



Article

A New Under-Sampling Method to Face Class Overlap and Imbalance

Angélica Guzmán-Ponce ^{1,†}, Rosa María Valdovinos ^{1,†} , José Salvador Sánchez ^{2,†}  and José Raymundo Marcial-Romero ^{1,*,†}

¹ Facultad de Ingeniería, Universidad Autónoma del Estado de Mexico, Cerro de Coatepec s/n, Ciudad Universitaria, Toluca 50100, Mexico; angelicagp1416@hotmail.com (A.G.-P.); rvaldovinosr@uaemex.mx (R.M.V.)

² Department of Computer Languages and Systems, Institute of New Imaging Technologies, Universitat Jaume I, 12071 Castelló de la Plana, Spain; sanchez@uji.es

* Correspondence: jrmarcialr@uaemex.mx

† These authors contributed equally to this work.

Received: 7 May 2020; Accepted: 1 June 2020; Published: 27 July 2020



Abstract: Class overlap and class imbalance are two data complexities that challenge the design of effective classifiers in Pattern Recognition and Data Mining as they may cause a significant loss in performance. Several solutions have been proposed to face both data difficulties, but most of these approaches tackle each problem separately. In this paper, we propose a two-stage under-sampling technique that combines the DBSCAN clustering algorithm to remove noisy samples and clean the decision boundary with a minimum spanning tree algorithm to face the class imbalance, thus handling class overlap and imbalance simultaneously with the aim of improving the performance of classifiers. An extensive experimental study shows a significantly better behavior of the new algorithm as compared to 12 state-of-the-art under-sampling methods using three standard classification models (nearest neighbor rule, J48 decision tree, and support vector machine with a linear kernel) on both real-life and synthetic databases.

Keywords: class imbalance; class overlap; under-sampling; clustering; DBSCAN; minimum spanning tree

1. Introduction

The class imbalance problem is a challenging situation common to many real-world applications such as fraud detection [1], fault/failure diagnosis [2], face recognition [3], text classification [4], sentiment analysis [5], and credit risk prediction [6], among others. A binary data set is said to be imbalanced when one of the classes (the minority or positive class, C^+) has a significantly lower number of instances in comparison to the other class (the majority or negative class, C^-) [7]. The disproportion between the number of positive and negative instances leads to a bias towards the majority class that may imply an important deterioration of the classification performance on the minority class.

Many authors have asserted that the class imbalance distribution by itself does not represent a critical problem for classification, but when it is associated with other data complexity factors, then it can significantly decrease the classification performance because traditional classifiers tend to err on many positive instances [8]. García et al. [9] viewed that the combination of class imbalance and highly overlapping class distributions results in a significant performance deterioration of instance-based classifiers. Class overlap refers to ambiguous regions of the feature space where the prior probability of

the classes are approximately equal (i.e., all classes have a similar amount of data), which makes most classifiers more likely to produce a large number of misclassifications [10].

Class noise, which refers to mislabeled instances, can also lead to incorrect classification decisions, especially when combined with class imbalance [11]. The presence of both noise and class imbalance can significantly affect the classification process: mislabeling from the minority class to the majority class increases the imbalance ratio in the data set, whereas mislabeling from the majority class to the minority class produces an even higher difficulty to correctly classify the positive instances [12]. Both class overlap and noise have been commonly tackled through the application of data filtering methods, which are intended to identify and remove mislabeled and atypical data and also to clean possible overlapping between classes [13,14].

The different strategies to address the class imbalance problem are usually categorized into four groups [7,15]:

1. Algorithmic-level methods. These consist in internally biasing the discrimination-based process so as to compensate for the class imbalance.
2. Data-level methods. These perform some sort of data preprocessing with the aim of reducing the imbalance ratio.
3. Cost-sensitive methods. These incorporate distinct misclassification costs into the classification process and assign higher misclassification costs to the errors on the minority class.
4. Ensemble-based techniques. These consist of a combination between an ensemble algorithm and either the data-level or the cost-sensitive approaches. In the first one, data preprocessing is performed before training the classifier, whereas in the second one, the low cost guides the use of the ensemble algorithm.

Conclusions about what is the best solution for the class imbalance problem are divergent, but the data-level methods are likely the most investigated because they are not classifier-dependent, do not need any algorithmic adaptation to the data sets, and can be easily implemented for any problem [16]. Thus, focusing on the data-level approach, resampling methods consist of adjusting the data set size with the aim of balancing the class distribution, either by decreasing the number of majority class instances (under-sampling) or increasing the number of minority class instances (over-sampling). A hybrid resampling approach consists of combining both over-sampling and under-sampling strategies. As remarked by several authors, under-sampling methods may throw away many potentially useful data as a result of ignoring most of majority class, whereas the over-sampling methods can lead to overfitting on the minority class and an increase in complexity and execution time [17]. On the other hand, it is worth pointing out that the resampling methods have been also used to face the imbalance problem together with class overlap and/or the presence of noisy and borderline instances in the data set [8,18–21], a common situation that has motivated the research presented in this paper.

Although most traditional resampling methods are based on the k nearest neighbor (k NN) rule [22–24], clustering algorithms have been demonstrated to be even a more powerful strategy for addressing the class imbalance problem [16,25,26] because they may reduce the risk of removing potentially useful data and, therefore, they can achieve better results than the k NN-based resampling methods. On the other hand, despite the fact that over-sampling has been shown to be apparently superior to under-sampling [27,28], the under-sampling approach can be especially useful when dealing with big data applications [29] since these algorithms will lead to a reduction of the data set size and also a decrease in the computational requirements of the subsequent classification process. Bearing both these issues in mind, the present paper introduces a new under-sampling technique (here called DBMIST-US) to face not only the class imbalance but also the overlapping between classes. It deals with the noisy instances in the majority class via a noise filter based on the DBSCAN clustering algorithm [30] combined with the use of a minimum spanning tree

(MST) algorithm to reduce the size of the negative class. The reason to combine the DBSCAN clustering with the MST approach is because DBSCAN has demonstrated to be a powerful tool for identifying and removing noisy instances and cleaning the overlapping between classes [31], but it does not produce a well-balanced class distribution. By viewing the data set as a weighted complete graph, the MST algorithm allows for discovering the core of the majority class, which is further used to remove the amount of redundant negative instances needed to balance both classes.

Hereafter, the remainder of this paper is organized as follows. Section 2 introduces several well-known methods to tackle the class imbalance problem. Details of the DBMIST-US algorithm here proposed are described in Section 3. The experimental setup is reported in Section 4. In Section 5, the results are analyzed and discussed. Finally, Section 6 presents the main conclusions and outlines some open lines to be addressed in the future.

2. Resampling Algorithms to Face Class Imbalance

In this section, a collection of both k NN-based and clustering-based algorithms for dealing with the class imbalance problem is briefly described.

2.1. Neighborhood-Based Algorithms

Most k NN-based under-sampling algorithms are inherited from the literature on prototype selection. These techniques exploit the analysis of the feature space, either by removing instances far from the decision boundary of two classes to reduce redundancy (condensing or thinning) or removing those close to the boundary for generalization (editing, cleaning, or filtering) [32]. Wilson's editing [33] and Hart's condensing [22] are the most representative examples of these families of prototype selection methods.

Wilson's editing (ENN) removes all instances that are misclassified by the k NN rule. The Hart's condensing (CNN) algorithm uses the concept of consistent subset to eliminate the majority class instances that are sufficiently far away from the decision boundary because these are considered to be irrelevant for learning. Analogously, the Tomek links (TL) [23,34] have also been employed to remove the majority class instances since, if two instances form a Tomek link, then either one of these is noise or both instances are borderline.

The one-sided selection (OSS) technique developed by Kubat and Matwin [35] eliminates only the majority class instances that are redundant or noisy (i.e., those that are bordering the minority class). The concept of Tomek links is used to discover the border instances, whereas the Hart's condensing algorithm is applied to discover the redundant cases (i.e., those that are far enough from the decision boundary).

Laurikkala [24] proposed the neighborhood cleaning rule (NCL) algorithm, which is quite similar to the OSS technique. With NCL, the majority class instances whose class label differs from the class of at least two of their three nearest neighbors are removed by means of Wilson's editing [33]. In addition, this algorithm also discards the neighbors that belong to the majority class if a minority class instance is misclassified by its three nearest neighbors.

2.2. Clustering-Based Algorithms

Clustering-based under-sampling methods have become a well-grounded alternative to neighborhood-based algorithms for handling class imbalance because they can mitigate the problem of information loss caused by the removal of negative instances [16].

Yen and Lee introduced a cluster-based under-sampling algorithm named SBC (under-sampling based on clustering) [17], which rests upon the idea that there may exist various clusters in a data set and each cluster i may have a different ratio of majority class instances to minority class instances in the cluster

i. Thus, all instances are initially grouped into K clusters; then, a number of negative instances will be randomly selected from each cluster according to the ratio of majority class instances to minority class instances. Finally, it combines the selected negative instances from the K clusters with all the instances in the minority class. A similar under-sampling method corresponds to the algorithm proposed by Longadge et al. [36], which firstly clusters the majority class instances into K groups using the K -means algorithm and then selects $|C^+| \times IR_i$ majority class instances from each cluster i , where IR_i denotes the imbalance ratio in the cluster i . Note that the aim of this method is not to obtain a perfectly balanced class distribution, but to reduce the disproportion between the size of the majority and minority classes.

The ClusterOSS algorithm introduced by Barella et al. [37] is an improvement of the OSS method. It starts by using the K -means algorithm to cluster the majority class instances. Then, the closest instances to the center of each cluster are used to start the application of OSS, removing borderline and noisy negative instances using the Tomek links. Unlike OSS, this technique does not start the under-sampling process at random, but defines how many and which instances should be chosen by applying the K -means clustering.

Sowah et al. proposed an under-sampling technique named CUST [38] whose main objective is to remove redundant and noisy instances along with outliers from the majority class. In a first stage, the algorithm removes noisy negative instances using a techniques based on Tomek links. In a second stage, outliers and redundant negative instances are eliminated by using the K -means algorithm to cluster the remaining majority class instances.

Das et al. introduced the ClusBUS algorithm [39], which aims at discarding negative instances that lie in class overlapping regions. First, it clusters the entire data set into K clusters using the DBSCAN algorithm. Then, for those clusters that contain both positive and negative instances, the algorithm removes a number of majority class instances necessary to create a vacuum around the minority class instances. Tsai et al. presented the CBIS technique [40] based on clustering analysis and instance selection: the affinity propagation algorithm groups similar negative instances into K clusters, and then an instance selection process filters out instances in each cluster. Finally, all reduced K clusters together with the instances of minority class form a balanced data set.

Ng et al. proposed the diversified sensitivity-based under-sampling (DSUS) method [41]. It selects useful instances yielding high stochastic sensitivity with respect to the currently trained classifier. To guarantee that only instances close to the decision boundary are selected and preserve the data distribution, negative instances are clustered and only a representative instance of each cluster participates in the selection process

Lin et al. developed two under-sampling strategies in which the K -means clustering technique is used [26]. Unlike the algorithms described previously, the number of clusters with negative instances is here defined to be equal to the size of the minority class. The first strategy uses the cluster centers to represent the majority class, whereas the second strategy employs the nearest neighbors of the centers. In the Fast-CBUS method introduced by Ofek et al. [16], the minority class is clustered into K clusters by the K -means algorithm and, for each cluster, a similar number of majority class instances close to the minority class instances are sampled.

It is also worth pointing out that some hybrid algorithms combine clustering with other techniques. For instance, the clustering-based balancing (ClusterBal) method proposed by Sun et al. [42] combines clustering with classifier ensembles: it divides the majority class instances into K subsets by clustering, and then all the minority instances are added to each subset to train a focused classifier. On the other hand, the cluster-based evolutionary under-sampling (CBEUS) [43] combines the K -means clustering applied to the majority class with a genetic algorithm.

3. The DBMIST-US Algorithm

In this section, we describe the new algorithm DBMIST-US for handling class overlap and imbalance by under-sampling the data set. The purpose is to obtain a subset of representative instances from the majority class and avoid the risk of removing instances that may contribute to generate knowledge. The overall process of DBMIST-US is described in Algorithm 1, where IR_{max} is the unique free parameter representing the maximum imbalance ratio allowed. It starts by dividing a two-class imbalanced data set of n instances into a subset C^- with the majority class instances and a subset C^+ containing the minority class instances. After this initial step, the algorithm consists of two stages:

1. Cleaning stage (lines 3–7). The DBSCAN clustering method [30] is applied to the set with the negative instances to produce a noise-free subset.
2. Core stage (line 8). An MST is built to get a core representation from the majority class. The result is a subset of the majority class with less dispersion than the set obtained in the previous stage.

Algorithm 1 DBMIST-US

Input: $DS = \{p_1, p_2, \dots, p_n\}, IR_{max}$

Output: DS'

- 1: Split DS into two subsets C^- and C^+ .
 - 2: $C'^- \leftarrow C^-$
 - 3: **repeat**
 - 4: Compute ϵ and $minPts$.
 - 5: $C'^- \leftarrow DBSCAN(C'^-, \epsilon, minPts)$.
 - 6: **until** ϵ and $minPts$ do not change
 - 7: $DS' \leftarrow C^+ \cup C'^-$
 - 8: $C''- \leftarrow MSTGraph(DS', IR_{max})$
 - 9: $DS' \leftarrow C^+ \cup C''-$
-

DBSCAN is a clustering algorithm that assumes that the clusters correspond to dense regions in the feature space. It can discover groups without any prior knowledge about the number of clusters in the data, and it works pretty well with noisy data sets. DBSCAN requires two input parameters: ϵ defines the radius of the neighborhood region (i.e., how close instances should be to each other to be considered a part of the same cluster) and $minPts$ establishes the minimum number of instances that might be part of the ϵ -neighborhood to form a cluster. Unlike the original proposal [44] where the input values of ϵ and $minPts$ are computed by clustering the data set with the K -means algorithm, our DBSCAN formulation only takes care of $|C^-|$ and $|C^+|$ to compute ϵ and $minPts$, respectively.

DBSCAN begins by randomly selecting a negative instance p_i^- from the current set C'^- . If the analyzed instance p_i^- does not have at least $minPts$ neighboring instances at a distance ϵ , it is considered as noisy and removed from the set C'^- . Note that the rationale behind this is to remove those instances that are close enough to the decision boundary. This process is repeated until $minPts$ and ϵ do not change. After applying the DBSCAN algorithm on C'^- , an MST is computed by the process $MSTGraph$ described in Algorithm 2.

Algorithm 2 MSTGraph

Input: DS', IR_{max}

Output: $C'^- \subset C^-$

- 1: Split DS' into two subsets C^- and C^+ .
 - 2: Build $G_w = (V, E)$ a weighted complete graph from C^- .
 - 3: $S \leftarrow |C^-|$
 - 4: $IR \leftarrow \frac{S}{|C^+|}$
 - 5: $C'^- \leftarrow \bigcap \frac{S}{|C^+|}$
 - 6: $M_G \leftarrow \text{incidenceMatrix}(G_w)$
 - 7: $MST \leftarrow \text{GetMST}(G_w, M_G)$
 - 8: **while** $IR > IR_{max}$ **do**
 - 9: $S \leftarrow \frac{S \cdot \sigma^2 Z^2}{\sigma^2(|C^-|-1) + \sigma^2 Z^2}$
 - 10: $IR \leftarrow \frac{S}{|C^+|}$
 - 11: **end while**
 - 12: **for all** $\{v, u\}$ in $E(MST)$ **do**
 - 13: $C'^- \cup u$
 - 14: $C'^- \cup v$
 - 15: **if** $|C'^-| > S$ **then**
 - 16: **return** C'^-
 - 17: **end if**
 - 18: **end for**
-

3.1. Core Stage

After cleaning the majority class with DBSCAN, a graph-based approach is used to compute a candidate core of the set C'^- . A weighted complete graph $G_w = (V(G), E(G))$ is built from the set of majority class instances C'^- given by the DBSCAN-based stage as follows:

- $V(G) = \{i \mid p_i \in C'^-\}$ a set of vertices.
- $E(G) = \{\{v, u\} \mid v, u \in V(G)\}$ a set of edges.
- $\forall e = \{v, u\} \in E(G), w(e) = d(p_v, p_u)$ where $d(p_v, p_u)$ is the Euclidean distance between v and u .

The purpose of computing an MST is to build a subgraph that connects all vertices in G_w without forming cycles, that is, there is not a path from a vertex x to a vertex y such that $x = y$ and whose sum of edge weights is the smallest. Figure 1 shows an illustrative example of an MST (b) computed from a weighted graph (a).

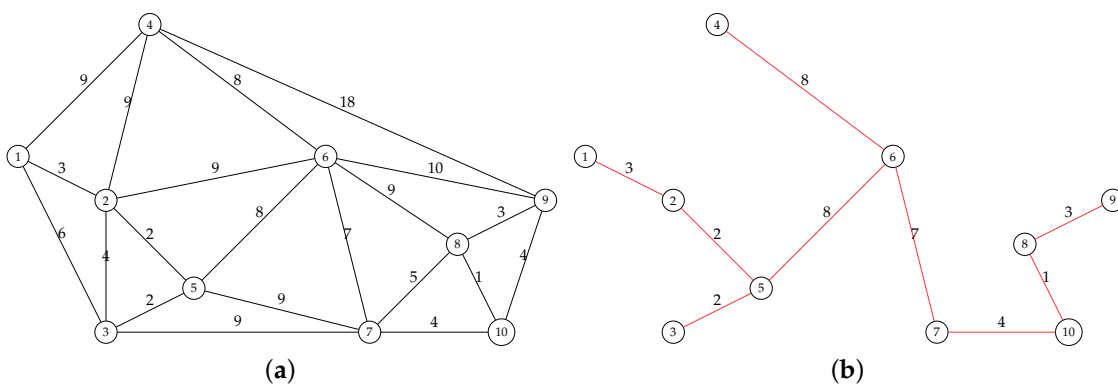


Figure 1. Illustration of weighted graphs. The graph on the right depicts an MST of the graph on the left. (a) weighted graph; (b) MST.

To build an MST, an incidence matrix M_G (line 6) computed by means of Prim's algorithm [45] is used. This process is as follows:

1. Choose an initial vertex v at random.
2. Select the edge $e = \{v, u\}$ with the lowest weight in M_G and mark v as visited. Now, the next vertex to be analyzed is u .
3. Repeat Step 2 now taking u as the initial vertex of the edge e and while there exists a vertex z such that z has not been visited yet, $e = \{u, z\}$.
4. The MST will be built doing backtracking on the already marked vertices until each vertex has been marked.

By definition, an MST contains each vertex of the graph. Our proposal takes only a subset of the MST: the first S -vertices to build the new majority class C'^- , where S denotes the maximum imbalance ratio required (lines 8–10). To validate the 95% of confidence in the imbalance ratio, the values chosen are $Z = 1.96$, $\sigma = 0.5$, and $e = 0.05$ according to the procedure given by Torres et al. [46].

3.2. Time Complexity

The time complexity of an algorithm is estimated by counting the number of operations performed [47]. Thus, the time complexity of the DBMIST-US algorithm depends on its two stages: the run time complexity of DBSCAN is $\mathcal{O}(n^2)$ in the worst-case [48] and the time complexity of MSTGraph is also $\mathcal{O}(n^2)$, which depends on the time complexity of the following processes:

- The computation of the incidence matrix from G_w takes n^2 steps.
- The computation of an MST based on the Prim's algorithm takes n^2 steps.

Therefore, as the three procedures are independent, the complexity of DBMIST-US is given by $\mathcal{O}(n^2)$ in the worst-case.

3.3. Differences between DBMIST-US and Related Works

The algorithm proposed in this work differs from other related methods in several aspects. First, most developments focus on handling either class imbalance or class overlap separately, while DBMIST-US faces both intrinsic data complexities together. Although one can find a few works addressing both problems, in general, they consist of the application of different techniques in a sequential manner. For instance, Chen et al. designed a methodology based on the use of the NCL algorithm to remove the overlapping majority class instances followed by an ensemble to randomly under-sample the majority class [49]. Similarly, Koziarski and Woźniak proposed an energy-based approach to clean the neighborhoods of minority class instances combined with an over-sampling procedure [50]. Other methods have been also proposed to handle imbalanced data sets with noisy and borderline instances together, but they are mostly based on modifying some previous resampling algorithm; in this line, for instance, Sáez et al. proposed an extension of the well-known SMOTE over-sampling algorithm through the inclusion of an iterative ensemble-based noise filter [8].

Regarding the differences between DBMIST-US and the existing clustering-based methods, it is worth pointing out that most under-sampling techniques rely upon the K -means and the fuzzy K -means algorithms [16,26,36–38,40,41,43,51]. However, it is well-known that K -means may not be sufficiently effective when applied to imbalanced data because it always generates clusters with similar sizes [52]. Another weakness of these clustering algorithms is that they assume some prior knowledge of the number of clusters to face the imbalance problem. To overcome these shortcomings, our proposal employs the DBSCAN algorithm in the filtering stage. Some of the advantages of using DBSCAN are: (i) it can identify

arbitrarily shaped clusters and detect noise, and (ii) the number of clusters is not a user-specified parameter. In addition, an MST-based approach is used to under-sample the majority class, which, to the best of our knowledge, this is the first development using the MST to face the imbalance problem. A related approach employs the Gabriel graph and the relative neighborhood graph to search for neighbors in SMOTE [53], which are supergraphs of the Euclidean MST.

Finally, differences between DBMIST-US and other resampling methods that use DBSCAN can also be remarked. Sanguanmak and Hanskunatai proposed the hybrid DBSM algorithm, which employs DBSCAN for under-sampling and the SMOTE technique for over-sampling[54]; here, DBSCAN is used to create clusters from the whole training set, and then 50% of the majority class instances are eliminated from each cluster. Another approach is the DBSMOTE algorithm [55], which integrates DBSCAN and SMOTE; DBSCAN is applied to discover arbitrarily shaped clusters and SMOTE generates more synthetic positive instances around the core of the minority class rather than at the border. The main difference between our proposal and both these methods is that, unlike DBMIST-US, these are for over-sampling. On the other hand, the DBMUTE technique combines DBSCAN with the under-sampling MUTE algorithm [56] to discover and remove majority class instances from the overlapping region [57], but this is not intended to identify and remove noisy instances.

4. Experimental Set-Up

Extensive experiments were conducted to evaluate the usefulness of the DBMIST-US method when compared to a pool of state-of-the-art under-sampling techniques. In summary, the research questions to investigate in this experimental study were:

- Q1. What is the classification performance of DBMIST-US in comparison to several state-of-the-art under-sampling algorithms?
- Q2. How robust is DBMIST-US across the use of different classification models?
- Q3. What is the impact of each under-sampling algorithm on the imbalance ratio?

4.1. Data Sets

We conducted the experiments on 20 real-life data sets (Table 1) and 12 synthetic data sets taken from the KEEL Data Set Repository (<https://sci2s.ugr.es/keel/imbanced.php#subA>). All data sets have two classes and various levels of imbalance (the imbalance ratio ranges from 9.35 to 82).

The experiments on the synthetic data were carried out on three databases with different shapes of the minority class (subclus, clover, and paw) whose instances are randomly and uniformly distributed in a two-dimensional feature space. In all cases, the positive instances are uniformly surrounded by the majority class. Each database comprises 800 instances with an imbalance ratio of 7 and different levels of random noise (0%, 30%, 50%, and 70%) on both classes: data from both the majority and minority classes were randomly mislabeled and some single feature-values were randomly changed.

In subclus, the positive instances are located inside rectangles that form small disjuncts. Clover represents a more complex, nonlinear problem, where the minority class resembles a flower with elliptic petals. In a paw database, the minority class is decomposed into three elliptic sub-regions of varying cardinalities, where two sub-regions are located close to each other, and the remaining smaller sub-region is separated. Figure 2 illustrates two examples of these data sets: 0% and 70% of noise.

Table 1. Summary of the real-life data sets.

	Data Set	Class Distribution	#Instances	IR
1	yeast-0-5-6-7-9_vs_4	51–477	528	9.35
2	glass-0-1-6_vs_2	17–175	192	10.29
3	glass2	17–197	214	11.59
4	shuttle-c0-vs-c4	123–1706	1829	13.87
5	yeast-1_vs_7	30–429	459	14.30
6	glass4	13–201	214	15.47
7	ecoli4	20–316	336	15.80
8	page-blocks-1-3_vs_4	28–444	472	15.86
9	glass-0-1-6_vs_5	9–175	184	19.44
10	yeast-1-4-5-8_vs_7	30–663	693	22.10
11	glass5	9–205	214	22.78
12	yeast-2_vs_8	20–462	482	23.10
13	flare-F	43–1023	1066	23.79
14	yeast4	51–1433	1484	28.10
15	yeast-1-2-8-9_vs_7	30–917	947	30.57
16	yeast5	44–1440	1484	32.73
17	ecoli-0-1-3-7_vs_2-6	7–274	281	39.14
18	abalone-17_vs_7-8-9-10	58–2280	2338	39.31
19	yeast6	35–1449	1484	41.40
20	poker-8-9_vs_5	25–2050	2075	82.00

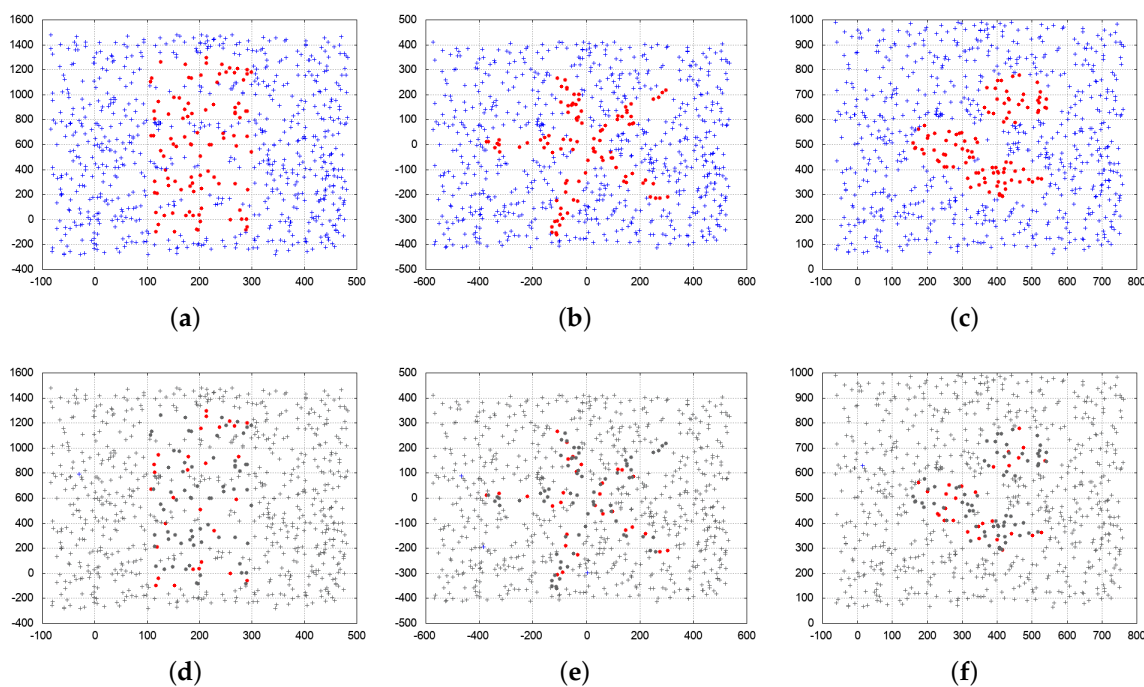


Figure 2. Scatter plots of the synthetic data sets with 0% and 70% of noise (points in gray color are for noisy data). (a) subclus-0; (b) clover-0; (c) paw-0; (d) subclus-70; (e) clover-70; (f) paw-70.

4.2. Reference Under-Sampling Algorithms and Classifiers

Besides the DBMIST-US method with a maximum imbalanced ratio set to 2, we also implemented some state-of-the-art under-sampling algorithms already described in Section 2: CNN, ENN, TL, NCL, OSS, SBC, and ClusterOSS. In addition, the experiments also included a collection of methods that are popular in the literature related to class imbalance (see Table 2).

Table 2. Other under-sampling methods included in the experiments.

Method	Description
Random under-sampling (RUS)	It balances the data set through the random elimination of instances that belong to the over-sized class.
Evolutionary under-sampling (EUS)	It consists of removing instances in the majority class by using the guide of a genetic-based algorithm [58].
Easy ensemble (EE)	The data set is divided into several subsets by random resampling with replacement, and each subset is then used to train a base classifier of the ensemble with AdaBoost [59].
Balance cascade (BC)	It performs bagging and removes negative instances that can be classified correctly with high confidence from future selections [59].
Random under-sampling boosting (RUSBOOST)	It combines RUS with the AdaBoost algorithm [60].

To investigate the robustness of the experimental results and obtain fair comparisons between the under-sampling algorithms, we evaluated the performance across three popular classification models of different nature: an instance-based learner (nearest neighbor, 1NN), a decision tree (non-pruned J48) and a linear classifier (support vector machine with a linear kernel, SVM). The parameters of these classifiers throughout the experiments were the default values of the WEKA open source machine learning software [61].

The evaluation of the under-sampling algorithms was estimated by 10-fold cross-validation in order to avoid biased results [62]. Each original data set was randomly divided into 10 stratified parts: for each fold, nine blocks were used as a training set, and the remaining portion was used for testing each of the three classifiers that were trained by the different under-sampling strategies. Finally, the performance measures of each method were averaged over the 10 trials.

4.3. Evaluation Metrics

For a two-class problem, Table 3 shows a 2×2 confusion matrix where each entry contains the number of correct/incorrect classifications [63]. Thus, the true-positive (TP) and the true-negative (TN) values represent the amount of instances from each class that were correctly classified, whereas the false-positive (FP) and false-negative (FN) indicate the amount of positive and negative instances misclassified.

Table 3. Confusion matrix.

	Predicted as Positive	Predicted as Negative
Actually positive	TP	FN
Actually negative	FP	TN

From this confusion matrix, two plain performance evaluation indices can be easily calculated:

- True-positive rate (or sensitivity). It measures the proportion of positive instances correctly classified, $TPR = TP / (TP + FN)$.
- True-negative rate (or specificity). It measures the proportion of negative examples correctly classified, $TNR = TN / (TN + FP)$.

The assessment of classification performance is often carried out using more powerful measures that can be derived from straightforward indices such as the true-positive and true-negative rates. The most popular measure is the overall accuracy, but it is not appropriate when the prior class probabilities are very different because it does not consider misclassification costs and is strongly biased towards the majority class [64]. Thus, the performance of classifiers in the context of class imbalance has commonly been evaluated using other metrics, such as the geometric mean, the *F*-measure, and the area under the

ROC curve. For the present experimental study, the geometric mean was chosen since it represents a good trade-off between the accuracy rates measured on each class:

$$Gmean = \sqrt{TPR \cdot TNR} \tag{1}$$

5. Results

In this section, we report the experimental results obtained by DBMIST-US and several state-of-the-art under-sampling algorithms, pursuing to answer the three research questions stated in Section 4.

5.1. Classification Performance Comparison with State-of-the-Art Methods

The aim of the first block of this experimental study was to compare the performance of DBMIST-US with that of reference methods to assess its competitiveness (Q1), and also to analyze its robustness across classification models (Q2). Tables A1–A3 in Appendix A report the geometric mean (averaged across the 10 runs) of each under-sampling algorithm using the three classifiers on all the data sets. The classification results on the original (non-preprocessed) data sets were also included as a baseline for performance assessment and usefulness testing.

Figure 3 displays the average Friedman rank of the under-sampling algorithm for each classifier. From these graphs, one can observe that DBMIST-US achieved the lowest average ranks regardless of the classifier used, which means that this method performed the best on both the real-life data sets and the synthetic data sets. In addition, as expected, under-sampling the data leads to an improvement in the classification performance, except when using the CNN algorithm; in general, this showed the highest average Friedman ranks (i.e., the worst performance as measured by the geometric mean), especially with the 1NN and J48 classifiers.

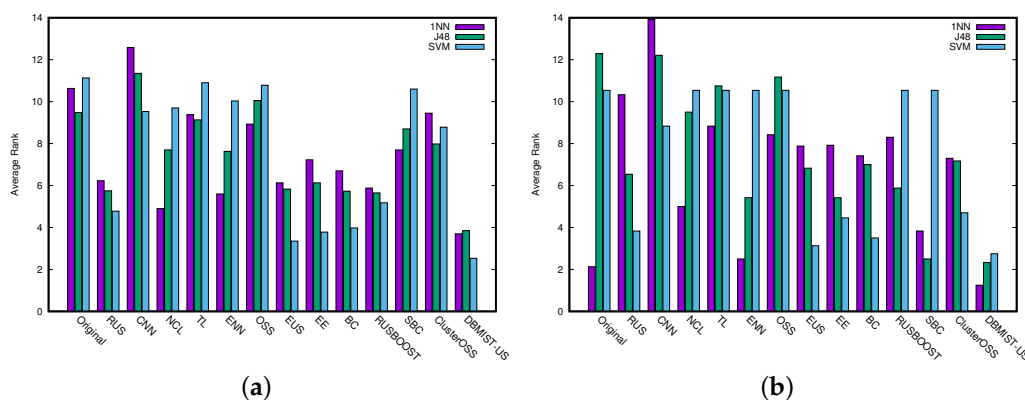


Figure 3. Comparison of average Friedman ranks. (a) real data sets; (b) synthetic data sets.

It is also interesting to draw attention to the behavior of CNN, NCL, TL, ENN, OSS, RUSBOOST, and SBC with the SVM classification model since the geometric mean obtained by these algorithms was 0 for many of the experimental data sets, as can be seen in Table A3. Taking into account that the geometric mean on the respective non-preprocessed data sets was also 0, this behavior indicates that those under-sampling algorithms could not handle either the class overlap or the class imbalance or even both correctly. To gain some insight into this situation, Figure 4 shows the TPR (red triangles) and TNR (black squares) obtained by using SVM on each under-sampled synthetic data set. From these spider plots, one can observe that the bad performance of most of those methods comes from a true-positive rate equal to 0, which means that the SVM model misclassified all instances in the minority class.

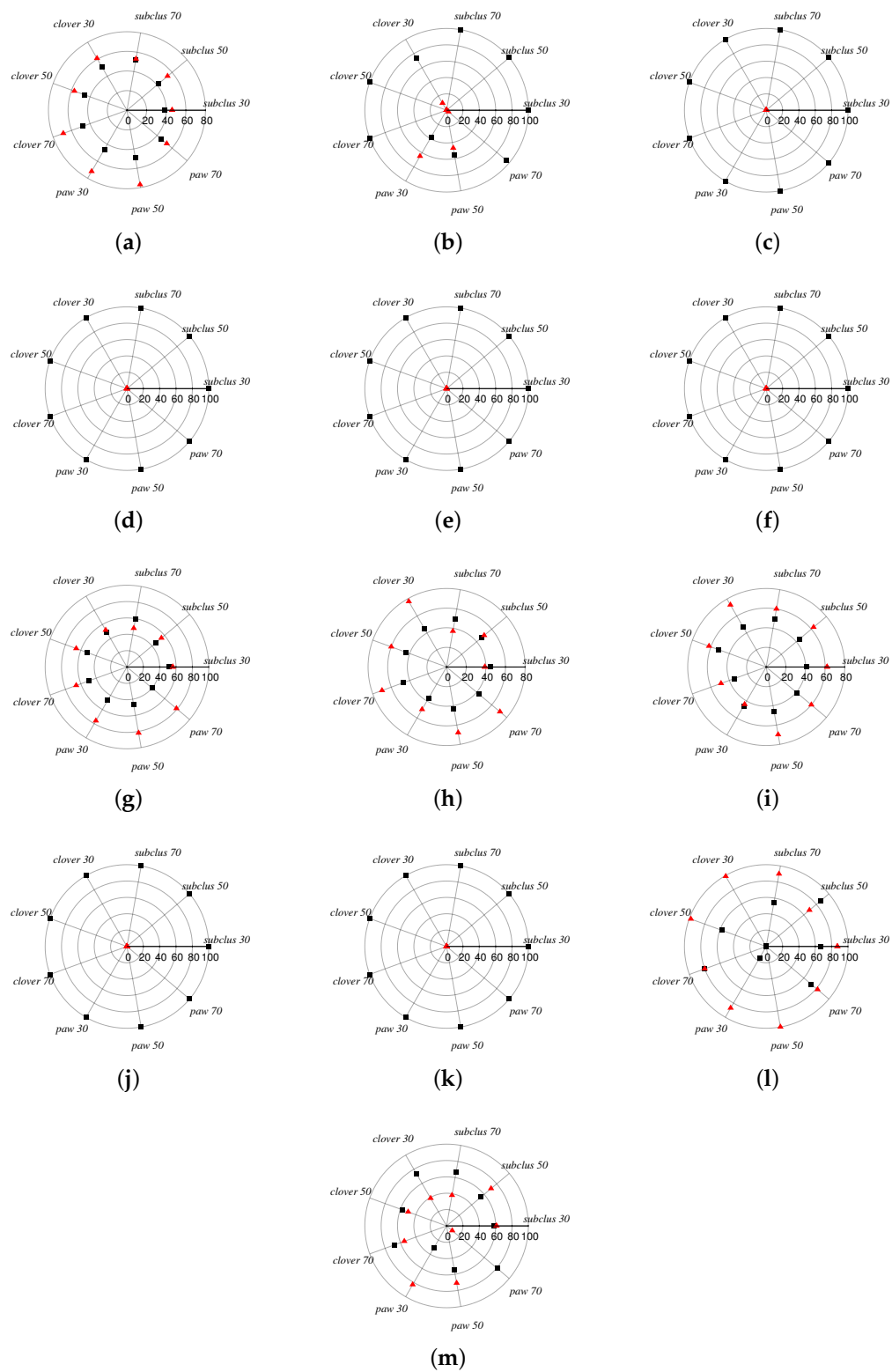


Figure 4. True-positive and true-negative rates using SVM on the synthetic data sets. (a) RUS; (b) CNN; (c) NCL; (d) TL; (e) ENN; (f) OSS; (g) EUS; (h) EE; (i) BC; (j) RUSBOOST; (k) SBC; (l) ClusterOSS; (m) DBMIST-US.

To shed some light onto why the true-positive rate given by those under-sampling algorithms was 0, Figure 5 plots the original paw-50 data set and the under-sampled sets when using either a neighborhood-based method (TL) or a clustering-based method (SBC). One can observe that these algorithms could not solve the class imbalance problem and, consequently, classification by SVM remained biased towards the majority class. On the other hand, the plots show that there still exists some overlapping between classes, indicating that not all noisy negative instances could be removed from the data sets.

A particularly interesting problem emerges with the ClusterOSS method where the performance on the minority class improved significantly, but at the cost of causing a severe degradation of the true-negative rate. In Figure 4, the spider plot for the paw-50 data set brings up an especially dramatic case since the TNR was equal to 0. For a deeper understanding of why such a loss of performance in the true-negative rate arose, the scatter plots in Figure 6 display the under-sampled sets when using ClusterOSS and DBMIST-US. As can be viewed, both methods eliminated the problem of class overlap by cleaning the decision boundary completely. However, the removal of many negative instances with the ClusterOSS algorithm interchanged the role of classes (i.e., the majority class became the minority one) and placed the negative instances further from each other than from the positive instances, whereas DBMIST-US behaved appropriately by producing a perfectly balanced data set and keeping a sufficient number of useful negative instances in well-defined clusters.

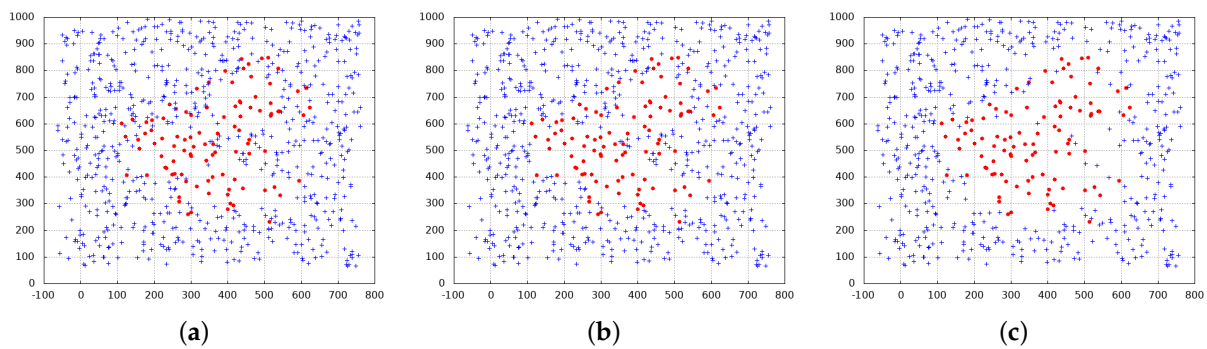


Figure 5. Scatter plots of the paw-50 database for the non-preprocessed set and the sets yielded by TL and SBC. (a) original; (b) TL; (c) SBC.

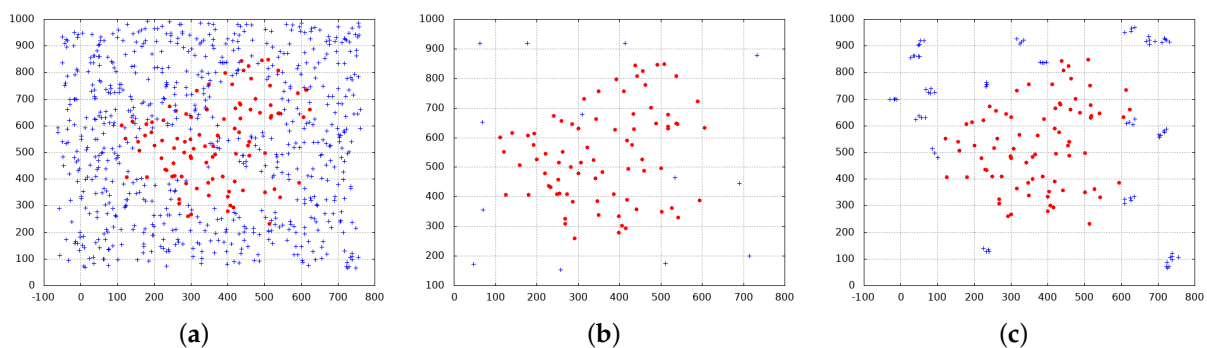


Figure 6. Scatter plots of the paw-50 database for the non-preprocessed set and the sets yielded by ClusterOSS and DBMIST-US. (a) Original; (b) ClusterOSS; (c) DBMIST-US.

For further confirmation of the behavior of DBMIST-US just discussed, the scatter plots in Figure 7 depict the three synthetic databases with 70% of noise in their original class distribution and also after

being under-sampled by the DBMIST-US algorithm. Despite all these corresponding to problems with high overlapping between classes, the under-sampling method was able to properly eliminate the noisy and borderline instances in the majority class and clean the decision boundary. In addition, DBMIST-US reduced the disproportion between positive and negative instances very importantly.

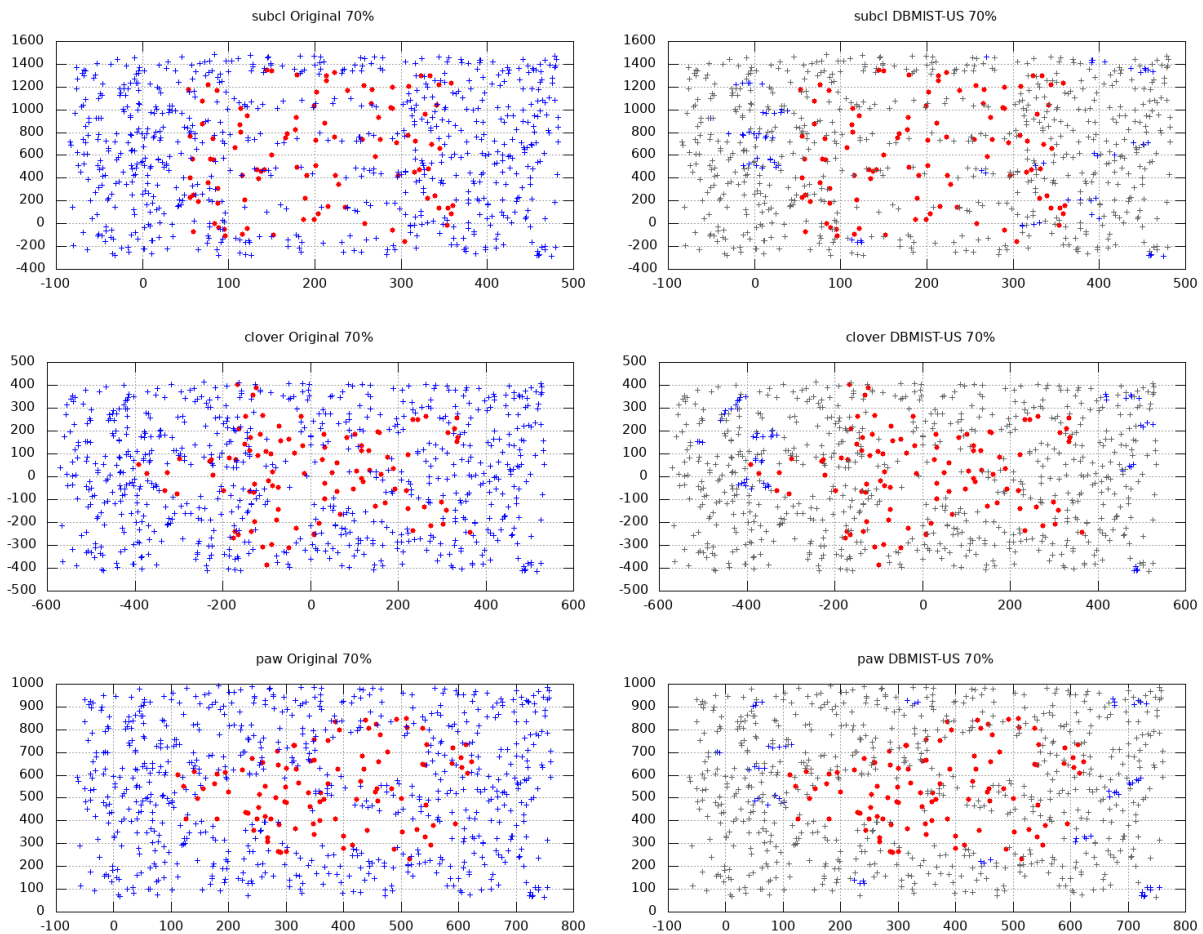


Figure 7. Scatter plots of the synthetic data sets (70% of noise) for the non-preprocessed sets and the sets yielded by DBMIST-US (points in gray color correspond to the instances that were removed).

5.2. Statistical Significance Analysis

The Wilcoxon’s paired signed-rank test to find out statistically significant differences between each pair of under-sampling algorithms for a significance level of $\alpha = 0.05$. In Figure 8, we present the results of a wins-ties-losses analysis, showing the number of data sets on which DBMIST-US obtained statistically significantly better, equal, or worse performance than the reference algorithms on a pairwise basis, for the three classifiers considered in this experimental study.

As can be seen, DBMIST-US significantly outperformed the state-of-the-art under-sampling methods on a majority of databases, independently of the classifier used. This result suggests that the DBMIST-US algorithm is robust against the classification models of different nature.

Figure 9 summarizes the Wilcoxon’s test to show whether or not there exists any significance difference between DBMIST-US and the reference methods for the three classifiers. In this case, the bars represent

how many times an algorithm was significantly better, equal to, or worse than the remaining techniques. The most important finding is that the DBMIST-US method was significantly the best under-sampling algorithm for all classifiers, which highlights the robustness of this method once again.

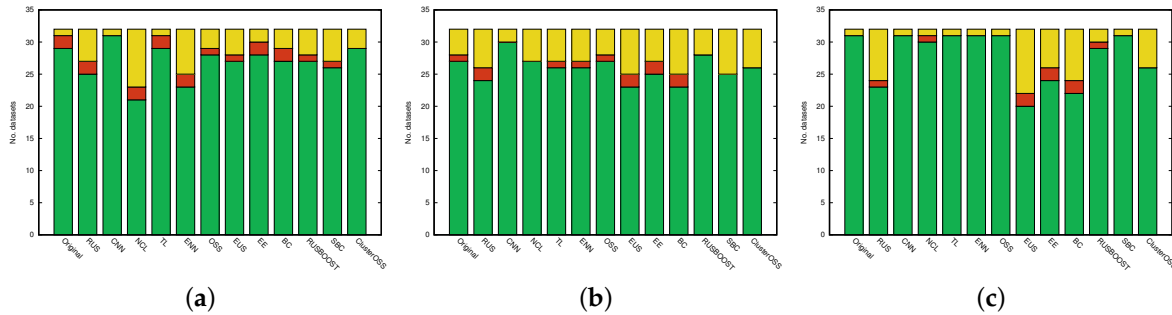


Figure 8. Comparison of DBMIST-US with reference methods with respect to the number of data sets on which DBMIST-US was statistically significantly better (green), equal (yellow), or worse (red). (a) 1NN; (b) J48; (c) SVM.

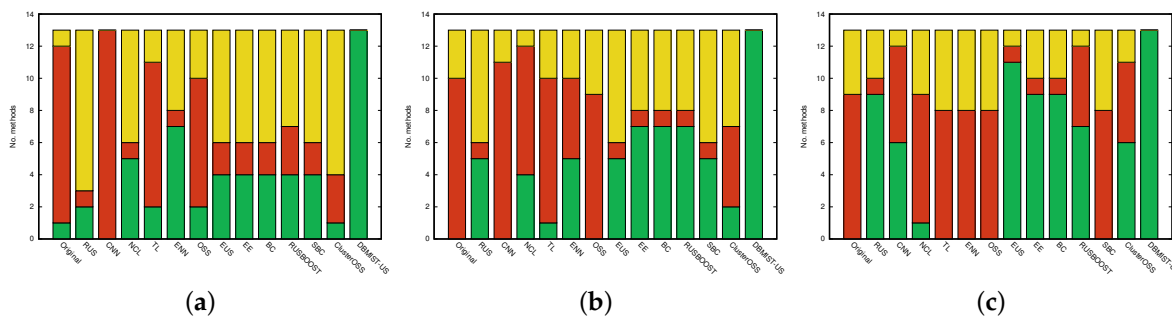


Figure 9. Summary of the Wilcoxon’s test. (a) 1NN; (b) J48; (c) SVM.

5.3. Evaluation of the Impact on the Imbalance Ratio

This block of the experimental study aimed to investigate the impact of each under-sampling algorithm on the imbalance ratio (Q3), that is, to check the ability of each method to provide well-balanced data sets.

Table 4 reports the imbalance ratio of each under-sampled data set. Initially, the imbalance ratio of the 20 real-life data sets was between 9.35 and 82, whereas the imbalance ratio of the synthetic databases was 7. After under-sampling the original sets, RUS, EUS, EE, and BC gave perfectly-balanced data sets with an imbalance ratio equal to 1; conversely, most data sets that were resampled by NCL, TL, ENN, OSS, and SBC are still heavily class-imbalanced, thus leading to an imbalance ratio really close to the original one. Regarding the DBMIST-US algorithm, the imbalance ratios equal or close to 1 indicate that the resulting data sets were well-balanced, demonstrating the effectiveness of the method proposed here. Finally, the ClusterOSS method produced many data sets with an imbalance ratio less than 1, which means that the huge amount of negative instances removed made the majority class smaller than the minority class, as already pointed out in Section 5.1.

Table 4. Imbalance ratio of the resampled data sets.

	Original	RUS	CNN	NCL	TL	ENN	OSS	EUS	EE	BC	RUSBOOST	SBC	ClusterOSS	DBMIST-US
yeast-0-5-6-7-9_vs_4	9.4	1.0	1.7	8.0	9.1	8.2	7.3	1.0	1.0	1.0	1.6	7.5	0.4	1.0
glass-0-1-6_vs_2	19.3	1.0	2.2	8.7	10.1	8.4	9.6	1.0	1.0	1.0	1.0	8.9	4.2	1.1
glass2	11.6	1.0	2.0	10.1	11.4	9.8	10.5	1.0	1.0	1.0	1.5	9.9	1.8	1.1
shuttle-c0-vs-c47	13.9	1.0	0.1	13.8	13.9	13.8	0.3	1.0	1.0	1.0	1.5	13.8	0.4	1.2
yeast-1_vs_7	14.3	1.0	2.8	12.1	14.0	12.5	12.1	1.0	1.0	1.0	1.5	11.4	2.0	1.0
glass4	15.5	1.0	1.0	14.5	15.5	14.9	4.2	1.0	1.0	1.0	1.5	15.6	5.0	1.1
ecoli4	15.8	1.0	1.0	15.2	15.8	15.4	10.3	1.0	1.0	1.0	1.5	14.9	0.9	1.1
page-blocks-1-3_vs_4	15.9	1.0	1.0	15.3	15.8	15.2	4.5	1.0	1.0	1.0	1.5	14.7	3.0	1.0
glass-0-1-6_vs_5	19.4	1.0	1.2	18.3	19.3	18.7	5.9	1.0	1.0	1.0	1.4	18.7	2.0	1.1
yeast-1-4-5-8_vs_7	22.1	1.0	4.1	19.6	21.8	19.8	21.8	1.0	1.0	1.0	1.5	16.9	2.3	1.0
glass5	22.8	1.0	1.2	21.6	22.7	22.0	7.6	1.0	1.0	1.0	1.4	22.0	3.8	1.1
yeast-2_vs_8	23.1	1.0	2.4	21.9	22.9	22.0	21.2	1.0	1.0	1.0	1.5	21.0	0.8	1.0
flare-F	23.8	1.0	3.1	22.0	23.7	21.8	23.7	1.0	1.0	1.0	1.5	23.2	0.1	1.1
yeast4	28.1	1.0	2.5	26.4	27.8	26.6	24.7	1.0	1.0	1.0	1.5	22.1	0.3	1.1
yeast-1-2-8-9_vs_7	30.6	1.0	4.3	28.3	30.2	28.4	30.1	1.0	1.0	1.0	1.5	29.6	0.4	1.0
yeast5	32.7	1.0	1.1	32.0	32.6	32.0	25.0	1.0	1.0	1.0	1.5	32.0	0.3	1.1
ecoli-0-1-3-7_vs_2-6	39.1	1.0	2.1	38.1	38.9	38.0	17.3	1.0	1.0	1.0	1.4	37.6	2.2	1.3
abalone-17_vs_7-8-9-10	39.3	1.0	2.5	37.5	39.1	38.2	37.3	1.1	1.0	1.0	1.5	31.0	0.3	1.1
yeast6	41.4	1.0	2.9	40.0	41.1	39.9	35.5	1.0	1.0	1.0	1.5	40.2	0.4	1.1
poker-8-9_vs_5	82.0	1.0	5.1	79.4	81.7	80.5	80.6	1.0	1.0	1.0	1.5	69.2	0.7	1.1
subcl-0	7.0	1.0	1.3	6.6	6.9	6.5	6.3	1.0	1.0	1.0	1.5	5.7	0.5	1.0
subcl-30	7.0	1.0	1.3	6.4	6.8	5.9	6.7	1.0	1.0	1.0	1.5	4.7	0.6	0.9
subcl-50	7.0	1.0	1.4	6.3	6.6	5.7	6.5	1.0	1.0	1.0	1.5	4.6	1.5	1.0
subcl-70	7.0	1.0	1.4	6.2	6.6	5.6	6.6	1.0	1.0	1.0	1.5	4.6	0.5	1.1
clover-0	7.0	1.0	1.4	6.6	6.9	6.6	6.5	1.0	1.0	1.0	1.5	5.5	1.4	0.9
clover-30	7.0	1.0	1.1	6.5	6.8	6.2	6.5	1.0	1.0	1.0	1.5	5.1	0.7	1.2
clover-50	7.0	1.0	1.3	6.3	6.7	5.8	6.2	1.0	1.0	1.0	1.5	5.1	0.4	1.0
clover-70	7.0	1.0	1.3	6.2	6.6	5.6	6.0	1.0	1.0	1.0	1.5	4.7	1.2	1.1
paw-0	7.0	1.0	2.2	6.8	7.0	6.7	7.0	1.0	1.0	1.0	1.5	6.3	0.1	0.9
paw-30	7.0	1.0	1.1	6.6	6.8	6.2	6.3	1.0	1.0	1.0	1.5	5.2	0.7	0.8
paw-50	7.0	1.0	1.0	6.5	6.7	6.0	6.1	1.0	1.0	1.0	1.5	5.4	0.2	1.0
paw-70	7.0	1.0	1.1	6.3	6.6	5.8	5.8	1.0	1.0	1.0	1.5	5.2	1.2	1.1
% Avg. reduction		91.55	86.98	8.55	3.28	10.18	23.24	91.51	91.55	91.55	87.34	17.33	90.26	91.29

Another way of evaluating the impact of each under-sampling algorithm on the imbalance ratio is looking at the percentage of average reduction of the imbalance ratio given in the last row of Table 4. Results reveal that the top-4 methods were RUS, EUS, EE, and BC with an average reduction of 91.55%, which correspond to the algorithms that produced perfectly-balanced data sets (imbalance ratio equal to 1). However, DBMIST-US with an average reduction of 91.29% was clearly very close to the top-4 methods. On the other hand, the average reductions achieved with NCL, TL, ENN, OSS, and SBC were extremely low, which suggests that these algorithms are not the most appropriate for under-sampling.

6. Conclusions

In this work, we have proposed a new under-sampling method called DBMIST-US to face class overlap and class imbalance through the combination of the DBSCAN clustering algorithm with the generation of an MST. The goal of the technique proposed here is to handle both data complexities simultaneously using a two-stage algorithm. In the first stage, the decision boundary is cleaned by removing negative instances that have been considered as noise, whereas the second stage is in charge of balancing the classes by selecting representative instances from the majority class according to a maximum imbalance ratio predefined by the user.

As already introduced in Section 3.3, several differences between DBMIST-US and some recent related under-sampling algorithms can be highlighted: (i) DBMIST-US faces class overlap and class imbalance simultaneously, whereas other methods are designed to tackle these data complexities separately or by applying a series of techniques in a sequential manner, (ii) the baseline clustering-based under-sampling approaches are based on the classical K -means algorithm, whereas our proposal combines DBSCAN with MST, and (iii) the algorithms that rest upon the use of DBSCAN are primarily developed for over-sampling the minority class instead of under-sampling the majority class.

We carried out a set of experiments to validate the performance of the DBMIST-US method against 12 state-of-the-art under-sampling algorithms using three supervised classifiers (1NN, J48, and SVM), tested on both real-life data sets and synthetic data sets with an imbalance ratio ranging from 7 to 82. The results have shown that the method here proposed performs well with noisy and borderline problems and better than the remaining algorithms when applied on heavily imbalanced data sets, especially using the SVM classification model. Another research question that has properly been answered in the experimental study refers to robustness, showing that DBMIST-US is high robust to the use of classifiers of different nature. It is also worth noting the percentage of average reduction of the imbalance ratio achieved by DBMIST-US, which was comparable to the reduction produced by the best reference algorithms.

The promising results of the DBMIST-US algorithm encourage us to examine some directions for future research. The most immediate issue relates to the generalization of DBMIST-US for handling the multi-class imbalance problem, which emerges as a more challenging situation in many practical applications. A second subject that deserves further research is the multi-label classification of imbalanced data where the classes are not mutually exclusive, which represents a much more complex classification task because an instance may belong to more than one class. However, another avenue for investigation is to analyze how the DBMIST-US algorithm can be used on large-scale data and data streams.

Author Contributions: A.G.-P., R.-M.V., and J.-S.S. designed the study; A.G.-P. implemented the algorithms and carried out the experiments; J.-S.S. and J.-R.M.-R. performed the analysis; J.-S.S. wrote the manuscript; J.-S.S. and R.-M.V. revised and validated the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Universitat Jaume I under grant [UJI-B2018-49], the 5046/2020CIC UAEM project and the Mexican Science and Technology Council (CONACYT) under scholarship [702275].

Conflicts of Interest: The authors declare no conflict of interests.

Abbreviations

The following abbreviations are used in this manuscript:

- DBSCAN Density Based Spatial Clustering of Applications with Noise
- MST Minimum Spanning Tree
- DBMIST-US DBSCAN and MST - Under-Sampling
- IR Imbalance ratio
- kNN k Nearest Neighbor
- SVM Support Vector Machine
- SMOTE Synthetic Minority Over-sampling Technique

Appendix A. Classification Results

This appendix summarizes the classification performance of each under-sampling algorithm using 1NN, J48, and SVM. Values in boldface highlight the best result for each database.

Table A1. Geometric mean results obtained with 1NN.

	Original	RUS	CNN	NCL	TL	ENN	OSS	EUS	EE	BC	RUSBOOST	SBC	ClusterOSS	DBMIST-US
yeast-0-5-6-7-9_vs_4	61.2	73.5	50.7	83.0	69.0	78.9	72.2	78.4	73.2	73.5	76.7	77.1	58.9	75.4
glass-0-1-6_vs_2	47.0	74.4	51.0	66.9	52.6	67.9	52.9	68.3	76.2	73.5	64.1	53.1	33.0	76.3
glass2	40.7	76.2	51.6	53.1	41.0	67.1	47.4	64.7	60.0	67.6	68.7	53.1	66.9	76.9
shuttle-c0-vs-c4_7	99.6	99.6	100.0	99.6	99.6	99.6	100.0	99.6	99.6	99.6	99.6	99.6	99.6	99.6
yeast-1_vs_7	62.3	74.8	46.2	76.8	67.6	70.4	69.9	66.6	59.6	58.1	65.9	67.2	62.4	69.2
glass4	86.6	79.9	65.3	91.3	86.6	87.7	83.6	80.7	84.3	84.3	87.4	86.6	56.9	92.1
ecoli4	86.3	95.0	81.4	89.4	86.3	92.2	86.2	92.5	100.0	100.0	92.2	91.9	98.9	100.0
page-blocks-1-3_vs_4	98.2	98.2	94.5	98.2	98.2	98.2	98.2	94.5	94.6	96.4	96.6	99.9	92.2	98.2
glass-0-1-6_vs_5	80.9	77.0	65.1	100.0	80.9	100.0	79.3	94.3	94.3	94.3	91.0	94.0	78.1	95.4
yeast-1-4-5-8_vs_7	44.1	64.5	39.7	62.7	47.9	48.0	54.2	69.7	69.3	60.0	65.1	57.0	64.9	61.3
glass5	81.1	94.3	75.2	93.8	81.2	94.0	79.8	88.2	72.0	72.0	91.2	93.3	44.3	92.7
yeast-2_vs_8	77.0	59.8	58.3	80.6	77.3	80.5	77.4	54.8	77.5	72.5	75.1	80.2	71.5	74.5
flare-F	30.3	81.4	30.6	65.9	33.7	72.6	26.2	78.9	76.6	68.5	77.3	26.1	0.0	87.3
yeast4	58.8	83.3	43.7	80.1	62.2	71.3	58.9	84.3	77.4	82.4	77.1	78.6	75.4	79.8
yeast-1-2-8-9_vs_7	47.6	56.6	42.5	65.4	54.2	60.4	60.0	64.5	63.2	69.9	65.0	47.6	69.2	66.1
yeast5	82.1	100.0	65.5	96.3	87.6	94.1	88.7	93.2	92.0	96.5	94.6	82.1	92.5	97.1
ecoli-0-1-3-7_vs_2-6	83.7	78.2	48.8	84.4	84.0	84.4	84.2	92.6	71.4	71.4	81.9	84.2	83.7	87.3
abalone-17_vs_7-8-9-10	50.6	77.6	44.8	66.7	45.3	61.4	50.7	73.9	78.2	73.2	78.6	58.3	44.8	78.9
yeast6	71.1	69.9	54.8	86.1	79.0	82.7	80.6	81.0	75.1	81.4	83.7	71.2	88.0	81.1
poker-8-9_vs_5	19.9	53.1	14.8	20.0	20.0	20.0	20.0	61.2	45.3	63.5	65.1	34.5	64.0	80.0
subcl-0	86.2	89.9	66.1	91.1	90.4	95.1	87.8	94.5	90.4	92.5	89.5	91.7	88.6	98.1
subcl-30	72.0	72.5	58.0	78.9	75.3	83.4	75.5	74.5	83.5	77.0	77.3	83.3	87.2	90.6
subcl-50	60.5	69.0	46.0	73.6	68.2	81.9	68.3	74.0	71.5	81.0	74.5	80.6	86.6	89.3
subcl-70	52.5	70.5	35.9	72.3	62.7	82.4	62.0	79.8	72.5	73.5	71.6	74.0	85.5	93.0
clover-0	89.1	90.4	59.5	98.7	94.8	96.8	95.1	89.9	89.4	93.4	91.6	98.1	87.6	100.0
clover-30	80.4	78.0	50.3	86.8	81.6	88.0	81.9	78.5	80.9	76.9	82.6	87.6	80.4	96.3
clover-50	70.0	74.5	48.6	83.1	75.9	87.5	75.6	75.8	75.4	76.9	76.3	85.9	81.6	98.5
clover-70	62.5	73.9	41.1	80.4	71.9	87.4	75.3	77.5	74.9	77.0	73.4	78.7	85.0	99.1
paw-0	93.5	94.9	60.3	98.4	96.7	98.5	95.6	97.0	93.0	92.4	95.1	98.7	71.9	97.3
paw-30	78.3	77.5	51.9	86.7	83.5	91.8	83.7	83.0	83.5	77.9	83.0	87.5	79.4	94.0
paw-50	74.1	81.5	46.3	86.0	81.8	89.4	83.1	82.8	83.0	87.0	79.9	86.1	45.9	98.1
paw-70	64.0	77.0	45.0	78.3	74.7	89.8	75.8	74.6	79.0	77.5	76.8	82.1	86.2	98.5

Table A2. Geometric mean results obtained with J48.

	Original	RUS	CNN	NCL	TL	ENN	OSS	EUS	EE	BC	RUSBOOST	SBC	ClusterOSS	DBMIST-US
yeast-0-5-6-7-9_vs_4	65.1	70.6	60.2	70.6	65.8	74.5	64.3	73.2	70.5	68.6	77.0	74.3	71.3	76.7
glass-0-1-6_vs_2	52.3	60.0	43.8	66.7	53.1	53.3	58.3	46.7	57.6	45.6	63.5	52.0	4.6	64.8
glass2	53.1	90.7	22.4	57.6	33.2	58.0	52.4	55.2	73.0	61.1	65.2	58.2	65.0	72.5
shuttle-c0-vs-c47	100.0	100.0	99.6	99.9	100.0	100.0	100.0	100.0	100.0	100.0	99.9	99.9	99.9	100.0
yeast-1_vs_7	54.3	64.5	57.2	57.2	65.2	62.6	54.2	71.6	58.3	81.5	67.4	62.8	68.4	69.7
glass4	82.2	73.0	73.0	82.1	82.2	86.6	79.3	88.4	92.3	92.3	89.1	72.4	9.0	92.3
ecoli4	82.9	87.5	92.3	80.1	82.9	80.1	86.0	76.5	87.5	92.5	86.3	77.1	93.5	82.8
page-blocks-1-3_vs_4	96.1	100.0	94.5	99.8	99.8	96.1	97.4	98.2	96.4	94.5	97.5	97.1	93.8	98.2
glass-0-1-6_vs_5	99.4	88.2	95.3	99.7	93.7	99.7	97.1	100.0	94.3	94.3	92.5	99.7	79.0	92.3
yeast-1-4-5-8_vs_7	0.0	59.6	0.0	0.0	0.0	0.0	0.0	66.6	60.6	44.7	53.1	25.8	64.3	55.0
glass5	98.8	100.0	85.3	99.7	99.5	99.7	98.5	94.3	94.3	94.3	93.9	100.0	49.5	95.8
yeast-2_vs_8	22.4	61.2	69.2	22.3	0.0	31.6	22.4	74.8	71.4	76.5	72.1	31.6	67.1	73.0
flare-F	15.2	85.9	32.7	62.4	0.0	56.8	0.0	84.8	86.0	79.5	84.2	0.0	0.0	90.5
yeast4	53.9	78.3	0.0	60.7	57.4	65.4	53.9	83.3	80.3	84.1	78.2	68.4	80.9	79.3
yeast-1-2-8-9_vs_7	48.2	53.7	47.9	44.6	44.6	44.6	44.6	59.9	54.2	69.7	67.3	40.8	68.4	62.8
yeast5	86.3	96.6	77.8	96.4	89.0	93.9	88.9	87.4	94.1	89.8	94.2	88.9	90.2	96.6
ecoli-0-1-3-7_vs_2-6	84.4	71.4	70.4	84.5	84.4	84.5	83.5	70.0	78.2	78.2	74.5	84.4	72.4	94.3
abalone-17_vs_7-8-9-10	34.7	75.0	0.0	43.3	34.6	39.3	32.1	78.8	75.0	74.0	75.5	57.0	45.9	76.0
yeast6	73.3	72.8	70.2	77.2	73.3	73.5	67.3	82.6	71.4	88.6	81.9	71.4	80.4	84.6
poker-8-9_vs_5	0.0	47.8	0.0	0.0	0.0	0.0	0.0	44.0	40.0	42.3	44.6	0.0	52.5	59.6
subcl-0	97.8	92.3	77.9	97.8	97.8	98.3	97.2	95.5	93.9	88.6	92.3	96.6	87.3	94.4
subcl-30	66.7	75.9	74.0	71.4	70.2	87.5	68.5	75.3	83.2	79.7	83.5	87.3	89.4	85.2
subcl-50	26.4	78.0	31.3	74.8	53.4	84.8	68.2	74.7	80.8	82.0	80.9	83.2	83.7	82.6
subcl-70	0.0	76.5	0.0	32.6	24.4	72.6	14.1	77.8	80.7	73.0	79.9	80.1	83.5	86.7
clover-0	63.3	85.6	11.6	65.1	66.7	66.9	66.6	83.4	77.5	78.2	81.5	92.7	82.1	83.9
clover-30	26.4	74.6	59.2	42.9	45.1	71.5	28.2	68.9	79.1	79.6	81.7	84.0	52.7	97.4
clover-50	0.0	71.8	0.0	59.1	47.9	64.3	10.0	77.7	82.1	75.5	75.8	84.2	78.4	94.7
clover-70	0.0	68.2	27.2	22.3	0.0	67.7	0.0	71.4	80.1	62.6	75.3	82.2	80.4	97.4
paw-0	80.4	93.9	39.0	65.9	65.1	71.6	80.1	91.2	89.0	90.3	92.2	97.2	35.1	94.6
paw-30	19.9	83.4	60.1	79.7	72.5	87.5	60.7	79.2	83.2	82.8	82.3	87.6	76.0	91.1
paw-50	28.2	80.6	52.0	82.5	51.8	87.5	51.5	86.7	86.9	85.6	82.0	89.0	30.0	94.4
paw-70	0.0	78.4	0.0	63.9	32.9	88.4	34.2	77.3	82.1	78.2	82.0	90.8	76.8	92.8

Table A3. Geometric mean results obtained with SVM.

	Original	RUS	CNN	NCL	TL	ENN	OSS	EUS	EE	BC	RUSBOOST	SBC	ClusterOSS	DBMIST-US
yeast-0-5-6-7-9_vs_4	0.0	78.3	55.0	42.0	0.0	37.0	0.0	78.4	75.3	74.5	77.7	44.2	30.1	79.5
glass-0-1-6_vs_2	0.0	55.2	0.0	0.0	0.0	0.0	0.0	45.6	47.8	44.0	32.1	0.0	0.0	57.4
glass2	0.0	68.6	0.0	0.0	0.0	0.0	0.0	47.1	59.4	54.2	21.8	0.0	29.5	71.7
shuttle-c0-vs-c47	99.6	100.0	99.6	100.0	99.6	99.6	99.6	100.0	100.0	100.0	100.0	99.6	99.9	100.0
yeast-1_vs_7	0.0	81.2	0.0	0.0	0.0	0.0	0.0	83.3	74.1	74.8	64.5	0.0	12.7	76.6
glass4	39.2	68.8	40.7	39.2	39.2	39.2	39.2	84.3	84.3	84.3	87.6	39.2	0.0	92.0
ecoli4	63.2	92.5	92.3	74.2	63.2	74.2	70.7	97.5	97.5	100.0	94.9	74.2	99.0	100.0
page-blocks-1-3_vs_4	65.4	71.9	74.5	65.4	65.4	65.4	65.5	79.4	79.9	75.1	73.1	65.4	25.6	84.5
glass-0-1-6_vs_5	0.0	81.6	73.9	0.0	0.0	0.0	0.0	94.3	81.6	81.6	86.4	0.0	42.3	90.2
yeast-1-4-5-8_vs_7	0.0	60.6	0.0	0.0	0.0	0.0	0.0	58.9	67.8	49.4	47.3	0.0	60.5	62.0
glass5	0.0	88.2	75.2	0.0	0.0	0.0	0.0	88.2	81.6	81.6	81.8	0.0	43.1	91.2
yeast-2_vs_8	74.1	72.3	73.4	74.2	74.2	74.2	74.2	74.2	74.2	77.5	73.3	74.2	70.6	73.0
flare-F	0.0	78.2	36.2	42.9	15.2	50.4	0.0	76.8	78.5	76.2	81.7	0.0	0.0	88.7
yeast4	0.0	82.4	0.0	0.0	0.0	0.0	0.0	85.2	82.3	84.3	80.9	0.0	24.6	82.3
yeast-1-2-8-9_vs_7	0.0	52.4	0.0	0.0	0.0	0.0	0.0	69.7	74.5	74.8	56.4	0.0	43.1	70.9
yeast5	0.0	98.9	70.9	33.7	0.0	39.9	30.2	92.9	94.1	94.1	95.8	0.0	92.6	96.3
ecoli-0-1-3-7_vs_2-6	84.1	78.2	78.7	84.2	84.0	84.2	83.5	92.6	84.5	84.5	84.1	84.0	82.8	92.6
abalone-17_vs_7-8-9-10	0.0	74.1	0.0	0.0	0.0	0.0	0.0	69.8	75.0	69.8	69.6	0.0	0.0	72.1
yeast6	0.0	78.5	0.0	0.0	0.0	0.0	0.0	88.4	81.3	87.1	87.7	0.0	86.1	86.3
poker-8-9_vs_5	0.0	51.8	0.0	0.0	0.0	0.0	0.0	44.9	37.9	35.8	30.1	0.0	2.5	62.1
subcl-0	0.0	51.3	0.0	0.0	0.0	0.0	0.0	54.5	47.0	56.8	0.0	0.0	0.0	67.5
subcl-30	0.0	41.8	0.0	0.0	0.0	0.0	0.0	54.0	41.9	50.4	0.0	0.0	76.3	59.4
subcl-50	0.0	47.6	0.0	0.0	0.0	0.0	0.0	50.3	48.5	52.6	0.0	0.0	77.3	62.3
subcl-70	0.0	52.5	0.0	0.0	0.0	0.0	0.0	53.2	43.0	54.8	0.0	0.0	67.6	50.3
clover-0	0.0	54.5	0.0	0.0	0.0	0.0	0.0	47.1	52.0	48.4	0.0	0.0	29.5	93.5
clover-30	0.0	55.8	27.2	0.0	0.0	0.0	0.0	50.5	58.9	58.6	0.0	0.0	2.2	54.0
clover-50	0.0	51.2	0.0	0.0	0.0	0.0	0.0	58.6	51.4	56.8	0.0	0.0	74.5	53.3
clover-70	0.0	57.5	0.0	0.0	0.0	0.0	0.0	57.4	57.4	41.4	0.0	0.0	80.0	61.2
paw-0	0.0	64.2	0.0	0.0	0.0	0.0	0.0	47.4	39.4	57.6	0.0	0.0	0.0	85.8
paw-30	0.0	57.5	49.6	0.0	0.0	0.0	0.0	59.8	43.0	45.0	0.0	0.0	32.4	50.6
paw-50	0.0	61.4	51.3	0.0	0.0	0.0	0.0	62.0	54.1	56.7	0.0	0.0	0.0	61.9
paw-70	0.0	49.4	17.0	0.0	0.0	0.0	0.0	55.9	55.3	49.6	0.0	0.0	77.0	27.1

References

1. Sun, Y.; Kamel, M.S.; Wong, A.K.; Wang, Y. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognit.* **2007**, *40*, 3358–3378. [\[CrossRef\]](#)
2. Codetta-Raiteri, D.; Portinale, L. Dynamic Bayesian networks for fault detection, identification, and recovery in autonomous spacecraft. *IEEE Trans. Syst. Man Cybern. Syst.* **2015**, *45*, 13–24. [\[CrossRef\]](#)
3. Zhang, Y.; Zhou, Z.H. Cost-sensitive face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *32*, 1758–1769. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Liu, C.L.; Hsaio, W.H.; Lee, C.H.; Tao, C.; Kuo, T.H. Semi-supervised text classification with universum learning. *IEEE Trans. Cybern.* **2015**, *46*, 462–473. [\[CrossRef\]](#)
5. Gopalakrishnan, V.; Ramaswamy, C. Sentiment learning from imbalanced dataset: An ensemble based method. *Int. J. Artif. Intell.* **2014**, *12*, 75–87.
6. García, V.; Marqués, A.I.; Sánchez, J.S. Improving risk predictions by preprocessing imbalanced credit data. In Proceedings of the 19th International Conference on Neural Information Processing, Doha, Qatar, 12–15 November 2012; pp. 68–75.
7. Fernández, A.; García, S.; Galar, M.; Prati, R.; Krawczyk, B.; Herrera, F. *Learning from Imbalanced Data Sets*; Springer: Cham, Switzerland, 2018; Volume 1.
8. Sáez, J.A.; Luengo, J.; Stefanowski, J.; Herrera, F. SMOTE-IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Inf. Sci.* **2015**, *291*, 184–203. [\[CrossRef\]](#)
9. García, V.; Alejo, R.; Sánchez, J.S.; Sotoca, J.M.; Mollineda, R.A. Combined effects of class imbalance and class overlap on instance-based classification. In Proceedings of the 6th International Conference on Intelligent Data Engineering and Automated Learning, Burgos, Spain, 20–23 September 2006; pp. 371–378.
10. Gupta, S.; Gupta, A. Handling class overlapping to detect noisy instances in classification. *Knowl. Eng. Rev.* **2018**, *33*, e8. [\[CrossRef\]](#)
11. Khoshgoftaar, T.M.; Van Hulse, J.; Napolitano, A. Supervised neural network modeling: An empirical investigation into learning from imbalanced data with labeling errors. *IEEE Trans. Neural Netw.* **2010**, *21*, 813–830. [\[CrossRef\]](#)
12. Van Hulse, J.; Khoshgoftaar, T.M.; Napolitano, A. A novel noise filtering algorithm for imbalanced data. In Proceedings of the 9th International Conference on Machine Learning and Applications, Washington, DC, USA, 12–14 December 2010; pp. 9–14.
13. Muhlenbach, F.; Lallich, S.; Zighed, D.A. Identifying and handling mislabelled instances. *J. Intell. Inf. Syst.* **2004**, *22*, 89–109. [\[CrossRef\]](#)
14. Dong, X.; He, H.; Li, C.; Liu, Y.; Xiong, H. Scene-based big data quality management framework. In *Data Science*; Springer: Singapore, 2018; pp. 122–139.
15. Barandela, R.; Sánchez, J.; García, V.; Rangel, E. Strategies for learning in class imbalance problems. *Pattern Recognit.* **2003**, *36*, 849–851. [\[CrossRef\]](#)
16. Ofek, N.; Rokach, L.; Stern, R.; Shabtai, A. Fast-CBUS: A fast clustering-based undersampling method for addressing the class imbalance problem. *Neurocomputing* **2017**, *243*, 88–102. [\[CrossRef\]](#)
17. Yen, S.J.; Lee, Y.S. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Syst. Appl.* **2009**, *36*, 5718–5727. [\[CrossRef\]](#)
18. Napierała, K.; Stefanowski, J.; Wilk, S. Learning from imbalanced data in the presence of noisy and borderline examples. In Proceedings of the 7th International Conference on Rough Sets and Current Trends in Computing, Warsaw, Poland, 28–30 June 2010; pp. 158–167.
19. Alejo, R.; Valdovinos, R.M.; García, V.; Pacheco-Sanchez, J.H. A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios. *Pattern Recognit. Lett.* **2013**, *34*, 380–388. [\[CrossRef\]](#)
20. García, V.; Sánchez, J.S.; Mollineda, R.A. An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. In Proceedings of the 5th Iberoamerican Congress on Pattern Recognition, Valparaiso, Chile, 13–16 November 2007; pp. 397–406.

21. Van-Hulse, J.; Khoshgoftaar, T. Knowledge discovery from imbalanced and noisy data. *Data Knowl. Eng.* **2009**, *68*, 1513–1542. [[CrossRef](#)]
22. Hart, P. The condensed nearest neighbor rule. *IEEE Trans. Inf. Theory* **1968**, *14*, 515–516. [[CrossRef](#)]
23. Tomek, I. Two modifications of CNN. *IEEE Trans. Syst. Man Cybern.* **1976**, *SMC-6*, 769–772. [[CrossRef](#)]
24. Laurikkala, J. Improving identification of difficult small classes by balancing class distribution. In Proceedings of the 8th Conference on Artificial Intelligence in Medicine, Cascais, Portugal, 1–4 July 2001; pp. 63–66.
25. Branco, P.; Torgo, L.; Ribeiro, R.P. A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv.* **2016**, *49*, 1–50. [[CrossRef](#)]
26. Lin, W.C.; Tsai, C.F.; Hu, Y.H.; Jhang, J.S. Clustering-based undersampling in class-imbalanced data. *Inf.Sci.* **2017**, *409–410*, 17–26. [[CrossRef](#)]
27. Drummond, C.; Holte, R.C. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In Proceedings of the Workshop on Learning from Imbalanced Datasets II, Washington, DC, USA, 21 August 2003; Volume 11.
28. García, V.; Sánchez, J.S.; Marqués, A.I.; Florencia, R.; Rivera, G. Understanding the apparent superiority of over-sampling through an analysis of local information for class-imbalanced data. *Expert Syst. Appl.* **2019**, 1–19. [[CrossRef](#)]
29. Basgall, M.J.; Hasperué, W.; Naiouf, M.; Fernández, A.; Herrera, F., An analysis of local and global solutions to address big data imbalanced classification: A case study with SMOTE Preprocessing. In *Cloud Computing and Big Data*; Springer International Publishing: Cham, Switzerland, 2019; pp. 75–85.
30. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, USA, 2–4 August 1996; AAAI Press: Portland, OR, USA, 1996; pp. 226–231.
31. Ijaz, M.F.; Attique, M.; Son, Y. Data-driven cervical cancer prediction model with outlier detection and over-sampling methods. *Sensors* **2020**, *20*, 2809. [[CrossRef](#)]
32. García, S.; Derrac, J.; Cano, J.; Herrera, F. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 417–435. [[CrossRef](#)] [[PubMed](#)]
33. Wilson, D.L. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Trans. Syst. Man Cybern.* **1972**, *SMC-2*, 408–421. [[CrossRef](#)]
34. Tomek, I. An experiment with the edited nearest-neighbor rule. *IEEE Trans. Syst. Man Cybern.* **1976**, *SMC-6*, 448–452. [[CrossRef](#)]
35. Kubat, M.; Matwin, S. Addressing the curse of imbalanced training sets: One-sided selection. In Proceedings of the 14th International Conference on Machine Learning, Nashville, TN, USA, 8–12 July 1997; pp. 179–186.
36. Longadge, R.; Dongre, S.S.; Malik, L. Multi-cluster based approach for skewed data in data mining. *IOSR J. Comput. Eng.* **2013**, *12*, 66–73. [[CrossRef](#)]
37. Barella, V.H.; Costa, E.P.; Carvalho, A.C.P.L.F. ClusterOSS: A new undersampling method for imbalanced learning. In Proceedings of the 3rd Brazilian Conference on Intelligent Systems, São Carlos, Brazil, 18–23 October 2014; pp. 453–458.
38. Sowah, R.A.; Agebure, M.A.; Mills, G.A.; Koumadi, K.M.; Fiwawo, S.Y. New cluster undersampling technique for class imbalance learning. *Int. J. Mach. Learn. Comput.* **2016**, *6*, 205. [[CrossRef](#)]
39. Das, B.; Krishnan, N.C.; Cook, D.J., Handling imbalanced and overlapping classes in smart environments prompting dataset. In *Data Mining for Service*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 199–219.
40. Tsai, C.F.; Lin, W.C.; Hu, Y.H.; Yao, G.T. Under-sampling class imbalanced datasets by combining clustering analysis and instance selection. *Inf. Sci.* **2019**, *477*, 47–54. [[CrossRef](#)]
41. Ng, W.W.Y.; Hu, J.; Yeung, D.S.; Yin, S.; Roli, F. Diversified sensitivity-based undersampling for imbalance classification problems. *IEEE Trans. Cybern.* **2015**, *45*, 2402–2412. [[CrossRef](#)] [[PubMed](#)]
42. Sun, Z.; Song, Q.; Zhu, X.; Sun, H.; Xu, B.; Zhou, Y. A novel ensemble method for classifying imbalanced data. *Pattern Recognit.* **2015**, *48*, 1623–1637. [[CrossRef](#)]
43. Kim, H.J.; Jo, N.O.; Shin, K.S. Optimization of cluster-based evolutionary undersampling for the artificial neural networks in corporate bankruptcy prediction. *Expert Syst. Appl.* **2016**, *59*, 226–234. [[CrossRef](#)]

44. Smiti, A.; Elouedi, Z. DBSCAN-GM: An improved clustering method based on Gaussian means and DBSCAN techniques. In Proceedings of the IEEE 16th International Conference on Intelligent Engineering Systems, Lisbon, Portugal, 13–15 June 2012; pp. 573–578.
45. Prim, R.C. Shortest connection networks and some generalizations. *Bell Syst. Tech. J.* **1957**, *36*, 1389–1401.
46. Torres, M.; Paz, K.; Salazar, F. Tamaño de una muestra para una investigación de mercado. *Boletín Electrónico* **2006**, *2*, 1–13.
47. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to Algorithms*; MIT Press: Cambridge, MA, USA, 2009.
48. Suthar, N.; Indr, P.; Vinit, P. A technical survey on DBSCAN clustering algorithm. *Int. J. Sci. Eng. Res.* **2013**, *4*, 1775–1781.
49. Chen, L.; Fang, B.; Shang, Z.; Tang, Y. Tackling class overlap and imbalance problems in software defect prediction. *Software Qual. J.* **2018**, *26*, 97–125. [[CrossRef](#)]
50. Koziarski, M.; Woźniak, M. CCR: A combined cleaning and resampling algorithm for imbalanced data classification. *Int. J. Appl. Math. Comput. Sci.* **2017**, *27*, 727–736. [[CrossRef](#)]
51. Xiao, L.; Gao, M.; Su, X. An under-sampling ensemble classification algorithm based on fuzzy C-means clustering for imbalanced data. *Data Anal. Knowl. Discov.* **2019**, *3*, 90–96. [[CrossRef](#)]
52. Liang, J.; Bai, L.; Dang, C.; Cao, F. The K-means-type algorithms versus imbalanced data distributions. *IEEE Trans. Fuzzy Syst.* **2012**, *20*, 728–745. [[CrossRef](#)]
53. García, V.; Sánchez, J.S.; Martín-Félez, R.; Mollineda, R.A. Surrounding neighborhood-based SMOTE for learning from imbalanced data sets. *Progr. Artif. Intell.* **2012**, *1*, 347–362. [[CrossRef](#)]
54. Sanguanmak, Y.; Hanskunatai, A. DBSM: The combination of DBSCAN and SMOTE for imbalanced data classification. In Proceedings of the 13th International Joint Conference on Computer Science and Software Engineering, Khon Kaen, Thailand, 13–15 July 2016; pp. 1–5.
55. Bunkhumpornpat, C.; Sinapiromsaran, K.; Lursinsap, C. DBSMOTE: Density-based synthetic minority over-sampling technique. *Appl. Intell.* **2012**, *36*, 664–684. [[CrossRef](#)]
56. Bunkhumpornpat, C.; Sinapiromsaran, K.; Lursinsap, C. MUTE: Majority under-sampling technique. In Proceedings of the 8th International Conference on Information, Communications & Signal Processing, Singapore, 13–16 December 2011; pp. 1–4.
57. Bunkhumpornpat, C.; Sinapiromsaran, K. DBMUTE: Density-based majority under-sampling technique. *Knowl. Inf. Syst.* **2017**, *50*, 827–850. [[CrossRef](#)]
58. García, S.; Herrera, F. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evol. Comput.* **2009**, *17*, 275–306. [[CrossRef](#)]
59. Liu, X.; Wu, J.; Zhou, Z. Exploratory undersampling for class-imbalance learning. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2009**, *39*, 539–550. [[CrossRef](#)]
60. Seiffert, C.; Khoshgoftaar, T.M.; Van Hulse, J.; Napolitano, A. RUSBoost: A hybrid approach to alleviating class imbalance. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2010**, *40*, 185–197. [[CrossRef](#)]
61. Witten, I.H.; Frank, E.; Hall, M.A.; Pal, C.J. *Data Mining: Practical Machine Learning Tools and Techniques*; Morgan Kaufmann: Cambridge, MA, USA, 2017.
62. 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. *Inf. Fusion* **2019**, *47*, 88–101. [[CrossRef](#)]
63. Galar, M.; Fernandez, A.; Barrenechea, E.; Bustince, H.; Herrera, F. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2012**, *42*, 463–484. [[CrossRef](#)]
64. García, V.; Mollineda, R.A.; Sánchez, J.S. A bias correction function for classification performance assessment in two-class imbalanced problems. *Knowl. Based Syst.* **2014**, *59*, 66–74. [[CrossRef](#)]

