Ensemble classification for imbalanced data based on feature space partitioning and hybrid metaheuristics

Article *in* Applied Intelligence · February 2019

DOI: 10.1007/s10489-019-01423-6

CITATIONS
4

READS
116

5 authors, including:

Pedro López García
Tecnalia Corporación Tecnológica
28 PUBLICATIONS **268** CITATIONS
SEE PROFILE

Antonio David Masegosa
Universtiy of Deusto/ Ikerbasque - Basque Foundation for Science
60 PUBLICATIONS **358** CITATIONS
SEE PROFILE

Eneko Osaba
Tecnalia Corporación Tecnológica
109 PUBLICATIONS **719** CITATIONS
SEE PROFILE

Enrique Onieva
University of Deusto
139 PUBLICATIONS **1,732** CITATIONS
SEE PROFILE

Noname manuscript No.
(will be inserted by the editor)

Ensemble Classification for Imbalance Data based on Feature Space Partitioning and Hybrid Metaheuristics

Pedro Lopez-Garcia · Antonio D. Masegosa · Eneko Osaba · Enrique Onieva · Asier Perallos

the date of receipt and acceptance should be inserted later

Abstract One of the most challenging issues when facing a classification problem is to deal with imbalanced datasets. In recent literature, ensemble classification techniques have proven to be very successful in addressing this problem. In this paper, we present an ensemble classification approach based on feature space partitioning for imbalanced classification. In order to optimize the different parameters related to the feature space partitioning, a hybrid metaheuristic called GACE is applied. To assess the performance of the proposal, an extensive experimentation over imbalanced and real-world datasets is accomplished by comparing different configurations and base classifiers. The results show that its performance is competitive with reference techniques in the literature.

1 Introduction

The classification task is one of the most important and basic tasks in the field of machine learning. Many approaches to this task have been developed over the years. Some of the classical methods that can be included in this topic are Decision Trees, Artificial Neural Networks, K-nearest Neighbor, or Support Vector Machines among others. These techniques operate under the assumption that the data contains a faithful balance between each of the classes represented in the problem they are applied to [14]. However, in many real-world problems, this assumption leads to poor performance when the number of instances of one class is much lower than the rest of the classes. If this situation occurs, the data-set is said to be imbalanced. In this kind of data, the class with the largest number of instances is called the majority class, while a class with fewer instances is called

Pedro Lopez-Garcia · Antonio D. Masegosa · Eneko Osaba · Enrique Onieva · Asier Perallos
DeustoTech-Fundacion Deusto, Deusto Foundation, 48007, Bilbao, Spain.

Pedro Lopez-Garcia · Antonio D. Masegosa · Eneko Osaba · Enrique Onieva · Asier Perallos
Faculty of Engineering, University of Deusto, 48007, Bilbao, Spain.

Antonio D. Masegosa
IKERBASQUE, Basque Foundation for Science, 48011, Bilbao, Spain.
E-mail: {p.lopez, ad.masegosa, e.osaba, enrique.onieva, perallos}@deusto.es

a minority class. Imbalanced data is present in real-world problems, such as disease diagnosis [57], traffic congestion [45], astronomy [58] or image classification [72, 74]. When machine learning methods are applied to imbalanced data, they should focus on achieving a good classification of the minority class due to the fact that the cost of misclassifying them is usually higher[22]. Using the traffic congestion forecasting problem as an example, it is more important to achieve a higher accuracy regarding instances of congestion (a minority class) than for instances of a normal state of traffic (the majority class) due to the loss of time this can involve for drivers in a real-world scenario.

Many approaches have been proposed in the literature to deal with learning from imbalanced data; among them are sampling methods, cost-sensitive algorithms, one-class classifiers, and ensemble classification techniques. These approaches can be placed into three different categories:

- Data-level approaches, which are focused on restructuring the training datasets in order to balance them. Oversampling and undersampling methods are the most common examples of this category;
- Algorithmic level approaches, which introduce modifications in the classification methods to improve their performance when classifying the minority class; and
- Ensemble methods, which combine the estimation of a set of individual classifiers trained over the same data. The most widely used approach of this class is the so-called boosting algorithm, which works under the premise that a set of weak classifiers works better than a strong one. Examples of boosting algorithms are SMOTEBoost [9], RUSBoost [56], or AdaBoost [62].

In this work, we focus on ensemble methods because they have shown to be one of the most successful approaches to deal with imbalance classification so far. Ensemble classification can be defined as the combination of a group of classifiers whose individual decisions are joined in some manner to provide a final output [34]. The principal idea behind the use of ensemble classification is to learn from data using multiple individual classifiers.

Generally, ensemble classification has proved to obtain better results than an individual classifier on its own, when they are applied to the same problems [12], and it has also been presented as a method to improve the performance of a single classifier [5, 12, 33]. When an ensemble is created, there are some design decisions to take into account, such as the algorithm or algorithms to use as individual classifier (also called base classifier), the sampling strategy, and the collective decision making method for the final output, i.e. the method for combining the outputs of the base classifiers. Other aspects to take into account could be the diversity generation or the way each classifier will be trained (with the whole training set or a part of it). This area of research has attracted significant interest in recent years. The interested reader is referred to [53] for a tutorial on this topic, describing a taxonomy for characterizing ensemble methods, and the general process of constructing classification ensembles.

Focusing on the selection method used to choose the most appropriate individual classifiers given a classification problem, we can find two approaches: static classifier selection, where the same ensemble is applied for all test samples; and dynamic classifier selection, where a different ensemble may be applied for each test sample [20]. Within dynamic classifier selection, an interesting concept is the

local specialization of the base classifiers on specific partitions of the feature space [39]. Some proposals in this direction assume the local specialization of the individual classifiers while others divide the feature space into partitions and establish a different classifier for each of them.

Regarding the choice of a collective decision-making method, two main groups of methods can be defined for this task. The first one includes algorithms that join the answers of their classifiers. Majority voting [55] and other kinds of popular voting variants [39, 41, 65] are part of this group. Advanced techniques include weighting the importance of the decisions coming from the base classifiers. Treating the process of weight selection as a separate learning process is an alternative method [23, 29, 40]. One of the advantages of these techniques is the effective counteraction to avoid the overtraining of the base classifiers [29]. The second group is formed by the procedures that use a posteriori probability estimator to classifier fusions on the level of their discriminating function. These methods do not require a learning procedure. However, they can be only used in clearly defined conditions [16].

The present paper relates to dynamic classifier selection based on local specialization and weighted voting as collective decision-making method. Concretely, we focus on the approach called AdaSS (Adaptive Splitting and Selection) that simultaneously divides the feature space into partitions and establishes a different classifier for each partition. This approach was proposed by [29] and recently used in other works as [30, 31] with very promising results. This approach for building ensembles entails the resolution of a complex optimization problem whose objective is the minimization of the error of the whole system.

In this work, we propose the use of ensemble methods based on AdaSS as a powerful tool to deal with imbalance datasets because, as far as we know, it has not been applied before in this context. The motivations behind this research are:

- To select the best area possible to create the partitions for each ensemble by optimizing the centroid's positions of the clusters that delimit the partitions of the feature space.
- To optimize the weights of each base classifier within the discriminant function of the collective decision-making method of each ensemble. This optimization will be unsupervised. In this way, it won't be necessary the knowledge of an expert or an external validation set to determinate the initial value of the weights. This also makes that the classifiers work in a no restrictive way, i.e. the final weights will be based in the level of expertise that the classifier had obtained along the execution in every class.
- To address the last two tasks (feature space centroids and individual weights optimization for each partition and ensemble, respectively) into one integrated process. This approach has proven to obtain very good results in other works as [29, 30, 31]. The main novelty of our proposal w.r.t. the mentioned works, it is the incorporation of a more powerful method to solve the underlying optimization problem. This optimization problem becomes even more complex in the context of imbalanced data because the prediction of the majority classes leads to local minima with big basins of attractions, which is a characteristic that is known to lead optimization methods to poorer performance. For this reason, in our opinion, the use of more advanced optimization algorithms is a must in order to ensure the best possible performance of the resulting ensemble. To

this end, we have used a hybrid metaheuristic, called GACE, that combines a Genetic Algorithm (GA) with a Cross Entropy (CE) method for the resolution of the mentioned optimization problem. The main advantage of this technique is the combination of the exploration ability of the GA and the exploitation ability of the CE. This method was successfully applied to the optimization of hierarchical fuzzy rule-based systems [45], and continuous functions [44].
– To be able to deal with imbalance data without the use of data-level methods as SMOTE, which is one of the most successful ones currently. As mentioned before, data-level approaches to deal with imbalance implies the modification of the training set, which leads to an extra cost in terms of the time required for the application of the technique. Our aim is to design methods that provide similar or better results without this extra-cost by avoiding modifications of the training set.

This paper is an extension of the work presented in [46]. The main novelties addressed here are listed below:

– A wider and more realistic benchmark: the total number of datasets has increased from 12 to 40 by incorporating new imbalanced datasets. Furthermore, ten out of forty correspond to real-world datasets for traffic congestion prediction.
– A new analysis of the algorithm's behaviour: a study of the influence of the subpopulation size in the performance of ensemble method has been included.
– An extension of the comparative study: the proposal has been compared with new high-performance and well-known methods from the literature on imbalanced classification.

The rest of this paper is structured as follows. Section 2 presents different articles related to ensemble methods and hybrid techniques with a special focus on its application to imbalanced data. The ensemble methodology based on AdaSS and GACE is exposited in Section 3. The experimental set-up is presented in Section 4. Finally, Section 5 contains the conclusions and avenues for future work.

2 Background

In this section, different approaches in the literature related to our proposal are presented. We focus this section on works in the state of the art from the three different areas that form part of the problem and proposed solution exposited in this paper: ensemble approaches applied not only to general themes but also to imbalanced data in Section 2.1, metaheuristics applied to imbalanced data in Section 2.2, and hybrid methods applied to both imbalanced and balanced domains in Section 2.3.

2.1 Ensemble learning applied to imbalanced classification

Ensemble learning can be defined as the use of multiple learning algorithms to obtain better predictive performance than could be obtained from any of these algorithms alone [50, 53]. Over the last decade, much research related to this approach has been presented in the literature, focusing on the classification problem

for imbalanced datasets. For example, in [52], a resampling ensemble algorithm is developed focused on the imbalanced classification problem. In this case, the minority classes are oversampled while the majority classes are undersampled. To construct the ensemble, machine learning methods are selected.

Another example can be found in [63], which presents a bagging technique where two learning algorithms are used to construct the ensemble to deal with an online class imbalanced learning problem.

In [42], a resampling ensemble algorithm is developed focused on the classification problems for imbalanced datasets. The optimization technique used in this case is the BAT algorithm, where the accuracy rate of all the classes is optimized at the same time. From the experimental results, the system can be used to reduce the time complexity as well as enhance the accuracy rate of the imbalanced classification process.

Another ensemble-based method is presented in [18], where Synthetic Minority Over-sampling Technique (SMOTE) and Rotation Forest algorithm are used to address the class imbalance problem. Twenty KEEL imbalanced datasets are used in the experimentation, where the proposal is compared with different classification ensemble methods, such as SMOTE-Boost, SMOTE-Bagging, and SMOTE-random subspace.

There are many papers related to this theme, which means that it is an active issue in the literature. For this reason, the state-of-the-art about ensemble imbalance classification is wide. Interested readers are referred to [34], [49], and [62] for different surveys of this issue.

2.2 Metaheuristics applied to imbalanced classification

Metaheuristic techniques have been used in many different fields over the last decades. This category includes algorithms such as Particle Swarmn Optimization (PSO) [35], Ant Colony Optimization (ACO) [15], Genetic Algorithm (GA) [26], Bat Algorithm (BA) [68], Data Gravitation Classification (DGC) [73], and so on [75]. Focusing on classification, and especially on imbalanced problems, these methods have been widely used on their own as well as in combination with other techniques in the literature. In [66], PSO is proposed for omics data classification. The algorithm is designed to handle different characteristics of omics data, such as high dimensionality, small sample size, and class imbalance.

For example, in this article [69] Authors proposed an undersampling method based on ACO for an imbalanced problem in DNA microarray data. The proposal is evaluated on four benchmark skewed DNA microarray datasets. The results outperform many other sampling approaches.

In case of [6], authors developed a cost-sensitive feature selection method using a type of GA called a chaos genetic algorithm. The evaluation function considers both feature-acquiring costs and misclassification costs in the field of network security, weakening the influence of the many instances from the majority classes in large-scale datasets. The proposal is tested on a large-scale dataset of network security, using two kinds of classifiers: C4.5 and K-nearest Neighbor.

Other works related with metaheuristics such as Support Vector Machine (SVM) or Genetic Programming (GP) can be found in [10], where a SVM based

framework is presented to optimize different metrics related to class imbalance situations, as well as in [48], where the authors analyzed the performance of evolving diverse ensembles using genetic programming for software defect prediction with imbalanced data.

2.3 Hybrid algorithms applied to imbalanced classification

In this subsection, different hybrid approaches in the literature are mentioned. Hybrid algorithms are a way of dealing with the weaknesses of the different methods, and, at the same time, maximizing their strengths. These algorithms have been used in many and varied domains, such as medicine [25], scheduling optimization [43], transportation systems [45], astronomy [57], and so on.

In the imbalanced classification field, many works can be found that use hybrid algorithms to deal with this problem. For example, in [67], a PSO is proposed for dealing with the class imbalance problem in medical and biological data mining. A PSO is combined with multiple classifiers and a performance metric for evaluation fusion. The majority classes are ranked using multiple objectives according to their merit and then combined with the minority class to create a balanced dataset.

Authors in [60] presented a Soft-Hybrid algorithm to improve classification performance. The hybrid algorithm is formed by different modified machine learning techniques whose results were combined at the end of an experimentation phase. Measures such as the true positive rate, the F-measure, and the G-mean were used as quality measures.

Another example can be found in [8], where a hybrid algorithm formed by a GA and an undersampling method is created to improve the accuracy of support vector machines on skewed datasets.

Lastly, in [3], the authors developed a hybrid Adaboost-SVM method using Gaussian Mixture Modeling (GMM) to investigate the impact of using GMM with the boosted SVM in a multi-class phoneme recognition problem with the aim of advancing the classification of imbalanced data.

The main novelty of the present paper with respect to the methods reviewed in this and the previous subsections is two-fold. On the one hand, the fact of using feature space partitioning as the approach to build the ensemble for imbalanced datasets, that has not been used in this context before. And on the other hand, the application of GACE as optimization method of the ensemble approach, a more powerful optimizer than those used in previous works on ensembles based on feature space partitioning. As mentioned before, the main motivation behind this is that the problem becomes harder when the datasets are imbalanced due to the big basins of attraction created by the majority classes.

The concept of feature space partitioning has been applied to imbalance classification in [36] and [37]. In the first work, the feature space partitioning consists on clustering strategies as c-means or fuzzy c-means, and the weights of the base classifiers are based on a heuristic function that takes into account the Euclidean distance between the object and the boundary of the respective class. In the second work, the feature space partitioning is based on random subspaces and the weights of the base classifiers are set in the same way as before. The main difference with these two works is the use of AdaSS as feature space partitioning technique whose main advantage is the simultaneous optimization of the partitions, the assignment

of classifiers to partitions and the weights of the base classifiers for inferring the output class.

3 Description of the ensemble approach

In this section, we describe the different elements that compose the proposed approach. First, we describe the AdaSS algorithm for the simultaneous partitioning of the feature space and assignment of classifiers to partitions (Section 3.1) and then, the details of the training algorithm based on the GACE hybrid metaheuristic, in Section 3.2.

3.1 Description of Adaptive Splitting And Selection Algorithm

The Adaptive Splitting and Selection Algorithm exploits the local competencies of given classifiers. Let us assume the feature space $\mathcal{X}$ is divided into a set of H clusters,

$$\mathcal{X} = \bigcup_{h=1}^{H} \hat{\mathcal{X}}_h, \;\; \forall k, l \in \{1, ..., H\}, \quad k \neq l, \;\; \hat{\mathcal{X}}_k \cap \hat{\mathcal{X}}_l = \varnothing \tag{1}$$

where $\hat{\mathcal{X}}_h$ denotes the h-th constituent (cluster). The clusters are defined by their centroids $C_h = \{c_h^1, \ldots, c_h^d\}$, where d is the feature space dimension. With this information we define:

$$\text{member}(C, x) = \arg\,\min_{h=1}^{H} dist(x, C_h) \tag{2}$$

as the functions that returns the index of the cluster where $C = \{C_1, \ldots, C_h\}$ is the set of centroids and *dist* refers to the Euclidean distance.In case of draw, the cluster with the lower index is selected. Then, the decision rule for the combined classifier Ψ is given by the formula:

$$\Psi(x) = \bar{\Psi}_{member(C, x_n)}(x_n) \tag{3}$$

where $\bar{\Psi}_h$ is the classifier assigned to the h-th cluster (called an area classifier). In this way, the compound classifier returns the output of the classifier assigned to the cluster where the instance x belongs. It could be a single classifier or an ensemble classifier, which is the case of our proposal. It is important to take into account that the parameter H plays a fundamental role in the performance of the ensemble. On the one hand, a larger number of clusters makes possible a wider exploration of the local competencies of the area classifiers, but on the other hand, it could lead to overfitting. In the present paper, the parameter H is kept fixed along all the experimentation.

In the following lines, we present the classification rule for the area or local classifiers $\bar{\Psi}_h$ which in turn are also ensembles. Let us assume that we count with k (base) classifiers $\Psi^1, \Psi^2, \ldots, \Psi^k$ to build these local (ensemble) classifiers (Note that we use $\bar{\Psi}$ with subindex to refer to the local classifier and Ψ with superindex to refer to the base classifiers that we use to build the local classifiers). For a given instance $x \in X$, each local classifier decides whether x belongs to class $i \in M = \{1, \ldots, m\}$

based on a discriminant function. Let $F^(l)(i,x)$ denote a function that is assigned to class i for a given value of x, and that is used by the l-th classifier. To calculate the response of each of the classifier, a W matrix is defined, that represents the weights use for the discriminant function F. This matrix has k rows and m values, where k is the number of classifiers used in the ensemble and m is the number of classes in the dataset. In this way, the weight matrix corresponding to the h-th cluster W_h could be formulated as follows:

$$W_h = [[w_h^1(1), \ldots, w_h^1(m)], \ldots, [w_h^k(1), \ldots, w_h^k(m)]] \tag{4}$$

Having said this, the local classifier Ψ_h uses the next decision rule:

$$\bar{\Psi}_h(x) = i \text{ if } \hat{F}_h(i,x) = max_{j \in M} \hat{F}_h(j,x) \tag{5}$$

where

$$\hat{F}_h(i,x) = \sum_{l=1}^{k} w_h^l(i) F^{(l)}(i,x), \tag{6}$$

and

$$\sum_{l=1}^{k} w_h^l = 1, \forall i \in M \tag{7}$$

Finally, let us assume that for the training of the classifier we have a the learning set LS, that consists in N learning objects. Then, LS is defined as:

$$LS = (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \tag{8}$$

where x_n denotes the values described in the n-th object, and y_n denotes its correct class label.As usual,LS is divided into two subsets: Training set $|TS| = T$, used during training, and Validation set $|VS| = N - T$. In this way, the optimization criteria for the global combined classifier is formulated as:

$$Q(\Psi) = \hat{Q}(\Psi, TS) \tag{9}$$

where $\hat{Q}$ refers to a specific performance metric of the classifier Ψ in the Training set TS (e.g. accuracy, Area Under the Curve, etc.) that is defined according to the user's preferences. In the next subsection, we explain the description and workflow of the training algorithm based on the hybrid metaheuristic GACE.

3.2 Description of the training algorithm

The objective of the training algorithm is to learn the best combination of cluster centroids $C = \{C_1, \ldots, C_H\}$ and ensemble weights $W = \{W_1, \ldots, W_H\}$ that minimizes the objective function described in Equation 9, given set of base classifiers Ψ^1, Ψ^2, ..., Ψ^k. To solve this optimization problem we used the GACE method, as mentioned before. GACE is a hybrid algorithm that combines Genetic Algorithm with Cross Entropy in order to take advantage of the exploratory ability of GA as a search algorithm and the exploitation capability of CE to create synergy between

them. The benefits of GACE as optimization method are supported by its goods results in areas as optimization of Hierarchical Fuzzy Rule-Based Systems [45] or continuous functions [44].

The general working of the training method is as follows (its pseudocode is given in Algorithm 1). First, the initial population (POP), with POP_{size} individuales, is randomly generated following the structure of the codification of the solution. In each generation, the population is then divided into two subpopulations, GA_{pop} and CE_{pop}, with GA_{size} and CE_{size} individuals ($POP_{\text{size}} = GA_{\text{size}}+CE_{\text{size}}$), respectively. The individuals of GA_{pop} are chosen using the corresponding selection operator, while the individuals in CE_{pop} are the CE_{size} best individuals in the current population POP_t. In GA_{pop}, the crossover and mutation operators of the GA are applied, while in CE_{pop}, the CE method is used to evolve the corresponding sub-population. Both subpopulations of new individuals are joined into a single population that then completely replaces the previous one. This process is iteratively repeated until a specified stop condition is reached. Interested readers are referred to [45] for more information about the hybrid algorithm.

Data: $POP_{size}, p_{ga},\ p_c,\ p_m,\ L_r,\ p_{up},\ T_{max}$
Result: *Best individual found*
$GA_{size} \leftarrow \parallel POP_{size} \cdot p_{ga} \parallel$
$CE_{size} \leftarrow POP_{size} - GA_{size}$
$n_{up} \leftarrow \parallel CE_{size} \cdot p_{up} \parallel$
$t \leftarrow 0$
$POP_0, \leftarrow$ Initialize(POP_{size})
$\overline{M} \leftarrow$ Initialize Means vector
$S \leftarrow$ Initialize Standard Deviation vector
Evaluate POP_0
while $t < T_{max}$ **do**
  $GA_{pop} \leftarrow$ SelectionOperator(POP_t, GA_{size})
  $CE_{pop} \leftarrow$ SelectBestSamples(POP_t, CE_{size})
  Offspring$_{GA} \leftarrow$ Crossover(GA_{pop}, p_c)
  Offspring$_{GA} \leftarrow$ Mutation(Offspring$_{GA}$,p_m)
  Offspring$_{CE} \leftarrow$ Generate($CE_{pop}, CE_{size}, \overline{M}, S$)
  $\overline{M} \leftarrow UpdateMeans(L_r, \overline{M}, \text{Offspring}_{CE}, n_{up})$
  $S \leftarrow UpdateDeviation(L_r, S, \text{Offspring}_{CE}, n_{up})$
  $POP_{t+1} \leftarrow$ Offspring$_{GA} \bigcup$ Offspring$_{CE}$
  Evaluate POP_{t+1}
  Add the best individual found to POP_{t+1} if it is not in the population
  $t \leftarrow t + 1$
end

Algorithm 1: Pseudocode of the workflow followed by the optimization method GACE

In the next part of this subsection, we will explain the codification used for the solutions, the initialization process of the population, and the specific crossover and mutation operators employed.

3.2.1 Codification of the solution

In a formal way, one individual in the population is composed of two different parts: one of them codifies the centroids of the partitions (C), and the other one

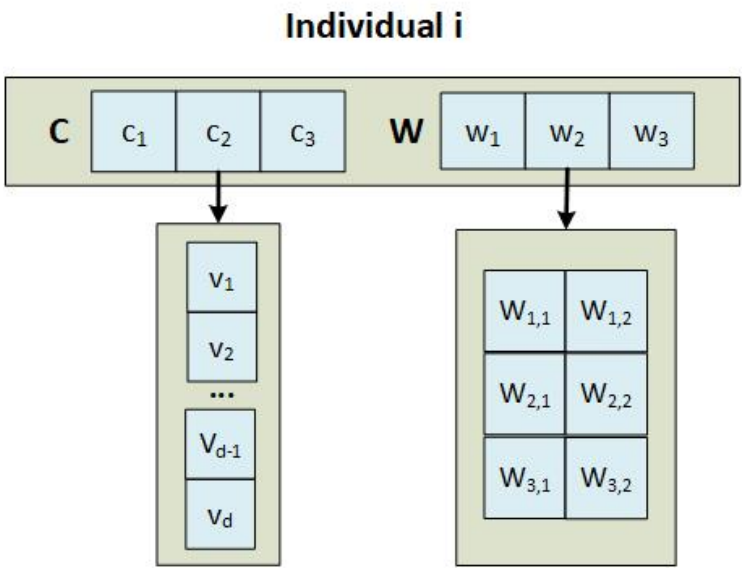

Fig. 1 Structure of an individual in the population with $H = 3$ areas, $k = 3$ classifiers and $m = 2$ classes.

contains the weights used in the discriminant functions of the ensemble classifiers (W). Figure 1 shows an individual with the described structure.

The part codifying the centroids C is represented as an array of H elements, where H is the number of areas or partitions defined by the user. Each centroid c_h, where $h \in [1 \cdots H]$ is the index of the partition, is a vector with the same number of elements as the dimension of the dataset d.

As mentioned in previous sections, there are H matrices contained in W. Each of these matrices, W_h, $h \in [1 \cdots H]$, has $k \times m$ values, where k is the number of classifiers used in the ensemble and m is the number of classes in the dataset, as mentioned before. Each value is the weight that the discriminant function of the classifier l to determine the class i of an instance assigned to in area h. For example, $w_3[1, 2]$ is the weight for the first classifier to determine the second class in the third partition. Each of the possible solutions is represented as shown in Equation 10.

$$Ind(C, W) = \begin{cases} C = (C_1, C_2, \dots C_H) \\ W = (W_1, W_2, \dots W_H) \end{cases} \tag{10}$$

Where $c_h = (v_1, v_2, \dots v_d)$ and $W_h = \{W_h[1, 1], ...W_h[k, m]\}$. Then, GACE is applied to achieve the following goals:

1. Tuning the position of the different centroids C in the feature space.
2. Adjust the values of the weight matrices W, for the different classifiers and classes.

3.2.2 Initialization of the population

For the initial population, each value in C_h is initialized with a random value in the interval $[\min_r, \max_r]$, where $\max_r$ and $\min_r$ are the upper and lower bounds of the r-th dimension of the feature space.

For the weights, each value of the matrices is initialized randomly in the interval $[0, 1]$ and then normalized to ensure that they sum one for each class. As mentioned, each C_h has a size equal to the number of the variables in the feature space, and each W_h has a size of k classifiers per m classes.

3.2.3 Operators of the sub-populations

Different operators are applied in each subpopulation. Selection, crossover, and mutation operators are used in the case of GA_{pop}. As selection operator, Tournament Selection [21] has been adopted. This operator chooses two random individuals in the population and selects the best one according to their fitness. A total of GA_{size} individuals are chosen by this operator to form the GA_{pop} subpopulation. The crossover operator chose was BLX-α [17]. Given two parents $X = (x_1 \dots x_z)$ and $Y = (y_1 \dots y_z)$, for each element i, BLX-α crossover creates two offspring by generating random values in the interval shown in Equation 11, with $\alpha \in [0, 1]$. The choice of this crossover is justified due to its good synergy between exploration and exploitation of the individual [24].

$$[min(x_r, y_r) - \alpha|x_r - y_r|,\ \max(x_r, y_r) + \alpha|x_r - y_r|] \tag{11}$$

Gaussian mutation [4] is taken as the mutation operator. Each element x_i of an individual is updated according to Equation 12:

$$a_r = \mathcal{N}(x_r, \frac{\max_r - \min_r}{10}) \tag{12}$$

where $\mathcal{N}$ is a normal distribution with mean x_r and standard deviation (max_r - min_r).

4 Experimentation

This section presents the results of the experimentation carried out. The objectives of this experimentation are listed below:

- To validate the performance of the proposed ensemble classification approach based on AdaSS and GACE over complex imbalanced data-sets and real-world problems.
- To analyse the influence of the algorithm used to generate the base classifiers and the sizes of the sub-populations of GACE in the performance of the proposal.
- To compare the approach with different state-of-the-art algorithms in imbalanced classification.

The section is structured as follow. In Section 4.1, the different datasets and their main characteristics are presented. The parameter settings and base classifiers are presented in Section 4.2. The analysis of the results and the comparison versus the state-of-the-art are presented in Section 4.3.

No.	Name	Objects	Features	IR
1	Ecoli1	220	7	3.36
2	Ecoli3	336	7	8.6
3	Glass1	214	9	1.82
4	Glass6	214	9	6.28
5	Iris0	150	4	2
6	Page-blocks0	5472	10	8.79
7	Pima	768	8	1.87
8	Vehicle1	846	18	2.9
9	Yeast1	1484	8	2.46
10	Yeast3	1484	8	8.1
11	Glass016vs2	192	9	10.29
12	Ecoli4	336	7	14.3
13	Glass016v5	184	9	19.44
14	Glass5	214	9	22.78
15	Dermatology6	358	34	16.9
16	Shuttle6	230	9	22
17	Poker9	244	10	29.5
18	Yeast28	482	8	23.1
19	Yeast4	1484	8	28.1
20	Led7digit	443	7	10.97
21	Ecoli0137	281	7	39.14
22	WineRed8	656	11	35.44
23	WineWhite9	168	11	32.6
24	Yeast6	1484	8	41.4
25	Poker896	1485	10	58.4
26	WineWhite395	1482	11	58.28
27	Shuttle25	3316	9	66.67
28	WineRed35	691	11	68.1
29	Poker895	2075	10	82
30	Poker86	1477	10	85.88

Table 1 Details of imbalanced datasets used in the experimentation

4.1 Datasets

A total of 30 imbalanced datasets of different complexity have been extracted from the KEEL repository[1] in order to test the performance of the proposal in different kinds of scenarios. These datasets chosen have been used extensively in the literature. Table 1 shows the characteristics of each dataset: name, number of instances, features, classes, and Imbalance Ratio (IR) [76, 77], which is the ratio between the number of instances from the majority and minority classes. The larger the ratio is, the more imbalanced the dataset. The number of classes in these imbalanced datasets is two, which means that we are dealing with imbalanced binary classification. These classes are defined as positive (minority class) and negative (majority class).

In addition, real-world datasets have been used in order to apply the proposal to traffic congestion forecasting in a road; the data collected comes from Lisbon highway A5 and was used in EU FP7 project ICSI[2]. This highway is a 25 km long motorway in Portugal that connects Lisbon with Cascais. Data from a total of 10 sensors in the road have been transformed into datasets, and the proposal has

[1] http://sci2s.ugr.es/keel/datasets.php

[2] http://www.ict-icsi.eu/

been applied to forecast the congestion in each one of them. Each dataset contains 9 variables: day of the week, hour of the day, number of motorbikes, number of cars, number of trucks, number of buses, number of other types of vehicles, total number of vehicles, and a class called *nextlevel*. This class contains the value of congestion that appears in the next hour at a certain point and can take as values $\{LOW, MED, HIG\}$. The level of congestion is defined to be LOW if the total number of vehicles counted is below the 15th percentile, MED (Medium) if it is above the 15th percentile but below the 30th, and HIG (High), otherwise. In the present paper, HIG is the positive class (minority class), and LOW and MED will form the negative class (majority class). The three first weeks of the month were used as the training set and the last week as the test set. This group of datasets will be referred to as A5-Traffic in the following sections.

4.2 Parameter Settings

This section presents the parameter settings establish for the experimentation and the definition of the base classifiers. Three different algorithms have been used for creating the baseline classifiers:

- Minimal distance classifier, which applies the 3-nearest neighbors (3-NN) algorithm [11].
- A Neural Network (NN) method [28], trained with back-propagation algorithm. The number of neurons depends on the dataset used: the size of the input layer is equal to the number of features. The size of the output layer is equal to the number of classes. The number of neurons in the hidden layer is equal to one-half of the sum of the numbers of neurons in the input and output layers. In this case, the total number of iterations was set to 2000 in order to have a fair comparison and not take so much time for bigger datasets.
- Support Vector Machine (SVM) classifier [7], using the sequential minimal optimization procedure with a polynomial kernel.

A homogeneous pool was used in this experimentation, that is, all the base classifiers in the ensemble are built with the same algorithm. To induce diversification, each classifier is trained with a subset of $1/k$-th instances from the training set, where $k = 3$ is the number of classifiers in the pool. Each subset is mutually exclusive from each other and contains the same distribution of examples as the training set. In this way, each ensemble in the experimentation created with our proposal will have tree base classifiers generated with the same algorithm (3-NN, NN or SVM) each one trained with 1/3 of the instances of the dataset. Focusing on the parameters of the algorithm used in the experimentation, the population size (POP_{size}) has been set to 50, and the size of the GA subpopulation to 40 or 45 individuals, $GA_{\text{size}} = \{40, 45\}$. The reason for setting POP_{size} to this value is because of the good performance shown in other classification and optimization tasks [44]. Besides, in those papers, a population with a higher value of GA_{size} than the size of the CE subpopulation (CE_{size}) tended to show better results [45]. As for the GA part parameters, the crossover probability p_c was set to 0.85 and the mutation probability p_m to 0.1. Regarding the CE parameters, the learning rate value L_r is usually recommended to be set within the interval $[0.7, 0.9]$. In this case, $L_r = 0.7$ was chosen. The parameter n_{up}, i.e. the number of individuals that

Parameter	Symbol	Values
Population size	POP_{size}	50
GA pop. size	GA_{size}	{40, 45}
CE pop. size	CE_{size}	{10, 5}
Crossover probability	p_c	0.85
Mutation probability	p_m	0.1
Learning rate	l_r	0.7
No. update	n_{up}	0.4 CE_{size}
No. of areas	H	3
No. of classifiers	k	3
No. of iterations	T_{max}	200

Table 2 Values of the parameters used in the experimentation

is used to update the CE means and standard deviations, was set to $n_{up} = 0.4 \times CE_{\text{size}}$. The number of partitions H is the same as in the present authors' previous paper, and was set to $H = 3$. The number k of classifiers in each pool was also set to 3. The stop condition was designed as follows: in first place, it checks if the best solution does not change in 20 generations, and if so the execution is stopped. Otherwise, it checks if a maximum number of generations $T_{max} = 200$ is fulfilled, stopping the execution in that case. The summary of the parameter settings is presented in Table 2.

4.3 Results

This section presents the results obtained by the proposal using the different configurations mentioned before. A broad comparison of these results with those obtained by state-of-the-art techniques from the literature is made. The following classification techniques from the literature have been used for this comparison:

- RUSBoost (RUS) [56] removes instances from the majority class by randomly undersampling the dataset in each iteration. After training a classifier, the weights of the original dataset instances are updated, and then another sampling phase is applied.
- UnderBagging to OverBagging (UOBag) [61] makes use of both oversampling and undersampling. One of the keys of this algorithm is that the diversity is boosted using a resampling rate in each iteration. This rate defines the number of instances taken from each class. Hence, the first classifiers are trained with a smaller number of instances than the last ones.
- Class and Prototype weighted classifier (CPW) [51] is a method of extracting weights associated to prototypes and classes, with the aim of enhancing the classification accuracy of the 1-NN rule.
- Adaboost [32] is a boosting algorithm which repeatedly invokes a learning algorithm to successively generate a committee of simple, low-quality classifiers.
- FARCHD [1] is a three-stage fuzzy association rule-based classification model which aims to obtain an accurate and compact fuzzy rule-based classifier with a low computational cost.
- Multilayer perceptron for Cost-Sensitive classification problems (NNCS) [71] uses a multilayer Perceptron to classify a dataset of examples with minimal cost.

It is important to note that some of the compared techniques use a pre-processing algorithm to modify the data before its execution. RUS uses SMOTE, and UOBag applies resampling to the data before the application of C4.5 as a base classifier. The techniques mentioned above have been included to determine whether the performance of the proposal of this paper, to which we will refer asAdaSSGACE, reaches or exceeds that of state-of-the-art techniques that do use pre-processing techniques.

KEEL [2] has been used for running the state of the art techniques, and MATLAB r2017 using PRTools Toolbox [3] for the execution of AdaSSGACE. The experiments were carried out on an Intel Xeon E5 2.30 GHz computer with 32 GB of RAM. For validation, 5-fold cross-validation was used. The number of repetitions made for each method was set to 10. The performance metric that we used to set the function $\hat{Q}(\Psi, TS)$ defined in Equation 9 is the Area Under the ROC Curve (AUC), which is calculated as in Equation 13:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{13}$$

where TP_{rate} and FP_{rate} correspond to the True Positive ratio and the False Positive ratio, respectively. This metric is used in order to compare imbalanced datasets in a fair way. It indicates the central tendency of the results obtained by each method. The configurations of the proposed technique are denoted by $AdaSSGACE_{BC,GA_{\text{size}}}$, where BC (either K-NN, NN, or SVM) is the algorithm used to generate the three base classifiers of the ensemble, and GA_{size} is the size of the GA subpopulation. In this way, $AdaSSGACE_{KNN.40}$ indicates that this ensemble has three base classifiers generated with KNN and the GA_{size} parameter was set to 40.

Table 3 shows the results obtained by the techniques in the imbalanced datasets with IR less than 10. Bold values represent the two best AUC values obtained on the corresponding dataset. The most remarkable configurations of the proposal are those formed by the couple (K-NN, 40) and by SVM with both population sizes. In the case of the techniques from the state of the art, RUS and UOBag are the two techniques with better results, with similar values which, in turn, are similar to those obtained by $AdaSSGACE_{KNN,40}$.

	Ecoli1	Ecoli3	Glass1	Glass6	Iris0	Page-blocks0	Pima	Vehicle1	Yeast1	Yeast3
$AdaSSGACE_{KNN.40}$	**0.890**	**0.864**	0.750	0.889	**0.999**	0.931	0.703	0.673	0.700	**0.897**
$AdaSSGACE_{NN.40}$	0.833	0.734	0.647	**0.903**	0.970	0.880	**0.738**	**0.778**	**0.711**	0.890
$AdaSSGACE_{SVM.40}$	0.872	0.785	0.639	0.641	0.841	0.751	0.589	0.651	0.688	0.891
$AdaSSGACE_{KNN.45}$	0.223	0.202	0.468	0.425	0.140	0.930	0.699	0.684	0.701	0.897
$AdaSSGACE_{NN.45}$	0.799	0.739	0.649	0.905	0.963	0.883	0.733	0.766	0.710	0.889
$AdaSSGACE_{SVM.45}$	0.875	0.761	0.635	0.647	0.910	0.757	0.577	0.652	0.681	0.892
RUS	**0.884**	0.840	**0.780**	**0.921**	0.990	**0.956**	0.725	**0.786**	0.701	**0.919**
UOBag	0.876	**0.886**	0.739	0.901	0.970	**0.953**	0.730	0.745	**0.720**	**0.919**
CPW	0.812	0.747	**0.775**	0.871	**1.000**	0.871	0.664	0.628	0.664	0.827
Adaboost	0.843	0.799	0.613	0.880	**1.000**	0.840	**0.742**	0.702	0.599	0.805
FARCHD	0.857	0.753	0.718	0.894	**1.000**	0.754	0.704	0.624	0.671	0.854
NNCS	0.854	0.856	0.607	0.851	**1.000**	0.736	0.727	0.660	0.677	0.743

Table 3 AUC obtained by the techniques on imbalanced datasets with IR less than 10

Table 4 shows the results obtained for those datasets with IR between 10 and 30. As in the previous table, bold values represent the two best AUC val-

[3] prtools.org/prtools/prtools-overview/

ues obtained on each dataset. In this case, those configurations that used K-NN to generate the base classifiers obtain, on average, better results than the rest of AdaSSGACE configurations, although the NN configurations also get similar results. As for the state-of-the-art techniques, RUS and UOBag continue obtaining good performance on almost every dataset. Also remarkable is the performance obtained by the FARCHD technique, which improves the values obtained when it was applied to those datasets with IR less than 10.

	Dermatology	Ecoli4	Glass016vs2	Glass016v5	Glass5	Led7digit	Poker9	Shuttle6	Yeast28	Yeast4
$AdaSSGACE_{KNN.40}$	0.938	**0.933**	**0.708**	0.867	0.804	0.851	0.740	0.960	**0.796**	0.798
$AdaSSGACE_{NN.40}$	**0.966**	0.760	0.630	0.796	0.718	0.856	0.563	0.920	0.720	0.739
$AdaSSGACE_{SVM.40}$	0.749	0.907	0.611	0.733	0.675	0.785	0.636	0.843	0.737	0.509
$AdaSSGACE_{KNN.45}$	0.946	**0.940**	0.678	0.863	0.785	0.845	0.705	**0.965**	**0.801**	**0.799**
$AdaSSGACE_{NN.45}$	0.979	0.805	0.594	0.832	0.764	0.839	0.586	0.894	0.711	0.709
$AdaSSGACE_{SVM.45}$	0.768	0.917	0.592	0.697	0.724	0.785	0.628	0.833	0.742	0.514
RUS	**0.966**	0.896	**0.700**	**0.954**	0.949	**0.894**	0.590	0.902	0.747	**0.827**
UOBag	0.938	0.867	0.629	**0.963**	**0.988**	0.881	0.556	0.948	0.778	0.763
CPW	0.500	0.870	0.577	0.836	0.893	0.500	**0.950**	0.900	0.769	0.677
Adaboost	0.500	0.842	0.494	0.891	**0.995**	**0.910**	0.572	0.900	0.770	0.548
FARCHD	**0.949**	0.872	0.491	0.789	0.745	0.883	**0.848**	**1**	0.700	0.565
NNCS	0.893	0.660	0.471	0.880	0.995	0.647	0.604	0.813	0.652	0.543

Table 4 AUC obtained by the techniques on imbalanced datasets with IR in the interval [10,30]

Finally, Table 5 contains the results obtained by the techniques in those datasets with IR greater than 30. As in previous cases, K-NN configurations lead to better performance. Following the results obtained in the previous datasets, RUS technique obtains the best results among the state-of-the-art techniques. In this case, FARCHD improves the results obtained by UOBAG, placing itself as the second best state-of-the-art technique in this case. While the results of the proposal configurations are close, equal or improve the results obtained by the best techniques in most cases, in some datasets such as Poker896 they are far from the best results. This may be due to the fact that in datasets with high IR a bagging or boosting method significantly improves the obtained results.

	Ecoli0137	Poker895	Poker896	Poker6	Shuttle25	WineRed35	WineRed8	WineWhite395	WineWhite9	Yeast6
$AdaSSGACE_{KNN.40}$	0.790	**0.631**	0.618	0.528	**0.986**	0.590	0.589	0.535	0.608	**0.875**
$AdaSSGACE_{NN.40}$	0.848	0.507	0.572	0.576	0.874	0.559	0.588	0.558	0.598	0.772
$AdaSSGACE_{SVM.40}$	0.682	0.588	0.557	0.469	0.672	0.580	0.528	0.531	0.576	0.515
$AdaSSGACE_{KNN.45}$	0.814	**0.629**	0.623	0.526	0.982	0.608	0.578	0.535	0.645	**0.868**
$AdaSSGACE_{NN.45}$	0.848	0.507	0.586	0.551	0.912	0.579	0.588	0.561	0.580	0.757
$AdaSSGACE_{SVM.45}$	0.685	0.596	0.483	0.494	0.642	0.597	0.535	0.560	0.566	0.524
RUS	**0.896**	0.547	**0.915**	0.631	**1**	**0.644**	**0.815**	**0.674**	**0.893**	0.851
UOBag	0.867	0.618	0.534	0.584	**1**	**0.615**	**0.700**	**0.576**	0.714	0.814
CPW	0.870	0.517	0.504	0.504	**1**	0.494	0.541	0.537	**0.894**	0.734
Adaboost	0.842	0.495	0.735	**0.864**	0.929	0.547	0.528	0.599	0.685	0.598
FARCHD	**0.872**	0.500	**0.960**	**0.900**	**1**	0.499	0.498	0.500	0.788	0.599
NNCS	0.660	0.529	0.503	0.451	0.590	0.583	0.664	0.444	0.418	0.686

Table 5 AUC obtained by the techniques on imbalanced datasets with IR greater than 30

The AUC obtained by the techniques on the A5-Traffic datasets are collected in Table 6. A total of 10 real-data datasets are used, and each execution was repeated 10 times. The name of each column corresponds to the name of the dataset. For this part of the experimentation, the results obtained by all the techniques are similar. The techniques mentioned in the previous experiments maintain good performance, while others with poorer results, such as CPW or Adaboost, improve their performance on these datasets.

	CL_{400}	CL_{600}	CL_{1505}	CL_{1980}	CL_{3600}	CL_{4000}	CL_{6800}	CL_{7100}	CL_{8050}	CL_{9400}
$AdaSSGACE_{KNN.40}$	0.913	0.869	0.945	**0.956**	0.881	0.945	0.810	0.922	0.961	0.845
$AdaSSGACE_{NN.40}$	0.925	0.846	**0.966**	**0.956**	0.887	0.956	0.809	0.946	0.961	0.905
$AdaSSGACE_{SVM.40}$	0.916	0.785	0.927	0.953	0.707	0.952	0.817	0.908	**0.963**	0.697
$AdaSSGACE_{KNN.45}$	0.909	0.868	0.949	0.954	0.873	0.947	0.806	0.923	0.960	0.850
$AdaSSGACE_{NN.45}$	0.915	0.828	**0.969**	0.953	0.891	0.954	0.828	0.940	0.959	0.886
$AdaSSGACE_{SVM.45}$	0.916	0.803	0.931	**0.957**	0.644	0.953	0.812	0.860	0.961	0.741
RUS	**0.975**	**0.978**	0.952	0.949	**0.940**	**0.961**	**0.889**	0.950	0.951	0.663
UOBag	**0.997**	**0.966**	**0.966**	0.949	0.922	**0.958**	**0.889**	0.937	**0.965**	0.825
CPW	0.967	0.868	0.965	0.500	0.852	0.500	**0.864**	**0.975**	0.500	**0.967**
Adaboost	0.950	0.894	**0.966**	0.938	**0.927**	0.938	0.839	0.958	0.948	**0.971**
FARCHD	0.905	0.841	0.984	0.945	0.873	0.935	0.864	**0.962**	0.957	0.929
NNCS	0.863	0.876	0.945	0.887	0.829	0.933	0.818	0.914	0.933	0.850

Table 6 AUC obtained by the techniques on A5-Traffic datasets

To assess whether the differences in performance observed in the previous tables are significant or not, it is necessary to perform statistical tests. For this reason, in this article we follow the guidelines proposed in [13], where non-parametric statistical testing is suggested in situations like the one faced in this study (several datasets, algorithms and configurations).

First, the Friedman test [13] has been used for multiple comparisons to check if significant differences exist among the set of algorithms. Besides this, the average rank return by this test allows sorting the algorithms in terms of performance. Each column of Table 7 shows the mean ranking provided by this non-parametric test for each group of datasets (imbalanced with IRs and A5-Traffic), and globally over all datasets. The best global rank is obtained by RUS on all the datasets, followed by UOBag. The proposal configuration which obtains the best rank so far is (K-NN, 40), followed by (NN, 40). Looking at each group of datasets, we can see that RUS gets the best average ranking in the groups with IR lower than 10 and higher than 30, $AdaSSGACE_{KNN,40}$ in the datasets with IR between 10 and 30, and UOBag in A5-Traffic.

	< 10	[10, 30]	> 30	A5-Traffic	Global
$AdaSSGACE_{KNN.40}$	4.45	3.75	5.25	7.1	5.61
$AdaSSGACE_{NN.40}$	5.65	7.25	6.95	5.15	6.56
$AdaSSGACE_{SVM.40}$	8.6	8.75	9.45	8.5	9.075
$AdaSSGACE_{KNN.45}$	8.7	3.8	5.45	7.55	6.75
$AdaSSGACE_{NN.45}$	6	7.6	6.75	5.75	6.86
$AdaSSGACE_{SVM.45}$	8.3	8.65	8.7	8.15	8.77
RUS	2.8	3.85	2.25	4.6	3.73
UOBAG	3.2	5.05	3.75	3.8	4.53
CPW	8.05	7.2	6.75	6.7	5.57
Adaboost	6.95	6.65	6.85	5.1	5.52
FARCHD	7.45	6.5	6.35	6.4	7.06
NNCS	7.85	8.95	9.5	9.2	7.92

Table 7 Results of Friedman test for all the techniques used

To assess if the performance of the best technique in the experimentation is significantly different from the other techniques from the state-of-the-art, we applied Holm's [27] and Finner's [19] post-hoc tests. Table 8 shows the results returned by the Holm's and Finner's post-hoc tests using RUS as the control method since it obtained the best average rank. These tests were applied for each group of dataset, and globally over all datasets. The differences are considered

	<10		[10, 30]		>30		A5-Traffic		Global	
	Holm	Finner	Holm	Finner	Holm	Finner	Holm	Finner	Holm	Finner
$AdaSSGACE_{KNN.40}$	0.612	0.331	1.901	0.963	0.142	0.069	0.285	0.087	0.080	0.027
$AdaSSGACE_{NN.40}$	0.231	0.093	0.210	0.065	0.028	0.010	1.207	0.467	0.002	0.001
$AdaSSGACE_{SVM.40}$	0.003	0.003	0.019	0.014	0	0	0.036	0.019	0	0
$AdaSSGACE_{KNN.45}$	0.003	0.003	1.901	0.975	0.142	0.057	0.160	0.054	0.001	0
$AdaSSGACE_{NN.45}$	0.189	0.064	0.136	0.046	0.032	0.010	0.906	0.298	0.001	0
$AdaSSGACE_{SVM.45}$	0.006	0.003	0.021	0.014	0.001	0	0.063	0.025	0	0
UOBAG	0.804	0.804	1.260	0.486	0.352	0.352	1.207	0.620	0.321	0.321
CPW	0.009	0.003	0.210	0.065	0.032	0.010	0.433	0.128	0.080	0.028
Adaboost	0.050	0.016	0.360	0.111	0.030	0.010	1.207	0.467	0.080	0.029
FARCHD	0.024	0.007	0.360	0.119	0.044	0.015	0.534	0.163	0	0
NNCS	0.012	0.004	0.014	0.014	0	0	0.009	0.009	0	0

Table 8 Results of Holm and Finner tests for the experimental techniques

significant when the p-value return by the test is lower than 0.05. The values are rounded up to a maximum of 3 decimal places for the sake of the visualization.

These tests show that the results obtained by the proposed technique have no significant differences from those obtained by the state-of-the-art technique when the base classifiers are generated by K-NN and GA_{size} is set to 40, according to Holm's test, and only when the results are considered in a global way, according to Finner's test. In the rest of the configurations, with can find more significant differences w.r.t. RUSBoost, especially in datasets with IR higher than 30 and globally. Finally, in the case of state-of-the-art techniques, the reference technique obtains significantly better results than all the techniques except UOBAG in global values.

5 Conclusions

In this paper, we have presented a new ensemble classification approach for imbalanced data based on feature space partitioning and hybrid metaheuristics, and concretely, on the Adaptive Splitting and Selection Strategy and the GACE metaheuristic, respectively. The main objective of this new method was to deal with imbalance data without the use of data-level methods that usually entails an extra-cost in terms of the time required for preprocessing the data.

The developed technique has been applied to a total of 40 datasets of different types: datasets with different imbalance ratios, and real imbalanced data-set with traffic information. Furthermore, the proposal has been compared with state-of-the-art classification techniques in the literature, such as RUSBoost, FARCHD, CPW, Adaboost, and NNCS. The performance obtained by the proposed method in most cases is similar to or better than the results obtained by the compared techniques, regardless of whether or not they used data-level methods. The best results so far have been obtained with configurations with KNN and NN as the algorithm to generate the base classifiers, with different sizes for the genetic populations. Statistical tests have been applied in order to corroborate the results obtained.

As future lines of research, it would be interesting to use a heterogeneous pool of classifiers instead of a homogeneous one. Also, another type of algorithms to generate the base classifiers could be explored. Configurations with various different sizes of the GA sub-population could be applied in order to show a more dedicated analysis to the optimization method in this theme. Besides, more techniques from

the literature could be used for the comparison. In this paper, the experimentation is focused on the performance of the presented proposal in binary imbalanced classification. In future works, OVO and OVA-based ensembles will be used with multi-class imbalanced datasets, and a study about the time consuming and the diversity of different population configurations will be made. Finally, in this work, the hybrid method is formed by joining a GA and CE techniques. Other methods such as PSO, ACO or BAT Algorithm could be used to replace either of the two components in order to compare the performance against the proposal in ensemble classification.

Acknowledgments

This work has been supported by the research projects TEC2013-45585-C2-2-R and TIN2014-56042-JIN from the Spanish Ministry of Economy and Competitiveness, and the TIMON project, which received funding from the European Union Horizon 2020 research and innovation programme under grant agreement No. 636220.

References

1. Alcala-Fdez J, Alcala R, Herrera F (2011) A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning. IEEE Transactions on Fuzzy Systems, 19(5):857–872
2. Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, Garca S, Sánchez L, Herrera F (2011) KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic and Soft Computing, 17(2–3):255–287
3. Amami R, Ben Ayed D, Ellouze N (2013) Adaboost with SVM using GMM supervector for imbalanced phoneme data. Human System Interaction (HSI), 2013 The 6th International conference on, 328–333
4. Bäck T, Schwefel H (1993) An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation, 1(1):1–23
5. Bi Y, Guan J, Bell D (2008) The combination of multiple classifiers using an evidential reasoning approach. Artificial Intelligence, 172(15):1731–1751
6. Bian J, Peng XG, Wang Y, Zhang H (2016) An efficient cost-sensitive feature selection using chaos genetic algorithm for class imbalance problem. Mathematical Problems in Engineering 2016
7. Burges C (1998) A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2):121–167
8. Cervantes J, Huang DS, Garca-Lamont F, Chau A (2014) A hybrid algorithm to improve the accuracy of support vector machines on skewed data-sets. International Conference on Intelligent Computing, 782–788.
9. Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) SMOTEBoost: Improving prediction of the minority class in boosting. In: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 107–119

10. Cheng F, Yang K, Zhang L (2015) A structural SVM based approach for binary classification under class imbalance. Mathematical Problems in Engineering 2015
11. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1):21–27
12. Danesh A, Moshiri B, Fatemi O (2007) Improve text classification accuracy based on classifier fusion methods. 10th International Conference on Information Fusion, 1–6
13. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation, 1(1):3–18
14. Díez-Pastor JF, Rodríguez GOC Juan J, Kuncheva LI (2015) Random balance: Ensembles of variable priors classifiers for imbalanced data. Knowledge-Based Systems, 85:96–111
15. Dorigo M, Gambardella LM (1997) Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation, 1(1):53–66
16. Duin RP (2002) The combining classifier: To train or not to train? Proceedings 16th International Conference Patter Recognition, IEEE, 2:765–770
17. Eshelman LJ, Schaffer JD (1992) Real-coded genetic algorithms and interval-schemata. Foundations of genetic algorithms, 2:187-202
18. Fattahi S, Othman Z, Othman Z (2015) New approach with ensemble method to address class imbalance problem. Journal of Theoretical and Applied Information Technology, 72(1)
19. Finner H (1993) On a monotonicity problem in step-down multiple test procedures. Journal of the American Statistical Association, 88(423):920–923
20. Giacinto G, Roli F (2001) Dynamic classifier selection based on multiple classifier behaviour. Pattern Recognition, 34(9):1879–1881
21. Goldberg DE, Deb K (1991) A comparative analysis of selection schemes used in genetic algorithms. Foundations of Genetic Algorithms, 1:69–93
22. Haixiang G, Xiuwu L, Kejun Z, Chang D, Yanhui G (2011) Optimizing reservoir features in oil exploration management based on fusion of soft computing. Applied Soft Computing, 11(1):1144–1155
23. Hashem S (1997) Optimal linear combinations of neural networks. Neural networks, 10(4):599–614
24. Herrera F, Lozano M, Verdegay JL (1998) Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. Artificial Intelligence Review, 12(4):265–319
25. Ho D, Drake T, Bentley R, Valea F, Wax A (2015) Evaluation of hybrid algorithm for analysis of scattered light using ex vivo nuclear morphology measurements of cervical epithelium. Biomedical Optics Express, 6(8):2755–2765
26. Holland JH (1992) Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press
27. Holm S (1979) A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics, 6:65–70

28. Hopfield J (1982) Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences of the United States of America, 79(8):2554–2558
29. Jackowski K, Wozniak M (2009) Algorithm of designing compound recognition system on the basis of combining classifiers with simultaneous splitting feature space into competence areas. Pattern Analysis and Applications, 12(4):415–425
30. Jackowski K, Krawczyk B, Woniak M (2014). Improved adaptive splitting and selection: the hybrid training method of a classifier based on a feature space partitioning. International journal of neural systems, 24(03):1430007.
31. Jackowski, K. (2015) Adaptive splitting and selection algorithm for regression. New Generation Computing, 33(4):425–448.
32. del Jesus M, Hoffmann F, Junco L, Sánchez L (2004) Induction of fuzzy-rule-based classifiers with evolutionary boosting algorithms. IEEE Transactions on Fuzzy Systems, 12(3):296–308
33. Jurek A, Bi Y, Wu S, Nugent C (2011) Classification by cluster analysis: A new meta-learning based approach. Multiple Classifier Systems, 259–268
34. Jurek A, Bi Y, Wu S, Nugent C (2014) A survey of commonly used ensemble-based classification techniques. The Knowledge Engineering Review, 29(5):551–581
35. Kennedy J (2011) Particle swarm optimization. Encyclopedia of Machine Learning, Springer, 760–766
36. Krawczyk B, Cyganek B. (2017). Selecting locally specialised classifiers for one-class classification ensembles. Pattern analysis and applications, 20(2):427–439.
37. Krawczyk B, McInnes B T (2018). Local ensemble learning from imbalanced and noisy data for word sense disambiguation. Pattern Recognition, 78:103–119.
38. Kuncheva L (2000) Clustering-and-selection model for classifier combination. Proceedings of the Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 1:185–188
39. Kuncheva LI (2004) Combining Pattern Classifiers: Methods and Algorithms. John Wiley & Sons.
40. Kuncheva LI, Jain LC (2000) Designing classifier fusion systems by genetic algorithms. IEEE Transactions on Evolutionary computation, 4(4):327–336
41. Kuncheva LI, Whitaker CJ, Shipp CA, Duin RP (2003) Limits on the majority vote accuracy in classifier fusion. Pattern Analysis & Applications, 6(1):22–31
42. Lavanya S, Palaniswami S, Divyabharathi M (2015) Resampling ensemble algorithm for class imbalance problem using optimization algorithm. International Journal of Applied Engineering Research, 10(13):11520-11526
43. Liu X, Lin J, Deng K (2011) Scheduling optimization in re-entrant lines based on a GA and PSO hybrid algorithm. Tongji Daxue Xuebao/Journal of Tongji University, 39:726-729
44. Lopez-Garcia P, Onieva E, Osaba E, Masegosa A, Perallos A (2016) Gace: A meta-heuristic based in the hybridization of genetic algorithms and cross entropy methods for continuous optimization. Expert Systems with Applications, 55:508–519

45. Lopez-Garcia P, Onieva E, Osaba E, Masegosa AD, Perallos A (2016) A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy. IEEE Transactions on Intelligent Transportation Systems, 17(2):557–569
46. Lopez-Garcia P, Woniak M, Onieva E, Perallos A (2016) Hybrid optimization method applied to adaptive splitting and selection algorithm. Lecture Notes in Computer Science, Springer, 9648:742–750.
47. Mannor S, Peleg B, Rubinstein R (2005) The cross entropy method for classification. Proceedings of the 22nd International Conference on Machine Learning, ACM, 561–568
48. Maua G, Galinac Grbac T (2017) Co-evolutionary multi-population genetic programming for classification in software defect prediction: An empirical case study. Applied Soft Computing Journal, 55:331-351
49. Mokeddem D, Belbachir H (2009) A survey of distributed classification based ensemble data mining methods. Journal of Applied Sciences, 9(20):3739-3745
50. Opitz DW, Maclin R (1999) Popular ensemble methods: An empirical study. Journal of Artificial Intelligence Research, 11:169–198
51. Paredes R, Vidal E (2006) Learning weighted metrics to minimize nearest-neighbor classification error. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(7):1100–1110
52. Qian Y, Liang Y, Li M, Feng G, Shi X (2014) A resampling ensemble algorithm for classification of imbalance problems. Neurocomputing, 143:57-67
53. Rokach L (2010) Ensemble-based classifiers. Artificial Intelligence Review, 33(1):1–39
54. Rubinstein R (1999) The cross-entropy method for combinatorial and continuous optimization. Methodology and Computing in Applied Probability, 1(2):127–190
55. Ruta D, Gabrys B (2005) Classifier selection for majority voting. Information Fusion, 6(1):63–81
56. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2010) RUSBoost: A hybrid approach to alleviating class imbalance. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 40(1):185–197
57. Sentinella M, Casalino L (2009) Cooperative evolutionary algorithm for space trajectory optimization. Celestial Mechanics and Dynamical Astronomy, 105(1-3):211
58. Stanciu S, Tranca D, Stanciu G, Hristu R, Bueno J (2016) Perspectives on combining nonlinear laser scanning microscopy and bag-of-features data classification strategies for automated disease diagnostics. Optical and Quantum Electronics, 48(6):320
59. Talbi EG (2002) A taxonomy of hybrid metaheuristics. Journal of Heuristics 8(5):541–564
60. Vorraboot P, Rasmequan S, Chinnasarn K, Lursinsap C (2015) Improving classification rate constrained to imbalanced data between overlapped and non-overlapped regions by hybrid algorithms. Neurocomputing, 152:429-443
61. Wang S, Yao X (2009) Diversity analysis on imbalanced data sets by using ensemble models. Proceedings of IEEE Symposium in Computational Intelligence and Data Mining, 2009, CIDM'09, 324–331
62. Wang S, Yao X (2012) Multiclass imbalance problems: Analysis and potential solutions. IEEE Transactions on Systems, Man, and Cybernetics, Part B

(Cybernetics), 42(4):1119–1130
63. Wang S, Minku L, Yao X (2015) Resampling-based ensemble methods for online class imbalance learning. IEEE Transactions on Knowledge and Data Engineering, 27(5):1356-1368
64. Wang Z, Wong Y, Rahman M (2004) Optimisation of multi-pass milling using genetic algorithm and genetic simulated annealing. The International Journal of Advanced Manufacturing Technology, 24(9–10):727–732
65. Xu L, Krzyzak A, Suen CY (1992) Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Transactions on Systems, Man, and Cybernetics, 22(3):418–435
66. Yang J, Ji Z, Xie W, Zhu Z (2016) Model selection based on particle swarm optimization for omics data classification. Shenzhen Daxue Xuebao (Ligong Ban)/Journal of Shenzhen University Science and Engineering, 33(3):264–271
67. Yang P, Xu L, Zhou B, Zhang Z, Zomaya A (2009) A particle swarm based hybrid system for imbalanced medical data sampling. BMC Genomics 10(Suppl. 3) DOI: 10.1186/1471-2164-10-S3-S34
68. Yang XS (2010) A new metaheuristic bat-inspired algorithm. Studies in Computational Intelligence, 284:65–74
69. Yu H, Ni J, Zhao J (2013) ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data. Neurocomputing, 101:309–318
70. Zheng J, Yan R (2012) Attribute reduction based on cross entropy in rough set theory. Journal of Information and Computational Science, 9(3):745–750
71. Zhou ZH, Liu XY (2006) Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Transactions on Knowledge and Data Engineering, 18(1):63–77
72. Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence, 5(4):221–232.
73. Cano, A., Zafra, A., Ventura, S. (2013). Weighted data gravitation classification for standard and imbalanced data. IEEE transactions on cybernetics, 43(6):1672–1687.
74. Mahdizadehaghdam, S., Dai, L., Krim, H., Skau, E., Wang, H. (2017). Image classification: A hierarchical dictionary learning approach. IEEE International Conference In Acoustics, Speech and Signal Processing (ICASSP), 2017, 2597–2601
75. Khari, M., Kumar, P., Burgos, D., Crespo, R. G. (2017). Optimized test suites for automated testing using different optimization techniques. Soft Computing, 1–12.
76. Fernndez, A., Garca, S., Herrera, F. (2011). Addressing the classification with imbalanced data: open problems and new challenges on class distribution. Hybrid Artificial Intelligent Systems, 1–10.
77. Sun, Y., Wong, A. K., Kamel, M. S. (2009). Classification of imbalanced data: A review. International Journal of Pattern Recognition and Artificial Intelligence, 23(04):687–719.
78. Tselentis, D.I., Vlahogianni, E.I., Karlaftis, M.G. (2014). Improving short-term traffic forecasts: to combine models or not to combine?. IET Intelligent Transport Systems, 9(2):193–201.