Attribute Selection via Multi-Objective Evolutionary Computation Applied to Multi-Skill Contact Center Data Classification

Fernando Jiménez
University of Murcia
Murcia - Spain
fernan@um.es

Enrico Marzano
Gap S.R.L.
Udine - Italy
e.marzano@gapitalia.it

Gracia Sánchez
University of Murcia
Murcia - Spain
gracia@um.es

Guido Sciavicco
University of Murcia
Murcia - Spain
guido@um.es

Nicola Vitacolonna
University of Udine
Udine - Italy
nicola.vitacolonna@uniud.it

***Abstract*—Attribute or feature selection is one of the basic strategies to improve the performances of data classification tasks, and, at the same time to reduce the complexity of classifiers, and it is a particularly fundamental one when the number of attributes is relatively high. Evolutionary computation has already proven itself to be a very effective choice to consistently reduce the number of attributes towards a better classification rate and a simpler semantic interpretation of the inferred classifiers. We propose the application of the multi-objective evolutionary algorithm ENORA to the task of feature selection for multi-class classification of data extracted from an integrated multi-channel multi-skill contact center, which include technical, service and central data for each session. Additionally, we propose a methodology to integrate feature selection for classification, model evaluation, and decision making to choose the most satisfactory model according to a *a posteriori* process in a multi-objective context. We check out our results by comparing the performance and the classification rate against the well-known multi-objective evolutionary algorithm NSGA-II. Finally, the best obtained solution is validated by a data expert's semantic interpretation of the classifier.**

I. Introduction

At its core, a *call center* is a set of resources, personnel, computers, and telecommunication equipment, which enable the delivery of services via the telephone. A current trend, made possible by the advancements in information technology, is the extension of call centers into *contact centers* (as integrated part of BPOs—Business Process Outsourcers), in which the phone-operator role of the agents is complemented by services offered via other media, ranging from faxing to instant messaging, to web portals, among others. Contact centers handle both *inbound* and *outbound* communications, with different purposes, including customer care and follow-up, as well as marketing and quality control. The distinguishing feature of *multi-skill* contact centers is that services vary over a wide range of possibly very different types (e.g., specific product client follow-up and travel reservation systems) [1].

The authors acknowledge the support from the Spanish Project *TIN12-39353-C04-01*, the Spanish Project *TIN2013-45491-R*, the Spanish fellowship program *'Ramon y Cajal' RYC-2011-07821* (G. Sciavicco), and the Italian regional grant *Interventi per la ricerca applicata, sviluppo sperimentale e innovazione 'Active Contact System', Friuli Venezia Giulia*, by the Regional Law 47/1978. We would also like to thank the 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.

As it operates, a large contact center generates vast amounts of data, which can be broadly classified as *operational* data, which includes all the technical information needed to reconstruct a detailed history of the events that take place during each communication, and *service* data, which is specific to the particular service for which the contact has taken place. The former category includes, for instance, the dialled or dialling phone number, the involved Contact Service Resource (also referred to as an *agent*), possible call transfers, and any kind of time-stamps. The latter category might include all the answers given by each interviewed subject during an outbound survey. The very large quantity of logged data produced over time becomes the ideal environment for *data mining* processes (DM). DM [2] consists of applying algorithms to collected data for distinct purposes such as finding patterns, creating prediction models or obtaining statistical data; besides, as the amount of processed data grows towards "big data" sizes, the performance of such algorithms becomes more and more critical. DM techniques are often offered by commercially available tools (see, e.g., [3]) at the user level, but data mining and, particularly, data classification are hardly user-level processes: they require not only a deep knowledge of the semantics of the analyzed data, but also a complex data preparation and integration phase, as well as a researcher-level expertise on the vast range of available tools and techniques, to choose the best one to be applied to each concrete case. We are interested in supervised data classification, that is, classification from a data set in which instances are labeled with a given class [2]. A typical approach to solve this problem consists in using a *decision tree* algorithm [4], applied to the original data set or to a data set reduced by a *feature* (or *attribute*) *selection* method [5]. Feature selection has become increasingly frequent in classification or regression applications in genomics, health sciences, economics, finance, among others (see, e.g., [6], [7]). Feature selection is an independent process whose main objective is to reduce the dimension of the data set (the "number of columns") in order to perform a more efficient and more easily interpretable classification, which is also more accurate because the noise introduced by irrelevant features has been removed. Supervised feature selection methods (see [8] for a survey) roughly range from *filter models* to *wrapper models*, to *embedded models*, according to whether selection and classification/regression are integrated step-by-step (as in the wrapper and the embedded methods) or not, and to whether statistical criteria are included in the selection (as

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.78

in the filter and the embedded methods); the complexity of their implementation as well as their performances increase from the simpler ones (filter methods) to the more elaborate ones (embedded), the latter being naturally less general and adaptable.

In this paper we apply a wrapper feature selection mechanism via the adaptation of a multi-objective evolutionary algorithm known as ENORA [9], [10], and we compare its performance against the classical multi-objective evolutionary algorithm NSGA-II [11] along two directives: *(i)* performance of the multi-objective search strategy via measuring the hypervolume, and *(ii)* quality of the classifiers that have been built over the selected features. The goal is to build a classifier for the session data of a multi-skill contact center, which allows us to predict the outcome of a communication based on a limited set of features. The data we have used have been provided by Northern Italy company Gap S.R.L., and consists of more than sixty-thousand *inbound phone-based* sessions over a period of three months, for which the values of forty-nine features, including the outcome (with eleven possible distinct values), has been recorded. Compared to previous data mining experiments on contact center databases, the quality of the information at our disposal is considerably higher. Not only did previous experiments such as [12] approach the classification problem via a hybrid model that uses several different DM techniques, without feature selection; they did also operate on a very restricted set of attributes, consequently limiting the range of their results. Other work concerning evolutionary approaches to contact center data sets includes [13], which, unlike the present paper, is focused on service tree optimization with the purpose of minimizing operation costs, which is part of the design and not the evaluation process in a contact center.

II. Background

Feature selection. *Feature selection* [5] is the process of eliminating features from the data set that are irrelevant with respect to the task to be performed. Its main aim is to determine a minimal subset of features from a problem domain while retaining a suitably high accuracy in representing the original features. Feature selection finds useful features to represent the data and remove non-relevant ones, and simplifies the implementation of the classifier itself by determining what features should be made available to it. Furthermore, feature selection tends to speed up the processing rate of the classifier; at the same time, it improves response times by reducing the dimensionality of the input space. Additionally, feature selection can improve the quality of the classification in terms of accuracy and interpretability of the outcome.

According to whether the training set is labelled (classified) or not, feature selection algorithms can be categorized into *supervised* and *unsupervised*, respectively. In between lies a third category, *semi-supervised* feature selection, which includes algorithms that make use of both labeled and unlabelled data to estimate the relevance of a feature. Supervised feature selection methods can be further categorized into *filter models*, *wrapper models* and *embedded models*. The filter model separates feature selection from classifier learning, so that the bias of a learning algorithm does not interact with the bias of a feature selection algorithm. Features may be ranked independently of the feature space (*univariate scheme*); alternatively, features may be evaluated in batch (*multivariate scheme*), being in this way naturally capable of handling redundancy. Filter techniques easily scale to very high-dimensional data sets, they are computationally simple and fast, and they are independent of the classification algorithm. Filter methods, however, ignore the interaction with the classifier. Wrapper models use the predictive accuracy of a predetermined learning algorithm to determine the quality of the selected features, with the advantages of including the interaction between feature subset search and model selection, and the ability to take into account feature dependencies. A common drawback of these techniques is that they have a higher risk of *overfitting* than filter techniques, and they are computationally expensive. Finally, embedded models achieve model fitting and feature selection simultaneously. They incorporate the statistical criteria, as filter models do, to select several candidate feature subsets with a given cardinality, and they choose the subset with the highest classification accuracy. While embedded methods include the interaction with the classification model, and are less computationally expensive than wrapper methods, they are also more complex to implement and test, and are less adaptable from case to case. According to the type of the output, feature selection algorithms can be classified into *feature weighting* algorithms and *subset selection* algorithms: feature selection algorithms in filter and embedded models may return either a subset of selected features or weights that measure the relevance of each feature. On the other hand, feature selection algorithms with wrapper model usually return feature subsets, and are therefore classified as subset selection algorithms.

Subset feature selection methods typically consist of four basic steps: *subset generation*, *subset evaluation*, *stopping criterion*, and *result validation*. In the first step, a candidate feature subset is chosen based on a given search strategy, and sent, in the second step, to be evaluated according to a certain evaluation criterion. The subset that best fits the evaluation criterion is chosen among all the candidates that have been evaluated after the stopping criterion are met. In the final step, the chosen subset is validated using domain knowledge or a validation set. *Sequential and exponential* search methods for subset generation include [5], [6], [14], [15]. Instead, we propose the use of a *random* search method, as in [16], [17], [18], and, in particular, of a multi-objective evolutionary algorithm.

Multi-objective evolutionary feature selection. In the evolutionary computational model, a problem plays the role of an environment populated by a set of individuals, each one representing a possible solution to the problem. The degree of adaptation of each individual to its environment is expressed by an adequacy measure known as a *fitness function*. Starting with an initial population of random solutions, in each iteration the best individuals are selected and combined using variation operators such as *crossing* and *mutation* to build the next generation. This process is repeated until some stop criterion is met, typically based on the number of iterations. The use of evolutionary strategies for the selection of features has been initially proposed in [19]. Since then, it has been regarded as a powerful tool for feature selection in machine learning [17] and proposed by numerous authors as a search strategy (see, e.g., [20], [21]). *Multi-objective* evolutionary algorithms are designed to solve a set of minimization/maximization problems for a tuple of n functions $f_1(\overrightarrow{x}), \ldots, f_n(\overrightarrow{x})$, where $\overrightarrow{x}$ is a

vector of parameters belonging to a given domain. Let $\mathcal{F}$ be the search space for a multi-objective optimization problem. A solution $\overrightarrow{x} \in \mathcal{F}$ is said to be a *non dominated* (or *Pareto optimal*) if and only if there exists no $\overrightarrow{y} \in \mathcal{F}$ for which: *(i)* there exists $1 \leq i \leq n$ such that $f_i(\overrightarrow{y})$ improves $f_i(\overrightarrow{x})$, and *(ii)* for each $j \neq i$, $f_j(\overrightarrow{x})$ does not improve $f_i(\overrightarrow{y})$.Tbe set of non dominated solutions from $\mathcal{F}$ is called *Pareto front*.

Multi-objective approaches are particularly suitable for multi-objective optimization, as they search for multiple optimal solutions in parallel; such algorithms are capable of finding a set of optimal solutions in its final population in a single run, and once the set of optimal solutions is available, the most satisfactory one can be chosen by applying a preference criterion. In subset feature selection, each solution in the Pareto front represents a subset of features with an associated trade-off between, for example, accuracy and data set dimension.

In the first evolutionary approach involving multi-objective feature selection [22], three criteria (accuracy, number of features and number of instances) are aggregated and then a single-objective optimization algorithm is applied. A formulation of feature selection as a multi-objective optimisation problem using a neuro-fuzzy based wrapper has been proposed in [23]. Other approaches, such as [24], [25], [26], [27], [28], propose the use of the multi-objective algorithm known as NSGA-II [11] in combination with wrapper methods that use decision tree (such as C4.5 [27]), support vector machines [25], [26], maximal entropy based models [24], or a filter method [28] that include measures of consistency, dependency and distance information.

The classifier J48. J48 is the Weka [29] implementation of the decision tree C4.5 introduced in [30] (as an improvement of algorithm ID3, by the same author). It is known to be computationally very efficient and to guarantee the interpretability of the results. Briefly, C4.5 builds decision trees from a set of training data by using the *information entropy gain* criterion. At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets, each one belonging to one of the predefined classes. The splitting criterion is the normalized information gain: the feature with the highest normalized information gain is chosen to make the decision.

Discussion. As we have seen, numerous approaches that use multi-objective evolutionary algorithms for feature selection have been proposed in last years. Although both filters as wrapper methods have been proposed, most of the authors use subset selection wrapper methods. The optimization model most commonly used has been maximizing the accuracy of the classifier along with minimizing the number of features, although many other models have been proposed for specific contexts. The NSGA-II algorithm has been without doubt the most widely used, as well as the search strategy used in the proposal or as reference algorithm in comparison with other search strategies. In this paper we focus on the problem of classification in supervised learning. In order to reach simultaneously very high accuracy and very low cardinality of the subset of the features, we are interested in a subset selection wrapper method with multi-objective evolutionary search strategy. A new multi-objective evolutionary algorithm (ENORA) is proposed in this paper for this purpose. Since the algorithm NSGA-II has been the reference in most of the studies in the literature, in this work NSGA-II is closely compared with ENORA, both from the point of view of the statistical results as well as the interpretation of the search process. Since wrapper methods base the subset evaluation on the predictive accuracy of a predetermined learning algorithm, the required computational time of their execution directly depends on the learning algorithm computational efficiency. For this reason we use the learning method C4.5 for classification. Moreover, we exploit the natural interpretability of the generated classifier by C4.5, which is a requirement for expert validation of the result. The purpose of this study is to classify the outcome of sessions from the data set of the multi-skill contact center Gap S.R.L.

Algorithm 1 $(\mu + \lambda)$ strategy for multi-objective optimization

Require: $T > 1$ {Number of iterations}
Require: $N > 1$ {Number of individuals in population}
Initialize P with N individuals
Evaluate all individuals of P
$t \leftarrow 0$
while $t < T$ **do**
  $Q \leftarrow \emptyset$
  $i \leftarrow 0$
  while $i < N$ **do**
    $Parent1 \leftarrow$ Binary tournament selection from P
    $Parent2 \leftarrow$ Binary tournament selection from P
    $Child1, Child2 \leftarrow$ Self-adaptative variation $Parent1, Parent2$
    Evaluate $Child1$
    Evaluate $Child2$
    $Q \leftarrow Q \bigcup \{Child1, Child2\}$
    $i \leftarrow i + 2$
  end while
  $R \leftarrow P \bigcup Q$
  $P \leftarrow N$ Best individuals from R according to the Rank-crowding better function in population R
  $t \leftarrow t + 1$
end while
return Non dominated individuals from P

III. ENORA: an Elitist Pareto-Based Multi-Objective Evolutionary Algorithm

We propose the use of an evolutionary search strategy for multiple Pareto-optimal solutions (subsets) simultaneously, taking into account both *accuracy* (classification rate) and *cardinality* (of the set of features) criteria. Its main components, that is, representation, fitness functions, initial population, selection and sampling mechanisms, generational replacement schemata, and variation operators are described in this section.

The ENORA algorithm. ENORA is an elitist Pareto-based multi-objective evolutionary algorithm that uses a $(\mu + \lambda)$ survival, where μ corresponds to the population size and λ refers to the number of generated children. The $(\mu + \lambda)$ strategy was originally developed in [31] as an *evolution strategy*, using selection, adapting mutation and a population of size one, called $(1 + 1)$-ES. Recombination and populations with more than one individual were later introduced in [32]. The $(\mu + \lambda)$ technique allows the μ best children and parents to survive and is, therefore, an elitist method. ENORA uses a $(\mu + \lambda)$ survival with $\mu = \lambda$, where μ and λ are equal to the population size, binary tournament selection, and self-adaptive crossover and mutation for multi-objective evolutionary optimization (Alg. 1). After the initialization and evaluation of a population P of N individuals, and for each of the T generations, a pair of parents are selected by a *binary tournament selection* from the

Algorithm 2 Binary tournament selection

Require: P {Population}
1: $I \leftarrow$ Random selection from P
2: $J \leftarrow$ Random selection from P
3: **if** I is better than J according to the rank crowding better function in population P **then**
4: **return** I
5: **else**
6: **return** J
7: **end if**

Algorithm 3 Rank-Crowding-Better function

Require: P {Population}
Require: I, J {Individuals to compare}
1: **if** $rank\,(P, I) < rank\,(P, J)$ **then**
2: **return** $True$
3: **end if**
4: **if** $rank\,(P, J) < rank\,(P, I)$ **then**
5: **return** $False$
6: **end if**
7: **return** $crowding\_distance\left(P^I, I\right) > crowding\_distance\left(P^J, J\right)$

population P (Alg. 2). This algorithm returns the best between two random individuals according to the *rank crowding better function* (Alg. 3). An individual I is considered better than an individual J if the rank of individual I is better (lower) than the rank of individual J in the population P. The *rank* of an individual I in a population P, $rank\,(P, I)$, is the *non-domination level* of the individual I among the individuals J of the population P so that $slot\,(I)$=$slot\,(J)$, where the radial *slot* ($slot\,(I)$) is the portion of the search space to which I belongs, and it is defined as:

$$slot\,(I) = \sum_{j=1}^{n-1} d^{j-1} \lfloor d \frac{\alpha_j^I}{\pi/2} \rfloor$$

$$\alpha_j^I = \begin{cases} \frac{\pi}{2} & \text{if } h_j^I = 0 \\ \arctan(\frac{h_{j+1}^I}{h_j^I}) & \text{if } h_j^I \neq 0 \end{cases}$$

where $d = \left\lfloor \sqrt[n-1]{N} \right\rfloor$ and h_j^I is the objective function f_j^I normalized in $[0, 1]$. If two individuals I and J have the same rank, the best individual is the individual with the greater crowding distance in its front (the front of I and J are denoted by P^I and P^J, respectively, Alg. 3). The selected pair of parents is crossed, mutated, evaluated and added to an initially empty auxiliary population Q. This process is repeated until Q contains a number N of individuals. An auxiliary population R is obtained with the union of the populations P and Q. Then, the rank of all individuals in the population R is calculated (Alg. 3). Finally, the N best individuals of R according to the rank crowding better function (Alg. 3) survive to the next generation.

The crowding distance of an individual I in a population P is a measure of the search space around I which is not occupied by any other individual in the population P. This quantity serves as an estimate of the perimeter of the cuboid formed by using the nearest neighbours as the vertices. If we define $f_j^{max} = \max_{I \in P} \{f_j^I\}$, $f_j^{min} = \min_{I \in P} \{f_j^I\}$, and $f_j^{sup_j^I}$ (resp., $f_j^{inf_j^I}$) is the value of the jth objective function for the individual higher adjacent (resp., lower adjacent) in the jth objective function to the individual I, then the crowding distance $crowding\_distance(P, I)$ is ∞ if for each j it is the case that $f_j^I = f_j^{max}$ or $f_j^I = f_j^{min}$, and it is

$$crowding\_distance\,(P, I) = \sum_{j=1}^{n} \frac{f_j^{sup_j^I} - f_j^{inf_j^I}}{f_j^{max} - f_j^{min}}$$

otherwise.

Representation, evaluation and variation. We use a fixed-length representation where each individual consists of a bit set, and each bit represents a selected (1) or non selected (0) feature. Additionally, to carry out self-adaptive crossing and mutation, each individual has two discrete parameters $d_I \in \{0, \ldots, \delta\}$ and $e_I \in \{0, \ldots, \epsilon\}$ associated to, respectively, crossing and mutation, where $\delta \geq 0$ is the number of crossing operators and $\varepsilon \geq 0$ is the number of mutation operators. Therefore, an individual I in the feature selection problem with M features is represented as:

$$I = \left\{b_1^I, \ldots, b_M^I, d_I, e_I\right\}$$

where for each i $b_i^i \in \{0, 1\}$, and where $d_I \in \{0, \ldots, \delta\}$, $e_I \in \{0, \ldots, \epsilon\}$. An individual I is evaluated with two fitness functions, $f_1\,(I)$ and $f_2\,(I)$, corresponding to the two objectives of the multi-objective optimization model:

$$\begin{cases} f_1\,(I) = \mathcal{ACC}\,(I) \\ f_2\,(I) = \mathcal{C}\,(I) \end{cases}$$

where $\mathcal{ACC}\,(I)$, defined as:

$$\mathcal{ACC}\,(I) = \frac{N_c}{N_t}$$

(being N_c and N_t are the number of correctly classified instances and the number of total instances, respectively) is the accuracy of the classifier, when classification is performed using only the attributes in individual I; $f_1\,(I)$ must be maximized. On the other hand, $\mathcal{C}\,(I)$ is the cardinality of the subset of the selected features represented by individual I, so that $f_2\,(I)$ must be minimized.

The initial population is generated randomly. For each individual I in the population, q randomly chosen bits are set to 1, and the remaining $M - q$ to 0 (in this way, we ensure the diversity of the initial population); moreover, the parameters d_I and e_I are also randomly generated in their respective domains $\{0, \delta\}$ and $\{0, \epsilon\}$. Self-adaptive crossover and mutation are used to maintain diversity in the population and to sustain the convergence capacity of the evolutionary algorithm. In a self-adaptive evolutionary algorithm [33], the probabilities of crossover and mutation vary according to the fitness value of the solutions. By using self-adaptive variation operators, it is not necessary to set the probabilities of the application of the different operators *a priori*. We use *uniform crossover* and *one flip mutation*, although other variation operators may be considered. The selection of the operators is made by means of the adaptive technique according to the parameters d_I and e_I that indicate which crossover and mutation is carried out for individual I.

IV. Experiment Design and Results

In this section we present the results of the experiment over our data set, obtained by a methodology which includes pre-

Algorithm 4 Variation

Require: $Parent1$, $Parent2$ {Individuals to vary}
1: $Child1 \leftarrow Parent1$
2: $Child2 \leftarrow Parent2$
3: Self-adaptive crossover $Child1$, $Child2$
4: Self-adaptive mutation $Child1$
5: Self-adaptive mutation $Child2$
6: **return** $Child1, Child2$

Algorithm 5 Adaptive crossover

Require: I, J {Individuals to cross}
Require: p_v $(0 < p_v < 1)$ {Probability of operator change}
Require: $\delta > 0$ {Number of different crossover operators ($\delta = 1$ in our case)}
1: **if** A random Bernoulli variable of probability p_v takes the value 1 **then**
2: $d_I \leftarrow$ Int *Random* from $\{0, \delta\}$
3: **end if**
4: $d_J \leftarrow d_I$
5: Carry out the type of crossover specified by d_I:
{0: No cross}
{1: Uniform crossover}

Algorithm 6 Adaptive mutation

Require: I {Individual to mutate}
Require: p_v $(0 < p_v < 1)$ {Probability of operator change}
Require: $\epsilon > 0$ {Number of different mutation operators ($\epsilon = 1$ in our case)}
1: **if** A random Bernoulli variable of probability p_v takes the value 1 **then**
2: $e_I \leftarrow$ Int *Random* from $\{0, \epsilon\}$
3: **end if**
4: Carry out the type of mutation specified by e_I:
{0: No mutation}
{1: One flip mutation}

processing of the data, feature selection, optimizers' performances comparison (based on hypervolume metrics), classifier learning construction, and test, as shown in Fig. 1.

The Gap data set. Gap S.R.L. [1] is a Business Process Outsourcer operating in various fields, such as tele-marketing and customer care, and offers, among the rest, inbound and outbound phone-based services. Via a complex integration of several data systems, Gap keeps a rich and detailed record of each agent-user communication (i.e., of each *session*), including both operational and service data. For inbound phone-based communications, operational data are automatically generated; they include time stamps that record the initial and final (absolute) time of each session, the waiting interval, the duration of the communication, the total duration of the session (which can be different from the duration of the communication because it extends across the in-queue period and the post-call activity carried out by an agent), the time slot (hour of the day, day of the week, month of the year) in which the session has taken place, the technical result (closed, renounced, transferred, off time, waiting time exceeded), the type (mobile vs home phone), and location of the caller. Service data are related to the particular service for which the call has taken place, and include cumulative workload/service up to the current time, service category, and service priority. The feature over which we classify session data is composed by eleven classes, explained in Tab. I).

Besides operational and service data, Gap maintains a database of *central* data, which includes various parameters regarding agents and services, such as the time schedule of the personnel, each agent's expertise level in each service, some personal data of the agents (such as sex and age). Overall, the raw data for our experiments includes 49 features. Some continuous features (e.g., call duration) have been discretized

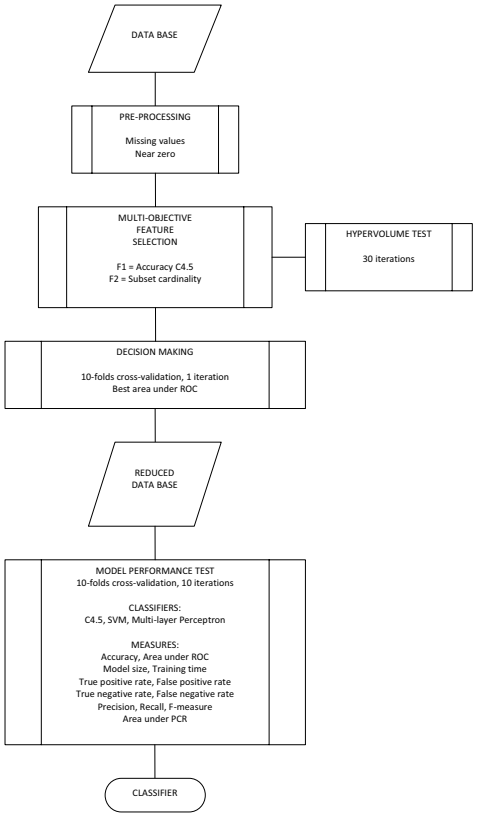

Fig. 1. Proposed methodology

Result	Description
CALL_DROPPED	The call dropped before reaching its natural ending
CLOSED	The call has ended properly, either positively or negatively
SMS_EMAIL	The call has resulted in a SMS or email dispach
NEGATIVE	The call has ended properly with a negative result for the caller
NOT_TRANSFERRED	The call has been transferred without success
INVALID	The call has no valid result
POSITIVE	The call has ended properly with a positive result for the caller
RECALL	The call has ended properly, and a recall has been scheduled
SIGNAL	The call has ended properly, and an external warning has been issued
TEST	The call is an internal test
TRANSFERRED	The call has been successfully transferred

TABLE I. AN EXPLANATION OF THE POSSIBLE SESSION RESULTS.

(e.g., short, medium, long).

Data pre-processing. The initial data set composed of 49 features has been pre-processed as follows. First, we have replaced all the missing values for nominal and numeric attributes with, respectively, the modes and means from the training data; to this end, the procedure *ReplaceMissingValues* from the *weka.filters.unsupervised.attribute* package has been used. Second, we have eliminated the features with too small variation; we have used the procedure *NearZeroVar* from Caret R [34] for this task. Among the removed features, we mention the *type of service*, as over 90% of the sessions were tagged as customer care, the *caller area region*, as over 60% of the sessions did not have a value set (they were inbound calls from mobile devices), and six aggregated attributes about agents such as the average conversation time per day and the average break time per day: this is probably due to the fact that similar aggregate data have been collected for each month, and the

	ENORA	NSGA-II
Minimum	0.0105	0.0153
Maximum	0.0153	1.0000
Mean	0.0129	0.7047
S.D.	0.0011	0.4589
C.I. Low	0.0124	0.5333
C.I. High	0.0133	0.8760

S.D = Standard Deviation of Mean
C.I. = Confidence Interval for the Mean (90%)

TABLE II. STATISTICS FOR THE HYPERVOLUME OBTAINED WITH 30 RUNS: ENORA AGAINST NSGA-II.

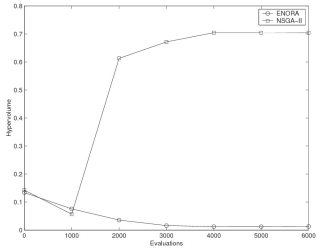

Fig. 2. Hypervolume evolution: ENORA against NSGA-II.

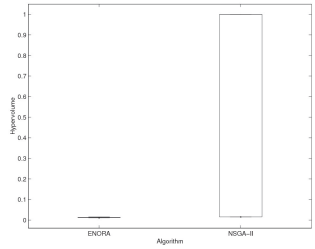

Fig. 3. Hypervolume boxplots: ENORA against NSGA-II.

distribution over a single day is not informative. As a result, the data set has been reduced to 41 features.

Feature selection. Both search strategies ENORA and NSGA-II have been integrated into a C4.5-based wrapper feature selection method, using the two objective functions described in the previous section: accuracy maximization and cardinality minimization. After 30 run, to each non-dominated individual of the last population of each strategy, we performed a 10-folds cross-validation to each non-dominated solution of the last population by using the $\mathcal{AUC}$ parameter (*area under ROC curve*), and identified the solution with the best cross-validation value (one for ENORA, and one for NSGA-II). Finally, for each of the two solutions, we built a reduced data set based on the the selected features. The two search strategies have been implemented with dynamically adapted parameters [33], and written in Java by using the Weka [29] package. For each run, we used the following evaluator: *weka.attributeSelection.WrapperSubsetEval -B weka.classifiers.trees.J48 -F 5 -T 0.01 -R 1 -E acc – -C 0.25 -M 2*. Both search strategies ENORA and NSGA-II have been run with the number of evaluations set to 100 and the number of individuals in each population set to 100, for a total of 10000 evaluations. The search strategy ENORA has been incorporated by authors into Weka as official package by means of the *MultiObjetiveEvolutionarySearch* package [35].

Hypervolume test. The measure of hypervolume is defined, in general terms [36], as the volume of the search space dominated by a population P, expressed as:

$$HV\left(P\right) = \bigcup_{i=1}^{|Q|} v_i$$

where $Q \subseteq P$ is the set of non-dominated individuals of P, and v_i is the volume of the individual i. We use, for technical convenience, an equivalent hypervolume indicator defined as the ratio of the volume of the non-dominated search space over the volume of the entire search space, as follows:

$$HV'\left(P\right) = 1 - \frac{HV\left(P\right)}{volS}$$

where $volS$ is the volume of the search space, and we compare the indicator of ENORA with the one of NSGA-II.

NSGA-II [11] is an elitist Pareto-based multi-objective evolutionary algorithm which improves the previous NSGA algorithm by incorporating an explicit diversity technique, and it is, perhaps, one of the most used Pareto-based multi-objective evolutionary algorithms described in the literature. It uses, as ENORA, a $(\mu + \lambda)$ strategy with a binary tournament selection and a rank crowding better function. The difference between NSGA-II and ENORA is how the calculation of the rank of the individuals in the population is performed. In ENORA, the rank of an individual in a population is the non-domination level of the individual in its slot, whereas in NSGA-II the rank of an individual in a population is the non-domination level of the individual in all the population. The corresponding graphical evolution as well as their respective boxplots are shown in Fig. 2 and Fig. 3. To perform a statistical comparison of the results obtained by the two optimizers, a confidence interval of 90% has been used for the mean obtained with a pairwise t-test [37], as there is no dependency among the populations involved.

Classification model performance test. Once the two reduced data sets (ENORA-DS and NSGA-II-DS) have been obtained, we tested and compared them. It turns out that both reduced data sets contain 8 of the original 41 features, shown and explained in Tab. IV. We configured the *Experimenter* tool available in Weka with the two data sets to perform a 10-fold cross-validation (10 iterations) with the following classifiers: J48, MLPClassifier (which trains a *multi-layer perceptron* with one hidden layer using Weka's Optimization class, by minimizing the squared error plus a quadratic penalty with the so-called *BFGS method*), and SMO (which implements Platt's sequential minimal optimization algorithm for training a support vector classifier [38]), all run with default parameters set by Weka. To analyze the result of the experiment we performed a paired t-test *corrected*, with 0.05 significance (being ENORA-DS the test base) and the following measures have been compared: *(i)* the percent of correct classifications; *(ii)* the (weighted) area under the ROC curve; *(iii)* the average model size; *(iv)* the CPU time required for training; *(v)* the (weighted) average IR precision; *(vi)* the (weighted) average IR

Attribute	ENORA-DS	NSGA-II-DS	Validation
CONVERSATION_PERIOD	yes	yes	The conversation time might influence the outcome of a session: short times might indicate connection problems, and long times might indicate difficulties for the agent to satisfy customer's needs
POST_CALL_PERIOD	yes	yes	The post call period characterizes the session, the type of service, and the agent: long periods might indicate a problematic managing of the session after the end of the conversation
OP_AVAIL_MONTH	yes	yes	The available time over the month characterizes the agent: high values might indicate that that specific agent works inbound sessions only seldom and he/she is focused on a restricted set of services
SESS_SERV_MONTH	yes	no	Some services are generally more common than others, and cumulate more service time: such services might generally indicate different outcomes
CALL_SERV_MONTH	no	yes	Some services are generally more common than others, and cumulate more calls over time: such services might generally indicate different outcomes
PHONE_TYPE	yes	yes	The inbound call being made from a mobile or a land line might influence the outcome: for example, mobile might generically entail a higher probability of connection problems
SERVICE_PRIORITY	yes	yes	The priority associated to a particular type of service might influence the outcome of a session for that service: higher priorities are served in a better and quicker way
PRACTICE_AIM	yes	yes	The aim of the session (the reason why the call has been placed) might influence its outcome: certain aims might be easier or harder to be managed than others

TABLE III. EXPERT'S VALIDATION OF THE SELECTED ATTRIBUTES.

	ENORA-DS	NSGA-II-DS	ORIGINAL-DS
	percent correct		
trees.J48	**95.4536(0.2457)**	95.4534(0.2460)	95.0932(0.2793)*
functions.SMO	93.4509(0.2334)	93.4509(0.2321)	**94.8253(0.2632)**^{v}
functions.MLPClassifier	87.7157(1.2001)	87.8714(1.5174)	**88.6387(2.1868)**
	Weighted avg. area under ROC		
trees.J48	**0.7922(0.0032)**	**0.7922(0.0032)**	0.7385(0.0023) *
functions.SMO	0.8068(0.0031)	0.8069(0.0031)	**0.8520(0.0038)** ^{v}
functions.MLPClassifier	0.7713(0.0467)	0.7686(0.0502)	**0.9139(0.0459)** ^{v}
	Model size		
trees.J48	**135379.7200(9654.7690)**	135505.7200(9818.4742)	468671.6600(25013.6411) *
functions.SMO	36734.4400(53.4696)	**36731.0000(55.7755)**	96148.9200(151.1192) *
functions.MLPClassifier	**14789.0000(0.0000)**	14790.0000(0.0000)*	33761.0000(0.0000) *
	CPU time for training		
trees.J48	0.9473(0.0718)	**0.9039(0.0840)**	3.9074(0.1210)*
functions.SMO	**69.3231(4.8032)**	69.3481(4.6594)	297.7571(20.0965)*
functions.MLPClassifier	**0.4715(0.2256)**	0.4996(0.3073)	4.0965(4.0989)*
	Weighted avg. IR precision		
trees.J48	**0.9497(0.0030)**	0.9496(0.0030)	0.9477(0.0032)
functions.SMO	0.9134(0.0025)	0.9134(0.0024)	**0.9357(0.0048)**^{v}
functions.MLPClassifier	0.7743(0.0320)	0.7778(0.0387)	**0.7974(0.0554)**
	Weighted avg. IR recall		
trees.J48	**0.9545(0.0025)**	**0.9545(0.0025)**	0.9509(0.0028)*
functions.SMO	0.9345(0.0023)	0.9345(0.0023)	**0.9483(0.0026)**^{v}
functions.MLPClassifier	0.8772(0.0120)	0.8787(0.0152)	**0.8864(0.0219)**
	Weighted avg. F-measure		
trees.J48	**0.9497(0.0029)**	**0.9497(0.0029)**	0.9486(0.0030)
functions.SMO	0.9191(0.0030)	**0.9191(0.0029)**	0.9389(0.0032) ^{v}
functions.MLPClassifier	0.8220(0.0224)	0.8244(0.0271)	**0.8384(0.0390)**
	Weighted avg. are under PRC		
trees.J48	**0.7454(0.0037)**	**0.7454(0.0037)**	0.6972(0.0038) *
functions.SMO	0.7162(0.0037)	0.7162(0.0036)	**0.7585(0.0048)** ^{v}
functions.MLPClassifier	0.6733(0.0586)	0.6718(0.0670)	**0.8035(0.0516)** ^{v}
	Weighted avg. true positive rate		
trees.J48	**0.9545(0.0025)**	**0.9545(0.0025)**	0.9509(0.0028) *
functions.SMO	0.9345(0.0023)	0.9345(0.0023)	**0.9483(0.0026)** ^{v}
functions.MLPClassifier	0.8772(0.0120)	0.8787(0.0152)	**0.8864(0.0219)**
	Weighted avg. false positive rate		
trees.J48	0.0537(0.0046)	0.0537(0.0047)	**0.0490(0.0041)** ^{v}
functions.SMO	0.0940(0.0074)	0.0940(0.0074)	**0.0608(0.0075)** ^{v}
functions.MLPClassifier	0.2243(0.0554)	0.2224(0.0559)	**0.1989(0.0777)**
	Weighted avg. true negative rate		
trees.J48	0.9463(0.0046)	0.9463(0.0047)	**0.9510(0.0041)** ^{v}
functions.SMO	0.9060(0.0074)	0.9060(0.0074)	**0.9392(0.0075)** ^{v}
functions.MLPClassifier	0.7757(0.0554)	0.7776(0.0559)	**0.8011(0.0777)**
	Weighted avg. false negative rate		
trees.J48	**0.0455(0.0025)**	**0.0455(0.0025)**	0.0491(0.0028) *
functions.SMO	0.0655(0.0023)	0.0655(0.0023)	**0.0517(0.0026)** ^{v}
functions.MLPClassifier	0.1228(0.0120)	0.1213(0.0152)	**0.1136(0.0219)**

TABLE IV. COMPARATIVE RESULTS OF THE TEST.

recall; *(vii)* the (weighted) average F-measure (that combines precision and recall); *(viii)* the (weighted) area under the PRC (precision-recall) curve; *(ix)* the (weighted) ratio of true positive, true negative, false positive, and false negative.

Analysis of the solutions and discussion. In this section we analyze the obtained solution based on the result of the above test, shown in Tab. IV, where by ORIGINAL-DS we denote the original data set (after the pre-processing). For each result, a mark * denotes that the result is statistically worse than the test base (ENORA-DS); similarly, a mark ^{v} denotes a statistically better result, and no mark denotes no statistically meaningful difference. The values between brackets are the standard deviations, and the boldfaced results are the best ones.

- In terms of the hypervolume indicator, as a result of the comparisons, we can conclude that:
  1) ENORA provides lower values of the hypervolume indicator than NSGA-II; therefore, the approximations obtained with ENORA are better, according to the hypervolume indicator. Observe that our t-test must be considered robust, since our samples contain more than 30 individuals [37], leading us to conclude that the differences between the hypervolume values obtained with the two algorithms are statistically significant.
  2) NSGA-II is unable to keep the hypervolume of the initial population, which, instead, worsen from generation to generation; on the other hand, ENORA is capable to keep the hypervolume and improve it.
  3) Although NSGA-II and ENORA algorithms are similar, their behaviors are really quite different, the main difference being the following: when NSGA-II compares two individuals through binary tournament, a dominated individual is never selected, while in ENORA a dominated individual can be winner of the tournament; then, ENORA encourages diversity, allowing individuals in all slots to perform towards the Pareto front although these individuals are not the best ones when they are compared, and, thus, obtaining best hypervolume that NSGA-II along the successive generations.
- In terms of the performances of the classification model, we observe that:
  1) Both feature selection methods improve the performances with respect to all tested classifiers (and, particularly, with respect to J48), and effectively reduce the number of features.
  2) Although the performances of the models built on the two different data sets is similar, and does not present significant statistical values, ENORA has obtained a better $\mathcal{ACC}$ and the same $\mathcal{AUC}$, reducing the size of the model compared to NSGA-II.
  3) Evaluation with J48 turned out to be better than the evaluation with the other two classifiers, which was predictable, given that the feature selection mechanisms have been configured with J48.

V. CONCLUSIONS

We proposed the application of the multi-objective evolutionary algorithm ENORA to the task of feature selection for multi-class classification of data extracted from an integrated contact center. We also proposed a methodology to integrate

feature selection for classification, model evaluation, and decision making to choose the most satisfactory model according to a *a posteriori* process in a multi-objective context. We concluded that ENORA is a serious alternative for feature selection with aim of simultaneously maximization of the accuracy and minimization of subset cardinality of the models. Based on our results we claim that ENORA is statistically better with respect to the hypervolume indicator, obtaining classification models that are not worse than those obtained by NSGA-II. Finally, the chosen solution by our methodology presents, with respect to the original data set, a significant reduction of the number of features together with an improvement of important measures such as the accuracy, the area under the ROC curve, the size of the model, the CPU time requested for training, the precision, the recall, the F-measure, and the area under the PRC curve. The selected attributes have been validated by a GAP manager who is an expert in this domain.

REFERENCES

[1] A. Brunello, "A data warehouse for a contact center with multiple channels and skills," Master's thesis, University of Udine, 2015.

[2] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "Advances in knowledge discovery and data mining," U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. AAAI, 1996, ch. From Data Mining to Knowledge Discovery: An Overview, pp. 1–34.

[3] "Data mining in call centers," https://msdn.microsoft.com/en-us/library/dd206982.aspx, 2015.

[4] S. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica*, vol. 31, no. 3, pp. 249–268, 2007.

[5] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer, 1998.

[6] R. Caruana and D. Freitag, "Greedy attribute selection," in *Proc. of the 11th International Conference on Machine Learning (ICML)*. Morgan Kaufmann, 1994, pp. 28–36.

[7] A. Arauzo-Azofra, J. Benitez, and J. Castro, "Consistency measures for feature selection," *Journal of Intelligence Information Systems*, vol. 30, no. 3, pp. 273–292, 2008.

[8] V. Kumar and S. Minz, "Feature selection: A literature review," *Smart Computing Review*, vol. 4, no. 3, pp. 211–229, 2014.

[9] F. Jiménez, A. Gómez-Skarmeta, G. Sánchez, and K. Deb, "An evolutionary algorithm for constrained multi-objective optimization," in *Proc. of the Congress on Evolutionary Computation (CEC)*, vol. 2. IEEE, 2002, pp. 1133–1138.

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

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

[12] 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.

[13] 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.

[14] A. Marcano-Cedeno, J. Quintanilla-Dominguez, M. Cortina-Januchs, and D. Andina, "Feature selection using sequential forward selection and classification applying artificial metaplasticity neural network," in *Proc. of the 46th Annual Conference on IEEE Industrial Electronics Society (IECON)*, 2010, pp. 2845–2850.

[15] P. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," *IEEE Trans. on Computers*, vol. 26, no. 9, pp. 917–922, 1977.

[16] G. Nandi, "An enhanced approach to las vegas filter (lvf) feature selection algorithm," in *Proc. of the 2nd National Conference on Emerging Trends and Applications in Computer Science (NCETACS)*, 2011, pp. 1–3.

[17] H. Vafaie and K. D. Jong, "Genetic algorithms as a tool for feature selection in machine learning," in *Proc. of the 4th International Conference on Tools with Artificial Intelligence (TAI)*, 1992, pp. 200–204.

[18] S. Dreyer, "Evolutionary feature selection," Master's thesis, Institutt for datateknikk og informasjonsvitenskap, 2013.

[19] 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.

[20] M. ElAlami, "A filter model for feature subset selection based on genetic algorithm," *Knowledge-Based Systems*, vol. 22, no. 5, pp. 356 – 362, 2009.

[21] R. Anirudha, R. Kannan, and N. Patil, "Genetic algorithm based wrapper feature selection on hybrid prediction model for analysis of high dimensional data," in *Proc. of the 9th International Conference on Industrial and Information Systems (ICIIS)*, 2014, pp. 1–6.

[22] H. Ishibuchi, "Multi-objective pattern and feature selection by a genetic algorithm," in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO)*, 2000, pp. 1069–1076.

[23] C. Emmanouilidis, A. Hunter, J. MacIntyre, and C. Cox, "A multi-objective genetic algorithm approach to feature selection in neural and fuzzy modeling," *Journal of Evolutionary Optimization*, vol. 3, no. 1, pp. 1–26, 2001.

[24] A. Ekbal, S. Saha, and C. Garbe, "Feature selection using multiobjective optimization for named entity recognition," in *Proc. of the 20th International Conference on Pattern Recognition (ICPR)*, 2010, pp. 1937–1940.

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

[26] 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.

[27] A. Jara, R. Martínez, D. Vigueras, G. Sánchez, and F. Jiménez, "Attribute selection by multiobjective evolutionary computation applied to mortality from infection in severe burns patients," in *Proc. of the International Conference on Health Informatics (HEALTHINF)*, 2011, pp. 467–471.

[28] M. Venkatadri and K. S. Rao, "A multiobjective genetic algorithm for feature selection in data mining," *International Journal of Computer Science and Information Technologies*, vol. 1, no. 5, pp. 443–448, 2010.

[29] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, 2005.

[30] J. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[31] I. Rechenberg, *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution*. Frommann, 1973.

[32] H. Schwefel, *Numerical Optimization of Computer Models*. Wiley, 1981.

[33] M. Srinivas and L. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.

[34] "Package caret," http://cran.r-project.org/web/packages/caret/caret.pdf, 2015.

[35] "Multiobjective evolutionary search." [Online]. Available: http://sourceforge.net/projects/moea/files/MultiObjetiveEvolutionarySearch1.0.0.zip/download?use_mirror=vorboss\&r=\&use_mirror=vorboss

[36] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, 2001.

[37] M. O'Mahony, *Sensory Evaluation of Food: Statistical Methods and Procedures*. CRC Press, 1986.

[38] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," Microsof Research, Tech. Rep., 1998, mSR-TR-98-14.