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

TAMIDA 5:
PREPROCESAMIENTO DE DATOS

Feature Dimensionality vs. Distribution of Sample Types: A Preliminary Study on Gene-Expression Microarrays

J. Salvador Sánchez
Institute of New Imaging Technologies
Department of Computer Languages and Systems
Universitat Jaume I
Castelló de la Plana, Spain
sanchez@uji.es

Vicente García
División Multidisciplinaria de Ciudad Universitaria
Universidad Autónoma de Ciudad Juárez
Ciudad Juárez, Chihuahua, Mexico
vicente.jimenez@uacj.mx

***Abstract*—In gene-expression microarray data sets each sample is defined by hundreds or thousands of measurements. High-dimensionality data spaces have been reported as a significant obstacle to apply machine learning algorithms, owing to the associated phenomenon called 'curse of dimensionality'. The analysis and interpretation of these data sets have been defined as a very challenging problem. The hypothesis proposed in this paper is that there may exist some correlation between dimensionality and the types of samples (safe, borderline, rare and outlier). To examine our hypothesis, we have carried out a series of experiments over four gene-expression microarray databases because these data correspond to a typical example of the so-called 'curse of dimensionality' phenomenon. The results show that there indeed exist meaningful relationships between dimensionality and the proportion of each type of samples, demonstrating that the amount of safe samples increases and the total number of borderline samples decreases as dimensionality of the feature space decreases.**

***Index Terms*—Gene-expression microarray, feature dimensionality, sample types, feature ranking, classification**

I. Introduction

The 'curse of dimensionality' phenomenon (also known as the Hughes phenomenon) constitutes a challenging problem in many real-life applications. It refers to a situation in which the number of samples needed to estimate an arbitrary function with a given level of accuracy grows exponentially with respect to the number of input variables (dimensionality) of the function [1]. An illustrative example of this problem corresponds to gene-expression microarray data [2], [3] where the number of genes (G) massively exceeds the sample size (n): there are typically over tens of thousands of gene-expression levels and often less than 100 samples in the data set. This is a problem in itself because it may increase the complexity of classification/prediction, degrade the generalization ability of classifiers and hinder the understanding of the underlying relationships between the genes and the samples [4], [5]. Besides, overfitting is also a major issue in a high-dimensional, low-sample scenario [6].

Feature selection is the standard way to tackle this problem by choosing a subset of informative variables from the whole set of features for further analysis. In the specific context of microarray data, there exists an apparent need for dimensionality reduction not only because of the huge number of input variables, but also because many of them can be highly correlated with other variables. Throughout the last decades, many different feature (gene) selection algorithms have been proposed using filter, wrapper, embedded, ensembles and hybrid methods [7]–[11].

A particularly popular strategy for feature selection over microarray data refers to the use of gene ranking algorithms, which are filters that comprise some univariate scoring metric to quantify how much more statistically significant each gene is than the others [12]. These methods rank genes in decreasing order of the estimated scores under the assumption that the top-ranked genes correspond to the most informative (or differentially expressed) ones across different classes without redundancy.

The central question the present study intends to answer is how dimensionality of the feature space and some intrinsic data characteristics are related to each other. More specifically, this paper examines whether or not dimensionality reduction may alter the distribution of the different types of samples defined by several authors [13], [14]. To gain some insight into this question, we analyze the tendency of the amount of each type of samples when varying the dimensionality of the feature space. For the experiments, we consider four public data sets of gene-expression microarrays.

Over the past years, the potential links between feature dimensionality and several data complexities in microarrays have been a matter of concern for researchers. For instance, Baumgartner and Somorjai [15] used five real-life biomedical databases of increasing difficulty to show how the data complexity of a given classification problem can be related to the performance of regularized linear classifiers. Okun and Priisalu [16] explored the connections between data complexity and classification performance defined by low-variance and low-biased bolstered resubstitution error made by k-nearest neighbor classifiers. Souto et al. [17] computed different

measures characterizing the complexity of gene expression data sets for cancer diagnosis, and then investigated how those measures were related to the classification performances of support vector machines. Bolón-Canedo et al. [18] presented a review of a set of feature selection methods applied to DNA microarray data and analyzed the impact of class imbalance, class overlapping or data set shift on the classification results. Similarly, Sánchez and García [19] demonstrated that there exist meaningful relationships between dimensionality and class separability in gene-expression microarray data sets. Lorena et al. [20] studied the complexity of several microarray data sets with and without dimensionality reduction using a support vector machine. Seijo-Pardo et al. [21] proposed the use of three data complexity measures to automatically set a threshold value, which is then employed to obtain a subset of genes from the ordered ranking given by a ranker algorithm. Morán-Fernández et al. [22] demonstrated that there is some correlation between microarray data complexity and the classification error rates of a set of classifiers. Sun et al [23] proposed an ECOC algorithm to address the small sample size and class imbalance problems in multi-class microarray data sets by exploring data distributions based on data complexity theory.

Henceforth, the rest of the paper is organized as follows. Section II presents the types of samples according to the taxonomy proposed by Napierala and Stefanowski [14]. Section III provides the experimental set-up and the description of the databases used in our experiments. Next, the results are reported and discussed in Section IV. Finally, Section V summarizes the main conclusions and points out some directions for future research.

II. Types of Samples

Following the categorization proposed by several authors [13], [14], [24], two main types of samples should be distinguished: *safe* and *unsafe*. Safe samples refer to those located in homogeneous regions with data of a single class and are sufficiently separated from examples of other classes, whereas the rest of samples have to be considered as unsafe. The safe samples will be classified correctly by most models, but the classification of unsafe samples will usually be a very tricky task with a high error rate.

The general feature of unsafe samples is that they are placed close to examples from some other classes. However, this type of data can be further divided into three subgroups depending on their particular characteristics [14], [25]: *borderline*, *rare* and *outlier*. Borderline samples are located near the decision boundaries between classes. Rare samples are small groups of examples located far from the core of their class, creating small data chunks or subclusters. Finally, the outliers are single samples being surrounded by examples that belong to some other class.

A simple method to identify each type of samples is based on analyzing the local neighborhood of the examples. This can be performed either by searching for the k neighbors of a sample or by using some kernel function. Thus, one can guess that a safe sample x will be characterized by having a neighborhood with a majority of examples that belong to its same class; rare examples and outliers will be mainly surrounded by examples from different classes, whereas borderline samples will be surrounded by examples both from their same class and also from different classes.

Many authors often choose $k = 5$ because smaller values may poorly distinguish the nature of samples, while higher values would violate the assumption of the local neighborhood [14], [24]–[26]. Following this procedure, we can define the following cases:

- A sample x will be classified in the safe category if at least 4 out of the 5 nearest neighbors belong to the class of x.
- A sample x will be classified in the borderline category if 2–3 out of its 5 nearest neighbors belong to the class of x.
- A sample x will be classified in the rare category if only one nearest neighbor belongs to the class of x, and this has no more than one neighbor from its same class.
- A sample x will be classified in the outlier category if all its nearest neighbors are from the opposite class.

III. Databases and Experimental Protocol

We conducted a pool of experiments on a collection of publicly available gene-expression microarray data sets, which were taken from the Kent Ridge Biomedical Data Set Repository (http://datam.i2r.a-star.edu.sg/datasets/krbd). Table I reports the main characteristics of these databases, including the number of genes (features), the number of samples, and the size of each class (here designated as positive and negative).

TABLE I
Characteristics of the gene-expression microarray data sets

	#Genes	#Samples	#Positive	#Negative
Breast	24481	97	46	51
CNS	7129	60	21	39
Colon	2000	62	22	40
Prostate	12600	136	59	77

For the present study, we varied the percentage of genes from 5% to 100% with a step size of 5% by using the ReliefF algorithm, thus yielding 20 different subsets (each one with a percentage of the top-ranked features) for each database. The experiments have been circumscribed to the ReliefF algorithm because this paper aims to analyze how dimensionality of the feature space might affect the proportion of the different types of samples, not to find the best feature selection/ranking method.

A. The ReliefF Algorithm

The basic idea of the ReliefF algorithm [27] lies on adjusting the weights of a vector $W = [w(1), w(2), \ldots, w(G)]$ with the objective of giving more relevance to features that better discriminate the samples from neighbors of some different class.

It randomly picks out a sample x and searches for k nearest neighbors of the same class (hits, h_i) and k nearest neighbors from each of the different classes (misses, m_i). If x and h_i have different values on feature f, then the weight $w(f)$ is decreased because it is interpreted as a bad property of this feature. In contrast, if x and m_i have different values on the feature f, then $w(f)$ is increased. This process is repeated t times, and the values of the weight vector W are updated as follows:

$$w(f) = w(f) - \frac{\sum_{i=1}^{k} dist(f,x,h_i)}{t \cdot k} + \sum_{c \neq class(x)} \frac{P(c)}{1 - P(class(x))} \cdot \frac{\sum_{i=1}^{k} dist(f,x,m_i)}{t \cdot k} \quad (1)$$

where $P(c)$ is the prior probability of class c, $P(class(x))$ denotes the probability for the class of x, and $dist(f,x,m_i)$ represents the absolute distance between samples x and m_i in the feature f.

The algorithm assigns negative values to features that are completely irrelevant and the highest scores for the most informative features. In general, one will then select the g top-ranked features in order to build the classifier with a presumably much smaller subset of features ($g \ll G$). In addition, unlike other well-known ranking methods such as those based on information theory (e.g., mutual information or information gain), the ReliefF algorithm takes care of the dependencies between genes [28].

IV. Results and Discussion

This section is devoted to explore how the number of genes may have an effect on the amount of samples that belong to each type. As far as we know, there has been no systematic analysis on this problem; in fact, previous studies have focused on identifying the types of samples from the minority class in class imbalanced data sets and analyzing how the resampling techniques may alter the distribution/proportion of safe, borderline, rare and outlier samples [14], [24]–[26], [29], [30]

Bearing our purpose in mind, the experiments were as follows. First, we calculated the percentages of positive and negative samples from each type when varying the percentage of genes. Afterwards, we also run six classifiers of different nature over each subset of features: the 1-nearest neighbor (1-NN) rule with the Euclidean distance, a pruned C4.5 decision tree, a support vector machine (SVM) with a linear kernel using the sequential minimal optimization algorithm and a soft-margin $C = 1.0$, a normalized Gaussian radial basis function (RBF) neural network with the K-means clustering algorithm to provide the basis functions, the naive Bayes classifier (NBayes), and a multi-layer perceptron (MLP) with one hidden layer, a learning rate of 0.3 and 500 training epochs.

Fig. 1 shows the percentages of each positive sample type when varying the dimensionality of the feature space for each database. As can be seen, the percentage of safe samples in the positive class increases and the percentage of borderline positive samples decreases as dimensionality decreases. Although the percentages of rare and outlier samples are generally low, it was observed a very similar behavior to that of the borderline samples. This result could allow to gain some insight into the reasons why classification in lower dimensions is usually easier than in higher dimensions.

Analogously, Fig. 2 displays the percentages of the negative sample types when varying the dimensionality of the feature space for each database. In general, lines in these plots closely match the trend patterns recognized in the plots of Fig. 1, that is, the percentage of safe samples increases and the percentages of the different types of unsafe samples decrease as dimensionality decreases. Notwithstanding, for the safe and borderline samples, we observed an essential difference of behavior between the positive class and the negative class: while the percentages of safe positive samples were usually lower than those of the borderline positive samples, the percentages of safe negative samples always resulted much higher than those of the borderline negative samples. This behavior agrees with the expected one because the negative class corresponds to the majority class and therefore, the probability for a negative sample to be identified as safe is higher than the probability of being classified in some group of the unsafe samples.

Regarding the rare and outlier samples that belong to the negative class, we found that there was no substantial relationship between dimensionality of the feature space and the number of samples in both these types. Nevertheless, this fact should not become especially critical for a given classification problem because the amount of samples that belong to the rare and outlier types is minimal as compared to the total number of safe and borderline samples.

Plots in Fig. 3 correspond to the accuracy achieved by each classification model when applied to each of the 20 subsets. It is possible to observe that the accuracy of all classifiers tends to decrease as the amount of genes increases. A visual comparison between this figure and those of the sample types allows to demonstrate that there exists some significant link (positive correlation) between the dimensionality of the feature space and the distribution of sample types since the highest accuracies were achieved for the subsets with the largest number of safe samples and the smallest number of unsafe samples.

V. Concluding Remarks

As one of the earliest works on investigating the potential connections between feature dimensionality and sample types, this paper has to be viewed as a preliminary study of the effects of dimensionality reduction on the distribution of the different types of samples in a data set.

From the experiments carried out, we have observed that the proportions of safe, borderline, rare and outlier samples vary as the dimensionality of the feature space changes. More specifically, reduction in dimensionality generally leads to a

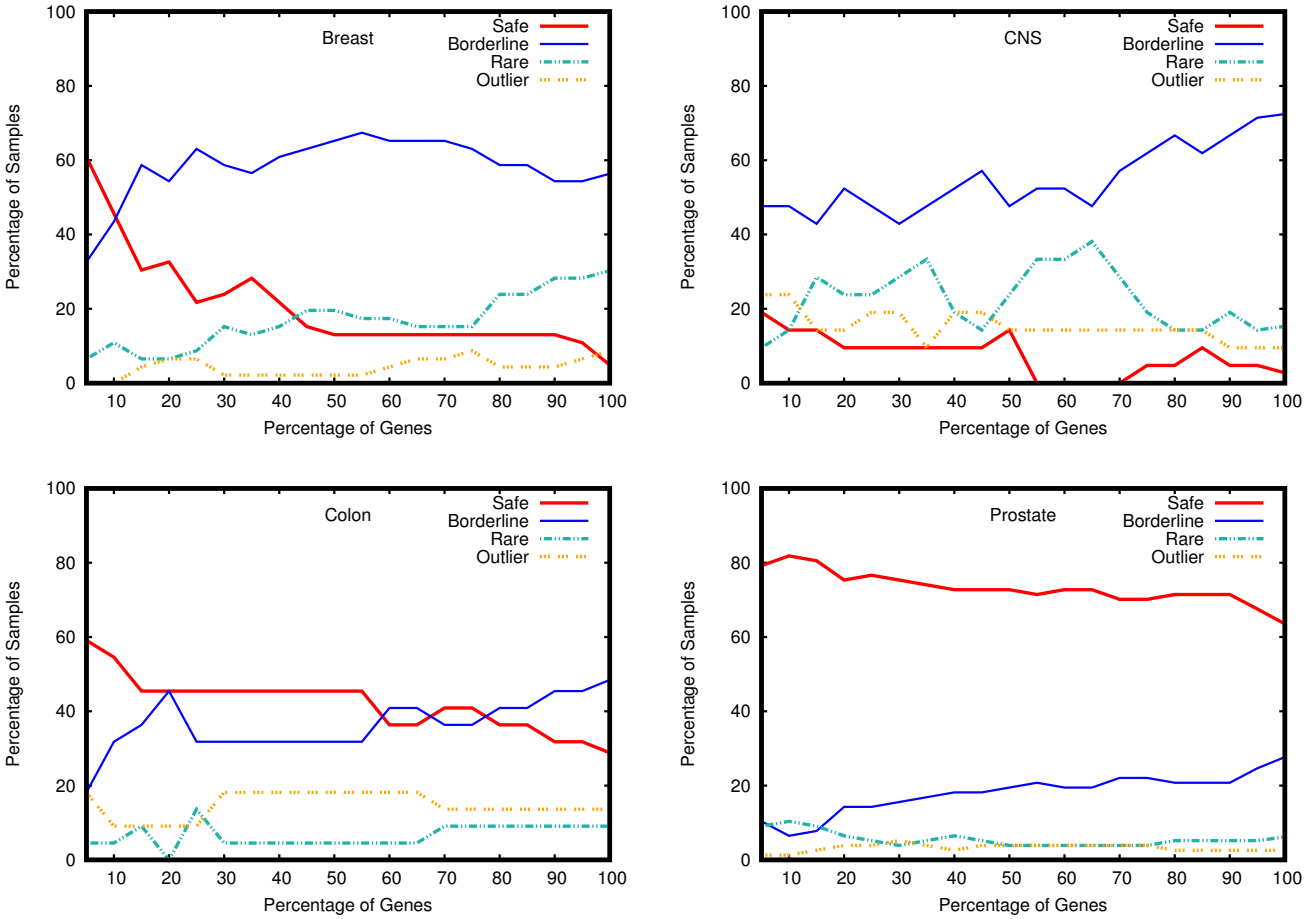


Fig. 1. Plots of the percentage of each type of positive samples when varying the percentage of genes

significant decrease in the amount of borderline samples and an increase in the number of safe samples. As showed in the experiments, this has a direct impact on the performance of classifiers because the classification of safe samples results much easier than the classification of any type of unsafe samples.

Through the characterization of databases by the distribution of their sample types, our hypothesis for further research is that it would be possible to define a meta-learning framework to choose the feature subset with the highest classification performance. Another direction for extending the present paper consists in the combined use of sample types and data complexity measures for the implementation of accurate preprocessing methods.

Acknowledgment

This research work has partially been supported by the Mexican PRODEP under Grant No. DSA/103.5/15/7004, the Generalitat Valenciana under Grant No. PROMETEOII/2014/062, and the Universitat Jaume I under Grant No. P1-1B2015-74.

References

[1] L. Chen, "Curse of dimensionality," in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Boston, MA: Springer, 2009, pp. 545–546.

[2] R. Clarke, H. W. Ressom, A. Wang, J. Xuan, M. C. Liu, E. A. Gehan, and Y. Wang, "The properties of high-dimensional data spaces: implications for exploring gene and protein expression data," *Nat. Rev. Cancer*, vol. 8, no. 1, pp. 37–49, 2008.

[3] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "Recent advances and emerging challenges of feature selection in the context of big data," *Knowl.-Based Syst.*, vol. 86, pp. 33–45, 2015.

[4] E. R. Dougherty, "Small sample issues for microarray-based classification," *Compar. Func. Genom.*, vol. 2, no. 1, pp. 28–34, 2001.

[5] L. Wang, F. Chu, and W. Xie, "Accurate cancer classification using expressions of very few genes," *IEEE-ACM T. Comput. Biol. Bioinform.*, vol. 4, no. 1, pp. 40–53, 2007.

[6] R. Somorjai, B. Dolenko, and R. Baumgartner, "Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions," *Bioinformatics*, vol. 19, no. 12, pp. 1484–1491, 2003.

[7] I. Inza, P. Larrañaga, R. Blanco, and A. J. Cerrolaza, "Filter versus wrapper gene selection approaches in DNA microarray domains," *Artif. Intell. Med.*, vol. 31, no. 2, pp. 91–103, 2004.

[8] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[9] C. Lazar, J. Taminau, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, H. Bersini, and A. Nowe, "A survey on filter techniques for feature selection in gene expression microarray analysis,"

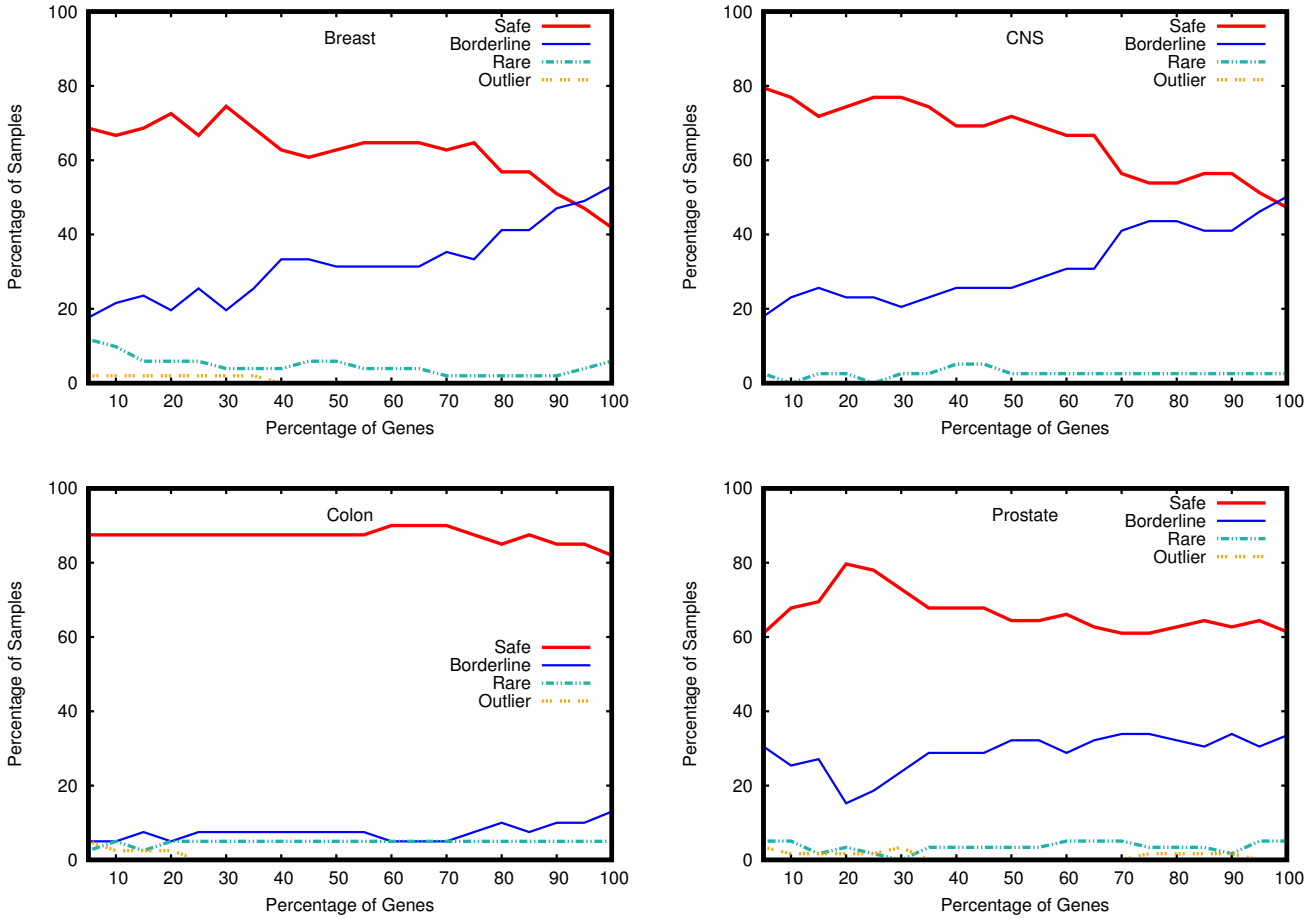


Fig. 2. Plots of the percentage of each type of negative samples when varying the percentage of genes

IEEE-ACM T. Comput. Biol. Bioinform., vol. 9, no. 4, pp. 1106–1119, 2012.

[10] Z. M. Hira and D. F. Gillies, "A review of feature selection and feature extraction methods applied on microarray data," *Adv. Bioinformatics*, vol. 2015, no. ID 198363, pp. 1–13, 2015.

[11] J. C. Ang, A. Mirzal, H. Haron, and H. N. A. Hamed, "Supervised, unsupervised, and semi-supervised feature selection: A review on gene selection," *IEEE-ACM T. Comput. Biol. Bioinform.*, vol. 13, no. 5, pp. 971–989, 2016.

[12] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[13] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in *Proc. 14th International Conference on Machine Learning*, Nashville, TN, 1997, pp. 179–186.

[14] K. Napierala and J. Stefanowski, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *J. Intell. Inf. Syst.*, vol. 46, no. 3, pp. 563–597, 2016.

[15] R. Baumgartner and R. Somorjai, "Data complexity assessment in undersampled classification of high-dimensional biomedical data," *Pattern Recogn. Lett.*, vol. 27, no. 12, pp. 1383–1389, 2006.

[16] O. Okun and H. Priisalu, "Dataset complexity in gene expression based cancer classification using ensembles of k-nearest neighbors," *Artif. Intell. Med.*, vol. 45, no. 2, pp. 151–162, 2009.

[17] M. C. P. de Souto, A. C. Lorena, N. Spolaôr, and I. G. Costa, "Complexity measures of supervised classifications tasks: A case study for cancer gene expression data," in *Proc. International Joint Conference on Neural Networks*, Barcelona, Spain, 2010, pp. 1–7.

[18] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, J. Benítez, and F. Herrera, "A review of microarray datasets and applied feature selection methods," *Inform. Sciences*, vol. 282, pp. 111–135, 2014.

[19] J. S. Sánchez and V. García, "Addressing the links between dimensionality and data characteristics in gene-expression microarrays," in *Proc. International Conference on Learning and Optimization Algorithms: Theory and Applications*, Rabat, Morocco, 2018, pp. 1–6.

[20] A. C. Lorena, I. G. Costa, N. Spolaor, and M. C. de Souto, "Analysis of complexity indices for classification problems: Cancer gene expression data," *Neurocomputing*, vol. 75, no. 1, pp. 33–42, 2012.

[21] B. Seijo-Pardo, V. Bolón-Canedo, and A. Alonso-Betanzos, "Using data complexity measures for thresholding in feature selection rankers," in *Advances in Artificial Intelligence*. Lecture Notes in Computer Science, Springer, 2016, vol. 9868, pp. 121–131.

[22] L. Morán-Fernández, V. Bolón-Canedo, and A. Alonso-Betanzos, "Can classification performance be predicted by complexity measures? A study using microarray data," *Knowl. Inf. Syst.*, vol. 51, no. 3, pp. 1067–1090, 2017.

[23] M. Sun, K. Liu, and Q. Hong, "An ECOC approach for microarray data classification based on minimizing feature related complexities," in *Proc. 10th International Symposium on Computational Intelligence and Design*, Hangzhou, China, 2017, pp. 300–303.

[24] J. A. Sáez, B. Krawczyk, and M. Woźniak, "Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets," *Pattern Recogn.*, vol. 57, pp. 164–178, 2016.

[25] B. Krawczyk, M. Woniak, and F. Herrera, "Weighted one-class classification for different types of minority class examples in imbalanced data," in *Proc. IEEE Symposium on Computational Intelligence and Data Mining*, Piscataway, NJ, 2014, pp. 337–344.

[26] P. Skryjomski and B. Krawczyk, "Influence of minority class instance types on SMOTE imbalanced data oversampling," in *Proc. 1st Interna-*

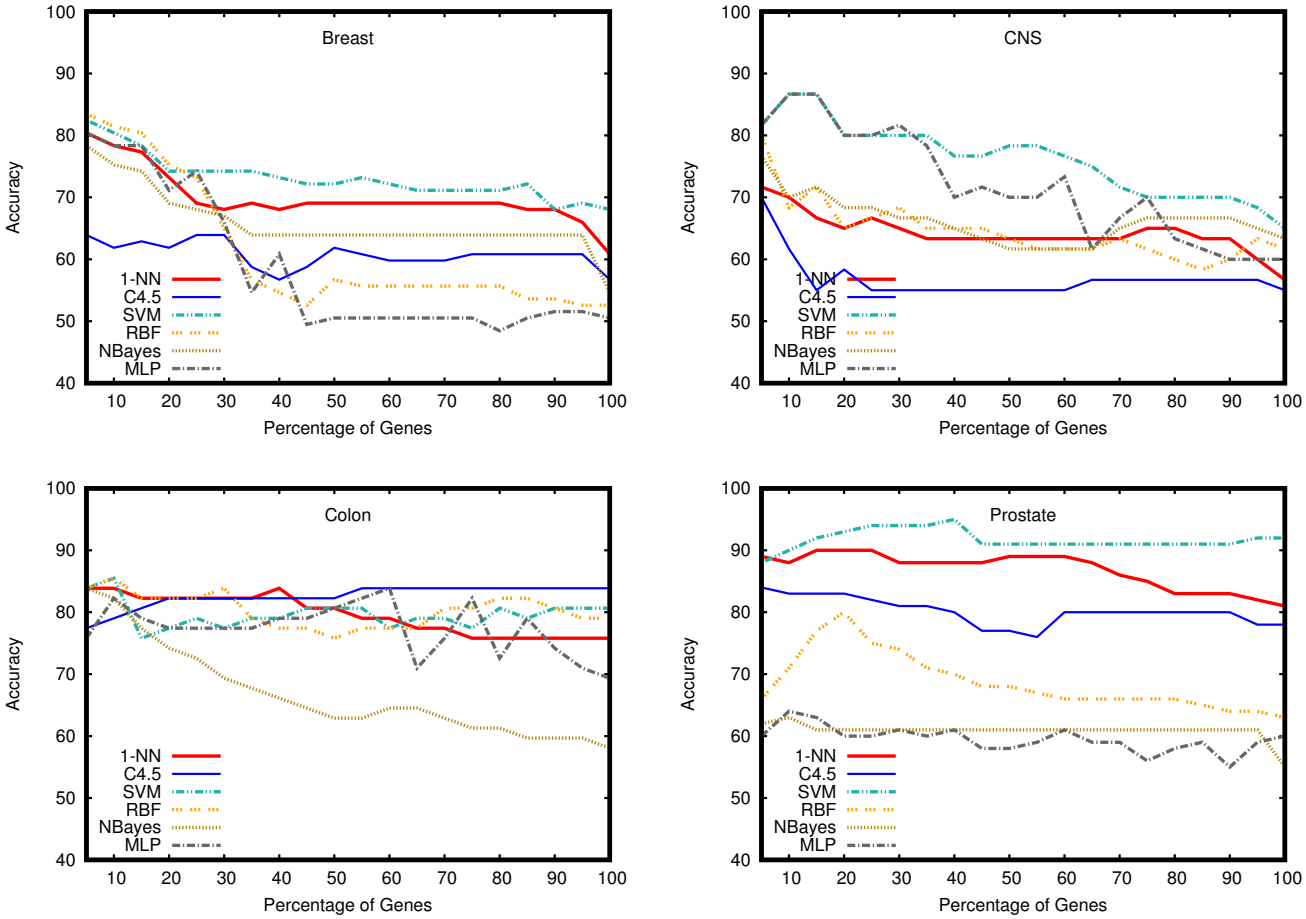


Fig. 3. Classification accuracies when varying the percentage of genes

tional Workshop on Learning with Imbalanced Domains: Theory and Applications, Skopje, Macedonia, 2017, vol. 74, pp. 7–21.

[27] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Mach. Learn.*, vol. 53, no. 1–2, pp. 23–69, 2003.

[28] Y. Peng, W. Li, and Y. Liu, "A hybrid approach for biomarker discovery from microarray gene expression data for cancer classification," *Cancer Inform.*, vol. 2, pp. 301–311, 2006.

[29] S. Wang, L. L. Minku, and X. Yao, "Resampling-based ensemble methods for online class imbalance learning," *IEEE T. Knowl. Data En.*, vol. 27, no. 5, pp. 1356–1368, 2015.

[30] M. Lango and J. Stefanowski, "The usefulness of roughly balanced bagging for complex and high-dimensional imbalanced data," in *Proc. 4th International Workshop on New Frontiers in Mining Complex Patterns*. Porto, Portugal: Springer International Publishing, 2016, pp. 93–107.

Selección de características distribuida en entornos heterogéneos

Verónica Bolón-Canedo
Grupo LIDIA. DCITIC.
Universidade da Coruña
A Coruña, España
veronica.bolon@udc.es

Rubén Seoane-Martínez
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España

José Luis Morillo-Salas
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
jose.luis.morillo@udc.es

Amparo Alonso-Betanzos
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
ciamparo@udc.es

***Resumen*—Los avances en las Tecnologías de la Información y las Comunicaciones han contribuido a la proliferación de grandes bases de datos. En algunos casos estos datos ya están distribuidos en su origen, pero en otros casos su gran escala hace que el procesamiento en un único nodo sea imposible, y en consecuencia la distribución en varios nodos de cómputo es una opción natural para su manejo. En este trabajo, proponemos una metodología que nos permite distribuir el proceso de selección de características, la mayoría de las veces un paso de preprocesado imprescindible en los conjuntos de alta dimensión actuales, ya que nos permite reducir la dimensión de entrada, seleccionando las características relevantes y eliminando las redundantes y/o irrelevantes. En particular, nuestra propuesta en este artículo se centra en el problema de los conjuntos de datos desbalanceados, bien porque la situación se da ya en origen o bien cuando este contexto en que las distintas clases de datos no están igualmente representadas en las distintas particiones se produce debido a que se debe distribuir el conjunto único original para poder tratarlo. Los resultados experimentales obtenidos demuestran que nuestra aproximación distribuida obtiene resultados de error comparables a la aproximación centralizada, aportando como ventajas una reducción apreciable del tiempo computacional y la capacidad de trabajar eficientemente en entornos de desbalanceo de clases.**

***Index Terms*—selección de características, algoritmos distribuidos, conjuntos de datos desbalanceados.**

I. Introducción

La selección de características (SC) es una técnica de aprendizaje automático en la que se seleccionan los atributos que permiten que un problema esté claramente definido, mientras que los irrelevantes o redundantes se ignoran [1]. Tradicionalmente, un algoritmo de SC se aplica de manera centralizada, es decir, se utiliza un único modelo selector de características sobre todos los datos del conjunto para resolver un problema determinado. Sin embargo, en algunos casos, los datos pueden o bien estar ya distribuidos en varias localizaciones, o bien se puede usar una estrategia de aprendizaje distribuido para repartir en varios nodos de cómputo un conjunto de datos que es demasiado grande para poder ser procesado en un único nodo. De esta forma podemos aprovechar el procesamiento de estos múltiples subconjuntos de datos bien en secuencia o en paralelo. Existen varias formas de distribuir una tarea de selección de características, aunque las más comunes son:

- los datos están juntos en un conjunto de datos muy grande, por lo que se distribuyen en varios procesadores, se ejecuta un algoritmo de SC idéntico en cada uno y luego los resultados parciales se combinan para obtener un resultado final, y
- los datos pueden estar en diferentes conjuntos de datos situados en diferentes ubicaciones, por lo que se ejecuta un algoritmo de selección de características idéntico en cada uno y los resultados se combinan para obtener un resultado final.

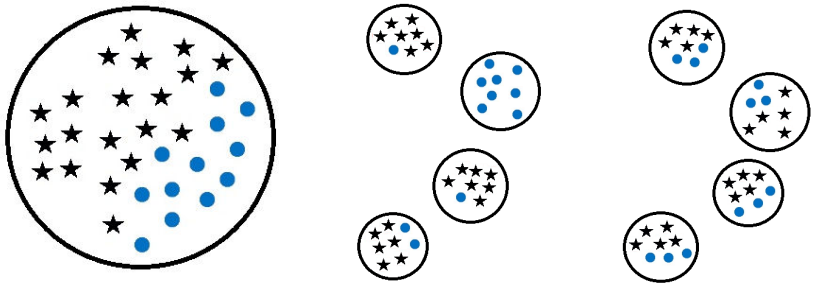
Al respecto, existen varios trabajos en la literatura que realizan la selección de características de forma distribuida [2], [3]. Sin embargo, cuando los datos se distribuyen en varios procesadores, pueden aparecer algunos problemas adicionales, como un alto desequilibrio entre clases en algunos de los nodos, o incluso la situación extrema en la que algunas clases no están representadas en absoluto en algunos de los subconjuntos de datos. El *problema de desequilibrio de clase o desbalanceo* se produce cuando un conjunto de datos está dominado por una clase mayoritaria que tiene significativamente muchas más instancias que las otras clases, llamadas minoritarias. En este caso, los algoritmos de aprendizaje computacional suelen presentar un sesgo hacia las clases mayoritarias, ya que las reglas que predicen correctamente esas instancias se ponderan positivamente a favor de la métrica de precisión, mientras que las reglas específicas que predicen ejemplos de la clase minoritaria generalmente se ignoran. Por lo tanto, las muestras de las clases minoritarias se clasifican erróneamente más a menudo que las de las otras clases [4].º

En este trabajo presentamos una metodología para distribuir el proceso de SC, que tiene en cuenta este problema de la posible heterogeneidad de los subconjuntos. Para ello usamos dos alternativas: (i) forzar las particiones del conjunto de datos para mantener el equilibrio entre las clases, y (ii) aplicar técnicas de sobremuestreo (oversampling) cuando el desequilibrio es inevitable.

II. Metodología distribuida

En este trabajo, se detalla la aplicación de una metodología para distribuir el proceso de SC sobre la base de trabajos pre-

Este trabajo ha sido parcialmente financiado por el Ministerio de Economía y Competitividad (proyectos de investigación TIN 2015-65069-C2-1-R y la Red Española de Big Data y Análisis de datos escalable, TIN2016-82013-REDT), y por Fondos de Desarrollo Regional de la Unión Europea.



(a) Centralizado (b) Partición aleatoria (c) Partición homogénea

Figura 1. Escenarios centralizado (a) y particiones aleatoria (b) y homogénea (c) en un proceso de selección de características distribuido

vios [5], [6]. Esta metodología consta de tres pasos principales, que son los siguientes:

1. partición de los datos, si éstos no estuviesen ya distribuidos en origen,
2. aplicación del método de SC a cada una de las diferentes particiones realizadas
3. combinación de los resultados.

Debemos de tener en cuenta que los dos primeros pasos se repiten varias rondas (r), para garantizar la captura de suficiente información para el paso de combinación de los resultados parciales.

El primer paso de la metodología anterior es el núcleo de este trabajo y consiste en dividir sin reemplazo los datos del conjunto original, asignando grupos de n muestras a cada subconjunto de datos. Se seguirán dos enfoques principales: *partición aleatoria*, en la que se realizará una distribución aleatoria de los datos en los distintos nodos, y *partición homogénea*, en la que se mantienen las proporciones del conjunto original en cada uno de los subconjuntos obtenidos. Un ejemplo de estos dos tipos de partición, junto con el escenario centralizado en el que todos los datos están juntos, se puede ver en la figura 1.

Después de realizar una partición, el conjunto de datos podría estar desequilibrado (ya sea porque la partición se realizó al azar o porque el conjunto de datos ya estaba desbalanceado en origen). En este caso, nuestra propuesta consiste en aplicar el método de sobremuestreo SMOTE [7], que agrega ejemplos sintéticos de la clase minoritaria al conjunto de datos original hasta que la distribución de clases se equilibre. Para poder conseguir esto, SMOTE genera ejemplos sintéticos de la clase minoritaria utilizando los ejemplos originales de la misma de la siguiente manera: en primer lugar, busca los k vecinos más cercanos de la muestra de la clase minoritaria que se utilizará como base para la nueva muestra sintética. Luego, en el segmento que une la muestra de la clase minoritaria con uno o todos sus vecinos, se toma aleatoriamente una muestra sintética y se agrega al nuevo conjunto de datos sobremuestreados.

El siguiente paso en la metodología general consiste en aplicar un método de SC en cada partición. Las características que se seleccionan para ser eliminadas reciben un voto y luego, se realiza una nueva ronda que conduce a una nueva partición del conjunto de datos y se lleva a cabo una nueva iteración de la votación hasta alcanzar el número predefinido de rondas r. Finalmente, las características que han recibido una cantidad de votos por encima de un umbral predefinido se eliminan. Por lo tanto, se obtiene finalmente un conjunto único de características que se pueden utilizar para entrenar un clasificador C y probar su rendimiento en un nuevo conjunto de muestras (conjunto de datos de test). Más detalles sobre cómo elegir el umbral de votos se pueden encontrar en [5], [6]. El pseudocódigo de la metodología propuesta se muestra en el algoritmo 1.

```
1  inicializar el vector de votos a 0
2  para cada ronda hacer
3      dividir el conjunto de datos d aleatoriamente o
         mantener las proporciones de las clases en
         subconjuntos de datos disjuntos
4      para cada subconjunto de datos hacer
5          si los datos están desbalanceados entonces
6              aplicar SMOTE
           fin
7          aplicar un algoritmo de selección de características
8          incrementar un voto para cada característica a ser
             eliminada
       fin
   fin
9  eliminar las características cuyo número de votos sea
     superior a un umbral
10 clasificar con el subconjunto de características obtenido
```

Algoritmo 1: Pseudo-código de la metodología propuesta

III. Resultados experimentales

En esta sección presentaremos el esquema de experimentación y los resultados obtenidos. Recordemos que los objetivos de la experimentación son dos, (i) poder establecer qué tipo de partición es la más adecuada y (ii) cúal es la influencia del algoritmo de sobremuestreo SMOTE cuando las particiones presentan datos desbalanceados.

III-A. Comparación entre las aproximaciones distribuidas y la centralizada

Con este objetivo, hemos seleccionado 6 conjuntos de datos, cuyas características se resumen en la Tabla I, y que están

Cuadro I
Características de los conjuntos utilizados en la primera parte de la experimentación, en donde solamente se utiliza SMOTE en la clase minoritaria.

Conjunto	Nº Muestras	Nº Características	Nº Clases	% clase mayoritaria
Connect4	67557	42	3	65.83
Isolet	7797	617	26	3.85
Madelon	2400	500	2	50
Mnist	60000	717	2	50
Ozone	2536	72	2	97.12
Spambase	4601	57	2	60.6

disponibles para su descarga en el UCI Machine Learning Repository [1].

Aunque la metodología propuesta es genérica, y por lo tanto se puede usar con cualquier método de SC, en este trabajo hemos elegido una suite de cuatro filtros, basados en diferentes tipos de métricas. Concretamente hemos utilizado Correlation-Based Feature Selection (CFS), Consistency-based, Information Gain y ReliefF, todos ellos disponibles en la herramienta de software libre Weka [2]. Para posteriormente poder evaluar los resultados de la selección de características realizada, hemos elegido cuatro clasificadores populares en el estado del arte: C4.5, Naive Bayes, IB1 y Vectores de Máquinas Soporte (en inglés, Support Vector Machine –SVM–). Los experimentos se realizaron en una CPU Core ™i3-6100 Intel ®3.70 GHz con 16 GB de memoria RAM.

En el primer estudio experimental se compararon tres escenarios diferentes: (i) la aproximación centralizada estándard, (ii) la distribución aleatoria, y (iii) el particionado homogéneo. Para las dos aproximaciones distribuidas (la aleatoria y la homogénea), el número de rondas utilizado fue de 5. Para asegurar una buena fiabilidad en los resultados obtenidos, se realizó una validación *hold-out* estándard, es decir, se dividieron los distintos conjuntos de la tabla I en dos subconjuntos diferentes, con la proporción 2/3 para entrenamiento y 1/3 para prueba, y se repitió esta operación 5 veces. También se han usado test de significación estadística, en primer lugar un test de Friedman para comprobar si existían diferencias significativas para un nivel de significación $\alpha = 0{,}5$, y posteriormente de Nemenyi para obtener aquellos modelos que no son significativamente diferentes a los que obtienen la mayor precisión. Las tablas detalladas con los resultados obtenidos para todas las combinaciones entre conjuntos de datos, métodos de SC y clasificadores pueden verse en el material suplementario que se encuentra en [3].

La Tabla II muestra un resumen de este primer conjunto de experimentos, en los que la meta es comparar los tres escenarios (centralizado, partición aleatoria y partición homogénea), independientemente de la aplicación de la técnica de *oversampling* SMOTE. La tabla muestra los resultados para cada combinación de conjunto de datos y escenario, teniendo en cuenta dos medidas de evaluación diferentes: la precisión de la clasificación y el valor del índice kappa. El motivo de incluir el valor de Kappa es porque éste evalúa la calidad del aprendizaje teniendo en cuenta las situaciones en las que el conjunto de datos está desequilibrado y el clasificador aprende correctamente la clase mayoritaria, pero sistemáticamente clasifica erróneamente las instancias de la clase minoritaria. En las primeras dos filas de cada conjunto de datos, se puede consultar el promedio de la precisión de clasificación y Kappa; y en las últimas dos filas se muestran los valores máximos de precisión y Kappa (y la combinación que lo obtiene entre paréntesis). Como se puede ver, los enfoques distribuidos (partición aleatoria y homogénea) son una buena solución para disminuir el tiempo computacional sin implicar una degradación en el rendimiento de clasificación. Comparando los dos enfoques distribuidos, vale la pena señalar que, en general, el enfoque homogéneo parece obtener resultados más estables, mientras que con la partición aleatoria puede ocurrir que en un caso particular la proporción de las clases sea óptima y por esa razón obtiene los mejores resultados en algunos casos.

Como era de esperar, los enfoques distribuidos reducen significativamente el tiempo de ejecución en comparación con el enfoque centralizado (ver detalles en el material complementario 3^0), aunque depende concretamente tanto del método de selección empleado como del conjunto de datos. Cuando el conjunto de datos es pequeño, la mejora es leve (por ejemplo, de 0.40s a 0.34s en el conjunto Ozone) pero en conjuntos de datos más grandes, la mejora es considerable (por ejemplo, de 820.46s a 0.83s en el conjunto Connect-4). Es remarcable también el buen rendimiento obtenido por los métodos de selección ReliefF y Consistency-based.

III-B. Utilización de SMOTE en las particiones

El segundo grupo de experimentos consiste en la evaluación de la efectividad de SMOTE ante el problema del desbalanceo de clases. La comparación se realizó entre los dos escenarios distribuidos (partición aleatoria y homogénea). Debemos tener en cuenta que, al aplicar la partición aleatoria, es posible que algunos subconjuntos de datos estén desbalanceados, incluso si el conjunto de datos completo no lo estaba. Por lo tanto, hemos aplicado SMOTE cuando el subconjunto de datos no estaba balanceado (ya sea bien debido a la existencia de esta circunstancia en clase original, o bien debido a la partición aleatoria). Se han realizado diferentes experimentos con diferentes porcentajes de sobremuestreo. Por ejemplo, si la clase minoritaria tiene 40 muestras y aplicamos SMOTE

[1] http://archive.ics.uci.edu/ml/index.php
[2] http://www.cs.waikato.ac.nz/ ml/weka/
[3] http://lidiagroup.org/index.php/en/materials-en.html

Conjunto		Centralizado	Aleatorio	Homogéneo
Connect4	Precisión (media)	65.72	67.52	67.52
	Kappa (media)	0.175	0.184	0.188
	Precisión (max)	73.37 (Cons+C4.5)	74.12 (Rel+C4.5)	72.81 (IG+C4.5)
	Kappa (max)	0.454 (Cons+C4.5)	0.425 (Rel+C4.5)	0.399 (Rel+C4.5)
Isolet	Precisión (media)	63.92	69.12	68.64
	Kappa (media)	0.624	0.678	0.673
	Precisión (max)	84.60 (Rel+SVM)	85.51 (Rel+SVM)	83.87 (Rel+SVM)
	Kappa (max)	0.839 (Rel+SVM)	0.849 (Rel+SVM)	0.832 (Rel+SVM)
Madelon	Accuracy (media)	74.64	72.25	75.32
	Kappa (media)	0.492	0.444	0.506
	Accuracy (max)	88.75 (Varios+IB1)	81.75 (Rel+C4.5)	89.62 (Rel+IB1)
	Kappa (max)	0.774 (Varios+IB1)	0.636 (Rel+C4.5)	0.792 (Rel.+IB1)
MNIST	Precisión (media)	81.04	83.68	83.83
	Kappa (media)	0.620	0.672	0.675
	Precisión (max)	89.96 (Rel+IB1)	96.33 (Cons+IB1)	95.83 (Rel+IB1)
	Kappa (max)	0.799 (Rel+IB1)	0.926 (Cons+IB1)	0.916 (Rel+IB1)
Ozone	Precisión (media)	92.09	91.06	90.87
	Kappa (media)	0.1014	0.1012	0.092
	Precisión (max)	97.12 (Todos+SVM)	96.99 (Todos+SVM)	96.97 (Todos+SVM)
	Kappa (max)	0.189 (Cons+C4.5)	0.215 (Cons+C4.5)	0.180 (IG+IB1)
Spambase	Precisión (media)	86.66	87.18	87.63
	Kappa (media)	0.723	0.732	0.742
	Precisión (max)	91.42 (CFS+C4.5)	91.24 (CFS+C4.5)	91.73 (CFS+C4.5)
	Kappa (max)	0.819 (CFS+C4.5)	0.816 (CFS+C4.5)	0.826 (CFS+C4.5)

Cuadro II

Resumen de los resultados obtenidos para las aproximaciones distribuidas y centralizada. No se ha utilizado el método SMOTE en las aproximaciones distribuidas.

con un nivel de 100, significa que se generan 40 muestras sintéticas, si el nivel es 200, significa que se generan 80 nuevas muestras. Además, incluimos la opción "auto", que consiste en aplicar SMOTE de tal forma que las clases queden completamente balanceadas.

Conjunto	Precisión	Kappa	Escenario	Combinación	SMOTE
Isolet	85.68	0.851	Aleatorio	Rel+SVM	Auto
Madelon	89.63	0.792	Homogéneo	Rel+IB1	0
MNIST	96.34	0.926	Aleatorio	Cons+IB1	0
Connect4	74.12	0.425	Aleatorio	Rel+C4.5	0
	72.90	0.442	Aleatorio	Rel+C4.5	100
Ozone	97.28	0	Homogéneo	All+SVM	100
	90.82	0.302	Homogéneo	Rel+SVM	600
Spambase	91.73	0.826	Homogéneo	CFS+C4.5	0
	91.68	0.827	Homogéneo	CFS+C4.5	300

Cuadro III

Resumen de los resultados obtenidos usando SMOTE en las clases minoritarias.

```
1  para repeticiones hacer
2      dividir el conjunto de datos d aleatoriamente en
         subconjuntos disjuntos de datos train y test
3      calcular la clase mayoritaria del conjunto de train
4      para cada nivel Smote minoritaria hacer
5          para cada nivel Smote mayoritaria hacer
6              train = SMOTE(clase_mayoritaria, train)
7              train = SMOTE(clase_minoritaria, train)
8              para filtros hacer
9                  train_filtrado = sel_car(filtro,train)
10                 para clasificador hacer
11                     clasificar(clasificador, train_filtrado,
                         test)
                   fin
               fin
           fin
       fin
   fin
```

Algoritmo 2: Pseudo-código de la metodología propuesta usando SMOTE también en la clase mayoritaria

La tabla III muestra el resumen de los mejores resultados obtenidos al aplicar diferentes niveles de sobremuestreo con SMOTE a los subconjuntos de datos. En la primera fila de cada conjunto de datos, se muestra la opción con la mayor precisión, mientras que la segunda fila representa la opción con el valor Kappa más alto. Cuando el mejor resultado para ambas mediciones de evaluación se logra mediante la misma combinación y escenario, solo se muestra una fila. Como era de esperar, la aplicación de una técnica de sobremuestreo no es necesaria en el caso de conjuntos de datos equilibrados (Isolet, Madelon, MNIST). En el caso de Isolet, la aplicación de SMOTE en el escenario de partición aleatoria ha resultado provechosa, ya que en este caso es posible que los conjuntos de datos equilibrados produzcan subconjuntos de datos no balanceados (especialmente para Isolet, con un alto número de clases). Por el contrario, los conjuntos de datos desbalanceados (Connect4, Ozone y Spambase) son buenos candidatos para mejorar sus resultados después de aplicar SMOTE. De hecho, la Tabla III muestra que la aplicación del método de sobremuestreo mejora los valores Kappa, lo que significa que el aprendizaje de las clases es mejor. Hay que recordar que la clase mayoritaria de ozono tiene el 97.12 % de las muestras, por lo que al obtener una precisión de clasificación del 97.28 % es posible que clasifique correctamente todas las muestras de la clase mayoritaria, pero solo unas pocas de la clase minoritaria. Después de aplicar el método de sobremuestreo, la precisión cae al 90.82 %, probablemente porque el clasificador no está tan sobreajustado para aprender la clase mayoritaria

y tiene una tasa de verdaderos positivos más alta en la clase minoritaria.

Finalmente, se realizó un tercer conjunto de experimentos, en los que se utiliza también la técnica SMOTE para añadir también muestras sintéticas en la clase mayoritaria, no sólo en la minoritaria, de forma que todas las clases, mayoritarias y minoritarias, cuenten en sus subconjuntos con muestras sintéticas, como se puede ver en el algoritmo 2.

Para realizar este tercer bloque de experimentos se utilizaron dos tipos de conjuntos, los que denominamos con la etiqueta *normal* en la tabla IV, que son conjuntos de datos en los que el número de muestras es mucho mayor que el número de características, y conjuntos de datos del tipo *Microarrays* [8], obtenidos de investigaciones sobre la clasificación de casos de cáncer, que tienen un elevado número de características y un número muy pequeño de muestras (ver tabla IV). La idea es comprobar no sólo si el realizar SMOTE en todas las clases mejora el resultado, al tener muestras sintéticas en todas ellas, sino también si el balance entre muestras y características influye en los resultados. Se han repetido de nuevo los experimentos, pero en este caso además se han añadido muestras sintéticas también en la clase mayoritaria, utilizando SMOTE con porcentajes de 20, 40 y 100. Al igual que anteriormente, se han obtenido valores para todas las posibles combinaciones de clasificador, filtro y combinación de porcentajes de SMOTE en la clase mayoritaria. En la tabla V se pueden ver los resultados obtenidos para todos los conjuntos de la tabla IV sin SMOTE, con la alternativa de SMOTE en la clase minoritaria y con la alternativa de usar SMOTE en todas las clases.

Como podemos ver en la tabla V, la alternativa SMOTE en las clases minoritaria y mayoritaria conjuntamente es siempre la opción que alcanza la precisión máxima, con los valores de índice kappa más altos (en ocasiones, otras alternativas consiguen idénticas kappas, y la alternativa SMOTE sólo en minoritaria empata en precisión máxima en 5 de los 12 conjuntos). No parecen existir grandes diferencias entre los dos tipos de conjuntos, si bien la diferencia en precisión media entre las dos alternativas usando SMOTE en los conjuntos microarray es menor que en el caso de los conjuntos que hemos denominado normales.

IV. Conclusiones y Trabajo Futuro

Hemos presentado una metodología para la selección de características distribuida, tratando de resolver el problema del desbalanceo de los datos en las diferentes particiones. Para ello, hemos forzado a las particiones de datos en los diferentes nodos a mantener la misma distribución de clase que el conjunto de datos original y hemos aplicado la técnica de sobremuestreo (oversampling) SMOTE. Los resultados experimentales en siete conocidos conjuntos de datos han demostrado que:

- El enfoque distribuido — partición aleatoria u homogénea — es competitivo cuando se compara con el enfoque centralizado estándar, incluso en algunos casos mejorando el rendimiento de clasificación.
- La partición homogénea obtiene resultados más estables que la partición aleatoria.
- La aplicación de SMOTE en las clases minoritarias (uso estándard del procedimiento), mejora la calidad del aprendizaje en conjuntos de datos desbalanceados, en algunos casos a expensas de una ligera disminución en la precisión general.
- Además al aplicar el método propuesto con un porcentaje pequeño de SMOTE también en la clase mayoritaria se aprecia una mejora en la precisión máxima obtenida en todos los conjuntos de datos. Además, si bien es cierto que esta última alternativa no obtiene prácticamente en ningún caso los mejores valores de precisiones medias, sí que consigue obtener los valores de kappa más altos, por lo que el método presenta una mayor robustez.

Como trabajo futuro, nos planteamos probar otros métodos para tratar la heterogeneidad, como puede ser el caso de las técnicas de submuestreo (undersampling en inglés), ponderación, etc.

Referencias

[1] I. Guyon. *Feature extraction: foundations and applications*, volume 207. Springer, 2006.

[2] L. Morán-Fernández, V. Bolón-Canedo, and A. Alonso-Betanzos. Centralized vs. distributed feature selection methods based on data complexity measures. *Knowledge-Based Systems*, 117:27–45, 2017.

[3] V. Bolón-Canedo, N. Sánchez-Maroño, and Alonso-Betanzos. Distributed feature selection: An application to microarray data classification. *Applied Soft Computing*, 30:136–150, 2015.

[4] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[5] V. Bolón-Canedo, N. Sánchez-Maroño, and J. Cerviño-Rabuñal. Scaling up feature selection: a distributed filter approach. In *Conference of the Spanish Association for Artificial Intelligence*, pages 121–130. Springer, 2013.

[6] V. Bolón-Canedo, N. Sánchez-Marono, and J. Cervino-Rabunal. Toward parallel feature selection from vertically partitioned data. In *Proceedings of ESANN 2014, pp. 395–400*, 2014.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *arXiv preprint arXiv:1106.1813*, 2011.

[8] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, J.M. Benítez, and F. Herrera. A review of microarray datasets and applied feature selection methods. *Information Sciences*, 282:111–135, 2014.

Cuadro IV
Características de los conjuntos del tercer bloque de experimentación, con muestras sintéticas en la clase mayoritaria

Conjunto	Tipo	Nº Muestras	Nº Características	Nº Clases	% clase mayoritaria
Arrhythmia	Normal	452	279	16	54.2
Connect4	Normal	67557	42	3	65.83
Musk2	Normal	6598	168	2	63
Nomao	Normal	34465	120	2	71.44
Ozone	Normal	2536	72	2	97.12
Spambase	Normal	4601	57	2	60.6
Weight	Normal	4024	90	5	34.04
Brain	Microarray	21	12625	2	67
CNS	Microarray	60	7129	2	75
Colon	Microarray	62	2000	2	65
Gli85	Microarray	85	22283	2	69
Ovarian	Microarray	253	15154	2	64

Conjunto		Sin SMOTE	Solo SMOTE minoritaria	SMOTE minoritaria + mayoritaria
Arrhythmia	Precisión (media)	**63.07**	62.63	62.07
	Kappa (media)	0.739	0.746	**0.774**
	Precisión (max)	68.34 (CFS+Naive)	67.68 (CFS+Naive+100)	**68.74** (Rel+SVM+300+40)
	Kappa (max)	**1** (Todos+IB1)	**1** (Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)
Connect4	Precisión (media)	66.14	**66.20**	62.15
	Kappa (media)	0.163	0.430	**0.478**
	Precisión (max)	76.12 (Cons+C4.5)	77.88 (Cons+C4.5+600)	**78.21** (Cons+C4.5+100+20)
	Kappa (max)	0.777 (Cons+C4.5)	**1** (Cons+IB1+Todos)	**1** (Cons+IB1+Todos+Todos)
Musk2	Precisión (media)	**89.41**	87.35	85.17
	Kappa (media)	0.672	0.687	**0.739**
	Precisión (max)	95.62 (Cons+C4.5)	95.44 (CFS+C4.5+100)	**95.72** (Cons+C4.5+100+40)
	Kappa (max)	**1** (Todos+IB1)	**1** (Varios+IB1+Varios)	**1** (Varios+IB1+Todos+Todos)
Nomao	Precisión (media)	**85.75**	84.22	83.94
	Kappa (media)	0.646	0.693	**0.723**
	Precisión (max)	94.34 (Cons+C4.5)	94.52 (Cons+C4.5+300)	**94.70** (Cons+C4.5+Auto+100)
	Kappa (max)	0.964 (Cons+C4.5)	**1** (Cons+IB1+Todos)	**1** (Cons+IB1+Todos+Todos)
Ozone	Precisión (media)	**92.36**	91.88	88.58
	Kappa (media)	0.271	0.506	**0.628**
	Precisión (max)	96.99 (Cons+Todos)	**97.02** (Rel+SVM+600)	**97.02** (Rel+SVM+300+Varios)
	Kappa (max)	0.982 (Info+Rel)	**1** (CFS+IB1+Todos)	**1** (CFS+IB1+Todos+Todos)
Spambase	Precisión (media)	85.92	**87.45**	87.11
	Kappa (media)	0.800	0.858	**0.859**
	Precisión (max)	92.27 (CFS+C4.5)	92.79 (CFS+C4.5+300)	**92.85** (CFS+C4.5+Auto+100)
	Kappa (max)	0.998 (Cons+IB1)	**1** (Cons+IB1+Auto)	**1** (Cons+IB1+Varios+Varios)
Weight	Precisión (media)	84.40	85.44	**86.33**
	Kappa (media)	0.803	0.829	**0.851**
	Precisión (max)	**100** (Cons+Varios)	**100** (Cons+Varios+Todos)	**100** (Varios+C4.5+Todos+Todos)
	Kappa (max)	**1** (Cons+Varios)	**1** (Varios+IB1+Varios)	**1** (Varios+IB1+Varios+Varios)
Brain	Precisión (media)	59.11	**62.27**	62.14
	Kappa (media)	0.882	0.889	**0.904**
	Precisión (max)	82.86 (Info+C4.5)	**94.29** (CFS+C4.5+Todos)	**94.29** (CFS+C4.5+Todos+Todos)
	Kappa (max)	**1** (CFS+Todos, Info+Todos)	**1** (CFS+Todos, Info+Todos)	1 (CFS+Todos, Info+Todos)
CNS	Precisión (media)	55.38	**58.98**	58.13
	Kappa (media)	0.867	0.910	**0.921**
	Precisión (max)	65 (Cons+SVM)	**70** (CFS+C4.5+600)	**70** (CFS+Naive+300+20)
	Kappa (max)	**1** (Todos+IB1)	**1** (Todos+IB1+Todos)	1 (Todos+IB1+Todos+Todos)
Colon	Precisión (media)	**77.62**	77.17	76.81
	Kappa (media)	0.861	0.913	**0.920**
	Precisión (max)	86.67 (Naive+Rel)	87.62 (Info+Naive+Auto)	**88.67** (Rel+Naive+Auto+20)
	Kappa (max)	1 (Todos+IB1)	**1** (Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)
Gli85	Precisión (media)	77.99	**80.40**	80.08
	Kappa (media)	0.944	0.969	**0.976**
	Precisión (max)	85.71 (Info+SVM)	88.57 (Cons+IB1+Varios)	**90** (CFS+IB1+Auto+100)
	Kappa (max)	**1** (Todos+IB1)	**1** (Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)
Ovarian	Precisión (media)	97.68	98.04	**98.09**
	Kappa (media)	0.989	0.994	**0.995**
	Precisión (max)	**100** (CFS+SVM)	**100** (CFS+SVM+Todos)	**100** (CFS+Varios+Varios)
	Kappa (max)	**1** (CFS+SVM, Todos+IB1)	**1** (CFS+SVM+Todos, Todos+IB1+Todos)	**1** (Todos+IB1+Todos+Todos)

Cuadro V
Resumen de los resultados obtenidos utilizando las tres aproximaciones, sin utilizar SMOTE, usando SMOTE sólo en la clase minoritaria–utilización estándard– y usando SMOTE en todas las clases.

Local sets for multi-label instance selection*

Álvar Arnaiz-González
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
alvarag@ubu.es

José-Francisco Díez-Pastor
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
jfdpastor@ubu.es

Juan J Rodríguez
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
jjrodriguez@ubu.es

César García-Osorio
Dpto. Ingeniería Civil
Universidad de Burgos
Burgos, Spain
cgosorio@ubu.es

***Abstract*—This is a summary of our article published in Applied Soft Computing [1], presented to the Multi-Conference CAEPIA'18 KeyWorks.**

***Index Terms*—multi-label classification, data reduction, instance selection, nearest neighbor, local set**

I. Summary

Single-label classification is a predictive data mining task that consists of assigning a label to an instance for which the label is unknown. Multi-label classification presents a similar task, although the difference is that the instances have a collection of labels, known as a labelset, rather than only one label. The maximum size of the labelset is determined by the number of different labels in the data set. The aforementioned labelset concept can also be considered as a sequence of binary output attributes (as many attributes as there are labels in the whole data set). Each attribute indicates whether the corresponding label is applicable to the instance. Only one of the attributes is active in single-label problems, while several attributes may be active in multi-label problems [5]. In other words, the labels in multi-label learning are not mutually exclusive [8]. This feature implies a much harder and more challenging problem, due to the high relevance of the relations between the different labels [10].

Despite the well-established usefulness of single-label instance selection, there are still very few methods for multi-label classification. To the best of our knowledge, only two instance selection methods for multi-label have been developed. Since both algorithms are based on Wilson Editing (ENN) [9], to avoid any confusion with the acronyms, we refer to them by the initials of their authors: the KADT method [6] and the CRJH method [3]. In this paper, we have attempted to fill that gap by proposing a new technique for computing local sets in multi-label data sets. This new proposal was used to adapt two single-label instance selection methods, LSSm and LSBo, for multi-label problems. The adaptation was tested against the few instance selection methods existing for multi-label learning and against the classifiers (MLkNN [11] and IBLR-ML [4]) trained on the whole data sets.

The main contributions of the paper were:

We would like to thank the *Ministerio de Economía y Competitividad* of the Spanish Government for financing the project TIN2015-67534-P (MINECO/FEDER, UE) and the *Junta de Castilla y León* for financing the project BU085P17 (JCyL/FEDER, UE) both cofinanced from European Union FEDER funds.

- The definition of the local set concept in the context of multi-label data sets.
- The proposal that defines two new instance selection methods, based on the adaptation of single-label classification algorithms to multi-label learning: LSBo and LSSm [7].
- The experimental evaluation of the new algorithms. The new methods were compared with the few existing algorithms.

Instance selection methods usually focus on boundaries between classes. Boundaries are the keystone of the predictive process, because they define whether an instance belongs to one class or another. The simplest classification problem is a binary class data set: there is only one class, thus one instance may or may not belong to it (in practice, this task is similar to determining one of two categories to which the instance belongs). In multi-class classification, more classes are present but, as in the previous case, each instance can only belong to one. The challenge that emerges in multi-label data sets is that instances can belong to more than one class at the same time, which blurs the boundaries (because different labels overlap).

The concept of local set has been used for designing several instance selection algorithms for single-label data sets [2], [7]. Local sets are defined by the nearest enemy, which is straightforward to compute in single-label data sets. The problem with multi-label data sets is how the nearest enemy is defined: it is no trivial task, because every single instance has a set of labels, rather than only one, as in single-label classification. An intuitive solution would be to consider each labelset (the vector of labels of an instance) as a class in itself. However, the results of several experiments have demonstrated that this approach is of little or no use, due to the large amount of different labelsets that multi-label data sets usually have. For example, for a data set with three different classes, the number of different labelsets could be up to $2^3 = 8$; if a data set has nine labels, the number of labelsets could reach $2^9 = 512$. The number of possible labelsets therefore increases exponentially with the number of labels. Hence, local sets calculated in this way will be too small (many of them only made up of a single instance) and, therefore, the algorithms based on local sets would not work properly.

The proposal that was presented in the paper was to use the Hamming loss (calculated over labelsets) to measure the

degree of difference in the labelsets[1]. If the Hamming loss between the labelsets of two instances is greater than a predefined threshold, the instances are considered to be of different '*classes*'. This concept of *class* can be seen as a '*soft-class*' in the same sense as in regression data sets. The Hamming distance is computed as follows:

$$Hamming\ distance(\mathbf{a}, \mathbf{b}) = |\omega_{\mathbf{a}} \triangle \omega_{\mathbf{b}}| \quad (1)$$

where, $\omega_{\mathbf{a}}$ and $\omega_{\mathbf{b}}$ are the labelsets of instances $\mathbf{a}$ and $\mathbf{b}$, respectively, and $\triangle$ is the symmetric difference between two labelsets[2].

The Hamming distance according to the previous definition is a whole number. The Hamming loss value is commonly used in multi-label learning HL $\in [0, 1]$.

$$Hamming\ loss(\mathbf{a}, \mathbf{b}) = \frac{1}{|\Omega|}|\omega_{\mathbf{a}} \triangle \omega_{\mathbf{b}}| \quad (2)$$

Pseudocode 1 shows the proposed method for local set calculation in multi-label data sets. It has two inputs: the multi-label data set and the value of the Hamming loss threshold that determines when two labelsets are considered distinct. The function has two outputs: an array of local sets and an array of nearest enemies. Every single instance has its local set (made of one or more instances) and its nearest enemy.

Algorithm 1: Function `computeLocalSets`: computes the local sets of a multi-label data set.

Input: A training set $X = \{(\mathbf{x}_1, \omega_1), ..., (\mathbf{x}_n, \omega_n)\}$, a threshold θ

Output: The local sets $LSS = \{\mathbf{lss}_1, ..., \mathbf{lss}_n\}$, the nearest enemy of each instance $NE = \{ne_1, ..., ne_n\}$

```
1  for i ∈ {1...n} do
2      lss_i ← ∅
3      dist_ne_i ← ∞
       /* Find the nearest enemy of x_i      */
4      for j ∈ {1...n} do
5          d ← EuclideanDistance(x_i, x_j)
6          if HammingLoss(ω_i, ω_j) > θ and d < dist_ne_i
           then
7              ne_i ← x_j
8              dist_ne_i ← d
       /* Compute the local set of x_i       */
9      for j ∈ {1...n} do
10         if EuclideanDistance(x_i, x_j) < dist_ne_i then
11             lss_i ← lss_i ∪ {x_j}
```

After the calculation of local sets, any local set-based algorithm can be used without changes. In the experimental study, we considered LSSm and LSBo, because their use of local sets is more robust than the use of local sets in ICF (the heuristic used in ICF has fundamental problems that were reported in [7]).

The experimental study used a broad range of data sets from different domains, several multi-label measures and statistical tests. The results revealed the two main benefits of our proposal: *i*) HDLSSm, as an edition algorithm, is not only capable of outperforming the other instance selection methods in terms of its results, but it also capable of outperforming the classifier trained with the whole data set; *ii*) HDLSBo, as a condensed algorithm, achieved a remarkable compression, while maintaining a statistically equivalent performance to the performance of the other methods. Furthermore, the existence of a threshold for controlling local set sizes implies an adaptable and versatile proposal.

[1]We decided to use Hamming loss, because its computation is fast and it is a commonly used measure in multi-label learning.

[2]The symmetric difference is the exclusive disjunction (XOR) of two sets, that is the set of all elements that are in one set, but not in the other set.

References

[1] Álvar Arnaiz-González, José F. Díez-Pastor, Juan J. Rodríguez, and César García-Osorio. Local sets for multi-label instance selection. *Applied Soft Computing*, 68:651 – 666, 2018.

[2] Henry Brighton and Chris Mellish. *On the Consistency of Information Filters for Lazy Learning Algorithms*, pages 283–288. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[3] Francisco Charte, Antonio J. Rivera, María J. del Jesus, and Francisco Herrera. MLeNN: A first approach to heuristic multilabel undersampling. In *Intelligent Data Engineering and Automated Learning – IDEAL 2014: 15th International Conference, Salamanca, Spain, September 10-12, 2014. Proceedings*, pages 1–9, Cham, 2014. Springer International Publishing.

[4] Weiwei Cheng and Eyke Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2):211–225, Sep 2009.

[5] Francisco Herrera, Francisco Charte, Antonio J. Rivera, and María J. del Jesus. *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer Publishing Company, Incorporated, 2016.

[6] Sawsan Kanj, Fahed Abdallah, Thierry Denœux, and Kifah Tout. Editing training data for multi-label classification with the k-nearest neighbor rule. *Pattern Analysis and Applications*, 19(1):145–161, 2016.

[7] Enrique Leyva, Antonio González, and Raúl Pérez. Three new instance selection methods based on local sets: A comparative study with several approaches from a bi-objective perspective. *Pattern Recognition*, 48(4):1523 – 1537, 2015.

[8] Newton Spolaôr, Maria Carolina Monard, Grigorios Tsoumakas, and Huei Diana Lee. A systematic review of multi-label feature selection and a new method based on label construction. *Neurocomputing*, 180:3 – 15, 2016. Progress in Intelligent Systems Design Selected papers from the 4th Brazilian Conference on Intelligent Systems (BRACIS 2014).

[9] Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-2(3):408–421, July 1972.

[10] Zoulficar Younes, Fahed Abdallah, Thierry Denœux, and Hichem Snoussi. A dependent multilabel classification method derived from the k-nearest neighbor rule. *EURASIP Journal on Advances in Signal Processing*, 2011(1):1–14, 2011.

[11] Min-Ling Zhang and Zhi-Hua Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038 – 2048, 2007.

Emerging topics and challenges of learning from noisy data in non-standard classification: A survey beyond binary class noise

Ronaldo C. Prati
Center of Mathematics, Computer Science and Cognition
Federal University of ABC
Santo André, São Paulo, Brazil
ronaldo.prati@ufabc.edu.br

Julián Luengo
Data Science and Computational Intelligence Institute
University of Granada
Granada, Spain
julianlm@decsai.ugr.es

Francisco Herrera
Data Science and Computational Intelligence Institute
University of Granada
Granada, Spain
herrera@decsai.ugr.es

***Abstract*—This is a summary of our article published in Knowledge and Information Systems [1] to be part of the MultiConference CAEPIA'18 Key Works.**

***Index Terms*—Data preprocessing, non-standard classification, noise data, multiclass classification, multi-instance learning, multi-label learning, multitask problems, ordinal classification, data streams, non-stationary environments**

I. Summary

Learning from noisy data is an important topic in machine learning, data mining and pattern recognition, as real world data sets may suffer from imperfections in data acquisition, transmission, storage, integration and categorization. Indeed, over the last few decades, noisy data has attracted a considerable amount of attention from researchers and practitioners, and the research community has developed numerous techniques and algorithms in order to deal with the issue [2]–[4].

These approaches include the development of learning algorithms which are robust to noise as well as data pre-processing techniques that remove or "repair" noisy instances. Although noise can affect both input and class attributes, class noise is generally considered more harmful to the learning process, and methods for dealing with class noise are becoming more frequent in the literature [3].

Class noise may have many reasons, such as errors or subjectivity in the data labeling process, as well as the use of inadequate information for labeling. For instance, in some medical applications, the true status of some diseases can only be determined by expensive or intrusive procedures, some of which can only be carried out after a patient's death. Another reason is that data labeling by domain experts is generally costly, and several applications use labels which are automatically defined by autonomous taggers (e.g., sentiment analysis polarization [5]), or by non-domain experts. This approach is common in, e.g., social media analysis [5], where hashtags used by users or information provided by a pool of non-domain experts (crowdsourcing) are used to derive labels.

Even though class noise is predominant in the literature (see [2], [6] for recent surveys and comparison studies), most of the research has been focused on noise handling in binary class problems. However, new real-life problems have motivated the development of classification paradigms beyond binary classification [7]. These paradigms include ordinal class [8], multiclass [9], multilabel [10] and multi-instance [11] as well as learning from data streams and non-stationary environments [12] and joint exploiting related tasks [13]. Due to the ubiquity of noise, it is of fundamental importance to better understand the relationships and implications of class noise within these paradigms. Each paradigm has its own particularities which impose new challenges and research questions for noise handling. Although research for class noise handling in these paradigms is somewhat present in the literature, it remains quite scarce and requires general discussion of issues, challenges and research practices regarding it.

The related paper aims to discuss open-ended challenges and future research directions for learning with class noise data, focusing on the aforementioned non-binary classification domains. The main contributions of such a paper are:

- We discuss some current research, as well as the need of adaptation or development of new techniques for handling class noise within non-binary classification paradigms.
- We also discuss issues related to the simulation of noise scenarios (inclusion of artificial noise) within these paradigms, an experimental artifact frequently adopted

This work have been partially supported by the São Paulo State (Brazil) research council FAPESP under project 2015/20606-6, the Spanish Ministry of Science and Technology under project TIN2014-57251-P and the Andalusian Research Plan under project P12-TIC-2958.

for analysis of noise dealing techniques. These issues are important for simulating noise scenarios that may occur in real world applications, and can serve as the basis for uniforming procedures by providing an objective ground in order to assess the robustness of the learning methods.
- We present some important open-ended issues and offer some possible solutions to the existing problems.

We are aware of some studies already considering multiclass noise problems. Different multiclass noise patterns impose numerous challenges, some of them infrequently addressed in the literature. Even state of the art methods for dealing with binary class noise present considerable variation in performance when considering different multiclass noise patterns at the same noise ratio. Despite this, these issued are seldom considered in the literature. In the related paper we focus on some of aspects that could be studied further, providing a guideline of open challenges for researchers, such as:

- Would these different types of noise patterns pose the same or different challenges when dealing with multiple class noise?
- Which one would be more difficult to tackle?
- Which aspects of the problem would be more affected by considering different noise multiclass pattern?
- How do existing methods behave considering these different noise patterns?

One interesting topic for further research is how to extend methods, originally developed only for binary class, to the multiclass case. Some data transformation approaches for transforming multiclass to binary problems, e.g., One-versus-One (OVO) or One-versus-ALL (OVA), could be applied [14]. However, research on this topic generally involves random noise completely at random, with uniform class noise distribution. Investigating how these approaches are affected by different noise patterns is an interesting topic for research. For instance, when applying a filter using a OVA decomposition, does the order in which class noise is removed matter? If so, is this influence stronger for different noise distribution among the classes?

Another open-ended problem is the relationship with imbalanced classification [15] and multiclass noise. It is reported in the literature that noise in minority classes is more harmful than in majority classes [16]. However, multiclass imbalance [17] has further issues to consider, as multiple predominant or infrequent classes may occur. It is unclear what learning difficulties multiclass noise can cause under highly imbalanced class distributions, and how to handle it effectively is an open-ended issue. Furthermore, different noise patterns can change the observed class ratio, which may influence the behavior of class imbalance techniques. Uniform class noise, for instance, may mask the observed class ratio of multiple rare classes even for low noise levels. Default class may also introduce an artificial predominant class, thus generating an artificial imbalanced problem due to the presence of noise. Possible ways to handle noise in imbalanced problems include cost sensitive noise handling [18], [19], attributing and the development of class ratio aware filtering approaches [20] considering the multiclass context.

We believe this discussion will encourage researchers and practitioners to explore the problem of class noise handling in new scenarios and different learning paradigms in more detail.

References

[1] R. C. Prati, J. Luengo, and F. Herrera, "Emerging topics and challenges of learning from noisy data in non-standard classification: A survey beyond binary class noise," *Knowledge and Information Systems*, vol. in press, 2018.

[2] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," vol. 25, no. 5, pp. 845–869, 2014.

[3] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 177–210, 2004.

[4] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.

[5] B. Liu, *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge University Press, 2015.

[6] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artif. Intell. Rev.*, vol. 33, no. 4, pp. 275–306, 2010.

[7] J. Hernández-González, I. Inza, and J. A. Lozano, "Weak supervision and other non-standard classification problems: a taxonomy," *Pattern Recognit. Lett.*, vol. 69, pp. 49–55, 2016.

[8] P. A. Gutiérrez, M. Perez-Ortiz, J. Sanchez-Monedero, F. Fernández-Navarro, and C. Hervas-Martinez, "Ordinal regression methods: survey and experimental study," vol. 28, no. 1, pp. 127–146, 2016.

[9] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of artificial intelligence research*, vol. 2, pp. 263–286, 1995.

[10] F. Herrera, F. Charte, A. J. Rivera, and M. J. del Jesus, *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer, 2016.

[11] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Sánchez-Tarragó, and S. Vluymans, *Multiple Instance Learning: Foundations and Algorithms*. Springer, 2016.

[12] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: a survey," vol. 10, no. 4, pp. 12–25, 2015.

[13] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 615–637, 2005.

[14] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition," *Knowledge and information systems*, vol. 38, no. 1, pp. 179–206, 2014.

[15] R. C. Prati, G. E. A. P. A. Batista, and D. F. Silva, "Class imbalance revisited: a new experimental setup to assess the performance of treatment methods," *Knowl. Inf. Syst.*, vol. 45, no. 1, pp. 247–270, 2015.

[16] J. Van Hulse and T. Khoshgoftaar, "Knowledge discovery from imbalanced and noisy data," *Data Knowl. Eng.*, vol. 68, no. 12, pp. 1513–1542, 2009.

[17] S. Wang and X. Yao, "Multiclass imbalance problems: Analysis and potential solutions," vol. 42, no. 4, pp. 1119–1130, 2012.

[18] X. Zhu and X. Wu, "Cost-guided class noise handling for effective cost-sensitive learning," in *IEEE International Conference on Data Mining (ICDM)*. IEEE, 2004, pp. 297–304.

[19] X. Zhu, X. Wu, T. M. Khoshgoftaar, and Y. Shi, "An empirical study of the noise impact on cost-sensitive learning." in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 7, 2007, pp. 1168–1173.

[20] J. A. Sáez, J. Luengo, J. Stefanowski, and F. Herrera, "Smote–ipf: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering," *Inf. Sci.*, vol. 291, pp. 184–203, 2015.

Training Set Selection for Monotonic Ordinal Classification

José-Ramón Cano
Dept. of Computer Science
University of Jaén
Linares, Spain
jrcano@ujaen.es

Salvador García
Dept. of Computer Science and Artificial Intelligence
University of Granada
Granada, Spain
salvagl@decsai.ugr.es

***Abstract*—This is a summary of our article published in Data & Knowledge Engineering [1] to be part of the MultiConference CAEPIA'18 KeyWorks.**

***Index Terms*—Monotonic Classification, Ordinal Classification, Training Set Selection, Data Preprocessing, Machine Learning**

I. Summary

Learning with ordinal data sets has increased the attention of the machine learning community in recent years. These data sets are characterized by the presence of an ordinal output and they are commonly found in real life.

Monotonic classification is an ordinal classification problem where monotonic constraints are present in the sense that a higher value of a feature in an instance, fixing the other values, should not decrease its class assignment [2]. Monotonicity is a property commonly found in many environments of our lives like economics, natural language or game theory [4]. A classical example of monotonicity is in the case of bankruptcy prediction in companies, where appropriate actions can be taken in time, considering the information based on financial indicators taken from their annual reports. The comparison of two companies where one dominates the other on all financial indicators shows clearly where the monotonicity is present, which supposes that the overall evaluation of the second cannot be higher than the evaluation of the first. This strategy could be applied to the credit rating score used by banks as well as for the bankruptcy prediction strategy .

In the specialized literature we can find multiple monotonic classifiers proposed. As a restriction, some of them require the training set to be purely monotone to work properly. Other classifiers can handle non-monotonic data sets, but they do not guarantee monotone predictions.

In addition, real-life data sets are likely to have noise, which obscures the relationship between features and the class. This fact affects the prediction capabilities of the learning algorithms which learn models from those data sets.

This work was supported by TIN2014-57251-P, by the Spanish "Ministerio de Economía y Competitividad" and by "Fondo Europeo de Desarrollo Regional" (FEDER) under Project TEC2015-69496-R and the Foundation BBVA project 75/2016 BigDaPTOOLS.

In order to address these shortcomings and to test the prediction competences of the monotonic classifiers, the usual trend is to generate data sets which completely satisfy the monotonicity conditions. The intuitive idea behind this is that the models trained on monotonic data sets should offer better predictive performance than the models trained on the original data. In the specialized literature, we find two possible techniques to generate monotonic data sets. Monotonic data sets can be created by generating artificial data [5] and by relabeling the real data [6]. The latter restores the monotonicity of the data set by changing the class labels in those instances which violate the monotonicity constraints. Class relabeling is the only approach which can be applied in real life data sets, and has shown promising results in the literature.

As an alternative to relabel, Training Set Selection (TSS) is known as an application of instance selection methods [3] over the training set used to build any predictive model. The effects produced by TSS are: reduction in space complexity, decrease in computational cost and the selection of the most representative instances by discarding noisy ones.

In this paper we propose a TSS algorithm to manage monotonic classification problems, called Monotonic Training Set Selection (MonTSS). MonTSS can be considered as the first in the literature for performing TSS in monotonic classification problems. It is a data preprocessing technique which, by means of a suitable TSS process for monotonic domains, offers an alternative without modifying the class labels of the data set, it instead removes harmful instances. MonTSS incorporates proper measurements to identify and select the most suitable instances in the training set to enhance both the accuracy and the monotonic nature of the models produced by different classifiers.

The whole process is presented in Figure 1, and as can be seen it is composed of three stages:

1) The MonTSS process starts with a preprocessing step where MonTSS analyzes the original data set by quantifying the relationship between each input feature and the output class. This relation is estimated with a metric called Rank Mutual Information (RMI). With it, we

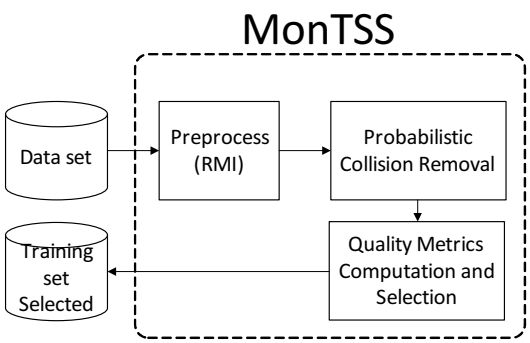


Fig. 1. MonTSS process.

know the features which have a real direct or inverse monotonic relation with the class or no relation as well (including unordered categorical features). The RMI value is evaluated in the training data set to decide which features are used in the computation of collisions between instances.

In essence, rank mutual information can be considered as the degree of monotonicity between features A_1,...,A_f and the feature class Y. Given any feature A_j and feature class Y, the value of RMI for the feature A_j is calculated as follows:

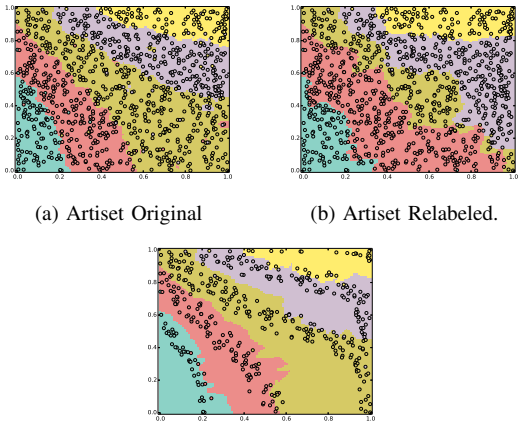
$$\text{RMI}(A_j, Y) = -\frac{1}{n}\sum_{i=1}^{n} log \frac{\#[x_i]^{\leq}_{A_j} \cdot \#[x_i]^{\leq}_{Y}}{n \cdot \#([x_i]^{\leq}_{A_j} \cap [x_i]^{\leq}_{Y})} \quad (1)$$

where n is the number of instances in data set D, $[x_i]^{\leq}_{A_j}$ is the set formed by all the instances of the set D whose feature A_j is less or equal than feature A_j of instance x_i, and $[x_i]^{\leq}_{Y}$ is the set composed of the instances of the set D whose feature class Y is less or equal than feature class Y of instance x_i.

2) In the second stage, the probabilistic collision removal mechanism is applied, which eliminates most of the instances which produce collisions. The remaining instances are used as input in the last stage.
3) Here, the quality metrics are computed and based on them, the selection procedure is developed considering the following rule:

$$\text{Select } x_i = \begin{cases} \text{true} & \textbf{if } \text{Del}(x_i) < \text{Infl}(x_i) \\ & \text{or } \text{Del}(x_i) \geq 0.9 \\ \text{false} & \text{otherwise.} \end{cases} \quad (2)$$

The rationale behind this rule is to retain the instances which are closer to the class boundaries, using a straightforward threshold of 0.9 which is independent from the diversity of their neighborhood. Furthermore, a relationship between $\text{Del}(x_i)$ and $\text{Infl}(x_i)$ can be easily established as they represent a measurement in the same range of the relative rate of the situation and the neighborhood variety of every instance. In this respect, the rule is built as a function of both measures. As a result, for instance, the rule preserves the instances belonging to central areas if there are instances of other classes around.



(a) Artiset Original (b) Artiset Relabeled.

(c) Artiset MonTSS.

Fig. 2. Artificial data set (Artiset) preprocessed by Relabeling and MonTSS with the borders calculated by MkNN with 3 neighbors.

We have compared the results offered by well-known classical monotonic classifiers over 30 data sets with and without the use of MonTSS as a data preprocessing stage. As graphical example of use we present the Fig. 2.

The results show that MonTSS is able to select the most representative instances, which leads monotonic classifiers to always offer equal or better results than without preprocessing.

MonTSS is able to select the most representative instances independently of the classifier to be applied later. This leads monotonic classifiers to always offer equal or better results than without preprocessing. Furthermore, data related metrics are notably improved, fully satisfying the monotonicity restrictions without affecting or modifying the nature of the original data. At the same time, it reduces the number of non-comparable pairs of instances and the size of the training data sets before the learning stage starts.

References

[1] J.-R. Cano and S. García, "Training set selection for monotonic ordinal classification," Data & Knowledge Engineering, vol. 112, pp. 94–105, 2017.
[2] H. Daniels and M. Velikova,"Monotone and partially monotone neural networks," IEEE Transactions on Neural Networks, vol. 21, num. 6, pp. 906–917, 2010.
[3] J.-R. Cano, N.R. Aljohani, R.A. Abbasi, J.S. Alowidbi and S. García, "Prototype selection to improve monotonic nearest neighbor," Engineering Applications of Artificial Intelligence, vol. 60, pp. 128–135, 2017.
[4] W. Kotłowski and R. Słowiński, "On nonparametric ordinal classification with monotonicity constraints," IEEE Transactions on Knowledge and Data Engineering, vol. 25, num. 11, pp. 2576–2589, 2013.
[5] R. Potharst, A. Ben-David and M.C. van Wezel, "Two algorithms for generating structured and unstructured monotone ordinal data sets," Engineering Applications of Artificial Intelligence, vol. 22, num. 4-5, pp- 491–496, 2009.
[6] M. Rademaker, B. De Baets and H. De Meyer, "Optimal monotone relabelling of partially non-monotone ordinal data," Optimization Methods and Software, vol. 27, num. 1, pp. 17–31, 2012.

Data source analysis in mood disorder research

Pavél Llamocca Portella
Universidad Complutense de Madrid
pavel.llamocca@hotmail.com

Axel Junestrand
Universidad Complutense de Madrid
axel.junestrand@hotmail.com

Milena Čukić
University of Belgrade
milena.cukic@gmail.com

Diego Urgelés
Hospital Ntra. Sra. De la Paz
diego.urgeles@gmail.com

Victoria López López
Universidad Complutense de Madrid
vlopezlo@ucm.es

***Abstract*— Mood disorders have been a relevant topic for the last few years. Nowadays, there are projects in the mental health area which are supported by technological devices that improve the efficiency of treatments by effortlessly allowing the gathering of biological and psychological indicators from patients. One of the goals of this document is to describe the most common methods for collecting most of those indicators and to study which of them can be applied to the Bip4Cast project. The purpose of this article is to analyze the sources of information that have been used successfully in the study of emotional disorders as well as alternative sources of information from the monitoring of movement and sounds in the patient's environment. This article shows the results of the analysis of traditional information sources. The results show a lack of precision in the data on fundamental variables such as sleep quality and motor activity. Therefore, the study demonstrates the need to include new sources of information to increase the quality of the data before applying crisis prediction algorithms. The need to monitor the sleep and movement of patients in order to achieve a sufficient quality in the source data from the evolutionary analysis of patients is concluded.**

Keywords—bipolar disorder; mood disorder; data gathering; machine learning; data analysis.

I. Introduction

The quality of treatments in mood disorders has acquired a high attention for researchers in the mental health field. However, despite current efforts, there is still plenty of room for improvement. Many practicians agree that knowing how patients react to treatments in advance and predicting when their mood could vary significantly are two of the most important issues to solve in order to ensure the quality of new treatments.

The proposal of the Bip4Cast project is to keep using the current monitoring of personal sessions between patients and psychiatrists, but also to add new data sources and their analysis to improve the prediction of Bipolar Disorder crises in patients. The main idea is to get advantage of the new developments in data gathering, data cleaning, and Machine Learning to monitor a set of patients and make a new approach with these data. The patients are encouraged to follow certain methods for gathering psychological and biological indicators during a particular period of time. The goal is to analyze the data gathered in order to find some common patterns that could trigger a crisis. For the process of pattern detection, some Machine Learning tools and mathematical models are being used.

The goal of this document is to cover a discussion about some methods for gathering indicators and the feasibility of their usage in this project. Apart from this introduction, this document includes the following sections: section 2 presents the state of the art related research in mood changes and patient monitoring. In section 3, several methods for gathering psychological and biological parameters are described. Section 4 covers the preliminary analytics on the Bip4Cast data sets and, finally, section 5 includes the conclusions.

II. State of the Art

Several studies about Bipolar Disorder state that a relationship exists between the different behaviors of the patients before the occurrence of a crisis [1-2]. For example, during a manic or a depressive crisis, some of these studies agree that sleeping rates are very important indicators. Vocal features as well as the rate of speech are other important indicators and there are some studies stating that the pitch is lower in a depressed state [3]. Also, parameters like the time of exposure to dark or sunny places and the physical activity are considered.

In [4], the author introduces a mobile health system using several sensors for mood detection. In [5], the author presents a research that includes Machine Learning models in a mobile application in order to estimate the mood in depressive patients. However, no objective psychological or psychiatric markers are considered due to the recording of the data being done manually by patients. Furthermore, there is an interesting study which describes the use of electroencephalography for the gathering of brain signals. It also uses non-linear features

like Higuchi's Fractal Dimension and Sample Entropy to feed different Machine Learning methods [6]. In [7], a mobile application is presented for supporting the treatments of patients with Bipolar Disorder. Its key is to compare objective and subjective data. It records objective data using some features given by mobile phones like accelerometers and phone call rates. This information is used for predicting trends in the mood of the patients. However, the focus of this application is to record subjective data using a self-reporting approach.

At the present time, there are some projects within the scope of mental health which have similar approaches and try to obtain certain markers from which a common pattern can be inferred to help with the treatments. One of these projects is PSYCHE [8], whose main idea is for the patients to use a special garment made of a proper material, of which the main goal is to collect parameters from the patient in his/her daily life. The outcomes of PSYCHE are positive. However, patients said that the main inconvenient was to use the same clothes all the time, which implicates a high discomfort for the patient and therefore its non-use. Another really interesting project is MONARCA [9]. It emphasizes the use of mobile phones for the electronical monitoring of patients. The number of parameters that can be obtained through the use of a mobile phone is really high, but nevertheless, none of them are physiological parameters. Furthermore, after 3 years of activity with this application, an analysis of some non-functional requirements for the treatment of patients with Bipolar Disorder concluded that, for new developments, some details have to be taken into consideration in order to improve the ease of use, e.g.: ensuring that the patients have a data plan for their 3G connection, the need for teaching the clinicians how to operate the system, and the overheating of the smart phone from the use of an application that requires both GPS and Bluetooth.

Since few years ago, Body Sensor Networks (BSN) have made an appearance, which are a branch of wireless sensor networks (WSNs) that conform one of the core technologies of IoT developments in the healthcare system [10]. Its purpose is to provide an integrated hardware and software platform which facilitates the future development of pervasive monitoring systems. BSN allow the monitoring of patients by using a collection of tiny-powered and lightweight wireless sensor nodes. These nodes are placed on the skin and sometimes integrated with different garments, so that the patient's health-related data can be collected and transferred to the healthcare staff in real time. However, the development of this new technology in healthcare applications without considering security makes patient privacy vulnerable. For this reason, several research projects are currently being carried out to try to cover this vulnerability [11].

III. Data Source Analysis

All the research that is currently being conducted suggests a wide variety of indicators for taking mood disorder treatments into account. The indicators themselves and the way in which they are collected are strongly related. In this section, several data sources for gathering those indicators are described. Depending on the indicator they gather or the type of device, they are classified into 6 groups as shown below.

1. Smart wristband/smart band. These devices include a set of sensors which measure daily activity by means of accelerometers. They create variables such as an activity tracker (resting, moving or sleeping), a pedometer (steps taken and distance traveled), the calories burned, a sleep monitor (awake, slight and deep sleeping), the heart rate and the blood oxygen level. The most relevant variable measured from this type of device is the sleep indicator. Almost all researchers agree that sleep quality is the best indicator in Bipolar Disorder treatments. The CHOICE [12] study states that lower levels of depression are correlated with improvements in insomnia treatments, and on the other hand, high levels of mania are correlated with less need for sleep. Furthermore, a pilot randomized controlled trial demonstrated that sleep disturbance appears to be an important pathway contributing to Bipolar Disorder [13]. These data can easily be gathered from popular apps. Fig. 1 (a) shows one of the monitors used in the Bip4Cast project.

Fig. 1. Two sleep monitors in Bip4Cast (a- Garmin Vivofit3; b- Sleep Cycle for iPhone)

2. Medical bands. They allow measuring more parameters because they usually include more hardware and better features like atmospheric pressure (barometer), GPS location and magnetometer. There is some research which links episodes to disturbances in circadian rhythms and lifestyle regularity. Those indicators can be collected through these devices using the activity tracker or gyroscope. Furthermore, this research suggests that methods for tracking behavior, nutrition, blood pressure and lipid profile as well as physical/social activity and sleep-awake routines may improve treatments. In the Bip4Cast project we are using GENEActif v.1.2 for a total of 25 patients, (see [14] for more details about their use in Bip4Cast).

3. Mobile Sensors. In this group, any other kind of sensors that can be worn by the user on any other part of the body is included, e.g. for wearing on the leg, ActivPAL is a kind of device used to investigate the correlation between physical behaviors and chronic disease [15]. For using as a necklace, LeafUrban is an option (there are some versions for wearing on the wrist or attached to the clothes). It is a device designed for women and what it makes different from other devices is the tracking of the menstrual cycle, the fertility and the breathing [16]. For wearing on the head, ELF Emmit is a headband that helps the user improve the state of both mind and body by

using pulsed electromagnetic stimulation (PEMS) [17]. Relevant variables include skin and breath changes, electrocardiogram and respirogram data, stress level and menstrual cycle among others.

4. Sleep Activity Recording Devices. In this group, any device specialized in recording sounds and activities during sleep is included. There are hundreds of mobile applications that record sounds during sleep. One of the main objectives is to detect snoring, for which four of the most popular applications at the moment are SleepGenius, SleepCycle (see Fig. 1 (b)), SleepBot and SleepTime. However, there are other kinds of devices with different non-invasive designs, e.g. devices attached to the mattress which can track sounds as well as heart rate, breathing, movement, etc. The reason for using these methods is to improve sleeping conditions. Almost all of these methods have one parameter in common: "breathing", which allows the detection of snoring. Habitual snoring is a prevalent condition that is not only a marker for Obstructive Sleep Apnea (OSA) but can also lead to vascular risks [18]. Some researchers have found a relationship between OSA and Major Depressive Disorder/Bipolar Disorder.

5. Forms and Questionnaires. This group contains any method which uses a questionnaire or a form for the self-reporting of mood. In current literature, these methods were designed by psychiatrists and are presented as scales. There are several scales for detecting the risk of a euphoria episode outbreak: Altman Self-Rating Mania Scale (ASRM) [19], the Clinician-Administered Rating Scale for Mania (CARS-M), the Internal State Scale (ISS), the Self-Report Manic Inventory (SRMI), etc. For depression episodes, there are several scales, like the Patient Health Questionnaire (PHQ-9) [20]. All of them consist in questionnaires which can be performed by patients. This presents the opportunity of developing digital forms based on these patients in order to facilitate their use.

Scales for detecting the risk of euphoria or mania episodes, like the Young Mania Rating Scale (YMRS) and the Bech-Rafaelsen Mania Scale (MAS), or the Hamilton Depression Rating Scale (HDRS), which detects the risk of depression, are not included because they are performed by the clinician (however, for the scope of this project, these scales are included in normal monitoring sessions). All of these questionnaires collect variables from which it is possible to measure the presence and severity of mania, depression, affective, psychological and somatic symptoms. Fig. 2 shows the interface of an application developed for collecting these data. All the details about this work are in [21].

6. Mobile Apps / Time Consumption. This group includes mobile applications that support BPD treatments and/or record smart phone use. For the aim of this project, these mobile applications were classified into two subclasses: the first one, named Bipolar Disorder Apps (BPDA) in this document, includes any applications that have been developed for supporting the treatment itself, and the second one, named Time Consumption Apps (TCA) in this document, includes any application that is able to measure the time spent using a specific mobile application.

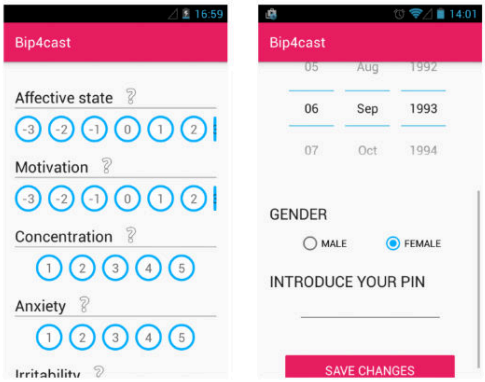


Fig. 2. First version of the app for collecting daily personal data

7. Conventional methods. Finally, patients are also being assessed periodically through interviews. This evaluation is done by psychiatrists in medical centers. For the patient, this does not imply any kind of alteration in the current treatment. However, the procedure will need the psychiatrist to send the collected data from those sessions to the data server. It is important to mention that in this phase, the Young Mania Rating Scale (YMRS) and the Hamilton Depression Rating Scale (HRDS) are included for detecting mania or depression episodes.

Also, psychiatrists can take advantage of these interviews for downloading the data recorded from wristbands and medical bands in order to later send them to a dedicated server (just in case these devices are not able to send the recorded data automatically).

IV. THE BIP4CAST PROJECT

Patients with Bipolar Disorder are characterized by a behavior which is difficult to predict. There is great deal of information which can be retrieved from biological, physiological and physical signals in order to detect episodes. Knowing which variables are correlated and which features or parameters are important is essential to build a model that will successfully predict the target of a study. The aim of this study is to investigate which features have the highest importance in health. In order to achieve this, Machine Learning algorithms and techniques are used for feature ranking.

The data used for this project is anonymized patient data gathered by psychiatrists at Clínica Nuestra Señora de la Paz in Madrid. All the data were available in an Excel file with different sheets. Even though 25 patients are already wearing a medical band (GENEActif 1.2) and we have developed an application for gathering their daily activity, for this study most of the data have been gathered in a supervised manner during medical appointments with four different patients that suffer from Bipolar Disorder. The goal for the future is that these data are both recorded by the psychiatrists in appointments and with the help of mobile applications. This way, the patients can actively participate in their own diagnosis. The data consist of 4 data sets: Episodes, which represents different episode periods in the patients (depression/mania) from a total of four

patients; YMRS data set, which contains Young Mania Rating Scale [22] data (to assess mania symptoms) from a total of 48 days; HDRS data set, which contains Hamilton Depression Rating Scale [23] data (for depression), also from a total of 48 days; Interview data set, which contains 728 registers about physical and psychological items, the latter including variables like anxiety, irritability or concentration problems, and the former including more objective data, as could be the number of cigarettes smoked by the patient or the time in which the patient woke up or went to bed. The last data set used in the study is Interventions. It includes data about all the medical interventions that different doctors have had with the patients, in a total of 92 registers. For the gathering of data included in the Interview data set, a mobile application [21] has been developed, which patients can use daily to store quantitative data (number of cigarettes, menstruation, etc.) and qualitative data (feeling of stress, anxiety, etc.). In the project, we have also included studies with data from a medical bracelet (GENEActif 2.1 for 25 patients) and an application for recording night sounds. The data collected by these last two exercises will be included in subsequent studies.

The programming language used in this project is Python 2.7, which has a lot of libraries that make data cleaning and visualization less complicated, as well as applying Machine Learning algorithms. The environment used is Jupyter Notebook [24]. Scikit-learn [25] is the Machine Learning library of choice for this project because it includes preprocessing and cross-validation tools as well as all the known baseline Machine Learning algorithms. This project is shared in a public GitHub repository, which can be found at [26].

A. *Data Cleaning*

The first step of the project consisted in the data cleaning which included the gathering of the data that we would be working with. In order to gather the data, we exported each sheet, from the Excel file that was given to us by the hospital, to csv format. The initial Excel file was divided into five different sheets: Episodes, YMRS, HDRS, Interview Data Set (IDS) and Interventions. In order to export them to a format readable by Pandas [27], we saved each sheet as CSV UTF-8 in Microsoft Excel. Some other improvement was done in relation to data cleaning: filling the empty values, converting them from Float to Integer and data type revision.

B. *Exploratory Data Analysis*

After the data cleaning, we performed an Exploratory Data Analysis, in order to visualize how the data behaved. We used histograms, heatmaps and scatterplots in this part. For instance, the YMRS data set correlation heatmap showed that aggressiveness and verbal expression were correlated. This could mean that if the patient talks a lot (excessive speech rate), this behavior would probably be accompanied by excessive energy or hyperactivity (Disruptive-Aggressive Behavior). The scattterplot matrix from the YMRS data showed that hyperactivity and irritability have a similar distribution as well as a correlation between verbal expression and euphoria. The HDRS data set analysis showed a similar distribution between suicide and precocious insomnia (difficulty of sleep early in the night). The HDRS data set correlation heatmap showed a high correlation between depressed mood and work: the less a patient is willing to work or do other activities, the more depressed he or she will probably feel. At this stage of the research, the best data set regarding both size and accuracy was the Interview Data Set (IDS). During its analysis, we found a clear linear relationship between the variables mood and motivation. The Intervention data set presented a lack of correlation between the level of relief in a patient and the GAF (Global Assessment of Functioning). A good summary of all these results can be read at [26].

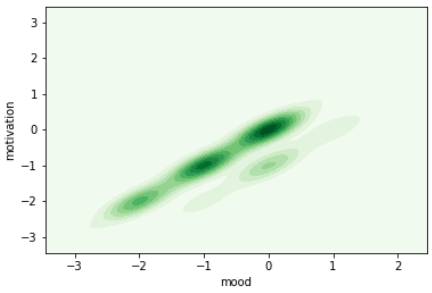


Fig. 3. 2D kernel density plot of *mood* and *motivation* in Interview data set

C. *Data Combination*

The goal of this phase was to find data set combinations that had enough data for the algorithms to process so that we could later see which data sets returned the highest accuracy. In order to get the combinations right, we defined a function that obtained the date of each entry and compared it with the different episodes of depression and mania in the Episode data set, which was the target of the prediction. For the entries or rows that were not recorded in the Episode data set, we assumed that the patient was in a euthymic state. This way, we got three possible states that a patient could be in: Depression (D), Mania (M) and Euthymic (N). For instance, with the HDRS and Episode combination we could see that when the patients had a depression episode, the value of depressed mood was much higher, almost always between 2 and 3, which meant that they either spontaneously reported feeling depressed or they communicated feeling depressed in a non-verbal way, judging by the rating items from the Hamilton Depression Rating Scale (HDRS). We could also see that when the patients were in a depression state (because of the predominance of green points on higher values of the work axis, where green represented patients in a state of depression), they started feeling loss of interest in activities they usually performed or there was a decrease in the time spent on work and other activities, which made perfect sense according to this rating scale.

D. *Application of the Algorithms*

The data sets on which we tested the algorithms were: YMRS (Young Mania Rating Scale data), HDRS (Hamilton Depression Rating Scale data), Interviews (interview data, IDS), Interventions (intervention data), YMRS-HDRS (combination of the YMRS and HDRS data) and Interviews-Interventions (combination of the Interview and Intervention

data). Fig.4 shows the diagram of the process followed during this study.

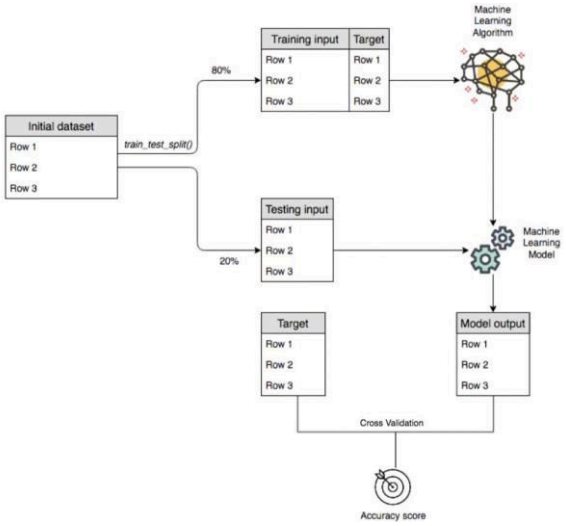


Fig. 4. Diagram of the Machine Learning algorithm application process

The algorithms that we used for this part were: Decision Tree [27], Random Forest [28], Support Vector Machines [29] and Logistic Regression [30]. The reason why these algorithms have been chosen for this project is explained in [26], in the section belonging to each algorithm.

The fact that these algorithms have been applied in this project does not mean that they are the best option for the classification of Bipolar Disorder states, but rather that they are the most suitable ones given the amount of data and the number of features used for the project. In future studies that make use of this project, if the data sets are larger it could be interesting to apply other algorithms too, like the Naïve Bayes [31] algorithm or any kind of Boosting algorithm [32], as to see how they perform on this particular classification problem.

Before applying each algorithm, as is necessary for every classification problem, the original data needed to be split into training and testing sets. Later, the training data would be used to train the prediction models and the testing data would be used to compare the output of the model with the real targets by cross-validation, a technique presented first by M. Stone in 1974 [33], that is used widely in Machine Learning for algorithm performance comparison.

The testing set is used for obtaining the accuracy of the model, as mentioned above, which is done by comparing the output obtained from the testing input and the real output of the testing set. In order to divide the original data sets into training and testing sets, we used the train_test_split() function from the scikit-learn library [25], where the test size represents the percentage of the data that is used for the testing set. After the algorithms were applied, the cross_val_score() function, which is also included in the scikit-learn library, was called in order to evaluate the score by k-fold cross-validation [33].

The best way to compare the accuracies obtained with the different algorithms on all the data sets was to make an algorithm performance matrix, which is shown in Table 1. This matrix showed that, in average, the data set that returned the best prediction accuracy (69%) with the algorithms was the Interview data set, as seen on the right column. The algorithm that performed the best, also in average (62%), was the Logistic Regression algorithm, as seen on the column farthest down.

Even though the algorithm that had the best accuracy average was the Logistic Regression, we stated that the Random Forest algorithm made the most accurate predictions in the sense that they were very reasonable given the behavior of the patients, which we tested with randomized data. These predictions made possible the implementation of a small program with the Random Forest classifier that we obtained, and which can be seen in [26].

TABLE I. ACCURACIES WITH DIFFERENT ALGORITHMS

Algorithms/Datasets	Decision Tree	Random Forest	SVM	Logistic Regression	Average
YMRS	36%	38%	38%	76%	47%
HDRS	78%	55%	36%	62%	58%
Interviews	70%	72%	68%	67%	69%
Interventions	44%	65%	57%	51%	54%
YMRS-HDRS	63%	63%	70%	33%	57%
Interviews-Interventions	67%	17%	42%	83%	52%
Average	60%	52%	52%	62%	

V. Conclusions and Future Work

Having a deep understanding of the data is essential in any Machine Learning project focused on a branch of medical science like psychiatry, where knowing which behaviors are normal and abnormal in the patients can help us create much more precise prediction models. The amount of data used and the nature of the data source are very important factors because with a larger suitable amount of data we will be able to get prediction accuracies with a much higher level of confidence. In the same way that understanding the data is important, having a deep perception of the theory behind each algorithm used, as well as their many implementations, is crucial in order to get the models to perform in the best possible way.

In this project, several groups of data collected in a supervised way have been analyzed and a set of Machine Learning algorithms has been applied. The results allow us to make decisions about the new sources of relevant information to be incorporated in consequent studies. It is concluded that the data from sleep and daily activity, measured both by movement and sounds, are relevant for improving the prediction of a crisis in patients with Bipolar Disorder. Therefore, a future project that includes group 1 bracelets instead of the current medical wristbands is proposed, because the latter are too expensive and invasive, and the development of a new mobile application that, in addition to the daily data, includes sensitization data and sounds. Future work will also analyze the EEG data collected during supervised monitoring for the purpose of performing a comparative analysis. The implementation with Jupyter will also allow us to perform the same studies on larger databases when the number of patients in the experiment is higher.

The most immediate use of the results obtained in this project would be to train the same algorithms used but with

larger amounts of data, in order to see if they perform in a similar way. Gathering objective data from devices like phones or wristbands is something that can be accomplished quite easily according to the work already done in this sense. The goal of this task would be to compare the performance of different algorithms on the objective data gathered from these devices with the performance results obtained on the subjective data used in this project.

As for other more indirect applications of the results obtained during this project, the implementation of a drug recommending system for patients with Bipolar Disorder could be made by predicting the states in which the patients are during a certain period of time. These predictions could be stored in a database which also contains the medicine that these patients have been prescribed with during the same period of time, thus providing the possibility of seeing how each patient reacts to the different types of drugs used.

References

[1] N. Vanello et al., "Speech analysis for mood state characterization in bipolar patients," Annual International Conference of the IEEE Engineering in Medicine and Biology Society, San Diego, CA, pp. 2104-2107, 2012

[2] T. Beiwinkel, et al. "Using Smartphones to Monitor Bipolar Disorder Symptoms: A Pilot Study". Eysenbach G, ed. JMIR Mental Health. 2016.

[3] MN. Burns, et al., "Harnessing Context Sensing to Develop a Mobile Intervention for Depression". J Med Internet Res 2011;13(3):e55.

[4] M. Cukic, et al. "EEG machine learning with Higuchi fractal dimension and Sample Entropy as features for successful detection of depression" CoRR abs/1803.05985 (2018): n. pag..

[5] M. Frost, A. Doryab, M. Faurholt-Jepsen, L. Vedel Kessing, and J. E. Bardram. "Supporting disease insight through data analysis: refinements of the monarca self-assessment system". In Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (UbiComp '13). ACM, New York, NY, USA, pp. 133-142, September-2013

[6] H. Javelot, et al., "Telemonitoring with respect to Mood Disorders and Information and Communication Technologies: Overview and Presentation of the PSYCHE Project", BioMed Research International, vol. 2014, Article ID 104658, 12 pages, 2014.

[7] O. Mayora, et. al., "Mobile Health Systems for Bipolar Disorder: The relevance of Non-Functional Requirements in MONARCA Project". IGI International Journal of Handheld Computing Research (IJHCR). 10.4018/978-1-4666-8756-1.ch070.

[8] Z. Guan, T. Yang, X. Du and M. Guizani, "Secure data access for wireless body sensor networks," 2016 IEEE Wireless Communications and Networking Conference, Doha, pp. 1-6. 2016.

[9] P. Gope and T. Hwang, "BSN-Care: A Secure IoT-Based Modern Healthcare System Using Body Sensor Network," in IEEE Sensors Journal, vol. 16, no. 5, pp. 1368-1376, March, 2016.

[10] L. Sylvia, et. al., "Sleep disturbance may impact treatment outcome in bipolar disorder: A preliminary investigation in the context of a large comparative effectiveness trial", Journal of Affective Disorders, Volume 225, 2018, Pages 563-568, ISSN 0165-0327.

[11] AG. Harvey, et. al., "Treating Insomnia Improves Mood State, Sleep, and Functioning in Bipolar Disorder: A Pilot Randomized Controlled Trial". Journal of consulting and clinical psychology. 83. 10.1037/a0038655.

[12] J. Anchiraico, "Diseño de una Arquitectura Big Data para la Predicción de Crisis en el Trastorno Bipolar", Trabajo Fin de Master en Ingeniería Informática, ePrints-UCM, Madrid, 2016.

[13] ActivePAL (2018). Retrived from http://www.palt.com/.

[14] Bellabeat – Urban Collection, Health Trackers (2018). Retrieved from https://webshop.bellabeat.com/pages/leaf-urban

[15] ELF Emmit Device from NewMed (2018). Retrieved from https://www.news-medical.net/ELF-Emmit-Device-from-NewMed

[16] H. Nakano, et al. "Monitoring Sound To Quantify Snoring and Sleep Apnea Severity Using a Smartphone: Proof of Concept." Journal of Clinical Sleep Medicine : JCSM : Official Publication of the American Academy of Sleep Medicine 10.1 (2014): pp. 73–78, *PMC*. Web. 12 June 2018.

[17] The Altman Self-Rating Mania Scale (2018). Retrieved from https://psychology-tools.com/altman-self-rating-mania-scale/

[18] The Patient Health Questionnaire (PHQ-9) (2018). Retrieved from https://patient.info/doctor/patient-health-questionnaire-phq-9

[19] A. Martínez, "Introduction to Big Data and First Steps in a Big Data Project", Trabajo Fin de Doble Grado Matemáticas-Informática, ePrints-UCM, Madrid, 2016.

[20] R.C. Young et al., "A rating scale for mania: reliability, validity and sensitivity", British Journal of Psychiatry, 1978

[21] M. Hamilton, "A rating scale for depression", Journal of Neurology, Neurosurgery, and Psychiatry, 1960.

[22] F. Perez and B. E. Granger, "IPython: A System for Interactive Scientific Computing," in Computing in Science & Engineering, vol. 9, no. 3, pp. 21-29, May-June 2007.

[23] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research 12 (2011), pp. 2825-283. November, 2011.

[24] A. Junestrand, "Application of Machine Learning Algorithms for Bipolar Disorder Crisis Prediction", Trabajo Fin de Grado Ingeniería Informática, ePrints-UCM, Madrid, 2018.

[25] W. McKinney, "Data Structures for Statistical Computing in Python", Proceedings of the 9th Python in Science Conference, 2010.

[26] T. Hastie, R. Tibshirani and J. Friedman, "The Elements of Statistical Learning", Springer Series in Statistics, 2009.

[27] Garima, H. Gulati and P. K. Singh, "Clustering techniques in data mining: A comparison", New Delhi: 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015.

[28] J. R. Quinlan, "Induction of Decision Trees", Hingham, Massachusetts: Kluwer Academic Publishers, 1986.

[29] L. Breiman, "Random Forests", Berkeley, Califiornia: Statistics Department Technical Report, 2001.

[30] C. W. Hsu, C. C. Chang and C. J. Lin, "A Practical Guide to Support Vector Classification", Taipei: National Taiwan University, 2003.

[31] C. Y. J. Peng, K. L. Lee and G. M. Ingersoll, "An Introduction to Logistic Regression Analysis and Reporting", Bloomington, Indiana: The Journal of Educational Research, 2002.

[32] T. Patil and S. Sherekar, "Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification", International Journal Of Computer Science And Applications , 2013.

[33] P. Bühlmann and T. Hothorn, "Boosting Algorithms: Regularization, Prediction and Model Fitting", Zürich: Seminar für Statistik, ETH Zürich, 2007.

[34] M. Stone, "Cross-validatory choice and assessment of statistical predictions", London: Royal Statistical Society, 1974.