A Metahierarchical Rule Decision System to Design Robust Fuzzy Classifiers Based on Data Complexity

Javier Cózar, Alberto Fernández, Francisco Herrera, and José A. Gámez

Abstract—**There is a wide variety of studies that propose different classifiers to solve a large amount of problems in distinct classification scenarios. The no free lunch theorem states that if we use a big enough set of varied problems, all classifiers would be equivalent in performance. From another point of view, the performance of the classifiers is dependant of the scope and properties of the datasets. In this sense, new proposals on the topic often focus on a given context, aiming at improving the related state-of-the-art approaches. Data complexity metrics have been traditionally used to determine the inner characteristics of datasets. This way, researchers are able to categorize the problems in different scenarios. Then, this taxonomy can be applied to determine inner characteristics of the datasets in order to determine intervals of good and bad behavior for a given classifier. In this paper, we will take advantage of the data complexity metrics in order to design a fuzzy metaclassifier. The final goal is to create decision rules based on the inner characteristics of the data to apply a different version of the fuzzy classifier for a given problem. To do so, we will make use of the FARC-HD classifier, an evolutionary fuzzy system that has led to different extensions in the specialized literature. Experimental results show the goodness of this novel approach as it is able to outperform all versions of FARC-HD on a wide set of problems, and obtain competitive results (in terms of performance and interpretability) versus two selected state-of-the-art rule-based classification system, C4.5 and FURIA.**

Index Terms—**Data complexity metrics (DCM), evolutionary fuzzy system, fuzzy rule based classification system, metaclassifier.**

Manuscript received August 3, 2016; revised April 28, 2017 and March 15, 2018; accepted August 16, 2018. Date of publication August 23, 2018; date of current version March 29, 2019. This work was supported in part by the Spanish Ministry of Science and Technology under Grant TIN2016-77902-C3-1-P, Grant TIN2016-82013-REDT, Grant TIN2014-57251-P, and Grant TIN2015-68454-R; in part by the FEDER funds; and in part by the Junta de Comunidades de Castilla-La Mancha through Project SBPLY/17/18050/000493. The work of J. Cózar was supported by the MICINN under Grant FPU12/05102. *(Corresponding author: Javier Cózar.)*

J. Cózar and J. A. Gámez are with the Department of Computing Systems, University of Castilla-La Mancha, Albacete 02071, Spain (e-mail: javier.cozar@uclm.es; jose.gamez@uclm.es).

A. Fernándezis is with the Andalusian Research Institute on Data Science and Computational Intelligence, University of Granada, Granada 18071, Spain (e-mail: alberto@decsai.ugr.es).

F. Herrera is with the Andalusian Research Institute on Data Science and Computational Intelligence, University of Granada, Granada 18071, Spain, and also with the Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia (e-mail: herrera@decsai.ugr.es).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This includes datasets generation and DCM. The total size of the file is 38.9 KB.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2018.2866967

I. Introduction

Over the last decades, much effort has been invested into designing new classifiers. However, their performance is very dependent on the problem to solve. In fact, following the no free lunch theorem [1], [2], if all classifiers are evaluated using a big enough set of problems, all those classifiers would be equivalent.

When a fuzzy classifier is designed, it is usually evaluated using a set of problems with certain properties, i.e., imbalanced [3], [4], high-dimensional problems [5], [6] among others. In this way, we can guess a relation between the characteristics of the problem and the performance of the classifier. Related to it, data complexity metrics (DCMs) describe problem's properties that can be used to know in advance the behavior for each classifier [6]–[9]. These properties may focus on different aspects, as the class distribution, the level of overlapping between features, and so on. For example, a given metric can provide us information whether a problem can be solved by linear programming just computing the minimum sum of error distance of each dataset's point to a hyperplane, which separates these points into two groups or classes. If it equals 0 means that the problem can be solved with no error by simple linear programming. Therefore, this metric can be used as an indicator of the ease of a problem.

In [10] 12 of these metrics were used to discover intervals of good and bad behavior for a set of three classifiers. In other words, they discovered subspaces in the hyperspace of the 12 DCMs where a classifier performs good, bad, or unstable. These intervals might be used to extract Domains of Competence (DoC) for a classifier and derive usage rules which determine *a priori* its performance for a problem with the interval characteristics.

Our aim in this paper is to design a data complexity guided classifier based on the previous ideas. This process is divided into the following two steps.

1) First of all, we will start from a set of classifiers and we will analyze their behavior on different type of problems, where each type is described by DCMs.
2) With the former information, we will design a hierarchical rule decision system (HRDS) to decide which fuzzy classifier would have the best performance on a certain problem.

As a case study, we have selected the family of Fuzzy Association Rule-Based Classification model for high-dimensional (FARC-HD) problems classifiers, i.e., the original approach [11] and three extensions (IVTURS [12], IVTURS-Imb [13], and FARC-FW [14]) designed to focus on problems with specific characteristics. As a consequence, it is interesting to analyze the metaclassifier, called FAR metaclassifier (FAR-MC), to check whether the specializations of the extensions are used for their respective specific problems or not.

The rationale of the selection of fuzzy rule based classification systems (FRBCS) as baseline algorithms is based on two criteria: First, their good performance and interpretability in different contexts of applications; and second, they are models that can natively deal with the uncertainty of the data from real-world problems, leading to very robust classification systems. In addition, FARC-HD is a state-of-the-art algorithm which has proved to be a robust classifier in different scenarios [15], [16] as well as its variants [17], [18].

Our proposed metaclassifier approach can be embedded with any family of classifiers. However, the benefits of using interpretable models, such as fuzzy classifiers, add more advantages to the output model. It provides a simple yet powerful set of linguistic rules that provides a clearer description of the phenomena and, therefore, allows users and experts to easily understand the problem. We must point out that a metaclassifier is quite different from an ensemble: while the first one combines the individual outputs of multiple classifiers, the second just selects one classifier to predict. Therefore, the use of an ensemble of fuzzy classifiers affects drastically to the interpretability of the output model, whereas our proposed metaclassifier maintains the original one from the selected classifiers.

Finally, regarding the experimentation phase, we will use a large set of binary class problems (up to 421) to train and evaluate the performance of the proposed algorithm. We have applied a novelty procedure to split the group of datasets, called distribution-balanced DCM(DB), preserving the characteristic distribution of the problems. We have left 251 datasets for training and 170 for test. In the evaluation process, we have carried out a comparison between FAR-MC and its base classifiers. Also to keep in mind the upper bound of the performance of FAR-MC, we show the results of the perfect metaclassifier which always selects the best base classifier for each problem, called *Oracle*. To conclude, our experimental analysis is devoted to confirm the necessity of using the intrinsic data characteristics of a problem (complexity metrics) to determine the best suited single FRBCS to be applied. In order to carry out a fair comparison, we will contrast our FAR-MC proposal versus classifiers with similar properties and capabilities, namely C4.5 [19] and FURIA [20]. Motivated by the problem of imbalanced classification [22] in our experimental analysis we consider two different scenarios: using or not SMOTE [21] as a previous preprocessing step.

To sum up, the main contributions of this research can be enumerated as follows.

1) We make use of DCMs to generate intervals of behavior, based on [7] and [10], to understand the inner characteristics of the problems from which each classifier is better suited.
2) We build a metaclassifier based on an HRDS from the intervals of behavior. This current research supposes one step forward to the findings extracted in [10]. Specifically, apart from discovering the properties of the intervals of behavior, we combine this information to automatically compose an HRDS in order to generate a metaclassifier which benefits from the best behaviors of the individual classifiers. Therefore, it is able to automatically decide which is the most suitable fuzzy classifier to be applied to a given problem to achieve the highest performance.
3) We will use a family of fuzzy classifiers to build the metaclassifier, called FAR-MC.
4) Our conclusions are supported by a thorough experimental study using a large set of problems. For the validation, we have splitted it into train and test set of problems using DB-DCM, a procedure which preserves the characteristics of the datasets in each group, leading to more robust and reliable conclusions.

This paper is structured into five sections. In Section II, we define DMCs and present how these metrics can be used to define the behavior of different classifiers depending on the dataset characteristics. Afterward, we describe how we use these definitions to build FAR-MC. Then, in Sections III and IV, we describe the experimental framework and the study performed in this paper, respectively. Finally, in Section V, we summarize the conclusions and expose some future work on the topic. Furthermore, as complementary material,[1] we provide additional information about the used multiclass and derived binary datasets. It also includes information about characteristics of the datasets, which each base classifier process in the HRDS of FAR-MC, as well as the percentage that they represent in relation with the full set of train or test problems.

II. Meta-HRDSs to Design Robust Fuzzy Classifiers

In this research, we propose using a family of fuzzy classifiers in order to generate a metaclassifier that is able to outperform the single algorithms it is composed of. To do so, we describe the algorithms' behavior by means of DCMs, and then, we use this information to generate a meta-HRD.

This section is divided as follows. First, in Section II-A, we describe DCMs and how these metrics can be used to categorize problems based on their inner characteristics. Then, we explain how these characteristics are used to generate the DoC (good, bad, and unstable behavior) for a set of classifiers. Afterward, in Section II-B, we adapt the usage of an automatic software to generate the aforementioned DoC to our requirements. Finally, in Section II-C, we detail the procedure to generate the meta-HRDS used by FAR-MC based on the DoC.

A. Describing Algorithms' Behavior by Means of DCMs

DCMs are measures that characterize datasets, i.e., the difficulty of a classification problem [23]. The nature of dataset properties can vary, so it does the definition of DCMs. For example, some problems have nonzero Bayes error [7]. Others have a complex decision boundary and/or subclass structures. Certain problems have a high-dimensional feature space and sparseness of available samples which lead to estimation difficulties, etc.

Ho and Basu [7] focused on a set of 12 geometrical characteristics of the class distributions, as they support that these are more discriminant than other metrics for classification problems.

[1] http://simd.albacete.org/supplements/FARMC.htm

TABLE I
DATA COMPLEXITY MEASURES

Type	Id.	Description
Measures of overlaps in feature values from different classes	F1	Maximum Fisher's discriminant ratio
	F2	Volume of overlap region
	F3	Maximum (individual) feature efficiency
Measures of separability of classes	L1	Minimized sum of error distance by linear programming
	L2	Error rate of linear classifier by linear programming
	N1	Fraction of points on class boundary
	N2	Ratio of average intra/inter class NN distance
	N3	Error rate of 1NN classifier
Measures of geometry, topology and density of manifolds	L3	Nonlinearity of linear classifier by linear programming
	N4	Nonlinearity of 1NN classifier
	T1	Fraction of points with associated adherence subsets retained
	T2	Average number of points per dimension

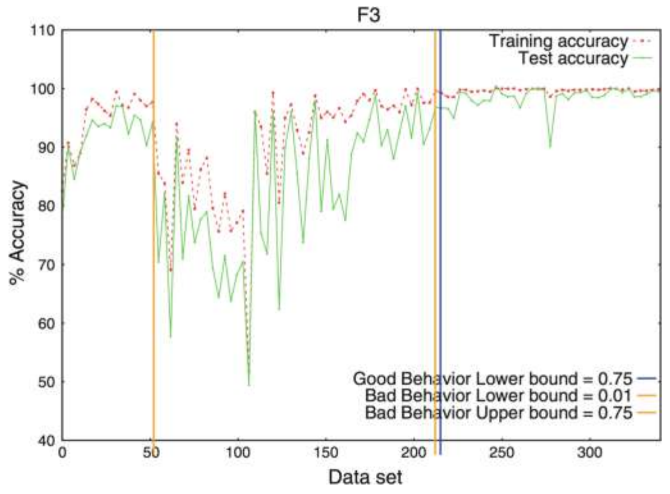

Fig. 1. Accuracy of SVM for problems sorted by F3 DCM.

"**If** DCM$_i \in [a, b]$ **then** the behaviour of classifier C **is** *good/bad/not characterised*"

Fig. 2. Form of the rules to characterize the behavior of a classifier.

This set of DCMs was divided into three blocks (see Table I). The first one contains DCMs which measures the overlaps in feature values from different classes. The second measures the separability of classes. Finally, the last block is formed by measures of geometry, topology, and density of manifolds.

To test if these metrics describe well or not the difficulty of a classification problem, in [7] they treat each dataset as a point in a twelve-dimensional hyperspace, and examined the distribution of these points in this space by the density plots and pairwise scatter plots for interesting structures. They employ a set of 944 binary class (real and synthetic) problems, where some of the synthetic are random noise (they assign a random class to each instance). First, they conclude that the distribution of real-world problems is significantly different from that of random noise. Therefore, real-world problems have learnable structures that can be used to describe them. Regarding the difficulty of a classification problem, they found that there exist structures in the twelve-dimensional hyperspace that reveal the intricate relationships among the factors, which affects the difficulty of a problem.

However, the performance achieved by a classifier is dependant on both the difficulty of a problem and the classifier. In [10] the authors use this set of DCMs to describe the characteristics of the datasets in order to identify regions of good, bad, and not characterized behavior for different classifiers. The objective is to know *a priori* if a certain classifier would perform well or poorly for a specific problem.

The main idea behind the performance prediction is based on the relation between the DCMs of a problem and the accuracy obtained with the classifier. To better understand this concept, they show plots where problems (in *x*-axis) are sorted by a specific DCM and the accuracy is in the *y*-axis. One of these plots is shown in Fig. 1, which depicts an example of the former behavior for the DCM F3, in which there is a region defined by values of this metric between 0.01 and 0.75 where the accuracy obtained by the support vector machine (SVM) classifier is unstable and in most cases below 90%. On the contrary, for F3 values upon 0.75 the accuracy is more stable and generally over 95%.

For each classifier, there can be more than one region of good/bad/not characterized behavior, as they are using several DCMs. In order to guess the performance of a classifier for a certain problem, it is necessary to combine all this information. For example, a problem may have a low value of L1, where the classifier SVM behaves good, and a low value of F1, where its behavior is bad. Ho and Basu [7] performs deriving one rule per good, bad behavior, and not characterized regions with the form depicted in Fig. 2, where DCM$_i$ refers to one of the used DCMs.

Then, all the good behavior rules are combined into a single one using the *or* operator. They call this rule positive rule disjunction (PRD). Similary, they do the same with the bad behavior rules, calling this rule negative rule disjunction (NRD). The PRD and NRD rules may present overlapping in their support (the problems that they cover). However, mutually exclusive description of the good and bad regions is desirable in order to estimate the behavior of the classifier. In order to tackle this issue, they consider the conjunctive operator *and* ($\wedge$) and the difference operator *and not* ($\wedge^{\neg}$) between the PRD and NRD rules. After analyzing different combinations of PRD and NRD rules using these two operators, they conclude that good behavior regions are described directly by the rule "PRD," bad regions are described by the rule "NRD $\wedge^{\neg}$ PRD," and not characterized regions are described by the rule "*not* PRD and *not* (NRD $\wedge^{\neg}$ PRD)". To check the behavior of these rules, they show some pictures that show the accuracy for a classifier for each group of datasets described by the good and bad behavior

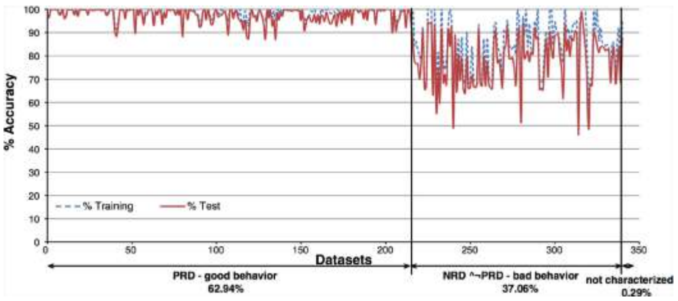

Fig. 3. Accuracy of SVM for problems grouped by PRD and NDR $\wedge\neg$ PRD.

and the noncharacterized region. One of these figures, for the SVM classifier and the set of 340 training problems, is shown in Fig. 3.

In [10], a software tool was also developed for the automatic extraction method of the DoC,[2] called *ComplexityRuleExtraction*. This software generates the intervals for good, bad behaviors, and not characterized regions for each DCM. At the same time, it describes which datasets match with each rule. The main outline of the automatic extraction method is described in Fig. 4. It manages four definitions, two for good and bad behavior elements (Definitions 1 and 2), and other two for intervals of good and bad behavior (Definitions 3 and 4), where $\overline{U}^{\text{tra}}$, $\overline{U}^{\text{tst}}$ and $\overline{U}^{\text{diff}}$ refers to the mean training, test and training minus test accuracy for the whole set of problems ($u_i \in U$), and $\overline{V}^{\text{tra}}$, $\overline{V}^{\text{tst}}$ and similary $\overline{V}^{\text{diff}}$ refers to the mean training, test and training minus test accuracy for the datasets in the interval V ($u_i \in V$). Also, this software requires two input parameters, *minGoodElementTest* and *threshold*, which refers to the minimum accuracy level for a good behavior element and the improvement required in terms of the mean test accuracy for an interval of datasets against the mean test accuracy for the whole set of problems.

Definition 1: A **good behavior element** u_i is such that
1) $u_i^{\text{test}} \geq$ minGoodElementTest; and
2) $u_i^{\text{tra}} - u_i^{\text{tst}} \leq \overline{U}^{\text{diff}}$

Definition 2: A **bad behavior element** u_i is such that
1) $u_i^{\text{test}} <$ minGoodElementTest; and
2) $u_i^{\text{tra}} - u_i^{\text{tst}} > \overline{U}^{\text{diff}}$

Definition 3: An **interval of good behavior** $V = \{u_i, \ldots, u_j\}$ is such that
1) $\overline{V}^{\text{diff}} \leq \overline{U}^{\text{diff}}$; and
2) $\overline{V}^{\text{tst}} \geq \overline{U}^{\text{tst}} +$ threshold; and
3) $\forall u_j \in V; u_j^{\text{tst}} \geq$ minGoodElementTest

Definition 4: An **interval of bad behavior** $V = \{u_i, \ldots, u_j\}$ is such that
1) $\overline{V}^{\text{diff}} > \overline{U}^{\text{diff}}$; and
2) $\overline{V}^{\text{tst}} < \overline{U}^{\text{tst}} -$ threshold

In order to extract the good, bad behavior, and not characterized intervals a bottom-up process is followed. First, the algorithm arranges the datasets in U based on the values of one of the twelve DCMs (CM_j), generating a sorted list U_{CM_j}. Afterward, this list is explored from the lowest to the highest value

[2]http://sci2s.ugr.es/DC-automatic-method

```
INPUT: A list of datasets $U = \{u_1, u_2, \ldots, u_n\}$. Each dataset $u_i$ has associated a tuple T containing the training and test accuracy values for a particular learning method and its 12 data complexity values.
OUTPUT: A set of intervals $G$ in which the learning method shows good behaviour, and a set of intervals $B$ where the learning method shows bad behaviour.
$G \leftarrow \{\}$
$B \leftarrow \{\}$
for each $CM_j \in$ DCMs do
    //Sort the list $U$ by each data complexity measure $CM_j$
    $U_{CM_j} \leftarrow$ sort($U$,$CM_j$)
    //Search for good behaviour intervals
    $i \leftarrow 1$
    while $i < n$ do
        pos $\leftarrow$ nextImportantGoodPoint($u_i$,$U_{CM_j}$)
        if pos $\neq -1$ then
            $V \leftarrow$ extendGoodInterval(pos,$U_{CM_j}$)
            $G \leftarrow G \cup \{V\}$
            $u_i \leftarrow M_{up}(V)$
        end if
    end while
    //Search for bad behaviour intervals
    $i \leftarrow 1$
    while $i < n$ do
        pos $\leftarrow$ nextImportantBadPoint($u_i$,$U_{CM_j}$)
        if pos $\neq -1$ then
            $V \leftarrow$ extendBadInterval(pos,$U_{CM_j}$)
            $B \leftarrow B \cup \{V\}$
            $u_i \leftarrow M_{up}(V)$
        end if
    end while
end for
//Merge and filter the intervals if necessary
$G \leftarrow$ mergeOverlappedIntervals($G$)
$G \leftarrow$ dropSmallIntervals($G$)
$B \leftarrow$ mergeOverlappedIntervals($B$)
$B \leftarrow$ dropSmallIntervals($B$)
return $\{G, B\}$
```

Fig. 4. Automatic Extraction Method.

of CM_j: when a good or bad behavior element $u_i \in U_{CM_j}$ is found (Definitions 1 and 2), the exploration stops and considers such element as an initial interval $V = u_i$. This interval is extended by adding adjacent elements to u_i while such interval verifies the Definitions 3 or 4 accordingly.

Once all the possible intervals have been extracted, a generalization process is applied in order to merge intervals of the same type which are overlapped or slightly separated. Finally, the algorithm runs a filtering process to remove nonsignificant intervals (which contain a low number of elements). The regions that have not been labeled as good or bad behavior are the noncharacterized intervals.

B. Automatic Method to Obtain the DoC

To extract the DoC for each classifier we will adopt the methodology proposed in [10], which was described in Section II-A.

The concept of DoC for a certain classifier is different in this paper, due to our aim is to design an HRDS which is able to determine which classifier performs better than the others. That means the performance of a classifier can be poor but the best among the rest. Therefore, we define a score value which contrasts the quality of the classifiers among themselves and we use it to define the DoC. This matter implies the following two changes.

1) Score instead of accuracy as input performance metric.
2) Parameters needs to be adapted to the score.

Regarding the first point, one option could be to use the ranking (using the accuracy, area under the ROC curve, or other performance metric), as it gives us information about their relative performance, being 1 the best measurement and n the worst, being n the number of classifiers. However, this approach performs poorly because it loses information about the relative difference in terms of performance. The strategy adopted in this paper consists on using the difference of the performance between a classifier and other labeled as the default classifier. For instance, let C_b be the default classifier and the individual performance of each classifier for a problem p be $m_p^{C_1}$, $m_p^{C_2}, \ldots, m_p^{C_n}$. Then, the score is the difference $m_p^{C_i} - m_p^{C_b}$ for $i = 1, \ldots, n$. In Section II-C, we will detail how we select the default classifier.

The ComplexityRuleExtraction software uses two parameters as inputs: *minGoodElementTest* and *threshold*. Their interpretation are, respectively, the minimum performance of an element to be considered good and the mean improvement for a set of problems compared to the mean performance for all the datasets to be considered as a domain of competence (interval of good behavior). Because the score has a different domain, we cannot use the parameterization recommended by the authors. Instead, we will explore different configurations for both parameters.

C. Metaclassifier Hierarchical Rule Extraction Method

Once the DoC have been obtained for all the selected classifiers, we design an HRDS to decide *a priori* which classifier is the best suited one given the characteristics of a certain problem.

As the DoC might overlap between classifiers, the priority order is crucial. In order to determine the classifier's priority order, we will evaluate all the possible combinations and we will choose the best. The number of possible combinations is $(n-1)!$, as the base classifier is not taken into account because it is always the last classifier in the hierarchical rule system (it is the default classifier). However, n should be a small number (four in our case) so the number of combinations should be small and easily tackled by any conventional computer.

In addition, the impact of the base classifier on the mean performance is a constant, as it will be used for the datasets out of the DoC of the other classifiers independently of the order. Therefore, it is not necessary to evaluate its performance on these remaining datasets while searching for the best classifier's priority order, which supposes a lower computational effort.

For the selection of the default classifier, we have designed a wrapper algorithm that evaluate the HRDS obtained with each classifier as the base one, and selects the best one according to the mean performance metric. The pseudocode is shown in Fig. 5.

```
1: INPUT: $\mathcal{D}$, classifiers, $mget$, $th$
2: OUTPUT: HRDS
3: bestPerformance ← $-\inf$
4: for all c ∈ classifiers do
5:    baseClassifier ← c
6:    for i = 1 to $(|classifiers| - 1)!$ do
7:       ordClassifiers ← getOrder(i, classifiers - {c})
8:       DoC ← getDoC(ordClassifiers, $mget$, $th$)
9:       performance ← evaluateRDS(DoC, $\mathcal{D}$)
10:      if performance > bestPerformance then
11:         bestPerformance ← performance
12:         bestDoC ← DoC
13:         bestBaseClassifier ← c
14:      end if
15:   end for
16: end for
17: HRDS ← generateHRDS(bestDoC, bestBaseClassifier)
18: return HRDS
```

Fig. 5. Algorithm to generate the best HRDS.

The inputs are a set of datasets, the classifiers, and the two parameters for the ComplexityRuleExtraction software, *minGoodElementTest,* and *threshold*. In the following and for readability reasons, these two parameters are renamed as *mget* and *th*, respectively. The output is the HRDS.

First, all the classifiers are tested as the base one. Then, it evaluates the DoC using the ComplexityRuleExtraction software testing all the orderings for the remaining classifiers. From the best configuration of DoC and base classifier (among all the evaluated configurations) it builds the HRDS appending the base classifier to the end of the system.

III. Case Study Based on the FARC-HD Family: Experimental Framework

In this section, we design FAR-MC, a case study for the meta-HRDS based on the FARC-HD family. First, in Section III-A, we describe the evolutionary fuzzy systems (EFS) [18] used to learn the decision system. Then, in Section III-B, we describe the datasets used to generate and validate FAR-MC. Furthermore, we describe DB-DCM, the strategy used to perform the split the datasets into the train and test sets of problems. Afterward, in Section III-C, we justify the selection of the performance metric and we describe the statistical tests used for the evaluation. Finally, in Section III-D, we detail the parameterization used for the different methods.

A. Family of FARC-HD Algorithms: Standard Approach and Current Extensions

FRBCS are highly interpretable models that can also deal with the imprecision associated with real-world data acquisition. When the data used to build these models consist of a high number of variables and/or instances, the learning process suffers from exponential growth of the fuzzy rule search space. Also to generate the database definition (which contains the fuzzy partition for the variables of the problem) becomes a complex task which have a huge impact in the performance of the classifier.

In such complex scenarios evolutionary algorithms are very suitable and usually lead to robust solutions. EFS carries out a global search, evolving simultaneously the rulebase and the database definition. A well-known state-of-the-art EFS is FARC-HD (from FARC for high-dimensional problems) [11]. In addition to its scalability and robustness, we have selected it because there exist a family of classifiers, variants of FARC-HD, focused on solving different classification contexts.

In the following Sections, we will describe the basics of FARC-HD and its variations.

1) Baseline FARC-HD: FARC-HD is a fuzzy association rule-based classification algorithm for high-dimensional problems [11]. It starts from a predefined fuzzy partition, and builds a set of candidate rules. This process is done building a search tree to list all the possible frequent fuzzy item sets, which corresponds directly to the antecedent of a candidate rule.

However, dealing with the whole set of candidate rules is impracticable even for small problems. In order to reduce the number of candidate rules, it selects the most important based on their support, which measures their coverage with respect to the data. To carry out this process efficiently, the search tree is pruned based on the *a priori* principle [24]. If a fuzzy item set is not frequent (its support does not reach a minimum support threshold), all the item sets derived from it by adding a fuzzy predicate are not frequent either, so there is no need to calculate their support and this branch of the search tree can be pruned.

Moreover, one of the main characteristics of FRBCSs is the interpretability, which is dramatically reduced when using rules with a high number of terms in the antecedent. In order to generate a tractable and interpretable set of candidate rules, the number of antecedents can be also limited to a maximum (by limiting the maximum depth of the tree).

In a second phase, it reduces even more the number of candidate rules though a process called prescreening. This is done because the number of candidate rules might sill be huge for the subsequent search algorithm. In order to retain only the best candidate rules, it follows a weighted instance scheme where iteratively the best candidate rule is selected, and weights associated with patterns are updated for the next iteration. It stops when all the patterns are covered by more than k_t rules.

Finally, in the third phase, it applies an evolutionary algorithm to select a subset of the candidate rules to be present in the rule base, and to tune the membership functions in the data base.

2) Interval Valued Fuzzy Reasoning Method with Tuning and Rule Selection (IVTURS): One of the most important points in the definition of FRBCSs is the membership functions of the fuzzy variables. This is a difficult task due to the uncertainty related to their definition. Interval-valued fuzzy sets [25] allows to model the ignorance definition of the fuzzy terms [26], as it provides an interval (instead of a single number) as the membership degree of each element to this set. In [12] interval-valued fuzzy sets are used to define the membership functions.

IVTURS starts from an initial FRBCS generated by means of the base classifier FARC-HD, and then adapts the definition of the membership functions to use the interval-valued fuzzy sets. Finally, it uses a genetic algorithm to tune the definition of the interval-valued fuzzy sets and to perform a rule selection process. As the partition of the variables does not use classical fuzzy sets, the reasoning method has been extended to deal with this type of fuzzy sets.

Apart of the improvement in the design of the FRBCSs, it uses more information in the membership function definitions. Therefore, it is expected to deal better with problems where the density of manifolds is high, but the output of that instances are slightly different (as it considers the uncertainty of the membership degrees for that instances).

3) IVTURS-Imbalanced: Imbalanced problems have received a special interest in the last decades as a large number of real-world datasets suffer from this problem. Imbalanced datasets refer to problems where one or more classes are represented by a large number of examples [known as majority class(es)] while the other class(es) are represented by only a few examples [known as minority class(es)] [22]. This unbalanced distribution leads the classifier to predict the examples as one of the majority classes, completely ignoring the minority ones.

IVTURS-Imbalanced [13] is designed to cope with imbalanced problems. It is a modification of the previous IVTURS algorithm. The learning process is similar to the one described in [12], but adding a new method just before applying the evolutionary algorithm to select a subset of the candidate rules and tune the membership functions. This method rescales the rule weights of the generated rule base in order to avoid low confidence levels of rules for the minority class. Also, the inference process has been modified to predict instances which do not fire any rule. In this case, instead of using a default prediction rule (as in [11]), it uses a weighted combination of the most suitable rules in the rule base to classify the uncovered instances.

4) Overlapping Classes: FARC-FW: The problem of overlapping or class separability refers to regions where similar number of instances of both classes (in binary classification problems) are present. This issue is directly proportional to the hardness of classifying a problem, i.e., any linearly separable dataset (absence of the overlapping problem) can be addressed by a naive classifier, regardless the class distribution [27].

In [14], the FARC-HD algorithm is adapted to deal with class separability problems. In order to do that, it assigns weights to input variables to allow giving more importance to some variables, which suffers to a lesser extent the problem of overlapping. In order to learn the best combination of weights, they use a wrapper approach, in which for each combination of weights they apply the evolutionary algorithm used in FARC-HD.

B. Datasets: Characteristics and Validation Procedure

The existing metrics to characterize the domain of competence are designed only for binary class datasets. Also, and in order to obtain good DoC, we need as many datasets as possible and with different characteristics. In order to do that, we have followed the same strategy used in [10], taking a large number of multiclass datasets and deriving a set of binary datasets from

TABLE II
STATISTICAL INFORMATION FOR THE DCMS FOR ALL THE USED BINARY DATASETS

DCM	mean	s.d.	min	max
F1	2.3251 ±	3.7874	0.0019	50.9500
F2	0.2277 ±	0.3411	0.0000	1.0000
F3	0.7106 ±	0.6982	0.0000	1.9970
N1	0.2163 ±	0.1803	0.0018	0.7440
N2	0.4448 ±	0.2488	0.0112	1.0240
N3	0.1308 ±	0.1388	0.0000	0.5500
N4	0.1548 ±	0.1376	0.0000	0.4988
L1	0.5690 ±	0.4186	0.0738	4.4810
L2	0.2018 ±	0.1301	0.0000	0.4940
L3	0.3229 ±	0.1969	0.0000	0.5042
T1	0.9304 ±	0.0966	0.2100	1.0000
T2	78.8510 ±	190.1388	8.2310	1458.0000

them avoiding those which are linearly separable. These binary problems have been generated from pairwise combinations of the classes. In order to obtain additional datasets, this methodology has been also applied grouping the classes by pairs.

Moreover, we have made a selection of the problems used in [10] limiting the number of predictive variables to a maximum of 15. This decision was taken since the classifier FARC-FW is very time-consuming in terms of dimensionality. In addition to these datasets, we have considered four new problems: *haberman*, *optdigits*, *pima,* and *shuttle*.

The number of attributes of the multiclass problems ranges from 3 to 53, and the number of classes from 2 to 28. From the derived 74994 binary datasets, only 481 are used after the filtering process. For more information, Table I in the complementary website shows the number of attributes and classes for the multiclass problems, as well as the derived and used binary class datasets (*Derived b.ds.* and *Used b.ds.,* respectively). Regarding the used binary problems, we also noted their characteristics in terms of DCMs in Table II.

The validation process is used to measure how well a method generalise when new input data is received. In data mining, the validation process usually consists on dividing the dataset into the training and test datasets. Then, the method is build using the training dataset, and its performance is assessed over the test dataset. In this paper, we have two levels of validations.

1) Performance of a single classifier on a single dataset. Focused on validating the performance of the base classifiers during the FAR-MC construction process.
2) Performance of a single classifier on a set of datasets. Focused on validating the performance of FAR-MC on a set of "unseen" problems.

To evaluate the performance of a classifier on a dataset we have used a tenfold cross validation. It is a commonly used strategy that divides the dataset in k folds (10 in this case) and performs k evaluation processes, training with $k-1$ folds and testing with the remaining (each time, the test fold is different). We have ran three times (using the number of execution as seed) the tenfold cross validation, and the average of the 30 executions is reported.

In the second level, to evaluate the performance of the meta-classifier FAR-MC, we split the datasets into training (D^{train}) and test (D^{test}) sets. The way we make this partition is a key factor in both modeling the classifier and testing its performance. We propose to use a similar set of training and test datasets in terms of DCM distributions. In this way, we avoid a different data complexity metric distribution between the training and test datasets that could lead to erroneous conclusions.

```
1: **Input:** points = $\{p^1,\ldots,p^m\}/p^i=(p^i_1,\ldots,p^i_{n_i})$
2: **Output:** folds = $\{f_1,\ldots,f_k\}$
3: $k \leftarrow 10$
4: $f_i \leftarrow \emptyset, \forall i \in 1$ **to** $k$
5: points $\leftarrow$ NormalizeDomains(points)
6: meanPoint $\leftarrow (mp_1,\ldots,mp_n)/mp_i=\overline{p_i}$
7: distances $\leftarrow \{\sum_{j=1}^{n_i}(p^i_j-mp_j)^2 \forall i \in \{1,\ldots,m\}\}$
8: $idx \leftarrow \underset{i}{\text{argmax}}\, d_i \in$ distances
9: $fold \leftarrow 0$
10: **while** points $\neq \emptyset$ **do**
11:     $f_{fold} \leftarrow f_{fold} \cup \{p^{idx}\}$
12:     points $\leftarrow$ points $-\{p^{idx}\}$
13:     distances $\leftarrow \{\sum_{j=1}^{n_i}(p^i_j-p^{idx}_j)^2 \forall i \in \{1,\ldots,m\}\}$
14:     $idx \leftarrow \underset{i}{\text{argmax}}\, d_i \in$ distances
15:     $fold \leftarrow (fold+1) \mod k$
16: **end while**
```

Fig. 6. Algorithm to split the datasets into training and test sets.

Our strategy, called DB-DCM, splits datasets into k folds for classification problems. The main idea is to stratify the datasets in k folds. The pseudocode is depicted in Fig. 6.

First of all, we normalize the domain of the variables in the range $[0,1]$. Afterward, we select an initial point and iteratively the nearest unassigned neighbour is assigned to the next fold until all the points are assigned. At the end of this process, we have a number of folds which contains a similar distribution of points. We have chosen a number of folds of ten, six for training, and four for test.

Respect to the initial point, instead of choosing it randomly, we select the point whose distance to the mean point of the point cloud is maximum [see Fig. 7(a)]. That way, we ensure it is a point in the cortex of the point cloud. This is better than to be in the center because the latest points will be very different as the distances between them would be greater, i.e., in Fig. 7(b) and (c) we show the paths using the farthest and mean point, respectively.

Next, we show in the Table III the statistics of each DCM in the training and test set using the previous algorithm (Table III). As we can see, the values are quite similar to those in Table II, which indicates a good stratification (only for the metric F1 we can observe some differences).

C. Selection of a Performance Metric and Statistical Tests for Experiment Validation

The evaluation criterion has a direct impact on the study, as it is used to evaluate the classification performance and also to guide the classifier modeling. The accuracy metric is a combination of the values of the confusion matrix, as shown in Table IV, and is one of the most used in classification [see (1)]. However,

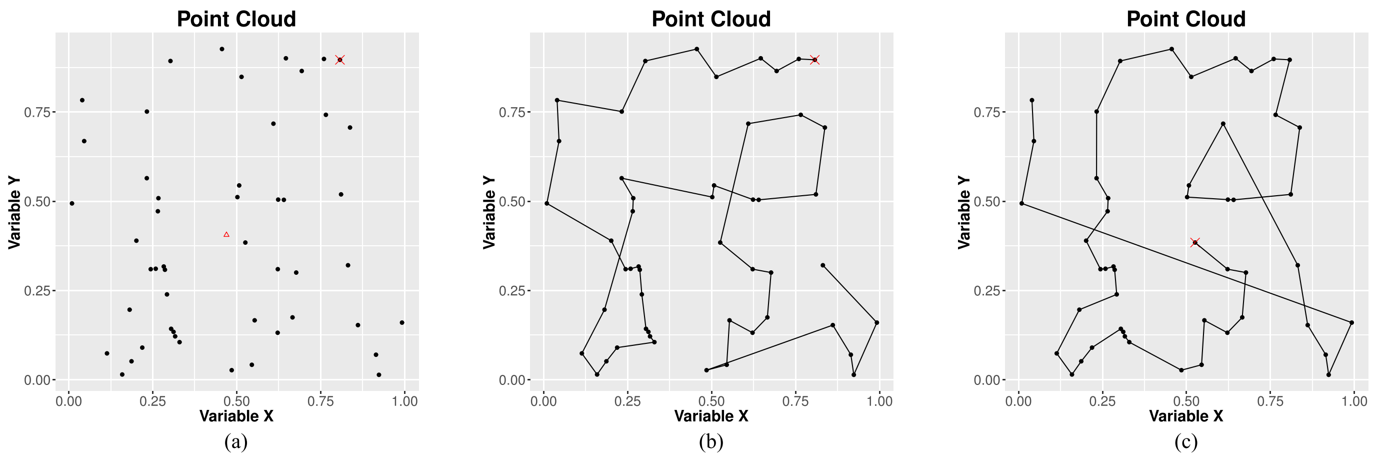

Fig. 7. Example of different initial points for the DCMs stratiffication process. (a) Example of point cloud, mean and starting point (red triangle and cross) for the algorithm. (b) Point selection order using the farthest point from the mean point as starting (red cross). (c) Point selection order using the closest point from the mean point as starting (red cross).

TABLE III
STATISTICAL INFORMATION FOR THE DCMs FOR SPLITTED SETS OF THE BINARY PROBLEMS

DCM	mean	s.d.	min	max
F1	2.4951 ±	4.4888	0.0019	50.9500
F2	0.2247 ±	0.3382	0.0000	1.0000
F3	0.7092 ±	0.7013	0.0000	1.9970
N1	0.2134 ±	0.1778	0.0022	0.7440
N2	0.4460 ±	0.2461	0.0112	1.0240
N3	0.1300 ±	0.1383	0.0000	0.5476
N4	0.1525 ±	0.1354	0.0000	0.4988
L1	0.5652 ±	0.4364	0.0784	4.4810
L2	0.2000 ±	0.1300	0.0000	0.4911
L3	0.3230 ±	0.1982	0.0000	0.5042
T1	0.9314 ±	0.0954	0.2100	1.0000
T2	75.8409 ±	176.1891	8.2310	1298.0000

(a) Data Complexity Metrics for the training binary datasets.

DCM	mean	s.d.	min	max
F1	2.0741 ±	2.3810	0.0066	16.4300
F2	0.2321 ±	0.3453	0.0000	1.0000
F3	0.7127 ±	0.6937	0.0000	1.9970
N1	0.2206 ±	0.1838	0.0018	0.7400
N2	0.4431 ±	0.2527	0.0129	0.9901
N3	0.1321 ±	0.1395	0.0000	0.5500
N4	0.1583 ±	0.1408	0.0000	0.4986
L1	0.5746 ±	0.3906	0.0738	3.8880
L2	0.2046 ±	0.1302	0.0000	0.4940
L3	0.3227 ±	0.1950	0.0000	0.5000
T1	0.9291 ±	0.0983	0.3163	1.0000
T2	83.2954 ±	208.9653	9.1540	1458.0000

(b) Data Complexity Metrics for the testing Binary Datasets.

TABLE IV
CONFUSION MATRIX FOR A BINARY CLASS PROBLEM

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

this metric does not take into account the class distribution. In this paper, we deal with problems with different ratios between the majority and minority class (from 1 to more than 23). In this framework accuracy might lead to erroneous conclusions since the minority/negative class has little impact on accuracy compared to the majority/positive class [13]. As an example, for a dataset whose imbalanced ratio (IR, which is the ratio between the majority and minority class) is equal 9, a naive classifier which classifies all the examples as negative would achieve an accuracy of 0.9

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}. \tag{1}$$

For imbalanced datasets, which are problems with an unbalanced distribution between the majority and minority class, it is more appropriate to use metrics which take into account the class distribution [13], [28].

In this paper, we will use the area under the ROC curve (AUC) as the performance metric, which is commonly used in imbalanced problems. AUC combines the true positive and false positive rates [28] [see (2)], where the TP_{rate} is the percentage of positive instances correctly classified ($\frac{\text{TP}}{\text{TP}+\text{FN}}$) and the FP_{rate} is the percentage of negative instances misclassified ($\frac{\text{FP}}{\text{FP}+\text{TN}}$)

$$\text{AUC} = \frac{1 + \text{TP}_{\text{rate}} - \text{FP}_{\text{rate}}}{2}. \tag{2}$$

For the sake of complementing the analysis of FAR-MC versus the state-of-the-art classifiers, we will take into account the values of the precision and the recall [29], which are also widely used in imbalanced domains. Specifically, we will combine both values into the well-known F1-score [see (3)] as a single measure that balances both precision and recall, thus, allowing us to assess the quality of our proposed methodology from an additional point of view

$$\text{F1-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

For the sake of clarity, in the following we will use D^{train} and D^{test} for the sets of train and test datasets, $\text{AUC}^{\text{train}}_{\text{train}}$ and $\text{AUC}^{\text{train}}_{\text{test}}$ for the train and test AUC metric for the datasets of the training set of problems, and similary, $\text{AUC}^{\text{test}}_{\text{train}}$ and $\text{AUC}^{\text{test}}_{\text{test}}$ for the train and test AUC metric for the datasets of the test set of problems.

To evaluate the performance of FAR-MC we have divided the validation process in two comparisons. First of all, we will compare the metaclassifier FAR-MC versus the FARC-HD and variants. Afterward, we will compare the performance of FAR-MC and the state-of-the-art classifiers C4.5 [19] and FURIA [20] in two variants: by using and not the preprocessing technique SMOTE [21], which is an oversampling technique commonly used to deal with imbalanced problems [22], [30].

For each comparison, we carry out a standardized methodology composed of a sequence of statistical tests described in [31], and then extended in [32]. The statistical study pipeline consists on the following steps. First of all, we will apply a Friedman test [33] to check the differences between the evaluated classifiers. Afterward, we apply a set of paired Wilcoxon signed rank tests [34] between the best classifier (in terms of best mean ranking) and the rest. In order to reduce the familywise or type 1 error, we apply a p-value correction using the Holm's procedure [35].

It is worth pointing that FAR-MC can never outperforms all other FARC-HD familywise classifiers, as at least it will draw with another. As a consequence, we have used the version of the Wilcoxon signed rank test described in [36] and also recommended in [31] which is able to deal with draws, splitting the ranks for ties evenly among the statistics R^+ and R^-.

D. Parameterization Setup

Each base classifier is already implemented in the software KEEL [37]. It is a tool which provides a way to design experiments with different datasets and computational intelligence algorithms. For each classifier, we will use the default configuration for the parameters in such software tool. The parameters configuration shared for all the classifiers are *nLabels*=5, *minSup*=0.05, *minSup*=0.8, *depth*=3, *k*=2, *popSize*=50, *bitsgen*=30, and *FRM*=Additive. The rest of the parameters are, respectively for FARC-HD, IVTURS, IVTURS-Imb and FARC-FW, *maxTrials*={15000, 15000, 20000, 1000 + 15000}, and *alpha*={0.15, 0.15, 0.2, 0.15}.

In the case of the state-of-the-art algorithms, C4.5, FURIA and the preprocessing technique SMOTE, also use the default parameterization. In the case of FURIA, the number of optimizations is 2 and the number of folds is 3. For C4.5, the confidence level will be set at 0.25, with 2 being the minimum number of item sets per leaf, and the application of pruning will be used to obtain the final tree. SMOTE configuration will also be the standard with a 50% class distribution, 5 neighbors for generating the synthetic samples, and heterogeneous value difference metric for computing the distance among the examples.

In the case of the parameters for the automatic domain of competence extraction, we have used seven possible values for the minimum performance parameter $mget$ and two possible values for th per each $mget$: the same value as $mget$ and this value divided by 10. Hence, we have used fourteen pair of values for *mget* and *th* which are shown in Table V.

IV. Building a Metaclassifier: Experimental Analysis With FAR-MC

This section is divided in three blocks. In Section IV-A, we build the metaclassifier FAR-MC using D^{train} and analyze its behavior using the same set of problems. Afterward, we evaluate the performance of FAR-MC using the set of test datasets D^{test}. In Section IV-B, we analyze the performance of FAR-MC comparing it against FARC-HD and its variants. Finally, in Section IV-C we compare FAR-MC versus the state-of-the-art FURIA and C4.5 classifiers, the last one with and without applying the preprocessing technique SMOTE.

In addition, experiments will include the results of an ideal metaclassifier called *Oracle*, which always select the best base classifier for each problem. It is useful to keep in mind the best reachable results of FAR-MC, so we can check its performance knowing its upper bound.

TABLE V
Parameterization for the Automatic Domain of Competence Extraction Software

$mget$	$5e^{-4}$	$1e^{-3}$	$2.5e^{-3}$	$5e^{-3}$	$7.5e^{-3}$	$1e^{-2}$	$1.5e^{-2}$
th	/1, /10	/1, /10	/1, /10	/1, /10	/1, /10	/1, /10	/1, /10

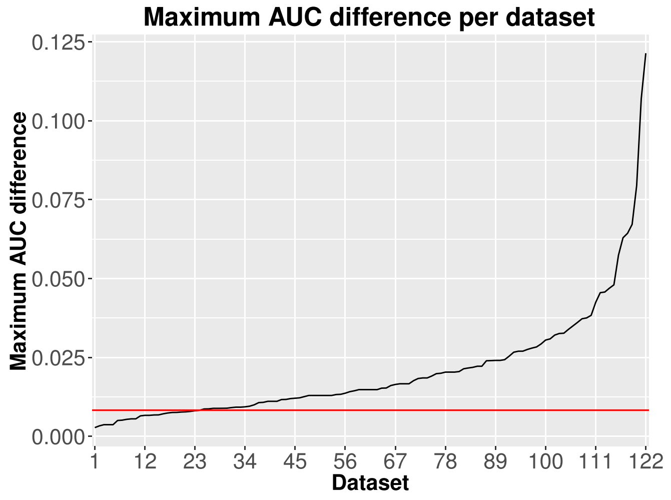

Fig. 8. Difference between the best and worst classifier in terms of test AUC and the cut level for discarded datasets.

A. Building the Metaclassifier (Using D^{train})

Here, we use the training datasets D^{train} to build the HRDS of the metaclassifier FAR-MC. In Section IV-A1, we will present a problem related with datasets whose performance is similar for all the base classifiers. In order to avoid such problem, we will first apply a preprocessing technique to filter out those problematic datasets. After that, in Section IV-A2, we perform the construction of the HRDS following the algorithm described in Fig. 5. Then, in Section IV-A3, we make an overview of the performance of FAR-MC using the same set of problems D^{train} used to build the metaclassifier.

1) D^{train} filtering: The DoC represent the relative good behavior, which refers to the performance compared to a fixed default classifier. If we analyze the performance differences, we may observe that for some problems the differences between the best and worst AUC are quite small (see Fig. 8). In other words, the performance of all the FARC-HD familywise classifiers are very similar. These datasets do not provide us useful knowledge about which classifier is the best, so a percentage of them (whose

TABLE VI
SUMMARY RESULTS FOR ALL THE TRAINING DATASETS

classifier	train ± s.d.	test ± s.d.	hits
FARC-FW	0.9395 ± 0.009	0.8685 ± 0.072	0.20
FARC-HD	0.9328 ± 0.009	0.8675 ± 0.070	0.23
IVTURS	0.9279 ± 0.009	0.8688 ± 0.069	0.27
IVTURS-Imb	0.9423 ± 0.008	0.8754 ± 0.069	0.37
Oracle	0.9425 ± 0.008	0.8845 ± 0.065	1.00

(a) Training and test AUC for all the training datasets.

	FARC-FW	FARC-HD	IVTURS	IVTURS-Imb	Oracle
FARC-FW	0/251/0	124/7/120	113/5/133	85/6/160	0/49/202
FARC-HD	120/7/124	0/251/0	120/6/125	91/5/155	0/57/194
IVTURS	133/5/113	125/6/120	0/251/0	95/7/149	0/68/183
IVTURS-Imb	160/6/85	155/5/91	149/7/95	0/251/0	0/93/158
Oracle	202/49/0	194/57/0	183/68/0	158/93/0	0/251/0

(b) Win/tie/loss comparison using test AUC for all the training datasets.

TABLE VII
SUMMARY RESULTS FOR BALANCED TRAINING DATASETS

classifiers	train ± s.d.	test ± s.d.	hits
FARC-FW	0.9295 ± 0.006	0.8703 ± 0.049	0.20
FARC-HD	0.9256 ± 0.006	0.8701 ± 0.049	0.25
IVTURS	0.9187 ± 0.006	0.8723 ± 0.048	0.38
IVTURS-Imb	0.9228 ± 0.006	0.8714 ± 0.049	0.29
Oracle	0.9270 ± 0.005	0.8810 ± 0.045	1.00

(a) Training and test AUC for balanced training datasets.

	FARC-FW	FARC-HD	IVTURS	IVTURS-Imb	Oracle
FARC-FW	0/122/0	56/6/60	47/4/71	46/5/71	0/24/98
FARC-HD	60/6/56	0/122/0	50/3/69	53/4/65	0/30/92
IVTURS	71/4/47	69/3/50	0/122/0	59/6/57	0/46/76
IVTURS-Imb	71/5/46	65/4/53	57/6/59	0/122/0	0/35/87
Oracle	98/24/0	92/30/0	76/46/0	87/35/0	0/122/0

(b) Win/tie/loss comparison using test AUC for balanced training datasets.

differences are the lowest) will be removed. Analyzing Fig. 8, we can see that there are some datasets whose differences are very low, and then, these differences starts to increase very fast. In this paper, we have filtered out the 20% of the datasets (which are below the red line), allowing to still have a reasonable high number of datasets.

2) Generating the HRDS: We have analyzed the distribution of the dataset characteristics in D^{train} and we have found that balanced and imbalanced datasets are equally distributed (as well as in [10], we have considered $IR \leq 1.5$ for balanced and $IR > 1.5$ for imbalanced problems). As one of the base classifiers is focused on imbalanced problems, the results might be skewed in benefit of IVTURS-Imb, with a clear bias to select it as the best classifier in a higher proportion than the rest of classifiers. Therefore, we have divided the training datasets D^{training} into balanced $D^{\text{training}}_{\text{bal}}$ and imbalanced $D^{\text{training}}_{\text{imbal}}$ in order to analyze them separately and build independent hierarchical rule systems, that will be combined later. In fact, if we compare the results for the training datasets D^{train} (see Table VI), we can see that the performance of IVTURS-Imb is noticeable better than the other base classifiers. However, if we observe the results for the balanced and imbalanced training datasets (Tables VII and VIII, respectively) the results are quite different. In the case of balanced datasets, paying attention to the AUC, FARC-FW, and FARC-HD would be the best, but if we focus on the percentage of perfect hits (same results than the Oracle) IVTURS seems to be the best. On the contrary, in the case of imbalanced problems, IVTURS-Imb is clearly the outstanding classifier both in terms of AUC and Hits. Attending the win/tie/loss metric, no other classifier seems to behave similar IVTURS-Imb. Also, if we see the Hits it reaches the 45%, while the second best classifier reaches only the 21%.

As a consequence, we have determined to use always IVTURS-Imb in the case of imbalanced problems (rule "If $IR > 1.5$ then IVTURS-Imb"). In contrast, in the case of balanced datasets there is not an outstanding classifier, implying the necessity of applying our methodology to discover a hierarchical rule system to select the best model for different contexts.

TABLE VIII
SUMMARY RESULTS FOR IMBALANCED TRAINING DATASETS

classifier	train ± s.d.	test ± s.d.	hits
FARC-FW	0.9490 ± 0.012	0.8667 ± 0.093	0.19
FARC-HD	0.9397 ± 0.012	0.8650 ± 0.090	0.21
IVTURS	0.9366 ± 0.011	0.8654 ± 0.090	0.17
IVTURS-Imb	0.9607 ± 0.010	0.8792 ± 0.088	0.45
Oracle	0.9571 ± 0.010	0.8878 ± 0.085	1.00

(a) Training and test AUC for imbalanced training datasets.

	FARC-FW	FARC-HD	IVTURS	IVTURS-Imb	Oracle
FARC-FW	0/129/0	68/1/60	66/1/62	39/1/89	0/25/104
FARC-HD	60/1/68	0/129/0	70/3/56	38/1/90	0/27/102
IVTURS	62/1/66	56/3/70	0/129/0	36/1/92	0/22/107
IVTURS-Imb	89/1/39	90/1/38	92/1/36	0/129/0	0/58/71
Oracle	104/25/0	102/27/0	107/22/0	71/58/0	0/129/0

(b) Win/tie/loss comparison using test AUC for imbalanced training datasets.

3) Evaluation of FAR-MC: After running the algorithm described in Fig. 5 using the 14 combinations of parameters for $mget$ and th, the best results were obtained using $mget = 0.01$ and $th = 0.001$, and the best default classifier was *FARC-FW*. The rule system, combined with the ad hoc designed rule for the imbalanced datasets, is shown in Fig. 9.

In the case of the rule for IVTURS-Imb it uses three DCMs based on the class separability: middle values for N2, small for N3 and high values for N4. Respect to the rule for IVTURS it uses two measures of class overlapping (highest value for F2 and smallest for F3), and two of geometry, topology, and density of manifolds (small values for N4 and the highest value for T1).

This HRDS suggests that the scope of the base classifiers and the DCMs used in the decision rules differ. However, we have analyzed the DCM characteristics of the datasets that fires each rule, and from this point of view results agree with what we expected. We think it is related to the generation process: the DoC used for each decision rule are selected from the best combination of *mget*, *th* and rule orders in terms of AUC, so the

```
**if** IR∈(1.5,Infinity] **then**
    IVTURS-Imb
**else if** N2∈[0.3190,0.5127] **or** N3 ∈ [0.0097,0.1458] **or** N4 ∈ [0.3529,0.4585] **then**
    IVTURS-Imb
**else if** F2 = 1.0 **or** F3 = 0.0 **or** N4 ∈ [0.0472,0.0778] **or** T1 = 1.0 **then**
    IVTURS
**else**
    FARC-FW
**end if**
```

Fig. 9. Hierarchical Rule System generated from D^{train}.

best dataset split for each classifier is achieved with this system. However, also the dataset properties for each rule agree with the scope of its classifier.

The information related with the DCM characteristics for the set of problems that fire each decision rule can be consulted in the Appendix, which can be downloaded in the complementary website. In the table related to the training datasets, we may observe from these results that in the case of IVTURS-Imb for balanced datasets, the IR is greater than in the other two cases. This makes sense as IVTURS-Imb was designed for imbalanced problems. Analyzing the DCM for the datasets fired by IVTURS, the most remarkable statistics are high T2 values, which refers to the density of manifolds (ratio between the number of instances and the number of input variables), and low values for F1 which implies high overlapped data. Then, the datasets which fires the FARC-FW classifier have very high values for F1 (which refers to low overlaping between features) and low values for T2 that means a low ratio between the number of instances and the number of features. These results are reasonable from the point of view of the scope of each classifier. Only for the case of FARC-FW we expected a set of problems with high overlaping between features, but we found the opposite. This might be because IVTURS-Imb also deals well with this type of datasets and FARC-FW is in a lower level of priority inside the hierarchical rule system.

The results for the proposed metaclassifier FAR-MC are depicted in Table IX. As it was carried out previously, the mean training and test AUC is reported for all the classifiers, including FAR-MC and the Oracle, and the percentage of times each classifier matches with the decision of the Oracle. Also, the metric win/tie/loss is shown.

If we focus on the average train and test AUC values, all of them are quite similar, even the Oracle classifier. This is due to the behavior of all the base classifiers, which in mean are very similar (here we notice the effect of the no free lunch theorem). However, if we pay attention to the hits or win/tie loss metric, results are very different. In the case of imbalanced datasets, obviously FAR-MC and IVTURS-Imb have the same results as they behave equally.

Regarding the set of balanced problems, we may extract the following conclusions. First, in terms of hits (same results as the Oracle), IVTURS, and FAR-MC seems to be better than the

TABLE IX
RESULTS FOR THE TRAINING DATASETS: TRAINING AND TEST AUC (± STANDARD DEVIATION), PERCENTAGE OF TIMES WHICH REACHES THE BEST POSSIBLE RESULT (ORACLE) AND THE WIN/TIE/LOSS METRIC COMPARED WITH THE BEST IN TERMS OF MEAN RANKING (FAR-MC)

classifier	train ± s.d.	test ± s.d.	hits	w/t/l
FARC-FW	0.9395 ± 0.009	0.8685 ± 0.072	0.20	147/45/59
FARC-HD	0.9328 ± 0.009	0.8675 ± 0.070	0.23	163/6/82
FAR-MC	0.9451 ± 0.008	0.8775 ± 0.069	0.42	-/-/-
IVTURS	0.9279 ± 0.009	0.8688 ± 0.069	0.27	142/30/79
IVTURS-Imb	0.9423 ± 0.008	0.8754 ± 0.069	0.37	39/186/26
Oracle	0.9425 ± 0.008	0.8845 ± 0.065	1.00	0/106/145

(a) The full set of datasets.

classifier	train ± s.d.	test ± s.d.	hits	w/t/l
FARC-FW	0.9295 ± 0.006	0.8703 ± 0.049	0.20	58/44/20
FARC-HD	0.9256 ± 0.006	0.8701 ± 0.049	0.25	73/5/44
FAR-MC	0.9287 ± 0.095	0.8757 ± 0.131	0.39	-/-/-
IVTURS	0.9187 ± 0.006	0.8723 ± 0.048	0.38	50/29/43
IVTURS-Imb	0.9228 ± 0.006	0.8714 ± 0.049	0.29	39/57/26
Oracle	0.9270 ± 0.005	0.8810 ± 0.045	1.00	0/48/74

(b) Balanced datasets.

classifier	train ± s.d.	test ± s.d.	hits	w/t/l
FARC-FW	0.9490 ± 0.012	0.8667 ± 0.093	0.19	89/1/39
FARC-HD	0.9397 ± 0.012	0.8650 ± 0.090	0.21	90/1/38
FAR-MC	0.9607 ± 0.010	0.8792 ± 0.088	0.45	-/-/-
IVTURS	0.9366 ± 0.011	0.8654 ± 0.090	0.17	92/1/36
IVTURS-Imb	0.9607 ± 0.010	0.8792 ± 0.088	0.45	0/129/0
Oracle	0.9571 ± 0.010	0.8878 ± 0.085	1.00	0/58/71

(c) Imbalanced datasets.

rest. Between these two classifiers, the win/tie/loss point out that FAR-MC is more robust than IVTURS.

B. Testing FAR-MC Against the FARC-HD Family Classifiers

Once we have learnt the hierarchical rule system of FAR-MC, we will evaluate its performance over the test set of problems D^{test}.

As in the standard classification task, this will allow us to determine the goodness of our approach on a set of unseen problems, so that we can determine whether our FAR-MC classifier is able to achieve a good generalization, i.e., the rules have been properly learnt and are valid for new unseen problems.

Similarly than in Section IV-3A, for each rule we show in the complementary material (table of test datasets) the number of datasets from D^{test} which fire each rule, and the characteristics in terms of DCM for this set of problems. If we compare these results with those for the training datasets (the two tables shown in the previous website), we can extract similar conclusions. In fact, if we compare the percentage of datasets which fire each rule for both sets of problems D^{train} and D^{test}, we can see that these numbers are quite similar. That means the knowledge extracted in the training phase can also be applied to unseen problems, which implies a good generalization capability of the HRDS. It also confirms our initial hypothesis and relates the DCM values and the performance of the classifiers, supporting the research carried out in this paper.

The results for D^{test} are shown in Table X. For all the classifiers we show the training and test AUC (± the standard deviation), the percentage of problems where each classifier obtains

TABLE X
RESULTS FOR THE TEST DATASETS: TRAINING AND TEST AUC (± STANDARD DEVIATION), PERCENTAGE OF TIMES WHICH REACHES THE BEST POSSIBLE RESULT (ORACLE) AND THE WIN/TIE/LOSS METRIC COMPARED WITH THE BEST IN TERMS OF MEAN RANKING (FAR-MC)

classifier	train ± s.d.	test ± s.d.	hits	w/t/l
FARC-FW	0.9398 ± 0.009	0.8686 ± 0.069	0.24	92/31/47
FARC-HD	0.9324 ± 0.009	0.8667 ± 0.066	0.19	103/5/62
FAR-MC	0.9431 ± 0.007	0.8773 ± 0.067	0.41	-/-/-
IVTURS	0.9268 ± 0.008	0.8693 ± 0.067	0.32	87/26/57
IVTURS-Imb	0.9395 ± 0.008	0.8737 ± 0.068	0.34	31/121/18
Oracle	0.9417 ± 0.007	0.8844 ± 0.062	1.00	0/70/100

(a) The full set of datasets.

classifier	train ± s.d.	test ± s.d.	hits	w/t/l
FARC-FW	0.9241 ± 0.005	0.8652 ± 0.049	0.26	37/29/16
FARC-HD	0.9178 ± 0.006	0.8610 ± 0.048	0.22	46/4/32
FAR-MC	0.9219 ± 0.006	0.8698 ± 0.048	0.46	-/-/-
IVTURS	0.9113 ± 0.006	0.8646 ± 0.048	0.33	36/25/21
IVTURS-Imb	0.9145 ± 0.007	0.8622 ± 0.050	0.32	31/33/18
Oracle	0.9230 ± 0.006	0.8750 ± 0.044	1.00	0/38/44

(b) Balanced datasets.

classifier	train ± s.d.	test ± s.d.	hits	w/t/l
FARC-FW	0.9545 ± 0.012	0.8718 ± 0.088	0.22	55/2/31
FARC-HD	0.9460 ± 0.011	0.8720 ± 0.083	0.17	57/1/30
FAR-MC	0.9628 ± 0.009	0.8843 ± 0.084	0.36	-/-/-
IVTURS	0.9413 ± 0.010	0.8736 ± 0.085	0.31	51/1/36
IVTURS-Imb	0.9628 ± 0.009	0.8843 ± 0.084	0.36	0/88/0
Oracle	0.9592 ± 0.008	0.8932 ± 0.079	1.00	0/32/56

(c) Imbalanced datasets.

the same performance as the Oracle, and the win/tie/loss metric between the best classifier in terms of mean rank (FAR-MC) and the rest.

In the case of balanced datasets, we can see that FAR-MC has increased in seven points the percentage of hits respect to the results for $D^{\text{train}}_{\text{bal}}$. Comparing with the base classifiers, we can appreciate a slightly decrease in the relative performance versus FARC-HD (the ratio win/loses for $D^{\text{train}}_{\text{bal}}$ is 1.66, and for $D^{\text{test}}_{\text{bal}}$ is 1.44, and a slightly increase versus IVTURS-Imb (1.5 for $D^{\text{train}}_{\text{bal}}$ and 1.72 for $D^{\text{test}}_{\text{bal}}$). In the case of imbalanced problems, we can see more uniform results between IVTURS and IVTURS-Imb. However, focusing on the w/t/l metric, it is still the outstanding. Moreover, in general we can extract similar conclusions as for the training datasets D^{train}, which means that the HRDS has correctly adapted to the new problems, concluding that FAR-MC has a good generalization power.

We also performed a statistical test to compare the performance of FAR-MC against the FARC-HD familywise classifiers. We will divide the analysis for the balanced, imbalanced and the full set of test datasets D^{test}. The results can be seen in the Table XI.

In accordance with these experimental results, FAR-MC is the best classifier in terms of the mean rank in all the cases. Moreover, if we observe the corrected p-values FAR-MC is statistically better than the other methods. This fact supports the conclusions that we extracted previously, stressing FAR-MC as the best strategy among the FARC-HD familywise classifiers. As we discussed before, the results are noticeable better in terms of the win/tie/loss metric.

TABLE XI
STATISTICAL TEST ANALYSIS BETWEEN FARC-SELECTOR AND FARC-HD AND ITS VARIANTS

classifier	rank	p-value	p-value (Holm)	w / t / l
FAR-MC	2.62	-	-	- / - / -
IVTURS-Imb	2.78	0.052167	0.052167	31 /121/18
IVTURS	3.02	0.000472	0.000945	87 / 26 /57
FARC-FW	3.25	0.000032	0.000095	92 / 31 /47
FARC-HD	3.33	0.000002	0.000009	103/ 5 /62

(a) The full set of datasets (Friedman p-value = 6.23e-05).

classifier	rank	p-value	p-value (Holm)	w / t / l
FAR-MC	2.62	-	-	- / - / -
IVTURS-Imb	2.95	0.010919	0.021838	31/33/18
IVTURS	3.02	0.014540	0.021838	36/25/21
FARC-HD	3.19	0.002930	0.011718	46/ 4 /32
FARC-FW	3.22	0.003776	0.011718	37/29/16

(b) Balanced datasets (Friedman p-value = 1.00e-01).

classifier	rank	p-value	p-value (Holm)	w / t / l
FAR-MC	2.62	-	-	- / - / -
IVTURS-Imb	2.62	0.500826	0.500826	0 /88/ 0
IVTURS	3.02	0.010574	0.021148	51/ 1 /36
FARC-FW	3.28	0.000886	0.002658	55/ 2 /31
FARC-HD	3.45	0.000152	0.000606	57/ 1 /30

(c) Imbalanced datasets (Friedman p-value = 5.20e-4).

C. Analyzing FARC-Selector Versus State of the art

In the context of classification problems, maybe one of the most widely used rule-based algorithms is the C4.5 decision tree [19], [38]. The reasons are its robustness, efficiency, and good performance [39], [40].

FURIA [20] is also a well-known and accurate state-of-the-art fuzzy classifier, which has been recently used in several works as a baseline algorithm to compare with [41]–[45].

Moreover, both algorithms are designed to be used for standard classification problems. For imbalanced datasets they has been also widely applied in conjunction with the SMOTE preprocessing technique [22], [30] (aiming at rebalancing the training set).

In this section, we will compare these state-of-the-art classifiers with our proposal FAR-MC. As stated in Section III-C, we will make use of both AUC and F1-score metrics in order to provide well-founded conclusions from our study. To make the comparison, we will apply the same methodology used in Section IV-B. The prediction performance can be seen in Tables XII and XIII (errors and the percentage of the test improvement obtained with FAR-MC using AUC and F1-score metrics, respectively) and the statistical analysis in Tables XIV and XV using AUC and F1-score, respectively.

First of all, we can see similar results comparing AUC and F1-score, being F1-score slightly more favorable to FAR-MC. This fact gives more robustness to the conclusions we discuss next.

If we compare FAR-MC versus C4.5 (both variants), the results point out our proposal as the outperforming classifier, both in terms of mean rank and the win/tie/loss metric. Paying attention to the corrected p-values, it is especially worth pointing that FAR-MC is rather better, which supports again the quality and robustness of FAR-MC. In relation to the test

TABLE XII
RESULTS FOR THE TEST DATASETS: TRAINING AND TEST AUC (± STANDARD DEVIATION) AND THE WIN/TIE/LOSS METRIC COMPARED WITH THE BEST IN TERMS OF MEAN RANKING (SMOTE+FURIA)

classifier	train ± s.d.	test ± s.d.	% test
FAR-MC	0.9431 ± 0.007	0.8773 ± 0.067	-
FURIA	0.9233 ± 0.017	0.8679 ± 0.068	1.08%
SMOTE+FURIA	0.9400 ± 0.013	0.8806 ± 0.063	-0.38%
C4.5	0.9284 ± 0.015	0.8620 ± 0.068	1.77%
SMOTE+C4.5	0.9474 ± 0.013	0.8675 ± 0.070	1.13%

(a) The full set of datasets.

classifier	train ± s.d.	test ± s.d.	% test
FAR-MC	0.9219 ± 0.006	0.8698 ± 0.048	-
FURIA	0.9123 ± 0.012	0.8709 ± 0.048	-0.13%
SMOTE+FURIA	0.9131 ± 0.012	0.8728 ± 0.049	-0.34%
C4.5	0.9225 ± 0.010	0.8562 ± 0.052	1.59%
SMOTE+C4.5	0.9266 ± 0.011	0.8562 ± 0.054	1.59%

(b) Balanced datasets.

classifier	train ± s.d.	test ± s.d.	% test
FAR-MC	0.9628 ± 0.009	0.8843 ± 0.084	-
FURIA	0.9335 ± 0.022	0.8652 ± 0.087	2.21%
SMOTE+FURIA	0.9650 ± 0.014	0.8879 ± 0.077	-0.41%
C4.5	0.9339 ± 0.021	0.8674 ± 0.082	1.94%
SMOTE+C4.5	0.9667 ± 0.014	0.8780 ± 0.084	0.72%

(c) Imbalanced datasets.

TABLE XIII
RESULTS FOR THE TEST DATASETS: TRAINING AND TEST F1-SCORE (± STANDARD DEVIATION) AND THE WIN/TIE/LOSS METRIC COMPARED WITH THE BEST IN TERMS OF MEAN RANKING (SMOTE+FURIA)

classifier	train ± s.d.	test ± s.d.	% test
FAR-MC	0.9224 ± 0.013	0.8329 ± 0.091	-
FURIA	0.9048 ± 0.024	0.8201 ± 0.099	1.54%
SMOTE+FURIA	0.9193 ± 0.020	0.8340 ± 0.086	-0.13%
C4.5	0.9053 ± 0.022	0.8096 ± 0.096	2.79%
SMOTE+C4.5	0.9225 ± 0.021	0.8143 ± 0.094	2.23%

(a) The full set of datasets.

classifier	train ± s.d.	test ± s.d.	% test
FAR-MC	0.9134 ± 0.007	0.8533 ± 0.055	-
FURIA	0.9004 ± 0.016	0.8513 ± 0.058	0.23%
SMOTE+FURIA	0.9059 ± 0.013	0.8600 ± 0.055	-0.79%
C4.5	0.9073 ± 0.013	0.8315 ± 0.062	2.55%
SMOTE+C4.5	0.9199 ± 0.013	0.8412 ± 0.063	1.41%

(b) Balanced datasets.

classifier	train ± s.d.	test ± s.d.	% test
FAR-MC	0.9308 ± 0.018	0.8139 ± 0.124	-
FURIA	0.9088 ± 0.031	0.7910 ± 0.137	2.81%
SMOTE+FURIA	0.9317 ± 0.027	0.8097 ± 0.114	0.52%
C4.5	0.9034 ± 0.031	0.7891 ± 0.128	3.04%
SMOTE+C4.5	0.9249 ± 0.028	0.7891 ± 0.123	3.04%

(c) Imbalanced datasets.

TABLE XIV
STATISTICAL TEST ANALYSIS BETWEEN FAR-MC AND THE STATE-OF-THE-ART CLASSIFIERS, USING THE AUC METRIC

classifier	rank	p-value	p-value (Holm)	w / t / l
SMOTE+FURIA	2.41	0.474016	0.474016	- / - / -
FAR-MC	2.56	-	-	80 / 2 /88
FURIA	3.17	0.000557	0.001114	102/30/38
SMOTE+C4.5	3.34	0.000004	0.000013	117/11/42
C4.5	3.51	0.000000	0.000001	115/10/45

(a) Study with the whole set of test datasets (Friedman p-value = 2.43e-13).

classifier	rank	p-value	p-value (Holm)	w / t / l
SMOTE+FURIA	2.47	0.425236	0.625860	- / - / -
FAR-MC	2.49	-	-	38/ 0 /44
FURIA	2.68	0.312930	0.625860	33/28/21
SMOTE+C4.5	3.66	0.000020	0.000061	59/ 7 /16
C4.5	3.70	0.000013	0.000054	57/ 6 /19

(b) Study with the set of balanced test datasets (Friedman p-value = 1.46e-10).

classifier	rank	p-value	p-value (Holm)	w/t/ l
SMOTE+FURIA	2.35	0.459607	0.459607	- /-/ -
FAR-MC	2.62	-	-	42/2/44
SMOTE+C4.5	3.05	0.035801	0.071601	58/4/26
C4.5	3.34	0.003833	0.011499	58/4/26
FURIA	3.64	0.000033	0.000132	69/2/17

(c) Study with the set of imbalanced test datasets (Friedman p-value = 1.05e-07).

TABLE XV
STATISTICAL TEST ANALYSIS BETWEEN FAR-MC AND THE STATE-OF-THE-ART CLASSIFIERS, USING THE F1-SCORE METRIC

classifier	rank	p-value	p-value (Holm)	w /t/ l
FAR-MC	2.50	-	-	- /-/ -
SMOTE+FURIA	2.56	0.512490	0.512490	97 /2/71
FURIA	2.91	0.000757	0.001513	97 /3/70
C4.5	3.41	0.000000	0.000000	107/2/61
SMOTE+C4.5	3.61	0.000000	0.000000	120/2/48

(a) Study with the whole set of test datasets (Friedman p-value = 6.74e-14).

classifier	rank	p-value	p-value (Holm)	w/ t / l
SMOTE+FURIA	2.34	0.423877	0.467564	- / - / -
FAR-MC	2.52	-	-	40/ 1 /41
FURIA	2.74	0.233782	0.467564	35/28/19
SMOTE+C4.5	3.59	0.000145	0.000435	61/ 6 /15
C4.5	3.81	0.000004	0.000016	62/ 6 /14

(b) Study with the set of balanced test datasets (Friedman p-value = 1.32e-11).

classifier	rank	p-value	p-value (Holm)	w/t/ l
FAR-MC	2.47	-	-	- /-/ -
SMOTE+FURIA	2.78	0.011582	0.022136	56/1/31
C4.5	3.05	0.011068	0.022136	48/1/39
FURIA	3.07	0.001765	0.005294	52/2/34
SMOTE+C4.5	3.64	0.000002	0.000009	64/1/23

(c) Study with the set of imbalanced test datasets (Friedman p-value = 3.10e-05).

error, we can see an improvement with respect to the state-of-the-art algorithm greater than 1% and 2% (for AUC and F1-score), except for balanced datasets (which is 0.72% and 1.41%, respectively).

Focusing on the comparison between the fuzzy classifiers, we observe a similar behavior in terms of performance between FAR-MC and SMOTE+FURIA, whereas FAR-MC is significantly better than FURIA in the general case study (all datasets) and for imbalanced problems. From the point of view of interpretability, FAR-MC is remarkably more interpretable. FURIA, based on the well-known RIPPER algorithm [46], tends to generate large systems of specialized rules (which are formed by many antecedents). Moreover, it does not generate directly fuzzy rules. Instead, it generates interval based rules, and the process is followed by a fuzzy fication phase, which generates hardly interpretable database definitions. On the other hand, FAR-based classifiers used by FAR-MC aim to produce simple systems, both in terms of the number of rules and the number of antecedents per rule (usually parameterized to generate rules formed by three antecedents at most).

V. Concluding Remarks

In this paper, we have proposed FAR-MC, a new metaclassifier that aims to use the best base classifier among a set of them based on the input dataset properties. To do so, we have gathered a set of 12 different DCM to create DoC for the associated classifiers. These DCM describe the dataset properties allowing to determine if a specific classifier may perform better than the others.

We have generated these DoC using the software tool developed in [10]. To use it properly in the scope of this problem we have designed a score based on the relative performance between each classifier and other labeled as the default classifier.

Finally, we have built a hierarchical rule system that aims to select the best base classifier accordingly to the dataset properties. The experimental results show a good performance of FAR-MC obtaining significative statistical differences comparing it versus the base classifiers, especially in the case of the datasets selected for the validation. We also compared the results versus the state-of-the-art classifiers C4.5 and FURIA, using and not the preprocessing technique SMOTE. Results show that FAR-MC is much better than C4.5, and not statistically different from SMOTE+FURIA. However, FAR-MC produces simpler and more interpretable models. Moreover, based on the percentage of hits with respect to the Oracle, we believe that there is field to improve the results following this research line.

It is also worth pointing that, in terms of computational time, FAR-MC is comparable to the base classifiers. Our methodology is based on two stages. The first one is the computation of the DCMs of the input data. The second one is the selection of the FARC-HD classifier by means of the hierarchical rules, and the application of the classifier itself. Regarding the first stage, some of the DCMs are really fast to compute as they rely on statistical properties of the input attributes. For the second stage, our proposal only uses a small set of rules to determine the base classifier, thus, moving the computational efforts to the application of the FARC-HD based classifiers.

As future work, we propose the usage of other alternatives for the score based on the relative performance. One option could be to use a metric that uses the information of the relative performance of all the classifiers at once, as it is the ranking, but without losing the information about the differences. This can be done by normalizing the performance metric for all the classifiers in the range $[0 - 1]$. Other alternative could be to use the ranking of the classifier performances for each problem, and design a new methodology to derive the DoC taking into account that we deal with this particular metric. Moreover, it could be interesting to analyze the performance of an ensemble using the same family of FAR classifiers. A comparative of these results versus our proposed metaclassifier could point out useful conclusions.

References

[1] D. Gómez and A. Rojas, "An empirical overview of the no free lunch theorem and its effect on real-world machine learning classification," *Neural Comput.*, vol. 28, pp. 216–228, 2016.

[2] D. H. Wolpert *et al.*, "No free lunch theorems for search," Technical Report SFI-TR-95-02-010, Santa Fe Institute, Tech. Rep., 1995.

[3] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognit.*, vol. 40, no. 12, pp. 3358–3378, 2007.

[4] S. Sukhanov, A. Merentitis, C. Debes, J. Hahn, and A. Zoubir, "Bootstrap-based SVM aggregation for class imbalance problems," in *Proc. Eur. Signal Process. Conf.*, 2015, pp. 165–169.

[5] O. P. Panagopoulos, V. Pappu, P. Xanthopoulos, and P. M. Pardalos, "Constrained subspace classifier for high dimensional datasets," *Omega*, vol. 59, pp. 40–46, 2016.

[6] L. Byczkowska-Lipinska and A. Wosiak, "Hybrid classification of high-dimensional biomedical tumour datasets," *Adv. Intell. Comput. Diagn. Control*, vol. 386, pp. 287–298, 2016.

[7] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 289–300, Mar. 2002.

[8] S. Singh, "Multiresolution estimates of classification complexity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1534–1539, Dec. 2003.

[9] M. J. Flores, J. A. Gámez, and A. M. Martínez, "Domains of competence of the semi-naive Bayesian network classifiers," *Inf. Sci.*, vol. 260, pp. 120–148, 2014.

[10] J. Luengo and F. Herrera, "An automatic extraction method of the domains of competence for learning classifiers using data complexity measures," *Know. Inf. Syst.*, vol. 42, no. 1, pp. 147–180, 2015.

[11] J. Alcala-Fdez, R. Alcala, and F. Herrera, "A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 5, pp. 857–872, Oct. 2011.

[12] J. A. Sanz, A. Fernandez, H. Bustince, and F. Herrera, "IVTURS: A linguistic fuzzy rule-based classification system based on a new interval-valued fuzzy reasoning method with tuning and rule selection," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 3, pp. 399–411, Jun. 2013.

[13] J. A. Sanz, D. Bernardo, F. Herrera, H. Bustince, and H. Hagras, "A compact evolutionary interval-valued fuzzy rule-based classification system for the modeling and prediction of real-world financial applications with imbalanced data," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 4, pp. 973–990, Aug. 2015.

[14] S. Alshomrani, A. Bawakid, S.-O. Shim, A. Fernández, and F. Herrera, "A proposal for evolutionary fuzzy systems using feature weighting: Dealing with overlapping in imbalanced datasets," *Knowl.-Based Syst.*, vol. 73, pp. 1–17, 2015.

[15] G. Lucca *et al.*, "Preaggregation functions: Construction and an application," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 2, pp. 260–272, Apr. 2016.

[16] P. Villar, A. Fernández, and F. Herrera, "On the combination of pairwise and granularity learning for improving fuzzy rule-based classification systems: GL-FARCHD-OVO," in *Proc. 9th Int. Conf. Comput. Recognit. Syst.*, 2016, pp. 135–146.

[17] M. Cintra, H. Camargo, and M. Monard, "Genetic generation of fuzzy systems with rule extraction using formal concept analysis," *Inf. Sci.*, vol. 349, pp. 199–215, 2016.

[18] A. Fernández, V. López, M. J. del Jesus, and F. Herrera, "Revisiting evolutionary fuzzy systems: Taxonomy, applications, new trends and challenges," *Knowl.-Based Syst.*, vol. 80, pp. 109–121, 2015.

[19] J. R. Quinlan, "C4.5: programs for machine learning," Amsterdam, The Netherlands: Elsevier, 2014.

[20] J. Hühn and E. Hüllermeier, "FURIA: An algorithm for unordered fuzzy rule induction," *Data Mining Know. Discovery*, vol. 19, no. 3, pp. 293–319, 2009.

[21] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artifi. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[22] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Inf. Sci.*, vol. 250, pp. 113–141, 2013.

[23] T. K. Ho and M. Basu, "Measuring the complexity of classification problems," in *Proc. 15th Int. Conf. Pattern Recognit.*, 2000, vol. 2, pp. 43–47.

[24] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *ACM SIGMOD Rec.*, vol. 22, no. 2, pp. 207–216, 1993.
[25] R. Sambuc, "Function ø-flous, application a laide au diagnostic en pathologie thyroidienne," Ph.D. dissertation, Univ. Marseille, Marseille, France, 1975.
[26] H. Bustince *et al.*, "Ignorance functions. An application to the calculation of the threshold in prostate ultrasound images," *Fuzzy Sets Syst.*, vol. 161, no. 1, pp. 20–36, 2010.
[27] R. C. Prati, G. E. Batista, and M. C. Monard, "Class imbalances versus class overlapping: an analysis of a learning system behavior," in *Proc. Mexican Int. Conf. Artif. Intell.*, 2004, pp. 312–321.
[28] J. Huang and C. X. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE Trans. Know. Data Eng.*, vol. 17, no. 3, pp. 299–310, Mar. 2005.
[29] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Know. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
[30] N. V. Chawla, "C4.5 and imbalanced data sets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure," in *Proc. ICML'03 Workshop Class Imbalances*, 2003.
[31] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.
[32] S. García and F. Herrera, "An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons," *J. Mach. Learn. Res.*, vol. 9, pp. 2677–2694, 2008.
[33] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Ann. Math. Statist.*, vol. 11, pp. 86–92, 1940.
[34] M. Hollander, D. A. Wolfe, and E. Chicken, "Nonparametric statistical methods," Hoboken, NJ, USA: Wiley, 2013.
[35] S. Holm, "A simple sequentially rejective multiple test procedure," *Scand. J. Statist.*, vol. 6, pp. 65–70, 1979.
[36] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.
[37] J. Alcal *et al.*, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, no. 2–3, pp. 255–287, 2010.
[38] X. W.-V. K.-J. Ross and Z.-H. Zhou, "Top 10 algorithms in data mining," *Know. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, 2007.
[39] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, vol. 3, pp. 856–863.
[40] E. Kretschmann, W. Fleischmann, and R. Apweiler, "Automatic rule generation for protein annotation with the c4.5 data mining algorithm applied on swiss-prot," *Bioinformatics*, vol. 17, no. 10, pp. 920–926, 2001.
[41] M. Antonelli, P. Ducange, and F. Marcelloni, "A fast and efficient multi-objective evolutionary learning scheme for fuzzy rule-based classifiers," *Inf. Sci.*, vol. 283, pp. 36–54, 2014.
[42] J. Thongkam and V. Sukmak, "Enhancing the performance of association rule models by filtering instances in colorectal cancer patients," *Eng. Appl. Sci. Res.*, vol. 44, no. 2, pp. 76–83, 2017.
[43] A. S. Koshiyama, M. M. B. R. Vellasco, and R. Tanscheit, "Gpfis-class: A genetic fuzzy system based on genetic programming for classification problems," *Appl. Soft Comput. J.*, vol. 37, pp. 561–571, 2015.
[44] A. Palacios, L. Sánchez, I. Couso, and S. Destercke, "An extension of the furia classification algorithm to low quality data through fuzzy rankings and its application to the early diagnosis of dyslexia," *Neurocomputing*, vol. 176, pp. 60–71, 2016.
[45] M. Elkano, M. Galar, J. Sanz, and H. Bustince, "Fuzzy rule-based classification systems for multi-class problems using binary decomposition strategies: On the influence of n-dimensional overlap functions in the fuzzy reasoning method," *Inf. Sci.*, vol. 332, pp. 94–114, 2016.
[46] W. Cohen, "Fast effective rule induction," in *Proc. 12th Int. Conf. Mach. Learn.*, 1995, pp. 115–123.

Authors' photographs and biographies not available at the time of publication.