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.

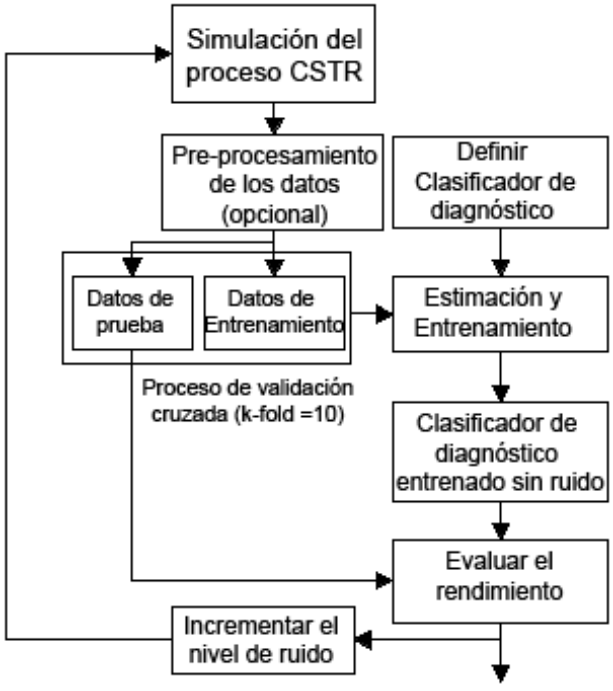


Figura 1. Flujograma empleado para los experimentos en el CSTR.

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.

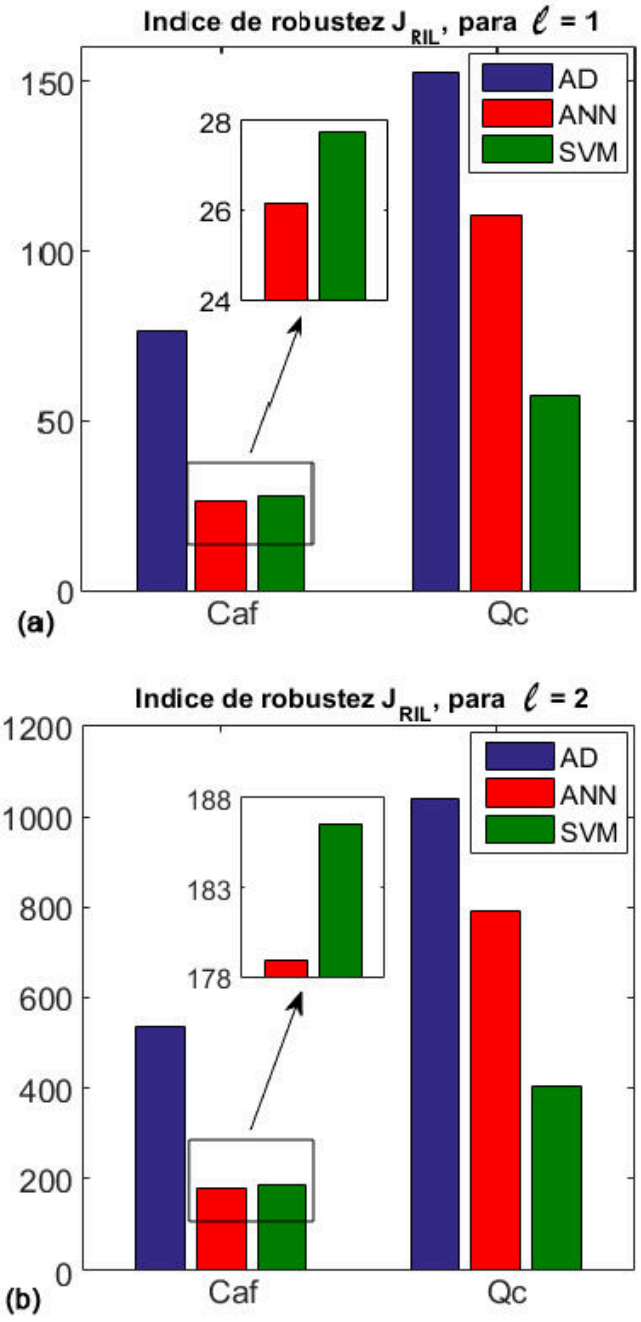


Figura 2. Comparando clasificadores mediante el índice de robustez J_{RIL}.

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.

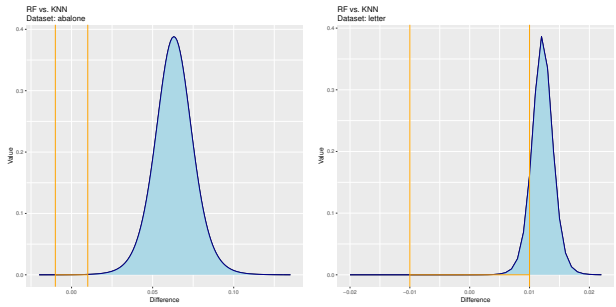
Vemos en la tabla Tabla III que la hipótesis nula $\mathcal{H}_0 : \mu_{LR} = \mu_{KNN}$ puede rechazarse debido a que el p-valor asintótico es menor que $0{,}05$, de manera que este test identifica una diferencia significativa entre estos dos algoritmos. Para determinar cuál obtiene mejores resultados, 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.

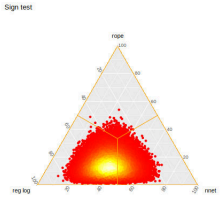
III-B4. Test bayesiano de signo: Para este test usamos los resultados promediados de dos algoritmos en todos los conjuntos de datos.

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`.

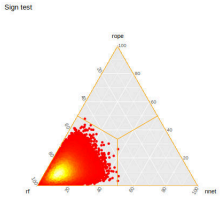
III-B5. Test bayesiano de rangos con signo: Repetimos la experimentación usando el test bayesiano de rangos con signo, mostrando la gráfica asociada en la Fig. 3 y los resultados numéricos en la Tabla VII.

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

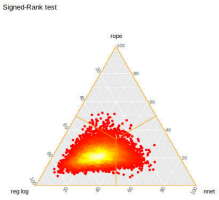
La experimentación inherente a la naturaleza del aprendizaje automático y el rápido crecimiento del número de algoritmos propuestos conlleva la necesidad de establecer un método claro de comparación del desempeño de estos algoritmos y

Tabla VII: Bayesian Signed-Rank test

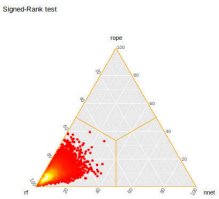
test		
probabilities	left	0.4921
	rope	0.2220
	right	0.2859

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.

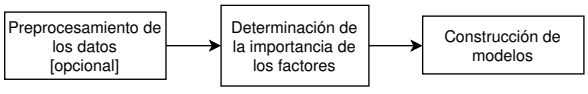


Figura 1. Esquema general de la metodología.

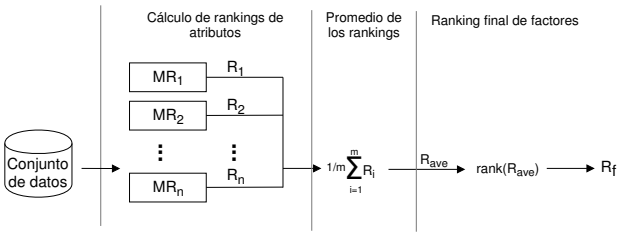


Figura 2. Cálculo del ranking final de factores.

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

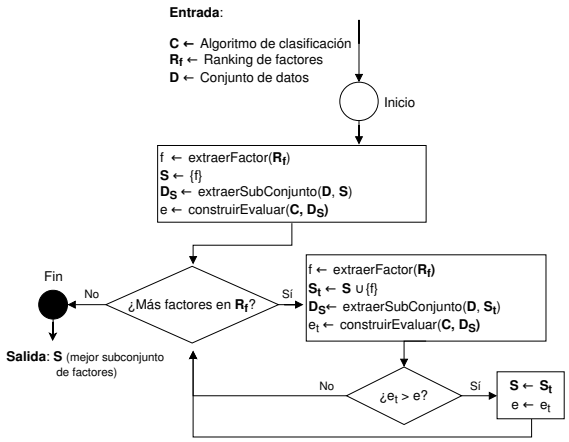


Figura 3. Búsqueda heurística del mejor subconjunto de factores.

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.

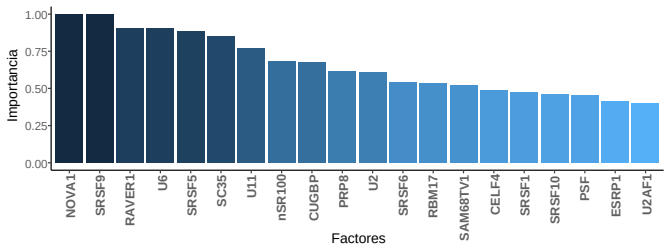


Figura 4. Ranking de factores para diferenciar entre muestras normales y tumorales.

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

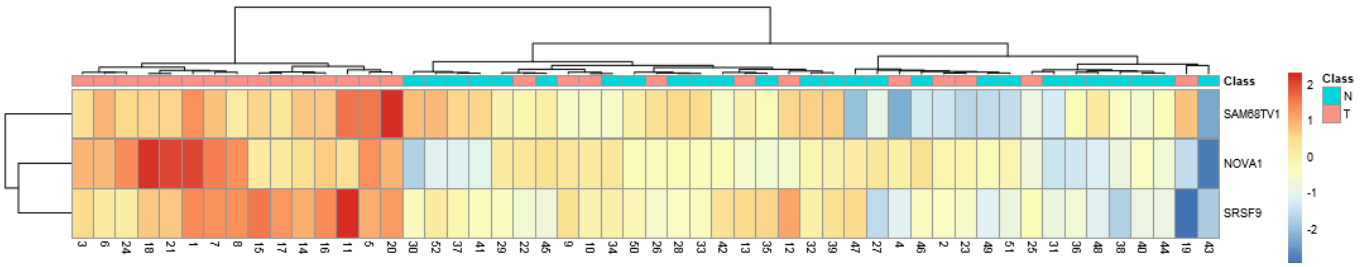


Figura 5. Heatmap generado a partir de un subconjunto de tres factores.

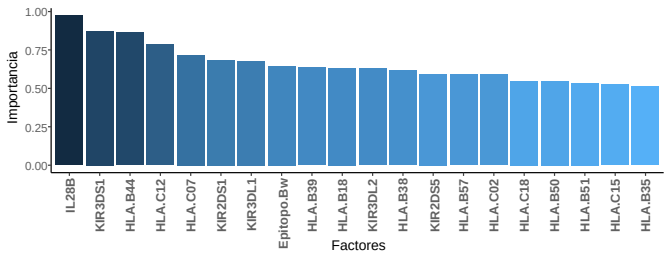


Figura 6. Ranking de factores para el problema de aclaramiento espontáneo en VHC.

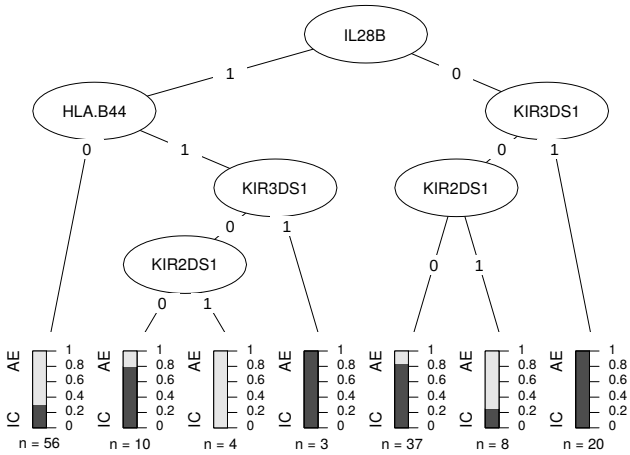


Figura 7. Árbol de decisión generado por uno de los modelos de C4.5 para el problema de VHC.

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.