Multi-objective Evolutionary Feature Selection for Fuzzy Classification

F. Jiménez, C. Martínez, E. Marzano, J. Palma (Member IEEE), G. Sánchez, G. Sciavicco

***Abstract*—The interpretability of classification systems refers to the ability of these to express their behaviour in a way that is easily understandable by a user. Interpretable classification models allow for external validation by an expert and, in certain disciplines such as medicine or business, providing information about decision making is essential for ethical and human reasons. Fuzzy rule-based classification systems are consolidated powerful classification tools based on fuzzy logic and designed to produce interpretable models; however, in presence of a large number of attributes, even rule-based models tend to be too complex to be easily interpreted. In this work, we propose a novel multivariate feature selection method in which both search strategy and classifier are based on multi-objective evolutionary computation. We designed a set of experiments to establish an acceptable setting with respect to the number of evaluations required by the search strategy and by the classifier, and we tested our strategy on a real-life dataset. Then, we compared our results against a wide range of feature selection methods that includes filter, wrapper, multivariate and univariate methods, with deterministic and probabilistic search strategies, and with evaluators of diverse nature. Finally, the fuzzy rule-based classification model obtained with the proposed method has been evaluated with standard performance metrics and compared with other well-known fuzzy rule-based classifiers. We have used two real-life datasets extracted from a contact center; in one case, with the proposed method we obtained an accuracy of 0.7857 with 8 rules, while the best fuzzy classifier compared obtained 0.7679 with 8 rules, and in the second case, we obtained an accuracy of 0.7403 with 5 rules, while the best fuzzy classifier compared obtained 0.6364 with 4 rules.**

***Index Terms*—Data Classification, Multi-objective Evolutionary Algorithms, Feature Selection, Fuzzy Rules Based Learning.**

I. Introduction

The *interpretability of classification systems* refers to the ability of these to express their behaviour in a way that is easily understandable by a user. Interpretable classification models allow for external validation by an expert and, in certain disciplines such as medicine or business providing information about decision making is essential for ethical and human reasons. The *fuzzy rule-based classification systems* (*FRBCS*) [1], [2], [3], [4] have been strongly developed in the last years, and they are now consolidated powerful classification tools that also allow the interpretation of the model in a direct and clear way since they use linguistic labels in a similar way as human reasoning does. *Multi-objective Evolutionary Algorithms* (MOEAs) [5] have been applied successfully in recent years for the optimization of *FRBCS*. There are two main motivations behind using MOEAs for FRBCS. On the one hand, evolutionary computation is a very powerful technique for the approximation of optimization and search problems in high complexity non-linear spaces, and, on the other hand, multi-objective optimization allows to simultaneously optimize the accuracy and the complexity of the FRBCS, by identifying a set of Pareto solutions. In [6], a MOEA is used to concurrently learn rule base and data base of a FRBS. In this case, two objectives are considered: the first measures the complexity as the sum of the input variable labels used in each of the rules, and the second corresponds to the mean squared error. In [7], fuzzy classifiers for imbalanced and cost-sensitive datasets are generated with a three objective MOEA. The first and second objectives are sensitivity and specificity. The third objective is a complexity measure computed as the sum of the conditions that compose the antecedents of the rules, which is minimized. In [8], *PAES-RCS* method is used to maximize accuracy and minimize the total rule length for internet traffic classification. In [9], *IT2-PAES-RCS* extends *PAES-RCS* to employ Type-2 fuzzy sets, where sensitivity, specificity and total rule length are optimized for financial data classification.

However, the use of fuzzy logic may not be enough for the classification model to be interpretable. Interpretability not only implies *transparency* (expressibility capabilities) but also *compactness*. In a *FRBCS*, improving the compactness implies reducing the number of attributes, the number of linguistic labels for each variable, and the number of rules. Fuzzy rule-based classification systems are designed to produce interpretable models; however, in presence of a large number of attributes, the resulting classifiers may be too complex to be easily interpreted (for example, rules with more than five attributes may be intractable for a human being). In this sense, a feature selection process [10], prior to the fuzzy rule extraction phase, may be crucial step. Although feature selection problem is *NP-hard* [11], with a search space ranging $O(2^N)$ elements, where N is the number of features, a *heuristic* or *meta-heuristic* search strategy can obtain good approximate solutions in reasonable times, thus reducing the complexity to build the final classifier. The three most common feature selection schemata are the so-called *filter*, *wrapper* and *embedded*. Filter selection methods [12], [13], [14] apply statistical measures to evaluate the attribute subset, whereas wrapper methods [15], [16] interact with a classifier to evaluate the attribute subset using some performance metric. Filter methods are computationally faster, but less accurate, than

F. Jiménez, G. Sánchez, and J. Palma are with the Department of Information Engineering and Communications, University of Murcia, Spain, *fernan,gracia,jtpalma@um.es*

C. Martínez is with the International Doctorate School of the University of Murcia, Spain, *carlos.martinez6@um.es*

E. Marzano is CEO of G.A.P. Srl, Udine, Italy, *e.marzano@gapitalia.it*

G. Sciavicco is with the Department of Mathematics and Computer Science, University of Ferrara, Italy, *guido.sciavicco@unife.it*

wrapper methods. In addition, a disadvantage of wrapper methods is that the performances of the selected subsets are often very dependent on the learning algorithm that is used for subset evaluation, so that, for example, a good selection of attributes performed with a decision tree-based wrapper method may result in a poor one when the selected attributes are used in a support vector machine. In [17] a filter feature selection method is proposed where a greedy algorithm is used as search strategy and a dependency measure between fuzzy decision and condition attributes is employed to evaluate the significance of a candidate feature. In [18] a wrapper feature selection method is proposed where best-first is used as search strategy, and Wang and Mendel method to generate fuzzy rule base is used as evaluator. Finally, embedded methods [19] achieve model fitting and feature selection simultaneously. The use of MOEAs as a search strategy for feature selection methods [20], [21], [22], [23], [24] is justified, on the one hand, by the very high cardinality of the search space, and on the other hand, by the intrinsic multi-objective nature of the problem, that requires minimizing the number of chosen attributes and maximizing some performance metrics, such as correlation, consistency, information gain, entropy, accuracy of the classifier, etc.

In [25] we proposed a *FRBCS* based on *multi-objective evolutionary constrained real-parameter optimization*, which maximizes the accuracy and minimize the number of rules, and imposes a constraint for the similarity of fuzzy sets. The maximum number of rules of the model, the maximum number of linguistic labels for each variable, and the maximum similarity of fuzzy sets, are parametrizable, so that they can be established by a user in order to obtain compact models. Once the fuzzy rule set has been extracted, a final linguistic labelling process assigns a linguistic label to each fuzzy set. In addition, this fuzzy classification method is itself a feature selection method, since it detects 'don't care conditions' attributes that can be eliminated from the classification model. However, although the classification method detects 'do not care conditions' attributes, this may not be enough in the presence of many attributes in the database, and a feature selection method prior to the fuzzy rule extraction phase is highly recommended. Therefore, if feature selection is used for a later classification based on fuzzy rules, the best choice would be a feature selection wrapper method that uses a fuzzy rule-based classifier for evaluation. However, this configuration for a wrapper method is not without drawbacks. Due to the high computational cost required by fuzzy rule-based systems in the presence of a large number of attributes, a wrapper method that uses a fuzzy rule-based classifier may be non-viable. It is therefore necessary to carefully analyse the parameters of both search strategy and evaluator to obtain a good trade-off between accuracy and run time.

In this work we propose a novel wrapper based multivariate feature selection method for *FRBCS* which presents the following novelties and benefits with respect to existing methods in the literature:

1) To the best of our knowledge, this is the first work for fuzzy classification that proposes a wrapper feature selection method prior to the phase of fuzzy rule extraction, where both search strategy and evaluator are performed with independent multi-objective evolutionary algorithms. In the current literature, usually, feature selection is embedded in the fuzzy rules extraction algorithm itself by identifying *"don't care conditions"* (*embedded feature selection methods* [7], [8], [9]).
2) Our method for fuzzy rule extraction consists of multi-objective constrained optimization of real parameters, instead of multi-objective combinatorial optimization as does the rest of the methods in the literature. In the rest of the methods, the rule base is explicitly separated from the (previously built) knowledge base containing the definition of the fuzzy sets. Then, the fuzzy sets are combined in the rule base using combinatorial optimization techniques [6], [7], [8], [9]. These approaches use a mixed optimization model: combinatorial optimization for rule and fuzzy sets selection, and real parameter optimization for parameter tuning. All these evolutionary algorithms use a fixed representation with triangular fuzzy sets in a Pittsburgh approach, and they use a mark equal to 0 if the rule is not selected, and a mark equal to 1 when the rule is selected, and an integer number to identify de fuzzy set, including in some cases "don't care conditions". Our approach does not build a explicit knowledge base with the definitions of the fuzzy sets, but fuzzy sets are directly embedded in the rule base in a random fashion within the domains of each variable, with a variable-length float-point representation with gaussian fuzzy sets in a Pittsburgh approach. Since the definitions of the fuzzy sets are random, this can produce intermingled and, therefore, non-interpretable partitions. To prevent this, a similarity constraint for fuzzy sets is imposed in the optimization model, which is handled by the multi-objective evolutionary algorithm by a repair technique and applied after the initialization, crossing and mutation. The gaussian fuzzy sets are represented by their centre and variance as real-coded parameters, and therefore the crossover and mutation operators used by the evolutionary algorithm are those of the float-point representations, varying the centres and the variances of the gaussian fuzzy sets separately.
3) The interpretability of the rule base can be adjusted, not only by imposing a maximum of rules and a maximum of linguistic labels, but also a maximum threshold of similarity (defined by the user) between the fuzzy sets. Thus, when the similarity threshold is close to 0, the sets fuzzy sets are sufficiently separated, giving rise to descriptive fuzzy models, whereas when the similarity threshold is close to 1, the fuzzy sets can be very similar, producing fuzzy approximative models (non-interpretable).
4) Our method allows to deal with databases composed of numeric as well as categorical (or nominal) attributes. This is important because in many real-life problems both types of attributes are present. Our evolutionary multi-objective algorithm treats both types of attributes separately in the representation of individuals as well as in crossing and mutation operators. Chromosomes are divided into two parts (one for the numerical attributes

and another for the categorical ones); the numeric attributes are treated as fuzzy sets and optimized by real-parameter constrained optimization, and the categorical attributes are represented with integers and optimized by combinatorial optimization. Both types of attributes are merged by the inference engine to provide the classifier's predictions.

5) Both the multi-objective evolutionary algorithm for the search strategy and for fuzzy rules extraction have been designed with self-adaptive variation operators. In this way, it is not necessary to do preliminary experimentations to adjust the crossover and mutation probabilities.
6) The search strategy and the fuzzy classifier have both been included in the *Weka* platform [26] as official packages. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can be applied directly to a dataset or called from proprietary Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. Weka is open source software issued under the *GNU General Public License*. The proposed feature selection method can be easily configured using the Weka *Graphical User Interface*. Therefore, the proposed method can be compared with the rest of feature selection methods and classifiers implemented in the platform, as well as undergo statistical tests on a wide set of performance measures.

With respect to the methodology used for the validation of the proposed method, the following techniques have been applied:

1) Due to the high computational cost, the experiments have been oriented to establishing an acceptable setting with respect to the number of evaluations required by the search strategy and by the fuzzy rule-based classifier. To this end, statistical tests have been carried out on the run time, accuracy and number of selected attributes performance metrics.
2) We compared the proposed method in its best setting to a wide range of feature selection methods (79 in total) that includes filter, wrapper, univariate, and multivariate methods, with deterministic and probabilistic search strategies, and with evaluators of diverse natures.
3) To compare the performance of the 79 feature selection methods, the accuracy, weighted area under the *ROC* curve, root mean squared error and model size metrics have been used. To select the best methods, we propose a multi-objective decision making process to identify the non-dominated solutions of a multi-objective combinatorial optimization problem with 4 objectives (one for each performance metric).
4) Finally, the fuzzy rule-based classification model obtained with the proposed method has been evaluated with standard performance metrics and compared with other well-known fuzzy rule-based classifiers.

For the realization of the experiments we used a real-life dataset. *Operational* and *service* data have been extracted from a medium-sized *contact center* that operates both inbound and outbound communications, with different purposes, including customer care and follow-up, as well as marketing and quality control, and the aim of this classification problem is to evaluate *agents'* performances. Operational data include all the technical information needed to reconstruct a detailed history of the events that take place during each communication, and include, for example, the dialled or dialling phone number, the agent(s) that has (have) been involved, possible call transfers, and time-stamps. On the other hand, service data are specific to the particular service for which the contact has taken place, and may include, for example, all answers given by the interviewed subject during an outbound survey. Therefore, we may define this problem as a feature selection problem, whose purpose is to establish which subset of variables best *objectively* indicates of the performances of an agent. To this end, the center has collected the cumulative data, represented by agent, of a significant period of time and a significant range of different services, and asked to three, independent, supervisors to evaluate each involved agent. Such an evaluation plays the role of the *expert's view* of this problem, and the model we are searching for tries to predict such judgement.

The paper is organized as follows. Section II introduces basic concepts of *Multi-objective Constrained Optimization* and two *Multi-objective Evolutionary Algorithms* widely used in literature. It also briefly describes the feature selection process and how feature selection methods are implemented in Weka, as well as the *MultiObjectiveEvolutionarySearch* and *MultiObjectiveEvolutionaryFuzzyClassifier* classes implemented by the authors of this work and available in the Weka platform; Section III describes the datasets of agents of a contact center used for the realization of experiments, the preprocessing phase, and the feature selection method proposed in this paper; Section IV shows the experiments performed, the parameter setting, their results, an analysis of these based on the statistical comparison with other feature selection methods and other fuzzy rule-based classifiers, and an interpretation of the fuzzy model obtained with the proposed methodology in the context of the problem. Finally, in Section V we draw some final conclusions.

II. Background

Multi-objective constrained optimization. The term *optimization* [27] refers to selection of a best element, with regard to some criteria, from a set of alternative elements. *Mathematical programming* [28] deals with the theory, algorithms, methods and techniques to represent and solve optimization problems. In this paper we are interested in a class of mathematical programming problems called *multi-objective constrained optimization problems* [29], which can be formally defined, for l objectives and m constraints, as follows:

$$\begin{array}{lll} Min./Max. & f_i(\mathbf{x}), & i = 1, \ldots, l \\ subject\ to & g_j(\mathbf{x}) \leq 0, & j = 1, \ldots, m \end{array} \quad (1)$$

where $f_i(\mathbf{x})$ (usually called *objectives*) and $g_j(\mathbf{x})$ are linear or non-linear arbitrary functions. Optimization problems can be naturally separated into two categories: those with

discrete variables, which we call *combinatorial*, an those with continuous variables. In combinatorial problems, we are looking for objects from a finite, or countably infinite, set $\mathcal{X}$, typically integers, sets, permutations, or graphs. In problems with continuous variables, instead, we look for real parameters belonging to some continuous domain. In (1), $\mathbf{x} = \{x_1, x_2, \ldots, x_n\} \in \mathcal{X}^n$ represents the set of decision variables, where $\mathcal{X}$ is the domain for each variable x_k, $k = 1, \ldots, n$. Note that maximization problems can be defined symmetrically, and solved in an equivalent way.

Now, let $\mathcal{F} = \{\mathbf{x} \in \mathcal{X}^n \mid g_j(\mathbf{x}) \leq 0,\ j = 1, \ldots, m\}$ be the set of all feasible solutions to (1). We want to find a subset of solutions $\mathcal{S} \subseteq \mathcal{F}$ called *non-dominated set* (or *Pareto optimal set*). A solution $\mathbf{x} \in \mathcal{F}$ is *non-dominated* if there is not other solution $\mathbf{x}' \in \mathcal{F}$ that dominates $\mathbf{x}$, and a solution $\mathbf{x}'$ *dominates* $\mathbf{x}$ if and only if (1) there exists i ($1 \leq i \leq l$) such that $f_i(\mathbf{x}')$ improves $f_i(\mathbf{x})$, and (2) for every i ($1 \leq i \leq l$), $f_i(\mathbf{x})$ does not improve $f_i(\mathbf{x}')$. In other words, $\mathbf{x}'$ *dominates* $\mathbf{x}$ if and only if $\mathbf{x}'$ is better than $\mathbf{x}$ for at least one objective, and not worse than $\mathbf{x}$ for any other objective. The set $\mathcal{S}$ of non dominated solutions of (1) can be formally defined as:

$$\mathcal{S} = \{\mathbf{x} \in \mathcal{F} \mid \nexists\, \mathbf{x}'(\mathbf{x}' \in \mathcal{F} \wedge \mathcal{D}(\mathbf{x}', \mathbf{x}))\}$$

where:

$$\begin{aligned} \mathcal{D}(\mathbf{x}', \mathbf{x}) \quad \equiv \quad & \exists i (1 \leq i \leq l, f_i(\mathbf{x}') < f_i(\mathbf{x})) \wedge \\ & \forall i (1 \leq i \leq l, f_i(\mathbf{x}') \leq f_i(\mathbf{x})). \end{aligned}$$

Once the set of optimal solutions is available, the most satisfactory one can be chosen by applying a preference criterion. When all the functions f_i are linear, then the problem is a *linear programming problem* [30], which is the classical problem of mathematical programming and extremely efficient algorithms exist to obtain the optimal solution (e.g., the *simplex method* [31]). When any of the functions f_i is non-linear then we have a *non-linear programming problem* [32]. A non-linear programming problem in which the objectives are arbitrary functions is, in general, intractable. In principle, any search algorithm can be used to solve combinatorial optimization problems, although they are not guaranteed to find an optimal solution. *Metaheuristics* methods such as *evolutionary algorithms* [33] are typically used to find approximate solutions for complex multi-objective optimization problems, including feature selection and fuzzy classification.

The multi-objective evolutionary algorithms *ENORA* and *NSGA-II*. The multi-objective evolutionary algorithms *ENORA* [25], [34], [35], [36], [37] and *NSGA-II* [5], [38] use a $(\mu + \lambda)$ strategy with $\mu = \lambda = popsize$, where μ corresponds to the number of parents and λ refers to the number of children ($popsize$ is the population size), with *binary tournament selection* and a rank function based on Pareto fronts and *crowding*. The difference between *NSGA-II* and *ENORA* is how the calculation of the ranking of the individuals in the population is performed. In *ENORA* each individual belongs to a slot (as established in [37]) of the objective search space, and the rank of an individual in a population is the non-domination level of the individual in its slot. In *NSGA-II*, the rank of an individual in a population is the non-domination level of the individual in the whole population. *ENORA* uses the *fast non-dominated sorting* as *NSGA-II*. It compares each solution with the rest of the solutions and stores the results to avoid duplicate comparisons between every pair of solutions. For a problem with l objectives and a population with P solutions, the *fast non-dominated sorting* needs to conduct $l \cdot P \cdot (P-1)$ objective comparisons, which means that it has a algorithmic complexity of $O(l \cdot P^2)$. However, *ENORA* distributes the population in P slots (in the best case), therefore, the algorithmic complexity of *ENORA* is $O(l \cdot P^2)$ in the worst case, and $O(l \cdot P)$ in the best case.

Feature selection. *Feature Selection* is defined in [10] as the process of eliminating features from the dataset that are irrelevant to the task to be performed. It facilitates data understanding, reduces the measurement and storage requirements, reduces the computation time, and reduces the size of a dataset, so that model learning becomes an easier process. A selection method is basically a *search strategy* where the performance of candidate subsets is measured with a given *evaluator*. The search space for candidate subsets has cardinality $O(2^n)$, where n is the number of features. A *stopping criterion* establishes when the feature selection process must finish. It can be defined as a control procedure that ensures that no further addition or deletion of features does produce a better subset, or it can be as simple as a counter of iterations.

As discussed in the Introduction section, feature selection methods are typically categorized into *wrapper*, *filter* and *embedded*, as well as, orthogonally, into *univariate* and *multivariate* methods. Wrapper methods [15] use a predetermined learning algorithm to determine the quality of selected features by using some evaluation metric [16]; filter methods apply statistical measures to evaluate the set of attributes [12], [13], [14], while embedded methods achieve model fitting and feature selection simultaneously [19]. Finally, multivariate methods evaluate features in batches, while univariate methods rank each feature independently of the feature space.

Multi-objective evolutionary search. We use in this paper our *MultiObjectiveEvolutionarySearch* package of Weka. Our search strategy identifies non-dominated solutions to the following multi-objective optimization problem, which can be formulated as an instance of the problem (1) with $l = 2$ (two objectives) and $m = 0$ (no constraints) [39]:

$$\begin{array}{ll} Max./Min. & \mathcal{F}_{\mathcal{D}}(\mathbf{x}) \\ Min. & \mathcal{C}(\mathbf{x}) \end{array} \tag{2}$$

where $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ is a boolean set of decision variables, i.e. $x_k \in \{true, false\}$, $k = 1, \ldots, n$, being n the number of input attributes of a database $\mathcal{D}$. The problem (2) is, therefore, a *multi-objective boolean combinatorial optimization problem* where $x_k = 1$ represents that the variable x_k is selected, and $x_k = 0$ represents that the variable x_k is not selected, for all $k = 1, \ldots, n$. Depending on the type of the $\mathcal{F}_{\mathcal{D}}(\mathbf{x})$ function, which is defined with the evaluator, the resulting selection method can be a filter (when $\mathcal{F}_{\mathcal{D}}(\mathbf{x})$ is a statistical measure over the database $\mathcal{D}$) or a wrapper (when $\mathcal{F}_{\mathcal{D}}(\mathbf{x})$ is a performance measure - to be minimized

or maximized according to the metric - that evaluates the performance of a learning algorithm over the database $\mathcal{D}$). The function $\mathcal{C}(\mathbf{x})$ measures the number of selected features, i.e.

$$\mathcal{C}(\mathbf{x}) = \sum_{k=1}^{n} \eta(x_k),$$

where η is a function that transforms a boolean value into numeric ($true = 1$ and $false = 0$).

To solve (2), *ENORA* and *NSGA-II* have been implemented with *fixed-length binary representation*, *uniform random initialization*, *binary tournament selection*, ranking based on *non-domination level* with *crowding distance*, and *self-adaptive uniform crossover* and *one flip mutation* operators.

Multi-objective evolutionary fuzzy classification. We use in this paper our *MultiObjectiveEvolutionaryFuzzyClassifier* package of Weka. Our method identifies non-dominated solutions (fuzzy rule-based classifiers) of the following *multi-objective constrained optimization problem*, which can be formulated as an instance of the problem (1) with $l = 2$ (two objectives) and $m = 4$ (four constraints) [25]:

$$\begin{array}{ll} Max./Min. & \mathcal{F}_{\mathcal{D}}(\mathbf{\Gamma}) \\ Min. & \mathcal{NR}(\mathbf{\Gamma}) \\ subject\ to: & \mathcal{NR}(\mathbf{\Gamma}) \geq w \\ & \mathcal{NR}(\mathbf{\Gamma}) \leq M_{max} \\ & \mathcal{NL}(\mathbf{\Gamma}) \leq L_{max} \\ & \mathcal{S}(\mathbf{\Gamma}) \leq g_s \end{array} \tag{3}$$

where $\mathbf{\Gamma}$ is a fuzzy rule-based classifier composed by $\mathcal{NR}(\mathbf{\Gamma})$ fuzzy rules. Each fuzzy rule R_j^{Γ}, $j = 1, \ldots, \mathcal{NR}(\mathbf{\Gamma})$ has the following structure:

$$\begin{array}{ll} R_j^{\Gamma}: & if\ x_1\ is\ A_{1j}^{\Gamma} \wedge \ldots \wedge x_p\ is\ A_{pj}^{\Gamma} \wedge \\ & y_1\ is\ B_{1j}^{\Gamma} \wedge \ldots \wedge y_q\ is\ B_{qj}^{\Gamma} \quad \rightarrow z\ is\ C_j^{\Gamma}, \end{array}$$

where $x_i \in [l_i, u_i] \subset \mathbb{R}$, $i = 1, \ldots, p$, $p \geq 0$, are real input attributes, $y_i \in \{1, \ldots, v_i\}$, $i = 1, \ldots, q$, $q \geq 0$, $v_i > 1$, are categorical input attributes, and $z \in \{1, \ldots, w\}$, $w > 1$ is a categorical output attribute. Each fuzzy set A_{ij}^{Γ}, $i = 1, \ldots, p$, $j = 1, \ldots, \mathcal{NR}(\mathbf{\Gamma})$ is defined with a *gaussian membership function* [40]. In the problem (3), the function $\mathcal{F}_{\mathcal{D}}(\mathbf{\Gamma})$ is a performance measure of the classifier $\mathbf{\Gamma}$ over the database $\mathcal{D}$. The function $\mathcal{NR}(\mathbf{\Gamma})$ is minimized, and the constraints $\mathcal{NR}(\mathbf{\Gamma}) \geq w$ and $\mathcal{NR}(\mathbf{\Gamma}) \leq M_{max}$ limit the number of rules of the classifier $\mathbf{\Gamma}$ to the interval $[w, M_{max}]$, where w is the number of classes of the output attribute, and M_{max} is given by user. The constraint $\mathcal{NL}(\mathbf{\Gamma}) \leq L_{max}$ limits the number of linguistic labels of the real input variables to L_{max}. Finally, the constraint $\mathcal{S}(\mathbf{\Gamma}) \leq g_s$ ensures a maximum similarity g_s ($0 < g_s \leq 1$) between the fuzzy sets; the similarity value of a classifier $\mathbf{\Gamma}$ represents the maximum value of overlapping among their fuzzy sets for any input variable. The constraint $\mathcal{S}(\mathbf{\Gamma}) \leq g_s$ is handled by the multi-objective evolutionary algorithm by means of a repair algorithm, which is applied after the initialization of solutions, and after the crossing and mutation.

As reasoning method we use the *maximum matching* where, the *compatibility degree* of the rule R_j^{Γ} for the example $(\mathbf{x}, \mathbf{y})$ is calculated as:

$$\varphi_j^{\Gamma}(\mathbf{x}, \mathbf{y}) = \left(\phi_j^{\Gamma}(\mathbf{y}) + 1\right) \prod_{i=1}^{p} \mu_{A_{ij}^{\Gamma}}(x_i)$$

where $\phi_j^{\Gamma}(\mathbf{y})$ is the number of categorical input attributes, so that $y_i = B_{ij}^{\Gamma}$. The *compatibility degree* is obtained by applying a t-norm product to the degree of satisfaction of the clauses $x_i\ \ is\ A_{ij}^{\Gamma}$ multiplied by the number of matches of the categorical input data $y_i\ \ is\ B_{ij}^{\Gamma}$. The *association degree* of the example $(\mathbf{x}, \mathbf{y})$ with the class C, is calculated by summing the compatibility degrees of each rule R_j^{Γ} whose value for the categorical output attribute C_j^{Γ} is equal to C, that is:

$$\lambda_C^{\Gamma}(\mathbf{x}, \mathbf{y}) = \sum_{\substack{j = 1, \ldots, M_{\Gamma} \\ C_j^{\Gamma} = C}} \varphi_j^{\Gamma}(\mathbf{x}, \mathbf{y})$$

The *classification* for the example $(\mathbf{x}, \mathbf{y})$ or output of the classifier Γ, corresponds to the class C whose association degree is maximum, that is:

$$f_{\Gamma}(\mathbf{x}, \mathbf{y}) = \arg_C \max_{C=1}^{w} \lambda_C^{\Gamma}(\mathbf{x}, \mathbf{y})$$

For the problem (3), *ENORA* and *NSGA-II* have been implemented with *variable-length representation* with float-point and categorical input variables with a *Pittsburgh approach*, *uniform random initialization*, *binary tournament selection*, *handling constraints* using a *repair algorithm*, ranking based on *non-domination level* with *crowding distance*, *self-adaptive variation operators* which work on different levels of the fuzzy classifier: *fuzzy set crossover*, *rule crossover*, *rule incremental crossover*, *gaussian set center mutation*, *gaussian set variance mutation*, *fuzzy set mutation*, *rule incremental mutation*, and *integer mutation* (for categorical data). Once the fuzzy rule set has been extracted, a linguistic label is assigned to each fuzzy set.

III. Assessing Agents' Quality in a Contact Center

Evaluating the quality of the work that is being done by the employees is a central problem in modern business. Such an evaluation should be correct, fair, systematic and reliable, and, to this end, it should be measurable. For the purpose of experimenting the capabilities of our feature selection/fuzzy classification schema, we considered the problem of evaluating the quality of the work of operators (also called *agents*) in a medium-sized contact center. A *call center* is a set of resources, personnel, computers, and telecommunication equipment, which enable the delivery of services via telephone. Thanks to the advancements in information technology, call centers are gradually evolving into *contact centers*, in which the phone-operator role of agents is complemented, and sometimes substituted, by services offered through other technologies, such as faxing, instant messaging, web portals, and so on. Contact centers handle both inbound and outbound

communications, with different purposes, including customer care and follow-up, as well as marketing and quality control. As we shall see, compared to previous data mining experiments on contact center databases, the quality of the information at our disposal is considerably higher. Not only previous experiments such as [41] made no use of feature selection; they also operated on a very restricted set of attributes, consequently limiting the significance of their results. Moreover, all previous experiments, including [35], [42], were not designed to evaluate the performances of the agents.

Datasets. The data we have used have been provided by Northern Italy contact center GAP S.R.L., and consists of the cumulative performances data of 77 agents over a period of 6 months. Contacts in GAP are managed and organized as follows. The *flux* of information is categorized into *inbound* (that is, contacts that GAP receives, such as phone calls) and *outbound* (i.e., surveys made by GAP). Each of these is classified by commissions: a *commission* is the unit of contract between GAP and a client (i.e., the ACME airline commissions to GAP the phone ticket selling service for their customers), and each commission may be declined into several services. A *service* is a specific type of interaction that the client wants GAP to operate with (i.e., ACME wants GAP to deal with ticket selling but not lost-and-found), and each service includes several *sub-types* (i.e., ACME ticket selling includes a channel for information, a channel for reservation managing, and so on). For the purpose of this experiment, we considered phone-based communications only. Of all agents, 56 were employed for outbound, inbound, and back office services, while the remaining 21 had no inbound communications, naturally leading to two datasets: *ALL_AGENTS* (i.e., those who managed only outbound communications and back office services) and *INBOUND_AGENTS* (i.e., those who managed, among others, also inbound communications). The work of all agents has been described via 69 attributes, while for those agents with at least some inbound communications over the analysed period, we were able to add 6 more features (that make sense for inbound communications only).

The set of variables common to both datasets (the one containing the cumulative performance indicators of all agents and the one containing the cumulative performance indicators of only those agents that had inbound communications) can be classified into several categories, depending on the particular aspect they describe, for a better understanding. The first category is *agent related variables* (see Table I - top[1]), and includes their seniority (from 6 months to 5 and an half year), their gender (31 males versus 46 females), their age (from 19 to 65 years old), their level of education (from 1 - minimum compulsory education, to 5 - university degree or more), and their *skill* average and variance: GAP has internally engineered a skill-function that takes into account several aspects, recomputed weekly for each agent, and of which we consider the average and the variance over the entire period. A second category of variables is *work's diversity*, by means of which we want to measure how heterogeneous has been the agent's work in the analysed period. This category includes the number of *distinct sessions*[2] *and distinct commissions* the agent has worked on, the daily frequency of *context switches* (that takes into account switching between flows, or services, or service sub-types, weighted: farthest jumps weight the most), the daily frequency of *flow switches* (inbound vs. outbound), the daily frequency of *service switches*, and the the daily frequency of *sub-type switches*, and it is given in Table I (bottom). Moreover, we have taken into account how the agents' work has been *distributed* (Table II - top), by including the average and the variance over days of the number of minutes during which he/she has been effectively working (*management*), on inbound (*management inbound*), on outbound (*management outbound*) communications, or on back office (*management back office*), along with their fraction on the entire workload, that takes into account how many times the agent has declared him/herself *available* (in idle state), for how many minutes in total, *on break*, and for how many minutes, and *inactive* (that is, on break or available). The distribution takes also into account the *icc* index, which is an internal evaluation of the importance, complexity and criticality of the service being worked on. Finally, Table II (bottom) shows the variables relative to agents' *turns distribution*, that take into account in which part of the day and of the week each agent's shifts are mainly scheduled, as well as the fraction, over the entire observed period, of break, available, and inactive time of the agent.

Six more attributes have been considered for those agents whose job during the observed period included inbound communications. Such variables take into account the structure, the understandability, and the type of call-related *notes* written by the agent. These may be *articulated*, *non-articulated*, *domain-related*, *hybrid*, or *unrecognizable*.

The dataset *ALL_AGENTS* contains 69 input attributes and 77 instances, while the dataset *INBOUND_AGENTS* contains 75 input attributes and 56 instances. Both datasets have been enriched with a variable that describes the agent performance value. This has been obtained by asking to three independent supervisors a fair judgement of each agent *to the best of their expertise*. Their judgement, on a scale from 1 (lowest) to 5 (highest), takes into account the overall impression of the agents and their performances. Then, the three votes have been combined into a single one by averaging them. Four class labels are identified according to the average judgement: 'Low' $[1, 2)$, '$Medium$' $[2, 3)$, '$High$' $[3, 4)$, '$Excellent$' $[4, 5]$.

Preprocessing. For each of the two datasets we applied a simple preprocessing methodology. First, we have replaced all the missing values with their respective mean; to this end, the class *weka.filters.unsupervised.attribute.ReplaceMissingValues* has been used. Second, we have searched for those features with too small variation: no features have been eliminated via this process, indicating that, potentially, all of them might influence the agent judgement. We have used

[1]Unless otherwise specified, every numeric variable is in fact a pair of variables that takes into account average and variance of each aspect.

[2]A *session* is the most basic unit of work done by the agent, to which it is possible to assign a result, for example a phone call.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2019.2892363, IEEE Transactions on Fuzzy Systems

Agent related variables	
attribute	**semantics**
agent_seniority	# of days of service of the agent
agent_gender	whether the agent is a male or female
agent_age	age of the agent
agent_education	level of education of the agent
agent_skill	weekly avg. and var. of agents' skill
Diversity variables	
attribute	**semantics**
num_sessions	daily avg. and var. of the # of distinct sessions
num_commissions	daily avg. and var. of the # of distinct commissions
switch_index	daily avg. and var. of (all) switches
switch_index_flow_type	daily avg. and var. of flow switches
switch_index_ser_type	daily avg. and var. of service switches
switch_index_ser_same_type	daily avg. and var. of sub-service type switches
icc_inbound_av	daily avg. and var. of avg. icc index in inbound
icc_outbound_av	daily avg. and var. of avg. icc index in outbound
icc_inbound_var	daily avg. and var. of var. icc index in inbound
icc_outbound_var	daily avg. and var. of var. icc index in outbound

TABLE I
VARIABLES RELATED TO THE AGENT AND VARIABLES RELATED TO THE SWITCHING FREQUENCY OF THE AGENT.

Work distribution variables	
attribute	**semantics**
management	daily avg. and var. of # min. working
management_inbound	daily avg. and var. of # min. working on inbound comm.
management_outbound	daily avg. and var. of # min. working on outbound comm.
management_backoffice	daily avg. and var. of # min. working on back office
fraction_inbound	daily avg. and var. of the % of min. on inbound
fraction_outbound	daily avg. and var. of the % of min. on outbound
fraction_backoffice	daily avg. and var. of the % of min. on back office
available_sessions	daily avg. and var. of the # of available sessions
available	daily avg. and var. of # min. available
break_sessions	daily avg. and var. of the # of break sessions
break	daily avg. and var. of # min. on break
inactive_sessions	daily avg. and var. of the # of inactive sessions
inactive	daily avg. and var. of # min. inactive
Turn distribution variables	
attribute	**semantics**
turn_duration	daily avg. and var. of turn length in # min.
fraction_weekend	fraction of weekend workdays
fraction_night	daily avg. and var. of the % of min. working during nights
fraction_morning	daily avg. and var. of the % of min. working during mornings
fraction_early_afternoon	daily avg. and var. of the % of min. working during early aft.
fraction_late_afternoon	daily avg. and var. of the % of min. working during late aft.
fraction_evening	daily avg. and var. of the % of min. working during evening
inactivity_time	fraction of total inactivity time over total turn duration
available_time	fraction of total availability time over total turn duration
break_time	fraction of total break time over total turn duration

TABLE II
VARIABLES RELATED TO THE AGENT'S WORK DISTRIBUTION AND HETEROGENEITY, AND TURN DISTRIBUTION.

Notes' structure variables	
attribute	**semantics**
fraction_abbreviated	fraction of abbreviated notes
fraction_articulated	fraction of articulated notes
fraction_non_articulated	fraction of non articulated notes
fraction_hybrid	fraction of hybrid notes
fraction_unrecognized	fraction of unrecognized notes
fraction_domain	fraction of domain-related notes

TABLE III
VARIABLES RELATED TO THE AGENT'S NOTES.

the class *weka.filters.unsupervised.attribute.RemoveUseless* for this task.

Feature selection for fuzzy classification. As we have mentioned, one of the drawbacks of fuzzy classification is that it generates little interpretable models in the presence of many attributes. In these cases, a process of selection of attributes, prior to fuzzy rule extraction, is required. In this paper we propose the following multivariate wrapper feature selection method for later use in fuzzy classification. As for the **search strategy**, we propose our *MultiObjectiveEvolutionarySearch* method with the multi-objective evolutionary algorithm *ENORA*. As shown in [35], [36], the performance of *ENORA* are generally better than those of *NSGA-II* in terms of *hypervolume* [43], and better than other single-objective search strategies. Our **evaluator** consists of a wrapper with the fuzzy rule-based **classifier** based, again, on *ENORA*, which outperforms *NSGA-II* in this task as well [25], driven by the accuracy (*ACC*) as a **measure**. The *ACC*-guided search has given better results in the preliminary experiments than the search using the *area under the ROC curve* (*AUC*), which gives very poor values of *ACC*. As **stopping criterion** we use a simple limit on the number of generations. The proposed method basically consists of a multi-objective evolutionary algorithm (search strategy) where, for the evaluation of a candidate attribute subset, a wrapper (*WrapperSubsetEval*) based on the *MultiObjectiveEvolutionaryFuzzyClassifier* classifier and accuracy performance metric is used. Therefore, to evaluate a candidate attribute subset, a multi-objective evolutionary algorithm is executed to extract a fuzzy rule set on the reduced database, which is evaluated with cross-validation on the accuracy metric. The following steps are then performed for the evaluation of an attribute subset $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$, $x_k \in \{true, false\}$, $k = 1, \ldots, N$, in a database $\mathcal{D}$:

STEP 1. Remove from the database $\mathcal{D}$ those attributes x_k such that $x_k = false$, obtaining a reduced database $\mathcal{D}$';

STEP 2. Run *MultiObjectiveEvolutionaryFuzzyClassifier* over the database $\mathcal{D}$' to extract a fuzzy rule set.

STEP 3. Evaluate the fuzzy rule set with cross-validation using the accuracy performance metric;

STEP 4. Return the 'merit' of the attribute subset **x**.

Considering the wideness of the search space ($O(2^N)$) for feature selection, and the intrinsic complexity of fuzzy rule extraction (for each attempted candidate), it is crucial to adjust the evolution parameters of both evolutionary multi-objective algorithms to obtain a good trade-off between accuracy and run time.

IV. EXPERIMENTS, RESULTS AND DISCUSSION

In this section, we describe the set of experiments that have been performed to show the effectiveness of the proposed methodology. Three blocks of experiments have been carried out: the first one aims to find the optimal number of generations of both the search strategy and the evaluator for an adequate trade-off between performance and run time; in the second block, the proposed method is compared with other multivariate, univariate, filter and wrapper feature selection methods; in the third block, we compare the fuzzy classifiers obtained by the proposed method with the fuzzy classifiers obtained by well-known methods. Finally, the rules of our best fuzzy models are interpreted in the context of the contact centre. All our experiments have been run on a machine with 8 processors Intel Xeon X7550 at 2.00 GHz, 1TByte of RAM at 1067MHz, and storage Lustre Distributed File System v2.5.2 - Interconnection network: Infiniband QDR (40Gbps).

# Configuration	Number of generations Evolutionary search	Number of generations Fuzzy classifier
#1	10	10
#2	10	100
#3	100	10

TABLE IV
THREE PARAMETER CONFIGURATIONS STUDIED IN THIS PAPER.

Configuration	Run time	Accuracy	Number of attributes
INBOUND_AGENTS			
#1	158300955.5	0.50607	9.7
#2	1140188340.7	0.57998	6.0
#3	553075217.9	0.56	3.5
ALL_AGENTS			
#1	148756602.7	0.50857	7.6
#2	1200222302.3	0.56128	8.0
#3	688512492.7	0.54831	5.5

TABLE V
AVERAGE RUN TIME, ACCURACY AND NUMBER OF ATTRIBUTES

Optimal number of generations. Given that the proposed feature selection method consists of an evolutionary algorithm that, in turn, uses another evolutionary algorithm to evaluate each individual population, the run time may be intolerable if the number of generations for each of the evolutionary algorithms is not properly adjusted. To this aim, we have established three different configurations, shown in the Tab. IV. We executed 10 times our feature selection method in each of the three configurations. Table V shows a summary of results with the averages of run time (ms.), accuracy and number of selected attributes. In order to test if the differences between the means of each of the configurations are statistically significant, statistical tests have been performed. First, the conditions to apply the parametric test *one way repeated measures ANOVA*, normality and sphericity, have been checked. Only the dataset *ALL_AGENTS* for the number of attributes meets these conditions (Shapiro-Wilk normality test p-values 0.119, 0.381 and 0.8 for configurations #1, #2 and #3 respectively; Mauchly test for sphericity p-value 0.245). In this case, one way repeated measures ANOVA has been performed. For the rest of cases, Friedman test has been applied. When statistically significant differences are detected in the Friedman test, the *Nemenyi post-hoc* test has been applied to locate where these differences are.

Regarding the run time, configuration #1 was the best one, as expected. Configuration #3 behaved better than the configuration #2, although the differences are not statistically significant. Regarding accuracy, no statistically significant differences between the configurations #2 and #3 were found, and both behaved better than configuration #1. Finally, regarding the number of selected attributes, there are no statistically significant differences between the three configurations. This analysis allowed us to conclude that the worst configuration is #1 regarding to accuracy and, since there are no statistically significant differences between configurations #2 and #3, we opted for configuration #3 according to the *principle of minimum description length*. Therefore, configuration #3 has been used for feature selection. The remaining parameters were set as follows: *populationSize* was 100 in both the search and the classification; *maxRules* was 14 (i.e., 10 plus the number of classes); *maxSimilarity* was 0.1 (allowing only minimally overlapping fuzzy sets); *minV* was 2.0 and *maxV* was 30.0 (default values). Among the 3000 individuals of the final population in the model with configuration #3, we have chosen the one with the best accuracy; given that the number of chosen attributes is low enough, no further *a posteriori* decision method was necessary.

Dataset	Selected attributes	Rank	Importance
INBOUND_AGENTS	switch_index_ser_type	3	0.01071
	switch_index_ser_same_type	4	-0.00357
	icc_inbound_av	2	0.02857
	turn_duration	1	0.06786
ALL_AGENTS	management	1	0.418
	break_sessions	4	0.301
	icc_inbound_var	3	0.348
	fraction_night	2	0.377

TABLE VI
SELECTED ATTRIBUTES AND THEIR RANKS WITH THE PROPOSED FEATURE SELECTION METHOD FOR *INBOUND_AGENTS* AND *ALL_AGENTS* DATASETS.

Identifying the best attributes and their importance. Tab. VI shows the selected attributes and their ranks and importances for each of the datasets. The rank and importance of the attributes has been obtained through a univariate wrapper feature selection method, where the search strategy is the *ranker* method, and the *evaluator* is *ClassifierAttributeEval* with *classifier* = *MultiObjectiveEvolutionaryFuzzyClassifier* (*algorithm* = *ENORA*, *generations* = 1000, *populationSize* = 100, *evaluationMeasure* = *ACC*, *maxLabels* = 3, *maxRules* = 4, *maxSimilarity* = 0.4, *minV* = 30.0, *maxV* = 2.0, *seed* = 1), *evaluationMeasure* = *ACC*, and *leaveOneAttributeOut* = *true*. An attribute is evaluated by measuring the impact of leaving it out from the full set. Before analyzing the behaviour of our method from the numerical point of view, and comparing its performances with those of other methods, let us observe how the selected features may help assessing the quality of agents' work. Recall that each agents has been previously classified by three, independent experts, and that the combination of their judgment has been used as class. As for the dataset *INBOUND_AGENTS*, the most important attribute that has been selected is *turn_duration*, while the second most important is *icc_inbound_av*. This means that in order to automatically predict the quality of an agent with inbound communications, the average duration of their turn, and the average complexity of their task are key values. On the other hand, for agents with outbound tasks only (*ALL_AGENTS* dataset), the average quantity of -effective- working time, and how much of the agents' work takes place during night, seem to determine his/her quality. These elements can be used to design a simple system that, using the rules discussed later, may help the management to optimize turn and work distribution to ensure higher standards.

Comparing the results with other selection methods. The experiments performed in this section aim to answer the following questions:

1) Which feature selection method produces the best performance for fuzzy classification?

2) How does the proposed feature selection method behave with other classifiers?

In order to answer these questions we have systematically applied a very wide range of feature selection methods, all available in the literature. Each method is, in itself, a combination of a specific choice among the search strategies, the evaluators, and the evaluation metrics (in the case of wrapper methods). In the following, we describe our choices for search methods and evaluators.

Let us consider possible search methods. As for univariate methods, we used the *Ranker* method [44], which ranks attributes by their individual evaluations. As for multivariate methods, among **deterministic search strategies** we considered: *BestFirst* [45], *GreedyStepwise* [46], *LinearForwardSelection* [47], and *InfoGain* [48], while the employed **probabilistic algorithms** are: *MultiObjectiveEvolutionarySearch* (already described in Section II), *PSOSearch* [49], and *GeneticSearch* [50]. *BestFirst* implements *beam search*, and searches the space of attribute subsets by greedy hill climbing augmented with a backtracking capability; the amount of backtracking may be customized by specifying the beam width. It supports forward, backward, and bi-directional search directions. *GreedyStepwise* performs a greedy forward or backward search through the space of attribute subsets, stopping when the addition (forward direction) or deletion (backward direction) of any of the remaining attributes results in a decrease in evaluation, thus, it has no backtracking capability. *LinearForwardSelection* is an extension of *BestFirst*, supporting simple forward or floating forward search directions. The latter considers a number of consecutive single-attribute elimination steps after each forward step, as long as this results in an improvement. The algorithm takes only a restricted number of k attributes into account, with the goal of reducing the number of evaluations performed during the search and producing a compact final subset, by two possible modes of operation: *fixed-set* or *fixed-width*. According to the former, all single attributes are initially ranked, and then the top-k are passed as input to forward selection. The latter employs a similar initial ranking criterion, starting the search with the top-k attributes; however, it maintains a fixed number of k candidates also in each of the subsequent forward selection steps, by adding further attributes from the initial ranked list (as long as any remain). Finally, the *InfoGain* strategy works by listing all features, ordered by their individual scores, as determined by measuring the information gain score with respect to the class. As far as probabilistic algorithms are concerned, *genetic* (or *evolutionary*) algorithms are the most common choice. Genetic algorithms were first proposed for attribute selection in [51], and are now considered an important tool for the selection of features [52]. They are inspired by the process of natural selection and, through the application of *elitist selection*, iteratively generate better and better solutions to optimization and search problems, by employing operators such as *mutation* and *crossover*. The goodness of a solution is determined through the use of one (single-objective) or more (multi-objective) *fitness* functions. In the present work, for the purpose of attribute selection (in those cases in which we choose multi-objective optimization), two objectives are optimized: the first one is chosen by the evaluator, and it is to be maximized, while the second one is the attribute subset cardinality, and it is to be minimized. The final output is given by the non-dominated solution in the last population having the best fitness score for the first objective. *GeneticSearch* implements the simple, classical Goldberg's (single-objective) *Genetic Algorithm* for searching. Finally, *PSOSearch* explores the attribute space employing the *Particle Swarm Optimization* (*PSO*) algorithm. *PSO* optimizes a problem iteratively, trying to improve a candidate solution with regard to a given measure of quality. Similarly to evolutionary computation techniques, it considers a population of candidate solutions, called particles. Elements are moved around the search space according to mathematical formulae, considering each particle's characteristics and the overall "swarm knowledge", following an agent-oriented paradigm.

Now, let us examine the different evaluators that we considered. As much as **multivariate filters** are concerned, we used *CfsSubsetEval* [53] and *ConsistencySubsetEval* [54]. *CfsSubsetEval* evaluates the worthiness of an entire subset of features by considering the individual predictive power of each attribute, together with the degree of redundancy between them. Subsets containing attributes that are highly correlated with the class, and not strongly correlated with one another, are preferred. On the contrary, *ConsistencySubsetEval* scores a subset of features as a whole, by projecting the training instances according to the attribute subset, and considering the consistency of class values in the obtained instance sets. For possible **univariate filters**, *GainRatioAttributeEval* [55], *SignificanceAttributeEval* [56] and *SymmetricalUncertAttributeEval* [57] were considered. *GainRatioAttributeEval* evaluates the worthiness of a single attribute by measuring its *gain ratio* value with respect to the class labels. Gain ratio is a well-known, commonly used assessment measure, calculated as the difference between the entropy of class distribution minus the conditional entropy of the classes given the values of the attribute, divided by the entropy of the attribute itself. *SignificanceAttributeEval* scores a single attribute by computing its *probabilistic significance* as a two-way function of its association to the class decision. The intuition behind this algorithm is that if an attribute is significant with respect to the class labels, then it is expected that different sets of elements with complementary sets of values for the attribute will also belong to complementary sets of classes. Finally, *SymmetricalUncertAttributeEval* evaluates the worthiness of a given attribute by measuring its *symmetrical uncertainty* with respect to the class. Finally, as possible **wrappers**, we used *WrapperSubsetEval* [15] for multivariate methods and *ClassifierAttributeEval* [58] for univariate methods, in conjunction with the classifiers *J48* (C4.5 [19]), *LibSVM* [59] and *RandomForest* [60], and with the metrics *ACC*, *weighted area under the ROC curve* (*WAUC*), and *root mean squared error* (*RMSE*). *J48* is a Java implementation of the widely-used decision tree learner *C4.5*, which is known to be computationally efficient. The learning algorithm builds a decision tree from a set of labelled training instances in a recursive fashion, starting from the root node, by using the *information gain*

ratio criterion. *LibSVM* is a library for *support vector machines* learning. A support vector machine is a supervised machine learning algorithm, which can be used for both regression and (typically binary) classification problems. Each instance is mapped to a point in n-dimensional space, where n is the number of features characterizing the instance. Then, in a binary classification setting, a hyperplane is constructed, that optimally divides the instances in homogeneous groups with respect to the class labels. *RandomForest* is an *ensemble learning* method which constructs a forest of random trees, for classification or regression purposes. A typical problem of decision trees is their propensity to overfit, if not properly pruned: in the literature, they are regarded as models having low bias, but high variance. In *RandomForest* each tree is built from a separate part of the same training set, reducing the variance, thus contrasting the tendency of a large, single tree to overfit. Given a new instance to classify, the final output is obtained by combining the results given by the different trained models. *ACC* measures the amount of correctly labelled instances, as classified by a model. It is given by the ratio between the number of correctly classified instances and the number of total instances. *WAUC* metric is calculated on a *ROC* curve [61], [62], which is a graphical representation of the *sensitivity* versus *specificity* for a classifier system, obtained by varying the model class discrimination threshold. The *WAUC* value belongs to the interval $[0, 1]$; a score of 1 represents the perfect classifier, while 0.5 is typical of a random classification behaviour; this number is computed taking into account also the cardinality/weight of each class. *RMSE* measures the difference between values predicted by a model and the values actually observed.

By combining the above choices one may end up obtaining as much as 79 feature selection methods, each one of them optimized following a different criterium. In order to choose the best reduced databases, we considered the following multi-objective combinatorial optimization problem:

$$\begin{aligned} Maximize \quad & f_1(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n} ACC(\mathbf{x}, j) \\ Maximize \quad & f_2(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n} WAUC(\mathbf{x}, j) \\ Minimize \quad & f_3(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n} RMSE(\mathbf{x}, j) \\ Minimize \quad & f_4(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n} MS(\mathbf{x}, j) \end{aligned} \tag{4}$$

The problem (4) is an instance of the problem (1) with $l = 4$ (four objectives) and $m = 0$ (no constraints), where $\mathbf{x} \in DB$ is a database ($DB = \{1, \ldots, 79\}$), and $n = 3$ is the number of classifiers (*J48*, *RandomForest* and *LibSVM*). The function $ACC(\mathbf{x}, j)$ is the accuracy of the classifier j for the database $\mathbf{x}$, $WAUC(\mathbf{x}, j)$ is the weighted area under the *ROC* curve of the classifier j for the database $\mathbf{x}$, $RMSE(\mathbf{x}, j)$ is the root mean squared error of the classifier j for the database $\mathbf{x}$, and $MS(\mathbf{x}, j)$ is the serialized model size of the classifier j for the database $\mathbf{x}$. The solution to the problem (4) is a set of 4 non-dominated databases for *INBOUND_AGENTS* and a set of 9 non-dominated databases for *ALL_AGENTS* (all shown in Tab. VII). We compared the performances of these selections (i.e., the best selections obtained with classical methods) against the performances of the selection obtained by the proposed method (hereafter called, generically, *MOES-MOEFC-ACC*). Tab. VIII shows the result of such a comparison for *INBOUND_AGENTS* under the accuracy metric (top) and under the area under the *ROC* metric (bottom) when we tried to learn a classifier with *J48*, *RandomForest*, *LibSVM*, and the multi-objective fuzzy rule-based classifier *MOEFC* (cfr. Section II), the latter having been run with *populationSize* set to 100, *generations* set to 1000, *maxRules* set to 14, *maxLabels* set to 7, *maxSimilarity* set to 0.1, *minV* set to 2.0, and *maxV* set to 30.0. Tab. IX shows the same comparison *ALL_AGENTS*. Both comparisons include the respective original dataset (no selection). In both tables, the results of the experiments have been analysed through a *paired t-test corrected*, with 0.05 significance (being *MOES-MOEFC-ACC* the test base). For each result, a mark $*$ denotes that the result is statistically worse than the test base; similarly, a mark v denotes a statistically better result, and no mark denotes no statistically meaningful difference.

Comparing the results with other fuzzy classifiers. Here we compare our classification results with those obtained by other fuzzy rule-based classifiers from the *R package frbs* [63] and from Weka. The following four algorithms have been considered: *FRBCS.CHI* [2], *FRBCS.W* [3], and *FH.GBML* [4]. *FRBCS.CHI* extends Wang and Mendel's method [1] for tackling classification problems, and it is quite similar to their technique. However, since it is based on the Fuzzy Rule-Based Classification Systems (FRBCS) model, Chi's method only takes class labels on each data to be consequent parts of fuzzy IF-THEN rules. In other words, rules are generated as in Wang and Mendel's technique and, then, the consequents are replaced with their classes. Regarding calculating degrees of each rule, they are determined by the antecedent part of the rules, and redundant rules can be deleted by considering their degrees. *FRBCS.W:* implements the *second type* of FRBCS, which has certainty grades (weights) in the consequent parts of the rules. The antecedents are determined by a grid-type fuzzy partition from the training data. The consequent class is defined as the dominant class in the fuzzy subspace corresponding to the antecedents of each fuzzy IF-THEN rule. The class of a new instance is determined by the consequent class of the rule with the maximum product of its compatibility and certainty grades. The compatibility grade is determined by aggregating degrees of the membership function of antecedents, while the certainty grade is calculated from the ratio among the consequent class. *FH.GBML* is an hybrid algorithm of two fuzzy genetics-based machine learning approaches (i.e., Michigan and Pittsburgh) for designing fuzzy rule-based classification systems. Finally, *FURIA* [64] (Fuzzy Unordered Rule Induction Algorithm) is implemented as a Weka package that extends the well-known RIPPER rule learner, while preserving its advantages, such as simple and comprehensible rule sets. In addition, it includes a number of modifications and extensions. In particular, *FURIA* learns fuzzy rules instead of conventional rules and unordered rule

#Id	Database name	Search strategy	Evaluator
		INBOUND_AGENTS	
#1	*BF-RF-RMSE*	*BestFirst*	*RandomForest (RMSE)*
#2	*GS-LSVM-ACC*	*GreedyStepwise*	*LibSVM (ACC)*
#3	*MOES-J48-ACC*	*MultiObjectiveEvolutionarySearch*	*J48 (ACC)*
#4	*MOES-RF-WAUC*	*MultiObjectiveEvolutionarySearch*	*RandomForest (WAUC)*
		ALL_AGENTS	
#1	*BF-LSVM-ACC*	*BestFirst*	*LibSVM (ACC)*
#2	*BF-RF-ACC*	*BestFirst*	*RandomForest (ACC)*
#3	*BF-RF-RMSE*	*BestFirst*	*RandomForest (RMSE)*
#4	*LFS-LSVM-WAUC*	*LinearForwardSelection*	*LibSVM (WAUC)*
#5	*MOES-J48-RMSE*	*MultiObjectiveEvolutionarySearch*	*J48 (RMSE)*
#6	*MOES-LSVM-RMSE*	*MultiObjectiveEvolutionarySearch*	*LibSVM (RMSE)*
#7	*MOES-RF-ACC*	*MultiObjectiveEvolutionarySearch*	*RandomForest (ACC)*
#8	*MOES-RF-RMSE*	*MultiObjectiveEvolutionarySearch*	*RandomForest (RMSE)*
#9	*PSOS-LSVM-RMSE*	*PSOSearch*	*LibSVM (RMSE)*

TABLE VII
REDUCED DATABASES OBTAINED BOTH PROBLEMS AS A RESULT OF SOLVING (4).

	MOES-MOEFC-ACC	***#1***	***#2***	***#3***	***#4***	***INBOUND_AGENTS***
			ACC			
MOEFC	59.40	38.53∗	42.67∗	55.23	55.17	3.33∗
J48	50.10	42.80	40.13	65.60	61.57	38.90
RandomForest	45.73	42.80	53.87	67.10v	68.77v	56.90
LibSVM	41.43	39.20	74.27v	55.03v	55.03v	42.80
ZeroR	42.80	42.80	42.80	42.80	42.80	42.80
			WAUC			
MOEFC	0.63	0.50∗	0.53	0.60	0.59	0.50∗
J48	0.62	0.50∗	0.55	0.74	0.72	0.58
RandomForest	0.64	0.50∗	0.66	0.78v	0.81v	0.68
LibSVM	0.49	0.50	0.80v	0.62v	0.62v	0.50
ZeroR	0.50	0.50	0.50	0.50	0.50	0.50

TABLE VIII
RESULTS OF 10-FOLD CROSS-VALIDATION 10 ITERATIONS FOR THE *INBOUND_AGENTS* PROBLEM.

	MOES-MOEFC-ACC	***#1***	***#2***	***#3***	***#4***	***#5***	***#6***	***#7***	***#8***	***#9***	***ALL_AGENTS***
					ACC						
MOEFC	60.12	44.89∗	56.77	55.91	43.98∗	48.21∗	52.07	55.00	51.62	43.64∗	6.25∗
J48	55.00	43.55	53.75	60.87	52.43	68.43v	44.79	60.27	58.21	47.18	45.98
RandomForest	59.16	54.14	66.70	64.59	52.68	57.95	54.36	69.25	68.27	54.29	56.34
LibSVM	46.41	64.48v	41.64	41.64	65.05v	46.16	65.05v	41.64	45.30	66.23v	41.64
ZeroR	41.64	41.64	41.64	41.64	41.64	41.64	41.64	41.64	41.64	41.64	41.64
					WAUC						
MOEFC	0.62	0.50∗	0.58	0.59	0.52∗	0.58	0.51∗	0.58	0.57	0.52∗	0.50∗
J48	0.66	0.59	0.68	0.72	0.65	0.77	0.59	0.72	0.69	0.62	0.60
RandomForest	0.74	0.68	0.80	0.82	0.69	0.73	0.72	0.81	0.84v	0.71	0.71
LibSVM	0.54	0.71v	0.50	0.50	0.72v	0.56	0.72v	0.50	0.53	0.73v	0.50
ZeroR	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50

TABLE IX
RESULTS OF 10-FOLD CROSS-VALIDATION 10 ITERATIONS FOR THE *ALL_AGENTS* PROBLEM.

sets instead of rule lists. Moreover, to deal with uncovered examples, it makes use of an efficient rule stretching method.

The results of comparing multi-objective fuzzy rule-based classification with the models obtained with *FRBCS.CHI*, *FRBCS.W* and *FH.GBML* over the reduced database *MOES-MOEFC-ACC* are shown in Tab. X. We considered the accuracy of the resulting classifiers both in full training and in 10-fold cross-validation with 10 repetitions, as well as their kappa statistic, number of rules, fuzzy set form (gaussian, trapezoidal, triangular), number of linguistic labels and run time in full training.

Analysis of the results. Tab. VIII and IX can be read in two orthogonal ways:

- First, we may compare how the fuzzy rule-based classifier based on ENORA behaved on selections of features different from *MOES-MOEFC-ACC*, that is, on the best selections obtained by other methods. In this sense, our selection resulted the most performing one for both problems, beating the second best selection by 7.17 accuracy points in the case of *INBOUND_AGENTS* and by 3.21 points in the case of *ALL_AGENTS*. Comparing them by weighted area under the *ROC* curve shows the same behaviour.
- Second, we may compare the accuracies (and the areas under the *ROC* curve) of the models learned by different classifiers on the selection *MOES-MOEFC-ACC*. The fuzzy rule-based classifier obtained, by far, better models than any other classifier, beating the second best one by

	MOEFC	*FRBCS.CHI*	*FRBCS.W*	*FH.GBML*	*FURIA*	*MOEFC*	*FRBCS.CHI*	*FRBCS.W*	*FH.GBML*	*FURIA*
	INBOUND_AGENTS					*ALL_AGENTS*				
Accuracy full training set	0.7857	0.6607	0.7143	0.7679	0.5357	0.7403	0.5714	0.5714	0.6364	0.6234
Kappa statistic full training set	0.666	0.485	0.5762	0.6396	0.2385	0.5877	0.3465	0.3465	0.4111	0.4385
Average accuracy 10-fold CV (10 rep.)	0.5940	0.5544	0.5431	0.5723	0.4649	0.6013	0.5459	0.6004	0.5567	0.5714
Average kappa statistic 10-fold CV (10 rep.)	0.3751	0.3018	0.296	0.3243	0.2448	0.3749	0.2728	0.3661	0.2761	0.3265
Maximum number of rules	14	–	–	14	–	14	–	–	14	–
Found number of rules	8	21	43	8	3	5	49	46	4	6
Fuzzy set form	Gaussian	Trapezoidal	Gaussian	Triangular	Trapezoidal	Gaussian	Trapezoidal	Gaussian	Triangular	Trapezoidal
Maximum number of linguistic labels for each variable	7	3	21	14	–	7	15	15	4	–
Run time full training set	36.12 s.	1.230 s.	0.399 s.	4.11 h.	0.01 s.	40.5 s.	1.552 s.	0.523 s.	4.86 h.	0.01 s.
Number of generations	1000	–	–	1000	–	1000	–	–	1000	–
Population size	100	–	–	100	–	100	–	–	100	–
Fuzzy sets maximum similarity	0.1	–	–	–	–	0.1	–	–	–	–

TABLE X
COMPARING FUZZY RULE-BASED CLASSIFIERS' METRICS.

9.30 points in the case of *INBOUND_AGENTS* and by 0.96 points in the case of *ALL_AGENTS*. It is worth to observe that it is somehow unfair to compare very interpretable classifiers such as the one based on fuzzy rules against non-interpretable ones such as *RandomForest*; in this sense, in the case of *ALL_AGENTS*, the difference between the fuzzy model and the next interpretable one, based on a decision tree, is 5.12 points in favour of the former. Moreover, the selection *MOES-MOEFC-ACC* almost always improves the accuracy of every classifier learning method that we have tried against the original dataset, in both problems.

As far as the behaviour of other fuzzy rule-based classifiers is concerned, the accuracy obtained in full training mode by the multi-objective learner is much better than the one obtained by any other method; such a difference is smaller in cross-validation mode, but it is still there. In terms of interpretability, *MOEFC* found (much) less rules than *FRBCS.CHI* and than *FRBCS.W*, while the number of rules found by *FH.GBML* is the same (8 rules) in the case of *INBOUND_AGENTS*, and one less (4 rules instead of 5 rules) in the case of *ALL_AGENTS*. Although *FURIA* has found very compact models (3 and 6 rules for *INBOUND_AGENTS* and *ALL_AGENTS* respectively), the accuracy in both full training set and 10-fold cross-validation is much worse than the accuracy obtained with *MOEFC*. Therefore, we can conclude that *MOEFC* behaves generally better than all three other fuzzy classifiers. In particular, observe that the only classifier whose performances are comparable with those of *MOEFC* on this problem, that is, *FH.GBML*, presented a run time several orders of magnitude higher: a few seconds (*MOEFC*) against a few hours (*FH.GBML*) with the same number of generations and the same cardinality for the population.

Final model(s). The experiments described and discussed above aimed at comparing the results of the proposed selection method with those of other, classical, selection methods, as well as the performances of the multi-objective fuzzy rule-based learner with those of other, classical, fuzzy rule system learners. In this section we compute a final model over the attributes selected earlier, and we discuss its characteristics in the context of assessing the quality of the agents. For this purpose, the *MOEFC* classifier has been executed over the *MOES-MOEFC-ACC* database with the following parameters:

- We set *maxRules* to 4 (where 4 is the number of output classes) and *maxLabels* = 3 (*Low*, *Medium* and *High*), that is, the minimum number of possible rules and a very low number of labels, to maximize the simplicity, that is, the interpretability, of the resulting model;
- We set *maxSimilarity* to 0.4, that is, a value sufficiently high to guarantee enough search space, and sufficiently low to guarantee that the linguistic labels do not overlap too much and can be distinguished from each other;
- We set *generations* to 100000, to maximize the accuracy.

	INBOUND_AGENTS	*ALL_AGENTS*
Accuracy full training set	0.7143	0.7143
Kappa statistic full training set	0.5564	0.5445
Accuracy 10-fold CV	0.6607	0.6234
Kappa statistic 10-fold CV	0.4748	0.4002
Number of rules	4	4
Fuzzy set form	Gaussian	Gaussian
Maximum number of linguistic labels for each variable	3	3
Run time full training set	1271.59 s.	1635.38 s.
Number of generations	100000	100000
Population size	100	100
Fuzzy sets maximum similarity	0.4	0.4

TABLE XI
PERFORMANCES OF THE FINAL MODELS.

Tab. XII shows the models, and Tab. XI their performances. For their interpretation, focus, first, on the problem *INBOUND_AGENTS*. Recall that this is the problem of evaluating the performances (Low, $Medium$, $High$ or $Excellent$) of *versatile* agents, that is, those that manage al types of communications. By interpreting our model, we learn that agents with higher rate of switching among *services* are generally worse than those with lower rates, but that higher rates of switching among *sub-services* generically indicates better agents. That is, for agents that deal with all types of communication, switching among services is somehow delicate: changing too often from a service to another negatively influences the (perceived) performances, while changing from a sub-service to another has a positive result. Moreover, and somehow unexpectedly, agents with lower rates of switching and longer turns seem to show better performances.

Let us focus, now, on the problem of classifying agents that manage only outbound communications, that is, *specialized* agents. The overall daily workload, that is, *management*, emerges as the most important variable: consistently with the case of *INBOUND_AGENTS*, the higher workload the better the agent. Moreover, higher rates of *icc* changes, that is, higher rates of changes, during the day, of the relative importance of the tasks being carried out, as well as too low or too high level of night work, are associated with extreme evaluations

INBOUND_AGENTS										
Fuzzy rule set										
IF	x_1 is $Moderately\ High$	AND	x_2 is $Moderately\ Low$	AND	x_3 is $Medium$	AND	x_4 is Low	THEN	y is Low	
IF	x_1 is $Moderately\ High$	AND	x_2 is Low	AND	x_3 is $Medium$	AND	x_4 is Low	THEN	y is $Medium$	
IF	x_1 is Low	AND	x_2 is $Moderately\ Low$	AND	x_3 is $Medium$	AND	x_4 is $High$	THEN	y is $High$	
IF	x_1 is Low	AND	x_2 is Low	AND	x_3 is $High$	AND	x_4 is $High$	THEN	y is $Excellent$	
Gaussian fuzzy sets										
Attribute			**Name**		**Center**		**Standard Deviation**		**Linguistic label**	
x_1			switch_index_ser_type		960.7785		1152.5188		*Low*	
					3466.7129		1286.9339		*Moderately High*	
x_2			switch_index_ser_same_type		0.1893		0.6861		*Low*	
					1.5343		0.5784		*Moderately Low*	
x_3			icc_inbound_av		350.6354		203.0594		*Medium*	
					798.2903		213.9748		*High*	
x_4			turn_duration		242.0740		50.3521		*Low*	
					571.1476		84.8794		*High*	

ALL_AGENTS										
Fuzzy rule set										
IF	x_1 is $Moderately\ Low$	AND	x_2 is $Moderately\ Low$	AND	x_3 is $High$	AND	x_4 is $High$	THEN	y is Low	
IF	x_1 is $Moderately\ Low$	AND	x_2 is $Moderately\ High$	AND	x_3 is Low	AND	x_4 is Low	THEN	y is $Medium$	
IF	x_1 is $Moderately\ High$	AND	x_2 is $Moderately\ Low$	AND	x_3 is Low	AND	x_4 is $Moderately\ High$	THEN	y is $High$	
IF	x_1 is $Moderately\ High$	AND	x_2 is $Moderately\ Low$	AND	x_3 is $High$	AND	x_4 is $High$	THEN	y is $Excellent$	
Gaussian fuzzy sets										
Attribute			**Name**		**Center**		**Standard Deviation**		**Linguistic label**	
x_1			management		257.4638		63.1698		*Moderately Low*	
					417.3272		91.7403		*Moderately High*	
x_2			break_sessions		261.3662		164.4175		*Moderately Low*	
					593.1362		164.4175		*Moderately High*	
x_3			icc_inbound_var		60345.5613		16629.5431		*Low*	
					142820.7940		34768.2436		*High*	
x_4			fraction_night		0.0019		0.0017		*Low*	
					0.0053		0.0017		*Moderately High*	
					0.0086		0.0017		*High*	

TABLE XII
FUZZY RULE-BASED CLASSIFICATION MODELS.

(Low and $Excellent$), that is, agents seem to show better performances when they are assigned to both night and day turns, instead of just one.

Guide for parameter settings. Due to the complexity of the proposed method, we finally show a guide for parameter setting in the complete process of feature selection plus fuzzy classification in order to maintain an adequate trade-off between accuracy and interpretability. The names shown below are the names that appear in the Weka user interface.

1) Feature selection phase:

a) Search strategy: MultiObjectiveEvolutionarySearch with the following parameters:

- *algorithm: ENORA*. Although *NSGA-II* can also be chosen, it has been empirically demonstrated that *ENORA* obtains better hypervolume values than *NSGA-II* for feature selection in classification tasks [35], [36].
- *generations:* 100 (as tested in Section IV Optimal number of generations).
- *populationSize:* 100. This population size is widely accepted by the scientific community on Evolutionary Computation.
- *reportFrecuency:* 100. This parameter establishes the frequency with which the information relative to the population in a generation is printed. It is useful to check the evolution of the algorithm in the testing phase. For the final execution it is convenient to set it equal to the number of generations, thus printing only two reports (at the beginning and at the end).
- *seed:* 1. This parameter is necessary for reproducibility.

b) Evaluator: WrapperSubsetEval with the following configuration:

- *classifier: MultiObjectiveEvolutionaryFuzzyClassifier* with the following parameters:
  - *algorithm: ENORA*. Although *NSGA-II* can also be chosen, it has been empirically demonstrated that *ENORA* obtains better hypervolume values than *NSGA-II* for classification tasks [25].
  - generations: 10 (as tested in Section IV Optimal number of generations).
  - *populationSize:* 100 (as accepted by the scientific community on Evolutionary Computation).
  - *reportFrecuency:* 10 (a report at the beginning and another report at the end).
  - *evaluationMeasure: ACC*. This parameter configures the function $\mathcal{F}_{\mathcal{D}}(\mathbf{\Gamma})$ of equation (3) which is used in the optimization process. *AUC* and *RMSE* can also be chosen.
  - *maxLabels:* $[3, 7]$. This parameter corresponds to L_{max} in the equation (3). It is assumed that more than 7 linguistic labels lead to a non-interpretable classifier. Set *maxLabels* = 3 for maximum compactness.
  - *maxRules:* $[w, \max\{w, 10\}]$. This parameter corresponds to M_{max} in the equation (3). It is assumed that more than 10 rules lead to a non-interpretable classifier. Set *maxRules* = w (number of classes) for maximum compactness. Set *maxRules* = $\max\{w, 10\}$ for maximum accuracy.
  - *maxSimilarity:* $[0.1, 0.4]$. his parameter corresponds to

g_s in the equation (3). It is assumed that a similarity between fuzzy sets greater than 0.4 leads to a non-interpretable classifier. Set *maxSimilarity* = 0.1 for a maximum separation between the fuzzy set. Set *maxSimilarity* = 0.4 for maximum accuracy.
  - *minV:* 30.0. This internal parameter sets the value for which the domain of a variable is divided to obtain the minimum variance.
  - *maxV:* 2.0. This internal parameter sets the value for which the domain of a variable is divided to obtain the maximum variance.
  - *seed:* 1 (necessary for reproducibility).
- *evaluationMeasure: ACC*. This parameter configures the function $\mathcal{F}_{\mathcal{D}}(\mathbf{x})$ of equation (2) which is used in the optimization process. *AUC*, *RMSE* (of the class probabilities for discrete class), *MAE* (mean absolute error of the class probabilities for discrete class), *F-measure*, and *AUPRC* (area under the precision-recall curve) can also be chosen.
- *folds:* 5 (for k-fold cross-validation). A higher number of folds can produce an excessive run time.
- *threshold:* 0.01. Cross-validation is repeated if standard deviation of mean exceeds this value.
- *seed:* 1 (necessary for reproducibility).

2) Fuzzy classification phase:

MultiObjectiveEvolutionaryFuzzyClassifier with the same values of the parameters as in the feature selection phase, with the same parameter values as in the feature selection phase, except the number of generations that must be set to a higher number for a fine tuning of the classifier. Depending on the size of the dataset, we suggest a number of generations between 1000 and 100000.

V. Conclusions and future works

In this work we proposed a novel multivariate feature selection method in which both the search strategy and the classifier are based on multi-objective evolutionary computation. We designed a set of experiments to establish an acceptable setting with respect to the number of evaluations required by the search strategy as well as by the classifier, and we tested our strategy on a real-life dataset. Our proposal is essentially novel: it solves the problem of selecting the *best* attributes for a very specific classification learning task based on fuzzy rules; as a matter of fact, the performances of a given classifier are very sensible to the attributes that are selected, and using filter selections, which are based on generic statistical values, or wrapper selections obtained with non-fuzzy classifier training not always gives good results. We were able to solve a classification problem in the context of a contact center that required to *internally* classify the quality of the services being provided, and our classification model turned out to be more accurate and more interpretable than a wide range of non-fuzzy classifier and, also, than other classical fuzzy classifiers.

After the FS phase, all selected attributes are (ideally) used in every rule of a classifier learned by our optimization model. By simply relaxing such a constraint, and by suitably redefining the complexity objective in the optimization model (e.g., by minimizing the sum of the lengths of all rules, or similar measures), the resulting classifiers will encompass rules different subsets of the selected attributes (clearly, the implementation must be adapted to obtain an initial population in which the classifiers have rules of different lengths as well as mutation operators that allow a rule to grow or shrink). It is natural to imagine that such classifiers may be even more accurate, and more interpretable at the same time, and such an improvement is currently considered as possible future work. In addition, we are currently working on the implementation of our own version of multi-objective differential evolution (MODE) for the selection of features and the classification based on rules, as well as their inclusion in the open source software Weka published under the GNU General public license.

Acknowledgements

This study was partially supported by computing facilities of Extremadura Research Centre for Advanced Technologies CETA-CIEMAT), funded by the European Regional Development Fund (ERDF). CETA-CIEMAT belongs to CIEMAT and the Government of Spain. This research was also partially supported by Spanish Ministry of Economy and Competitiveness (Spain) under project TIN2013-45491-R.

References

[1] L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," in *Proceedings of the 1991 IEEE International Symposium on Intelligent Control*, 1991, pp. 263–268.

[2] Z. Chi, H. Yan, and T. Pham, *Fuzzy Algorithms: With Applications to Image Processing and Pattern Recognition*. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 1996.

[3] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 4, pp. 506–515, 2001.

[4] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Hybridization of fuzzy gbml approaches for pattern classification problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 2, pp. 359–365, 2005.

[5] K. Deb, *Multi-objective optimization using evolutionary algorithms*. London, UK: Wiley, 2001.

[6] M. Gacto, R. Alcalá, and F. Herrera, "Adaptation and application of multi-objective evolutionary algorithms for rule reduction and parameter tuning of fuzzy rule-based systems," *Soft Computing*, vol. 13, no. 5, pp. 419–436, Dec. 2009.

[7] P. Ducange, B. Lazzerini, and F. Marcelloni, "Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets," *Soft Computing*, vol. 14, no. 7, pp. 713–728, 2010.

[8] P. Ducange, G. Mannara, F. Marcelloni, R. Pecori, and M. Vecchio, "A novel approach for internet traffic classification based on multi-objective evolutionary fuzzy classifiers," in *Fuzzy Systems (FUZZ-IEEE), 2017 IEEE International Conference on. IEEE*, 07 2017, pp. 1–6.

[9] M. Antonelli, D. Bernardo, H. Hagras, and F. Marcelloni, "Multiobjective evolutionary optimization of type-2 fuzzy rule-based systems for financial data classification," *IEEE Trans. Fuzzy Systems*, vol. 25, no. 2, pp. 249–264, 2017. [Online]. Available: https://doi.org/10.1109/TFUZZ.2016.2578341

[10] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*. Norwell, MA, USA: Kluwer Academic Publishers, 1998.

[11] E. Amaldi and V. Kann, "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems," *Theoretical Computer Science*, vol. 209, no. 1, pp. 237 – 260, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397597001151

[12] A. G. Karegowda, A. S. Manjunath, and M. A. Jayaram, "Comparative study of attribute selection using gain ratio and correlation based feature selection," *International Journal of Information Technology and Knowledge Management*, vol. 2, no. 2, pp. 271–277, 2010.

[13] M. A. Hall, "Correlation-based feature selection for machine learning," University of Waikato, Tech. Rep., 1999.

[14] A. Ahmad and L. Dey, "A feature selection technique for classificatory analysis," *Pattern Recognition Letters*, vol. 26, no. 1, pp. 43–56, 2005.

[15] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[16] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*. New York, NY, USA: Cambridge University Press, 2011.

[17] C. Wang, M. Shao, Q. He, Y. Qian, and Y. Qi, "Feature subset selection based on fuzzy neighborhood rough sets," *Knowledge-Based Systems*, vol. 111, pp. 173 – 179, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705116302714

[18] M. Cintra, H. Camargo, and M.-C. Monard, "Fuzzy feature subset selection using the wang & mendel method," in *Proceedings - 8th International Conference on Hybrid Intelligent Systems, HIS 2008*, 10 2008, pp. 590–595.

[19] S. Salzberg, "C4.5: Programs for machine learning by J. Ross Quinlan," *Machine Learning*, vol. 16, no. 3, pp. 235–240, 1994.

[20] Y. Jin, Ed., *Multi-Objective Machine Learning*, ser. Studies in Computational Intelligence. Warsaw, Poland: Springer, 2006, vol. 16.

[21] J. García-Nieto, E. Alba, L. Jourdan, and E. Talbi, "Sensitivity and specificity based multiobjective approach for feature selection: Application to cancer diagnosis," *Information Processing Letters*, vol. 109, no. 16, pp. 887 – 896, 2009.

[22] J. Zhao, V. B. Fernandes, L. Jiao, I. Yevseyeva, A. Maulana, R. Li, T. Bäck, and M. T. M. Emmerich, "Multiobjective optimization of classifiers by means of 3-d convex hull based evolutionary algorithm," *CoRR*, vol. abs/1412.5710, 2014.

[23] A. Gaspar-Cunha, G. Recio, L. Costa, and C. Estébanez, "Self-adaptive moea feature selection for classification of bankruptcy prediction data," *The Scientific World Journal*, vol. 2014, 2014.

[24] S. K. Nayak, P. K. Rout, A. K. Jagadev, and T. Swarnkar, "Elitism based multi-objective differential evolution for feature selection: A filter approach with an efficient redundancy measure," *Journal of King Saud University - Computer and Information Sciences*, 2017.

[25] F. Jiménez, G. Sánchez, and J. M. Juárez, "Multi-objective evolutionary algorithms for fuzzy classification in survival prediction," *Artificial Intelligence in Medicine*, vol. 60, no. 3, pp. 197–219, 2014.

[26] I. H. Witten, E. Frank, and M. A. Hall, "Introduction to weka," in *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*, ser. The Morgan Kaufmann Series in Data Management Systems. Boston: Morgan Kaufmann, 2011, pp. 403 – 406.

[27] T. Hubertus, M. Klaus, and T. Eberhard, *Optimization theory*. Dordrecht: Kluwer Academic, 2004.

[28] S. Sinha, *Mathematical Programming: Theory and Methods*. Elsevier Science Limited, 2006.

[29] Y. Collette and P. Siarry, *Multiobjective Optimization: Principles and Case Studies*. Springer Berlin Heidelberg, 2004.

[30] H. Karloff, *Linear Programming*. Boston, MA: Birkhauser Basel, 1991.

[31] I. Maros and G. Mitra, *Simplex algorithms*. Oxford Science, 1996, ch. 1, pp. 1–46.

[32] D. Bertsekas, *Nonlinear Programming (Second ed.)*. Cambridge, MA: Athena Scientific, 1999.

[33] F. Jiménez and J. L. Verdegay, *Evolutionary Computation and Mathematical Programming*. Heidelberg: Physica-Verlag HD, 2001, pp. 167–182.

[34] F. Jiménez, A. Gómez-Skarmeta, G. Sánchez, and K. Deb, "An evolutionary algorithm for constrained multi-objective optimization," in *Proceedings of the Evolutionary Computation on 2002. CEC '02. Proceedings of the 2002 Congress*, ser. CEC '02, vol. 2. Washington, DC, USA: IEEE Computer Society, 2002, pp. 1133–1138.

[35] F. Jiménez, E. Marzano, G. Sánchez, G. Sciavicco, and N. Vitacolonna, "Attribute selection via multi-objective evolutionary computation applied to multi-skill contact center data classification," in *Proc. of the IEEE Symposium on Computational Intelligence in Big Data (IEEE CIBD 15)*. IEEE, 2015, pp. 488–495.

[36] F. Jiménez, R. Jodár, G. Sánchez, M. Martín, and G. Sciavicco, "Multi-objective evolutionary computation based feature selection applied to behaviour assessment of children," in *Proc. of the 2016 International Conference on Educational Data Mining (ICEDM)*, vol. 2(6), 2016, pp. 1888–1897.

[37] F. Jiménez, G. Sánchez, J. García, G. Sciavicco, and L. Miralles, "Multi-objective evolutionary feature selection for online sales forecasting," *Neurocomputing*, vol. 234, pp. 75–92, 2017.

[38] K. Deb, A. Pratab, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii." *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182 – 197, 2002.

[39] F. Jiménez, G. Sánchez, J. García, G. Sciavicco, and L. Miralles, "Multi-objective evolutionary feature selection for online sales forecasting," *Neurocomputing*, no. 234, pp. 75 – 92, 2017.

[40] V. Kreinovich, C. Quintana, and L. Reznik, "Gaussian membership functions are most adequate in representing uncertainty in measurements," in *Proceedings of the NAFIPS'92*. Puerto Vallarta: North American Fuzzy Information Processing Society Conference, 1992, pp. 618–624.

[41] M. Paprzycki, A. Abraham, R. Guo, and S. Mukkamala, "Data mining approach for analyzing call center performance," in *Proc. of the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE)*, 2004, pp. 1092–1101.

[42] S. Salcedo-Sanz, M. Naldi, A. Pérez-Bellido, J. Portilla-Figueras, and E. Ortíz-García, "Evolutionary optimization of service times in interactive voice response systems," *IEEE Trans. Evolutionary Computation*, vol. 14, no. 4, pp. 602–617, 2010.

[43] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173 – 195, 2000.

[44] J. Novakovic, "Toward optimal feature selection using ranking methods and classification algorithms," *Yugoslav Journal of Operations Research*, vol. 21, no. 1, 2016.

[45] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley Publishing Company, 1984.

[46] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Prentice-Hall, 2003.

[47] M. Gutlein, E. Frank, M. Hall, and A. Karwath, "Large-scale attribute selection using wrappers," in *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining*, 2009, pp. 332–339.

[48] S. Dinakaran and P. R. J. Thangaiah, "Role of attribute selection in classification algorithms," *International Journal of Scientific & Engineering Research*, vol. 4, no. 6, pp. 67–71, 2013.

[49] A. Moraglio, C. Di Chio, J. Togelius, and R. Poli, "Geometric particle swarm optimization," in *Proceedings of the 10th European Conference on Genetic Programming*, 2007, pp. 125–136.

[50] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[51] W. Siedlecki and J. Sklansky, "A note on genetic algorithms for large-scale feature selection," *Pattern recognition letters*, vol. 10, no. 5, pp. 335–347, 1989.

[52] H. Vafaie and K. De Jong, "Genetic algorithms as a tool for feature selection in machine learning," in *Proceedings of the 4th International Conference on Tools with Artificial Intelligence (TAI)*, 1992, pp. 200–203.

[53] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, The University of Waikato, 1999.

[54] H. Liu and R. Setiono, "A probabilistic approach to feature selection - a filter solution," in *Proceedings of the 13th International Conference on Machine Learning (ICML)*, vol. 96, 1996, pp. 319–327.

[55] A. G. Karegowda, A. S. Manjunath, and M. A. Jayaram, "Comparative study of attribute selection using gain ratio and correlation based feature selection," *International Journal of Information Technology and Knowledge Management*, vol. 2, no. 2, pp. 271–277, 2010.

[56] A. Ahmad and L. Dey, "A feature selection technique for classificatory analysis," *Pattern Recognition Letters*, vol. 26, no. 1, pp. 43–56, 2005.

[57] S. I. Ali and W. Shahzad, "A feature subset selection method based on symmetric uncertainty and ant colony optimization," *International Journal of Computer Applications*, vol. 60, pp. 5–10, 2012.

[58] R. Schafer, "Accurate and efficient general-purpose boilerplate detection for crawled web corpora," *Language Resources and Evaluation*, vol. 51, no. 3, pp. 873–889, 2017.

[59] C.-C. Chang and C. J. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, 2011.

[60] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[61] C. E. Metz, "Basic principles of ROC analysis," in *Seminars in nuclear medicine*, vol. 8, 1978, pp. 283–298.

[62] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[63] L. Riza, C. Bergmeir, F. Herrera, and J. Bentez, "frbs: Fuzzy rule-based systems for classification and regression in R," *Journal of Statistical Software, Articles*, vol. 65, no. 6, pp. 1–30, 2015.

[64] J. Hühn and E. Hüllermeier, "Furia: an algorithm for unordered fuzzy rule induction," *Data Mining and Knowledge Discovery*, vol. 19, no. 3, pp. 293–319, 2009.