Predicting the Risk of Academic Dropout with Temporal Multi-Objective Optimization

F. Jiménez, A. Paoletti, G. Sánchez, and G. Sciavicco

***Abstract*—In the European academic systems, the public founding to single universities depends on many factors, which are periodically evaluated. One of such factors is the rate of success, that is, the rate of students that do complete their course of study. At many levels, therefore, there is an increasing interest in being able to predict the risk that a student will abandon the studies, so that (specific, personal) corrective actions may be designed. In this paper, we propose an innovative temporal optimization model that is able to identify the earliest moment in a student's career in which a reliable prediction can be made concerning his/her risk of dropping out from the course of studies. Unlike most available models, our solution can be based on the academic behaviour alone, and our evidence suggests that by ignoring classically used attributes such as the gender or the results of pre-academic studies one obtains more accurate, and less biased, models. We tested our system on real data from the three-years Degree in Computer Science offered by the University of Ferrara (Italy).**

***Index Terms*—Multi-objective optimization; Evolutionary computation; Academic dropout risk prediction.**

I. Introduction

Educational Data Mining refers to the set of techniques specialized in analyzing and extracting useful knowledge from students' data. In recent years the importance of such a discipline has clearly grown, witnessed, among others, by the extensive literature. In many academic systems in Europe, public founding is the single most important contribution (source: European University Association - EUA, 2018) to universities' economical sustainability. In particular, in the Italian academic system, the public founding schema has changed rapidly in the past 20 years, and a very complex mechanism, linked to several different parameters, is used for modulating the amount of money that a particular institution receives every year. One of such parameters is the *rate of success*, that is, the ratio between the number of students that decide to start a particular course of studies and the number of students that actually complete it. Educational data mining techniques can be used to predict such rate, provided that one finds the correct model in terms of accuracy, easiness of implementation, and availability of data. For the purposes of this paper, we shall define *academic dropout* the event of not finishing a course of studies, and officially abandoning it. A student may be automatically classified as a dropped out if he/she has not paid the due fees for an entire year, or more, or because he/she has communicated to the institution the intention of abandoning that particular course of studies and/or the institution per se.

F. Jiménez (*fernan@um.es*) and G. Sánchez (*gracia@um.es*) are with the Dept. of Information Eng. and Comms., University of Murcia, Campus de Espinardo, 30100, Murcia, Spain.

A. Paoletti (*alessia.paoletti@studenti.units.it*) is with the Dept. of Mathematics and Geosciences, University of Trieste, Piazzale Europa 1, 34127, Trieste, Italy.

G. Sciavicco (*guido.sciavicco@unife.it*) is with the Dept. of Mathematics and Computer Science, University of Ferrara, Via Saragat 1, 44124, Ferrara, Italy.

Background. There are three prevailing strategies for *academic prediction*: predicting the chance of completing the course of studies, analyzing the risk of dropout, and predicting the final grade. Moreover, academic prediction can be also orthogonally classified by the type of predictors, which can be *static* students' data (age, gender, school of origin, and so on), *phycological and attitudinal* (usually from the result of questionnaires' administration), and *dynamic* (that is, concerning the behaviour of the student in the first part of his/her career). Clearly different types of data can be mixed in some models, but they refer, nonetheless, to different aspects: static data cannot be controlled at all, and a prediction model based on static data alone is basically a descriptive statistical model; attitudinal data require the design, administration, and interpretation of specific tests, and the availability of such data cannot be always guaranteed; instead, dynamic data are always available, and by modelling the essential aspects of a student's career they allow a very reliable prediction. Examples of recent work that use static data include, among many others, [1], [2], [3], that use personal and social behaviour data (obtained by questionnaires' administration) to predict the performance during the first semester, [4], in which personal information, family information, and economical level information have been used to predict the academic performance, [5], that includes specific psychological factors (obtained, as before, by questionnaires' administration) to predict the academic behaviour of students, and [6], which models the prediction problem with a neural network. Examples of predictive models based on attitudinal or psychological tests include [7], in which a model combining standardized test scores and psychosocial variables is proposed, and [8], in which the authors propose an investigation of a visualization psychometric test to facilitate the early diagnosis of the academic performances of technical drawing students. Recent studies concerning the predictive capabilities of dynamic variables include [9], in which elements such as attendance to classes, obtained results, preparatory school information, among others, have been taken into account to predict the performance of a student, [10], in which all static aspects have been ignored, and the performances of specific indicators have been condensed and used to predict the global performance during a semester, [11], in which the authors considered data based on student activity provided by private university, and searched for early predictors of

students' success and failure, in reference, in particular, to the amount and the type of students' use of the institution's online resources, and [12], in which indicators of the initial performances are used to predict the behaviour of students. Finally, recent work that use, to some extent, combinations of static, attitudinal, and dynamic predictors include [13], focused on e-learning students, and [14], which is an open source initiative consisting of building a prediction model on data taken from one institution and testing its performance on data taken from other partner institutions, associating the model to a concrete and effective intervention strategy. Predicting the risk of dropout has been the focus of [15], in which the authors, similarly to what we study in the present work, approach the problem of an early prediction of dropout in high school, which is an essential condition for a successful intervention of problematic students; unlike [15], we focus our attention on university students and define the problem as an optimization model. Moreover, a dropout prediction model has also been developed in [16], in which the authors develop a (temporal) Cox model to study the nature of students' retention by focusing on dropout as predicted class, and by using several static attributes plus five, cumulative, dynamic attributes; unlike [16], our methodology focuses on obtaining an optimal moment of prediction using mainly dynamic, cumulative and non-cumulative attributes.

A temporal optimization model. Unlike most previous researches on academic performance (and in particular, on academic dropout) prediction, we propose in this paper a multi-objective optimization model; we focus on dynamic data, although our model can be extended to include all classical predictors, as discussed earlier. The ultimate purpose of the proposed optimization model is to find the *earliest moment*, in a student's career, in which a reliable prediction is possible. This allow us to build an ah-hoc prediction model for any university course, without committing, for example, to perform the prediction based on the results during the first semester or the first year. We define the concept of *horizon*, which can be seen, roughly, as the *moment*, in a student's career, *in which the prediction is made*, and we aim to optimize, at the same time, the period of time that we have to wait before the prediction (to be minimized) and the accuracy of our prediction (to be maximized). In this way, we can perform a *dynamic preprocessing* to a training set of student's careers (of which we know the outcome) to find the best horizon for that course of studies, and, then, build a prediction model using any learner. The resulting classifier can be then applied to a new student of the same course (after he/she has passed the horizon), to predict his/her outcome. We tested the potential of our approach on real data taken from the University of Ferrara (Italy), and, specifically, from the Bachelor Degree in Computer Science. The objectives of this paper are:

- Proposing a temporal multi-objective optimization model for dynamic preprocessing of student's data, so that the earliest horizon can be found at which a reliable prediction on academic dropout can be made;
- Proposing an implementation of our model;
- Testing the potential of our model on real data.

Structure of the paper. This paper is structured as follows. First, in Section II, we describe the underlying idea and define a multi-objective optimization model. In Section III we describe our solution to the optimization problem, and all its components. Then, we describe our data set, and our experiment, in Section IV, along with the results and their validation. We conclude the paper in Section V by explaining how our system can be seamlessly implemented in an academic context and which corrective actions can be initiated for students that are predicted to drop out.

II. A Multi-Objective Temporal Optimization Model for Prediction of Academic Dropout

The problem. Most European academic systems have a fixed structure. Consider the case, for example, of the Italian system. Once a student is enrolled at a specific year in an undergraduate course, such as the Bachelor Degree in Computer Science, which is the object of experimental part of this paper, automatically gains the right of being examined for all subjects of that year, and he/she retains such right for as long as he/she is enrolled. So, for example, given that *Introduction to Programming* is a subject of the first semester, an enrolled student has the right of being examined at every official call for as long as his/her enrolling continues; he/she may be examined for that subject, during, say, an official call of the fourth semester, provided that he/she has paid all due fees for the second year of course. Moreover, a given degree has a fixed duration (typically, a bachelor degree course is three academic years), but a student that has not fulfilled all required subjects may continue to enroll for a fourth, fifth, and, in general, n-th academic year (such a student will be called *out-of-course*). Finally, especially for bachelor degrees, universities tend to impose a policy of a fixed number of exam calls per academic year, which is, in average, six. So, continuing with our example, *Introduction to Programming*, taught during the first semester, allows for two attempts to pass the relative exam right at the end of the semester, two at the end of the second semester (in which it is not taught), and two right before the next academic year begins. A realistic model of this system, however, should take into account only the *exam period*, instead of the number of calls; in the Italian system, therefore, we consider that there are three exam periods (February-March, June-July, and September). Given this relatively fixed structure, and given that we may easily identify a subset of subjects in a specific degree that are well-known to be *determinative* (i.e., mandatory, on-topic, and known to be relatively difficult), in this paper we propose a model that is able to mix both static and dynamic parameters, the latter focusing on which determining subjects have been already passed, the passing grade, and *how many attempts were necessary* to learn a predictive model. In our approach, therefore, a student can be described by a set of classical attributes (such as gender, or type of high school diploma), and a set of attributes that depend on the performances up until the observation moment; as we shall see, our purpose is to determine the optimal observation moment to perform a reliable prediction. The predicted class is binary: *dropout/no*

dropout. For the purposes of this paper, we say that a student has dropped out from the course of studies if one of two conditions arises: either he/she has not paid any fee for at least one academic year (one should take into account that any delay in paying any due fee prevents the student to register any exam until the debt is cancelled), or the student has official communicated to the University that he/she wants to change the course of studies or simply leave it. In both cases, the class is determined by the system automatically.

We designed the learning process as a *multi-objective optimization* model with two objectives: the accuracy of classification (to be maximized) and the *weight of the horizon*. An horizon can be seen as a curve that represents, for each determining subject, for how many attempts one should wait before deciding that not having passed that subject has a relevant role in predicting whether the student will drop out, and of course, its weight, which represents how long should we be waiting before predicting, is to be minimized. To better understand this concept, consider two limit examples. In the first situation, we build a model with complete horizon: in this model we have waited until the end of the third academic year before deciding if a particular student will drop out; this is of course extremely precise, as well as useless (because we cannot help that student any more). In the second example, we build a model with almost zero horizon: in this model, we decide whether or not a particular student is a risk of dropping out based almost solely on static information; this would be desirable, but unavoidably imprecise (because there is not enough information to decide, yet). Thus, accuracy and weight of the horizon must be optimized together, forming a typical Pareto problem which makes sense, *mutatis mutandis*, in other academic systems with a fixed or quasi-fixed structure. To solve this optimization problem we use an evolutionary algorithm to build a *wrapper* in which individuals, that is, horizons, are generated and evaluated. We call this approach *dynamic preprocessing*, as it consists of generating, dynamically, a version of the data set that it is then used to learn a classification model.

Multi-objective optimization. *Optimization* problems [17], [18] are naturally classified into problems of continuous or problems of discrete variables, the latter often referred to as *combinatorial* problems. In combinatorial problems, we are looking for objects from a finite, or possible countably infinite, set $\mathcal{X}$ - typically integers, sets, permutations, or graphs, and, in *multi-objective* problems [19], we aim to solve a system of the type:

$$\begin{array}{ll} min/max & f_1(\overline{x}) \\ min/max & f_2(\overline{x}) \\ \ldots & \\ min/max & f_k(\overline{x}), \end{array} \tag{1}$$

where each $f_l(\overline{x})$ is an arbitrary linear or non-linear function, $\overline{x} = (x_1, x_2, \ldots, x_m) \in \mathcal{X}^m$ represents the set of decision variables. Solving a multi-objective optimization problem naturally leads to a Pareto set of *solutions* (also called *individuals* or *particles*). A solution set $\overline{x} \in \mathcal{X}^m$ is *Pareto* (or *non-dominated*) if and only if there is not other solution $\overline{x}' \in \mathcal{X}^m$ that dominates $\overline{x}$. Solution $\overline{x}'$ *dominates* $\overline{x}$ if and only if: *(i)* there exists $1 \leq l \leq k$ such that $f_l(\overline{x}')$ improves $f_l(\overline{x})$, and *(ii)* for every $1 \leq l \leq k$, $f_l(\overline{x})$ does not improve $f_l(\overline{x}')$. In other words, $\overline{x}'$ dominates $\overline{x}$ if and only if $\overline{x}'$ is better than $\overline{x}$ in at least one objective, and not worse than $\overline{x}$ in any objective. So, solving an multi-objective optimization problem generates a set of solutions instead of a single solution; each individual solution of the front is optimal in its way, because it is not dominated by any other possible solution (see Fig. II). When necessary, we choose a single solution out of the optimal front by means of a *a posteriori* decision making process. When each function f_l is linear, (1) is a *linear programming problem* [20], for which extremely efficient algorithms exist (e.g., the *simplex method* - see [21], and derived methods). When any of the functions f_i is non-linear, then we have a *non-linear programming problem* [22]. 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; however, generic search algorithms are not guaranteed to find an optimal solution. Methods such as *branch and bound* [23], and *meta-heuristics*, such as *evolutionary algorithms* [24] or *particle swarm optimization* [25], are typically used to find approximate solutions for many complicated optimization problems, including multi-objective combinatorial optimization problems.

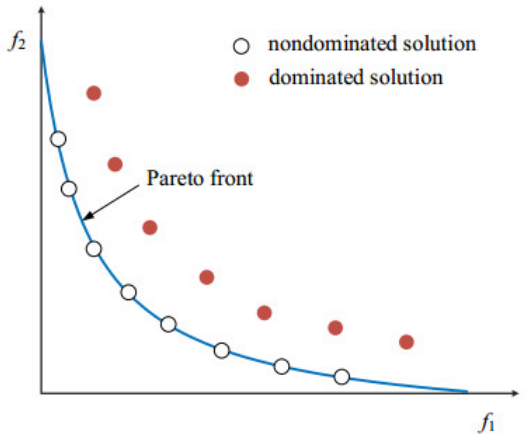

Fig. 1. An example of Pareto front.

Temporal optimization of the horizon. Consider, now, a generic data set $\mathcal{S}$ (*students*) whose instances are tuples as:

$$s = (S_1, \ldots, S_n, D_1, \ldots, D_m, C).$$

In the most general case, we describe a student with n *static* attributes $S_1, \ldots, S_n$, m *dynamic* attributes $D_1, \ldots, D_m$, and a class C. Static attributes include typical information such as *gender*, *age*, or *previous academic results*. Dynamic attributes, on the other hand, describe the behaviour of s in m chosen subjects. Each D_j may be thought as a pair (a_j, g_j), where a_j represents the number of attempts that were necessary to pass subject j (and it is 0 when subject j has not been

	year	*sex*	S1	S2	S3	S4	S5	S6	Class
Student 1	2004	M	5,2	4,3	7,1	3,2	6,4	7,3	1
Student 2	2007	F	5,2	8,3	2,2	9,2	0,0	0,0	0
Student 1	2004	M	2	3	0	2	0	0	1
Student 2	2007	F	2	0	2	0	0	0	0

TABLE I
INSTANCES CORRESPONDING TO TWO EXAMPLE STUDENTS (TOP SIDE), AND THE SAME INSTANCES UNDER THE HORIZON $(5, 5, 5, 5, 5, 5)$ (BOTTOM SIDE). IN THIS EXAMPLE, $r = 0$, THAT IS, THERE ARE NO CUMULATIVE INDICATORS.

passed yet), while g_j is any categorization of the grade that s obtained for subject j (that include a special value, for the case in which $a_j = 0$). Learning a classifier over $\mathcal{S}$ as we have just described it corresponds to learning a classifier with complete knowledge. Indeed, to construct $\mathcal{S}$ we may consider a statistically relevant number of students enrolled into a specific degree and who have had enough time (at least as long as the programmed duration) to complete their studies. Having chosen a set of m relevant subjects that are taught, say, in the first two years of a three-years degree, a classifier trained over such data will be extremely precise but it will present two drawbacks: it will suffer of a high level of over-fitting, and it will be inapplicable in real cases, because we would know that some student is at high risk of dropping out when it is already too late to do anything about it. To solve this problem, we designed a transformation $\mathcal{H}$ that works as follows. Given an element $\overline{x} = (x_1, \ldots, x_m) \in \mathbb{N}^m$ (an *horizon*), and a student s described as above:

$$s = (S_1, \ldots, S_n, D_1, \ldots, D_m, C),$$

we say that:

$$\mathcal{H}(s, \overline{x}) = (S_1, \ldots, S_n, T_1, \ldots, T_r, D'_1, \ldots, D'_m, C),$$

where, for each j, we have:

$$D'_j = \begin{cases} g_j & \text{if } a_j < x_j \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

So, x_j represents the number of attempts that we are prepared to wait for the subject j, before predicting the outcome for a given student. The new attributes $T_1, \ldots, T_k$ represent *cumulative* indicators, such as *number of off-course academic years* in which he/she has been enrolled, or *number of exams passed with honours*. These, if available, must be added during this transformation, and taken from some other source (paired with $\mathcal{S}$), because their value is dynamic and depends on the particular horizon. Indeed, the horizon gives us, implicitly, an observation time for the student (say, for example, after the third available exam period after his/her enrollment), and, at that time, it makes sense to highlight his/her cumulative indicators. This transformation can be applied to the entire data set; by overloading the symbol $\mathcal{H}$, we denote by:

$$\mathcal{H}(\mathcal{S}, \overline{x}) = \{\mathcal{H}(s, \overline{x}) \mid s \in \mathcal{S}\},$$

the transformation applied to the entire data set $\mathcal{S}$. Then, we can define the *weight* of an horizon $\overline{x}$:

$$W(\overline{x}) = \sum_{j=1}^{m} x_j. \quad (3)$$

The smaller is the weight of $\overline{x}$, the less information is carried by $\mathcal{H}(\mathcal{S}, \overline{x})$, and the less precise will be a classifier learned on it; but, at the same time, the less time we have to wait before being able to predict the outcome for a student. Let MAX be maximum number of exam periods that any student has needed to pass an exam as recorded in $\mathcal{S}$: a classifier learned on the data set $\mathcal{H}(\mathcal{S}, (0, \ldots, 0))$ will be called *zero-knowledge* classifier (and it corresponds to learning a classifier with static data only), while a classifier learned on $\mathcal{H}(\mathcal{S}, (MAX, \ldots, MAX))$ will be called *full-knowledge* classifier. To better understand this concept, consider the case of the Italian university system, which has a somehow fixed structure, and consider two example students *Student 1* and *Student 2* that present the situation reported in Table I (top side). In this example, $n = 2$, $m = 6$, and grades are categorized into $0, \ldots, 4$, with 0 corresponding to no grade at all, 1 to the best possible grade, and 4 to the worse possible grade. *Student 1* is a male, first enrolled in 2004 who, as of today, has passed precisely all chosen subjects (and his/her class is 1); in particular, we know that it took him 5 exam sessions to pass subject 1. Under the horizon $(5, 5, 5, 5, 5, 5)$, we have the situation described in Table I (bottom side): we decided to wait at most 5 attempts per subject, and the table shows us a partial information regarding *Student 1*. A full knowledge classifier learned on the original data set would have complete knowledge of how a student behaved regarding the most important and characterizing subjects; such model would be useless because it would require to wait 'unlimited' time to decide the predicted outcome, and when it came out negative (even if that prediction would be extremely reliable) it would be too late to anything about it. Optimizing the horizon solves this problem.

Thus, we aim to solve the following multi-objective combinatorial optimization problem:

$$\begin{array}{ll} max & ACC(\overline{x}) \\ min & W(\overline{x}), \end{array} \quad (4)$$

which is an instance of (1) with $l = 2$. With ACC we denote the accuracy of a classifier learned over the data set $\mathcal{H}(\mathcal{S}, \overline{x})$. Since $\mathcal{H}(\mathcal{S}, \overline{x})$ is a data set of students whose information is restricted to the temporal horizon $\overline{x}$, we say that (4) is a *temporal* multi-objective combinatorial optimization problem. It may be argued that using the accuracy of a classifier is reductive, as the performances of classifiers can be measured

in many different ways. But previous experiments of feature selection based on similar mechanisms to the one presented here (although with very different purposes) seem to indicate that optimizing the accuracy of a classifier is not essentially different from optimizing other measures (see, e.g., [26]).

III. An Evolutionary Algorithm Solution

Genetic algorithms are search heuristics inspired by natural evolution. They reflect the process of natural selection, where the fittest individuals are selected for reproduction in order to produce offspring of the next generation. The process starts with the selection of fittest individuals from a population. They produce offspring which inherit the characteristics of the parents and will be added to the next generation. If parents have better fitness, their offspring will be better than parents and have a better chance at surviving. This process keeps on iterating and at the end, a generation with the fittest individuals will be found. All genetic algorithms share this methodology, but they differ in the way the initial populations are generated, the best individuals are selected, and the new individuals are generated. In the recent literature, genetic algorithms have been successfully used to solve a wide range of search and optimization problems, and, in particular, intractable data analysis and data mining problems, such as, for example, *feature selection* [27], [28], [29] for classification and for clustering. The algorithm NSGA-II (*Non-dominated Sorting Genetic Algorithm II*), introduced in [30], is a well-known *elitist Pareto-based genetic algorithm* which can be used to solve problems as (1). The population is sorted using a domination rank function, where each individual is assigned a rank representing the front to which it belongs. Individuals with same rank are sorted with the *crowding distance*, calculated as the perimeter of the hypercube formed by taking the nearest individuals to it as the hypercube's vertices. NSGA-II uses a $(\mu + \lambda)$ strategy with $\mu = \lambda$, where μ corresponds to the population size and λ refers to the number of children, with *binary tournament selection*. For us, an individual is an horizon, and at each generation we evaluate the current population with two objective functions as described in (4). NSGA-II is a well-established algorithms whose performances have been tested in many occasions and in very different applicative domains. There are more recent genetic algorithms in the literature, such as, for example, ENORA [31], [32], [33], among many others; nonetheless, our purpose in this paper is describing and testing a new methodology based on a multi-objective optimization problem, rather than finding how it can be best implemented.

Individual representation is simple: each individual I is a tuple in $[0, \ldots, MAX]^m$. The initial population is randomly generated with a uniform distribution, as shown in Algorithm 1. Each individual I is evaluated with two fitness functions, W and ACC, corresponding to the two objectives of the multi-objective combinatorial optimization model (3). The function W (weight of the horizon) is calculated as in (2). Algorithm 2 shows the steps required to calculate the value of the function ACC. As much as *variation operators* are concerned, we use *self-adaptive* [34] *uniform crossover*, *arithmetical crossover*, and *uniform mutation* [35], described in Algorithm 3, 4, and 5, respectively. These have been suitably adapted to integer representation for the purposes of this work.

Algorithm 1 Initial population

Require: $popsize$ {Population size}
Require: m {Number of subjects}
1: **for** $i = 1$ to $popsize$ **do**
2: let I be a new individual;
3: **for** $j = 1$ to m **do**
4: $I[j] \leftarrow rand(\{1, \ldots, MAX\})$;
5: **end for**
6: $POP[i] \leftarrow I$;
7: **end for**
8: **return** POP

Algorithm 2 ACC function

Require: I {Individual to evaluate}
Require: $\mathcal{S}$ {Data Set}
Require: Γ {Classifier}
1: Compute $\mathcal{S}'$ from $\mathcal{H}(\mathcal{S}, I)$;
2: Build a classifier Γ trained with $\mathcal{S}'$;
3: Evaluate Γ in full training mode;
4: **return** Accuracy(Γ)

As classifier learning algorithms, we propose two solutions, the first one using *random forest* [36], and the second one using *logistic regression* [37]. The former is a well-known tree-based learning algorithm known for its low tendency to over-training and its high accuracy. It is an ensemble learning method which constructs a forest of random decision trees, for classification or regression purposes. A typical problem of decision trees is their propensity to over-fit, if not properly pruned: in the literature, they are regarded as models having low bias, but high variance. In random forest 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 over-fit. Given a new instance to classify, the final output is obtained by combining the results given by the different trained models. Logistic regression, on the contrary, is a function-based learner, and it is named after the function used at the core of the method, the *logit* function. Such function, also called *sigmoid*, was developed to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits, and it is particularly fit for binary classification problems.

Finally, we propose two alternative *decision making* strategies. Once a final Pareto front $\mathcal{P} = \{\overline{x}_1, \ldots, \overline{x}_p\}$ has been identified, a solution may be selected according to Algorithm 6. This simple methodology has the advantage of allowing an automatic, independent choice that takes into account both objectives. Consider, for example, two horizons $\overline{x}_1$ and $\overline{x}_2$, the former with weight 15 and accuracy 0.951, and the latter with weight 16 and accuracy 0.955: the former, with a

Algorithm 3 Uniform crossover

Require: I, J {Parents}
1: **for** $j = 1$ to m **do**
2: $r \leftarrow rand(\{0, 1\})$;
3: **if** $r = 0$ **then**
4: $I'[j] = J[j]$;
5: $J'[j] = I[j]$;
6: **else**
7: $I'[j] = I[j]$;
8: $J'[j] = J[j]$;
9: **end if**
10: **end for**
11: **return** I', J' {Children}

Algorithm 4 Arithmetical crossover

Require: I, J {Parents}
1: $\alpha \leftarrow real\_rand([0, 1])$;
2: **for** $j = 1$ to m **do**
3: $I'[j] = round(\alpha \cdot I[j] + (1 - \alpha) \cdot J[j])$;
4: $J'[j] = round(\alpha \cdot J[j] + (1 - \alpha) \cdot I[i])$;
5: **end for**
6: **return** I', J' {Children}

ratio accuracy/weight of 0.06, must be preferred over the latter, with a ratio 0.05, regardless the (small) gaining in accuracy of the latter over the former, and assuming that its accuracy is at least better than the average of the front. Since the search space may be not extremely wide, final Pareto fronts with just a few distinct non-dominated solutions may appear; in the particular case of a front with just two solutions, the one with the best ratio may be simply chosen. In some cases, however, other strategies can be used if special conditions apply. Indeed, each individual can be associated to a *temporal indicator*; this can be thought of as a quantitative description of how long we should wait before that particular individual (that corresponds to an horizon) can be applied to a student, and it can be measured, for example, in number of months or number of available exam sessions. Then, if the resulting Pareto front is composed by a sufficiently small number of individuals, and if a minimum accepted accuracy and a maximum accepted temporal indicator of the model can be identified, we can also use Algorithm 7. In this case, we: *(i)* associate to each individual of the front to its temporal indicator; *(ii)* for each different temporal indicator, consider the individual with the highest accuracy (if there is more than one), and *(iii)* choose the individual with highest accuracy among the remaining ones that pass the minimum accuracy and maximum temporal indicator thresholds.

IV. Experiment

In this section, we test the potential of our proposal with real data.

Data description and preparation. The data warehouse of the University of Ferrara contains the entire history of each student ever enrolled. We considered the bachelor degree in Computer Science, which has a programmed duration of three

Algorithm 5 Uniform mutation

Require: I {Parent}
1: **for** $j = 1$ to m **do**
2: $r \leftarrow rand(\{0, 1\})$;
3: **if** $r = 0$ **then**
4: $I'[j] \leftarrow rand(\{1, \ldots, MAX\})$;
5: **else**
6: $I'[j] = I[j]$;
7: **end if**
8: **end for**
9: **return** I' {Child}

Algorithm 6 Decision making (solution 1)

Require: $\mathcal{P} = \{\overline{x}_1, \ldots, \overline{x}_p\}$ {Pareto front}
1: $avg \leftarrow 0$;
2: **for all** $x_i \in \mathcal{P}$ **do**
3: $avg \leftarrow avg + ACC(\overline{x}_i)$;
4: **end for**
5: $avg \leftarrow \frac{avg}{p}$;
6: $Best \leftarrow 0$;
7: **for all** $x_i \in \mathcal{P}$ **do**
8: **if** $\frac{ACC(\overline{x}_i)}{W(\overline{x}_i)} > Best$ **and** $\mathcal{ACC}(\overline{x}_i) \geq avg$ **then**
9: $\overline{x} \leftarrow \overline{x}_i$;
10: $Best \leftarrow \frac{ACC(\overline{x}_i)}{W(\overline{x}_i)}$;
11: **end if**
12: **end for**
13: **return** $\overline{x}$

academic years, and which has been opened in 2001. Since then, some changes occurred in the denomination of subjects, the material that is presented, and, of course, the teachers that are responsible for each subject. Nevertheless, the essential structure of the course has been preserved. For each student, several static data are available. In particular, we know, for each student, the type of high school diploma, the gender, and the province of residence. Other attributes may be available for a single student; for example, we may have included the student's year of first enrollment, or the GPA from hir/her preparatory school, among others. Unfortunately, as it often happens, some features must be simply be excluded from classifier learning (such as the year of first enrollment: a model that uses such an information is unable to predict the outcome for students that will enroll in future years) or may not be available/standardized for all students (as it is the case of GPAs: they may be not comparable among different schools, or unavailable for certain students). Static data are shown in Tab. II. The cumulative data that we have considered for each student are those listed in Table III. The attribute *#out of course* takes into account the number of times that he/she enrolled besides the programmed duration of the course. These include the number of times that he/she has enrolled after the programmed duration of course, plus the number of times that he/she has not been able to pass the predetermined minimum limit of credits for the current year of enrollment (in this case, he/she has to enroll in a out of course academic year even before the programmed duration has not been reached yet).

Algorithm 7 Decision making (solution 2)

Require: $\mathcal{P} = \{\overline{x}_1, \ldots, \overline{x}_p\}$ {Pareto front}
Require: $\mathcal{T} = \{t_1, \ldots, t_p\}$ {Temporal indicators}
Require: l {Minimal accepted accuracy}
Require: t {Maximal accepted temporal indicator}
$\mathcal{D} \leftarrow \mathcal{P}$;
for all $x_i \in \mathcal{P}$ **do**
  for all $x_j \in \mathcal{P}$ **do**
    if ($t_i = t_j$ **and** $ACC(\overline{x}_i) < ACC(\overline{x}_j)$) **then**
      $\mathcal{D} \leftarrow \mathcal{D} - \{x_i\}$;
    end if
  end for
end for
$\mathcal{F} \leftarrow \mathcal{D}$;
for all $x_i \in \mathcal{D}$ **do**
  if ($ACC(\overline{x}_i) < l$ **or** $t_i > t$) **then**
    $\mathcal{F} \leftarrow \mathcal{F} - \{x_i\}$;
  end if
end for
$Best \leftarrow 0$;
for all $x_i \in \mathcal{F}$ **do**
  if $ACC(\overline{x}_i) > Best$ **then**
    $\overline{x} \leftarrow \overline{x}_i$;
    $Best \leftarrow ACC(\overline{x}_i)$;
  end if
end for
return $\overline{x}$

attribute	explanation
gender	male or female student
residence	whether lives in Ferrara or not
prep school	type of preparatory school

TABLE II
STATIC DATA

For example, a student s enrolls in the first year, does not pass enough credits during the academic year, and enrolls, again, in the first year (out of course); then, he/she fulfills all requirements, and may enroll in the second year. In such a situation, the attribute *#out of course* is set to 1. Furthermore, in the Italian systems, the fees due for an academic year are usually split into several tranches, each with a specific deadline; each time that a student s fails to fulfill a deadline, a tax flag is raised and accounted for in the attribute *#tax flags*. Finally, there are several possible financial benefits that may be granted to a student, depending on his/her academic performance, his/her economical status, among others; an economical benefit may have the form of a fee reduction or exemption, or a scholarship, and the number of times that any such benefit is ever granted to a student is registered in the attribute *#benefits*. Therefore, for us, $r = 3$. Dynamic data take into account the entire academic performance of a student with respect to a specific, predefined, subset of subjects, as we have explained in the previous section. We have chosen $m = 11$ core subjects that are being taught at the Computer Science bachelor degree following very intuitive guidelines; for a subject to be chosen, we imposed that: *(i)* it is a mandatory subject, and it has always been so; *(ii)* it is a characterizing subject, and *(iii)* the content of the subject has not changed (besides the necessary updating), and its declared weight (in terms of European credits) has not changed more than ± 3. The 11 chosen subjects are listed in Table IV. In the Italian academic system, grades for each passed subject are expressed in thirtieths, whereas the minimum grade to pass is 18. Nevertheless, given that most exams are oral or include an oral part, distinguishing between similar grades (eg., 19/30 or 20/30) is meaningless on a statistical basis. Thus, for $j = 1, \ldots, m$, we categorized the possible grades as follows:

$$g_j = \begin{cases} 4 & \text{if } grade(j) \geq 27/30 \\ 3 & \text{if } 24/30 \leq grade(j) < 27/30 \\ 2 & \text{if } 21/30 \leq grade(j) < 24/30 \\ 1 & \text{if } 18/30 \leq grade(j) < 21/30 \\ 0 & \text{if exam } j \text{ has not been passed} \end{cases}$$

If g_j is different from 0 for a given j, then, as we have explained, its corresponding a_j is different from 0 as well:

$$a_j = \begin{cases} q & \text{if } q \text{ attempts were necessary to pass } j \\ 0 & \text{if exam } j \text{ has not been passed} \end{cases}$$

The *class* $C \in \{0, 1\}$ that we want to predict is: has the student finished his/her studies, or dropped out?

attribute	explanation
#out of course	number of out of course enrollments
#tax flags	number of times that fees were overdue
#benefits	number of times that some economical benefit was granted

TABLE III
CUMULATIVE DATA

code	subject
S1	Algorithms and Data Structures
S2	Computer Architecture
S3	Introduction to Databases Systems
S4	Probability and Statistics
S5	Numerical Analysis
S6	Introduction to Calculus
S7	Introduction to Physics
S8	Programming Languages and Paradigms
S9	Algebra and Discrete Mathematics
S10	Introduction to Programming
S11	Operative Systems

TABLE IV
CHOSEN SUBJECTS, IN NO PARTICULAR ORDER

Experimental setup. We have chosen 498 students first enrolled in the bachelor degree in Computer Science at the University of Ferrara (Italy) in a year between 2001 and 2014; of these, 74% completed their studies, and the remaining ones are classified as dropout. As explained in the previous section, for us $n = 3$, $r = 3$, and $m = 11$. In order to remedy the class imbalance, we randomly extracted a *training set*, with precisely 200 students, with 50% positive and 50% negative w.r.t. to the class; the remaining students were inserted into a *test set*. From the former, we have further selected a training subset by setting $n = 0$. In this way, in parallel to testing our proposal, we prove that static data are actually unnecessary

learner	best	average	worst
logistic	0.0353	0.0353	0.0354
random forest	0.0560	0.0561	0.0562

TABLE V
BEST, AVERAGE, AND WORST HYPERVOLUME OF THE LAST POPULATION AMONG 10 EXECUTIONS FOR EACH LEARNER

for dropout prediction, and eliminate typical biases from the learning process (such as gender in certain scientific courses, or school or provenance). We have executed our model, with no static data, 10 times with *random forest* and 10 times with *logistic regression*. For each execution, the number of evaluation has been set to 10000 (100 generations, for a population of 100 individuals). The probabilities of crossover and mutation are self-adaptive, as in [38]. No pre-processing, in the classical sense, is necessary, considering that the system itself performs a (dynamic) pre-processing at each generation, as we have explained, and taking into account that there are no null values, that every column presents non-zero variance, and that columns are independent from each other. In terms of computational complexity, observe that NSGA-II uses the so-called *fast non-dominated sorting* algorithm [30], which compares each solution with the rest of the solutions and stores the results so as to avoid duplicate comparisons between every pair of solutions; this means that for a problem with k objectives ($k = 2$, for us) and a population with N solutions ($N = 100$, for us), this method needs to conduct $k \cdot N \cdot (N-1)$ objective comparisons, which means that it has time complexity $O(k \cdot N^2)$ [39].

Results and validation. The classical measure to compare different executions of the same evolutionary model is the *hypervolume* of the last population, which can be defined as a measure of the space under the solutions' front. The smaller the hypervolume, the better is the set of solutions in terms of objectives. The hypervolume statistics for learner are shown in Table V; as we can see, there are only minimal differences. Among the 10 execution for each learner, we selected the one whose last population has best hypervolume.

Thanks to the fixed structure of the Italian academic system, we can now associate each individual of both fronts with its corresponding temporal indicator, as explained in the previous section. The exam sessions are denoted with February, July, and September, and each one of them is referred to a particular year. So, for example, an individual may be associated with 'September 1', meaning that, in order to apply the corresponding horizon to a particular student, we need to wait until the third exam session of the first year of enrollment. In this respect, though, one has to pay attention to the fact that, we have recalled, the structure of the course of studies may change along the years. We have already taken into account, in this experiment, these changes: as a matter of fact, we have linked every student to the specific structure of the course that corresponds to his/her academic year of first enrollment. Nevertheless, the temporal indicator must be uniform, and the natural choice is to use the *current structure* to compute it. Applying Algorithm 7 with minimum accepted accuracy 0.7 and maximum temporal indicator 'July 2' (that is, assuming that it is reasonable to wait until the second exam session of the second year before judging the academic performance of a student, and accepting a model that misjudges 3 out 10 students) gives us the highlighted results in Table VI and Table VII, in which we show every individual of the last front of the chosen execution, along with the following indicators: accuracy in training phase, accuracy in test phase, Choen's Kappa (which is a measure of the reliability of a classification), mean absolute error, and root mean absolute error. Consider, first, the chosen individual from the logistic model. The corresponding horizon requires to wait one exam session for *Computer Architecture* and one exam session for *Probability and Statistics*, which, in turn, requires to wait (in the current structure of studies) until the summer session of the first year of enrollment. The model that has been learned on the training data projected on this horizon (as explained in Section II) shows 0.732 of accuracy in training and 0.746 of accuracy in test, and this particular horizon will be referred to as *horizon 1*. Similarly, the chosen individual for the random forest model requires waiting two exam sessions for *Computer Architecture* and one exam session for *Programming Languages and Paradigms*, which, in turn, requires (in the current structure) to wait until the summer session of the second year of enrollment; the training accuracy is 0.743 and the test accuracy is 0.836, and this model will be referred to as *horizon 2*. Recall that our models were not designed to be interpretable. Unlike most solutions in the literature, by means of learning a prediction model we do not aim to *explain* why a certain student drops out from the studies; instead, we only aim to be able to predict his/her intention to drop out, therefore allowing for corrective actions. Therefore, an horizon, such the one chosen among those produced by the logistic regression schema, should not be misinterpreted by deducing, for example, that a student that has not passed *Computer Architecture* and *Probability and Statistics* in their respective first available sessions is classified as being at risk of dropping out. Instead, the correct interpretation is: a student can be classified as being at risk of dropping out (or not) by observing his/her academic results concerning, in particular, the results on the subjects *Computer Architecture* and *Probability and Statistics* after the respective first available exam session. Interestingly enough, the accuracy in training and in test show the following behaviour: for very short horizons, the training accuracy and the test accuracy are not extraordinarily high, but the models are very resilient to over-fitting, witnessed by the fact that training and test accuracy are very similar (and test accuracy is even higher than training accuracy in some cases); as the horizons become longer, the accuracies tend to grow (as expected) but the resulting models are less reliable on new data. By means of our decision making algorithm, we are able to identify a good equilibrium between the training accuracy, the over-fitting degree, and the applicability of the horizon (recall that long horizons are not really useful, as we have explained in Section II). The case of the horizon chosen among the random forest models is illuminating, in this sense, considering that the accuracy increases by almost 0.1 points. Moreover, this particular model shows a Cohen's Kappa of 0.363, which can be considered relatively high, indicating

		training		test				
weight	horizon	accuracy	indicator	kappa stat.	mean abs. err.	root mean sq. err.	accuracy	static data acc.
2	0 1 0 0 0 0 0 1 0 0 0	0.732	July 2	0.247	0.345	0.422	0.733	0.726
2	**0 1 0 1 0 0 0 0 0 0 0**	**0.732**	**July 1**	0.263	0.349	0.415	0.746	0.740
3	0 1 0 0 0 0 0 2 0 0 0	0.747	September 2	0.263	0.329	0.417	0.746	0.723
4	0 0 0 0 0 0 0 4 0 0 0 0	0.777	July 3	0.206	0.325	0.448	0.603	0.606
5	0 0 0 0 0 0 0 5 0 0 0 0	0.797	September 3	0.249	0.293	0.429	0.656	0.656
6	0 0 0 0 0 0 0 6 0 0 0 0	0.823	February 4	0.332	0.247	0.397	0.736	0.733
7	0 0 0 0 0 0 0 7 0 0 0 0	0.848	July 4	0.418	0.208	0.367	0.800	0.790
8	0 0 0 0 0 0 0 8 0 0 0 0	0.873	September 4	0.494	0.173	0.335	0.843	0.830
9	0 0 0 0 0 0 0 9 0 0 0 0	0.883	February 5	0.520	0.163	0.324	0.856	0.846
10	0 0 0 0 0 0 0 10 0 0 0 0	0.919	July 5	0.580	0.140	0.309	0.886	0.873
11	0 0 0 0 0 0 0 11 0 0 0 0	0.929	September 5	0.642	0.120	0.283	0.910	0.893
19	0 0 0 6 0 0 0 12 0 1 0	0.939	February 6	0.725	0.075	0.221	0.936	0.916
21	0 0 0 0 0 0 0 21 0 0 0	0.944	February 9	0.806	0.062	0.162	0.960	0.936
23	0 0 0 0 0 0 0 22 0 1 0	0.959	July 9	0.848	0.060	0.159	0.970	0.956

TABLE VI
DECISION MAKING (LOGISTIC REGRESSION)

individual		training		test				
weight	horizon	accuracy	indicator	kappa stat.	mean abs. err.	root mean sq. err.	accuracy	static data acc.
1	0 1 0 0 0 0 0 0 0 0 0	0.717	July 1	0.217	0.367	0.429	0.726	0.683
3	**0 2 0 0 0 0 0 1 0 0 0**	**0.742**	**July 2**	0.363	0.331	0.407	0.836	0.743
4	0 0 0 0 0 0 0 4 0 0 0	0.782	July 3	0.206	0.330	0.457	0.603	0.620
5	0 0 0 0 0 0 0 5 0 0 0	0.797	September 3	0.249	0.299	0.436	0.656	0.620
6	0 0 0 0 0 0 0 6 0 0 0	0.823	February 4	0.340	0.239	0.400	0.743	0.743
7	0 0 0 0 0 0 0 7 0 0 0	0.848	July 4	0.429	0.193	0.363	0.806	0.813
8	0 0 0 0 0 0 0 8 0 0 0	0.873	September 4	0.507	0.158	0.328	0.850	0.856
9	0 0 0 0 0 0 0 9 0 0 0	0.888	February 5	0.507	0.144	0.312	0.850	0.893
10	0 0 0 0 0 0 0 10 0 0 0	0.919	July 5	0.614	0.116	0.286	0.900	0.896
11	0 0 0 0 0 0 0 11 0 0 0	0.929	September 5	0.681	0.097	0.262	0.923	0.920
16	0 0 0 4 0 0 0 12 0 0 0	0.944	February 6	0.702	0.082	0.224	0.930	0.940
20	0 1 0 1 0 0 0 18 0 0 0	0.949	February 8	0.834	0.075	0.169	0.966	0.963
22	0 1 0 1 0 0 0 20 0 0 0	0.959	September 8	0.911	0.058	0.128	0.983	0.983
39	0 0 0 0 0 17 0 22 0 0 0	0.984	July 9	0.895	0.045	0.138	0.980	0.983

TABLE VII
DECISION MAKING (RANDOM FOREST)

that this model behaves in a reliable way. Recall that, in our experiment, $n = 0$. The reason underlying this choice is avoiding learning a biased classifier, and building a model that consider students only by their performances; when training data are unbalanced with respect to some category (such as male students, for example), they generate less accurate models with respect to new data corresponding to female students, and, more in general, every static attribute that is known to show, statistically, high information gain, works *against* the overall reliability of a model as we have designed it, and this is why we decided to exclude them completely. For further confirmation of this effect, observe the last column of Table VI and Table VII, in which we have computed the test accuracy of the same horizons enriched with the three available static attributes: gender, province of residence, and type of school (that takes into account the type of preparatory school attended before first enrolling at the University of Ferrara). Focus, in particular, on the logistic models: adding the static data causes the accuracy to *worsen* in 12 cases out of 14; as we have already observed, this phenomenon is probably due to the fact that university students *do not* distribute uniformly among genders and other static characteristics, especially in specific courses such as Computer Science. Random forest models are less prone to this effect; nevertheless, the accuracy decreases in 6 out of 14 cases, and it remains unchanged in other 2 cases. Moreover, the cases in which the accuracy improves when static data are added are cases in which the horizon is extremely long, therefore, as we have noted, they are cases in which the model is at very high risk of over-fitting and is useless from a practical point of view.

To conclude this section, we show the results of testing the chosen horizons (*horizon 1* and *horizon 2* - in both cases, we have also tested their respective version with static data) with other classifiers. To widen the range of classifier types, we have used logistic regression, random forest, and *decision trees*, in particular the C4.5 implementation known as J48, with standard parameters (covering, in this way, functions and trees, but also interpretable and non-interpretable models). As it can be see from Tab. VIII, the chosen horizons allow to learn relatively accurate classifiers, even if a model different from the one used for training is used. From this experiment, it seems the best performing model is the one trained with random forest, learned with a decision tree, and without static data.

Statistical tests. In the current literature, when random processes are involved, statistical tests are often performed to establish whether the results are (statistically) reliable. In our case, it so happens that the 10 executions for each learning algorithm produced Pareto fronts very similar to each other; they are, in actuality, identical to each other up to horizons before the start of the fourth year (that is, the first out-of-

	logistic regression							
	without static data				with static data			
	accuracy	kappa stat.	root mean sq. err.	area under curve	accuracy	kappa stat.	root mean sq. err.	area under curve
horizon 1	0.74	0.25	0.41	0.81	0.74	0.25	0.42	0.77
horizon 2	0.78	0.30	0.41	0.83	0.69	0.23	0.42	0.81
	random forest							
horizon 1	0.74	0.24	0.42	0.78	0.68	0.20	0.44	0.78
horizon 2	0.83	0.35	0.41	0.80	0.74	0.26	0.43	0.80
	decision tree							
horizon 1	0.77	0.25	0.41	0.71	0.66	0.21	0.43	0.76
horizon 2	0.82	0.32	0.41	0.73	0.82	0.32	0.41	0.73

TABLE VIII
TEST RESULTS WITH DIFFERENT CLASSIFIERS

course year). This is a clear indication that our process has found, in general, solutions that are very close to the global optimums of the problems, and that comparing solutions from different executions does not make sense. However, in order to prove that our solution is statistically reliable, we have performed the following experiment. We have considered the two chosen horizons (which entail two dynamically pre-processed training data sets of 200 instances each), and tested their ability to predict the class using six standard learning algorithms, in 10-fold cross-validation mode, against 6 other data sets each: 5 of these correspond to *randomly-generated* horizons (all generated under the condition of entailing a temporal indicators before 'September 2'), and the last one is the training data set with *static data* only. The purpose of this test is twofold: first, we prove that the horizons that have been found behave, indeed, much better than randomly generated ones (which, paired with the fact that all executions produced similar fronts, is consistent with the fact that our process found very good solutions of the problem), and, second, we justify the increased computational time of finding the horizons instead of using simple, plain, static data. The chosen classifiers are: decision tree, random forest, logistic regression, support vector machine, multi-layer perceptron (with one hidden layer), and a deep learning architecture; in the latter case, the chosen configuration is: one internal dense layer of 100 units with activation function *ActivationRELU*, and output layer of 2 units with activation function *ActivationSoftmax* and loss function *LossMCXENT*. Observe that an horizon that has been randomly generated under the constraint of corresponding to a not-too-far temporal indicator does entail a data set for which the classification problem makes perfect sense, and it is perfectly consistent with the fact that the resulting accuracies are still acceptable.

As Tab. IX and Tab. X show, both horizons chosen after the execution of the implemented evolutionary algorithm behave much better than other possible horizons, and better than the data set obtained by static data only.

V. CONCLUSIONS

Being able to consistently predict the intention of a student to abandon the course of studies in which he or she has enrolled is very important for educators. In some European university systems, it has also some economical value, as public founding depends, among many other parameters, on the rate of success of students that decide to enroll. In the recent literature many different models have been proposed. Some of them are based on classical static descriptors, such as the gender or the age of a student, and try to identify the reasons behind the success or the failure of students; some other proposed solutions include psychological and attitude tests to be administered to the students at some stage. We have proposed an optimization model that allows us to identify the earliest possible moment in which a reliable prediction is possible, and showed that a prediction model can be built that is not gender-, or age-biased, but it is based solely on the performances of the student. We tested our methodology with real data from the Degree in Computer Science of the University of Ferrara (Italy). In particular, we have been able to build two prediction models. In the first one, the career of a student is analyzed after two semesters, and the accuracy in predicting his/her intention of dropping out ranges from 74% to 77%. In the second one, the career is analyzed after four semesters, and the accuracy of the prediction ranges between 78% and 83%. Therefore, we have proved that our system allows one to build a prediction model with a sufficiently high accuracy, but, at the same time, is able to predict the intention of dropping out early enough in the academic career so that some kind of corrective action can be attempted. In our knowledge, this is the first proposal of a temporal optimization model for academic success prediction. As it has been designed, our system can be seamlessly implemented in a university data warehouse. After an initial phase in which the model is optimized and a specific horizon is chosen for every specific degree of studies, the classifier can be transformed in an algorithm that simply checks, periodically, every student at the horizon time. Since we have decided that the outcome prediction is reliable enough at that moment, we can program the system to alert the university's administration if a student is classified as being at risk of dropping out. Simple corrective actions can be started at that point, that range from offering counseling to the student to proposing personalized tutoring.

There are two outstanding tasks at this point. First, we would like to test our systems for other courses of studies and other university systems, in order to verify that the model is adaptable. Second, we would like to build a prototype that verifies the realizability of a early alert system as the one described above. Both research directions are currently being explored.

Acknowledgments. The authors would like to thank the

	horizon 1	*random 1*	*random 2*	*random 3*	*random 4*	*random 5*	*static data*
decision tree	0.68	0.68	0.68	0.62	0.69	0.67	0.53
random forest	0.69	0.68	0.67	0.62	0.68	0.67	0.48
logistic regr.	0.72	0.68	0.67	0.65	0.63	0.63	0.51
support vect. m.	0.72	0.68	0.68	0.65	0.69	0.67	0.52
multi-layer perc.	0.71	0.68	0.67	0.64	0.69	0.69	0.49
deep learning	0.72	0.68	0.66	0.63	0.63	0.63	0.51

TABLE IX
TWO-TAILED, PAIRED T-TEST CORRECTED, WITH CONFIDENCE 0.05, FOR *horizon 1*, MEASURING THE ACCURACY.

	horizon 2	*random 6*	*random 7*	*random 8*	*random 9*	*random 10*	*static data*
decision tree	0.74	0.61	0.66	0.63	0.67	0.57	0.53
random forest	0.73	0.61	0.66	0.65	0.67	0.57	0.48
logistic regr.	0.73	0.61	0.66	0.66	0.67	0.58	0.51
support vect. m.	0.72	0.61	0.66	0.66	0.67	0.58	0.52
multi-layer perc.	0.72	0.61	0.66	0.66	0.67	0.59	0.49
deep learning	0.73	0.61	0.66	0.66	0.67	0.59	0.51

TABLE X
TWO-TAILED, PAIRED T-TEST CORRECTED, WITH CONFIDENCE 0.05, FOR *horizon 2*, MEASURING THE ACCURACY.

University of Ferrara project FIR 2017: *Academic Success and Abandoning Rate Prediction in the Degree in Computer Science of the University of Ferrara (Italy).*

REFERENCES

[1] A. Saa, "Educational data mining and students' perfomance prediction," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 5, pp. 212 – 220, 2016.

[2] B. Bahritidinov and E. Sánchez, "Predicting academic success by using context variables and probabilistic classification," *Expert Systems*, vol. 34, no. 4, 2017.

[3] K. Tani and A. Gilbey, "Predicting academic success for business and computing students," *International Journal of Information and Communication Technology Education*, vol. 12, no. 4, pp. 15–24, 2016.

[4] A. Daud, N. Aljohani, R. Abbasi, M. Lytras, F. Abbas, and J. Alowibdi, "Predicting student performance using advanced learning analytics," in *Proc. of the 26th International Conference on World Wide Web Companion*, 2017, pp. 415–421.

[5] R. Kappe and H. van der Flier, "Predicting academic success in higher education: what's more important than being smart?" *European Journal of Psychology of Education*, vol. 27, no. 4, pp. 605–619, 2012.

[6] V. Martinho, C. Nunes, and C. Minussi, "Prediction of school dropout risk group using neural network fuzzy ARTMAP," in *Proc. of the Federated Conference on Computer Science and Information Systems*, 2013, pp. 111–114.

[7] S. Ting, "Predicting academic success of first-year engineering students from standardized test scores and psychosocial variables," *International Journal of Engeneering Education*, vol. 17, no. 1, pp. 75–80, 2001.

[8] G. Prieto-Adanez and A. Dias-Velasco, "Predicting academic success of engineering students in technical drawing from visualization test scores," *Journal for Geometry and Graphics*, vol. 6, no. 1, pp. 99–109, 2002.

[9] S. Borkar and K. Rajeswari, "Attributes selection for predicting students' academic performance using education data mining and artificial neural network," *International Journal of Computer Applications*, vol. 86, no. 10, 2014.

[10] B. Bharadwaj and S. Pal, "Mining educational data to analyze students' perfomance," *International Journal of Advanced Computer Science and Applications*, vol. 1, no. 6, 2011.

[11] R. Baker, D. Lindrum, M. Lindrum, and D. Perkowski, "Analyzing early at-risk factors in higher education e-learning courses," in *Proc. of the 8th International Conference on Educational Data Mining*, 2015, pp. 150–155.

[12] G. Dekker, M. Pechenizkiy, and J. Vleeshouwers, "Predicting students drop out: A case study," in *Proc. of the Educational Data Mining International Conference*, 2009, pp. 41–50.

[13] M. Tan and P. Shao, "Prediction of student dropout in e-learning program through the use of machine learning method," *International Journal of Education Techniology*, vol. 10, no. 1, pp. 11–17, 2015.

[14] E. M. Lauría, E. Moody, S. Jayaprakash, N. Jonnalagadda, and J. Baron, "Open academic analytics initiative: initial research findings," in *Proc. of the 3rd Conference on Learning Analytics and Knowledge*, 2013, pp. 150–154.

[15] C. Márquez-Vera, A. Cano, C. Romero, A. Noaman, H. Fardoun, and S. Ventura, "Early dropout prediction using data mining: a case study with high school students," *Expert Systems*, vol. 33, no. 1, pp. 107–124, 2016.

[16] S. Ameri, M. Fard, R. Chinnam, and C. Reddy, "Survival analysis based framework for early prediction of student dropouts," in *Proc. of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 903–912.

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

[18] S. Sinha, *Mathematical Programming: Theory and Methods*. Elsevier, 2006.

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

[20] H. Karloff, *Linear Programming*. Birkhauser Basel, 1991.

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

[22] D. Bertsekas, *Non-Linear Programming (Second ed.)*. Athena Scientific, 1999.

[23] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[24] F. Jiménez and J. Verdegay, *Evolutionary Computation and Mathematical Programming*. Springer, 2001, pp. 167–182.

[25] A. Olsson, *Particle Swarm Optimization: Theory, Techniques and Applications.* Nova Science Publishers, 2011.

[26] A. Brunello, F. Jiménez, E. Marzano, J. Palma, G. Sánchez, and G. Sciavicco, "Towards semi-automatic human performance evaluation: The case study of a contact center," *Intelligent Data Analysis*, vol. 22, no. 4, pp. 867–880, 2018.

[27] A. Karegowda, A. Manjunath, and M. 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.

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

[29] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011.

[30] 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.

[31] 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.

[32] 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.

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

[34] 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.

[35] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs (3rd Ed.)*. Berlin, Heidelberg: Springer, 1996.

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

[37] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

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

[39] C. Bao, L. Xu, E. Goodman, and L. Cao, "A novel non-dominated sorting algorithm for evolutionary multi-objective optimization," *Journal of Computational Science*, vol. 23, pp. 31 – 43, 2017.

AUTHORS' BIOGRAPHIES

Fernando Jiménez is Associate Professor at the University of Murcia (Spain), and holds a M.D. in Computer Science from the University of Granada (Spain), and a Ph.D. in Computer Science from the University of Murcia (Spain). His research interests include evolutionary computation, multi-objective constrained optimization, soft computing, evolutionary fuzzy systems and data mining.

Alessia Paoletti is a graduate student in the Master Degree of Computer Science, at the University of Trieste. She completed a final project in educational data mining applied to the prediction of academic success rate for students during her Bachelor Degree at the University of Ferrara.

Gracia Sánchez is Associate Professor at the University of Murcia (Spain), and holds a M.D. in Computer Science from the Polytechnic University of Valencia (Spain) and a Ph.D. in Computer Science from the University of Murcia (Spain). Her research interests include multi-objective constrained optimization, soft computing, evolutionary fuzzy systems and data mining.

Guido Sciavicco is Associate Professor at the University of Ferrara (Italy), and holds a M.D. in Computer Science and a Ph.D. in Computer Science, both from the University of Udine (Italy). He has been researching in temporal, spatial, and modal logic for over 10 years, especially in satisfiability and model-checking problems. Recently, he has focused on studying temporal data mining problems with applications.