Evolutionary Undersampling for Classification with Imbalanced Datasets: Proposals and Taxonomy

Salvador García salvagl@decsai.ugr.es
Department of Computer Science and Artificial Intelligence, University of Granada, Granada, 18071, Spain

Francisco Herrera herrera@decsai.ugr.es
Department of Computer Science and Artificial Intelligence, University of Granada, Granada, 18071, Spain

Abstract

Learning with imbalanced data is one of the recent challenges in machine learning. Various solutions have been proposed in order to find a treatment for this problem, such as modifying methods or the application of a preprocessing stage. Within the preprocessing focused on balancing data, two tendencies exist: reduce the set of examples (undersampling) or replicate minority class examples (oversampling). Undersampling with imbalanced datasets could be considered as a prototype selection procedure with the purpose of balancing datasets to achieve a high classification rate, avoiding the bias toward majority class examples. Evolutionary algorithms have been used for classical prototype selection showing good results, where the fitness function is associated to the classification and reduction rates. In this paper, we propose a set of methods called evolutionary undersampling that take into consideration the nature of the problem and use different fitness functions for getting a good trade-off between balance of distribution of classes and performance. The study includes a taxonomy of the approaches and an overall comparison among our models and state of the art undersampling methods. The results have been contrasted by using nonparametric statistical procedures and show that evolutionary undersampling outperforms the nonevolutionary models when the degree of imbalance is increased.

Keywords

Classification, class imbalance problem, undersampling, prototype selection, evolutionary algorithms.

1 Introduction

In recent years, the class imbalance problem has been one of the emergent challenges in machine learning. The problem appears when the data presents a class imbalance, consisting of containing many more examples of one class than the other one (Chawla et al., 2004; Xie and Qiu, 2007). Many applications have appeared in learning with imbalanced domains, such as fraud detection, intrusion detection (Cieslak et al., 2006), and biological and medical identification (Cohen et al., 2006).

Usually, the instances are grouped into two classes: the majority or negative class, and the minority or positive class. The latter class, in imbalanced domains, usually

represents a concept with the same or greater interest than the negative class. A standard classifier might ignore the importance of the minority class because its representation inside the dataset is not strong enough. As a classic example, if the ratio of imbalance presented in the data is 1:100 (i.e., there is one positive instance in a total set of 100 instances), the error of ignoring this class is only 1%.

A large number of approaches have been proposed to deal with the class imbalance problem. These approaches can be divided into three groups, depending on the way they work.

1. At the algorithmic level, the approaches are *internal* approaches. This group of methods tries to adapt the decision threshold to impose a bias on the minority class or to improve the prediction performance by adjusting weights for each class. In any case, they are based on modifying previous algorithms or making specific proposals for dealing with imbalanced data (Grzymala-Busse et al., 2005; Huang et al., 2006). Recently, in the field of evolutionary learning, some studies have been presented analyzing the behavior of XCS (Bernadó-Mansilla and Garrell-Guiu, 2003; Kovacs and Kerber, 2006; Butz et al., 2006) for imbalanced datasets (Orriols-Puig and Bernadó-Mansilla, 2006).

2. At the data level, the approaches are *external* approaches. This group of methods does not consist of modifying existing algorithms. On the contrary, they consist of resampling the data in order to decrease the effect caused by the imbalance of data (Batista et al., 2004). The main advantage of these techniques is that they are independent of the classifier used, so they can be considered as preprocessing approaches. A preliminary study by using evolutionary algorithms (EAs) as resampling for balancing data has been done (García et al., 2006), where we proposed a new evolutionary method. This study provides a method that uses a fitness function designed for performing a prototype selection process with the aim of balancing data, improving the generalization capability, and reducing the training data.

3. Combining the data and the algorithmic level, the approaches are *boosting* approaches. This set is composed of methods that consist of ensembles with the objective of improving the performance of weak classification algorithms. In boosting, the performance of weak classifiers is improved by means of focusing on hard examples that are difficult to classify. These approaches learn a way of combining several classifiers by using weighted examples in order to focus attention on hard examples. Thus, they preprocess the data through the incorporation of weights. In imbalanced data, as well as handling weights associated with each hard example, the replication of minority class instances is also used. Moreover, they constitute an adapted ensemble of classifiers developed depending on the data, so the algorithms are also modified for obtaining appropriate models to tackle imbalanced domains. Two main approaches have been developed with promising results in this group: the SMOTEBoost approach (Chawla et al., 2003) and the DataBoost-IM approach (Guo and Viktor, 2004).

Resampling approaches can be categorized into two tendencies: *undersampling*, which consists of reducing the data by eliminating examples belonging to the majority class with the objective of equalizing the number of examples of each class; and *oversampling*, which aims to replicate or generate new positive examples in order to gain importance (Chawla et al., 2002).

In specialized literature, we can find papers for resampling techniques from the point of view of studying the effect of the class distribution in classification (Weiss and Provost, 2003; Estabrooks et al., 2004) and adaptations of Prototype Selection (PS) methods (Wilson and Martinez, 2000) to work with imbalanced datasets (Kubat and Matwin, 1997; Batista et al., 2004).

Data may be categorized depending on its *imbalance ratio* (IR), which is defined as the relation between the majority class and minority class instances, by the expression

$$IR = \frac{N^-}{N^+}, \tag{1}$$

where N^- is the number of instances belonging to the majority class, and N^+ is the number of instances belonging to the minority class. Logically, a dataset is imbalanced when its IR is greater than 1. We will consider that an IR above 9 represents a high IR in a dataset, due to the fact that ignoring the minority class instances by a classifier supposes an error of 0.1 in accuracy, which has poor relevance. We will separately study the datasets belonging to this group in order to analyze the behavior of the proposals over them, given that when datasets present a high IR, the difficulty of the learning process increases.

EAs have been used for data reduction with promising results. They have been successfully used for feature selection (Whitley et al., 1998; Guerra-Salcedo and Whitley, 1998; Guerra-Salcedo et al., 1999; Papadakis and Theocharis, 2006; Wang et al., 2007; Sikora and Piramuthu, 2007) and PS (Cano et al., 2003, 2005), calling this last one the evolutionary prototype selection (EPS). EAs also have a good behavior for training set selection in terms of getting a trade-off between precision and interpretability with classification rules (Cano et al., 2007).

PS is an instance reduction process whose results are used as a reference set of examples for the nearest neighbor rule (1-NN) in order to classify new patterns. This reduces the number of rows in the dataset with no loss of classification accuracy and even has an improvement in the classifier. Various approaches of PS algorithms were proposed in the literature (see Wilson and Martinez, 2000, for review). A distinction is needed among those methods that are centered on an efficient selection of prototypes in order to increase or maintain the global accuracy rate and to reduce the size of the training data, and those that are focused on balancing data by selecting samples in order to prevent bad behaviors in a subsequent classification process.

In this paper, we propose the use of EAs for undersampling imbalanced datasets, which we call the evolutionary undersampling (EUS) approach. The objective is to increase the accuracy of the classifier by means of reducing instances mainly belonging to the majority class. In fact, the fitness functions proposed are designed to achieve a good trade-off between reduction, data balancing, and accuracy in classification. We propose eight EUS methods and categorize them into a taxonomy depending on their objective, scheme of selection, and metrics of performance employed.

We will distinguish two levels of imbalanced degree among datasets. A high degree of imbalance may have a remarkable influence on performance in a classification task and may cause problems in preprocessing stages carried out by many algorithms at the data level. For this reason, we analyze the use of the EUS method under these conditions by empirically comparing different methods among themselves and arranging them into a taxonomy. In addition to this, we compare our approach to other

Table 1: Confusion Matrix for Two-Class Problem

	Positive prediction	Negative prediction
Positive class	True positive (TP)	False negative (FN)
Negative class	False positive (FP)	True negative (TN)

undersampling methods studied in the literature. The empirical study has been contrasted via nonparametrical statistical testing.

The rest of the paper is organized as follows: Section 2 gives an explanation about issues on evaluating imbalanced learning. In Section 3, the EPS concepts are explained, together with a description of each model used. Section 4 gives the proposed taxonomy and expands the description of all the proposed EUS models. In Sections 5 and 6, the experimentation framework and the results and their analysis are presented. Finally, in Section 7, we present our conclusion. Appendix A contains a complete description of undersampling methods focused on balancing data and prototype selection methods.

2 How to Evaluate a Classifier in Imbalanced Domains?

When we want to evaluate a classifier over imbalanced domains, classical ways of evaluating, such as classification accuracy, have no sense. A standard classifier that uses an accuracy rate may be biased toward the majority class due to the bias inherent in the measure, which is directly related to the ratio between the number of instances of each class. A typical example of this fact is the following: if the ratio of imbalance presented in the data is 1:100, the error of ignoring this class is only 1%.

The most correct way of evaluating the performance of classifiers is based on the analysis of the confusion matrix. In Table 1, a confusion matrix is illustrated for a problem of two classes, with the values for the positive and negative classes. From this matrix it is possible to extract a number of widely used metrics to measure the performance of learning systems, such as *Error Rate*, defined as $Err = \frac{FP+FN}{TP+FN+FP+TN}$ and *Accuracy*, defined as $Acc = \frac{TP+TN}{TP+FN+FP+TN} = 1 - Err$.

In relation to the use of error (or accuracy) rate, another type of metric in the domain of the imbalanced problems is considered more correct. Concretely, from Table 1 it is possible to obtain four metrics of performance that measure the classification performance for the positive and negative classes independently.

1. The false negative rate $FN_{rate} = \frac{FN}{TP+FN}$ is the percentage of true positive cases misclassified as negative.
2. The false positive rate $FP_{rate} = \frac{FN}{FP+TN}$ is the percentage of true negative cases misclassified as positive.
3. The true negative rate $TN_{rate} = \frac{TN}{FP+TN}$ is the percentage of true negative cases correctly classified as negative.
4. The true positive rate $TP_{rate} = \frac{TP}{TP+FN}$ is the percentage of true positive cases correctly classified as positive.

The goal of a classifier is to minimize the false positive and false negative rates or, in a similar way, to maximize the true positive and true negative rates.

In Barandela et al. (2003) a metric called the *geometric mean (GM)* was introduced, defined as $g = \sqrt{a^+ \cdot a^-}$, where a^+ denotes the accuracy in positive examples (TP_{rate}), and a^- is the accuracy in negative examples (TN_{rate}). With this measure we try to maximize accuracy in order to balance both classes at the same time. This evaluation measure allows Barandela et al. to simultaneously maximize the accuracy in positive and negative examples with a favorable trade-off.

Another metric that could be used to measure the performance of classification over imbalanced datasets is the receiver operating characteristic (ROC) graphic (Bradley, 1997). In this graphic, the relationship between FN_{rate} and FP_{rate} can be visualized. The area under the ROC curve (AUC) corresponds to the probability of correctly identifying which of the two stimuli is noise and which is signal plus noise. AUC provides a single-number summary for the performance of learning algorithms.

Working with imbalanced domains and resampling techniques, the ROC analysis can be carried out by using a parameter of quantity of sampling that indicates the IR desired at the end of the preprocessing task (Chawla et al., 2002). In this paper, most of the methods evaluated do not allow for an adjustment of this parameter given that the IR obtained cannot be previously defined as a parameter.

We have used two measures to evaluate the performance of the methods studied in this paper: GM and AUC.

3 Evolutionary Prototype Selection: Representation and Fitness Function

Let us assume that there is a training set TR with N instances that consists of pairs $(x_i, y_i), i = 1, ..., N$, where x_i defines an input vector of attributes and y_i defines the corresponding class label. Each of the N instances has M input attributes and they should belong to a positive or a negative class. Let $S \subseteq TR$ be the subset of selected instances resulting from the execution of an algorithm.

PS can be considered as a search problem in which EAs can be applied. To accomplish this, we take into account two important issues: the specification of the representation of the solutions and the definition of the fitness function.

1. *Representation* The search space associated is constituted by all the subsets of TR. This is accomplished by using a binary representation. A chromosome consists of N genes (one for each instance in TR) with two possible states: 0 and 1. If the gene is 1, its associated instance is included in the subset of TR represented by the chromosome. If it is 0, this does not occur (Kuncheva and Bezdek, 1998).

2. *Fitness Function* Let S be a subset of instances of TR and be coded by a chromosome. Classically, we define a fitness function that combines two values: the classification rate ($clas\_rat$) associated with S and the percentage of reduction ($perc\_red$) of instances of S with regard to TR (Cano et al., 2003).

$$Fitness(S) = \alpha \cdot clas\_rat + (1 - \alpha) \cdot perc\_red. \tag{2}$$

The 1-NN classifier is used for measuring the classification rate, $clas\_rat$, associated with S. It denotes the percentage of correctly classified objects from TR using only S to find the nearest neighbor. For each object y in S, the nearest neighbor is

searched for among those in the set $S \setminus \{y\}$. In contrast, $perc\_red$ is defined as

$$perc\_red = 100 \cdot \frac{|TR| - |S|}{|TR|}. \tag{3}$$

The objective of the EAs is to maximize the fitness function defined, that is, maximize the classification rate and minimize the number of instances obtained. The EAs with this fitness function will be denoted by the extension PS in the name.

We also consider the crossover operation for data reduction. In order to achieve a good reduction rate, heuristic uniform crossover (HUX) implemented for CHC undergoes a change that makes it more difficult to include instances inside the selected subset. Therefore, if an HUX switches a bit on in a gene, then the bit could be switched off depending on a certain probability (which will be specified in Section 5.1).

As the evolutionary computation method in the core of EPS, we have used the CHC model (Eshelman, 1991; Cano et al., 2003) and the intelligent genetic algorithm (IGA; Ho et al., 2002).

3.1 CHC

CHC is a classical evolutionary model that introduces different features to obtain a trade-off between exploration and exploitation; such as incest prevention, reinitialization of the search process when it becomes blocked, and the competition between parents and offspring in the replacement process. Recently, CHC has been used in many applications; for example, as a method for optimizing learning models (Alcalá et al., 2007), information processing (Alba et al., 2006), and image registration (Cordón et al., 2006).

During each generation the CHC develops the following steps.

1. It uses a parent population of size N to generate an intermediate population of N individuals that are randomly paired and used to generate N potential offspring.

2. A survival competition is held where the best N chromosomes from the parent and offspring populations are selected to form the next generation.

CHC also implements a form of heterogeneous recombination using HUX, a special recombination operator. HUX exchanges half of the bits that differ between parents, where the bit position to be exchanged is randomly determined. CHC also employs a method of incest prevention. Before applying HUX to the two parents, the Hamming distance between them is measured. Only those parents that differ from each other by some number of bits (mating threshold) are mated. The initial threshold is set at $L/4$, where L is the length of the chromosomes. If no offspring are inserted into the new population, then the threshold is reduced by one.

No mutation is applied during the recombination phase. Instead, when the population converges or the search stops making progress (i.e., the difference threshold has dropped to zero and no new offspring are being generated that are better than any member of the parent population) the population is reinitialized to introduce new diversity to the search. The chromosome representing the best solution found over the course of the search is used as a template to reseed the population. Reseeding of the population is accomplished by randomly changing 35% of the bits in the template chromosome to form each of the other $N - 1$ new chromosomes in the population. The search is then resumed.

3.2 IGA

IGA is a generational genetic algorithm (GGA) that incorporates an intelligent crossover (IC) operator. IC builds an orthogonal array (OA; see Ho et al., 2002) from two parents of chromosomes and searches within the OA to find the two best individuals according to the fitness function. It takes about $2^{\lceil \log_2(\gamma+1) \rceil}$ fitness evaluations to perform an IC operation, where γ is the number of bits that differs between both parents. IGA is based on orthogonal experimental design used for PS and feature selection.

IGA is a GGA with elitist strategy in initialization, evaluation, selection, and mutation. It randomly generates N individuals at the beginning. The selection uses the rank that replaces the worst $P_s N$ by the best $P_s N$ individuals to form a new population, where P_s is a selection probability. It randomly selects $P_c N$ individuals to perform IC, where P_c is the crossover probability. The mutation operator is the conventional bit inverse mutation operator. The best individual is retained without being subject to the mutation operator. The algorithm finishes when the number of evaluations achieves a certain value.

The IC operator is based upon the development of an OA that consists of a set of orthogonal representations obtained from two chromosomes. Once the OA is constructed, the IC evaluates all candidates belonging to it and returns the best and the second best individuals. The development of the OA is expensive and the algorithm for doing it is detailed elsewhere (Ho et al., 2002).

4 Evolutionary Undersampling: Models and Taxonomy

We will present a taxonomy for EUS methods, identifying the main issues used for the classification of the respective models and including each method in its corresponding place. We will use two ways of division, the objective that they pursue and the way that they do the selection of instances.

Regarding the objective, there are two goals of interest in EUS:

1. Aiming for an optimal balancing of data without a loss of effectiveness in classification accuracy. EUS models that follow this tendency will be called evolutionary balancing undersampling (EBUS).

2. Aiming for an optimal power of classification without taking into account the balancing of data, considering the latter as a subobjective that may be an implicit process. EUS models that follow this tendency will be called evolutionary undersampling guided by classification measures (EUSCM).

With respect to the types of instance selection that can be carried out in EUS, we distinguish:

- If the selection scheme proceeds over any kind of instance, then it is called global selection (GS). That is, the chromosome contains the state of all instances belonging to the training dataset and removals of minority class instances (those belonging to the positive class) are allowed.

- If the selection scheme only proceeds over majority class instances, then it is called majority selection (MS). In this case, the chromosome saves the state of instances that belong to the negative class and a removal of a positive or minority class instance is not allowed.

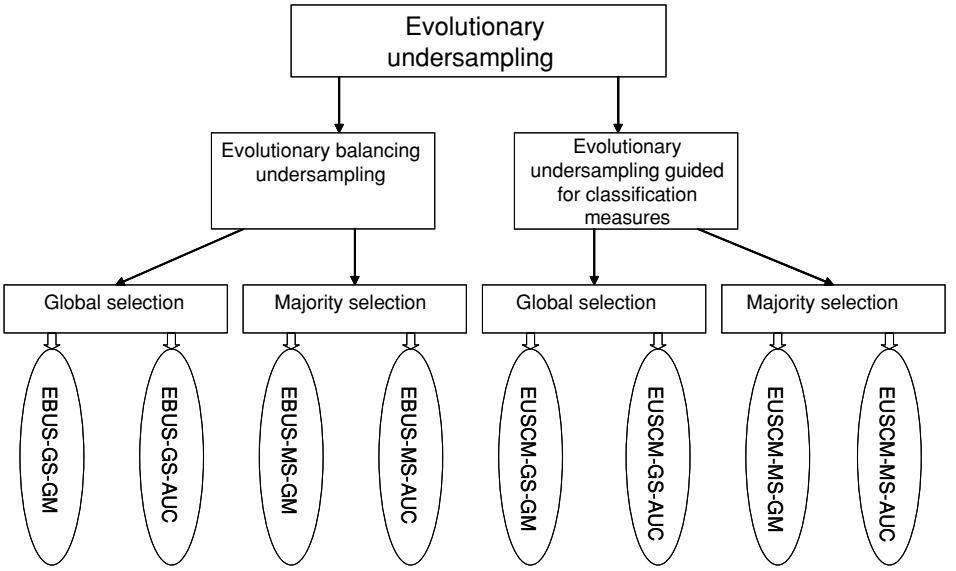

Figure 1: Evolutionary undersampling taxonomy.

This categorization produces four subgroups of EUS methods. Furthermore, two methods that differ in the measure of evaluation are included in each group (GM and AUC), obtaining a total number of eight EUS methods.

These methods will be described in the following subsections. They can be easily distinguished by their names.

- If the term *GM* appears, then it means that the model uses the geometric mean as the evaluator of accuracy. The other case uses the term *AUC* for evaluation with an AUC measure.
- If the term *GS* appears, this implies that the selection scheme is global selection, as opposed to *MS*, which implies that the selection scheme is majority selection.
- The method will be a EBUS or a EUSCM model and this fact is specified in its name.

The EUS approaches have been developed by using the CHC evolutionary model.

Figure 1 summarizes the proposed taxonomy for EUS approach and includes the eight methods studied in this paper.

In the following two subsections, we will describe the EBUS and EUSCM models.

4.1 Evolutionary Balancing Undersampling

The EBUS subgroup contains four methods, EBUS-GS-GM, EBUS-MS-GM, EBUS-GS-AUC, and EBUS-MS-AUC.

4.1.1 EBUS-GS-GM

This is the model used in García et al. (2006), consisting of applying the same fitness function defined in the expression in Equation (4) and the selection over the majority

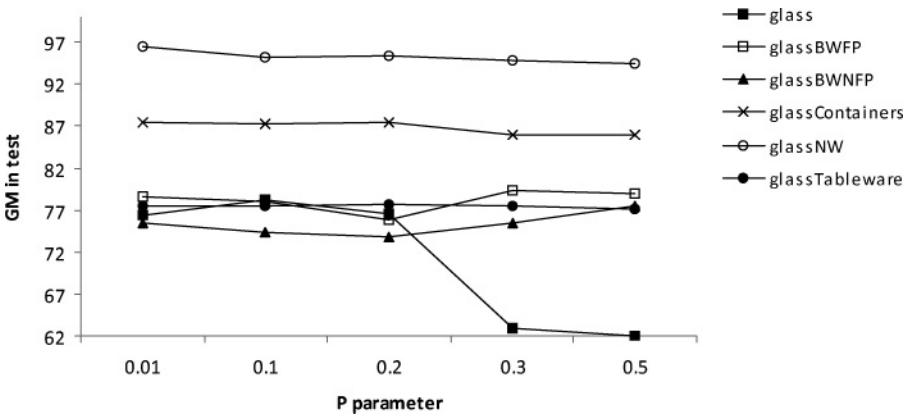

Figure 2: Influence of the P factor in EBUS-MS-GM.

and minority class samples simultaneously. This model aims to remove instances of both classes, identifying minority class examples that have a negative influence over the classification task and achieving a maximal reduction in positive instances. A penalization factor used for preserving the same number of instances belonging to each class helps to maintain a generalization capability in the reduction task, in the way it does not specialize the data subset only for the positive class. The fitness function for EBUS-GS-GM is

$$Fitness_{Bal}(S) = \begin{cases} g - |1 - \frac{n^+}{n^-}| \cdot P & \text{if } n^- > 0 \\ g - P & \text{if } n^- = 0 \end{cases} \quad (4)$$

where g is geometric mean of balanced accuracy defined in Section 2, n^+ is the number of positive instances selected (minority class), n^- is the number of negative instances selected (majority class), and P is the penalization factor.

The P parameter is considered an influential value that controls the intensity and importance of the balance during the evolutionary search. It defines the maximum penalization applied over the classification measure if there were a total unbalancing between both classes, that is, either no positive instances are selected or no negative instances are selected. We have empirically determined that the penalization over the classification measure should be closer to 0.02 so that a low value does not sufficiently affect the achievement of the balance. Moreover, a high value implies that the trade-off between accuracy and balancing could be lost. So, the parameter P value that we will use is 0.2.

Figures 2 and 3 represent the GM accuracy when parameter P is set from 0.01 to 0.5, in the EBUS model with majority (which will be detailed in the next section) and global selection, respectively. In this case study, we have used the imbalanced dataset derived from the Glass dataset (see Table 2), which is composed of six versions with different IR values. The graphics show how employing low and high P values could lead to extremely bad results using some datasets. A value of $P = 0.1$ or $P = 0.2$ remains stable over all datasets.

This fitness function tries to find subsets of instances making a trade-off between the classification balanced accuracy and an equal number of examples selected from

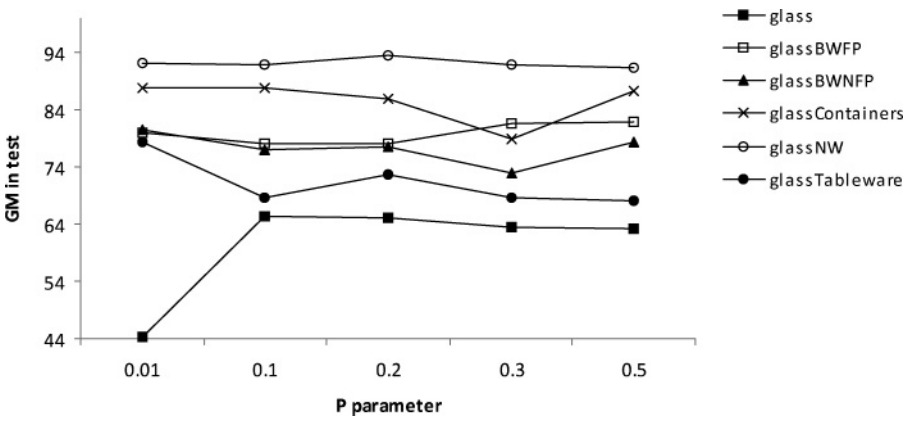

Figure 3: Influence of the P factor in EBUS-GS-GM.

each class. This second objective is obtained through the penalization applied to g in the fitness value equation.

4.1.2 EBUS-MS-GM

EBUS-MS-GM is the same model as before, but it only selects instances belonging to the negative class (i.e., majority class samples). The fitness function corresponds to the expression in Equation (5). With this scheme, the reduction only affects the negative instances, but the process is controlled in the same way as the previous model; thus, the reduction is not free, it depends on the minority class examples.

$$Fitness_{Bal}(S) = \begin{cases} g - |1 - \frac{N^+}{n^-}| \cdot P & \text{if } n^- > 0 \\ g - P & \text{if } n^- = 0 \end{cases} \tag{5}$$

Given that the instances belonging to the positive class are not affected in this model, N^+ is a constant that represents the number of original positive instances within the training dataset.

4.1.3 EBUS-GS-AUC

The EBUS-GS-AUC model is obtained from the EBUS-GS-GM model by replacing the geometric mean to measure the accuracy of the training data using the AUC measure (see Section 2). The fitness function corresponds to the expression in Equation (6).

$$Fitness_{Bal}(S) = \begin{cases} AUC - |1 - \frac{n^+}{n^-}| \cdot P & \text{if } n^- > 0 \\ AUC - P & \text{if } n^- = 0 \end{cases} \tag{6}$$

Although the AUC measure is totally valid to achieve a good balance between accuracy in both classes, it does not control the resulting balance of instances selected in both classes.

Table 2: Imbalanced Datasets

Dataset	#Examples	#Attributes	Class (min., maj.)	%Class(min.,maj.)	IR
GlassBWNFP	214	9	(build-window-(non_float-proc, remainder)	(35.51, 64.49)	1.82
EcoliCP-IM	220	7	(im, cp)	(35.00, 65.00)	1.86
Pima	768	8	(1, 0)	(34.77, 66.23)	1.9
GlassBWFP	214	9	(build-window-float-proc, remainder)	(32.71, 67.29)	2.06
German	1,000	20	(1, 0)	(30.00, 70.00)	2.33
Haberman	306	3	(Die, Survive)	(26.47, 73.53)	2.68
Splice-ie	3,176	60	(ie, remainder)	(24.09, 75.91)	3.15
Splice-ei	3,176	60	(ei, remainder)	(23.99, 76.01)	3.17
GlassNW	214	9	(non-windows glass, remainder)	(23.93, 76.17)	3.19
VehicleVAN	846	18	(van, remainder)	(23.52, 76.48)	3.25
EcoliIM	336	7	(im, remainder)	(22.92, 77.08)	3.36
New-thyroid	215	5	(hypo, remainder)	(16.28, 83.72)	4.92
Segment1	2,310	19	(1, remainder)	(14.29, 85.71)	6.00
EcoliIMU	336	7	(iMU, remainder)	(10.42, 89.58)	8.19
Optdigits0	5,564	64	(0, remainder)	(9.90, 90.10)	9.10
Satimage4	6,435	36	(4, remainder)	(9.73, 90.27)	9.28
Vowel0	990	13	(0, remainder)	(9.01, 90.99)	10.1
GlassVWFP	214	9	(Ve-win-float-proc, remainder)	(7.94, 92.06)	10.39
EcoliOM	336	7	(om, remainder)	(6.74, 93.26)	13.84
GlassContainers	214	9	(containers, remainder)	(6.07, 93.93)	15.47
Abalone9-18	731	9	(18, 9)	(5.75, 94.25)	16.68
GlassTableware	214	9	(tableware, remainder)	(4.2, 95.8)	22.81
YeastCYT-POX	483	8	(POX, CYT)	(4.14, 95.86)	23.15
YeastME2	1,484	8	(ME2, remainder)	(3.43, 96.57)	28.41
YeastME1	1,484	8	(ME1, remainder)	(2.96, 97.04)	32.78
YeastEXC	1,484	8	(EXC, remainder)	(2.49, 97.51)	39.16
Car	1,728	6	(good, remainder)	(3.99, 96.01)	71.94
Abalone19	4,177	9	(19, remainder)	(0.77, 99.23)	128.87

4.1.4 EBUS-MS-AUC

Using AUC as the performance measure, the EBUS-MS-AUC model does not remove examples belonging to the positive class. The fitness function corresponds to the expression in Equation (7).

$$Fitness_{Bal}(S) = \begin{cases} AUC - |1 - \frac{N^+}{n^-}| \cdot P & \text{if } n^- > 0 \\ AUC - P & \text{if } n^- = 0 \end{cases} \quad (7)$$

4.2 Evolutionary Undersampling Guided by Classification Measures

This subgroup is composed of four methods of EUSCM, namely, EUSCM-GS-GM, EUSCM-MS-GM, EUSCM-GS-AUC, and EUSCM-MS-AUC.

4.2.1 EUSCM-GS-GM

EUSCM-GS-GM is the same model as the first one, but no penalization factor (P) is applied during the selection, implying that the balancing between classes is not expected. The fitness function corresponds to the expression in Equation (8). Usually, the objective of this model is to select a minimal number of examples representing the whole training dataset. The disadvantage of lack of generalization capability may be pointed out in this model.

$$Fitness(S) = g \tag{8}$$

4.2.2 EUSCM-MS-GM

In the EUSCM-MS-GM model, the selection is applied over negative instances. The fitness function can be represented by the expression in Equation (8). Note that the penalization factor P is not included in the fitness function. This implies that the difference of the number of instances among both classes does not tend to be equal to 0, so there is no control over the selection process in terms of getting a balance. However, control over negative instances is present, given that removing instances only affects the majority class.

4.2.3 EUSCM-GS-AUC

The EUSCM-GS-AUC model is obtained from the EUSCM-MS-GM model described previously by using the AUC instead of the geometric mean to measure the accuracy of the training data (see Section 2). The fitness function corresponds to the expression in Equation (9).

$$Fitness(S) = AUC \tag{9}$$

The AUC measure is involved in a trade-off between improving the accuracy rate over positive instances and not losing accuracy rate power over negative instances.

4.2.4 EUSCM-MS-AUC

The EUSCM-MS-AUC model is the same as the previous model, with the exception of the selection carried out, which is only performed in examples belonging to the majority class. The fitness function used corresponds to the expression in Equation (9).

5 Experimental Framework

This section describes the methodology followed in the experimental study of the undersampling methods compared. We will explain the configuration of the experiment including the datasets used and the parameters for the algorithms. The PS methods used in the study and the proposals of undersampling found in more specialized literature are (for a detailed description, see Appendix A) the prototype selection methods, and the classical undersampling methods for balancing class distribution.

Prototype Selection Methods

- Instance-based 3 (IB3; Aha et al., 1991) is an incremental instance selection algorithm that introduces the *acceptable* concept in the selection.

- Decremental reduction optimization procedure 3 (DROP3; Wilson and Martinez, 2000) is based on the rule "Any instance incorrectly classified by its *k*-NN is removed."

Classical Undersampling Methods for Balancing Class Distribution

- Random undersampling (RUS) is a nonheuristic method that aims to balance class distribution through the random elimination of majority class examples to get a balanced instance set.
- Tomek links (TL; Tomek, 1976) searches for Tomek links and eliminates examples belonging to the majority class in each Tomek link found.
- Condensed nearest neighbor rule (US-CNN; Hart, 1968) is a modification of the classic CNN rule for imbalanced learning.
- One-sided selection (OSS; Kubat and Matwin, 1997) is an undersampling method resulting from the application of Tomek links followed by the application of US-CNN.
- US-CNN + TL (Batista et al., 2004) is similar to OSS, but the method US-CNN is applied before the Tomek links.
- Neighborhood cleaning rule (NCL; Laurikkala, 2001) is an adaptation of the ENN rule for imbalanced learning.
- Class purity maximization (CPM; Yoon and Kwek, 2005) is a clustering based method for imbalanced learning that manages the *impurity* concept.
- Undersampling based on clustering (SBC; Yen and Lee, 2006) is a random undersampling based on clustering.

Finally, we will briefly introduce the use of nonparametric statistical tests employed for the comparison of the results obtained.

5.1 Configuration of the Experiment

Performance of PS methods and undersampling for balancing data, which will be described in Appendix A, is analyzed by using 28 datasets taken from the UCI machine learning database repository (Newman et al., 1998). Multiclass datasets are modified to obtain two-class nonbalanced problems, defining one class as positive and one or more classes as negative. Missing values have been replaced with the lowest possible value of the attribute associated domain.

The datasets are sorted by their IR values in an incremental way. The main characteristics of these datasets are summarized in Table 2. For each dataset, the table gives the number of examples (#Examples), number of attributes (#Attributes), and class name (minority and majority) together with the class distribution and the IR value. Some of these datasets have already been used in previous works (Weiss and Provost, 2003; Batista et al., 2004; Guo and Viktor, 2004; Akbani et al., 2004).

The datasets considered are partitioned using the *10fold cross-validation (10-fcv)* procedure. The parameters of the used algorithms (see Appendix A for detailed descriptions of PS and undersampling methods) are presented in Table 3. The EUS approach

Table 3: Parameters Considered for the Algorithms

Algorithm	Parameters
IB3	$Acept.\,Level = 0.9,\ Drop\,Level = 0.7$
EPS-CHC	$Pop = 50,\ Eval = 10{,}000,\ \alpha = 0.5$
EPS-IGA	$Pop = 10,\ Eval = 10{,}000,\ \alpha = 0.5$
RUS	$Balancing\ Ratio = 1:1$
SBC	$Balancing\ Ratio = 1:1,\ N.\,Clusters = 3$
EUS	$Pop = 50,\ Eval = 10{,}1000,\ P = 0.2,\ Prob.\,inclusion\ HUX = 0.25$

always uses the same parameters independent of the fitness function it considers, in order to make them more comparable in performance. All the parameters for the algorithms are the ones recommended by their respective authors.

5.2 Nonparametric Statistical Tests for Statistical Analysis

In this paper we have used a *10-fcv*, which is a way of estimating the real error of a classifier by testing it against all instances of the dataset, while training the classifier with instances independent of the testing instances. The results obtained from this validation are not completely independent, therefore the results neither present normal distribution nor homogeneity of variance. In this situation, we consider the use of nonparametric tests, according to the recommendations made in Demšar (2006).

As such, these nonparametric tests can be applied to classification accuracies, error ratios, or any other measure for technique evaluation, even including model sizes and computation times. Empirical results suggest that they are also more powerful than the parametric tests. Demšar recommends a set of simple, safe, and robust nonparametric tests for statistical comparisons of classifiers. We will briefly describe the two tests used in this study.

The first one is Friedman's test (Sheskin, 2003), which is a nonparametric test equivalent to the repeated-measures ANOVA. The null hypothesis is that all the algorithms are equivalent, so a rejection of this hypothesis implies the existence of differences among the performance of all the algorithms studied. After this, a post hoc test could be used in order to find whether the control or proposed algorithm presents statistical differences with regard to the remaining methods into the comparison. The simplest of them is the Bonferroni-Dunn test, but we can use more powerful tests that control the familywise error rate and reject more hypotheses than the Bonferroni-Dunn test does; one example that accomplishes this is Holm's test.

Due to the fact that Friedman's test can be too conservative, we have used a derivation of it, Iman and Davenport's test (Iman and Davenport, 1980). The descriptions and computations of the tests are explained in Demšar (2006).

As a post hoc test of the Friedman statistic, we use Holm's procedure (Holm, 1979), which is a multiple comparison procedure that works with a control algorithm (normally, the best procedure is chosen) and compares it to the remaining methods. The results obtained in each comparison by using Holm's procedure will be reported through p values.

6 Experimental Results and Analysis

This section shows the experimental results and their associated statistical analysis in the evaluation of the imbalanced datasets. It is divided into three parts, corresponding

Table 4: Average Results for PS Algorithms over Imbalanced Datasets

	PS Method	$\%Red$	$\%Red^-$	$\%Red^+$	a^-_{tra}	a^+_{tra}	GM_{tra}	a^-_{tst}	a^+_{tst}	GM_{tst}	AUC
Mean	None	0.0	0.0	0.0	0.9399	0.6414	0.7485	0.9387	0.6175	**0.6958**	**0.7606**
SD		0.0	0.0	0.0	0.1832	0.1514	0.1635	0.1831	0.1485	0.1576	0.1648
Mean	IB3	61.82	62.00	80.60	0.9196	0.475	0.5965	0.9227	0.4615	0.5267	0.6746
SD		1.49	1.49	1.70	0.1812	0.1302	0.146	0.1815	0.1284	0.1372	0.1552
Mean	DROP3	91.76	94.00	78.13	0.8879	0.7027	0.7657	0.8761	**0.6299**	0.6751	0.7359
SD		1.81	1.83	1.67	0.1781	0.1584	0.1654	0.1769	0.15	0.1553	0.1621
Mean	EPS-CHC	**98.85**	**99.00**	**97.17**	0.9735	0.5747	0.6757	**0.9584**	0.5183	0.6037	0.7206
SD		1.88	1.88	1.86	0.1865	0.1433	0.1553	0.185	0.1361	0.1468	0.1604
Mean	EPS-IGA	89.49	90.00	81.91	**0.9767**	**0.7206**	**0.8044**	0.9459	0.5919	0.6691	0.7516
SD		1.79	1.80	1.71	0.1868	0.1604	0.1695	0.1838	0.1454	0.1546	0.1638

to different studies that aim to achieve a certain conclusion. The objectives of the three parts are as follows. Section 6.1 shows a study applying and evaluating PS methods over imbalanced data. Section 6.2 compares the approaches of EUS proposed among themselves. Section 6.3 includes a study between the most promising EUS model and the algorithms of undersampling focused on balancing data obtained from the state of the art.

6.1 Using PS Methods over Imbalanced Domains

In Section 2, the metric of GM was described as a good way of evaluating the performance of classifiers over imbalanced domains. Using a preprocessing stage with the aim of improving the performance of a posterior classifier should obtain a better GM rate than not using it.

This first study is composed of a comparison among the four PS methods considered in this study and the classifier 1-NN without preprocessing.

Table 4 shows us the average and standard deviations of the results offered by the PS algorithms over the imbalanced datasets. Each column shows:

- The PS method employed.
- The percentage of reduction with respect to the original dataset size. Furthermore, the percentage of reduction associated with each class is shown in posterior columns.
- The accuracy for each class by using an 1-NN classifier (a^+ and a^-), where subindex tra refers to training data and subindex tst refers to test data. The GM value is also shown for training and test data.
- Finally, the AUC measure in test data is reported.
- *None* indicates that no balancing method is employed (i.e., the original dataset is used for classification with 1-NN).

Table 4 reports that, in general, all PS methods lose accuracy and AUC in test data, given that the usage of this preprocessing stage performs worse than 1-NN. EPS-CHC

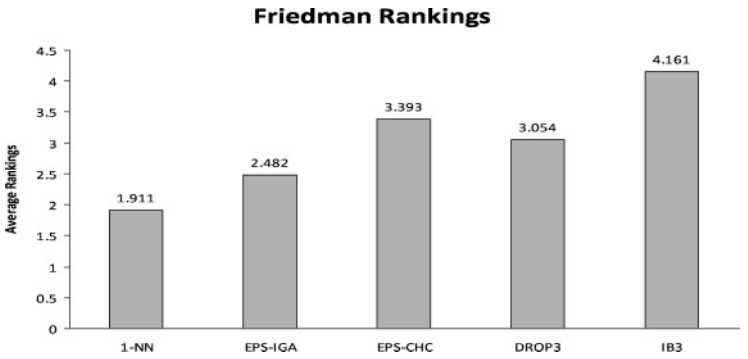

Figure 4: Friedman rankings for classical PS and EPS algorithms.

Table 5: Statistics and Critical Values for Friedman's and Iman-Davenport's Test

Friedman's statistic	Critical value	Hypothesis
33.143	9.488	Rejected
Iman-Davenport statistic	Critical value	Hypothesis
11.348	2.456	Rejected

is the algorithm that obtains the highest reduction rate and EPS-IGA obtains the best result in training data, indicating to us that it overfits the selected instances to the training data.

In Figure 4, the values of the average rankings using Friedman's method are specified. Each column represents the average ranking obtained by an algorithm; that is, if a certain algorithm achieves rankings 1, 3, 1, 4, and 2, on five datasets, then the average ranking is $\frac{1+3+1+4+2}{5} = \frac{11}{5}$. The height of each column is proportional to the ranking; the lower a column is, the better its associated algorithm is.

Next, we apply Friedman's and Iman-Davenport's tests (considering a level of significance $\alpha = .05$) to check whether differences exist among all the methods by using the GM measure, presenting the results in Table 5.

Table 5 indicates to us that both Friedman's and Iman-Davenport's statistics are higher than their associated critical value, so the hypothesis of equivalence of results is rejected. A post hoc test is then necessary to distinguish whether the control method (1-NN without preprocessing in this case) is significantly better than the other remaining control methods. We will apply the post hoc statistical analysis considering all datasets. We use Holm's procedure to check this over the GM measure, and the results are offered in Figure 5. Following the indications given in Section 5.2, this procedure computes the z_i value and obtains the associated p_i value by using the normal distribution for each hypothesis i to evaluate. The figure represents the p values associated with each comparison between 1-NN without preprocessing and the corresponding PS algorithm indicated in the x axis. The discontinuous line similar to a staircase represents the α/i value established for each comparison following Holm's method. If a p value of a certain comparison exceeds this line, this hypothesis cannot be rejected for the Holm test and it implies the need to stop checking the remaining hypotheses. Otherwise, when a p value does not exceed the discontinuous line, this implies the rejecting of the hypothesis associated and allows the next test to be done.

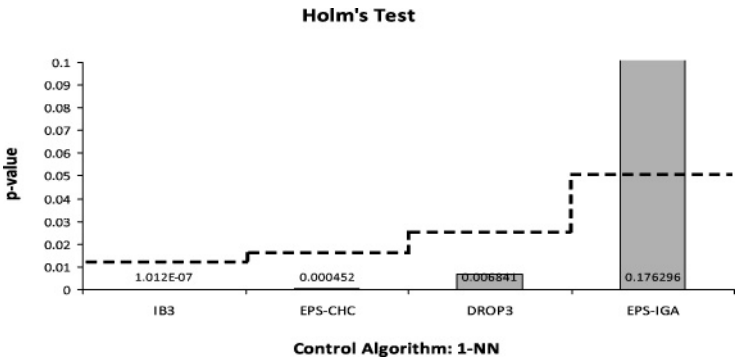

Figure 5: Holm's test: The control algorithm is 1-NN without preprocessing.

The statistical analysis of this comparison declares that the use of PS methods is not recommended for nonbalanced domains given that the accuracy of 1-NN is significantly better than the three PS methods studied. With respect to the EPS-IGA algorithm, we cannot state that 1-NN is better than EPS-IGA preprocessing, but EPS-IGA does not obtain a better value of ranking than 1-NN. The exhaustive search process performed by EPS-IGA and its capability of reduction (lower than EPS-CHC) of the training data allow subsets of instances to be found that overfit the training set but in test accuracy perform worse on imbalanced domains with respect to no use of a PS method. Note that the EPS-CHC algorithm notably performs poorly when we treat with highly imbalanced datasets, so the excessive reduction achieved by this method does not produce a benefit in imbalanced domains.

6.2 EUS Methods

As a second objective, we analyze all EUS models proposed over all imbalanced datasets. The eight algorithms that compose the taxonomy explained in Section 4 will be analyzed in terms of efficacy and efficiency in order to obtain the most appropriate configuration of EUS over a set of imbalanced datasets. Firstly, we will study the EUS models on the full set of data considered. Then, we will divide the datasets into two groups: those that have an IR below 9 and those that have an IR above 9. All the studies will include statistical analysis of the results.

Beginning with the study that considers the 28 imbalanced datasets, Table 6 shows us the average and standard deviations of the results offered by the models proposed. It follows the same structure as Table 4.

By analyzing Table 6, we can point out the following.

- The best average results are offered by the models EBUS, by measuring the performance with GM accuracy and AUC.
- An observable difference between the use of global selection and majority selection exists. In all cases, the majority selection is preferable to global selection.
- The employment of GM or AUC in the fitness does not overly affect the results obtained.

Table 6: Average Results for the Proposed Models over Imbalanced Datasets

	EUS Method	$\%Red$	$\%Red^-$	$\%Red^+$	a^-_{tra}	a^+_{tra}	GM_{tra}	a^-_{tst}	a^+_{tst}	GM_{tst}	AUC
Mean	EBUS-	70.04	81.00	0.00	0.8473	0.9323	0.8878	0.8289	**0.8189**	0.7955	0.8071
SD	MS-AUC	1.58	1.70	0.00	0.174	0.1825	0.1781	0.1721	0.171	0.1686	0.1698
Mean	EBUS-	69.93	80.00	0.00	0.8504	0.9252	0.8862	0.8319	0.8188	**0.7971**	**0.8085**
SD	MS-GM	1.58	1.69	0.00	0.1743	0.1818	0.1779	0.1724	0.171	0.1687	0.1699
Mean	EBUS-	**96.30**	**98.09**	82.12	0.8749	0.9259	0.8991	0.8566	0.7826	0.7872	0.8024
SD	GS-AUC	1.85	1.87	1.71	0.1768	0.1818	0.1792	0.1749	0.1672	0.1677	0.1693
Mean	EBUS-	96.23	98.00	82.13	0.8812	0.9195	0.8996	0.8595	0.7863	0.7927	0.8058
SD	GS-GM	1.85	1.87	1.71	0.1774	0.1812	0.1792	0.1752	0.1676	0.1683	0.1696
Mean	EUSCM-	76.86	90.00	0.00	0.8639	**0.9371**	0.8961	0.8285	0.8084	0.7795	0.8014
SD	MS-AUC	1.66	1.80	0.00	0.1757	0.1829	0.1789	0.172	0.1699	0.1669	0.1692
Mean	EUSCM-	76.18	89.00	0.00	0.8714	0.9313	0.8983	0.8354	0.8081	0.7861	0.805
SD	MS-GM	1.65	1.79	0.00	0.1764	0.1824	0.1791	0.1727	0.1699	0.1676	0.1696
Mean	EUSCM-	94.46	95.00	84.01	0.9144	0.9116	**0.9092**	**0.8916**	0.7374	0.7712	0.797
SD	GS-AUC	1.84	1.85	1.73	0.1807	0.1804	0.1802	0.1784	0.1623	0.166	0.1687
Mean	EUSCM-	94.34	95.00	**84.19**	**0.9155**	0.9054	0.9068	0.8894	0.7278	0.7575	0.7912
SD	GS-GM	1.84	1.84	1.73	0.1808	0.1798	0.18	0.1782	0.1612	0.1645	0.1681

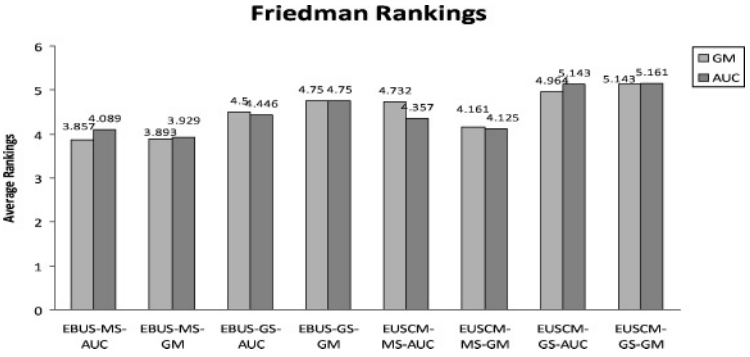

Figure 6: Friedman rankings for all EUS models over all imbalanced datasets.

We are interested in checking whether these differences are significant by using nonparametrical statistical tests. For this, we compute the average rankings by using Friedman's test over the results obtained for all imbalanced datasets, as well as for the results on datasets with $IR < 9$ and $IR > 9$. In Figures 6, 7, and 8 (they follow the same scheme as Figure 4), we represent the ranking values for each algorithm and for GM and AUC measures. With these values, we have computed Iman-Davenport's statistic (considering a level of confidence $\alpha = .05$) and the results are shown in Table 7.

Iman's and Davenport's multiple comparison test procedure cannot reject, in the majority of the cases, the hypothesis of equivalence of means, so we can conclude that significant differences do not exist among the distinct models of EUS studied. An exception is produced when we evaluate the models with GM over datasets with $IR > 9$. In this case, Iman-Davenport's test rejects the null hypothesis (we have proved

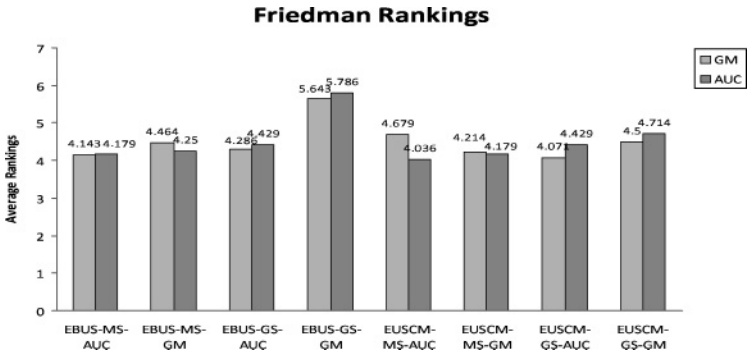

Figure 7: Friedman rankings for all EUS models over imbalanced datasets with $IR < 9$.

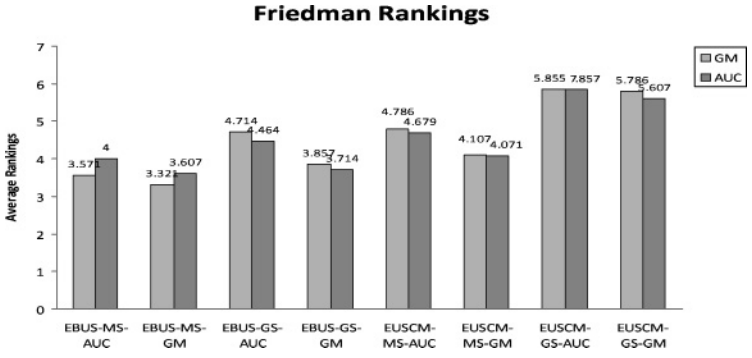

Figure 8: Friedman rankings for all EUS models over imbalanced datasets with $IR > 9$.

Table 7: Statistics and Critical Values for Iman-Davenport Test ($\alpha = .05$)

Imbalance datasets	Iman-Davenport statistic for GM	Iman-Davenport statistic for AUC	Critical value	Hypothesis
All	1.099	1.049	2.058	Both nonrejected
$IR < 9$	0.575	0.716	2.112	Both nonrejected
$IR > 9$	2.355	1.736	2.112	Rejected for GM measure

that Friedman's test also rejects it). Due to the fact that Iman-Davenport's test is more powerful than Friedman's test, if it is not able to reject the null hypothesis, Friedman's test cannot do it either.

An analysis based upon the results obtained from the rankings computed following the guidelines for Friedman's test allows us to state the following.

First, with respect to imbalanced datasets with $IR < 9$ (Figure 7), we find the three points that follow.

- The parameter addressed to balancing data (P factor) lacks interest when the data is not imbalanced enough. The EUSCM model obtains good results without

balancing mechanisms. Hence, in general, the EUSCM approach behaves better than EBUS.

- However, the best performing method is EBUS-MS-AUC, because it obtains low rankings in both measures, although it is not the best in GM (EUSCM-GS-AUC outperforms it) and in AUC (EUSCM-MS-AUC is the best in this case).
- The differences between the use of global and majority selection or GM and AUC in the fitness function do not follow a specific bias toward carrying out the best choice.

Second, with respect to imbalanced datasets with $IR > 9$ (Figure 8), we find the following.

- When the IR becomes high, a GS mechanism has no sense due to the reduced number of examples belonging to the minority class. Thus, the MS mechanism obtains better results than the GS mechanism.
- We can observe that EBUS models behave better than the EUSCM model. Therefore, a balancing mechanism may help the undersampling process over extreme circumstances of imbalance.
- In particular, an algorithm that belongs to the group of EBUS models with majority selection, that is, EBUS-MS-GM, is the best performing method in this case.

In spite of the conclusion obtained from Iman-Davenport's test, which is that there are not notable differences among the models, we have to choose a certain model for performing a comparison with the state of the art techniques in order to stress the benefit of using EUS. Thus, we will select the most accurate model, EBUS-MS-GM, which presents the best result in highly imbalanced datasets ($IR > 9$) and considering all of them (see Figure 6).

Finally, Figure 9 shows a set of bar charts that represent the runtime spent by each type of model of EUS on some datasets with different IRs. Obviously, runtime is influenced by the size of the datasets, due to the fact that chromosome size grows along with this increase in size. On the other hand, it is observably the fact that GS is less affected by the IR, and that MS is very influenced by it. In the *pima* dataset, with a low IR, the runtime of the EUS model with MS is high because of the evaluation cost of the minority class examples, which are retained in all evaluations. In EBUS, this fact is more notable due to the interest in balancing both classes. However, when IR is high (as in the case of the *yeastEXC* dataset), the EBUS-MS model is favored in efficiency by its interest in balancing.

6.3 EUS versus other Undersampling Methods

In this subsection we will compare the EBUS-MS-GM model with 1-NN (called none, as previously mentioned) and the remaining undersampling techniques.

Table 8 shows us the average and standard deviations of the results offered by each algorithm over the 28 imbalanced datasets considered.

The results offered in Table 8 suggest that the most accurate method is EBUS-MS-GM by considering GM and AUC. In general, the classical undersampling methods behave well, with the exception of the SBC algorithm. In relation to the reduction of

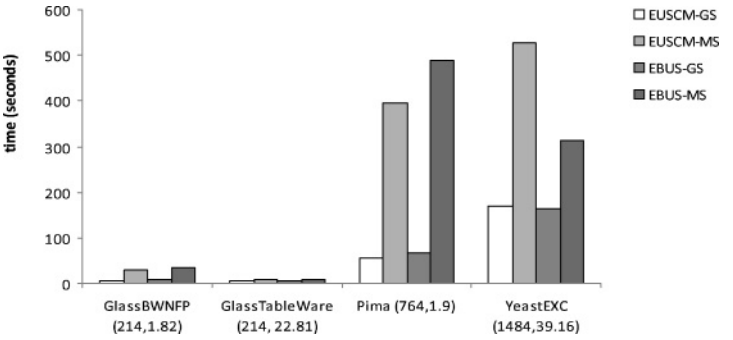

Figure 9: Runtime of the EUS models.

Table 8: Average Results Obtained for the State of the Art Methods and the Two Proposed Algorithms Chosen over Imbalanced Datasets

	EUS Method	$\%Red$	$\%Red^-$	$\%Red^+$	a^-_{tra}	a^+_{tra}	GM_{tra}	a^-_{tst}	a^+_{tst}	GM_{tst}	AUC
Mean	None	0.00	0.00	0.00	**0.9399**	0.6414	0.7485	**0.9387**	0.6175	0.6958	0.7606
SD		0.00	0.00	0.00	0.1832	0.1514	0.1635	0.1831	0.1485	0.1576	0.1648
Mean	US-CNN +	81.31	**96.00**	0.00	0.6949	0.8975	0.7649	0.7093	0.8444	0.7193	0.7618
SD	TL	1.70	1.85	0.00	0.1575	0.179	0.1653	0.1592	0.1737	0.1603	0.1649
Mean	US-CNN	72.95	85.00	0.00	0.8702	0.6882	0.747	0.8855	0.6882	0.7195	0.7696
SD		1.61	1.746	0.00	0.1763	0.1568	0.1633	0.1778	0.1568	0.1603	0.1658
Mean	CPM	81.12	84.00	51.74	0.8854	0.5778	0.6906	0.898	0.6345	0.7039	0.7487
SD		1.70	1.74	1.36	0.1778	0.1437	0.157	0.1791	0.1505	0.1586	0.1635
Mean	NCL	10.04	13.00	0.00	0.8966	0.822	0.8378	0.8907	0.7162	0.7385	0.7862
SD		0.60	0.69	0.00	0.1789	0.1713	0.173	0.1784	0.1599	0.1624	0.1676
Mean	OSS	76.37	90.00	0.00	0.838	0.8177	0.8067	0.8475	0.7543	0.7455	0.7837
SD		1.65	1.78	0.00	0.173	0.1709	0.1697	0.174	0.1641	0.1632	0.1673
Mean	RUS	69.28	79.0	0.0	0.8062	0.8425	0.8222	0.8045	0.8045	0.7757	0.7892
SD		1.57	1.69	0.00	0.1697	0.1735	0.1714	0.1695	0.1695	0.1664	0.1679
Mean	SBC	76.84	90.00	0.00	0.3275	**0.9279**	0.3458	0.3268	**0.8857**	0.3382	0.6063
SD		1.67	1.79	0.00	0.1082	0.182	0.1111	0.108	0.1779	0.1099	0.1472
Mean	TL	6.67	9.00	0.00	0.9191	0.7804	0.8241	0.9079	0.6925	0.7338	0.7829
SD		0.49	0.56	0.90	0.1812	0.1669	0.1716	0.1801	0.1573	0.1619	0.1672
Mean	EBUS-	69.93	80.00	0.00	0.8504	0.9252	0.8862	0.8319	0.8188	**0.7971**	**0.8085**
SD	MS-GM	1.58	1.69	0.00	0.1743	0.1818	0.1779	0.1724	0.171	0.1687	0.1699

the training set achieved, three classes of algorithms exist: algorithms whose reduction rate is low (TL and NCL), algorithms whose reduction capability is high (in general, all EUS models with GS, as we saw in Section 4.2), and algorithms that adapt the reduction to the optimum accuracy (to whose class belong the remaining methods which usually achieve a reduction rate closer to 70–80%).

Our interest lies in knowing whether the EUS models may be considered better than the classical undersampling algorithms in order to recommend their use. In order to do this, the results will be contrasted through a multiple comparison test, which will

Table 9: Statistics and Critical Values for Friedman's and Iman-Davenport's Tests

Friedman statistic for GM	Friedman statistic for AUC	Critical value	Iman-Davenport statistic for GM	Iman-Davenport statistic for AUC	Critical value	Hypotheses
74.170	78.789	16.919	11.261	12.282	1.918	All rejected

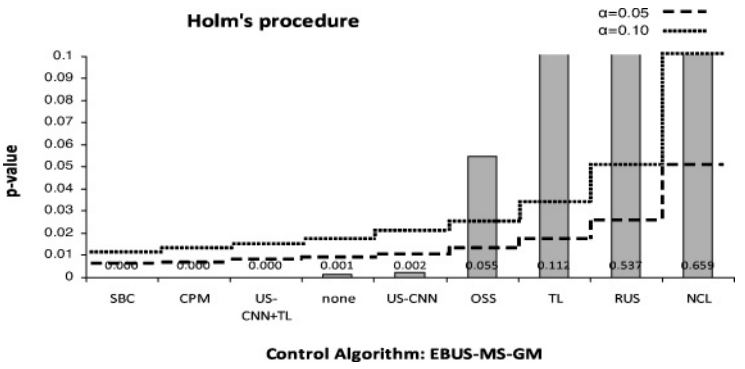

Figure 10: Holm's test on all datasets with GM; the control algorithm is EBUS-MS-GM.

be Holm's procedure. We will represent the results of this test by using the model of figures already employed in Section 4.1 (Figure 5).

The Friedman's and Iman-Davenport's tests results with a level of significance $\alpha = .05$ by considering the 28 imbalanced datasets and the methods enumerated in Table 8 are reported in Table 9.

Both tests find significant differences in the results obtained in this study. Therefore, we can apply Holm's procedure as a post hoc test in order to detect the set of methods that are significantly worse than the control method. Figures 10 and 11 display the p values and the threshold of significance for Holm's procedure with $\alpha = .05$ and $\alpha = .10$. The control method is set as the one that achieves the highest value of performance in GM and AUC.

The EBUS-MS-GM model is the one that achieves the best ranking, so it is the control method in both comparisons. As we can see in both Figures 10 and Figure 11, EBUS-MS-GM outperforms five undersampling methods: SBC, CPM, US-CNN, US-CNN+TL, and no application of undersampling. Although EBUS-MS-GM obtains a better performance than the four remaining algorithms, Holm's procedure is not able to detect these differences as significant, measuring the performance with GM or AUC.

One of the factors that makes learning on imbalanced domains more difficult, as we had already commented, is the increase of the degree of imbalance between classes. In relation to this, we will make a second study composed of the four algorithms that have no statistical differences with respect to the EBUS-MS-GM model by dividing the group of imbalanced datasets into two subgroups, in the same way that we did in the previous section: those that have an $IR < 9$ and those that have an $IR > 9$. Note that although the number of algorithms to be compared is lower than originally, the number of datasets is also reduced by half, so the results reported by the nonparametric tests are not influenced in favor or against a desired result.

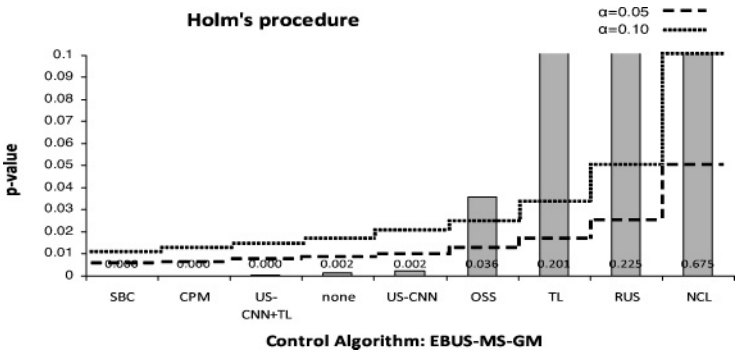

Figure 11: Holm's test on all datasets with AUC; the control algorithm is EBUS-MS-GM.

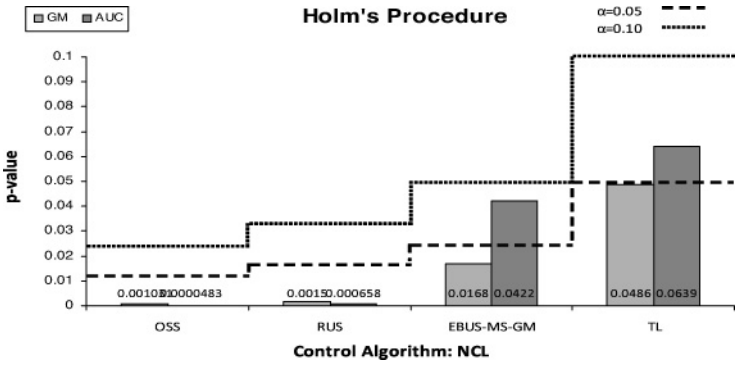

Figure 12: Holm's test on datasets with $IR < 9$; the control algorithm is NCL.

Firstly, we study the case where imbalanced datasets have $IR < 9$. Figure 12 shows a graphical representation of Holm's procedure.

In the case of $IR < 9$, the best method is NCL. Measuring the performance by means of GM, NCL is statistically better than all the methods considered with $\alpha = .05$ and $\alpha = .10$. However, with AUC as the performance metric, EBUS-MS-GM and TL behave as well as NCL when considering $\alpha = .05$.

Secondly, we study the case where imbalanced datasets have $IR > 9$. Figure 13 shows a graphical representation of Holm's procedure.

- Considering $\alpha = .05$, EBUS-MS-GM is similar in performance to RUS and it is also similar to OSS when evaluating with AUC.
- Considering $\alpha = .10$, EBUS-MS-GM is similar to RUS in performance when using GM as the performance measure.
- EBUS-MS-GM is the best method for a level of significance of $\alpha = .10$ with AUC as the performance measure.

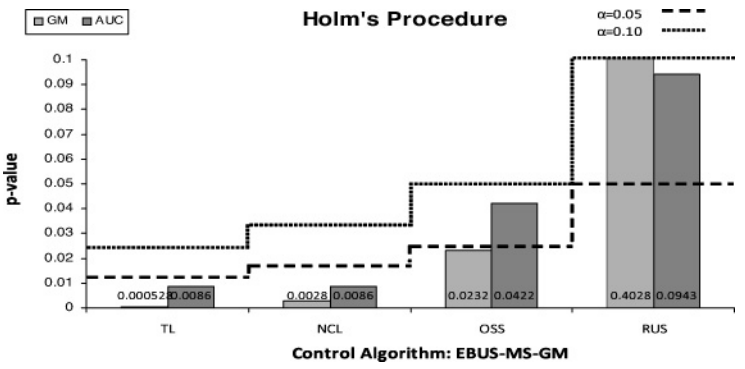

Figure 13: Holm's test on datasets with $IR > 9$; the control algorithm is EBUS-MS-GM.

The conclusions that we can extract analyzing these tables and figures are as follows:

- EUS models usually present an equal or better performance than the remaining methods, independent of the degree of imbalance of data.
- The best performing undersampling model over imbalance datasets is EBUS-MS-GM (Table 8).
- EBUS-MS-GM is not the best model when we use imbalanced datasets with low IR, although it obtains good results. The NCL algorithm is the most appropriate to be used in this type of dataset (Figure 12), but when IR increases, it does not behave well. Hence, NCL is not appropriate to use over datasets with high IR.
- The tendency of the EUS models follows an improvement of the behavior in classification when the data turn to a high degree of imbalance.
- The EBUS-MS-GM model is the most accurate when we deal with datasets with $IR > 9$. This fact is proven by observing Figure 13 in which it is significantly better than the remaining algorithms by using the AUC measure.
- An observable difference exists when measuring the behavior of the classical and EUS methods between GM and AUC. For instance, with GM evaluation, the algorithms RUS and EBUS-MS-GM are significantly equivalent to Holm's procedure. GM evaluates a trade-off between accuracy on positive and negative classes. RUS maintains all the positive examples and randomly selects a subset of negative instances. This subset of instances, although randomly selected, may become a good representative of the negative set of instances. On the other hand, AUC measures a trade-off between true positives and false positives, so it penalizes the misclassification of positive instances. Just as it is easy to obtain a random subset of instances that is accurate with respect to examples of the same class, it is not so easy to find a random subset of instances of a certain class that does not harm the classification of the opposite class. For this reason, the RUS algorithm performs well when considering GM and not as well with AUC.
- Classical undersampling algorithms, such as NCL and TL, lose accuracy when IR becomes high. This is logical because of the fact that their intention is to preserve

minority class instances as well as to not produce massive removal of majority class instances.

- The model EBUS-MS-GM (in general EUS) can adapt to distinct situations of imbalance and it is not problem dependent.

7 Conclusions

This paper addressed the analysis of prototype selection and undersampling algorithms over imbalance classification problems when they are applied in different imbalance ratios in the distribution of classes. A proposal of taxonomy of evolutionary undersampling methods is offered, categorizing all models according to the objective of interest, the selection scheme, and the evaluation measure.

An experimental study was carried out to compare the results of the evolutionary undersampling approach with nonevolutionary techniques.

The main conclusions achieved are as follows.

- Prototype selection algorithms must not be used for handling imbalanced problems. They are prone to gain global performance by eliminating examples belonging to the minority class by considering them to be noisy examples.
- During the evolutionary undersampling process, the employment of the majority selection mechanism helps to obtain more accurate subsets of instances than does the use of the global selection mechanism. However, the global selection mechanism is necessary to achieve the highest reduction rates.
- A significant difference between the use of GM or AUC in the evaluation of solutions in EUS approaches is not observed.
- Datasets with a low imbalance ratio may be faced by EUSCM models, especially by using the model with a global mechanism of selection and evaluation through the GM measure.
- Although over datasets with high imbalance ratio, all EUS models obtain good results, we emphasize the EBUS models with a special interest in the one that performs a majority selection by using the GM measure. The superiority of this model in relation to state of the art undersampling algorithms has been empirically proven.

Finally, we would like to point out that the *EUS* approach is a good choice for undersampling imbalanced datasets, especially when the data presents a high imbalance ratio among the classes. We recommend the use of the *EBUS-MS-GM* model over imbalanced datasets.

As future research lines, the following topics remain to be explored.

- The use of evolutionary undersampling for training set selection (Cano et al., 2007) in order to analyze the behavior of other classification methods (C4.5, SVMs, etc.), combined with subset selection for imbalanced datasets.
- A study on scalability for making it feasible to apply evolutionary undersampling for very large datasets (Song et al., 2005; Cano et al., 2005).

- The analysis of evolutionary undersampling in terms of data complexity (Ho and Basu, 2002; Bernadó-Mansilla and Ho, 2005) for a better understanding of the behavior of our approach over dataset depending on the data complexity measure values.

Acknowledgments

This research has been supported by the project TIN2005-08386-C05-01. S. García holds an FPU scholarship from the Spanish Ministry of Education and Science. The authors are very grateful to the anonymous reviewers for their valuable suggestions and comments to improve the quality of this paper.

References

Aha, D. W., Kibbler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6:37–66.

Akbani, R., Kwek, S., and Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *ECML, LNCS 3201*, pp. 39–50.

Alba, E., Luque, G., and Araujo, L. (2006). Natural language tagging with genetic algorithms. *Information Processing Letters*, 100(5):173–182.

Alcalá, R., Alcala-Fdez, J., Herrera, F., and Otero, J. (2007). Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation. *International Journal of Approximate Reasoning*, 44(1):45–64.

Barandela, R., Sánchez, J. S., García, V., and Rangel, E. (2003). Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851.

Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6(1):20–29.

Bernadó-Mansilla, E., and Garrell-Guiu, J. M. (2003). Accuracy-based learning classifier systems: Models, analysis and applications to classification tasks. *Evolutionary Computation*, 11(3):209–238.

Bernadó-Mansilla, E., and Ho, T. K. (2005). Domain of competence of XCS classifier system in complexity measurement space. *IEEE Transactions on Evolutionary Computation*, 9(1):82–104.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Butz, M. V., Pelikan, M., Llorá, X., and Goldberg, D. E. (2006). Automated global structure extraction for effective local building block processing in XCS. *Evolutionary Computation*, 14(3):345–380.

Cano, J. R., Herrera, F., and Lozano, M. (2003). Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study. *IEEE Transactions on Evolutionary Computation*, 7(6):561–575.

Cano, J. R., Herrera, F., and Lozano, M. (2005). Stratification for scaling up evolutionary prototype selection. *Pattern Recognition Letters*, 26(7):953–963.

Cano, J. R., Herrera, F., and Lozano, M. (2007). Evolutionary stratified training set selection for extracting classification rules with trade-off precision-interpretability. *Data and Knowledge Engineering*, 60:90–108.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Chawla, N. V., Japkowicz, N., and Kotcz, A. (2004). Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explorations*, 6(1):1–6.

Chawla, N. V., Lazarevic, A., Hall, L. O., and Bowyer, K. W. (2003). SMOTEBoost: Improving prediction of the minority class in boosting. In *Seventh European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD*, pp. 107–119.

Cieslak, D. A., Chawla, N. V., and Striegel, A. (2006). Combating imbalance in network intrusion datasets. In *Proceedings of the IEEE International Conference on Granular Computing*, pp. 732–737.

Cohen, G., Hilario, M., Sax, H., Hugonnet, S., and Geissbühler, A. (2006). Learning from imbalanced data in surveillance of nosocomial infection. *Artificial Intelligence in Medicine*, 37(1):7–18.

Cordón, O., Damas, S., and Santamaría, J. (2006). Feature-based image registration by means of the CHC evolutionary algorithm. *Image Vision Computing*, 24(5):525–533.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

Eshelman, L. J. (1991). The CHC adaptive search algorithm: How to safe search when engaging in nontraditional genetic recombination. In G. J. E. Rawlings (Ed.), *Foundations of genetic algorithms* (pp. 265–283). San Mateo, CA: Morgan Kauffman.

Estabrooks, A., Jo, T., and Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1):18–36.

García, S., Cano, J. R., Fernández, A., and Herrera, F. (2006). A proposal of evolutionary prototype selection for class imbalance problems. In *IDEAL, LNCS 4224*, pp. 1415–1423.

Grzymala-Busse, J. W., Stefanowski, J., and Wilk, S. M. (2005). A comparison of two approaches to data mining from imbalanced data. *Journal of Intelligent Manufacturing*, 16:565–573.

Guerra-Salcedo, C., Chen, S., Whitley, D., and Smith, S. (1999). Fast and accurate feature selection using hybrid genetic strategies. In *Proceedings of the Congress on Evolutionary Computation, CEC*, pp. 177–184.

Guerra-Salcedo, C., and Whitley, D. (1998). Genetic search for feature subset selection: A comparison between CHC and GENESIS. In *Proceedings of the Third Annual Conference of Genetic Programming*, pp. 504–509.

Guo, H., and Viktor, H. L. (2004). Learning from imbalanced data sets with boosting and data generation: The databoost-IM approach. *SIGKDD Explorations*, 6(1):30–39.

Hart, P. E. (1968). The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 18:515–516.

Ho, S.-Y., Liu, C.-C., and Liu, S. (2002). Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. *Pattern Recognition Letters*, 23(13):1495–1503.

Ho, T. K., and Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70.

Huang, K., Yang, H., King, I., and Lyu, M. R. (2006). Imbalanced learning with a biased minimax probability machine. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(4):913–923.

Iman, R. L., and Davenport, J. M. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics*, 18:571–595.

Kovacs, T., and Kerber, M. (2006). A study of structural and parametric learning in XCS. *Evolutionary Computation*, 14(1):1–19.

Kubat, M., and Matwin, S. (1997). Addressing the course of imbalanced training sets: One-sided selection. In *Proceedings of the International Conference on Machine Learning, ICML*, pp. 179–186.

Kuncheva, L. I., and Bezdek, J. C. (1998). Nearest prototype classification: Clustering, genetic algorithms, or random search? *IEEE Transactions on Systems, Man, and Cybernetics*, 28(1):160–164.

Laurikkala, J. (2001). Improving identification of difficult small classes by balancing class distribution. In *AIME '01: Proceedings of the 8th Conference on AI in Medicine in Europe*, pp. 63–66.

Newman, D. J., Hettich, S., Blake, C. L., and Merz, C. J. (1998). UCI repository of machine learning databases.

Orriols-Puig, A., and Bernadó-Mansilla, E. (2006). Bounding XCS's parameters for unbalanced datasets. In *GECCO '06: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 1561–1568.

Papadakis, E., and Theocharis, B. (2006). A genetic method for designing TSK models based on objective weighting: Application to classification problems. *Soft Computing*, 10(9):805–824.

Sheskin, D. (2003). *Handbook of parametric and nonparametric statistical procedures.* Boca Raton, FL: Chapman & Hall/CRC.

Sikora, R., and Piramuthu, S. (2007). Framework for efficient feature selection in genetic algorithm based data mining. *European Journal of Operational Research*, 180:723–737.

Song, D., Heywood, M. I., and Zincir-Heywood, A. N. (2005). Training genetic programming on half a million patterns: An example from anomaly detection. *IEEE Transactions on Evolutionary Computation*, 9(3):225–239.

Tomek, I. (1976). Two modifications of CNN. *IEEE Transactions on Systems, Man, and Communications*, 6:769–772.

Wang, X., Yang, J., Teng, X., Xia, W., and Jensen, R. (2007). Feature selection based on rough sets and particle swarm optimization. *Pattern Recognition Letters*, 28(4):459–471.

Weiss, G. M., and Provost, F. J. (2003). Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, 19:315–354.

Whitley, D., Beveridge, R., Guerra, C., and Graves, C. (1998). Messy genetic algorithms for subset feature selection. In *Proceedings of the International Conference on Genetic Algorithms*, pp. 568–575.

Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, 2:408–421.

Wilson, D. R., and Martinez, T. R. (2000). Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3):257–286.

Xie, J., and Qiu, Z. (2007). The effect of imbalanced data sets on LDA: A theoretical and empirical analysis. *Pattern Recognition*, 40:557–562.

Yen, S., and Lee, Y. (2006). Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset. In *ICIC, LNCIS 344*, pp. 731–740.

Yoon, K., and Kwek, S. (2005). An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics. In *HIS '05: Proceedings of the Fifth International Conference on Hybrid Intelligent Systems*, pp. 303–308.

Appendix A Undersampling Methods Focused on Balancing Data versus Prototype Selection Methods

This Appendix summarizes and describes the methods used in the experimental study of this paper. We distinguish between methods used in the PS task (Section A.1) and classical undersampling methods focused on reducing data with the aim of balancing data (Section A.2).

A.1 Prototype Selection Methods

Two classical models for PS are used in this study: an incremental well-known technique, IB3 (Aha et al., 1991), and a decremental one, DROP3 (Wilson and Martinez, 2000). In addition to this, we point out that the study also includes the models CHC and IGA for PS as defined in Section 3. These models will be called EPS-CHC and EPS-IGA, respectively.

Next, we describe the two classical methods.

- IB3 Instance x from the training set TR is added to the new set S if the nearest *acceptable* instance in S (if there are not *acceptable* instances, then a random instance is used) has a different class from x. The *acceptable* concept is defined as the confidence interval:

$$\frac{p + \frac{z^2}{2n} \pm z\sqrt{\frac{p(p-1)}{n} + \frac{z^2}{2n^2}}}{1 + \frac{z^2}{n}}, \tag{10}$$

  where z is the confidence factor (0.9 is used to accept, 0.7 to reject); p is the classification accuracy of an x instance (while x is added to S); and n is the number of classification-trials for a given instance (while added to S).

- DROP3 TR is copied to the subset selected S. It uses a noise filtering pass before sorting the instances in S. This is done using the rule that any instance not classified by its k nearest neighbors is removed (we use $k = 3$). After removing noisy instances from S in this manner, the instances are sorted by distance to their nearest enemy remaining in S, and thus points far from the real decision boundary are removed first. This allows points internal to clusters to be removed early in the process, even if there were noisy points nearby. After the noise removal, the steps are described in Figure 14 (Wilson and Martinez, 2000).

A.2 Classical Undersampling Methods for Balancing Class Distribution

In this work, we evaluate eight different methods of undersampling to balance the class distribution on training data:

- Random Undersampling (RUS) RUS is a nonheuristic method that aims to balance class distribution through the random elimination of majority class examples to get a balanced instance set. The final ratio of balancing can be adjusted.

- Tomek Links (TL; Tomek, 1976) Tomek links can be defined as follows: given two examples $E_i = (x_i, y_i)$ and $E_j = (x_j, y_j)$ where $y_i \neq y_j$ and $d(E_i, E_j)$ is the distance between E_i and E_j. A pair (E_i, E_j) is called a Tomek link if there is not an example

1. Let $S = TR$
2. For each instance s in S
   3. Find s.N1..k+1, the k+1 nearest neighbors of s in S
   4. Add s to each of its neighbors' lists of associates
5. For each instance s in S
   6. Let *with* = ♯ of associates of s classified correctly with s as a neighbor
   7. Let *without* = ♯ of associates of s classified correctly without s
   8. If (without – with) = 0
      9. Remove s from S if at least as many of its associates in TR would be classified correctly without s.
      10. For each associate a of s
         11. Remove s from as list of nearest neighbors
         12. Find a new nearest neighbor for a
         13. Add a to its new neighbors' list of associates
      14. For each neighbor k of s
         15. Remove s from ks lists of associates
16. Return S

Figure 14: Pseudocode of the DROP3 algorithm.

E_l, such that $d(E_i, E_l) < d(E_i, E_j)$ or $d(E_j, E_l) < d(E_i, E_j)$. Tomek links can be used as an undersampling method eliminating only examples belonging to the majority class in each Tomek link found.

- Condensed Nearest Neighbor Rule (US-CNN; Hart, 1968) First, randomly draw one majority class example and all examples from the minority class and put these examples in S. Afterward, use a 1-NN over the examples in S to classify the examples in TR. Every misclassified example from TR is moved to S.

- One-Sided Selection (OSS; Kubat and Matwin, 1997) OSS is an undersampling method resulting from the application of Tomek links followed by the application of US-CNN.

- US-CNN + TL (Batista et al., 2004) US-CNN + TL is similar to OSS, but the method US-CNN is applied before the Tomek links.

- Neighborhood Cleaning Rule (NCL; Laurikkala, 2001) The NCL uses the Wilson's edited nearest neighbor rule (ENN; Wilson, 1972) to remove majority class examples. For each example $E_i = (x_i, y_i)$ in the training set, its three nearest neighbors are found. If E_i belongs to the majority class and the classification given by its three nearest neighbors contradicts the original class of E_i, then E_i is removed. If E_i belongs to the minority class and its three nearest neighbors misclassify E_i, then the nearest neighbors that belong to the majority class are removed.

- Class Purity Maximization (CPM; Yoon and Kwek, 2005) CPM attempts to find a pair of centers, one being a minority class instance while the other is a majority class instance. Using these centers, CPM partitions all the instances into two clusters, C_1 and C_2. If either of the clusters have less class impurity than the parent's impurity (Imp) then we have found our clusters. The impurity of a set of instances is simply the proportion of minority class instances. It then recursively partitions each of these clusters into subclusters. Thus, it forms a hierarchical clustering. If the impurity cannot be improved, then we stop the recursion. The algorithm is described in Figure 15.

- Undersampling Based on Clustering (SBC; Yen and Lee, 2006) Considering that the number of samples in the class-imbalanced dataset is N, within it, the number of samples belonging to the majority class is N^- and the number of minority class samples is N^+. SBC first clusters all samples in the dataset into K clusters. The number of majority class and minority class samples is N_i^- and N_i^+, respectively. Therefore, the ratio of the number of majority class samples to the number of minority class samples in the ith cluster is N_i^-/N_i^+. If the ratio of N_i^- to N_i^+ in the training dataset is set to be $m : 1$, the number of selected majority class samples in the ith cluster is shown in the expression in Equation (11).

$$SN_i^- = (m \cdot N^+) \cdot \frac{N_i^-/N_i^+}{\sum_{i=1}^{K}(N_i^-/N_i^+)} \tag{11}$$

After determining the number of majority class samples in each cluster, SBC randomly chooses majority class samples in the ith cluster.

Input: Imp: cluster impurity of parent cluster

$parent$: parent cluster ID

Output: subclusters C_i rooted at parent

CPM(Imp,$parent$)

1. $impurity \leftarrow \infty$

2. While $Imp \leq impurity$

   3. If all the instance pairs in $parent$ were tested then return

   4. Pick a pair of majority and minority class instances as centers

   5. Partition all instances into two clusters C_1 and C_2

      according to nearest center

   6. $impurity \leftarrow min(impurity(C_1), impurity(C_2))$

7. CPM($impurity(C_1), C_1$)

8. CPM($impurity(C_2), C_2$)

Figure 15: Pseudocode of CPM algorithm.