Knowledge-Based Systems 38 (2013) 85–104

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

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

A hierarchical genetic fuzzy system based on genetic programming for addressing classification with highly imbalanced and borderline data-sets

Victoria López [a,*], Alberto Fernández [b], María José del Jesus [b], Francisco Herrera [a]

[a] Dept. of Computer Science and Artificial Intelligence, CITIC-UGR, Research Center on Information and Communications Technology, University of Granada, 18071 Granada, Spain
[b] Dept. of Computer Science, University of Jaén, 23071 Jaén, Spain

ARTICLE INFO

Article history:
Received 18 October 2011
Received in revised form 3 June 2012
Accepted 24 August 2012
Available online 3 October 2012

Keywords:
Fuzzy rule based classification systems
Hierarchical fuzzy partitions
Genetic rule selection
Tuning
Imbalanced data-sets
Borderline examples

ABSTRACT

Lots of real world applications appear to be a matter of classification with imbalanced data-sets. This problem arises when the number of instances from one class is quite different to the number of instances from the other class. Traditionally, classification algorithms are unable to correctly deal with this issue as they are biased towards the majority class. Therefore, algorithms tend to misclassify the minority class which usually is the most interesting one for the application that is being sorted out.

Among the available learning approaches, fuzzy rule-based classification systems have obtained a good behavior in the scenario of imbalanced data-sets. In this work, we focus on some modifications to further improve the performance of these systems considering the usage of information granulation. Specifically, a positive synergy between data sampling methods and algorithmic modifications is proposed, creating a genetic programming approach that uses linguistic variables in a hierarchical way. These linguistic variables are adapted to the context of the problem with a genetic process that combines rule selection with the adjustment of the lateral position of the labels based on the 2-tuples linguistic model.

An experimental study is carried out over highly imbalanced and borderline imbalanced data-sets which is completed by a statistical comparative analysis. The results obtained show that the proposed model outperforms several fuzzy rule based classification systems, including a hierarchical approach and presents a better behavior than the C4.5 decision tree.

1. Introduction

Learning from imbalanced data-sets is an issue that has attracted a lot of attention in machine learning research [29,51]. This problem is characterized by a class distribution where the number of examples in one class is outnumbered by the number of examples in the other class. The presence of imbalanced data-sets is dominant in a high number of real problems including, but not limited to, medical diagnosis, fraud detection, finances, risk management, network intrusion and so on. Additionally, the positive or minority class is usually the one that has the highest interest from the learning point of view and it also implies a great cost when it is not well classified [17,57].

A standard classifier that seeks accuracy over a full range of instances is frequently not suitable to deal with imbalanced learning tasks, since it tends to be overwhelmed by the majority class thus misclassifying the minority examples. This situation becomes critical when the minority class is greatly outnumbered by the majority class, generating an scenario of highly imbalanced data-sets where the performance deterioration is amplified. However, some studies have shown that imbalance for itself is not the only factor that hinders the classification performance [37]. There are several data intrinsic characteristics which lower the learning effectiveness. Some of these handicaps within the data are the presence of small disjuncts [53], the overlap between the classes [26] or the existence of noisy [49] and borderline [44] samples. There is no need to say that when the classification data share an skewed data distribution together with any of the aforementioned situations, the performance degradation is intensified [19,42,53].

A large number of approaches have been proposed to deal with the class imbalance problem. Those solutions fall largely into two major categories. The first is data sampling in which the training data distribution is modified to obtain a set with a balanced distribution. Standard classifiers are thus helped to obtain a correct identification of data [9,6]. The second is through algorithmic modification where the base learning methods are modified to consider the imbalanced distribution of the data. In this manner, base learning methods change some of its internal operations accordingly [57].

Fuzzy Rule-Based Classification Systems (FRBCSs) [34] are useful and well-known tools in the machine learning framework. They provide a good trade-off between the empirical precision of

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: vlopez@decsai.ugr.es (V. López), alberto.fernandez@ujaen.es (A. Fernández), mjjesus@ujaen.es (M.J. del Jesus), herrera@decsai.ugr.es (F. Herrera).

0950-7051/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2012.08.025

traditional engineering techniques and the interpretability achieved through the use of linguistic labels whose semantic is close to the natural language. Specifically, recent works have shown that FRBCSs have a good behavior dealing with imbalanced data-sets by means of the application of instance preprocessing techniques [20].

The hybridization between fuzzy logic and genetic algorithms leading to Genetic Fuzzy Systems (GFSs) [12,30] is one of the most popular approaches used when different computational intelligence techniques are combined. A GFS is basically a fuzzy system augmented by a learning process based on evolutionary computation. Among evolutionary algorithms, Genetic Programming (GP) [39] is a development of classical genetic algorithms that evolve tree-shaped solutions using variable length chromosomes. GP has been used in FRBCSs to learn fuzzy rule bases [7] profitting from its high expressive power and flexibility.

However, the disadvantage of FRBCSs is the inflexibility of the concept of linguistic variable because it imposes hard restrictions on the fuzzy rule structure [5] which may suppose a loss in accuracy when dealing with some complex systems, such as high dimensional problems, the presence of noise or overlapped classes. Many different possibilities to enhance the linguistic fuzzy modeling have been considered in the specialized literature. All of these approaches share the common idea of improving the way in which the linguistic fuzzy model performs the interpolative reasoning by inducing a better cooperation among the rules in the Knowledge Base (KB). This rule cooperation may be induced acting on three different model components:

- Approaches acting on the whole KB. This includes the KB derivation [43] and a hierarchical linguistic rule learning [14].
- Approaches acting on the Rule Base (RB). The most common approach is rule selection [35] but also multiple rule consequent learning [11] could be considered.
- Approaches acting on the Data Base (DB). For example a priori granularity learning [13] or membership function tuning [1].

In this work, we present a procedure to obtain an Hierarchical Fuzzy Rule Based Classification System (HFRBCS) to deal with imbalanced data-sets. In order to do so, this model introduces modifications both at the data and algorithm level. This procedure is divided into three different steps:

1. A preprocessing technique, the Synthetic Minority Over-sampling Technique (SMOTE) [9], is used to balance the distribution of training examples in both classes.
2. A hierarchical knowledge base (HKB) [14] is generated, using the GP-COACH (Genetic Programming-based learning of COmpact and ACcurate fuzzy rule-based classification systems for High-dimensional problems) algorithm [7] to build the RB. The GP-COACH algorithm has been modified to extend a classical KB into a HKB, integrating a rule expansion process to create high granularity rules in each generation of the algorithm. The usage of a HKB implies an adaptation of the components to allow the interaction of the different granularities in the RB population.
3. A post-processing step involving rule selection and the application of the 2-tuples based genetic tuning is applied to improve the overall performance.

The combination of these steps constitutes a convenient approach to solve the problem of classification with imbalanced data-sets. First of all, the preprocessing technique compensates the number of instances for each class easing the learning process for the consequent procedures. Then, the step to learn the HKB is used to address the imbalanced problem together with some of the data intrinsic characteristics that difficult the learning. This HKB process is appropriate because it increases the accuracy by reinforcing those problem subspaces that are specially difficult in this environment, such as borderline instances [44], small disjuncts [37] or overlapping regions [26]. Finally, the post-processing step refines the results achieved by the previous process. The integration of these schemes completes our proposal, which will be denoted as GP-COACH-H (GP-COACH Hierarchical).

We will focus on two difficult situations in the scenario of imbalanced data, such as highly imbalanced and borderline imbalanced classification problems. For that, we have selected a benchmark of 44 and 30 problems respectively from KEEL data-set repository[1] [2]. We will perform our experimental analysis focusing on the precision of the models using the Geometric Mean of the true rates (GM) [4]. This study will be carried out using non-parametric tests to check whether there are significant differences among the obtained results [25].

This work is structured in the following way. First, Section 2 presents an introduction of classification with imbalanced problems, describing its features, the SMOTE algorithm and the metrics that are used in this framework. Next, Section 3 introduces the proposed approach. Sections 4 and 5 describe the experimental framework used and the analysis of results, respectively. Next, the conclusions achieved in this work are shown in Section 6. Finally, we include an appendix with the detailed results for the experiments performed in the experimental study.

2. Imbalanced data-sets in classification

In this section we delimit the context in which this work is content, briefly introducing the problem of imbalanced classification. Then, we will describe the preprocessing technique that we have applied in order to deal with the imbalanced data-sets: the SMOTE algorithm [9]. We finish this section describing the evaluation metrics that are used in this specific problem with respect to the most common ones in classification.

2.1. The problem of imbalanced data-sets

In some classification problems, the number of examples that represent the diverse classes is very different. Specifically, the imbalance problem occurs when one class is represented only by a few number of examples, while the others are represented by a large number of examples [51,29]. In this paper, we focus on two-class imbalanced data-sets, where there is a positive (minority) class, with the lowest number of instances, and a negative (majority) class, with the highest number of instances.

This problem is prevalent in many real world applications, such as medical diagnosis [45,48], anomaly detection [38], image analysis [8] or bioinformatics [28], just referencing some of them. Furthermore, it is usual that positive classes are the most interesting from the application point of view so it is crucial to correctly identify these cases. The importance of this problem in the aforementioned uses has increased the attention towards it, which has been considered one of the 10 challenging problems in data mining [56].

Although these issues occur frequently in data, many data mining methods do not naturally perform well under these circumstances. In fact, many only work optimally when the classes in data are relatively balanced. Furthermore, the performance of algorithms is usually more degraded when the imbalance increases because positive examples are more easily forgotten. That situation is critical in highly imbalanced data-sets because the number of

[1] http://www.keel.es/datasets.php.

positive instances in the data-set is negligible and that situation increases the difficulty that most learning algorithms have in detecting positive regions. Figs. 1 and 2 depict two data-sets with low imbalance and high imbalance respectively.

However, the imbalanced data-set is also affected by some circumstances that make the learning more difficult. For example, metrics that have been used traditionally seem inappropriate in this scenario when they ascribe a high performance to a trivial classifier that predicts all samples as negative. This behavior is wrapped up in the inner way of building an accurate model, preferring general rules with good coverage for the negative class and disregarding more specific rules which are the ones associated to the positive class.

An important issue that appear in imbalanced data-sets is the presence of borderline examples. Inspired by Kubat and Matwin [40] we may distinguish between safe, noisy and borderline examples. Safe examples are placed in relatively homogeneous areas with respect to the class label. By noisy examples we understand individuals from one class occurring in safe areas of the other class. Finally, borderline examples are located in the area surrounding class boundaries, where the positive and negative classes overlap. These borderline examples make difficult to determine a correct discrimination of the classes. For instance, Napierala et al. [44] present in a series of experiments in which it is shown that the degradation in performance of a classifier in an imbalanced scenario is strongly affected by the number of borderline examples.

2.2. Addressing imbalanced data-sets: use of preprocessing and SMOTE algorithm

A large number of approaches have been proposed to deal with the class-imbalance problem [51,41,42]. These approaches can be categorized in two groups: the internal approaches that create new algorithms or modify existing ones to take the class-imbalance problem into consideration [4] and external approaches that preprocess the data in order to diminish the effect of their class imbalance [6,23,27]. Furthermore, cost-sensitive learning solutions incorporating both approaches assume higher misclassification costs with samples in the positive class and seek to minimize the high cost errors [17,57]. The great advantage of the external approaches is that they are more versatile, since their use is independent of the classifier selected. Furthermore, we may preprocess all data-sets before-hand in order to use them to train different classifiers. In this manner, the computation time needed to prepare the data is only required once. According to this, in this work we have chosen an oversampling method which is a reference in this area: the SMOTE algorithm [9] and a variant called SMOTE + ENN [6].

In this approach, the positive class is over-sampled by taking each positive class sample and introducing synthetic examples along the line segments joining any/all of the k positive class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen. This process is illustrated in Fig. 3, where x_i is the selected point, x_{i1} to x_{i4} are some selected nearest neighbors and r_1 to r_4 the synthetic data points created by the randomized interpolation.

Synthetic samples are generated in the following way: take the difference between the feature vector (sample) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific features. This approach effectively forces the decision region of the positive class to become more general. An example is detailed in Fig. 4.

In short, its main feature is to form new positive class examples by interpolating between several positive class examples that lie together. Thus, the overfitting problem is avoided and causes the decision boundaries for the positive class to spread further into the negative class space.

Nevertheless, class clusters may be not well defined in cases where some negative class examples might be invading the positive class space. The opposite can also be true, since interpolating positive class examples can expand the positive class clusters, introducing artificial positive class examples too deeply into the negative class space. Inducing a classifier in such a situation can lead to over-fitting. For this reason we will also consider in this work a hybrid approach, "SMOTE+ENN", where the Wilson's Edited Nearest Neighbor Rule [54] is used after the SMOTE application to remove any example from the training set misclassified by its three nearest neighbors.

2.3. Evaluation in imbalanced domains

The measures of the quality of classification are built from a confusion matrix (shown in Table 1) which records correctly and incorrectly recognized examples for each class.

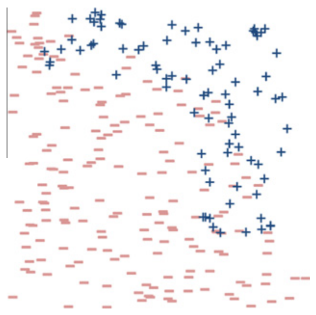
Fig. 1. Data-set with low imbalance (IR = 2.23).

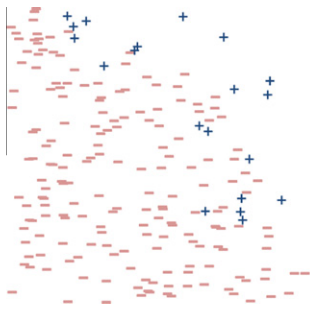
Fig. 2. Data-set with high imbalance (IR = 9.15).

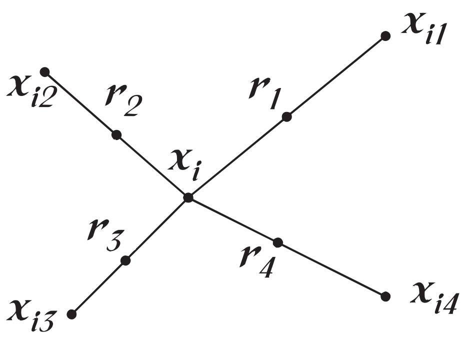

Fig. 3. An illustration of how to create the synthetic data points in the SMOTE algorithm.

```
Consider a sample (6,4) and let (4,3) be its nearest neighbor.
(6,4) is the sample for which k-nearest neighbors
are being identified and (4,3) is one of its k-nearest neighbors.
Let: f1_1 = 6 f2_1 = 4, f2_1 - f1_1 = -2
f1_2 = 4 f2_2 = 3, f2_2 - f1_2 = -1
The new samples will be generated as
(f1',f2') = (6,4) + rand(0-1) * (-2,-1)
rand(0-1) generates a random number between 0 and 1.
```

Fig. 4. Example of the SMOTE application.

The most used empirical measure, accuracy (Eq. (1)), does not distinguish between the number of correct labels of different classes, which in the ambit of imbalanced problems may lead to erroneous conclusions. For example a classifier that obtains an accuracy of 90% in a data-set with a 90% of negative instances, might not be accurate if it does not cover correctly any positive class instance.

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

Because of this, instead of using accuracy, more appropriate metrics in this situation are considered. Two common measures, sensitivity and specificity (Eqs. (2) and (3)), approximate the probability of the positive (negative) label being true. In other words, they assess the effectiveness of the algorithm on a single class.

$$sensitivity = \frac{TP}{TP + FN} \tag{2}$$

$$specificity = \frac{TN}{FP + TN} \tag{3}$$

The metric used in this work is the geometric mean of the true rates [4,40], which can be defined as

$$GM = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{FP + TN}} \tag{4}$$

This metric attempts to maximize the accuracy of each one of the two classes with a good balance. It is a performance metric that links both objectives.

3. The hierarchical genetic programming fuzzy rule based classification system with rule selection and tuning (GP-COACH-H)

In this section, we will describe our proposal to obtain a hierarchical FRBCS through the usage of GP and applying rule selection together with 2-tuples lateral tuning, denoted as GP-COACH-H. This proposal is defined through its components in the following way: Section 3.1 presents a brief introduction of FRBCSs in order to contextualize the algorithm; next, Section 3.2 describes the GP-COACH algorithm [7] which is the linguistic rule generation method based on GP that we have used as base for our proposal of a hierarchical rule base generation method; later, in Section 3.3, the building of the hierarchical fuzzy rule based classification is detailed, mentioning the modifications the hierarchical procedure introduces in the knowledge base generation and in the basic running of the GP-COACH algorithm; subsequently, Section 3.4 shows the selection of the best cooperative rules and the tuning of the databases in a genetic process where both objectives collaborate; and finally, Section 3.5 summarizes the description of the proposal.

Table 1
Confusion matrix for a two-class problem.

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

3.1. Fuzzy rule based classification systems

FRBCSs are useful and well-known tools in the machine learning framework since they can provide an interpretable model for the end user. A FRBCS has two main components: the Inference System and the KB. In a linguistic FRBCS, the KB is composed of a RB, constituted by a set of fuzzy rules, and the DB that stores the membership functions of the fuzzy partitions associated to the input variables. If expert knowledge of the problem is not available, it is necessary to use some Machine Learning process to obtain the KB from examples.

Any classification problem consists of N training patterns $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn})$, $p = 1,2,\ldots,m$ from M classes where x_{pi} is the ith attribute value ($i = 1,2,\ldots,n$) of the pth training pattern.

In this work, we use fuzzy rules of the following form to build our classifier:

$$\text{Rule } R_j : \text{If } x_1 \text{ is } \widehat{A}_1^j \text{ and } \ldots \text{ and } x_n \text{ is } \widehat{A}_n^j \text{ then Class} = C_j \text{ with } RW_j \tag{5}$$

where R_j is the label of the jth rule, $\mathbf{x} = (x_1, \ldots, x_n)$ is an n-dimensional pattern vector, $\widehat{A}_i^j$ is a set of linguistic labels $\{L_i^1 \text{ or } \ldots \text{ or } L_i^{l_k}\}$ joined by a disjunctive operator, C_j is a class label, and RW_j is the rule weight [33]. We use triangular membership functions as linguistic labels whose combination will form an antecedent fuzzy set. This kind of rule is called a DNF fuzzy rule.

To compute the rule weight, many heuristics have been proposed [36]. In our proposal, we compute the rule weight as the fuzzy confidence or Certainty Factor (CF) [15], showed in Eq. (6):

$$RW_j = CF_j = \frac{\sum_{x_p \in ClassC_j} \mu_{\widehat{A}_j}(x_p)}{\sum_{p=1}^{N} \mu_{\widehat{A}_j}(x_p)} \tag{6}$$

where $\mu_{\widehat{A}_j}(x_p)$ is the matching degree of the pattern x_p with the antecedent part of the fuzzy rule R_j.

GP-COACH-H uses the normalized sum fuzzy reasoning method [15] for classifying new patterns by the RB, a general reasoning model for combining information provided by different rules, where each rule promotes the classification with its consequent class according to the matching degree of the pattern with the antecedent part of the fuzzy rule together with its weight. The total sum for each class is computed as follows:

$$Sum_{Class\ h}(x_p) = \frac{\sum_{R_j \in RB, C_j = h} \mu_{\widehat{A}_j}(x_p) \cdot CF_j}{\max\limits_{c=1,\ldots,M} \sum_{R_j \in RB, C_j = c} \mu_{\widehat{A}_j}(x_p) \cdot CF_j} \tag{7}$$

$$Class(x_p) = \arg\max(Sum_{Class\ h}(x_p)) \tag{8}$$

3.2. The GP-COACH algorithm

The GP-COACH algorithm [7] is a genetic programming-based algorithm for the learning of fuzzy rule bases. We will use this method as a base for our hierarchical model modifying its behavior to include the different granularity levels into its inner way of running.

This algorithm is a genetic cooperative-competitive learning approach where the whole population represents the RB obtained. Each individual in the population codifies a rule. These rules are DNF fuzzy rules (Eq. (5)) which allow the absence of some input features and are generated according to the production rules of a context-free grammar. As DB we are using linguistic partitions with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions.

There are two evaluation functions in the GP-COACH algorithm: a local fitness function, known as *raw_fitness*, to evaluate the performance of each rule and a global fitness function, known as *global_fitness*, to evaluate the behavior of the whole rule population. The *raw_fitness* is computed according to *Confidence* (shown in Eq. (6)) and *Support*, which measure the accuracy of the rule and the extent of knowledge of the rule respectively:

$$Support(R_j) = \frac{\sum_{x_p \in ClassC_j} \mu_{A_j}(x_p)}{N_{C^j}} \tag{9}$$

where N_{C^j} is the number of examples that belong to the same class that the one determined in the consequent of the rule. Therefore, the *raw_fitness* is computed in the following way:

$$raw\_fitness(R_j) = \alpha \cdot Confidence(R_j) + (1 - \alpha) \cdot Support(R_j) \tag{10}$$

Finally, it is important to point out that each time that an individual is evaluated it is also necessary to modify its certainty degree. On the other hand, the *global_fitness* score measure is defined as follows:

$$global\_fitness = w_1 \cdot accuracy + w_2 \cdot (1.0 - Var_N) + w_3 \cdot (1.0 - Cond_N) + w_4 \cdot (1.0 - Rul_N) \tag{11}$$

where Var_N and $Cond_N$ are the normalized values of the average number of variables and conditions in the rules, and Rul_N is the normalized number of rules in the population respectively.

The GP-COACH algorithm also includes a mechanism for maintaining the diversity in the population: the token competition procedure [55], inspired by the following natural behavior: when an individual finds a good place to live, it will maintain its position there preventing the others to share its position unless they are stronger. Each example in the training set is called a token and the rules in the population compete to acquire as many tokens as possible. When a rule matches an example, it tries to seize the token, however, this token will be assigned to the stronger rule that matches the example. Stronger individuals exploit their dominant position by seizing as many tokens as they can. The other ones entering the same position will have their strength decreased because they cannot compete with the stronger ones, by the addition of a penalization in the fitness score of the individual. Therefore, to model this behavior, a *penalized_function* is defined:

$$penalized\_fitness(R_j) = \begin{cases} raw\_fitness(R_j) \cdot \frac{count(R_j)}{ideal(R_j)} & \text{if } ideal(R_j) > 0, \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

where *raw_fitness*(R_j) is the fitness score obtained from the evaluation function (Eq. (10)), *count*(R_j) is the number of tokens that the individual actually seized and *ideal*(R_j) is the total number of tokens that it can seize, which is equal to the number of examples that the individual matches.

As a result of the token competition, there can be individuals that cannot grab any token. These individuals are considered as irrelevant, and they are eliminated from the population because all of their examples are covered by other stronger individuals.

Once the token competition mechanism has been applied, it is possible that some of the examples in the training set are not covered by any of the rules in the population. The generation of new specific rules covering these examples improves the diversity in the population, and helps the evolutionary process to easily find stronger and more general rules covering these examples. Therefore, GP-COACH learns rule sets having two different types of fuzzy rules: a core of strong and general rules (primary rules) that cover most of the examples, and a small set of weaker and more specific rules (secondary rules) that are only used if there are not any primary rule matching the example. These secondary rules are generated by the Chi et al. algorithm [10] over the set of training examples that are left uncovered by the primary rules. This scaly scheme is used in rule based algorithms to cover in a better way the data space [52]. GP-COACH uses four different genetic operators to generate new individuals during the evolutionary process:

1. *Crossover*: A part in the first parent is randomly selected and exchanged by another part, randomly selected, in the second one.
2. *Mutation*: It is applied to a variable in the rule randomly chosen. The mutation can add a new label to the label set associated to the variable, remove a label from the label set associated to the variable or exchange one label in the label set associated to the variable with another one not included.
3. *Insertion*: It adds a new variable to the parent rule with at least one linguistic label.
4. *Dropping condition*: It selects one variable and removes its conditions from the rule.

These operations only generate one offspring each time they are applied.

Fig. 5 shows the pseudocode associated to the GP-COACH algorithm. This method begins creating a random initial population according to the rules in the context-free grammar. Each individual in this population is then evaluated. After that, the initial population is kept as the best evolved population and its global fitness score is computed. Then, the initial population is copied to the current population and the evolutionary process begins:

1. An offspring population, with the same size than the current one, is created. Parents are selected by using the tournament selection mechanism and children are created by using one of the four genetic operators. The genetic operator selection is done in a probabilistic way according to a given probability.
2. Once the offspring population is created, it is joined to the current population, creating a new population whose size is double the current population size. Individuals in this new population are sorted according to their fitness and the token competition mechanism is applied. Secondary rules are created if some examples remain uncovered.

```
CurrentPop = InitPopulation();
EvaluatePopRules (CurrentPop);
CurrentFitness = EvaluatePopulation (CurrentPop);
BestPop = CurrentPop;
BestFitness = CurrentFitness;
$N = 0$;
while $N < N_{eval}$ do
    OffspringsPop = $\emptyset$;
    while OffspringsPop.size() $\neq$ CurrentPop.size() do
        ParentRule = tournamentSelection (CurrentPop);
        GeneratedRule = geneticOperator (ParentRule, CurrentPop);
        OffspringsPop.add(GeneratedRule);
    end
    EvaluatePopRules (OffspringsPop);
    JointPop = CurrentPop $\cup$ OffspringsPop;
    JointPop = tokenCompetition (JointPop);
    CurrentPop = JointPop $\cup$ generateSecondaryRules (JointPop);
    CurrentFitness = EvaluatePopulation (CurrentPop);
    if CurrentFitness > BestFitness then
        BestPop = CurrentPop;
        BestFitness = CurrentFitness;
    end
    $N$++;
end
Output: BestPop
```

Fig. 5. The GP-COACH algorithm.

3. The global fitness score measure is then calculated for this new population. We check whether this new fitness is better than the one stored for the best population, updating the best population and fitness if necessary. In any case, the new population is copied as the current population in order to be able to apply the evolutionary process again.

The evolutionary process ends when the stop condition is verified, that is when a number of evaluations is reached. Then, the population kept as the best one is returned as the solution to the problem and GP-COACH finishes.

3.3. Hierarchical fuzzy rule based classification system construction

HFRBCs try to improve the performance of fuzzy rule based systems in data subspaces that are particularly difficult. In order to do so, instead of the classical definition of the KB, we use an extension known as HKB [14], which is composed of a set of layers. We will divide this subsection in two parts: the first part is devoted to the presentation of the HKB, its components and some general guidelines about how to build it; the second part is devoted to the integration of the HKB into the inner way of running of the GP-COACH algorithm which we have used as base for our proposal.

3.3.1. Hierarchical knowledge base

In order to overcome the inflexibility of the concept of linguistic variable which degrades the performance of algorithms in complex search spaces, we extend the definition of the standard KB into an hierarchical one that preserves the original model descriptive power and increases its accuracy. This HKB is composed of a set of layers. We define a layer by its components in the following way:

$$layer(t, n(t)) = DB(t, n(t)) + RB(t, n(t)) \tag{13}$$

with $n(t)$ being the number of linguistic terms that compose the partitions of layer, $DB(t,n(t))$ (*t-linguistic partitions*) being the DB which contains the linguistic partitions with granularity level $n(t)$ of layer, and $RB(t,n(t))$ (*t-linguistic rules*) being the RB formed by those linguistic rules whose linguistic variables take values in the former partitions. The number of linguistic terms in the *t-linguistic partitions* is defined in the following way:

$$n(t) = (n(1) - 1) \cdot 2^{t-1} + 1 \tag{14}$$

with $n(1)$ being the granularity of the initial fuzzy partitions.

This set of layers is organized as a hierarchy, where the order is given by the granularity level of the linguistic partition defined in each layer. That is, given two successive layers t and $t + 1$ then the granularity level of the linguistic partitions of layer $t + 1$ is greater than the ones of layer t. This causes a refinement of the previous layer linguistic partitions. As a consequence of the previous definitions, we can now define the HKB as the union of every layer t

$$HKB = \bigcup_t layer(t, n(t)) \tag{15}$$

Our proposal considers a two-layer HKB, i.e. starting with an initial layer t, we produce layer $t + 1$ in order to extract the final system of linguistic rules. This fact allows the approach to build a significantly more accurate modeling of the problem space.

First of all, we need to build the two-layer HDB. The first level layer is built by the usage of *linguistic partitions* with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions. The second layer, is built preserving all the membership function modal points, corresponding to each linguistic term, through the higher layers of the hierarchy and adding a new linguistic term between each two consecutive terms of the *t-linguistic partition* reducing the support of these linguistic terms in order to keep place for the new one, which is located in the middle of them. Fig. 6 shows the *linguistic partitions* from one level to another, with $n(1) = 3$ and $n(2) = 5$.

The second step affects the generation of the HRB which is composed by the RB of layer t and a RB of layer $t + 1$. Two measures of error are usually used to build a RB of layer $t + 1$ from a layer RB of layer t: a global measure, which is used to evaluate the complete RB, and a local measure, used to determine the goodness of the rules. We calculate these measures similarly to other HFRBCS methodologies focused on classification problems [21]. The global measure used is the accuracy per class, computed as:

$$Acc_i(X_i, RB) = \frac{|x_p \in X_i / FRM(x_p, RB) = Class(x_p)|}{|X_i|} \tag{16}$$

where $||$ is the number of patterns, X_i is the set of examples of the training set that belong to the ith class, $FRM(x_p,RB)$ is the class prediction of the pattern using the rules in the RB with the FRM used by the GP-COACH algorithm, and $Class(x_p)$ is the class label for example x_p. The local measure utilized is the accuracy for a rule, computed over the whole training set as

$$Acc(X, R_j) = \frac{|X^+(R_j)|}{|X(R_j)|} \tag{17}$$

It is important to remind that since we are using the normalized sum approach as FRM, $X^+(R_j)$ and $X(R_j)$ are defined as

- $X(R_j)$ is the set of examples that have a matching degree with the rule higher than 0 where this compatibility has contributed to classify the sample as the class label of the rule.
- $X^+(R_j)$ is the set of examples that have a matching degree with the rule higher than 0 where this compatibility has contributed to classify the sample as the class label of the rule and where the predicted class corresponds with the class label of the example.

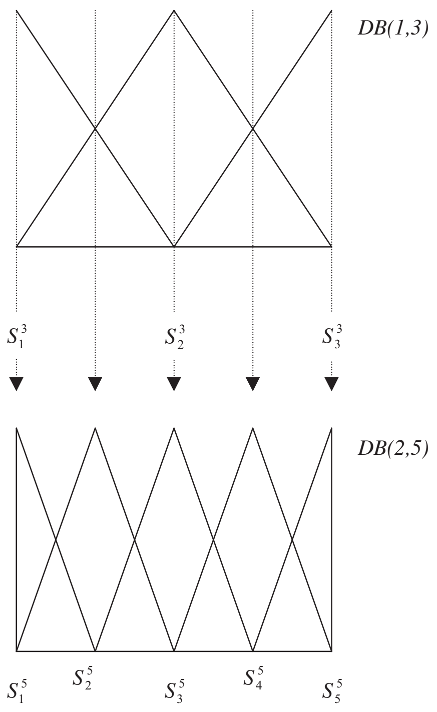

Fig. 6. Transition from a partition in DB(1,3) to another one in DB(2,5).

For each example in the training set, we obtain a set of rules that have contributed to the classification when we compute the global measure. Therefore, when we try to compute $X^{+}(R_j)$ and $X(R_j)$ we have for each rule the set of examples where the current rule has contributed to its classification.

Once we have computed the global measure and the local measure, we characterize the rules as *good* or *bad* according to the following calculation:

```
If $(Acc(X,R_j) \leqslant (1-\alpha) \cdot Acc_i(X_i,RB))$ Then
    $R_j = good\,rule$
Else
    $R_j = bad\,rule$
```

Good rules are kept in the rule population while *bad rules* are deleted from the current population. Then, new high granularity rules are created using as linguistic rule generator with the DB associated to layer $t + 1$ and adopting as training set for this task a subset of the original training set including examples that meets some specified conditions. If after the generation of these rules we find repeated rules we only keep one copy of them, or if we find contradictory rules (rules with the same antecedent but with different consequents) we maintain the rule with a higher rule weight in the RB while the others are removed.

3.3.2. Integration of a HKB in the GP-COACH algorithm

The usage of a HKB in the inner way of running of the GP-COACH algorithm induces some changes in its structure. For example, the existence of the HRB which is composed by the RB of layer t and a RB of layer $t + 1$ forces the GP-COACH algorithm to provide a mechanism to maintain these two RB levels. In our case, these RBs are merged and are evolved together in the different generations computed in the GP-COACH algorithm.

The rule population used in the algorithm is now a mixed population that combines primary rules and secondary rules where the secondary rules present different granularities. In this kind of population, genetic operators obtain rules according to the type of parent rule: primary rules obtain primary rules while secondary rules obtain secondary rules maintaining the granularity of the original rule. The only restriction in the application of the genetic operations appears in the usage of the crossover operation where the rules selected for the generation of a new rule must have the same granularity.

The global fitness score is modified to consider the different granularities of the rules in the population. The new global fitness function is:

$$global\_fitness = w_1 \cdot accuracy + w_2 \cdot (1.0 - Var_N) + w_3 \cdot \left(1.0 - \frac{(Cond\_Low_N \cdot R\_Low + Cond\_High_N \cdot R\_High)}{R}\right) + w_4 \cdot (1.0 - Rul_N) \tag{18}$$

where Var_N is the normalized average number of variables, $Cond\_Low_N$ is the normalized average number of conditions in low granularity rules, $Cond\_High_N$ is the normalized average number of conditions in high granularity rules, Rul_N is the normalized number of rules and R_Low, R_High, R are the number of low granularity rules, high granularity rules and total number of rules respectively.

To generate the high granularity rules some additional steps are performed just after the final step of a GP-COACH generation which is the construction of secondary rules for examples that have not been covered with the current rule base. This process is done performing the following operations:

1. The rules that compose the rule set are classified as *good rules* or *bad rules* as explained in the previous subsection.
2. *Good rules* are kept in the rule population and *bad rules* are directly deleted.
3. New high granularity rules are created using as linguistic rule generator the Chi et al. algorithm [10] with the DB associated to layer $t + 1$ and adopting as training set for this task the examples that were classified by the rules that were considered *bad rules*.
4. Repeated and contradictory rules are searched for and only one copy of the best performing is kept.

Usually, when creating a hierarchical rule base, another step is added to improve the performance of the final model: a hierarchical rule selection step. In our case, since the hierarchical expansion of rules is embedded into each generation of the GP-COACH algorithm, adding a genetic selection process would increase considerably the run time of the approach. Therefore, this rule selection step is appended after the GP-COACH generations end combined with a tuning step to take advantage of the synergy between these refinements of the KB. Furthermore, GP-COACH tries to obtain a compact rule population with the token competition procedure making thus this delay of the rule selection step possible.

3.4. Hierarchical rule base selection and lateral tuning

In this last step, we analyze the use of genetic algorithms to select and tune a compact and cooperative set of fuzzy rules that obtain a high performance starting from the hierarchical rules generated in the previous step. In order to do so, we consider the approach used by Alcalá et al. [1] that uses the linguistic 2-tuples representation [32]. This representation allows the lateral displacement of the labels considering only one parameter (symbolic translation parameter), which involves a simplification of the tuning search space that aids the obtaining of optimal models. Particularly this happens when it is combined with a rule selection within the same process enabling it to take advantage of the positive synergy that both techniques present. In this way, this process for contextualizing the membership functions permits them to achieve a better covering degree while maintaining the original shapes, which results in accuracy improvements without a significant loss in the interpretability of the fuzzy labels. The symbolic translation parameter of a linguistic term is a number within the interval $[-0.5,0.5)$ that expresses the domain of a label when it is moving between its two lateral labels. Let us consider a set of labels S representing a fuzzy partition. Formally, we have the pair, $(s_i,\alpha_i), s_i \in S, \alpha_i \in [-0.5,0.5)$. An example is illustrated in Fig. 7 where we show the symbolic translation of a label represented by the pair $(S_2, -0.3)$.

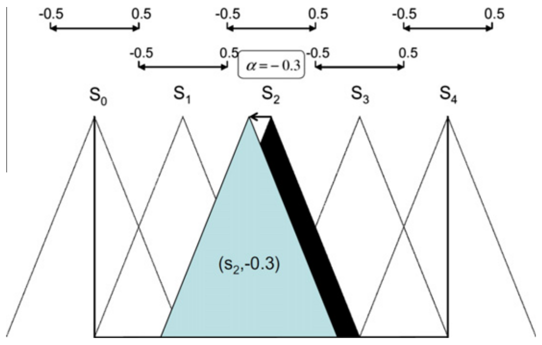

Fig. 7. Lateral displacement of a MF.

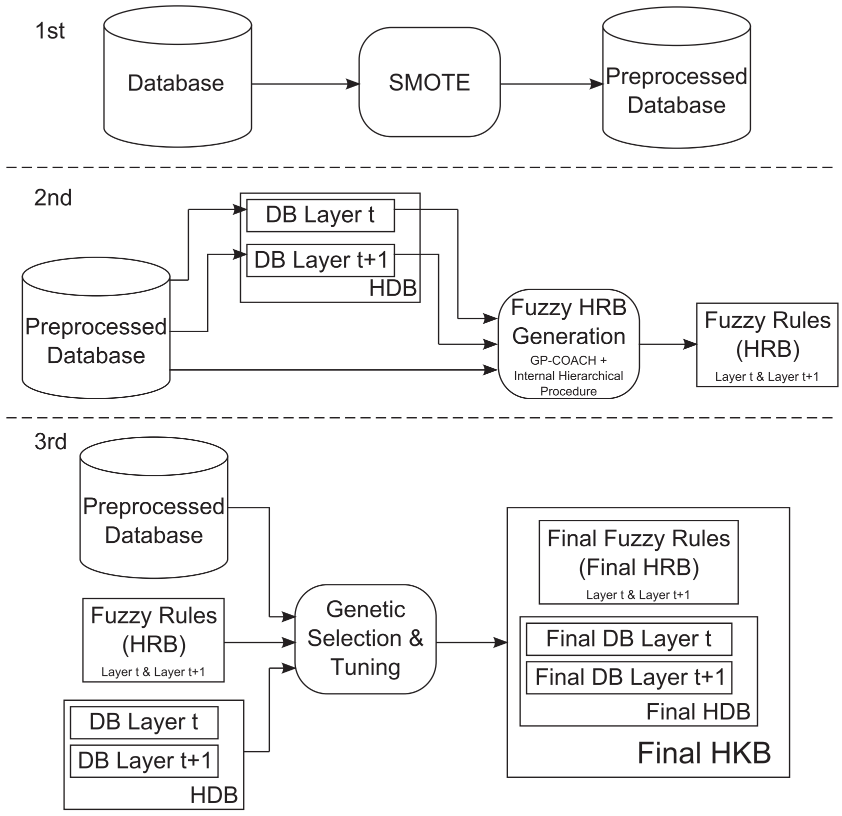

Fig. 8. Flowchart of GP-COACH-H.

Alcalá et al. [1] proposed two different rule representation approaches, a global approach and a local approach. In our algorithm, the tuning is applied to the level of linguistic partitions (global approach). In this way, the pair (X_i, label) takes the same tuning value in all the rules where it is considered. For example, X_1-is(High, 0.3) will present the same value for those rules in which the pair "X_1 is High" was initially considered. This proposal decreases the tuning problem complexity, greatly easing the derivation of optimal models.

To accomplish this rule selection and lateral tuning process, we consider the use of a specific genetic algorithm, the CHC evolutionary algorithm [18] with the same scheme described in our previous works [21,22]. In the remainder of this section, we describe the specific features of our new tuning approach, which involves the codification of the solutions and initial gene pool, chromosome evaluation, crossover operator and restarting approach.

1. *Codification and Initial Gene Pool:* To combine the rule selection with the global lateral tuning, a double coding scheme for both rule selection (C_S) and lateral tuning (C_T) is used:
   - For the C_S part, each chromosome is a binary vector that determines when a rule is selected or not (alleles '1' and '0' respectively). Considering the M rules contained in the candidate rule set (rules from the two hierarchical levels considered), the corresponding part $C_S = \{c_1, \ldots, c_M\}$ represents a subset of rules composing the final rule base, so that, $If\, c_j = 1\, then (R_j \in RB)\, else (R_j \notin RB)$, with R_j being the corresponding jth rule in the candidate rule set and RB being the final RB.
   - For the C_T part, a real coding is considered. This part is the joint of the α parameters of each fuzzy partition. Let us consider the following number of labels per variable: (ml^1, ml^2, $\ldots$, ml^n) for low granularity rules and (mh^1, $mh^2, \ldots, mh^n$) for high granularity rules, with n being the number of system variables. Then, this part has the following form (where each gene is associated to the tuning value of the corresponding label): $C_T = (cl_{11}, \ldots, cl_{1ml^1}, cl_{21}, \ldots, cl_{2ml^2}, \ldots, cl_{n1}, \ldots, cl_{nml^n}, ch_{11}, \ldots, ch_{1mh^1}, ch_{21}, \ldots, ch_{2mh^2}, \ldots, ch_{n1}, \ldots, ch_{nmh^n})$.

   Finally, a chromosome C is coded in the following way: $C = C_S C_T$. To make use of the available information, all the candidate rules are included in the population as an initial solution. To do this, the initial pool is obtained with the first individual having all genes with value '1' in the C_S part and all genes with value '0.0' in the C_T part. The remaining individuals are generated at random.
2. *Chromosome Evaluation:* To evaluate a determined chromosome we compute its accuracy over the training set. If two individuals obtain the same value, then the individual with the lower number of selected rules is preferred.
3. *Crossover Operator:* The crossover operator will depend on the chromosome part where it is applied:
   - In the C_S part, the half uniform crossover scheme (HUX) is employed.
   - For the C_T part, we consider the Parent Centric BLX (PCBLX) operator [31], which is based on BLX-α.
4. *Restarting Approach:* To get away from local optima, this algorithm uses a restart approach that is performed to improve the diversity of the population that may be reduced by the strong elitist pressure of the replacement scheme.

For details about the remainder features of the optimization process, please refer to Fernández et al. [21] and Fernández et al. [22].

3.5. Summary of the GP-COACH-H algorithm

Once every step of the algorithm has been explained we briefly sum up how the GP-COACH-H algorithm works. Fig. 8 depicts a flowchart of the GP-COACH-H algorithm.

Table 2
Parameter specification for the algorithms tested in the experimentation.

Algorithm	Parameters
FRBCS parameters	
GP-COACH and GP-COACH-H	Minimum t-norm, Maximum t-conorm, Rule Weight = Certainty Factor, Fuzzy Reasoning Method = Normalized Sum, Number of Fuzzy Labels (for basic GP-COACH) = 5 or 9, Number of Fuzzy Labels (for GP-COACH-H) = 5 for Low Granularity Rules and 9 for High Granularity Rules
HFRBCS(Chi)	Product t-norm, Rule Weight = Penalized Certainty Factor, Fuzzy Reasoning Method = Winning Rule, Number of Fuzzy Labels = 3 for Low Granularity Rules and 5 for High Granularity Rules
GP-COACH parameters	
GP-COACH and GP-COACH-H	Evaluations = 20000, Initial Population Size = 200, α (raw fitness) = 0.7, Crossover Probability = 0.5, Mutation Probability = 0.2, Dropping Condition Probability = 0.15, Insertion Probability = 0.15, Tournament size = 2, $w_1 = 0.8$, $w_2 = w_3 = 0.05$, $w_4 = 0.1$
Hierarchical procedure parameters	
GP-COACH-H and HFRBCS(Chi)	α (rule expansion) = 0.2, CHC Evaluations = 10,000, CHC Population Size = 61, CHC bits per gene (for GP-COACH-H) = 30
C4.5 parameters	
C4.5	Pruned=true, Confidence = 0.25 and Minimum number of item-sets per leaf = 2

There are three different steps in the building of the model:

1. *Preprocessing stage:* In this first step, GP-COACH-H preprocesses the original data-set to balance the class distribution. In order to do so, the SMOTE algorithm is used, as described in subSection 2.2.
2. *Generation of the HKB:* This stage is devoted to the generation of a two-layer HKB from the balanced data-set. This HKB is composed by two different DBs (each one with a different granularity level) and one RB that contains rules from the two hierarchies:
   (a) *HDB Generation*: The first layer DB is created with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions. The second layer, is built preserving all the membership function modal points, corresponding to each linguistic term.
   (b) *HRB Generation:* In order to generate the HRB we use as a base the GP-COACH algorithm, which has been modified to incorporate in its internal way of running the creation of hierarchical rules. The adjustments reinforce the connection between the GP-COACH algorithm and the hierarchical methodology because they have been designed to get the greatest possible performance. Specifically, these modifications include:
      - A step to identify *good* and *bad* rules, where *bad* rules are deleted and the examples covered by them are used to create new high granularity rules.
      - Changes in the global fitness function considering the different granularities in the rule population.
      - A variation on the conditions of the application of the crossover operator where only rules with the same granularity level are allowed to produce an offspring.

   This HRB generation procedure uses the preprocessed data-set from the previous step and the membership functions defined by the HDB.
3. *Refinement of the HKB:* After the building of an initial HKB in the previous phase, another genetic procedure is applied to improve the final performance of this solution. In this step, rules that cooperate properly in the population are selected and the HDB is tuned with the 2-tuples linguistic representation. These optimizations are done in a single step to take advantage of the synergy that both techniques can achieve. The set of selected rules define the final HRB given as solution and the tuning parameters obtained modify the original HDB to create the final HDB which is the output of the algorithm.

4. Experimental framework

In this section, we present the set up of the experimental framework used to develop the analysis of our proposal. First we introduce the algorithms selected for the comparison with the proposed approach and their configuration parameters (subSection 4.1). Next, we provide details of the problems chosen for the experimentation (subSection 4.2). Finally, we present the statistical tests applied to compare the results obtained with the different classifiers (subSection 4.3).

4.1. Algorithms selected for the study and parameters

In order to test the performance of our approach, GP-COACH-H, several classification methods have been selected to perform the experimental study. These methods are:

- *GP-COACH* [7]: The original FRBCS that was used as base for our approach, a GP-based algorithm for the learning of compact and interpretable fuzzy rule bases that obtains good accuracy in high dimensional classification problems.
- *HFRBCS(Chi)* [21]: This approach obtains a Hierarchical Fuzzy Rule Base Classification System (HFRBCS) using the Chi et al. algorithm [10] as the linguistic rule generation method and has reported good results in imbalanced data-sets.
- *C4.5* [47]: A well-known decision tree which has shown a good behavior in the framework of imbalanced data-sets [6].

The configuration parameters used for these algorithms are shown in Table 2. All the methods were run using KEEL software[2] [3], following the default parameter values given in the KEEL platform to configure the methods, which were selected according to the recommendation of the corresponding authors of each algorithm, assuming that the choice of the values of the parameters was optimal.

Regarding the use of the SMOTE [9] and SMOTE+ENN [6] preprocessing methods, we consider only the 1-nearest neighbor (using the euclidean distance) to generate the synthetic samples, and we balance the training data to the 50% distribution. We only use SMOTE + ENN for C4.5 because it shows a positive synergy when pruning the tree [16].

4.2. Data-sets and data partitions

In order to analyze the quality of our approach GP-COACH-H against the algorithms introduced in the previous section, we have

[2] http://www.keel.es/.

Table 3
Summary of imbalanced data-sets.

Data-sets	#Ex.	#Atts.	Class (−; +)	%Class (−; +)	IR
ecoli034vs5	200	7	(p, imL, imU; om)	(10.00, 90.00)	9.00
yeast2vs4	514	8	(cyt; me2)	(9.92, 90.08)	9.08
ecoli067vs35	222	7	(cp, omL, pp; imL, om)	(9.91, 90.09)	9.09
ecoli0234vs5	202	7	(cp, imS, imL, imU; om)	(9.90, 90.10)	9.10
glass015vs2	172	9	(build-win-non_float-proc, tableware, build-win-float-proc; ve-win-float-proc)	(9.88, 90.12)	9.12
yeast0359vs78	506	8	(mit, me1, me3, erl; vac, pox)	(9.88, 90.12)	9.12
yeast02579vs368	1004	8	(mit, cyt, me3, vac, erl; me1, exc, pox)	(9.86, 90.14)	9.14
yeast0256vs3789	1004	8	(mit, cyt, me3, exc; me1, vac, pox, erl)	(9.86, 90.14)	9.14
ecoli046vs5	203	6	(cp, imU, omL; om)	(9.85, 90.15)	9.15
ecoli01vs235	244	7	(cp, im; imS, imL, om)	(9.83, 90.17)	9.17
ecoli0267vs35	224	7	(cp, imS, omL, pp; imL, om)	(9.82, 90.18)	9.18
glass04vs5	92	9	(build-win-float-proc, containers; tableware)	(9.78, 90.22)	9.22
ecoli0346vs5	205	7	(cp, imL, imU, omL; om)	(9.76, 90.24)	9.25
ecoli0347vs56	257	7	(cp, imL, imU, pp; om, omL)	(9.73, 90.27)	9.28
yeast05679vs4	528	8	(me2; mit, me3, exc, vac, erl)	(9.66, 90.34)	9.35
ecoli067vs5	220	6	(cp, omL, pp; om)	(9.09, 90.91)	10.00
vowel0	988	13	(hid; remainder)	(9.01, 90.99)	10.10
glass016vs2	192	9	(ve-win-float-proc; build-win-float-proc, build-win-non_float-proc,headlamps)	(8.89, 91.11)	10.29
glass2	214	9	(Ve-win-float-proc; remainder)	(8.78, 91.22)	10.39
ecoli0147vs2356	336	7	(cp, im, imU, pp; imS, imL, om, omL)	(8.63, 91.37)	10.59
led7digit02456789vs1	443	7	(0, 2, 4, 5, 6, 7, 8, 9; 1)	(8.35, 91.65)	10.97
glass06vs5	108	9	(build-win-float-proc, headlamps; tableware)	(8.33, 91.67)	11.00
ecoli01vs5	240	6	(cp, im; om)	(8.33, 91.67)	11.00
glass0146vs2	205	9	(build-win-float-proc, containers, headlamps, build-win-non_float-proc;ve-win-float-proc)	(8.29, 91.71)	11.06
ecoli0147vs56	332	6	(cp, im, imU, pp; om, omL)	(7.53, 92.47)	12.28
cleveland0vs4	177	13	(0; 4)	(7.34, 92.66)	12.62
ecoli0146vs5	280	6	(cp, im, imU, omL; om)	(7.14, 92.86)	13.00
ecoli4	336	7	(om; remainder)	(6.74, 93.26)	13.84
yeast1vs7	459	8	(nuc; vac)	(6.72, 93.28)	13.87
shuttle0vs4	1829	9	(Rad Flow; Bypass)	(6.72, 93.28)	13.87
glass4	214	9	(containers; remainder)	(6.07, 93.93)	15.47
page-blocks13vs2	472	10	(graphic; horiz.line, picture)	(5.93, 94.07)	15.85
abalone9vs18	731	8	(18; 9)	(5.65, 94.25)	16.68
glass016vs5	184	9	(tableware; build-win-float-proc, build-win-non_float-proc,headlamps)	(4.89, 95.11)	19.44
shuttle2vs4	129	9	(Fpv Open; Bypass)	(4.65, 95.35)	20.5
yeast1458vs7	693	8	(vac; nuc, me2, me3, pox)	(4.33, 95.67)	22.10
glass5	214	9	(tableware; remainder)	(4.20, 95.80)	22.81
yeast2vs8	482	8	(pox; cyt)	(4.15, 95.85)	23.10
yeast4	1484	8	(me2; remainder)	(3.43, 96.57)	28.41
yeast1289vs7	947	8	(vac; nuc, cyt, pox, erl)	(3.17, 96.83)	30.56
yeast5	1484	8	(me1; remainder)	(2.96, 97.04)	32.78
ecoli0137vs26	281	7	(pp, imL; cp, im, imU, imS)	(2.49, 97.51)	39.15
yeast6	1484	8	(exc; remainder)	(2.49, 97.51)	39.15
abalone19	4174	8	(19; remainder)	(0.77, 99.23)	128.87

selected several highly imbalanced and borderline imbalanced data-sets.

Specifically, as highly imbalanced data-sets, we have selected 44 data-sets from KEEL data-set repository[3] [2] with an imbalance ratio (IR) [46] greater than 9. The data are summarized in Table 3, where we denote the number of examples (#Ex.), number of attributes (#Atts.), class name of each class (positive and negative), class attribute distribution and IR. This table is in ascending order according to the IR.

Inspired by Kubat and Matwin [40], Napierala et al. [44] created several artificial data-sets that contain borderline examples in an imbalanced scenario to address the correct identification of those examples. These data-sets have three different shapes of the positive class: *subclus* (Fig. 9), *clover* (Fig. 10) and *paw* (Fig. 11), all surrounded uniformly by the negative class. For each shape, we have data-sets from two different sizes and IR: data-sets with 600 examples with an IR of 5 and data-sets with 800 examples with an IR of 7. Each one of these data-sets is affected by different disturbance ratio levels (0%, 30%, 50%, 60% and 70%). The disturbance ratio is simulated increasing the ratio of borderline examples from the positive class subregions.

To develop the different experiments we consider a *5-fold cross-validation model*, i.e., five random partitions of data with a 20% and the combination of 4 of them (80%) as training and the remaining ones as test. For each data-set we consider the average results of the five partitions. The data-sets used in this study use the partitions provided by the KEEL data-set repository in the imbalanced classification data-set section.[4]

4.3. Statistical tests for performance comparison

Statistical analysis needs to be carried out in order to find significant differences among the results obtained by the studied methods [24]. We consider the use of non-parametric tests, according to the recommendations made in [25,24] where a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers is presented. These tests are used due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility [50].

The Wilcoxon test [50] will be used as a non-parametric statistical procedure in order to conduct pairwise comparisons between two algorithms. For multiple comparisons we use the Iman-Davenport

[3] http://www.keel.es/datasets.php.

[4] http://www.keel.es/imbalanced.php.

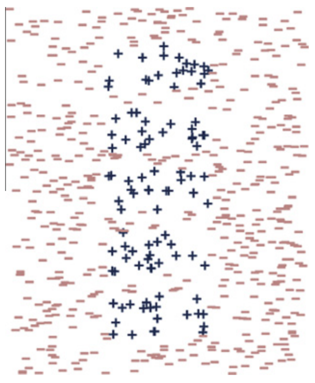
Fig. 9. Subclus.

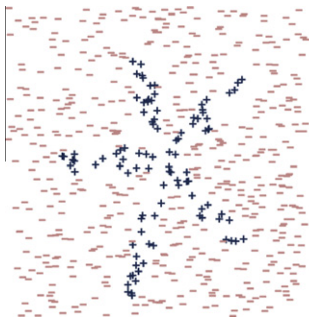
Fig. 10. Clover.

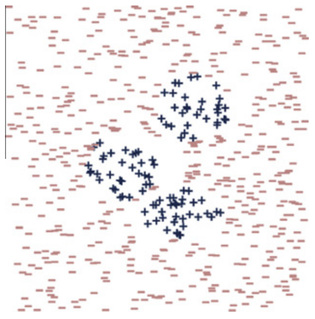
Fig. 11. Paw.

test to detect statistical differences among a group of results, and the Holm post-hoc test in order to find which algorithms are distinctive among a $1 \times n$ comparison.

The post-hoc procedure allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance α. However, it is very interesting to compute the p-value associated with each comparison, which represents the lowest level of significance of a hypothesis that results in a rejection. It is the adjusted p-value. In this manner, we can know whether two algorithms are significantly different and how different they are.

Furthermore, we consider the average ranking of the algorithms, in order to show how good a method is with respect to its partners. This ranking is obtained by assigning a position to each algorithm depending on its performance for each data-set. The algorithm which achieves the best accuracy in a specific data-set will have the first ranking (value 1); then, the algorithm with the second best accuracy is assigned rank 2, and so forth. This task is carried out for all data-sets and finally an average ranking is computed as the mean value of all rankings.

These tests are suggested in the studies presented in [25,24], where their use in the field of machine learning is highly recommended. For a wider description of the use of these tests, please refer to the website on *Statistical Inference in Computational Intelligence and Data Mining*.[5]

5. Experimental study

In this section, we present a set of experiments to illustrate and demonstrate the behavior of GP-COACH-H. These experiments are designed towards two objectives: to exemplify how the GP-COACH-H algorithm works, and to determine its robustness for highly and borderline imbalanced data-sets.

We organize those experiments in the following way. First, Section 5.1 presents a case of study over one one of the highly imbalanced data-sets presented in the previous section. Next, Section 5.2 contains an analysis of the impact of the hierarchical step in the algorithm. Section 5.3 studies the the importance of the usage of a preprocessing step when dealing with highly imbalanced data-sets. Later, Section 5.4 performs a global comparison among the fuzzy classification methods and C4.5 over the highly imbalanced data-sets. Finally, in Section 5.5, this global comparison is also carried out over the borderline imbalanced data-sets.

5.1. Sample procedure of the GP-COACH-H algorithm: a case of study

In order to illustrate how GP-COACH-H works we have selected the *glass0146vs2* data-set. We will follow the algorithm operations and the results it provides. The *glass0146vs2* data-set is a highly imbalanced data-set from the KEEL data-set repository,[6] with 9 input attributes, 205 instances and an IR equal to 11.06. We have selected this data-set as one with a small size whose results can be easily interpreted.

For this specific run, we have chosen the 3rd partition from the 5-fcv used in all the experiments. This partition uses 164 instances for training (14 positive and 150 negative) and 41 for test (3 positive and 38 negative), using the 9 input attributes of the whole data-set. The first step of the GP-COACH-H algorithm (see Fig. 8) uses the SMOTE algorithm to balance the class distribution. Therefore, we apply the SMOTE algorithm and we obtain a new training set that contains 300 instances, 150 instances for each class.

The second step starts using the preprocessed data-set to generate the HKB. In order to generate the HKB, we first generate the HDB from the available data. The HDB is generated (as was explained in the previous sections) with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions. The second layer, is built preserving all the membership function modal points, corresponding to each linguistic term. Figs. 12 and 13 show the linguistic variables generated for the *Mg* attribute, according to the given instructions.

Once we have generated the HDB, we start the GP procedure to generate the HRB. This procedure evolves a rule population through several generations, including the usage of genetic operators to generate new individuals, the token competition procedure to delete irrelevant rules and the hierarchical creation of new rules in each step. At the end of the iterations, a rule base with different granularity rules is obtained. In Fig. 14, the rules generated using the generated HDB and the preprocessed training set are shown.

At this point, we start the last step of the algorithm which is the genetic rule selection and lateral tuning of the variables. To obtain the final solution, we use the preprocessed set from the first step and the HKB generated previously. The genetic search looks for a

[5] http://sci2s.ugr.es/sicidm/.
[6] http://www.keel.es/imbalanced.php.

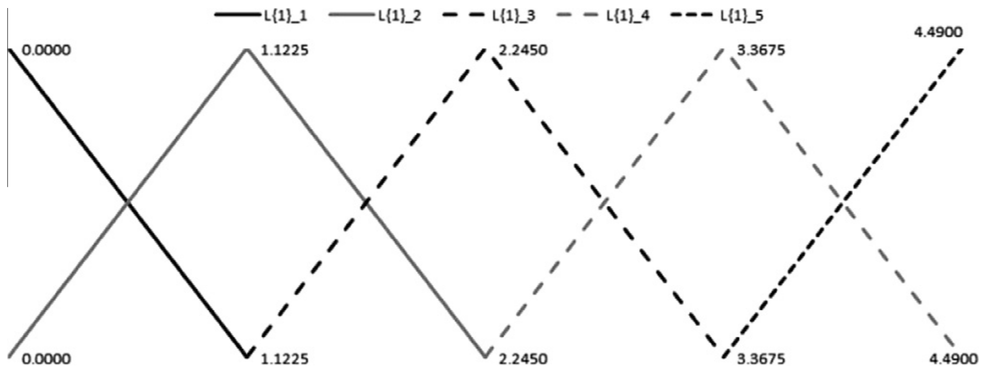

Fig. 12. Database Layer 1 with 5 labels, M_g attribute.

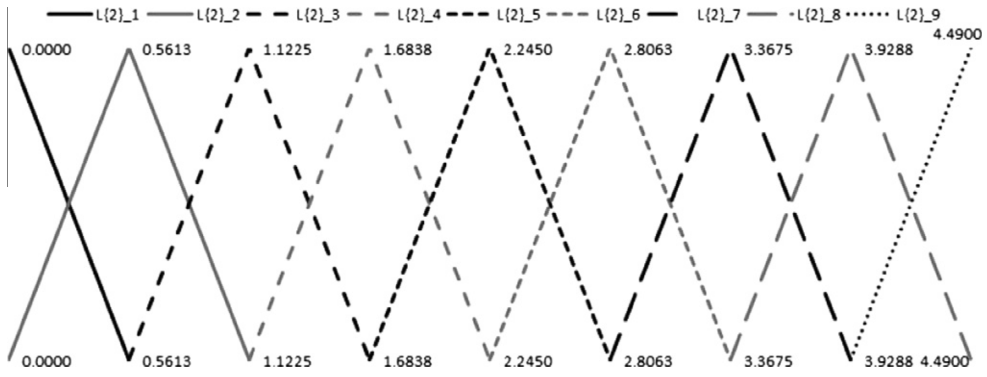

Fig. 13. Database Layer 2 with 9 labels, M_g attribute.

```
@Number of rules: 13

1{1}: IF Mg IS (L{1}_1 OR L{1}_2 OR L{1}_3) THEN negative with RW: .9969
2{1}: IF Ca IS (L{1}_1 OR L{1}_4 OR L{1}_5) THEN negative with RW: .9999
3{1}: IF RI IS (L{1}_4 OR L{1}_5) THEN negative with RW: 1.0
4{1}: IF Al IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Fe IS (L{1}_3 OR L{1}_5) THEN negative with RW: 1.0
5{1}: IF Al IS L{1}_3 AND Si IS (L{1}_1 OR L{1}_2) AND Fe IS (L{1}_2 OR L{1}_5) THEN negative with RW: 1.0
6{1}: IF Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Si IS (L{1}_1 OR L{1}_2 OR L{1}_5) THEN negative with RW: .9969
7{1}: IF RI IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Al IS L{1}_3 AND Fe IS (L{1}_2 OR L{1}_5) THEN negative with RW:
      .9904
8{1}: IF RI IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Fe IS (L{1}_2 OR L{1}_5)
      THEN negative with RW: .9590
9{2}: IF RI IS L{1}_3 AND Na IS L{1}_4 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND
      Ca IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_2 THEN positive with RW: .8156
10{2}: IF RI IS L{1}_2 AND Na IS L{1}_3 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND
       Ca IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_1 THEN positive with RW: .6675
11{2}: IF RI IS L{2}_3 AND Na IS L{2}_5 AND Mg IS L{2}_7 AND Al IS L{2}_2 AND Si IS L{2}_6 AND K IS L{2}_1 AND
       Ca IS L{2}_3 AND Ba IS L{2}_1 AND Fe IS L{2}_1 THEN positive with RW: .9654
12{2}: IF RI IS L{2}_4 AND Na IS L{2}_5 AND Mg IS L{2}_7 AND Al IS L{2}_2 AND Si IS L{2}_5 AND K IS L{2}_1 AND
       Ca IS L{2}_4 AND Ba IS L{2}_1 AND Fe IS L{2}_2 THEN positive with RW: .7443
13{2}: IF RI IS L{2}_5 AND Na IS L{2}_7 AND Mg IS L{2}_1 AND Al IS L{2}_3 AND Si IS L{2}_5 AND K IS L{2}_1 AND
       Ca IS L{2}_6 AND Ba IS L{2}_1 AND Fe IS L{2}_2 THEN negative with RW: 1.0
```

Fig. 14. Rules generated after the Fuzzy HRB Generation.

new HKB that better represents the data. Figs. 15–17 show the new HDB and HRB obtained, which are the final output of the GP-COACH-H algorithm.

5.2. Analysis of the impact of the hierarchical levels over the imbalanced data-sets

This subsection is devoted to the impact of the usage of the HKB in the GP-COACH-H algorithm in relation to not using a HKB and use a traditional KB instead. In this manner, we will detect the influence of this component of the GP-COACH-H algorithm thus justifying its use.

We will compare the results of the GP-COACH-H algorithm according to the fuzzy HKB generated after the application of the GP procedure to the results of the basic GP-COACH algorithm with 5 and 9 labels, using SMOTE as preprocessing algorithm in both cases. The performance measures used are sensitivity and specificity to observe the impact for each class. Table 4 shows the average results for each algorithm over the highly imbalanced data-sets. The complete table of results for all data-sets can be found in the appendix of this work.

Considering the sensitivity measure the best performing average algorithm is the basic GP-COACH with 5 labels, however, if we look at the specificity measure then the best performing algorithm is the basic GP-COACH with 9 labels. Therefore, we need to consider the effectiveness for each class separately.

Contemplating the positive class, we can observe that the best performance in training is higher for the hierarchical version, being

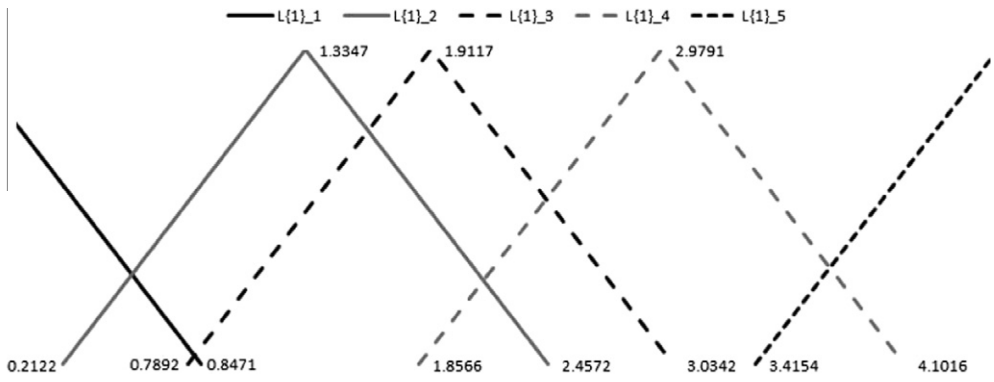

Fig. 15. Final database Layer 1 with 5 labels, M_g attribute.

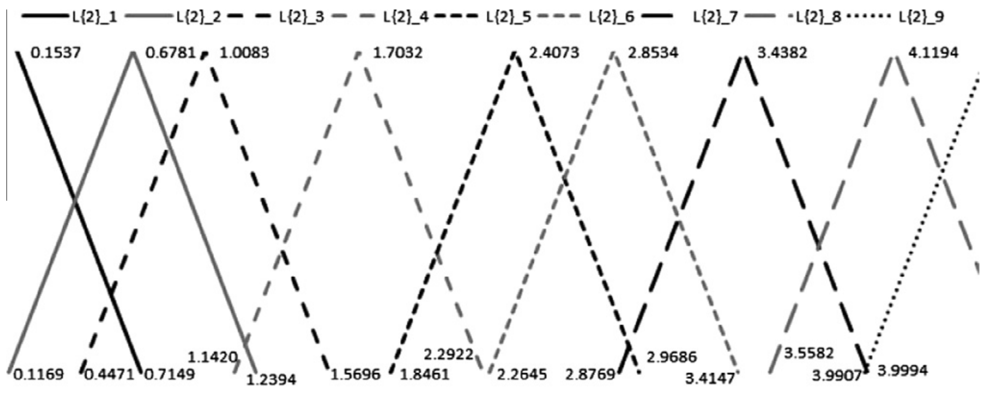

Fig. 16. Final database Layer 2 with 9 labels, M_g attribute.

Table 4
Average results for GP-COACH-5, GP-COACH-9 and GP-COACH-H for the highly imbalanced data-sets.

Data-set	Sensitivity$_{tr}$	Sensitivity$_{tst}$	Specificity$_{tr}$	Specificity$_{tst}$
GP-COACH-5	.9097 ± .0307	**.7809 ± .1212**	.8643 ± .0307	.8531 ± .1212
GP-COACH-9	.8983 ± .0267	.7319 ± .1334	.9231 ± .0267	**.9055 ± .1334**
GP-COACH-H	.9398 ± .0204	.7797 ± .1233	.9025 ± .0204	.8855 ± .1233

Table 5
Average results for GP-COACH versions with and without SMOTE preprocessing for the highly imbalanced data-sets.

Data-set	No preprocessing		SMOTE preprocessing	
	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$
GP-COACH-5	.4789 ± .1017	.3677 ± .1922	.8763 ± .0307	**.7897 ± .1212**
GP-COACH-9	.5074 ± .0871	.3929 ± .1996	.9056 ± .0267	**.7845 ± .1334**
GP-COACH-H	.4536 ± .1216	.3439 ± .1697	.9576 ± .0121	**.8175 ± .1193**

able to describe the training set more accurately than in the presence of low granularity rules only. Therefore, our initial intuition where the HKB was able to better describe difficult data spaces is confirmed. Comparing the training results in relation to the test results we notice a drop in performance for all the algorithms where GP-COACH-5 gets the best results, GP-COACH-H obtains similar results to GP-COACH-5 and GP-COACH-9 accomplishes lower results than the other two.

Analyzing the results associated to the negative class, we see an almost opposite situation. For training results the GP-COACH-9 algorithm is the algorithm that best describes the data, a situation where GP-COACH-H is supposed to be found. Nevertheless, GP-COACH-H is designed to specifically deal with imbalanced data-sets concentrating on the positive class so is logical that it does not characterize the negative class as well as the previous case. Confronting the training results with the test results we find a drop in the performance on equal levels for each approaches. Therefore, GP-COACH-9 is the best performing algorithm for the negative class, closely followed by GP-COACH-H where GP-COACH-5 performance falls behind those two approaches.

After checking the performance in each class, we discover that the basic GP-COACH is a powerful tool to describe one of our classes depending on the number of labels used. Nevertheless, if we choose a specific number of labels to focus on one class the final performance is degraded in the other one. Consequently, the GP-COACH-H approach that combines low granularity and high granularity rules is able to address the description of both classes accordingly. Its performance does not exceed the results of the basic algorithm, however, it goes closely after them in each class. Furthermore, there is not a high decrease in performance for the class

```
@Number of rules: 10

1{1}: IF Mg IS (L{1}_1 OR L{1}_2 OR L{1}_3) THEN negative with RW: 1.0
2{1}: IF Ca IS (L{1}_1 OR L{1}_4 OR L{1}_5) THEN negative with RW: .9871
3{1}: IF RI IS (L{1}_4 OR L{1}_5) THEN negative with RW: .9981
4{1}: IF Al IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Fe IS (L{1}_3 OR L{1}_5) THEN negative with RW: .9892
5{1}: IF Al IS L{1}_3 AND Si IS (L{1}_1 OR L{1}_2) AND Fe IS (L{1}_2 OR L{1}_5) THEN negative with RW: .9902
6{1}: IF Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Si IS (L{1}_1 OR L{1}_2 OR L{1}_5) THEN negative with RW: 1.0
7{1}: IF RI IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Fe IS (L{1}_2 OR L{1}_5)
      THEN negative with RW: .9461
8{2}: IF RI IS L{1}_3 AND Na IS L{1}_4 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND Ca
      IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_2 THEN positive with RW: .6544
9{2}: IF RI IS L{1}_2 AND Na IS L{1}_3 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND Ca
      IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_1 THEN positive with RW: .6719
10{2}: IF RI IS L{2}_3 AND Na IS L{2}_5 AND Mg IS L{2}_7 AND Al IS L{2}_2 AND Si IS L{2}_6 AND K IS L{2}_1 AND Ca
       IS L{2}_3 AND Ba IS L{2}_1 AND Fe IS L{2}_1 THEN positive with RW: .9561
```

Fig. 17. Final rules generated with the GP-COACH-H algorithm.

Table 6
Average results for FRBCS methods and C4.5 for the highly imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE+ENN for C4.5.

Data-set	GM_{tr}	GM_{tst}
GP-COACH-5	.8763 ± .0307	.7897 ± .1212
GP-COACH-9	.9056 ± .0267	.7845 ± .1334
HFRBCS(Chi)	.9331 ± .0117	.7901 ± .1325
GP-COACH-H	.9576 ± .0121	**.8175 ± .1193**
C4.5	.9549 ± .0180	.7848 ± .1452

Table 7
Average rankings and adjusted *p*-values using Holm's post-hoc procedure for FRBCS methods and C4.5 adopting the GM measure for the highly imbalanced data-sets.

Algorithm	Average ranking	Adjusted *p*-value (Holm's test)
GP-COACH-H	2.4091	
GP-COACH-9	3.0227	0.0862
GP-COACH-5	3.0909	0.0862
C4.5	3.2045	0.0549
HFRBCS(Chi)	3.2727	0.0416

Table 8
Average results for FRBCS methods and C4.5 for the borderline imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE+ENN for C4.5.

Data-set	GM_{tr}	GM_{tst}
GP-COACH-5	.7899 ± .0218	.7630 ± .0578
GP-COACH-9	.8103 ± .0330	.7628 ± .0705
HFRBCS(Chi)	.8316 ± .0195	.7992 ± .0461
GP-COACH-H	.8674 ± .0157	**.8234 ± .0428**
C4.5	.8881 ± .0244	.8208 ± .0462

Table 9
Average rankings and adjusted *p*-values using Holm's post-hoc procedure for FRBCS methods and C4.5 adopting the GM measure for the borderline imbalanced data-sets.

Algorithm	Average ranking	Adjusted *p*-value (Holm's test)
GP-COACH-H	1.7333	
C4.5	1.9000	0.6831
HFRBCS(Chi)	3.0667	0.0022
GP-COACH-9	3.8667	0.0000
GP-COACH-5	4.4333	0.0000

Table 10
Wilcoxon test to compare GP-COACH-H against C4.5 in borderline imbalanced data-sets. R^+ corresponds to the sum of the ranks for GP-COACH-H and R^- to C4.5.

Comparison	R^+	R^-	*p*-Value
GP-COACH-H vs C4.5	261.0	204.0	0.551

as in the basic algorithm. In this manner, GP-COACH-H is able to profit from the descriptive power of each granularity level obtaining a good balance between the performance of both classes.

5.3. Analysis of the suitability of the preprocessing step for imbalanced problems

In this part of the study, our aim is to show the suitability of the preprocessing step included in GP-COACH-H as the first step of the algorithm. We also check the performance of applying this preprocessing step to the basic GP-COACH algorithm in order to show the necessity of this procedure when dealing with imbalanced data-sets, thus justifying the inclusion of this step in our proposal.

According to this objective, we show the average GM results in training and test in Table 5, together with the corresponding standard deviation, for the basic GP-COACH algorithm and for the hierarchical GP-COACH-H with and without SMOTE preprocessing over the highly imbalanced data-sets presented in Section 4.2. The complete table of results for all data-sets is shown in the appendix of this work. We observe that the best result in test (which is stressed in boldface) always corresponds to the one obtained when the SMOTE preprocessing is applied. Furthermore, there is an enormous difference between the usage or not usage of preprocessing. Therefore, we conclude that the usage of SMOTE as preprocessing clearly outperforms the usage of the original data-sets making the use of this methodology a necessity in the framework of imbalanced data-sets.

5.4. Analysis of GP-COACH-H on highly imbalanced data-sets

The following part of the study will consider the performance of the GP-COACH-H algorithm in contrast with other FRBCS learning proposals and with the C4.5 algorithm. Table 6 shows the average GM results in training and test together with the corresponding standard deviation for the highly imbalanced data-sets considered. By rows, we can observe the results for the basic GP-COACH method with 5 and 9 labels (GP-COACH-5 and GP-COACH-9), the HFRBCS(Chi), the proposed GP-COACH-H and the C4.5 decision tree. The best average case in test is highlighted in bold. The complete table of results for all data-sets is also shown in the appendix of this work together with the results of the previous experiments. We remind that SMOTE is used for the FRBCS whereas SMOTE+ENN is applied in conjunction with C4.5 along all the experiments.

According to the average values shown in this table the best method in highly imbalanced data-sets is the GP-COACH-H. To carry out the statistical study we first check for significant differences among the algorithms using an Iman-Davenport test. The *p*-value (0.0779) is low enough to reject the null equality hypothesis with a high confidence level. Thus, we can conclude that significant differences do exist, proceeding by showing in Table 7 the average ranks of the algorithms and the adjusted p-values computed by the Holm test. Looking at this table we can notice that GP-COACH-H obtains the lower ranking which makes it the control method used for the post-hoc computation. As all the adjusted p-values are sufficiently low to reject the null-hypothesis in all cases, the assumption where GP-COACH-H is the best performing method considered for highly imbalanced data-sets is reinforced.

5.5. Analysis of GP-COACH-H on borderline imbalanced data-sets

In the last part of our study, we want to analyze the behavior of the GP-COACH-H proposal in the scenario of imbalance borderline data-sets. We will take into account the same algorithms considered in the analysis for highly imbalanced data-sets, namely, the basic GP-COACH method with 5 and 9 labels (GP-COACH-5 and GP-COACH-9), HFRBCS(Chi), GP-COACH-H and the C4.5 decision tree. Table 8 shows the average results in training and test together with the corresponding standard deviation for the algorithms used in the study over the borderline imbalanced data-sets. As in previous tables, the best average case in test is highlighted in bold and the complete table of results for the borderline imbalanced data-sets is also shown in the appendix of this work.

Observing the average results table we detect GP-COACH-H as the method with the best average results. Similarly to the procedure used in the highly imbalanced data-sets comparison we start the statistical study for borderline imbalanced data-sets computing the Iman-Davenport test to discern if there are significant differences among the algorithms. The p-value computed is zero, implying that there are differences between the algorithms. Therefore, we perform the Holm test as post-hoc procedure. Table 9 contains the ranks of the algorithms and the adjusted p-values computed using the Holm test.

According to Table 9 the lowest ranking corresponds to GP-COACH-H turning it into the control method used in the Holm test as the best performing method for borderline data-sets. In this case, the adjusted p-values associated to the basic GP-COACH (with 5 and 9 labels) and to HFRBCS(Chi) are low enough to reject the null-hypothesis with a high confidence level. This means, that our proposal GP-COACH-H is the best performing FRBCS in borderline imbalanced data-sets. In the remaining case (C4.5), we perform a Wilcoxon test (Table 10) in order to check if we find differences between both algorithms.

In this case, the p-value computed does not reject the null hypothesis. Nevertheless, GP-COACH-H achieves a higher sum of ranks, which means that GP-COACH-H has obtained a greater performance in a superior number of data-sets than C4.5, turning GP-COACH-H into a competitive method. Furthermore, the average performance of GP-COACH-H is better than the performance of C4.5 and the standard deviation is lower which causes GP-COACH-H to be a more robust method in each occasion.

To sum up, our experimental study has shown that GP-COACH-H is an algorithm that presents a good behavior in the framework of imbalanced data-sets, specifically, when dealing with high imbalanced data and borderline imbalanced data. The design of GP-COACH-H integrates different strategies to deal with the problem that help to overcome the difficulties when they appear. Specifically, the preprocessing step is used to counter the imbalance problem, the hierarchical procedure is added to the FRBCS used as base to obtain a better representation of the data-set in difficult areas such as small disjuncts or borderline samples and the rule selection combined with tuning refines the results obtained improving the overall results. These schemes combined together deal with the mentioned problems in conjunction generating good results.

6. Concluding remarks

In this paper we have presented a FRBCS with different granulation levels that integrates rule selection and the 2-tuples tuning approach to improve the performance in imbalanced data-sets. The proposal integrates data sampling together with algorithm modifications to the basic approach and adapts its behavior to the different granulation levels considered, adding a post-processing step that helps the hierarchical fuzzy rule base classification system to have a better adaptation to the context of each problem and therefore to enhance its global behavior.

The proposed hierarchical fuzzy rule based classification was compared to the GP-COACH algorithm, HFRBCS algorithm and the C4.5 decision tree in order to demonstrate its good performance. The experimental study justifies the combination of SMOTE with the algorithmic modifications such as the usage of a hierarchical knowledge base in order to increase the performance in the imbalanced data-set scenario. Moreover, the results obtained when we deal with this scenario evidence the interest of this proposal. Specifically, this proposal outperforms the other approaches in the framework of highly imbalanced data-sets, which usually is an scenario where most algorithms have lots of difficulties to perform properly.

For borderline imbalanced data-sets our approach shows a better behavior than other FRBCSs used in the experimental studio and maintains a competitive performance when it is compared with C4.5. These results have been contrasted by several non-parametric statistical procedures that reinforce the extracted conclusions.

As future work, we consider several lines of work centered on the features of GP-COACH-H that can still be enhanced to obtain a better performance. One possibility includes the modification of the genetic operations to achieve a multi-objective procedure that enables a trade-off between interpretability and accuracy. Moreover, we want to study in depth the data intrinsic characteristics that hinder the performance in imbalanced data-sets and incorporate this knowledge into the model with a specialized strategy for each case. Another possibility focus on the balance level of the preprocessing step. If an equal balance is not needed and can be substituted by a lower number of instances then the run time of the algorithm will decrease.

Acknowledgments

This work has been supported by the Spanish Ministry of Science and Technology under Projects TIN2011-28488 and TIN2008-06681-C06-02, and the Andalusian Research Plan P10-TIC-6858 and TIC-3928. V. López holds a FPU scholarship from Spanish Ministry of Education.

Appendix A. Detailed results for the experimental study

In this appendix we present the complete results tables for all the algorithms used in this work. Thus, the reader can observe the full training and test results, with their associated standard deviation, in order to compare the performance of each approach. In Table 11 we show the detailed results for the GP-COACH-5, GP-COACH-9 and GP-COACH-H versions with SMOTE preprocessing for the GP procedure using the specificity and sensitivity measures. Next, in Table 12 we show the results for the basic GP-COACH method and the hierarchical GP-COACH-H with and without SMOTE preprocessing. Later, the results for each FRBCS method with SMOTE preprocessing and C4.5 with SMOTE+ENN preprocessing over the highly imbalanced data-sets are shown in Table 13. Finally, Table 14 presents the results for the same algorithms as Table 13 over the borderline data-sets considered.

Table 11
Complete table of results for GP-COACH-5, GP-COACH-9 and GP-COACH-H after the GP procedure using the specificity and sensitivity measures.

Data-set	GP-COACH-5				GP-COACH-9				GP-COACH-H			
	Sensitivity$_{tr}$	Sensitivity$_{tst}$	Specificity$_{tr}$	Specificity$_{tst}$	Sensitivity$_{tr}$	Sensitivity$_{tst}$	Specificity$_{tr}$	Specificity$_{tst}$	Sensitivity$_{tr}$	Sensitivity$_{tst}$	Specificity$_{tr}$	Specificity$_{tst}$
ecoli034vs5	.9750 ± .0165	.8500 ± .1282	.9764 ± .0165	.9722 ± .1282	.9875 ± .0158	.9000 ± .0709	.9653 ± .0158	.9556 ± .0709	.9375 ± .0567	.8500 ± .0759	.9806 ± .0567	.9667 ± .0759
yeast2vs4	.9316 ± .0092	.8818 ± .0412	.9196 ± .0092	.9179 ± .0412	.9265 ± .0044	.8818 ± .0381	.9319 ± .0044	.9288 ± .0381	.9365 ± .0166	.8636 ± .0471	.9352 ± .0166	.9308 ± .0471
ecoli067vs35	.9654 ± .0193	.8400 ± .2093	.9200 ± .0193	.8650 ± .2093	.9438 ± .0210	.8400 ± .2265	.9412 ± .0210	.9250 ± .2265	.9660 ± .0292	.8400 ± .2248	.9463 ± .0292	.9200 ± .2248
ecoli0234vs5	.9625 ± .0095	.7500 ± .1552	.9794 ± .0095	.9338 ± .1552	.9625 ± .0409	.8000 ± .1648	.9670 ± .0409	.9174 ± .1648	.9875 ± .0111	.8500 ± .1239	.9822 ± .0111	.9392 ± .1239
glass015vs2	.7429 ± .1337	.4833 ± .2183	.4774 ± .1337	.4968 ± .2183	.8077 ± .0511	.1833 ± .3032	.8581 ± .0511	.7677 ± .3032	.8978 ± .0440	.6000 ± .0739	.7742 ± .0440	.7677 ± .0739
yeast0359vs78	.7650 ± .0840	.6400 ± .1244	.5273 ± .0840	.5312 ± .1244	.3700 ± .0258	.3600 ± .0833	.8499 ± .0258	.8418 ± .0833	.8450 ± .0199	.7000 ± .0820	.8164 ± .0199	.8026 ± .0820
yeast02579vs368	.8763 ± .0093	.8700 ± .0376	.9577 ± .0093	.9514 ± .0376	.8864 ± .0109	.8900 ± .0395	.9279 ± .0109	.9204 ± .0395	.8788 ± .0093	.8600 ± .0488	.9577 ± .0093	.9547 ± .0488
yeast0256vs3789	.7096 ± .0136	.6858 ± .0676	.9251 ± .0136	.9271 ± .0676	.7322 ± .0149	.7063 ± .0563	.9022 ± .0149	.8994 ± .0563	.7247 ± .0154	.7063 ± .0598	.9191 ± .0154	.9182 ± .0598
ecoli046vs5	.9750 ± .0168	.9000 ± .1248	.9740 ± .0168	.9509 ± .1248	.9875 ± .0174	.8500 ± .2166	.9836 ± .0174	.9566 ± .2166	1.0000 ± .0073	.8500 ± .2117	.9727 ± .0073	.9401 ± .2117
ecoli01vs235	.9689 ± .0151	.8600 ± .1131	.9125 ± .0151	.8955 ± .1131	.9689 ± .0152	.9100 ± .0670	.9398 ± .0152	.9227 ± .0670	.9479 ± .0184	.7700 ± .1915	.9443 ± .0184	.9364 ± .1915
ecoli0267vs35	.9216 ± .0211	.8000 ± .1311	.9209 ± .0211	.9156 ± .1311	.9778 ± .0334	.8000 ± .1125	.9220 ± .0334	.8916 ± .1125	.9444 ± .0260	.8000 ± .0928	.9073 ± .0260	.8709 ± .0928
glass04vs5	1.0000 ± .0208	.9000 ± .1277	.9338 ± .0208	.9287 ± .1277	1.0000 ± .0247	1.0000 ± .0134	.9426 ± .0247	.9279 ± .0134	1.0000 ± .0365	.8000 ± .4020	.9190 ± .0365	.8412 ± .4020
ecoli0346vs5	1.0000 ± .0028	.8000 ± .1132	.9919 ± .0028	.9784 ± .1132	.9875 ± .0159	.8500 ± .0632	.9811 ± .0159	.9459 ± .0632	1.0000 ± .0107	.8500 ± .0608	.9770 ± .0107	.9622 ± .0608
ecoli0347vs56	.9700 ± .0104	.8000 ± .1039	.9461 ± .0104	.9224 ± .1039	.9700 ± .0166	.8000 ± .1459	.9590 ± .0166	.9397 ± .1459	.9800 ± .0069	.8400 ± .0923	.9483 ± .0069	.9309 ± .0923
yeast05679vs4	.7843 ± .0198	.7836 ± .0759	.8569 ± .0198	.8490 ± .0759	.7990 ± .0158	.7636 ± .0858	.8753 ± .0158	.8616 ± .0858	.8184 ± .0117	.7255 ± .0653	.8496 ± .0117	.8449 ± .0653
ecoli067vs5	1.0000 ± .0144	.8000 ± .1277	.9113 ± .0144	.8850 ± .1277	.9625 ± .0162	.8500 ± .0884	.9488 ± .0162	.9400 ± .0884	.9625 ± .0153	.8500 ± .0513	.9463 ± .0153	.9350 ± .0513
vowel0	.9861 ± .0116	.9667 ± .0154	.9527 ± .0116	.9532 ± .0154	.9778 ± .0075	.9444 ± .0093	.9510 ± .0075	.9365 ± .0093	.9611 ± .0160	.9333 ± .0232	.9630 ± .0160	.9588 ± .0232
glass016vs2	.9560 ± .0558	.5500 ± .1228	.6443 ± .0558	.5886 ± .1228	.8231 ± .0355	.3000 ± .2385	.8257 ± .0355	.7886 ± .2385	.9714 ± .0171	.5833 ± .1741	.7286 ± .0171	.7086 ± .1741
glass2	.9264 ± .0256	.7667 ± .0841	.5330 ± .0256	.5074 ± .0841	.7626 ± .0615	.3500 ± .3725	.8514 ± .0615	.8027 ± .3725	.9407 ± .0533	.4667 ± .1544	.8225 ± .0533	.7505 ± .1544
ecoli0147vs2356	.9228 ± .0241	.7200 ± .1060	.9267 ± .0241	.8990 ± .1060	.8808 ± .0521	.7133 ± .1467	.9218 ± .0521	.9056 ± .1467	.9221 ± .0232	.8333 ± .0477	.9120 ± .0232	.9154 ± .0477
led7digit02456789vs1	.8582 ± .0246	.8607 ± .0829	.9421 ± .0246	.9483 ± .0829	.8582 ± .0274	.8321 ± .0708	.9434 ± .0274	.9507 ± .0708	.8648 ± .0216	.8607 ± .0851	.9403 ± .0216	.9458 ± .0851
glass06vs5	1.0000 ± .0073	1.0000 ± .0113	.9798 ± .0073	.9900 ± .0113	1.0000 ± .0069	.9000 ± .1332	.9849 ± .0069	.9300 ± .1332	1.0000 ± .0141	1.0000 ± .0215	.9722 ± .0141	.9595 ± .0215
ecoli01vs5	1.0000 ± .0137	.8000 ± .1248	.9636 ± .0137	.9682 ± .1248	1.0000 ± .0075	.9000 ± .0908	.9784 ± .0075	.9409 ± .0908	1.0000 ± .0043	.8500 ± .0868	.9784 ± .0043	.9545 ± .0868
glass0146vs2	.9253 ± .0483	.7833 ± .0450	.6208 ± .0483	.5909 ± .0450	.8978 ± .0665	.5167 ± .3191	.8085 ± .0665	.8027 ± .3191	.8956 ± .0273	.5833 ± .0748	.7486 ± .0273	.7343 ± .0748
ecoli0147vs56	.9700 ± .0222	.8000 ± .0385	.9455 ± .0222	.8987 ± .0385	.9900 ± .0154	.8000 ± .0661	.9577 ± .0154	.9282 ± .0661	.9800 ± .0198	.8400 ± .0898	.9381 ± .0198	.9118 ± .0898
cleveland0vs4	.9800 ± .0194	.5667 ± .1710	.9687 ± .0194	.9697 ± .1710	.9418 ± .0266	.6333 ± .2114	.9781 ± .0266	.9634 ± .2114	.9600 ± .0378	.8000 ± .1632	.9439 ± .0378	.9146 ± .1632
ecoli0146vs5	1.0000 ± .0171	.8500 ± .1133	.9577 ± .0171	.9385 ± .1133	.9875 ± .0168	.8500 ± .1158	.9750 ± .0168	.9423 ± .1158	1.0000 ± .0111	.8500 ± .1162	.9712 ± .0111	.9462 ± .1162
ecoli4	.9750 ± .0151	.9000 ± .0717	.9755 ± .0151	.9811 ± .0717	.9625 ± .0201	.8500 ± .0806	.9723 ± .0201	.9588 ± .0806	.9750 ± .0143	.9000 ± .0724	.9723 ± .0143	.9684 ± .0724
yeast1vs7	.9333 ± .0568	.8333 ± .0539	.5640 ± .0568	.5644 ± .0539	.8667 ± .0832	.4667 ± .1465	.8736 ± .0832	.8483 ± .1465	.9000 ± .0314	.6667 ± .0899	.7506 ± .0314	.6945 ± .0899
shuttle0vs4	1.0000 ± .0000	1.0000 ± .0020	1.0000 ± .0000	.9982 ± .0020	1.0000 ± .0003	.9920 ± .0103	.9990 ± .0003	.9988 ± .0103	.9980 ± .0023	.9917 ± .0094	1.0000 ± .0023	1.0000 ± .0094
glass4	1.0000 ± .0168	.7333 ± .4090	.9615 ± .0168	.9200 ± .4090	.9800 ± .0212	.8333 ± .1306	.9739 ± .0212	.9500 ± .1306	1.0000 ± .0187	.6667 ± .3937	.9478 ± .0187	.9200 ± .3937
page-blocks13vs4	.9273 ± .0477	.9000 ± .0679	.9189 ± .0477	.9144 ± .0679	.9391 ± .0703	.8400 ± .1574	.9262 ± .0703	.9233 ± .1574	.9735 ± .0116	.7933 ± .1273	.9825 ± .0116	.9752 ± .1273
abalone9-18	.7439 ± .0355	.7306 ± .0996	.6589 ± .0355	.6705 ± .0996	.8275 ± .0149	.5889 ± .1103	.7885 ± .0149	.7851 ± .1103	.8446 ± .0191	.7778 ± .0917	.8004 ± .0191	.7823 ± .0917
glass016vs5	1.0000 ± .0106	1.0000 ± .0422	.9643 ± .0106	.9314 ± .0422	1.0000 ± .0058	.9000 ± .1312	.9643 ± .0058	.9314 ± .1312	1.0000 ± .0108	.8000 ± .1672	.9557 ± .0108	.9429 ± .1672
shuttle2vs4	1.0000 ± .0496	1.0000 ± .0667	.9310 ± .0496	.9190 ± .0667	1.0000 ± .0046	1.0000 ± .0090	.9959 ± .0046	.9920 ± .0090	1.0000 ± .0046	1.0000 ± .0000	.9939 ± .0046	1.0000 ± .0000
yeast1458vs7	.6917 ± .1104	.4333 ± .2086	.6012 ± .1104	.5829 ± .2086	.7750 ± .0498	.4000 ± .1118	.7492 ± .0498	.7451 ± .1118	.8583 ± .0314	.5000 ± .1284	.6923 ± .0314	.6756 ± .1284
glass5	.9714 ± .0317	.6000 ± .5297	.9720 ± .0317	.9561 ± .5297	.9714 ± .0340	.6000 ± .4085	.9854 ± .0340	.9805 ± .4085	1.0000 ± .0368	.8000 ± .4225	.9183 ± .0368	.9122 ± .4225
yeast2vs8	.5750 ± .0319	.5500 ± .1487	.9919 ± .0319	.9957 ± .1487	.6500 ± .0211	.6000 ± .1606	.9973 ± .0211	.9978 ± .1606	.9625 ± .0175	.5500 ± .1322	.9259 ± .0175	.9111 ± .1322
yeast4	.8434 ± .0095	.7236 ± .0527	.8789 ± .0095	.8772 ± .0527	.7988 ± .0140	.6873 ± .0469	.8939 ± .0140	.8905 ± .0469	.9216 ± .0137	.8018 ± .0438	.8238 ± .0137	.8248 ± .0438
yeast1289vs7	.7583 ± .1556	.5333 ± .1253	.6065 ± .1556	.6122 ± .1253	.7917 ± .0672	.4667 ± .1633	.8277 ± .0672	.8079 ± .1633	.9000 ± .0495	.7000 ± .0902	.7132 ± .0495	.6925 ± .0902
yeast5	.9208 ± .0262	.8611 ± .0478	.9493 ± .0262	.9479 ± .0478	.9324 ± .0289	.8833 ± .0434	.9642 ± .0289	.9667 ± .0434	.9487 ± .0131	.9083 ± .0477	.9488 ± .0131	.9479 ± .0477
ecoli0137vs26	.8867 ± .0418	.7000 ± .4202	.9516 ± .0418	.9490 ± .4202	.8533 ± .0354	.7000 ± .4200	.9562 ± .0354	.9562 ± .4200	1.0000 ± .0106	.8000 ± .4201	.9362 ± .0106	.9015 ± .4201
yeast6	.8786 ± .0128	.8571 ± .0907	.9208 ± .0128	.9248 ± .0907	.8571 ± .0187	.8000 ± .1348	.9367 ± .0187	.9399 ± .1348	.8857 ± .0130	.8286 ± .0988	.9294 ± .0130	.9310 ± .0988
abalone19	.8508 ± .0131	.6952 ± .0824	.6165 ± .0131	.6200 ± .0824	.9298 ± .0196	.4714 ± .0569	.7402 ± .0196	.7359 ± .0569	.8588 ± .0165	.4667 ± .1476	.7233 ± .0165	.7216 ± .1476
Mean	.9097 ± .0307	**.7809 ± .1212**	.8643 ± .0307	.8531 ± .1212	.8983 ± .0267	.7319 ± .1334	.9231 ± .0267	**.9055 ± .1334**	.9398 ± .0204	.7797 ± .1233	.9025 ± .0204	.8855 ± .1233

Table 12
Complete table of results for GP-COACH versions with and without SMOTE preprocessing.

Data-set	No preprocessing						SMOTE preprocessing					
	GP-COACH-5		GP-COACH-9		GP-COACH-H		GP-COACH-5		GP-COACH-9		GP-COACH-H	
	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$
ecoli034vs5	.8348 ± .0623	.7293 ± .1508	.8436 ± .0337	.7854 ± .1903	.5871 ± .5366	.5412 ± .4967	.9755 ± .0165	.9018 ± .1282	.9761 ± .0158	.9250 ± .0709	.9833 ± .0276	.8660 ± .1252
yeast2vs4	.8111 ± .0405	.7934 ± .1018	.5085 ± .4646	.4557 ± .4232	.8855 ± .0243	.7817 ± .0850	.9252 ± .0092	.8987 ± .0412	.9283 ± .0044	.9036 ± .0381	.9647 ± .0095	.9304 ± .0288
ecoli067vs35	.7120 ± .1319	.5262 ± .3357	.8724 ± .0453	.6667 ± .3799	.9087 ± .0578	.6631 ± .3861	.9421 ± .0193	.8185 ± .2093	.9420 ± .0210	.8509 ± .2265	.9707 ± .0140	.7286 ± .4095
ecoli0234vs5	.7559 ± .2269	.6610 ± .3759	.8802 ± .0194	.7854 ± .1903	.2500 ± .4330	.1973 ± .4411	.9707 ± .0095	.8286 ± .1552	.9638 ± .0409	.8472 ± .1648	.9966 ± .0024	.8473 ± .1526
glass015vs2	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.5017 ± .1337	.3732 ± .2183	.8301 ± .0511	.2115 ± .3032	.9503 ± .0127	.6301 ± .0922
yeast0359vs78	.3816 ± .0722	.2151 ± .2033	.4467 ± .0198	.4595 ± .0941	.5025 ± .0540	.4136 ± .0989	.5764 ± .0840	.5111 ± .1244	.4804 ± .0258	.4467 ± .0833	.8919 ± .0188	.7189 ± .1013
yeast02579vs368	.4423 ± .2722	.4263 ± .3592	.2948 ± .0425	.2621 ± .1685	.9015 ± .0256	.8413 ± .0321	.9160 ± .0093	.9087 ± .0376	.9068 ± .0109	.9044 ± .0395	.9298 ± .0080	.9107 ± .0303
yeast0256vs3789	.3581 ± .0988	.1078 ± .1513	.3997 ± .1358	.2928 ± .2341	.7495 ± .0269	.6353 ± .1073	.8101 ± .0136	.7954 ± .0676	.8127 ± .0149	.7955 ± .0563	.8348 ± .0146	.7982 ± .0673
ecoli046vs5	.8263 ± .0684	.6878 ± .1833	.8725 ± .0144	.6514 ± .3834	.9551 ± .0180	.8105 ± .2129	.9744 ± .0168	.9171 ± .1248	.9855 ± .0174	.8793 ± .2166	.9952 ± .0039	.8677 ± .2102
ecoli01vs235	.7225 ± .1149	.5614 ± .3222	.7925 ± .1083	.5727 ± .3727	.4644 ± .2395	.2426 ± .3508	.9398 ± .0151	.8682 ± .1131	.9540 ± .0152	.9138 ± .0670	.9845 ± .0143	.8471 ± .0944
ecoli0267vs35	.8223 ± .0748	.5828 ± .3713	.8973 ± .0300	.8518 ± .1458	.7825 ± .3031	.6055 ± .3454	.9201 ± .0211	.8407 ± .1311	.9494 ± .0334	.8365 ± .1125	.9707 ± .0163	.9028 ± .0739
glass04vs5	.3917 ± .4067	.1414 ± .3162	.7777 ± .1085	.6243 ± .3713	.0000 ± .0000	.0000 ± .0000	.9662 ± .0208	.9064 ± .1277	.9706 ± .0247	.9632 ± .0134	.9909 ± .0125	.9429 ± .0419
ecoli0346vs5	.8211 ± .0317	.7560 ± .1884	.8353 ± .0390	.8005 ± .0897	.3742 ± .5123	.3732 ± .5132	.9959 ± .0028	.8772 ± .1132	.9842 ± .0159	.8934 ± .0632	.9993 ± .0015	.8847 ± .0690
ecoli0347vs56	.4067 ± .1337	.1779 ± .2436	.6756 ± .0664	.6729 ± .1658	.5673 ± .1876	.2654 ± .3956	.9576 ± .0104	.8525 ± .1039	.9644 ± .0166	.8571 ± .1459	.9881 ± .0036	.8767 ± .0977
yeast05679vs4	.5301 ± .1708	.4502 ± .2955	.0000 ± .0000	.0000 ± .0000	.7518 ± .1110	.5433 ± .1763	.8194 ± .0198	.8136 ± .0759	.8360 ± .0158	.8080 ± .0858	.8961 ± .0280	.6988 ± .0530
ecoli067vs5	.7960 ± .0691	.5861 ± .3606	.9076 ± .0156	.7474 ± .1815	.9218 ± .0186	.7505 ± .1554	.9545 ± .0144	.8356 ± .1277	.9554 ± .0162	.8923 ± .0884	.9849 ± .0041	.8671 ± .0629
vowel0	.7763 ± .0717	.7098 ± .0795	.8520 ± .0443	.8207 ± .0619	.8046 ± .1343	.7599 ± .1505	.9691 ± .0116	.9598 ± .0154	.9642 ± .0075	.9400 ± .0093	.9947 ± .0057	.9465 ± .0622
glass016vs2	.0555 ± .1240	.0000 ± .0000	.0555 ± .1240	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.7820 ± .0558	.5419 ± .1228	.8120 ± .0355	.4211 ± .2385	.9415 ± .0218	.6467 ± .2206
glass2	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0535 ± .1195	.0000 ± .0000	.7019 ± .0256	.6223 ± .0841	.7981 ± .0615	.3949 ± .3725	.9663 ± .0188	.5886 ± .1299
ecoli0147vs2356	.5083 ± .1123	.3942 ± .2360	.4271 ± .0894	.0816 ± .1826	.5234 ± .0427	.3036 ± .3019	.9247 ± .0241	.7976 ± .1060	.8994 ± .0521	.7972 ± .1467	.9594 ± .0153	.8263 ± .0687
led7digit02456789vs1	.8988 ± .0194	.9013 ± .0830	.9081 ± .0156	.8958 ± .0864	.5328 ± .4872	.5099 ± .4682	.8981 ± .0246	.9011 ± .0829	.8986 ± .0274	.8874 ± .0708	.9142 ± .0158	.9000 ± .0809
glass06vs5	.7986 ± .0637	.7548 ± .1379	.5142 ± .3470	.1949 ± .4359	.1512 ± .3381	.1414 ± .3162	.9898 ± .0073	.9949 ± .0113	.9924 ± .0069	.9060 ± .1332	.9975 ± .0035	.9120 ± .1263
ecoli01vs5	.7401 ± .0531	.6549 ± .1580	.7686 ± .1074	.4130 ± .3987	.9673 ± .0323	.8196 ± .1238	.9816 ± .0137	.8739 ± .1248	.9891 ± .0075	.9190 ± .0908	.9977 ± .0031	.8946 ± .0823
glass0146vs2	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.7552 ± .0483	.6651 ± .0450	.8501 ± .0665	.5675 ± .3191	.9313 ± .0074	.7300 ± .0534
ecoli0147vs56	.3948 ± .1736	.1789 ± .2449	.5243 ± .1376	.2444 ± .3541	.7058 ± .2287	.3563 ± .3715	.9574 ± .0222	.8472 ± .0385	.9735 ± .0154	.8577 ± .0661	.9852 ± .0045	.8372 ± .0595
cleveland0vs4	.8312 ± .0470	.5287 ± .3100	.8656 ± .0464	.3969 ± .5436	.5326 ± .3671	.2784 ± .3812	.9740 ± .0194	.7232 ± .1710	.9595 ± .0266	.7578 ± .2114	.9719 ± .0322	.8646 ± .1627
ecoli0146vs5	.6469 ± .0906	.4560 ± .4213	.8714 ± .0461	.6805 ± .2062	.9284 ± .0280	.7762 ± .2119	.9785 ± .0171	.8832 ± .1133	.9811 ± .0168	.8862 ± .1158	.9952 ± .0038	.9194 ± .0597
ecoli4	.6336 ± .1716	.5146 ± .3263	.6761 ± .3818	.4696 ± .3039	.0000 ± .0000	.0000 ± .0000	.9751 ± .0151	.9373 ± .0717	.9673 ± .0201	.9008 ± .0806	.9936 ± .0043	.9357 ± .0702
yeast1vs7	.0986 ± .1382	.0000 ± .0000	.2234 ± .2285	.0000 ± .0000	.1632 ± .1706	.0000 ± .0000	.7229 ± .0568	.6802 ± .0539	.8676 ± .0832	.6093 ± .1465	.8988 ± .0301	.6900 ± .0646
shuttle0vs4	.8361 ± .0143	.8300 ± .0595	.9877 ± .0101	.9744 ± .0387	1.0000 ± .0000	.9960 ± .0090	1.0000 ± .0000	.9991 ± .0020	.9995 ± .0003	.9954 ± .0103	1.0000 ± .0000	1.0000 ± .0000
glass4	.4689 ± .1175	.1155 ± .2582	.7019 ± .0955	.4737 ± .4550	.2963 ± .1948	.0000 ± .0000	.9804 ± .0168	.7231 ± .4090	.9766 ± .0212	.8811 ± .1306	.9906 ± .0077	.7303 ± .4132
page-blocks13vs4	.7063 ± .0430	.6917 ± .1876	.7397 ± .0877	.6815 ± .2216	.7321 ± .1134	.6016 ± .1727	.9205 ± .0477	.9035 ± .0679	.9316 ± .0703	.8706 ± .1574	.9994 ± .0008	.9482 ± .0502
abalone9-18	.3172 ± .0875	.2357 ± .2205	.3648 ± .2181	.2412 ± .2282	.4393 ± .1049	.2565 ± .2510	.6884 ± .0355	.6922 ± .0996	.8070 ± .0149	.6699 ± .1103	.8595 ± .0265	.7500 ± .0599
glass016vs5	.4309 ± .4010	.1414 ± .3162	.7755 ± .0843	.4828 ± .4567	.0000 ± .0000	.0000 ± .0000	.9819 ± .0106	.9644 ± .0422	.9820 ± .0058	.9090 ± .1312	.9921 ± .0078	.8550 ± .1596
shuttle2vs4	.6367 ± .2363	.6000 ± .5477	.2159 ± .3028	.2000 ± .4472	.8257 ± .1510	.8000 ± .4472	.9639 ± .0496	.9568 ± .0667	.9979 ± .0046	.9960 ± .0090	1.0000 ± .0000	.9918 ± .0183
yeast1458vs7	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.5775 ± .1104	.3546 ± .2086	.7566 ± .0498	.5353 ± .1118	.8952 ± .0261	.6304 ± .1095
glass5	.2926 ± .4010	.2000 ± .4472	.7766 ± .0977	.5372 ± .5051	.0000 ± .0000	.0000 ± .0000	.9711 ± .0317	.5801 ± .5297	.9780 ± .0340	.6758 ± .4085	.9957 ± .0027	.7877 ± .4404
yeast2vs8	.7401 ± .0348	.7283 ± .1497	.7401 ± .0348	.7283 ± .1497	.7410 ± .0351	.7283 ± .1497	.7544 ± .0319	.7274 ± .1487	.8049 ± .0211	.7601 ± .1606	.9937 ± .0047	.7381 ± .1765
yeast4	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0853 ± .1236	.0000 ± .0000	.8602 ± .0095	.7923 ± .0527	.8443 ± .0140	.7807 ± .0469	.9001 ± .0156	.8175 ± .0391
yeast1289vs7	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.6325 ± .1556	.5262 ± .1253	.7996 ± .0672	.5860 ± .1633	.8843 ± .0292	.6939 ± .1205
yeast5	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0333 ± .0745	.0000 ± .0000	.9344 ± .0262	.9020 ± .0478	.9477 ± .0289	.9229 ± .0434	.9724 ± .0066	.9428 ± .0526
ecoli0137vs26	.6472 ± .0986	.1414 ± .3162	.3344 ± .1877	.1414 ± .3162	.8430 ± .0583	.1401 ± .3133	.9167 ± .0418	.7215 ± .4202	.9021 ± .0354	.7203 ± .4200	.9843 ± .0107	.7067 ± .4136
yeast6	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.8994 ± .0128	.8856 ± .0907	.8960 ± .0187	.8574 ± .1348	.9319 ± .0155	.8170 ± .0977
abalone19	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.7196 ± .0131	.6425 ± .0824	.8290 ± .0196	.5828 ± .0569	.8558 ± .0193	.5532 ± .1487
Mean	.4789 ± .1017	.3677 ± .1922	.5074 ± .0871	.3929 ± .1996	.4536 ± .1216	.3439 ± .1697	.8763 ± .0307	.7897 ± .1212	.9056 ± .0267	.7845 ± .1334	.9576 ± .0121	**.8175 ± .1193**

Table 13
Complete table of results for FRBCS methods and C4.5 in highly imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE + ENN for C4.5.

Data-set	GP-COACH-5		GP-COACH-9		HFRBCS(Chi)		GP-COACH-H		C4.5	
	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}
ecoli034vs5	.9755 ± .0165	.9018 ± .1282	.9761 ± .0158	.9250 ± .0709	.9930 ± .0050	.8421 ± .1471	.9833 ± .0276	.8660 ± .1252	.9762 ± .0149	.8761 ± .0492
yeast2vs4	.9252 ± .0092	.8987 ± .0412	.9283 ± .0044	.9036 ± .0381	.9527 ± .0105	.8932 ± .0418	.9647 ± .0095	.9304 ± .0288	.9745 ± .0066	.9029 ± .0406
ecoli067vs35	.9421 ± .0193	.8185 ± .2093	.9420 ± .0210	.8509 ± .2265	.9574 ± .0163	.8267 ± .1415	.9707 ± .0140	.7286 ± .4095	.9771 ± .0204	.7206 ± .4072
ecoli0234vs5	.9707 ± .0095	.8286 ± .1552	.9638 ± .0409	.8472 ± .1648	.9910 ± .0031	.8425 ± .1504	.9966 ± .0024	.8473 ± .1526	.9827 ± .0074	.8861 ± .1245
glass015vs2	.5017 ± .1337	.3732 ± .2183	.8301 ± .0511	.2115 ± .3032	.6967 ± .0269	.5590 ± .0851	.9503 ± .0127	.6301 ± .0922	.9066 ± .0263	.7788 ± .2089
yeast0359vs78	.5764 ± .0840	.5111 ± .1244	.4804 ± .0258	.4467 ± .0833	.8401 ± .0040	.7330 ± .0403	.8919 ± .0188	.7189 ± .1013	.9213 ± .0214	.6894 ± .0888
yeast02579vs368	.9160 ± .0093	.9087 ± .0376	.9068 ± .0109	.9044 ± .0395	.9063 ± .0103	.8946 ± .0355	.9298 ± .0080	.9107 ± .0303	.9572 ± .0206	.9125 ± .0336
yeast0256vs3789	.8101 ± .0136	.7954 ± .0676	.8127 ± .0149	.7955 ± .0563	.8106 ± .0165	.7927 ± .0674	.8348 ± .0146	.7982 ± .0673	.9173 ± .0180	.7707 ± .0366
ecoli046vs5	.9744 ± .0168	.9171 ± .1248	.9855 ± .0174	.8793 ± .2166	.9925 ± .0029	.8800 ± .1156	.9952 ± .0039	.8677 ± .2102	.9834 ± .0079	.8776 ± .1148
ecoli01vs235	.9398 ± .0151	.8682 ± .1131	.9540 ± .0152	.9138 ± .0670	.9844 ± .0132	.8709 ± .1076	.9845 ± .0143	.8471 ± .0944	.9649 ± .0302	.8277 ± .1191
ecoli0267vs35	.9201 ± .0211	.8407 ± .1311	.9494 ± .0334	.8365 ± .1125	.9609 ± .0140	.8247 ± .1089	.9707 ± .0163	.9028 ± .0739	.9825 ± .0058	.8061 ± .1065
glass04vs5	.9662 ± .0208	.9064 ± .1277	.9706 ± .0247	.9632 ± .0134	.9457 ± .0185	.7092 ± .3976	.9909 ± .0125	.9429 ± .0419	.9909 ± .0063	.9748 ± .0269
ecoli0346vs5	.9959 ± .0028	.8772 ± .1132	.9842 ± .0159	.8934 ± .0632	.9918 ± .0057	.8729 ± .1175	.9993 ± .0015	.8847 ± .0690	.9884 ± .0046	.8946 ± .0793
ecoli0347vs56	.9576 ± .0104	.8525 ± .1039	.9644 ± .0166	.8571 ± .1459	.9682 ± .0099	.9007 ± .0962	.9881 ± .0036	.8767 ± .0977	.9566 ± .0176	.8413 ± .1377
yeast05679vs4	.8194 ± .0198	.8136 ± .0759	.8360 ± .0158	.8080 ± .0858	.9290 ± .0103	.7318 ± .0747	.8961 ± .0280	.6988 ± .0530	.9197 ± .0128	.7678 ± .1029
ecoli067vs5	.9545 ± .0144	.8356 ± .1277	.9554 ± .0162	.8923 ± .0884	.9531 ± .0211	.8559 ± .0542	.9849 ± .0041	.8671 ± .0629	.9740 ± .0103	.8376 ± .1167
vowel0	.9691 ± .0116	.9598 ± .0154	.9642 ± .0075	.9400 ± .0093	.9999 ± .0003	.9882 ± .0162	.9947 ± .0057	.9465 ± .0622	.9943 ± .0047	.9417 ± .0815
glass016vs2	.7820 ± .0558	.5419 ± .1228	.8120 ± .0355	.4211 ± .2385	.8726 ± .0230	.5837 ± .2004	.9415 ± .0218	.6467 ± .2206	.9365 ± .0323	.6063 ± .1173
glass2	.7019 ± .0256	.6223 ± .0841	.7981 ± .0615	.3949 ± .3725	.8299 ± .0174	.5484 ± .2057	.9663 ± .0188	.5886 ± .1299	.9261 ± .0342	.7377 ± .1633
ecoli0147vs2356	.9247 ± .0241	.7976 ± .1060	.8994 ± .0521	.7972 ± .1467	.9517 ± .0109	.8477 ± .0655	.9594 ± .0153	.8263 ± .0687	.9563 ± .0318	.8119 ± .0459
led7digit02456789vs1	.8981 ± .0246	.9011 ± .0829	.8986 ± .0274	.8874 ± .0708	.9380 ± .0212	.8276 ± .0778	.9142 ± .0158	.9000 ± .0809	.9217 ± .0192	.8370 ± .0475
glass06vs5	.9898 ± .0073	.9949 ± .0113	.9924 ± .0069	.9060 ± .1332	.9744 ± .0046	.8907 ± .1178	.9975 ± .0035	.9120 ± .1263	.9911 ± .0035	.9628 ± .0556
ecoli01vs5	.9816 ± .0137	.8739 ± .1248	.9891 ± .0075	.9190 ± .0908	.9932 ± .0043	.8689 ± .1166	.9977 ± .0031	.8946 ± .0823	.9828 ± .0068	.8081 ± .1213
glass0146vs2	.7552 ± .0483	.6651 ± .0450	.8501 ± .0665	.5675 ± .3191	.7005 ± .0077	.5117 ± .1026	.9313 ± .0074	.7300 ± .0534	.9010 ± .0596	.6157 ± .3465
ecoli0147vs56	.9574 ± .0222	.8472 ± .0385	.9735 ± .0154	.8577 ± .0661	.9790 ± .0059	.8886 ± .0918	.9852 ± .0045	.8372 ± .0595	.9608 ± .0173	.8250 ± .1380
cleveland0vs4	.9740 ± .0194	.7232 ± .1710	.9595 ± .0266	.7578 ± .2114	.9992 ± .0018	.3961 ± .3827	.9719 ± .0322	.8646 ± .1627	.9819 ± .0187	.7307 ± .1517
ecoli0146vs5	.9785 ± .0171	.8832 ± .1133	.9811 ± .0168	.8862 ± .1158	.9913 ± .0047	.8674 ± .1069	.9952 ± .0038	.9194 ± .0597	.9850 ± .0061	.8880 ± .1148
ecoli4	.9751 ± .0151	.9373 ± .0717	.9673 ± .0201	.9008 ± .0806	.9869 ± .0141	.9302 ± .0817	.9936 ± .0043	.9357 ± .0702	.9826 ± .0170	.8947 ± .1202
yeast1vs7	.7229 ± .0568	.6802 ± .0539	.8676 ± .0832	.6093 ± .1465	.9163 ± .0225	.7074 ± .1240	.8988 ± .0301	.6900 ± .0646	.9093 ± .0332	.7222 ± .0532
shuttle0vs4	1.0000 ± .0000	.9991 ± .0020	.9995 ± .0003	.9954 ± .0103	1.0000 ± .0000	.9912 ± .0115	1.0000 ± .0000	1.0000 ± .0000	.9999 ± .0002	.9997 ± .0007
glass4	.9804 ± .0168	.7231 ± .4090	.9766 ± .0212	.8811 ± .1306	.9981 ± .0017	.7039 ± .4049	.9906 ± .0077	.7303 ± .4132	.9665 ± .0149	.7639 ± .4279
page-blocks13vs4	.9205 ± .0477	.9035 ± .0679	.9316 ± .0703	.8706 ± .1574	.9989 ± .0012	.9864 ± .0065	.9994 ± .0008	.9482 ± .0502	.9975 ± .0018	.9909 ± .0065
abalone9-18	.6884 ± .0355	.6922 ± .0996	.8070 ± .0149	.6699 ± .1103	.8396 ± .0303	.6756 ± .1401	.8595 ± .0265	.7500 ± .0599	.9273 ± .0074	.6884 ± .1181
glass016vs5	.9819 ± .0106	.9644 ± .0422	.9820 ± .0058	.9090 ± .1312	.9971 ± .0030	.7796 ± .4361	.9921 ± .0078	.8550 ± .1596	.9863 ± .0047	.7738 ± .4328
shuttle2vs4	.9639 ± .0496	.9568 ± .0667	.9979 ± .0046	.9960 ± .0090	.9990 ± .0023	.9749 ± .0271	1.0000 ± .0000	.9918 ± .0183	1.0000 ± .0000	1.0000 ± .0000
yeast1458vs7	.5775 ± .1104	.3546 ± .2086	.7566 ± .0498	.5353 ± .1118	.9037 ± .0133	.6249 ± .0626	.8952 ± .0261	.6304 ± .1095	.8717 ± .0492	.3345 ± .3342
glass5	.9711 ± .0317	.5801 ± .5297	.9780 ± .0340	.6758 ± .4085	.9764 ± .0221	.6873 ± .3956	.9957 ± .0027	.7877 ± .4404	.9698 ± .0296	.5851 ± .5343
yeast2vs8	.7544 ± .0319	.7274 ± .1487	.8049 ± .0211	.7601 ± .1606	.8334 ± .0164	.7247 ± .1510	.9937 ± .0047	.7381 ± .1765	.8923 ± .0447	.8033 ± .1167
yeast4	.8602 ± .0095	.7923 ± .0527	.8443 ± .0140	.7807 ± .0469	.9001 ± .0194	.8264 ± .0229	.9001 ± .0156	.8175 ± .0391	.8984 ± .0123	.6897 ± .0769
yeast1289vs7	.6325 ± .1556	.5262 ± .1253	.7996 ± .0672	.5860 ± .1633	.8699 ± .0224	.6937 ± .0437	.8843 ± .0292	.6939 ± .1205	.9408 ± .0259	.5522 ± .1662
yeast5	.9344 ± .0262	.9020 ± .0478	.9477 ± .0289	.9229 ± .0434	.9782 ± .0033	.9420 ± .0259	.9724 ± .0066	.9428 ± .0526	.9819 ± .0077	.9390 ± .0474
ecoli0137vs26	.9167 ± .0418	.7215 ± .4202	.9021 ± .0354	.7203 ± .4200	.9867 ± .0079	.7148 ± .4180	.9843 ± .0107	.7067 ± .4136	.9650 ± .0320	.7062 ± .4093
yeast6	.8994 ± .0128	.8856 ± .0907	.8960 ± .0187	.8574 ± .1348	.9341 ± .0177	.8492 ± .1288	.9319 ± .0155	.8170 ± .0977	.9301 ± .0157	.8029 ± .1541
abalone19	.7196 ± .0131	.6425 ± .0824	.8290 ± .0196	.5828 ± .0569	.8343 ± .0280	.7019 ± .0856	.8558 ± .0193	.5532 ± .1487	.8838 ± .0300	.1550 ± .2125
Mean	.8763 ± .0307	.7897 ± .1212	.9056 ± .0267	.7845 ± .1334	.9331 ± .0117	.7901 ± .1325	.9576 ± .0121	**.8175 ± .1193**	.9549 ± .0180	.7848 ± .1452

Table 14
Complete table of results for FRBCS methods and C4.5 in borderline imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE+ENN for C4.5.

Data-set	GP-COACH-5		GP-COACH-9		HFRBCS(Chi)		GP-COACH-H		C4.5	
	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}
paw02a-600-5-70-BI	.7969 ± .0223	.7892 ± .0297	.7898 ± .0311	.7178 ± .0867	.8730 ± .0129	.8460 ± .0395	.8824 ± .0068	.8523 ± .0342	.8879 ± .0176	.8310 ± .0292
paw02a-600-5-60-BI	.7849 ± .0185	.7603 ± .0698	.7364 ± .1198	.6329 ± .1650	.8485 ± .0311	.8157 ± .0309	.8753 ± .0046	.8501 ± .0235	.8755 ± .0287	.8094 ± .0279
paw02a-600-5-50-BI	.8188 ± .0129	.7869 ± .0676	.8541 ± .0123	.8117 ± .0348	.8595 ± .0307	.8226 ± .0514	.9002 ± .0067	.8402 ± .0227	.8936 ± .0213	.8301 ± .0468
paw02a-600-5-30-BI	.8418 ± .0121	.8193 ± .0639	.8567 ± .0209	.8281 ± .0695	.8749 ± .0185	.8573 ± .0352	.9036 ± .0143	.8605 ± .0660	.8990 ± .0328	.8604 ± .0542
paw02a-600-5-0-BI	.8427 ± .0313	.8489 ± .0311	.9130 ± .0299	.8765 ± .0836	.9339 ± .0076	.9142 ± .0179	.9596 ± .0055	.9367 ± .0124	.9512 ± .0389	.9473 ± .0259
04clover5z-600-5-70-BI	.7706 ± .0217	.7457 ± .1094	.7832 ± .0233	.7094 ± .0921	.7790 ± .0371	.7427 ± .0816	.8250 ± .0126	.7795 ± .0574	.8665 ± .0173	.7557 ± .0468
04clover5z-600-5-60-BI	.7789 ± .0209	.7464 ± .0290	.7243 ± .0473	.6445 ± .0800	.7713 ± .0242	.7725 ± .0514	.8405 ± .0146	.7986 ± .0328	.8895 ± .0151	.7990 ± .0757
04clover5z-600-5-50-BI	.7658 ± .0362	.7434 ± .0872	.7804 ± .0385	.7496 ± .1121	.7925 ± .0390	.7582 ± .0536	.8537 ± .0280	.8080 ± .0530	.8771 ± .0286	.8063 ± .0575
04clover5z-600-5-30-BI	.7839 ± .0294	.7683 ± .0638	.8029 ± .0699	.7453 ± .0767	.8087 ± .0332	.8097 ± .0474	.8737 ± .0182	.8093 ± .0688	.8957 ± .0175	.8291 ± .0337
04clover5z-600-5-0-BI	.7842 ± .0151	.7663 ± .0536	.8389 ± .0146	.7705 ± .0179	.8104 ± .0288	.8008 ± .0309	.8900 ± .0363	.8519 ± .0517	.9269 ± .0333	.8652 ± .0271
03subcl5-600-5-70-BI	.6807 ± .0416	.6319 ± .0582	.7940 ± .0278	.7478 ± .0572	.7947 ± .0138	.7278 ± .0277	.8528 ± .0098	.8006 ± .0419	.8381 ± .0367	.7617 ± .0406
03subcl5-600-5-60-BI	.7471 ± .0251	.6898 ± .0903	.8047 ± .0196	.7789 ± .0483	.8083 ± .0182	.7498 ± .0719	.8067 ± .0434	.7379 ± .0545	.8322 ± .0348	.7641 ± .0732
03subcl5-600-5-50-BI	.7688 ± .0094	.7320 ± .0902	.7920 ± .0172	.7500 ± .0869	.8020 ± .0227	.7465 ± .0537	.8269 ± .0257	.7563 ± .0600	.8332 ± .0109	.7753 ± .0895
03subcl5-600-5-30-BI	.8022 ± .0333	.7860 ± .0504	.8183 ± .0137	.8020 ± .0930	.8148 ± .0276	.7713 ± .0552	.8474 ± .0169	.8307 ± .0379	.8615 ± .0208	.8140 ± .0552
03subcl5-600-5-0-BI	.8795 ± .0173	.8685 ± .0470	.8952 ± .0145	.8965 ± .0183	.8985 ± .0048	.8765 ± .0329	.9364 ± .0015	.9179 ± .0132	.9336 ± .0265	.8969 ± .0368
paw02a-800-7-70-BI	.7947 ± .0138	.7733 ± .0423	.7872 ± .0391	.6775 ± .1242	.8601 ± .0118	.8197 ± .0415	.8741 ± .0104	.8421 ± .0274	.8923 ± .0147	.8001 ± .0486
paw02a-800-7-60-BI	.8028 ± .0132	.7410 ± .0558	.8089 ± .0167	.7235 ± .0549	.8514 ± .0070	.8113 ± .0439	.8706 ± .0105	.8253 ± .0523	.8817 ± .0188	.8043 ± .0352
paw02a-800-7-50-BI	.8211 ± .0026	.7736 ± .0546	.8317 ± .0152	.7905 ± .0352	.8719 ± .0074	.8373 ± .0416	.8863 ± .0081	.8164 ± .0704	.9072 ± .0238	.8448 ± .0447
paw02a-800-7-30-BI	.8391 ± .0135	.8170 ± .0565	.8445 ± .0423	.7998 ± .0312	.8894 ± .0059	.8672 ± .0145	.9030 ± .0077	.8299 ± .0398	.9135 ± .0282	.8449 ± .0455
paw02a-800-7-0-BI	.8493 ± .0472	.8300 ± .0418	.9197 ± .0506	.9100 ± .0433	.9288 ± .0068	.9307 ± .0192	.9576 ± .0092	.9351 ± .0245	.9532 ± .0318	.9371 ± .0334
04clover5z-800-7-70-BI	.7708 ± .0199	.7351 ± .0673	.7839 ± .0335	.7100 ± .1069	.7898 ± .0143	.7237 ± .1008	.8182 ± .0161	.7857 ± .0643	.8723 ± .0429	.7525 ± .0956
04clover5z-800-7-60-BI	.7722 ± .0159	.7796 ± .0340	.7108 ± .0542	.6688 ± .0708	.7799 ± .0194	.7842 ± .0591	.8256 ± .0073	.7707 ± .0461	.8889 ± .0276	.7704 ± .0396
04clover5z-800-7-50-BI	.7860 ± .0215	.7436 ± .0766	.7744 ± .0198	.7544 ± .1039	.7928 ± .0290	.7543 ± .0608	.8390 ± .0138	.7897 ± .0645	.8946 ± .0099	.8261 ± .0701
04clover5z-800-7-30-BI	.7879 ± .0219	.7488 ± .0362	.8162 ± .0221	.7545 ± .0775	.8075 ± .0324	.7750 ± .0583	.8513 ± .0171	.7805 ± .0541	.8930 ± .0215	.8256 ± .0360
04clover5z-800-7-0-BI	.7962 ± .0210	.7578 ± .0503	.8249 ± .0347	.7694 ± .0373	.8091 ± .0283	.7693 ± .0608	.8958 ± .0213	.8541 ± .0578	.9412 ± .0370	.8730 ± .0413
03subcl5-800-7-70-BI	.6662 ± .0352	.6456 ± .0446	.7107 ± .1049	.6454 ± .1444	.7784 ± .0059	.7552 ± .0490	.8186 ± .0330	.7868 ± .0250	.8255 ± .0211	.7735 ± .0376
03subcl5-800-7-60-BI	.7250 ± .0225	.6991 ± .0392	.7962 ± .0137	.7696 ± .0284	.7896 ± .0195	.7331 ± .0436	.8102 ± .0325	.7729 ± .0347	.8374 ± .0244	.7513 ± .0357
03subcl5-800-7-50-BI	.7584 ± .0152	.7261 ± .0908	.8021 ± .0058	.7516 ± .0509	.7927 ± .0158	.7239 ± .0340	.8204 ± .0200	.7436 ± .0310	.8396 ± .0089	.7507 ± .0438
03subcl5-800-7-30-BI	.7993 ± .0219	.7689 ± .0630	.8133 ± .0269	.7901 ± .0594	.8327 ± .0228	.7955 ± .0439	.8552 ± .0081	.8297 ± .0390	.8812 ± .0155	.7941 ± .0220
03subcl5-800-7-0-BI	.8814 ± .0223	.8663 ± .0396	.8998 ± .0099	.9061 ± .0238	.9036 ± .0081	.8851 ± .0321	.9217 ± .0126	.9081 ± .0224	.9588 ± .0249	.9292 ± .0370
Mean	.7899 ± .0218	.7630 ± .0578	.8103 ± .0330	.7628 ± .0705	.8316 ± .0195	.7992 ± .0461	.8674 ± .0157	**.8234 ± .0428**	.8881 ± .0244	.8208 ± .0462

References

[1] R. Alcalá, J. Alcalá-Fdez, F. Herrera, A proposal for the genetic lateral tuning of linguistic fuzzy systems and its interaction with rule selection, IEEE Transactions on Fuzzy Systems 15 (2007) 616–635.

[2] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Journal of Multi-Valued Logic and Soft Computing 17 (2011) 255–287.

[3] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2009) 307–318.

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

[5] A. Bastian, How to handle the flexibility of linguistic variables with applications, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 2 (1994) 463–484.

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

[7] F.J. Berlanga, A.J. Rivera, M.J. del Jesus, F. Herrera, GP-COACH: genetic programming-based learning of compact and accurate fuzzy rule-based classification systems for high-dimensional problems, Information Sciences 180 (2010) 1183–1200.

[8] J. Charles, Automatic recognition of complete palynomorphs in digital images, Machine Vision and Applications 22 (2011) 53–60.

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

[10] Z. Chi, H. Yan, T. Pham, Fuzzy Algorithms with Applications to Image Processing and Pattern Recognition, World Scientific, 1996.

[11] O. Cordón, F. Herrera, A proposal for improving the accuracy of linguistic modeling, IEEE Transactions on Fuzzy Systems 8 (2000) 335–344.

[12] O. Cordón, F. Herrera, F. Hoffmann, L. Magdalena, Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases, Advances in Fuzzy Systems – Applications and Theory, vol. 19, World Scientific, 2001.

[13] O. Cordón, F. Herrera, P. Villar, Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base, IEEE Transactions on Fuzzy Systems 9 (2001) 667–674.

[14] O. Cordon, F. Herrera, I. Zwir, Linguistic modeling by hierarchical systems of linguistic rules, IEEE Transactions on Fuzzy Systems 10 (2002) 2–20.

[15] O. Cordón, M. del Jesus, F. Herrera, A proposal on reasoning methods in fuzzy rule-based classification systems, International Journal of Approximate 20 (1999) 21–45.

[16] C. Drummond, R.C. Holte, C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling, in: Proceedings of the International Conference on Machine Learning (ICML 2003) Workshop on Learning from Imbalanced Data Sets II, 2003, pp. 1–8.

[17] C. Elkan, The foundations of cost-sensitive learning, in: Proceedings of the 17th IEEE International Joint Conference on Artificial Intelligence (IJCAI'01), 2001, pp. 973–978.

[18] L.J. Eshelman, The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination, in: Foudations of Genetic Algorithms, Morgan Kaufmann, 1990, pp. 265–283.

[19] A. Fernández, S. García, F. Herrera, Addressing the classification with imbalanced data: open problems and new challenges on class distribution, in: Proceedings of the 6th International Conference on Hybrid Artificial Intelligence Systems (HAIS'11), 2011, pp. 1–10.

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

[21] A. Fernández, M.J. del Jesus, F. Herrera, Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets, International Journal of Approximate Reasoning 50 (2009) 561–577.

[22] A. Fernández, M.J. del Jesus, F. Herrera, On the 2-tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data-sets, Information Sciences 180 (2010) 1268–1291.

[23] S. García, J. Derrac, I. Triguero, C.J. Carmona, F. Herrera, Evolutionary-based selection of generalized instances for imbalanced classification, Knowledge-Based Systems 25 (2012) 3–12.

[24] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Computing 13 (2009) 959–977.

[25] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2607–2624.

[26] V. García, R. Mollineda, J.S. Sánchez, On the k-NN performance in a challenging scenario of imbalance and overlapping, Pattern Analysis Applications 11 (2008) 269–280.

[27] V. García, J. Sánchez, R. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowledge-Based Systems 25 (2012) 13–21.

[28] N. García-Pedrajas, J. Pérez-Rodríguez, M. García-Pedrajas, D. Ortiz-Boyer, C. Fyfe, Class imbalance methods for translation initiation site recognition in DNA sequences, Knowledge-Based Systems 25 (2012) 22–34.

[29] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering 21 (2009) 1263–1284.

[30] F. Herrera, Genetic fuzzy systems: taxonomy, current research trends and prospects, Evolutionary Intelligence 1 (2008) 27–46.

[31] F. Herrera, M. Lozano, A.M. Sánchez, A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study, International Journal of Intelligent Systems 18 (2003) 309–338.

[32] F. Herrera, L. Martínez, A 2-tuple fuzzy linguistic representation model for computing with words, IEEE Transactions on Fuzzy Systems 8 (2000) 746–752.

[33] H. Ishibuchi, T. Nakashima, Effect of rule weights in fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 9 (2001) 506–515.

[34] H. Ishibuchi, T. Nakashima, M. Nii, Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining, Springer-Verlag, 2004.

[35] H. Ishibuchi, K. Nozaki, N. Yamamoto, H. Tanaka, Selecting fuzzy if–then rules from classification problems using genetic algorithms, IEEE Transactions on Fuzzy Systems 9 (1995) 260–270.

[36] H. Ishibuchi, T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 13 (2005) 428–435.

[37] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intelligent Data Analysis Journal 6 (2002) 429–450.

[38] W. Khreich, E. Granger, A. Miri, R. Sabourin, Iterative boolean combination of classifiers in the ROC space: an application to anomaly detection with HMMs, Pattern Recognition 43 (2010) 2732–2752.

[39] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, The MIT Press, 1992.

[40] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection, in: 14th International Conference on Machine Learning(ICML97), 1997, pp. 179–186.

[41] J. Liu, Q. Hu, D. Yu, A comparative study on rough set based class imbalance learning, Knowledge-Based Systems 21 (2008) 753–763.

[42] V. López, A. Fernández, J.G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics, Expert Systems with Applications 39 (2012) 6585–6608.

[43] L. Magdalena, F. Monasterio-Huelin, A fuzzy logic controller with learning through the evolution of its knowledge base, International Journal of Approximate Reasoning 16 (1997) 335–358.

[44] K. Napierala, J. Stefanowski, S. Wilk, Learning from imbalanced data in presence of noisy and borderline examples, in: 7th International Conference on Rough Sets and Current Trends in Computing (RSCTC2010), 2010, pp. 158–167.

[45] S. Oh, M. Lee, B.T. Zhang, Ensemble learning with active example selection for imbalanced biomedical data classification, IEEE/ACM Transactions on Computational Biology and Bioinformatics 8 (2011) 316–325.

[46] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced datasets, Soft Computing 13 (2009) 213–225.

[47] J. Quinlan, C4.5: Programs for Machine Learning., Morgan Kauffman, 1993.

[48] J. Ren, ANN vs. SVM: which one performs better in classification of MCCs in mammogram imaging, Knowledge-Based Systems 26 (2012) 144–153.

[49] C. Seiffert, T.M. Khoshgoftaar, J.V. Hulse, A. Folleco, An empirical study of the classification performance of learners on imbalanced and noisy software quality data, Information Sciences, in press. http://dx.doi.org/10.1016/j.ins.2010.12.016.

[50] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC, 2011.

[51] Y. Sun, A.K.C. Wong, M.S. Kamel, Classification of imbalanced data: a review, International Journal of Pattern Recognition and Artificial Intelligence 23 (2009) 687–719.

[52] D. Wang, T.S. Dillon, Extraction of classification rules characterized by ellipsoidal regions using soft-computing techniques, International Journal of Systems Science 37 (2006) 969–980.

[53] G.M. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (2004) 7–19.

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

[55] M.L. Wong, K.S. Leung, Data Mining Using Grammar-Based Genetic Programming and Applications, Kluwer Academic Publishers, 2000.

[56] Q. Yang, X. Wu, 10 challenging problems in data mining research, International Journal of Information Technology & Decision Making 5 (2006) 597–604.

[57] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), 2001, pp. 204–213.