temas como entrada de otras técnicas de aprendizaje automático.

Respecto a la caracterización de los conjuntos de datos experimentales del Cuadro III, se observa que el modelado de temas se aplica sobre múltiples entidades de los repositorios (CD2), y que el número de entidades varía mucho, desde 3,447,987 hasta 1,345. Los valores altos de CD2 se corresponden con trabajos donde se aplica el modelado de temas con entidades de múltiples repositorios (CD4). Esta correlación también ocurre con el número de repositorios (CD3, CD4). Algunos de los trabajos que realizan diseños experimentales aplicando LDA sobre un único repositorio, eliminan inicialmente algunos repositorios por falta de número de entidades y de contenido textual para experimentar. El acceso a los conjuntos de datos es abierto, pero necesitan eliminar ruido para mejorar la experimentación. Una carencia importante observada es que no hay ningún trabajo que experimente sobre datos de empresa que no procedan de proyectos *OpenSource*.

Respecto a la caracterización de cómo se aplica el algoritmo de modelado de temas del Cuadro IV, se observa que LDA es un algoritmo de referencia en el modelado de temas, pero la configuración de sus parámetros varía entre experimentos. Este es el caso del número de iteraciones (AMT2) y número de temas (AMT3). Además, otros parámetros rara vez se incluyen en el artículos, este es el caso de la transformación del corpus utilizada (AMT6). La transformación del corpus predominante en lo trabajos es IDF-TF. En el 30 % de los trabajos, la salida de LDA es supervisada por una persona para garantizar la coherencia de los temas obtenidos. Solo el 30 % de los trabajos validan empíricamente que los parámetros elegidos para la ejecución del algoritmo tiene un desempeño óptimo (AMT5). Una tendencia actual es validar estos parámetros utilizando algoritmos genéticos sobre diferentes ejecuciones del algoritmo con distintos parámetros. En este sentido en la bibliografía se han identificado dos nuevas fuentes bibliográficas [11] y [1] que detallan como calibrar los parámetros de LDA.

Respecto a la caracterización de técnicas del lenguaje natural recogidas en el Cuadro V, se observa que las técnicas clásicas de eliminación de palabras vacías (TP1) y lematización (TP3) con el algoritmo de Porter son especificadas en el 100 % de los trabajos. Otras técnicas documentadas que mejoran la calidad de la representación del documentos como: filtrado de términos por frecuencia en el corpus documentos (TP4), análisis de elementos sintácticos (TP5) e identificación de secuencia de palabras (TP6) son aplicadas en el 30 % de los artículos, 10 % y 10 % respectivamente.

Como conclusión final se considera que el modelado de temas aplicado a tareas de mantenimiento del software tiene una proyección prometedora, pero su aplicación necesita madurar. En este sentido, en los artículos revisados no se ha encontrado la documentación experimental necesaria que facilite el replicado del experimento, es decir, repositorios *OpenSource* con conjuntos de datos y los programas para comparar y mejorar los diseños experimentales.

Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2015-67534-P (MINECO/FEDER, UE) del Ministerio de Economía y Competitividad del Gobierno de España y por el proyecto BU085P17 (JCyL/FEDER, UE) de la Junta de Castilla y León ambos cofinanciados con los fondos FEDER de la Unión Europea.

Un especial agradecimiento a la Doctora Yulan He de la Aston University por acogernos en sus seminarios y enseñarnos técnicas sobre procesamiento del lenguaje natural y aprendizaje automático basado en texto.

Referencias

[1] Amritanshu Agrawal, Wei Fu, and Tim Menzies. What is wrong with topic modeling? and how to fix it using search-based software engineering. *Information and Software Technology*, 98:74–88, jun 2018.

[2] H. U. Asuncion, A. U. Asuncion, and R. N. Taylor. Software traceability with topic modeling. volume 1, pages 95–104, 2010.

[3] A. Barua, S.W. Thomas, and A.E. Hassan. What are developers talking about? an analysis of topics and trends in stack overflow. *Empirical Software Engineering*, 19(3):619–654, 2014.

[4] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, 2003.

[5] M. Fejzer, M. Wojtyna, M. Burzańska, P. Wiśniewski, and K. Stencel. Open source is a continual bugfixing by a few. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8716:153–162, 2014.

[6] Jungil Kim and Eunjoo Lee. Understanding review expertise of developers: A reviewer recommendation approach based on latent dirichlet allocation. *Symmetry*, 10(4), 2018.

[7] S. Kumaresh and R. Baskaran. Mining software repositories for defect categorization. *Journal of Communications Software and Systems*, 11(1):31–36, 2015.

[8] Z. Liao, Y. Li, D. He, J. Wu, Y. Zhang, and X. Fan. Topic-based integrator matching for pull request. volume 2018-January, pages 1–6, 2018.

[9] S. K. Lukins, N. A. Kraft, and L. H. Etzkorn. Bug localization using latent dirichlet allocation. *Information and Software Technology*, 52(9):972–990, 2010.

[10] G. Maskeri, S. Sarkar, and K. Heafield. Mining business topics in source code using latent dirichlet allocation. pages 113–120, 2008.

[11] Annibale Panichella, Bogdan Dit, Rocco Oliveto, Massimilano Di Penta, Denys Poshynanyk, and Andrea De Lucia. How to effectively use topic models for software engineering tasks? an approach based on genetic algorithms. In *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, may 2013.

[12] M.M. Rahman and C.K. Roy. An insight into the pull requests of github. pages 364–367, 2014.

[13] A. Sharma, F. Thung, P.S. Kochhar, A. Sulistya, and D. Lo. Cataloging github repositories. volume Part F128635, pages 314–319, 2017.

[14] X. Sun, B. Li, H. Leung, B. Li, and Y. Li. Msr4sm: Using topic models to effectively mining software repositories for software maintenance tasks. *Information and Software Technology*, 66:1–12, 2015.

[15] Y. Xiong, Z. Meng, B. Shen, and W. Yin. Developer identity linkage and behavior mining across github and stackoverflow. *International Journal of Software Engineering and Knowledge Engineering*, 27(9-10):1409–1425, 2017.

[16] Y. Xiong, Z. Meng, B. Shen, and W. Yin. Mining developer behavior across git hub and stack overflow. pages 578–583, 2017.

[17] W.E. Zhang, Q.Z. Sheng, E. Abebe, M. Ali Babar, and A. Zhou. Mining source code topics through topic model and words embedding. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10086 LNAI:664–676, 2016.

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 5:
PREPROCESAMIENTO DE DATOS

Feature Dimensionality vs. Distribution of Sample Types: A Preliminary Study on Gene-Expression Microarrays

J. Salvador Sánchez
Institute of New Imaging Technologies
Department of Computer Languages and Systems
Universitat Jaume I
Castelló de la Plana, Spain
sanchez@uji.es

Vicente García
División Multidisciplinaria de Ciudad Universitaria
Universidad Autónoma de Ciudad Juárez
Ciudad Juárez, Chihuahua, Mexico
vicente.jimenez@uacj.mx

***Abstract*—In gene-expression microarray data sets each sample is defined by hundreds or thousands of measurements. High-dimensionality data spaces have been reported as a significant obstacle to apply machine learning algorithms, owing to the associated phenomenon called 'curse of dimensionality'. The analysis and interpretation of these data sets have been defined as a very challenging problem. The hypothesis proposed in this paper is that there may exist some correlation between dimensionality and the types of samples (safe, borderline, rare and outlier). To examine our hypothesis, we have carried out a series of experiments over four gene-expression microarray databases because these data correspond to a typical example of the so-called 'curse of dimensionality' phenomenon. The results show that there indeed exist meaningful relationships between dimensionality and the proportion of each type of samples, demonstrating that the amount of safe samples increases and the total number of borderline samples decreases as dimensionality of the feature space decreases.**

***Index Terms*—Gene-expression microarray, feature dimensionality, sample types, feature ranking, classification**

I. Introduction

The 'curse of dimensionality' phenomenon (also known as the Hughes phenomenon) constitutes a challenging problem in many real-life applications. It refers to a situation in which the number of samples needed to estimate an arbitrary function with a given level of accuracy grows exponentially with respect to the number of input variables (dimensionality) of the function [1]. An illustrative example of this problem corresponds to gene-expression microarray data [2], [3] where the number of genes (G) massively exceeds the sample size (n): there are typically over tens of thousands of gene-expression levels and often less than 100 samples in the data set. This is a problem in itself because it may increase the complexity of classification/prediction, degrade the generalization ability of classifiers and hinder the understanding of the underlying relationships between the genes and the samples [4], [5]. Besides, overfitting is also a major issue in a high-dimensional, low-sample scenario [6].

Feature selection is the standard way to tackle this problem by choosing a subset of informative variables from the whole set of features for further analysis. In the specific context of microarray data, there exists an apparent need for dimensionality reduction not only because of the huge number of input variables, but also because many of them can be highly correlated with other variables. Throughout the last decades, many different feature (gene) selection algorithms have been proposed using filter, wrapper, embedded, ensembles and hybrid methods [7]–[11].

A particularly popular strategy for feature selection over microarray data refers to the use of gene ranking algorithms, which are filters that comprise some univariate scoring metric to quantify how much more statistically significant each gene is than the others [12]. These methods rank genes in decreasing order of the estimated scores under the assumption that the top-ranked genes correspond to the most informative (or differentially expressed) ones across different classes without redundancy.

The central question the present study intends to answer is how dimensionality of the feature space and some intrinsic data characteristics are related to each other. More specifically, this paper examines whether or not dimensionality reduction may alter the distribution of the different types of samples defined by several authors [13], [14]. To gain some insight into this question, we analyze the tendency of the amount of each type of samples when varying the dimensionality of the feature space. For the experiments, we consider four public data sets of gene-expression microarrays.

Over the past years, the potential links between feature dimensionality and several data complexities in microarrays have been a matter of concern for researchers. For instance, Baumgartner and Somorjai [15] used five real-life biomedical databases of increasing difficulty to show how the data complexity of a given classification problem can be related to the performance of regularized linear classifiers. Okun and Priisalu [16] explored the connections between data complexity and classification performance defined by low-variance and low-biased bolstered resubstitution error made by k-nearest neighbor classifiers. Souto et al. [17] computed different

measures characterizing the complexity of gene expression data sets for cancer diagnosis, and then investigated how those measures were related to the classification performances of support vector machines. Bolón-Canedo et al. [18] presented a review of a set of feature selection methods applied to DNA microarray data and analyzed the impact of class imbalance, class overlapping or data set shift on the classification results. Similarly, Sánchez and García [19] demonstrated that there exist meaningful relationships between dimensionality and class separability in gene-expression microarray data sets. Lorena et al. [20] studied the complexity of several microarray data sets with and without dimensionality reduction using a support vector machine. Seijo-Pardo et al. [21] proposed the use of three data complexity measures to automatically set a threshold value, which is then employed to obtain a subset of genes from the ordered ranking given by a ranker algorithm. Morán-Fernández et al. [22] demonstrated that there is some correlation between microarray data complexity and the classification error rates of a set of classifiers. Sun et al [23] proposed an ECOC algorithm to address the small sample size and class imbalance problems in multi-class microarray data sets by exploring data distributions based on data complexity theory.

Henceforth, the rest of the paper is organized as follows. Section II presents the types of samples according to the taxonomy proposed by Napierala and Stefanowski [14]. Section III provides the experimental set-up and the description of the databases used in our experiments. Next, the results are reported and discussed in Section IV. Finally, Section V summarizes the main conclusions and points out some directions for future research.

II. Types of Samples

Following the categorization proposed by several authors [13], [14], [24], two main types of samples should be distinguished: *safe* and *unsafe*. Safe samples refer to those located in homogeneous regions with data of a single class and are sufficiently separated from examples of other classes, whereas the rest of samples have to be considered as unsafe. The safe samples will be classified correctly by most models, but the classification of unsafe samples will usually be a very tricky task with a high error rate.

The general feature of unsafe samples is that they are placed close to examples from some other classes. However, this type of data can be further divided into three subgroups depending on their particular characteristics [14], [25]: *borderline*, *rare* and *outlier*. Borderline samples are located near the decision boundaries between classes. Rare samples are small groups of examples located far from the core of their class, creating small data chunks or subclusters. Finally, the outliers are single samples being surrounded by examples that belong to some other class.

A simple method to identify each type of samples is based on analyzing the local neighborhood of the examples. This can be performed either by searching for the k neighbors of a sample or by using some kernel function. Thus, one can guess that a safe sample x will be characterized by having a neighborhood with a majority of examples that belong to its same class; rare examples and outliers will be mainly surrounded by examples from different classes, whereas borderline samples will be surrounded by examples both from their same class and also from different classes.

Many authors often choose $k = 5$ because smaller values may poorly distinguish the nature of samples, while higher values would violate the assumption of the local neighborhood [14], [24]–[26]. Following this procedure, we can define the following cases:

- A sample x will be classified in the safe category if at least 4 out of the 5 nearest neighbors belong to the class of x.
- A sample x will be classified in the borderline category if 2–3 out of its 5 nearest neighbors belong to the class of x.
- A sample x will be classified in the rare category if only one nearest neighbor belongs to the class of x, and this has no more than one neighbor from its same class.
- A sample x will be classified in the outlier category if all its nearest neighbors are from the opposite class.

III. Databases and Experimental Protocol

We conducted a pool of experiments on a collection of publicly available gene-expression microarray data sets, which were taken from the Kent Ridge Biomedical Data Set Repository (http://datam.i2r.a-star.edu.sg/datasets/krbd). Table I reports the main characteristics of these databases, including the number of genes (features), the number of samples, and the size of each class (here designated as positive and negative).

TABLE I
Characteristics of the gene-expression microarray data sets

	#Genes	#Samples	#Positive	#Negative
Breast	24481	97	46	51
CNS	7129	60	21	39
Colon	2000	62	22	40
Prostate	12600	136	59	77

For the present study, we varied the percentage of genes from 5% to 100% with a step size of 5% by using the ReliefF algorithm, thus yielding 20 different subsets (each one with a percentage of the top-ranked features) for each database. The experiments have been circumscribed to the ReliefF algorithm because this paper aims to analyze how dimensionality of the feature space might affect the proportion of the different types of samples, not to find the best feature selection/ranking method.

A. The ReliefF Algorithm

The basic idea of the ReliefF algorithm [27] lies on adjusting the weights of a vector $W = [w(1), w(2), \ldots, w(G)]$ with the objective of giving more relevance to features that better discriminate the samples from neighbors of some different class.

It randomly picks out a sample x and searches for k nearest neighbors of the same class (hits, h_i) and k nearest neighbors from each of the different classes (misses, m_i). If x and h_i have different values on feature f, then the weight $w(f)$ is decreased because it is interpreted as a bad property of this feature. In contrast, if x and m_i have different values on the feature f, then $w(f)$ is increased. This process is repeated t times, and the values of the weight vector W are updated as follows:

$$w(f) = w(f) - \frac{\sum_{i=1}^{k} dist(f,x,h_i)}{t \cdot k} + \sum_{c \neq class(x)} \frac{P(c)}{1 - P(class(x))} \cdot \frac{\sum_{i=1}^{k} dist(f,x,m_i)}{t \cdot k} \quad (1)$$

where $P(c)$ is the prior probability of class c, $P(class(x))$ denotes the probability for the class of x, and $dist(f,x,m_i)$ represents the absolute distance between samples x and m_i in the feature f.

The algorithm assigns negative values to features that are completely irrelevant and the highest scores for the most informative features. In general, one will then select the g top-ranked features in order to build the classifier with a presumably much smaller subset of features ($g \ll G$). In addition, unlike other well-known ranking methods such as those based on information theory (e.g., mutual information or information gain), the ReliefF algorithm takes care of the dependencies between genes [28].

IV. Results and Discussion

This section is devoted to explore how the number of genes may have an effect on the amount of samples that belong to each type. As far as we know, there has been no systematic analysis on this problem; in fact, previous studies have focused on identifying the types of samples from the minority class in class imbalanced data sets and analyzing how the resampling techniques may alter the distribution/proportion of safe, borderline, rare and outlier samples [14], [24]–[26], [29], [30]

Bearing our purpose in mind, the experiments were as follows. First, we calculated the percentages of positive and negative samples from each type when varying the percentage of genes. Afterwards, we also run six classifiers of different nature over each subset of features: the 1-nearest neighbor (1-NN) rule with the Euclidean distance, a pruned C4.5 decision tree, a support vector machine (SVM) with a linear kernel using the sequential minimal optimization algorithm and a soft-margin $C = 1.0$, a normalized Gaussian radial basis function (RBF) neural network with the K-means clustering algorithm to provide the basis functions, the naive Bayes classifier (NBayes), and a multi-layer perceptron (MLP) with one hidden layer, a learning rate of 0.3 and 500 training epochs.

Fig. 1 shows the percentages of each positive sample type when varying the dimensionality of the feature space for each database. As can be seen, the percentage of safe samples in the positive class increases and the percentage of borderline positive samples decreases as dimensionality decreases. Although the percentages of rare and outlier samples are generally low, it was observed a very similar behavior to that of the borderline samples. This result could allow to gain some insight into the reasons why classification in lower dimensions is usually easier than in higher dimensions.

Analogously, Fig. 2 displays the percentages of the negative sample types when varying the dimensionality of the feature space for each database. In general, lines in these plots closely match the trend patterns recognized in the plots of Fig. 1, that is, the percentage of safe samples increases and the percentages of the different types of unsafe samples decrease as dimensionality decreases. Notwithstanding, for the safe and borderline samples, we observed an essential difference of behavior between the positive class and the negative class: while the percentages of safe positive samples were usually lower than those of the borderline positive samples, the percentages of safe negative samples always resulted much higher than those of the borderline negative samples. This behavior agrees with the expected one because the negative class corresponds to the majority class and therefore, the probability for a negative sample to be identified as safe is higher than the probability of being classified in some group of the unsafe samples.

Regarding the rare and outlier samples that belong to the negative class, we found that there was no substantial relationship between dimensionality of the feature space and the number of samples in both these types. Nevertheless, this fact should not become especially critical for a given classification problem because the amount of samples that belong to the rare and outlier types is minimal as compared to the total number of safe and borderline samples.

Plots in Fig. 3 correspond to the accuracy achieved by each classification model when applied to each of the 20 subsets. It is possible to observe that the accuracy of all classifiers tends to decrease as the amount of genes increases. A visual comparison between this figure and those of the sample types allows to demonstrate that there exists some significant link (positive correlation) between the dimensionality of the feature space and the distribution of sample types since the highest accuracies were achieved for the subsets with the largest number of safe samples and the smallest number of unsafe samples.

V. Concluding Remarks

As one of the earliest works on investigating the potential connections between feature dimensionality and sample types, this paper has to be viewed as a preliminary study of the effects of dimensionality reduction on the distribution of the different types of samples in a data set.

From the experiments carried out, we have observed that the proportions of safe, borderline, rare and outlier samples vary as the dimensionality of the feature space changes. More specifically, reduction in dimensionality generally leads to a

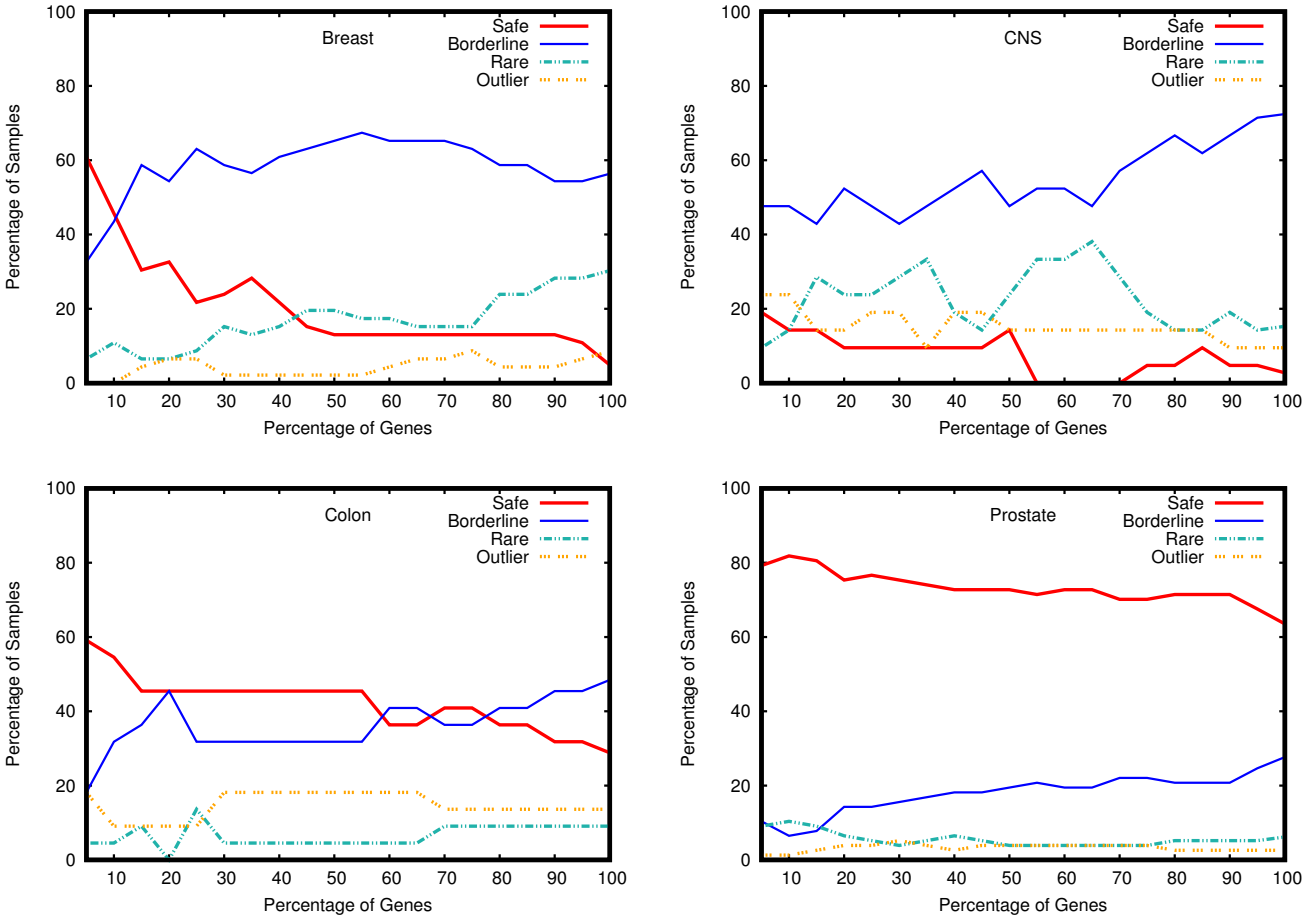


Fig. 1. Plots of the percentage of each type of positive samples when varying the percentage of genes

significant decrease in the amount of borderline samples and an increase in the number of safe samples. As showed in the experiments, this has a direct impact on the performance of classifiers because the classification of safe samples results much easier than the classification of any type of unsafe samples.

Through the characterization of databases by the distribution of their sample types, our hypothesis for further research is that it would be possible to define a meta-learning framework to choose the feature subset with the highest classification performance. Another direction for extending the present paper consists in the combined use of sample types and data complexity measures for the implementation of accurate preprocessing methods.

Acknowledgment

This research work has partially been supported by the Mexican PRODEP under Grant No. DSA/103.5/15/7004, the Generalitat Valenciana under Grant No. PROMETEOII/2014/062, and the Universitat Jaume I under Grant No. P1-1B2015-74.

References

[1] L. Chen, "Curse of dimensionality," in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Boston, MA: Springer, 2009, pp. 545–546.

[2] R. Clarke, H. W. Ressom, A. Wang, J. Xuan, M. C. Liu, E. A. Gehan, and Y. Wang, "The properties of high-dimensional data spaces: implications for exploring gene and protein expression data," *Nat. Rev. Cancer*, vol. 8, no. 1, pp. 37–49, 2008.

[3] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "Recent advances and emerging challenges of feature selection in the context of big data," *Knowl.-Based Syst.*, vol. 86, pp. 33–45, 2015.

[4] E. R. Dougherty, "Small sample issues for microarray-based classification," *Compar. Func. Genom.*, vol. 2, no. 1, pp. 28–34, 2001.

[5] L. Wang, F. Chu, and W. Xie, "Accurate cancer classification using expressions of very few genes," *IEEE-ACM T. Comput. Biol. Bioinform.*, vol. 4, no. 1, pp. 40–53, 2007.

[6] R. Somorjai, B. Dolenko, and R. Baumgartner, "Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions," *Bioinformatics*, vol. 19, no. 12, pp. 1484–1491, 2003.

[7] I. Inza, P. Larrañaga, R. Blanco, and A. J. Cerrolaza, "Filter versus wrapper gene selection approaches in DNA microarray domains," *Artif. Intell. Med.*, vol. 31, no. 2, pp. 91–103, 2004.

[8] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[9] C. Lazar, J. Taminau, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, H. Bersini, and A. Nowe, "A survey on filter techniques for feature selection in gene expression microarray analysis,"

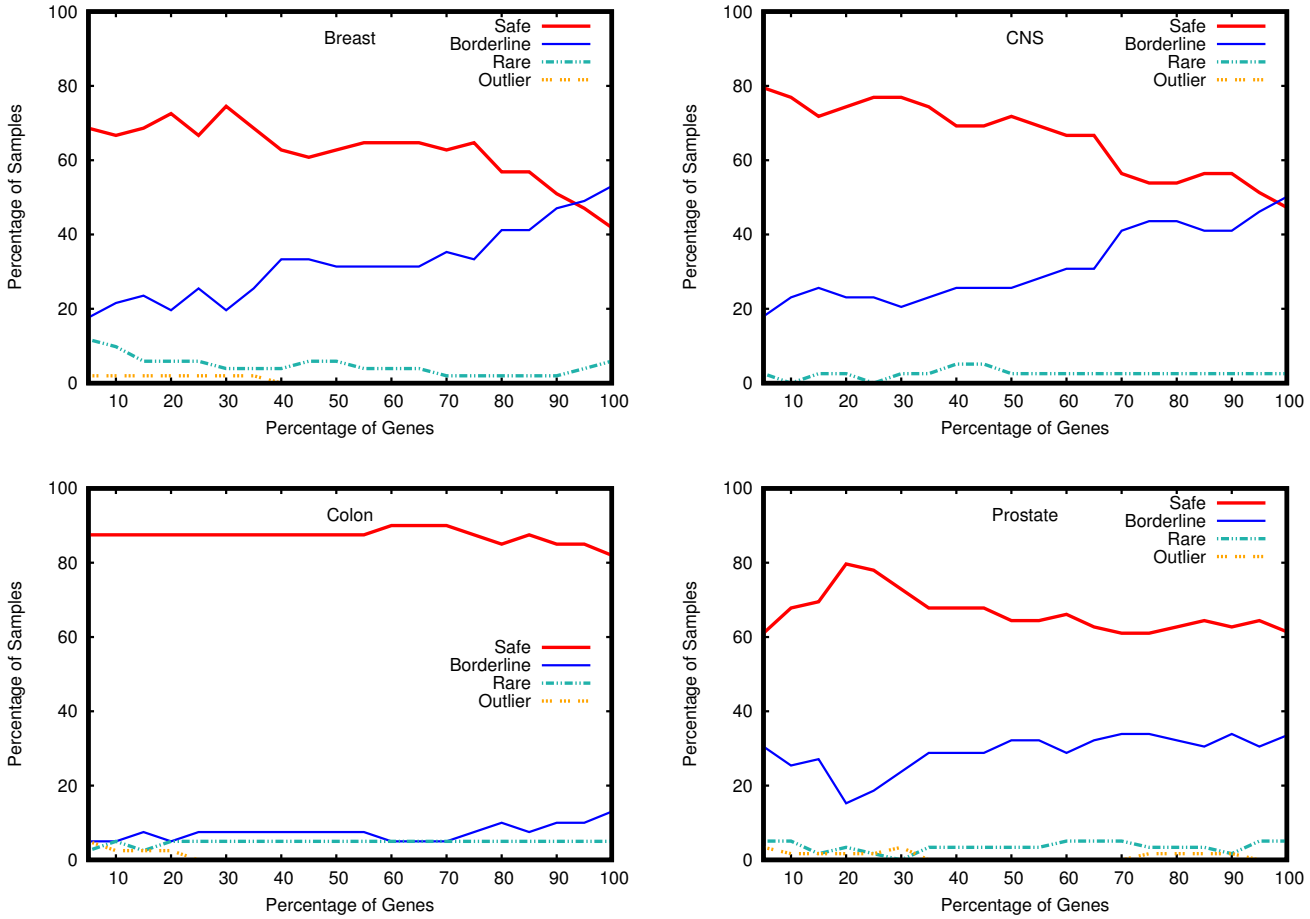


Fig. 2. Plots of the percentage of each type of negative samples when varying the percentage of genes

IEEE-ACM T. Comput. Biol. Bioinform., vol. 9, no. 4, pp. 1106–1119, 2012.

[10] Z. M. Hira and D. F. Gillies, "A review of feature selection and feature extraction methods applied on microarray data," *Adv. Bioinformatics*, vol. 2015, no. ID 198363, pp. 1–13, 2015.

[11] J. C. Ang, A. Mirzal, H. Haron, and H. N. A. Hamed, "Supervised, unsupervised, and semi-supervised feature selection: A review on gene selection," *IEEE-ACM T. Comput. Biol. Bioinform.*, vol. 13, no. 5, pp. 971–989, 2016.

[12] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[13] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in *Proc. 14th International Conference on Machine Learning*, Nashville, TN, 1997, pp. 179–186.

[14] K. Napierala and J. Stefanowski, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *J. Intell. Inf. Syst.*, vol. 46, no. 3, pp. 563–597, 2016.

[15] R. Baumgartner and R. Somorjai, "Data complexity assessment in undersampled classification of high-dimensional biomedical data," *Pattern Recogn. Lett.*, vol. 27, no. 12, pp. 1383–1389, 2006.

[16] O. Okun and H. Priisalu, "Dataset complexity in gene expression based cancer classification using ensembles of k-nearest neighbors," *Artif. Intell. Med.*, vol. 45, no. 2, pp. 151–162, 2009.

[17] M. C. P. de Souto, A. C. Lorena, N. Spolaôr, and I. G. Costa, "Complexity measures of supervised classifications tasks: A case study for cancer gene expression data," in *Proc. International Joint Conference on Neural Networks*, Barcelona, Spain, 2010, pp. 1–7.

[18] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, J. Benítez, and F. Herrera, "A review of microarray datasets and applied feature selection methods," *Inform. Sciences*, vol. 282, pp. 111–135, 2014.

[19] J. S. Sánchez and V. García, "Addressing the links between dimensionality and data characteristics in gene-expression microarrays," in *Proc. International Conference on Learning and Optimization Algorithms: Theory and Applications*, Rabat, Morocco, 2018, pp. 1–6.

[20] A. C. Lorena, I. G. Costa, N. Spolaor, and M. C. de Souto, "Analysis of complexity indices for classification problems: Cancer gene expression data," *Neurocomputing*, vol. 75, no. 1, pp. 33–42, 2012.

[21] B. Seijo-Pardo, V. Bolón-Canedo, and A. Alonso-Betanzos, "Using data complexity measures for thresholding in feature selection rankers," in *Advances in Artificial Intelligence*. Lecture Notes in Computer Science, Springer, 2016, vol. 9868, pp. 121–131.

[22] L. Morán-Fernández, V. Bolón-Canedo, and A. Alonso-Betanzos, "Can classification performance be predicted by complexity measures? A study using microarray data," *Knowl. Inf. Syst.*, vol. 51, no. 3, pp. 1067–1090, 2017.

[23] M. Sun, K. Liu, and Q. Hong, "An ECOC approach for microarray data classification based on minimizing feature related complexities," in *Proc. 10th International Symposium on Computational Intelligence and Design*, Hangzhou, China, 2017, pp. 300–303.

[24] J. A. Sáez, B. Krawczyk, and M. Woźniak, "Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets," *Pattern Recogn.*, vol. 57, pp. 164–178, 2016.

[25] B. Krawczyk, M. Woniak, and F. Herrera, "Weighted one-class classification for different types of minority class examples in imbalanced data," in *Proc. IEEE Symposium on Computational Intelligence and Data Mining*, Piscataway, NJ, 2014, pp. 337–344.

[26] P. Skryjomski and B. Krawczyk, "Influence of minority class instance types on SMOTE imbalanced data oversampling," in *Proc. 1st Interna-*

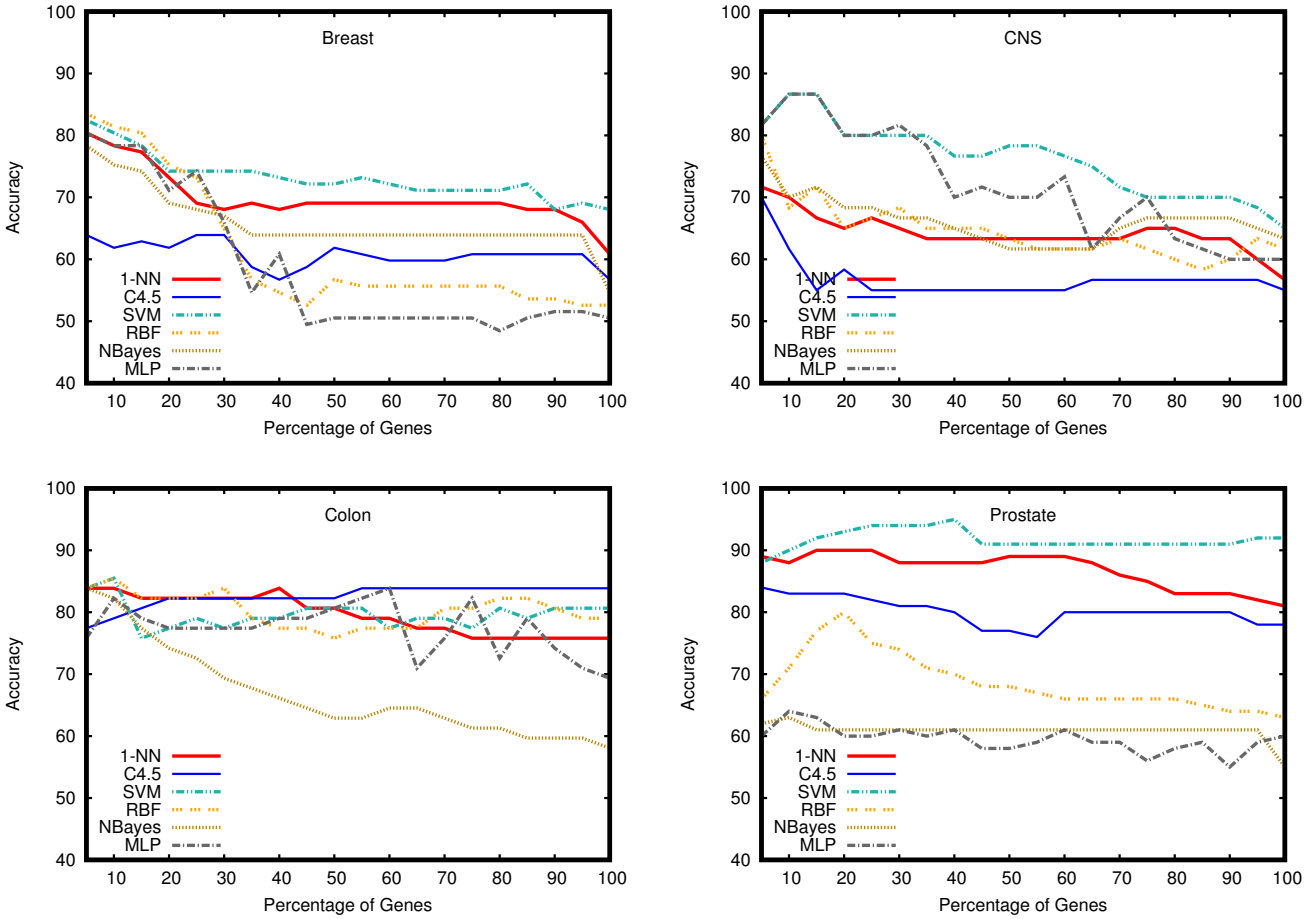


Fig. 3. Classification accuracies when varying the percentage of genes

tional Workshop on Learning with Imbalanced Domains: Theory and Applications, Skopje, Macedonia, 2017, vol. 74, pp. 7–21.

[27] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Mach. Learn.*, vol. 53, no. 1–2, pp. 23–69, 2003.

[28] Y. Peng, W. Li, and Y. Liu, "A hybrid approach for biomarker discovery from microarray gene expression data for cancer classification," *Cancer Inform.*, vol. 2, pp. 301–311, 2006.

[29] S. Wang, L. L. Minku, and X. Yao, "Resampling-based ensemble methods for online class imbalance learning," *IEEE T. Knowl. Data En.*, vol. 27, no. 5, pp. 1356–1368, 2015.

[30] M. Lango and J. Stefanowski, "The usefulness of roughly balanced bagging for complex and high-dimensional imbalanced data," in *Proc. 4th International Workshop on New Frontiers in Mining Complex Patterns*. Porto, Portugal: Springer International Publishing, 2016, pp. 93–107.

Selección de características distribuida en entornos heterogéneos

Verónica Bolón-Canedo
Grupo LIDIA. DCITIC.
Universidade da Coruña
A Coruña, España
veronica.bolon@udc.es

Rubén Seoane-Martínez
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España

José Luis Morillo-Salas
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
jose.luis.morillo@udc.es

Amparo Alonso-Betanzos
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
ciamparo@udc.es

***Resumen*—Los avances en las Tecnologías de la Información y las Comunicaciones han contribuido a la proliferación de grandes bases de datos. En algunos casos estos datos ya están distribuidos en su origen, pero en otros casos su gran escala hace que el procesamiento en un único nodo sea imposible, y en consecuencia la distribución en varios nodos de cómputo es una opción natural para su manejo. En este trabajo, proponemos una metodología que nos permite distribuir el proceso de selección de características, la mayoría de las veces un paso de preprocesado imprescindible en los conjuntos de alta dimensión actuales, ya que nos permite reducir la dimensión de entrada, seleccionando las características relevantes y eliminando las redundantes y/o irrelevantes. En particular, nuestra propuesta en este artículo se centra en el problema de los conjuntos de datos desbalanceados, bien porque la situación se da ya en origen o bien cuando este contexto en que las distintas clases de datos no están igualmente representadas en las distintas particiones se produce debido a que se debe distribuir el conjunto único original para poder tratarlo. Los resultados experimentales obtenidos demuestran que nuestra aproximación distribuida obtiene resultados de error comparables a la aproximación centralizada, aportando como ventajas una reducción apreciable del tiempo computacional y la capacidad de trabajar eficientemente en entornos de desbalanceo de clases.**

***Index Terms*—selección de características, algoritmos distribuidos, conjuntos de datos desbalanceados.**

I. Introducción

La selección de características (SC) es una técnica de aprendizaje automático en la que se seleccionan los atributos que permiten que un problema esté claramente definido, mientras que los irrelevantes o redundantes se ignoran [1]. Tradicionalmente, un algoritmo de SC se aplica de manera centralizada, es decir, se utiliza un único modelo selector de características sobre todos los datos del conjunto para resolver un problema determinado. Sin embargo, en algunos casos, los datos pueden o bien estar ya distribuidos en varias localizaciones, o bien se puede usar una estrategia de aprendizaje distribuido para repartir en varios nodos de cómputo un conjunto de datos que es demasiado grande para poder ser procesado en un único nodo. De esta forma podemos aprovechar el procesamiento de estos múltiples subconjuntos de datos bien en secuencia o en paralelo. Existen varias formas de distribuir una tarea de selección de características, aunque las más comunes son:

- los datos están juntos en un conjunto de datos muy grande, por lo que se distribuyen en varios procesadores, se ejecuta un algoritmo de SC idéntico en cada uno y luego los resultados parciales se combinan para obtener un resultado final, y
- los datos pueden estar en diferentes conjuntos de datos situados en diferentes ubicaciones, por lo que se ejecuta un algoritmo de selección de características idéntico en cada uno y los resultados se combinan para obtener un resultado final.

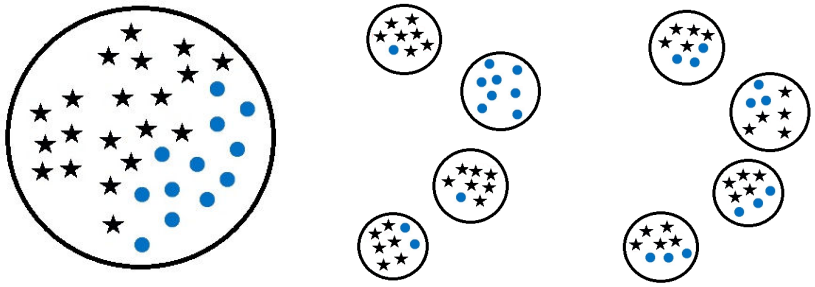
Al respecto, existen varios trabajos en la literatura que realizan la selección de características de forma distribuida [2], [3]. Sin embargo, cuando los datos se distribuyen en varios procesadores, pueden aparecer algunos problemas adicionales, como un alto desequilibrio entre clases en algunos de los nodos, o incluso la situación extrema en la que algunas clases no están representadas en absoluto en algunos de los subconjuntos de datos. El *problema de desequilibrio de clase o desbalanceo* se produce cuando un conjunto de datos está dominado por una clase mayoritaria que tiene significativamente muchas más instancias que las otras clases, llamadas minoritarias. En este caso, los algoritmos de aprendizaje computacional suelen presentar un sesgo hacia las clases mayoritarias, ya que las reglas que predicen correctamente esas instancias se ponderan positivamente a favor de la métrica de precisión, mientras que las reglas específicas que predicen ejemplos de la clase minoritaria generalmente se ignoran. Por lo tanto, las muestras de las clases minoritarias se clasifican erróneamente más a menudo que las de las otras clases [4].º

En este trabajo presentamos una metodología para distribuir el proceso de SC, que tiene en cuenta este problema de la posible heterogeneidad de los subconjuntos. Para ello usamos dos alternativas: (i) forzar las particiones del conjunto de datos para mantener el equilibrio entre las clases, y (ii) aplicar técnicas de sobremuestreo (oversampling) cuando el desequilibrio es inevitable.

II. Metodología distribuida

En este trabajo, se detalla la aplicación de una metodología para distribuir el proceso de SC sobre la base de trabajos pre-

Este trabajo ha sido parcialmente financiado por el Ministerio de Economía y Competitividad (proyectos de investigación TIN 2015-65069-C2-1-R y la Red Española de Big Data y Análisis de datos escalable, TIN2016-82013-REDT), y por Fondos de Desarrollo Regional de la Unión Europea.



(a) Centralizado (b) Partición aleatoria (c) Partición homogénea

Figura 1. Escenarios centralizado (a) y particiones aleatoria (b) y homogénea (c) en un proceso de selección de características distribuido

vios [5], [6]. Esta metodología consta de tres pasos principales, que son los siguientes:

1. partición de los datos, si éstos no estuviesen ya distribuidos en origen,
2. aplicación del método de SC a cada una de las diferentes particiones realizadas
3. combinación de los resultados.

Debemos de tener en cuenta que los dos primeros pasos se repiten varias rondas (r), para garantizar la captura de suficiente información para el paso de combinación de los resultados parciales.

El primer paso de la metodología anterior es el núcleo de este trabajo y consiste en dividir sin reemplazo los datos del conjunto original, asignando grupos de n muestras a cada subconjunto de datos. Se seguirán dos enfoques principales: *partición aleatoria*, en la que se realizará una distribución aleatoria de los datos en los distintos nodos, y *partición homogénea*, en la que se mantienen las proporciones del conjunto original en cada uno de los subconjuntos obtenidos. Un ejemplo de estos dos tipos de partición, junto con el escenario centralizado en el que todos los datos están juntos, se puede ver en la figura 1.

Después de realizar una partición, el conjunto de datos podría estar desequilibrado (ya sea porque la partición se realizó al azar o porque el conjunto de datos ya estaba desbalanceado en origen). En este caso, nuestra propuesta consiste en aplicar el método de sobremuestreo SMOTE [7], que agrega ejemplos sintéticos de la clase minoritaria al conjunto de datos original hasta que la distribución de clases se equilibre. Para poder conseguir esto, SMOTE genera ejemplos sintéticos de la clase minoritaria utilizando los ejemplos originales de la misma de la siguiente manera: en primer lugar, busca los k vecinos más cercanos de la muestra de la clase minoritaria que se utilizará como base para la nueva muestra sintética. Luego, en el segmento que une la muestra de la clase minoritaria con uno o todos sus vecinos, se toma aleatoriamente una muestra sintética y se agrega al nuevo conjunto de datos sobremuestreados.

El siguiente paso en la metodología general consiste en aplicar un método de SC en cada partición. Las características que se seleccionan para ser eliminadas reciben un voto y luego, se realiza una nueva ronda que conduce a una nueva partición del conjunto de datos y se lleva a cabo una nueva iteración de la votación hasta alcanzar el número predefinido de rondas r. Finalmente, las características que han recibido una cantidad de votos por encima de un umbral predefinido se eliminan. Por lo tanto, se obtiene finalmente un conjunto único de características que se pueden utilizar para entrenar un clasificador C y probar su rendimiento en un nuevo conjunto de muestras (conjunto de datos de test). Más detalles sobre cómo elegir el umbral de votos se pueden encontrar en [5], [6]. El pseudocódigo de la metodología propuesta se muestra en el algoritmo 1.

```
1  inicializar el vector de votos a 0
2  para cada ronda hacer
3      dividir el conjunto de datos d aleatoriamente o
         mantener las proporciones de las clases en
         subconjuntos de datos disjuntos
4      para cada subconjunto de datos hacer
5          si los datos están desbalanceados entonces
6              aplicar SMOTE
           fin
7          aplicar un algoritmo de selección de características
8          incrementar un voto para cada característica a ser
             eliminada
       fin
   fin
9  eliminar las características cuyo número de votos sea
     superior a un umbral
10 clasificar con el subconjunto de características obtenido
```

Algoritmo 1: Pseudo-código de la metodología propuesta

III. Resultados experimentales

En esta sección presentaremos el esquema de experimentación y los resultados obtenidos. Recordemos que los objetivos de la experimentación son dos, (i) poder establecer qué tipo de partición es la más adecuada y (ii) cúal es la influencia del algoritmo de sobremuestreo SMOTE cuando las particiones presentan datos desbalanceados.

III-A. Comparación entre las aproximaciones distribuidas y la centralizada

Con este objetivo, hemos seleccionado 6 conjuntos de datos, cuyas características se resumen en la Tabla I, y que están

Cuadro I
Características de los conjuntos utilizados en la primera parte de la experimentación, en donde solamente se utiliza SMOTE en la clase minoritaria.

Conjunto	Nº Muestras	Nº Características	Nº Clases	% clase mayoritaria
Connect4	67557	42	3	65.83
Isolet	7797	617	26	3.85
Madelon	2400	500	2	50
Mnist	60000	717	2	50
Ozone	2536	72	2	97.12
Spambase	4601	57	2	60.6

disponibles para su descarga en el UCI Machine Learning Repository [1].

Aunque la metodología propuesta es genérica, y por lo tanto se puede usar con cualquier método de SC, en este trabajo hemos elegido una suite de cuatro filtros, basados en diferentes tipos de métricas. Concretamente hemos utilizado Correlation-Based Feature Selection (CFS), Consistency-based, Information Gain y ReliefF, todos ellos disponibles en la herramienta de software libre Weka [2]. Para posteriormente poder evaluar los resultados de la selección de características realizada, hemos elegido cuatro clasificadores populares en el estado del arte: C4.5, Naive Bayes, IB1 y Vectores de Máquinas Soporte (en inglés, Support Vector Machine –SVM–). Los experimentos se realizaron en una CPU Core ™i3-6100 Intel ®3.70 GHz con 16 GB de memoria RAM.

En el primer estudio experimental se compararon tres escenarios diferentes: (i) la aproximación centralizada estándard, (ii) la distribución aleatoria, y (iii) el particionado homogéneo. Para las dos aproximaciones distribuidas (la aleatoria y la homogénea), el número de rondas utilizado fue de 5. Para asegurar una buena fiabilidad en los resultados obtenidos, se realizó una validación *hold-out* estándard, es decir, se dividieron los distintos conjuntos de la tabla I en dos subconjuntos diferentes, con la proporción 2/3 para entrenamiento y 1/3 para prueba, y se repitió esta operación 5 veces. También se han usado test de significación estadística, en primer lugar un test de Friedman para comprobar si existían diferencias significativas para un nivel de significación $\alpha = 0{,}5$, y posteriormente de Nemenyi para obtener aquellos modelos que no son significativamente diferentes a los que obtienen la mayor precisión. Las tablas detalladas con los resultados obtenidos para todas las combinaciones entre conjuntos de datos, métodos de SC y clasificadores pueden verse en el material suplementario que se encuentra en [3].

La Tabla II muestra un resumen de este primer conjunto de experimentos, en los que la meta es comparar los tres escenarios (centralizado, partición aleatoria y partición homogénea), independientemente de la aplicación de la técnica de *oversampling* SMOTE. La tabla muestra los resultados para cada combinación de conjunto de datos y escenario, teniendo en cuenta dos medidas de evaluación diferentes: la precisión de la clasificación y el valor del índice kappa. El motivo de incluir el valor de Kappa es porque éste evalúa la calidad del aprendizaje teniendo en cuenta las situaciones en las que el conjunto de datos está desequilibrado y el clasificador aprende correctamente la clase mayoritaria, pero sistemáticamente clasifica erróneamente las instancias de la clase minoritaria. En las primeras dos filas de cada conjunto de datos, se puede consultar el promedio de la precisión de clasificación y Kappa; y en las últimas dos filas se muestran los valores máximos de precisión y Kappa (y la combinación que lo obtiene entre paréntesis). Como se puede ver, los enfoques distribuidos (partición aleatoria y homogénea) son una buena solución para disminuir el tiempo computacional sin implicar una degradación en el rendimiento de clasificación. Comparando los dos enfoques distribuidos, vale la pena señalar que, en general, el enfoque homogéneo parece obtener resultados más estables, mientras que con la partición aleatoria puede ocurrir que en un caso particular la proporción de las clases sea óptima y por esa razón obtiene los mejores resultados en algunos casos.

Como era de esperar, los enfoques distribuidos reducen significativamente el tiempo de ejecución en comparación con el enfoque centralizado (ver detalles en el material complementario 3^0), aunque depende concretamente tanto del método de selección empleado como del conjunto de datos. Cuando el conjunto de datos es pequeño, la mejora es leve (por ejemplo, de 0.40s a 0.34s en el conjunto Ozone) pero en conjuntos de datos más grandes, la mejora es considerable (por ejemplo, de 820.46s a 0.83s en el conjunto Connect-4). Es remarcable también el buen rendimiento obtenido por los métodos de selección ReliefF y Consistency-based.

III-B. Utilización de SMOTE en las particiones

El segundo grupo de experimentos consiste en la evaluación de la efectividad de SMOTE ante el problema del desbalanceo de clases. La comparación se realizó entre los dos escenarios distribuidos (partición aleatoria y homogénea). Debemos tener en cuenta que, al aplicar la partición aleatoria, es posible que algunos subconjuntos de datos estén desbalanceados, incluso si el conjunto de datos completo no lo estaba. Por lo tanto, hemos aplicado SMOTE cuando el subconjunto de datos no estaba balanceado (ya sea bien debido a la existencia de esta circunstancia en clase original, o bien debido a la partición aleatoria). Se han realizado diferentes experimentos con diferentes porcentajes de sobremuestreo. Por ejemplo, si la clase minoritaria tiene 40 muestras y aplicamos SMOTE

[1] http://archive.ics.uci.edu/ml/index.php
[2] http://www.cs.waikato.ac.nz/ ml/weka/
[3] http://lidiagroup.org/index.php/en/materials-en.html

Conjunto		Centralizado	Aleatorio	Homogéneo
Connect4	Precisión (media)	65.72	67.52	67.52
	Kappa (media)	0.175	0.184	0.188
	Precisión (max)	73.37 (Cons+C4.5)	74.12 (Rel+C4.5)	72.81 (IG+C4.5)
	Kappa (max)	0.454 (Cons+C4.5)	0.425 (Rel+C4.5)	0.399 (Rel+C4.5)
Isolet	Precisión (media)	63.92	69.12	68.64
	Kappa (media)	0.624	0.678	0.673
	Precisión (max)	84.60 (Rel+SVM)	85.51 (Rel+SVM)	83.87 (Rel+SVM)
	Kappa (max)	0.839 (Rel+SVM)	0.849 (Rel+SVM)	0.832 (Rel+SVM)
Madelon	Accuracy (media)	74.64	72.25	75.32
	Kappa (media)	0.492	0.444	0.506
	Accuracy (max)	88.75 (Varios+IB1)	81.75 (Rel+C4.5)	89.62 (Rel+IB1)
	Kappa (max)	0.774 (Varios+IB1)	0.636 (Rel+C4.5)	0.792 (Rel.+IB1)
MNIST	Precisión (media)	81.04	83.68	83.83
	Kappa (media)	0.620	0.672	0.675
	Precisión (max)	89.96 (Rel+IB1)	96.33 (Cons+IB1)	95.83 (Rel+IB1)
	Kappa (max)	0.799 (Rel+IB1)	0.926 (Cons+IB1)	0.916 (Rel+IB1)
Ozone	Precisión (media)	92.09	91.06	90.87
	Kappa (media)	0.1014	0.1012	0.092
	Precisión (max)	97.12 (Todos+SVM)	96.99 (Todos+SVM)	96.97 (Todos+SVM)
	Kappa (max)	0.189 (Cons+C4.5)	0.215 (Cons+C4.5)	0.180 (IG+IB1)
Spambase	Precisión (media)	86.66	87.18	87.63
	Kappa (media)	0.723	0.732	0.742
	Precisión (max)	91.42 (CFS+C4.5)	91.24 (CFS+C4.5)	91.73 (CFS+C4.5)
	Kappa (max)	0.819 (CFS+C4.5)	0.816 (CFS+C4.5)	0.826 (CFS+C4.5)

Cuadro II

Resumen de los resultados obtenidos para las aproximaciones distribuidas y centralizada. No se ha utilizado el método SMOTE en las aproximaciones distribuidas.

con un nivel de 100, significa que se generan 40 muestras sintéticas, si el nivel es 200, significa que se generan 80 nuevas muestras. Además, incluimos la opción "auto", que consiste en aplicar SMOTE de tal forma que las clases queden completamente balanceadas.

Conjunto	Precisión	Kappa	Escenario	Combinación	SMOTE
Isolet	85.68	0.851	Aleatorio	Rel+SVM	Auto
Madelon	89.63	0.792	Homogéneo	Rel+IB1	0
MNIST	96.34	0.926	Aleatorio	Cons+IB1	0
Connect4	74.12	0.425	Aleatorio	Rel+C4.5	0
	72.90	0.442	Aleatorio	Rel+C4.5	100
Ozone	97.28	0	Homogéneo	All+SVM	100
	90.82	0.302	Homogéneo	Rel+SVM	600
Spambase	91.73	0.826	Homogéneo	CFS+C4.5	0
	91.68	0.827	Homogéneo	CFS+C4.5	300

Cuadro III

Resumen de los resultados obtenidos usando SMOTE en las clases minoritarias.

```
1  para repeticiones hacer
2      dividir el conjunto de datos d aleatoriamente en
         subconjuntos disjuntos de datos train y test
3      calcular la clase mayoritaria del conjunto de train
4      para cada nivel Smote minoritaria hacer
5          para cada nivel Smote mayoritaria hacer
6              train = SMOTE(clase_mayoritaria, train)
7              train = SMOTE(clase_minoritaria, train)
8              para filtros hacer
9                  train_filtrado = sel_car(filtro,train)
10                 para clasificador hacer
11                     clasificar(clasificador, train_filtrado,
                         test)
                   fin
               fin
           fin
       fin
   fin
```

Algoritmo 2: Pseudo-código de la metodología propuesta usando SMOTE también en la clase mayoritaria

La tabla III muestra el resumen de los mejores resultados obtenidos al aplicar diferentes niveles de sobremuestreo con SMOTE a los subconjuntos de datos. En la primera fila de cada conjunto de datos, se muestra la opción con la mayor precisión, mientras que la segunda fila representa la opción con el valor Kappa más alto. Cuando el mejor resultado para ambas mediciones de evaluación se logra mediante la misma combinación y escenario, solo se muestra una fila. Como era de esperar, la aplicación de una técnica de sobremuestreo no es necesaria en el caso de conjuntos de datos equilibrados (Isolet, Madelon, MNIST). En el caso de Isolet, la aplicación de SMOTE en el escenario de partición aleatoria ha resultado provechosa, ya que en este caso es posible que los conjuntos de datos equilibrados produzcan subconjuntos de datos no balanceados (especialmente para Isolet, con un alto número de clases). Por el contrario, los conjuntos de datos desbalanceados (Connect4, Ozone y Spambase) son buenos candidatos para mejorar sus resultados después de aplicar SMOTE. De hecho, la Tabla III muestra que la aplicación del método de sobremuestreo mejora los valores Kappa, lo que significa que el aprendizaje de las clases es mejor. Hay que recordar que la clase mayoritaria de ozono tiene el 97.12 % de las muestras, por lo que al obtener una precisión de clasificación del 97.28 % es posible que clasifique correctamente todas las muestras de la clase mayoritaria, pero solo unas pocas de la clase minoritaria. Después de aplicar el método de sobremuestreo, la precisión cae al 90.82 %, probablemente porque el clasificador no está tan sobreajustado para aprender la clase mayoritaria

y tiene una tasa de verdaderos positivos más alta en la clase minoritaria.

Finalmente, se realizó un tercer conjunto de experimentos, en los que se utiliza también la técnica SMOTE para añadir también muestras sintéticas en la clase mayoritaria, no sólo en la minoritaria, de forma que todas las clases, mayoritarias y minoritarias, cuenten en sus subconjuntos con muestras sintéticas, como se puede ver en el algoritmo 2.

Para realizar este tercer bloque de experimentos se utilizaron dos tipos de conjuntos, los que denominamos con la etiqueta *normal* en la tabla IV, que son conjuntos de datos en los que el número de muestras es mucho mayor que el número de características, y conjuntos de datos del tipo *Microarrays* [8], obtenidos de investigaciones sobre la clasificación de casos de cáncer, que tienen un elevado número de características y un número muy pequeño de muestras (ver tabla IV). La idea es comprobar no sólo si el realizar SMOTE en todas las clases mejora el resultado, al tener muestras sintéticas en todas ellas, sino también si el balance entre muestras y características influye en los resultados. Se han repetido de nuevo los experimentos, pero en este caso además se han añadido muestras sintéticas también en la clase mayoritaria, utilizando SMOTE con porcentajes de 20, 40 y 100. Al igual que anteriormente, se han obtenido valores para todas las posibles combinaciones de clasificador, filtro y combinación de porcentajes de SMOTE en la clase mayoritaria. En la tabla V se pueden ver los resultados obtenidos para todos los conjuntos de la tabla IV sin SMOTE, con la alternativa de SMOTE en la clase minoritaria y con la alternativa de usar SMOTE en todas las clases.

Como podemos ver en la tabla V, la alternativa SMOTE en las clases minoritaria y mayoritaria conjuntamente es siempre la opción que alcanza la precisión máxima, con los valores de índice kappa más altos (en ocasiones, otras alternativas consiguen idénticas kappas, y la alternativa SMOTE sólo en minoritaria empata en precisión máxima en 5 de los 12 conjuntos). No parecen existir grandes diferencias entre los dos tipos de conjuntos, si bien la diferencia en precisión media entre las dos alternativas usando SMOTE en los conjuntos microarray es menor que en el caso de los conjuntos que hemos denominado normales.

IV. Conclusiones y Trabajo Futuro

Hemos presentado una metodología para la selección de características distribuida, tratando de resolver el problema del desbalanceo de los datos en las diferentes particiones. Para ello, hemos forzado a las particiones de datos en los diferentes nodos a mantener la misma distribución de clase que el conjunto de datos original y hemos aplicado la técnica de sobremuestreo (oversampling) SMOTE. Los resultados experimentales en siete conocidos conjuntos de datos han demostrado que:

- El enfoque distribuido — partición aleatoria u homogénea — es competitivo cuando se compara con el enfoque centralizado estándar, incluso en algunos casos mejorando el rendimiento de clasificación.
- La partición homogénea obtiene resultados más estables que la partición aleatoria.
- La aplicación de SMOTE en las clases minoritarias (uso estándard del procedimiento), mejora la calidad del aprendizaje en conjuntos de datos desbalanceados, en algunos casos a expensas de una ligera disminución en la precisión general.
- Además al aplicar el método propuesto con un porcentaje pequeño de SMOTE también en la clase mayoritaria se aprecia una mejora en la precisión máxima obtenida en todos los conjuntos de datos. Además, si bien es cierto que esta última alternativa no obtiene prácticamente en ningún caso los mejores valores de precisiones medias, sí que consigue obtener los valores de kappa más altos, por lo que el método presenta una mayor robustez.

Como trabajo futuro, nos planteamos probar otros métodos para tratar la heterogeneidad, como puede ser el caso de las técnicas de submuestreo (undersampling en inglés), ponderación, etc.

Referencias

[1] I. Guyon. *Feature extraction: foundations and applications*, volume 207. Springer, 2006.

[2] L. Morán-Fernández, V. Bolón-Canedo, and A. Alonso-Betanzos. Centralized vs. distributed feature selection methods based on data complexity measures. *Knowledge-Based Systems*, 117:27–45, 2017.

[3] V. Bolón-Canedo, N. Sánchez-Maroño, and Alonso-Betanzos. Distributed feature selection: An application to microarray data classification. *Applied Soft Computing*, 30:136–150, 2015.

[4] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[5] V. Bolón-Canedo, N. Sánchez-Maroño, and J. Cerviño-Rabuñal. Scaling up feature selection: a distributed filter approach. In *Conference of the Spanish Association for Artificial Intelligence*, pages 121–130. Springer, 2013.

[6] V. Bolón-Canedo, N. Sánchez-Marono, and J. Cervino-Rabunal. Toward parallel feature selection from vertically partitioned data. In *Proceedings of ESANN 2014, pp. 395–400*, 2014.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *arXiv preprint arXiv:1106.1813*, 2011.

[8] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, J.M. Benítez, and F. Herrera. A review of microarray datasets and applied feature selection methods. *Information Sciences*, 282:111–135, 2014.

Cuadro IV
Características de los conjuntos del tercer bloque de experimentación, con muestras sintéticas en la clase mayoritaria

Conjunto	Tipo	Nº Muestras	Nº Características	Nº Clases	% clase mayoritaria
Arrhythmia	Normal	452	279	16	54.2
Connect4	Normal	67557	42	3	65.83
Musk2	Normal	6598	168	2	63
Nomao	Normal	34465	120	2	71.44
Ozone	Normal	2536	72	2	97.12
Spambase	Normal	4601	57	2	60.6
Weight	Normal	4024	90	5	34.04
Brain	Microarray	21	12625	2	67
CNS	Microarray	60	7129	2	75
Colon	Microarray	62	2000	2	65
Gli85	Microarray	85	22283	2	69
Ovarian	Microarray	253	15154	2	64

Conjunto		Sin SMOTE	Solo SMOTE minoritaria	SMOTE minoritaria + mayoritaria
Arrhythmia	Precisión (media)	**63.07**	62.63	62.07
	Kappa (media)	0.739	0.746	**0.774**
	Precisión (max)	68.34 (CFS+Naive)	67.68 (CFS+Naive+100)	**68.74** (Rel+SVM+300+40)
	Kappa (max)	**1** (Todos+IB1)	**1** (Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)
Connect4	Precisión (media)	66.14	**66.20**	62.15
	Kappa (media)	0.163	0.430	**0.478**
	Precisión (max)	76.12 (Cons+C4.5)	77.88 (Cons+C4.5+600)	**78.21** (Cons+C4.5+100+20)
	Kappa (max)	0.777 (Cons+C4.5)	**1** (Cons+IB1+Todos)	**1** (Cons+IB1+Todos+Todos)
Musk2	Precisión (media)	**89.41**	87.35	85.17
	Kappa (media)	0.672	0.687	**0.739**
	Precisión (max)	95.62 (Cons+C4.5)	95.44 (CFS+C4.5+100)	**95.72** (Cons+C4.5+100+40)
	Kappa (max)	**1** (Todos+IB1)	**1** (Varios+IB1+Varios)	**1** (Varios+IB1+Todos+Todos)
Nomao	Precisión (media)	**85.75**	84.22	83.94
	Kappa (media)	0.646	0.693	**0.723**
	Precisión (max)	94.34 (Cons+C4.5)	94.52 (Cons+C4.5+300)	**94.70** (Cons+C4.5+Auto+100)
	Kappa (max)	0.964 (Cons+C4.5)	**1** (Cons+IB1+Todos)	**1** (Cons+IB1+Todos+Todos)
Ozone	Precisión (media)	**92.36**	91.88	88.58
	Kappa (media)	0.271	0.506	**0.628**
	Precisión (max)	96.99 (Cons+Todos)	**97.02** (Rel+SVM+600)	**97.02** (Rel+SVM+300+Varios)
	Kappa (max)	0.982 (Info+Rel)	**1** (CFS+IB1+Todos)	**1** (CFS+IB1+Todos+Todos)
Spambase	Precisión (media)	85.92	**87.45**	87.11
	Kappa (media)	0.800	0.858	**0.859**
	Precisión (max)	92.27 (CFS+C4.5)	92.79 (CFS+C4.5+300)	**92.85** (CFS+C4.5+Auto+100)
	Kappa (max)	0.998 (Cons+IB1)	**1** (Cons+IB1+Auto)	**1** (Cons+IB1+Varios+Varios)
Weight	Precisión (media)	84.40	85.44	**86.33**
	Kappa (media)	0.803	0.829	**0.851**
	Precisión (max)	**100** (Cons+Varios)	**100** (Cons+Varios+Todos)	**100** (Varios+C4.5+Todos+Todos)
	Kappa (max)	**1** (Cons+Varios)	**1** (Varios+IB1+Varios)	**1** (Varios+IB1+Varios+Varios)
Brain	Precisión (media)	59.11	**62.27**	62.14
	Kappa (media)	0.882	0.889	**0.904**
	Precisión (max)	82.86 (Info+C4.5)	**94.29** (CFS+C4.5+Todos)	**94.29** (CFS+C4.5+Todos+Todos)
	Kappa (max)	**1** (CFS+Todos, Info+Todos)	**1** (CFS+Todos, Info+Todos)	1 (CFS+Todos, Info+Todos)
CNS	Precisión (media)	55.38	**58.98**	58.13
	Kappa (media)	0.867	0.910	**0.921**
	Precisión (max)	65 (Cons+SVM)	**70** (CFS+C4.5+600)	**70** (CFS+Naive+300+20)
	Kappa (max)	**1** (Todos+IB1)	**1** (Todos+IB1+Todos)	1 (Todos+IB1+Todos+Todos)
Colon	Precisión (media)	**77.62**	77.17	76.81
	Kappa (media)	0.861	0.913	**0.920**
	Precisión (max)	86.67 (Naive+Rel)	87.62 (Info+Naive+Auto)	**88.67** (Rel+Naive+Auto+20)
	Kappa (max)	1 (Todos+IB1)	**1** (Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)
Gli85	Precisión (media)	77.99	**80.40**	80.08
	Kappa (media)	0.944	0.969	**0.976**
	Precisión (max)	85.71 (Info+SVM)	88.57 (Cons+IB1+Varios)	**90** (CFS+IB1+Auto+100)
	Kappa (max)	**1** (Todos+IB1)	**1** (Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)
Ovarian	Precisión (media)	97.68	98.04	**98.09**
	Kappa (media)	0.989	0.994	**0.995**
	Precisión (max)	**100** (CFS+SVM)	**100** (CFS+SVM+Todos)	**100** (CFS+Varios+Varios)
	Kappa (max)	**1** (CFS+SVM, Todos+IB1)	**1** (CFS+SVM+Todos, Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)

Cuadro V
Resumen de los resultados obtenidos utilizando las tres aproximaciones, sin utilizar SMOTE, usando SMOTE sólo en la clase minoritaria–utilización estándard– y usando SMOTE en todas las clases.

Local sets for multi-label instance selection*

Álvar Arnaiz-González
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
alvarag@ubu.es

José-Francisco Díez-Pastor
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
jfdpastor@ubu.es

Juan J Rodríguez
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
jjrodriguez@ubu.es

César García-Osorio
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
cgosorio@ubu.es

***Abstract*—This is a summary of our article published in Applied Soft Computing [1], presented to the Multi-Conference CAEPIA'18 KeyWorks.**

***Index Terms*—multi-label classification, data reduction, instance selection, nearest neighbor, local set**

I. Summary

Single-label classification is a predictive data mining task that consists of assigning a label to an instance for which the label is unknown. Multi-label classification presents a similar task, although the difference is that the instances have a collection of labels, known as a labelset, rather than only one label. The maximum size of the labelset is determined by the number of different labels in the data set. The aforementioned labelset concept can also be considered as a sequence of binary output attributes (as many attributes as there are labels in the whole data set). Each attribute indicates whether the corresponding label is applicable to the instance. Only one of the attributes is active in single-label problems, while several attributes may be active in multi-label problems [5]. In other words, the labels in multi-label learning are not mutually exclusive [8]. This feature implies a much harder and more challenging problem, due to the high relevance of the relations between the different labels [10].

Despite the well-established usefulness of single-label instance selection, there are still very few methods for multi-label classification. To the best of our knowledge, only two instance selection methods for multi-label have been developed. Since both algorithms are based on Wilson Editing (ENN) [9], to avoid any confusion with the acronyms, we refer to them by the initials of their authors: the KADT method [6] and the CRJH method [3]. In this paper, we have attempted to fill that gap by proposing a new technique for computing local sets in multi-label data sets. This new proposal was used to adapt two single-label instance selection methods, LSSm and LSBo, for multi-label problems. The adaptation was tested against the few instance selection methods existing for multi-label learning and against the classifiers (MLkNN [11] and IBLR-ML [4]) trained on the whole data sets.

The main contributions of the paper were:

We would like to thank the *Ministerio de Economía y Competitividad* of the Spanish Government for financing the project TIN2015-67534-P (MINECO/FEDER, UE) and the *Junta de Castilla y León* for financing the project BU085P17 (JCyL/FEDER, UE) both cofinanced from European Union FEDER funds.

- The definition of the local set concept in the context of multi-label data sets.
- The proposal that defines two new instance selection methods, based on the adaptation of single-label classification algorithms to multi-label learning: LSBo and LSSm [7].
- The experimental evaluation of the new algorithms. The new methods were compared with the few existing algorithms.

Instance selection methods usually focus on boundaries between classes. Boundaries are the keystone of the predictive process, because they define whether an instance belongs to one class or another. The simplest classification problem is a binary class data set: there is only one class, thus one instance may or may not belong to it (in practice, this task is similar to determining one of two categories to which the instance belongs). In multi-class classification, more classes are present but, as in the previous case, each instance can only belong to one. The challenge that emerges in multi-label data sets is that instances can belong to more than one class at the same time, which blurs the boundaries (because different labels overlap).

The concept of local set has been used for designing several instance selection algorithms for single-label data sets [2], [7]. Local sets are defined by the nearest enemy, which is straightforward to compute in single-label data sets. The problem with multi-label data sets is how the nearest enemy is defined: it is no trivial task, because every single instance has a set of labels, rather than only one, as in single-label classification. An intuitive solution would be to consider each labelset (the vector of labels of an instance) as a class in itself. However, the results of several experiments have demonstrated that this approach is of little or no use, due to the large amount of different labelsets that multi-label data sets usually have. For example, for a data set with three different classes, the number of different labelsets could be up to $2^3 = 8$; if a data set has nine labels, the number of labelsets could reach $2^9 = 512$. The number of possible labelsets therefore increases exponentially with the number of labels. Hence, local sets calculated in this way will be too small (many of them only made up of a single instance) and, therefore, the algorithms based on local sets would not work properly.

The proposal that was presented in the paper was to use the Hamming loss (calculated over labelsets) to measure the

degree of difference in the labelsets[1]. If the Hamming loss between the labelsets of two instances is greater than a predefined threshold, the instances are considered to be of different '*classes*'. This concept of *class* can be seen as a '*soft-class*' in the same sense as in regression data sets. The Hamming distance is computed as follows:

$$Hamming\ distance(\mathbf{a}, \mathbf{b}) = |\omega_{\mathbf{a}} \triangle \omega_{\mathbf{b}}| \quad (1)$$

where, $\omega_{\mathbf{a}}$ and $\omega_{\mathbf{b}}$ are the labelsets of instances $\mathbf{a}$ and $\mathbf{b}$, respectively, and $\triangle$ is the symmetric difference between two labelsets[2].

The Hamming distance according to the previous definition is a whole number. The Hamming loss value is commonly used in multi-label learning HL $\in [0, 1]$.

$$Hamming\ loss(\mathbf{a}, \mathbf{b}) = \frac{1}{|\Omega|}|\omega_{\mathbf{a}} \triangle \omega_{\mathbf{b}}| \quad (2)$$

Pseudocode 1 shows the proposed method for local set calculation in multi-label data sets. It has two inputs: the multi-label data set and the value of the Hamming loss threshold that determines when two labelsets are considered distinct. The function has two outputs: an array of local sets and an array of nearest enemies. Every single instance has its local set (made of one or more instances) and its nearest enemy.

Algorithm 1: Function `computeLocalSets`: computes the local sets of a multi-label data set.

Input: A training set $X = \{(\mathbf{x}_1, \omega_1), ..., (\mathbf{x}_n, \omega_n)\}$, a threshold θ

Output: The local sets $LSS = \{\mathbf{lss}_1, ..., \mathbf{lss}_n\}$, the nearest enemy of each instance $NE = \{ne_1, ..., ne_n\}$

```
1  for i ∈ {1...n} do
2      lss_i ← ∅
3      dist_ne_i ← ∞
       /* Find the nearest enemy of x_i      */
4      for j ∈ {1...n} do
5          d ← EuclideanDistance(x_i, x_j)
6          if HammingLoss(ω_i, ω_j) > θ and d < dist_ne_i
           then
7              ne_i ← x_j
8              dist_ne_i ← d
       /* Compute the local set of x_i       */
9      for j ∈ {1...n} do
10         if EuclideanDistance(x_i, x_j) < dist_ne_i then
11             lss_i ← lss_i ∪ {x_j}
```

After the calculation of local sets, any local set-based algorithm can be used without changes. In the experimental study, we considered LSSm and LSBo, because their use of local sets is more robust than the use of local sets in ICF (the heuristic used in ICF has fundamental problems that were reported in [7]).

The experimental study used a broad range of data sets from different domains, several multi-label measures and statistical tests. The results revealed the two main benefits of our proposal: *i*) HDLSSm, as an edition algorithm, is not only capable of outperforming the other instance selection methods in terms of its results, but it also capable of outperforming the classifier trained with the whole data set; *ii*) HDLSBo, as a condensed algorithm, achieved a remarkable compression, while maintaining a statistically equivalent performance to the performance of the other methods. Furthermore, the existence of a threshold for controlling local set sizes implies an adaptable and versatile proposal.

[1]We decided to use Hamming loss, because its computation is fast and it is a commonly used measure in multi-label learning.

[2]The symmetric difference is the exclusive disjunction (XOR) of two sets, that is the set of all elements that are in one set, but not in the other set.

References

[1] Álvar Arnaiz-González, José F. Díez-Pastor, Juan J. Rodríguez, and César García-Osorio. Local sets for multi-label instance selection. *Applied Soft Computing*, 68:651 – 666, 2018.

[2] Henry Brighton and Chris Mellish. *On the Consistency of Information Filters for Lazy Learning Algorithms*, pages 283–288. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[3] Francisco Charte, Antonio J. Rivera, María J. del Jesus, and Francisco Herrera. MLeNN: A first approach to heuristic multilabel undersampling. In *Intelligent Data Engineering and Automated Learning – IDEAL 2014: 15th International Conference, Salamanca, Spain, September 10-12, 2014. Proceedings*, pages 1–9, Cham, 2014. Springer International Publishing.

[4] Weiwei Cheng and Eyke Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2):211–225, Sep 2009.

[5] Francisco Herrera, Francisco Charte, Antonio J. Rivera, and María J. del Jesus. *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer Publishing Company, Incorporated, 2016.

[6] Sawsan Kanj, Fahed Abdallah, Thierry Denœux, and Kifah Tout. Editing training data for multi-label classification with the k-nearest neighbor rule. *Pattern Analysis and Applications*, 19(1):145–161, 2016.

[7] Enrique Leyva, Antonio González, and Raúl Pérez. Three new instance selection methods based on local sets: A comparative study with several approaches from a bi-objective perspective. *Pattern Recognition*, 48(4):1523 – 1537, 2015.

[8] Newton Spolaôr, Maria Carolina Monard, Grigorios Tsoumakas, and Huei Diana Lee. A systematic review of multi-label feature selection and a new method based on label construction. *Neurocomputing*, 180:3 – 15, 2016. Progress in Intelligent Systems Design Selected papers from the 4th Brazilian Conference on Intelligent Systems (BRACIS 2014).

[9] Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-2(3):408–421, July 1972.

[10] Zoulficar Younes, Fahed Abdallah, Thierry Denœux, and Hichem Snoussi. A dependent multilabel classification method derived from the k-nearest neighbor rule. *EURASIP Journal on Advances in Signal Processing*, 2011(1):1–14, 2011.

[11] Min-Ling Zhang and Zhi-Hua Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038 – 2048, 2007.

Emerging topics and challenges of learning from noisy data in non-standard classification: A survey beyond binary class noise

Ronaldo C. Prati
Center of Mathematics, Computer Science and Cognition
Federal University of ABC
Santo André, São Paulo, Brazil
ronaldo.prati@ufabc.edu.br

Julián Luengo
Data Science and Computational Intelligence Institute
University of Granada
Granada, Spain
julianlm@decsai.ugr.es

Francisco Herrera
Data Science and Computational Intelligence Institute
University of Granada
Granada, Spain
herrera@decsai.ugr.es

***Abstract*—This is a summary of our article published in Knowledge and Information Systems [1] to be part of the MultiConference CAEPIA'18 Key Works.**

***Index Terms*—Data preprocessing, non-standard classification, noise data, multiclass classification, multi-instance learning, multi-label learning, multitask problems, ordinal classification, data streams, non-stationary environments**

I. Summary

Learning from noisy data is an important topic in machine learning, data mining and pattern recognition, as real world data sets may suffer from imperfections in data acquisition, transmission, storage, integration and categorization. Indeed, over the last few decades, noisy data has attracted a considerable amount of attention from researchers and practitioners, and the research community has developed numerous techniques and algorithms in order to deal with the issue [2]–[4].

These approaches include the development of learning algorithms which are robust to noise as well as data pre-processing techniques that remove or "repair" noisy instances. Although noise can affect both input and class attributes, class noise is generally considered more harmful to the learning process, and methods for dealing with class noise are becoming more frequent in the literature [3].

Class noise may have many reasons, such as errors or subjectivity in the data labeling process, as well as the use of inadequate information for labeling. For instance, in some medical applications, the true status of some diseases can only be determined by expensive or intrusive procedures, some of which can only be carried out after a patient's death. Another reason is that data labeling by domain experts is generally costly, and several applications use labels which are automatically defined by autonomous taggers (e.g., sentiment analysis polarization [5]), or by non-domain experts. This approach is common in, e.g., social media analysis [5], where hashtags used by users or information provided by a pool of non-domain experts (crowdsourcing) are used to derive labels.

Even though class noise is predominant in the literature (see [2], [6] for recent surveys and comparison studies), most of the research has been focused on noise handling in binary class problems. However, new real-life problems have motivated the development of classification paradigms beyond binary classification [7]. These paradigms include ordinal class [8], multiclass [9], multilabel [10] and multi-instance [11] as well as learning from data streams and non-stationary environments [12] and joint exploiting related tasks [13]. Due to the ubiquity of noise, it is of fundamental importance to better understand the relationships and implications of class noise within these paradigms. Each paradigm has its own particularities which impose new challenges and research questions for noise handling. Although research for class noise handling in these paradigms is somewhat present in the literature, it remains quite scarce and requires general discussion of issues, challenges and research practices regarding it.

The related paper aims to discuss open-ended challenges and future research directions for learning with class noise data, focusing on the aforementioned non-binary classification domains. The main contributions of such a paper are:

- We discuss some current research, as well as the need of adaptation or development of new techniques for handling class noise within non-binary classification paradigms.
- We also discuss issues related to the simulation of noise scenarios (inclusion of artificial noise) within these paradigms, an experimental artifact frequently adopted

This work have been partially supported by the São Paulo State (Brazil) research council FAPESP under project 2015/20606-6, the Spanish Ministry of Science and Technology under project TIN2014-57251-P and the Andalusian Research Plan under project P12-TIC-2958.

for analysis of noise dealing techniques. These issues are important for simulating noise scenarios that may occur in real world applications, and can serve as the basis for uniforming procedures by providing an objective ground in order to assess the robustness of the learning methods.
- We present some important open-ended issues and offer some possible solutions to the existing problems.

We are aware of some studies already considering multiclass noise problems. Different multiclass noise patterns impose numerous challenges, some of them infrequently addressed in the literature. Even state of the art methods for dealing with binary class noise present considerable variation in performance when considering different multiclass noise patterns at the same noise ratio. Despite this, these issued are seldom considered in the literature. In the related paper we focus on some of aspects that could be studied further, providing a guideline of open challenges for researchers, such as:

- Would these different types of noise patterns pose the same or different challenges when dealing with multiple class noise?
- Which one would be more difficult to tackle?
- Which aspects of the problem would be more affected by considering different noise multiclass pattern?
- How do existing methods behave considering these different noise patterns?

One interesting topic for further research is how to extend methods, originally developed only for binary class, to the multiclass case. Some data transformation approaches for transforming multiclass to binary problems, e.g., One-versus-One (OVO) or One-versus-ALL (OVA), could be applied [14]. However, research on this topic generally involves random noise completely at random, with uniform class noise distribution. Investigating how these approaches are affected by different noise patterns is an interesting topic for research. For instance, when applying a filter using a OVA decomposition, does the order in which class noise is removed matter? If so, is this influence stronger for different noise distribution among the classes?

Another open-ended problem is the relationship with imbalanced classification [15] and multiclass noise. It is reported in the literature that noise in minority classes is more harmful than in majority classes [16]. However, multiclass imbalance [17] has further issues to consider, as multiple predominant or infrequent classes may occur. It is unclear what learning difficulties multiclass noise can cause under highly imbalanced class distributions, and how to handle it effectively is an open-ended issue. Furthermore, different noise patterns can change the observed class ratio, which may influence the behavior of class imbalance techniques. Uniform class noise, for instance, may mask the observed class ratio of multiple rare classes even for low noise levels. Default class may also introduce an artificial predominant class, thus generating an artificial imbalanced problem due to the presence of noise. Possible ways to handle noise in imbalanced problems include cost sensitive noise handling [18], [19], attributing and the development of class ratio aware filtering approaches [20] considering the multiclass context.

We believe this discussion will encourage researchers and practitioners to explore the problem of class noise handling in new scenarios and different learning paradigms in more detail.

References

[1] R. C. Prati, J. Luengo, and F. Herrera, "Emerging topics and challenges of learning from noisy data in non-standard classification: A survey beyond binary class noise," *Knowledge and Information Systems*, vol. in press, 2018.

[2] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," vol. 25, no. 5, pp. 845–869, 2014.

[3] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 177–210, 2004.

[4] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.

[5] B. Liu, *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge University Press, 2015.

[6] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artif. Intell. Rev.*, vol. 33, no. 4, pp. 275–306, 2010.

[7] J. Hernández-González, I. Inza, and J. A. Lozano, "Weak supervision and other non-standard classification problems: a taxonomy," *Pattern Recognit. Lett.*, vol. 69, pp. 49–55, 2016.

[8] P. A. Gutiérrez, M. Perez-Ortiz, J. Sanchez-Monedero, F. Fernández-Navarro, and C. Hervas-Martinez, "Ordinal regression methods: survey and experimental study," vol. 28, no. 1, pp. 127–146, 2016.

[9] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of artificial intelligence research*, vol. 2, pp. 263–286, 1995.

[10] F. Herrera, F. Charte, A. J. Rivera, and M. J. del Jesus, *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer, 2016.

[11] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Sánchez-Tarragó, and S. Vluymans, *Multiple Instance Learning: Foundations and Algorithms*. Springer, 2016.

[12] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: a survey," vol. 10, no. 4, pp. 12–25, 2015.

[13] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 615–637, 2005.

[14] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition," *Knowledge and information systems*, vol. 38, no. 1, pp. 179–206, 2014.

[15] R. C. Prati, G. E. A. P. A. Batista, and D. F. Silva, "Class imbalance revisited: a new experimental setup to assess the performance of treatment methods," *Knowl. Inf. Syst.*, vol. 45, no. 1, pp. 247–270, 2015.

[16] J. Van Hulse and T. Khoshgoftaar, "Knowledge discovery from imbalanced and noisy data," *Data Knowl. Eng.*, vol. 68, no. 12, pp. 1513–1542, 2009.

[17] S. Wang and X. Yao, "Multiclass imbalance problems: Analysis and potential solutions," vol. 42, no. 4, pp. 1119–1130, 2012.

[18] X. Zhu and X. Wu, "Cost-guided class noise handling for effective cost-sensitive learning," in *IEEE International Conference on Data Mining (ICDM)*. IEEE, 2004, pp. 297–304.

[19] X. Zhu, X. Wu, T. M. Khoshgoftaar, and Y. Shi, "An empirical study of the noise impact on cost-sensitive learning." in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 7, 2007, pp. 1168–1173.

[20] J. A. Sáez, J. Luengo, J. Stefanowski, and F. Herrera, "Smote–ipf: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering," *Inf. Sci.*, vol. 291, pp. 184–203, 2015.

Training Set Selection for Monotonic Ordinal Classification

José-Ramón Cano
Dept. of Computer Science
University of Jaén
Linares, Spain
jrcano@ujaen.es

Salvador García
Dept. of Computer Science and Artificial Intelligence
University of Granada
Granada, Spain
salvagl@decsai.ugr.es

***Abstract*—This is a summary of our article published in Data & Knowledge Engineering [1] to be part of the MultiConference CAEPIA'18 KeyWorks.**

***Index Terms*—Monotonic Classification, Ordinal Classification, Training Set Selection, Data Preprocessing, Machine Learning**

I. Summary

Learning with ordinal data sets has increased the attention of the machine learning community in recent years. These data sets are characterized by the presence of an ordinal output and they are commonly found in real life.

Monotonic classification is an ordinal classification problem where monotonic constraints are present in the sense that a higher value of a feature in an instance, fixing the other values, should not decrease its class assignment [2]. Monotonicity is a property commonly found in many environments of our lives like economics, natural language or game theory [4]. A classical example of monotonicity is in the case of bankruptcy prediction in companies, where appropriate actions can be taken in time, considering the information based on financial indicators taken from their annual reports. The comparison of two companies where one dominates the other on all financial indicators shows clearly where the monotonicity is present, which supposes that the overall evaluation of the second cannot be higher than the evaluation of the first. This strategy could be applied to the credit rating score used by banks as well as for the bankruptcy prediction strategy .

In the specialized literature we can find multiple monotonic classifiers proposed. As a restriction, some of them require the training set to be purely monotone to work properly. Other classifiers can handle non-monotonic data sets, but they do not guarantee monotone predictions.

In addition, real-life data sets are likely to have noise, which obscures the relationship between features and the class. This fact affects the prediction capabilities of the learning algorithms which learn models from those data sets.

This work was supported by TIN2014-57251-P, by the Spanish "Ministerio de Economía y Competitividad" and by "Fondo Europeo de Desarrollo Regional" (FEDER) under Project TEC2015-69496-R and the Foundation BBVA project 75/2016 BigDaPTOOLS.

In order to address these shortcomings and to test the prediction competences of the monotonic classifiers, the usual trend is to generate data sets which completely satisfy the monotonicity conditions. The intuitive idea behind this is that the models trained on monotonic data sets should offer better predictive performance than the models trained on the original data. In the specialized literature, we find two possible techniques to generate monotonic data sets. Monotonic data sets can be created by generating artificial data [5] and by relabeling the real data [6]. The latter restores the monotonicity of the data set by changing the class labels in those instances which violate the monotonicity constraints. Class relabeling is the only approach which can be applied in real life data sets, and has shown promising results in the literature.

As an alternative to relabel, Training Set Selection (TSS) is known as an application of instance selection methods [3] over the training set used to build any predictive model. The effects produced by TSS are: reduction in space complexity, decrease in computational cost and the selection of the most representative instances by discarding noisy ones.

In this paper we propose a TSS algorithm to manage monotonic classification problems, called Monotonic Training Set Selection (MonTSS). MonTSS can be considered as the first in the literature for performing TSS in monotonic classification problems. It is a data preprocessing technique which, by means of a suitable TSS process for monotonic domains, offers an alternative without modifying the class labels of the data set, it instead removes harmful instances. MonTSS incorporates proper measurements to identify and select the most suitable instances in the training set to enhance both the accuracy and the monotonic nature of the models produced by different classifiers.

The whole process is presented in Figure 1, and as can be seen it is composed of three stages:

1) The MonTSS process starts with a preprocessing step where MonTSS analyzes the original data set by quantifying the relationship between each input feature and the output class. This relation is estimated with a metric called Rank Mutual Information (RMI). With it, we

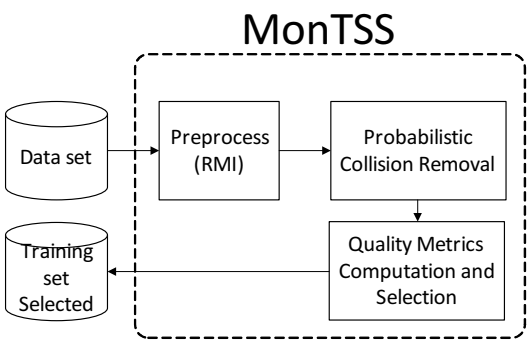


Fig. 1. MonTSS process.

know the features which have a real direct or inverse monotonic relation with the class or no relation as well (including unordered categorical features). The RMI value is evaluated in the training data set to decide which features are used in the computation of collisions between instances.

In essence, rank mutual information can be considered as the degree of monotonicity between features A_1,...,A_f and the feature class Y. Given any feature A_j and feature class Y, the value of RMI for the feature A_j is calculated as follows:

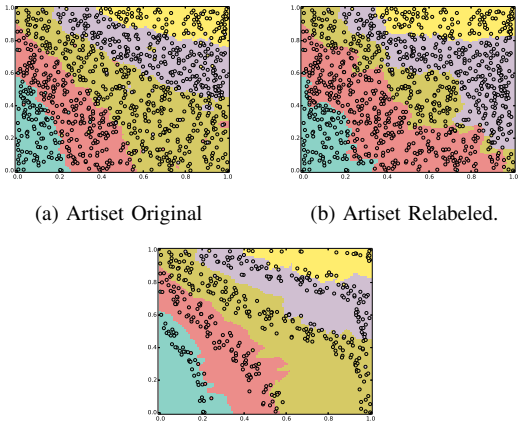
$$\text{RMI}(A_j, Y) = -\frac{1}{n}\sum_{i=1}^{n} log \frac{\#[x_i]^{\leq}_{A_j} \cdot \#[x_i]^{\leq}_{Y}}{n \cdot \#([x_i]^{\leq}_{A_j} \cap [x_i]^{\leq}_{Y})} \quad (1)$$

where n is the number of instances in data set D, $[x_i]^{\leq}_{A_j}$ is the set formed by all the instances of the set D whose feature A_j is less or equal than feature A_j of instance x_i, and $[x_i]^{\leq}_{Y}$ is the set composed of the instances of the set D whose feature class Y is less or equal than feature class Y of instance x_i.

2) In the second stage, the probabilistic collision removal mechanism is applied, which eliminates most of the instances which produce collisions. The remaining instances are used as input in the last stage.
3) Here, the quality metrics are computed and based on them, the selection procedure is developed considering the following rule:

$$\text{Select } x_i = \begin{cases} \text{true} & \textbf{if } \text{Del}(x_i) < \text{Infl}(x_i) \\ & \text{or } \text{Del}(x_i) \geq 0.9 \\ \text{false} & \text{otherwise.} \end{cases} \quad (2)$$

The rationale behind this rule is to retain the instances which are closer to the class boundaries, using a straightforward threshold of 0.9 which is independent from the diversity of their neighborhood. Furthermore, a relationship between $\text{Del}(x_i)$ and $\text{Infl}(x_i)$ can be easily established as they represent a measurement in the same range of the relative rate of the situation and the neighborhood variety of every instance. In this respect, the rule is built as a function of both measures. As a result, for instance, the rule preserves the instances belonging to central areas if there are instances of other classes around.



(a) Artiset Original (b) Artiset Relabeled.

(c) Artiset MonTSS.

Fig. 2. Artificial data set (Artiset) preprocessed by Relabeling and MonTSS with the borders calculated by MkNN with 3 neighbors.

We have compared the results offered by well-known classical monotonic classifiers over 30 data sets with and without the use of MonTSS as a data preprocessing stage. As graphical example of use we present the Fig. 2.

The results show that MonTSS is able to select the most representative instances, which leads monotonic classifiers to always offer equal or better results than without preprocessing.

MonTSS is able to select the most representative instances independently of the classifier to be applied later. This leads monotonic classifiers to always offer equal or better results than without preprocessing. Furthermore, data related metrics are notably improved, fully satisfying the monotonicity restrictions without affecting or modifying the nature of the original data. At the same time, it reduces the number of non-comparable pairs of instances and the size of the training data sets before the learning stage starts.

References

[1] J.-R. Cano and S. García, "Training set selection for monotonic ordinal classification," Data & Knowledge Engineering, vol. 112, pp. 94–105, 2017.
[2] H. Daniels and M. Velikova,"Monotone and partially monotone neural networks," IEEE Transactions on Neural Networks, vol. 21, num. 6, pp. 906–917, 2010.
[3] J.-R. Cano, N.R. Aljohani, R.A. Abbasi, J.S. Alowidbi and S. García, "Prototype selection to improve monotonic nearest neighbor," Engineering Applications of Artificial Intelligence, vol. 60, pp. 128–135, 2017.
[4] W. Kotłowski and R. Słowiński, "On nonparametric ordinal classification with monotonicity constraints," IEEE Transactions on Knowledge and Data Engineering, vol. 25, num. 11, pp. 2576–2589, 2013.
[5] R. Potharst, A. Ben-David and M.C. van Wezel, "Two algorithms for generating structured and unstructured monotone ordinal data sets," Engineering Applications of Artificial Intelligence, vol. 22, num. 4-5, pp- 491–496, 2009.
[6] M. Rademaker, B. De Baets and H. De Meyer, "Optimal monotone relabelling of partially non-monotone ordinal data," Optimization Methods and Software, vol. 27, num. 1, pp. 17–31, 2012.

Data source analysis in mood disorder research

Pavél Llamocca Portella
Universidad Complutense de Madrid
pavel.llamocca@hotmail.com

Axel Junestrand
Universidad Complutense de Madrid
axel.junestrand@hotmail.com

Milena Čukić
University of Belgrade
milena.cukic@gmail.com

Diego Urgelés
Hospital Ntra. Sra. De la Paz
diego.urgeles@gmail.com

Victoria López López
Universidad Complutense de Madrid
vlopezlo@ucm.es

***Abstract*— Mood disorders have been a relevant topic for the last few years. Nowadays, there are projects in the mental health area which are supported by technological devices that improve the efficiency of treatments by effortlessly allowing the gathering of biological and psychological indicators from patients. One of the goals of this document is to describe the most common methods for collecting most of those indicators and to study which of them can be applied to the Bip4Cast project. The purpose of this article is to analyze the sources of information that have been used successfully in the study of emotional disorders as well as alternative sources of information from the monitoring of movement and sounds in the patient's environment. This article shows the results of the analysis of traditional information sources. The results show a lack of precision in the data on fundamental variables such as sleep quality and motor activity. Therefore, the study demonstrates the need to include new sources of information to increase the quality of the data before applying crisis prediction algorithms. The need to monitor the sleep and movement of patients in order to achieve a sufficient quality in the source data from the evolutionary analysis of patients is concluded.**

Keywords—bipolar disorder; mood disorder; data gathering; machine learning; data analysis.

I. Introduction

The quality of treatments in mood disorders has acquired a high attention for researchers in the mental health field. However, despite current efforts, there is still plenty of room for improvement. Many practicians agree that knowing how patients react to treatments in advance and predicting when their mood could vary significantly are two of the most important issues to solve in order to ensure the quality of new treatments.

The proposal of the Bip4Cast project is to keep using the current monitoring of personal sessions between patients and psychiatrists, but also to add new data sources and their analysis to improve the prediction of Bipolar Disorder crises in patients. The main idea is to get advantage of the new developments in data gathering, data cleaning, and Machine Learning to monitor a set of patients and make a new approach with these data. The patients are encouraged to follow certain methods for gathering psychological and biological indicators during a particular period of time. The goal is to analyze the data gathered in order to find some common patterns that could trigger a crisis. For the process of pattern detection, some Machine Learning tools and mathematical models are being used.

The goal of this document is to cover a discussion about some methods for gathering indicators and the feasibility of their usage in this project. Apart from this introduction, this document includes the following sections: section 2 presents the state of the art related research in mood changes and patient monitoring. In section 3, several methods for gathering psychological and biological parameters are described. Section 4 covers the preliminary analytics on the Bip4Cast data sets and, finally, section 5 includes the conclusions.

II. State of the Art

Several studies about Bipolar Disorder state that a relationship exists between the different behaviors of the patients before the occurrence of a crisis [1-2]. For example, during a manic or a depressive crisis, some of these studies agree that sleeping rates are very important indicators. Vocal features as well as the rate of speech are other important indicators and there are some studies stating that the pitch is lower in a depressed state [3]. Also, parameters like the time of exposure to dark or sunny places and the physical activity are considered.

In [4], the author introduces a mobile health system using several sensors for mood detection. In [5], the author presents a research that includes Machine Learning models in a mobile application in order to estimate the mood in depressive patients. However, no objective psychological or psychiatric markers are considered due to the recording of the data being done manually by patients. Furthermore, there is an interesting study which describes the use of electroencephalography for the gathering of brain signals. It also uses non-linear features

like Higuchi's Fractal Dimension and Sample Entropy to feed different Machine Learning methods [6]. In [7], a mobile application is presented for supporting the treatments of patients with Bipolar Disorder. Its key is to compare objective and subjective data. It records objective data using some features given by mobile phones like accelerometers and phone call rates. This information is used for predicting trends in the mood of the patients. However, the focus of this application is to record subjective data using a self-reporting approach.

At the present time, there are some projects within the scope of mental health which have similar approaches and try to obtain certain markers from which a common pattern can be inferred to help with the treatments. One of these projects is PSYCHE [8], whose main idea is for the patients to use a special garment made of a proper material, of which the main goal is to collect parameters from the patient in his/her daily life. The outcomes of PSYCHE are positive. However, patients said that the main inconvenient was to use the same clothes all the time, which implicates a high discomfort for the patient and therefore its non-use. Another really interesting project is MONARCA [9]. It emphasizes the use of mobile phones for the electronical monitoring of patients. The number of parameters that can be obtained through the use of a mobile phone is really high, but nevertheless, none of them are physiological parameters. Furthermore, after 3 years of activity with this application, an analysis of some non-functional requirements for the treatment of patients with Bipolar Disorder concluded that, for new developments, some details have to be taken into consideration in order to improve the ease of use, e.g.: ensuring that the patients have a data plan for their 3G connection, the need for teaching the clinicians how to operate the system, and the overheating of the smart phone from the use of an application that requires both GPS and Bluetooth.

Since few years ago, Body Sensor Networks (BSN) have made an appearance, which are a branch of wireless sensor networks (WSNs) that conform one of the core technologies of IoT developments in the healthcare system [10]. Its purpose is to provide an integrated hardware and software platform which facilitates the future development of pervasive monitoring systems. BSN allow the monitoring of patients by using a collection of tiny-powered and lightweight wireless sensor nodes. These nodes are placed on the skin and sometimes integrated with different garments, so that the patient's health-related data can be collected and transferred to the healthcare staff in real time. However, the development of this new technology in healthcare applications without considering security makes patient privacy vulnerable. For this reason, several research projects are currently being carried out to try to cover this vulnerability [11].

III. Data Source Analysis

All the research that is currently being conducted suggests a wide variety of indicators for taking mood disorder treatments into account. The indicators themselves and the way in which they are collected are strongly related. In this section, several data sources for gathering those indicators are described. Depending on the indicator they gather or the type of device, they are classified into 6 groups as shown below.

1. Smart wristband/smart band. These devices include a set of sensors which measure daily activity by means of accelerometers. They create variables such as an activity tracker (resting, moving or sleeping), a pedometer (steps taken and distance traveled), the calories burned, a sleep monitor (awake, slight and deep sleeping), the heart rate and the blood oxygen level. The most relevant variable measured from this type of device is the sleep indicator. Almost all researchers agree that sleep quality is the best indicator in Bipolar Disorder treatments. The CHOICE [12] study states that lower levels of depression are correlated with improvements in insomnia treatments, and on the other hand, high levels of mania are correlated with less need for sleep. Furthermore, a pilot randomized controlled trial demonstrated that sleep disturbance appears to be an important pathway contributing to Bipolar Disorder [13]. These data can easily be gathered from popular apps. Fig. 1 (a) shows one of the monitors used in the Bip4Cast project.

Fig. 1. Two sleep monitors in Bip4Cast (a- Garmin Vivofit3; b- Sleep Cycle for iPhone)

2. Medical bands. They allow measuring more parameters because they usually include more hardware and better features like atmospheric pressure (barometer), GPS location and magnetometer. There is some research which links episodes to disturbances in circadian rhythms and lifestyle regularity. Those indicators can be collected through these devices using the activity tracker or gyroscope. Furthermore, this research suggests that methods for tracking behavior, nutrition, blood pressure and lipid profile as well as physical/social activity and sleep-awake routines may improve treatments. In the Bip4Cast project we are using GENEActif v.1.2 for a total of 25 patients, (see [14] for more details about their use in Bip4Cast).

3. Mobile Sensors. In this group, any other kind of sensors that can be worn by the user on any other part of the body is included, e.g. for wearing on the leg, ActivPAL is a kind of device used to investigate the correlation between physical behaviors and chronic disease [15]. For using as a necklace, LeafUrban is an option (there are some versions for wearing on the wrist or attached to the clothes). It is a device designed for women and what it makes different from other devices is the tracking of the menstrual cycle, the fertility and the breathing [16]. For wearing on the head, ELF Emmit is a headband that helps the user improve the state of both mind and body by

using pulsed electromagnetic stimulation (PEMS) [17]. Relevant variables include skin and breath changes, electrocardiogram and respirogram data, stress level and menstrual cycle among others.

4. Sleep Activity Recording Devices. In this group, any device specialized in recording sounds and activities during sleep is included. There are hundreds of mobile applications that record sounds during sleep. One of the main objectives is to detect snoring, for which four of the most popular applications at the moment are SleepGenius, SleepCycle (see Fig. 1 (b)), SleepBot and SleepTime. However, there are other kinds of devices with different non-invasive designs, e.g. devices attached to the mattress which can track sounds as well as heart rate, breathing, movement, etc. The reason for using these methods is to improve sleeping conditions. Almost all of these methods have one parameter in common: "breathing", which allows the detection of snoring. Habitual snoring is a prevalent condition that is not only a marker for Obstructive Sleep Apnea (OSA) but can also lead to vascular risks [18]. Some researchers have found a relationship between OSA and Major Depressive Disorder/Bipolar Disorder.

5. Forms and Questionnaires. This group contains any method which uses a questionnaire or a form for the self-reporting of mood. In current literature, these methods were designed by psychiatrists and are presented as scales. There are several scales for detecting the risk of a euphoria episode outbreak: Altman Self-Rating Mania Scale (ASRM) [19], the Clinician-Administered Rating Scale for Mania (CARS-M), the Internal State Scale (ISS), the Self-Report Manic Inventory (SRMI), etc. For depression episodes, there are several scales, like the Patient Health Questionnaire (PHQ-9) [20]. All of them consist in questionnaires which can be performed by patients. This presents the opportunity of developing digital forms based on these patients in order to facilitate their use.

Scales for detecting the risk of euphoria or mania episodes, like the Young Mania Rating Scale (YMRS) and the Bech-Rafaelsen Mania Scale (MAS), or the Hamilton Depression Rating Scale (HDRS), which detects the risk of depression, are not included because they are performed by the clinician (however, for the scope of this project, these scales are included in normal monitoring sessions). All of these questionnaires collect variables from which it is possible to measure the presence and severity of mania, depression, affective, psychological and somatic symptoms. Fig. 2 shows the interface of an application developed for collecting these data. All the details about this work are in [21].

6. Mobile Apps / Time Consumption. This group includes mobile applications that support BPD treatments and/or record smart phone use. For the aim of this project, these mobile applications were classified into two subclasses: the first one, named Bipolar Disorder Apps (BPDA) in this document, includes any applications that have been developed for supporting the treatment itself, and the second one, named Time Consumption Apps (TCA) in this document, includes any application that is able to measure the time spent using a specific mobile application.

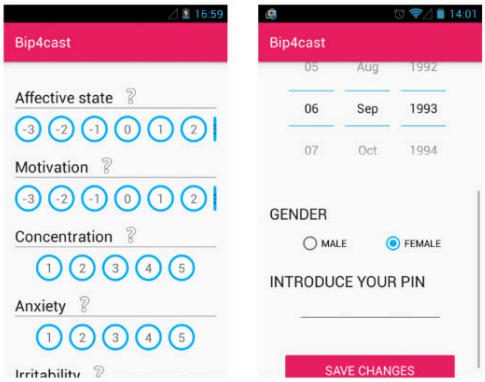


Fig. 2. First version of the app for collecting daily personal data

7. Conventional methods. Finally, patients are also being assessed periodically through interviews. This evaluation is done by psychiatrists in medical centers. For the patient, this does not imply any kind of alteration in the current treatment. However, the procedure will need the psychiatrist to send the collected data from those sessions to the data server. It is important to mention that in this phase, the Young Mania Rating Scale (YMRS) and the Hamilton Depression Rating Scale (HRDS) are included for detecting mania or depression episodes.

Also, psychiatrists can take advantage of these interviews for downloading the data recorded from wristbands and medical bands in order to later send them to a dedicated server (just in case these devices are not able to send the recorded data automatically).

IV. THE BIP4CAST PROJECT

Patients with Bipolar Disorder are characterized by a behavior which is difficult to predict. There is great deal of information which can be retrieved from biological, physiological and physical signals in order to detect episodes. Knowing which variables are correlated and which features or parameters are important is essential to build a model that will successfully predict the target of a study. The aim of this study is to investigate which features have the highest importance in health. In order to achieve this, Machine Learning algorithms and techniques are used for feature ranking.

The data used for this project is anonymized patient data gathered by psychiatrists at Clínica Nuestra Señora de la Paz in Madrid. All the data were available in an Excel file with different sheets. Even though 25 patients are already wearing a medical band (GENEActif 1.2) and we have developed an application for gathering their daily activity, for this study most of the data have been gathered in a supervised manner during medical appointments with four different patients that suffer from Bipolar Disorder. The goal for the future is that these data are both recorded by the psychiatrists in appointments and with the help of mobile applications. This way, the patients can actively participate in their own diagnosis. The data consist of 4 data sets: Episodes, which represents different episode periods in the patients (depression/mania) from a total of four

patients; YMRS data set, which contains Young Mania Rating Scale [22] data (to assess mania symptoms) from a total of 48 days; HDRS data set, which contains Hamilton Depression Rating Scale [23] data (for depression), also from a total of 48 days; Interview data set, which contains 728 registers about physical and psychological items, the latter including variables like anxiety, irritability or concentration problems, and the former including more objective data, as could be the number of cigarettes smoked by the patient or the time in which the patient woke up or went to bed. The last data set used in the study is Interventions. It includes data about all the medical interventions that different doctors have had with the patients, in a total of 92 registers. For the gathering of data included in the Interview data set, a mobile application [21] has been developed, which patients can use daily to store quantitative data (number of cigarettes, menstruation, etc.) and qualitative data (feeling of stress, anxiety, etc.). In the project, we have also included studies with data from a medical bracelet (GENEActif 2.1 for 25 patients) and an application for recording night sounds. The data collected by these last two exercises will be included in subsequent studies.

The programming language used in this project is Python 2.7, which has a lot of libraries that make data cleaning and visualization less complicated, as well as applying Machine Learning algorithms. The environment used is Jupyter Notebook [24]. Scikit-learn [25] is the Machine Learning library of choice for this project because it includes preprocessing and cross-validation tools as well as all the known baseline Machine Learning algorithms. This project is shared in a public GitHub repository, which can be found at [26].

A. *Data Cleaning*

The first step of the project consisted in the data cleaning which included the gathering of the data that we would be working with. In order to gather the data, we exported each sheet, from the Excel file that was given to us by the hospital, to csv format. The initial Excel file was divided into five different sheets: Episodes, YMRS, HDRS, Interview Data Set (IDS) and Interventions. In order to export them to a format readable by Pandas [27], we saved each sheet as CSV UTF-8 in Microsoft Excel. Some other improvement was done in relation to data cleaning: filling the empty values, converting them from Float to Integer and data type revision.

B. *Exploratory Data Analysis*

After the data cleaning, we performed an Exploratory Data Analysis, in order to visualize how the data behaved. We used histograms, heatmaps and scatterplots in this part. For instance, the YMRS data set correlation heatmap showed that aggressiveness and verbal expression were correlated. This could mean that if the patient talks a lot (excessive speech rate), this behavior would probably be accompanied by excessive energy or hyperactivity (Disruptive-Aggressive Behavior). The scattterplot matrix from the YMRS data showed that hyperactivity and irritability have a similar distribution as well as a correlation between verbal expression and euphoria. The HDRS data set analysis showed a similar distribution between suicide and precocious insomnia (difficulty of sleep early in the night). The HDRS data set correlation heatmap showed a high correlation between depressed mood and work: the less a patient is willing to work or do other activities, the more depressed he or she will probably feel. At this stage of the research, the best data set regarding both size and accuracy was the Interview Data Set (IDS). During its analysis, we found a clear linear relationship between the variables mood and motivation. The Intervention data set presented a lack of correlation between the level of relief in a patient and the GAF (Global Assessment of Functioning). A good summary of all these results can be read at [26].

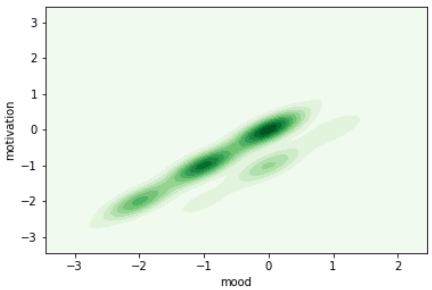


Fig. 3. 2D kernel density plot of *mood* and *motivation* in Interview data set

C. *Data Combination*

The goal of this phase was to find data set combinations that had enough data for the algorithms to process so that we could later see which data sets returned the highest accuracy. In order to get the combinations right, we defined a function that obtained the date of each entry and compared it with the different episodes of depression and mania in the Episode data set, which was the target of the prediction. For the entries or rows that were not recorded in the Episode data set, we assumed that the patient was in a euthymic state. This way, we got three possible states that a patient could be in: Depression (D), Mania (M) and Euthymic (N). For instance, with the HDRS and Episode combination we could see that when the patients had a depression episode, the value of depressed mood was much higher, almost always between 2 and 3, which meant that they either spontaneously reported feeling depressed or they communicated feeling depressed in a non-verbal way, judging by the rating items from the Hamilton Depression Rating Scale (HDRS). We could also see that when the patients were in a depression state (because of the predominance of green points on higher values of the work axis, where green represented patients in a state of depression), they started feeling loss of interest in activities they usually performed or there was a decrease in the time spent on work and other activities, which made perfect sense according to this rating scale.

D. *Application of the Algorithms*

The data sets on which we tested the algorithms were: YMRS (Young Mania Rating Scale data), HDRS (Hamilton Depression Rating Scale data), Interviews (interview data, IDS), Interventions (intervention data), YMRS-HDRS (combination of the YMRS and HDRS data) and Interviews-Interventions (combination of the Interview and Intervention

data). Fig.4 shows the diagram of the process followed during this study.

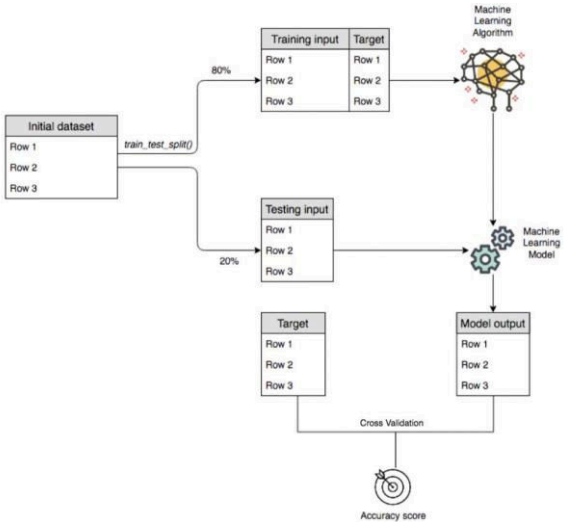


Fig. 4. Diagram of the Machine Learning algorithm application process

The algorithms that we used for this part were: Decision Tree [27], Random Forest [28], Support Vector Machines [29] and Logistic Regression [30]. The reason why these algorithms have been chosen for this project is explained in [26], in the section belonging to each algorithm.

The fact that these algorithms have been applied in this project does not mean that they are the best option for the classification of Bipolar Disorder states, but rather that they are the most suitable ones given the amount of data and the number of features used for the project. In future studies that make use of this project, if the data sets are larger it could be interesting to apply other algorithms too, like the Naïve Bayes [31] algorithm or any kind of Boosting algorithm [32], as to see how they perform on this particular classification problem.

Before applying each algorithm, as is necessary for every classification problem, the original data needed to be split into training and testing sets. Later, the training data would be used to train the prediction models and the testing data would be used to compare the output of the model with the real targets by cross-validation, a technique presented first by M. Stone in 1974 [33], that is used widely in Machine Learning for algorithm performance comparison.

The testing set is used for obtaining the accuracy of the model, as mentioned above, which is done by comparing the output obtained from the testing input and the real output of the testing set. In order to divide the original data sets into training and testing sets, we used the train_test_split() function from the scikit-learn library [25], where the test size represents the percentage of the data that is used for the testing set. After the algorithms were applied, the cross_val_score() function, which is also included in the scikit-learn library, was called in order to evaluate the score by k-fold cross-validation [33].

The best way to compare the accuracies obtained with the different algorithms on all the data sets was to make an algorithm performance matrix, which is shown in Table 1. This matrix showed that, in average, the data set that returned the best prediction accuracy (69%) with the algorithms was the Interview data set, as seen on the right column. The algorithm that performed the best, also in average (62%), was the Logistic Regression algorithm, as seen on the column farthest down.

Even though the algorithm that had the best accuracy average was the Logistic Regression, we stated that the Random Forest algorithm made the most accurate predictions in the sense that they were very reasonable given the behavior of the patients, which we tested with randomized data. These predictions made possible the implementation of a small program with the Random Forest classifier that we obtained, and which can be seen in [26].

TABLE I. ACCURACIES WITH DIFFERENT ALGORITHMS

Algorithms/Datasets	Decision Tree	Random Forest	SVM	Logistic Regression	Average
YMRS	36%	38%	38%	76%	47%
HDRS	78%	55%	36%	62%	58%
Interviews	70%	72%	68%	67%	69%
Interventions	44%	65%	57%	51%	54%
YMRS-HDRS	63%	63%	70%	33%	57%
Interviews-Interventions	67%	17%	42%	83%	52%
Average	60%	52%	52%	62%	

V. Conclusions and Future Work

Having a deep understanding of the data is essential in any Machine Learning project focused on a branch of medical science like psychiatry, where knowing which behaviors are normal and abnormal in the patients can help us create much more precise prediction models. The amount of data used and the nature of the data source are very important factors because with a larger suitable amount of data we will be able to get prediction accuracies with a much higher level of confidence. In the same way that understanding the data is important, having a deep perception of the theory behind each algorithm used, as well as their many implementations, is crucial in order to get the models to perform in the best possible way.

In this project, several groups of data collected in a supervised way have been analyzed and a set of Machine Learning algorithms has been applied. The results allow us to make decisions about the new sources of relevant information to be incorporated in consequent studies. It is concluded that the data from sleep and daily activity, measured both by movement and sounds, are relevant for improving the prediction of a crisis in patients with Bipolar Disorder. Therefore, a future project that includes group 1 bracelets instead of the current medical wristbands is proposed, because the latter are too expensive and invasive, and the development of a new mobile application that, in addition to the daily data, includes sensitization data and sounds. Future work will also analyze the EEG data collected during supervised monitoring for the purpose of performing a comparative analysis. The implementation with Jupyter will also allow us to perform the same studies on larger databases when the number of patients in the experiment is higher.

The most immediate use of the results obtained in this project would be to train the same algorithms used but with

larger amounts of data, in order to see if they perform in a similar way. Gathering objective data from devices like phones or wristbands is something that can be accomplished quite easily according to the work already done in this sense. The goal of this task would be to compare the performance of different algorithms on the objective data gathered from these devices with the performance results obtained on the subjective data used in this project.

As for other more indirect applications of the results obtained during this project, the implementation of a drug recommending system for patients with Bipolar Disorder could be made by predicting the states in which the patients are during a certain period of time. These predictions could be stored in a database which also contains the medicine that these patients have been prescribed with during the same period of time, thus providing the possibility of seeing how each patient reacts to the different types of drugs used.

References

[1] N. Vanello et al., "Speech analysis for mood state characterization in bipolar patients," Annual International Conference of the IEEE Engineering in Medicine and Biology Society, San Diego, CA, pp. 2104-2107, 2012

[2] T. Beiwinkel, et al. "Using Smartphones to Monitor Bipolar Disorder Symptoms: A Pilot Study". Eysenbach G, ed. JMIR Mental Health. 2016.

[3] MN. Burns, et al., "Harnessing Context Sensing to Develop a Mobile Intervention for Depression". J Med Internet Res 2011;13(3):e55.

[4] M. Cukic, et al. "EEG machine learning with Higuchi fractal dimension and Sample Entropy as features for successful detection of depression" CoRR abs/1803.05985 (2018): n. pag..

[5] M. Frost, A. Doryab, M. Faurholt-Jepsen, L. Vedel Kessing, and J. E. Bardram. "Supporting disease insight through data analysis: refinements of the monarca self-assessment system". In Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (UbiComp '13). ACM, New York, NY, USA, pp. 133-142, September-2013

[6] H. Javelot, et al., "Telemonitoring with respect to Mood Disorders and Information and Communication Technologies: Overview and Presentation of the PSYCHE Project", BioMed Research International, vol. 2014, Article ID 104658, 12 pages, 2014.

[7] O. Mayora, et. al., "Mobile Health Systems for Bipolar Disorder: The relevance of Non-Functional Requirements in MONARCA Project". IGI International Journal of Handheld Computing Research (IJHCR). 10.4018/978-1-4666-8756-1.ch070.

[8] Z. Guan, T. Yang, X. Du and M. Guizani, "Secure data access for wireless body sensor networks," 2016 IEEE Wireless Communications and Networking Conference, Doha, pp. 1-6. 2016.

[9] P. Gope and T. Hwang, "BSN-Care: A Secure IoT-Based Modern Healthcare System Using Body Sensor Network," in IEEE Sensors Journal, vol. 16, no. 5, pp. 1368-1376, March, 2016.

[10] L. Sylvia, et. al., "Sleep disturbance may impact treatment outcome in bipolar disorder: A preliminary investigation in the context of a large comparative effectiveness trial", Journal of Affective Disorders, Volume 225, 2018, Pages 563-568, ISSN 0165-0327.

[11] AG. Harvey, et. al., "Treating Insomnia Improves Mood State, Sleep, and Functioning in Bipolar Disorder: A Pilot Randomized Controlled Trial". Journal of consulting and clinical psychology. 83. 10.1037/a0038655.

[12] J. Anchiraico, "Diseño de una Arquitectura Big Data para la Predicción de Crisis en el Trastorno Bipolar", Trabajo Fin de Master en Ingeniería Informática, ePrints-UCM, Madrid, 2016.

[13] ActivePAL (2018). Retrived from http://www.palt.com/.

[14] Bellabeat – Urban Collection, Health Trackers (2018). Retrieved from https://webshop.bellabeat.com/pages/leaf-urban

[15] ELF Emmit Device from NewMed (2018). Retrieved from https://www.news-medical.net/ELF-Emmit-Device-from-NewMed

[16] H. Nakano, et al. "Monitoring Sound To Quantify Snoring and Sleep Apnea Severity Using a Smartphone: Proof of Concept." Journal of Clinical Sleep Medicine : JCSM : Official Publication of the American Academy of Sleep Medicine 10.1 (2014): pp. 73–78, *PMC*. Web. 12 June 2018.

[17] The Altman Self-Rating Mania Scale (2018). Retrieved from https://psychology-tools.com/altman-self-rating-mania-scale/

[18] The Patient Health Questionnaire (PHQ-9) (2018). Retrieved from https://patient.info/doctor/patient-health-questionnaire-phq-9

[19] A. Martínez, "Introduction to Big Data and First Steps in a Big Data Project", Trabajo Fin de Doble Grado Matemáticas-Informática, ePrints-UCM, Madrid, 2016.

[20] R.C. Young et al., "A rating scale for mania: reliability, validity and sensitivity", British Journal of Psychiatry, 1978

[21] M. Hamilton, "A rating scale for depression", Journal of Neurology, Neurosurgery, and Psychiatry, 1960.

[22] F. Perez and B. E. Granger, "IPython: A System for Interactive Scientific Computing," in Computing in Science & Engineering, vol. 9, no. 3, pp. 21-29, May-June 2007.

[23] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research 12 (2011), pp. 2825-283. November, 2011.

[24] A. Junestrand, "Application of Machine Learning Algorithms for Bipolar Disorder Crisis Prediction", Trabajo Fin de Grado Ingeniería Informática, ePrints-UCM, Madrid, 2018.

[25] W. McKinney, "Data Structures for Statistical Computing in Python", Proceedings of the 9th Python in Science Conference, 2010.

[26] T. Hastie, R. Tibshirani and J. Friedman, "The Elements of Statistical Learning", Springer Series in Statistics, 2009.

[27] Garima, H. Gulati and P. K. Singh, "Clustering techniques in data mining: A comparison", New Delhi: 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015.

[28] J. R. Quinlan, "Induction of Decision Trees", Hingham, Massachusetts: Kluwer Academic Publishers, 1986.

[29] L. Breiman, "Random Forests", Berkeley, Califiornia: Statistics Department Technical Report, 2001.

[30] C. W. Hsu, C. C. Chang and C. J. Lin, "A Practical Guide to Support Vector Classification", Taipei: National Taiwan University, 2003.

[31] C. Y. J. Peng, K. L. Lee and G. M. Ingersoll, "An Introduction to Logistic Regression Analysis and Reporting", Bloomington, Indiana: The Journal of Educational Research, 2002.

[32] T. Patil and S. Sherekar, "Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification", International Journal Of Computer Science And Applications , 2013.

[33] P. Bühlmann and T. Hothorn, "Boosting Algorithms: Regularization, Prediction and Model Fitting", Zürich: Seminar für Statistik, ETH Zürich, 2007.

[34] M. Stone, "Cross-validatory choice and assessment of statistical predictions", London: Royal Statistical Society, 1974.

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 6:

PROBLEMAS NO ESTÁNDAR

JCLAL 2.0: mejoras y nuevas funcionalidades en la herramienta Java de código abierto para el aprendizaje activo

Eduardo Pérez
Instituto Maimónides de Investigación Biomédica de Córdoba
Email: eduardo.perez@imibic.org

Luis D. González
Dpto. de Informática
Universidad de Holguín
Email: ldgonzalezo@uho.edu.cu

Luis M. Sánchez
Dpto. de Informática
Universidad de Holguín
Email: lsanchezv@uho.edu.cu

Oscar Reyes
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Email: ogreyes@uco.es

Sebastián Ventura
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Email: sventura@uco.es

***Resumen*—El aprendizaje activo tiene como objetivo la construcción de mejores clasificadores mediante el etiquetado iterativo de conjuntos de ejemplos no etiquetados. Este paradigma de aprendizaje resulta de suma importancia en la actualidad, debido principalmente al alto coste que tiene el proceso de clasificación de grandes volúmenes de datos. JCLAL es una librería Java de código abierto para la investigación en el área de aprendizaje activo, que fue creada con el objetivo de facilitar el desarrollo de algoritmos y la ejecución de experimentos en esta área de estudio. En este trabajo, se presentan las mejoras y nuevas funcionalidades que incorpora JCLAL en su versión 2.0. Esta nueva versión tiene varias características que hacen de JCLAL una herramienta más potente para la investigación en el área del aprendizaje activo. Se introduce un entorno visual que facilita aún más el uso de JCLAL por usuarios no expertos en el dominio. Además, esta nueva versión aprovecha los beneficios de la computación paralela y distribuida, posibilitando que JCLAL sea utilizado eficientemente en entornos actuales donde se manejan grandes conjuntos de datos.**

I. Introducción

El aprendizaje supervisado tiene como objetivo la construcción de modelos a partir de datos que han sido etiquetados previamente. Sin embargo, uno de los principales problemas de este paradigma de aprendizaje es que generalmente se requiere de una considerable cantidad de datos etiquetados para lograr construir modelos con una alta precisión. El etiquetado de datos no es una tarea sencilla de hacer, ya que requiere comúnmente un esfuerzo considerable por parte de los expertos que etiquetan los datos. En consecuencia, hoy en día se pueden encontrar muchas colecciones de datos que tienen un número reducido de ejemplos etiquetados en conjunto con una cantidad enorme de ejemplos que quedaron sin etiquetar [1]. En este tipo de escenario las técnicas supervisadas son insuficientes al no ser capaces de explotar la información útil que se encuentra implícita en los datos no etiquetados. Ante este tipo de problema, surge el aprendizaje semi-supervisado y el aprendizaje activo (AL), que tienen como principal objetivo la construcción de modelos precisos usando tanto los datos etiquetados como los no etiquetados.

Los métodos de AL emplean como principal hipótesis que si el algoritmo tiene la oportunidad de elegir los datos para el proceso de aprendizaje, entonces se podrán construir mejores modelos con un menor costo. Las técnicas de AL tratan de superar los obstáculos del etiquetado de datos mediante consultas donde se intenta descubrir aquellos ejemplos no etiquetados que pueden aportar información útil en la construcción de un modelo. Los métodos de AL son iterativos, donde en cada paso los ejemplos no etiquetados más significativos son presentados a un oráculo (por ejemplo, un experto), el cual los etiqueta y posteriormente estos ejemplos son incorporados al conjunto de entrenamiento a partir del cual se construye un nuevo modelo. De esta manera se intenta constuir un modelo más preciso usando la menor cantidad posible de información [2, 3].

En la actualidad, el rápido y adecuado desarrollo de un área de investigación está condicionado a contar con las herramientas computacionales que provean, al menos, las siguientes funcionalidades: (I) la disponibilidad de un API que permita reducir el tiempo en la implementación de nuevos algoritmos; (II) la facilidad de reproducción de experimentos; y (III) que tenga implementado un número considerable de métodos del estado del arte, de manera que le ahorre tiempo a los investigadores, así como disminuya la posibilidad de cometer errores en la implementación de dichos algoritmos. En este sentido, el AL ha estado bastante limitado, ya que las heramientas computacionales para la investigación en esta área son muy escasas, y la mayoría de las existentes no proveen de las funcionalidades mencionadas anteriormente. Entre las librerías que podemos encontrar en Internet destacan LibAct [4] desarollada en Python, y JCLAL [5] que está completamente desarrollada en el lenguaje Java.

Si comparamos estas dos librerías de clases, es de destacar que JCLAL contiene todos los elementos tratados en LibAct,

pero además incorpora un amplio conjunto de funcionalidades que permiten controlar todo el ciclo de AL a través de un correcto y flexible diseño de clases e interfaces. De esta manera, la implementación de nuevas estrategias de AL, así como la definición de nuevos escenarios de consulta, tipos de oráculos, condiciones de paradas, entre otras muchas funcionalidades, resulta un proceso bastante sencillo de hacer. Por otra parte, JCLAL cuenta con una detallada documentación y un conjunto amplio de ejemplos.

Como parte de la evolución a la cual está sujeto cualquier software, la librería JCLAL se ha seguido mejorando y desarrollando desde su publicación inicial. El objetivo del presente trabajo es presentar las nuevas características y funcionalidades que podemos encontrar en la nueva versión de este framework. Si se analiza la versión 1.0 de JCLAL, esta no disponía de una interfaz de usuario, obligando a los investigadores a usar el sistema de configuración de experimentos (basado en ficheros XML) o bien a codificar directamente los procedimientos usando su API. En este sentido, la nueva versión de JCLAL dispone de una interfaz de usuario que lo convierte en un producto más asequible y usable, lo que posibilita conocer sus potencialidades sin tener que entrar en el proceso de asimilación de su API. Por otra parte, la antigua versión de JCLAL está limitada a trabajar solamente con conjuntos de datos que pueden ser cargados en la memoria de estaciones de trabajo individuales, lo cual complicaba el uso de este framework en escenarios con enormes volúmenes de datos (problemas *Big Data*). En este trabajo, se explica además como la nueva versión de JCLAL supera esta última limitación gracias al desarrollo de un módulo que se integra con el conocido framework para procesamiento de datos masivos Apache Spark [6].

El resto de este trabajo se organiza de la manera siguiente. En la Sección II se detallan los cambios y nuevas funcionalidades en el API de JCLAL, así como se describe la nueva manera en la que se gestiona la modularidad del framework. En la sección III se introduce el entorno visual de JCLAL, detallando las funcionalidades del mismo. La integración de la librería con el framework Spark se explica en la Sección IV. Finalmente, en la Sección V se presentan las conclusiones del presente trabajo.

II. Estructura de clases y nuevas funcionalidades en el API de JCLAL 2.0

Desde su primer lanzamiento JCLAL ha continuado mejorándose y desarrollándose como parte del proceso de actualización y mantenimiento de cualquier software. En su versión 2.0, JCLAL cuenta con una nueva estructura de clases y funcionalidades que lo convierten en un framework cada vez más cercano a los estándares que se esperan de este tipo de herramienta computacional.

Una de las nuevas funcionalidades es la posibilidad de aprovechar todas las capacidades de cómputo de las máquinas modernas. El AL es un proceso iterativo, donde en cada iteración comúnmente los pasos más costosos son la construcción del modelo, la evaluación del modelo y el cálculo de la importancia de cada instancia no etiquetada según la estrategia de consulta. JCLAL brinda la posibilidad de paralelizar el experimento en una estación de trabajo individual y también a través del framework Apache Spark. El investigador además tiene la posibilidad de elaborar sus propias estrategias de paralelización a través de la interfaz *IParallelContext*. Si el usuario desea hacer la paralelización de forma personalizada deberá registrar su clase en un archivo de configuración, como se muestra en el ejemplo siguiente:

```
<query-strategy type="...EntropySamplingQueryStrategy">
 <parallel-unlabeled type="...TestUnlabeledDataSinglePC"/>
 <wrapper-classifier type="...WekaClassifier">
  <parallel-test type="...WekaTestModelSinglePC"/>
  <parallel-build type="...WekaBuildClassifierSinglePC"/>
  <classifier type="weka.classifiers.functions.SMO"/>
 </wrapper-classifier>
</query-strategy>
```

Anteriormente, JCLAL estaba restringido al uso de conjuntos de datos que siguieran el formato establecido por el popular framework Weka [7]. En esta nueva versión, se modificó la jerarquía de clases e interfaces que están relacionadas con la manipulación de los conjuntos de datos. De esta manera, se logra una mayor flexibilidad e independencia en el trabajo con los conjuntos de datos, haciendo posible integrar a JCLAL con datos en formato de cualquier otro framework. Si un investigador quiere aplicar JCLAL en conjuntos de datos no soportados nativamente, solamente tendrá que implementar las interfaces *IInstance* e *IDataset*; además estas interfaces dan soporte a los tipos de instancias densas y dispersas.

Por otra parte, en JCLAL 2.0 se le ha dado mayor soporte a los clasificadores del framework MOA [8], lo cual permite utilizar algoritmos de aprendizaje incremental en el proceso de AL. De esta manera, el modelo no necesita ser entrenado desde cero en cada iteración de AL (a diferencia de la mayoría de los clasificadores de Weka), lo cual mejora significativamente el rendimiento en escenarios donde se disponen de grandes volúmenes de datos no etiquetados o con datos en *streamming*.

Por último, en esta nueva versión se ha añadido un paquete de clases que implementan varios test estadísticos no paramétricos. Dicho paquete está inspirado en el módulo de test estadísticos de la librería Keel [9]. Los test estadísticos son de suma importancia para el análisis de resultados experimentales, y JCLAL incluye varios test que permiten realizar comparaciones múltiples entre estrategias de AL.

II-A. Modularidad de JCLAL

Una de las nuevas mejoras es la modularidad del framework, lo cual facilita su portabilidad y distribución. La idea surgió al observar que JCLAL crecía en tamaño (un tamaño considerable al incluir el paquete de integración con Spark), lo cual hacía que el árbol de dependencia fuera bien extenso y además afectaba la portabilidad y usabilidad de JCLAL en aquellos usuarios que solamente querían usar las características básicas del framework.

En la nueva versión se optó por un diseño modular inspirado en el diseño de Apache Spark y Spring [10], separando las características del framework en los distintos módulos que componen el sistema. De esta manera, el proyecto fue

dividido en 3 módulos: **jclal-core** que tiene las funcionalidades básicas de JCLAL, el módulo **jclal-gui** que incluye la nueva interfaz visual y **jclal-spark** que comprende el procesamiento distribuido con Spark.

La modularidad de JCLAL se gestionó mediante Apache Maven [11], el cual es una herramienta popular para la gestión de proyectos software, que facilita la administración de dependencias y un amplio conjunto de estándares para el proceso de desarrollo del software. Si el usuario decide emplear Maven como administrador de dependencias, el *groupid* para JCLAL sería **net.sf.jclal**. El siguiente ejemplo muestra cómo instalar el núcleo de JCLAL (**jclal-core**) en una aplicación; de esta manera Maven gestiona las dependencias necesarias para usar solamente el módulo **jclal-core**.

```
<dependencies>
  <dependency>
    <groupId>net.sf.jclal</groupId>
    <artifactId>jclal-core</artifactId>
   <version>2.0</version>
  </dependency>
</dependencies>
```

III. Entorno visual para JCLAL

Es común que los framework de aprendizaje automático dispongan de una interfaz gráfica de usuario (GUI, por sus siglas en inglés) para facilitar la configuración, ejecución y análisis de resultados de los experimentos, reduciendo así el tiempo y trabajo necesario de los investigadores. Tal es el caso de Weka, VisualJCLEL [12], Eva2 [13], OTA [14] y HeuristicLab [15] que permiten a los investigadores sin un amplio conocimiento de su funcionamiento interno, o incluso sin habilidades de programación, utilizar estos framework de forma intuitiva.

VisualJCLAL es un módulo GUI que permite trabajar con el framework de forma intuitiva y flexible, acercando JCLAL a los usuarios no expertos en el dominio. Este entorno visual está inspirado en la arquitectura de Weka y VisualJCLEC. VisualJCLAL permite aprovechar la mayoría de las funcionalidades que brinda su framework base, y hasta el momento brinda la opción de utilizar estrategias de AL diseñadas para crear modelos predictivos sobre datos clasificados con una sola etiqueta (*single-label learning*) y datos clasificados con múltiples etiquetas (*multi-label learning*) [2].

VisualJCLAL brinda también la posibilidad de incluir escenarios de AL, algoritmos y estrategias de consulta mediante *plugins* creados por el usuario, gracias al uso de una arquitectura basada en componentes y al diseño flexible de clases e interfaces de JCLAL. Para ello el usuario utilizará la opción de agregar un nuevo *plugin* al sistema donde sus clases serán añadidas a los archivos de configuración en formato **.props** y serán mostrados en la GUI aquellos componentes que implementen la interfaz *IPlugin*. Estos parámetros se incluirán automáticamente dentro de las opciones de la interfaz gráfica usando la estructura modular y el generador de interfaces del entorno de usuario.

Por otra parte, este módulo aprovecha las posibilidades de JCLAL en su versión 2.0 para la ejecución de los algoritmos en forma paralela y distribuida (esta última utilizando el framework Apache Spark), además de otras funcionalidades que resultan muy útiles para la realización de experimentos de AL.

La Figura 1 muestra la pantalla inicial que aparece tras ejecutar VisualJCLAL. A partir de ella se tienen tres opciones principales:

- Crear nuevo experimento (*New Experiment*): permite crear una nueva configuración desde cero para la ejecución de un experimento.
- Cargar la configuración de un experimento (*Make from XML*): permite cargar una configuración de un experimento creado anteriormente y que se encuentra almacenado en un archivo XML.
- Ver resultados (*View results*): permite analizar de forma gráfica los resultados de los experimentos ejecutados en JCLAL.

Figura 1. Interfaz inicial.

A la interfaz de configuración de un experimento se puede acceder a través de la opción *New Experiment* o cargando una configuración existente mediante la opción *Make from XML*. Como se puede observar en la Figura 2, en las diferentes pestañas de la interfaz se pueden configurar todos los parámetros necesarios para la ejecución del experimento.

En cuanto a la opción *View results*, esta nos permite cargar los ficheros generados por JCLAL y analizar gráficamente los resultados de un experimento, como se muestra en la Figura 3. Mediante la selección de una medida de evaluación se puede observar el comportamiento de varias curvas de aprendizaje al mismo tiempo, permitiendo realizar una comparación visual entre diferentes estrategias de AL. Además, esta interfaz permite configurar y analizar las gráficas mediante el cambio de colores, ocultar o mostrar curvas, definir rangos de evaluación, cambiar las formas de las líneas y calcular el área bajo una curva de aprendizaje.

Otra opción importante que brinda VisualJCLAL es la posibilidad de crear un estudio experimental con múltiples configuraciones a partir de una configuración básica (la configuración base debe contar con los parámetros mínimos

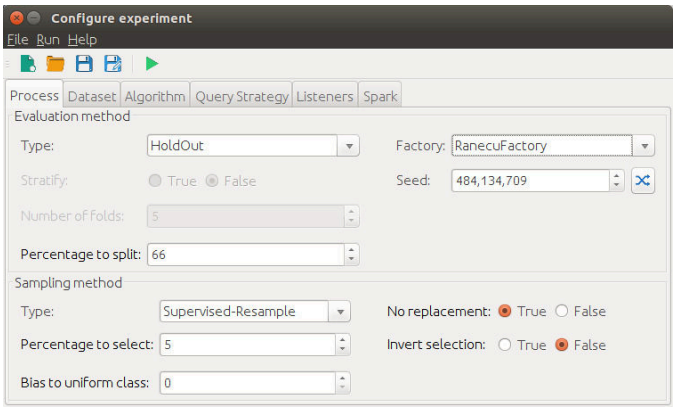


Figura 2. Interfaz para la configuración de un experimento.

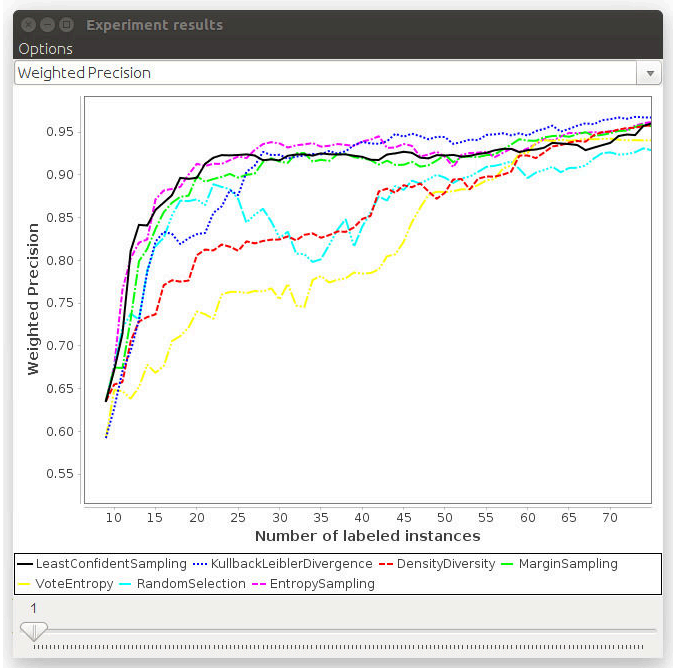


Figura 3. Interfaz de usuario para la visualización de resultados.

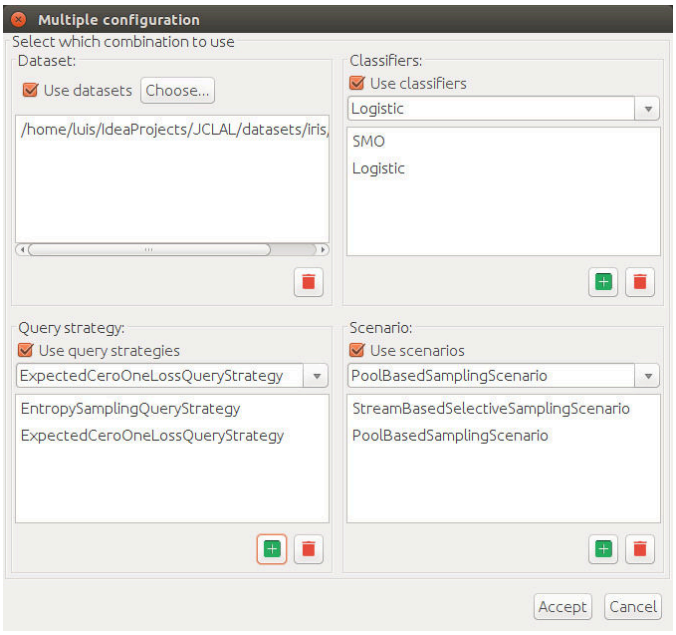


Figura 4. Creación de un estudio experimental.

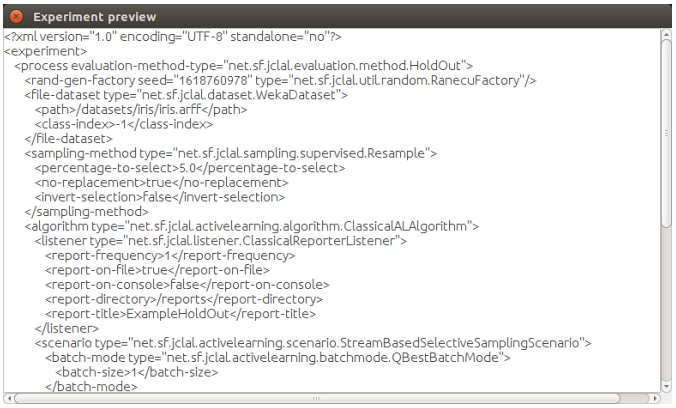


Figura 5. Vista previa del archivo de configuración.

necesarios para la ejecución de un experimento). En la Figura 4 se puede observar cómo se pueden generar N configuraciones en dependencia de las opciones escogidas.

Por último, se cuenta con la opción de chequear cómo quedaría la configuración del experimento antes de ser almacenada en un fichero, como se muestra en la Figura 5.

IV. Integración de JCLAL con el framework Spark

Big Data es un término muy popular en la actualidad, que hace referencia a problemas que involucran volúmenes de datos que superan la capacidad de los software clásicos y las máquinas convencionales para ser capturados, gestionados y procesados en un tiempo razonable. Apache Spark es uno de los framework más populares para el procesamiento de enormes conjuntos de datos, el cual implementa el paradigma de programación MapReduce [16] propuesto por Google. Spark almacena los resultados intermedios en memoria, logrando ejecutar aplicaciones mucho más rápido que las soluciones desarrolladas con Hadoop [17], lo cual hace de Spark una solución más adecuada para el desarrollo de algoritmos de aprendizaje automático altamente escalables. En este sentido, Spark cuenta con la librería MLlib [18] para el desarrollo de algoritmos distribuidos, la cual tiene implementaciones distribuidas de algunos de los algoritmos supervisados y no supervisados más populares, como Naive Bayes [19], árboles de decisión [20] y *k-means* [21].

La versión 1.0 de JCLAL se encontraba limitada a la hora de ejecutar experimentos con grandes conjuntos de datos, principalmente porque los algoritmos de aprendizaje que provee el framework Weka, o Mulan [22] para el caso de datos multi-etiqueta, no están diseñados para trabajar en este tipo escenarios. En la versión actual de JCLAL se ha desarrollado

un módulo que integra a JCLAL con Spark, aprovechando las ventajas que tiene este último para el procesamiento paralelo y distribuido de datos masivos. El módulo de AL distribuido (**jclal-spark**) está diseñado de forma tal que puede trabajar con los modelos de clasificación que provee la librería MLlib. La arquitectura es similar a la de **jclal-core**, siendo la estructura de paquetes **net.sf.jclal.spark.***.

Para utilizar **jclal-spark** a través de Maven hay que añadir el *artifactid* de este módulo, de esta manera se instalan automáticamente todas las dependencias necesarias para el trabajo con Spark. Además, en el fichero de configuración de un experimento, se debe incluir la etiqueta que indica la configuración para conectarse al servidor Spark. La etiqueta **master** especifica la dirección donde se encuentra el nodo maestro y en **app-name** se indica el nombre que se desea asignar al trabajo.

```
<spark>
  <master>spark://localhost:7077</master>
  <app-name>MyAlJob</app-name>
<spark>
```

Debido a que Spark provee sus propios modelos de clasificación, **jclal-spark** implementa las clases *SparkClassifier* y *SparkCommitteeClassifier* que sirven de interfaz para acceder a dichos clasificadores directamente (ver Figura 6). La clase encargada de realizar las evaluaciones de los clasificadores de Spark es *SparkSingleLabelEvaluation*, que además provee las medidas de tiempo de ejecución e información acerca de los conjuntos de datos.

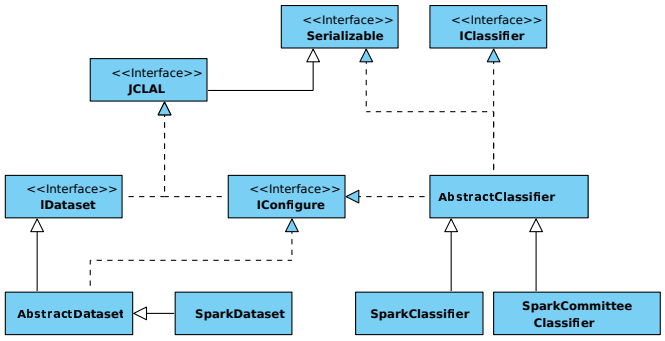


Figura 6. Diagrama de las clases principales del paquete **jclal-spark**.

Es válido destacar que el procesamiento de los conjuntos de datos en un ambiente distribuido es completamente diferente a como se hace en **jclal-core**. La clase *SparkDataset* se ha implementado para el manejo de datos distribuidos a través del conjunto de acciones y operadores que brinda la librería Spark. Los datos pueden ser cargados directamente desde disco o en un sistema de archivos distribuidos los cuales son primeramente indexados para luego ser procesados en el framework. En la nueva versión de JCLAL se añadió la etiqueta **format** para especificar el formato del conjunto de datos, logrando una mayor compatibilidad con Spark y otras librerías que se puedan incluir en el futuro. El ejemplo siguiente muestra como utilizar un conjunto de datos en formato **libsvm** empleando *SparkDataset*.

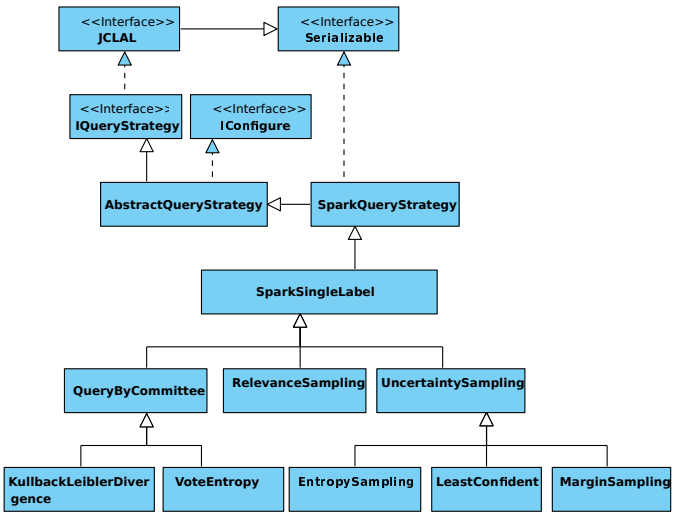


Figura 7. Diagrama de clases de las estrategias de consulta implementadas en el módulo **jclal-spark**.

```
<file-dataset type="net.sf.jclal.spark.dataset.SparkDataset">
  <path>path/to/dataset</path>
  <format>libsvm</format>
<file-dataset>
```

Respecto a las estrategias de consultas de AL implementadas en el paquete **jclal-spark**, hasta el momento están implementadas aquellas estrategias de consultas que se adaptan directamente al paradigma MapReduce, pero se espera implementar próximamente el resto de estrategias contenidas en **jclal-core**. La figura 7 muestra el diagrama de clases de las estrategias de consulta que actualmente soporta el módulo **jclal-spark**; estas son *Relevance Sampling*, y todas aquellas derivadas de *Uncertanty Sampling* (*Entropy Sampling*, *Least Confident* y *Margin Sampling*) y *Query By Committee* (*Kullback Leibler Divergence* y *Vote Entropy*).

Por último, en **jclal-spark** están implementados los métodos de evaluación clásicos: *Hold-Out*, *k-Fold Cross Validation*, *5X2 Cross Validation* y *Leave One Out Cross Validation*. En el caso de los métodos de validación cruzada se utilizaron funciones de Spark que están optimizadas para la creación de los *folds* en conjuntos de datos distribuidos.

V. Conclusiones

En este trabajo se describieron las mejoras y nuevas funcionalidades que se incluyen en la nueva versión del framework JCLAL. En su nueva versión, el núcleo de JCLAL cuenta con una nueva estructura de clases y funcionalidades que lo convierten en un framework más potente para la investigación en el área de AL. Se logró una mayor flexibilidad e independencia en la manipulación de los conjuntos de datos, haciendo posible integrar a JCLAL con datos provenientes de cualquier otro framework. Además, se le dio un mayor soporte a algoritmos de aprendizaje incremental, lo cual posibilita una mayor eficiencia en el proceso de construcción de modelos predictivos. Por otra parte, se añadió un paquete que permite realizar contrastes de hipótesis mediantes test estadísticos no

paramétricos. Otra nueva mejora radica en la descomposición del framework en módulos independientes, lo cual facilita su portabilidad y distribución. De esta manera se puede trabajar con las funciones deseadas del framework sin tener instaladas todas y cada una de las librerías del árbol de dependencias.

JCLAL 2.0 también incorpora el entorno VisualJCLAL que permite trabajar con el framework de forma intuitiva y flexible, acercando JCLAL a los usuarios no expertos en el dominio y permitiendo un ahorro significativo de tiempo. Por último, se incorporó un nuevo e importante módulo, que permite de forma sencilla aplicar programación distribuida en JCLAL. El módulo de AL distribuido utiliza la popular librería Spark, lo cual permite el uso de JCLAL 2.0 en problemas *Big Data* y posibilita una mejor acogida por parte de la comunidad científica.

Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2017-83445-P del Ministerio de Economía y Competitividad y Fondos FEDER. También ha sido financiado por el contrato i-PFIS no. IFI17/00015 otorgado por el Instituto de Salud Carlos III.

Referencias

[1] O. Reyes, A. H. Altalhi, and S. Ventura, "Statistical comparisons of active learning strategies over multiple datasets," *Knowledge-Based Systems*, vol. 145, pp. 274–288, 2018.

[2] O. Reyes, C. Morell, and S. Ventura, "Effective active learning strategy for multi-label learning," *Neurocomputing*, vol. 273, pp. 494–508, 2018.

[3] O. Reyes and S. Ventura, "Evolutionary strategy to perform batch-mode active learning on multi-label data," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 4, pp. 46:1–46:26, 2018.

[4] Y. Y. Yang, S. C. Lee, Y. A. Chung, T. E. Wu, S. A. Chen, and H. T. Lin, "libact: Pool-based active learning in python," *arXiv preprint arXiv:1710.00379*, 2017.

[5] O. Reyes, E. Pérez, M. C. Rodríguez-Hernández, H. M. Fardoun, and S. Ventura, "JCLAL: a java framework for active learning," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 3271–3275, 2016.

[6] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 10–10.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[8] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: massive online analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010.

[9] J. Alcalá-Fdez, L. Sanchez, S. Garcia, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas *et al.*, "Keel: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[10] R. Johnson, *Expert One-on-one J2EE Design and Development*. Wrox Press, 2002.

[11] F. P. Miller, A. F. Vandome, and J. McBrewster, "Apache maven," 2010.

[12] J. I. Jaen, J. R. Romero, and S. Ventura, "VisualJCLEC: A visual framework for evolutionary computation," in *Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on*. IEEE, 2012, pp. 119–125.

[13] M. Kronfeld, H. Planatscher, and A. Zell, "The EvA2 optimization framework," in *International Conference on Learning and Intelligent Optimization*. Springer, 2010, pp. 247–250.

[14] J. Brownlee *et al.*, "Oat: The optimization algorithm toolkit," *Centre for Information Technology Research (CITR), Swinburne University of Technology, Victoria, Australia, Technical Report A*, vol. 20071220, 2007.

[15] S. Wagner and M. Affenzeller, "Heuristiclab: A generic and extensible optimization environment," in *Adaptive and Natural Computing Algorithms*. Springer, 2005, pp. 538–541.

[16] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[17] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. Mccauley, M. Franklin, S. Shenker, and I. Stoica, "Fast and interactive analytics over hadoop data with spark," *Usenix Login*, vol. 37, no. 4, pp. 45–51, 2012.

[18] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.

[19] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[20] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[21] S. Lloyd, "Least squares quantization in PCM," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[22] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas, "Mulan: A java library for multi-label learning," *Journal of Machine Learning Research*, vol. 12, pp. 2411–2414, 2011.

Resolviendo el problema de regresión multi-salida mediante *Gene Expression Programming*

Oscar Reyes
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Email: ogreyes@uco.es

Jose M. Moyano
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Email: jmoyano@uco.es

Jose M. Luna
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Email: jmluna@uco.es

Sebastián Ventura
Dpto. Informática y Análisis Numérico
Universidad de Córdoba
Email: sventura@uco.es

***Resumen*—En los últimos años, el estudio de problemas donde los ejemplos están asociados a múltiples variables objetivo al mismo tiempo ha ganado un creciente interés en la comunidad científica. En este trabajo se propone un método basado en *Gene Expression Programming* para darle solución al problema de regresión multi-salida. El método propuesto se enfoca en la regresión simbólica, lo cual permite crear un modelo predictivo multi-salida sin tener conocimiento previo de las relaciones existentes en los datos. La codificación de un individuo de la población se realiza mediante un cromosoma de varios genes, donde cada gen codifica una expresión matemática que produce la salida para una variable objetivo, y así cada individuo de la población representa directamente una posible solución al problema. Los operadores usados en el proceso evolutivo permiten la constante creación de nuevo material genético, y algunos de ellos favorecen la detección de las dependencias existentes entre variables objetivo. El estudio experimental realizado en 8 conjuntos de datos muestra los beneficios y efectividad del método propuesto para resolver el problema de regresión multi-salida.**

I. Introducción

En la última década, el estudio de problemas donde cada ejemplo del conjunto de datos está asociado a múltiples variables objetivo (variables de salida) ha tenido un creciente interés en la comunidad de aprendizaje automático, debido principalmente al elevado número de aplicaciones reales que pueden ser modeladas dentro de este paradigma. El objeto de estudio de la regresión multi-salida o *multi-target regression* (MTR), se enfoca en la predicción simultánea de múltiples variables objetivo continuas usando un único conjunto de variables descriptoras (variables de entrada) [1, 2]. MTR está presente en varios dominios de aplicación, tales como en el modelado ecológico [3], procesado de señales [4], y la eficiencia energética [5].

A día de hoy, se han propuesto un número considerable de métodos para resolver el problema MTR, los cuales se pueden categorizar en dos grupos: métodos de transformación de problemas y métodos de adaptación de algoritmos [1]. Los métodos de transformación descomponen un problema de regresión multi-salida en varios problemas de regresión simple (es decir, con una sola variable objetivo), y por último mediante un proceso de agregación se obtienen las predicciones finales. En este sentido, la mayoría de los métodos de transformación para MTR han sido inspirados en aproximaciones existentes para el aprendizaje multi-etiqueta [2, 6]; en este último paradigma cada ejemplo está asociado a múltiples variables objetivo binarias. Por otro lado, la categoría de adaptación de algoritmos incluye aquellos métodos que no descomponen el problema MTR en problemas de regresión simple, sino que tratan directamente los datos multi-salida. Dentro de esta categoría se han propuesto un amplio número de métodos, tales como métodos estadísticos [7], árboles de regresión [3], algoritmos basados en reglas [8], máquinas de vectores soporte [9] y métodos locales [10].

Entre los paradigmas existentes más populares para resolver problemas de regresión se encuentra la regresión simbólica, donde el objetivo es buscar una expresión matemática que se ajuste a un conjunto de datos dado. En este tipo de técnica no se proporciona ningún modelo en particular como punto de partida, sino que las expresiones se forman por medio de la codificación de bloques matemáticos, lo cual provee de una selección de características implícita en el proceso de construcción de expresiones, y además puede beneficiar la detección de relaciones complejas en los datos. Las técnicas de regresión simbólica han sido aplicadas a un gran número de problemas reales [11], destacando el uso de algoritmos evolutivos para su resolución [12]. Entre las técnicas de computación evolutiva más adecuadas para el desarrollo de métodos de regresión simbólica se encuentra *Gene Expression Programming* (GEP) [12–14], una aproximación que se sitúa entre los algoritmos genéticos y la programación genética, aprovechando así las ventajas de ambas técnicas. [13]. GEP emplea una población de individuos, selecciona padres de acuerdo a su *fitness* y evoluciona la población utilizando operadores genéticos, tal y como hacen los algoritmos genéticos y la programación genética. Sin embargo, la principal diferencia radica en la forma en que los individuos son codificados. En este caso, los individuos se codifican como cadenas lineales de longitud fija (como en algoritmos genéticos), que luego se expresan como árboles de diferentes tamaños y formas (como en programación genética) [13].

Hasta el momento, los trabajos existentes que aplican GEP para regresión simbólica se han restringido a solucionar problemas de regresión simple. Es de destacar que al tratar de solucionar el problema MTR mediante regresión simbólica usando la técnica GEP aparecen dificultades adicionales que no existían en el contexto de la regresión simple, como es la dependencia estadística que puede existir entre las variables objetivo. En este sentido, varios estudios recientes han demostrado que es vital detectar y explotar correctamente dichas dependencias de cara a mejorar el rendimiento predictivo de los algoritmos de regresión [2, 9, 10, 15].

En este trabajo, se propone un método basado en GEP que resuelve el problema MTR mediante regresión simbólica. El método diseñado no divide el problema multi-salida en varias tareas de salida simple, sino que trata directamente con los datos multi-salida, por lo que se obtiene un método con un coste computacional aceptable en conjuntos de datos con un elevado número de variables objetivo. Los individuos se representan mediante un cromosoma con codificación multigénica, donde cada gen representa una función matemática que predice el valor de una variable objetivo; de esta manera cada individuo representa una solución completa al problema MTR. El poder creativo de GEP permite la constante creación de nuevo material genético, permitiendo una mejor exploración del espacio de búsqueda. Además, el método puede detectar las dependencias existentes entre variables objetivo por medio de los operadores de transposición o recombinación de genes entre cromosomas.

La contribución principal de este trabajo es la introducción de un método basado en GEP para el problema MTR. Este trabajo representa un estudio inicial, donde se pretende analizar los beneficios del paradigma GEP para construir modelos de regresión multi-salida. La efectividad del método desarrollado se comprobó mediante un estudio experimental en 8 conjuntos de datos, demostrando que se obtienen resultados prometedores mediante la comparación contra otros dos métodos del estado del arte.

El resto del este trabajo se organiza de la siguiente manera: en la Sección II se describen los fundamentos del método propuesto; la configuración del estudio experimental, así como la discusión de los resultados son presentados en la Sección III; finalmente, en la Sección IV se presentan las conclusiones del presente trabajo.

II. Método Basado en GEP

En esta sección se describe el método propuesto basado en GEP para resolver el problema MTR mediante regresión simbólica. Antes de describir el método en sí, se definen algunas notaciones que son utilizadas a lo largo del trabajo.

Sea $S = \{(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2), \ldots, (\mathbf{x}^n, \mathbf{y}^n)\}$ un conjunto de datos de n ejemplos de entrenamiento. Una instancia $i \in S$ se representa como una tupla $(\mathbf{x}^i, \mathbf{y}^i)$, donde $\mathbf{x}^i \in \mathcal{X}$ e $\mathbf{y}^i \in \mathcal{Y}$ son los vectores de entrada y salida de i, respectivamente. $\mathcal{X}$ representa el espacio de entrada que contiene d variables descriptivas $x_1, x_2 \ldots, x_d$, mientras que $\mathcal{Y}$ representa el espacio de salida, que contiene q variables objetivo $y_1, y_2 \ldots, y_q$. Por otro lado, x_ℓ^i denota el valor de la ℓ-ésima variable descriptiva para el ejemplo i, mientras que y_ℓ^i representa el valor de su ℓ-ésima variable objetivo. En el problema MTR el objetivo principal es construir una función $f : \mathcal{X} \rightarrow \mathcal{Y}$ que prediga un vector de variables objetivo $\hat{\mathbf{y}}$ a partir de un vector de variables descriptivas nunca antes visto $\mathbf{x}$. Por otro lado, como se hace uso de regresión simbólica, es necesaria la definición del conjunto de símbolos no terminales (funciones) F=$\{-, +, /, *, \ldots\}$ y el conjunto de símbolos terminales T=$\{x_1, x_2, \ldots, x_d\}$, donde cada símbolo terminal representa una variable descriptiva $x_\ell \in \mathcal{X}$.

En el paradigma GEP, un gen está compuesto por una cabeza y una cola. La cabeza puede contener símbolos que pertenezcan tanto a los conjuntos F como T, mientras que la cola puede contener solo símbolos terminales. La longitud de la cabeza (h) se escoge de acuerdo al problema, mientras que la longitud de la cola (t) se calcula como t=$h(a-1)+1$, donde a es el número de argumentos de la función con máxima aridad en F. Tenga en cuenta que a mayor h, más larga y compleja será la expresión matemática que se podrá codificar, sin embargo hay que considerar que también será mayor el espacio de búsqueda. Una desventaja de la mayoría de métodos basados en GEP reside en la continua transformación de los cromosomas en árboles de expresión (ETs, por sus siglas en inglés) y su correspondiente evaluación para obtener el *fitness* de los individuos. Para evitar esto, en este trabajo los genes se codifican mediante notación prefija, siguiendo el enfoque propuesto por Peng et al. en [12], permitiendo la evaluación de los individuos sin construir los ETs, y mejorando por tanto significativamente la eficiencia computacional de GEP. La Figura 1 representa un gen de un cromosoma, que en este caso contiene la notación prefija de la ecuación matemática $f(\mathbf{x}) = \sqrt{\left(\frac{x_1}{x_4 * x_5} - \sqrt{x_1}\right)} + x_1$.

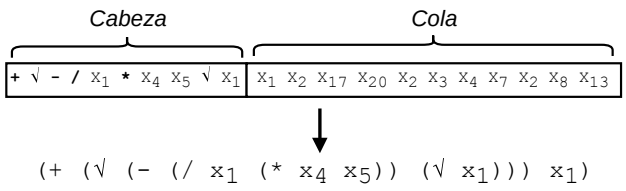


Figura 1. Un gen con $h = 10$ y su correspondiente fenotipo representado en notación prefija.

Se define una población de p individuos, donde cada individuo representa una posible solución completa al problema MTR. Para ello, un individuo es representado mediante un cromosoma multi-génico, compuesto por q genes de igual longitud $c = [g_1, g_1, \ldots, g_q]$, conteniendo un gen por cada variable objetivo del problema. Por tanto, el gen ℓ-ésimo de un cromosoma codifica una función matemática que se puede utilizar posteriormente como modelo predictivo para estimar el valor de la variable objetivo y_ℓ^i de un ejemplo de prueba i.

Para calcular el nivel de adaptación de un individuo se calcula el error cuadrático medio relativo (*average relative*

root mean square error, aRRMSE) en promedio para todas las variables objetivo sobre el conjunto de entrenamiento S,

$$\frac{1}{q}\sum_{\ell=1}^{q}\sqrt{\frac{\sum_{i\in S}(y_\ell^i-\hat{y}_\ell^i)^2}{\sum_{i\in S}(y_\ell^i-\bar{y}_\ell)^2}}\,, \tag{1}$$

donde y_ℓ^i e $\hat{y}_\ell^i$ son los valores de la ℓ-ésima variable objetivo en los vectores de salida conocido ($\mathbf{y}^i$) y predicho ($\hat{\mathbf{y}}^i$) para el ejemplo i, respectivamente. Por otro lado, $\bar{y}_\ell$ es el valor medio de la variable objetivo ℓ-ésima en el conjunto de entrenamiento S. Resumiendo, dado un individuo de la población, la función de *fitness* mide el error cuadrático medio relativo (RRMSE) para la primera variable objetivo, evaluando la función matemática codificada en el primer gen del cromosoma en cada ejemplo de entrenamiento $i \in S$, y así este proceso se realiza para todas las variables objetivo obteniendo finalmente el valor promedio entre todas ellas.

La población inicial se crea de manera aleatoria, pero siempre controlando la diversidad entre los individuos generados. Un cromosoma se forma creando cada uno de sus genes de la siguiente manera: (I) los símbolos en la cabeza del gen se seleccionan aleatoriamente de $F \cup T$, asegurando que el primer elemento (raíz) de la cabeza sea un símbolo no terminal; y (II) los símbolos de la cola se seleccionan aleatoriamente del conjunto T. Un individuo se añade a la población inicial si tiene una similaridad respecto al resto de la población menor que un umbral específico. La similaridad entre dos individuos se calcula como

$$s(c^i, c^j) = 1 - \frac{\sum_{\ell=1}^{q} h(c_\ell^i, c_\ell^j)}{q^2(t+h)}, \tag{2}$$

donde c^i y c^j son los cromosomas de los individuos i y j, respectivamente. La función $h(c_\ell^i, c_\ell^j)$ mide el número de símbolos distintos, sumados a lo largo de las $t+h$ posiciones, en el ℓ-ésimo gen de los cromosomas c^i y c^j. Este método permite la creación de una población inicial con suficiente diversidad, lo cual evita la convergencia temprana del método a un mínimo local.

Respecto al operador de selección, se ha utilizado una selección por torneo binario para crear la población intermedia de padres. Se empleó una baja presión selectiva para que no solo se favorezca a los mejores individuos en el proceso de selección, sino que se estimule también la diversidad entre los individuos seleccionados.

En cuanto al operador de mutación, los padres mutan con una tasa de mutación p_m y se ha diseñado un operador que siempre genera individuos válidos. Se han empleado varios puntos de mutación por cromosoma (n_{mp}), y dicho operador realiza los siguientes pasos para cada uno de los puntos de mutación: (I) se selecciona aleatoriamente uno de los genes del individuo; (II) se selecciona aleatoriamente una posición dentro del gen; (III) si la posición pertenece a la cabeza, el símbolo en dicha posición se cambia aleatoriamente por un símbolo que pertenece a $F \cup T$, excepto en el caso de que la posición seleccionada sea la raíz, donde se reemplazará necesariamente por un símbolo en F; (IV) si la posición pertenece a la cola, el símbolo en la posición seleccionada se cambia por otro símbolo en T.

Una característica de GEP que lo diferencia de otros paradigmas evolutivos es que fragmentos del genoma se pueden activar y saltar a otro lugar en el cromosoma (elementos transponibles), y esta característica da lugar a la definición de los siguientes operadores de transposición: *insertion sequence* (IS), *root insertion sequence* (RIS) y *gene transposition* (GT). El operador IS se utiliza con una pequeña probabilidad p_{is}, y permite que pequeños fragmentos de un gen (fragmentos con una función o un terminal en la primera posición) se transpongan a la cabeza de los genes, excepto a la raíz; este operador se ejecuta n_{is} veces sobre un cromosoma. El operador IS realiza los siguientes pasos: (I) dado un cromosoma, se selecciona un gen a partir del cual un fragmento de longitud entre $[1, l_{is}]$ (l_{is}-máxima longitud de un fragmento) se selecciona aleatoriamente; y (II) se selecciona un gen de destino donde insertar el fragmento copiado empezando en una posición aleatoria de la cabeza del gen (distinta de la raíz).

Por otra parte, el operador RIS se usa con una probabilidad baja p_{ris}, y permite que fragmentos pequeños con un símbolo no terminal en la primera posición se transpongan a la raíz de los genes; este operador se realiza n_{ris} veces sobre un cromosoma. Para ello, este operador realiza los siguientes pasos: (I) dado un cromosoma, se selecciona aleatoriamente un gen; (II) se selecciona un punto aleatorio en la cabeza del gen seleccionado y se busca hacia atrás hasta encontrar un símbolo no terminal; (III) y este fragmento se copia empezando en la raíz del gen. Durante la transposición RIS, los símbolos en la cabeza se desplazan hacia la derecha hasta acomodar completamente el nuevo fragmento copiado. Al igual que en el operador de transposición IS, la cola del gen que se transpone y los genes adyacentes no se modifican.

Por último, el operador GT se usa con una pequeña probabilidad p_{gt}, y permite que genes completos se transpongan al principio de los cromosomas. Para ellos, este operador realiza los siguientes pasos: (I) dado un cromosoma, se selecciona aleatoriamente un gen y se elimina de su lugar de origen; (II) y el gen se copia al principio del cromosoma, mientras que los genes restantes se desplazan a la derecha, manteniéndose la longitud original del cromosoma. Los operadores IS, RIS y GT producen siempre individuos válidos, sin embargo, hay que tener en cuenta que dichos operadores pueden modificar drásticamente la expresión de un gen; cuanto más cercano a la raíz del gen ocurra la transposición, más profundo será el cambio en la expresión del gen [13].

En el método propuesto también se aplicaron tres tipos de recombinación o cruce: recombinación en un punto (OP), recombinación en dos puntos (TP), y recombinación de genes (GP), con probabilidades de aplicación p_{op}, p_{tp}, y p_{gp}, respectivamente. Dados dos cromosomas, el operador OP realiza el cruce sobre un punto seleccionado aleatoriamente. El operador TP seleccionan aleatoriamente dos puntos y los fragmentos entre dichos puntos se intercambian entre ambos padres. En GP se intercambian los genes, ubicados en una

posición seleccionada aleatoriamente, entre ambos padres. Es de destacar que en los tres operadores, se cruzan dos padres y se obtienen dos nuevos individuos válidos. Además, el operador TP tiene más poder de transformación que el OP en problemas más complejos, especialmente cuando se utilizan cromosomas multigénicos. Por último, el operador GP no crea genes nuevos como OP y TP, sin embargo, cabe destacar que este operador puede introducir nuevo material en la población, ya que los genes intercambiados entre padres pueden ser muy diferentes.

Finalmente, se ha seguido un esquema generacional para actualizar la población que pasa de una generación a otra, donde el peor individuo de la nueva población es reemplazado por el mejor individuo de la población anterior. El método propuesto (GEPMTR, de ahora en adelante) sigue los pasos que se muestran en la Figura 2. Estos pasos se realizan n_g veces (número de generaciones), y al final del proceso se selecciona el mejor individuo encontrado a lo largo de todas las generaciones, cuyo cromosoma representa un modelo de regresión multi-salida que puede ser utilizado posteriormente para predecir las variables objetivo de conjuntos de ejemplos nunca antes visto.

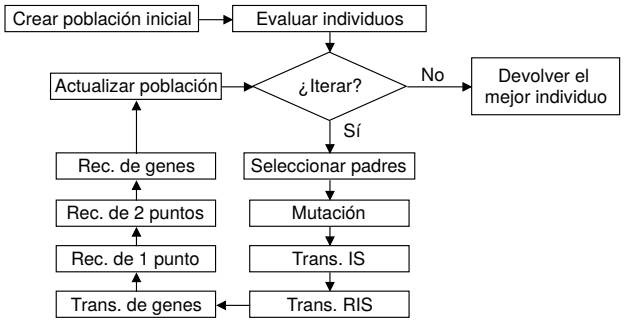


Figura 2. Diagrama de flujo del método GEPMTR.

III. Estudio Experimental

En esta sección primero se explica la configuración de los experimentos realizados en este trabajo, y posteriormente se presenta un análisis y discusión de los resultados obtenidos.

III-A. Configuración de los Experimentos

Los experimentos se han ejecutado sobre ocho conjuntos de datos de regresión multi-salida[1]. La Tabla I muestra los conjuntos de datos utilizados y sus características, como el número de instancias (n), variables descriptivas (d) y variables objetivo (q).

En el estudio experimental, se compara nuestra propuesta (GEPMTR) con otros dos algoritmos del estado del arte para regresión multi-salida, *Single Target* (ST) y *Regressor Chains* (RC) [2]. Tanto ST como RC usan árboles de regresión como predictores de regresión simple, tal y como recomiendan sus autores. Los parámetros utilizados para la ejecución de GEPMTR se muestran en la Tabla II; la mayoría de los valores son los recomendados por Ferreira [13]. Además, se utilizó una probabilidad global de cruce de 0.8, que corresponde con la suma de las probabilidades de los tres operadores de cruce utilizados.

[1]Los conjuntos de datos están disponibles públicamente en http://mulan.sourceforge.net/datasets-mtr.html

Tabla I
Conjuntos de datos de regresión multi-salida.

	n	d	q
slump	103	7	3
jura	359	15	3
sf1	323	10	3
atp1d	337	411	6
osales	639	413	12
wq	1060	16	14
oes97	334	263	16
oes10	403	298	16

Tabla II
Parámetros del algoritmo GEPMTR.

Parámetro	**Valor**
Conjunto de funciones (F)	$\{-, +, /, *, \sqrt{}, sin, cos, log\}$
Conjunto de terminales (T)	$\{x_1, x_2, \ldots, x_d\}$
Tamaño de la población (p)	100
Número de puntos de mutación (n_{mp})	2
Probabilidad de mutación (n_{mp})	0,1
Número de fragmentos IS (n_{is})	3
Máx. longitud de los fragmentos IS (l_{is})	3
Probabilidad IS (p_{is})	0,1
Número de fragmentos RIS (n_{ris})	3
Probabilidad RIS (p_{ris})	0,1
Probabilidad GT (p_{gt})	0,1
Probabilidad OP (p_{op})	0,2
Probabilidad TP (p_{tp})	0,5
Probabilidad GP (p_{gp})	0,1

Para la evaluación de los algoritmos, se utilizó la medida aRRMSE, que ha sido ampliamente usada para la evaluación de métodos de regresión multi-salida [1, 2]. La medida aRRMSE se calcula como se mostró en la Ecuación 1. En todos los experimentos, el valor de aRRMSE se estimó realizando un *10-folds cross-validation*, midiendo el error en cada uno de los conjuntos de test y promediando finalmente los resultados.

Para realizar comparaciones múltiples entre algoritmos, se ha utilizado el test de Friedman [16]. En casos donde el test de Friedman indicase que existen diferencias significativas en el rendimiento de los algoritmos, se ha realizado el test *post-hoc* de Holm [17] para realizar una comparación múltiple con un algoritmo de control. Por otra parte, en los casos donde se compararon dos métodos independientes, se empleó el test de Wilcoxon [18].

III-B. Resultados Experimentales

En el estudio experimental, primero se analizó el comportamiento de GEPMTR considerando diferentes valores del parámetro h= $\{5, 10, 15, 20\}$ y número de generaciones. La Tabla III muestra los resultados, donde en cada fila se han resaltado en negrita los mejores valores para cada h.

Se puede observar como en prácticamente todos los casos, cuando GEPMTR se ejecutó con 500 generaciones, los individuos evolucionaron hacia un mejor modelo predictivo,

Tabla III
Resultados de GEPMTR dependiendo de h y del número de generaciones.

	$h=5$		$h=10$		$h=15$		$h=20$	
	100 g.	**500 g.**	**100 g.**	**500 g.**	**100 g.**	**500 g.**	**100 g.**	**500 g.**
slump	**0,860**	0,872	0,848	**0,772**	0,864	**0,767**	0,845	**0,752**
jura	0,692	**0,669**	0,681	**0,646**	0,686	**0,644**	0,684	**0,642**
sf1	1,129	**1,083**	1,128	**1,128**	**1,142**	1,164	1,156	**1,141**
atp1d	0,555	**0,446**	0,541	**0,453**	0,545	**0,450**	0,544	**0,450**
osales	0,954	**0,860**	0,910	**0,861**	0,908	**0,881**	0,921	**0,888**
wq	0,976	**0,968**	0,974	**0,963**	0,975	**0,961**	0,976	**0,960**
oes97	0,660	**0,589**	0,650	**0,601**	0,658	**0,598**	0,666	**0,623**
oes10	0,530	**0,471**	0,530	**0,520**	0,546	**0,484**	0,541	**0,487**
p-value	**0,023**		**0,022**		**0,030**		**0,014**	

obteniendo un mejor rendimiento. Para cada valor diferente de h, se realizó el test de Wilcoxon para comparar el rendimiento de GEPMTR con 100 y 500 generaciones. Los *p-values* del test de Wilcoxon se muestran en la última fila de la Tabla III. El test estadístico rechazó la hipótesis nula en todos los casos para un nivel de significación α=$0,05$, lo cual demuestra que el algoritmo con 500 generaciones obtiene resultados significativamente mejores que con 100 generaciones.

Una vez verificado que GEPMTR funciona mejor con un mayor número de generaciones, el estudio se centra en el valor de h. La Tabla IV muestra los *rankings* de GEPMTR ejecutado con diferentes valores de h y con 500 generaciones sobre cada conjunto de datos. Esta tabla es un resumen de la Tabla III, pero esta se centra en los valores de *ranking* en lugar de aRRMSE. En cada fila, el algoritmo con un mejor rendimiento obtiene un *ranking* de 1, el siguiente un *ranking* de 2, y así sucesivamente. La última fila muestra el valor de *ranking* medio devuelto por el test de Friedman.

Tabla IV
Valores de ranking con diferentes valores de h y 500 generaciones.

	h=5	h=10	h=15	h=20
slump	4,00	3,00	2,00	**1,00**
jura	4,00	3,00	2,00	**1,00**
sf1	**1,00**	2,00	4,00	3,00
atp1d	**1,00**	4,00	2,50	2,50
osales	**1,00**	2,00	3,00	4,00
wq	4,00	3,00	2,00	**1,00**
oes97	**1,00**	3,00	2,00	4,00
oes10	**1,00**	4,00	2,00	3,00
Ranking medio	**2,13**	3,00	2,44	2,44

Se puede observar que, a pesar de que el test de Friedman no encontró diferencias significativas en el rendimiento de GEPMTR con los diferentes valores del parámetro h, la configuración con h=5 fue la que mejor *ranking* medio obtuvo, mostrando una tendencia a que cuanto más simples sean los modelos, mejor es su rendimiento predictivo.

En la segunda fase del estudio experimental, se realizó una comparación entre GEPMTR y dos algoritmos del estado del arte en MTR. La Tabla V muestra los resultados; para esta comparación, los resultados que se muestran para GEPMTR son el mínimo de las diferentes configuraciones de h con 500 generaciones. La última fila de la tabla muestra los valores de *ranking* medio obtenidos por el test de Friedman. En cada fila, el mejor valor de error se resalta en negrita. Los resultados muestran que el método propuesto obtiene mejores resultados que los otros dos algoritmos en siete de los ocho conjuntos de datos considerados, obteniendo también el mejor valor de *ranking* medio.

Tabla V
Comparación de GEPMTR con dos algoritmos del estado del arte en MTR.

	GEPMTR	**ST**	**RC**
slump	**0,752**	0,814	0,829
jura	**0,642**	0,696	0,704
sf1	1,083	1,127	**1,046**
atp1d	**0,446**	0,479	0,484
osales	**0,860**	0,925	0,965
wq	**0,960**	0,966	0,974
oes97	**0,589**	0,716	0,719
oes10	**0,471**	0,594	0,595
Ranking medio	**1,125**	2,125	2,750

El estadístico del test de Friedman fue igual a $10,75$, y la hipótesis nula fue rechazada con un *p-value* de $0,005$, existiendo diferencias significativas en el rendimiento de los algoritmos. Para detectar dónde estaban localizadas dichas diferencias se ejecutó el test de Holm, cuyos resultados se muestran en la Tabla VI, indicando que GEPMTR tiene un rendimiento predictivo significativamente mejor que el resto de algoritmos para un nivel de significación $\alpha = 0,05$.

Tabla VI
p-values ajustados calculados por el test de Holm.

	ST	**RC**
GEPMTR vs.	0,045	0,002

III-C. Discusión

La representación multi-génica utilizada resultó ser una manera efectiva para representar los cromosomas, permitiendo que cada individuo constituya una solución diferente y completa para el problema MTR. La manera de crear la población inicial, así como el uso de un operador de selección con menor presión, permitió la evolución de una población con suficiente diversidad previniendo que el método quedase atrapado en un mínimo local o en valles aplanados en las primeras generaciones.

Operadores como el de transposición IS, y los de recombinación OP y TP pueden introducir material genético beneficioso. Así, buenos bloques existentes en genes más adaptados se pueden transferir a otros genes, favoreciendo el intercambio de información entre genes, y por tanto, el modelado de relaciones estadísticas entre variables objetivo.

En cuanto a la eficiencia computacional de GEPMTR, el hecho de haber usado notación prefija, y por tanto que los genes puedan ser evaluados sin construir su correspondiente árbol de expresión, hace que la evaluación de los individuos se haya acelerado significativamente. GEPMTR puede ser paralelizado fácilmente, y por tanto, se puede reducir significativamente su

tiempo de cómputo. Por otro lado, no se consume tiempo en la descomposición del problema MTR en varios problemas de regresión simple, sino que el método propuesto trata directamente con los datos multi-salida. Además, GEPMTR tiene implícito un proceso de selección de características que puede ser muy beneficioso en el proceso de aprendizaje de modelos de regresión; la expresión matemática codificada por un gen involucra un número de variables mucho menor que el total de variables descriptivas existentes. Todas las características antes mencionadas permiten la aplicación efectiva del método propuesto en conjuntos de datos de gran escala.

GEPMTR obtuvo resultados prometedores en los experimentos realizados. Sin embargo, la principal desventaja de GEPMTR radica en el análisis de un considerable número de hyperparámetros. Para futuros trabajos será importante expandir el estudio experimental, por ejemplo analizando diferentes valores para las probabilidades de los operadores.

IV. Conclusiones

En este trabajo se ha presentado un método basado en GEP que resuelve el problema MTR mediante la técnica de regresión simbólica. Tanto la forma de codificación y evaluación de los individuos, como la posibilidad de paralelización, resultan en un método con un coste computacional aceptable y que permite ser usado en conjuntos de datos de gran escala.

Este trabajo corresponde con un estudio inicial, donde se busca analizar los beneficios del paradigma GEP para construir modelos de regresión multi-salida. En trabajos futuros, se estima mejorar algunos de los componentes del algoritmo, como un operador de selección que permita ajustar la presión selectiva. Por otro lado, sería interesante el desarrollo de *ensembles* usando GEPMTR.

Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2017-83445-P del Ministerio de Economía y Competitividad y Fondos FEDER, y además por la ayuda FPU del Ministerio de Educación FPU15/02948.

Referencias

[1] H. Borchani, G. Varando, C. Bielza, and P. Larrañaga, "A survey on multi-output regression," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 5, pp. 216–233, 2015.

[2] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas., "Multi-target regression via input space expansion: Treating targets as inputs," *Machine Learning*, vol. 104, no. 1, pp. 55–98, 2016.

[3] D. Kocev, S. Džeroski, M. D. White, G. R. Newell, and P. Griffioen, "Using single and multi-target regression trees and ensembles to model a compound index of vegetation condition," *Ecological Modelling*, vol. 220, no. 8, pp. 1159–1168, 2009.

[4] D. Tuia, J. Verrelst, L. Alonso, F. Pérez-Cruz, and G.Camps-Valls, "Multioutput support vector regression for remote sensing biophysical parameter estimation," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 4, pp. 804–808, 2011.

[5] A. Tsanas and A. Xifara, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," *Energy and Buildings*, vol. 49, no. 560-567, 2012.

[6] O. Reyes, C. Morell, and S. Ventura, "Evolutionary feature weighting to improve the performance of multi-label lazy algorithms," *Integrated Computer-Aided Engineering*, vol. 21, no. 4, pp. 339–354, 2014.

[7] T. Similä and J. Tikka, "Input selection and shrinkage in multiresponse linear regression," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 406–422, 2007.

[8] T. Aho, S. D. B. Ženko, and T. Elomaa, "Multi-target regression with rule ensembles," *Journal of Machine Learning Research*, vol. 373, pp. 2055–2066, 2009.

[9] G. Melki, A. Cano, V. Kecman, and S. Ventura, "Multi-target support vector regression via correlation regressor chains," *Information Sciences*, vol. 415-416, pp. 53–69, 2017.

[10] O. Reyes, A. Cano, H. Fardoun, and S. Ventura, "A locally weighted learning method based on a data gravitation model for multi-target regression," *International Journal of Computational Intelligence Systems*, vol. 11, pp. 282–295, 2018.

[11] S. Stijven, E. Vladislavleva, A. Kordon, L. Willem, and M. E. Kotanchek, *Genetic Programming Theory and Practice XIII*. Cham: Springer, 2016, ch. Prime-Time: Symbolic Regression Takes Its Place in the Real World, pp. 241–260.

[12] Y. Peng, C. Yuan, X. Qin, J. Huang, and Y. Shi, "An improved gene expression programming approach for symbolic regression problems," *Neurocomputing*, vol. 137, pp. 293–301, 2014.

[13] C. Ferreira, "Gene expression programming: A new adaptive algorithm for solving problems," *Complex Systems*, vol. 13, pp. 87–129, 2001.

[14] H. S. Lopes and W. R. Weinert, "Egipsys: An enhanced gene expression programming approach for symbolic regression problems," *International Journal of Applied Mathematics and Computer Science*, vol. 14, no. 3, pp. 375–384, 2004.

[15] J. M. Moyano, E. Gibaja, and S. Ventura, "An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains," in *IEEE Congress on Evolutionary Computation*, 2017, pp. 2015–2021.

[16] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 1940.

[17] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian journal of statistics*, pp. 65–70, 1979.

[18] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

Análisis de algoritmos de cuantificación basados en ajuste de distribuciones*

Alberto Castaño
Centro de Inteligencia Artificial
Universidad de Oviedo
Gijón, España
castanoalberto@uniovi.es

Laura Morán-Fernández
Departamento de Computación
Universidade da Coruña
A Coruña, España
laura.moranf@udc.es

Jaime Alonso
Centro de Inteligencia Artificial
Universidad de Oviedo
Gijón, España
jalonso@uniovi.es

Verónica Bolón-Canedo
Departamento de Computación
Universidade da Coruña
A Coruña, España
veronica.bolon@udc.es

Amparo Alonso-Betanzos
Departamento de Computación
Universidade da Coruña
A Coruña, España
ciamparo@udc.es

Juan José del Coz
Centro de Inteligencia Artificial
Universidad de Oviedo
Gijón, España
juanjo@uniovi.es

***Resumen*—La cuantificación consiste en predecir la proporción de las distintas clases presentes en una muestra dada de ejemplos. Un tipo de algoritmos de cuantificación se basa en estimar y ajustar las distribuciones subyacentes del conjunto de entrenamiento y de la muestra a predecir. La diferencia clave entre los métodos propuestos reside en cómo estimar las distribuciones: usando los propios atributos o las predicciones dadas por un clasificador. Este artículo analiza ambas alternativas y propone dos nuevos algoritmos. La conclusión es que los métodos basados en usar las predicciones de un clasificador no son en general peores y en ciertos casos producen mejores estimaciones.**

***Palabras clave:*—Cuantificación, Estimación de la prevalencia, Adaptación al dominio**

I. Introducción

Existen diversas aplicaciones reales que demandan predecir la distribución de probabilidad de las clases en un conjunto de ejemplos. Formalmente, este problema se denomina cuantificación [1]. Ejemplos típicos son predecir la proporción de comentarios positivos y negativos en una red social sobre un acontecimiento o producto durante un período de tiempo [2], cuantificar la cantidad de células dañadas en un tejido [3] o predecir el porcentaje de incidencias de cada tipo en un centro de atención al usuario [1]. En estas aplicaciones no se requiere dar una predicción individual para cada ejemplo sino que basta con retornar una predicción única para toda la muestra.

Aunque a primera vista la cuantificación pueda verse como un subproducto de la clasificación, se pueden diseñar algoritmos de cuantificación que no se basen simplemente en agregar las predicciones dadas por un clasificador. Es el caso de los métodos que estiman y ajustan distribuciones de muestras de ejemplos. Si nos centramos en la cuantificación binaria, la idea consiste en estimar durante el entrenamiento la distribución de la clase positiva y negativa de alguna forma y, en el momento de predecir una nueva muestra, estimar de la misma manera su distribución y aproximarla con una combinación de la distribución de la clase positiva y negativa dependiendo de la proporción de cada clase. La proporción que más aproxime ambas distribuciones será la que retorne el método [3], [4].

Se han propuesto sistemas basados en el ajuste de distribuciones no sólo en el campo de la cuantificación [3], sino también en otro problema relacionado como es el de los algoritmos de adaptación al dominio [5]. En estos últimos la idea es calcular la probabilidad a priori de las clases para actualizar un clasificador, sin necesidad de reentrenarlo, cuando la distribución de las clases cambia. El primer objetivo de este artículo es aunar y analizar ambas líneas de trabajo. Una diferencia clave entre ambas corrientes es cómo estimar las distribuciones. Mientras los algoritmos de adaptación al dominio se basan en estimar las distribuciones usando la información con la que se describen los ejemplos, los métodos de cuantificación proponen usar también las predicciones de un clasificador, asumiendo que ejemplos similares deben obtener predicciones parecidas. Un segundo objetivo de este artículo es comparar ambas alterativas. Asimismo, proponemos dos nuevos métodos que extienden algoritmos de adaptación al dominio, pero usando las predicciones de un clasificador.

Con todo ello, las contribuciones de este trabajo son: 1) presentar dos nuevos cuantificadores basados en ajustar distribuciones usando las predicciones de un clasificador, y 2) realizar un análisis experimental riguroso de los métodos basados en ajuste de distribuciones.

II. Cuantificación Binaria

Sea $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ un conjunto de entrenamiento, donde $\boldsymbol{x}_i$ es la representación de un ejemplo en el espacio de entrada $\mathcal{X} \subset \mathbb{R}^d$, e $y_i \in \mathcal{Y} = \{-1, +1\}$ su clase. El objetivo de la cuantificación binaria es obtener un modelo, $\bar{h}$, que dado un conjunto de ejemplos sin etiquetar, $T = \{\boldsymbol{x}_j\}_{j=1}^{m}$,

*Esta investigación ha sido financiada en parte por el Ministerio de Economía y Competitividad (MINECO) y el Fondo Europeo de Desarrollo Regional (FEDER), a través del proyecto coordinado TIN2015-65069-C2.

prediga la proporción de la clase positiva y la clase negativa. Dado que ambas son complementarias, basta con predecir la proporción o prevalencia de la clase positiva $\hat{p}$, siendo la de la clase negativa $1-\hat{p}$. En símbolos: $\bar{h} : \mathbb{N}^{\mathcal{X}} \longrightarrow [0,1]$, donde $\mathbb{N}^{\mathcal{X}}$ denota un multi-conjunto de ejemplos, representado por el número de veces que cada posible ejemplo $\boldsymbol{x} \in \mathcal{X}$ aparece en dicho multi-conjunto.

La aproximación más sencilla para producir un cuantificador es la denominada Clasificar y Contar (CC), que consiste en entrenar un clasificador binario, h, clasificar con él todos los ejemplos de T y contar los ejemplos de cada clase: $\hat{p}_{CC} = \bar{h}(T) = \frac{1}{m}\sum_{\boldsymbol{x}_i \in T} I(h(\boldsymbol{x}_i) = +1)$, siendo I la función indicatriz. Sin embargo, los estudios muestran [6] que CC produce en general resultados subóptimos, especialmente cuando la distribución de las clases cambia significativamente entre el entrenamiento y la fase de predicción. Por ese motivo se han propuesto distintos algoritmos de cuantificación. Una revisión completa de ellos puede encontrarse en [7].

La cuantificación, por su propia definición, es uno de esos problemas de aprendizaje [8] en los que se sabe de antemano que la distribución de los datos cambia, es decir, que $P_D(x,y) \neq P_T(x,y)$, ya que es obvio que al menos cambia $P_D(y) \neq P_T(y)$[1]. La cuestión es determinar qué elementos de la distribución, entre $P(x)$, $P(x|y)$ y $P(y|x)$, se asume que no cambian para facilitar el aprendizaje. De hecho, la clave y el primer paso para diseñar un algoritmo de cuantificación es definir precisamente eso. La mayoría de los algoritmos de cuantificación están diseñados bajo la asunción de que $P(x|y)$ se mantiene constante y que el resto de factores puede cambiar.

Esta asunción la sigue por ejemplo el método AC [1], que probablemente es el cuantificador más usado y que tomaremos como algoritmo básico de comparación en este estudio. La idea del método AC es que, dado que $P(x|y)$ es constante, los ratios de verdaderos positivos, tpr, (*true positive rate*) y de falsos positivos, fpr, (*false positive rate*) del clasificador usado por el método CC serán constantes también, aunque cambie la distribución de probabilidad de las clases. Eso implica que la prevalencia que predice el método CC, $\hat{p}_{CC}$, se puede escribir en función de la prevalencia real p:

$$\hat{p}_{CC} = tpr \cdot p + fpr \cdot (1-p). \tag{1}$$

Despejando p calculamos la prevalencia estimada por AC:

$$\hat{p}_{AC} = \frac{\hat{p}_{CC} - fpr}{tpr - fpr}. \tag{2}$$

En base a este desarrollo matemático, los pasos del método AC consisten en, primero, entrenar un clasificador y estimar su *tpr* y *fpr*, después usar el método CC para calcular $\hat{p}_{CC}$ y finalmente aplicar (2) para obtener la predicción final, $\hat{p}_{AC}$. En teoría AC produce predicciones perfectas siempre que se cumpla que: 1) $P(x|y)$ es constante y 2) las estimaciones del *tpr* y *fpr* son perfectas. Obviamente, es difícil que ambas condiciones se cumplan totalmente en problemas reales de una cierta complejidad, pero aún así el método suele ofrecer un buen rendimiento.

[1]Nótese que si $P_D(y) = P_T(y)$ el problema de cuantificación sería trivial

III. Métodos Basados en Ajuste de Distribuciones

El enfoque para el problema de la cuantificación en el que se centra este artículo se basa en estimar y ajustar las distribuciones de entrenamiento y test. La idea fundamental consiste en modificar la distribución de entrenamiento, que abusando de notación denotaremos por D', mediante una mezcla de la distribución de los ejemplos positivos, D^+, y la distribución de los negativos, D^-, en función de la prevalencia estimada de ambas clases, es decir,

$$D' = D^- \cdot (1-\hat{p}) + D^+ \cdot \hat{p}. \tag{3}$$

El objetivo es tratar de aproximar D' lo más posible a la distribución estimada para el conjunto de test T. La Figura 1 trata de ilustrar esta idea. En la figura izquierda se observa la distribución de los ejemplos positivos y de los negativos de entrenamiento. En la parte derecha se muestra la distribución observable en el conjunto de test. Asumiendo de nuevo que $P(x|y)$ es constante, y que por tanto las distribuciones de D^+ y D^- cambiarán uniformemente en función de la prevalencia de las clases, el objetivo es minimizar la diferencia entre la distribución de test y la distribución modificada D':

$$\min_{\hat{p}} \Delta(T, D') = \min_{\hat{p}} \Delta(T, D^- \cdot (1-\hat{p}) + D^+ \cdot \hat{p}). \tag{4}$$

En el ejemplo de la Figura 1 es evidente que la prevalencia de la clase positiva en el conjunto de test es menor que la observada en los datos de entrenamiento y que por lo tanto, tenemos que disminuir ese valor para ajustar las distribuciones.

Los métodos basados en ajustar distribuciones comparten un marco común que consta de los siguientes elementos:

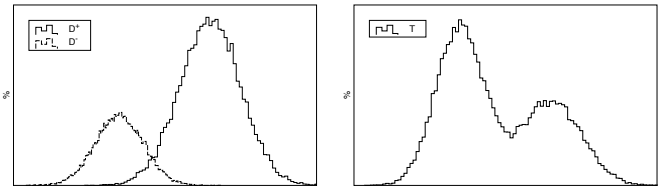
- una forma de estimar o representar las distribuciones,
- una medida Δ para compararlas, y
- un método para calcular el mínimo de (4).

La diferencia entre los métodos que se estudian en este artículo está en alguno de estos tres elementos, pero nuestro interés principal es analizar las distintas formas usadas para estimar las distribuciones y en menor medida las métricas empleadas para compararlas.

En la Figura 1 se ha dejado deliberadamente sin etiquetar el eje horizontal para indicar que la distribución se puede representar usando diferentes datos. No obstante, hay dos corrientes fundamentales: usar la propia descripción de los ejemplos (lo que algunos métodos llaman usar la información de las X's), o usar las predicciones dadas por un clasificador (usar las predicciones y)[2]. Las distribuciones de la Figura 1 podrían perfectamente representar funciones de densidad de probabilidad tanto del valor de un atributo del espacio de entrada $\mathcal{X}$, como de la probabilidad a posteriori dada por un clasificador de que un ejemplo pertenezca a la clase positiva, $P(y=+1|x)$.

En cuanto a la medida Δ, hay múltiples opciones incluyendo normas como $L1$ o $L2$ y medidas de similitud o divergencia entre distribuciones de probabilidad como la divergencia

[2]No confundir con usar las etiquetas reales presentes en D



(a) Distribuciones del cjto de entrenamiento (b) Distribución del cjto de test

Figura 1: Los métodos basados en comparación y ajuste estiman de alguna forma la distribución de (a) los ejemplos positivos y negativos disponibles en el entrenamiento, D^+ y D^-, y (b) de los ejemplos de la muestra a predecir, T. Después tratan de ajustar la combinación de las distribuciones de D^+ y D^- usando (3) para aproximar la distribución de T

de Kullback-Leibler (KLD) y la distancia de Hellinger. En el artículo analizaremos las medidas que se han propuesto hasta ahora y propondremos dos nuevos métodos usando una función basada en distancias y otra en rankings.

III-A. Ajuste Mediante la Distancia de Hellinger

Analizando la literatura de cuantificación, los métodos más relevantes basados en ajuste de distribuciones son los propuestos en [3]. Se trata de dos métodos basados en la distancia de Hellinger (HD) para comparar las distribuciones, donde éstas se representan mediante histogramas: en un caso del valor de los atributos (versión llamada HDX) y en el otro del valor de las predicciones de un clasificador (HDy). Usar histogramas hace que estos métodos tengan un hiperparámetro, b, que es el número de intervalos o *bins* usados en los histogramas.

Usando la definición de la distancia de Hellinger para el caso discreto multivariante, y aplicando (3) para representar D', resulta en el siguiente problema a minimizar:

$$\min_{\hat{p}} \frac{1}{d}\sum_{l=1}^{d}\sqrt{\sum_{k=1}^{b}\left(\sqrt{\frac{|T_{k,l}|}{m}}-\sqrt{\frac{|D^-_{k,l}|}{n^-}(1-\hat{p})+\frac{|D^+_{k,l}|}{n^+}\hat{p}}\right)^2}, \tag{5}$$

donde n^- y n^+ son el número de ejemplos en D^- y D^+ respectivamente y $|T_{k,l}|/m$, $|D^-_{k,l}|/n^-$ y $|D^+_{k,l}|/n^+$ son la proporción de ejemplos de T, D^- y D^+ que pertenece al bin k en la dimensión l. En el caso de HDy, solamente tenemos una dimensión que se corresponde con las predicciones del clasificador, luego el primer sumatorio desaparece.

Los autores proponen una búsqueda lineal, variando $\hat{p}$ en el rango $[0,1]$ en pequeños incrementos, para resolver (5). Sin embargo, la solución se puede hallar analíticamente, con más precisión y menos coste computacional, teniendo en cuenta la equivalencia entre la distancia de Hellinger y el coeficiente de Bhattacharyya, $HD(T,D') = \sqrt{1-BC(T,D')}$ [9], y resolviendo el siguiente problema de optimización:

$$\begin{aligned} \min_{\pi} \quad & 1-\sum_k \sqrt{T_k(D'\pi)_k}, \\ s.a. \quad & \sum \pi = 1, \quad \pi_i \geq 0, \end{aligned} \tag{6}$$

donde, sobrecargando la notación, definimos:

- T como una matriz con $bd \times 1$ en el caso del método HDX, con las proporciones de los bins del primer atributo en las b primeras filas, y así sucesivamente con el resto de atributos, y en el caso de HDy con solamente b filas y una columna con las proporciones de cada bin calculadas con las predicciones del clasificador,
- D' es una matriz de $bd \times 2$ en el caso del método HDX, y $b \times 2$ en el caso del HDy, con idéntica estructura a la matriz T, donde las dos columnas representan, en este orden, las proporciones de la clase negativa y positiva, y
- π una matriz 2×1 con dos variables, π_1 la prevalencia de la clase negativa, y π_2 la de la positiva, $\hat{p}$.

Este problema puede resolverse con cualquier librería de optimización convexa; nosotros empleamos `cvxpy` para Python.

En teoría, la ventaja de HDX es usar directamente la información disponible en los atributos con los que se representan los ejemplos, lo cual puede ser una desventaja en espacios de alta dimensionalidad, o cuando los problemas dependan mucho de la interacción entre los valores de los atributos, ya que estos se consideran de forma en cierto modo independiente al calcular la HD, como hemos descrito anteriormente. La ventaja del método HDy es que resume en una sola dimensión, a través de un clasificador, la distribución de los ejemplos, asumiendo que ejemplos parecidos deben obtener predicciones similares. La desventaja del método HDy es que el clasificador debe entrenarse adecuadamente y las estimaciones de las predicciones de los ejemplos deben hacerse no en reescritura, sino usando validaciones cruzadas con un alto número de particiones para evitar el sobreajuste en las predicciones.

III-B. Ajuste Mediante la Distancia Energy

Varios métodos de adaptación al dominio [5], [10] se basan en calcular la nueva distribución de las clases (denominadas probabilidades a priori en este contexto). La idea es usar esas nuevas probabilidades a priori para ajustar el clasificador disponible sin necesidad de reentrenarlo. Es obvio que dichos métodos también pueden usarse como algoritmos de cuantificación, aunque su objetivo sea otro bien distinto. En la literatura de adaptación al dominio es habitual que la comparación entre las distribuciones de entrenamiento y test se haga solamente usando el valor de los atributos [4], [11]. Dichos métodos plantean los mismos problemas que comentamos anteriormente para el método HDX: a veces es complejo calcular las densidades antes de compararlas. No obstante, siguiendo ese enfoque, recientemente se ha publicado un método [12] que destaca entre los propuestos en ese campo ya que es computacionalmente muy eficiente y tiene un mejor rendimiento que otros métodos anteriores [4].

Dicho algoritmo, que denotaremos por EDX, se basa en minimizar la distancia *Energy* (ED) entre la distribución de T y la distribución modificada D' definida por:

$$\begin{aligned} \min_{\hat{p}} \quad & 2\cdot\mathbb{E}_{\boldsymbol{x}_j \backsim T, \boldsymbol{x}_i \backsim D'}||\boldsymbol{x}_j-\boldsymbol{x}_i|| \\ & -\mathbb{E}_{\boldsymbol{x}_j,\boldsymbol{x}'_j \backsim T}||\boldsymbol{x}_j-\boldsymbol{x}'_j|| - \mathbb{E}_{\boldsymbol{x}_i,\boldsymbol{x}'_i \backsim D'}||\boldsymbol{x}_i-\boldsymbol{x}'_i||, \end{aligned} \tag{7}$$

donde $||\cdot||$ representa la distancia euclídea. Nótese que el segundo término puede suprimirse al ser independiente de $\hat{p}$ debido a que solo interviene la distribución de T. Omitimos aquí la derivación matemática para resolver este problema ya que es similar al que detallaremos en la sección siguiente.

IV. Métodos Propuestos

Nuestra primera propuesta consiste en hacer el método complementario al EDX, que llamaremos EDy y que emplea las predicciones de un clasificador h en lugar de los atributos que definen $\mathcal{X}$. Para ello debemos minimizar la expresión:

$$\begin{aligned}\min_{\hat{p}} \quad & 2 \cdot \mathbb{E}_{\boldsymbol{x}_j \backsim T, \boldsymbol{x}_i \backsim D'} ||h(\boldsymbol{x}_j) - h(\boldsymbol{x}_j)||_1 \\ & -\mathbb{E}_{\boldsymbol{x}_i, \boldsymbol{x}'_i \backsim D'} ||h(\boldsymbol{x}_i) - h(\boldsymbol{x}'_i)||_1.\end{aligned} \tag{8}$$

Si desdoblamos ambos términos aplicando (3) tenemos

$$\begin{aligned}\min_{\hat{p}} \quad & 2 \cdot (1-\hat{p}) \cdot \mathbb{E}_{\boldsymbol{x}_j \backsim T, \boldsymbol{x}_i \backsim D^-} ||h(\boldsymbol{x}_j) - h(\boldsymbol{x}_i)||_1 \\ & +2 \cdot \hat{p} \cdot \mathbb{E}_{\boldsymbol{x}_j \backsim T, \boldsymbol{x}_i \backsim D^+} ||h(\boldsymbol{x}_j) - h(\boldsymbol{x}_i)||_1 \\ & -(1-\hat{p})^2 \cdot \mathbb{E}_{\boldsymbol{x}_i, \boldsymbol{x}'_i \backsim D^-} ||h(\boldsymbol{x}_i) - h(\boldsymbol{x}'_i)||_1 \\ & -2 \cdot \hat{p} \cdot (1-\hat{p}) \mathbb{E}_{\boldsymbol{x}_i \backsim D^+, \boldsymbol{x}'_i \backsim D^-} ||h(\boldsymbol{x}_i) - h(\boldsymbol{x}'_i)||_1 \\ & -\hat{p}^2 \cdot \mathbb{E}_{\boldsymbol{x}_i, \boldsymbol{x}'_i \backsim D^+} ||h(\boldsymbol{x}_i) - h(\boldsymbol{x}'_i)||_1.\end{aligned} \tag{9}$$

En la práctica podemos aproximar cada esperanza usando las correspondientes muestras de ejemplos disponibles:

$$\mathbb{E}_{\substack{\boldsymbol{x}_j \backsim T \\ \boldsymbol{x}_i \backsim D^-}} \ldots \approx \mu_{T,D^-} = \frac{1}{mn^-} \sum_{\boldsymbol{x}_j \in T} \sum_{\boldsymbol{x}_i \in D^-} ||h(\boldsymbol{x}_j) - h(\boldsymbol{x}_i)||_1. \tag{10}$$

Sustituyendo todos los términos tenemos

$$\begin{aligned}\min_{\hat{p}} \quad & 2(1-\hat{p})\mu_{T,D^-} + 2\hat{p}\mu_{T,D^+} \\ & -(1-\hat{p})^2 \mu_{D^-,D^-} - 2\hat{p}(1-\hat{p})\mu_{D^-,D^+} - \hat{p}^2 \mu_{D^+,D^+},\end{aligned} \tag{11}$$

un problema fuertemente convexo [12] en el que derivando e igualando a cero obtenemos que

$$\hat{p} = \frac{\mu_{T,D^-} - \mu_{T,D^+} - \mu_{D^-,D^-} + \mu_{D^-,D^+}}{-\mu_{D^-,D^-} + 2\mu_{D^-,D^+} - \mu_{D^+,D^+}}. \tag{12}$$

El segundo método propuesto trata de completar el tipo de medidas consideradas para comparar distribuciones. Por un lado tenemos una medida basada en histogramas (Hellinger), una segunda basada en distancias (*Energy*) y se podría considerar una tercera opción que se basase en rankings, por ejemplo usando el criterio Cramér-von Mises (CvM). Después de analizar varias alternativas que utilizaban medidas derivadas de CvM [13], observamos que planteaban ciertos problemas en el contexto de la cuantificación binaria. El principal es que los errores en los rankings dados por un clasificador no son simétricos respecto a las clases: la clase positiva tiende a dar valores más altos, por ejemplo, al usar un clasificador probabilístico que devuelva $P(y = +1|x)$, lo que hace que los errores sean mayores y la medida sea sesgada hacia esa clase. Por ese motivo, finalmente decidimos usar el enfoque propuesto en [14] y crear el método CvMy que consiste en:

1. Calcular un ranking conjunto de los ejemplos disponibles en T, D^- y D^+,
2. Utilizar la ED para ajustar las distribuciones.

Es decir, aplicamos el mismo algoritmo que en el método EDy, pero en lugar de calcular la distancia entre las predicciones, calculamos la distancia entre los rankings de las predicciones. En este caso μ_{T,D^-} se calcularía como

$$\mu_{T,D^-} = \frac{1}{mn^-} \sum_{\boldsymbol{x}_j \in T} \sum_{\boldsymbol{x}_i \in D^-} ||r(h(\boldsymbol{x}_j)) - r(h(\boldsymbol{x}_i))||, \tag{13}$$

donde $r(h(\boldsymbol{x}))$ nos devuelve la posición en el ranking conjunto para todos los ejemplos disponibles, tanto del conjunto de entrenamiento como de la muestra de test a predecir. Luego la diferencia entre EDy y CvMy es simplemente que en un caso usamos las distancias entre las predicciones y en otro caso las diferencias entre los rankings de esas predicciones.

V. Experimentos

El objetivo de los experimentos[3] era comparar los métodos descritos en las secciones III y IV sobre varios conjuntos de datos. Se prestó especial atención a aquellos pares de métodos que, empleando la misma técnica, actúan sobre X ó y. Otro aspecto a analizar es la diferencia entre los métodos que utilizan distintas métricas para ajustar las distribuciones. Con el fin de tener una base de comparación, se incluyó el método AC. Para realizar los experimentos se emplearon 41 conjuntos de datos. Entre ellos se encuentran conjuntos binarios y otros que son versiones binarizadas de conjuntos originalmente multiclase. Así, por ejemplo, el conjunto balance.1 es un conjunto binario en el que la clase 1 original es la clase positiva, formando el resto la clase negativa.

Los métodos que necesitan un clasificador (AC, HDy, EDy y CvMy) fueron entrenados garantizando que usen exactamente los mismos clasificadores. Como algoritmo de clasificación se optó por utilizar *Random Forest* (RF) con salida probabilística para obtener modelos no lineales. La representación usada por el método HDy se generó mediante 8 *bins* por ser el valor que da mejores resultados en estudios previos [15]. Respecto a la hiperparametrización de RF, se llevó a cabo una búsqueda donde la profundidad variaba entre $[1, 5, 10, 15, 20, 25, 30]$, el número de árboles entre $[9, 18, 27, 36, 45, 54, 63]$, y el número mínimo de ejemplos para los nodos hoja entre $[1, 2, 4, 6, 8, 10]$. Dicha búsqueda se realizó optimizando la media geométrica mediante una validación cruzada de tres particiones de tal forma que se obtuvieran buenos clasificadores incluso cuando las clases no estuvieran balanceadas.

Todos los modelos se entrenaron sobre las mismas particiones de los conjuntos de datos en subconjuntos de entrenamiento y test, usando el $70\,\%$ de los datos para entrenar y el $30\,\%$ restante para testear el modelo, haciendo 20 repeticiones. Con cada partición de test se generaron a su vez 50 conjuntos aleatorios con diferente prevalencia mediante muestreo con reemplazamiento. La Tabla I presenta el error absoluto medio de las 1000 muestras totales de test, $\frac{1}{1000}\sum_{i=1}^{1000} |\hat{p}_i - p_i|$.

[3]Con el fin de facilitar la reproducibilidad de estos experimentos, los conjuntos de datos, el código y los resultados obtenidos están disponibles en http://github.com/albertorepo/analisis-cuantificacion.

Tabla I: Error absoluto medio para cada método sobre cada conjunto de datos

conjunto	AC	CvMy	EDX	EDy	HDX	HDy
acute.a	**0.037108**	0.063550	0.043139	0.045118	0.055545	0.038207
acute.b	**0.032788**	0.066146	0.036890	0.041999	0.041406	0.054910
balance.1	0.035783	0.033945	**0.023380**	0.030977	0.031761	0.029968
balance.3	0.036954	0.041549	0.031478	0.036346	0.038860	**0.028690**
breast-cancer	**0.016127**	0.041405	0.021433	0.019083	0.020661	0.016980
cmc.1	0.086332	0.078364	0.078307	0.076136	0.069633	**0.066261**
cmc.2	0.107941	0.119897	**0.067238**	0.115609	0.070623	0.108277
cmc.3	0.123313	0.102359	0.097244	0.094503	**0.089708**	0.099939
coil	**0.074547**	0.088761	0.091519	0.084081	0.152487	0.111920
ctg.1	**0.015548**	0.024147	0.029277	0.017414	0.039771	0.015466
ctg.2	0.029396	0.034666	0.033221	0.029420	0.035380	**0.027457**
ctg.3	0.036554	0.034602	0.037974	**0.028830**	0.072301	0.029866
default_credit	0.019715	0.022824	0.019998	0.020298	**0.019029**	0.022613
diabetes	0.072729	0.054864	0.059925	**0.053669**	0.079258	0.057511
german	0.087824	0.094541	**0.078250**	0.094154	0.100175	0.108456
haberman	0.243842	0.249899	**0.166582**	0.235816	0.263772	0.210012
ionosphere	**0.048585**	0.066306	0.060841	0.055436	0.055167	0.051017
iris.1	0.002824	0.093470	**0.016286**	0.019482	0.019413	0.036264
iris.2	0.055456	0.072105	0.083458	**0.048620**	0.050365	0.060695
iris.3	0.056763	0.066537	0.048433	0.046074	**0.040583**	0.075169
lettersH	0.020787	0.026393	0.027549	0.025537	0.024807	**0.016682**
mammographic	0.055443	0.049573	0.047413	0.044584	0.040516	**0.039722**
normtrans	0.126085	0.115678	**0.083582**	0.111233	0.184914	0.140455
normwine.1	0.044364	0.053990	**0.031585**	0.037656	0.036819	0.041440
normwine.2	0.050073	0.052678	**0.041247**	0.044185	0.067116	0.052616
normwine.3	0.043011	0.058478	**0.037641**	0.037738	0.058761	0.059418
pageblocks.5	0.042875	0.045982	0.056530	0.042546	0.116209	**0.037511**
phoneme	0.016554	0.019391	0.020960	0.015883	0.017325	**0.012115**
semeion.8	0.058507	0.058614	0.086534	0.058085	0.053439	**0.042278**
sonar	0.113939	0.108738	**0.103082**	0.107576	0.126948	0.110015
spambase	**0.008127**	0.015965	0.011639	0.010721	0.020363	0.008359
spectf	0.184524	0.162914	0.105445	0.150938	**0.088333**	0.134393
tictactoe	**0.048306**	0.061411	0.081694	0.056250	0.078846	0.048768
transfusion	0.107033	0.132669	**0.079578**	0.116017	0.162173	0.139883
wdbc	0.022888	0.030576	0.025309	0.016853	0.027982	**0.016830**
wine-quality-red	0.045189	0.037138	0.042694	**0.035102**	0.047469	0.037303
wine-quality-white	0.031770	0.030004	**0.028205**	0.028864	0.029644	0.025918
wine.1	0.037356	0.051096	0.034395	**0.033942**	0.036904	0.044647
wine.2	0.044871	0.056931	0.051078	**0.041804**	0.068114	0.051526
wine.3	0.037569	0.057219	0.036618	**0.035190**	0.048123	0.038674
yeast	0.060799	0.062919	**0.051120**	0.063225	0.060161	0.052552

El método ganador para cada conjunto se destaca en negrita. Puede verse que no hay un método que gane claramente a los demás. Tampoco se observan diferencias concluyentes entre las formas de estimar las distribuciones ni entre las medidas para ajustarlas. Entre los métodos propuestos, el algoritmo EDy es superior, siendo competitivo con los algoritmos previos, lo que parece no ocurrir en el caso del método CvMy.

Para analizar los resultados desde un punto de vista estadístico hemos aplicado análisis bayesianos en lugar de los tradicionales tests de contraste de la hipótesis nula ya que estamos de acuerdo con las desventajas de estos últimos apuntadas en [16]. Previo a la aplicación de estos tests, es necesario definir la *región de equivalencia práctica* (*rope*, por sus siglas en inglés): dos métodos se consideran prácticamente equivalentes si la diferencia dada una métrica no supera un cierto umbral. En nuestro caso, consideramos dos cuantificadores equivalentes si la diferencia en error absoluto es menor del $1\,\%$. La elección del error absoluto como medida de evaluación se debe a que es una métrica acotada entre 0 y 1 y por lo tanto, *rope* puede definirse como un porcentaje de variación dentro de ese intervalo. Para otras métricas como la KLD no sería trivial la elección de tal región, dado que su valor tiene un rango infinito, además de una difícil interpretación práctica. Una vez fijado el valor de *rope* es posible realizar el análisis para cada par de métodos tanto en cada conjunto individual (Tabla II) como para todos los conjuntos usando un test jerárquico [16] (Tabla III). En el primer caso podemos calcular la probabilidad de cada una de las tres posibilidades: que gane uno de los dos cuantificadores o que los resultados caigan en la zona de equivalencia. Cuando una de esas tres probabilidades es mayor que el $95\,\%$ consideramos que hay una diferencia significativa en favor de esa alternativa, y si no, lo marcamos como un conjunto *sin decisión*. Analizando la Tabla II vemos una ligera ventaja en favor de los métodos y frente a los métodos X, más clara en el caso de HDy frente a HDX que en la comparación EDy vs. EDX. En el caso de CvMy, es inferior a HDy y EDy, especialmente frente al segundo. En cambio EDy obtiene mejores resultados que el resto de métodos excepto HDy, pero la diferencia entre ambos es pequeña. Si analizamos los resultados a nivel global con el

Tabla II: Número de conjuntos para los que el test bayesiano decide que existe diferencia significativa ($\geq 95\,\%$)

Par ($m1$-$m2$)	$m1$	$rope$	$m2$	Sin decisión
EDX-AC	16	7	16	2
EDX-CvMy	17	11	7	6
EDX-EDy	9	16	12	4
EDX-HDX	10	15	7	9
EDX-HDy	10	15	14	2
EDy-AC	14	14	9	4
EDy-CvMy	11	24	0	6
EDy-HDX	15	11	9	6
EDy-HDy	8	13	10	10
HDX-AC	11	9	17	4
HDX-CvMy	16	9	13	3
HDX-HDy	6	13	19	3
HDy-AC	11	19	11	0
HDy-CvMy	15	15	6	5
AC-CvMy	16	11	9	5

Tabla III: Resultados del test bayesiano jerárquico

Par ($m1$-$m2$)	$P(m1 \gg m2)$	$P(rope)$	$P(m2 \gg m1)$
EDX-AC	0.537	0.000	0.463
EDX-CvMy	0.970	0.013	0.018
EDX-EDy	0.239	0.252	0.509
EDX-HDX	0.347	0.598	0.055
EDX-HDy	0.342	0.009	0.649
EDy-AC	0.523	0.202	0.275
EDy-CvMy	0.286	0.714	0.000
EDy-HDX	0.768	0.134	0.099
EDy-HDy	0.105	0.744	0.150
HDX-AC	0.162	0.002	0.837
HDX-CvMy	0.706	0.001	0.293
HDX-HDy	0.022	0.005	0.973
HDy-AC	0.451	0.072	0.477
HDy-CvMy	0.971	0.005	0.024
AC-CvMy	0.814	0.005	0.180

test jerárquico (Tabla III) vemos que hay pocas diferencias significativas (HDy vs. CvMy, HDy vs. HDX y EDy vs. CvMy). Se observa de nuevo la ligera ventaja de los métodos y, EDy y HDy, frente a EDX y HDX. Estos datos son un resumen de la distribución de las diferencias entre cada par de métodos. La forma de la distribución puede verse mediante un gráfico simplex (Figura 2). Por ejemplo, es interesante ver que en la comparación EDy-CvMy la nube de puntos está muy alejada del vértice de CvMy, a pesar de que la diferencia no resulte significativa, mostrando una superioridad clara.

VI. Conclusiones

Este trabajo tenía como principal objetivo realizar un análisis comparativo entre diferentes algoritmos de cuantificación basados en ajuste de distribuciones. Los resultados experimentales obtenidos muestran que los métodos basados en las y's, es decir, en el uso de clasificadores, no son en general peores, obteniendo mejores estimaciones en ciertas ocasiones. En cuanto a los métodos propuestos, EDy se configura como una buena alternativa, pero no así el algoritmo propuesto basado en ranking, CvMy. Como futura línea de investigación, se plantea la aplicación de algún algoritmo de selección de características en los métodos X para tratar de obtener un subconjunto con los mejores atributos. Además, sería interesante realizar una búsqueda de características en los problemas

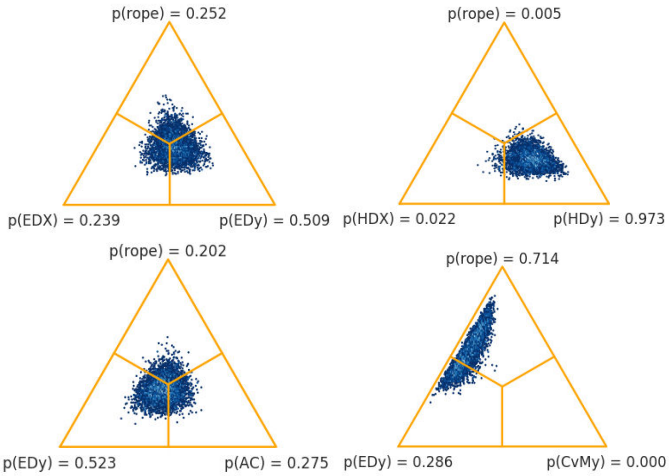


Figura 2: Gráficos simplex de la comparación mediante el test jerárquico bayesiano de varios pares de métodos

(alta dimensionalidad de entrada, presencia de ruido, etc.), que puedan implicar un mejor comportamiento en cada método.

Referencias

[1] G. Forman, "Quantifying counts and costs via classification," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 164–206, 2008.

[2] P. González, J. Díez, N. Chawla, and J. J. del Coz, "Why is quantification an interesting learning problem?" *Progress in Artificial Intelligence*, pp. 1–6, 2016.

[3] V. González-Castro, R. Alaiz-Rodríguez, and E. Alegre, "Class distribution estimation based on the hellinger distance," *Information Sciences*, vol. 218, pp. 146–164, 2013.

[4] M. Sugiyama, T. Kanamori, T. Suzuki, M. C. du Plessis, S. Liu, and I. Takeuchi, "Density-difference estimation," *Neural Computation*, vol. 25, no. 10, pp. 2734–2775, 2013.

[5] A. Margolis, "A literature review of domain adaptation with unlabeled data," University of Washington, Tech. Rep., 2011.

[6] J. Barranquero, P. González, J. Díez, and J. J. del Coz, "On the study of nearest neighbor algorithms for prevalence estimation in binary problems," *Pattern Recognition*, vol. 46, no. 2, pp. 472–482, 2013.

[7] P. González, A. Castaño, N. V. Chawla, and J. J. del Coz, "A review on quantification learning," *ACM Computing Surveys*, vol. 50, no. 5, pp. 74:1–74:40, 2017.

[8] J. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012.

[9] A. Firat, "Unified framework for quantification," *arXiv preprint arXiv:1606.00868*, 2016.

[10] H. Daume III and D. Marcu, "Domain adaptation for statistical classifiers," *JAIR*, vol. 26, pp. 101–126, 2006.

[11] A. Iyer, S. Nath, and S. Sarawagi, "Maximum mean discrepancy for class ratio estimation: Convergence bounds and kernel selection." in *ICML*, 2014, pp. 530–538.

[12] H. Kawakubo, M. C. Du Plessis, and M. Sugiyama, "Computationally efficient class-prior estimation under class balance change using energy distance," *IEICE Tran. on Inf. and Sys.*, vol. 99, pp. 176–186, 2016.

[13] L. Morán-Fernández, V. Bolón-Canedo, and A. Alonso-Betanzos, "A distributed approach for classification using distance metrics," in *ESANN*, 2017.

[14] J. Curry, X. Dang, and H. Sang, "A rank-based Cramér-von-Mises-type test for two samples," *arXiv preprint arXiv:1802.06332*, 2018.

[15] P. Pérez-Gállego, J. R. Quevedo, and J. J. del Coz, "Using ensembles for problems with characterizable changes in data distribution: A case study on quantification," *Information Fusion*, vol. 34, pp. 87–100, 2017.

[16] A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis," *Journal of Machine Learning Research*, vol. 18, no. 77, pp. 1–36, 2017.

k-Vecinos más Cercanos Difuso para Clasificación Monotónica

Sergio González, Salvador García, Francisco Herrera
Dept. de Ciencias de la Computación e Inteligencia Artificial
Universidad de Granada
Granada, España
{sergiogvz, salvagl, herrera}@decsai.ugr.es

Sheng-Tun Li
Dept. de Gestión Industrial y de Información
Universidad Nacional Cheng Kung
Tainan, Taiwan
stli@mail.ncku.edu.tw

Robert John
Escuela de Ciencias de la Computación
Universidad de Nottingham
Nottingham, Reino Unido
robert.john@nottingham.ac.uk

***Resumen*—La clasificación con restricciones monotónicas proviene de una necesidad de algunos problemas reales, en los que la variable de salida no disminuyen con un aumento de las entrada, o viceversa. Los conjuntos de datos reales con frecuencia tiene una gran cantidad de ruido de clase. Este incrementa el número de violaciones monotónicas y repercute negativamente en el rendimiento de los clasificadores. Por lo que, algunos métodos deben recurrir a técnicas de re-etiquetado que pueden alterar el conocimiento del problema. Además, la inclusión de estas restricciones obligan a los clasificadores a diversificar sus objetivos entre precisión y monotonicidad.**

Con esta contribución, se propone un nuevo modelo basado en los k-Vecinos más Ceranos Difuso para la clasificación con restricciones monotónicas, MonFkNN. Este clasificador incorpora un nuevo cálculo de pertenencias difusas, que lo dota de robustez frente ruido monotónico sin necesidad de re-etiquetado. Asimismo, se ha diseñado para ser adaptable a las diferentes necesidades del problema. Tras varios estudios experimentales, ha demostrado ser tremendamente más preciso igualando el grado de monotonicidad de los mejores métodos de la literatura.

***Palabras Clave*—Ciencia de datos, aprendizaje perezoso, vecinos más cercanos difuso, restricciones monotónicas, clasificación ordinal, regresión ordinal**

I. Introducción

La evaluación de activos o individuos [1], [2] tienen como objetivo determinar los artículos más valiosos según sus virtudes, es decir, la clasificación en etiquetas ordinales según sus atributos ordinales. Además, estas aplicaciones suelen conllevar una restricción monotónica entre las entradas y la clase. Es decir, la predicción de clase de un individuo no debe disminuir con un mejor valor para una determinada variable, fijando el resto. Estos problemas se conocen como clasificación con restricciones de monotonicidad [3] y la ruptura de estas, como violaciones de monotonicidad.

Este trabajo ha contado con el apoyo del Proyecto Español TIN2017-89517-P y una beca de investigación (FPU) otorgada a S. González por el MECD.

Estas tareas de aprendizaje exigen otros requisitos además de modelos precisos, como la consistencia monótona de las predicciones o la minimización de los errores de clasificación. Sin embargo, estos otros objetivos pueden perjudicar la precisión. Por lo tanto, se debe buscar un equilibro justo entre las diferentes necesidades de cada problema.

En estos últimos años, se ha diseñado nuevos algoritmos que consideran estas restricciones [1], [3]–[6]. El aprendizaje basado en la instancia ha demostrado ser una buena aproximación para la clasificación monotónica [4], [6]. Sin embargo, algunos de estos métodos, como la variante monotónica de k-Vecinos más Cercanos [4] (MkNN), necesitan aprender de un conjunto completamente monótono para asegurar predicciones monótonas. Esta circunstancia rara vez se cumple en conjuntos reales, donde el ruido y las discrepancias son comunes. Por ello, se hace uso de estrategias de re-etiquetado que alteran las etiquetas de clase de algunos ejemplos forzando un conjunto monotónico [4].

En clasificación estándar, la versión difusa de los k-Vecinos más Cercanos [7] (FkNN) es un método muy sólido con un gran rendimiento, gracias a su alta robustez al ruido [8]. En esta contribución, se presenta un nuevo modelo diseñado sobre FkNN con nociones de MkNN para tener en cuenta las restricciones de monotonicidad, llamado MonFkNN. La robustez de FkNN se ha adaptado para reducir el impacto de las violaciones monotónicas sin la necesidad de re-etiquetado. Adicionalmente, se han añadido otras técnicas para tratar las pertenencias de instancia de una manera más adecuada según la monotonicidad. Nuestra propuesta MonFkNN es un clasificador flexible que cubre diferentes necesidades de monotonicidad y precisión, con buen equilibrio entre ambos, ajustando sus parámetros.

Este documento está organizado de la siguiente manera. En la Sección II, presentamos el problema de la clasificación con restricciones monotónicas y los métodos relacionados con

nuestra propuesta. La sección III está dedicada a explicar con lujo detalle la propuesta MonFkNN. En la Sección IV, se presenta el marco experimental utilizado en los diferentes estudios empíricos. La sección V plantea tres estudios experimentales: un análisis sobre los parámetros de nuestro método y sus dos configuraciones más relevantes, una profunda comparación con el Estado-del-Arte y un estudio de la robustez de MonFkNN frente al ruido. Finalmente, en la Sección VI, se presentan las principales conclusiones de este estudio.

II. Preliminares

En esta sección, se introducen los preliminares sobre los que se construye este trabajo.

II-A. Clasificación Monotónica

La clasificación monotónica tiene como objetivo predecir la etiqueta de la clase y a partir del vector de la característica de entrada x, donde $y \in \mathcal{Y} = \{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_c\}$, $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^K$. Como propiedad principal, los atributos y sus predicciones están monotónicamente restringidos por conocimiento previo del problema, es decir, $\mathbf{x} \succeq \mathbf{x}' \rightarrow f(\mathbf{x}) \geq f(\mathbf{x}')$ [9], donde $\mathbf{x} \succeq \mathbf{x}'$ implica $\forall_{j=1,\ldots,K}, \mathbf{x}_j \geq \mathbf{x}'_j$, esto es, $\mathbf{x}$ domina $\mathbf{x}'$. Por lo tanto, el objetivo principal es construir clasificadores que no violen estas restricciones, conocidos como clasificadores monótonos.

Se pueden distinguir dos tipos diferentes de clasificadores monotónicos: los modelos monotónicos aproximados, que minimizan el número de violaciones monotónicas en sus decisiones, y los clasificadores monotónicos puros, cuyas predicciones son siempre monotónicas en relación con el entrenamiento. Esto último es difícil de lograr, especialmente en aplicaciones de la vida real, donde los conjuntos de datos de formación rara vez son puramente monótonos. Para ser considerado monotónico, un conjunto de datos debe tener todos sus pares de instancias monotónicas entre sí [6]: $\mathbf{x}_i \succeq \mathbf{x}_j \rightarrow y_i \geq y_j, \forall_{i,j}$.

II-B. k-Vecinos más Cercanos Monotónico

La aproximación de los k-Vecinos más Cercanos para la clasificación monotónica [4] (MkNN) modifica la regla estándar del vecino más cercano para evitar violaciones monotónicas en sus predicciones. Para ello, MkNN calcula para cada nuevo ejemplo x_i el rango $r_i = [y_{min}, y_{max}]$ de etiquetas de clase válidas, que satisfagan las restricciones monotónicas. El y_{min} inferior de r_i se calcula como la etiqueta de clase más alta de todas las instancias por debajo del ejemplo x_i. Análogamente, el límite superior y_{max} es la etiqueta de clase mínima de las instancias superiores a x_i (véase la Ec. 1).

$$r_i = \begin{cases} y_{min} = \text{máx}\{y | (x,y) \in \mathcal{D} \wedge x_i \succeq x\} \\ y_{max} = \text{mín}\{y | (x,y) \in \mathcal{D} \wedge x \succeq x_i\} \end{cases} \quad (1)$$

Se pueden distinguir dos variantes diferentes dependiendo de cómo se extraen los vecinos para una nueva instancia i. La variante *enRango* considera los ejemplos k más cercanos x_j con sus etiquetas de clase y_j en el rango $[y_{min}, y_{max}]$. La versión *fueraDeRango* extrae los vecinos, independientemente de su clases, y luego, se eliminan aquellos cuyas clases están fuera del rango r_i. Como en el k-NN estándar, la clase mayoritaria entre los vecinos de k se usa como la etiqueta predicha.

MkNN requiere conjuntos de datos monotónicos para funcionar correctamente. Por ello, se utiliza una técnica de re-etiquetado para transformar los datos de entrenamiento. Estas técnicas pretenden identificar las violaciones de la monotonicidad y hacer los mínimos cambios posibles con la mínima diferencia de clase para eliminarlas [4].

II-C. k-Vecinos más Cercanos Difuso

Los métodos difusos de los k-Vecinos más Cercanos [8] incluyen conceptos difusos en la clásica regla k-NN para aprender de conjuntos difusos y producir reglas de clasificación difusas. Para una nueva instancia x_i, el algoritmo original FkNN [7] extrae sus K vecinos más cercanos de la misma manera que el k-NN estándar. Sin embargo, sus pertenecías a cada clase l se calculan con la siguiente expresión:

$$u(x,l) = \frac{\sum_{j=1}^{K} u(x_j, l) * \frac{1}{||x - x_j||^{(m-1)}}}{\sum_{j=1}^{K} \frac{1}{||x - x_j||^{(m-1)}}} \quad (2)$$

Como se muestra en 2, la pertenencia $u(x_i, l) = u_{il}$ de la muestra x_i a la clase l se asigna con el producto de la pertenencia a la clase $u(x_j, l)$ de los vecinos x_j y la inversa de sus distancias a x_i. Este último sirve como un peso que sesga hacia las pertenencias de las muestras más cercanas. El parámetro m determina el grado de influencia de las distancias a los vecinos. La etiqueta final de clase para el ejemplo x_i será aquella l con el mayor grado de pertenencia u_{il}.

Partiendo de un conjunto de entrenamiento etiquetado, FkNN [7] lo transforma en un conjunto difuso con pertenencias de clase usando la regla del vecino más cercano. Para cada muestra de entrenamiento x, se extraen sus k vecinos más cercanos y, luego, sus pertenencias se calculan de acuerdo a 3. Esta transformación ha demostrado ser útil frente a muestras ruidosas, ya que sus pertenencias se repartirán por las clases circundantes perdiendo su influencia.

$$u(x,l) = \begin{cases} 0{,}51 + 0{,}49 * (nn_l/k) & \text{if} \quad l = C(x) \\ 0{,}49 * (nn_l/k) \end{cases} \quad (3)$$

III. k-Vecinos más Cercanos Difuso para clasificación monotónica

En esta sección, se explica en detalle nuestra propuesta de k-Vecinos más Cercanos Difuso para clasificación monotónica.

III-A. De probabilidades a etiqueta final de clase

Es esencial elegir el mecanismo ideal según la monotonicidad para obtener una clase final a partir de una función de probabilidad. La clase más probable es el método más

común. Sin embargo, esto podría ser inapropiado para escenarios con restricciones monótonas. Por ejemplo, sea $x \leq y$ y sus probabilidades $f_x = (0{,}2, 0{,}2, 0{,}4, 0{,}2, 0{,}0)$ y $f_y = (0{,}0, 0{,}4, 0{,}3, 0{,}2, 0{,}1)$, entonces sus clases finales rompen la monotonía: $argmax(f_x) = l_3 > l_2 = argmax(f_y)$. Aunque, la instancia y tenga mayor peso en etiquetas más altas que x. Es más, f_y domina débilmente a f_x según el primer grado dominancia estocástica (FSD) [10], ya que la función de distribución acumulativa F_x de x, es mayor, elemento por elemento, que F_y, es decir, $f_x \preceq_{FSD} f_y \iff (\forall l \in \mathcal{Y})(F_x(l) \geq F_y(l))$. La FSD ha sido de utilidad para definir las restricciones de monotonicidad en las clasificaciones estocásticas [5], [11], con la expresión $x \leq y \implies f_x \preceq_{FSD} f_y$.

La mediana satisface dicha expresión, preservando la monotonicidad [5], [10]. Esta se calcula siguiendo la definición de percentil 50 como el rango$[l_m, l_M]$:

$$\begin{aligned} l_m &= \text{mín}\{l \in \mathcal{Y} | Prob\{X \leq l\} \geq 1/2\} \\ l_M &= \text{máx}\{l \in \mathcal{Y} | Prob\{X \geq l\} \geq 1/2\} \end{aligned} \tag{4}$$

Volviendo al ejemplo anterior, las clases para x y y elegidas por la mediana no rompen la monotonicidad: $med(f_x) = med(f_y) = 3$.

III-B. Pertenencias de clase robustas al ruido monotónico

En esta subsección, se explica el cálculo de pertenencias de clase rediseñado para reducir la influencia de las instancias no-monotónicas. En primer lugar, se ha de tratar las violaciones monótonas más simples, es decir, instancias repetidas con diferentes clases. MonFkNN primero sustituye las réplicas de cualquier ejemplo x por solo vector de característica x y sus pertenencias $u(x)$. La probabilidad $u(x, l)$ de la instancia x y la clase l se calcula con frecuencia de ejemplos duplicados pertenecientes a l, como se muestra en la siguiente expresión:

$$u(x, l) = \frac{|\{z \in \mathcal{D} | z = x \wedge C(z) = l\}|}{|\{z \in \mathcal{D} | z = x\}|} \tag{5}$$

Posteriormente, MonFkNN estima las pertenencias del resto de instancias con la información de los vecinos más cercanos a cada una. Sin embargo, estos vecinos son extraídos con un MkNN configurado como *enRango* en lugar de uno tradicional. Una vez obtenidos los vecinos de cada ejemplo x, la información de clase se fusiona en pertenencias de x. La probabilidad $u(x, l)$ de pertenecer a la clase l se calcula con la siguiente expresión:

$$u(x, l) = \begin{cases} \textit{RCr} + (nn_l/k) * (1 - \textit{RCr}) & \text{if} \quad l = C(x) \\ (nn_l/k) * (1 - \textit{RCr}) \end{cases} \tag{6}$$

donde nn_l es el número de vecinos de la clase l, k el número total de vecinos y $C(x)$ es la etiqueta original del ejemplo x. *RCr* es un nuevo parámetro llamado *Relevancia de la clase real"*. *RCr* puede ser visto como la probabilidad mínima asignada a la clase original $C(x)$ de la instancia x, en caso de que no haya vecinos etiquetados con $C(x)$.

Hay algunos valores para *RCr* en el rango $[0, 1]$ que tienen comportamientos muy interesantes y distintos. En el caso de un conjunto de datos realmente ruidoso, *RCr* podría establecerse en 0, dejando toda la responsabilidad al cálculo del rango r_i de las clases válidas y los vecinos más cercanos. En presencia de solo instancias repetidas, el usuario puede optar por tratarlas con *RCr* $= 1$. Finalmente, si se desea considerar las instancias etiquetadas originalmente, se recomienda asignar *RCr* a $0{,}5$. Este valor asegura que la clase real esté dentro del conjunto de medianas. En contraste con FkNN y sus $0{,}51$, si todos los vecinos pertenecen a la misma clase diferente a la actual, nuestro método obliga a elegir una clase intermedia. Por lo general, este último valor ofrece un comportamiento balanceado, estable y con un mejor rendimiento.

III-C. Agregación flexible de las pertenencias

Después de estimar la pertenencia de clase de cada instancia de entrenamiento, nuestro algoritmo está listo para predecir nuevos ejemplos. Esta fase de predicción ha sido diseñada con una mayor flexibilidad, permitiendo al usuario elegir entre predicciones más precisas o puramente monótonas.

Para ello, nuestro método utiliza otro MkNN para obtener los vecinos a utilizar en la predicción final. Este MkNN tiene también dos versiones, *enRango* y *fueraDeRange*. Pero son sustancialmente diferentes en comparación con las variantes originales, especialmente para la variante *fueraDeRange*.

La alternativa *enRango* se basa en la misma idea del MkNN original, donde los vecinos de un ejemplo deben pertenecer a un conjunto de clases monotónicamente válidas. Sin embargo, este rango de clases se obtiene utilizando las medianas adquiridas de las pertenencias de clase de las instancias de entrenamiento. Este avance mejora nuestro método hacia la robustez del ruido monotónico. La primera fase de MkNN se centra en la preservación de la monotonía de las medianas resultantes, para servir como una ventaja para las predicciones finales.

La versión *fueraDeRango* de nuestro método es completamente diferente de la anterior regla *fueraDeRango*. Ha sido diseñada con la intención de priorizar la precisión del clasificador sobre la monotonicidad. Con este propósito, nuestro método considera cualquier ejemplo como un vecino válido sin importar su clase. A diferencia del modelo original, no se realiza ningún filtrado o eliminación de vecinos fuera del rango válido. Sin embargo, su relevancia en la agregación de pertenencias se puede reducir, gracias a un factor de penalización introducido en la expresión de agregación.

Luego, para un nuevo ejemplo x, se obtienen sus vecinos más cercanos según la variante elegida. Sus pertenencias se agregan con la misma fórmula utilizada por el FkNN original con la adición del factor de penalización para la versión *fueraDeRange*. La siguiente expresión muestra cómo se integra este parámetro:

$$u(x,l) = \frac{\sum_{j=1}^{K} u(x_j,l) * \frac{pOR_j}{||x-x_j||^{(m-1)}}}{\sum_{j=1}^{K} \frac{pOR_j}{||x-x_j||^{(m-1)}}} \quad (7)$$

Como anteriormente, la pertenencia $u(x_i,l)$ de la nueva muestra x_i a la clase l es el resultado de la suma de las pertenencias a dicha clase $u(x_j,l)$ de los vecinos x_j ponderadas inversamente con la distancia a x_i. En la versión *fueraDeRange* de nuestro método, hay otro factor de ponderación en la contribución a las probabilidades finales, el parámetro mencionado como *"penalización de fueraDeRango"* (*pOR*). El factor pOR_j sólo se aplica si la clase y_j del vecino x_j no se encuentra en el rango de clases válidas r_i de x_i. Se puede configurar con valores continuos de 0 a 1. Cuando se asigna a 1, no se aplica ninguna penalización. El valor 0 significa una penalización completa, es decir, los vecinos con clases inválidas no participarán en la agregación. Este último comportamiento es equivalente a la variante *fueraDeRango* del MkNN original. Se recomienda el uso de 0.5, ya que es un buen equilibrio entre reducir su relevancia y considerarlos en la decisión.

Finalmente, la predicción de clase del nuevo ejemplo x es la mediana de las pertenencias de clase normalizadas resultantes.

MonFkNN se ha desarrollado para ser robusto al ruido monotónico y versátil en múltiples escenarios gracias a sus parámetros. Entre sus posibilidades, destacan dos configuraciones con un comportamiento muy distintivo: Monotónico Puro (PM) y Monotónico Aproximado (AM). La configuración monotónica pura corresponde a un valor de 0.5 para el parámetro *RCr* y el uso de la regla *enRango* para obtener las pertenencias de nuevas instancias. Este enfoque pretende dar predicciones con las mínimas violaciones de la monotonicidad.

La configuración monotónica aproximada prioriza la capacidad de predicción y relaja las restricciones monótonas. Las pertenencias del entrenamiento se obtienen mediante el tratamiento de las muestras repetidas. Los ejemplos únicos tendrán una probabilidad de 1 para la clase real y 0 para el resto. Este comportamiento se consigue con $RCr = 1$. Ya que buscamos predicciones más precisas, se consideran todas las instancias como vecinos válidos. Sin embargo, aquellas con etiquetas de clase inválidas contribuirán con la mitad de sus probabilidades ($pOR = 0{,}5$).

IV. Metodología experimental

Esta sección introduce el marco experimental utilizado en los diferentes estudios empíricos del trabajo. En dichos experimentos, se han incluido 12 conjuntos de datos diferentes. Estos son retratados en la Tabla I, donde se detallan el número de instancias, atributos y clases de cada conjunto en las columnas Ins., At. y Cl., respectivamente. La columna Direcciones indica la dirección de la relación monotónica entre cada atributo y la clase: monotonía directa (+) o inversa (-).

Se ha llevado a cabo un esquema de validación cruzada de 10 particiones (10-fcv) para ejecutar los diferentes clasificadores sobre estos conjuntos. Sus particiones han sido extraídas del repositorio KEEL [12].

Tabla I: Descripción de los 12 conjunto de datos usados.

Conjuntos	Ins.	At.	Cl.	Direcciones
artiset	1000	2	10	directas
balance	625	4	3	{-, -, +, +}
bostonhousing4cl	506	13	4	{-, +, -, +, -, +, -, +, -, -, -, +, -}
car	1728	6	4	directas
ERA	1000	4	9	directas
ESL	488	4	9	directas
LEV	1000	4	5	directas
machineCPU	209	6	4	{-, +, +, +, +, +}
qualitative_bankruptcy	250	6	2	directas
SWD	1000	10	4	directas
windsorhousing	546	11	2	directas
wisconsin	683	9	2	directas

La tabla II muestra los clasificadores involucrados en la comparación experimental y sus parámetros.

Tabla II: Parámetros considerados para los algoritmos a comparar.

Algoritmos	Parámetros
MkNN [4]	k = 5, distancia = euclidiana, tipoVecinos = enRango
OSDL [5]	equilibrado = No, tipoClasificación = mediana, límiteInferior = 0, límiteSuperior = 1 ajusteInterpolación = No, ponderado = No, pasoInterpolación = 10, gradoInterpolación = 0.5
OLM [6]	resolución = conservativa tipoClasificación = conservativa
MID [3]	R = 1, confianza = 0.25, elementosPorHoja = 2
Mon Fuzzy$k$NN	k = 5, K = 9, distancia = euclidiana
Monotónico Puro	RCr = 0.5, tipoVecinos = enRango
Monotónico Aprox.	$RCr = 1$, tipoVecinos = fueraDeRango, $pOR = 0{,}5$

Para evaluar la competencia de los clasificadores, se han empleado tres medidas que cubren diferentes aspectos de su rendimiento: capacidad predictiva con la precisión estándar, coste de los errores con el error absoluto medio (MAE) y monotonicidad con el Índice No Monotónico (NMI). El NMI mide la proporción de pares de muestras que rompen la monotonicidad respecto al total de pares.

V. Resultados y análisis

Esta sección presenta los resultados de diferentes estudios empíricos y sus análisis.

V-A. Evaluación de las propuestas de k-Vecinos más Cercanos Difuso Monotónico

A continuación, una comparación entre las versiones Pura y Aproximada de MonFkNN, destacando los diferentes comportamientos y aspectos de su rendimiento. La Tabla III contienen los resultados de las dos configuraciones de la propuesta en términos de Precisión, MAE y NMI. La fuente en negrita indica los mejores resultados obtenidos para cada conjunto de datos y métrica.

En la Tabla III, las diferencias entre ambos enfoques quedan claramente patentes. Tal como fueron diseñados, MonFkNN Aproximado (AM) tiene una mejor precisión en promedio en

Tabla III: Resultados de las versiones Puramente y Cuasi Monotónicas de MonFkNN.

	Precisión		MAE		NMI	
	PM ($RCr = 0,5$)	**AM** ($pOR = 0,5$)	**PM** ($RCr = 0,5$)	**AM** ($pOR = 0,5$)	**PM** ($RCr = 0,5$)	**AM** ($pOR = 0,5$)
artiset	0,9309	**0,9349**	0,0691	**0,0651**	**0,0000**	**0,0000**
balance	**0,9307**	0,9008	**0,0853**	0,1168	**0,0000**	0,0001
bostonhousing4cl	0,6561	**0,7134**	0,3972	**0,3261**	**0,0000**	0,0001
car	0,9740	**0,9834**	0,0295	**0,0195**	**0,0000**	**0,0000**
ERA	0,2420	**0,2430**	**1,2813**	1,2993	**0,0052**	**0,0052**
ESL	0,7036	**0,7131**	0,3149	**0,3053**	0,0004	**0,0003**
LEV	**0,6377**	0,6110	**0,3927**	0,4223	**0,0004**	0,0009
machineCPU	**0,7033**	0,6699	**0,3158**	0,3493	**0,0002**	0,0017
qualitative_bankruptcy	**0,9960**	**0,9960**	**0,0040**	**0,0040**	**0,0000**	**0,0000**
SWD	0,5807	**0,5833**	**0,4370**	0,4380	0,0007	**0,0003**
windsorhousing	0,7576	**0,7839**	0,2424	**0,2161**	**0,0005**	0,0051
wisconsin	0,9653	**0,9663**	0,0347	**0,0337**	**0,0000**	**0,0000**
Media:	0,7565	**0,7583**	0,3003	**0,2996**	**0,0006**	0,0012

el 75 % de los conjuntos utilizados. Por otro lado, MonFkNN Puro (PM) logra predicciones monotónicamente más confiables en 10 de los 12 conjuntos de datos utilizados, con grandes diferencias en los problemas de *Windsorhousing* y *MachineCPU*. Ambos tienen un resultado estable y bueno en términos de MAE, siendo la versión AM un poco mejor.

Ya que la monotonicidad se priorizada la mayoría de las veces en la clasificación con restricciones monotónicas, usaremos la variante Pura de MonFkNN en los siguientes estudios empíricos.

V-B. Comparación con el Estado-del-Arte

A lo largo de este experimento, se evalúa el rendimiento de MonFkNN en comparación a los clasificadores monotónicos del Estado-del-Arte. En esta, se busca un equilibrio entre predicciones precisas y monotónicas.

La Tabla IV recoge los resultados de precisión para los diferentes conjuntos de datos obtenidos por los algoritmos probados. En términos de precisión, MonFkNN rinde bastante mejor con un amplio margen. Nuestro método logra predicciones más precisas para 7 conjuntos de datos, con casos particularmente notables, como *balance* o *ESL*.

Tabla IV: Resultados en precisión obtenidos por los algoritmos evaluados.

	MonFkNN-PM	**MkNN**	**OSDL**	**OLM**	**MID**
artiset	**0,9309**	0,9199	0,1952	0,7948	0,7237
balance	**0,9307**	0,8624	0,6352	0,8320	0,7808
bostonhousing4cl	0,6561	0,6126	0,2787	0,5277	**0,6739**
car	**0,9740**	0,9711	0,9549	0,9543	0,8027
ERA	0,2420	0,1990	0,2320	0,1690	**0,2760**
ESL	**0,7036**	0,6332	0,6721	0,5738	0,6414
LEV	0,6377	0,4630	**0,6400**	0,4250	0,6070
machineCPU	**0,7033**	0,6890	0,2919	0,6746	0,6220
qualitative_bankruptcy	**0,9960**	**0,9960**	0,9160	0,9800	0,9840
SWD	0,5807	0,5200	**0,5840**	0,4160	0,5540
windsorhousing	0,7576	0,5861	0,4927	0,7564	**0,8205**
wisconsin	**0,9653**	0,9649	0,9590	0,8873	0,9517
Media:	**0,7565**	0,7014	0,5710	0,6659	0,7031

En la clasificación con restricciones monotónicas, el coste de los errores puede ser esencial. La Tabla V muestra el error en forma de MAE cometido por cada clasificador. Al igual que el rendimiento de precisión, MonFkNN es claramente mejor que el resto con el error más pequeño en promedio y para 8 de los conjuntos de datos. En aquellos problemas donde otros son superiores, como *LEV* o *wisconsin*, este consigue resultados realmente cercanos.

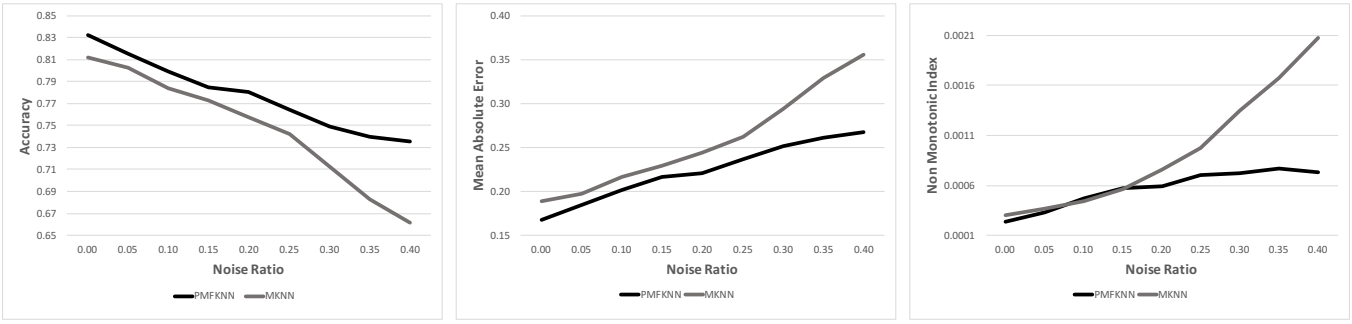
Tabla V: Resultados en MAE obtenidos por los algoritmos evaluados.

	MonFkNN-PM	**MkNN**	**OSDL**	**OLM**	**MID**
artiset	**0,0691**	0,0771	1,6897	0,2082	0,3123
balance	**0,0853**	0,1504	0,4912	0,1920	0,3360
bostonhousing4cl	0,3972	0,4901	0,9368	0,5988	**0,3893**
car	**0,0295**	0,0359	0,0475	0,0538	0,2506
ERA	**1,2813**	1,4270	1,2850	2,1500	1,2970
ESL	**0,3149**	0,3791	0,3607	0,4734	0,3934
LEV	0,3927	0,5740	**0,3920**	0,6680	0,4290
machineCPU	**0,3158**	0,3301	0,9043	0,3589	0,4211
qualitative_bankruptcy	**0,0040**	**0,0040**	0,0840	0,0200	0,0160
SWD	**0,4370**	0,4840	**0,4370**	0,7630	0,4750
windsorhousing	0,2424	0,4304	0,5073	0,2436	**0,1795**
wisconsin	0,0347	**0,0337**	0,0410	0,1127	0,0483
Media:	**0,3003**	0,3680	0,5980	0,4869	0,3790

En relación a la monotonicidad, la Tabla VI presenta los resultados de los NMI alcanzados. En este caso, la competencia por ser el mejor está mucho más reñida. OLM y MID obtienen predicciones claramente menos monótonas. Este último tiene el peor comportamiento considerando sólo la monotonicidad. MonFkNN, MkNN y OSDL funcionan de manera similar. MonFkNN y OSDL son ligeramente mejores en promedio.

Tabla VI: Resultados en NMI obtenidos por los algoritmos evaluados.

	MonFkNN-PM	**MkNN**	**OSDL**	**OLM**	**MID**
artiset	**0,0000**	**0,0000**	**0,0000**	**0,0000**	0,0039
balance	**0,0000**	0,0001	0,0006	**0,0000**	0,0017
bostonhousing4cl	**0,0000**	**0,0000**	**0,0000**	0,0003	0,0022
car	**0,0000**	**0,0000**	**0,0000**	**0,0000**	0,0046
ERA	0,0052	0,0056	**0,0049**	0,0063	0,0082
ESL	**0,0004**	0,0012	0,0006	0,0025	0,0021
LEV	**0,0004**	0,0010	**0,0004**	0,0043	0,0018
machineCPU	0,0002	**0,0000**	**0,0000**	0,0014	0,0037
qualitative_bankruptcy	**0,0000**	**0,0000**	0,0003	**0,0000**	0,0002
SWD	0,0007	**0,0005**	0,0009	0,0015	0,0020
windsorhousing	0,0005	**0,0000**	**0,0000**	**0,0000**	0,0030
wisconsin	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Media:	**0,0006**	0,0007	**0,0006**	0,0014	0,0028



(a) Efecto del ruido en la Precisión. (b) Efecto del ruido en el MAE. (c) Efecto del ruido en el NMI.

Figura 1: Rendimiento de Puro MonFkNN y MkNN con los diferentes conjuntos ruidos de *Artiset*.

V-C. Sobre la robustez al ruido monotónico de k-Vecinos más Cercanos Difuso Monotónico

Con este último estudio empírico, pretendemos probar la robustez de MonFkNN frente a la presencia de violaciones monótonas o ruido en los sets de entrenamiento en contraste con MkNN. Para ello, se ha introducido diferentes cantidades de instancias ruidosas en las particiones de entrenamiento del conjunto de datos artificial *Artiset*. Luego, se han comparado el comportamiento de MonFkNN y MkNN en términos de precisión, MAE y NMI, mientras que el ratio de ruido aumenta.

La figura 1 muestra el comportamiento de MonFkNN y MkNN (líneas más oscuras y más claras, respectivamente) en base a la precisión (1a), MAE (1b) y NMI (1c), con la progresión del ruido. Como era de esperar, mientras la cantidad de ruido aumenta, el rendimiento de ambos métodos empeora. Sin embargo, hay alguna gran diferencia entre ambos clasificadores.

En primer lugar, el comportamiento de cara al ruido de MonFkNN es claramente mejor que el del MkNN en todos los aspectos probados. Las líneas negras siempre están sobre las más claras de la figura 1a, lo que indica siempre una precisión mayor, y debajo de ellas en las figuras 1b y 1c, lo que significa mejores MAE y NMI para MonFkNN. Por lo general, la distancia entre ambos métodos es amplia, con excepción de los resultados de NMI obtenidos para los valores más pequeños de ruido. Además, mientras que el índice de ruido aumenta, sus diferencias también aumentan.

Con estos experimentos, MonFk-NN ha demostrado una fuerte robustez al ruido monotónico preservando su buen rendimiento en términos de precisión, costes de error y monotonicidad.

VI. Conclusiones

En este trabajo, se ha propuesto un modelo difuso de k-Vecinos más Cercanos para la clasificación con restricciones monotónicas. La transferencia de una función de probabilidades a una clase final se ha adaptado para que respete dichas restricciones. Se ha diseñado mecanismos para reducir la influencia de las violaciones monotónicas. Como demostración de su flexibilidad, se han presentados dos configuraciones diferentes del modelo con comportamientos muy distintos.

Durante el análisis experimental, se ha mostrado el gran potencial en relación a la monotonicidad de la configuración puramente monotónica y la buena precisión de la versión aproximada de MonFkNN. En comparación con otros métodos, MonFkNN es significativamente mejor en términos de precisión y coste de error, igualando los mejores resultados de NMI. Además, ha demostrado una fuerte robustez a grandes cantidades de ruido preservando su buen rendimiento.

Referencias

[1] C.-C. Chen and S.-T. Li, "Credit rating with a monotonicity-constrained support vector machine model," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7235–7247, 2014.

[2] J.-R. Cano, N. R. Aljohani, R. A. Abbasi, J. S. Alowidbi, and S. Garcia, "Prototype selection to improve monotonic nearest neighbor," *Engineering Applications of Artificial Intelligence*, vol. 60, pp. 128–135, 2017.

[3] A. Ben-David, "Monotonicity maintenance in information-theoretic machine learning algorithms," *Machine Learning*, vol. 19, no. 1, pp. 29–43, 1995.

[4] W. Duivesteijn and A. Feelders, "Nearest neighbour classification with monotonicity constraints," in *ECML/PKDD (1)*, 2008, pp. 301–316.

[5] S. Lievens, B. De Baets, and K. Cao-Van, "A probabilistic framework for the design of instance-based supervised ranking algorithms in an ordinal setting," *Annals of Operations Research*, vol. 163, no. 1, pp. 115–142, 2008.

[6] A. Ben-David, "Automatic generation of symbolic multiattribute ordinal knowledge-based dsss: methodology and applications," *Decision Sciences*, vol. 23, pp. 1357–1372, 1992.

[7] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[8] J. Derrac, S. García, and F. Herrera, "Fuzzy nearest neighbor algorithms: Taxonomy, experimental analysis and prospects," *Information Sciences*, vol. 260, pp. 98–119, 2014.

[9] W. Kotłowski and R. Słowiński, "On nonparametric ordinal classification with monotonicity constraints," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 11, pp. 2576–2589, 2013.

[10] H. Levy, *Stochastic dominance: Investment decision making under uncertainty*. Springer, 2015.

[11] S. Lievens and B. De Baets, "Supervised ranking in the weka environment," *Information Sciences*, vol. 180, no. 24, pp. 4763–4771, 2010.

[12] I. Triguero, S. González, J. M. Moyano, S. García, J. Alcalá-Fdez, J. Luengo, A. Fernández, M. J. del Jesús, L. Sánchez, and F. Herrera, "Keel 3.0: an open source software for multi-stage analysis in data mining," *International Journal of Computational Intelligence Systems*, vol. 10, no. 1, pp. 1238–1249, 2017.

Eventos raros, anomalías y novedades vistas desde el paraguas de la clasificación supervisada

Ander Carreño
Intelligent Systems Group (ISG)
UPV/EHU
P. Manuel Lardizabal 1
20018 San Sebastián (Spain)
Email: ander.carreno@ehu.eus

Iñaki Inza
Intelligent Systems Group (ISG)
UPV/EHU
P. Manuel Lardizabal 1
20018 San Sebastián (Spain)
Email: inaki.inza@ehu.eus

Jose A. Lozano
Intelligent Systems Group (ISG)
UPV/EHU, 28018 San Sebastián (Spain)
Baque Center For Applied Mathematics
BCAM, 48009 Bilbao (Spain)
Email: ja.lozano@ehu.eus

***Resumen*—En los últimos años, muchas áreas han contribuido a resolver problemas denominados: eventos raros (*rare events*), anomalías (*anomalies*), novedades (*novelties*) y datos atípicos (*outliers*). Estas áreas han creado una mezcla entre términos y problemas. Por un lado, problemas similares han sido descritos con diferentes términos. Por el otro, el mismo término ha sido utilizado para describir problemas diferentes. Dada esta confusión, este artículo tiene como objetivo poner orden en el área viendo todos estos problemas desde el prisma de la clasificación supervisada. En concreto, se asocia a cada término un escenario de clasificación, se muestran ejemplos, y se consideran aquellas características asociadas a cada término que comparten la mayoría de artículos de la literatura. Por lo tanto, los escenarios se formalizan desde el punto de vista de la clasificación supervisada, se revisan los objetivos de la tarea de clasificación, las métricas usadas para evaluar el rendimiento, las características de los datos de entrada y las técnicas utilizadas en cada uno de ellos.**

***Index Terms*—Eventos raros, anomalías, novedades, datos atípicos, outliers, aprendizaje automático, clasificación supervisada, aprendizaje desbalanceado.**

I. Introducción

Una gran cantidad de aplicaciones requieren filtrar o detectar observaciones anormales en los datos. En algunas situaciones, las anormalidades son descritas como eventos raros, anomalías, novedades, outliers, excepciones, aberraciones, sorpresas, peculiaridades, ruido o contaminantes entre otras. De todas estas acepciones, las más comunes son: eventos raros, anomalías, novedades y outliers. La importancia de detectar estas anormalidades es debido a que estas pueden estar relacionadas con situaciones críticas o interesantes en una gran cantidad de dominios. Por ejemplo, en seguridad, las intrusiones son consideradas anormalidades [1]–[3]; en seguridad vial, los accidentes de tráfico son considerados anormalidades [4]; en geología la erupción de un volcán es una anormalidad [5]; en el control de alimentos, cuerpos extraños dentro de envoltorios son también anormalidades [6]; o en el caso de la neurociencia, un estímulo que no se ha experimentado anteriormente es también una anormalidad [7].

Considerando la importancia de las anormalidades, se ha desarrollado una larga lista de trabajos relacionados con estos términos. Sin embargo, se ha formado una mezcla entre los términos y los problemas. En primer lugar, se han utilizado diferentes términos para describir el mismo problema. Por ejemplo, en [8], los autores tratan un problema en el que predicen si se va a producir un desprendimiento en una zona concreta y en un periodo de tiempo acotado. Para ello, crean un mapa de zonas susceptibles a que ocurra un desprendimiento. Cada zona tiene asociado un riesgo de desprendimiento. Este riesgo se predice con un modelo aprendido con datos sobre zonas donde ha ocurrido un desprendimiento y zonas en las que no. Es decir, el modelo se aprende con datos etiquetados en dos categorías: desprendimiento y no desprendimiento. Estos desprendimientos de tierra son descritos como *eventos raros*, denotando que son eventos que ocurren pocas veces. No obstante, en [1], se trata un problema similar desde el punto de vista de la clasificación supervisada, pero que se describe con otro término. En éste, se realiza un estudio en la industria ferroviaria. Las puertas de pasajeros cuentan con diferentes sistemas para abrir y cerrar las puertas de los trenes, que aseguran el confort y la seguridad de los pasajeros abordo. En algunas situaciones, por el deterioro de estos sistemas, el sistema de apertura y cierre de las puertas falla. Por lo tanto, la tarea de predicción se enfoca en predecir si la puerta va a fallar en un periodo de tiempo acotado. Para ello, se aprende un modelo con datos etiquetados en dos categorías: fallos (puertas que fallan) y situaciones normales. En este estudio, los errores en las puertas se denominan *anomalías*, denotando también situaciones que ocurren pocas veces. Como se puede ver, los dos problemas son prácticamente idénticos desde el punto de vista de la clasificación supervisada, pero se han utilizado diferentes términos para describir las anormalidades. Por lo tanto, analizando los problemas desde el punto de vista de la clasificación supervisada, ambos problemas se pueden modelar como problemas de clasificación binarios con distribuciones de probabilidad desbalanceadas en las clases.

Por otro lado, también el mismo término ha sido utilizado para describir problemas diferentes. En los siguientes dos problemas, los autores utilizan el término *novedad*, *novelty* en inglés, para describir la anormalidades. En [3], detectan si un paciente está sufriendo un ataque epiléptico. Dada la fuerza de las convulsiones que sufren estos pacientes durante un ataque, pueden hacerse daño a ellos mismos. En consecuencia, detectar estos ataques lo más rápido posible es muy importante

para evitar los daños. Por lo tanto, el objetivo de este estudio es detectar si un paciente está sufriendo un ataque epiléptico. Para ello, se aprende un modelo predictivo con datos de movimientos de pacientes recogidos con un acelerómetro 3D. Estos datos previamente han sido grabados durante varios días en los cuales, en algún momento, los pacientes han sufrido una o varias convulsiones. Después, estos datos son segmentados en ventanas de tiempo fijas y, estas particiones son etiquetadas como ataque (porque ha sufrido un ataque en este tramo temporal) o como normal. Por lo tanto, cuando llega un nuevo caso, es decir, al recibir los datos de un paciente que está constantemente monitorizado, el clasificador detecta si está o no sufriendo un ataque epiléptico (i.e. identificado como novedad). Sin embargo en [6], se detectan cuerpos extraños dentro de contenedores de alimentos. En algunas ocasiones, ciertos cuerpos extraños como insectos, piedras o plásticos son encontrados dentro de los envoltorios de alimentos, tales como bandejas o tetrabriks. Detectar estos cuerpos extraños es muy importante para la imagen de las empresas productoras de este tipo de productos y para los consumidores finales. Por lo tanto, partiendo únicamente de imágenes de rayos X de alimentos limpios de cuerpos extraños, es decir, desde nuestro prisma, de una única clase, se aprende un modelo que es capaz de detectar estos cuerpos extraños, que son descritos como novedades en este estudio. Como se puede apreciar en ambos ejemplos, el mismo término (novedad) ha sido utilizado para describir las anormalidades. No obstante, ambos trabajos no se basan en el mismo escenario de aprendizaje, y en [3] el modelo predictivo es aprendido con tanto instancias normales como anormales mientras que en [6] únicamente se aprende el modelo con datos normales (sin cuerpos extraños).

Sin embargo, no sólo los problemas descritos anteriormente, sino la mayoría de los problemas reseñados con los términos eventos raros, anomalías y novedades, se pueden formalizar como problemas de clasificación supervisada. Por lo tanto, este artículo trata de poner orden en esta mezcla y confusión de términos y problemas, asignando un único escenario de aprendizaje a cada uno de los términos, que viene a ser el más compartido en la literatura. Después, describe cada uno de los escenarios de clasificación mediante la asignación de aquellas características que más comparten los problemas de la literatura asociados a cada término.

El artículo se organiza de la siguiente manera. Cada sección corresponde a un escenario de clasificación, siendo: la Sección II para la detección de eventos raros, la Sección III para la detección de anomalías y la Sección IV para las novedades. En la Sección V se discute sobre el término outlier y, finalmente, en la Sección VI se exponen las conclusiones del artículo.

II. Detección de Eventos Raros

La mayoría de los artículos que utilizan el término evento raro para describir las anormalidades tienen la dimensión temporal como una característica clave. Por ejemplo en [4], se trata un problema relacionado con la seguridad vial en la autopista Attica (Grecia). Los autores dividen la autopista en diferentes secciones en las que quieren detectar si ha ocurrido un accidente en la última hora. Para ello, crean un modelo basándose en información histórica tanto de sensores terrestres como de cámaras de tráfico en el que conocen si ha ocurrido un accidente en las diferentes secciones y en intervalos de 1 hora. Por lo tanto, llegada una nueva información de 1 hora en una sección, el clasificador predice la existencia de un accidente. Otra aplicación relacionada con el término evento raro, es [5]. En este artículo se realiza un estudio sobre la erupción de dos volcanes de Sudamérica. Los autores quieren detectar si estos volcanes van a entrar en erupción en una ventana de tiempo acotada. Para ello, aprenden un modelo con datos históricos de actividad volcánica en el que eventualmente hay erupciones. Esta estimación la realizan mediante un proceso de Poisson en el que se predice si va a haber o no una erupción en un horizonte acotado. Otro ejemplo diferente relacionado es [9]. En este artículo se realiza un estudio sobre la probabilidad de fallo en cascada de una red eléctrica. Para ello, los autores realizan un estudio por simulación numérica en el que someten a la red a diferentes situaciones climatologícas extremas de manera intencionada. Como puede observarse, la dimensión temporal de todos estos problemas es clave. Sin embargo, se observan dos objetivos diferentes en todos estos trabajos. Por un lado, la estimación de la probabilidad de ocurrencia de un evento raro. Por otro, la detección de eventos raros en una ventana de tiempo fija.

En el caso de la estimación de la probabilidad de ocurrencia de un evento raro, la probabilidad es estimada por simulación. Este tema se ha tratado especialmente en la física y la ingeniería. Ejemplos de aplicaciones de estimación de la probabilidad de un evento raro son: la estimación de fallo de una infraestructura [9], la probabilidad de fallo de sistemas técnicos [10], o la probabilidad de desarrollos climáticos extremos [11].

En la clasificación de eventos raros, las instancias son series temporales [12]. El objetivo es clasificar nuevas series temporales utilizando un modelo aprendido anteriormente. Sin embargo, dada la temporalidad de las instancias, existen dos aproximaciones de clasificación diferentes. En primer lugar, se encuentra la clasificación de eventos raros completos, es decir, de series temporales completas. Por ejemplo, en [13], se hace uso del conjunto de datos SMART [1] para predecir si el disco duro va a fallar en un periodo de tiempo acotado. Los datos consisten en una colección de medidas tomadas a lo largo del tiempo por sensores del disco duro cuando este está en uso. En [14], se hace uso de un conjunto de datos de tecnología térmica en el que se ha almacenado información sobre sistemas de calefacción. El objetivo es predecir si el sistema de calefacción está fallando o no. En segundo lugar, la clasificación temprana de eventos raros se ha tratado en la literatura [15] (*early classification*). El objetivo es clasificar las nuevas observaciones lo antes posible, preferiblemente, antes de que se complete la serie temporal. Esta temprana clasificación es crítica en una variedad de aplicaciones. Por ejemplo, en

[1]El conjunto de datos SMART se compone de datos de uso de discos duros en los que se pueden ver errores como comportamiento normal.

[8] se realiza una predicción temprana de desprendimientos de tierra usando imágenes por satélite. O en [5], donde basándose en erupciones volcánicas del pasado, se predice si un volcán va a entrar en erupción o no en una ventana de tiempo fija.

De acuerdo a las distribuciones de probabilidad de los eventos raros ($\mathcal{A}$) y normales ($\mathcal{N}$), asumiendo que los datos son generados por el mecanismo generador $P(\mathbf{x}, c)$ [16], $P(C = \mathcal{A}) \ll P(C = \mathcal{N})$, los datos cuentan con una distribución desbalanceada de las clases. Por ello, la clasificación de eventos raros se puede formalizar bajo el marco de clasificación supervisada de series temporales con un (alto) desbalanceo de probabilidad sobre las clases [17], [18].

Sin embargo, en algunos problemas de clasificación de eventos raros, las instancias se transforman de tal forma que no se tiene en cuenta la información temporal en el proceso. Por esto, la clasificación de eventos raros se puede convertir en un problema de clasificación atemporal desbalanceado [4], [8].

Para evaluar el rendimiento de los clasificadores en la detección de eventos raros, métricas populares como el AUC [14], [19], [20] y el *recall* de la clase asociada al evento raro [14], [20] son utilizadas en la literatura.

II-A. *Características de los datos de entrada*

En la mayoría de los artículos referidos a la detección de eventos raros, los datos están etiquetados en dos categorías: normal ($\mathcal{N}$) y rara ($\mathcal{A}$). Las instancias son series temporales, una secuencia ordenada de pares (tiempo, valor) de longitud fija N; $TS = \{(t_i, x_i), i = 1, \ldots, N\}$ [12]. Tanto en el escenario de clasificación de eventos raros, como en el escenario de detección de anomalías, existe una distribución desbalanceada de las clases donde $P(C = \mathcal{A}) \ll P(C = \mathcal{N})$. En este escenario de clasificación, tanto las instancias normales como las anormales se encuentran en el conjunto de entrenamiento del modelo.

II-B. *Técnicas*

Entre las técnicas utilizadas en la literatura para estimar la probabilidad de ocurrencia de un evento raro se encuentran: *importance sampling*, simulaciones de Monte Carlo [21], [22], *kriging* [22] o *first order reliability methods (FORM)* [23]. De acuerdo con la clasificación de eventos raros, la técnica *rare event logistic regression*, una adaptación de la regresión logística, ha sido explícitamente propuesta en este escenario de clasificación [4], [8], [20], [24]. No obstante, técnicas populares de clasificación han sido aplicadas con éxito, como son la discriminación por divergencia de *Kullback-Leibler* [19], las redes neuronales de memoria a corto y largo plazo [14], los algoritmos genéticos [25], *naïve Bayes* para múltiples instancias [13], los procesos de Poisson [5], los *support vector data regression surrogates* [26], las redes Bayesianas [27], las máquinas de soporte vectorial [28] o el *AdaBoost* [29].

Por otro lado, teniendo en cuenta la distribución desbalanceada de las clases, se han encontrado técnicas en la literatura que tratan con este tipo de situaciones, como son *Structure Preserving Over Sampling (SPO)* [18] o una técnica de muestreo similar al SMOTE [17].

III. Detección de Anomalías

En el escenario de detección de anomalías, comúnmente los datos son atemporales y etiquetados. El objetivo es detectar anomalías basándose en un modelo aprendido con datos históricos. El conjunto de entrenamiento se compone de tanto instancias normales como anormales (anomalías). Nótese que se espera que las instancias anómalas nuevas (no vistas) sean parecidas a aquellas provistas en el conjunto de entrenamiento del modelo. Por lo tanto, este escenario puede formalizarse bajo el marco de la clasificación supervisada con un (alto) desbalanceo de las clases, asumiendo que los datos se generan por un mecanismo generador $P(\mathbf{x}, c)$ [16], $P(C = \mathcal{A}) \ll P(C = \mathcal{N})$. Esta distribución desbalanceada de los datos hace que los clasificadores estándar estén abrumados por la clase mayoritaria (anormal) e ignoren la minoritaria (normal) [30].

Para evaluar el rendimiento de los clasificadores en este escenario, dado el (alto) desbalanceo de las clases, métricas comunes como el *accuracy* no son lo suficiente representativas. Por ello, focalizándose en la correcta clasificación de las anomalías; o lo que es lo mismo, minimizando el error de tipo II, una métrica de evaluación popular utilizada en la literatura es el *recall* de la clase minoritaria [1], [31]. Sin embargo, otras métricas de evaluación pueden ser utilizadas como las que se revisan en [32].

III-A. *Características de los datos de entrada*

En la detección de anomalías, los datos son atemporales y, en la mayoría de las aplicaciones, están etiquetados en dos categorías: normal ($\mathcal{N}$) y anómalo ($\mathcal{A}$). En este escenario de clasificación existe una distribución desbalanceada de estas dos categorías, por lo que se tienen muchas más instancias de la clase normal que de la anormal. Sin embargo, en el conjunto de entrenamiento, se tienen tanto instancias de la clase normal como de la clase anómala.

III-B. *Técnicas*

Entre las técnicas especificas creadas bajo el entorno de detección de anomalías, se encuentra el popular *Isolation Forest* [33]. Por otro lado, se han adaptado técnicas de clasificación para resolver problemas en este ámbito, como son: las máquinas de soporte vectorial [34], los *random forests* [35], los clasificadores Bayesianos [36], las redes neuronales [37], o los modelos basados en mixturas de gausianas [38]. Nótese que dado que la detección de anomalías se puede formalizar como un escenario de clasificación supervisada con distribuciones de probabilidad (altamente) desbalanceadas, es posible utilizar técnicas específicas para el desbalanceo, como puede ser el *Synthetic Minority Oversampling Technique* (SMOTE) [39], [40].

IV. Detección de Novedades

En detección de novedades, comúnmente las instancias son atemporales y durante la etapa de entrenamiento, no se proporcionan instancias anormales [2], [6], [7]. El objetivo es clasificar si una nueva instancia pertenece al conjunto

normal provisto en entrenamiento o no, basándose en un modelo aprendido únicamente con instancias de una clase. Después, el modelo debe ser capaz de detectar cualquier grupo de instancias discordantes que es posible que formen una nueva clase [41]. Nótese que quizá sea necesaria la intervención de un experto para poder consolidar esta última anotación de una nueva clase. Por ejemplo, en este estudio de neurociencia [7], una novedad se considera un estímulo que no ha sido experimentado anteriormente. Este nuevo estímulo puede compartir características con otros nuevos estímulos no vistos en momento de entrenamiento, formando una clase o un nuevo tipo de estímulo. En el control alimenticio, los objetos extraños dentro de los envoltorios de los alimentos, como las piedras, los plásticos o insectos, son considerados novedades [6]. En este caso, es posible que las piedras de un alimento tengan características parecidas a otras observaciones no vistas en el conjunto de entrenamiento que contenga también una piedra.

En la detección de novedades, el conjunto de entrenamiento es generado únicamente por $P(\mathbf{X}|C = \mathcal{N})$. Consecuentemente, no se posee conocimiento alguno sobre las novedades cuando se aprende el primer modelo. Por lo tanto, en la creación del modelo, se debe asumir que las novedades estarán uniformemente distribuidas en el espacio de características. Esto es razonable dado que al no poseer ninguna información a priori, cualquier suposición es inconsistente. Sin embargo, una vez que se reconocen algunas novedades, estas pueden formar una clase nueva, por lo que el problema puede pasar a ser un problema de clasificación supervisada con distribuciones de probabilidad de clases desbalanceadas [41]. Para clarificar este último concepto, considérese un estudio médico en el que se proporcionan electrocardiogramas asociados a una enfermedad concreta como conjunto de entrenamiento. En la etapa de predicción, ciertos electrocardiogramas muestran una discordancia notable con aquellos de la etapa de entrenamiento. Por lo tanto, el clasificador los etiqueta como novedades. No obstante, se sabe que estos electrocardiogramas marcados como novedad, están relacionados con otra enfermedad. Por ello, es razonable pensar que estos electrocardiogramas comparten características entre ellos y que por tanto, crear una nueva clase para representar a los mismos sería una buena aproximación.

Por otro lado, también se ha tratado la detección de novedades en un entorno dinámico o *stream*. Por ejemplo en [41], partiendo de un conjunto de clasificación supervisada, los autores detectan nuevas clases combinando técnicas de clasificación supervisada y no supervisada. En concreto, si las instancias etiquetadas como novedades son suficientemente densas y distintas a las demás, estas son etiquetadas como una nueva clase.

Finalmente, para evaluar el rendimiento de estos clasificadores, es necesario señalar que la clasificación se centra principalmente en detectar las instancias normales proporcionadas en momento de entrenamiento [3], [33], [42]. Por lo tanto, las métricas de evaluación se basan en la minimización del error de tipo I. Una medida popular es la maximización del *recall* de la clase normal [3], [42].

IV-A. Características de los datos de entrada

En el escenario de detección de novedades, no se provee de instancias nóveles en el conjunto de entrenamiento. El modelo es aprendido únicamente con instancias de una clase. Se espera que en momento de predicción puedan llegar tanto instancias de una como de otra (novel) clase. Sin embargo, también se espera una distribución de probabilidad desbalanceada sobre las clases normal y novel $P(C = \mathcal{N}) \ll P(C \neq \mathcal{N})$ [7]. Además, se asume uniformidad sobre la distribución de las instancias novel.

IV-B. Técnicas

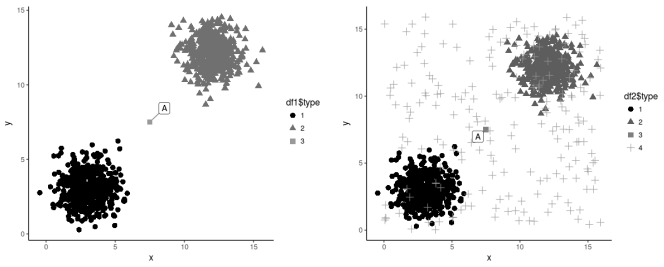
Las técnicas especialmente diseñadas para la detección de novedades tienen en cuenta que aprenden únicamente de una clase y son capaces de discernir si una nueva instancia forma parte o no de este comportamiento. Por lo tanto, las técnicas más representativas de este escenario de clasificación son aquellas relacionadas con el marco *one class*. Por ejemplo, *one class SVM* [28], [43], [44], *K-Nearest Neighbors data description* [45], *graph embeded one-class classifiers* [46] y *one-class random forests* [47] han sido propuestas en la literatura en escenarios de detección de novedades.

V. Discusión: Detección de Outliers

El término dato atípico, *outlier* en inglés tiene características comunes a los demás términos. Por ejemplo, las instancias pueden ser tanto temporales (series temporales) [48] como atemporales [49]–[51].

No obstante, aunque similar, el término outlier tiene diferencias clave con el resto de términos analizados. La diferencia más significativa es que los datos no están etiquetados, es decir, la tarea de predicción se realiza bajo el marco de la clasificación no supervisada [49]. Por lo tanto, en contraposición al resto de escenarios, no se asume que al menos existen dos clases en el problema de predicción. En ocasiones, el término outlier también se ha relacionado con el *ruido* sugiriendo que estas observaciones pueden ser inconsistentes o incorrectas [49]. Por ejemplo, cuando se cometen errores humanos al recoger los datos, estos comportamientos erráticos son considerados outliers [52]. En cambio, en otras situaciones, el término outlier se ha relacionado con instancias con alta varianza [51], [53]. Por ejemplo en [51], los autores detectan jugadores *all-star* de la NBA en un conjunto de datos constituido por jugadores de baloncesto entre los años 1973 y 2003. Los jugadores excepcionales son considerados outliers. Para detectarlos, los autores realizan un *clustering* y aquellos puntos que se desvían significativamente son considerados outliers.

Sin embargo, la detección de los outliers es subjetiva [49]. ¿Cuánto debe desviarse una observación del resto para considerarla como atípica? En la literatura existen múltiples criterios para responder a esta pregunta [51], [54], [55]. Para clarificarla, en la Figura 1 se describen dos situaciones para la detección de outliers. En 1a el punto marcado con A es considerado dato atípico dada la desviación significativa hacia uno y otro grupo. No obstante, cuando existe mucha más



(a) Sin ruido (b) Con ruido

Figura 1: Diferentes escenarios de outliers. Por [49].

Cuadro I: Resumen de las características de los diferentes escenarios de aprendizaje.

	Eventos raros	Anomalías	Novedades	Outliers
Siempre relacionado con dominios temporales	✓	✗	✗	✗
Distribución desbalanceada de las clases	✓	✓	✓	-
Observaciones anormales durante la etapa de entrenamiento	✓	✓	✗	-
Se realiza predicción sobre clases previamente vistas	✓	✓	✗	✗
Clasificación supervisada	✓	✓	✓	✗

dispersión en los datos, este concepto puede ser más difuso y complejo. Este comportamiento se puede ver en 1b. En 1b, dados los demás outliers alrededor del punto A, la densidad del punto A es mayor y puede que quede difuminado el criterio de separación, haciendo una detección más difícil.

VI. Conclusiones

Por un lado, los términos eventos raros, anomalías y novedades se han utilizado para describir problemas de clasificación similares en la literatura. Por el otro, problemas similares han sido descritos con diferentes términos. Además, estos términos se han utilizado indistintamente sin tener en cuenta las características que los diferencian. Por ello, en este artículo se han remarcado las características genuinas de cada uno de los términos. Para ello, se han extraído las características que más comparten los trabajos de la literatura referidos a cada término y se han revisado los objetivos, las características de los datos de entrada, y las técnicas utilizadas en los respectivos trabajos. Además, se ha argumentado que estos escenarios de aprendizaje pueden ser formalizados como problemas de clasificación supervisada con un (alto) desbalanceo entre las distribuciones de probabilidad de las clases. Consecuentemente, las técnicas específicas para tratar el desbalanceo pueden ser adaptadas y utilizadas en estos escenarios. En la Tabla I se muestra un resumen de las principales características de los diferentes escenarios de aprendizaje tratadas en este trabajo.

VII. Agradecimientos

Este trabajo ha sido parcialmente financiado por el Gobierno Vasco (IT609-13) y por el Ministerio de Economía, Industria y Competitividad (TIN2016-78365-R). Ander Carreño posee una beca del Ministerio de Economía, Industria y Competitividad (BES-2017-080016). J.A. Lozano posee financiación tanto del programa BERC 2018-2021 (Gobierno Vasco) como del programa Severo Ochoa SEV-2017-0718 (Ministerio de Economía, Industria y Competitividad).

Referencias

[1] R. P. Ribeiro, P. Pereira, and J. Gama, "Sequential anomalies: a study in the Railway Industry," *Machine Learning*, vol. 105, no. 1, pp. 127–153, oct 2016.

[2] M. A. F. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[3] S. Luca, D. A. Clifton, and B. Vanrumste, "One-class classification of point patterns of extremes," *Journal of Machine Learning Research*, vol. 17, pp. 1–21, 2016.

[4] A. Theofilatos, G. Yannis, P. Kopelias, and F. Papadimitriou, "Predicting Road Accidents: A Rare-events Modeling Approach," in *Transportation Research Procedia*, 2016.

[5] Y. Dzierma and H. Wehrmann, "Eruption time series statistically examined: Probabilities of future eruptions at Villarrica and Llaima Volcanoes, Southern Volcanic Zone, Chile," *Journal of Volcanology and Geothermal Research*, 2010.

[6] H. Einarsdóttir, M. J. Emerson, L. H. Clemmensen, K. Scherer, K. Willer, M. Bech, R. Larsen, B. K. Ersbøll, and F. Pfeiffer, "Novelty detection of foreign objects in food using multi-modal X-ray imaging," *Food Control*, vol. 67, pp. 39–47, sep 2016.

[7] A. Kafkas and D. Montaldi, "How do memory systems detect and respond to novelty?" *Neuroscience Letters*, feb 2018.

[8] M. Van Den Eeckhaut, T. Vanwalleghem, J. Poesen, G. Govers, G. Verstraeten, and L. Vandekerckhove, "Prediction of landslide susceptibility using rare events logistic regression: A case-study in the Flemish Ardennes (Belgium)," *Geomorphology*, vol. 76, no. 3-4, pp. 392–410, 2006.

[9] L. Dueñas-Osorio and S. M. Vemuru, "Cascading failures in complex infrastructure systems," *Structural Safety*, vol. 31, no. 2, pp. 157–167, mar 2009.

[10] T. Bedford and R. M. Cooke, *Probabilistic risk analysis: foundations and methods*. Cambridge University Press, 2001.

[11] S. Dessai and M. Hulme, "Does climate adaptation policy need probabilities?" *Climate Policy*, vol. 4, no. 2, pp. 107–128, jan 2004.

[12] P. Esling and C. Agon, "Time-series data mining," *ACM Computing Surveys*, vol. 45, no. 1, pp. 1–34, nov 2012.

[13] J. F. Murray, G. F. Hughes, and K. Kreutz-Delgado, "Machine Learning Methods for Predicting Failures in Hard Drives: A Multiple-Instance Application," *J. Mach. Learn. Res.*, 2005.

[14] S. Zhang, S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Deep learning on symbolic representations for large-scale heterogeneous time-series event prediction," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, mar 2017, pp. 5970–5974.

[15] U. Mori, "Contributions to time series data mining departing from the problem of road travel time modeling," Ph.D. dissertation, University of the Basque Country, 2015.

[16] T. M. T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[17] S. Köknar-Tezel and L. J. Latecki, "Improving SVM classification on imbalanced time series data sets with ghost points," *Knowledge and Information Systems*, vol. 28, no. 1, pp. 1–23, jul 2011.

[18] H. Cao, X.-L. Li, Y.-K. Woon, and S.-K. Ng, "SPO: Structure Preserving Oversampling for Imbalanced Time Series Classification," in *2011 IEEE 11th International Conference on Data Mining*. IEEE, dec 2011, pp. 1008–1013.

[19] J. Xu, S. Denman, C. Fookes, and S. Sridharan, "Detecting rare events using Kullback-Leibler divergence: A weakly supervised approach," *Expert Systems with Applications*, 2016.

[20] Y. Ren, Y. Wang, X. Wu, G. Yu, and C. Ding, "Influential factors of red-light running at signalized intersection and prediction using a rare events logistic regression model," *Accident Analysis and Prevention*, 2016.
[21] M. Balesdent, J. Morio, and L. Brevault, "Rare Event Probability Estimation in the Presence of Epistemic Uncertainty on Input Probability Distribution Parameters," *Methodology and Computing in Applied Probability*, 2016.
[22] Y. Auffray, P. Barbillon, and J. M. Marin, "Bounding rare event probabilities in computer experiments," *Computational Statistics and Data Analysis*, 2014.
[23] D. Straub, I. Papaioannou, and W. Betz, "Bayesian analysis of rare events," *Journal of Computational Physics*, 2016.
[24] G. King, G. Langche Zeng, J. Fowler, E. Katz, M. Tomz for research assistance, J. Alt, J. Freeman, K. Gleditsch, G. Imbens, C. Manski, P. McCullagh, W. Mebane, J. Nagler, B. Russett, K. Scheve, P. Schrodt, M. Tanner, R. Tucker for helpful suggestions, S. Bennett, P. Huth, and R. Tucker, "Logistic Regression in Rare Events Data," *Political Analysis*, vol. 9, no. 2, pp. 137–163, 2001.
[25] G. M. Weiss and H. Hirsh, "Learning to Predict Rare Events in Event Sequences," *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, 1998.
[26] J.-M. Bourinet, "Rare-event probability estimation with adaptive support vector regression surrogates," *Reliability Engineering and System Safety*, 2016.
[27] S.-P. Cheon, S. Kim, S.-Y. Lee, and C.-B. Lee, "Bayesian networks based rare event prediction with sensor data," *Knowledge-Based Systems*, 2009.
[28] W. Khreich, B. Khosravifar, A. Hamou-Lhadj, and C. Talhi, "An anomaly detection system based on variable N-gram features and one-class SVM," *Information and Software Technology*, 2017.
[29] J. Wu, J. M. Rehg, and M. D. Mullin, "Learning a Rare Event Detection Cascade by Direct Feature Selection," *Neural Information Processing Systems (NIPS)*, vol. 16, 2003.
[30] N. V. Chawla, N. Japkowicz, and A. Ko, "Editorial: Special Issue on Learning from Imbalanced Data Sets," *Sigkdd Explorations*, vol. 6, no. 1, p. 1, 2004.
[31] A. Ogbechie, J. Díaz-Rozo, P. Larrañaga, and C. Bielza, "Dynamic Bayesian Network-Based Anomaly Detection for In-Process Visual Inspection of Laser Surface Heat Treatment," *Machine Learning for Cyber Physical Systems*, pp. 17–24, 2017.
[32] J. Ortigosa-Hernández, I. Inza, and J. A. Lozano, "Towards competitive classifiers for unbalanc classification problems: a study on the performance scores," 2016.
[33] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "iBAT: Detecting anomalous taxi trajectories from GPS traces," *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp '11)*, pp. 99–108, 2011.
[34] Y. Zhou, W. Su, L. Ding, H. Luo, and P. Love, "Predicting Safety Risks in Deep Foundation Pits in Subway Infrastructure Projects: Support Vector Machine Approach," *Journal of Computing in Civil Engineering*, vol. 31, no. 5, p. 04017052, 2017.
[35] S. Fan, G. Liu, and Z. Chen, "Anomaly detection methods for bankruptcy prediction," *2017 4th International Conference on Systems and Informatics (ICSAI)*, no. 17, pp. 1456–1460, 2017.
[36] N. A. Heard, D. J. Weston, K. Platanioti, and D. J. Hand, "Bayesian anomaly detection methods for social networks," *Annals of Applied Statistics*, vol. 4, no. 2, pp. 645–662, 2010.
[37] K. Noto, C. Brodley, and D. Slonim, "FRaC: a feature-modeling approach for semi-supervised and unsupervised anomaly detection." *Data mining and knowledge discovery*, vol. 25, no. 1, pp. 109–133, 2012.
[38] D. Reynolds, "Gaussian Mixture Models," in *Encyclopedia of Biometrics*. Boston, MA: Springer US, 2015, pp. 827–832.
[39] S. Miri Rostami and M. Ahmadzadeh, "Extracting Predictor Variables to Construct Breast Cancer Survivability Model with Class Imbalance Problem," *Shahrood University of Technology*, vol. 6, no. 2, pp. 263–276, jul 2018.
[40] M. Araujo, R. Bhojwani, J. Srivastava, L. Kazaglis, and C. Iber, "ML Approach for Early Detection of Sleep Apnea Treatment Abandonment," *Proceedings of the 2018 International Conference on Digital Health - DH '18*, pp. 75–79, 2018.
[41] M. M. Masud, Q. Chen, L. Khan, C. C. Aggarwal, J. Gao, J. Han, A. Srivastava, and N. C. Oza, "Classification and Adaptive Novel Class Detection of Feature-Evolving Data Streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 7, pp. 1484–1497, jul 2013.
[42] M. Swarnkar and N. Hubballi, "OCPAD: One class Naive Bayes classifier for payload based anomaly detection," *Expert Systems with Applications*, 2016.
[43] F. Dufrenois and J. C. Noyer, "One class proximal support vector machines," *Pattern Recognition*, vol. 52, pp. 96–112, 2016.
[44] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *Pattern Recognition*, 2016.
[45] D. M. J. Tax, "One-class classification," 2001.
[46] V. Mygdalis, A. Iosifidis, A. Tefas, and I. Pitas, "Graph Embedded One-Class Classifiers for media data classification," *Pattern Recognition*, vol. 60, pp. 585–595, dec 2016.
[47] C. Désir, S. Bernard, C. Petitjean, and L. Heutte, "One class random forests," *Pattern Recognition*, vol. 46, no. 12, pp. 3490–3506, dec 2013.
[48] M. Gupta, J. Gao, and C. C. Aggarwal, "Outlier Detection for Temporal Data : A Survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 1–20, 2013.
[49] C. C. Aggarwal, *Outlier Analysis*. Springer International Publishing, 2017.
[50] L. Swersky, H. O. Marques, R. J. G. B. Campello, and A. Zimek, "On the Evaluation of Outlier Detection and One-Class Classification Methods," *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2016.
[51] M. Radovanović, A. Nanopoulos, and M. Ivanović, "Reverse Nearest Neighbors in Unsupervised Distance-Based Outlier Detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 4347, no. OCTOBER, pp. 1–14, 2014.
[52] A. Barai and D. Lopamudra, "Outlier Detection and Removal Algorithm in K-Means and Hierarchical Clustering," *World Journal of Computer Application and Technology*, vol. 5, no. 2, pp. 24–29, 2017.
[53] X. H. Dang, I. Assent, R. T. Ng, A. Zimek, E. Schubert, Xuan Hong Dang, I. Assent, R. T. Ng, A. Zimek, and E. Schubert, "Discriminative features for identifying and interpreting outliers," *Proceedings - International Conference on Data Engineering*, pp. 88–99, mar 2014.
[54] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Kdd*, vol. 96, no. 34, pp. 226–231, 1996.
[55] S. Ramaswamy, R. Rastogi, and K. Shim KAIST, "Efficient Algorithms for Mining Outliers from Large Data Sets," *ACM Sigmod Record 2000*, vol. 29, no. 2, pp. 427–438, 2016.

Una librería para el aprendizaje Multi-instancia Multi-etiqueta

Álvaro Belmonte, Amelia Zafra, Eva Gibaja, Sebastián Ventura
Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Córdoba, España

***Resumen*—Este artículo presenta una librería para trabajar en la resolución de problemas de clasificación con múltiples instancias y múltiples etiquetas. Se describe el formato de datos, la arquitectura software, así como las diferentes propuestas algorítmicas que incorpora. La librería permite añadir nuevos algoritmos de forma sencilla, facilitando a los investigadores en esta área el desarrollo, prueba y comparación de nuevas propuestas. Además, es libre y de código abierto y está implementada en Java, usando las librerías Weka y Mulan. De este modo, los usuarios que trabajan tanto en el aprendizaje con múltiples instancias como en el aprendizaje con múltiples etiquetas, se encontrarán con un entorno de desarrollo con el que están familiarizados.**

***Index Terms*—multi-instance learning, multi-label learning, software**

I. Introducción

A medida que los problemas de clasificación se vuelven más complejos, encontrar una representación adecuada de la información se convierte en una tarea cada vez más complicada. La experiencia demuestra que una representación precisa, que sea capaz de reflejar todas las relaciones e interacciones existentes en los datos, influye directamente en una resolución más efectiva del problema. En este contexto, el aprendizaje con múltiples instancias y múltiples etiquetas (*multi-instance multi-label learning*, MIML) se presenta como una alternativa prometedora que permite realizar una formulación natural que se adapta a las condiciones particulares de la representación de objetos complejos que suelen tener los problemas reales. Entre los problemas donde ha sido aplicado con éxito destacan la categorización de textos o imágenes [1]–[3], la detección de vídeo y audio [4] o la bioinformática [5], [6].

El aprendizaje MIML introduce una mayor flexibilidad en la representación de objetos, tanto en el espacio de entrada como en el de salida, debido a los dos paradigmas de aprendizaje que combina. Por un lado, la representación basada en el aprendizaje multi-instancia (*multi-instance*, MI) [7] ofrece una alternativa a la representación de instancias simples. En el aprendizaje MI un objeto o patrón, conocido habitualmente como *bolsa*, es representado mediante un conjunto variable de instancias, todas ellas con el mismo número de atributos. De este modo, un objeto puede ser representado con descripciones alternativas [7], con diferentes componentes [2], o bien mostrando su evolución en el tiempo [8]. Por otro lado, el aprendizaje multi-etiqueta (*multi-label*, ML) [9] introduce una mayor flexibilidad, esta vez en el espacio de salida, permitiendo que cada patrón tenga la posibilidad de pertenecer de forma simultánea a varias clases (etiquetas). Por ejemplo, para la resolución del problema de clasificación de imágenes, una imagen podría ser representada mediante múltiples instancias donde cada instancia se correspondería con diferentes regiones de la imagen, pudiendo a su vez tener diferentes etiquetas, tales como nubes, leones, y paisajes.

Actualmente, existen herramientas que permiten resolver problemas utilizando el aprendizaje MI y ML. Así, Weka [10] permite trabajar con problemas MI, y las librerías MEKA [11] y MULAN [12] permiten abordar el aprendizaje ML. No obstante, ninguna de ellas permite trabajar con el aprendizaje MIML. Hasta donde los autores conocemos, los únicos algoritmos públicamente disponibles para resolver problemas MIML han sido desarrollados por el grupo de investigación LAMDA [13]. Las principales limitaciones de estas implementaciones son que: están codificados en Matlab, con lo que es necesario disponer de licencia de este software para poder ejecutarlos y están formados por paquetes independientes, que habitualmente contienen la implementación de un único algoritmo que utiliza un formato de entrada y salida distinto al de otros paquetes, lo que dificulta la tarea de poder realizar un estudio experimental con ellos.

En este contexto, este artículo presenta una librería para aprendizaje MIML basada en las librerías Weka y MULAN, con lo que los investigadores en MI y ML, que empleen las librerías anteriores, estarán familiarizados con su estructura y formato. Entre sus características más relevantes se puede destacar: que utiliza un formato de datos diseñado específicamente para este aprendizaje, que cuenta con un conjunto de algoritmos desarrollados que directamente pueden ser ejecutados, que permite el uso de los algoritmos implementados para aprendizaje MI y ML de Weka y Mulan en un contexto MIML, que facilita el diseño y desarrollo de nuevos modelos que resuelvan problemas de clasificación con una representación MIML, que permite llevar a cabo un estudio experimental utilizando validación cruzada, y finalmente, su uso y ejecución resultan sencillos mediante la configuración de ficheros xml.

El resto del artículo se organiza como sigue. En la sección 2 se describe diferentes algoritmos MIML propuestos en la literatura. La sección 3, muestra la descripción de la librería, considerando el formato de los datos, su funcionalidad, su arquitectura y los principales paquetes y elementos de los que consta. En la sección 4, se describe algunos ejemplos de uso. Finalmente, la sección 5 describe las conclusiones más

relevantes del trabajo realizado.

II. Trabajo Previo

Los algoritmos de clasificación desarrollados para aprendizaje MIML se pueden clasificar en dos enfoques [14]. Por un lado, algoritmos basados en una estrategia de transformación del problema MIML, y por otro lado, algoritmos que abordan el problema MIML directamente.

Debido a que tanto el aprendizaje MI como el aprendizaje ML son parte de MIML, se pueden usar como vía o puente para la transformación y posterior resolución de problemas MIML. Una primera aproximación consiste en aplicar alguna técnica de descomposición de etiquetas que transforme el problema MIML a un problema MI. Después de esto, puede ser utilizado cualquier algoritmo MI para resolver el problema. Otra aproximación consiste en emplear alguna técnica que permita unificar las diferentes instancias de una bolsa en una única instancia, transformando el problema MIML en un problema ML al que se le puede aplicar cualquier algoritmo de ML. En la literatura se pueden encontrar diferentes algoritmos que realizan una transformación del problema, entre ellos encontramos métodos basados en *ensembles* [15], [4], máquinas de vector soporte [5] y métodos basados en redes neuronales [1].

El rendimiento de los algoritmos anteriores puede verse limitado debido a la pérdida de información incurrida durante el proceso de simplificación/transformación. Idealmente, las conexiones entre instancias y etiquetas, así como la correlación entre las propias etiquetas deberían ser tenidas en cuenta. Por este motivo, también se han propuesto técnicas para abordar el problema MIML de forma directa basadas en redes neuronales [16], *ensembles* [6], máquinas de vector soporte [3] o vecinos más cercanos [2]. En [17] puede encontrarse una revisión más exhaustiva de algoritmos publicados en el aprendizaje MIML.

III. Descripción de la librería

En esta sección se comentará el formato de los datos, los algoritmos y funcionalidades que se encuentran implementados, los paquetes de los que se compone y el formato de los ficheros de configuración para ejecutar cualquiera de sus algoritmos de clasificación.

III-A. Formato de los datos

El formato diseñado para la representación de la estructura de un problema MIML está basado en los formatos de datos utilizados por Weka y MULAN. De este modo, un conjunto de datos estará representado mediante dos ficheros:

- Un fichero *xml* cuya función principal es identificar a los atributos del fichero *arff* que representan las etiquetas. De esta manera, las etiquetas no tienen por qué ser necesariamente los últimos atributos que se definan. Además, este fichero permite representar, anidando etiquetas, una jerarquía de clases. A continuación, se muestra un ejemplo con la definición de cuatro etiquetas.

```
<?xml version="1.0" encoding="utf-8"?>
<labels xmlns="http://mulan.sourceforge.net/labels">
  <label name="label1"></label>
  <label name="label2"></label>
  <label name="label3"></label>
  <label name="label4"></label>
</labels>
```

- Un fichero *arff* (*Attribute-Relation File Format*) cuya estructura presenta dos partes, cabecera y datos:
  - La **cabecera** contiene el nombre de la relación junto con una lista de atributos y sus tipos.
    - En la primera línea del fichero se encuentra la sentencia *@relation <relation-name>* que define el nombre del conjunto de datos. Es una cadena de caracteres que, si contiene espacios, debe de ir entrecomillada.
    - A continuación, se definen dos atributos: el primero de tipo nominal que identifica unívocamente a cada bolsa y el segundo atributo, relacional, que contiene los atributos que describen a las instancias. Para definir los atributos se utilizan las sentencias *@attribute <attribute-name><data-type>*. Habrá una línea por atributo:
      - Los tipos de dato numérico se especifican con *numeric*.
      - Si se trata de un conjunto posible de valores nominales debe ir especificado entre llaves y separado por comas. Por ejemplo: $\{valor_1, valor_2, ..., valor_N\}$.
    - Finalmente, se definen los atributos correspondientes a las etiquetas. Estos atributos tienen que ser de tipo binario (nominales con valores 0 y 1).
  - La sección de **datos** comienza con un *@data*. Cada una de las líneas siguientes representará una bolsa. Los atributos deben de estar ordenados según la declaración realizada en la cabecera. El valor de cada atributo se separa por comas y todas las filas deben tener el mismo número de columnas. Los decimales deben de separarse con el punto. El contenido del atributo relacional estará entrecomillado (comillas simples o dobles), separando cada instancia de la bolsa por un $\backslash n$. A continuación, se muestra un ejemplo de fichero *arff* donde cada bolsa está formada por tres atributos de tipo numérico y tiene cuatro etiquetas. La primera bolsa está formada por 3 instancias, y la segunda bolsa por 2 instancias.

```
@relation toy
@attribute id {bag1,bag2}
@attribute bag relational
  @attribute f1 numeric
  @attribute f2 numeric
  @attribute f3 numeric
@end bag
@attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}
```

```
@data
bag1,"42,−198,−109\n42,−191,−142\n3,4,6",1,0,0,1
bag2,"12,−98,10\n42,−19,−12",0,1,1,0
```

La librería cuenta con las clases necesarias tanto para representar bolsas individuales (clase *Bag* del paquete *data*) como para representar un conjunto de datos completo (clase *MIMLInstances*). También proporciona medios para guardar un conjunto de datos en sus respectivos ficheros *arff* y *xml* y para obtener diferentes métricas que te permitan estudiarlos en más profundidad (*MIMLStatistics* en *data.statistics*).

III-B. Funcionalidad incluida en la librería

La librería contiene métodos para transformar un conjunto de datos MIML a otro tipo de representación como MI o ML, algoritmos de clasificación MIML y clases para dar soporte al desarrollo de experimentos.

III-B1. Métodos para transformar datos MIML: se han incluido los siguientes métodos:

- Métodos para transformar a MI (formato de datos usado en Weka) [9]:
  - *Binary Relevance Transformation*: transforma un conjunto de datos MIML en tantos conjuntos de datos MI binarios como etiquetas tenga el problema.
  - *Label Powerset Transformation*: transforma un conjunto de datos MIML en uno multiclase en el que cada posible combinación de etiquetas del conjunto de datos original es considerado una clase diferente.
- Métodos para transformar a ML (formato de datos usado en MULAN) [18]:
  - *Arithmetic Transformation*: transforma cada bolsa en una única instancia donde el valor para cada atributo es su valor medio dentro de la bolsa.
  - *Geometric Transformation*: transforma cada bolsa en una única instancia donde el valor para cada atributo es el centro geométrico de sus valores máximo y mínimo dentro de la bolsa.
  - *Min-Max Transformation*: transforma cada bolsa en una única instancia que contiene, para cada atributo, sus valores mínimo y máximo dentro de esa bolsa. Cada instancia está definida por el doble de atributos de los que tenía previamente.

III-B2. Algoritmos de clasificación: se han incluido los siguientes algoritmos:

- *MIMLClassifierMI*: realiza una transformación del problema MIML para obtener un problema MI aplicando una transformación LP o BR. Posteriormente, obtiene un resultado resolviendo el problema con un algoritmo MI que se especifique. Al ser compatible con la librería Weka, en la tabla I se muestra un ejemplo de 15 algoritmos de Weka que podrían ser utilizados directamente este clasificador.
- *MIMLClassifierML*: realiza una transformación del problema MIML para obtener un problema ML aplicando una transformación aritmética, geométrica o min-max. Posteriormente, obtiene un resultado resolviendo el problema con un algoritmo ML que se especifique. Al ser compatible con la librería MULAN, en la tabla I se muestra un ejemplo de 15 algoritmos de MULAN que podrían ser utilizados directamente este clasificador.
- *MIML-kNN* [2]: utiliza los vecinos y referencias más cercanos a una bolsa para estimar las posibles clases a las que pertenece, trabajando directamente sobre datos MIML.

III-B3. Otra funcionalidad: la librería presenta un marco de trabajo para manejar conjuntos de datos MIML, obtener informes, calcular medidas de distancia, realizar experimentos utilizando validación cruzada o indicando los conjuntos de datos de entrenamiento y test, así como las clases e interfaces genéricas y necesarias para desarrollar nuevos clasificadores, que permite que el desarrollo de nuevas propuestas y la realización de un estudio experimental sea una tarea sencilla.

III-C. Arquitectura de la librería

La librería, desarrollada en Java y de código abierto, hace uso de las librerías MULAN y Weka. Se encuentra estructurada en siete paquetes, algunos de ellos organizados en subpaquetes, de acuerdo con su funcionalidad:

- **core**: incluye diferentes funcionalidades que serán utilizadas por otras clases de la librería. Este paquete contiene las clases relacionadas con la configuración de los métodos utilizando ficheros *xml*.
- **core.distance**: contiene diferentes variantes de la distancia de *Hausdorff* para calcular la distancia entre bolsas.
- **data**: incluye clases para trabajar con el formato de datos detallado en la sección III-A. Entre las funcionalidades que soporta se encuentra: cargar en memoria un conjunto de datos, conocer propiedades de los datos como el número de instancias de una bolsa concreta, el número de atributos, el número de bolsas totales, el número de etiquetas, etc. Además, permite acceder tanto a una bolsa particular, como a sus instancias y etiquetas.
- **data.statistics**: incluye clases encargadas de obtener información relevante de los conjuntos de datos MIML. Se considera tanto información referente a MI (atributos por

Tabla I: Algoritmos compatibles

Clasificadores MI (Weka)	Clasificadores ML (MULAN)
CitationKNN	BRkNN
MDD	DMLkNN
MIDD	IBLR
MIBoost	mLkNN
MIEMDD	HOMER
MILR	RAkEL
MINND	EPS
MIOptimalBall	ECC
MIRI	MLStacking
MISMO	BR
MISVM	LP
MITI	PS
MIWrapper	CC
SimpleMI	RPC

bolsa, número medio de instancias por bolsa, número de instancias mínimo y máximo por bolsa, etc.), como a ML (frecuencia de los conjuntos de etiquetas, distribución de etiquetas por bolsa, co-ocurrencias entre etiquetas, etc.).

- **transformation.mimlTOml**: incluye los métodos para transformar un problema MIML en otro ML.
- **transformation.mimlTOmi**: incluye los métodos para transformar un problema MIML en otro MI.
- **mimlclassifier**: incluye las interfaces y las clases abstractas necesarias en el desarrollo de los algoritmos de clasificación para resolver problemas MIML.
- **mimclassifier.mimlTOmi**: incluye los algoritmos de clasificación que resuelven el problema MIML transformando previamente el problema a MI.
- **mimclassifier.mimlTOml**: incluye los algoritmos de clasificación que resuelven el problema MIML transformando previamente el problema a ML.
- **mimclassifier.lazy**: contiene el algoritmo MIMLkNN.
- **report**: incluye clases que permiten generar informes de resultados de los experimentos realizados. La librería incorpora un informe general con principales métricas que se pueden utilizar para evaluar un modelo MIML.
- **tutorial**: incluye ejemplos de uso de las diferentes funcionalidades de la librería: ejecutar y obtener los resultados de un algoritmo de clasificación MIML, transforma un conjunto de datos MIML a ML o MI, etc.
- **tutorial.config**: incluye ejemplos de ficheros de configuración para poder ejecutar y obtener resultados de los algoritmos de clasificación MIML incluidos en la librería.
- **miml**: incluye la clase para ejecutar cualquier clasificador de la librería mediante un fichero de configuración.

Todos los paquetes contienen las interfaces y las clases necesarias para extender su funcionalidad heredando e implementado la interfaz concreta. De este modo, es posible desarrollar de forma sencilla un nuevo método de transformación del conjunto de datos, y/o un nuevo algoritmo de clasificación. La figura 1 muestra un diagrama que representa las diferentes clases de la librería y sus relaciones.

III-D. Ficheros de configuración

Para facilitar la ejecución de los diferentes algoritmos (ver la sección III-B), la librería cuenta con una clase principal *RunAlgorithm*, que junto con un fichero de configuración en formato *xml*, que especifica el clasificador que se va a utilizar, el conjunto de datos y el informe de salida que se desea obtener, ejecuta y obtiene los resultados de cualquier clasificador MIML que se desee emplear. Estos ficheros tienen que seguir el siguiente formato para que la librería, a partir de la clase *ConfigLoader*, sea capaz de leerlos y configurar todo el proceso que se debe de seguir, desde la creación del modelo hasta la generación de los informes con los resultados:

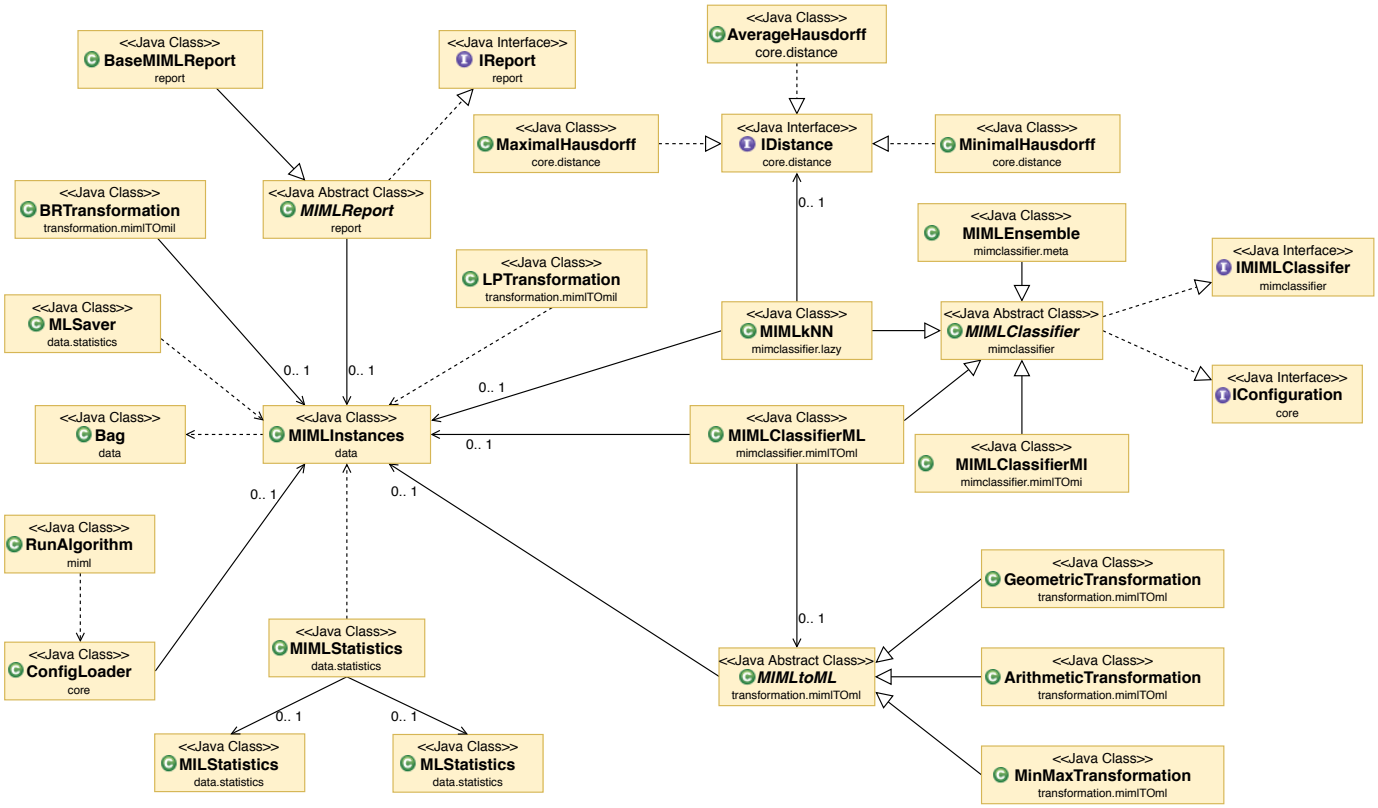


Figura 1: Diagrama de clases de la librería

```
<configuration>
  <classifier>  </classifier>
  <evaluator>  </evaluator>
  <report> </report>
</configuration>
```

Todos los ficheros deben empezar con el elemento *configuration*. En la etiqueta *classifier*, se especificará el clasificador de la librería que se quiere utilizar. Dentro del clasificador será donde se detalle su configuración: los parámetros específicos que necesite, el método de transformación a utilizar, y/o el clasificador MI o ML que se requiera en las versiones que transforman el problema. La etiqueta *evaluator* detallará el conjunto de datos a utilizar, y si se lleva a cabo una validación cruzada o se especifican directamente los ficheros *train/test*. Finalmente, la etiqueta *report*, indicará el informe que genera el fichero de salida. Esta clase puede ser extendida fácilmente para que los usuarios especifiquen el formato de salida más conveniente.

IV. Ejemplos de uso

La experimentación mediante ficheros *xml* se puede llevar a cabo haciendo uso de la clase ejecutable *RunAlgorithm* pasándole como argumento, a través de la opción *-c*, la ruta del fichero de configuración. A continuación, se mostrarán ejemplos para un conjunto de casos ilustrativos.

IV-A. Ejecución de un algoritmo de clasificación MIML reduciendo el problema a MI

El ejemplo de fichero *xml* para este caso es el siguiente:

```
<configuration>
  <classifier name="mimlclassifier.mimlTOmi.MIMLClassifierMI"
      >
    <multiInstanceClassifier name="weka.classifiers.mi.MISMO">
        <listOptions>-C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V
            -1 -W 1 -K "weka.classifiers.mi.supportVector.
            MIPolyKernel -E 1.0 -C 250007"</listOptions>
        </multiInstanceClassifier>
        <transformMethod name="mulan.classifier.transformation.
            BinaryRelevance"/>
  </classifier>
  <evaluator method="train-test">
    <data>
      <trainFile>data/miml_03_data.arff</trainFile>
      <testFile>data/miml_04_data.arff</testFile>
      <xmlFile>data/miml_03_data.xml</xmlFile>
    </data>
  </evaluator>
  <report name="report.BaseMIMLReport">
      <fileName>results/results.csv</fileName>
    </report>
</configuration>
```

- El clasificador que se desea utilizar debe especificarse en el atributo *name* del elemento *classifier*. Se va a emplear el clasificador *MIMLClassifierMI*. Este clasificador necesita que se especifique en *transformMethod* el método de transformación que se va a utilizar para transformar el problema MIML en un problema MI. En el ejemplo se hace uso del método BR de la librería de Mulan.
- Como configuración de este algoritmo, se debe detallar también en la etiqueta *multiInstanceClassifier* el algoritmo MI que se utilizará una vez realizada la transformación. En este caso se ha utilizado MISMO, un clasificador de la librería Weka para problemas MI. En la etiqueta *listOptions* detallaremos los parámetros que se quieren utilizar en el algoritmo MISMO. Si no se detalla ninguno, se tomarán los valores por defecto establecidos para cada algoritmo.
- El atributo *method* de la etiqueta *evaluator* especifica el proceso de evaluación que se llevará a cabo.
- En el caso de que se vaya a realizar un método *train/test* es necesario especificar en la etiqueta *data*, que pertenece al *evaluator*, la ruta de los ficheros *arff* de entrenamiento y test, así como la del fichero *xml* con las etiquetas.
- Por último, en el elemento *report* se indicará el informe que se desea utilizar, y la configuración de cada informe. En el informe que se ha especificado en el ejemplo, solamente es necesario especificar en *fileName* la ruta donde se quiere guardar el informe de resultados.

IV-B. Ejecución de algoritmo degenerativo utilizando ML

Para la realización de un experimento utilizando ML como vía, un ejemplo de fichero de configuración sería:

```
<configuration>
  <classifier name="mimlclassifier.mimlTOml.MIMLClassifierML"
      >
    <multiLabelClassifier name="mulan.classifier.lazy.MLkNN">
      <parameters>
        <classParameters>int.class</classParameters>
        <classParameters>double.class</classParameters>
        <valueParameters>1</valueParameters>
        <valueParameters>1.0</valueParameters>
      </parameters>
    </multiLabelClassifier>
    <transformMethod name="transformation.mimlTOml.
        ArithmeticTransformation"/>
  </classifier>
  <evaluator method="cross-validation">
    <numFolds>5</numFolds>
    <data>
      <file>data/miml_03_data.arff</file>
      <xmlFile>data/miml_03_data.xml</xmlFile>
    </data>
  </evaluator>
  <report name="report.BaseMIMLReport">
      <fileName>results/results.csv</fileName>
    </report>
</configuration>
```

- En este caso, el elemento *multiInstanceClassifier* es sustituido por *multiLabelClassifier*.
- En caso de que el clasificador ML que se utilice se pueda configurar, los parámetros se deben de especificar de la siguiente forma (para respetar la forma de ejecución que tiene establecida MULAN):
  - En primer lugar se pondrán tantos elementos *classParameters* como parámetros tenga el constructor del clasificador. Su contenido debe de ser el tipo al que

pertenecen dichos parámetros, tienen que estar en el orden en el que se declaren en el constructor.
    - A continuación, se especifican los valores que se quiere que tengan dichos parámetros, indicando cada uno de ellos en un elemento *valueParameters* diferente que, nuevamente, tienen que estar en orden.
- El clasificador que se ha utilizado (*MIMLClassifierML*) se adapta a todos los métodos de transformación disponibles en *transformation.mimlToml*, es necesario especificar en su configuración el elemento *transformMethod*.
- Este experimento se ha configurado para que realice una evaluación basada en validación cruzada, especificado en el atributo *method* la cadena *"cross-validation"*. Es necesario indicar el número de *folds* en el elemento *numFolds* y el fichero que contiene el conjunto de datos. La etiqueta *report* sería similar al ejemplo anterior.

IV-C. Ejecución de algoritmo basado en vecinos más cercanos

Por último, se muestra un ejemplo de fichero *xml* para ejecutar el algoritmo MIMLkNN:

```
<configuration>
  <classifier name="mimlclassifier.lazy.MIMLkNN">
    <nReferences>2</nReferences>
    <nCiters>2</nCiters>
    <metric name="core.distance.AverageHausdorff">
    </metric>
  </classifier>
  <evaluator method="train-test">
    <data>
      <trainFile>data/miml_03_data.arff</trainFile>
      <testFile>data/miml_04_data.arff</testFile>
      <xmlFile>data/miml_03_data.xml</xmlFile>
    </data>
  </evaluator>
  <report name="report.BaseMIMLReport">
      <fileName>results/results.csv</fileName>
    </report>
```

- Con los algoritmos que trabajen directamente con el formato MIML tan solo es necesario especificar en su configuración los parámetros para su ejecución. En este ejemplo, el algoritmo MIML-kNN necesita configurar el número de referencias y citas con los que trabajará, así como la métrica que va a utilizar para medir la distancia entre las bolsas de un *data set*. Las etiquetas *evaluator* y *report*, serían similares a los ejemplos anteriores.

V. Conclusiones

En este trabajo se presenta una librería Java basada en Weka y Mulan para el desarrollo, prueba y comparación de algoritmos dentro del marco de trabajo del aprendizaje MIML. Además de un conjunto de algoritmos MIML, de referencia en la temática, la estructura de la librería permite el desarrollo de nuevas propuestas de forma sencilla. Finalmente, se ha diseñado un formato de datos específico para MIML. La librería es sencilla de utilizar mediante el uso de ficheros de configuración *xml*.

Agradecimientos

Este trabajo está financiado por el proyecto de investigación TIN2017-83445-P del Ministerio de Economía y Competitividad y el Fondo Europeo de Desarrollo Regional.

Referencias

[1] K. Yan, Z. Li, and C. Zhang. A new multi-instance multi-label learning approach for image and text classification. *Multimedia Tools and Applications*, 75(13):7875–7890, 2016.

[2] M.L. Zhang. A k-nearest neighbor based multi-instance multi-label learning algorithm. In *Proceedings of the 22nd International Conference on Tools with Artificial Intelligence*, volume 2, pages 207–212, 2010.

[3] C. Tong-tong, L. Chan-juan, Z. Hai-lin, Z. Shu-sen, L. Ying, and D. Xin-miao. A multi-instance multi-label scene classification method based on multi-kernel fusion. In *Proceedings of the Conference on Intelligent Systems*, pages 782–787, 2015.

[4] X.S. Xu, X. Xue, and Z. Zhou. Ensemble multi-instance multi-label learning approach for video annotation task. In *Proceedings of the 19th International Conference on Multimedia*, pages 1153–1156. ACM, 2011.

[5] Y.X. Li, S. Ji, S. Kumar, J. Ye, and Z.H. Zhou. Drosophila gene expression pattern annotation through multi-instance multi-label learning. *Transactions on Computational Biology and Bioinformatics*, 9(1):98–112, 2012.

[6] J.S. Wu, S.J. Huang, and Z.H. Zhou. Genome-wide protein function prediction through multi-instance multi-label learning. *Transactions on Computational Biology and Bioinformatics*, 11(5):891–902, 2014.

[7] T.G. Dietterich, R.H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1):31–71, 1997.

[8] S. Kotsiantis, D. Kanellopoulos, and V. Tampakas. Financial application of multi-instance learning: two greek case studies. *Journal of Convergence Information Technology*, 5(8):42–53, 2010.

[9] E. Gibaja and journal=Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery volume=4 number=6 pages=411–444 year=2014 publisher=Wiley Online Library Ventura, S. Multi-label learning: a review of the state of the art and ongoing research.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The weka data mining software: An update. *SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[11] J. Read, P. Reutemann, B. Pfahringer, and G. Holmes. Meka: a multi-label/multi-target extension to weka. *Journal of Machine Learning Research*, 17(1):667–671, 2016.

[12] G. Tsoumakas, E. Spyromitros-Xioufis, Jozef Vilcek, and I. Vlahavas. Mulan: A java library for multi-label learning. *Journal Machine Learning Research*, 12:2411–2414, 2011.

[13] LAMDA Learning and Mining from DatA. http://lamda.nju.edu.cn/Data.ashx. Accessed: 2018-06-19.

[14] Z.H. Zhou, M.L. Zhang, S.J. Huang, and Y.F. Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.

[15] Z.H. Zhou and M.L. Zhang. Multi-instance multi-label learning with application to scene classification. In *Proceedings of the Advances in neural information processing systems*, pages 1609–1616, 2006.

[16] C. Li and G. Shi. Weights optimization for multi-instance multi-label rbf neural networks using steepest descent method. *Neural Computing and Applications*, 22(7-8):1563–1569, 2013.

[17] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Tarragó, and S. Vluymans. *Multiple Instance Learning. Foundations and Algorithms*. Springer, 230 pages, 2016.

[18] Eibe Frank and Xin Xu. Applying propositional learning algorithms to multi-instance data. pages 1–13, *Computer Science Working Papers. University of Waikato, Department of Computer Science*. 2003.

IX Simposio de Teoría y Aplicaciones de la Minería de Datos (IX TAMIDA)

TAMIDA 7:
DEEP LEARNING

Una primera aproximación a la predicción de variables turísticas con Deep Learning

Daniel Trujillo, Antonio Jesús Rivera, Francisco Charte, María José del Jesus
Instituto Andaluz de Investigación en Ciencia de Datos e Inteligencia Computacional (DaSCI), Departamento de Informática
Universidad de Jaén
Jaén, España
{dtviedma,arivera,fcharte,mjjesus}@ujaen.es

***Resumen*—El turismo es una de las actividades económicas más importantes a nivel mundial, por lo que una correcta planificación de los recursos existentes en función de la demanda es fundamental. En este sentido, el trabajo desarrollado permite comparar la bondad de un nuevo modelo de *deep learning*, LSTM, frente a un modelo clásico ampliamente reconocido, ARIMA. Se ha llevado a cabo un proceso de entrenamiento para obtener los modelos LSTM y ARIMA que, posteriormente se han validado utilizando datos no disponibles durante el aprendizaje.**

Nuestros resultados muestran que los nuevos modelos LSTM obtienen una precisión mayor que el clásico ARIMA, tanto en la validación a priori como en la predicción posterior.

***Palabras clave*—LSTM; ARIMA; time series forecasting**

I. Introducción

La industria del turismo a nivel mundial se considera una de las actividades económicas más importantes, sólo superado por la industria del petróleo y productos derivados [2], según algunos autores. En el caso español, al sector turístico se vinculan un 12,1% del total de empleos en 2013, creciendo desde 9,8% en 2001 [3]. En una actividad tan importante es fundamental una correcta planificación, de manera que, por un lado, se pueda atender adecuadamente a la demanda, y por otro, se pueda realizar el mantenimiento del patrimonio que da lugar a la actividad turística.

Por su naturaleza, de este tipo de actividades emanan conjuntos de datos con una fuerte dependencia temporal, lo que también se conoce como **series temporales** [**?**]. En estos conjuntos de datos, las instancias aparecen ordenadas por un criterio cronológico, al tiempo que la variable de estudio presenta una fuerte dependencia cronológica. Ejemplos de este tipo de conjuntos de datos son la evolución del precio de un bien, del número de pasajeros de un aeropuerto, o del grado de ocupación hotelera en una provincia concreta.

Tradicionalmente, el método que se ha utilizado para analizar este tipo de conjuntos de datos ha sido ARIMA [4]. Este modelo integra análisis de medias móviles y análisis autoregresivo, conocido como modelo ARMA, y lo generaliza para series diferenciadas. Este modelo, junto con las estrategias heurísticas desarrolladas para encontrar sus mejores parámetros, tiene una muy importante fundamentación estadística y se le considera como el estado del arte en el ámbito de predicción de series temporales.

Por su parte, las técnicas de *deep learning* han supuesto una revolución importante en el campo del aprendizaje automático, gracias a evoluciones en los algoritmos de entrenamiento disponibles, así como el aumento de la potencia de cómputo de los ordenadores actuales. Esta combinación de factores posibilitan el uso de estos modelos capaces de reproducir procesos cognitivos complejos reservados a los humanos. Modelos de *deep learning* han conseguido resultados sorprendentes en tareas como el reconocimiento de escritura [5], lo cual los hace prometedores sucesores de los algoritmos tradicionales de aprendizaje automático.

Uno de estos modelos, LSTM (*Long Short Term Memory*), está especialmente diseñado para tareas de análisis de series temporal. Su estructura le permite almacenar conocimiento útil para razonar sobre una serie temporal con un contexto mayor que las técnicas tradicionales, estableciendo una nueva línea de estudio que quizás termine por suceder al actual estado del arte, el modelo ARIMA.

El objeto de este estudio es establecer una comparativa entre el algoritmo más comúnmente usado en predicción de series temporales, ARIMA, y las redes neuronales de tipo LSTM, en un problema real de predicción de datos turísticos de interés para la provincia de Jaén.

Este artículo se estructura de la siguiente forma: En la segunda sección se aportará una breve introducción de los métodos considerados en el estudio. A continuación, se detallará la experimentación llevada a cabo para establecer la comparativa, tras lo cual se mostrarán los resultados obtenidos para finalizar con una discusión de los mismos.

II. Métodos

En esta sección se aporta una descripción a nivel introductorio de los métodos que utilizaremos para llevar a cabo la experimentación, LSTM y ARIMA. Al no ser objetivo de este estudio proporcionar una descripción suficientemente detallada, se aportan las referencias más relevantes en cada caso.

Universidad de Jaén.

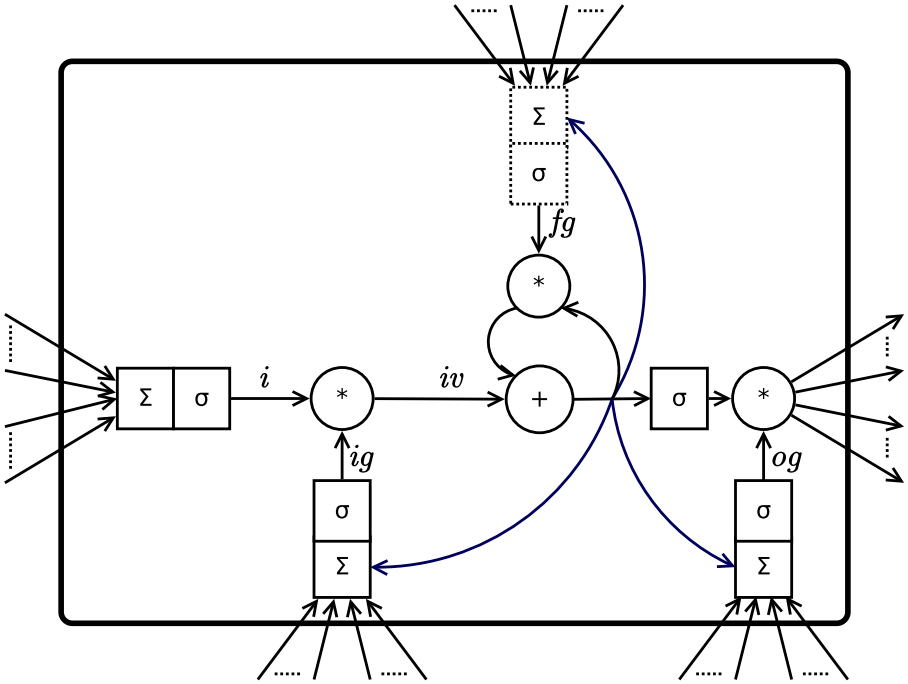


Fig. 1. Representación esquemática de una neurona tipo LSTM.

A. LSTM

Las redes neuronales de tipo LSTM surgen como respuesta a los diversos problemas en torno al flujo del gradiente que sufren los algoritmos de aprendizaje basados en gradiente de redes neuronales artificiales, los cuales se discuten ampliamente en [6], siendo el más recurrente en este tipo de redes el llamado ”vanishing gradient”. Este problema hace referencia al decreciente efecto de las variaciones de un parámetro en la salida final de la red, en la medida en la que el parámetro se aleja de la salida de la red. Esto afecta a las capas de entrada de redes prealimentadas con un número elevado de capas, pero también a redes neuronales recurrentes con amplias dependencias temporales en los datos de entrada.

Las redes neuronales de tipo LSTM han supuesto un importante impulso para las redes neuronales recurrentes, ya que, por los motivos expuestos, los algoritmos de entrenamiento de redes neuronales recurrentes más utilizados no ofrecían ninguna ventaja práctica frente a los mismos algoritmos en redes neuronales prealimentadas [6].

Long Short Term Memory es una arquitectura de neurona artificial cuya estructura le permite mantener un estado interno formado a partir de las evaluaciones que se han realizado, y que influye en las posteriores. Esta estructura le permite memorizar conocimiento que utilizará y actualizará en evaluaciones posteriores.

En su formulación original [7], esta célula define un flujo de información a través del cual, los valores de entrada se modifican mediante operaciones multiplicativas y funciones de activación para obtener su salida.

Sin embargo, posteriores modificaciones a la estructura original se han incorporado de facto en esta formulación, dando lugar a la estructura LSTM más utilizada en la literatura actual, que consta de los siguientes elementos:

- *Input*, *input gate*, *forget gate* y *output gate*: unidades simples, que primeramente aplican pesos a los valores de entrada, a los que se suman el valor de salida de *CEC* resultado de la última evaluación del *CEC* (*peephole connections*), y posteriormente aplica una función de activación.
- CEC (*Constant Error Carrousel*): situado en la parte central de la neurona, está formado por una autoconexión recurrente. Esta autoconexión constituye la memoria de la neurona, cuyo valor se suma a la entrada del *CEC* y se actualiza en cada evaluación de la neurona a su salida.

En la fig. 1 se detalla el interior de una neurona LSTM. En ella se puede apreciar muy facilmente el flujo de informacion dentro de la neurona, de izquierda a derecha, mientras es transformada por las 3 válvulas de información. Asimismo, se puede ver el CEC, justo en el centro de la neurona.

A continuación se proporciona una descripción textual de la neurona LSTM:

1) Los valores de entrada se suministran a las unidades *input*, *input gate* y *forget gate*, obteniendo los valores *i*, *ig* y *fg*, respectivamente
2) *i* e *ig* se combinan multiplicando sus valores elemento a elemento, obtenido *iv*.
3) *fg* se multiplica, elemento a elemento, con el valor de salida del *CEC* en el instante de tiempo anterior, obteniendo el nuevo valor de entrada de *CEC*.
4) Este valor de entrada de *CEC* se suma con *iv*, elemento a elemento, obteniendo la salida de *CEC*. Ésta, además de continuar por el flujo de información, se copia a

las *input*, *forget* y *output gate* a través de las *peephole connections*.

5) Se calcula la salida del *output gate*, *og*, dados los valores originales de entrada y el nuevo valor de salida del *CEC* a través de las peephole connection.
6) Por último, la salida de *CEC* se suministra a una función de activación, cuyo resultado se multiplica elemento a elemento con *og* para obtener la salida final de toda la neurona LSTM.

Esta estructura interna está diseñada para trabajar con series de datos en intervalos mucho más amplios, ya que se asegura, junto con un entrenamiento adecuado, generalmente mediante backpropagation, un flujo de error constante en el *CEC*, mientras que la *input gate* protege el valor recordado en el *CEC* de entradas irrelevantes, y la *output gate* protege a otras células de valores irrelevantes en el *CEC*.

Esta descripción de la estructura estándar es producto, como se ha mencionado antes, de modificaciones sobre la estructura original que se define en [7]. Dichas modificaciones incluyen:

- *Forget gate*: Previamente a su introducción en [8], el valor de salida del *CEC* simplemente se copia a la entrada del *CEC*, mientras que este valor, agregado a los datos de entrada, se suministran a una función de activación. Tras su introducción, se elimina la función de activación, y la salida del *CEC* pasa a ser simplemente la agregación del valor anterior del *CEC* a los datos de entrada.
- *Peephole connections*: Son conexiones con peso desde la salida del *CEC* hasta las *input gate* y *forget gate*, donde se usarán en la siguiente evaluación; y hasta la *output gate*, donde se usará inmediatamente para calcular el valor *og* y la salida final de la LSTM.

B. ARIMA

ARIMA [4] es un modelo ampliamente utilizado en la literatura, por su especial bondad en tareas de predicción de series temporales. El funcionamiento general de este modelo consiste en aproximar la serie temporal mediante una función matemática que pueda ser evaluada en las condiciones que se desean predecir. Esto significa que ARIMA no sólo es útil en tareas predictivas, sino que el ajuste que realiza de la serie temporal puede ser analizado desde un punto de vista descriptivo.

ARIMA se construye a partir de otro modelo, llamado ARMA, generalizándolo para series temporales que han sido diferenciadas. Esto es, una serie temporal obtenida como las diferencias entre cada elemento y el anterior de la serie temporal inicial. El nombre de ARMA proviene de las iniciales *AutoRegressive and Moving Averages*, y consiste en encontrar, primeramente un modelo autoregresivo sobre las diferencias de los valores a predecir, y posteriormente modelar el error de la anterior regresión a través de un modelo de medias móviles.

III. Experimentación

Para conocer el desempeño de los algoritmos se han realizado ejecuciones de ambos sobre un conjunto de datos real,

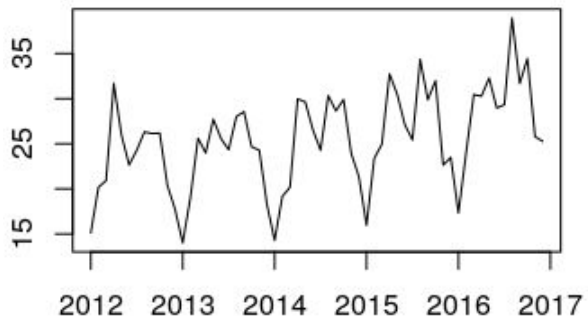


Fig. 2. Visualización de la serie temporal.

obteniendo para cada una de las ejecuciones unas métricas de error que serán objeto de comparación más adelante.

Con el objeto de hacer la comparativa lo más justa posible, el criterio que se ha utilizado a la hora de seleccionar los mejores parámetros del modelo LSTM es la precisión de predicción en el conjunto de entrenamiento. Tras seleccionar los mejores conjuntos de parámetros según este criterio, se ha procedido a reevaluar los modelos en el conjunto de test. De este modo el conjunto de test no interviene en el entrenamiento, como es natural, pero tampoco interviene en la selección del modelo, simulando con mayor precisión un escenario real cuyo objetivo sea el de predecir valores futuros.

A. Métricas de error

En la elaboración de esta experimentación se han considerado 2 métricas de error distintas: La raíz del error cuadrático medio (*rmse*); y el error porcentual absoluto medio (*mape*).

Las métricas de error cuantifican el desvío de una muestra (en este caso, representa la predicción realizada por nuestros modelos) con respecto de otra (los valores reales para la serie temporal).

Siendo E el vector de valores esperados y P el vector de valores predichos, teniendo E y P el mismo número de valores, ambas métricas se definen como sigue:

1) RMSE:

$$RMSE(E,P) = \sqrt{MSE(E,P)} \tag{1}$$

$$MSE(E,P) = \frac{\sum_{i=1}^{n}(E_i - P_i)^2}{n} \tag{2}$$

2) MAPE:

$$MAPE(E,P) = \frac{100}{n}\sum_{i=1}^{n}\frac{E_i - P_i}{E_i} \tag{3}$$

B. Conjunto de datos

Los datos empleados se han obtenido del repositorio público de Instituto Nacional de Estadística. En concreto, se han utilizado los relativos a la Encuesta de Ocupación Hotelera, serie correspondiente al grado de ocupación por plazas de la provincia de Jaén. Estos datos han sido agrupados por la fuente en intervalos de tiempo de un mes, siendo este intervalo la resolución de la que disponemos. En la fig. 2 se puede ver una representación de los datos obtenidos.

Como cabe esperar de una serie temporal relacionada con el sector turístico, se puede apreciar a simple vista una periodicidad de 12 meses en los valores, además de picos de ocupación en los meses de marzo-abril y julio-agosto, correspondiente con los períodos vacacionales de semana santa y verano.

De todos los datos disponibles, se han utilizado los comprendidos entre enero de 2012 y diciembre de 2016. Posteriormente, se han construido 2 conjuntos de datos: El primero de ellos es el conjunto de entrenamiento, y está formado por los datos comprendidos entre enero de 2013 y diciembre de 2015. El resto de datos, los comprendidos entre enero y diciembre de 2016 compondrán el conjunto de datos de test.

A las series temporales extraídas se les han añadido regresores externos adicionales para aumentar la eficiencia de los métodos. Estas columnas adicionales se han decidido con cuidado de no introducir incertidumbre en el conjunto de datos, considerando sólo aquellos regresores que se puedan obtener de forma precisa de antemano. En concreto, se ha añadido información sobre:

- Año
- Mes del año
- Semana santa: Indica si el mes en cuestión contiene parte de la semana santa de ese año.
- Puentes largos: Número de festivos en días martes o jueves en el mes dado.
- Puentes cortos: Número de festivos en días lunes o viernes en el mes dado.

Si bien esta es la estructura general de los conjuntos de datos de partida, por exigencias de cada implementación la estructura se ha adaptado en cada caso:

LSTM: Cada instancia suministrada a la red durante el entrenamiento tiene la forma $(V_n, V_{n-1}, V_{n-11}, V_{n-12}, SS, PC, PL)$, mientras que para la predicción se elimina el valor V_n por motivos obvios.

ARIMA: En el caso de ARIMA, cada instancia tiene la forma $(V_n, V_{n-12}, SS, PC, PL)$

Siendo:

n El mes correspondiente a la instancia en cuestión.
V_i El valor de la serie temporal correspondiente al mes i-ésimo.
SS Indica si en ese mes tiene lugar la Semana Santa.
PC El número de puentes cortos del mes correspondiente.
PL El número de puentes largos del mes correspondiente.

En el caso de la LSTM, se ha optado por añadir el valor de la serie temporal de mes anterior, del año anterior, y del undécimo mes anterior. Esta selección de retardos obedece a los resultados de un análisis preliminar de autocorrelación de la serie temporal, fig. 3, que muestra que estos valores son los más significativos.

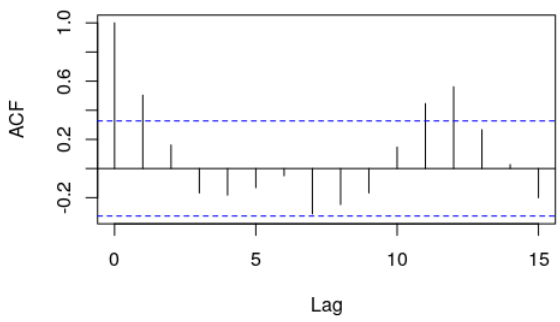


Fig. 3. Función de autocorrelación.

C. Métodos

La experimentación con las redes neuronales recurrentes de tipo LSTM se ha implementado en *Python*, utilizando la librería *Keras* con el backend *TensorFlow*, configurado por defecto. Asimismo, se ha utilizado la librería *Pandas* para la gestión de los datos, *Numpy* para el cálculo numérico, *Matplotlib* para la visualización de los resultados y *Scikit-learn* para el cálculo de métricas de error, así como la normalización de los datos de entrada.

A pesar de que las redes neuronales recurrentes LSTM son un modelo de tipo *deep learning*, el tiempo de entrenamiento por modelo en esta experimentación no ha sido excesivo.

Una vez realizadas ejecuciones preliminares explorando distintas zonas del espacio de configuraciones de parámetros, a la vista de las métricas de error obtenidas, se ha procedido a seleccionar la región del espacio donde encontraremos los modelos que calculen las mejores predicciones:

- Número de LSTM: 25, 50 y 100
- Iteraciones: 100 y 200
- *batch_size*: 1 y 2

Dado que el algoritmo utilizado para el entrenamiento de las redes LSTM, BackPropagation Through Time, parte de un estado inicial aleatorio, para reducir el efecto del azar de los elementos del estado inicial de la red, se han realizado 20 ejecuciones con cada una de las combinaciones de parámetros del espacio de búsqueda reducido definido anteriormente. De esta forma el error medio obtenido está muy escasamente influenciado por algunas ejecuciones cuyos resultados pueden verse desviados por motivos del azar.

Para el entrenamiento del modelo ARIMA se ha utilizado el entorno R de computación estadística, en lugar del lenguaje Python, como se ha hecho anteriormente. Esto es debido a que la implementación de referencia del modelo ARIMA está desarrollada en este lenguaje. En concreto, se ha utilizado la función *auto.arima* del paquete *forecast* [4].

El programa de entrenamiento del modelo ARIMA es visiblemente más simple que en el caso de la red LSTM. Esto es debido a que todo el análisis de parámetros y entrenamiento del modelo se realiza de forma automática por la implementación elegida, utilizando técnicas como las descritas en los trabajos [10], [11]. La existencia de distintas heurísticas

TABLA I
Resultados ejecuciones LSTM - RMSE Entrenamiento

Num LSTM	Iterac.	batch size	RMSE	
			Promedio	Desv. Est.
25	200	1	2,17954	0,01534
50	200	1	2,18941	0,01802
25	200	2	2,19675	0,01309
50	200	2	2,20690	0,00849
100	200	1	2,21396	0,03308
100	200	2	2,22174	0,01000
25	100	1	2,22621	0,01257
50	100	1	2,23581	0,01208
25	100	2	2,23769	0,01105
100	100	1	2,24942	0,02027
50	100	2	2,24990	0,01107
100	100	2	2,27130	0,01539

TABLA II
Resultado ejecución ARIMA

Entrenamiento	Test	
RMSE	RMSE	MAPE
2,40561	1,89010	5,01111

hace posible que este proceso se pueda automatizar teniendo en cuenta métricas comparativas como *Akaike information criterion* [12]

Además, dado que ni ARIMA ni las heurísticas que buscan los mejores parámetros son probabilísticos, no es necesario realizar varias ejecuciones del algoritmo, ya que todas arrojarían exactamente los mismos resultados.

IV. Resultados

En la tabla I se pueden ver los resultados de las mejores ejecuciones conseguidas con redes neuronales de tipo LSTM. En ella se muestra, para la métrica de error RMSE en la partición de entrenamiento del conjunto de datos, el promedio y la desviación típica de 20 repeticiones completas del experimento. Los distintos modelos se muetran en la tabla ordenados según el error cometido en la experimentación. A la vista de estos resultados, seleccionamos el modelo de menor error para la comparativa con ARIMA.

En la tabla II se muestra el resultado de la ejecución realizada con el modelo ARIMA. Como se ha introducido antes, dado que este modelo no es probabilístico, carece de sentido realizar varias repeticiones, ya que todas predicen exactamente igual, por lo que las métricas obtenidas serán idénticas.

La tabla comparativa III se muestran los resultados obtenidos en la partición de test, tanto para el modelo ARIMA como para el mejor modelo LSTM, el cual se ha seleccionado siguiendo el criterio del menor error en la partición de entrenamiento (tabla I). Es fácil observar de esta forma que LSTM obtiene mejores resultados que ARIMA en ambas métricas de error consideradas.

V. Conclusión

En este artículo se ha realizado una comparativa entre un nuevo modelo de red neuronal recurrente, LSTM, y el algoritmo considerado estado del arte en predicción de series temporales, ARIMA, sobre un problema de predicción de series temporales en turismo, justificado por la importancia del sector turístico en la economía mundial.

TABLA III
Comparativa ARIMA - LSTM

	RMSE	MAPE
LSTM	**1,70039**	**4,61407**
ARIMA	1,89010	5,01111

Los resultados de la experimentación llevada a cabo, resumidos en la tabla III, indican que un modelo de red neuronal de tipo LSTM, seleccionado basándonos exclusivamente en datos de entrenamiento, obtiene una mayor precisión en la predicción de los datos de test que el modelo ARIMA clásico, ampliamente utilizado en el campo de series temporales.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de España de Ciencia y Tecnología bajo el proyecto TIN2015-68454-R.

Referencias

[1] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Advances in neural information processing systems*, 1997, pp. 473–479.
[2] C. Altés, "Marketing y turismo," *Madrid: Editorial Síntesis*, 1993.
[3] J. R. Cuadrado Roura, J. M. López Morales *et al.*, "El turismo, motor del crecimiento y de la recuperación de la economía española," 2015.
[4] R. J. Hyndman, Y. Khandakar *et al.*, *Automatic time series for forecasting: the forecast package for R*. Monash University, Department of Econometrics and Business Statistics, 2007, no. 6/07.
[5] V. V. Romanjuk, "Training data expansion and boosting of convolutional neural networks for reducing the mnist dataset error rate," *Naukovi Visti NTUU KPI*, no. 6, pp. 29–34, 2016.
[6] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.
[7] J. Schmidhuber and S. Hochreiter, "Long short-term memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.
[8] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," 1999.
[9] J. Brownlee, "Multivariate time series forecasting with lstms in keras-machine learning mastery," *Machine Learning Mastery*, 2017.
[10] E. J. Hannan and J. Rissanen, "Recursive estimation of mixed autoregressive-moving average order," *Biometrika*, vol. 69, no. 1, pp. 81–94, 1982.
[11] V. Gómez, *Automatic model identification in the presence of missing observations and outliers*. Ministerio de Economía y Hacienda, Dirección General de Análisis y Programación Presupuestaria, 1998.
[12] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected Papers of Hirotugu Akaike*. Springer, 1998, pp. 199–213.
[13] J. E. Ball, D. T. Anderson, and C. S. Chan, "Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community," *Journal of Applied Remote Sensing*, vol. 11, no. 4, p. 042609, 2017.

Detección de cáncer de piel usando técnicas de aprendizaje profundo

1st Alejandro Polvillo Hall
Departamento de Datos Geographica
Sevilla, España
alejandro@geographica.gs

2nd Juan A. Álvarez-García
Dpto. de Lenguajes y Sistemas Informáticos Universidad de Sevilla
Sevilla, España
jaalvarez@us.es

3rd Cristina Rubio-Escudero
Dpto. de Lenguajes y Sistemas Informáticos Universidad de Sevilla
Sevilla, España
crubioescudero@us.es

***Resumen*—El aprendizaje profundo ha sido muy utilizado para la clasificación de imágenes a partir de la competición ImageNet en 2012. Esta clasificación de imágenes es de gran utilidad en el campo de la medicina, en el que ha habido un gran crecimiento de uso de técnicas de minería de datos en los últimos años. En este trabajo seleccionamos y entrenamos una red de aprendizaje profundo para el análisis de un conjunto de datos de cáncer de piel, obteniendo resultados muy satisfactorios, ya que el modelo ha superado los resultados de clasificación de dermatólogos entrenados haciendo uso de un dermatoscopio, de otras técnicas de aprendizaje automático, y de otras técnicas de aprendizaje profundo.**

***Index Terms*—aprendizaje profundo, imágenes médicas, análisis de datos clínicos**

I. Introducción

El aprendizaje profundo ha sido utilizado en el campo de la visión por computadora durante décadas [5], [9]. Sin embargo, su verdadero valor no se había descubierto hasta la competición de ImageNet en 2012 [8], un éxito que provocó una revolución a través del uso eficiente del procesamiento de unidades gráficas (GPU). El principal poder del aprendizaje profundo radica en su arquitectura [10], [11], que permite discriminación en múltiples niveles de abstracción para un conjunto de características.

Las técnicas de aprendizaje profundo han sido utilizadas con éxito en campos como el de la medicina, en el que el aprendizaje profundo viene a solucionar problemas que presentan los algoritmos de aprendizaje automático con algunas estructuras de datos muy utilizadas en medicina como son las imágenes. La minería de datos clínicos es la aplicación de técnicas de minería de datos a los datos clínicos, con el objetivo de interpretar los datos disponibles. Permite la creación de modelos de conocimiento y proporciona asistencia para la toma de decisiones clínicas. En los últimos 10 años, ha habido un interés creciente en la aplicación de técnicas de minería de datos a los datos clínicos. MEDLINE ha visto un fuerte aumento de factor 10 en el número de trabajos con el término ”minería de datos.[en] su título [7].

Para hacer un entrenamiento completo, el aprendizaje profundo requiere una gran cantidad de datos de entrenamiento etiquetados, un requisito que puede ser difícil para cumplir en el campo de la medicina, donde la anotación de expertos es costosa y las enfermedades (por ejemplo, lesiones) no cuentan con grandes conjuntos de datos. Además, requiere una gran cantidad de recursos computacionales para que el entrenamiento no sea excesivamente lento.

En este trabajo hemos aplicado técnicas de aprendizaje profundo para analizar un conjunto de datos de imágenes de cáncer de piel, obteniendo resultados muy satisfactorios.

En las siguientes secciones describimos las metodologías y resultados obtenidos.

II. Metodología

En esta sección se describen las metodologías utilizadas para el desarrollo de este trabajo.

II-A. Tensorflow

Tensorflow [1] es uno de los mejores frameworks de aprendizaje profundo existente. Ha sido adoptado por un montón de grandes empresas como Airbus, Twitter, IBM y otras más debido a su flexibilidad y polivalencia. Tensorflow es desarrollado por Google, que la usa en todos sus proyectos de aprendizaje automático y aprendizaje profundo.

Tensorflow no es en sí un framework de aprendizaje profundo, es un framework que te permite trabajar de forma muy rápida con matrices gracias a su paralelización en GPU's. Como casi todos los cálculos realizados para entrenar y predecir con una red neuronal son cálculos matriciales hacen de esta herramienta que sea ideal para usarla en la construcción de redes neuronales.

Tensorflow tiene un paquete interno que viene con la funcionalidad necesaria para hacer funcionar una red neuronal convolucional: Capas convolucionales, optimizadores, funciones de optimización, etc...

Una desventaja de Tensorflow es que es necesario escribir mucho código para conseguir algo funcional. El hecho de escribir todo ese código hace que tengas un control total sobre todos los elementos de la arquitectura de la red neuronal. Aunque también puede hacer que cometas muchos errores.

II-B. Keras

Keras [2] es un framework específico de aprendizaje profundo. No es competidor de Tensorflow, pues Keras se ejecuta "encima" de Tensorflow. Keras provee una sintaxis extremadamente fácil para la creación de redes neuronales, y después convierte esta sintaxis a modelos de Tensorflow, usando la potencia de éste para ejecutar toda la maquinaria de aprendizaje.

Debido a que nuestro caso de uso será un caso de uso de red neuronal, y no necesitaremos en principio la manipulación a muy bajo nivel de los modelos de aprendizaje profundo, nosotros optamos por el uso de Keras para el desarrollo del presente proyecto.

III. Resultados

El conjunto de datos seleccionado para evaluar nuestra propuesta ha sido el conjunto de imáganenes de cáncer de piel de The International Skin Imaging Collaboration (ISIC) (https://isic-archive.com/). Es una plataforma que intenta aunar a los profesionales dermatológicos con el objetivo de luchar contra el cáncer de piel. El conjunto de datos está formado por 23906 imágenes. Además de las imágenes, se proporcionan algunos metadatos entre los que destacamos edad, sexo, lugar anatómico del melanoma, tipo de diagnóstico, clase de melanoma y espesor del melanoma.

En primer lugar analizamos el conjunto de datos con respecto a la clase benigno/maligno. Como se puede observar en la Figura 1 está muy desbalanceado.

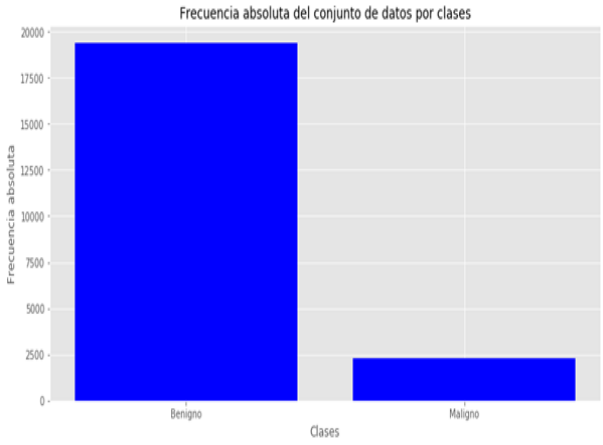


Figura 1. Conjunto de datos de cáncer de piel desbalanceado.

Debido a esto recurrimos a una técnica de remuestreo sobre la clase minoritaria que se encarga de escoger elementos al azar con reemplazo y añadirlos a la clase minoritaria. La ventaja de ésta técnica es que no elimina información, simplemente refuerza la información existente sobre la clase minoritaria. Tras aplicar la técnica el resultado obtenido se puede ver en la Figura 2.

En segundo lugar, partimos el conjunto de datos en entrenamiento (80 %) y test (20 %) asegurándonos de mantener balanceadas las clases en cada uno de los conjuntos.

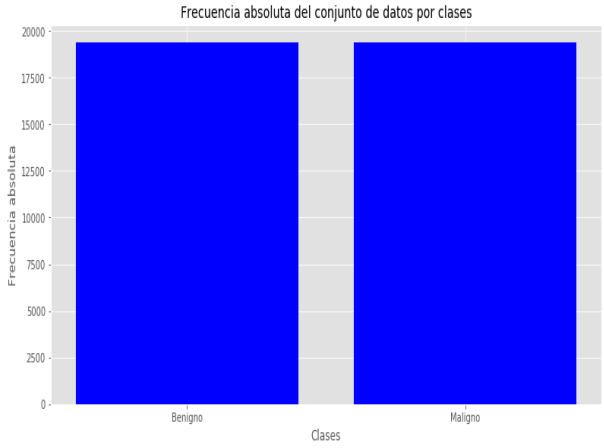


Figura 2. Conjunto de datos de cáncer de piel después del balanceo.

En tercer lugar, aplicamos un mecanismo de aumento de datos consistente en realizar distintas transformaciones aleatorias sobre cada uno de los elementos del conjunto de entrenamiento. Al realizar transformaciones aleatorias sobre el conjunto de datos conseguimos un doble efecto: aumentar el tamaño de nuestros datos y hacer que el sistema generalice mucho mejor. En particular aplicamos las técnicas conocidas como rotación automática, traslación vertical y horizontal de píxeles, cizallamiento, zoom, volteo, filtro de sal y pimienta.

III-A. Selección y aplicación del modelo de aprendizaje profundo

A la hora de enfrentarse a la selección del modelo nos encontramos en un momento de incertidumbre máxima debido a que son muchos los parámetros que tenemos que elegir, entre estos parámetros se encuentran:

- Arquitectura de la red neuronal: Propia, InceptionV3, VGG16, VGG19, Xception, etc...
- Tipo de entrenamiento: Transferencia de conocimiento, desde cero, etc…
- Parámetros de la transferencia de conocimiento: capas a entrenar, entrenamiento completo, etc...
- Algoritmo de aprendizaje: Descenso del gradiente, RMSProp, Adagrad, etc...
- Parámetros del algoritmo de aprendizaje: Tasa de aprendizaje, funciones de activación, épocas, lotes, etc…

Para seleccionar los parámetros adecuados hemos utilizado la búsqueda por rejilla (o grid search en inglés). Esta técnica se basa en la definición de un conjunto de posibles valores que pueden tomar los parámetros, y generar el producto cartesiano de todos los valores de todos los parámetros entre sí, usando cada uno de estos conjuntos de parámetros generados para construir un modelo y evaluar la bondad. En la Tabla I podemos ver todos los parámetros que se han considerado a la hora de buscar el modelo más adecuado.

En nuestra búsqueda rejilla se han fijado 3 parámetros: épocas, lotes y funciones de activación. Las épocas se han fijado puesto que es obvio que a más épocas mejor van a

Cuadro I
Parámetros considerados en la búsqueda del modelo más adecuado

Parámetro	Valores
Modelo	Xception, VGG16, VGG19 y InceptionV3
Tipo de entrenamiento	Transferencia de conocimiento, desde cero
Capas a entrenar	1, 2, 3 y todas
Algoritmo de aprendizaje	Descenso del gradiente, RMSProp, Adam, Adagrad
Tasa de aprendizaje	0.00001, 0.0001, 0.001, 0.01 y 0.1
Épocas	2
Lotes	10
Funciones de activación	ReLu

Cuadro II
Modelo y parámetros seleccionados

Parámetro	Valores
Modelo	InceptionV3
Tipo de entrenamiento	Transferencia de conocimiento
Capas a entrenar	todas
Algoritmo de aprendizaje	Adam
Tasa de aprendizaje	0.0001
Épocas	2
Lotes	10
Funciones de activación	ReLu

funcionar los modelos en su mayoría, así que se fijan a 2 épocas, para que así todas puedan recorrer el conjunto de datos 2 veces. Los lotes van de la mano de la capacidad de la unidad de procesamiento gráfico, aunque pueden influir en el entrenamiento, en este caso se han hecho los cálculos para que el conjunto de lotes sea el máximo que la unidad de procesamiento gráfica puede soportar, así hacemos que el tiempo de búsqueda por rejilla sea menor. La función de activación la he fijado a la función de activación ReLu debido a que todas las arquitecturas usan la ReLu debido a que el aprendizaje se realiza de una manera más rápida con este tipo de función de activación.

El modelo y parámetros que mejor desempeño han tenido, obteniendo un 81 % de exactitud se puede ver en la Tabla II:

El modelo seleccionado como óptimo es el modelo de Inception V3, creado por Google. Tal y como se describe en [12] era de esperar que se seleccionara la transferencia de conocimiento en vez del entrenamiento desde cero, pues posiblemente tenga un desempeño mayor. También se recomienda que, aunque se parta de unos pesos definidos (por la transferencia de conocimiento), se permita el reajuste de algunos pesos en las capas convolucionales, con el objetivo de que se adapte a nuestro problema. Como algoritmo de aprendizaje se usa Adam, con una tasa bastante pequeña. Esta tasa tan pequeña es debido a que gracias a la transferencia de conocimiento estamos muy cerca del óptimo, queremos dar pasos muy pequeños para acercarnos cada vez más al óptimo.

Una vez con nuestro modelo y parámetros definidos, procedemos a aumentar el número de épocas con el objetivo de obtener un mejor modelo y poder evaluar en la siguiente sección. Una vez el entrenamiento del modelo ha finalizado, hemos aplicado el clasificador sobre el conjunto de pruebas, obteniendo los resultados que se pueden ver en la Figura 3.

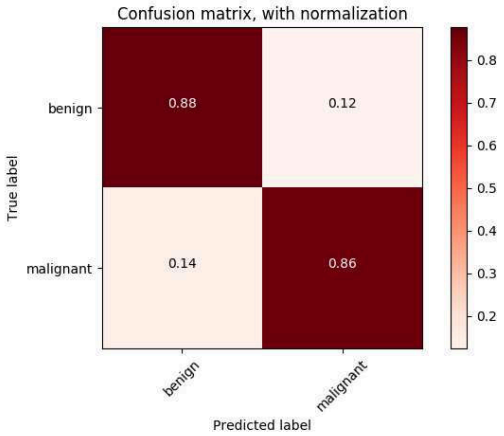


Figura 3. Matriz de confusión normalizada.

III-B. Evaluación del modelo

A continuación hablamos sobre la evaluación del modelo. La exactitud es la métrica por defecto que usa Keras para representar la bondad del modelo a lo largo de las distintas épocas de entrenamiento. En la Figura 4 podemos observar cómo se comporta el clasificador respecto a la exactitud a lo largo de las distintas épocas para los conjuntos de validación y de entrenamiento.

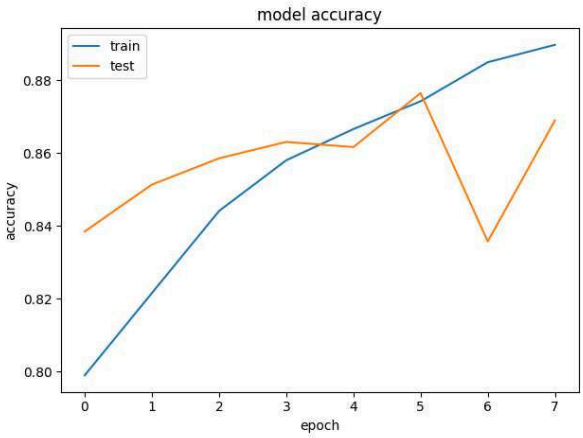


Figura 4. Exactitud del modelo a lo largo de las distintas épocas.

Como se puede observar en la gráfica siempre se encuentran en crecimiento, exceptuando en la época 6 que el conjunto de prueba hace algo difícil de interpretar, pero en la siguiente época se recupera.

El clasificador se ha puesto a entrenar durante 10 épocas, pero hemos activado una opción que permite ahorrar recursos. Esta opción consiste en que, si el clasificador durante la etapa de clasificación comienza a permanecer durante algún tiempo sin variar su exactitud, se pare la etapa de aprendizaje cuando se acabe la época. Por eso se puede observar que, aunque se haya puesto a entrenar durante 10 épocas, en la gráfica sólo se observan 8.

Cuadro III
Resultados obtenidos por nuestro modelo de aprendizaje profundo.

Medida	Valor
Exactitud	86.90 %
Precisión	87.47 %
Sensitividad	86.14 %
Especificidad	87.66 %
Área ROC	0.87

Se proporcionan también las medidas que se pueden ver en la Tabla III con respecto al desempeño de nuestro modelo de aprendizaje profundo.

La Figura 5 representa el área bajo la curva ROC

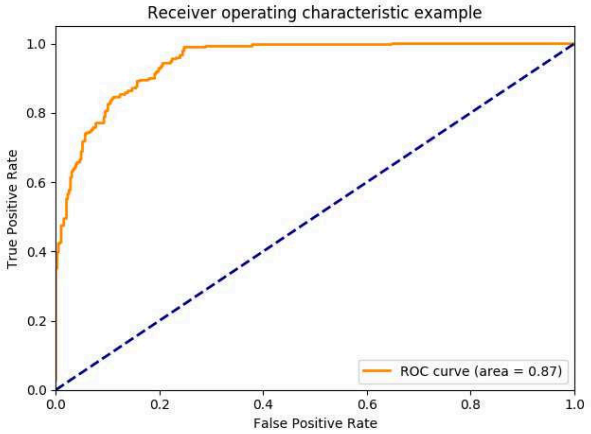


Figura 5. Área bajo la curva ROC.

Las inspecciones visuales sin ayuda de los expertos dermatológicos obtienen una media de un 60 % de exactitud [3], debido a la complejidad de observación de características diferenciadoras a simple vista. Con la ayuda de un experto entrenado junto a un dermatoscopio, la exactitud puede aumentar hasta un 75 % - 84 %.

Partiendo de la base de las afirmaciones anteriormente expuestas, y de las métricas obtenidas por nuestro clasificador, podemos observar a simple vista que nuestro clasificador ha superado en acierto a un experto dermatológico entrenado haciendo uso de un dermatoscopio. Esto parece un buen punto de partida.

Sabiendo que nuestro clasificador es capaz de superar a un experto dermatólogo, procedemos ahora a comparar nuestro clasificador con otras técnicas.

La primera intuición al intentar aplicar técnicas de inteligencia artificial en el análisis de imágenes médicas es usar aprendizaje automático o aprendizaje profundo. Con lo cual usaremos estos dos enfoques con el objetivo de comparar nuestro clasificador y ver qué bueno o malo es respecto a otros clasificadores

En primer lugar, nos centraremos en la comparación de nuestro clasificador con las técnicas de aprendizaje automático.

Cuadro IV
Comparación de resultados entre aprendizaje profundo y aprendizaje automático.

Medida	Nuestro clasificador	Aprendizaje automático
Exactitud	86.90 %	77 %
Precisión	87.47 %	71 %
Sensitividad	86.14 %	73 %

Cuadro V
Comparativa entre resultados de nuestro aprendizaje profundo y otro aprendizaje profundo.

Medida	Nuestro clasificador	Otro clasificador
Exactitud	86.90 %	85.5 %
Precisión	87.47 %	63.7 %
Sensitividad	86.14 %	50.7 %
Especificidad	87.66 %	94.1 %
Área ROC	0.87	0.8

El aprendizaje automático sobre el conjunto de datos de ISIC ha obtenido las métricas que se pueden ver en la Tabla IV [4].

Como se puede observar en la tabla, nuestro clasificador supera con creces a los clasificadores de aprendizaje automático enfocados al análisis de imágenes médicas para el conjunto de datos de ISIC. el resultado es lógico, pues los algoritmos de aprendizaje automático no están pensados para extraer características de imágenes, por lo que siempre se presupone un rendimiento menor.

En segundo, compararemos nuestro clasificador con otro clasificador de aprendizaje profundo [6]. El clasificador con el que compararemos nuestro clasificador, es un clasificador usado por un equipo participante de la competición de ISIC. Los resultados se pueden ver en la Tabla V:

Se puede observar, que, aunque el desempeño en la exactitud en ambos clasificadores es parecido, nuestro clasificador se comporta mucho mejor en la mayoría de situaciones. Ellos han conseguido obtener una especificidad muy alta, que es lo que le permite compensar la sensibilidad para obtener una exactitud y AUC ROC bastante aceptables.

IV. Conclusiones

En este trabajo hemos presentado el resultado de aplicar técnicas de aprendizaje profundo a un conjunto de datos de imágenes de cáncer de piel. Se ha descrito la fase de preprocesamiento de los datos, selección de parámetros para el modelo seleccionado y se han calculado distintas métricas para los resultados obtenidos.

En las métricas se pueden observar que el clasificador es bastante homogéneo, y no hay resultados de difícil interpretación. El clasificador tiene un nivel de acierto en ambas clases bastante estable, lo que nos hace indicar que ha generalizado correctamente la detección de cáncer en las imágenes. Nuestro modelo ha superado los resultados de clasificación de dermatólogo entrenado haciendo uso de un dermatoscopio, de otras técnicas de aprendizaje automático, y de otras técnicas de aprendizaje profundo.

En definitiva, el clasificador se ha comportado de manera muy buena frente a este conjunto de imágenes respecto a

otros clasificadores y técnicas. Aún sabiendo que las imágenes médicas son complicadas de tratar, el clasificador ha sabido generalizar los detalles que separan una imagen de un paciente que padece cáncer de piel de otro que no lo padece, consiguiendo así predecir con exactitud en la mayoría de casos.

Agradecimientos

Este trabajo ha sido financiado por los proyecto del Ministerio de Economía y Competitividad TIN2014-55894-C2-1-R y TIN2017-82113-C2-1-R.

Referencias

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] François Chollet et al. Keras, 2015.

[3] Noel CF Codella, David Gutman, M Emre Celebi, Brian Helba, Michael A Marchetti, Stephen W Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, et al. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In *Biomedical Imaging (ISBI 2018), 2018 IEEE 15th International Symposium on*, pages 168–172. IEEE, 2018.

[4] Machine Learning for ISIC Skin Cancer Classification Challenge. https://hackernoon.com/machine-learning-for-isic-skin-cancer-classification-challenge-part-1-ccddea4ec44a.

[5] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[6] David Gutman, Noel CF Codella, Emre Celebi, Brian Helba, Michael Marchetti, Nabin Mishra, and Allan Halpern. Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (isbi) 2016, hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1605.01397*, 2016.

[7] Jimison Iavindrasana, Gilles Cohen, Adrien Depeursinge, H Müller, R Meyer, and Antoine Geissbuhler. Clinical data mining: a review. *Yearbook of medical informatics*, 18(01):121–133, 2009.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015.

[12] Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312, 2016.

A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines

David Charte
Dept. CS and AI
University of Granada
Granada, España
fdavidcl@ugr.es

Francisco Charte
Dept. CS
University of Jaén
Jaén, España
fcharte@ujaen.es

Salvador García
Dept. CS and AI
University of Granada
Granada, España
salvagl@decsai.ugr.es

María J. del Jesus
Dept. CS
University of Jaén
Jaén, España
mjjesus@ujaen.es

Francisco Herrera
Dept. CS and AI
University of Granada
Granada, España
herrera@decsai.ugr.es

***Abstract*—This is a summary of our article published in Information Fusion [1] to be part of the CAEPIA-18 Key Works.**

***Index Terms*—Autoencoders, Feature fusion, Feature extraction, Representation learning, Deep Learning, Machine Learning**

I. Summary

The performance of most machine learning algorithms depends on the quality of the input data, and in particular on its features. Many techniques exist which combine their information into a new feature set, with the aim of improving learned models. This new set is usually of lower dimension and contains more abstract variables. Procedures which result in new feature sets can be named after several terms: feature engineering, feature learning, representation learning, feature selection, feature extraction and feature fusion. Manifold learning algorithms, especially those based on artificial neural networks (ANNs), fall into the last category.

Autoencoders (AEs) are feedforward ANNs with a symmetric *encoder-decoder* structure (see Fig. 1), where the middle layer represents an encoding of the input data. They are trained to reconstruct their inputs onto their output layer while some restrictions prevent them from copying the data along. AE components can generally be represented as functions: an encoder f and a decoder g which map inputs x to outputs r: $r = g(f(x))$. The desired encoding would be $y = f(x)$.

Typical activation functions such as linear, binary or ReLU are of limited use in AEs, while logistic, hyperbolic tangent and SELU are more common. According to whether the dimension of their encoding is lower or higher than that of the data, AEs can be either *undercomplete* or *overcomplete*, respectively. If they have just one hidden layer they are called *shallow*, and otherwise they are considered *deep*.

A taxonomy of the main variants of AEs has been proposed according to the properties of the inferred model:

This work is supported by the Spanish National Research Projects TIN2015-68454-R and TIN2014-57251-P, and Project BigDaP-TOOLS - Ayudas Fundación BBVA a Equipos de Investigación Científica 2016.

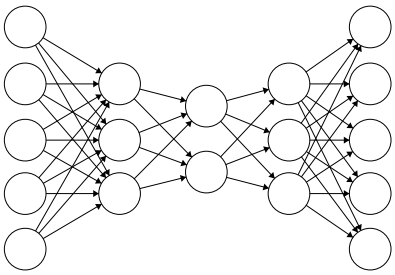


Fig. 1. A possible neural architecture for an autoencoder.

- Lower dimensionality
  - Basic [2]: the basis for most autoencoders, a symmetric feedforward ANN with a distance between the outputs and the inputs as the objective function, trainable with stochastic variants of gradient descent (SGD, AdaGrad, RMSProp, Adam...). Training can be done layer by layer in a stacked fashion.
  - Convolutional [3]: layers in this variant are convolutional, which explicitly consider a 2-dimensional structure for processing images.
  - LSTM-based [4]: models sequential data by placing Long-Short-Term Memory units as encoders and decoders.
- Regularization
  - Sparse [5]: introduces a penalty term for encodings with many activations, thus resulting in a low activation average. The Kullback-Leibler divergence can be used to attract the AE to the desired average activation value.
  - Contractive [6]: achieves local invariance to changes in many directions around the training samples by adding a penalty term based on the Frobenius norm of the Jacobian matrix of the encoder.
- Noise tolerance
  - Denoising [7]: its generated features are less sensitive to noise by training with corrupted versions of

input data.
  - Robust [8]: uses a special loss function, correntropy, in order to build more robust features. Correntropy measures the probability density that two events are equal, and is less affected by outliers than other distance metrics.

- Generative model
  - Variational [9]: applies a variational Bayesian approach to encoding, assuming a latent, unobserved random variable generates the observations. It tries to approximate the distribution of the latent variable given the data.
  - Adversarial [10]: brings adversarial learning to AEs, simultaneously training a discriminator and a generator (the encoder), each competing with the other.

AEs are also the inspiration for other, more complex, neural structures. For example, autoencoder trees which involve decision trees, dual-autoencoders which learn two latent representations, and recursive AEs which introduce more pieces of input as the model deepens.

AEs can be related to the wide range of existing feature fusion methods: in the linear case, they are equivalent to PCA when linear activations and mean squared error loss function are used. As a consequence, they can be seen as generalizations of PCA which allow nonlinearities and other objective functions. They are more easily applicable than Kernel PCA since in that case the choice of kernel highly alters the behavior of the model. Other nonlinear approaches such as Isomap and Locally Linear Embedding can be compared to the contractive AE, as they attempt to preserve the local structure of the data. Restricted Boltzmann Machines are alternatives to AEs for greedy layer-wise initialization, but their models are hard to simulate than those of AEs.

The usual purpose of AEs is performing feature fusion in order to improve classification and regression performance, and to facilitate other unsupervised tasks which can be hard in high-dimensional scenarios, such as clustering. Most applications of AEs can be summarized in the following categories:

- Classification: reducing or transforming the training data in order to achieve better performance in a classifier.
- Data compression: training AEs for specific types of data to learn efficient compressions.
- Detection of abnormal patterns: identification of discordant in- stances by analyzing generated encodings.
- Hashing: summarizing input data onto a binary vector for faster search.
- Visualization: projecting data onto 2 or 3 dimensions with an AE for graphical representation.
- Other purposes: reconstruction of deteriorated images, noise reduction in automatic speech recognition, curation of biological databases, tagging digital resources.

In [1] we propose some guidelines which can help when designing AEs for different tasks. The following is an overview of these:

- Architecture: a starting point is choosing the desired length of the encoding and the type of units for the neural structure.
- Activations and loss function: sigmoid-like functions generally work well in the encoding layer. Place linear or ReLU at the output when using mean squared error, or logistic activation when using cross-entropy error.
- Regularizations: can be used according to the desired properties in the encoding. Weight decay can prevent overfitting, sparsity can be useful in many scenarios and contraction can find a lower-dimensional manifold.

The main software libraries and frameworks which allow for the construction and training of AEs are Tensorflow, Caffe, Torch, MXNet, and Keras. Software pieces which specifically implement AEs are packages Autoencoder and SAENET of the CRAN repository, H2O and yadlt for Python.

A case study is provided in [1] with the well known MNIST handwritten digits dataset, covering several adjustable parameters such as the encoding length, the optimization algorithm, activation functions and some of the predominant AE variants.

References

[1] D. Charte, F. Charte, S. García, M. J. del Jesus, and F. Herrera, "A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines," *Information Fusion*, vol. 44, pp. 78 – 96, 2018.

[2] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological cybernetics*, vol. 59, no. 4, pp. 291–294, 1988.

[3] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Artificial Neural Networks and Machine Learning – ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds. Springer Berlin Heidelberg, 2011, pp. 52–59.

[4] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using LSTMs," in *International Conference on Machine Learning*, 2015, pp. 843–852.

[5] A. Ng, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[6] S. Rifai, Y. Bengio, P. Vincent, and Y. N. Dauphin, "A generative process for sampling contractive auto-encoders," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ser. ICML'12. Omnipress, 2012, pp. 1811–1818.

[7] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08. ACM, 2008, pp. 1096–1103.

[8] Y. Qi, Y. Wang, X. Zheng, and Z. Wu, "Robust feature learning by stacked autoencoder with maximum correntropy criterion," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6716–6720.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[10] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.