Neurocomputing 286 (2018) 179–192

Contents lists available at ScienceDirect

Neurocomputing

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

A study on combining dynamic selection and data preprocessing for imbalance learning

Anandarup Roy[a], Rafael M.O. Cruz[a,*], Robert Sabourin[a], George D.C. Cavalcanti[b]

[a] École de Technologie Supérieure, University of Quebec, Montreal, Quebec, Canada
[b] Centro de Informática, Universidade Federal de Pernambuco, Recife, PE, Brazil

ARTICLE INFO

Article history:
Received 27 April 2017
Revised 8 October 2017
Accepted 24 January 2018
Available online 2 February 2018

Communicated by Dr. Xiaofeng Zhu

Keywords:
Imbalanced learning
Ensemble of classifiers
Multi-class imbalance
Dynamic ensemble selection
Preprocessing
SMOTE

ABSTRACT

In real life, classifier learning may encounter a dataset in which the number of instances of a given class is much higher than for other classes. Such imbalanced datasets require special attention because traditional classifiers generally favor the majority class which has a large number of instances. Ensemble classifiers, in such cases, have been reported to yield promising results. Most often, ensembles are specially designed for data level preprocessing techniques that aim to balance class proportions by applying under-sampling and/or over-sampling. Most available studies concentrate on static ensembles designed for different preprocessing techniques. Contrary to static ensembles, dynamic ensembles became popular thanks to their performance in the context of ill defined problems (small size datasets). A dynamic ensemble includes a dynamic selection module for choosing the best ensemble given a test instance. This paper experimentally evaluates the argument that dynamic selection combined with a preprocessing technique can achieve higher performance than static ensemble for imbalanced classification problems. For this evaluation, we collect 84 two-class and 26 multi-class datasets of varying degrees of class-imbalance. In addition, we consider five variations of preprocessing methods and four dynamic selection methods. We further design a useful experimental framework to integrate preprocessing and dynamic selection. Our experiments show that the dynamic ensemble improves the F-measure and the G-mean as compared to the static ensemble. Moreover, considering different levels of imbalance, dynamic selection methods secure higher ranks than other alternatives.

1. Introduction

Class-imbalance [1] refers to classification problems in which many more instances are available for certain classes than for others. Particularly, in a two-class scenario, one class contains the majority of instances (the *majority class*), while the other (the *minority class*) contains fewer instances. Imbalanced datasets may originate from real life problems including the detection of fraudulent bank account transactions or telephone calls [2,3], biomedical diagnosis [1], text classification [4], image retrieval [5] and, more recently, college student retention [6]. Moreover, class imbalance is also a common problems when dealing with data streams and concept drift [7–10].

A number of surveys examining recent advances in imbalance learning have been published over the past few years. For example, He and Ma [11], in their book, provided an overview of the nature of the imbalance learning problem and covered important relevant issues such as sampling strategies, active learning and streaming data. A more recent review was proposed by Branco et al. [12], where they concentrated on general issues of imbalanced predictive modeling. Among specialized discussions, Galar et al. [13] provided a thorough survey on ensemble learning for the imbalanced data problem. Krawczyk [14] recently discussed open challenges for imbalance learning, and mentioned seven potential areas of research on the subject.

When a dataset is imbalanced, conventional classifiers typically favor the majority class, and thus fail to correctly classify the minority instances, which then results in performance loss [15]. Now that the importance of class-imbalance in classification has been acknowledged, a variety of techniques have been developed to address the problem. These approaches can be broken down into the following four categories, according to Galar et al. [13].

1. *Algorithm level* approaches adapt existing classifier learning algorithms in order to tune them for class-imbalance. Examples include the modifications for k-nearest neighbors

* Corresponding author.
E-mail addresses: roy.anandarup@gmail.com (A. Roy), rafaelmenelau@gmail.com (R.M.O. Cruz), Robert.Sabourin@etsmtl.ca (R. Sabourin), gdcc@cin.ufpe.br (G.D.C. Cavalcanti).

https://doi.org/10.1016/j.neucom.2018.01.060
0925-2312/

[16,17], SVM [18] and Hellinger Distance Decision Trees (HDDT) [19].

2. *Data level* approaches [20] include preprocessing algorithms (e.g., SMOTE [21]) that re-balance the class distribution by resampling the data space, thus avoiding modifications of the learning algorithm. For this reason, they are the most commonly used approaches for handling imbalanced data [11,21,22]. In fact, in this paper, we consider preprocessing approaches for re-balancing. A recent survey in this context was prepared by Branco et al. [12], and in Section 3, we will discuss preprocessing based on it.
3. The *Cost-sensitive* learning framework approach, which includes cost-sensitive MLP [23] and RBF [24], lies in between the data and algorithm level approaches. It assigns different costs to instances, and modifies the learning algorithm to accept the costs. Well-known methods in this category are cost sensitive SVM [2] and the AdaCost family [25].
4. *Multiple classifiers ensemble* (MCE) approaches combine an ensemble learning algorithm [26] with one of the techniques (usually, a preprocessing technique) mentioned above. In this work, we explore ensemble learning for handling class-imbalance. An overview of related works is presented hereafter.

The key idea with ensemble learning is to train a pool of base classifiers on different versions of the training dataset and then aggregate their decisions to classify an unknown instance. In recent years, MCE has been introduced for solving the class-imbalance problem. In this context, MCE has mostly been combined with data level preprocessing approaches. For example, in the SMOTEBoost method [27], the SMOTE procedure is combined with the AdaBoost [28] ensemble algorithm. Similar boosting-based ensemble methods are RUSBoost [29] and RAMOBoost [30]. On the other hand, many approaches have used bagging [31] instead of boosting. The hybridization of bagging and preprocessing is usually simpler than integrating preprocessing in boosting. Unlike boosting, bagging does not compute weights, and does not therefore need to adapt the weight update formula. Examples in this regard include SMOTE-Bagging [32], where each classifier is built with a dataset processed by SMOTE. Another well-known method is UnderBagging [13], which performs a random under-sampling in each bagging iteration to equate the size of the classes. Other notable methods can be found in [13,33]. Bagging-based ensembles have produced satisfactory performances, as demonstrated in [33,34]. Based on this fact, and because of its simplicity, we apply bagging with preprocessing methods here to generate a diverse pool of classifiers.

In general, MCE is composed of three phases, namely, *generation, selection*, and *integration* [35]. The aim of the generation phase is to produce a pool consisting of base classifiers. All the above mentioned schemes combining preprocessing with MCE focus only on the generation phase. The generated base classifiers are integrated using a static ensemble approach (e.g., majority voting) [26]. However, during pool generation, different versions of the dataset have varying difficulties. Therefore, the base classifiers which are based on these datasets, have different individual qualities. According to Krawczyk [14], this issue will affect the local competency of individual base classifiers, and should be exploited before/during the integration phase. This observation leads us to the *dynamic ensemble* [36,37], which determines the competency of individual base classifiers. In the static ensemble, a unified ensemble is used for all test instances. Conversely, in a dynamic ensemble, given a test instance, only the competent base classifiers, based on a competence measure, are selected to form an ensemble. This strategy affects the selection phase of MCE, and is commonly known as dynamic selection (DS) [36–39]. DS has been studied for many years, and in a number of studies, its superiority over the static ensemble has been verified [36,39–41]. Readers wishing to delve deeper into this may consult the taxonomy of DS techniques proposed by Cruz et al. [37].

There is a dearth of articles covering the application of DS to imbalance learning. One relevant article by Xiao et al. [42] covered the problem of customer classification. They combined ensemble learning with cost-sensitive learning, and then proposed a novel DS method for handling imbalanced data. However, they did not explore different preprocessing variations combined with DS strategies. The advantage of applying a preprocessing is the resulting re-balancing of the corresponding training dataset. Following this re-balancing, the base classifiers more or less overcome the learning difficulties arising from imbalanced data. Moreover, due to its randomized nature, preprocessing enhances the diversity of the classifier pool [26]. On the other hand, since DS exploits the local competency of base classifiers, it may overcome the difficulties due to their varying quality [14]. These advantages gave us reason to believe that 1) an MCE that uses preprocessing at the pool generation phase and applies DS at the selection phase will produce improved results as compared to a static ensemble method, and 2) a preprocessing can influence a DS technique, and lead to an improvement of its performance. We performed extensive experiments to validate these two arguments. Our experiments consisted of 84 two-class and 26 multi-class imbalanced datasets, five preprocessing methods and four DS strategies. The area under the ROC curve (AUC), the F-measure and the G-mean are used as measures of performance. Our results reveal the improvement of DS over static ensemble, considering different degrees of class-imbalance. In particular, a combination of DS and preprocessing improves the F-measure and the G-mean considerably. We further considered the best performing DS and preprocessing methods in order to assess how much preprocessing influences the DS strategy. To that end, we used a synthetic two-class dataset. Our study shows that preprocessing helps improve the DS performance.

This rest of the paper is organized as follows. We start by reviewing DS strategies in Section 2, where we also describe the specific DS methods used in this paper. Section 3 introduces the preprocessing techniques we used in the experiments. In Section 4, we detail our experimental setup, and provide the guidelines we applied for using DS in the context of imbalanced distributions. In Section 5, we present and analyze the results obtained for binary imbalanced problems with 30 variations of ensemble procedures. Section 6 we present an analysis of DS techniques for dealing with multi-class imbalanced problems. Finally, we conclude our paper in Section 7 and outline possible directions for future research.

2. Overview of dynamic selection

A dynamic selection (DS) enables the selection of one or more base classifiers from a pool, given a test instance. Therefore, in this approach, it is assumed that the structure of the ensemble varies for each test instance. This is based on the assumption that each base classifier is an expert in a different local region in the feature space. Hence, the most competent classifiers should be selected in classifying a new instance. The notion of competence is used in DS as a way of selecting, from a pool of classifiers, the best classifiers to classify a given test instance. Usually, the competence of a base classifier is estimated based on a small region in the feature space surrounding a given test instance, called the *region of competence*. This region is formed using the k-nearest neighbors (KNN) technique, with a set of labeled samples, which can be either the training or validation set. This set is called the Dynamic Selection dataset (DSEL) [36]. To establish the competence, given a test instance and the DSEL, the literature reports a number of measures classified into two categories by Britto et al. [36]. Among the

categories, we focus on the individual-based measures, which consider individual base classifier accuracy for the region of competence. However, the competency measures are calculated differently by different methods in this category. For example, we consider methods in which the competency is measured by pure accuracy [38], by ranking of classifiers [43] or using oracle information [39]. Other possible alternatives were described in [36].

Instead of grouping DS strategies by competence measure, we may also group them by selection methodology. Currently, there are two kinds of DS strategies: dynamic classifier selection (DCS) and dynamic ensemble selection (DES). DCS selects a single classifier for a test instance whereas DES selects an ensemble of competent classifiers to classify a test instance. Both these strategies have been studied in recent years, and a number of papers are available examining them [35,39–41,44–46]. Among these, Ko et al. [39] introduced four DES strategies inspired by oracle, and further demonstrated the superiority of DES over DCS. The effectiveness of DES for small size datasets was studied in [40,44]. More recently, in [35,46], the authors predicted an optimal pool size prior to the selection phase in DES strategies.

In this paper, we consider two DCS and two DES strategies. These strategies are based on different notions of competence measure. For example, while both RANK and the LCA are DCS strategies, the former however, measures the competence based on classifier ranking, and the latter based on classifier accuracy. On the other hand, the two DES strategies (KNE and KNU) are based on oracle information. These DS strategies (except RANK) were recently used in context of imbalanced data by Xiao et al. [42]. They however considered only SMOTE in preprocessing the datasets. Next, we briefly describe the four DS strategies adopted in this paper.

- The Modified Classifier Rank (**RANK**) [36,43] is a DCS method that exploits ranks of individual classifiers in the pool for each test instance. The rank of a classifier is based on its local accuracy within a neighborhood of the test instance. More formally, given a test instance assigned to class C_i by a classifier, the ranking of the classifier is estimated as the number of consecutive nearest neighbors assigned to class C_i that have been correctly labeled. The most locally accurate classifier has the highest rank and is selected for classification.
- The Local Class Accuracy (**LCA**) [36,38] tries to estimate each classifiers accuracy in a local region around the given test instance and then uses the most locally accurate classifier to classify the test instance. The local accuracy is estimated for each base classifier as the percentage of correct classifications within the local region, but considering only those examples where the classifier predicted the same class as the one it gave for the test instance. By nature, LCA is a DCS method.
- Ko et al. [39] proposed the K-Nearest Oracles (KNORA) family of techniques, inspired by the Oracle [47] concept. Of their four proposed techniques, we consider the KNORA-Eliminate (**KNE**) and KNORA-Union (**KNU**). The KNE considers that a base classifier c_i is competent for the classification of the test instance $\boldsymbol{x}_j$ if c_i achieves a perfect accuracy for the whole region of competence. Only the base classifiers with a perfect accuracy are used during the voting scheme. On the other hand, in the KNU technique, the level of competence of a base classifier c_i is measured by the number of correctly classified instances in the defined region of competence. In this case, every classifier that correctly classified at least one instance can submit a vote.

These DS methods similarly use the base classifiers and the DSEL directly to evaluate their competence, given a test instance. Readers may note that two recent DS strategies, namely, META-DES [41] and META-DES.Oracle [48], have been reported to outperform many of the above DS strategies in several cases. These DS methods are based on meta-learning, and use a meta-classifier to assess whether a base classifier is competent. The meta-classifier requires training based on a separate meta-training dataset. The trained meta-classifier then uses DSEL to obtain competent base classifiers for a given test instance. Therefore, these two methods, unlike the above listed ones, take an indirect approach (via the meta-classifier) to evaluate competency. For this reason, we do not consider such DS strategies in this study.

A key factor in dynamic selection is the competence of a base classifier determined using the DSEL. As reported in [49], dynamic selection performance is very sensitive to DSEL. Looking deeper into this problem, Cruz et al. [50] proposed to apply a prototype selection scheme to eliminate possible noise from the DSEL. However, they did not focus on imbalanced datasets. The distribution of DSEL makes an important contribution to the classification of imbalanced data. If the distribution of DSEL itself becomes imbalanced, then there is a high probability that the region of competence for a test instance will become lopsided. With this in mind, we design a clear guideline here for using DS techniques in dealing with imbalanced data. Our protocol applies preprocessing for the DSEL to make it balanced. We will describe this later, in Section 4.3.

3. Preprocessing techniques for imbalance learning

Data preprocessing approaches include strategies that preprocess a given imbalanced dataset by changing the data distribution used according to the goal of the user. Afterwards, any standard algorithm can be applied to the pre-processed dataset.

According to Branco et al. [12], existing preprocessing approaches can be categorized into two main types: 1) methods that change the data distribution to obtain a more balanced dataset, and 2) methods that modify the training set distribution using information on misclassification costs, such that the learned model avoids costly errors. This latter strategy, proposed by Zadrozny et al. [51], represents a way of implementing cost-sensitive learning. Its major drawback is that it imposes the need to define misclassification costs, which are not usually available in the datasets. In this paper, we consider methods from the first category.

Changing the distribution of training data to compensate for poor representativeness of the minority class, is an effective solution for imbalanced problems, and a plethora of methods are available in this regards. Branco et al. [12] divided such methods into three categories, namely, stratified sampling, synthesizing new data, and combinations of the two previous methods. While the complete taxonomy is available in [12], we will center our attention on the methods that have been used with ensemble learning algorithms. Random under-sampling (RUS) [52] is one such method, which removes random instances from the majority class. RUS has been coupled with boosting (RUSBoost) [29] and with bagging [52]. A major drawback of RUS is that it can discard potentially useful data. Using RUS with ensemble learning may overcome this issue because instances that are absent in one iteration may be present during another. Błaszczyński and Stefanowski [34] empirically showed that integrating bagging with RUS is more powerful than doing so with over-sampling. Díez-Pastor et al. [33] also obtained satisfactory results by integrating RUS with boosting and bagging.

Another important preprocessing approach consists in the generation of new synthetic data. Synthesizing new instances has several known advantages [21], and a wide number of proposals are available for building new synthetic examples. In this context, a famous method that uses interpolation to generate new instances is SMOTE, proposed by Chawla et al. [21]. SMOTE over-samples the minority class by generating new synthetic data. A number of methods have been developed based on the principle

of SMOTE. Among them, Borderline-SMOTE [53], Safe-level SMOTE [54], ADASYN [55], RAMO [30] and Random balance [56] deserve mention. As we pointed out in Section 1, SMOTE has also been combined with boosting (SMOTEBoost) [27] and with bagging (SMOTE-Bagging) [32].

Coupling preprocessing approaches with ensemble learning methods has proven to be highly competitive and robust to difficult data. Almost all the preprocessing approaches mentioned above can be combined with a bagging-based ensemble. Therefore, selecting the best preprocessing methods from a set of such methods is a nontrivial task. In that regard, we eliminate RUS-based ensembles here for the following reasons. First, García et al. [20] observed that over-sampling consistently outperforms RUS for strongly imbalanced datasets. Secondly, according to Krawczyk [14], the issue of diversity comes into play when using RUS with bagging. Since RUS maintains the minority class intact, the ensemble may not exhibit enough diversity. This is a serious drawback that may affect all phases of the corresponding MCE. Finally, as we pointed out in Section 2, the distribution of DSEL plays a crucial role in dynamic selection. RUS may lead to issues in properly creating the DSEL. This aspect will be described in details in 4.3.

Having ruled out RUS, we consider the methods that generate synthetic instances to enrich the minority class. As discussed above, there are several different proposals in this category. In a recent article by Díez-Pastor et al. [33], bagging based ensembles of SMOTE and Random balance were reported as producing satisfactory outcomes, which motivates us to combine these preprocessing with bagging. Besides, we also considered RAMO, which can outperform a number of algorithms (including some SMOTE based methods), as observed by Chen et al. [30]. RAMO, however, has not ever been combined with bagging.

Next, we provide a brief description of our selected preprocessing methods.

- The Synthetic Minority Over-sampling Technique (**SMOTE**) [21], which creates artificial instances for the minority class. The process works as follows: Let $\boldsymbol{x}_i$ be an instance from the minority class. To create an artificial instance from $\boldsymbol{x}_i$, SMOTE first isolates the k-nearest neighbors of $\boldsymbol{x}_i$, from the minority class. Afterward, it randomly selects one neighbor and randomly generates a synthetic example along the imaginary line connecting $\boldsymbol{x}_i$ and the selected neighbor.
- The Ranked Minority Over-sampling (**RAMO**) [30], which performs a sampling of the minority class according to a probability distribution, followed by the creation of synthetic instances. The RAMO process works as follows: For each instance $\boldsymbol{x}_i$ in the minority class, its k_1 nearest neighbors (k_1 is a user defined neighborhood size) from the whole dataset are isolated. The weight r_i of $\boldsymbol{x}_i$ is defined as:

$$r_i = \frac{1}{1 + exp(-\alpha.\delta_i)}, \tag{1}$$

  where δ_i is the number of majority cases in the k-nearest neighborhood. Evidently, an instance with a large weight indicates that it is surrounded by majority class samples, and thus difficult to classify. This is different from SMOTE, where all the weights are the same.
  After determining all weights, the minority class is sampled using these weights to get a sampling minority dataset G. The synthetic samples are generated for each instance in G by using SMOTE on k_2 nearest neighbors where k_2 is a user-defined neighborhood size.
- The Random Balance (**RB**) [56], which relies on the amount of under-sampling and over-sampling that is problem specific and that has a significant influence on the performance of the classifier concerned. RB maintains the size of the dataset, but varies the proportion of the majority and minority classes, using a random ratio. This includes the case where the minority class is over represented and the imbalance ratio is inverted. SMOTE and random under-sampling are used to respectively increase or reduce the size of the classes to achieve the desired ratios. Given a dataset S, with minority class S_P and majority class S_N, the RB procedure can be described as follows[1]:
  1. The modified size, *newMajSize*, of the majority class, is defined by a random number generated between 2 and $|S| - 2$ (both inclusive). Accordingly, the modified size, *newMinSize*, of the minority class becomes $|S| - newMajSize$.
  2. If $newMajSize < |S_N|$, the majority class S'_N is created by RUS the original S_N so that the final size $|S'_N| = newMajSize$. Consequently, the new minority class S'_P is obtained from S_P using SMOTE to create $newMinSize - |S_P|$ artificial instances.
  3. Otherwise, S'_P is the class created by RUS S_P. On the other hand, S'_N is the class that includes artificial samples generated using SMOTE on S_N. Thus, finally, $|S'_P| = newMinSize$ and $|S'_N| = newMajSize$.

Among these techniques, SMOTE and RAMO over-sample the minority class in order to balance it with the majority class. For the multi-class datasets, the over-sampling methods are applied to generate samples for each minority class. RB, on the other hand, randomly applies under-sampling and over-sampling. In that sense, RB is a hybrid method. Moreover, repeated applications of RB produce datasets having a large imbalance ratio variability. Then, the set of classifiers, each trained during different iterations of RB, can possess sufficient diversity. Hence, RB is most efficient in conjunction with ensemble learning, as shown by Díez-Pastor et al. [56].

Table 1
Preprocessing methods used for classifier pool generation.

Bagging based methods		
Abbr.	Name	Description
Ba	Bagging	Simple bagging without preprocessing
Ba-RM100	Bagging+RAMO 100%	RAMO in each iteration to double the minority class
Ba-RM	Bagging+RAMO	RAMO in each iteration to make equal size for both classes.
Ba-SM100	Bagging+SAMOTE 100%	SMOTE in each iteration to double the minority class
Ba-SM	Bagging+SMOTE	SMOTE in each iteration to make equal size for both classes.
Ba-RB	Bagging+RB	RB in each iteration to randomly balance the two classes.

4. Experimental protocol

In this section, we provide a complete protocol for our experiments. We intend to demonstrate that a dynamic selection, combined with a preprocessing method, improves the performance of an ensemble for imbalance learning. From this viewpoint, we first enlist the dynamic selection and the preprocessing methods, in Section 4.1. Afterward, we describe the datasets and the evaluation procedure, in Section 4.2.

4.1. Ensemble methods for the experiments

In our experiments, the Bagging method was used as a baseline technique to promote diversity. All the preprocessing techniques were combined with Bagging during the pool generation phase (see Section 1). Table 1 lists such combinations.

[1] This description is adopted from [56]

Each of the methods in Table 1 generates a pool of base classifiers. During a static ensemble, these base classifiers were integrated using the average aggregation method implemented in Weka 3.8 [57]. In this implementation (used earlier in [33,58]), the base classifiers return posterior probabilities for the classes. These probabilities are averaged across the classifiers, and the most probable class is assigned. On the other hand, during dynamic selection, a preprocessing method was combined with the selected dynamic selection methods (see Section 2). Thus, for each preprocessing method, we have four combined versions, each named according to the corresponding dynamic selection method. For example, the combined versions of Ba-SM are Ba-SM+RANK, Ba-SM+LCA, Ba-SM+KNE and Ba-SM+KNU, respectively.

The pool size for all ensemble techniques was set to 100. The classifier used as a base classifier in all experiments was J48, which is the Java implementation of Quinlan's C4.5 [59], available in Weka 3.8. C4.5 is one of the most popular algorithms for handling imbalanced data [19,22,33]. Here, C4.5 was used with Laplace smoothing at the leaves, but without pruning and collapsing. This configuration was recommended by Chawla [60] and by Cieslak et al. [19] (where it was called C4.4).

The preprocessing techniques, RAMO and SMOTE, have user-specified parameters. In the case of RAMO, we used $k_1 = 10$, $k_2 = 5$ and $\alpha = 0.3$. For SMOTE and RB, the number of nearest neighbors was 5. These parameter settings were adopted from [33]. Finally, for all the dynamic selection methods, we used 7 nearest neighbors [41] to define the region of competence.

4.2. Tools

The Weka 3.8 [57] tool was used for the experiments, along with Matlab 8.4.0. The results were obtained with a 5×2 stratified cross-validation. Three criteria were used for performance evaluation, namely, 1) the area under the ROC curve (AUC) [61], 2) the F-measure [62] and 3) the Geometric Mean (G-mean) [63]. The F-measure and the G-mean are reported for the minority class. In this work, the AUC is computed from the Wilcoxon rank sum test statistic. The average of these measures over all 5×2 exercises are reported as final results.

4.3. The experimental framework

The complete framework for a single experiment is presented in Fig. 1. The original dataset was divided into two equal halves. One of them was set aside for testing, while the other half was used to train the base classifiers and to derive the DSEL for dynamic selection. During training, for each bagging iteration, a preprocessing method (*Preproc* in Fig. 1) was further applied. While bagging itself promotes diversity, the preprocessing injects even more of it by generating random synthetic instances for the minority class. For the multi-class datasets, the over-sampling methods are applied to generate samples for each minority class. Hence, *Preproc* helps reduce class-imbalance and increases diversity for the training datasets.

Let us now highlight the process of setting up the DSEL. Here, instead of dividing the training set, we augment it by preprocessing, to create the DSEL. As we can see in Fig. 1, the same preprocessing (*Preproc*) was applied for creating the DSEL and during the bagging iterations. Since we considered a single training dataset, the DSEL has an overlap with the datasets used during bagging iterations. However, the randomized nature of the preprocessing methods allows the DSEL not to be exactly same as the training datasets generated by bagging (followed by preprocessing). Thus, we avoid possible overfitting issues. During testing, the test data and the DSEL are made available for dynamic ensemble methods. The DSEL is then used to obtain the region of competence for a test instance. Afterward, the underlying DS strategy is applied to select the best classifier (considering DCS) or ensemble of classifiers (considering DES). Readers should note that, for a static ensemble, the DSEL is not required.

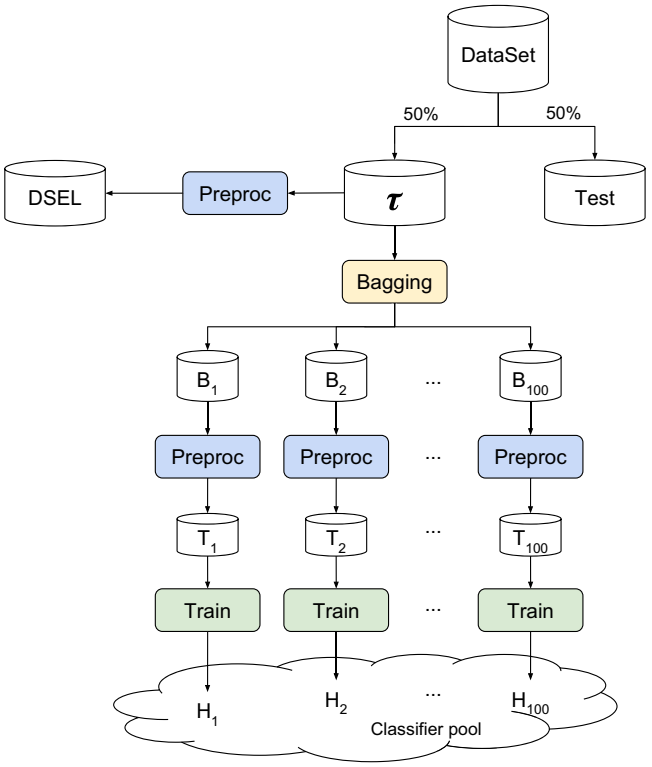


Fig. 1. The framework for training base classifiers and to prepare a DSEL for testing. Here, τ is the training data derived from the original dataset, B_i is the dataset generated from the ith bagging iteration, T_i is the dataset produced by preprocessing (*Preproc*) B_i and H_i is the ith base classifier.

Cruz et al. [49] observed that the size of the DSEL has an impact on dynamic selection. In fact, greater the number of instances in the DSEL, the higher the probability of selecting instances that are similar to the test instance. Hence, we applied a preprocessing to re-balance the DSEL. In this process, the preferred scenario has the preprocessing not decreasing the size of the DSEL. This is a vital reason for keeping RUS from consideration. The SMOTE and the RAMO methods synthesize new instances, thus ensuring that the DSEL will be large enough. On the other hand, the RB procedure may decrease the DSEL size, and may therefore produce degraded results, considering dynamic selection. We will verify this later in Section 5.

5. Experiment on binary imbalanced problems

For the binary imbalanced problems, we considered two collections of datasets. The HDDT collection contains 20 imbalanced datasets used in [19,33]. Table 2 presents the characteristics of this dataset. Secondly, we use the KEEL collection [33,64] of imbalanced datasets. This collection originally contained 66 binary imbalanced datasets, 64 of which we considered. Table 3 shows the characteristics of this dataset collection. The amount of imbalance for a given dataset is measured using the imbalance ratio (IR), which is the number of instances of the majority class per instance of the minority class. For our analysis, we grouped the datasets according to their IRs. We adopt the group definitions suggested by Fernández et al. [65]. A dataset has a low imbalance if its $IR < 3$; it has a medium imbalance if the IR lies between 3 and 9 (both inclusive), and it has a high imbalance if its $IR > 9$. Different groups

Table 2
Characteristics of the 20 datasets from the HDDT collection. Column #E shows the number of instances in the dataset, column #A the number of attributes, both numeric and nominal in the format (numeric/nominal), and column IR the imbalance ratio. The low imbalanced datasets are highlighted with dark gray whereas the medium imbalanced datasets are in light gray.

Dataset	#E	#A	IR	Dataset	#E	#A	IR
Credit-g	1000	(7/13)	2.33	Satimage	6430	(36/0)	9.29
German-numer	1000	(24/0)	2.33	Covtype	38500	(10/0)	13.02
Breast-y	286	(0/9)	2.36	PhosS	11411	(480/0)	17.62
Phoneme	5404	(5/0)	2.41	Cam	18916	(0/132)	19.08
Heart-v	200	(5/8)	2.92	Hypo	3163	(7/18)	19.95
Segment	2310	(19/0)	6	Oil	937	(49/0)	21.85
Estate	5322	(12/0)	7.37	Letter	20000	(16/0)	24.35
Pendigits	10992	(16/0)	8.63	Compustat	13657	(20/0)	25.26
Page	5473	(10/0)	8.77	Boundary	3505	(0/175)	27.5
Optdigits	5620	(64/0)	9.14	Ism	11180	(6/0)	42

Table 3
Characteristics of the 64 datasets from the KEEL collection. Column #E shows the number of instances in the dataset, column #A the number of attributes, both numeric and nominal in the format (numeric/nominal), and column IR the imbalance ratio. The low imbalanced datasets are highlighted with dark gray whereas the medium imbalanced datasets are in light gray.

Dataset	#E	#A	IR	Dataset	#E	#A	IR
Glass1	214	(9/0)	1.82	Ecoli-0-3-4-6_vs_5	205	(7/0)	9.25
Ecoli-0_vs_1	220	(7/0)	1.86	Ecoli-0-3-4-7_vs_5-6	257	(7/0)	9.28
Wisconsin	683	(9/0)	1.86	Yeast-0-5-6-7-9_vs_4	528	(8/0)	9.3529
Pima	768	(8/0)	1.87	Vowel0	988	(13/0)	9.9778
Iris0	150	(4/0)	2	Ecoli-0-6-7_vs_5	220	(6/0)	10
Glass0	214	(9/0)	2.06	Glass-0-1-6_vs_2	192	(9/0)	10.2941
Yeast1	1484	(8/0)	2.46	Ecoli-0-1-4-7_vs_2-3-5-6	336	(7/0)	10.586
Haberman	306	(3/0)	2.78	Led7digit-0-2-4-5-6-7-8-9_vs_1	443	(7/0)	10.973
Vehicle2	846	(18/0)	2.88	Ecoli-0-1_vs_5	240	(6/0)	11
Vehicle1	846	(18/0)	2.90	Glass-0-6_vs_5	108	(9/0)	11
Vehicle3	846	(18/0)	2.99	Glass-0-1-4-6_vs_2	205	(9/0)	11.0588
Glass-0-1-2-3_vs_4-5-6	214	(9/0)	3.20	Glass2	214	(9/0)	11.5882
Vehicle0	846	(18/0)	3.25	Ecoli-0-1-4-7_vs_5-6	332	(6/0)	12.28
Ecoli1	336	(7/0)	3.36	Cleveland-0_vs_4	177	(13/0)	12.6154
New-thyroid1	215	(5/0)	5.14	Ecoli-0-1-4-6_vs_5	280	(6/0)	13
Newthyroid2	215	(5/0)	5.14	Shuttle-c0_vs-c4	1829	(9/0)	13.87
Ecoli2	336	(7/0)	5.46	Yeast-1_vs_7	459	(7/0)	14.3
Glass6	214	(9/0)	6.38	Glass4	214	(9/0)	15.4615
Yeast3	1484	(8/0)	8.10	Ecoli4	336	(7/0)	15.8
Ecoli3	336	(7/0)	8.60	Page-blocks-1-3_vs_4	472	(10/0)	15.8571
Ecoli-0-3-4_vs_5	200	(7/0)	9	Abalone9-18	731	(7/1)	16.4048
Yeast-2_vs_4	514	(8/0)	9.0784	Glass-0-1-6_vs_5	184	(9/0)	19.4444
Ecoli-0-6-7_vs_3-5	222	(7/0)	9.0909	Shuttle-c2-vs-c4	129	(9/0)	20.5
Ecoli-0-2-3-4_vs_5	202	(7/0)	9.1	Yeast-1-4-5-8_vs_7	693	(8/0)	22.1
Glass-0-1-5_vs_2	172	(9/0)	9.1176	Glass5	214	(9/0)	22.7778
Yeast-0-3-5-9_vs_7-8	506	(8/0)	9.12	Yeast-2_vs_8	482	(8/0)	23.1
Yeast-0-2-5-6_vs_3-7-8-9	1004	(8/0)	9.1414	Yeast4	1484	(8/0)	28.098
Yeast-0-2-5-7-9_vs_3-6-8	1004	(8/0)	9.1414	Yeast-1-2-8-9_vs_7	947	(8/0)	30.5667
Ecoli-0-4-6_vs_5	203	(6/0)	9.15	Yeast5	1484	(8/0)	32.727
Ecoli-0-1_vs_2-3-5	244	(7/0)	9.1667	Ecoli-0-1-3-7_vs_2-6	281	(7/0)	39.143
Ecoli-0-2-6-7_vs_3-5	224	(7/0)	9.1818	Yeast6	1484	(8/0)	41.4
Glass-0-4_vs_5	92	(9/0)	9.2222	Abalone19	4174	(7/1)	129.44

Table 4
Average ranks with respect to AUC, to compare (a) different ensemble methods, and (b) among preprocessing methods for a particular ensemble. The best method (i.e., with the lowest rank) for (a) each row and (b) each column, is boldfaced. Methods that are equivalent to the best one are in brackets.

Preprocessing	Static	Dynamic selection				Preprocessing	Static	Dynamic selection			
method	ensemble	KNE	KNU	LCA	RANK	method	ensemble	KNE	KNU	LCA	RANK
Ba	[2.40]	3.19	**2.04**	[2.80]	4.58	Ba	**1.91**	**2.19**	**1.55**	**1.55**	**2.25**
Ba-RM100	**1.71**	3.22	[2.21]	3.38	4.48	Ba-RM100	2.85	3.40	2.41	2.95	3.13
Ba-RM	**1.66**	2.86	[2.06]	4.14	4.27	Ba-RM	3.99	3.80	3.31	3.98	3.54
Ba-SM100	**1.51**	2.92	3.12	3.06	4.39	Ba-SM100	3.04	3.14	4.21	3.16	3.34
Ba-SM	**1.46**	2.25	3.38	3.70	4.22	Ba-SM	4.23	3.77	5.21	3.99	3.82
Ba-RB	**1.84**	2.68	[2.09]	4.14	4.25	Ba-RB	4.98	4.70	4.31	5.38	4.92
		(a)						(b)			

Table 5
Average ranks with respect to F-measure, to compare (a) different ensemble methods, and (b) among preprocessing methods for a particular ensemble. The best method (i.e., with the lowest rank) for (a) each row and (b) each column, is boldfaced. Methods that are equivalent to the best one are in brackets.

Preprocessing	Static	Dynamic selection				Preprocessing	Static	Dynamic selection			
method	ensemble	KNE	KNU	LCA	RANK	method	ensemble	KNE	KNU	LCA	RANK
Ba	[2.79]	**2.31**	[2.32]	4.30	3.28	Ba	4.84	4.25	4.20	4.58	3.89
Ba-RM100	2.59	[2.34]	**1.96**	4.34	3.77	Ba-RM100	[3.40]	[3.24]	**2.44**	[2.99]	**2.83**
Ba-RM	2.52	[2.40]	**1.86**	4.34	3.88	Ba-RM	[3.12]	[3.15]	[2.47]	[2.82]	[3.26]
Ba-SM100	3.20	**2.06**	3.26	3.95	3.53	Ba-SM100	[3.49]	**2.78**	4.46	[2.97]	[3.09]
Ba-SM	[1.98]	**1.83**	3.47	3.98	3.75	Ba-SM	[3.10]	[3.09]	4.57	**2.66**	[3.24]
Ba-RB	**1.84**	2.55	[2.02]	4.70	3.88	Ba-RB	**3.05**	4.49	[2.86]	4.98	4.69
		(a)						(b)			

are highlighted in Table 2 and 3. During preprocessing of a dataset, we normalized the values for a numerical attribute to [0, 1].

We consider the performance of all the preprocessing methods (Table 1) and their combinations with static and dynamic ensemble methods. As stated in Section 4.2, we consider the average performances (over 5×2 experiments) with respect to each of AUC, F-measure and G-mean for purposes of comparison. Each method was assigned a rank for each dataset, based on its average performance, separately for each evaluation measure (i.e., AUC, F-measure and G-mean). The best method obtained rank 1. In the event of a tie, the ranks were shared. For example, if the top three methods tied for a dataset, they all received $(1+2+3)/3 = 2$ for that dataset. The ranks were averaged across all 84 datasets. The methods were arranged by their average rank, where the best method (i.e., the winner) had lowest average rank for an evaluation measure. This ranking procedure is similar to [33].

Given the ranks, we first performed Iman and Davenport's test [66], at a 95% significance level, to check whether there were any significant differences between the ranks of the methods compared. This test is a modification of the conventional Friedman test, which shows a conservative behavior [66,67]. The null hypothesis for the Iman and Davenport's test states that all the ranks are equivalent, i.e., all the algorithms perform equivalently. If the null hypothesis was rejected, we proceeded to perform a post hoc test. The Finner's [68] step-down procedure, for this purpose, is recommended by García et al. [67], because it is easy to understand and offers better results than other post hoc tests. We applied this procedure at a 95% significance level, to identify all methods that were equivalent to the best ranked method.

5.1. Comparison between static and dynamic ensemble methods

In this section, we intend to apply the preprocessing methods shown in Table 1, each with static and dynamic ensembles. For the sake of comparison, we present the average ranks calculated from the area under the curve (AUC) (Table 4), the F-measure (Table 5) and the G-mean (Table 6), respectively. The structure of these three tables is the same. The left table is row-wise, i.e., the ranks were calculated by rows. The possible ranks range from 1 to 5 (the static ensemble and four DS techniques). The right table is column-wise, meaning the ranks were computed by columns. Here, the ranks range from 1 to 6 (for the six preprocessing methods). In both cases, rank 1 corresponds to the best alternative.

The tables on the left indicate whether a dynamic selection performs better than the static ensemble. We observe that for AUC (Table 4), the static ensemble outperforms dynamic selection strategies in all but one case (bagging). However, the opposite can be observed for the F-measure and G-mean. Regarding the F-measure (Table 5), DS techniques clearly outperform the static ensemble, in all cases except for Ba-RB. The same is true for G-mean (Table 6). Moreover, for the G-mean, the KNU method becomes the best in three out of six cases. No such trend can be found with respect to F-measure. Finally, no method is statistically equivalent to Ba-SM100+KNE, for the F-measure and G-mean, which indicates KNE is the most suitable choice for Ba-SM100 preprocessing. It is also interesting to see that, for the F-measure and G-mean, a DES method usually has a lower average rank, as compared to a DCS method. We can therefore conclude that DES is preferable to DCS for combination with a preprocessing method.

The dissimilarities in the ranks with respect to AUC and the other two criteria may be due to the fact that the AUC measures a different aspect than does the F-measure or G-mean. For example, AUC considers probability of an instance given a class, whereas F-measure considers whether the instance is correctly classified. It is entirely possible that an instance having a higher probability for the minority class may not at all be correctly classified as a minority class. Thus, as Díez-Pastor et al. [33] pointed out, the preprocessing strategies have less impact on the AUC score. Since preprocessing is a key factor for the above comparisons, we may conclude that the AUC is less preferable for our comparison purpose.

The tables on the right (Tables 4(b), 5(b) and 6(b)) reveal the best preprocessing methods with respect to each ensemble method. For example, let us consider Tables 5(b) and 6(b) for the F-measure and G-mean. We see that for KNU and RANK, although the Ba-RM100 preprocessing method performs the best, Ba-RM however performs equivalently. For KNE, the Ba-SM100 preprocessing outperforms other preprocessing methods with respect to the F-measure and G-mean. It can be observed clearly that, for AUC, simple bagging is the best preprocessing method for all static and dynamic ensemble strategies.

Table 6
Average ranks with respect to G-mean, to compare (a) different ensemble methods, and (b) among preprocessing methods for a particular ensemble. The best method (i.e., with the lowest rank) for (a) each row and (b) each column, is boldfaced. Methods that are equivalent to the best one are in brackets.

Preprocessing	Static	Dynamic selection				Preprocessing	Static	Dynamic selection			
method	ensemble	KNE	KNU	LCA	RANK	method	ensemble	KNE	KNU	LCA	RANK
Ba	[2.68]	[2.45]	**2.20**	4.13	3.55	Ba	4.54	4.30	3.83	4.22	4.13
Ba-RM100	2.51	[2.45]	**1.98**	4.27	3.80	Ba-RM100	[3.35]	[3.38]	**2.52**	[3.07]	**2.94**
Ba-RM	2.48	[2.38]	**1.94**	4.31	3.88	Ba-RM	[3.24]	[3.15]	[2.63]	[3.04]	[3.13]
Ba-SM100	3.13	**2.03**	3.45	3.86	3.53	Ba-SM100	[3.43]	**2.88**	4.62	[3.06]	[3.20]
Ba-SM	[2.07]	**1.81**	3.53	3.98	3.61	Ba-SM	**3.16**	[3.10]	4.60	**2.90**	[3.23]
Ba-RB	**1.98**	[2.49]	[2.07]	4.67	3.78	Ba-RB	[3.27]	4.20	[2.81]	4.71	4.38
		(a)						(b)			

Table 7
Average ranks for the best ensemble methods. (a) According to AUC, (b) according to F-measure and (c) according to G-mean.

(a) AUC		(b) F-measure		(c) G-mean	
Methods	Rank	Method	Rank	Method	Rank
Ba+KNU	1.742	Ba-RM100+KNU	3.055	Ba-RM100+KNU	3.070
Ba-RM100	2.883	Ba-RM+KNU	3.117	Ba-RM+KNU	3.227
Ba-SM100	3.086	Ba-RB	3.242	Ba-RB	3.320
Ba-RM	4.023	Ba-SM100+KNE	3.445	Ba-SM100+KNE	3.477
Ba-SM	4.273	Ba-SM+KNE	3.641	Ba-SM+KNE	3.570
Ba-RB	4.992	Ba+KNE	4.500	Ba+KNU	4.336

Finally, we will now look at both the row-wise and column-wise tables. Here, an interesting fact is that Ba-RB can be considered as the best choice (w.r.t F-measure) for the static ensemble. While other preprocessing methods try to restore the balance of class proportions, RB randomly performs RUS and SMOTE, both in random amounts. In fact, after applying RB, the minority class may become the majority. Therefore, RB relies completely on random class proportions. For this reason, the base classifiers are expected to be diverse enough to improve the ensemble performance. Consequently, the contributions made by all the base classifiers may become significant. Thus, RB is a specialized method that is suitable for static ensemble. We should once more recall the importance of the DSEL size, which was indicated in Section 4.3. In the case of RB, the DSEL size may decrease due to RUS, and it affects the DS procedure considerably. This notwithstanding, we should note that Ba-RB+KNU performs similarly to the static ensemble for both the F-measure and G-mean.

Compiling all row-wise and column-wise results for the F-measure and G-mean, we may consider KNU as the most suitable DS strategy. On the other hand, we cannot find a single suitable candidate for preprocessing methods on the basis of these results. Nevertheless, we can conclude that preprocessing, combined with a DS strategy, may outperform a static ensemble. In the next section, we carry out a comparison of the best performing methods. Such studies reveal the overall performance of algorithms.

5.2. The overall winners

Let us now consider the best performing methods from each row of Tables 4(a), 5(a) and 6(a). New average ranks are calculated for these methods, considering all the datasets. We can find the positions of the winning methods according to their average ranks. The results are shown in Table 7. The Ba-RM100 preprocessing method, combined with KNU, dominates the other methods with respect to the F-measure and G-mean. On the other hand, for AUC, bagging with KNU (Ba-KNU) holds the best rank. This method, however, has the worst rank when the G-mean is considered. With respect to the F-measure, Ba-KNU is replaced by Ba-KNE, which has a better average rank. This indicates that AUC alone cannot guide the selection of a suitable algorithm for imbalanced data. Finally, we can see that the two RAMO-based methods hold the first two positions. Therefore, RAMO presented the best overall performance.

Table 8
Average ranks of the classifier ensemble methods with respect to medium and highly imbalanced datasets. (a) According to AUC, (b) according to F-measure and (c) according to G-mean.

(a) AUC		(b) F-measure		(c) G-mean	
Methods	Rank	Method	Rank	Method	Rank
Ba+KNU	1.566	Ba-RM100+KNU	2.906	Ba-RM100+KNU	2.943
Ba-RM100	2.604	Ba-RM+KNU	2.943	Ba-RM+KNU	3.151
Ba-SM100	3.038	Ba-RB	3.245	Ba-RB	3.415
Ba-RM	4.057	Ba-SM100+KNE	3.358	Ba-SM100+KNE	3.528
Ba-SM	4.396	Ba-SM+KNE	3.774	Ba-SM+KNE	3.717
Ba-RB	5.340	Ba+KNU	4.774	Ba+KNU	4.245

Table 9
Average ranks of the classifier ensemble methods with respect to highly imbalanced datasets. (a) According to AUC, (b) according to F-measure and (c) according to G-mean.

(a) AUC		(b) F-measure		(c) G-mean	
Methods	Rank	Method	Rank	Method	Rank
Ba+KNU	1.488	Ba-RM100+KNU	2.895	Ba-RM100+KNU	2.907
Ba-RM100	2.523	Ba-RM+KNU	3.012	Ba-RM+KNU	3.186
Ba-SM100	2.942	Ba-RB	3.360	Ba-RB	3.488
Ba-RM	4.151	Ba-SM100+KNE	3.488	Ba-SM100+KNE	3.558
Ba-SM	4.547	Ba-SM+KNE	3.756	Ba-SM+KNE	3.744
Ba-RB	5.349	Ba+KNE	4.488	Ba+KNU	4.116

5.3. Winners according to imbalance ratio

In this section, we consider the IR of a dataset, along with the classification performances. The aim of such a study is to observe the robustness of the algorithms for different values of IR. In this regards, our focus was on medium and highly imbalanced datasets. Thus, we designed our experiments as follows. First, we removed the datasets with low imbalance (i.e., with $IR < 3$) from consideration. Thereafter, we performed ranking and statistical tests similar to what is shown in Section 5.1. The ranking of the best performers are presented in Table 8. In a second experiment, we eliminated the datasets of medium imbalance (i.e., with $3 \geq IR \leq 9$), along with the low imbalanced datasets. After similar statistical tests, the ranks of the best performers are shown in Table 9.

Comparing Tables 7–9, it is clear that the topmost algorithms are the same regardless of the amount of imbalance present. This verifies that dynamic ensembles perform robustly even for highly imbalanced datasets. On the other hand, we see that the only static ensemble candidate is Ba-RB. This method holds the same position in the three tables. Ba-RB can be therefore conclusively be highly recommended for static ensembles. The RAMO-based dynamic ensemble methods outperform other methods for all three tables.

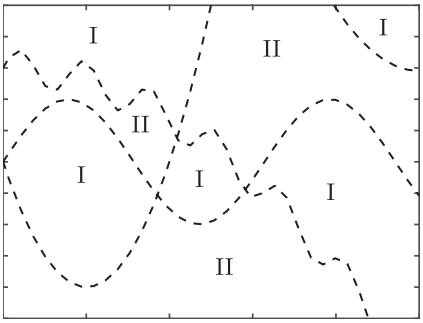


Fig. 2. The P2 problem. The symbols I and II indicate the majority and the minority classes.

Therefore, RAMO-based methods and the KNU dynamic selection methods are recommended for dynamic ensembles.

5.4. Performance analysis of Ba-RM100+KNU for DSEL with and without preprocessing

In Section 2, we pointed out the importance of DSEL in performing dynamic selection. In our experimental framework, we propose to apply a preprocessing on DSEL in order to re-balance it by generating synthetic instances. In all experiments in which a preprocessing was applied to the training data, the same preprocessing was applied for the DSEL as well. In this section, we intend to study the effects of preprocessing the DSEL on dynamic selection. In other words, we intend to compare the performance of a DS method for two versions of DSEL, one with preprocessing and the other without preprocessing. We perform this analysis on a two-class synthetic dataset and with our winning algorithm, Ba-RM100+KNU. Thus, we consider RAMO100 preprocessing for the DSEL and the KNORA-U dynamic selection method.

The P2 is a two-class problem; presented by Valentini [69], where each class is defined in multiple decision regions determined by polynomial and trigonometric functions. The P2 problem is illustrated in Fig. 2. While generating the instances for this problem, we changed the data distribution, so that the IR became 10. Thus, we caused the P2 problem to be highly imbalanced. Class II was the minority class. We separately generated the training, DSEL, and the test datasets. The majority to minority class size ratio was 200:20 for the training dataset and DSEL, and; 100:100 for testing dataset.

Table 10
Performance measures for two experiments with Ba-RM100+KNU on the P2 problem. The columns present average and standard deviations of AUC, F-measure and G-mean, over 5 experiments. A paired *t*-test was performed to assess whether the average values of a measure differ significantly in the two experiments. The test was performed at a 95% significance level.

Experiments	AUC	F-measure	G-mean
Experiment 1	0.779 (0.028)	0.467 (0.036)	0.546 (0.031)
Experiment 2	0.791 (0.022)	0.523 (0.027)	0.590 (0.022)
p-value for paired *t*-test	0.495	**0.023**	**0.032**

With the P2 problem, we first applied Ba-RM100 preprocessing on the training dataset in order to generate a pool of 100 base classifiers. Thereafter, we conducted two experimentation. In the first (called *Experiment 1*), we kept the DSEL as it was while in the second experiment (*Experiment 2*), we applied RAMO on DSEL to double its minority class size. For both experiments, we used KNORA-U (KNU) dynamic selection with 7 nearest neighbors. We had five sets of independent trials for each of Experiment 1 and Experiment 2. Each trial consisted of a training, a test and a DSEL generated independently. In Fig. 3, we present the testing results for a single trial of both experiments. There, we see that no instances from the majority class were misclassified. On the other hand, out of 100 minority class instances, 73 were misclassified in Experiment 1 and 62 in Experiment 2. Clearly, preprocessing the DSEL in Experiment 2 helped improve classification performance for the minority class.

Performance measures averages are presented in Table 10. We performed a paired *t*-test under the null hypothesis that the averages of a measure are equivalent for the two experiments. We can see in Table 10 that Experiment 2 improves all the three measures as compared to Experiment 1. The reported *p*-values clearly show that the improvement in the F-measure and G-mean are statistically significant. We also present the confusion matrix summed over all the five trials separately for the two experiments, in Table 11. We can observe that the classification accuracy for the

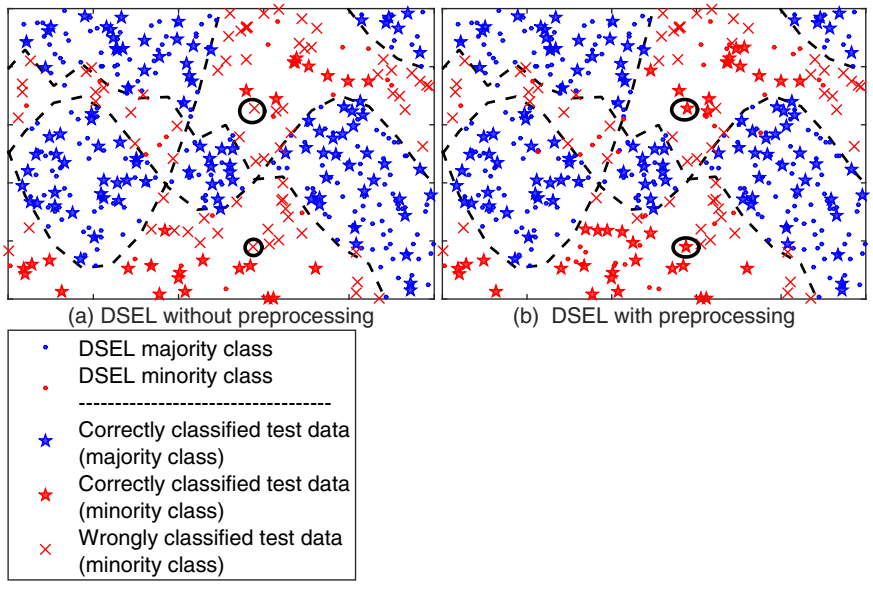


Fig. 3. Classification performance of Ba-RM100+KNU on the P2 problem. The DSEL (a) without and (b) with preprocessing, and the test instances are presented. No test data from the majority class was misclassified in either (a) or (b). The misclassification for minority class is more in (a) than (b). Two interesting instances from the minority class are circled. These instances were wrongly classified using DSEL without preprocessing whereas they were correctly classified using DSEL with preprocessing.

Table 11
Confusion matrix combining all the five trials for Experiment 1 and Experiment 2.

Classes	Experiment 1		Experiment 2	
	I	II	I	II
Majority (I)	495	5	495	5
Minority (II)	346	154	321	179

minority class improves in Experiment 2, whereas that for the majority class remain unchanged in both experiments.

Finally, we intended to carry out an in-depth study based on Fig. 3, in order to understand the differences in the region of competence for both experiments, given a test instance. Looking at Fig. 3, we can spot individual instances that are misclassified by Experiment 1 and correctly classified by Experiment 2. Two such instances were separated out along with their regions of competence. In Fig. 4, we present the region of competence of these two instances for both experiments.

Clearly, the difference between the regions of competence in Fig. 4 (a) and (b) is that, for the latter, more instances from the minority class are inside the region of competence. This is one of the main reasons for the improvement seen in performance. Again, since during preprocessing, we mostly synthesize new instances for the minority class, the above situation is very probable for real datasets. However, the performance of KNU also depends on the performances of base classifiers inside the region of competence. Nevertheless, we may conclude that preprocessing the DSEL helps in forming the region of competence better than using the DSEL as is.

6. Experiment on multi-class imbalanced problems

In this section, we perform a preliminary study on the application of dynamic selection and data preprocessing techniques for dealing with multi-class imbalance. Multi-class imbalanced classification is not as well developed as the binary case, with only a few papers handling this issue [8,70,71]. It is also considered as a more complicated problem, since the relation among the classes is no longer obvious. For instance, one class may be majority one when compared to some classes, and minority when compared to others [14]. In this case, data preprocessing techniques may have a bigger role due to the complex interaction between the different classes. Dynamic selection techniques is seen as an alternative to deal with this problem as it explores the local competence of each base classifier according to each new test sample [14,37].

A total of 26 multi-class imbalanced datasets taken from the Keel repository [64] was used in this analysis. These datasets were chosen based on previous works on multi-class imbalance [70,72,73]. The key features of the datasets are presented in Table 12. The IR is computed as the proportion of the number of

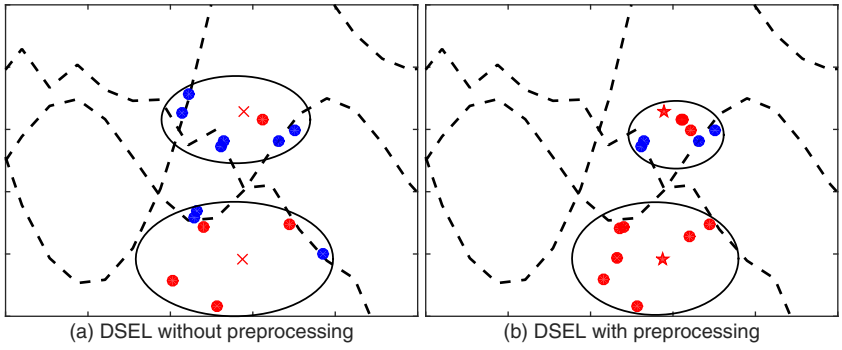


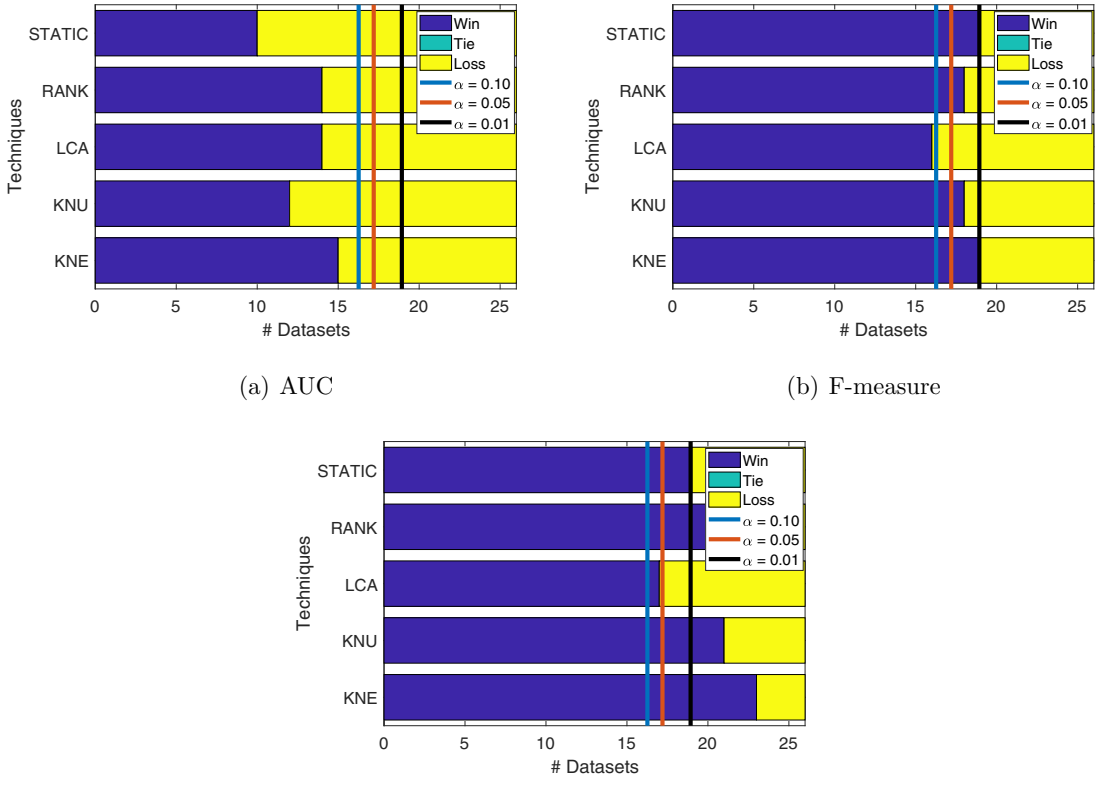
Fig. 4. The region of competence for the two interesting instances in Fig. 3. (a) Using DSEL without preprocessing and (i.e., Experiment 1) (b) using DSEL with preprocessing (i.e., Experiment 2). In both cases, the region of competence contains seven neighboring instances for the test instance.

Table 12
Characteristics of the 26 multi-class imbalanced datasets taken from the Keel repository. Column #E shows the number of instances in the dataset, column #A the number of attributes, #C shows the number of classes in the dataset, and column IR the imbalance ratio. The IR is computed as the proportion of the number of the majority class examples to the number of minority class examples. The low imbalanced datasets are highlighted with dark gray whereas the medium imbalanced datasets are in light gray.

Dataset	#E	#A	#C	IR	Dataset	#E	#A	#C	IR
Vehicle	846	(18/0)	4	1.09	CTG	2126	(21/0)	3	9.40
Wine	178	(13/0)	3	1.48	Zoo	101	(16/0)	7	10.25
Led7digit	500	(7/0)	10	1.54	Cleveland	467	(13/0)	5	12.62
Contraceptive	1473	(9/0)	3	1.89	Faults	1941	(27/0)	7	14.05
Hayes-Roth	160	(4/0)	3	2.10	Autos	159	(16/10)	6	16.00
Column3C	310	(6/0)	3	2.5	Thyroid	7200	(21/0)	3	40.16
Satimage	6435	(36/0)	7	2.45	Lymphography	148	(3/15)	4	40.50
Laryngeal3	353	(16/0)	3	4.19	Post-Operative	87	(1/7)	3	62.00
New-thyroid	215	(5/0)	3	5.00	Wine-quality red	1599	(11/0)	11	68.10
Dermatology	358	(33/0)	6	5.55	Ecoli	336	(7/0)	8	71.50
Balance	625	(4/0)	3	5.88	Page-blocks	5472	(10/0)	5	175.46
Flare	1066	(0/11)	6	7.70	Abalone	4139	(7/1)	18	45.93
Glass	214	(9/0)	6	8.44	Nursery	12690	(0/8)	5	2160.00

Table 13
Average ranks for the best ensemble methods considering the 26 multi-class imbalanced datasets. (a) According to AUC, (b) according to F-measure and (c) according to G-mean. Results that are statistically equivalent to the best one are in brackets.

(a) AUC		(b) F-measure		(c) G-mean	
Methods	Rank	Method	Rank	Method	Rank
Ba-RM100+KNU	**1.81**	Ba-RM100+KNU	**2.08**	Ba-RM100+KNE	**2.27**
Ba-RM100+KNE	[2.08]	Ba-RM100+KNE	[2.27]	Ba-RM100+KNU	[2.31]
Ba-RB	[2.81]	Ba-RB	[2.58]	Ba-RB	[2.38]
Ba-RM100+LCA	3.69	Ba-RM100+RANK	3.50	Ba-RM100+RANK	3.46
Ba-RM100+RANK	4.62	Ba-RM100+LCA	4.58	Ba-RM100+LCA	4.58



(a) AUC

(b) F-measure

(c) G-mean

Fig. 5. Pairwise comparison between the results achieved using the different DS techniques. The analysis is based in terms of wins, ties and losses. The vertical lines illustrate the critical values considering a confidence level $\alpha = \{0.10, 0.05, 0.01\}$.

the majority class examples to the number of minority class examples. In this case, the class with maximum number of examples is the majority class, and the class with the minimum number of examples is the minority one. Datasets with low IR (IR $<$ 3) are highlighted with dark gray, whereas datasets with medium IR (3 $<$ IR $<$ 9) are in light gray.

For the sake of simplicity, we only considered the best preprocessing algorithms from the previous analysis (Section 5.2). Hence, the RM100 was used for the DS techniques since it was the overall winner when applied with DS methods. In this case, the RAMO method is applied to generate synthetic samples for each minority class in the given dataset. Moreover, the RB was considered for the static combination since it was the best suited preprocessing method for static ensembles.

Performance evaluation is conducted using the multi-class generalization of the AUC, F-measure and G-mean [74]. The average rank of each technique according to AUC, F-measure and G-mean is presented in Tables 13(a), (b) and (c) respectively. Based on the average ranks, we can see that the DES techniques present a lower average rank when compared to that of the static combination for the three performance measures. The Ba-RM100+KNU technique presented the lowest average rank considering AUC and F-measure, while the Ba-RM100+KNE obtained the lowest average ranking according to G-mean.

Moreover, we performed the Iman and Davenport's test [66], at a 95% significance level, to check whether there were any significant differences between the ranks of the methods compared. The null hypothesis for the Iman and Davenport's test states that all the ranks are equivalent, i.e., all the algorithms perform equivalently. Since the null hypothesis was rejected, The Finner's [68] step-down procedure was conducted at a 95% significance level to identify all methods that were equivalent to the best ranked method. The performance of DES techniques (Ba-RM100+KNE and BA-RM100+KNU), and the best static approach (Ba-RB) were statistically equivalent considering the three performance evaluation criteria.

On the other hand, the DCS techniques (LCA and RANK) presented a higher average rank when compared to the static ensemble. Hence, similar to the experiments on binary problems, DES techniques outperforms DCS as well as static combination method for multi-class imbalanced problems. Both the Ba-RM100+KNU and

Ba-RM100+KNE are suitable DES strategies for dealing with multi-class imbalance. Also, DCS techniques are not suitable to deal with multi-class imbalance problems.

Moreover, Krawczyk [14] suggested that data preprocessing plays a bigger role when dealing with multi-class imbalance due to the relation among different classes. Hence, in order to demonstrate the importance of data preprocessing in this case, we conducted a pairwise comparison between the ensemble methods using data preprocessing (RM100 for DS and RB for static ensemble) with the same methods using only Bagging (Ba). The pairwise analysis is conducted using the Sign test, calculated on the number of wins, ties, and losses obtained by each method using preprocessing techniques, compared to the corresponding techniques without using preprocessing. The null hypothesis, H_0, meant that both techniques obtained statistically equivalent results. A rejection in H_0 meant that the classification performance obtained by the corresponding technique was significantly better at a predefined significance level α. In this case, the null hypothesis, H_0, is rejected when the number of wins is greater than or equal to a critical value, denoted by n_c. The critical value is computed using Eq. (2):

$$n_c = \frac{n_{exp}}{2} + z_\alpha \frac{\sqrt{n_{exp}}}{2} \tag{2}$$

where n_{exp} is the total number of experiments. We ran the test considering three levels of significance: $\alpha = \{0.10, 0.05, 0.01\}$. Fig. 5(a)–(c) shows the results of the Sign test according to AUC, F-measure and G-mean respectively. The different bars represent the critical values for each significance level.

The results of the pairwise analysis demonstrate that by using RM100 for DS techniques and RB for static combination as data preprocessing significantly improves the results of these techniques according to the F-measure and specially for the G-mean. Considering the F-measure, the KNU, KNE, RANK as well as the static combination obtained a significant number of wins at a confidence level $\alpha = 0.05$, while for G-mean these three techniques obtained a significant number of wins at a significance level of $\alpha = 0.01$. The only exception is the LCA method presented a significant number of wins only at a significance level of $\alpha = 0.05$. For the AUC metric, the results obtained with and without preprocessing are equivalent. Hence, the results obtained here confirm the hypothesis by Krawczyk [14], in which data preprocessing techniques play an important role when dealing with multi-class imbalanced problems.

7. Conclusions and future scope

This article presents an exhaustive study that combines dynamic selection (DS) methods with data preprocessing, for dealing with classification in an imbalanced environment. In the literature, data preprocessing has been widely adopted in imbalanced datasets. On the other hand, DS methods, have proven to be acceptable for many classification problems. In this paper, our aim was to couple these two methodologies and to observe the effects of this combination on the classification of imbalanced data. More precisely, we were interested into carrying out comparisons with static ensembles, because in previous studies, preprocessing was considered mostly with static ensembles. For the experiments, we considered three benchmark preprocessing algorithms, together with four DS strategies. Experiments conducted on 84 two-class and 26 multi-class datasets revealed that combining preprocessing with DS enhances the classification performance for the minority class, as compared to the combination of preprocessing and static ensembles. In particular, we can conclude that the DS method, named KNORA-Union (KNU), performs the best among the considered DS strategies. On the other hand, RAMO, in general becomes a preferred preprocessing strategy. Of course, the coupling of KNU and RAMO usually leads to the best classification performance, as seen in Table 7. The same conclusion holds for different levels of class-imbalance. Furthermore, an important by-product of this study is the observation that Random Balance (RB) is the best suited preprocessing method for static ensembles.

Our experimental framework (Fig. 1) enabled preprocessing the DSEL for improving the DS performance. In this article, we applied the same preprocessing for the training set and for the DSEL. An interesting future research avenue would consist in investigating the possibility of finding a preferable combination of preprocessing methods for both training set and the DSEL, which would lead to a better classification performance. Since DSEL is closely related to the definition of the region of competence, this study would include the characterization of the region of competence in terms of different versions of DSEL (obtained by preprocessing). Such a study may also be performed in order to obtain a preferred combination of preprocessing and the DS method, for a given problem. Moreover, a deeper study of the use of DS techniques, with different preprocessing methods, to cope with multi-class imbalanced classification problems is another interesting direction for future works.

References

[1] H. He, E. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[2] R. Akbani, S. Kwek, N. Japkowicz, Applying support vector machines to imbalanced datasets, in: Proceedings of the European Conference on Machine Learning (ECML), Pisa, Italy, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 39–50.
[3] W. Wei, J. Li, L. Cao, Y. Ou, J. Chen, Effective detection of sophisticated online banking fraud on extremely imbalanced data, World Wide Web 16 (4) (2013) 449–475.
[4] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, SIGKDD Explor. Newsl. 6 (1) (2004) 80–89.
[5] L. Piras, G. Giacinto, Synthetic pattern generation for imbalanced learning in image retrieval, Pattern Recognit. Lett. 33 (16) (2012) 2198–2205.
[6] D. Thammasiri, D. Delen, P. Meesad, N. Kasap, A critical assessment of imbalanced class distribution problem: the case of predicting freshmen student attrition, Expert Syst. Appl. 41 (2) (2014) 321–330.
[7] K. Wu, A. Edwards, W. Fan, J. Gao, K. Zhang, Classifying imbalanced data streams via dynamic feature group weighting with importance sampling, in: Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, Pennsylvania, USA, 2014, pp. 722–730.
[8] S. Wang, L.L. Minku, X. Yao, Dealing with multiple classes in online class imbalance learning, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, (IJCAI), New York, NY, USA, 2016, pp. 2118–2124.
[9] E.S. Xioufis, M. Spiliopoulou, G. Tsoumakas, I.P. Vlahavas, Dealing with concept drift and class imbalance in multi-label stream classification, in: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, (IJCAI), Barcelona, Catalonia, Spain, 2011, pp. 1583–1588.
[10] S. Chen, H. He, SERA: selectively recursive approach towards nonstationary imbalanced stream data mining, in: Proceedings of the International Joint Conference on Neural Networks, (IJCNN), Atlanta, Georgia, USA, 2009, pp. 522–529.
[11] H. He, Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications, first edn., Wiley-IEEE Press, 2013.
[12] P. Branco, L. Torgo, R.P. Ribeiro, A survey of predictive modeling on imbalanced domains, ACM Comput. Surv. 49 (2) (2016) 31:1–31:50.
[13] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches, Trans. Sys. Man Cyber Part C 42 (4) (2012) 463–484.
[14] B. Krawczyk, Learning from imbalanced data: open challenges and future directions, Prog. Artif. Intell. 5 (4) (2016) 221–232.
[15] R.C. Prati, G.E.A.P.A. Batista, D.F. Silva, Class imbalance revisited: a new experimental setup to assess the performance of treatment methods, Knowl. Inf. Syst. 45 (1) (2015) 247–270.
[16] V. García, R.A. Mollineda, J.S. Sánchez, On the k-nn performance in a challenging scenario of imbalance and overlapping, Pattern Anal. Appl. 11 (3–4) (2008) 269–280.
[17] J. Hu, Y. Li, W.-X. Yan, J.-Y. Yang, H.-B. Shen, D.-J. Yu, Knn-based dynamic query-driven sample rescaling strategy for class imbalance learning, Neurocomputing 191 (2016) 363–373. https://doi.org/10.1016/j.neucom.2016.01.043.
[18] R. Batuwita, V. Palade, FSVM-CIL: fuzzy support vector machines for class imbalance learning, IEEE Trans. Fuzzy Syst. 18 (3) (2010) 558–571, doi:10.1109/TFUZZ.2010.2042721.
[19] D.A. Cieslak, T.R. Hoens, N.V. Chawla, W.P. Kegelmeyer, Hellinger distance decision trees are robust and skew-insensitive, Data Mining Knowl. Discov. 24 (1) (2012) 136–158.

[20] V. García, J.S. Sánchez, R.A. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Know. Based Syst. 25 (1) (2012) 13–21.
[21] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: Synthetic minority over-sampling technique, J. Artif. Int. Res. 16 (1) (2002) 321–357.
[22] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, SIGKDD Explor. Newsl. 6 (1) (2004) 20–29.
[23] C.L. Castro, A.P. Braga, Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data, IEEE Trans. Neural Netw. Learn. Syst. 24 (6) (2013) 888–899.
[24] R. Alejo, V. García, J.M. Sotoca, R.A. Mollineda, J.S. Sánchez, Improving the performance of the rbf neural networks trained with imbalanced samples, in: Proceedings of the International Work-Conference on Artificial Neural Networks, (IWANN), Springer Berlin, Heidelberg, San Sebastián, Spain, 2007, pp. 162–169, doi:10.1007/978-3-540-73007-1_20.
[25] Y. Sun, M.S. Kamel, A.K. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recognit. 40 (12) (2007) 3358–3378.
[26] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, second edn., Wiley Publishing, 2014.
[27] N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, SMOTEBoost: improving prediction of the minority class in boosting, in: Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases, Springer, Berlin, Heidelberg, 2003, pp. 107–119.
[28] Y. Freund, R.E. Schapire, Experiments with a new boosting algorithm, in: Proceedings of the Thirteenth International Conference on Machine Learning (ICML), Morgan Kaufmann, 1996, pp. 148–156.
[29] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, RUSBoost: a hybrid approach to alleviating class imbalance, Trans. Sys. Man Cyber. Part A 40 (1) (2010) 185–197.
[30] S. Chen, H. He, E.A. Garcia, RAMOBoost: ranked minority oversampling in boosting, Trans. Neur. Netw. 21 (10) (2010) 1624–1642.
[31] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[32] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, 2009, pp. 324–331.
[33] J.F. Díez-Pastor, J.J. Rodríguez, C.I. García-Osorio, L.I. Kuncheva, Diversity techniques improve the performance of the best imbalance learning ensembles, Inf. Sci. 325 (2015) 98–117.
[34] J. Błaszczyński, J. Stefanowski, Neighbourhood sampling in bagging for imbalanced data, Neurocomputing 150, Part B (2015) 529–542.
[35] A. Roy, R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, Meta-learning recommendation of default size of classifier pool for meta-des, Neurocomputing 216 (2016) 351–362.
[36] A.S. Britto, R. Sabourin, L.E.S. Oliveira, Dynamic selection of classifiers - a comprehensive review, Pattern Recognit. 47 (11) (2014) 3665–3680.
[37] R.M. Cruz, R. Sabourin, G.D. Cavalcanti, Dynamic classifier selection: Recent advances and perspectives, Inf. Fus. 41 (2018) 195–216.
[38] K. Woods, W.P. Kegelmeyer, K. Bowyer, Combination of multiple classifiers using local accuracy estimates, IEEE Trans. Pattern Anal. Mach. Intell. 19 (4) (1997) 405–410.
[39] A. Ko, R. Sabourin, J.A. Britto, From dynamic classifier selection to dynamic ensemble selection, Pattern Recognit. 41 (5) (2008) 1718–1731.
[40] P.R. Cavalin, R. Sabourin, C.Y. Suen, Dynamic selection approaches for multiple classifier systems, Neural Comput. Appl. 22 (3–4) (2013) 673–688.
[41] R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, T.I. Ren, META-DES: a dynamic ensemble selection framework using meta-learning, Pattern Recognit. 48 (5) (2015) 1925–1935.
[42] J. Xiao, L. Xie, C. He, X. Jiang, Dynamic classifier ensemble model for customer classification with imbalanced class distribution, Expert Syst. Appl. 39 (3) (2012) 3668–3675.
[43] M. Sabourin, A. Mitiche, D. Thomas, G. Nagy, Classifier combination for hand-printed digit recognition, in: Proceedings of the 2nd International Conference on Document Analysis and Recognition, 1993, pp. 163–166.
[44] P.R. Cavalin, R. Sabourin, C.Y. Suen, Logid: An adaptive framework combining local and global incremental learning for dynamic selection of ensembles of HMMs, Pattern Recognit. 45 (9) (2012) 3544–3556.
[45] R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, META-DES.H: A dynamic ensemble selection technique using meta-learning and a dynamic weighting approach, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 2015, pp. 1–8.
[46] A. Roy, R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, Meta-regression based pool size prediction scheme for dynamic selection of classifiers, in: Proceedings of the 23rd International Conference on Pattern Recognition (ICPR), IEEE, 2016, pp. 211–216.
[47] L.I. Kuncheva, A theoretical study on six classifier fusion strategies, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2) (2002) 281–286.
[48] R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, META-DES.Oracle: Meta-learning and feature selection for dynamic ensemble selection, Inf. Fusion 38 (2017) 84–103.
[49] R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, A DEEP analysis of the META-DES framework for dynamic selection of ensemble of classifiers, CoRR, 2015 abs/1509.00825.
[50] R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, Prototype selection for dynamic classifier and ensemble selection, Neural Comput. Appl. 29 (2) (2018) 447–457.
[51] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Proceedings of the Third IEEE International Conference on Data Mining (ICDM), IEEE Computer Society, Washington, DC, USA, 2003, pp. 435–442.
[52] R. Barandela, R. Valdovinos, J. Sánchez, New applications of ensembles of classifiers, Pattern Anal. Appl. 6 (3) (2003) 245–256.
[53] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-smote: A new over-sampling method in imbalanced data sets learning, in: Proceedings of the International Conference on Advances in Intelligent Computing - Volume Part I (ICIC), Springer-Verlag, Berlin, Heidelberg, 2005, pp. 878–887.
[54] C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem, in: Proceedings of the Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD), Springer-Verlag, Berlin, Heidelberg, 2009, pp. 475–482.
[55] H. He, Y. Bai, E. Garcia, S. Li, Adasyn: adaptive synthetic sampling approach for imbalanced learning, in: Proceedings of the International Joint Conference on Neural Networks, 2008, pp. 1322–1328, doi:10.1109/IJCNN.2008.4633969.
[56] J.F. Díez-Pastor, J.J. Rodríguez, C.I. García-Osorio, L.I. Kuncheva, Random balance: ensembles of variable priors classifiers for imbalanced data, Knowl. Based Syst. 85 (2015) 96–111.
[57] I.H. Witten, E. Frank, M.A. Hall, C.J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, fourth, Morgan Kaufmann, Burlington, MA, 2016.
[58] J.J. Rodriguez, L.I. Kuncheva, C.J. Alonso, Rotation forest: A new classifier ensemble method, IEEE Trans. Pattern Anal. Mach. Intell. 28 (10) (2006) 1619–1630.
[59] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
[60] N.V. Chawla, C4.5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure, in: Proceedings of the ICML Workshop on Class Imbalances, 2003.
[61] T. Fawcett, An introduction to ROC analysis, Pattern Recognit. Lett. 27 (8) (2006) 861–874.
[62] C.J.V. Rijsbergen, Information Retrieval, second, Butterworth-Heinemann, Newton, MA, USA, 1979.
[63] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection, in: Proceedings of the 14th International Conference on Machine Learning (ICML), 1997, pp. 179–186.
[64] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, F. Sánchez, L. Herrera, Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, J. Multiple Valued Logic Soft Comput. 17 (2–3) (2011) 255–287.
[65] 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 Syst. 159 (18) (2008) 2378–2398.
[66] R.L. Iman, J.M. Davenport, Approximations of the critical region of the fbietkan statistic, Commun. Stat. Theory Methods 9 (6) (1980) 571–595.
[67] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Inf. Sci. 180 (10) (2010) 2044–2064.
[68] H. Finner, On a monotonicity problem in step-down multiple test procedures, J. Am. Stat. Assoc. 88 (423) (1993) 920–923.
[69] G. Valentini, An experimental bias-variance analysis of svm ensembles based on resampling techniques, IEEE Trans. Sys. Man Cybern. Part B Cybern. 35 (6) (2005) 1252–1271.
[70] L. Abdi, S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling techniques, IEEE Trans. Knowl. Data Eng. 28 (1) (2016) 238–251.
[71] L. Abdi, S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling and boosting techniques, Soft Comput. 19 (12) (2015) 3369–3385.
[72] A. Fernández, V. López, M. Galar, M.J. Del Jesus, F. Herrera, Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches, Knowl. Based Syst. 42 (2013) 97–110.
[73] F. Fernández-Navarro, C. Hervás-Martínez, P.A. Gutiérrez, A dynamic over-sampling procedure based on sensitivity for multi-class problems, Pattern Recognit. 44 (8) (2011) 1821–1833.
[74] Y. Sun, M.S. Kamel, Y. Wang, Boosting for learning multiple classes with imbalanced class distribution, in: Proceedings of the International Conference on Data Mining (ICDM), 2006, pp. 592–602.

Anandarup Roy completed his Ph.D. in Computer Science from Visva-Bharati University, Santiniketan, India in 2014. Currently he is a post-doctoral researcher at LIVIA (Laboratoire d imagerie, de vision et d'intelligence artificielle), cole de Technologie Supérieure (ÉTS), Université du Québec. His main research interests include statistical pattern recognition, mixture models, handwriting recognition and ensemble of classifiers.

Rafael M. O. Cruz obtained a Ph.D. in Engineering from the École de Technologie Supérieure (ÉTS), Université du Québec in 2016. Currently he is a post-doctoral researcher at LIVIA (Laboratoire d imagerie, de vision et d'intelligence artificielle). His main research interests are ensemble of classifiers, dynamic ensemble selection, meta-learning, prototype selection and handwritten recognition.

R. Sabourin joined in 1977 the physics department of the Montreal University where he was responsible for the design, experimentation and development of scientific instrumentation for the Mont Mégantic Astronomical Observatory. His main contribution was the design and the implementation of a microprocessor-based fine tracking system combined with a low-light level CCD detector. In 1983, he joined the staff of the cole de Technologie Supérieure, Université du Québec, in Montréal where he co-founded the Dept. of Automated Manufacturing Engineering where he is currently Full Professor and teaches Pattern Recognition, Evolutionary Algorithms, Neural Networks and Fuzzy Systems. In 1992, he joined also the Computer Science Department of the Pontifɖcia Universidade Catlica do Paran (Curitiba, Brazil) where he was, co-responsible for the implementation in 1995 of a master program and in 1998 a Ph.D. program in applied computer science. Since 1996, he is a senior member of the Centre for Pattern Recognition and Machine Intelligence (CENPARMI, Concordia University). Since 2012, he is the Research Chair holder specializingin Adaptive Surveillance Systems in Dynamic Environments. Dr Sabourin is the author (and co-author) of more than 350 scientific publications including journals and conference proceeding. He was co-chair of the program committee of CIFED'98 (Conférence Internationale Francophone sur l'Écrit et le Document, Québec, Canada) and IWFHR04 (9th International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan). He was nominated as Conference co-chair of ICDAR'07 (9th International Conference on Document Analysis and Recognition) that has been held in Curitiba, Brazil in 2007. His research interests are in the areas of adaptive biometric systems, adaptive surveillance systems in dynamic environments, intelligent watermarking systems, evolutionary computation and biocryptography.

George D. C. Cavalcanti received the D.Sc. degree in Computer Science from Center for Informatics, Federal University of Pernambuco, Brazil. He is currently an Associate Professor with the Center for Informatics, Federal University of Pernambuco, Brazil. His research interests include machine learning, pattern recognition, computer vision, and biometrics.