Applied Soft Computing 9 (2009) 1304–1314

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Enhancing the effectiveness and interpretability of decision tree and rule induction classifiers with evolutionary training set selection over imbalanced problems

Salvador García *, Alberto Fernández, Francisco Herrera

Dept. of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 19 October 2008
Accepted 18 April 2009
Available online 21 May 2009

Keywords:
Evolutionary algorithms
Imbalanced classification
Data reduction
Training set selection
Decision trees
Rule induction

ABSTRACT

Classification in imbalanced domains is a recent challenge in data mining. We refer to imbalanced classification when data presents many examples from one class and few from the other class, and the less representative class is the one which has more interest from the point of view of the learning task. One of the most used techniques to tackle this problem consists in preprocessing the data previously to the learning process. This preprocessing could be done through under-sampling; removing examples, mainly belonging to the majority class; and over-sampling, by means of replicating or generating new minority examples. In this paper, we propose an under-sampling procedure guided by evolutionary algorithms to perform a training set selection for enhancing the decision trees obtained by the C4.5 algorithm and the rule sets obtained by PART rule induction algorithm. The proposal has been compared with other under-sampling and over-sampling techniques and the results indicate that the new approach is very competitive in terms of accuracy when comparing with over-sampling and it outperforms standard under-sampling. Moreover, the obtained models are smaller in terms of number of leaves or rules generated and they can considered more interpretable. The results have been contrasted through non-parametric statistical tests over multiple data sets.

1. Introduction

The data used in a classification task could be not perfect. Data could present different types of imperfections, such as the presence of errors or missing values or imbalanced distribution of classes. In the last years, the class imbalance problem is one of the emergent challenges in data mining (DM) [45]. The problem appears when the data presents a class imbalance, which consists in containing many more examples of one class than the other one and the less representative class represents the most interesting concept from the point of view of learning [10]. The imbalance classification problem is very related with the cost-sensitive classification problem [9]. Imbalance in class distribution is pervasive in a variety of real-world applications, including but not limited to telecommunications [37], WWW, finance, ecology [29], biology and medicine [21].

Usually, in imbalanced classification problems, the instances are grouped into two types of classes: the majority or negative class, and the minority or positive class. The minority or positive class has more interest and it is also accompanied with a higher cost of making errors. A standard classifier might ignore the importance of the minority class because its representation inside the data set is not strong enough. As a classical example, if the ratio of imbalance presented in the data is 1:100 (that is, there is one positive instance versus one hundred negatives), the error of ignoring this class is only 1%, so many classifiers could ignore it or could not make any effort to learn an effective model for it.

Many approaches have been proposed to deal with the class imbalance problem. They can be divided into algorithmic approaches and data approaches. The first ones assume modifications in the operation of the algorithms, making them cost-sensitive towards the minority class [24,32,47,34]. The data approaches modify the data distribution, conditioned on an evaluation function. Re-sampling of data could be done by means of under-sampling, by removing instances from the data, and over-sampling, by replicating or generating new minority examples. There have been numerous papers and case studies exemplifying their advantages [8,3,19,9,18].

Decision trees and rule induction algorithms are very important techniques and they are used extensively in DM [26]. They are able to produce human-readable descriptions of trends in the underlying relationships of a data set and can be used for classification and prediction tasks. In the literature, many techniques of decision

* Corresponding author. Tel.: +34 958 240598.
E-mail addresses: salvagl@decsai.ugr.es.es (S. García), alberto@decsai.ugr.es (A. Fernández), herrera@decsai.ugr.es (F. Herrera).

1568-4946/$ – see front matter.
doi:10.1016/j.asoc.2009.04.004

trees and rule induction algorithms have been proposed [4,35,19,38]. In their conventional definition, these algorithms can be applied to imbalanced classification problems, although the performance that they achieve is not the adequate, unless we use appropriate algorithms which are adapted to imbalanced performance measures [17].

Evolutionary algorithms (EAs) [14] have been used for DM with promising results [20,11]. In data reduction, they have been successfully used for feature selection [42,25,40,46] and instance selection [5,6,22]. EAs also have a good behaviour for training set selection (TSS) in terms of getting a trade-off between precision and interpretability with classification rules [7].

In the field of class imbalanced classification, EAs are being applied recently. In [28], an EA is used to search an optimal tree in a global manner for cost-sensitive classification. In [13], the authors propose new heuristics and metrics for improving the performance of several genetic programming classifiers in imbalanced domains. EAs have also been applied for under-sampling the data in imbalanced domains in instance-based learning [23].

In this contribution, we propose the use of EAs for TSS in imbalanced data sets. Our objective is to increase the effectiveness of a well-known decision tree classifier, C4.5 [35], and a rule induction algorithm, PART [19] by means of removing instances guided by an evolutionary under-sampling algorithm. We compare our approach with other under-sampling, over-sampling methods and hybridization proposals of over-sampling and under-sampling [3] studied in the literature. The empirical study is contrasted via non-parametrical statistical testing in a multiple data set environment.

To achieve this objective, the rest of the contribution is organized as follows: Section 2 gives an overview about imbalanced classification. In Section 3, the evolutionary TSS issues are explained, together with a description of the used model. In Section 4 the experimentation framework, the results obtained and their analyses are presented. Section 5, remarks our conclusion. Finally, Appendix A is included in order to illustrate the comparisons of our proposal with other techniques through star plots.

2. Imbalanced data sets in classification: evaluation metrics and preprocessing techniques

In this section we will first introduce the data set imbalance problem. Then we will present the evaluation metrics for this kind of classification problem. Finally, we will show some preprocessing techniques that are commonly applied in order to deal with the imbalanced data sets.

2.1. The problem of imbalanced data sets

The imbalanced data set problem in classification domains occurs when the number of instances which represents one class is much larger than the other classes. Furthermore, the minority class is usually the one which has more interest from the point of view of the learning task [10]. This problem is very related with the cost-sensitive classification problem [21,47,32].

As we have mentioned, the classical machine learning algorithms may be biased towards the majority class and thus, may predict poorly the minority class examples.

To solve the imbalance data set problem there are two main types of solutions:

1. Solutions at the data level [8,3,9]: This kind of solution consists of balancing the class distribution by over-sampling the minority class (positive instances) or under-sampling the majority class (negative instances).
2. Solutions at the algorithmic level: In this case we may fit our method adjusting the cost per class [24], for example, adjusting the probability estimation in the leaves of a decision tree bias the positive class [41].

We focus on the two class imbalanced data sets, where there are only one positive and one negative class. We consider the positive class as the one with the lower number of examples and the negative class the one with the higher number of examples. In order to deal with the class imbalance problem we analyse the cooperation of some preprocessing methods of instances.

2.2. Evaluation in imbalanced domains

The most straightforward way to evaluate the performance of classifiers is based on the confusion matrix analysis. Table 1 illustrates a confusion matrix for a two-class problem having positive and negative class values. From such a table it is possible to extract a number of widely used metrics for measuring the performance of learning systems, such as error rate (1) and accuracy (2):

$$Err = \frac{FP + FN}{TP + FN + FP + TN} \tag{1}$$

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} = 1 - Err \tag{2}$$

In [41] it is shown that the error rate of the classification of the rules of the minority class is 2 or 3 times greater than the rules that identify the examples of the majority class and that the examples of the minority class are less probable to be predict than the examples of the majority one. Because of this, instead of using the error rate (or accuracy), in the ambit of imbalanced problems more correct metrics are considered. Specifically, from Table 1 it is possible to derive four performance metrics that directly measure the classification performance on positive and negative classes independently:

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

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

The metric used in this work is the geometric mean of the true rates [2], which can be defined as

$$GM = \sqrt{acc^+ \cdot acc^-} \tag{3}$$

where acc^+ means the accuracy in the positive examples (TP_{rate}) and acc^- is the accuracy in the negative examples (TN_{rate}). This metric tries to maximize the accuracy of each one of the two

Table 1
Confusion matrix for a two-class problem.

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

classes with a good balance. It is a performance metric that links both objectives.

2.3. Preprocessing imbalanced data sets

In the specialized literature, we may find some papers for re-sampling techniques from the point of view of the study of the effect of the class distribution in classification [41,16] and adaptations of instance selection methods [44,33] to treat with imbalanced data sets. It has been proved that applying a preprocessing step in order to balance the class distribution is a positive solution to the imbalance data set problem [3]. Besides, the main advantage of these techniques is that they are independent of the classifier used.

In this work we evaluate different instance selection methods together with over-sampling and hybrid techniques to adjust the class distribution in the training data. Specifically, we have used the methods which offer the best results in [3]. These methods are classified into three groups:

- **Under-sampling methods** that create a subset of the original database by eliminating some of the examples of the majority class.
- **Over-sampling methods** that create a superset of the original database by replicating some of the examples of the minority class or creating new ones from the original minority class instances.
- **Hybrid methods** that combine the two previous methods eliminating some of the minority class examples expanded by the over-sampling method in order to get rid of overfitting.

2.3.1. Under-sampling methods

- "**One-sided selection**" (OSS) [30] is an under-sampling method resulting from the application of Tomek links followed by the application of CNN. Tomek links are used as an under-sampling method and remove noisy and borderline majority class examples. Borderline examples can be considered "unsafe" since a small amount of noise can make them fall on the wrong side of the decision border. CNN aims to remove examples from the majority class that are distant from the decision border. The remainder examples, i.e., "safe" majority class examples and all minority class examples are used for learning.
- "**Neighborhood cleaning rule**" (NCL) uses the Wilson's Edited Nearest Neighbor Rule (ENN) [43,31] to remove majority class examples. ENN removes any example whose class label differs from the class of at least two of its three nearest neighbors. NCL modifies the ENN in order to increase the data cleaning. For a two-class problem the algorithm can be described in the following way: for each example e_i in the training set, its three nearest neighbors are found. If e_i belongs to the majority class and the classification given by its three nearest neighbors 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.

2.3.2. Over-sampling methods

- "**Synthetic Minority Over-sampling Technique**" (SMOTE) [8] is an over-sampling method. Its main idea is to form new minority class examples by interpolating between several minority class examples that lie together. Thus, the overfitting problem is avoided and causes the decision boundaries for the minority class to spread further into the majority class space.

2.3.3. Hybrid methods: over-sampling + under-sampling

- "**SMOTE + Tomek links (TL)**": Frequently, class clusters are not well defined since some majority class examples might be invading the minority class space. The opposite can also be true, since interpolating minority class examples can expand the minority class clusters, introducing artificial minority class examples too deeply in the majority class space. Inducing a classifier under such a situation can lead to overfitting. In order to create better-defined class clusters, we propose applying Tomek links [39] to the over-sampled training set as a data cleaning method. Thus, instead of removing only the majority class examples that form Tomek links, examples from both classes are removed.
- "**SMOTE + ENN**": The motivation behind this method is similar to SMOTE + Tomek links. ENN [43] tends to remove more examples than the Tomek links does, so it is expected that it will provide a more in depth data cleaning. Differently from NCL which is an under-sampling method, ENN is used to remove examples from both classes. Thus, any example that is misclassified by its three nearest neighbors is removed from the training set.

3. Evolutionary training set selection in imbalanced classification

Let us assume that there is a training set TR with N instances which consists of pairs $(x_i, y_i), i = 1, \ldots, N$, 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 positive or negative class. Let $S \subseteq TR$ be the subset of selected instances resulted in the execution of an algorithm.

TSS can be considered as a search problem in which EAs can be applied. Our approach will be denoted by Evolutionary Under-Sampling for Training Set Selection (EUSTSS). We take into account two important issues: the specification of the representation of the solutions and the definition of the fitness function:

- *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 (see Fig. 1).
- *Fitness function*: Let S be a subset of instances of TR and be coded by a chromosome. We define a fitness function based on the GM measure evaluated over TR:

$$fitness(S) = GM \tag{4}$$

This fitness function is related with the proposal of Evolutionary Under-Sampling for nearest neighbors classifier guided by Classification Measures (EUSCM) proposed in [23]. A decision tree or a rule induction algorithm can be used for measuring the accuracy associated with the model induced by using the instances selected in S. Obviously, the choice of the

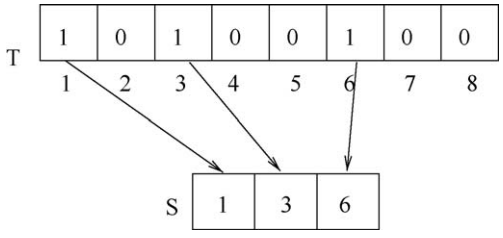


Fig. 1. Chromosome binary representation of a solution.

classifier is conditioned to the final evaluator classifier, following a wrapper scheme. The accuracy independently computed in each class is useful to obtain *GM* value associated to the chromosome. The objective of the EAs is to maximize the fitness function defined: maximize the *GM*rate.

A mechanism to avoid overlearning in training data is needed in the fitness function. Although most of the tree or rule induction algorithms, in their definition, usually incorporate a pruning mechanism to avoid overfitting, the inclusion of the induction process within an evolutionary cycle can guide the resulting model to be optimal for only known data, loosing the generalization ability. We incorporate a simple and effective mechanism which consists of providing to the classification costs a higher weight (W) to the instances that are no included in S than to the instances included in S. An instance of TR well classified scores a value W if it is not included in S and a value of 1 if it is included in S. This procedure encourages the reduction ability of the selected subset, due to the fact that it is more beneficial to evaluate chromosomes with a higher number of examples out of the selected ones. Obviously, the instance causes a substraction on accuracy of the same magnitude in case of misclassification. Our empirical studies have determined that a value of W equal to 3 works appropriately.

Fig. 2 represents the evolutionary under-sampling process followed by our proposal:

Algorithm 1. Pseudocode of CHC algorithm.

```
input : A population of chromosomes P_a
output: An optimized population of chromosomes P_a
t ← 0;
Initialize(P_a, ConvergenceCount);
while not EndingCondition(t, P_a) do
    Parents ← SelectionParents(P_a);
    Offspring ← HUX(Parents);
    Evaluate(Offspring);
    P_n ← ElitistSelection(Offspring, P_a);
    if not modified(P_a, P_n) then
        ConvergenceCount ← ConvergenceCount − 1;
        if ConvergenceCount = 0 then
            P_n ← Restart(P_a);
            Initialize(ConvergenceCount);
        end
    end
    t ← t + 1;
    P_a ← P_n;
end
```

- As the evolutionary computation method, we have used the CHC model [15,7]. 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 among parents and offspring into the replacement process.

  During each generation the CHC develops the following steps:

  - It uses a parent population of size N to generate an intermediate population of N individuals, which are randomly paired and used to generate N potential offspring.
  - Then, 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 who 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 is being generated which 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.

  The pseudocode of CHC appears in Algorithm 1.

- *Crossover operator for data reduction*: In order to achieve a good reduction rate, Heuristic Uniform Crossover (HUX) implemented for CHC undergoes a change that makes more difficult the inclusion of instances inside the selected subset. Therefore, if a HUX switches a bit on in a gene, then the bit could be switched off depending on a certain probability (its value will be specified in Section 4.1 and Table 3).

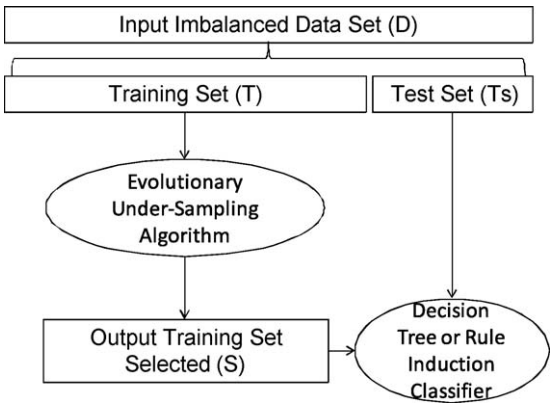


Fig. 2. Evolutionary under-sampling process.

4. Experimental framework and results

This section describes the methodology followed in the experimental study of the re-sampling compared techniques. We will explain the configuration of the experiment: used data sets and parameters for the algorithms. The algorithms used in the comparison are the same described in Section 2.3.

4.1. Experimental framework

Performance of the algorithms is analysed by using 25 data sets taken from the UCI Machine Learning Database Repository [1]. Multi-class data sets are modified to obtain two-class non-

Table 2
Imbalanced data sets.

Data set	# Examples	# Attributes	Class (min., maj.)	%Class (min., maj.)
Abalone9-18	731	9	(18, 9)	(5.75, 94.25)
Dermatology2	366	34	(2, remainder)	(16.67, 83.33)
EcoliCP-IM	220	7	(im, cp)	(35.00, 65.00)
EcoliIM	336	7	(im, remainder)	(22.92, 77.08)
EcoliIMU	336	7	(iMU, remainder)	(10.42, 89.58)
EcoliOM	336	7	(om, remainder)	(6.74, 93.26)
German	1000	20	(1, 0)	(30.00, 70.00)
GlassBWFP	214	9	(build-window-float-proc, remainder)	(32.71, 67.29)
GlassBWNFP	214	9	(build-window-non_float-proc, remainder)	(35.51, 64.49)
GlassNW	214	9	(non-windows glass, remainder)	(23.93, 76.17)
GlassVWFP	214	9	(Ve-win-float-proc, remainder)	(7.94, 92.06)
Haberman	306	3	(Die, Survive)	(26.47, 73.53)
New-thyroid	215	5	(hypo, remainder)	(16.28, 83.72)
PageBlocks(2,4,5)-3	559	10	(3, 2+4+5)	(5.01, 94.99)
Pima	768	8	(1,0)	(34.77, 66.23)
Segment1	2310	19	(1, remainder)	(14.29, 85.71)
VehicleVAN	846	18	(van, remainder)	(23.52, 76.48)
Vowel0	990	13	(0, remainder)	(9.01, 90.99)
Yeast(1)	467	8	(POX, MIT+ME3+EXC+ERL)	(4.28, 95.72)
Yeast(2)	1240	8	(POX+ERL, MIT+NUC+CYT+ME1+EXC)	(2.02, 97.98)
Yeast(3)	1334	8	(EXC, MIT+NUC+CYT+ME3)	(2.62, 97.38)
Yeast(4)	1120	8	(VAC, NUC+CYT+ME3+EXC)	(2.68, 97.32)
YeastCYT-POX	483	8	(POX, CYT)	(4.14, 95.86)
YeastNUC-POX	449	8	(POX, NUC)	(4.45, 95.55)
YeastPOX	1484	8	(POX, remainder)	(1.35, 98.65)

Table 3
Parameters considered for the algorithms.

Algorithm	Parameters
SMOTE	$k = 5$, balancing ratio = 1:1
EUSTSS	$Pop = 50$, $Eval = 10{,}000$ Prob. inclusion HUX = 0.25, $W = 3$

balanced problems, defining the joint of one or more classes as positive and the joint of one or more classes as negative.

The main characteristics of these data sets are summarized in Table 2. For each data set, it shows the number of examples (#Examples), number of attributes (#Attributes) and class name (minority and majority). The data sets considered are partitioned using the *tenfold cross-validation* (*10-fcv*) procedure. The parameters of the used algorithms are presented in Table 3.

4.2. Results and analysis for C4.5

Tables 4 and 5 show the results in training and test data obtained by the re-sampling approaches compared by means of *GM* evaluation measure. The column denoted by *none* corresponds to the case in which no re-sampling is performed previous to C4.5. The best case in each data set is remarked in bold.

Table 4
Results obtained by C4.5 using *GM* evaluation measure over training data.

Data set	None	NCL	OSS	SMOTE	SMOTE + ENN	SMOTE + TL	EUSTSS
abalone9-18	0.6611	0.7206	0.7218	**0.9348**	0.9337	0.8543	0.8449
dermatology2	0.9563	0.9240	0.9437	**0.9894**	0.9853	0.9845	0.9820
ecoliCP-IM	0.9869	0.9526	0.9869	**0.9906**	0.9860	0.9862	0.9869
ecoliIM	0.8602	0.9184	0.9275	**0.9502**	0.9483	0.9341	0.9428
ecoliMU	0.8794	0.8799	0.9234	**0.9722**	0.9625	0.9331	0.9374
ecoliOM	0.9416	0.9197	0.9576	0.9782	0.9891	0.9566	**0.9914**
german	0.7779	0.6881	0.7790	**0.8676**	0.8136	0.7773	0.7474
glassBWFP	0.9391	0.7557	0.8528	**0.9553**	0.8915	0.8906	0.9157
glassBWNFP	0.8684	0.6501	0.8766	**0.9450**	0.8964	0.8720	0.8856
glassNW	0.9770	0.8456	0.9670	**0.9899**	0.9679	0.9704	0.9783
glassVWFP	0.8476	0.8828	0.9691	**0.9779**	0.9611	0.8968	0.9608
haberman	0.4660	0.4856	0.7215	**0.7733**	0.7519	0.7520	0.7141
new-thyroid	0.9678	0.9507	0.9787	0.9869	0.9873	0.9854	**0.9963**
pageblocks(2,4,5)-3	0.9919	0.9542	0.9918	**1.0000**	**1.0000**	0.9980	**1.0000**
pima	0.8151	0.7115	0.8115	**0.8631**	0.8387	0.8210	0.8084
segment1	0.9908	0.9827	0.9957	**0.9991**	0.9988	0.9972	0.9969
vehicle	0.9856	0.8965	0.9696	**0.9889**	0.9784	0.9713	0.9666
vowel0	0.9973	0.9531	0.9973	0.9941	0.9949	0.9947	**0.9979**
yeast(1)	0.6699	0.7491	0.6171	**0.9467**	0.9460	0.8769	0.9357
yeast(2)	0.3938	0.7902	0.4203	0.8888	0.8918	0.8668	**0.8936**
yeast(3)	0.8862	0.9053	0.8973	**0.9642**	0.9675	0.9334	0.9554
yeast(4)	0.1086	0.1460	0.4341	0.7927	**0.8241**	0.6912	0.7793
yeastCYT-POX	0.2568	0.8052	0.3438	0.9072	0.9205	0.8793	**0.9377**
yeastNUC-POX	0.6742	0.8265	0.6742	0.9215	0.9379	0.8970	**0.9745**
yeastPOX	0.0000	0.7362	0.0000	0.8279	**0.8502**	0.8220	0.8473
Average	0.7560	0.8012	0.7903	**0.9362**	0.9289	0.9017	0.9191

Table 5
Results obtained by C4.5 using *GM* evaluation measure over test data.

Data set	None	NCL	OSS	SMOTE	SMOTE + ENN	SMOTE + TL	EUSTSS
abalone9-18	0.3763	0.4761	0.4963	0.6023	**0.6724**	**0.6724**	0.6697
dermatology2	0.8623	0.8988	0.8928	0.9194	0.9181	0.9098	**0.9505**
ecoliCP-IM	**0.9787**	0.9486	**0.9787**	0.9751	0.9748	**0.9787**	**0.9787**
ecoliIM	0.8167	0.8882	0.8860	0.8795	**0.9060**	0.8811	0.8809
ecoliMU	0.7709	0.7600	0.8092	0.8661	0.8137	**0.8671**	0.8579
ecoliOM	0.8073	0.8220	0.8749	0.8412	0.8010	0.8725	**0.9291**
german	0.5759	0.6437	**0.6753**	0.6410	0.6636	0.6658	0.6419
glassBWFP	0.8138	0.6652	0.7551	0.8216	0.7599	0.7971	**0.8425**
glassBWNFP	0.6934	0.5648	0.7353	0.7511	**0.7631**	0.7427	0.7235
glassNW	0.8942	0.8101	**0.9505**	0.9239	0.9373	0.9344	0.9321
glassVWFP	0.5286	0.6755	0.6884	0.6994	0.7572	0.4930	**0.7816**
haberman	0.4280	0.4329	0.6089	**0.6832**	0.6292	0.6022	0.6206
new-thyroid	0.9048	0.9132	0.8810	0.9193	**0.9492**	0.9414	0.9463
pageblocks(2,4,5)-3	0.9270	0.9327	0.9260	**0.9991**	**0.9991**	0.9807	**0.9991**
pima	0.6908	0.6457	0.7161	0.7155	0.6990	**0.7181**	0.7179
segment1	0.9852	0.9728	0.9849	0.9918	0.9947	0.9965	0.9891
vehicle	0.9172	0.8737	0.9118	0.9202	0.9216	**0.9241**	0.9239
vowel0	**0.9808**	0.9360	**0.9808**	0.9657	0.9764	0.9671	0.9734
yeast(1)	0.4121	0.5979	0.3414	0.5399	0.6073	**0.6883**	0.6271
yeast(2)	0.1155	0.7038	0.2151	0.6783	0.6940	**0.7477**	0.6846
yeast(3)	0.7343	0.8653	0.8313	0.7983	**0.8890**	0.8649	0.8759
yeast(4)	0.0000	0.0000	0.1144	0.3737	**0.4509**	0.3044	0.3749
yeastCYT-POX	0.0699	**0.7245**	0.1000	0.5585	0.6156	0.6176	0.6489
yeastNUC-POX	0.5828	0.6151	0.5536	**0.6974**	0.6630	0.5647	0.6819
yeastPOX	0.0000	0.6238	0.0000	0.5718	0.5408	**0.6410**	0.6154
Average	0.6347	0.7196	0.6763	0.7733	0.7839	0.7749	**0.7947**

Fig. 3 in Appendix A illustrates the comparison of EUSTSS with the remaining techniques considered in this study in terms of *GM* accuracy over test data and using C4.5 as classifier.

Table 6 shows the average number of leaves obtained by C4.5 in each data set.

Observing Tables 4–6, we can make the following analysis:

- In training data, the results are mainly favourable to the SMOTE and SMOTE + ENN algorithms. Nevertheless, when we take into account the results obtained in test data, we see that SMOTE, in average, loses performance with respect to the hybrid techniques and EUSTSS. This points out that, in spite of the fact that all the techniques used produce overlearning, the one produced by SMOTE is more remarkable.
- EUSTSS proposal obtains the best average result in *GM* evaluation measure. It clearly outperforms the other under-sampling methods (OSS and NCL) and it improves the accuracy even when comparing with over-sampling approaches.
- Over-sampling techniques obtain better accuracy than under-sampling procedures in combination with C4.5 (see [3]), but they cannot outperform EUSTSS proposal.
- Except for NCL, EUSTSS produces decision trees with lower number of leaves than the remaining methods. Although the combination NCL + C4.5 yields smaller trees, its accuracy in *GM* is

Table 6
Average number of leaves obtained by C4.5 decision tree.

Data set	None	NCL	OSS	SMOTE	SMOTE + ENN	SMOTE + TL	EUSTSS
abalone9-18	8.10	6.50	7.30	57.50	57.30	52.60	6.30
dermatology2	10.6	5.4	8.9	15.5	14.3	14.5	7.2
ecoliCP-IM	2.00	2.50	2.00	2.90	3.10	2.00	2.00
ecoliIM	5.30	5.10	6.20	10.40	10.10	10.40	6.00
ecoliMU	10.00	5.80	6.50	16.70	13.10	14.00	5.40
ecoliOM	3.90	3.40	4.40	7.80	6.60	6.80	5.40
german	91.00	35.30	57.60	159.90	121.00	82.40	33.60
glassBWFP	12.20	5.80	6.70	15.70	10.40	10.40	7.00
glassBWNFP	12.40	5.50	11.60	19.90	15.90	15.90	9.60
glassNW	6.70	4.10	4.40	9.70	6.90	7.10	5.60
glassVWFP	7.50	6.10	8.40	13.40	13.10	13.50	6.90
haberman	2.60	3.90	8.70	16.10	18.20	18.00	5.70
new-thyroid	4.10	2.60	4.30	4.90	4.90	5.00	4.30
pageblocks(2,4,5)-3	4.7	3.1	4.7	4.2	4.2	4.2	4
pima	22.40	16.10	24.60	39.50	38.90	34.90	14.50
segment1	10	8.9	12.4	12.5	12.3	12.6	7.5
vehicle	20.60	12.50	16.30	28.40	23.40	22.50	11.10
vowel0	7.80	5.00	7.80	10.70	11.40	10.50	7.90
yeast(1)	3	2.2	3.2	21.2	21.9	19.2	8.2
yeast(2)	3	3.9	3.1	38.9	39	36.7	7
yeast(3)	5	4.2	3.3	32.6	29.5	28.8	5
yeast(4)	1.4	1.3	5	58.2	61.7	54.2	7.4
yeastCYT-POX	1.70	3.70	2.30	23.30	19.70	21.20	7.60
yeastNUC-POX	2.9	4.2	3	15.1	15.9	18.5	8
yeastPOX	0	2	0	34.7	36.2	36.8	5
Average	10.36	**6.36**	8.91	26.79	24.36	22.11	7.93

worse than the one obtained by EUSTSS and over-sampling approaches.
- Over-sampling techniques force C4.5 to produce big trees. This fact is not desirable when our interest lies in interpretable models.

We have included a second type of table accomplishing a statistical comparison of methods over multiple data sets. Demšar [12] recommends a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers. We will use two non-parametric procedures for conducting the comparisons. One of them is Wilcoxon Signed-Ranks Test [36]. It is a pairwise test which can be used for comparing two algorithms. The second one is the Holm's procedure [27], which is a multiple comparison procedure used for contrasting the results obtained by a control algorithm against a set of algorithms. It is a $1 \times n$ comparison procedure which controls the family wise error rate [36] and it should be used when we want to compare a proposal that obtains the best results in a certain performance measure. Table 7 collects the results of applying Wilcoxon's and Holm's tests between our proposed methods and the rest of re-sampling algorithms studied in this paper over the 25 data sets considered. This table is divided into three parts: In the first part, the measure of performance used is the accuracy classification in test set through *GM* and the Wilcoxon's test is conducted. In the second part, we accomplish Wilcoxon's test by using as performance measure the number of leaves yielded by C4.5. Finally, the third part contains the results of Holm's test over *GM* evaluation measure. Note that Holm's test cannot be applied when comparing the number of leaves yielded by the trees, because by considering this performance measure, the NCL algorithm outperforms our proposal and a $1 \times n$ comparison has no sense. Each part of this table contains one column, representing our proposed methods, and N_a rows where N_a is the number of algorithms considered in this study. In each one of the cells, three symbols can appear: +, = or −. They represent that the proposal outperforms (+), is similar (=) or is worse (−) in performance than the algorithm which appears in the column (Table 7). The value in parentheses is the p-value obtained in the comparison and the level of significance considered is $\alpha = 0.05$.

Table 7
Non-parametric statistical tests results over *GM* and number of rules using C4.5.

Algorithm	EUSTSS		
	Wilcoxon		Holm
	GM	Num. leaves	*GM*
None	+ (.000)	= (.447)	+ (.000)
NCL	+ (.000)	− (.001)	+ (.000)
OSS	+ (.001)	= (.316)	+ (.005)
SMOTE	+ (.011)	+ (.000)	= (.248)
SMOTE + ENN	= (.391)	+ (.000)	= (1.000)
SMOTE + TL	= (.317)	+ (.000)	= (1.000)

We make a brief analysis of results summarized in Table 7:

- The use of Wilcoxon's and Holm's tests confirms the improvement caused by EUSTSS over OSS and NCL under-sampling methods. Curiously, it statistically outperforms SMOTE considering a pairwise comparison. We have seen in Table 5 that SMOTE obtains a similar average *GM* to SMOTE + TL, but Wilcoxon's test indicates us that SMOTE has an irregular behaviour depending on the data sets.
- In the case of interpretability, Wilcoxon's test confirms the results observed in Table 6. The combination EUSTSS + C4.5 yields a low number of rules.
- EUSTSS outperforms OSS, NCL and SMOTE in *GM* measure and behaves similarly to SMOTE + TL and SMOTE + ENN. However, the number of leaves produced by C4.5 when it is applied after EUSTSS is much lower than the produced by SMOTE and its hybridizations. EUSTSS allows C4.5 to induce very precise trees with small size.

4.3. Results and analysis for PART

Tables 8 and 9 show the results in training and test data obtained by the re-sampling approaches compared by means of *GM* evaluation measure. The column denoted by *none* corresponds to the case in which no re-sampling is performed

Table 8
Results obtained by PART using *GM* evaluation measure over training data.

Data set	None	NCL	OSS	SMOTE	SMOTE + ENN	SMOTE + TL	EUSTSS
abalone9-18	0.6828	0.8077	0.6243	**0.9316**	0.8634	0.9236	0.8345
dermatology2	0.9761	0.9750	0.9363	**0.9892**	0.9855	0.9841	0.9830
ecoliCP-IM	**0.9948**	0.9864	0.9302	0.9935	0.9865	0.9804	0.8920
ecoliIM	0.9061	0.9232	0.9079	**0.9418**	0.9254	0.9307	0.9116
ecoliMU	0.7950	0.9149	0.8477	**0.9641**	0.9235	0.9563	0.9250
ecoliOM	0.9775	0.9873	0.9294	**0.9879**	0.9664	0.9773	0.9787
german	0.9368	0.8525	0.7730	**0.9522**	0.8131	0.8818	0.7406
glassBWFP	0.9475	0.8661	0.7860	0.9246	0.8927	0.9035	**0.9251**
glassBWNFP	0.8154	0.8674	0.6698	**0.9145**	0.8535	0.8939	0.8747
glassNW	0.9793	0.9672	0.7973	**0.9862**	0.9678	0.9631	0.9778
glassVWFP	0.9062	0.9560	0.7902	**0.9701**	0.9062	0.9624	0.9358
haberman	0.5842	0.6973	0.5209	**0.7321**	0.7212	0.7050	0.6389
new-thyroid	**0.9923**	0.9909	0.9610	0.9907	0.9828	0.9871	0.9689
pageblocks(2,4,5)-3	0.9960	0.9956	0.9542	0.9998	0.9980	**1.0000**	0.9945
pima	0.7262	0.7937	0.7180	**0.7964**	0.7910	0.7789	0.6963
segment1	0.9983	0.9986	0.9835	**0.9993**	0.9977	0.9987	0.9856
vehicle	0.9899	0.9767	0.9477	**0.9950**	0.9820	0.9852	0.9699
vowel0	0.9963	0.9962	0.9692	0.9970	**0.9994**	0.9972	0.9725
yeast(1)	0.4147	0.6033	0.7491	0.9165	0.8762	**0.9556**	0.9313
yeast(2)	0.4637	0.4655	0.7929	0.8851	0.8695	**0.9044**	0.8974
yeast(3)	0.8947	0.9094	0.9127	0.9496	0.9303	**0.9618**	0.9502
yeast(4)	0.3732	0.4846	0.1601	0.7812	0.6616	**0.7856**	0.7839
yeastCYT-POX	0.3318	0.2663	0.8208	0.9387	0.8885	**0.9425**	0.9277
yeastNUC-POX	0.3556	0.4376	0.8102	0.9162	0.8861	0.9259	**0.9605**
yeastPOX	0.0000	0.0000	0.7362	**0.8755**	0.8201	0.8602	0.8590
Average	0.7614	0.7888	0.8011	**0.9332**	0.8995	0.9258	0.9006

Table 9
Results obtained by PART using *GM* evaluation measure over test data.

Data set	None	NCL	OSS	SMOTE	SMOTE + ENN	SMOTE + TL	EUSTSS
abalone9-18	0.4305	0.3741	0.4668	0.6047	0.5401	**0.6355**	0.5862
dermatology2	0.8776	0.8791	0.8882	0.9409	0.8855	0.9199	**0.9672**
ecoliCP-IM	0.9717	**0.9787**	0.9201	0.9751	**0.9787**	0.9606	0.8827
ecoliIM	0.8335	0.8687	0.8740	0.8651	0.8698	0.8805	**0.8806**
ecoliMU	0.6607	0.7921	0.7652	0.8436	**0.8648**	0.8447	0.8073
ecoliOM	0.7193	0.8144	0.8311	0.9014	0.7979	**0.9535**	0.8710
german	0.6305	0.6439	0.6137	0.6319	0.6148	**0.6453**	0.6126
glassBWFP	0.8136	0.7957	0.6973	0.8046	0.8102	0.7985	**0.8302**
glassBWNFP	0.6105	**0.7560**	0.5750	0.7371	0.6884	0.7136	0.7400
glassNW	0.8963	**0.9446**	0.7370	0.9131	0.9273	0.9088	0.9213
glassVWFP	0.6019	0.6928	0.4838	0.7019	**0.7638**	0.5089	0.7360
haberman	0.5161	0.6111	0.4754	0.6417	**0.6513**	0.5765	0.5478
new-thyroid	0.8891	0.9224	**0.9393**	0.9252	0.9204	0.9261	0.9231
pageblocks(2,4,5)-3	0.9553	0.9525	0.9327	0.9807	0.9624	0.9807	**0.9914**
pima	0.6867	0.6967	0.6651	0.7145	**0.7251**	0.7134	0.6373
segment1	0.9890	0.9810	0.9774	0.9911	**0.9921**	0.9893	0.9838
vehicle	0.9344	0.9271	0.9059	0.9308	0.9388	**0.9530**	0.9329
vowel0	0.9557	0.9557	0.9040	**0.9706**	0.9665	0.9557	0.9232
yeast(1)	0.2113	0.3105	0.5734	0.6219	**0.6774**	0.6061	0.5967
yeast(2)	0.1155	0.2151	0.6266	0.6787	**0.7248**	0.6418	0.6871
yeast(3)	0.8156	0.8700	0.8658	0.8484	**0.8926**	0.8651	0.8492
yeast(4)	0.0000	0.1147	0.0553	**0.3950**	0.1122	0.1845	0.2934
yeastCYT-POX	0.0000	0.0000	**0.8097**	0.4960	0.7181	0.7451	0.7502
yeastNUC-POX	0.2121	0.2414	0.5879	0.6928	0.6642	0.6239	**0.7254**
yeastPOX	0.0000	0.0000	0.6238	0.5709	0.6387	0.5439	**0.7016**
Average	0.6131	0.6535	0.7118	**0.7751**	0.7731	0.7630	**0.7751**

previous to PART. The best case in each data set is remarked in bold.

Fig. 4 in Appendix A illustrates the comparison of EUSTSS with the remaining techniques considered in this study in terms of *GM* accuracy over test data and using PART as classifier.

Table 10 shows the average number of leaves obtained by C4.5 in each data set.

Observing Tables 8–10, we can make the following analysis:

- In training data, the results are mainly favourable to the SMOTE and SMOTE + TL algorithms. In the case of PART, the overlearning in training data is less notorious than in the case of C4.5.
- EUSTSS proposal obtains the best average result in *GM* measure together with SMOTE. It again outperforms the other under-sampling methods (OSS and NCL) and it achieves similar rates of accuracy when comparing with over-sampling approaches.
- Except for OSS, EUSTSS produces smaller rule bases than the remaining methods. Although the combination OSS + PART yields the lowest number of rules, the accuracy in *GM* is lower than the achieved by EUSTSS.

Table 10
Average number of rules obtained by PART.

Data set	None	NCL	OSS	SMOTE	SMOTE + ENN	SMOTE + TL	EUSTSS
abalone9-18	8.30	9.10	5.70	29.10	28.00	27.00	4.90
dermatology2	7.10	5.80	3.40	9.60	8.10	9.90	3.10
ecoliCP-IM	4.10	3.60	2.60	4.80	2.60	4.10	3.50
ecoliIM	5.90	5.30	2.80	7.40	6.20	6.30	4.30
ecoliMU	6.00	5.50	4.20	9.30	8.10	6.90	4.60
ecoliOM	4.50	3.90	3.20	4.40	4.40	4.30	3.70
german	108.00	66.40	56.50	128.50	76.40	100.70	40.10
glassBWFP	7.50	5.00	3.90	7.90	6.60	7.10	5.20
glassBWNFP	5.20	6.50	4.80	9.00	7.40	8.30	5.50
glassNW	5.50	3.90	3.00	6.00	5.20	5.00	4.40
glassVWFP	6.50	6.30	4.50	9.20	8.90	8.00	6.10
haberman	3.40	6.10	3.20	7.30	8.20	9.90	4.20
new-thyroid	4.10	3.60	2.10	4.20	4.10	4.40	3.20
pageblocks(2,4,5)-3	4.00	4.00	2.00	2.50	2.60	2.60	2.90
pima	7.40	10.70	7.10	11.50	12.70	12.80	5.10
segment1	7.90	7.80	6.20	7.50	7.10	7.60	6.50
vehicle	13.70	11.50	9.50	16.40	14.10	13.70	8.70
vowel0	5.80	5.80	5.00	7.40	7.50	7.80	4.70
yeast(1)	4.00	4.60	2.10	12.10	11.70	13.90	5.60
yeast(2)	4.80	4.40	4.00	20.10	21.50	18.20	5.30
yeast(3)	5.00	3.50	4.20	14.30	14.70	14.70	4.50
yeast(4)	4.60	5.10	2.00	29.80	29.90	28.50	4.80
yeastCYT-POX	3.30	2.80	3.10	13.50	12.00	12.50	5.00
yeastNUC-POX	3.40	3.60	3.40	9.30	10.20	11.30	6.20
yeastPOX	1.00	1.00	2.00	24.10	19.90	21.30	4.60
Average	9.64	7.83	**6.02**	16.21	13.52	14.67	6.27

Table 11
Non-parametric statistical tests results over *GM* and number of rules using PART.

Algorithm	EUSTSS		
	Wilcoxon		Holm
	GM	Num. rules	*GM*
None	+ (.001)	= (.174)	+ (.000)
NCL	+ (.048)	= (.339)	+ (.000)
OSS	+ (.001)	− (.001)	+ (.020)
SMOTE	= (.667)	+ (.000)	= (1.000)
SMOTE + ENN	= (.989)	+ (.000)	= (1.000)
SMOTE + TL	= (.925)	+ (.000)	= (1.000)

- Over-sampling techniques force PART to produce many rules as the previous case.

Table 11 includes the results of applying the non-parametric statistical test between our proposed methods and the rest of re-sampling algorithms studied in this paper over the 25 data sets considered. It follows the same structure as Table 7.

We make a brief analysis of results summarized in Table 11:

- The use of Wilcoxon's test confirms the improvement caused by EUSTSS over OSS and NCL under-sampling methods. In the case of PART, SMOTE is more robust than in the C4.5 case and it lets to obtain accurate sets of rules without the requirement of hybridization with noise filters (ENN) or under-sampling techniques (TL).
- When we refer to interpretability, Wilcoxon's test again confirms the results observed in Table 10. The combination EUSTSS + PART yields a low number of rules.
- EUSTSS outperforms OSS and NCL in *GM* measure and behaves similarly to SMOTE and hybridizations. However, the number of rules produced by PART when it is applied after EUSTSS is much lower than the produced by SMOTE. As in the C4.5 case, EUSTSS allows PART to obtain very accurate sets of rules with small size.

5. Concluding remarks

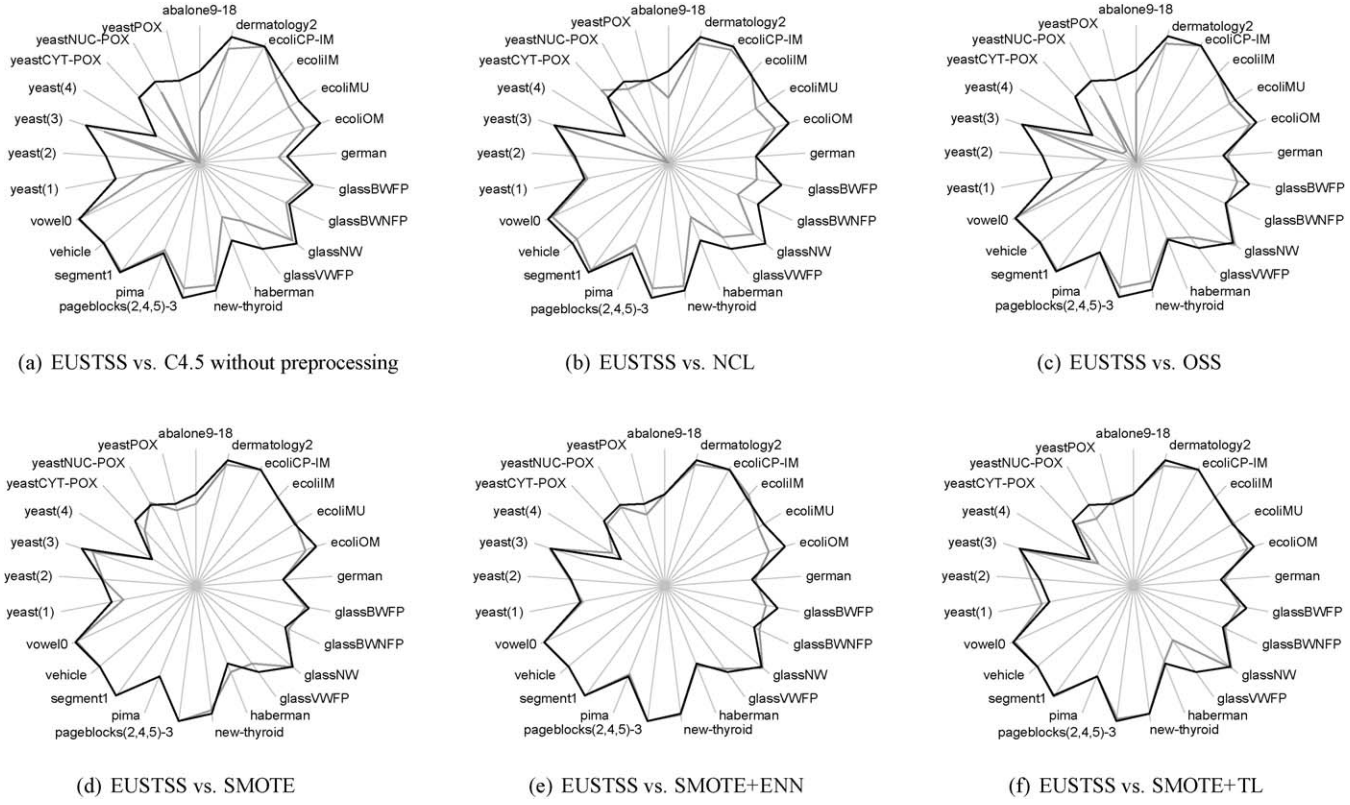
The purpose of this paper is to present a proposal of evolutionary training set selection algorithm for being applied over imbalanced data sets to improve the performance of decision tree or rule based induction classifiers. The study has been performed by using the C4.5 decision tree classifier and PART rule induction classifier. The results shows that our proposal allows to each one of the classifiers used to obtain very accurate models (trees or rule bases) with a low number of leaves or rules. The effectiveness of the models obtained is very competitive with respect to advanced hybrids of over-sampling. The proposal offers more accurate models than the offered by other under-sampling techniques, and the interpretability of the models obtained is increased due to the fact that the tree or rule bases yielded are made up by a lower number of leaves/rules.

Acknowledgement

This work was supported by TIN2005-08386-C05-01.

Appendix A. Star plot representations: EUSTSS vs. remaining methods

See Figs. 3 and 4.



(a) EUSTSS vs. C4.5 without preprocessing

(b) EUSTSS vs. NCL

(c) EUSTSS vs. OSS

(d) EUSTSS vs. SMOTE

(e) EUSTSS vs. SMOTE+ENN

(f) EUSTSS vs. SMOTE+TL

Fig. 3. Results obtained for C4.5 considering *GM* in test data.

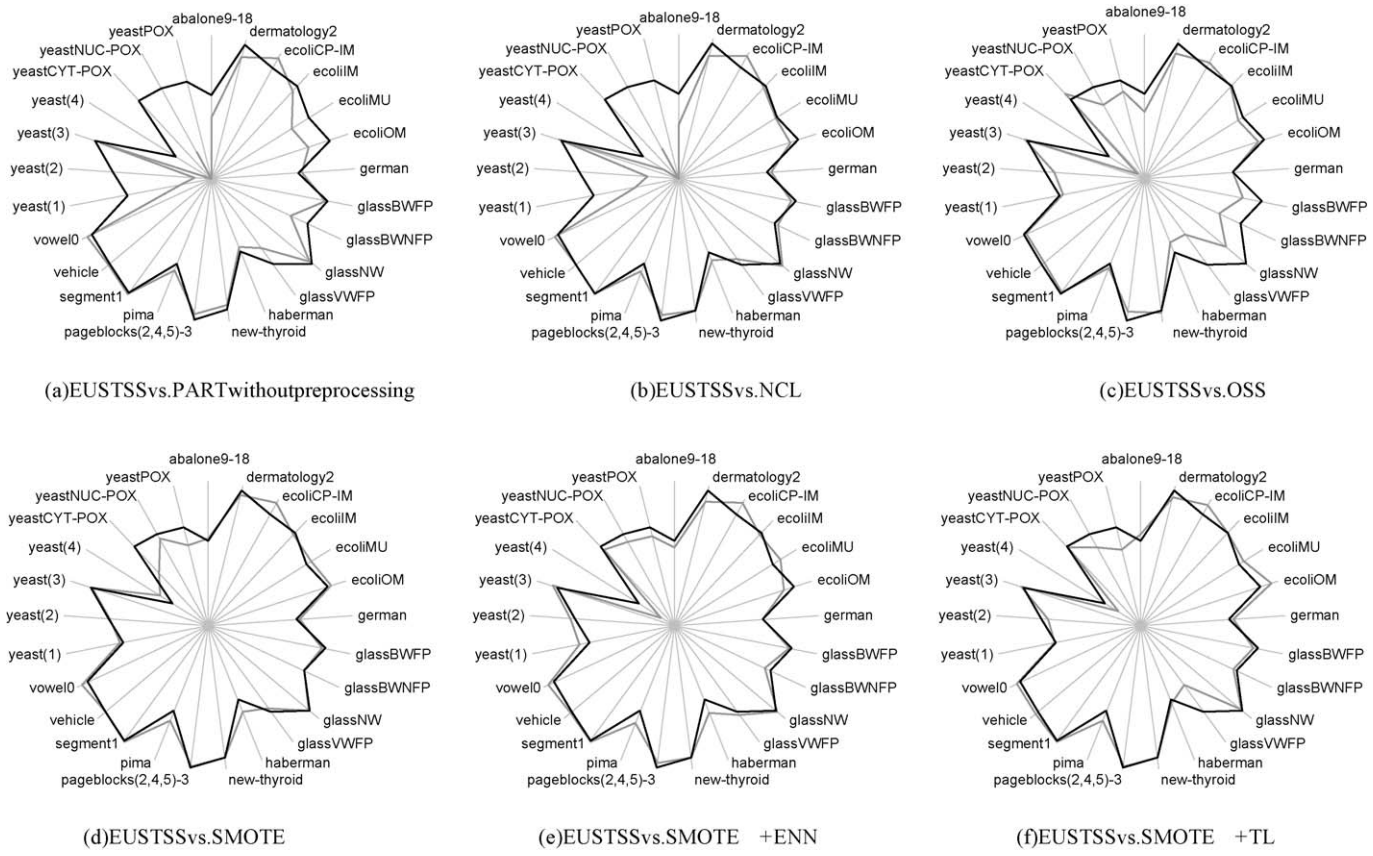


Fig. 4. Results obtained for PART considering *GM* in test data.

References

[1] A. Asuncion, D. Newman, UCI machine learning repository (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html.

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

[3] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for bal ancing machine learning training data, SIGKDD Explorations 6 (1) (2004) 20–29.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, 1984.

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

[6] J.R. Cano, F. Herrera, M. Lozano, On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining, Applied Soft Computing 6 (3) (2006) 323–332.

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

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

[9] N.V. Chawla, D.A. Cieslak, L.O. Hall, A. Joshi, Automatically countering imbalance and its empirical relationship to cost, Data Mining and Knowledge Discovery 17 (2008) 225–252.

[10] N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explorations 6 (1) (2004) 1–6.

[11] S. Dehuri, S. Patnaik, A. Ghosh, R. Mall, Application of elitist multiobjective genetic algorithm for classification rule generation, Applied Soft Computing 8 (1) (2008) 477–487.

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

[13] J. Doucette, M.I. Heywood, Gp classification under imbalanced data sets: active sub-sampling and auc approximation, in: EuroGP, Lecture Notes in Computer, Science, vol. 4971, 2008, 266–277.

[14] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer-Verlag, 2003 4971.

[15] L.J. Eshelman, 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, 1991, 265–283.

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

[17] T. Fawcett, PRIE: a system for generating rulelists to maximize roc performance, Data Mining and Knowledge Discovery 17 (2) (2008) 207–224.

[18] A. Fernández, S. García, M.J. del Jesus, F. Herrera, A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets, Fuzzy Sets and Systems 159 (18) (2008) 2378–2398.

[19] E. Frank, I.H. Witten, Generating accurate rule sets without global optimization, in: ICML'98: Proceedings of the Fifteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, (1998), pp. 144–151.

[20] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

[21] A. Freitas, A. da Costa Pereira, P. Brazdil, Cost-sensitive decision trees applied to medical data, in: DaWaK, Lecture Notes in Computer Science, 2007, 303–312.

[22] S. García, J.R. Cano, F. Herrera, A memetic algorithm for evolutionary prototype selection: a scaling up approach, Pattern Recognition 41 (8) (2008) 2693–2709.

[23] S. García, F. Herrera, Evolutionary under-sampling for classification with imbalanced data sets: proposals and taxonomy, Evolutionary Computation (in press).

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

[25] C. Guerra-Salcedo, S. Chen, D. Whitley, S. Smith, Fast and accurate feature selection using hybrid genetic strategies, in: Proceedings of the International Conference on Evolutionary Computation, 1999, pp. 177–184.

[26] J. Han, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

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

[28] M. Kretowski, M. Grzes, Evolutionary induction of decision trees for misclassification cost minimization, in: ICANNGA (1), Lecture Notes in Computer Science, 2007.

[29] M. Kubat, R.C. Holte, S. Matwin, Machine learning for the detection of oil spills in satellite radar images, Machine Learning 30 (2–3) (1998) 195–215.

[30] M. Kubat, S. Matwin, Addressing the course of imbalanced training sets: one-sided selection, in: ICML'97: Proceeding of the Fourteenth International Conference on Machine Learning, 1997, 179–186.

[31] J. Laurikkala, Improving identification of difficult small classes by balancing class distribution, in: AIME'01: Proceedings of the Eighth Conference on AI in Medicine in Europe, 2001, pp. 63–66.

[32] C.X. Ling, V.S. Sheng, Test strategies for cost-sensitive decision trees, IEEE Transactions on Knowledge and Data Engineering 18 (8) (2006) 1055–1067 (senior Member-Qiang Yang).
[33] E. Marchiori, Hit miss networks with applications to instance selection, Journal of Machine Learning Research 9 (2008) 997–1017.
[34] A. Orriols-Puig, E. Bernad'o-Mansilla, Evolutionary rule-based systems for imbalanced data sets, Soft Computing 13 (3) (2009) 213–225.
[35] J.R. Quinlan, C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning), Morgan Kaufmann, 1993.
[36] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC, 2006.
[37] A. Tajbakhsh, M. Rahmati, A. Mirzaei, Intrusion detection using fuzzy association rules, Applied Soft Computing 9 (2) (2009) 462–469.
[38] F.A. Thabtah, P.I. Cowling, A greedy classification algorithm based on association rule, Applied Soft Computing 7 (3) (2007) 1102–1111.
[39] I. Tomek, Two modifications of cnn., IEEE Transactions on Systems, Man and Communications 6 (1976) 769–772.
[40] B. Verma, P. Zhang, A novel neural-genetic algorithm to find the most significant combination of features in digital mammograms, Applied Soft Computing 7 (2) (2007) 612–625.
[41] G.M. Weiss, F.J. Provost, Learning when training data are costly: the effect of class distribution on tree induction, Journal of Artificial Intelligence Research 19 (2003) 315–354.
[42] D. Whitley, R. Beveridge, C. Guerra, C. Graves, Messy genetic algorithms for subset feature selection, in: Proceedings of the International Conference on Genetic Algorithms, 1998, pp. 568–575.
[43] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics 2 (1972) 408–421.
[44] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Machine Learning 38 (3) (2000) 257–286.
[45] Q. Yang, X. Wu, 10 challenging problems in data mining research, International Journal of Information Technology & Decision Making 5 (4) (2006) 597–604.
[46] H. Yan, J. Zheng, Y. Jiang, C. Peng, S. Xiao, Selecting critical clinical features for heart diseases diagnosis with a real-coded genetic algorithm, Applied Soft Computing 8 (2) (2008) 1105–1111.
[47] S. Zhang, L. Liu, X. Zhu, C. Zhang, A strategy for attributes selection in cost-sensitive decision trees induction, in: CITWORKSHOPS'08: Proceedings of the 2008 IEEE Eighth International Conference on Computer and Information Technology Workshops, IEEE Computer Society, Washing-ton, DC, USA, (2008), pp. 8–13.