Knowl Inf Syst
DOI 10.1007/s10115-011-0465-6

REGULAR PAPER

SMOTE-RSB_*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory

Enislay Ramentol · Yailé Caballero · Rafael Bello · Francisco Herrera

Received: 23 December 2009 / Revised: 8 September 2011 / Accepted: 17 November 2011

Abstract Imbalanced data is a common problem in classification. This phenomenon is growing in importance since it appears in most real domains. It has special relevance to highly imbalanced data-sets (when the ratio between classes is high). Many techniques have been developed to tackle the problem of imbalanced training sets in supervised learning. Such techniques have been divided into two large groups: those at the algorithm level and those at the data level. Data level groups that have been emphasized are those that try to balance the training sets by reducing the larger class through the elimination of samples or increasing the smaller one by constructing new samples, known as undersampling and oversampling, respectively. This paper proposes a new hybrid method for preprocessing imbalanced data-sets through the construction of new samples, using the Synthetic Minority Oversampling Technique together with the application of an editing technique based on the Rough Set Theory and the lower approximation of a subset. The proposed method has been validated by an experimental study showing good results using C4.5 as the learning algorithm.

Keywords Imbalanced data-sets · Classification · Data preparation · Oversampling · Undersampling · Rough sets theory

E. Ramentol · Y. Caballero
Department of Computer Science, University of Camagüey, Camagüey, Cuba
e-mail: enislayr@yahoo.es

Y. Caballero
e-mail: yailec@yahoo.com

R. Bello
Department of Computer Science, Universidad Central de Las Villas, Santa Clara, Cuba
e-mail: rbellop@uclv.edu.cu

F. Herrera (✉)
Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain
e-mail: herrera@decsai.ugr.es

Published online: 04 December 2011

1 Introduction

The imbalanced data-set problem in classification domains occurs when the number of instances that represent one class is much larger than the other classes. The minority class is usually more interesting from the point of view of the learning task [9]. There are many situations in which imbalance occurs between classes, such as satellite image classification [38], risk management [26], medical applications [29], and so on. As a result, this problem has been identified as a current challenge in Data Mining [48].

Classification algorithms often achieve high accuracy with the majority class whereas with the minority class, quite the opposite occurs. In imbalanced training sets, original knowledge often focuses on the minority class, whereas many classifiers consider the less frequent data to be rarities or noise, focusing exclusively on the results of the global measures [23,31,37].

Many techniques for dealing with class imbalance have arisen as a result of research and are grouped into two categories [9]: those at the level of the learning algorithm and those that modify data distribution (data level).

The great advantage of the data level approaches is that they are more versatile, since their use is independent of the classifier selected. Furthermore, we may preprocess all data-sets in order to use them to train different classifiers. In this manner, the computation time needed to prepare the data is only required once. In this paper, we consider the use of a well-known and widely used technique to balance the training set before the learning phase, the "Synthetic Minority Oversampling Technique" (SMOTE) methodology [8].

In this paper, we tackle the data level making the two following assumptions:

- It has special relevance to highly imbalanced data-sets, when the imbalance ratio (defined as the fraction between the number of instances of the majority class and the minority class) is high (higher than or equal to 9, so the minority class represents less than 10%).
- SMOTE can introduce examples of the minority class in the area of the majority class. This phenomenon is known as overgeneralization. In order to solve this problem, some cleaning (multi-edit) methods have been used in order to eliminate such examples, improving the SMOTE results, such as ENN and TomekLinks [4], Borderline [22], or Safe level [7].

This work introduces a new hybrid proposal for carrying out oversampling via SMOTE and undersampling over the synthetic instances for highly imbalanced data-sets, called SMOTE-RSB_*, and it is based on two steps:

- building new synthetic examples of the minority class using SMOTE and
- improving the quality of these new samples through editing techniques based on the Rough Set Theory (RST) and the lower approximation of a subset, acting over the artificial instances of the minority class created by the SMOTE algorithm.

Our main contribution is to introduce a new preprocessing method using SMOTE to generate synthetic examples and RST as a cleaning method. We propose the elimination of any synthetic example that does not belong to the lower approximation of the minority class, considering these examples in the boundary region as noisy and not useful for classification. We carried out experiments in order to show the goodness of this model in comparison with SMOTE, SMOTE-ENN, SMOTE-TomekLinks, Borderline-SMOTE1, Borderline-SMOTE2 and Safe-Level-SMOTE, using 44 data-sets from the UCI repository [3] with high imbalance ratios. The measure of performance is based on AUC [25], and the significance of the results is supported by the proper statistical analysis as suggested in the literature [12,16].

In order to do this, the paper is organized as follows. In Sect. 2, Related work, we introduce the imbalanced data-set problem, discuss the evaluation metric used in this work, describe

some preprocessing techniques for imbalanced data-sets, and introduce the RST. In Sect. 3 we present the algorithm called SMOTE-RSB_*. In Sect. 4 we introduce the experimental study, that is, the benchmark data-sets, the statistical tests for performance comparison and the experimental analysis in order to validate the goodness of our proposal. In Sect. 5 we draw some conclusions about the completed study.

2 Related work

This problem is closely related to the cost-sensitive classification problem [28,36,49]. The classical machine learning algorithms may be biased toward the majority class and, as a result, may predict the minority class examples poorly. This problem is growing in importance and has been identified as one of the 10 main challenges of Data Mining [48].

We focus our study on imbalanced data-sets with binary classes, that is, there is only one positive and one negative class, considering the former to be the one with the lower number of examples and the latter the one with the higher number of examples. This scenario can be viewed as a set of examples and counterexamples for a given concept to be learnt, which is the most common case in the specialized literature for imbalanced classification. When multiple classes are present, the binary-class approach may be directly applicable via pairwise coupling techniques [15]. Specifically, in a previous work by [14], the authors carried out an experimental analysis in which it is shown that this methodology allows the achievement of a good behavior for preprocessing in multiclass imbalanced data-sets.

As we have mentioned, the imbalanced data-set problem can be tackled using two main types of solutions:

1. **Solutions at the data level** [4,8,10,17]: this kind of solution consists of balancing the class distribution by oversampling the minority class (positive instances) or undersampling the majority class (negative instances) or by applying hybrid models which combine the previous techniques.
2. **Solutions at the algorithmic level**: in this case we need to adapt our method to deal directly with the imbalance between the classes, for example, modifying the cost per class [21] or adjusting the probability estimation in the leaves of a decision tree to favor the positive class [44].

In published research works, it has been shown that applying a preprocessing step in order to balance the class distribution is a positive solution to the problem of imbalanced data-sets [4,13]. Furthermore, the main advantage of these techniques is that they are independent of the classifier used. In [41] a system is presented that combines these two general solutions (data and algorithm level) obtaining good results.

In this work, we evaluate different instance selection methods together with oversampling and hybrid techniques to adjust the class distribution in the training data. Specifically we have chosen the methods that have been studied in [4]. These methods are classified into three groups:

- **Undersampling methods** that create a subset of the original data-set by eliminating some of the examples of the majority class.
- **Oversampling methods** that create a superset of the original data-set by replicating some of the examples of the minority class or creating new ones from the original minority class instances.
- **Hybrid methods** that combine the two previous methods, eliminating some of the minority class examples expanded by the oversampling method in order to eliminate overfitting.

In the remainder of this section, we will describe the methods used in this paper for the experimental study.

– **Undersampling methods**

- **"Tomek links"** [39] can be defined as follows: given two examples e_i and e_j belonging to different classes, with $d(e_i, e_j)$ the distance between e_i and e_j. A (e_i,e_j) pair is called a Tomek link if there is no example E_l, so that $d(e_i, e_l) < d(e_i, e_j)$ or $d(e_j, e_l) < d(e_i, e_j)$. If two examples form a Tomek link, then either one of these examples is noise or both examples are borderline. Tomek links can be used as an undersampling method or as a data cleaning method. As an undersampling method, only examples belonging to the majority class are eliminated, and as a data cleaning method, examples of both classes are removed.
- **"Neighborhood Cleaning Rule"** (NCL) uses the Wilson's Edited Nearest Neighbor Rule (ENN) [45] to remove majority class examples. ENN removes any example whose class label differs from the class of at least two of its three nearest neighbors. NCL modifies the ENN in order to increase the data cleaning. For a two-class problem, the algorithm can be described in the following way: for each example e_i in the training set, its three nearest neighbors are found. If e_i belongs to the majority class and the classification given by its three nearest neighbors contradict the original class of e_i, then e_i is removed. If e_i belongs to the minority class and its three nearest neighbors misclassify e_i, then the nearest neighbors that belong to the majority class are removed.

– **Oversampling methods**

- **"Synthetic Minority Oversampling Technique"** (SMOTE) [8] is an oversampling method. Its main idea is to form new minority class examples by interpolating between several minority class examples that lie together. Thus, the overfitting problem is avoided and the decision boundaries for the minority class spread further into the majority class space. This method is described in detail in the next part of this section.

– **Hybrid methods: Oversampling plus Undersampling**

- **"SMOTE—Tomek links"**. Frequently, class clusters are not well defined as some majority class examples might invade the minority class space. The opposite can also be true, since interpolating minority class examples can expand the minority class clusters, introducing artificial minority class examples too deeply into the majority class space. Inducing a classifier in such a circumstances can lead to overfitting. In order to create better-defined class clusters, Batista et al. [4] proposed applying Tomek links to the oversampled training set as a data cleaning method. Thus, instead of removing only the majority class examples that form Tomek links, examples from both classes are removed.
- **"SMOTE—ENN"**. The motivation behind this method is similar to SMOTE—Tomek links [4]. ENN tends to remove more examples than the Tomek links do, so it is expected to provide a more in depth data cleaning. In contrast to NCL, which is an undersampling method, ENN is used to remove examples from both classes. Thus, any example that is misclassified by its three nearest neighbors is removed from the training set.
- **"Borderline-SMOTE1".** This method only oversamples or strengthens the borderline minority examples [22]. First, it finds out the borderline minority examples P;

then, synthetic examples are generated from them and added to the original training set. This method can be described as follows: for every minority example (p_i) calculate its m nearest neighbors from the whole training set, if all the m nearest neighbors are majority examples, p_i is considered to be noise and is not operated in the following step. If $m/2 \leq m' < m$, namely the number of p_i's majority nearest neighbors is larger than the number of its minority ones, p_i is considered to be easily misclassified and put into a set DANGER. If $0 \leq m' < m/2$, p_i is safe and does not need to participate in the follows steps. The examples in DANGER are the borderline data of the minority class P. Finally, for each example in DANGER, we calculate its k-nearest neighbors from P and operate in a similar way to SMOTE.

- **"Borderline-SMOTE2".** This method is very similar to Borderline-SMOTE1; it not only generates synthetic examples from each example in DANGER and its positive nearest neighbors in P, but also does so for its nearest negative neighbor in N (majority class) [22]. The difference between it and its nearest negative neighbor is multiplied by a random number between 0 and 0.5; thus, the newly generated examples are closer to the minority class.
- **"Safe-Level-SMOTE".** This method assigns each positive instance its safe level before generating synthetic instances [7]. Each synthetic instance is positioned closer to the largest safe level so all synthetic instances are generated only in safe regions.

2.1 Evaluation in imbalanced domains

The performance of machine learning algorithms is typically evaluated using predictive accuracy. However, this is not appropriate when the data are imbalanced and/or when the costs of different errors vary markedly [9].

Weiss and Hirsh [43] showed that the error rate of the classification of the rules of the minority class is 2 or 3 times greater than the rules that identify the examples of the majority class and that the examples of the minority class are less likely to be predicted than the examples of the majority one. Because of this, instead of using the error rate (or accuracy), in the context of imbalanced problems more appropriate metrics are considered.

A confusion matrix is a form of contingency table showing the differences between the true and predicted classes for a set of labeled examples that introduces the well-known measures: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Using these, we obtain the classical rates of accuracy and error as follows:

$$\text{Error rate} = \frac{\text{FP} + \text{FN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{1}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} = 1 - \text{Error rate} \tag{2}$$

It is also possible to derive four performance metrics that directly measure the classification performance of positive and negative classes independently:

- **True positive rate** $\text{TP}_{\text{rate}} = \text{TP}/(\text{TP} + \text{FN})$ is the percentage of positive cases correctly classified as belonging to the positive class.
- **True negative rate** $\text{TN}_{\text{rate}} = \text{TN}/(\text{FP} + \text{TN})$ is the percentage of negative cases correctly classified as belonging to the negative class.
- **False positive rate** $\text{FP}_{\text{rate}} = \text{FP}/(\text{FP} + \text{TN})$ is the percentage of negative cases misclassified as belonging to the positive class.
- **False negative rate** $\text{FN}_{\text{rate}} = \text{FN}/(\text{TP} + \text{FN})$ is the percentage of positive cases misclassified as belonging to the negative class.

These four performance measures have the advantage of being independent of class costs and prior probabilities. The aim of a classifier is to minimize the false positive and negative rates or, similarly, to maximize the true negative and positive rates.

One appropriate metric that could be used to measure the performance of classification over imbalanced data-sets is the Receiver Operating Characteristic (ROC) graphics [6]. In these graphics, the tradeoff between the benefits (TP_{rate}) and costs (FP_{rate}) can be visualized, and it acknowledges the fact that the capacity of any classifier cannot increase the number of true positives without also increasing the false positives. The area under the ROC curve (AUC) [25] corresponds to the probability of correctly identifying which of the two stimuli is noise and which is signal plus noise. AUC provides a single-number summary for the performance of learning algorithms.

The way to build the ROC space is to plot on a two-dimensional chart the true positive rate (Y-axis) against the false positive rate (X-axis) as shown in Fig. 1. The points (0, 0) and (1,1) are trivial classifiers in which the output class is always predicted as negative and positive, respectively, while the point (0, 1) represents perfect classification. To compute the AUC, we just need to obtain the area of the graphic as:

$$\text{AUC} = \frac{1 + \text{TP}_{\text{rate}} - \text{FP}_{\text{rate}}}{2} \tag{3}$$

2.2 SMOTE: synthetic minority oversampling technique

The SMOTE algorithm [8] oversamples the minority class by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors.

With this approach, the positive class is oversampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of oversampling required, neighbors from the k-nearest neighbors are randomly chosen. This process is illustrated in Fig. 2, where x_i is the selected point, x_{i1} to x_{i4} are some selected nearest neighbors and r_1 to r_4 the synthetic data points created by the randomized interpolation.

Synthetic samples are generated in the following way: take the difference between the feature vector (sample) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific features.

This algorithm is detailed in Fig. 3, which we describe below:

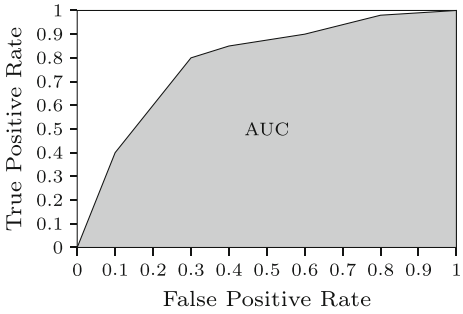

Fig. 1 The area under the ROC curve

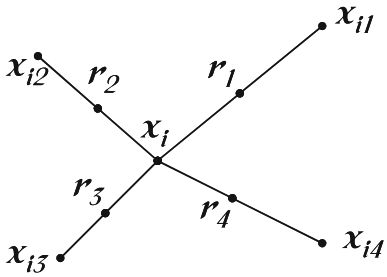

Fig. 2 An illustration of how to create the synthetic data points in the SMOTE algorithm

Input: Number of minority class samples T; *nattrs* Number of attributes; *Sample*[][] array for he original minority class samples, Amount of SMOTE N %; Number of nearest neighbors k.
Output: *Synthetic*[][] and array of (N/100) * T synthetic minority class samples
begin
1. if $N < 100$ **then**
1.1 Randomize the T minority class samples
1.2 Fill *Sample*[][] with the first $(N/100) * T$ samples
1.3 $T = (N/100) * T$
1.4 $N = 100$
end if
2. for $i \rightarrow 1$ to T
2.1 *Compute k nearest neighbors for i*, and save the indices in an array *nnarray*
2.2 *Populate*$(N, i, nnarray)$
end for
3. while $N \neq 100$ **do**
3.1 Choose a random number between 1 and k, call it nn
3.2 **for** $attr \leftarrow 1$ **to** *nattrs*
3.2.1 Compute: $dif = Sample[nnarray[nn]][attr] - Sample[i][attr]$
3.2.2 Compute: $gap = random\ number\ between\ 0\ and\ 1$
3.2.3 $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$
end for
3.3 $newindex++$
3.4 $N = N - 1$
end while
end

Fig. 3 SMOTE Algorithm pseudo-code

1. "Step 1" selects the number of minority class samples to be used for generating new instances when the degree of oversampling is lower than the 100%.
2. "Step 2" is aimed to computed all the k-neighbors for each sample to be replicated.
3. Finally, "Step 3" computes the interpolation as explained above. For the sake of clarity, an example of this usage is shown in Fig. 4.

In contrast to the common replication techniques (for example random oversampling), in which the decision region usually become more specific, with SMOTE the overfitting problem is somehow avoided by causing the decision boundaries for the minority class to be larger and to spread further into the majority class space, since it provides related minority class samples to learn from.

Finally, we must point out that the implementation of SMOTE used in this work has been taken from the open source KEEL project [1] [1,2].

[1] http://www.keel.es.

```
Consider a sample (6,4) and let (4,3) be its nearest neighbor.
(6,4) is the sample for which k-nearest neighbors are being
identified (4,3) is one of its k-nearest neighbors.
Let: f1_1 = 6 f2_1 = 4,  f2_1 - f1_1 = -2
f1_2 = 4  f2_2 = 3,  f2_2 - f1_2 = -1
The new samples will be generated as
(f1',f2') = (6,4) + rand(0-1) * (-2,-1)
rand(0-1) generates a random number between 0 and 1.
```

Fig. 4 Example of the SMOTE application

2.3 Rough set theory

Rough sets were presented in a paper published in 1982 [32]. Today, the RST has evolved into a methodology for dealing with different types of problems, such as uncertainty produced by inconsistencies in data [5].

RST is a good tool to model uncertainty when it appears as inconsistency. It is possible to manage quantitative as well as qualitative data, and it is unnecessary to eliminate inconsistencies prior to the analysis. The RST can be used with output information to determine the relevance of the attributes, generate relationships between them (in terms of rules), and so on. RST has allowed the development of methods for solving problems based on Information Systems such as classification, decision making, knowledge discovery [5,11,30,40,42,47].

The novelty of the RST is in the lower and upper approximations of a subset $X \subseteq U$. These concepts were originally introduced in reference to an indiscernibility relation R.

This classical approach of the RST is extended by accepting that objects that are not indiscernible but sufficiently close or similar can be grouped into the same class [20,35]. The aim is to construct a similarity relation $R^{'}$ from the indiscernibility relation R by relaxing the original conditions for indiscernibility. This relaxation can be performed in many ways, thus giving many possible definitions of similarity. Several functions of comparison of attributes exist (similarity functions), which are associated with the type of attribute that is compared. However, this similarity relation $R^{'}$ must satisfy some minimal requirements:

– R being an indiscernibility relation (equivalence relation) defined on U, $R^{'}$ is a similarity relation extending R if $\forall x \in U,\ R(x) \subseteq R^{'}(x)$and $\forall x \in U,\ \forall y \in R^{'}(x),\ R(y) \subseteq R^{'}(x)$, where $R^{'}(x)$is a similarity class of x, ie. $R^{'}(x) = \{y \in U : yR^{'}x\}$.

The approximation of the set $X \subset U$, using the inseparability relation R, has been induced as a pair of sets called $R-$ lower approximation of X and $R-$upper approximation of X. The lower approximation $B_*(X)$ and upper approximation $B^*(X)$ of X are defined respectively as shown in Eqs. 4 and 5.

$$B_*(X) = \{x \in X : R^{'}(x) \subseteq X\} \tag{4}$$

$$B^*(X) = \bigcup_{x \in X} R^{'}(x) \tag{5}$$

Taking into account the equations defined in 4 and 5, the boundary region of X is defined for the relation $R^{'}$ as:

$$BN_B(X) = B^*(X) - B_*(X) \tag{6}$$

If the set BN_B is empty, then the X set is exact with respect to the relation $R^{'}$. If, by contrast, $BN_B(X) \neq \theta$, the X set is inexact or approximated with respect to $R^{'}$.

The use of similarity relations offers greater possibilities for the construction of approximations.

Within RST, the meaning of the lower approximation of a decision system is of great interest for the revision of training sets. It allows us to analyze the consistencies associated with the different classes in a classification problem. The lower approximation consists of the objects that with absolute certainty belong to one class or another, guaranteeing that these instances are free of noise.

This idea is used for designing the hybrid preprocessing method that we present in the following section. We will use RST for clearing the synthetic instances created by SMOTE, avoiding noise and inconsistencies.

3 SMOTE-RSB_*: a preprocessing method combining SMOTE and RST

In this section, we present a new proposal for making the distribution between classes in imbalanced training sets uniform. The hybrid method has two stages:

1. first, we create new instances using the SMOTE algorithm.
2. second, a cleaning method based on RST is applied to include the original examples and the synthetic minority examples that belong to the lower approximation (B_*) of their class in the final training set (called $resultset$ in the algorithm).

The algorithm uses the extended approach of RST based on similarity relations and includes five steps that we detail below:

1. **"Step 1"** uses SMOTE for oversampling the original data-set and it matches with the first stage.
2. **"Step 2"** just builds the final set of instances (output of the algorithm) by including initially the original ones of the data-set.
3. **"Step 3"** constructs a similarity matrix for all instances of the original data-set. The function used to determine the degree of similarity between two instances x_i and x_j is defined as follows (assigning its value to the $similarityMatrix$):

$$similarityMatrix(i, j) = \frac{\sum_{k=1}^{n} w_k * \delta_k(x_{ik}, x_{jk})}{M} \tag{7}$$

where n is the number of features, w_k the weight for feature k, x_{ik} and x_{jk} are the values for feature k, respectively, δ_k is the function of comparison for feature k, M is the number of features considered in the equivalence relation, B is the features set considered in the equivalence relation.

The weight of a feature is defined as:

$$w_k = \begin{cases} 1 & \text{if } k \in B \\ 0 & \text{other case} \end{cases} \tag{8}$$

δ_k is calculated for discrete attributes in the following way:

$$\delta_k(x_{ik}, x_{jk}) = \begin{cases} 1 & \text{if } x_{ik} = x_{jk} \\ 0 & \text{other case} \end{cases} \tag{9}$$

and for continuous attributes:

$$\delta_k(x_{ik}, x_{jk}) = 1 - \frac{\left|x_{ik} - x_{jk}\right|}{\max A_k - \min A_k} \tag{10}$$

where $maxA_k$ and $minA_k$ are the extremes of the domain intervals for feature k.

4. **"Step 4"** analyzes which new synthetic data belong to the lower approximation, that is, their similarity value is lower than a given threshold, which means that there are no similar elements in the set, and they are added to the final "result set".
5. Finally, **"Step 5"**, checks whether all new generated instances are similar among them and returns the final "result set" as the output of the SMOTE algorithm.

Figure 5 shows the algorithm associated with the steps that are previously indicated. As we have point out previously, $resultSet$ is the output set of the algorithm, containing the original instances, and the final synthetic instances of the training data-set and $syntheticInstance$ is a vector containing the new instances generated by the SMOTE algorithm. This method is included within the KEEL software tool [1,2], so that any interested researcher can reproduce the experimental study.

We must pay special attention to the $similarityValue$ parameter, since it is the value used to determine the similarity of the instances for obtaining the lower and upper approximation. It will be analyzed in the experimental study in a section devoted to this parameter study.

With this algorithm, it is possible to obtain a training data-set eliminating inconsistencies from the synthetic examples. In the final training set, it inserts those synthetic instances, belonging to the lower approximation (B_*), that do not have similarities in the majority class. There are some exceptions for problems where all synthetic examples have similarities with negative instances. This justifies the inclusion of $Step\ 5$, due to the fact that SMOTE usually provides better behavior in the learning algorithm than the use of the original data-set without synthetic instances.

Algorithm steps
1. Step1: Using SMOTE we create a set synthetic data ($syntheticInstance[]$) for the minority class until the training set is balanced.
2. Step2: Create $resultSet$: including the original instances.
3. Step3: Construct the $similarityMatrix$ for synthetic instances organized in rows and negative instances (majority class) in columns, using expression 7 and considering all the features in the equivalence relation. In the $similarityMatrix(i, j)$ we will find the similarity degree between instances i and j.
4. Step4: For every synthetic example count the number of similar examples in the negative class.
$npos - syn$: number of synthetic instances
$similarityValue := 0.4$
While ($resultSet$ is empty) & ($similarityValue \leq 0.9$) **do**
  for $i \rightarrow 1$ **to** $npos - syn$
    for $j \rightarrow 1$ **to** $nneg$
      if ($similarityMatrix(i, j) > similarityValue$)
      then $cont[i] + +$
      endfor
    if $cont[i] = 0$ **then** // the instances are in the lower approximation,
     insert $syntheticInstance[i]$ in $resultset$ //(final training set)
  endfor
$similarityValue := similarityValue + 0.05$
endwhile
5. Step5: If there are no instances in the lower approximation, i.e all synthetic instances are similar to other positive ones, the solution is given as the set balanced with SMOTE, all synthetic instances are included in "resultset".

Fig. 5 Algorithm SMOTE-RSB_*

4 Experimental study

In this section, we first present the experimental framework, including the benchmark data-sets, the parameters, and the statistical tests used in order to carry out the performance comparison. Then, we introduce the experimental analysis, which is divided into two parts: first we carry out an analysis of the parameters for our model, then we develop the comparative analysis with some preprocessing techniques.

4.1 Experimental setup: data-sets, parameters, and statistical tests

In this section, we briefly describe the data-sets used for the experimental study and the statistical tests used alongside the experimental study. The learning algorithm used for the experimental study is $C4.5$ [33], which has been identified as one of the 10 top algorithms in Data Mining [46] and has been widely used in imbalanced problems [4].

4.1.1 Data-sets and parameters

To analyze our proposal, we have considered 44 data-sets from the UCI repository [3] with highly imbalanced rates (higher than 9). Multiclass data-sets were modified to obtain two-class non-balanced problems, so that the union of one or more classes of the minority class and the union of one or more of the remaining classes was labeled as the majority class. The description of these data-sets appears in Table 1 (column IR indicates the imbalance ratio).

The sets were divided in order to perform a fivefolds cross-validation, 80% for training and 20% for testing, where the 5 test data-sets form the whole set. For each data-set, we consider the average results of the five partitions. Partition was carried out in such a way that the quantity of elements in each class remained uniform [13]. They are available at the Website KEEL-dataset [1,2] at the link: http://www.keel.es/datasets.php.

For our experiments, we consider the following parameters for the SMOTE-RSB_* algorithm:

- k: Number of nearest neighbors that is fixed to 5.
- Distance function to obtain the nearest neighbors, the Euclidean distance is used.
- The class distribution will be rebalanced to 50–50%.

We must point out that these parameter values are those recommended by the authors of the SMOTE algorithm [8], and therefore we have used them as a standard for our experimentation.

4.1.2 Statistical tests

In this paper, we use the hypothesis testing techniques to provide statistical support to the analysis of the results [18,34]. Specifically, we will use non-parametric tests, due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility with these types of tests [12].

For multiple comparisons, we use the Iman-Davenport test [27] to detect statistical differences among a group of results and the Holm post hoc test [24] in order to find which algorithms reject the hypothesis of equality with respect to a selected control method.

The post hoc procedure allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance α. However, it is very interesting to compute the p-value associated with each comparison, which represents the lowest level of

Table 1 Description of the data-sets used in the experiments

Data-set	# Ex	#Attributes	%Class(min., maj.)	IR
ecoli0137vs26	281	7	(2.49, 97.51)	39.15
shuttle0vs4	1829	9	(6.72, 93.28)	13.87
yeastB1vs7	459	7	(6.53, 93.47)	14.3
shuttle2vs4	129	9	(4.65, 95.35)	20.5
glass016vs2	192	9	(8.85, 91.15)	10.29
glass016vs5	184	9	(4.89, 95.11)	19.44
pageblocks13vs4	472	10	(5.93, 94.07)	15.85
yeast05679vs4	528	8	(9.66, 90.34)	9.35
yeast1289vs7	947	8	(3.16, 96.84)	30.5
yeast1458vs7	693	8	(4.33, 95.67)	22.10
yeast2vs4	514	8	(9.92, 90.08)	9.08
Ecoli4	336	7	(6.74, 93.26)	13.84
Yeast4	1484	8	(3.43, 96.57)	28.41
Vowel0	988	13	(9.01, 90.99)	10.10
Yeast2vs8	482	8	(4.15, 95.85)	23.10
Glass4	214	9	(6.07, 93.93)	15.47
Glass5	214	9	(4.20, 95.80)	22.81
Glass2	214	9	(7.94, 92.06)	11.59
Yeast5	1484	8	(2.96, 97.04)	32.78
Yeast6	1484	8	(2.49, 97.51)	39.16
abalone19	4174	8	(0.77, 99.23)	128.87
abalone918	731	8	(5.65, 94.25)	16.68
cleveland0vs4	177	13	(7.34, 92.66)	12.61
ecoli01vs235	244	7	(2.86, 97.14)	9.16
ecoli01vs5	240	7	(2.91, 97.09)	11
ecoli0146vs5	280	7	(2.5, 97.5)	13
ecoli0147vs2356	336	7	(2.08, 97.92)	10.58
ecoli0147vs56	332	7	(2.1, 97.9)	12.28
ecoli0234vs5	202	7	(3.46, 96.54)	9.1
ecoli0267vs35	224	7	(3.12, 96.88)	9.18
ecoli034vs5	300	7	(2.33, 97.67)	9
ecoli0346vs5	205	7	(3.41, 96.59)	9.25
ecoli0347vs56	257	7	(2.72, 97.28)	9.28
ecoli046vs5	203	7	(3.44, 96.56)	9.15
ecoli067vs35	222	7	(3.15, 96.85)	9.09
ecoli067vs5	220	7	(3.18, 96.82)	10
glass0146vs2	205	9	(4.39, 95.61)	11.05
glass015vs2	172	9	(5.23, 94.77)	9.11
glass04vs5	92	9	(9.78, 90.22)	9.22
glass06vs5	108	9	(8.33, 91.67)	11

Table 1 continued

Data-set	# Ex	#Attributes	%Class(min., maj.)	IR
led7digit02456789vs1	443	7	(1.58, 98.42)	10.97
yeast0359vs78	506	8	(9.8, 90.2)	9.12
yeast0256vs3789	1004	8	(9.86, 90.14)	9.14
yeast02579vs368	1004	8	(9.86, 90.13)	9.14

significance of a hypothesis that results in a rejection. In this manner, we can know whether two algorithms are significantly different and how different they are.

Furthermore, we consider the average ranking of the algorithms in order to show graphically how good a method is with respect to its partners. This ranking is obtained by assigning a position to each algorithm depending on its performance for each data-set. The algorithm that achieves the best accuracy on a specific data-set will have the first ranking (value 1); then, the algorithm with the second best accuracy is assigned rank 2, and so forth. This task is carried out for all data-sets and finally an average ranking is computed as the mean value of all rankings.

These tests are suggested in the studies presented in [12,16,18,19], where its use in the field of machine learning is highly recommended. Any interested reader can find additional information on the Website http://sci2s.ugr.es/sicidm/, together with the software for applying the statistical tests.

4.2 SMOTE-RSB_*: parameter analysis

As previously introduced in Sect. 3, the $similarityvalue$ was fixed during the algorithm runtime. The algorithm started with 0.4 value; if the cleaning method does not find any instance in the lower approximation, the value is increased by 0.05 while the value remains lower than or equal to 0.9. This is a way to ensure the lowest similarity value but high enough to populate the lower approximation, obtaining good quality objects in the lower approximation.

Figure 6 shows the maximum similarity value for which each data-set gets balanced; each number in the x-axis represents a different data-set, whereas the y-axis shows the similarity values. As it can be observed, most of the data-sets are completely balanced when the similarity value is around 0.8, specifically those with a high imbalance ratio.

4.3 Comparative analysis

In this section, we will compare SMOTE-RSB_* with another six well-known preprocessing mechanisms based on SMOTE, that is, the SMOTE algorithm itself and four hybrid

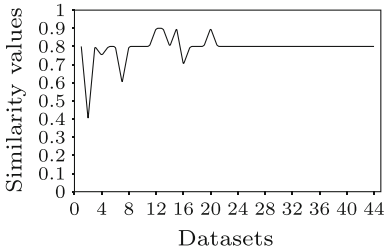

Fig. 6 Similarity value per data-set

Table 2 Comparison of the AUC results for 6 preprocessing algorithms for test

Data-set	Original	Smote	S-TL	S-ENN	Border1	Border2	Safelevel	S-RSB$_*$
ecoli0137vs26	0.7481	0.8136	0.8136	0.8209	**0.8445**	**0.8445**	0.8118	**0.8445**
shuttle0vs4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9988	**1.0000**
yeastB1vs7	0.6275	0.7003	0.7371	0.7277	0.6422	0.6407	0.6621	**0.8617**
shuttle2vs4	**1.0000**	0.9917	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	1.0000
glass016vs2	0.5938	0.6062	0.6388	**0.6390**	0.5738	0.5212	0.6338	0.6376
glass016vs5	**0.8943**	0.8129	0.8629	0.8743	0.8386	0.8300	0.8429	0.8800
pageblocks13vs4	**0.9978**	0.9955	0.9910	0.9888	**0.9978**	0.9944	0.9831	0.9978
yeast05679vs4	0.6802	0.7602	0.7802	0.7569	0.7473	0.7331	**0.7825**	0.7719
yeast1289vs7	0.6156	0.6832	0.6332	0.7037	0.6058	0.5473	0.5603	**0.7487**
yeast1458vs7	0.5000	0.5367	0.5563	0.5201	0.4955	0.4910	0.5891	**0.6183**
yeast2vs4	0.8307	0.8588	0.9042	0.9153	0.8635	0.8576	0.8647	**0.9681**
Ecoli4	0.8437	0.8310	0.8544	**0.9044**	0.8358	0.8155	0.8386	0.8544
Yeast4	0.6135	0.7004	0.7307	0.7257	0.7124	0.6882	**0.7945**	0.7609
Vowel0	0.9706	0.9494	0.9444	0.9455	0.9278	**0.9766**	0.9566	0.9678
Yeast2vs8	0.5250	0.8066	0.8045	**0.8197**	0.6827	0.6968	0.8112	0.7370
Glass4	0.7542	0.8508	**0.9150**	0.8650	0.7900	0.8325	0.9020	0.8768
Glass5	0.8976	0.8829	0.8805	0.7756	0.8854	0.8402	0.8939	**0.9232**
Glass2	0.7194	0.5424	0.6269	0.7457	0.7092	0.5701	0.6979	**0.7912**
Yeast5	0.8833	0.9233	0.9427	0.9406	0.9118	0.9219	0.9542	**0.9622**
Yeast6	0.7115	0.8280	**0.8287**	0.8270	0.7928	0.7485	0.8163	0.8208
abalone19	0.5000	0.5202	0.5162	0.5166	0.5202	0.5202	**0.5363**	0.5244
abalone918	0.5983	0.6215	0.6675	0.7193	0.7216	0.6819	**0.8112**	0.6791
cleveland0vs4	0.6878	0.7908	0.8376	0.7605	0.7194	0.7255	**0.8511**	0.7620
ecoli01vs235	0.7136	0.8377	**0.8495**	0.8332	0.7377	0.7514	0.7550	0.7777
ecoli01vs5	0.8159	0.7977	0.8432	0.8250	0.8318	0.8295	**0.8568**	0.7818
ecoli0146vs5	0.7885	**0.8981**	**0.8981**	**0.8981**	0.7558	0.8058	0.8519	0.8231
ecoli0147vs2356	0.8051	0.8277	0.8195	0.8228	0.7465	**0.8320**	0.8149	0.8154
ecoli0147vs56	0.8318	0.8592	0.8424	0.8424	0.8420	0.8453	0.8197	**0.8670**
ecoli0234vs5	0.8307	0.8974	0.8920	0.8947	0.8613	0.8586	0.8700	**0.9058**
ecoli0267vs35	0.7752	0.8155	**0.8604**	0.8179	0.8352	0.8102	0.8380	0.8227
ecoli034vs5	0.8389	0.9000	0.9361	0.8806	0.8806	0.9028	0.8306	**0.9417**
ecoli0346vs5	0.8615	**0.8980**	0.8703	**0.8980**	0.8534	0.8838	0.8520	0.8649
ecoli0347vs56	0.7757	0.8568	0.8482	0.8546	0.8427	0.8449	0.7995	**0.8984**
ecoli046vs5	0.8168	0.8701	0.8674	0.8869	0.8615	0.8892	0.8923	**0.9476**
ecoli067vs35	0.8250	0.8500	0.8125	0.8125	0.8550	**0.8750**	0.7950	0.8525
ecoli067vs5	0.7675	0.8475	0.8425	0.8450	0.8875	**0.8900**	0.7975	0.8800
glass0146vs2	0.6616	0.7842	0.7454	0.7095	0.6565	0.6958	0.7465	**0.7978**
glass015vs2	0.5011	0.6772	0.7040	**0.7957**	0.5196	0.5817	0.7215	0.7065
glass04vs5	0.9941	0.9816	0.9754	0.9754	0.9941	**1.0000**	0.9261	0.9941
glass06vs5	**0.9950**	0.9147	0.9597	0.9647	0.9950	0.9000	0.9137	0.9650
led7digit02456789vs1	0.8788	0.8908	0.8822	0.8379	0.8908	0.8908	**0.9023**	0.9019
yeast0359vs78	0.5868	0.7047	0.7214	0.7024	0.6228	0.6438	0.7296	**0.7400**

Table 2 continued

Data-set	Original	Smote	S-TL	S-ENN	Border1	Border2	Safelevel	S-RSB$_*$
yeast0256vs3789	0.6606	**0.7951**	0.7499	0.7817	0.7528	0.7644	0.7551	0.7857
yeast02579vs368	0.8432	**0.9143**	0.9007	0.9138	0.8810	0.8901	0.9003	0.9105
Mean	0.7673	0.8142	0.8247	0.8247	0.7937	0.7923	0.8173	**0.8402**

Table 3 Winner algorithm

	Original	Smote	S-TL	S-ENN	Border1	Border2	Safelevel	S-RSB$_*$	Total
Test	1/3	3/1	4/1	4/2	0/4	5/2	7/1	15/3	39/5 ties

Absolute winner/ties

approaches SMOTE-TomekLinks, SMOTE-ENN (they have been analyzed in [4], Borderline-SMOTE1, Borderline-SMOTE2, and Safe-Level-SMOTE.

The results of the experimental study for the test partitions are shown in Table 2, where in the first column we have included the result over the original data-sets, and the best method is highlighted in bold for each data-set. We can observe the goodness of the SMOTE-RSB_* approach since it obtains the highest performance value for all the methodologies that are being compared. Additionally, the good results for SMOTE-TomekLinks and SMOTE-ENN with respect to SMOTE emphasize the significance of the cleaning step in the oversampling for achieving a superior behavior at the classification stage. Finally, all the preprocessing approaches outperforms the results with the original data-sets as expected.

Table 3 shows the total number of times every compared algorithm obtains the highest AUC value. There are two numbers per cell. The first number represents how many times the algorithm is the absolute winner while the second one denotes how many times it shares the highest AUC with other algorithms (ties). The last column sums up results in the two numbers. The first one indicates how many times the highest AUC is achieved by a single algorithm while the second one shows the total amount of data-sets where the highest AUC is shared.

Table 4 shows the ranking of the algorithms on each data-set selected for this study and for the original data-sets. The reader can observe that our proposal appears 18 times in first place, 7 times in second position, 9 times in third position and only one time in the last three positions.

In order to compare the results, we will use a multiple comparison test to find the best preprocessing algorithm. In Table 5 we can observe that the best ranking is obtained by our proposal, and the two last positions correspond to Borderline-Smote1 and Borderline-Smote2.

An Iman–Davenport test is carried out (employing F-distribution with 6 and 258 degrees of freedom for Nds = 44) in order to find statistical differences among the algorithms, obtaining a p-value near to zero. In this manner, Table 6 shows the results of the Holm procedure for comparing our proposal with the remaining ones. The algorithms are ordered with respect to the z-value obtained. Thus, by using the normal distribution, we can obtain the corresponding p-value associated with each comparison and this can be compared with the associated α/i in the same row of the table to show whether the associated hypothesis of equal behavior is rejected in favor of the best ranking algorithm, as we can observe the test rejects all cases. We can observe that our approach is statistically superior to all compared methods.

Table 4 Performance ranking for test

Data-sets	1st	2nd	3rd	4th	5th	6th	7th	8th
ecoli0137vs26	S-RSB$_*^a$	Border2[a]	Border1[a]	S-ENN	S-TL	Smote	Safelevel	Original
shuttle0vs4	S-RSB$_*$	Border2	Border1	S-ENN	S-TL	Smote	Original	Safelevel
yeastB1vs7	S-RSB$_*$	S-TL	S-ENN	Smote	Safelevel	Border1	Border2	Original
shuttle2vs4	S-RSB$_*^a$	Safelevel[a]	Border2[a]	Border1[a]	S-ENN[a]	S-TL[a]	Original[a]	Smote
glass016vs2	S-ENN	S-TL	S-RSB$_*$	Safelevel	Smote	Original	Border1	Border2
glass016vs5	Original	S-RSB$_*$	S-ENN	S-TL	Safelevel	Border1	Border2	Smote
pageblocks13vs4	S-RSB$_*^a$	Original[a]	Border1[a]	Smote	Border2	S-TL	S-ENN	Safelevel
yeast05679vs4	Safelevel	S-TL	S-RSB$_*$	Smote	S-ENN	Border1	Border2	Original
yeast1289vs7	S-RSB$_*$	S-ENN	Smote	S-TL	Original	Border1	Safelevel	Border2
yeast1458vs7	S-RSB$_*$	Safelevel	S-TL	Smote	S-ENN	Original	Border1	Border2
yeast2vs4	S-RSB$_*$	S-ENN	S-TL	Safelevel	Border1	Smote	Border2	Original
Ecoli4	S-ENN	S-RSB$_*$	S-TL	Original	Safelevel	Border1	Smote	Border2
Yeast4	Safelevel	S-RSB$_*$	S-TL	S-ENN	Border1	Smote	Border2	Original
Vowel0	Border2	Original	S-RSB$_*$	Safelevel	Smote	S-ENN	S-TL	Border1
Yeast2vs8	S-ENN	Safelevel	Smote	S-TL	S-RSB$_*$	Border2	Border1	Original
Glass4	S-TL	Safelevel	S-RSB$_*$	S-ENN	Smote	Border2	Border1	Original
Glass5	S-RSB$_*$	Original	Safelevel	Border1	Smote	S-TL	Border2	S-ENN
Glass2	S-RSB$_*$	S-ENN	Original	Border1	Safelevel	S-TL	Border2	Smote
Yeast5	S-RSB$_*$	Safelevel	S-TL	S-ENN	Smote	Border2	Border1	Original
Yeast6	S-TL	Smote	S-ENN	S-RSB$_*$	Safelevel	Border1	Border2	Original
abalone19	Safelevel	S-RSB$_*$	Border2	Border1	Smote	S-ENN	S-TL	Original
abalone918	Safelevel	Border1	S-ENN	Border2	S-RSB$_*$	S-TL	Smote	Original
cleveland0vs4	Safelevel	S-TL	Smote	S-RSB$_*$	S-ENN	Border2	Border1	Original
ecoli01vs235	S-TL	Smote	S-ENN	S-RSB$_*$	Safelevel	Border2	Border1	Original
ecoli01vs5	Safelevel	S-TL	Border1	Border2	S-ENN	Original	Smote	S-RSB$_*$
ecoli0146vs5	Smote	S-ENN	S-TL	Safelevel	S-RSB$_*$	Border2	Original	Border1
ecoli0147vs2356	Border2	Smote	S-ENN	S-TL	S-RSB$_*$	Safelevel	Original	Border1
ecoli0147vs56	S-RSB$_*$	Smote	Border2	S-ENN	S-TL	Border1	Original	Safelevel
ecoli0234vs5	S-RSB$_*$	Smote	S-ENN	S-TL	Safelevel	Border1	Border2	Original
ecoli0267vs35	S-TL	Safelevel	Border1	S-RSB$_*$	S-ENN	Smote	Border2	Original
ecoli034vs5	S-RSB$_*$	S-TL	Border2	Smote	Border1	S-ENN	Original	Safelevel
ecoli0346vs5	S-ENN[a]	Smote[a]	Border2	S-TL	S-RSB$_*$	Original	Border1	Safelevel
ecoli0347vs56	S-RSB$_*$	Smote	S-ENN	S-TL	Border2	Border1	Safelevel	Original
ecoli046vs5	S-RSB$_*$	Safelevel	Border2	S-ENN	Smote	S-TL	Border1	Original
ecoli067vs35	Border2	Border1	S-RSB$_*$	Smote	Original	S-ENN	S-TL	Safelevel
ecoli067vs5	Border2	Border1	S-RSB$_*$	Smote	S-ENN	S-TL	Safelevel	Original
glass0146vs2	S-RSB$_*$	Smote	Safelevel	S-TL	S-ENN	Border2	Original	Border1
glass015vs2	S-ENN	Safelevel	S-RSB$_*$	S-TL	Smote	Border2	Border1	Original
glass04vs5	Border2	S-RSB$_*$	Border1	Original	Smote	S-ENN	S-TL	Safelevel
glass06vs5	Border1[a]	Original[a]	S-RSB$_*$	S-ENN	S-TL	Smote	Safelevel	Border2
led7digit02456789vs1	Safelevel	S-RSB$_*$	Border2	Border1	Smote	S-TL	Original	S-ENN
yeast0359vs78	S-RSB$_*$	Safelevel	S-TL	Smote	S-ENN	Border2	Border1	Original

Table 4 continued

Data-sets	1st	2nd	3rd	4th	5th	6th	7th	8th
yeast0256vs3789	Smote	S-RSB$_*$	S-ENN	Border2	Safelevel	Border1	S-TL	Original
yeast02579vs368	Smote	S-ENN	S-RSB$_*$	S-TL	Safelevel	Border2	Border1	Original

[a]The algorithms obtain the same result

Table 5 Rankings obtained through Friedman's test

Algorithm	Ranking
S-RSB$_*$	2.61364
S-ENN	3.92045
S-TL	3.96591
Smote	4.23864
Safelevel	4.34091
Boderline-SMOTE2	5.18182
Boderline-SMOTE1	5.36364

Table 6 Holm's table for $\alpha = 0.05$, S-RSB$_*$ is the control method

i	Algorithm	$z = (R_0 - R_i)/SE$	p	Holm/Hochberg/Hommel	Hypothesis
6	Borderline-SMOTE1	5.2658490926	1.395E-7	0.008333	Reject
5	Borderline-SMOTE2	4.9176937807	8.756E-7	0.01	Reject
4	Safelevel	3.3074754631	9.414E-4	0.0125	Reject
3	Smote	3.1116381002	0.001860	0.016667	Reject
2	S-TL	2.5894051323	0.009614	0.025	Reject
1	S-ENN	2.5023663043	0.012336	0.05	Reject

5 Concluding remarks

In this paper, we have presented a new proposal for editing training sets for highly imbalanced data-sets. The proposal belongs to the set of techniques known as hybrid oversampling and undersampling.

The novelty of this proposal is that the quality of the new synthetic instances is evaluated using RST and the lower approximation of a set of instances (the minority class in this case). This evaluation allows us to include only those artificial instances that are within the lower approximation of the minority class.

From the results of our experimental analysis, we have observed the good average results obtained by the SMOTE-RSB_* technique for preprocessing within the framework of imbalanced data-sets.

Acknowledgments This work had been supported by the Spanish Ministry of Science and Technology under Project TIN2011-28488 and the Andalusian Research Plan P10-TIC-6858. We will like to truly thank Dr. A. Ferndfndez and Dr. J. Luengo for their invaluable help during the development of this paper.

References

1. Alcalá-Fdez J, Sánchez L, García S, del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F (2009) KEEL: a software tool to assess evolutionary algorithms to data mining problems. Soft Comput 13(3):307–318
2. Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Multiple-Valued Log Soft Comput 17(2–3):255–287
3. Asuncion A, Newman D (2007) UCI Machine learning repository. http://mlearn.ics.uci.edu/MLRepository.html
4. Batista GEAPA, Prati RC, Monard MC (2004) A study of the behaviour of several methods for balancing machine learning training data. SIGKDD Explor 6(1):20–29
5. Bello R, Falcon R, Pedrycz W, Kacprzyk J (eds) (2008) Granular computing: at the junction of rough sets and fuzzy sets. Springer
6. Bradley AP (1997) The use of the Area Under the ROC Curve in the evaluation of machine learning algorithms. Pattern Recognit 30(7):1145–1159
7. Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C (2009) 'Safe-Level-SMOTE: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem'. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD09). LNCS 3644. Springer, pp 475–482
8. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: Synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
9. Chawla NV, Japkowicz N, Kolcz A (2004) Editorial: special issue on learning from imbalanced data sets. SIGKDD Explor 6(1):1–6
10. Chawla NV, Cieslak D, Hall L, Joshi A (2008) Automatically countering imbalance and its empirical relationship to cost. Data Min Knowl Discov 17(2):225–252
11. Chen Y-S, Cheng C-H (2010) Forecasting PGR of the financial industry using a rough sets classifier based on attribute-granularity. Knowl Inf Syst 25(1):57–79
12. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
13. Fernández A, García S, del Jesus MJ, Herrera F (2008) A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. Fuzzy Sets Syst 159(18): 2378–2398
14. Fernández A, del Jesus MJ, Herrera F (2010) Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning. 13th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU2010) LNAI 6178. pp 89–98. 159(18):2378–2398
15. Fürnkranz J (2002) Round robin classification. J Mach Learn Res 2:721–747
16. García S, Herrera F (2008) An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J Mach Learn Res 9:2677–2694
17. García S, Herrera F (2009) Evolutionary under-sampling for classification with imbalanced data sets: proposals and taxonomy. Evol Comput 17(3):275–306
18. García S, Fernández A, Luengo J, Herrera F (2009) A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. Soft Comput 13(10): 959–977
19. García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180:2044–2064
20. Greco S (2001) Rough sets theory for multicriteria decision analysis. Eur J Oper Res 129:1–47
21. Grzymala-Busse JW, Stefanowski J, Wilk S (2005) A comparison of two approaches to data mining from imbalanced data. J Intell Manuf 16(6):565–573
22. Han H, Wang WY, Mao BH (2005) Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. International conference on intelligent computing (ICIC05) LNCS 3644. Springer, pp 878–887
23. He H, García EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284
24. Holm S (1979) A simple sequentially rejective multiple test procedure, Scandinavian. J Stat 6:65–70
25. Huang J, Ling CX (2005) Using AUC and accuracy in evaluating learning algorithms. IEEE Trans Knowl Data Eng 17(3):299–310
26. Huan YM, Hung CM, Jiau HC (2006) Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem. Nonlinear Anal Real World Appl 7(4):720–747
27. Iman R, Davenport J (1980) Approximations of the critical region of the Friedman statistic. Commun Stat Part A Theory Methods 9:571–595

28. Ling C, Sheng V (2006) Test strategies for cost-sensitive decision trees. IEEE Trans Knowl Data Eng 18(8):1055–1057
29. Mazurowski M, Habas P, Zurada J, Lo J, Baker J, Tourassi G (2008) Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance. Neural Netw 21(2-3):427–436
30. Midelfar H, Komorowski J, Nørsett K, Yadetie F, Sandvik A, Lægreid A (2003) Learning rough set classifiers from gene expression and clinical data. Fundam Inf 53:155–183
31. Orriols-Puig A, Bernadó-Mansilla E (2009) Evolutionary rule-based systems for imbalanced datasets. Soft Comput 13(3):213–225
32. Pawlak Z (1982) Rough sets. Int J Comput Inf Sci 11:145–172
33. Quinlan J (1993) C4.5 programs for machine learning. Morgan Kaufmann, CA
34. Sheskin D (2003) Handbook of parametric and nonparametric statistical procedures. chapman & hall, CRC Press
35. Slowinski R, Vanderpooten D (1997) Similarity relation as a basis for rough approximations. Adv Mach Intell Soft-Comput 4:17–33
36. Sun Y, Kamel MS, Wong AK, Wang Y (2007) Cost-sensitive boosting for classification of imbalanced data. Pattern Recognit 40:3358–3378
37. Sun Y, Wong AK, Kamel MS (2009) Classification of imbalanced data: a review. Int J Pattern Recognit Artif Intell 23(4):687–719
38. Suresh S, Sundararajan N, Saratchandran P (2008) Risk-sensitive loss functions for sparse multi-category classification problems. Inf Sci 178(12):2621–2638
39. Tomek I (1976) Two modifications of CNN. IEEE Trans Syst Man Commun 6:769–772
40. Tsumoto S (2003) Automated extraction of hierarchical decision rules from clinical databases using rough set model. Expert Syst Appl 24:189–197
41. Wang BX, Japkowicz N (2010) Boosting support vector machines for imbalanced data sets. Knowl Inf Syst 25(1):1–20
42. Wei-hua X, Xiao-yan Z, Jian-min Z, Wen-xiu Z (2008) Attribute reduction in ordered information systems based on evidence theory. Knowl Inf Syst 178(5):1355–1371
43. Weiss GM, Hirsh H (2000) A quantitative study of small disjuncts, In: Proceedings of the 17th national conference on artificial inteligence. pp 665–670
44. Weiss GM, Provost F (2003) Learning when training data are costly: the effect of class distribution on tree induction. J Artif Intell Res 19:315–354
45. Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Commun 2(3):408–421
46. Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, McLachlan GJ, Ng A, Liu B, Yu PS, Zhou Z-H, Steinbach M, Hand DJ, Steinberg D (2008) Top 10 algorithms in data mining. Knowl Inf Syst 14(1):1–37
47. Xu W, Zhang X, Zhong J, Zhang W (2010) Attribute reduction in ordered information systems based on evidence theory. Knowl Inf Syst 25(1):169–184
48. Yang Q, Wu X (2006) 10 challenging problems in data mining research. Int J Inf Technol Decis Mak 5(4):597–604
49. Zhou Z-H, Liu X-Y (2006) Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans Knowl Data Eng 18(1):63–77

Author Biographies

Enislay Ramentol received the MSc degree in Informatics in 2008 in the University of Camagüey, Cuba. She is a currently Ph.D. student of University of Granada; in 2010 received the MSc in Soft Computing in University of Granada, Spain. She is currently a Lecturer in the Department of Computer Science at University of Camagüey, Cuba. Her research interests include imbalanced learning, resampling methods, feature selection and Rough Set Theory.

Yailé Caballero received the MSc in Cybernetic in 2001 and the Ph.D. degree in 2007, both from the Universidad Central "Marta Abreu" de Las Villas (UCLV), Cuba. She is currently the Dean of the Faculty of Informatics, University of Camagüey. She has published more than 60 papers in journals and proceedings of international congresses. Her current research interests include rough set theory, machine learning, metaheuristics and solution of problems of classification and prediction.

Rafael Bello received the MSc in Cybernetic and Mathematic in 1982 and the Ph.D. degree in Mathematics in 1987, both from the Universidad Central "Marta Abreu" de Las Villas (UCLV), Cuba. He is currently a Professor in the Department of Computer Science at UCLV. He has published more than 120 papers in journals and proceedings of international congresses. He has co-edited two international books, and he has been reviewer in several scientific journals (such as Information Sciences, Knowledge-based systems, Fuzzy Sets and Systems, and Journal of Medical Systems) and international congresses (such as ISDA, IEA/AIE, and ISKE). His current research interests include rough set theory, machine learning, metaheuristics and decision making.

Francisco Herrera received his MSc in Mathematics in 1988 and Ph.D. in Mathematics in 1991, both from the University of Granada, Spain. He is currently a Professor in the Department of Computer Science and Artificial Intelligence at the University of Granada. He has had more than 200 papers published in international journals. He is coauthor of the book *"Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases"* (World Scientific, 2001).
He currently acts as Editor in Chief of the international journal "Progress in Artificial Intelligence" (Springer) and serves as area editor of the Journal Soft Computing (area of evolutionary and bioinspired algorithms) and International Journal of Computational Intelligence Systems (area of information systems). He acts as associated editor of the journals: IEEE Transactions on Fuzzy Systems, Information Sciences, Advances in Fuzzy Systems, and International Journal of Applied Metaheuristics Computing; and he serves as member of several journal editorial boards, among others: Fuzzy Sets and Systems, Applied Intelligence, Knowledge and Information Systems, Information Fusion, Evolutionary Intelligence, International Journal of Hybrid Intelligent Systems, Memetic Computation, Swarm and Evolutionary Computation.
He received the following honors and awards: ECCAI Fellow 2009, 2010 Spanish National Award on Computer Science ARITMEL to the "Spanish Engineer on Computer Science", and International Cajastur "Mamdani" Prize for Soft Computing (Fourth Edition, 2010).
His current research interests include computing with words and decision making, data mining, bibliometrics, data preparation, instance selection, fuzzy rule-based systems, genetic fuzzy systems, knowledge extraction based on evolutionary algorithms, memetic algorithms, and genetic algorithms.