Feature Extraction for Next-Term Prediction of Poor Student Performance

Agoritsa Polyzou and George Karypis, *Fellow, IEEE*

***Abstract*—Developing tools to support students and learning in a traditional or online setting is a significant task in today's educational environment. The initial steps toward enabling such technologies using machine learning techniques focused on predicting the student's performance in terms of the achieved grades. However, these approaches do not perform as well in predicting poor-performing students. The objective of our work is twofold. First, in order to overcome this limitation, we explore if poorly performing students can be more accurately predicted by formulating the problem as binary classification, based on data provided before the start of the semester. Second, in order to gain insights as to which are the factors that can lead to poor performance, we engineered a number of human-interpretable features that quantify these factors. These features were derived from the students' grades from the University of Minnesota, an undergraduate public institution. Based on these features, we perform a study to identify different student groups of interest, while at the same time, identify their importance. As the resulting models provide us with different subsets of correct predictions, their combination can boost the overall performance.**

***Index Terms*—Academic student success, classification, performance prediction, feature importance.**

I. Introduction

Higher educational institutions constantly try to improve the retention and success of their enrolled students. According to the US National Center for Education Statistics [1], 60% of undergraduate students on four-year degrees will not graduate at the same institution where they started within the first six years. At the same time, 30% of college freshmen drop out after their first year of college. As a result, colleges look for ways to serve students more efficiently and effectively. Student experience and success are the prioritized factors for educational institutions. Educational data mining and learning analytics have been developed to provide tools for supporting the learning process, like monitor and measure student progress, but also, predict success or guide intervention strategies.

Most of the existing approaches focus on identifying students at risk who could benefit from further assistance in order to successfully complete a course or activity. A fundamental task in this process is to actually predict the student's performance in terms of grades, as this is the measurable quantity that is available in the data. While reasonable prediction accuracy has been achieved [2], [3], there is a significant weakness of the models proposed to identify the poor-performing students [4]. Usually, these models tend to be over-optimistic for the performance of students, assigning them a higher grade, while the student might as well fail the course. This is happening because the majority of the students do well, or have satisfactory enough performance, in order to complete successfully a course or activity.

In this paper, we investigate the problem of predicting the performance of a student at the end of the semester before he/she actually takes the course, i.e., before we get any indicators about the student performance during the semester. We do not use any data obtained during the semester that describe the performance, or the interactions of the students with an employed Learning Management System (LMS). This allows us to predict student performance and gain insights before student enrollment. We can provide these insights to students, advisors, and instructors in order to intervene early, while students can modify their course registration. We want to avoid students dropping or failing a course after they have already paid for its credits, and spent invaluable time and effort during the semester. In such a case, the student could consider another course that might be a better fit for him. This might also cause a delay in graduation time.

In order to focus on the poor-performing students, who are the ones that need these systems the most, the prediction problem is formulated as a classification task, where two groups of students are formed according to their course performance. Classification is the task of assigning instances to one of several predefined categories/classes. We essentially identify two complementary groups of students, the ones that are likely to successfully complete a course or activity, and the ones that seem to struggle. After identifying the latter group, we can provide additional resources and support to enhance their likelihood of success.

The question that arises at this point is what "success" and "failure" mean. Unique challenges related to higher edu–cation include methodological difficulties when measuring performance. Specifically, good performance can be relative or not. For example, a B− grade might be considered a bad grade for an excellent student, while being a good grade for a very weak

Manuscript received September 20, 2018; revised March 20, 2019; accepted April 9, 2019. Date of publication April 26, 2019; date of current version June 17, 2019. This work was supported in part by the National Science Foundation under Grants 1447788, 1704074, 1757916, and 1834251, in part by the Army Research Office (W911NF1810344), in part by the Intel Corporation, and in part by the Digital Technology Center at the University of Minnesota. *(Corresponding author: Agoritsa Polyzou.)*

The authors are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA
(e-mail: polyz001@umn.edu; karypis@umn.edu).

Digital Object Identifier 10.1109/TLT.2019.2913358

student. Similarly, a B+ for an easy class can be a rather bad grade, whereas a B in a hard class can be a good grade. To cover such cases, we investigated various ways to define groups of students, according to their achieved grades, which are interesting to predict from an intervention standpoint. In particular, we investigate the following groups and their complement, formed by students taking a class:

- failing students (**Fgr**),
- students dropping the class (**Wgr**),
- students performing worse than expected (**RelF**),
- students performing worse than expected, while taking into consideration the difficulty of the course (**RelCF**).

In order to gain more insight into the learning process and its most important characteristics, we have identified factors that might affect student performance. Using the historical grades of the students, we have created features that capture possible factors that influence the grades that a student will get at the end of the semester. The students' grades in courses are easily accessible to any institution. As a result, anyone can extract these features from the available data and perform the same analysis, irrespectively of the other factors. Using these features, we present a comprehensive study that is designed to answer the following questions: which features are good indicators of a student's performance? which features are the most important? The findings are interesting, as different features are the most important for different classification tasks.

We build models with different subsets of the constructed features, and notice that they have different performance, as expected. What we also verified is that these different models do not predict the same set of instances as positive. If we combine these models with each other, we can improve the performance of our classifiers.

The rest of the paper is organized as follows. Section II reviews the work in the area of predicting student performance at the end of the semester. In Section III, there is an overview of the data that we used. Section IV provides precise definitions of various concepts and terms used in the paper and formally introduces the notation that is used. The classification tasks, features extracted and methods tested are described in Section V. In Section VI, there is a detailed description of the experimental set up, and Section VII discusses the results about the different methods tested, the feature importance study, as well as the benefits of combining the predictions of two classifiers that use different input features. Section VIII contains the conclusions of the study.

II. Related Work

A. *Predicting Student Performance*

In the context of learning analytics, data collected by institutions or learning management systems are used to predict student performance and identify important factors that can lead to successful completion of a course. As we are interested in estimating next-term student performance, we will review the related work in this area of research.

The binary classification has been used in various educational problems, like when predicting if a student will drop out from high school [5] or to predict if a student will pass or fail a module in a distance learning setting [6]. Multi-label classification has been applied to provide a qualitative measure of students' performance. In [7], decision tree and naive Bayes classifiers are used with data from a survey. Attributes collected by a learning management system (LMS) have been employed to estimate the outcome as Fail, Pass, Good and Excellent [8], or to classify students with similar performance into buckets [9]. Some approaches [10], [11] test different ways to label the student performance, with two (pass or fail, above or below a threshold) or more labels (one label per letter grade, one label per interval). The majority of the aforementioned approaches are small-scale studies, that are applied to one or a limited number of courses. LMS log data from online courses were used in [12] to classify the students as "Successful Completers", "Non-successful Completers", or "Withdrawers". However, in that case, the classes were not so unbalanced, with 214, 70, 70 students, respectively.

Any conclusions made on the factors that influence the students' performance do not apply to other courses. The applicability of a single model over different courses is discussed in [13]. A larger-scale study [14] with 17 blended courses confirmed that the portability of models among different courses is low. Additionally, the study included the prediction of pass/failing performance using binary logistic regression based on LMS features. The additional information that comes from these features does not significantly improve the accuracy of the models. According to [15], [16], the predictive power of LMS features is limited, and selecting the most meaningful ones is a challenging task.

Other data mining techniques have also been used for similar problems, such as clustering. K-means algorithm is employed to cluster the students and study the factors that affect their progression through their studies [17], [18]. Clustering has also been applied for identifying key features of student performance in educational video games [19].

In recent years, approaches based on recommender systems have been also utilized in the area of learning analytics. Once we have collected enough data, these approaches are a good match for learning analytics, as they enable us to work in a larger scale. Initially, the term "next-term grade prediction" was introduced by Sweeney et al. [4] in the context of higher education, and it refers to the problem of predicting the grades for each student in the courses that he/she will take during the next semester. This means that we can utilize the history of courses and their corresponding grades, but not the in-term performance. In that paper, models based on SVD and factorization machines (FM) were tested. In another approach [20], the previous performance of students controls the grade estimation in two different ways, to build latent factors, but also to scale the influence of prior courses on the estimated grade. In [21], some additional state-of-the-art methods were used, as well as, a hybrid of FM and random forests (RF). The data used are the historical grades and additional content features, representing student, course and instructor characteristics. At the same setting, [2] and [3] developed course-specific methods to perform next-term grade prediction based on linear regression and matrix factorization.

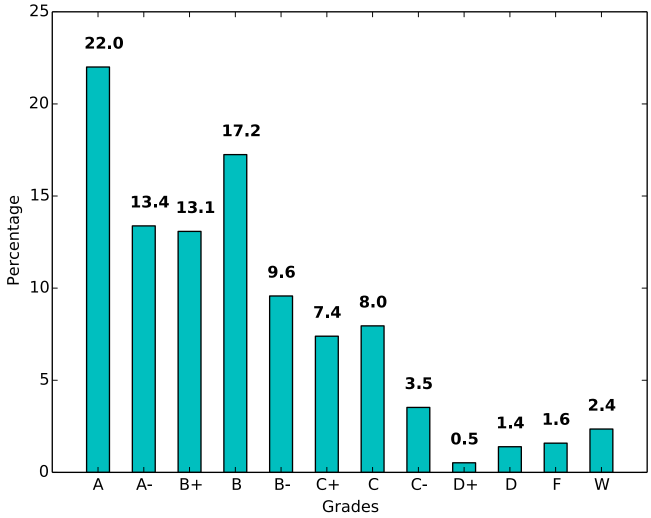


Fig. 1. Percentage of each letter grade with respect to the total grades.

All these methods assign a specific numerical grade to each student's attempt to take a course. A limitation identified in these approaches was that the developed models perform poorly for failing students. In [22], failing students have been completely removed from the dataset. As this is the subpopulation of students that needs additional support the most, it is very important for a model to be able to accurately identify these students at risk, and the factors that influence their performance.

This work is a more general study of the factors that influence the student performance, in a very large scale. The only observed data that we have available are the students' grades at the end of the semester. In our approach, we formulated this problem as a binary classification task, in order to detect different group of students.

B. Feature Selection

When there is a significant number of features without any quality guarantees, feature selection approaches are applied to gain insights on the underlying concepts. Supervised selection methods are of interest to us since we have the label information available. These methods are divided into three categories: filter, wrapper, and embedded models [23]. Filter models evaluate the features based only on the characteristics of the data, e.g., mutual information [24], relief [25]. The best features are selected first, and then, they are used to build any classifier. Wrapper models chose first the classifier, and then determine the quality of a subset of features by the accuracy achieved on the predetermined classifier [26]. Lastly, the embedded models combine the two models by performing feature selection and classifier building at the same time [27]. In our case, we will use an approach based on wrapper models, with predefined subset of features to examine.

III. Dataset

An undergraduate student enrolled to a college or university has to take some courses each semester, and receive a satisfactory grade on an A–F basis in order to successfully complete them. Depending on the student's degree program, these courses might be required, electives, or simply courses that the student takes for his/her own advancement, intellectual curiosity, or enjoyment.

The original dataset was obtained from the University of Minnesota and it spans 13 years. We removed any instances that received a letter grade not in the A–F grading scale (A, A−, B+, B, B−, C+, C, C−, D+, D, F). If a student withdraws from a course after the first two weeks of classes, it is denoted by the letter 'W' in the student's transcript. Statistics about the grades in the dataset are shown in Fig. 1. In our dataset, the letter grade A is the most common. We extract features for the instances occurring during the last 10 fall and spring semesters. Given a semester and a target course, we utilize all the students that had taken the course before, and for each student taking a course, we extract a set of features.

Additionally, we generate features for the instances awarded with the letter W, but we do not utilize them in any other way during the feature extraction process. These will be used only when trying to predict the students that drop-out from a course.

IV. Definitions and Notation

A *grade*, $g_{s,c}$, in this work refers to the performance of a student, s, in a course, c, at the end of the semester t. If we perform prediction for an instance, we also refer to s, c and t as *target* student, term and course respectively. All the courses that a student took in past semesters, before the target semester t, are the *prior courses*, denoted by C_{s,all_prior}. The set of courses for a single semester t is denoted as $C_{s,t}$. The cardinality of sets is denoted as $|C_{s,t}|$. Any course j worths a specified number of credits, cr_j. A student is active in the program for a number of terms, nterms_active$_{s,t}$, before target semester. Additionally, for a course c there might exist a stated set of courses that are required for a student to take before attempting c. We refer to this set as the *prerequisite courses*.

V. Classification Problems

A. Classification Tasks

Our motivation was to identify groups of students that need further assistance and guidance in order to successfully complete a course. These students could benefit from informed interventions. We consider this to be a binary classification problem, where these students form one of the classes and the remaining students form the other class.

We consider different ways of measuring when a student does not do well in a course to deal with the performance measurement challenges we mentioned earlier. Unsatisfactory performance can occur when the earned grade represents a performance that is below the student's potential. We considered the following four ways for labelling, resulting to four different classification tasks:

1) Failing student performance, i.e., letter grades D and F (denoted as the **Fgr** task).
2) The letter grade W (denoted as the **Wgr** task). This represents the instances when the student dropped the course. This behavior is worrisome as it shows that

TABLE I
FEATURE GROUPS DESCRIBING THE TARGET STUDENT s IN THE TARGET SEMESTER t

(1) Student's status in terms of grades. (**grades**)	• Average grade of s in prior courses C_{s,all_prior}. $\sum_j g_{s,j}/\lvert C_{s,all\_prior}\rvert$, for j in C_{s,all_prior}. • GPA of s, i.e., weighted average of the grades in prior courses w.r.t. the credits worth. $\sum_j g_{s,j} cr_j / \sum_j cr_j$, for j in C_{s,all_prior}. cr_j is the number of credits of course j. • GPA of s over the prior courses that belong in his/her major. • GPA of s over the prior courses that do not belong in his/her major. • GPA of s over the courses taken the previous semester, i.e., at the semester (t-1). • GPA of s over prior courses taken the past two semesters, i.e. at semesters (t-1) and (t-2). • GPA of s over prior courses taken on fall, spring and summer semesters. Essentially, here there are 3 features, one for each semester type. • Average grade of courses that s took with the same corresponding credit. There are 6 different features, each corresponding to prior courses that worth 1, 2, 3, 4, 5 or 6 credits. • GPA of courses that s took with different course levels. There are 6 levels (1xxx, 2xxx, 3xxx, 4xxx, 5xxx, or 8xxx). Higher level courses are more advanced.
(2) Other info indicating a student's status. (**status**)	• The number of prior courses, $\lvert C_{s,all\_prior}\rvert$. • Student's major. Included majors: Aerospace Engineering, Biomedical Engineering, Chemical Engineering, Chemistry, Civil Engineering, Computer Science, Electrical Engineering, Materials Science, Mathematics, Mechanical Engineering, Physics, and Statistics. • The total credits that s has earned in prior courses. $\sum_j cr_j$, for j in C_{s,all_prior}. • Indicator whether target semester t is a fall, spring, or summer semester. • Indicator whether the student has ever registered for the summer semester. This is an indicator of the past behavior of the student. • The number of semesters that the student is active, nterms_active$_{s,t}$. • The number of years student s is in the program. • The number of transferred credits. It is quite common for students to transfer some credits from other institutions, or from qualified courses they took at high school.
(3) Student's course load. (**load**)	• Average credits s earned in prior courses per semester. $\sum_j cr_j/$nterms_active$_{s,t}$, for j in C_{s,all_prior}. • The number of credits s earned in the past semester. $\sum_j cr_j$, for j in $C_{s,t-1}$. • The number of credits earned in the current semester. $\sum_j cr_j$, for j in $C_{s,t}$. • The number of courses taken the current semester. $\lvert C_{s,t}\rvert$. • Ratio of s's course load in the current semester to his/her average course load over the past semesters. This is a way to compare the usual load of the student with the load for the target semester. $(\sum_i cr_i)/(\sum_j cr_j/$nterms_active$_{s,t})$, for i in $C_{s,t}$ and j in C_{s,all_prior}.

The target student, course, semester are denoted as s, c, and t, respectively. The set C_{s,all_prior} represents the courses that the student took before the target semester t. For semester t, $C_{s,t}$ represents the set of courses that student s took on semester t. The term nterms_active$_{s,t}$ refers to the number of semesters that student s was active, i.e., taking courses, before semester t. The term cr_j refers to the credits that course j worths.

either the student was not interested in the course anymore or he/she expects to perform poorly.

3) Student performance that is worse than expected, i.e., the grade achieved is more than two letter grades lower than the student's GPA (denoted as the **RelF** task).
4) Student performance that is worse than expected while taking into consideration the difficulty of the course (denoted as the **RelCF** task). The difficulty of a course is expressed by the average grade achieved by the students that took the course in prior offerings. A positive instance is when the grade achieved is more than two letter grades lower than the average of the student's GPA and the course's prior average grade.

While the first two ways of labeling are absolute, the last two are relative to the student and the course of each instance. Statistics for the different classification tasks can be found at Table III.

As discussed at the related work section, it is easier to predict the successful students. In order to have a better understanding of the relative difficulty of this task compared with the four tasks mentioned above, we also examined the task of predicting the students that completed a course with the grade A (denoted as the **Agr** task).

B. Extracted Features

Having as input the historical grading data, we want to study what factors can discriminate students with different performance. We derived from the original data different features that capture different possible factors for a student's poor performance. The features can be separated into three distinct categories: the student-specific (they are independent from course c), course-specific features (they are independent from student s) and student- and course-specific features (they are a function of both s and c). All extracted features are described in Tables I and II, where related features are grouped together into eight different subcategories. The keywords on bold are used to indicate the corresponding group of features later.

Note that for each $\{s, t, c\}$, where student s took course c in semester t, we generate a different set of features. Every set of features characterize a student's attempt to take course c at the specific point of his/her studies.

These features are either numerical, categorical or indicator variables. For indicator features, we use the values of 0 or 1. The categorical features are encoded via a numerical value. For example, the feature about the current semester is categorical, and the values {fall, spring, summer} are transformed to {0, 1, 2}, respectively.

C. Methods Compared

In order to support students that need help to successfully complete a course, we will use classification techniques to identify them from the rest of the students. The instances of interest will be labeled as 1, and the rest as 0. The problem can be described

TABLE II
FEATURE GROUPS DESCRIBING THE STUDENT s IN TERM t AND COURSE c

(4) Course's difficulty and popularity. (**c-diff**)	• Relative course load in semester t (when s took c) w.r.t. the average credits of past students at the semester they had taken c. For each past student, compute the number of credits earned on that semester. Then, compute the fraction of $\sum_j cr_j$, for j in $C_{s,t}$, divided by the average number credits earned from past students on the same semester that they took course c. Values greater than 1 indicate heavier load than other students. • Average grade in c earned by past students. • Average grade in c of past students within the same major as the s. Now, filter the students in order to keep only the students that are in the same department as s. • Average grade in c of students belonging to c's major or not. This describes two features, by separating the past students to the ones that are in the same major as the department of c, and the ones that are out-of-the-department.
(5) Performance / Familiarity with the course's background and department. (**c-backgr**)	• Fraction of students in the same major as s that have taken the c. This feature measures how popular is course c across the students on the department of student s. • Fraction of students from s's major that took c. This feature shows how common is c in s's major. • Number of courses that s took and belong to c's department. Absolute measurement of how familiar is s with the department of the course c. • Ratio of courses that s took and belong to c's department. Relative measure of how familiar is s with the department of the course c. • Ratio of credits that s took and belong to c's department. Relative measurement of how familiar is s with the department of the course c, in terms of credits. • Ratio of credits that s took and belong to c's department and the average credits that past students took and belonged to c's department. This is a relative measurement of how familiar is s with the department of the course c, in comparison with past students. • GPA over the courses that s took and belong to c's department. This feature is a quantitative measure of student's performance in the c's department.
(6) Information about the prerequisites. (**prerequ**)	• GPA of the prerequisite and non-prerequisite courses that s has taken. Two features that show the performance of the student in prerequisite and other courses. • Number of the prerequisite courses taken by s, an absolute measurement. • Ratio of prerequisite courses taken by s. Relative measure to show how well-prepared the student is, in terms of the stated prerequisites. • Average terms past since prerequisite courses were taken by s. This feature will indicate how recently the prerequisite knowledge was acquired.
(7) Performance relative to the course's level. (**c-perform**)	• The number of lower, same and higher level courses w.r.t. the level of c. • GPA over lower, same, higher level courses w.r.t. the level of c.
(8) Course-specific features. (**c-spec**)	• Course level that c belongs to. • Indicator whether c is in the student's major or not. • Average grade earned by past students.

The target student, course, semester are denoted as s, c, and t, respectively. The set C_{s,all_prior} represents the courses that the student took before the target semester t. For semester t, $C_{s,t}$ represents the set of courses that student s took on semester t. The term nterms_active$_{s,t}$ refers to the number of semesters that student s was active, i.e., taking courses, before semester t. The term cr_j refers to the credits that course j worths.

TABLE III
STATISTICS FOR THE DIFFERENT CLASSIFICATION TASKS

Task	Fgr	Wgr	RelF	RelCF	Agr
Total instances	94,364	96,941	94,364	94,364	94,364
Positive instances	3,139	2,577	20,398	21,724	20,851
% of positive class	3.33	2.7	21.62	23.02	22.10

as follows. We are given a set of training examples that are in the form $(\mathbf{x}, y)$ and we want to learn their structure. We assume that there is some unknown function $y = f(\mathbf{x})$, that corresponds the feature vector $\mathbf{x}$ to a value y. In our case, $y = \{0, 1\}$. A classifier is an hypothesis about the true function f. Given unseen values of $\mathbf{x}$, it predicts the corresponding y values.

We tested the following classifiers [28], using scikit-learn library in Python [29]: Decision Tree (DT) [30] and Linear Support Vector Machine (SVM) [31] as base classifiers, and Random Forest (RF) [32] and Gradient Boosting (GB) [33] as ensemble classifiers.

While using Decision Trees, the classification process is modeled as a series of hierarchical decisions on the features, forming a tree-like structure. In other words, we ask a series of questions about the features of an instance, and based on the answer, we may ask more questions, until we reach to a conclusion about the class label of that instance. Each question, which is the splitting criterion, corresponds to an internal node of the tree, that separates the data into two or more parts. The goal is to get a split that results to as pure subsets (in terms of class labels) as possible, in order to make a confident prediction.

Consider the m-dimensional space that is defined by the feature vectors $\mathbf{x}$, of length m. There, every training instance corresponds to a single point. A Linear SVM looks for a decision boundary between two classes, a hyperplane that bisects the data with the largest possible margin between the two different classes. The margin on each side of the hyperplane is the area with no data points in it. Decision boundaries with large margins are expected to generalize well on previously unseen instances as they are robust to input uncertainty and noise.

Ensemble methods try to increase the prediction accuracy by combining the results from multiple weak classifiers. Assuming that base classifiers are more accurate than random guessing and that they are diverse, i.e., they make different errors on new entries, their combination will result in a more accurate classifier. Random Forest is a class of ensemble methods that uses decision trees as week learners. Randomness has been explicitly inserted in the model building process, as in each decision tree, every splitting criterion considers only a subset of features, randomly selected from the feature vector of **x**, to select the best split. This is repeated for each iteration, till the tree is fully grown, without any pruning. Once we build all the trees, the majority class from the various predictions of a test instance is reported. The strategy of this method is to reduce the correlation among the constructed decision trees so that the strength of the ensemble will improve.

In boosting, a weight is associated with each training instance, and the classifiers are constructed using these weights. In this iterative procedure, the weights are updated according to the classifier performance. Using the same algorithm, classifiers are trained on a weighted training set to focus on hard-to-classify instances. At the end of each iteration, the weights of instances with high misclassification error are relatively increased for future iterations. Each model created is applied to test instances and the final prediction is obtained by aggregating the predictions from the base classifiers. In Gradient Boosting for binary classification, a single regression tree is built, where in each splitting criterion, only a subset of the features is considered. Once the tree is built, then, the corresponding weight of the classifier in the current iteration is estimated.

VI. Experimental Design

A. Training and Testing Methodology

The models constructed are global, i.e., a single model predicts the performance of all students over all the courses. We used a 5-fold cross-validation approach to assess the performance of the different models. The data are partitioned into 5 disjoint subsets. For each fold, we test on one partition and use the remaining ones for training. As randomization takes part in the models while sampling and/or initialization, we run the same model with 3 different seeds and average out the performance achieved. The average of the performance achieved over the 5 folds was used as the performance of this 5-fold cross-validation experiment.

Note that our evaluation methodology can lead to scenarios in which students from the same cohort are used for both training and testing. However, we did not include the test instances during the feature extraction phase of any student that needed information from other students taking the target course. Given this, the protocol still ensures that no information is shared across training and test data, directly or indirectly, because no information was passed from the test to the training set during feature extraction.

Additionally, over the years and during our very long dataset, it does occur some shifting in course characteristics, like course content, difficulty, and popularity. As a result, two versions of a course offered now and before 10 years will not be the same. The current experimental setting does not take that into account, as it uses all the data for feature extraction, and learns a single model for all years. This can be considered by updating our models every semester or year and discarding k-fold validation. We could break the dataset into smaller subsets, that each includes a smaller range of years. As time progresses, we will utilize only a number of the most recent semesters for feature extraction and training. In this way, we can take into consideration these changes.

B. Metrics

Since the classes are unbalanced and the class of interest is the positive class (i.e., the students that fail, withdrawn from a class, etc.), the performance assessment metrics that we used focused on evaluating the performance on the positive class. We used three different metrics to compute the performance, which are the maximum F_1 score for the positive class, the area under the ROC curve, and the precision achieved by assigning a small percentage of test instances to the positive class.

All of the above metrics require that the classifier being evaluated assigns a score to each test instance indicating its prediction strength towards the positive class (i.e., the highest the score, the stronger the prediction towards the positive class is). Such a prediction score is naturally generated by the DT, RF and GB classifiers. For DT and GB, the predicted class probability is the fraction of samples of the same class in a leaf. For Random Forests, the prediction score of an input sample is computed as the mean predicted class probabilities of all the trees in the forest. Generating this for the SVM classifier is less obvious. During training time, we fit the data while computing and keeping track of the probability information. In this way, we can compute the probability of positive class assigned to a sample.

At a threshold θ, instances with prediction scores above θ are assigned to the positive class, otherwise to the negative class. For a given θ, we define $\mathrm{F}_1(\theta)$ as:

$$F_1(\theta) = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{1}$$

where, recall measures the ratio of true positives to all actual positives and precision measures ratio of true positives to all predicted positives. The $\max$ F_1 score is the highest F_1 score achieved for any threshold θ, according to the following steps:

1) Sort the prediction scores in non-increasing order.
2) For each point L in this sorted sequence, compute the F_1 score, using Eq. (1), by assuming that any instances that have a prediction score that is greater than that of the L-th instance are classified as positive and everything else is classified as negative.
3) The max F_1 score is the maximum F_1 value obtained.

The second metric is the area under the receiver operating characteristic (ROC) curve, AUC. The ROC curve plots the true positive rate against the false positive rate, at various thresholds. AUC corresponds to the probability that the classifier will rank a random positive instance higher than a negative one.

We also report precision@$k\%$, which assigns the highest ranked $k\%$ of the test instances to the positive class and then measures their correctly classified fraction. Moreover, if there is a tie among some instances with the lowest prediction score that is considered to be positive, then all these instances are labeled as positive. An advantage of this metric is that it

TABLE IV
PERFORMANCE OF THE VARIOUS CLASSIFIERS WHEN MODEL SELECTION IS BASED ON F_1 SCORES

Area under the ROC curve

Classiffier	Fgr	Wgr	RelF	RelCF	Agr
Random	0.500	0.500	0.501	0.501	0.500
DT	0.834	0.710	0.689	0.716	0.820
SVM	0.853	0.736	0.690	0.718	0.819
RF	0.873	0.778	0.748	0.759	0.850
GB	**0.877**	**0.780**	**0.755**	**0.765**	**0.854**

max F_1 score

Classifier	Fgr	Wgr	RelF	RelCF	Agr
Random	0.034	0.027	0.232	0.222	0.216
DT	0.255	0.123	0.450	0.466	0.573
SVM	0.276	0.171	0.452	0.469	0.570
RF	0.317	0.165	0.499	0.502	0.604
GB	**0.319**	**0.181**	**0.506**	**0.507**	**0.610**

TABLE V
PERFORMANCE OF THE VARIOUS CLASSIFIERS WHEN MODEL SELECTION IS BASED ON PRECISION SCORES

Precision@$k\%$

Classiffier	Fgr	Wgr	RelF	RelCF
DT	0.196	0.086	0.402	0.408
SVM	0.222	0.127	0.402	0.414
RF	**0.261**	0.123	0.468	0.459
GB	0.260	**0.134**	**0.475**	**0.466**

Precision@$k/2\%$

Classifier	Fgr	Wgr	RelF	RelCF
DT	0.249	0.111	0.456	0.451
SVM	0.296	0.178	0.456	0.452
RF	**0.365**	0.170	0.554	0.532
GB	**0.365**	**0.189**	**0.564**	**0.544**

provides an easy to interpret quantity as to the accuracy of the classifier at different values of k.

C. Model Selection

We will present results for two ways of model selection. Firstly, we perform model selection and parameter tuning based on the $\max$ F_1 score, obtained as described above. As an alternative way of evaluation, we utilize the precision@$k\%$ measure. In that case, precision is evaluated at a given threshold, considering as positive only the top $k\%$ of the instances with the highest prediction score.

Since the percentage of positive instances varies across the different classification tasks, the value of k depends on the task we have at hand, to match the expected percentage of positive instances, assuming that the same trend will be followed by the test data as well. In our experiments we report the results for two values of k. The first value of $k\%$ refers to the percentage of instances that are at least as much as the percentage of observed positive instances, i.e., 5%, 5%, 24%, and 24% for Fgr, Wgr, RelF, and RelCF tasks, respectively. In this way, we make sure that a perfect model would be able to capture all the positive instances. We also evaluate precision@$k/2\%$, where $k/2$ covers at least half of the observed percentage of positive instances in each classification problem, i.e., 2%, 2%, 12%, and 12% for Fgr, Wgr, RelF, and RelCF tasks, respectively. With this approach, we hope to more accurately predict a smaller set of instances. We believe that, even though precision at $k\%$ and at $k/2\%$ assess the performance at two discrete points, they provide insights as to how the classification methods and problems discussed in this paper can be used in an operational setting (i.e., deployed at a University).

D. Overlap Between Two Models

In order to compare how much the predictions of two models (i and j) agree, we assume that TP_i and TP_j are their corresponding sets of instances that were correctly identified as positive, with cardinalities $|TP_i|$ and $|TP_j|$. The percentage of overlap between TP_i and TP_j is computed as: $\text{Overlap}(i,j) = 100 \times |TP_i \cap TP_j|/|TP_i \cup TP_j|$. If we take into consideration that $|TP_i| \neq |TP_j|$, we can scale overlap by the maximum possible overlap: $\text{Max Overlap}(i,j) = 100 \times \min(|TP_i|, |TP_j|)/\max(|TP_i|, |TP_j|)$, and obtain the *Scaled Overlap*:

$$\text{Scaled Overlap}(i,j) = \frac{\text{Overlap}(i,j)}{\text{Max Overlap}(i,j)}. \tag{2}$$

Scaled overlap takes values from 0 to 1. When it is 0, there is no overlap. When it is 1, there is as much overlap as possible, i.e., the positive instances of the one model are included in the positive predictions of the other model.

VII. RESULTS

A. Performance Analysis

Table IV shows the performance of the various methods for the classification tasks, in terms of the AUC and the $\max$ F_1 score, when performing parameter tuning based on the $\max$ F_1 score.

Based on both metrics presented, GB is the best performing method, closely followed by the RF classifier. As expected, DT, which is the simplest method, has the lowest performance. These results are better compared to the performance of grade prediction methods for any classification task. When using Course-Specific Regression for predicting the failing students, we get a $\max$ F_1 score of 0.118, which is lower than any of the other methods we discuss.

While comparing the classification tasks, we can see that the tasks that predict relative performance have lower AUC values than when predicting absolute performance. In terms of $\max$ F_1 scores, we can see clearly that the A-students are the most accurately predicted. The F_1 scores of the different tasks are related to the percentage of positive instances in each task. If we consider the performance of a random classifier (which randomly predicts as positive the percentage of data equal to the number of positives), then we can achieve 9.5, 6.5, 2.3 and 2 times improvement for the tasks Fgr, Wgr, RelF and RelCF, respectively. The tasks Fgr and Wgr, that are highly unbalanced, have significantly lower F_1 scores, in a problem that is way more

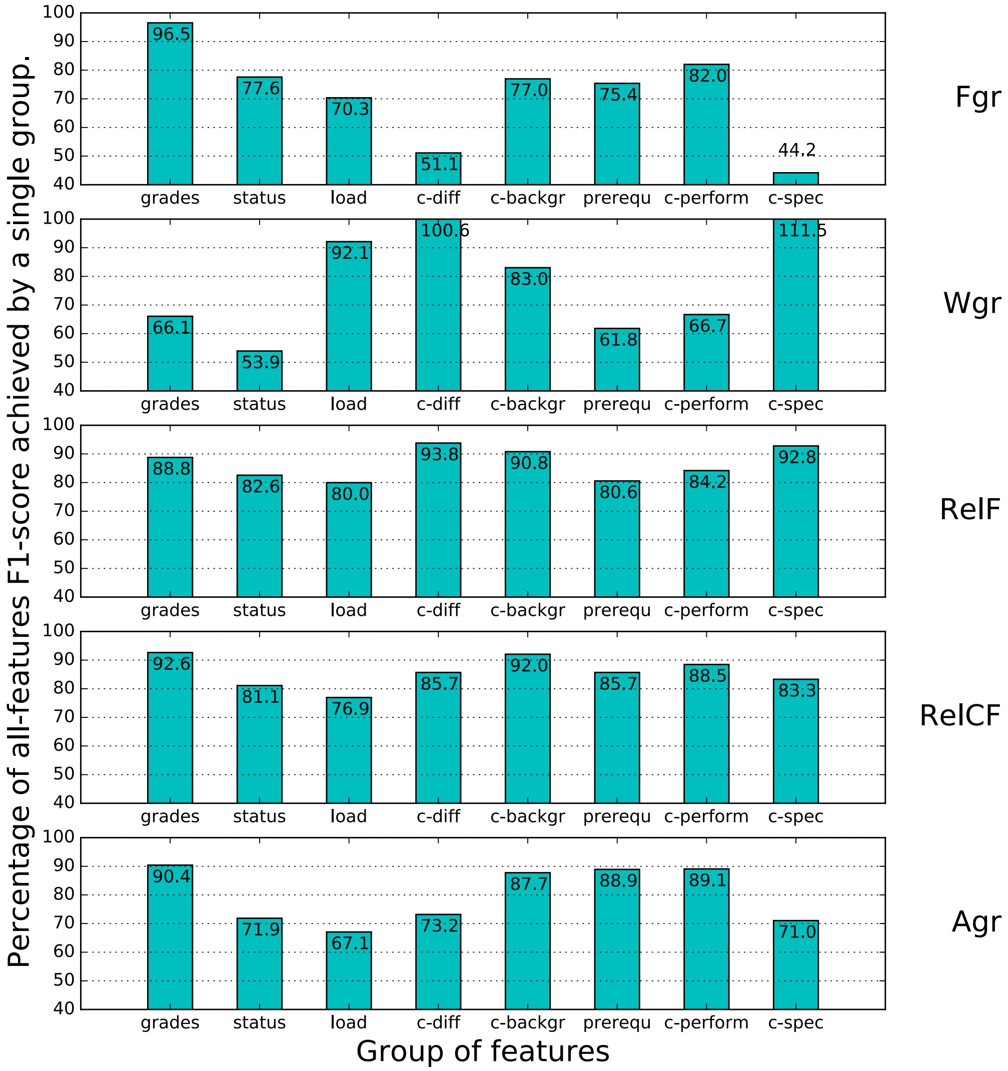


Fig. 2. Percentage of performance managed to recover using only one group of features.

difficult though. Moreover, as there is 81% overlap between the students that are positive for both RelF and RelCF, the tasks of RelF and RelCF have very similar performance.

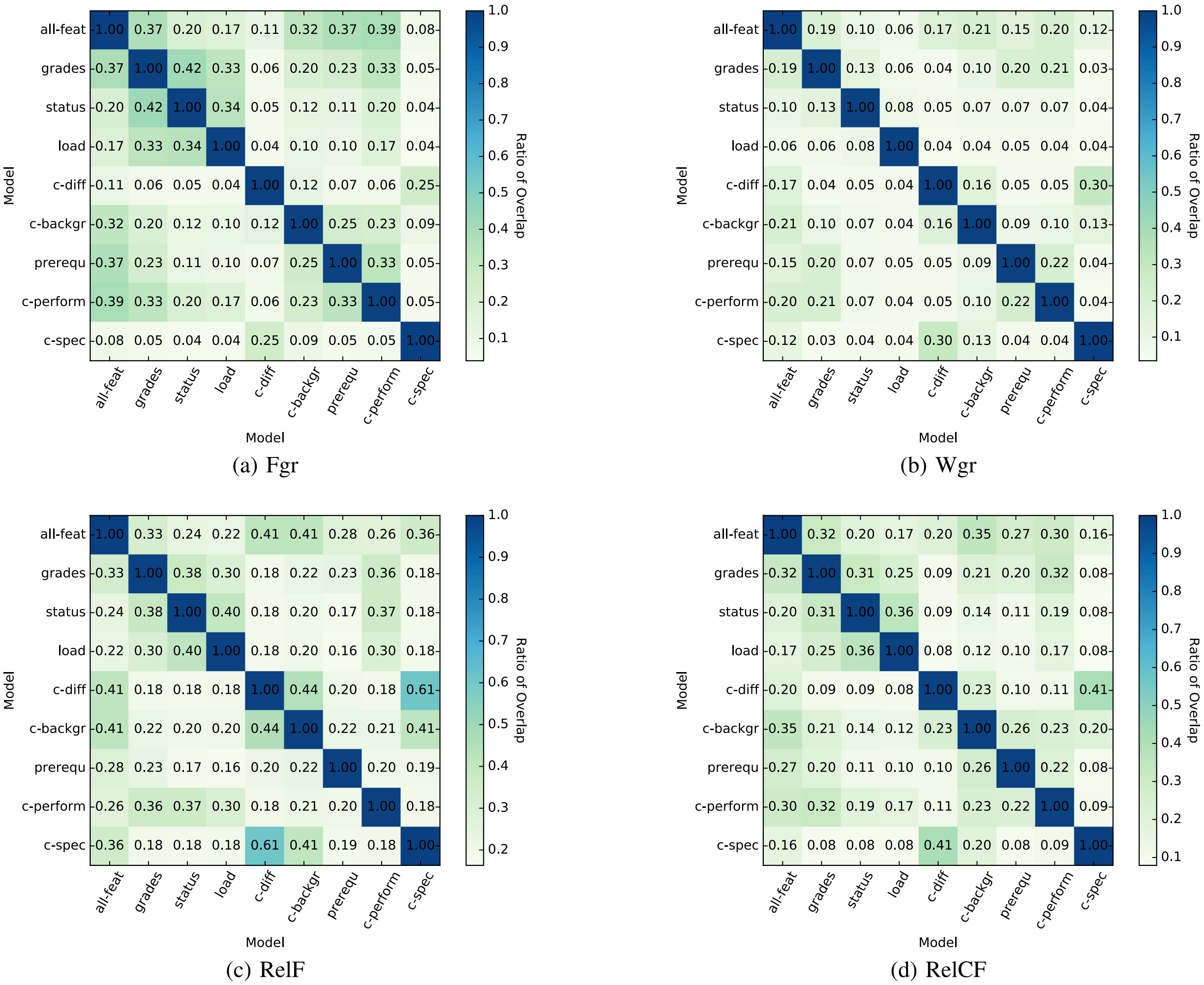
Table V presents the performance achieved in terms of precision, when model selection is based on precision@k% and precision@$k/2$%. Looking at the performance of the different models in terms of precision at k% and $k/2$% we see that GB tends to achieve the best performance, which is consistent with that previous observations.

These precision numbers also provide some insights as to what are the implications of actually using them to flag the students that may need to reconsider taking a particular course and potentially notifying them and/or their academic advisors. In general, the precision improves when we only consider the highest scoring predictions. If we look at the precision@k%, we can see that for the Fgr class, the models will be able to correctly flag 26% of the students, for the Wgr class precision is at 13.4%, whereas for RelF and RelCF, this number increases to around 46%. The corresponding numbers for the $k/2$% results are 35.5%, 19% and around 55%. As a result, when we use precision@$k/2$%, we also get fewer false positive instances.

Using the top k or $k/2$ percent of predictions (or even a smaller percentage) will be an academic unit's specific decision and needs to balance the costs associated with failing to warn a student about a course that he/she will potentially fail or not perform well in, versus having a high false positive rate. The latter case can either increase the workload of academic advisors (if they need to decide if the student will be warned or not) or increase the false positive rate to the point that students end up ignoring any warnings that were directly sent to them.

B. *Feature Importance Study*

In the problem that we examine, we do not know which are the most relevant influencing factors. In order to better represent the students, extracted features are introduced. Some of them may be irrelevant to the problem or repetitive. Our goals is to study which factors are important indicators of a student's performance. We performed the following experiment, that uses a wrapper method to evaluate the quality of the features [23]. We select a subset of features and use that subset to build a predetermined classifier. Evaluate the selected subset by the



(a) Fgr

(b) Wgr

(c) RelF

(d) RelCF

Fig. 3. Scaled Overlap of the models that use a single feature group. The two columns correspond to the experiments that are based on precision@k%.

performance of the classifier, and repeat as much as possible in order achieve some desired quality.

Since the number of candidate features is quite high, the exhaustive search of all possible subsets is impractical. As a result, we shrank the search space by defining the subsets ourselves. We categorize each extracted feature to one of the 8 groups, according to Table I. Afterwards, for each classification task, we built RF classifiers using only the features belonging to one of the above groups. We selected to use RF over GB, as they achieve similar performance in less training time. Model selection is performed based on the F_1 score.

The accuracy achieved for a model using a single group of features is expected to be less than the accuracy when using all the features. The percentage of accuracy that a model, using only the features belonging to one group, manages to achieve, in terms of the $\max F_1$ score, are presented on Fig. 2. In this bar chart, we can see the percentage of accuracy achieved from all the different feature groups for all the discussed classification tasks. The higher the percentage achieved by a single group of features, the more predictive ability these features have.

From this figure, we can get many insights on the factors that affect student performance. For example, the features related to the students' grades have a very good predictive capability in almost all the tasks, except the task of predicting the W grades. In this task, features related with the course's difficulty and popularity as well as features that are course-specific, manage to achieve the same accuracy as when using all the features. This indicates that the reasons that a student drops a course are related more to the course, rather than to the students themselves. The next best indicator is the feature group about the student's course load during the semester.

On the other hand, this is not the case for predicting the failing students, in the absolute sense, i.e., receive a D or F. When using only course-related groups (c-diff, c-spec) for predicting the students likely to fail a course (Fgr task), we manage to recover half or less from the $\max F_1$ score. As a result, these factors do not influence the absolute failing performance of a student, indicating that the reasons for that are mostly related with the student. As the students' grades manage to recover almost the same performance as when using all the features, they are the ones that affect the Fgr prediction the most. When using the other groups, it is very difficult to achieve comparable performance, as they recover 80% or less of the $\max F_1$ score.

The feature groups are behaving similarly for RelF and RelCF. However, we notice that for the RelCF task, the feature

TABLE VI
PRECISION OF THE THREE BEST PERFORMING COMBINATIONS FOR EACH TASK

	Precision@k%						Precision@$k/2$%					
Fgr	Best rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	grades	0.261	0.249	↑0.267	✳2.39	all-feat	grades	0.365	0.363	**0.382**	✳4.61
	all-feat	status	0.261	0.197	0.252	-3.48	all-feat	status	0.365	0.303	0.359	-1.64
	grades	c-backgr	0.249	0.197	0.250	0.35	grades	c-backgr	0.363	0.264	0.357	-1.64
	Average rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	grades	0.261	0.249	**0.269**	✳2.83	all-feat	grades	0.365	0.363	↑0.380	✳4.13
	all-feat	status	0.261	0.197	0.251	-3.88	all-feat	status	0.365	0.303	↑0.371	1.47
	all-feat	c-perform	0.261	0.210	0.246	-5.76	all-feat	load	0.365	0.276	0.357	-2.21
Wgr	Best rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	c-spec	0.123	0.134	**0.150**	✺12.48	load	c-spec	0.167	0.205	**0.262**	✺27.65
	load	c-spec	0.097	0.134	↑0.149	✺11.59	all-feat	c-spec	0.170	0.205	0.240	✺17.00
	load	c-diff	0.097	0.122	↑0.142	✺16.72	load	c-diff	0.167	0.174	0.233	✺34.22
	Average rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	c-spec	0.123	0.134	↑0.148	✺10.58	c-diff	c-spec	0.174	0.205	0.190	-7.29
	c-diff	c-spec	0.122	0.134	0.136	1.60	all-feat	c-spec	0.170	0.205	0.188	-8.32
	all-feat	c-diff	0.123	0.122	0.130	✳6.43	all-feat	load	0.170	0.167	0.187	✳9.62
RelF	Best rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	c-spec	0.468	0.422	↑0.458	-2.04	all-feat	c-diff	0.554	0.497	↑0.535	-3.40
	all-feat	c-diff	0.468	0.427	↑0.456	-2.43	all-feat	c-spec	0.554	0.493	0.532	-3.91
	all-feat	c-backgr	0.468	0.412	0.449	-3.97	all-feat	grades	0.554	0.455	0.529	-4.52
	Average rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	c-spec	0.468	0.422	**0.474**	✳1.32	grades	c-diff	0.455	0.497	**0.563**	✺13.22
	grades	c-diff	0.395	0.427	↑0.472	✺10.56	all-feat	c-spec	0.554	0.493	↑0.562	✳1.44
	all-feat	c-diff	0.468	0.427	↑0.470	0.44	all-feat	grades	0.554	0.455	↑0.562	✺1.47
RelCF	Best rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	all-feat	c-backgr	0.459	0.416	↑0.447	-2.71	all-feat	grades	0.532	0.477	0.514	-3.29
	all-feat	grades	0.459	0.413	↑0.444	-3.36	all-feat	c-backgr	0.532	0.476	0.513	-3.44
	all-feat	c-perform	0.459	0.383	0.432	-5.81	all-feat	status	0.532	0.425	0.502	-5.68
	Average rank											
	model1	model2	prec1	prec2	prec1&2	%impr	model1	model2	prec1	prec2	prec1&2	%impr
	grades	c-diff	0.413	0.379	**0.467**	✺13.07	grades	c-diff	0.477	0.421	**0.544**	✺13.99
	all-feat	c-spec	0.459	0.360	↑0.464	✳1.09	all-feat	c-spec	0.532	0.402	↑0.542	✳2.02
	grades	c-spec	0.413	0.360	↑0.462	✺11.86	all-feat	c-diff	0.532	0.421	↑0.542	✳1.97

For every classification task and precision@k% or precision@$k/2$%, the number in bold is the best combined performance achieved.
Model1 and model2 refer to the two initial models, with the corresponding precisions, prec1 and prec2. The combined performance is perc1&2. The last column, %impr, is the percentage of improvement of the combined model over the best performing model, that uses a single feature group.
For each of the tasks, the numbers marked with ↑ show the models that perform well, whose performance is within 5% of the best combined precision in bold.
In the column with %impr, the symbols ✳ and ✺ are used to show that the performance improvement of the combined model is significant based on t-test, with significance levels 10^{-1} and 10^{-3} respectively.

groups that are related with student-course specific features have slightly better performance, while the student-specific groups have slightly worst performance, compared to the task of RelF. This is happening because, for RelCF, we take into consideration how other students usually perform on the target course. Every single group has enough information for the RF to utilize to achieve performance which is as good as 75% of the best case, i.e., when using all the features.

Finally, for identifying the A-students, the feature groups 1, 5, 6, 7 (grades, c-backgr, prerequ, c-perform) are the ones that manage to have the best performance. These groups are related with students' grades in general, but also, with their grades relative to the target course's background, prerequisites and level. Using only one of them can provide us with the information we need in order to recover around 90% of the performance while using all the features.

C. Combining Predictions of Different Models

Many of the models shown in Fig. 2 tend to achieve similar performance for the classification tasks. To see if that performance similarity is due to the fact that all of them identify the same set of students or not, we compare how similar are the positive top predictions of these models. We compare every pair of models in Section VII-B, and the best performing RF model with all the features. Specifically, we analyze the top k and $k/2\%$ of their predictions and use the scaled overlap metric (Eq. (2)) to compute the overlap between the correctly predictive positive classes of these models (Fig. 3). Every (i, j) element in the heatmap is the average scaled overlap when combining models i and j, over the different seeds and folds. The heatmaps for precision@$k/2\%$ have similar patterns, but smaller overlap values. Surprisingly, for many of these models, the overlap between the best performing models is relatively low, suggesting that these models actually identify different subset of problematic student-course instances.

These results suggest that the performance of the classification methods can potentially be improved by combining the predictions of these different models. To test this hypothesis, for each model we rank the instances with respect to how likely it is they belong to the positive class. We use two heuristic strategies [34] to combine these rankings based on their initial rankings. Based on the first approach, an item's final rank is the item's best rank. It is equivalent with a Round Robin approach, where the item with the best rank is added to the final ranking in turn. The second strategy is a median rank approximation. An item's final rank is the item's average rank. In the end, we assign the positive label to the first $k\%$ or $k/2\%$ of the instances with the best ranking for each classification task.

The three best performing combinations for each classification problem and choice of $k\%$ and $k/2\%$ are shown in Table VI. In all (or nearly all the cases), by combining the predictions of different models, we get better results than that of the initial best performing model. When using precision@$k\%$, we get a percentage of improvement for all tasks between 1.32% up to 13.07%. For precision@$k/2\%$, we are able to improve accuracy starting from 4.61%, up to 27.65% for the students that are at risk of dropping a course. There are other additional pairs lower in the list, with lower combined precision, that nevertheless, managed to significantly improve the performance of a single model.

The above finding shows us that, even if some models have lower precision, they can provide us with some insights that were not apparent in other better-performing models. For example, the best RF model including all the features, noted as all-feat, appears quite often in Table VI. It is possible to improve its accuracy if it gets combined with another model that uses only a subset of the features. This might be happening because of the mismatch between the evaluation metric (precision), and the criterion used while model estimation. The function to build the decision trees and measure the quality of a split is either Gini impurity or information gain (tree-specific). Both criteria are not optimized for precision@$k\%$. As a result, when emphasizing a specific subset of the features by selecting it as the second base model, we get better precision values.

In order to successfully merge the predictions of two models, we need to decide on the right technique which we will use, as this selection is critical for the final result. The best-rank strategy is the best one for merging models for the classification problems Fgr and Wgr, that refer to absolute performance. For predicting Fgr with precision@$k\%$, the average rank strategy results in a slightly better precision. On the other hand, when using the average rank and taking into consideration the ranking of both models, the relative tasks of RelF and RelCF get a considerable boost in their performance.

We also performed the same procedure for the four methods using all the features, but there was no significant improvement. The highest improvement was almost 1%, when combining the predictions of the two best performing models, Random Forest and Gradient Boosting classifiers.

VIII. Conclusion

The purpose of this paper is to accurately identify students that are at risk before they even take a class. These students might fail the class, drop it, or perform worse than they usually do. We can notify the student's instructor, advisor and department to take the appropriate steps to assist that particular student before the student registers for a course for which he/she seems to fail or drop later, after spending valuable time and effort. We successfully extracted features from historical grading data, in order to test different simple and sophisticated classification methods. By performing our feature importance study and creating meaningful feature groups, we also got interesting findings that can explain student performance in a concise way. For example, we found that, while Fgr and Wgr tasks seem similar, the importance of the feature groups are very different. Additionally, for the RelF and RelCF tasks, every single group manages to hold enough information for accurate predictions. When combining the predictions from different models, we can further improve the performance achieved by a single model, since different feature groups capture different characteristics.

Acknowledgment

Access to research and computing facilities was provided by the Digital Technology Center at the University of Minnesota and the Minnesota Supercomputing Institute.

References

[1] J. McFarland *et al.*, "Undergraduate retention and graduation rates," in *Proc. Condition Educ. Nat. Center Edu. Statist.*, 2017, pp. 268–271.

[2] A. Polyzou and G. Karypis, "Grade prediction with models specific to students and courses," *Int. J. Data Sci. Analytics*, vol. 2, no. 3/4, pp. 159–171, 2016.

[3] S. Morsy and G. Karypis, "Cumulative knowledge-based regression models for next-term grade prediction," in *Proc. SIAM Int. Conf. Data Mining*, 2017, pp. 552–560.

[4] M. Sweeney, J. Lester, and H. Rangwala, "Next-term student grade prediction," in *Proc. IEEE Int. Conf. Big Data*, 2015, pp. 970–975.

[5] J. E. Knowles, "Of needles and haystacks: Building an accurate statewide dropout early warning system in wisconsin," *J. Educational Data Mining*, vol. 7, no. 3, pp. 18–67, 2015.

[6] S. Kotsiantis, C. Pierrakeas, and P. Pintelas, "Predicting students' performance in distance learning using machine learning techniques," *Appl. Artif. Intell.*, vol. 18, no. 5, pp. 411–426, 2004.

[7] A. A. Saa, "Educational data mining & students' performance prediction," *Int. J. Adv. Comput. Sci. Appl.*, vol. 1, pp. 212–220, 2016.

[8] C. Romero, S. Ventura, P. G. Espejo, and C. Hervás, "Data mining algorithms to classify students," in *Proc. 1st Int. Conf. Educational Data Mining*, 2008, pp. 8–17.

[9] A. Pardo, N. Mirriahi, R. Martinez-Maldonado, J. Jovanovic, S. Dawson, and D. Gašević, "Generating actionable predictive models of academic performance," in *Proc. 6th Int. Conf. Learn. Analytics Knowl.*, 2016, pp. 474–478.

[10] E. Osmanbegović and M. Suljić, "Data mining approach for predicting student performance," *Econ. Rev.*, vol. 10, no. 1, pp. 3–12, 2012.

[11] B. Minaei-Bidgoli, D. A. Kashy, G. Kortemeyer, and W. F. Punch, "Predicting student performance: An application of data mining methods with an educational web-based system," in *Proc. 33rd Annu. Frontiers Educ.*, 2003, vol. 1, pp. T2A–13.

[12] L. V. Morris, C. Finnegan, and S.-S. Wu, "Tracking student behavior, persistence, and achievement in online courses," *Internet Higher Educ.*, vol. 8, no. 3, pp. 221–231, 2005.

[13] D. Gašević, S. Dawson, T. Rogers, and D. Gasevic, "Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success," *Internet Higher Educ.*, vol. 28, pp. 68–84, 2016.

[14] R. Conijn, C. Snijders, A. Kleingeld, and U. Matzat, "Predicting student performance from LMS data: A comparison of 17 blended courses using moodle LMS," *IEEE Trans. Learn. Technologies*, vol. 10, no. 1, pp. 17–29, Jan.-Mar. 2017.

[15] D. T. Tempelaar, B. Rienties, and B. Giesbers, "In search for the most informative data for feedback generation: Learning analytics in a data-rich context," *Comput. Human Behav.*, vol. 47, pp. 157–167, 2015.

[16] N. Z. Zacharis, "A multivariate approach to predicting student outcomes in web-enabled blended learning courses," *Internet Higher Educ.*, vol. 27, pp. 44–53, 2015.

[17] R. Campagni, D. Merlini, and M. Verri, "The influence of first year behaviour in the progressions of university students," in *Proc. Int. Conf. Comput. Supported Educ.*, 2017, pp. 343–362.

[18] O. Oyelade, O. Oladipupo, and I. Obagbuwa, "Application of k-means clustering algorithm for prediction of students' academic performance," *Int. J. Comput. Sci. Inf. Secur.*, vol. 7, no. 1, pp. 292–295, 2010.

[19] D. Kerr and G. K. Chung, "Identifying key features of student performance in educational video games and simulations through cluster analysis," *J. Educational Data Mining*, vol. 4, no. 1, pp. 144–182, 2012.

[20] Z. Ren, X. Ning, and H. Rangwala, "Grade prediction with temporal course-wise influence," in *Proc. 10th Int. Conf. Educational Data Mining*, 2017, pp. 48–55.

[21] M. Sweeney, J. Lester, H. Rangwala, and A. Johri, "Next-term student performance prediction: A recommender systems approach," *J. Educational Data Mining*, vol. 8, no. 1, pp. 22–51, 2016.

[22] Z. Iqbal, J. Qadir, A. N. Mian, and F. Kamiran, "Machine learning based student grade prediction: A case study," 2017, *arXiv: 1708.08744*.

[23] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," in *Data Classification: Algorithms and Applications*, C. C. Aggarwal, Ed. Boca Raton, FL, USA: CRC Press, 2014, ch. 2, pp. 37–64.

[24] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.

[25] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Mach. Learn.*, vol. 53, no. 1/2, pp. 23–69, 2003.

[26] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artif. Intell.*, vol. 97, no. 1/2, pp. 273–324, 1997.

[27] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 4, pp. 491–502, Apr. 2005.

[28] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning (Series in Statistics)* vol. 1, no. 10. Berlin, Germany: Springer, 2001.

[29] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[30] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Boca Raton, FL, USA: CRC Press, 1984.

[31] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[32] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[33] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.

[34] R. Fagin, R. Kumar, and D. Sivakumar, "Efficient similarity search and classification via rank aggregation," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2003, pp. 301–312.

Agoritsa Polyzou received the Diploma degree in computer engineering and informatics from the University of Patras, Patras, Greece, in 2013. She is currently working toward the Ph.D. degree with the Department of Computer Science and Engineering, University of Minnesota, Twin Cities, Minneapolis, MN, USA. Her research interests include data mining, learning analytics, recommender systems, and the application of data-mining techniques within educational contexts.

George Karypis is a Distinguished McKnight University Professor and an ADC Chair of Digital Technology with the Department of Computer Science and Engineering, University of Minnesota, Twin Cities, Minneapolis, MN, USA. His research interests span the areas of data mining, high performance computing, information retrieval, collaborative filtering, bioinformatics, cheminformatics, and scientific computing. His research has resulted in the development of software libraries for serial and parallel graph partitioning (METIS and ParMETIS), hypergraph partitioning (hMETIS), for parallel Cholesky factorization (PSPASES), for collaborative filtering-based recommendation algorithms (SUGGEST), clustering high dimensional datasets (CLUTO), finding frequent patterns in diverse datasets (PAFI), and for protein secondary structure prediction (YASSPP). He has coauthored more than 280 papers on these topics and two books (*Introduction to Protein Structure Prediction: Methods and Algorithms* (Wiley, 2010) and *Introduction to Parallel Computing* (*Publisher Addison Wesley*, 2003, 2nd edition)). In addition, he is serving on the program committees of many conferences and workshops on these topics, and on the editorial boards of the IEEE Transactions on Knowledge and Data Engineering, *ACM Transactions on Knowledge Discovery from Data*, *Data Mining and Knowledge Discovery*, *Social Network Analysis and Data Mining Journal*, *International Journal of Data Mining and Bioinformatics*, *Journal on Current Proteomics*, and *Advances in Bioinformatics*, *Biomedicine and Biotechnology*. He is a Fellow of the IEEE.