Journal of Universal Computer Science, vol. 26, no. 6 (2020), 698-719
submitted: 3/11/19, accepted: 2/4/20, appeared: 28/6/20 CC BY-ND 4.0

Undersampling Instance Selection for Hybrid and Incomplete Imbalanced Data

Oscar Camacho-Nieto
(Instituto Politécnico Nacional, CIDETEC-IPN, CDMX, México
oscarc@cic.ipn.mx)

Cornelio Yáñez-Márquez
(Instituto Politécnico Nacional, CIC-IPN, CDMX, México
cyanez@cic.ipn.mx)

Yenny Villuendas-Rey
(Instituto Politécnico Nacional, CIDETEC-IPN, CDMX, México
yenny.villuendas@gmail.com)

Abstract: This paper proposes a novel undersampling method, for dealing with imbalanced datasets. The proposal is based on a novel instance importance measure (also introduced in this paper), and is able to balance hybrid and incomplete data. The numerical experiments carried out show the proposed undersampling algorithm outperforms others algorithms of the state of art, in well-known imbalanced datasets.

Keywords: undersampling, imbalanced data, hybrid and incomplete data
Categories: I.2.1, I.2.6, I.5.1

1 Introduction

Imbalanced datasets are a challenge for supervised classifiers. Several datasets contain imbalance between classes, and usually the minority class is the interest class (whose incorrect classification is less desirable) [Fernández et al., 18]. For instance, let us suppose that we have 1000 persons, 10 of which are HIV positive. A classifier with a 99% accuracy, misclassifying the persons with HIV, is not useful at all. As in the example, many applications are imbalanced by nature, such as the diagnosis of some diseases [Bagby et al., 19; Fotouhi et al., 19; Rajesh and Dhuli, 18], credit assignment [Cleofas-Sánchez et al., 16; Garcia et al., 19; Yu et al., 18], and many others.

Many of these applications also have associated additional complexity, because the attributes can be hybrid or mixed [Hu et al., 06; Kim and Hong, 17], and in addition, there can be absences of information in the attribute values [Cheng at al., 12; Li et al., 16; Režnáková et al., 17; Ruiz-Shulcloper, 08]. The sum of these three factors (imbalance, hybrid attributes and lack of information) complicates the classification task, and affects the performance of the classifiers [Fernández et al., 18].

There are several ways to deal with the three factors mentioned above, which are summarized in data-level solutions, algorithm-level solutions and cost-based classification [López et al., 12]. The solutions at the data level to coping with hybrid attributes and lack of information, consist of unifying the type of the attributes (either by discretizing the numerical attributes or by digitizing the categorical attributes) and by completing the absences of information by means of imputation of values techniques. On the other hand, solutions at the algorithm level consist in managing the mixed attributes and the lost values as part of the training and classification phases.

In the case of imbalance, the solutions at the level of algorithms consist fundamentally in modifying the operation of the classifiers to consider the imbalanced data [Galar et al., 13]. In addition, these solutions usually use cost techniques to weigh errors differently, so that the classifiers are robust to the imbalance; favoring in this way the correct classification of the minority class [López et al., 12].

Several researchers have proven the usefulness of instance selection to improve classifier performance [Carbonneau et al., 16; Leyva et al., 15; Li et al., 14]. That is why, on the other hand, data-level solutions to attack the imbalance consist of balancing the data set by undersampling the majority class, oversampling the minority class, or a combination of both sampling techniques [Fernández et al., 18]. The oversampling techniques are based on the creation of artificial instances that are "similar" to the original ones, as it does the well-known SMOTE [Chawla et al., 02] and other algorithms [Castellanos et al., 18; Sáez et al., 16]. Carrying out this task is very difficult when dealing with mixed attributes and with absences of information, because the techniques for oversampling create artificial instances by subtracting and multiplying the values of the original instances. Another drawback of oversampling is the additional computational cost coming from creating artificial instances. That is why we consider that, in the presence of data with mixed attributes and absences of information; it is preferred to use undersampling techniques.

However, there is no algorithm capable of outperforming all existing methods in the state of the art in all possible scenarios, so the problem of selecting instances for class balancing is still valid. In addition, many of the algorithms proposed for undersampling perform a random sampling of the majority class, or of the instances close to the decision boundaries [Batista et al., 04; Kubat and Matwin, 97; Yen and Lee, 06]. Thus, the distribution of the samples in the majority class is not considered, which leads to losses in the performance of the classifiers.

That is why in this article we present a new approach, which is based on the recently proposed Maximal Similarity Granular Rough Sets, and in a new measure of instance importance, for the balancing of instances by undersampling. The new measure is presented for the first time in this paper and provides conceptual support to the new approach. Our idea is to structure the majority class, and later, to use a new merging approach and the proposed importance measure to select the instances of the majority class. In this way, we guarantee that the selected instances are representative of the majority class, thus ensuring the correct classification of as many instances as possible.

The two fundamental contributions of this article are:

1. The proposal of a new measures to estimate the importance of objects.
2. The proposal of a new algorithm for the selection of instances by undersampling, based on the proposed measure.

The article is structured as follows: section 2 reviews existing methods for undersampling imbalanced data, section 3 details the proposed measure for instance importance, and the proposed undersampling algorithms. The experimental comparisons made on imbalanced data sets are given in section 4, and they show that our proposal obtains better results with respect to other algorithms of the state of the art. Finally, conclusions and references are included.

2 Data balancing by undersampling

Undersampling algorithms aim at matching the quantities of instances in each class, by sampling the majority classes [Fernández et al., 18]. Thus, instances that are considered less relevant are eliminated, so that all classes have approximately the same number of objects. Next, we briefly review some of the undersampling algorithms in the literature.

The first condensing algorithm proposed in the literature is the CNN (Condensed Nearest Neighbor) rule, by Hart [Hart, 68]. Although it is not designed for diminish the effects of data imbalance, it is still used in several experimental comparisons. Similarly, the Tomek's modification of CNN, named as TL (Tomek’s Link) [Tomek, 76], is also a dated algorithm for instance selection. In this method if two instances form a TL, then one of them is noise or the two instances are on the border. Prior to the application of the CNN, this method obtains a set of objects that contains only the objects near the decision boundaries.

One Sided Selection (OSS) [Kubat and Matwin, 97] is probably the first undersampling method, designed specifically to deal with class imbalance. In this method, all the instances belonging to the minority class are selected, and the instances of the majority class participating in a Tomek link are removed.

Another undersampling method is the NCL (Neighbourhood Cleaning) rule [Laurikkala, 01], which uses the ENN (Edited Nearest Neighbor) rule [Wilson, 72] to remove objects from the majority class. ENN removes any instance whose class differs from the class of at least three of its five closest neighbors.

In 2004, Batista et al. [Batista et al., 04] carried out a study of sampling techniques to deal with class imbalance. In such study, they proposed the RUS (Random Under Sampling) and the CNNTL (CNN + TL) algorithms. RUS randomly selects the desired number of instances from the majority class, while CNNTL applies first the CNN algorithm, followed by the TL algorithm.

To overcome the drawbacks of the previous undersampling techniques, several authors have use clustering algorithm to better represent the majority class. One of such proposals is the Class Purity Maximization (CPM) algorithm [Yoon and Kwek, 05]. In this method, two instances of both the minority and the majority class are selected as centres. The other instances are distributed in two subsets according to their closest centres, with at least one subset that has a high class purity. This process is recursively repeated for each of the two subsets until it can no longer form two groups, with at least one class of higher purity than the parent group. Only the instances belonging to a non-pure subset are passed to a decision tree committee to decide if they are removed from the set.

Another algorithm based on clustering is SBC (under-Sampling Based on Clustering) [Yen and Lee, 06]. SBC clusters the data into k clusters (k is a user parameter), and then it selects a suitable number of majority class instances from each

cluster by considering the ratio of the number of majority class instances to the number of minority class instances in the cluster.
All the previously mentioned methods have, to a lesser or greater extent, managed to solve the problems of data imbalance by applying undersampling techniques; however, they do not guarantee a good representation of the majority class, in terms of the distribution of their instances, which leads to poorly represented classes. This motivated us to invent the new measure of importance of the instances, using the Maximal Similarity Granular Rough Sets, to propose a new approach, which will be described in the next section.

3 Proposed method

Our proposal, which we have named as Balanced by Rough Importance Selection (BRIS), uses a clustering-based approach for undersampling. However, it differentiates from previous proposals in several aspects:

1. We separate the majority classes, and then we cluster each of them.
2. We do not predefine the number of initial clusters.
3. We successfully deal with multi-class (having more than two classes) data.
4. We also deal directly with hybrid as well as with incomplete data.

We first detect which the minority class is, among all classes. We consider the number of instances in the minority class as min. Then, we separate each of the non-minority classes, and we design a procedure to select, from each of those classes, a number of representative instances equal to min.

Section 3.1 details the procedure to compute the importance of instances, which later are used to select the representative ones (we select the most important instances as representatives of their classes).
Section 3.2 faces the procedure to preserve the distribution of the non-minority classes, in a way such that all regions are represented by the more important instances. Section 3.3 gives the overall procedure of the proposed algorithm for data balancing by undersampling

3.1 Measuring instance importance

One of the greatest challenges in undersampling techniques is to obtain a good representation of the majority class. For doing so, we design a strategy of measuring the importance of the instances, based on the recently proposed Maximal Similarity Granular Rough Sets (MSGRS) [Villuendas-Rey, 19].

3.1.1 Some definitions of Rough Sets

This subsection is strongly based on [Villuendas-Rey, 19]. In Rough Set Theory, an Information System (IS) is a pair (U, A) where U is a universe of instances, cases or examples, which are described by a set of attributes or features A. If we add a decision attribute d to the pair, we have a new pair $(U, A \cup \{d\})$, named Decision System (DS). According to Rough Set Theory, we can approximate a concept $X \subseteq U$ by means of two sets: lower and upper approximations. To do so, we use a relation among instances. Let (U, A) be an IS, let $B \subseteq A$ be a subset of attributes, let $x, y \in U$ be two instances, and let x_i and y_i be the values of the attribute $A_i \in B$ on instances x and y, respectively. In classic definitions of Rough Sets, the indiscernibility relation $R \subseteq U \times U$ with respect to B is denoted as R_B and it is defined as $xR_By \leftrightarrow \forall A_i \in B, x_i = y_i$. The equivalence class of an instance x with respect to R_B is denoted as $B(x)$ and is defined as $B(x) = \{y \in U: yR_Bx\}$.

Considering the equivalence classes, a set X has two approximations: lower and upper. The lower approximation is denoted as $LOW_{R_B}(X) = \{x \in U: B(x) \subseteq X\}$ and the upper approximation is $UPP^{R_B}(X) = \{x \in U: B(x) \cap X \neq \emptyset\}$.

3.1.2 MSGRS and instance importance

Different from classical Rough Sets, MSGRS has the advantage of dealing with hybrid as well as with incomplete data, and they do not need any parameter for the construction of the approximations, except a similarity function. In MSGRS, MS is defined as a maximum similarity operator as follows [Villuendas-Rey, 19]:

Definition ***1***. Let (U, A) be an Information System, let $B \subseteq A$, let $x \in U_B$ be any instance in the universe U described only by the attributes in B, denoted as U_B and let $sim_B(x, z)$ be any similarity function between two instances belonging to U_B. The maximum similarity operator is defined as the application of the max operator over the similarity function sim_B:

$$MS_B(x) = \max_{z \in U_B, z \neq x} (sim_B(x, z)) \tag{1}$$

Thus, the similarity class and the lower and upper approximations of MSGRS given a set $X \subseteq U_B$, $X \neq \emptyset$ are defined by the expressions 2 - 4:

$$R_B(x) = \{y \in \{U_B - \{x\}\}: MS_B(x) > 0\} \cup \{x\} \tag{2}$$

$$LOW_{R_B}(X) = \{x \in U_B: R_B(x) \subseteq X\} \tag{3}$$

$$UPP^{R_B}(X) = \{x \in U_B: R_B(x) \cap X \neq \emptyset\} \tag{4}$$

As shown, in a MSGRS, the instances in the lower approximation has in their similarity class, instances only from its decision class (fulfilling condition $MS_B(x) > 0$). However, we can rewrite the maximum similarity operator of an instance x, to know how many instances from its same decision class are closer than the most similar instance from another decision class. We consider the more instances are in the similarity classes, the more important the instance is.

Definition ***2***. Let (U, A) be an Information System, let $B \subseteq A$, let $x \in U_B$ be any instance in the universe U described only by the attributes in B, denoted as U_B, and let X be the decision class of the instance x. The maximum competitor similarity operator is defined as the application of the max operator over the similarity function sim_B:

$$MCS_B(x) = \max_{z \in U_B, z \notin X} (sim_B(x, z)) \tag{5}$$

By using definition 2, we introduce the friendly similarity class of an instance, which is the first contribution of this paper.

Definition ***3.*** Thus, the friendly similarity class $F_B(x)$ of the instance x is given by:

$$F_B(x) = \{y \in X: sim_B(x, y) > MCS_B(x)\} \tag{6}$$

Definition 4. Let $x \in U_B$ be any instance in the universe U described only by the attributes in B, let $X \subseteq U_B$ be a decision class, and $x \in$ X. The importance of the instance x with respect to the attribute set $B \subseteq A$ is denoted as $I_B(x)$, and will be computed as the number of instances related with x by F_B, belonging to the same decision class of x:

$$I_B(x) = |F_B(x)| \tag{7}$$

Thus, the greater $I_B(x)$ the more important the instance x is. Figure 1 shows the pseudocode for the procedure to compute the importance of an instance.

```
Algorithm # 1: Rough Importance
Inputs: Instance $x$, Universe of instances $U$, Set of attributes $B$, Similarity function $sim_B$
Output: Importance of the instance $x$ denoted as $I_B(x)$
Steps:
1. Let $X$ be the decision class of the instance $x$
2. For each instance $z \in U_B, z \notin X$
    2.1. Compute the similarity as $(sim_B(x, z)$
3. Find the maximum competitor similarity of $x$ as $MCS_B(x) = \max_{z \in U_B, z \notin X} (sim_B(x, z))$
4. $F_B(x) = \emptyset$
5. For each $y \in X$
    5.1. If $(sim_B(x, y) > MCS_B(x)$ then $F_B(x) \leftarrow F_B(x) \cup \{y\}$
6. $I_B(x) = |F_B(x)|$ // The |  | is the cardinality of a set operator
Return $I_B(x)$
```

Figure 1: Algorithm to compute the importance of an instance.

Definition 4 is one of the most important contributions of this paper, due to it allows us to introduce, by using definition 3, the concept of instance importance, key for the posterior development of the proposed undersampling algorithm.

3.2 Preserving distribution of non-minority classes

A challenge of undersampling methods is to obtain good representations of the majority class, using few instances (usually, just the number of instances in the minority class). Several density-based clustering algorithms can be used for these purpose, such as DBSCAN [Ester et al., 96], however, they have the inconvenient of dealing only with numeric or with categorical data, and no with hybrid data.

3.2.1 Initial preservation of data distribution

To address this issue, we opted for using a clustering algorithm from the Logical Combinatorial Approach to Pattern Recognition [Ruiz-Shulcloper, 08], able to deal with hybrid data, and designed to preserve the structure of the data. The Compact Set Structuralization (CSS) [Trinidad et al., 00] has as parameter a similarity function between instances and a minimum similarity threshold (β_0) usually set to zero. CSS first computes a Maximum Similarity Graph of the instances (each instance is connected to its most similar instances, if such maximum similarity overpasses the threshold), and then return the Compact Sets, which are the connected components of the Maximum Similarity Graph computed. Formally, a Compact Set is defined as follows:

Let U be a set of instances, β_0 be a similarity threshold, $B \subseteq A$ be an attribute set describing the instances, is and $sim_B(x, y)$ a similarity function among instances $x, y \in U$. A subset $cs \neq \emptyset$ of U is a β_0-compact set if and only if:

$$a)\forall \mathrm{y} \in U \left[\mathrm{x} \in cs \wedge \begin{pmatrix} \max\limits_{\substack{\mathrm{z} \in U \\ \mathrm{z} \neq \mathrm{x}}} \{ sim_B(\mathrm{x}, \mathrm{z}) \} = sim_B(\mathrm{x}, \mathrm{y}) \geq \beta_0 \\ \vee \max\limits_{\substack{\mathrm{z} \in U \\ \mathrm{z} \neq \mathrm{x}}} \{ sim_B(\mathrm{z}, \mathrm{x}) \} = sim_B(\mathrm{y}, \mathrm{x}) \geq \beta_0 \end{pmatrix} \right] \Rightarrow \mathrm{y} \in cs$$

$$b)\forall x, \mathrm{y} \in cs, \exists x_1, \cdots, x_q \in cs \left[\begin{matrix} \mathrm{x} = x_1 \wedge \mathrm{y} = x_q \wedge \forall p \, \{1, \cdots, q-1\} \\ \left[\begin{matrix} \max\limits_{\substack{\mathrm{z} \in U \\ \mathrm{z} \neq x_p}} \{ sim_B(x_p, \mathrm{z}) \} = sim_B(x_p, x_{p+1}) \geq \beta_0 \\ \vee \max\limits_{\substack{\mathrm{z} \in U \\ \mathrm{z} \neq x_p}} \{ sim_B(x_{p+1}, \mathrm{z}) \} = sim_B(x_{p+1}, x_p) \geq \beta_0 \end{matrix} \right] \end{matrix} \right]$$

$c)$Every β_0-isolated instance (not connected with other instance) is a degenerated Compact Set.

The main drawback of the CSS is that the number of clusters is unknown, and it is usually much greater than the number of instances in the minority class, min. To solve this problem, we designed an agglomerative approach, in a way such that we are able to return the desired amount of instances from each non-minority class. After computed the CSS of each non-minority class, we proceed to merge the Compact Sets, following an agglomerative approach. Our merging strategy (figure 2) differentiates from other in the literature in the following:

It uses the importance of instances ($I_B(x)$, section 3.1) to select the cluster representatives (we use the words cluster representative instead of cluster center, due to we are dealing with hybrid data, and thus we have no centroids. Instead, we selected from of the available instances the most important one).

It merges all most similar clusters at once, by using the similarity between cluster representatives as inter-cluster similarity. By doing so, we avoid order dependence, and we diminish the computational cost associate with agglomerative strategies for instance clustering. In addition, using the similarity between cluster representatives as inter-cluster similarity also reduces computational cost, due to the similarity matrix between instances is computed only once, and then, the values are used.

Note that, if the number of initial clusters in the non-minority class is lower than the number of instances in the minority class, the proposed algorithm compensates the difference by randomly selecting instances from the non-minority class clusters.

Algorithm # 2: Cluster Merging
Inputs: Compact Sets CS of the class to cluster Number of clusters to obtain (min) Universe of instances U Set of attributes B Similarity function sim_B
Output: Set of remaining clusters RC

```
1.   $RC \leftarrow CS$
2.   If ($|RC| < min$)
  2.1.   Do
      2.1.1.   $R \leftarrow \emptyset$
      2.1.2.   Foreach cluster $cs \in RC$
          2.1.2.1.   Randomly select an instance $x \in cs$
          2.1.2.2.   $R \leftarrow R \cup \{x\}$
         Until $|RC| + |R| = min$
  2.2.   $RC \leftarrow RC \cup R$
3.   While ($|RC| > min$)
  3.1.   Foreach cluster $cs \in RC$
      3.1.1.   Foreach instance $x \in cs$, compute its importance as
               $Rough\ Importance(x, U, B, sim_B)$.
               //The detailed pseudocode of the Rough Importance algorithm is given in
               Figure 1.
  3.2.   Select as cluster representative the instance with greater importance.
  3.3.   Merge all most similar clusters in $cs$, using as inter-cluster similarity the similarity
         between cluster representatives
  3.4.   Foreach cluster $cs \in RC$
      3.4.1.   Update cluster representatives
4.   Return $RC$
```

Figure 2: Algorithm to merge the clusters of a non-minority class.

3.2.2 Selecting representatives of the non-minority classes

After obtained the desired amount of clusters of the non-minority classes, and by maintaining as much as possible the data distribution, our proposal will select a representative instance from each cluster. As clusters were merged according to the similarity of the most important instances of each cluster, now we want to select an instance similar to all instances in the cluster.

To do so, we will compute the average similarity of each instance with respect other instances in its cluster, and we will select the instance with greater average similarity (Figure 3).

```
Algorithm # 3: Representatives
Inputs:
Set of remaining clusters $RC$ of a non-minority class
Set of attributes $B$
Similarity function $sim_B$
Output: Set of representative instances $RI$ of a non-minority class
1.  $RI \leftarrow \emptyset$
2.  Foreach cluster $cs \in RC$
    2.1. Foreach instance $x \in cs$, compute its average similarity with
         respect other instances as $Avg(x) = \underset{z \in cs}{\operatorname{avg}}(sim_B(x, z))$.
    2.2. Select as cluster representative the instance with greater average
         similarity, as $y = \underset{x \in cs}{\operatorname{argmax}} Avg(x)$.
    2.3. $RI \leftarrow RI \cup \{y\}$
Return $RI$
```

Figure 3: Algorithm to select representative instances of a non-minority class.

3.3 Overall procedure of BRIS algorithm

Let $(U, A \cup \{d\})$ be a Decision System, let $B \subseteq A$, where the attributes in A can be continuous, categorical or with missing values. Let $x \in U_B$ be any instance in the universe U described only by the attributes in B, denoted as U_B, and let the decision attribute d be multi-valued, and let $U = \bigcup_{i=1}^{n} X_i$, where the decision class X_i is formed by the instances having value d_i in the decision attribute d, and let $|X_i|$ the cardinality of the i-th decision class. The minority class of the Decision System will be noted as X_m, and defined as $X_m = \underset{i=1..n}{\operatorname{argmin}} |X_i|$.

We will consider all other classes, with $i \neq m$, as non-minority, and therefore, majority classes. We consider the number of instances in the minority class as $min = |X_m|$. The proposed undersampling procedure works as follows:

First, we separate the minority and non-minority classes. Automatically, the instances in the minority class will be selected. For each non-minority classes, we compute its Compact Sets [Trinidad et al., 00]. Each Compact Set is considered as a cluster initially. Then, the importance for each of the instances in each compact is calculated. The instance that has the greater value (potentially further away from the decision boundaries), is selected as the representative of the cluster. Then, to obtain min clusters to represent the class, we use an agglomerative process, according to the merging algorithm presented in figure 2, which is based in an instance importance measure (whose computation is described in figure 1).

The algorithm selects the most similar clusters (taking as similarity between clusters the similarity between representatives) and merges them. The algorithm merges all the most similar groups in one step, avoiding order dependency and making clusters faster. The agglomerative process continues until the desired number of groups is obtained (number of objects of the minority class, min). Later, we will select a representative instance of each cluster, to form the representative instance set for the corresponding non-minority class. The overall undersampling procedure is drafted in figure 4.

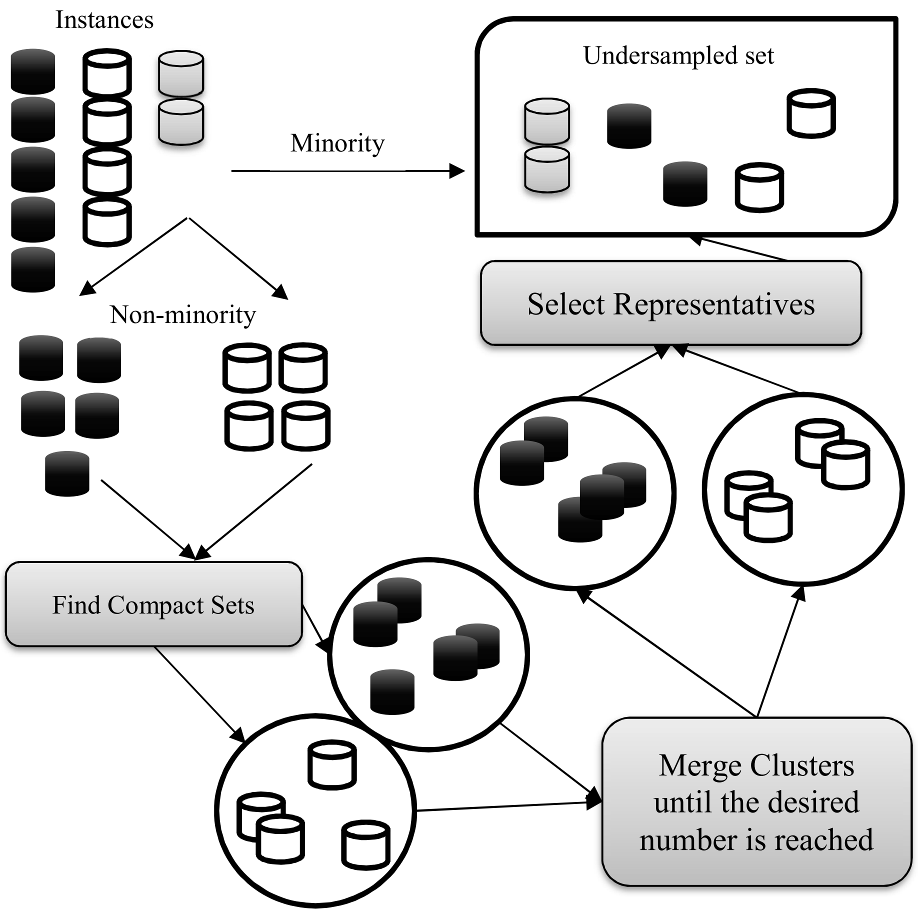

Figure 4: Overall scheme of the proposed undersampling algorithm.

4 Experimental comparisons

The experimental study was conducted on a computer with AMD Sempron SI-42 processor at a speed of 2.1 GHz, and usable RAM of 2.75 GB. This computer was not destined exclusively to the run of the algorithms, which always were executed with priority below normal. This is why it was not possible to perform a study of the execution time used by them.

4.1 Experimental framework

We decide using the Area under the ROC curve (AUC) and Matthews Correlation coefficient (MCC) as classifier performance measure. Considering the two classes confusion matrix of figure 5, the AUC is computed as [Sokolova and Lapalme, 09], and MCC [Matthews, 75] as:

$$AUC = \frac{1}{2}\left(\frac{tp}{tp+fn} + \frac{tn}{tn+fp}\right) \tag{8}$$

$$\begin{aligned} MCC &= \frac{\frac{tp}{N} - S * P}{\sqrt{P * S * (1-S) * (1-P)}} \\ N &= tp + tn + fp + fn \\ S &= \frac{tp+fn}{N} \\ P &= \frac{tp+fp}{N} \end{aligned} \tag{9}$$

In addition, we compute the Imbalance Ratio (IR) of the undersampling algorithms, measure as the instance ratio between the majority and minority classes.

Data class	Classified as *positive*	Classified as *negative*
positive	true positive (tp)	false negative (fn)
negative	false positive (fp)	true negative (tn)

Figure 5: Confusion matrix.

Dataset	**Atts**	**Instances**	**IR**	**Dataset**	**Atts**	**Instances**	**IR**
abalone19	8	4174	129.44	pageblocks13_vs_4	10	472	15.86
abalone918	8	731	16.40	pima	8	768	1.87
ecoli0_vs_1	7	220	1.86	segment0	19	2308	6.02
ecoli0137_vs_26	7	281	39.14	shuttlec0vsc4	9	1829	13.87
ecoli1	7	336	3.36	shuttlec2vsc4	9	129	20.50
ecoli2	7	336	5.46	vehicle0	18	846	3.25
ecoli3	7	336	8.60	vehicle1	18	846	2.90
ecoli4	7	336	15.8	vehicle2	18	846	2.88
glass0	9	214	2.06	vehicle3	18	846	2.99
glass0123_vs_456	9	214	3.20	vowel0	13	988	9.98
glass016_vs_2	9	192	10.29	wisconsin	9	683	1.86
glass016_vs_5	9	184	19.44	yeast05679_vs_4	8	528	9.35
glass1	9	214	1.82	yeast1	8	1484	2.46
glass2	9	214	11.59	yeast1_vs_7	8	459	14.30
glass4	9	214	15.47	yeast1289_vs_7	8	947	30.57
glass5	9	214	22.78	yeast1458_vs_7	8	693	22.10
glass6	9	214	6.38	yeast2_vs_4	8	514	9.08
haberman	3	306	2.78	yeast2_vs_8	8	482	23.10
iris0	4	150	2.00	yeast3	8	1484	8.10
newthyroid1	5	215	5.14	yeast4	8	1484	28.10
newthyroid2	5	215	5.14	yeast5	8	1484	32.73
pageblocks0	10	5472	8.79	yeast6	8	1484	41.40

Table 1: Description of the datasets used.

We use the 5-fold-cross validation procedure. We also use non-parametric tests (Friedman test [Friedman, 37; Friedman 40] and Holm post hoc test [Holm, 79]) for statistical comparisons [Garcia and Herrera, 08].

We used 44 imbalanced datasets from the KEEL repository [Alcalá-Fernández et al., 11], which are described in table 1. Their imbalance ratio (IR) ranges between 1.82 and 129.44, and their instance number ranges between 150 and 5472. The attribute number (Att) ranges between four and 19.

We tested our proposal (BRIS) against six undersampling algorithms: CNNTL, CPM, NCL, OSS, RUS, and SBC, all of them described previously in section 2. These undersampling algorithms are available at KEEL software [Alcalá-Fernández, 09; Alcalá-Fernandez, 11] in its release 3.0.

We maintain the default parameter values for all of the compared algorithms. We use the Nearest Neighbor [Cover and Hart, 67] (NN) classifier as base classifier.

Dataset	**Atts**	**Instances**	**IR**	**Dataset**	**Atts**	**Instances**	**IR**
abalone19	8	4174	129.44	pageblocks13_vs_4	10	472	15.86
abalone918	8	731	16.40	pima	8	768	1.87
ecoli0_vs_1	7	220	1.86	segment0	19	2308	6.02
ecoli0137_vs_26	7	281	39.14	shuttlec0vsc4	9	1829	13.87
ecoli1	7	336	3.36	shuttlec2vsc4	9	129	20.50
ecoli2	7	336	5.46	vehicle0	18	846	3.25
ecoli3	7	336	8.60	vehicle1	18	846	2.90
ecoli4	7	336	15.8	vehicle2	18	846	2.88
glass0	9	214	2.06	vehicle3	18	846	2.99
glass0123_vs_456	9	214	3.20	vowel0	13	988	9.98
glass016_vs_2	9	192	10.29	wisconsin	9	683	1.86
glass016_vs_5	9	184	19.44	yeast05679_vs_4	8	528	9.35
glass1	9	214	1.82	yeast1	8	1484	2.46
glass2	9	214	11.59	yeast1_vs_7	8	459	14.30
glass4	9	214	15.47	yeast1289_vs_7	8	947	30.57
glass5	9	214	22.78	yeast1458_vs_7	8	693	22.10
glass6	9	214	6.38	yeast2_vs_4	8	514	9.08
haberman	3	306	2.78	yeast2_vs_8	8	482	23.10
iris0	4	150	2.00	yeast3	8	1484	8.10
newthyroid1	5	215	5.14	yeast4	8	1484	28.10
newthyroid2	5	215	5.14	yeast5	8	1484	32.73
pageblocks0	10	5472	8.79	yeast6	8	1484	41.40

Table 1: Description of the datasets used.

4.2 Results

Tables 2 and 3 show the AUC and MCC results for the compared undersampling algorithms. In the first column we present the value obtained with the NN classifier in the original datasets, without having applied any instance selection algorithm.

Dataset	**NN**	**CNNTL**	**CPM**	**NCL**	**OSS**	**RUS**	**SBC**	**BRIS**
abalone19	0.52	0.50	0.51	0.50	0.50	0.51	0.20	**0.63**
abalone918	0.63	0.52	0.59	0.63	0.54	0.55	0.54	**0.73**
ecoli0_vs_1	**0.97**	0.89	0.94	0.96	0.93	0.95	0.18	**0.97**
ecoli0137_vs_26	0.72	0.42	0.64	0.67	0.54	0.55	0.01	**0.79**
ecoli1	0.84	0.76	0.74	0.81	0.78	0.80	0.25	**0.85**
ecoli2	0.85	0.67	0.85	0.86	0.72	0.76	0.08	**0.90**
ecoli3	0.71	0.68	0.69	0.72	0.72	0.68	0.05	**0.82**
ecoli4	0.86	0.72	0.81	0.86	0.83	0.72	0.03	**0.95**
glass0	0.78	0.74	0.74	0.80	0.75	0.78	0.16	**0.79**
glass0123_vs_456	0.89	0.80	**0.93**	0.92	0.88	0.92	0.91	**0.93**
glass016_vs_2	0.59	0.58	0.57	0.61	**0.64**	0.55	0.04	0.60
glass016_vs_5	0.83	0.74	0.80	0.76	0.76	0.65	0.02	**0.92**
glass1	0.70	0.72	0.70	0.70	0.73	0.76	0.27	**0.77**
glass2	0.61	0.61	0.59	0.59	0.61	0.56	0.04	**0.62**
glass4	0.79	0.66	0.65	0.78	0.74	0.67	0.03	**0.89**
glass5	0.86	0.86	0.89	0.81	0.81	0.63	0.02	**0.92**
glass6	0.86	0.69	0.90	0.93	0.77	0.89	0.07	**0.93**
haberman	0.55	**0.59**	0.53	0.56	0.57	0.58	0.55	0.58
iris0	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.17	**1.00**
newthyroid1	0.95	0.95	0.95	0.96	0.95	0.94	0.08	**0.99**
newthyroid2	0.95	0.89	0.92	0.96	0.93	0.92	0.08	**0.99**
pageblocks0	0.82	0.70	**0.83**	**0.83**	0.76	0.78	0.70	**0.83**
pageblocks13_vs_4	0.89	0.86	0.87	0.96	0.95	0.70	0.03	**0.96**
pimaimb	0.67	0.63	0.62	**0.69**	0.67	0.68	0.67	**0.69**
segment0	0.97	0.96	0.95	0.98	0.96	0.94	0.07	**0.99**
shuttlec0vsc4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.03	**1.00**
shuttlec2vsc4	0.90	0.77	**1.00**	**1.00**	**1.00**	0.83	0.02	**1.00**
vehicle0	0.86	0.84	0.86	0.87	0.86	0.85	0.12	**0.92**
vehicle1	0.65	0.63	0.61	**0.66**	0.63	0.61	0.13	**0.66**
vehicle2	0.89	0.83	0.89	0.91	0.86	0.87	0.38	**0.93**
vehicle3	0.66	0.66	0.66	0.70	0.68	0.66	0.13	**0.73**
vowel0	0.92	0.99	0.94	**1.00**	0.99	0.85	0.05	0.99
wisconsin	0.96	0.94	0.92	**0.97**	0.96	0.96	0.17	**0.97**
yeast05679_vs_4	0.68	0.62	0.68	0.71	0.65	0.63	0.05	**0.75**
yeast1	0.64	0.62	0.59	0.65	0.63	0.62	0.14	**0.67**
yeast1_vs_7	0.59	0.56	0.58	0.62	0.59	0.56	0.03	**0.69**
yeast1289_vs_7	0.58	0.52	0.54	0.56	0.51	0.52	0.02	**0.66**
yeast1458_vs_7	0.52	0.52	0.55	0.56	0.51	0.52	0.02	**0.60**
yeast2_vs_4	0.83	0.65	0.75	0.83	0.72	0.74	0.05	**0.88**
yeast2_vs_8	0.72	0.55	0.64	0.77	0.57	0.55	0.02	**0.81**
yeast3	0.84	0.67	0.72	0.79	0.73	0.71	0.05	**0.87**
yeast4	0.62	0.55	0.60	0.65	0.58	0.56	0.02	**0.79**
yeast5	0.79	0.74	0.73	0.81	0.77	0.66	0.01	**0.93**
yeast6	0.73	0.53	0.61	0.69	0.54	0.55	0.01	**0.80**

Table 2: AUC of the undersampling algorithms. Best results in bold.

The next six columns (CNNTL, CPM, NCL, OSS, RUS, and SBC) contain the values obtained with the NN classifier, after applying the corresponding undersampling algorithm to the data banks of our experimental study. Finally, in the BRIS column we present the value obtained with the NN classifier, after applying our proposal.

Dataset	**CNNTL**	**CPM**	**NCL**	**OSS**	**RUS**	**SBC**	**BRIS**
abalone19	0.02	0.03	0.01	0.01	**0.06**	0.02	0.05
abalone918	0.14	0.23	**0.34**	0.13	0.19	0.17	0.27
ecoli0_vs_1	0.10	**0.52**	0.54	0.25	0.26	-	0.26
ecoli0137_vs_26	0.78	0.83	0.88	0.81	0.86	-	**0.94**
ecoli1	0.55	0.41	0.66	0.59	**0.67**	-	0.66
ecoli2	0.45	0.69	**0.73**	0.53	0.60	-	0.71
ecoli3	0.50	0.38	0.48	0.57	**0.52**	-	0.48
ecoli4	0.60	0.41	**0.74**	0.69	0.65	-	0.73
glass0	0.65	0.84	**0.88**	0.78	0.84	0.86	0.83
glass0123_vs_456	0.24	0.26	0.17	**0.28**	0.19	-	0.10
glass016_vs_2	0.49	0.57	0.65	0.57	0.46	-	**0.68**
glass016_vs_5	0.52	0.46	**0.65**	0.50	0.58	-	0.55
glass1	0.38	0.47	0.35	0.44	**0.54**	-	**0.54**
glass2	**0.41**	0.25	0.22	0.23	0.23	-	0.13
glass4	0.57	0.32	0.56	**0.63**	0.56	-	0.60
glass5	0.75	0.72	**0.78**	0.74	0.48	-	0.55
glass6	0.51	0.66	**0.89**	0.52	0.82	-	0.83
haberman	0.18	0.02	0.16	0.15	**0.19**	0.11	0.14
iris0	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	-	**1.00**
newthyroid1	0.94	0.87	**0.96**	0.95	0.93	-	0.93
newthyroid2	0.81	0.81	0.95	0.89	0.88	-	**0.97**
pageblocks0	0.82	0.75	**0.90**	0.87	0.56	-	0.73
pageblocks13_vs_4	0.56	0.69	**0.72**	0.62	0.66	0.54	0.66
pimaimb	0.26	0.24	**0.40**	0.36	0.35	0.33	0.34
segment0	0.95	0.90	**0.97**	0.95	0.94	-	0.95
shuttlec0vsc4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	-	0.99
shuttlec2vsc4	0.55	0.83	0.83	0.83	0.47	-	**1.00**
vehicle0	0.76	0.71	0.76	0.76	0.76	-	**0.77**
vehicle1	0.30	0.19	**0.36**	0.27	0.24	-	0.30
vehicle2	0.72	0.78	**0.85**	0.76	0.77	-	0.79
vehicle3	0.36	0.34	**0.45**	0.39	0.36	-	0.40
vowel0	**1.00**	0.90	**1.00**	**1.00**	0.80	-	0.89
wisconsin	0.91	0.82	**0.94**	0.93	0.92	-	0.93
yeast05679_vs_4	0.38	0.42	**0.59**	0.41	0.39	-	0.37
yeast1	0.09	**0.14**	0.10	0.08	0.09	-	0.11
yeast1_vs_7	0.12	0.20	**0.26**	0.14	0.05	-	0.08
yeast1289_vs_7	0.23	0.30	**0.33**	0.33	0.24	-	0.20
yeast1458_vs_7	0.46	0.52	**0.71**	0.57	0.62	-	0.61
yeast2_vs_4	0.24	0.42	**0.60**	0.30	0.21	-	0.29
yeast2_vs_8	0.28	0.19	**0.36**	0.28	0.28	-	0.30
yeast3	0.52	0.51	**0.67**	0.62	0.58	-	0.55
yeast4	0.23	0.25	**0.40**	0.27	0.26	-	0.24
yeast5	0.60	0.55	**0.74**	0.62	0.54	-	0.53
yeast6	0.19	0.34	**0.50**	0.21	0.26	-	0.05

Table 3: MCC of the undersampling algorithms. Best results in bold.

Note that, according to the AUC measure, BRIS defeats the six algorithms with which it compares, in most of the datasets under study. It is important to note that we do not codify categorical values; instead, we use the HEOM distance [Wilson and Martínez, 97] to deal with hybrid datasets. The proposed undersampling algorithm, which was able to obtain a good representation of the majority class, and therefore, very good results according to AUC.

Regarding the MCC measure, the SBC algorithm obtained the worst results, due to MCC was undefined for most of the datasets. This is due to in such datasets, SBC based classification do not obtain any True Positive (*tp*) values. Both CNNTL and CPM obtained the best results according to MCC in four datasets, and OSS in five datasets. RUS and the proposed BRIS were the best in seven datasets, and NCL was the clear winner according to MCC, with the best results in 29 datasets.

Table 4 gives the Imbalance Ratios obtained by the algorithms. We also process the results files given by KEEL software, in order to determine to what extent majority and minority classes were preserved.

To do so, we analyze the files, and compute how many times of the 220 folds (44 datasets for 5 folds each) the majority class was the negative class, and how many times the majority class was the positive one. The overall results of such analysis is shown in figure 6.

As shown in Table 4, RUS and BRIS algorithms always obtained a perfectly balanced dataset, with the same number of instances in the minority and majority classes, and therefore, their results are not shown in figure 6.

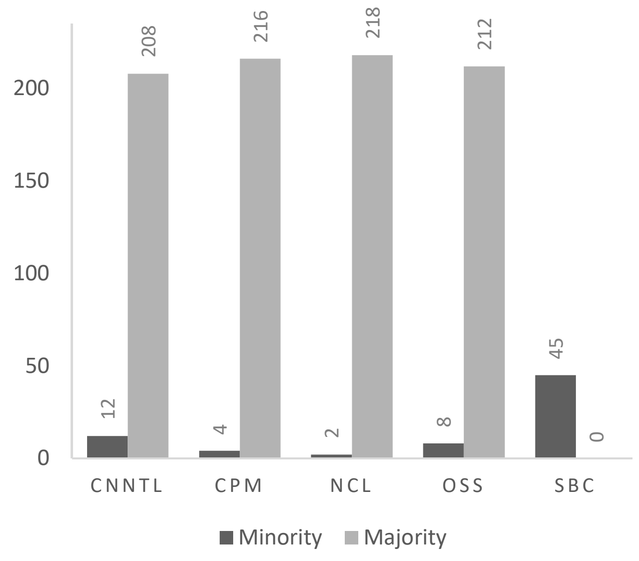

Figure 6: Folds in which the positive and negative classes were the majority ones.

Dataset	**CNNTL**	**CPM**	**NCL**	**OSS**	**RUS**	**SBC**	**BRIS**
abalone19	1.68	6.60	124.84	2.88	**1.00**	1.27	**1.00**
abalone918	1.26	2.86	13.16	1.50	**1.00**	1.72	**1.00**
ecoli0_vs_1	9.28	1.18	1.63	7.67	**1.00**	**1.00**	**1.00**
ecoli0137_vs_26	2.15	3.49	37.55	1.37	**1.00**	**1.00**	**1.00**
ecoli1	3.86	1.30	2.65	2.48	**1.00**	1.16	**1.00**
ecoli2	1.83	2.50	4.90	1.46	**1.00**	**1.00**	**1.00**
ecoli3	1.73	1.67	7.44	1.19	**1.00**	**1.00**	**1.00**
ecoli4	1.24	3.03	14.94	1.12	**1.00**	**1.00**	**1.00**
glass0	3.21	1.59	1.32	1.90	**1.00**	**1.00**	**1.00**
glass0123_vs_456	6.13	1.05	2.64	3.70	**1.00**	1.81	**1.00**
glass016_vs_2	1.49	2.73	7.38	2.15	**1.00**	**1.00**	**1.00**
glass016_vs_5	1.87	2.36	17.75	2.11	**1.00**	**1.00**	**1.00**
glass1	3.37	1.18	1.07	2.01	**1.00**	1.17	**1.00**
glass2	1.46	3.04	8.44	2.31	**1.00**	**1.00**	**1.00**
glass4	1.15	2.04	13.61	1.43	**1.00**	**1.00**	**1.00**
glass5	2.17	3.27	20.82	2.56	**1.00**	**1.00**	**1.00**
glass6	3.23	2.27	5.32	2.00	**1.00**	**1.00**	**1.00**
haberman	2.24	1.42	1.13	1.37	**1.00**	1.93	**1.00**
iris0	9.20	1.20	2.00	6.67	**1.00**	**1.00**	**1.00**
newthyroid1	3.44	1.29	4.72	2.67	**1.00**	**1.00**	**1.00**
newthyroid2	3.48	1.29	4.74	2.82	**1.00**	**1.00**	**1.00**
pageblocks0	2.72	1.46	7.90	1.88	**1.00**	1.98	**1.00**
pageblocks13_vs_4	1.38	2.09	15.07	1.48	**1.00**	**1.00**	**1.00**
pimaimb	4.89	1.21	1.27	2.45	**1.00**	2.02	**1.00**
segment0	2.95	2.64	5.95	2.58	**1.00**	**1.00**	**1.00**
shuttlec0vsc4	18.37	1.80	13.83	16.40	**1.00**	**1.00**	**1.00**
shuttlec2vsc4	1.25	1.20	18.78	1.16	**1.00**	**1.00**	**1.00**
vehicle0	2.60	1.08	2.81	2.04	**1.00**	**1.00**	**1.00**
vehicle1	2.19	1.56	1.58	1.40	**1.00**	**1.00**	**1.00**
vehicle2	1.98	1.71	2.59	1.58	**1.00**	1.67	**1.00**
vehicle3	2.23	1.51	1.68	1.47	**1.00**	**1.00**	**1.00**
vowel0	1.30	1.65	9.85	1.30	**1.00**	**1.00**	**1.00**
wisconsin	31.64	1.73	1.74	17.82	**1.00**	**1.00**	**1.00**
yeast05679_vs_4	1.26	2.13	7.61	1.25	**1.00**	**1.00**	**1.00**
yeast1	2.87	1.45	1.17	1.70	**1.00**	**1.00**	**1.00**
yeast1_vs_7	1.17	3.27	11.49	1.86	**1.00**	**1.00**	**1.00**
yeast1289_vs_7	1.59	4.51	26.77	2.36	**1.00**	**1.00**	**1.00**
yeast1458_vs_7	1.66	4.25	18.44	2.69	**1.00**	**1.00**	**1.00**
yeast2_vs_4	1.95	1.98	7.94	1.45	**1.00**	**1.00**	**1.00**
yeast2_vs_8	1.16	4.29	21.14	1.43	**1.00**	**1.00**	**1.00**
yeast3	2.08	1.84	7.06	1.29	**1.00**	**1.00**	**1.00**
yeast4	1.24	2.85	25.55	1.90	**1.00**	**1.00**	**1.00**
yeast5	1.08	1.72	31.65	1.26	**1.00**	**1.00**	**1.00**
yeast6	1.31	4.06	39.46	1.95	**1.00**	**1.00**	**1.00**

Table 4: Imbalance Ratio after applied the algorithms. Best results in bold

The remaining algorithms but SBC, usually maintain more instances of the negative (majority) class, with some exceptions in few folds, where the classes were inverted, due to the positive class contained more instances than the negative class (2 folds out of 220 for NCL, 4 for CPM, 8 for OSS and 12 for CNNTL).

However, the SBC algorithm inverted the classes in the 20% of the folds (9 datasets), although it obtained a perfectly balanced dataset in the remaining 35 datasets, but with no good majority class representation.

4.3 Statistical analysis

Despite the excellent results of the proposed algorithm, we use the Friedman test to determine the existence of significant differences in the performance of the compared algorithms. To do so, we again used the KEEL software [Alcalá-Fernández, 09; Alcalá-Fernández, 11]. The test rejects the null hypothesis for all the measures. Table 5 shows the rankings obtained by the test, considering the Area under the ROC Curve, the Matthews Correlation Coefficient, and the Imbalance Ratio measures.

After the rejection of the null hypothesis for all measures, we used the Holm post hoc test, to determine between which pair of algorithms were the significant differences in performance. Holm test was also computed by using the KEEL software [Alcalá-Fernández, 09; Alcalá-Fernández, 11].

Tables 6 - 8 give the results of the Holm tests, comparing the best ranked algorithms with respect all others, for AUC, MCC and IR, respectively.

AUC		**MCC**		**IR**	
Algorithm	**Ranking**	**Algorithm**	**Ranking**	**Algorithm**	**Ranking**
BRIS	1.4205	NCL	2.0114	RUS	1.8977
NCL	2.7045	BRIS	3.4432	BRIS	1.8977
NN	3.5568	OSS	3.4886	SBC	2.4773
OSS	4.625	RUS	3.8068	OSS	4.8068
CPM	4.7273	CPM	4.1364	CPM	5.1591
RUS	5.3295	CNNTL	4.3977	CNNTL	5.4432
CNNTL	5.8523	SBC	6.7159	NCL	6.3182
SBC	7.7841				

Table 5: Friedman rankings the undersampling algorithms

I	Algorithm	Z	p	Holm
7	SBC	12.185436	0.000000	0.007143
6	CNNTL	8.486286	0.000000	0.008333
5	RUS	7.485339	0.000000	0.010000
4	CPM	6.332075	0.000000	0.012500
3	OSS	6.136237	0.000000	0.016667
2	NN	4.090825	0.000043	0.025000
1	NCL	2.458847	0.013938	0.050000

Table 6: Results of the Holm test, comparing the best ranked algorithm with others, according to AUC. The hypothesis with $p \leq 0.05$ are rejected

The Holm test confirm that BRIS had a significantly better performance than all the compared undersampling algorithms, and also better than the original Nearest Neighbor classifier, according to AUC. However, NCL obtained best results for the MCC measure.

I	**Algorithm**	**Z**	**p**	**Holm**
6	SBC	10.2147	0	0.008333
5	CNNTL	5.18137	0	0.01
4	CPM	4.613886	0.000004	0.0125
3	RUS	3.898364	0.000097	0.016667
2	OSS	3.207515	0.001339	0.025
1	BRIS	3.108822	0.001878	0.05

Table 7: Results of the Holm test, comparing the best ranked algorithm with others, according to MCC. The hypothesis with $p \leq 0.05$ *are rejected*

I	**Algorithm**	**Z**	**p**	**Holm**
6	NCL	9.597870	0.000000	0.008333
5	CNNTL	7.698035	0.000000	0.010000
4	CPM	7.081205	0.000000	0.012500
3	OSS	6.316336	0.000000	0.016667
2	SBC	1.258333	0.208271	0.025000
1	BRIS	0.000000	1.000000	0.050000

Table 8: Results of the holm test, comparing the best ranked algorithm with others, according to IR. The hypothesis with $p \leq 0.05$ *are rejected*

With respect IR, BRIS outperforms OSS, CPM, CNNTL and NCL, and has no significant differences with RUS and SBC.

4.4 Discussion

As shown, the proposed method significantly improves the performance of the Nearest Neighbor classifier using imbalanced datasets. These results show that it is possible to increase the efficiency of the classification, by performing undersampling strategies for classes balancing. With respect to the other compared algorithms, BRIS had a very good performance, significantly surpassing all other algorithms according to the Area under the ROC Curve, and been the second best according to Matthews Correlation Coefficient. It is noteworthy that the BRIS achieves a perfect balance of the classes, since it selects from the majority class, as many instances as the minority class has, resulting in the imbalance ratio of equal to one in all cases.

5 Conclusions and Future Work

The experimental results clearly show that BRIS, the proposed undersampling algorithm, enriches the state of the art in the treatment of class imbalance. On the one hand, by applying BRIS to each of the 44 imbalanced datasets, the performance of the Nearest Neighbor classifier (the base classifier) is significantly improved, according to AUC, which is the measure of performance chosen. And on the other hand, BRIS overwhelmingly surpasses the performance obtained with the NN classifier, after applying the corresponding undersampling algorithm (CNNTL, CPM, NCL, OSS, RUS, and SBC) to the datasets of our experimental study. BRIs was also the second best algorithm according to the MCC measure.

In addition, it is noteworthy that the BRIS achieves a perfect balance of the classes; and the only one of the six comparison undersampling algorithms that achieves something similar is RUS, while the others do not achieve the perfect balance, reaching the point that the SBC algorithm inverted the classes in the 20% of the folds (9 datasets).

The success of BRIS is due, in large part, to the proposal of an original measure to estimate the importance of objects; and also, that the conceptual basis on which this new measure rests are Maximal Similarity Granular Rough Sets and Compact Set Structuralization, which are able to deal with hybrid data, and are designed to preserve the structure of the data.

As future work, we propose the development of an oversampling procedure for data balancing.

Acknowledgements

The authors would like to thank the Instituto Politécnico Nacional (Secretaría Académica, COFAA, SIP, CIDETEC, and CIC), the CONACyT, and SNI for their economic support to develop this work.

References

[Alcalá-Fernández et al., 09] Alcalá-Fernández, J., Sánchez, L., Garcia, S., del Jesus, M. J., Ventura, S., Garrell, J. M., . . . Rivas, V. M. “KEEL: a software tool to assess evolutionary algorithms for data mining problems”, Soft Computing, 13(3) (2009), 307-318.

[Alcalá-Fernández et al., 11] Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. “Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework”, Journal of Multiple-Valued Logic & Soft Computing, 17 (2011), 255-287.

[Bagby et al., 19] Bagby, S. P., Martin, D., Chung, S. T., & Rajapakse, N. “From the Outside In: Biological Mechanisms Linking Social and Environmental Exposures to Chronic Disease and to Health Disparities”, American journal of public health, 109(S1) (2019), S56-S63.

[Batista et al., 04] Batista, G. E., Prati, R. C., & Monard, M. C. “A study of the behavior of several methods for balancing machine learning training data”, ACM SIGKDD explorations newsletter, 6(1) (2004), 20-29.

[Carbonneau et al., 16] Carbonneau, M.-A., Granger, E., Raymond, A. J., & Gagnon, G. “Robust multiple-instance learning ensembles using random subspace instance selection”, Pattern Recognition, 58 (2016), 83-99.

[Castellanos et al, 18] Castellanos, F. J., Valero-Mas, J. J., Calvo-Zaragoza, J., & Rico-Juan, J. R. “Oversampling imbalanced data in the string space”, Pattern Recognition Letters, 103 (2018), 32-38.

[Chawla et al., 02] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. “SMOTE: synthetic minority over-sampling technique”, Journal of artificial intelligence research, 16 (2002), 321-357.

[Cheng et al., 12] Cheng, K.-O., Law, N.-F., & Siu, W.-C. “Iterative bicluster-based least square framework for estimation of missing values in microarray gene expression data”, Pattern Recognition, 45(4) (2012), 1281-1289.

[Cleofas-Sánchez et al., 16] Cleofas-Sánchez, L., García, V., Marqués, A., & Sánchez, J. S. “Financial distress prediction using the hybrid associative memory with translation”, Applied Soft Computing, 44 (2016), 144-152.

[Cover and Hart, 67] Cover, T. M., & Hart, P. E. “Nearest neighbor pattern classification”, IEEE transactions on information theory, 13(1) (1967), 21-27.

[Ester et al., 96] Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. “A density-based algorithm for discovering clusters in large spatial databases with noise”, Second International Conference on Knowledge Discovery and Data Mining, 34 (1996), 226-231

[Fernández et al., 18] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. Learning from imbalanced data sets: Springer (2018).

[Fotouthi et al., 19] Fotouhi, S., Asadi, S., & Kattan, M. W. “A comprehensive data level analysis for cancer diagnosis on imbalanced data”, Journal of biomedical informatics, 90 (2019), 103089.

[Friedman, 37] Friedman, M. “The use of ranks to avoid the assumption of normality implicit in the analysis of variance”, Journal of the american statistical association, 32(200) (1937), 675-701.

[Friedman, 40] Friedman, M. “A comparison of alternative tests of significance for the problem of m rankings”, The Annals of Mathematical Statistics, 11(1) (1940), 86-92.

[Galar et al., 13] Galar, M., Fernández, A., Barrenechea, E., & Herrera, F. “ EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling”, Pattern Recognition, 46(12) (2013), 3460-3471.

[Garcia and Herrera, 08] Garcia, S., & Herrera, F. “An extension on “statistical comparisons of classifiers over multiple data sets” for all pairwise comparisons”, Journal of Machine Learning Research, 9(Dec) (2008), 2677-2694.

[García et al., 19] García, V., Marqués, A. I., & Sánchez, J. S. “Exploring the synergetic effects of sample types on the performance of ensembles for credit risk and corporate bankruptcy prediction”, Information Fusion, 47 (2019), 88-101.

[Hart, 68] Hart, P. "The condensed nearest neighbour rule", IEEE transactions on information theory, 14(3) (1968), 515-516.

[Holm, 79] Holm, S. "A simple sequentially rejective multiple test procedure", Scandinavian journal of statistics, 6(2) (1979), 65-70.

[Hu et al., 06] Hu, Q., Yu, D., & Xie, Z. "Information-preserving hybrid data reduction based on fuzzy-rough techniques", Pattern Recognition Letters, 27(5) (2006), 414-423.

[Kim and Hong, 17] Kim, K., & Hong, J.-s. "A hybrid decision tree algorithm for mixed numeric and categorical data in regression analysis", Pattern Recognition Letters, 98 (2017), 39-45.

[Kubat and Matwin, 97] Kubat, M., & Matwin, S. "Addressing the curse of imbalanced training sets: one-sided selection", ICML (1997), 179-186

[Laurikkala, 01] Laurikkala, J. "Improving identification of difficult small classes by balancing class distribution", Conference on Artificial Intelligence in Medicine in Europe.AIME (2001), DOI:10.1007/3-540-48229-6_9

[Leyva et al., 15] Leyva, E., González, A., & Pérez, R. "Three new instance selection methods based on local sets: A comparative study with several approaches from a bi-objective perspective", Pattern Recognition, 48(4) (2015), 1523-1537.

[Li et al., 14] Li, Z., Geng, G.-H., Feng, J., Peng, J.-y., Wen, C., & Liang, J.-l. "Multiple instance learning based on positive instance selection and bag structure construction", Pattern Recognition Letters, 40, (2014) 19-26.

[Li et al., 16] Li, Y., Wu, B., Ghanem, B., Zhao, Y., Yao, H., & Ji, Q. "Facial action unit recognition under incomplete data based on multi-label learning with missing labels", Pattern Recognition, 60 (2016), 890-900.

[López et al., 12] López, V., Fernández, A., Moreno-Torres, J. G., & Herrera, F. "Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics", Expert Systems with Applications, 39(7) (2012), 6585-6608.

[Matthews, 75] Matthews, B. W. "Comparison of the predicted and observed secondary structure of T4 phage lysozyme", Biochimica et Biophysica Acta (BBA) - Protein Structure 405(2) (1975), 442–451

[Rajesh and Dhuli, 18] Rajesh, K. N., & Dhuli, R. "Classification of imbalanced ECG beats using re-sampling techniques and AdaBoost ensemble classifier", Biomedical Signal Processing and Control, 41 (2018), 242-254.

[Režnáková et al., 17] Režnáková, M., Tencer, L., Plamondon, R., & Cheriet, M. "Forgetting of unused classes in missing data environment using automatically generated data: application to on-line handwritten gesture command recognition" Pattern Recognition, 72 (2017), 355-367.

[Ruiz-Shulcloper, 08] Ruiz-Shulcloper, J. "Pattern recognition with mixed and incomplete data", Pattern Recognition and Image Analysis, 18(4) (2008), 563-576.

[Sáez et al., 16] Sáez, J. A., Krawczyk, B., & Woźniak, M. “Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets”, Pattern Recognition, 57 (2016), 164-178.

[Solokova and Lapalmme, 09] Sokolova, M., & Lapalme, G. “A systematic analysis of performance measures for classification tasks”, Information Processing & Management, 45(4) (2009), 427-437.

[Tomez, 76] Tomek, I. “Two modifications of CNN”, IEEE Transactions on Systems, Man and Cybernetics, 6 (1976), 769-772.

[Trinidad et al., 00] Trinidad, J. F. M. n., Shulcloper, J. R., & Cortés, M. S. L. “Structuralization of universes”, Fuzzy sets and systems, 112(3) (2000), 485-500.

[Villuendas-Rey, 19] Villuendas-Rey, Y. “Maximal similarity granular rough sets for mixed and incomplete information systems”, Soft Computing, 23(13) (2019), 4617-4631

[Wilson and Martinez, 97] Wilson, D. R., & Martinez, T. R. “Improved heterogeneous distance functions”, Journal of artificial intelligence research, 6 (1997), 1-34.

[Wilson, 72] Wilson, D. L. “Asymptotic properties of nearest neighbor rules using edited data”, IEEE Transactions on Systems, Man, and Cybernetics, (3) (1972), 408-421.

[Yen and Lee, 06] Yen, S.-J., & Lee, Y.-S. “Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset”, Intelligent Control and Automation LNCIS 344 (2006), 731-740.

[Yoon and Kwek, 05] Yoon, K., & Kwek, S. “An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics”, Fifth International Conference on Hybrid Intelligent Systems (HIS'05). DOI: 10.1109/ICHIS.2005.23

[Yu et al., 18] Yu, L., Zhou, R., Tang, L., & Chen, R. “A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data”, Applied Soft Computing, 69 (2018), 192-202.