Koutcheme, Charles; Sarsa, Sami; Hellas, Arto; Haaranen, Lassi; Leinonen, Juho

Methodological Considerations for Predicting At-risk Students

Published in:
ACE '22: Australasian Computing Education Conference

DOI:
10.1145/3511861.3511873

Published: 14/02/2022

Document Version
Peer reviewed version

Please cite the original version:
Koutcheme, C., Sarsa, S., Hellas, A., Haaranen, L., & Leinonen, J. (2022). Methodological Considerations for Predicting At-risk Students. In *ACE '22: Australasian Computing Education Conference* (pp. 105-113). ACM. https://doi.org/10.1145/3511861.3511873

Methodological Considerations for Predicting At-risk Students

Charles Koutcheme
Aalto University
Espoo, Finland
charles.koutcheme@aalto.fi

Sami Sarsa
Aalto University
Espoo, Finland
sami.sarsa@aalto.fi

Arto Hellas
Aalto University
Espoo, Finland
arto.hellas@aalto.fi

Lassi Haaranen
Aalto University
Espoo, Finland
lassi.haaranen@aalto.fi

Juho Leinonen
Aalto University
Espoo, Finland
juho.2.leinonen@aalto.fi

ABSTRACT

Educational researchers have long sought to increase student retention. One stream of research focusing on this seeks to automatically identify students who are at risk of dropping out. Studies tend to agree that earlier identification of at-risk students is better, providing more room for targeted interventions. We looked at the interplay of data and predictive power of machine learning models used to identify at-risk students. We critically examine the often used approach where data collected from weeks *1, 2, ..., n* is used to predict whether a student becomes inactive in the subsequent weeks w, $w \geq n+1$, pointing out issues with this approach that may inflate models' predictive power. Specifically, our empirical analysis highlights that including students who have become inactive on week n or before, where $n > 1$, to the data used to identify students who are inactive on the following weeks is a significant cause of bias. Including students who dropped out during the first week makes the problem significantly easier, since they have no data in the subsequent weeks. Based on our results, we recommend including only active students until week n when building and evaluating models for predicting dropouts in subsequent weeks and evaluating and reporting the particularities of the respective course contexts.

CCS CONCEPTS

• **Social and professional topics** → *Computing education.*

KEYWORDS

predicting performance, at-risk students, prediction, learning analytics, educational data mining

ACM Reference Format:
Charles Koutcheme, Sami Sarsa, Arto Hellas, Lassi Haaranen, and Juho Leinonen. 2022. Methodological Considerations for Predicting At-risk Students. In *Australasian Computing Education Conference (ACE '22), February 14–18, 2022, Virtual Event, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3511861.3511873

ACE '22, February 14–18, 2022, Virtual Event, Australia

ACM ISBN 978-1-4503-9643-1/22/02…$15.00
https://doi.org/10.1145/3511861.3511873

1 INTRODUCTION

Predicting academic performance of a student is a relatively popular research topic [20, 29, 45, 54]. Within computing education research, such studies use a variety of data that includes demographic factors [12, 16], surveys and questionnaires [16, 58, 61], data collected from course activities such as clicker data from lectures [25, 42, 52], log data from course materials [38], and submission data or log data from course assignments [1, 6, 9, 24, 33, 36, 58]. One common aspect in early work in the topic is including all the available data when building predictive models – this has been the norm for demographic data, collected survey and background data, and other type of data that is relatively static and does not by itself indicate activity or lack of it.

However, with the recent increase in collecting and using log data from courses, e.g. different types of snapshot and submission data from programming courses [1, 6, 9, 24, 33, 36, 58], using all the available data when building predictive models can lead to bias when informed decisions on what data is included or excluded are not made. As an example, building models for identifying students who have dropped out at the final week of a course using log data from all but the final week of the course can lead to accurate but uninformative models. The accuracy and lack of valuable information both stem from the same root cause: many students have already dropped out at the final week and thus the log data shows little to no activity for them on e.g. the week prior to the final week of a course, which is a relatively good indicator of a dropout. Another issue with such use of data is that it may mask away other more valuable information, such as information about assignments that students struggle with. For examples of studies that have used data in this way, see e.g. [3, 38, 44].

The above example highlights the methodological issue that we discuss in this article. Our overarching theme is exploring *to what extent including students who have likely already dropped out into the data used to train predictive models inflate the predictive performance of those models.* To achieve this, we study four datasets from three courses, and compare the performance of two approaches for building models for predicting who are likely to drop out. The first approach includes all past data (including already dropped out students), while the second approach includes only data from students who have been active up to the current point. While there exists a wide variety of works focusing on predicting at risk students [20], the closest matches to our work are articles addressing methodological aspects of such work (e.g. [60]).

This article is organized as follows. Next, in Section 2, we outline related work in predicting course dropouts, followed by Section 3 outlining our study methodology, including the context, data, and research questions. Section 4 presents our results, which are discussed in Section 5. Finally, in Section 6, we summarize our findings and outline future research directions.

2 BACKGROUND

The ability to identify students who are at risk of dropping out of studies [53], or simply under-performing [20], provides institutions and instructors with an opportunity to act to remedy the issue. Indeed, significant efforts have been invested into building early warning systems that analyze collected data and highlight students who might benefit from an intervention [4, 28, 43]. These interventions can be manual, e.g. providing counseling or coaching [7], but they can also be automated, e.g. the use of visualizations [23] or other automated feedback mechanisms [13]. As an example, even feedback on projected grade may change students' behavior [5].

The key issue in these systems is the functionality that is used to identify students who are performing poorly in their studies. Due to the complexity of the topic, there exist several intertwining research streams; some being very domain-specific such as identifying students' performance using log data collected from a programming environment [19, 21, 63], down to the level of utilizing keystrokes students take while programming [14, 32, 36], while other research streams may be domain-generic such as using past grades, completed courses or consistency of work to predict future performance [2, 31, 59]. Yet, even with the ability of identify students in need of assistance, it might be that the interventions themselves do not work [51], or only work for a subset of the students [23].

As with all predictions, like predicting weather or predicting the outcome of a cricket match, prediction results need to come sufficiently early, as having a system warn about upcoming rain is not useful if one already looks like a drowned rat. In the same vein, an early warning system that highlights already dropped out students is unlikely to be useful. The notion of *earliness* has been highlighted in research into predicting academic performance; although earliness between studies varies, it often indicates predicting academic performance using data from only a part of the course such as the first course weeks (see e.g. [1, 10, 17, 41]). However, approaches for predicting academic performance are often evaluated over multiple datasets, where each dataset has data from a particular period from a course or a set of courses (see e.g. [10, 30, 37, 57]).

Herein also lies the possibility for a methodological misstep that our present article discusses. Knowing that many students who drop out of courses drop out early, it is possible that – if not enough care is taken when choosing what data to use – data used to train and evaluate predictive models also includes data of students who have already dropped out. As an example, a dataset that is used to create a model for predicting who will drop out during the fourth week of a course might – if not carefully constructed – include students who have already dropped out after the first week, and thus no longer are active during the second and third week. This leads to a situation where – during training and evaluation of predictive models – the predictive models have information that they would not have in a realistic situation, or at least that would not be useful to those in the end using such models. If the models are not taken into use there is little harm however; including students who might already have become inactive likely will just inflate the performance of the models (see e.g. [3, 38, 44]).

Note that although our focus here is on weeks, the same phenomenon could be also observed within weeks. That is, if a student drops out in the middle of the week, completing just a few of the assignments, then the subsequent assignments that are left undone could be a good indicator of students' performance. This has been highlighted, e.g., in the analysis by Ahadi et al [1], who noted that the last assignments of a week were good predictors of future performance, although to their merit they also analyzed what students did in those particular assignments. Similarly, the phenomenon exists also on a higher abstraction level – as an example, one could seek to build a model that predicts dropouts from a degree program using e.g. enrollment data that would effectively already encode students' engagement.

3 METHODOLOGY

3.1 Context and Data

For the purposes of our study, we obtained log data from three courses held at a university in a Nordic country. All of the courses had different instructors. We refer to these courses as CS0-Web, CS1, and WD. CS0-Web & CS1 had deadlines after each round while WD only had a single deadline at the end of the course. The courses and their context are as follows.

CS0-Web is a 3 ECTS course intended for life-long learners to learn the basics of programming and web development with JavaScript. The course is taken by approximately 100 students annually and is realized as an interactive ebook with embedded exercises that are automatically graded. The course is focused on learning basics of HTML, CSS, and JavaScript as well as learn to manipulate DOM[1] with JavaScript. The latter half of the course focuses on simple server-side development.

CS1 is a 5 ECTS fully online[2] introductory programming course which teaches programming fundamentals, using the Scala language, and emphasizes both the object-oriented and functional paradigms. It is taken by approximately one thousand students annually, most of whom are from the university, though some life-long learners also take it. The course is given through an open online interactive textbook which integrates both the activities (assignments, exercises, multiple-choice questionnaires) and the theoretical side. Each week, students have to solve and submit a set of assignments integrated within the textbook.

WD is a 5 ECTS university course typically taken during the second year of studies. It is taken by a few hundred students annually, and it introduces students to the lifecycle of web applications, ranging from conceptualization and building of web applications to security testing and maintenance. Topics in the course are approached in a hands-on manner, where students learn to develop and test web applications using JavaScript, as well as to deploy, maintain, and monitor them in cloud environments.

[1] Document Object Model
[2] Due to the COVID-19 pandemic.

Table 1: Data summary.

Course	Students	Data
CS0-Web	75	6,382 code submissions
CS1-S	1,056	628,363 code submissions
CS1-B	1,407	Browsing activity
WD	175	60,195 code submissions

The course contents are usually offered in rounds, where the duration of a round is typically a week. During a round, each course introduces a specific set of related concepts and students solve a given number of assignments related to these concepts. For instance, in the CS1 and WD course, a round corresponds to a week. In CS0-Web rounds 1 & 8 last a week, but rounds 2 to 7 last two weeks. In the rest of the article, we will refer to the course rounds to abstract and unify the granularity in which the courses are conducted.

For the present analysis, the data includes code submissions (for CS0-Web, CS1, and WD) collected from their learning management systems and browsing behavior (for CS1) collected using a JavaScript plugin similar to the one described in [27, 38, 39]. For CS1, we use two separate datasets with separate anonymized student identifiers; thus, the course is described as two separate datasets from hereafter. The data used in this analysis is summarized in Table 1.

3.2 Research Questions and Approach

We adopted the same approach for all courses studied. Our research questions for this study are the following:

RQ1 What performance differences do we observe in predictive models when we include versus exclude data from students who have already dropped out?

RQ2 What features are most predictive of students dropping out when including versus excluding data from students who have already dropped out?

RQ3 How does the course context influence the performance of the predictive models?

There is no general consensus on when or how a student should be considered as having dropped out, which can even be affected by institutional policies [55]. We defined dropping out using student activity. We say that a student s has dropped out at round y if the student s is inactive at that round y and all the following rounds. Our definition of activity varies depending on the type of data collected. For submission data, we say that a student is inactive at a given round y if the student did not submit any assignments during that round. For browsing data, we say that a student is inactive at a given round y if the student has no browsing activity during that round (i.e., the student did not access/interact with the materials).

The *including approach* seeks to identify students who dropped out by a given round y using data from all students from all previous rounds. Studies using the including approach tend to include students in the data who have become inactive in the rounds preceding the target round. In *excluding approach*, when identifying students who are likely to drop out by a given round y, only students who did not drop out during the preceding rounds are included (i.e. students who already have dropped out are excluded). Compared to the including approach, the data from already dropped out students is not included, which creates a more realistic picture of the performance of the models.

To answer RQ1, we compare the prediction performance of the including approach and the excluding approach when predicting which students are likely to drop out. To answer RQ2, we analyze the features from the data that both of the approaches use for the predictive models, and to what extent they differ. Finally, to answer RQ3, we draw insights from the models' features and performance, and relate that with the differences in the courses and data.

In the analyses, where applicable, for the including approach, we predict which students are at risk of dropping out by each round of the courses, from round 2 to round 7[3], using iterative subsets of data from the previous rounds (1, 1-2, etc.). For predicting e.g. dropouts by round 7, we iteratively use data from: round 1, rounds 1-2, rounds 1-3, rounds 1-4, rounds 1-5, and rounds 1-6. In each subtask, we include all the student population. In our excluding approach, the task is the same, but we adjust the data to only include the students who did not dropout in one of the rounds beforehand.

3.3 Features Used

For the prediction tasks, we collected the following features:

Submission data We include the following features for each assignment per round:
- Attempt count $\in \mathbb{N}$: the number of times the student attempted the assignment
- Correctness $\in [0, 1]$: the percentage of maximum points student received for the assignment

Browsing data We include the following features for each round:
- Study session count $\in \mathbb{N}$: The number of study sessions
- Study session length $\in \mathbb{R}$: The average length of a study session (in minutes)
- Total time $\in \mathbb{R}$: The total time spent online (in minutes)
- Per page time $\in \mathbb{R}$: Time spent reading each chapter of the material pages corresponding to the given round

Overall, the features that we used for submission data are commonly used in the literature [22]. We opted for assignment specific features because for the courses selected, the assignments given do not have the same difficulty, and as such, we would lose information if we aggregated features over the rounds. Concerning browsing data, we computed per round study sessions statistics, but, in order to have features with close level of granularity across datasets, we also computed the time students spent reading on each page of the rounds included as data. This time spent per page approximates very roughly the time each student spent on (sets) of assignments – i.e. time-on-task, which is known to be a crucial part in learning [15, 18, 34, 35].

3.4 Models

Following the methodology employed by studies predicting students at risk, we used machine learning models for our task of predicting dropouts. As our machine learning models, we employ

[3]Round 1 is excluded, as there are no previous rounds to use as input data, and early rounds have a high dropout rate due to students sampling courses.

Random Forest classifier and Logistic Regression, which have also been previously used in the literature [20]. These models are both widely used in machine learning, often performant and are also interpretable in the sense that feature importance can be extracted for the trained models.

In addition to the two machine learning models, we use two baseline models to show the effectiveness of the machine learning models, i.e. that the models are able to generate useful non-trivial predictions. Our first baseline model is Majority Vote aka Zero Rate (ZeroR) model, which is a model that predicts all values as the most common prediction value in given data. As such it has no predictive value. As our second baseline model, we use a Naive Bayes classifier, which operates on the (unrealistic) assumption that each feature used to predict a class is independent of each other.

Table 2: Statistics of student activity and inactivity for each round in our data. Prefix R in the header indicates Round. Returning students are students who have been active previously and have since been inactive until the given round.

Dataset		R1	R2	R3	R4	R5	R6	R7
CS0-Web	All	73	73	73	73	73	73	73
	Active	73	68	68	48	36	33	26
	Inactive	0	5	5	25	37	40	47
	Returning	0	0	1	0	0	1	0
CS1-B	All	1196	1262	1305	1327	1360	1378	1407
	Active	1196	966	896	860	849	823	815
	Inactive	0	296	409	467	511	555	592
	Returning	0	0	26	37	26	21	30
CS1-S	All	1034	1044	1045	1047	1051	1053	1056
	Active	1034	847	787	790	776	755	746
	Inactive	0	197	258	257	275	298	310
	Returning	0	0	4	13	5	6	10
WD	All	169	174	174	174	174	175	
	Active	169	166	143	140	134	120	
	Inactive	0	8	31	34	40	55	
	Returning	0	0	0	1	0	0	

3.5 Procedure

For each subtask, we trained our machine learning models, random forest and logistic classifiers using the scikit-learn framework [48]. The approach here is similar to the approaches that have previously been applied in research on predicting academic performance (see e.g. [1]).

For model hyperparameter tuning and performance evaluation, we employ nested cross-validation using 5 folds for outer loop, and 3 folds for inner loop. We use PR-AUC (Precision-Recall Area Under Curve) as the metric to pick the best model hyperparameters. We report the performance of our models on the following commonly used performance metrics to vary the descriptiveness of our performance results: accuracy (intuitive metric with simple interpretation), RMSE (Root Mean Squared Error, a raw error metric for reliability), ROC-AUC (Receiving Operator Characteristic Area Under Curve a decision threshold independent metric that is unaffected by data skew), and F1-score (the harmonic mean of precision and recall which are both intuitive metrics for the problem).

We perform two preprocessing steps before feeding the data to our models: standardization and feature selection. Standardization balances the a priori importance of each feature, since different types of features have different magnitudes. We standardize each feature individually to have zero mean unit variance. Afterwards, we perform a feature selection step. We used Analysis of variance (ANOVA) to determine the $\sqrt{M}$ most relevant features as model inputs, where M is the total number of features. We also experimented with no feature selection at all, to rule out the possibility that feature selection removes important features.

In order to assess which features are most useful for dropout prediction, we examine the learned properties of the final models that are selected by the nested cross-validation process. We obtain the best subset of features from the final random forest model by looking at the accumulation of the impurity decrease within each tree of the model. We obtain the best subset of features from the final logistic regression model by looking at the (normalized) coefficients of the model.

3.6 Implementation

The code used in this work is available at Aalto Version Control System[4] with the full configuration of the trained models and evaluation pipeline. Unfortunately, we cannot publish the data used due to privacy concerns.

3.7 Descriptive Statistics

Descriptive statistics of the data, including active students, inactive students, and returning students for each course for each round is shown in Table 2. The total number of students is increasing in most of the datasets, indicating that it is common for the courses to include a few students who start late. The overall student count is also higher in CS1-B than in CS1-S, indicating that students browse the course materials without submitting anything, especially when joining late.

Additionally, besides the variance between the dropout statistics, returning students, i.e. students who become inactive on some round but are active again on a later round, are scarce within the data – as an example, for WD, there is only one returning student out of all the students who become inactive in the course. When considering CS1-S and CS1-B, the number of returning students is higher in the browsing data than in the submission data, possibly due to students being more likely to inspect the upcoming course material for reference than doing assignments when unsure of continuing on a course.

4 RESULTS

4.1 RQ1. Differences in Predictive Power

Students returning on a coming round after a break is rare as is shown in Table 2. Thus, intuitively, the inclusion of previous weeks' dropouts for predicting all future dropouts in the including approach helps models perform better as it should be easy for models

[4]https://version.aalto.fi/gitlab/sarsas2/methodological-considerations-for-predicting-at-risk-students

to pick up on the trend that previously inactive students remain inactive. On the other hand, the excluding approach cannot leverage such information and should therefore perform worse. The model performance comparison over different rounds to predict, in Figure 1, suggests that the later the round we are trying predict, the larger the difference between the two approaches becomes. This effect can be confirmed by inspecting the plot in Figure 2 which shows a comparison of random forest models and majority vote baseline scores for different metrics with different feature sets for predicting round 7 or the last round available. In the plot, we see that apart from a few outliers, the random forest scores are clearly higher for the including approach than the excluding approach.

Also, in the including approach the scores are improving mostly monotonically as more rounds (i.e. features) are introduced, whereas in the excluding approach this trend is not present. Instead, in the excluding approach, predicting dropouts appears at times harder as more rounds are introduced, which is evident especially in the F1-score, although the drastic difference for the F1-metric can be partly explained by a smaller number of positive examples (dropouts) in the data. Moreover, the model performance comparison over different rounds to predict, in Figure 1, shows that the later the round we are trying predict, the larger the difference between the two approaches becomes.

Also in Figure 2, in contrast to the F1-score results, differences in accuracy are rather small. The excluding approach even achieves better accuracy than the including approach for the CS1-B dataset when using features from all rounds that precede the prediction round. Achieving high accuracy, however, is increasingly easier for the excluding approach, since the proportion of dropouts decreases the further we are in a given course. Unlike in the including approach, the excluding approach random forest model accuracy scores are not much higher than those of the majority vote baseline and logistic regression fairs even worse. A similar pattern can be seen for RMSE score. Furthermore, when inspecting e.g. random forest model's F1-Score for feature rounds 1-6 and dataset CS1-S in the excluding approach, it is extremely poor, and similarly, accuracy is below that of the majority vote. Even though ROC-AUC scores are relatively high for both approaches, the overall results still show that the models may have major difficulties in reliably identifying future dropouts, a result which is not evident when using the including approach.

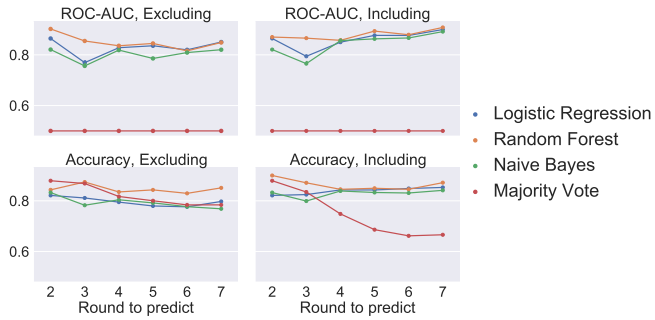

Figure 1: Performance of the models for accuracy and ROC-AUC metrics for each round to predict. Scores are averaged over all previous round feature sets over all evaluated datasets.

4.2 RQ2. Discriminative Features

We ranked the features given the importance given by each final model for each prediction task, and we selected the top 10 features. Figure 3 shows the percentage (from these top 10 features) of features of each type presented in subsection 3.3, for each type of dataset, averaged over all the weeks. The first thing that we observe is that both approaches rank the types of features rather similarly (within a given course). There are only variations in the magnitude of the importance while the feature ranking is the same between the two approaches. For the submission data, we see that the including approach has a tendency to give more importance to the correctness variable. Concerning browsing data, both the including and the excluding approach favor time spent as the most discriminating feature. However, this is also likely due to the higher proportion of time related features than study session related features. With that factor taken into account, it seems that the including approach makes equal use of study session statistics, while the excluding approach favors slightly more the study session length than the number of study sessions.

We show as illustration the top 10 submission features used by the RandomForest algorithm to discriminate dropouts from non dropouts at the 6th round, using data from all previous rounds, for the CS1 course (Table 3) and the CS0-Web course (Table 4). Figure 4 gives the distribution of the rounds from which these features come from. We observe that there are no major differences in terms of the type of feature used by both approaches in that scenario. For the CS1 course, there are also no differences in between approaches concerning the distribution of features with respect to rounds, as both approaches focus on assignments from the fifth round (Table 3). However, for the CS0-Web course, we observe that while the excluding approach leverages features from all rounds, the including approach only leverages features from the third and the fifth round (Table 4).

Nevertheless, we must be careful of over-interpreting these results as the ranking is prone to variance. For instance, simply choosing another number of top features to examine, restricting the selection to a specific round to predict or applying shuffling in cross-validation would show a slightly different picture.

4.3 RQ3. Effects of The Course Context

Overall, the performance of the models vary across the courses and across the rounds (Figure 2). While the performance seems to stay relatively similar across courses when using the including approach (same trend), the trend of the performance of the model under the excluding approach tends to be dissimilar (for accuracy and RMSE, dissimilar in relation to majority vote). One of the possible causes for this is the data – as discussed in descriptive statistics (Section 3.7), while many students drop out early on in the courses, there are fewer students dropping out later in the courses.

The bottom plot of Figure 3 highlights the difference and similarities in the top submission features when taking into account the course context. We can make a few observations about similarities and differences across courses. For the CS0-Web and the

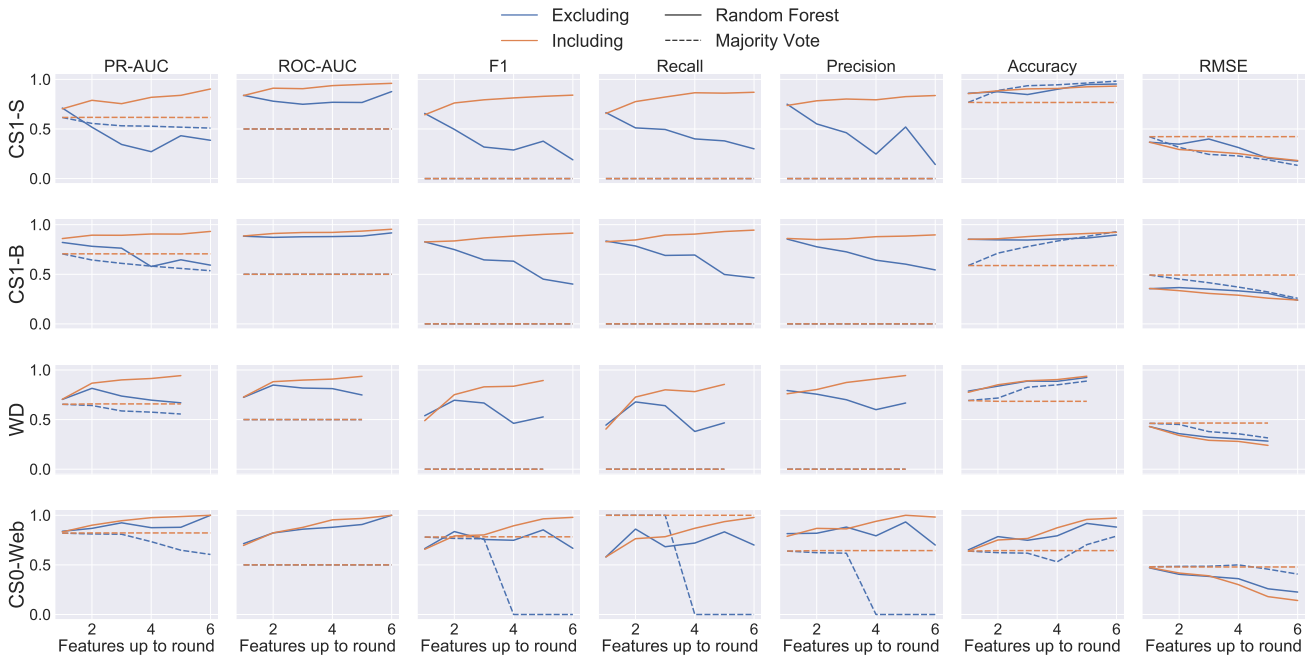

Figure 2: The difference of scores using including versus excluding approach for each dataset and metric when using features sets 1, 1-2, ... and 1-6 to predict dropouts on round 7. For WD course feature sets up to 1-5 are used to predict round 6 since it doesn't contain round 7.

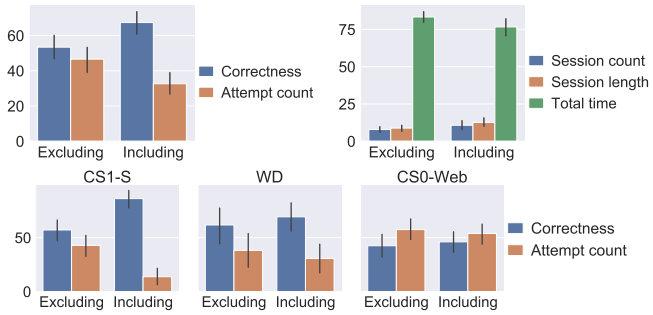

Figure 3: Percentage of type of features used in both approaches. On the top left (resp right), the percentage averaged over all prediction tasks and all submission (resp the browsing) datasets. On the bottom, the percentage individualized for each submission dataset. The percentage is computed as the average of using the top 10 features over each model and each subtask.

WD courses, the differences in the chosen features are not large between the excluding and the including approaches, while CS1-S shows a bigger difference; namely, that the including approach favors correctness over attempt count compared to the excluding approach. In practice, this suggests the need for course-specific predictive models, or the use of approaches such as transfer learning, as discussed in [30].

Table 3: Top 10 features used by the RandomForest model for predicting students dropping out at round 6 using data from all previous rounds with the excluding approach (top) and the including approach (bottom) for the CS1-S dataset.

	importance	name	round	feature type
1	0.080577	Assignment 26 (For 1)	5	Attempt count
2	0.080284	Assignment 35 (Flappybug 16)	5	Attempt count
3	0.048048	Assignment 28 (For 2)	5	Attempt count
4	0.044888	Assignment 30 (For 4)	5	Correctness
5	0.041555	Assignment 37 (For 7)	5	Attempt count
6	0.037730	Assignment 29 (For 3)	5	Attempt count
7	0.031279	Assignment 35 (Flappybug 16)	5	Correctness
8	0.026363	Assignment 24 (Purchases of)	5	Correctness
9	0.024191	Assignment 29 (For 3)	5	Correctness
10	0.022359	Assignment 26 (For 1)	5	Correctness

	importance	name	round	feature type
1	0.169364	Assignment 26 (For 1)	5	Correctness
2	0.129253	Assignment 28 (For 2)	5	Correctness
3	0.109589	Assignment 6 (Fixed price)	5	Correctness
4	0.079372	Assignment 29 (For 3)	5	Correctness
5	0.069453	Assignment 17 (Tempo)	5	Correctness
6	0.069156	Assignment 30 (For 4)	5	Correctness
7	0.049497	Assignment 23 (Number of Open Items)	5	Correctness
8	0.049484	Assignment 27	5	Correctness
9	0.049389	Assignment 24 (Purchases of)	5	Correctness
10	0.039598	Assignment 33 (District 1)	5	Correctness

Table 4: Top 10 features used by the RandomForest model for predicting students dropping out at round 6 using data from all previous rounds with the excluding approach (top) and the including approach (bottom) for the CS0-Web dataset. The data did not include assignment names but only database IDs for the assignments, which are shown in the name column in the table.

	importance	name	round	feature type
1	0.026288	476	3	Attempt count
2	0.021747	445	2	Attempt count
3	0.021477	441	2	Attempt count
4	0.020963	496	3	Attempt count
5	0.018968	444	2	Attempt count
6	0.018469	532	4	Attempt count
7	0.017219	472	3	Attempt count
8	0.015007	554	5	Attempt count
9	0.014855	570	5	Correctness
10	0.014628	331	1	Attempt count

	importance	name	round	feature type
1	0.058946	566	5	Attempt count
2	0.055328	566	5	Correctness
3	0.049541	573	5	Attempt count
4	0.044551	567	5	Correctness
5	0.038696	560	5	Attempt count
6	0.037591	561	5	Attempt count
7	0.037280	563	5	Attempt count
8	0.035821	562	5	Attempt count
9	0.030171	567	5	Attempt count
10	0.028314	474	3	Attempt count

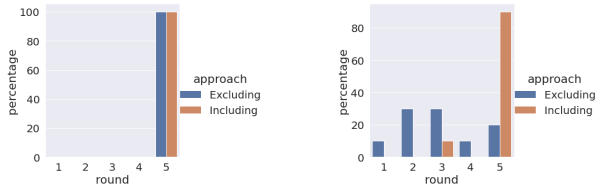

Figure 4: Percentage of the top 10 features of previous rounds used by the RandomForest classifier when predicting dropouts at round 6 using data from all previous rounds, for the CS1-course (left) and the CS0-Web course (right).

5 DISCUSSION

Overall, using two types of data from three separate courses, our results highlight two important intertwined aspects. First, the further into a course we go, the fewer students are actually dropping out, and second, predictive models that include all the data until the round to predict, i.e. the including approach, perform better than the predictive models that include only students who have been active at least until the previous round of the round to predict, i.e. the excluding approach. As an example, in the CS1-S data, there were only 20 dropouts from round 4 to round 5; the excluding approach seeks to identify these, while the including approach seeks to identify those in addition to the students who have already dropped out, which creates a target that is at least ten times larger. This effectively highlights a data imbalance issue, where one group to predict is larger than the other, which in turn makes the prediction task more difficult [62]. Approaches, such as sampling, can alleviate the issue to some extent [56], although we did not explore it in the present work.

The problem of students dropping out is especially important in the context of massive open online courses [8, 47], where the majority of the participants may drop out before the end of the course. Although one might argue that students might come back later though, as our data (Table 2) shows, this is fairly rare in our context. In practice, the relative number of students who drop out of courses tends to decrease the further we go into the course [47], although our data showed that there are differences in how this manifests. Building a predictive model that includes data from students who have already dropped out may lead to the models using features in data that are not useful in practice. This might hamper efforts where data from one course is used to predict the outcomes in another course, although in such cases one might also explore techniques such as transfer learning (see e.g. [30]). Although we identified studies that evaluate predictive models with data that likely includes students who have already dropped out, we typically did not observe evidence of the same models being taken into use as, e.g., early warning systems.

When considering the metrics that we used for studying the performance of the built models, we observed considerable differences between the excluding and including approach in most of the metrics in most of the courses (shown in Figure 2). While accuracy tended to be a poor metric in general due to the imbalanced dataset, the ROC-AUC masked performance differences for CS1-B and CS0-Web. On the other hand, these differences were clearly visible e.g. when studying the F1 score. This effectively highlights that when reporting results of such analyses, multiple metrics should be included.

When considering the problem of students dropping out in general, we acknowledge that there are a multitude of issues that are at play. While losing interest or preferring other work, highlighted e.g. in [26, 50], might be something that cannot be addressed, other factors – also highlighted in [26, 50] – such as plagiarism or excessively relying on others could be alleviated through interventions. It is a good question, however, what sorts of features should be used to identify factors such as confusion and the actions taken in such situations, as discussed in [40].

This study does not come without limitations, either. First, we defined dropout through inactivity, but students who drop out late might still get a passing grade. Second, despite our data showing that very few students become active after becoming inactive, this is still a possibility and a corner case, which the including approach could handle – despite its flaws – better. Third, as in many evaluations of predicting dropouts, we used data from the same course, which itself could lead to bias in the data. Finally, we also did not look into demographics or more fine-grained features and thus, cannot state how the two approaches would compare in those situations. Furthermore, our feature analysis is rather limited.

6 CONCLUSION

We have examined the differences in two approaches for identifying students who are likely to drop out. We observed that the including approach which includes also students who have already dropped out does not match the objective of most research, which typically

tries to assess the usability of data collected for developing real-time warning applications for educational teams. Importantly, we do not claim that everyone does this wrong, but rather, that there is a risk of bias that *needs* to be acknowledged and recognized. To summarize, our research questions and the answers are as follows.

RQ1: What performance differences do we observe in predictive models when we include versus exclude data from students who have already dropped out? First, we observed differences in the performance of the evaluated machine learning models; out of the explored models, random forest had the highest performance, which could be due to its behavior in identifying outliers. We also observed considerable differences between the excluding and including approaches, the latter including the students who have already dropped out in the data. In practice, due to the decreased number of dropouts over the weeks, the performance of the excluding approach tends to generally decline, while the opposite is true for the including approach. We did not, however, evaluate sampling (see e.g. [49]) or other approaches often used to alleviate the problem.

RQ2: What features are most predictive of students dropping out when including versus excluding data from students who have already dropped out? Comparing the features that the predictive models used between the two approaches, we observed slightly different emphases. However, contrary to our expectations, the differences are hardly evident enough to conclude the types of features used between the two approaches are inherently different. We did not, however, look into the individual features used by the models.

RQ3: How does the course context influence the performance of the predictive models? When comparing the performance of the predictive models between the courses, we observed significant differences. It is evident that the context and thus the data affects the results significantly. This further highlights the problem of using models developed in one context in other contexts, and also emphasizes the meaningfulness of reanalyses and replications of similar work in the future.

Effectively, our work highlights the importance of paying attention to the data in addition to the models. While research on predicting at-risk students inherently strives for high performance, focusing only on model evaluations and not paying sufficient attention to what data is being used can lead to wrong conclusions. An interesting question arises from our observations: how can we take into account the information about dropped out students in previous weeks to increase the performances of the models which try to detect which students drop out in the following weeks without risking unrealistic performance inflation? An option might be to revisit methodologies on transfer learning — shifting our focus from course to course behavior to learning about the dropout patterns within a specific course.

Finally, we need to consider the demographics of students in our courses and approaches. Dropouts, and other at-risk behavior, is also likely to stem from factors other than the differences in the course organization and the course materials used. Predictive models should be developed with care and consideration in order to achieve true fairness, and to avoid projecting our own biases into them in our attempts at improving the learning process for all [46]. As Corbett-Davies & Goel [11] so eloquently put it: *"Algorithms can avoid many of the implicit and explicit biases of human decisions makers, but they can also exacerbate historical inequities if not developed with care."*

ACKNOWLEDGMENTS

We acknowledge the computational resources provided by the Aalto Science-IT project. We are grateful for the grant by the Media Industry Research Foundation of Finland which partially funded this work.

REFERENCES

[1] Alireza Ahadi, Raymond Lister, Heikki Haapala, and Arto Vihavainen. 2015. Exploring machine learning methods to automatically identify students in need of assistance. In *Proceedings of the eleventh annual international Conf. on international computing education research*. 121–130.

[2] Alireza Ahadi, Raymond Lister, Shahil Lal, Juho Leinonen, and Arto Hellas. 2017. Performance and consistency in learning to program. In *Proceedings of the Nineteenth Australasian Computing Education Conference*. 11–16.

[3] Mohammad Majid al Rifaie, Matthew Yee-King, and Mark d'Inverno. 2016. Investigating swarm intelligence for performance prediction. (2016).

[4] Kimberly E Arnold and Matthew D Pistilli. 2012. Course signals at Purdue: Using learning analytics to increase student success. In *Proceedings of the 2nd international Conf. on learning analytics and knowledge*. 267–270.

[5] Tapio Auvinen, Lasse Hakulinen, and Lauri Malmi. 2015. Increasing students' awareness of their behavior in online learning environments with visualizations and achievement badges. *IEEE Transactions on Learning Technologies* 8, 3 (2015), 261–273.

[6] Brett A Becker. 2016. A new metric to quantify repeated compiler errors for novice programmers. In *Proceedings of the 2016 ACM Conf. on Innovation and Technology in Computer Science Education*. 296–301.

[7] Eric P Bettinger and Rachel B Baker. 2014. The effects of student coaching: An evaluation of a randomized experiment in student advising. *Educational Evaluation and Policy Analysis* 36, 1 (2014), 3–19.

[8] Luis N Bezerra and Márcia T Silva. 2017. A review of literature on the reasons that cause the high dropout rates in the MOOCS. *Revista Espacios* 38, 05 (2017).

[9] Adam S Carter, Christopher D Hundhausen, and Olusola Adesope. 2015. The normalized programming state model: Predicting student performance in computing courses based on programming behavior. In *Proceedings of the eleventh annual International Conf. on International Computing Education Research*. 141–150.

[10] Karo Castro-Wunsch, Alireza Ahadi, and Andrew Petersen. 2017. Evaluating neural networks as a method for identifying students in need of assistance. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education*. 111–116.

[11] Sam Corbett-Davies and Sharad Goel. 2018. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023* (2018).

[12] Timothy P Cronan, Phillip R Embry, and Steven D White. 1989. Identifying factors that influence performance of non-computing majors in the business computer information systems course. *J. of Research on Computing in Education* 21, 4 (1989), 431–446.

[13] Paul Denny, Jacqueline Whalley, and Juho Leinonen. 2021. Promoting Early Engagement with Programming Assignments Using Scheduled Automated Feedback. In *Australasian Computing Education Conference*. 88–95.

[14] John Edwards, Juho Leinonen, and Arto Hellas. 2020. A study of keystroke data in two contexts: Written language and programming language influence predictability of learning outcomes. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. 413–419.

[15] K Anders Ericsson, Ralf T Krampe, and Clemens Tesch-Römer. 1993. The role of deliberate practice in the acquisition of expert performance. *Psychological review* 100, 3 (1993), 363.

[16] Gerald E Evans and Mark G Simkin. 1989. What best predicts computer proficiency? *Commun. ACM* 32, 11 (1989), 1322–1327.

[17] Nickolas JG Falkner and Katrina E Falkner. 2012. A fast measure for identifying at-risk students in computer science. In *Proceedings of the ninth annual international Conf. on International computing education research*. 55–62.

[18] Darrell W Guillaume and Crist Simon Khachikian. 2011. The effect of time-on-task on student grades and grade expectations. *Assessment & Evaluation in Higher Education* 36, 3 (2011), 251–261.

[19] Joonas Häkkinen, Petri Ihantola, Matti Luukkainen, Antti Leinonen, and Juho Leinonen. 2021. Persistence of Time Management Behavior of Students and Its Relationship with Performance in Software Projects. In *Proceedings of the 17th ACM Conference on International Computing Education Research*. 92–100.

[20] Arto Hellas, Petri Ihantola, Andrew Petersen, Vangel V Ajanovski, Mirela Gutica, Timo Hynninen, Antti Knutas, Juho Leinonen, Chris Messom, and Soohyun Nam Liao. 2018. Predicting academic performance: a systematic literature review. In

Proceedings companion of the 23rd annual ACM Conf. on innovation and technology in computer science education. 175–199.

[21] Christopher David Hundhausen, Daniel M Olivares, and Adam S Carter. 2017. IDE-based learning analytics for computing education: a process model, critical review, and research agenda. *ACM Transactions on Computing Education (TOCE)* 17, 3 (2017), 1–26.

[22] Petri Ihantola, Arto Vihavainen, Alireza Ahadi, Matthew Butler, Jürgen Börstler, Stephen H Edwards, Essi Isohanni, Ari Korhonen, Andrew Petersen, Kelly Rivers, et al. 2015. Educational data mining and learning analytics in programming: Literature review and case studies. *Proceedings of the 2015 ITiCSE on Working Group Reports* (2015), 41–63.

[23] Kalle Ilves, Juho Leinonen, and Arto Hellas. 2018. Supporting self-regulated learning with visualizations in online learning environments. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 257–262.

[24] Matthew C Jadud. 2006. Methods and tools for exploring novice compilation behaviour. In *Proceedings of the second international workshop on Computing education research*. 73–84.

[25] Hassan Khosravi and Kendra ML Cooper. 2017. Using learning analytics to investigate patterns of performance and engagement in large classes. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education*. 309–314.

[26] Päivi Kinnunen and Lauri Malmi. 2006. Why students drop out CS1 course?. In *Proceedings of the second international workshop on Computing education research*. 97–108.

[27] Charles Koutcheme, Juho Leinonen, Juha Sorva, and Arto Hellas. 2021. Analyzing Fine-Grained Material Usage Behavior. In *CS Education Infrastructure for All III: From Ideas to Practice: Seventh SPLICE Project Workshop*.

[28] Andrew E Krumm, R Joseph Waddington, Stephanie D Teasley, and Steven Lonn. 2014. A learning management system-based early warning system for academic advising in undergraduate engineering. In *Learning analytics*. Springer, 103–119.

[29] Mukesh Kumar, AJ Singh, and Disha Handa. 2017. Literature survey on student's performance prediction in education using data mining techniques. *International J. of Education and Management Engineering* 7, 6 (2017), 40–49.

[30] Jarkko Lagus, Krista Longi, Arto Klami, and Arto Hellas. 2018. Transfer-learning methods in programming course outcome prediction. *ACM Transactions on Computing Education (TOCE)* 18, 4 (2018), 1–18.

[31] Robert R Leeper and James L Silver. 1982. Predicting success in a first programming course. *ACM SIGCSE Bulletin* 14, 1 (1982), 147–150.

[32] Juho Leinonen. 2019. *Keystroke Data in Programming Courses*. Ph.D. Dissertation. University of Helsinki.

[33] Juho Leinonen, Francisco Enrique Vicente Castro, and Arto Hellas. 2021. Does the Early Bird Catch the Worm? Earliness of Students' Work and its Relationship with Course Outcomes. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*. 373–379.

[34] Juho Leinonen, Francisco Enrique Vicente Castro, Arto Hellas, et al. 2021. Fine-Grained Versus Coarse-Grained Data for Estimating Time-on-Task in Learning Programming. In *Proceedings of The 14th International Conference on Educational Data Mining (EDM 2021)*. The International Educational Data Mining Society.

[35] Juho Leinonen, Leo Leppänen, Petri Ihantola, and Arto Hellas. 2017. Comparison of time metrics in programming. In *Proceedings of the 2017 acm conference on international computing education research*. 200–208.

[36] Juho Leinonen, Krista Longi, Arto Klami, and Arto Vihavainen. 2016. Automatic inference of programming performance and experience from typing patterns. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. 132–137.

[37] Leo Leppänen, Juho Leinonen, and Arto Hellas. 2016. Pauses and spacing in learning to program. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*. 41–50.

[38] Leo Leppänen, Juho Leinonen, Petri Ihantola, and Arto Hellas. 2017. Predicting academic success based on learning material usage. In *Proceedings of the 18th Annual Conf. on Information Technology Education*. 13–18.

[39] Leo Leppänen, Juho Leinonen, Petri Ihantola, and Arto Hellas. 2017. Using and collecting fine-grained usage data to improve online learning materials. In *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET)*. IEEE, 4–12.

[40] Soohyun Nam Liao, Sander Valstar, Kevin Thai, Christine Alvarado, Daniel Zingaro, William G Griswold, and Leo Porter. 2019. Behaviors of higher and lower performing students in CS1. In *Proc. of the 2019 ACM Conf. on Innovation and Technology in Computer Science Education*. 196–202.

[41] Soohyun Nam Liao, Daniel Zingaro, Michael A Laurenzano, William G Griswold, and Leo Porter. 2016. Lightweight, early identification of at-risk CS1 students. In *Proc. of the 2016 acm Conf. on international computing education research*. 123–131.

[42] Soohyun Nam Liao, Daniel Zingaro, Kevin Thai, Christine Alvarado, William G Griswold, and Leo Porter. 2019. A robust machine learning technique to predict low-performing students. *ACM Transactions on Computing Education (TOCE)* 19, 3 (2019), 1–19.

[43] Martín Liz-Domínguez, Manuel Caeiro Rodríguez, Martín Llamas Nistal, and Fernando A Mikic-Fonte. 2019. Predictors and early warning systems in higher education-A systematic literature review.. In *LASI-SPAIN*. 84–99.

[44] Ioanna Lykourentzou, Ioannis Giannoukos, Vassilis Nikolopoulos, George Mpardis, and Vassili Loumos. 2009. Dropout prediction in e-learning courses through the combination of machine learning techniques. *Computers & Education* 53, 3 (2009), 950–965.

[45] Kew Si Na and Zaidatun Tasir. 2017. Identifying at-risk students in online learning by analysing learning behaviour: A systematic review. In *2017 IEEE Conf. on Big Data and Analytics (ICBDA)*. IEEE, 118–123.

[46] Eirini Ntoutsi, Pavlos Fafalios, Ujwal Gadiraju, Vasileios Iosifidis, Wolfgang Nejdl, Maria-Esther Vidal, Salvatore Ruggieri, Franco Turini, Symeon Papadopoulos, Emmanouil Krasanakis, et al. 2020. Bias in data-driven artificial intelligence systems—An introductory survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10, 3 (2020), e1356.

[47] Daniel FO Onah, Jane Sinclair, and Russell Boyatt. 2014. Dropout rates of massive open online courses: behavioural patterns. *EDULEARN14 proceedings* 1 (2014), 5825–5834.

[48] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *J. of Machine Learning Research* 12 (2011), 2825–2830.

[49] Filipe Dwan Pereira, Samuel C Fonseca, Elaine HT Oliveira, Alexandra I Cristea, Henrik Bellhäuser, Luiz Rodrigues, David BF Oliveira, Seiji Isotani, and Leandro SG Carvalho. 2021. Explaining Individual and Collective Programming Students' Behavior by Interpreting a Black-Box Predictive Model. *IEEE Access* 9 (2021), 117097–117119.

[50] Andrew Petersen, Michelle Craig, Jennifer Campbell, and Anya Tafliovich. 2016. Revisiting why students drop CS1. In *Proceedings of the 16th Koli Calling International Conf. on Computing Education Research*. 71–80.

[51] Simone Plak, Ilja Cornelisz, Martijn Meeter, and Chris van Klaveren. 2019. Early warning systems for more effective student counselling in higher education: Evidence from a Dutch field experiment. *Higher Education Quarterly* (2019).

[52] Leo Porter, Daniel Zingaro, and Raymond Lister. 2014. Predicting student success using fine grain clicker data. In *Proc. of the tenth annual Conf. on International computing education research*. 51–58.

[53] Bardh Prenkaj, Paola Velardi, Giovanni Stilo, Damiano Distante, and Stefano Faralli. 2020. A Survey of Machine Learning Approaches for Student Dropout Prediction in Online Courses. *ACM Comput. Surv.* 53, 3, Article 57 (May 2020), 34 pages. https://doi.org/10.1145/3388792

[54] Amirah Mohamed Shahiri, Wahidah Husain, et al. 2015. A review on predicting student's performance using data mining techniques. *Procedia Computer Science* 72 (2015), 414–422.

[55] Simon, Andrew Luxton-Reilly, Vangel V Ajanovski, Eric Fouh, Christabel Gonsalvez, Juho Leinonen, Jack Parkinson, Matthew Poole, and Neena Thota. 2019. Pass rates in introductory programming and in other STEM disciplines. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*. 53–71.

[56] Dech Thammasiri, Dursun Delen, Phayung Meesad, and Nihat Kasap. 2014. A critical assessment of imbalanced class distribution problem: The case of predicting freshmen student attrition. *Expert Systems with Applications* 41, 2 (2014), 321–330.

[57] Christopher Watson, Frederick WB Li, and Jamie L Godwin. 2013. Predicting performance in an introductory programming course by logging and analyzing student programming behavior. In *2013 IEEE 13th international Conf. on advanced learning technologies*. IEEE, 319–323.

[58] Christopher Watson, Frederick WB Li, and Jamie L Godwin. 2014. No tests required: comparing traditional and dynamic predictors of programming success. In *Proceedings of the 45th ACM technical symposium on Computer science education*. 469–474.

[59] Laurie Honour Werth. 1986. Predicting student performance in a beginning computer science class. *ACM SIGCSE Bulletin* 18, 1 (1986), 138–143.

[60] Jacob Whitehill, Kiran Mohan, Daniel Seaton, Yigal Rosen, and Dustin Tingley. 2017. MOOC dropout prediction: How to measure accuracy?. In *Proceedings of the fourth (2017) acm Conf. on learning@ scale*. 161–164.

[61] Brenda Cantwell Wilson and Sharon Shrock. 2001. Contributing to success in an introductory computer science course: a study of twelve factors. *Acm sigcse bulletin* 33, 1 (2001), 184–188.

[62] L. Xu and Mo-Yuen Chow. 2006. A classification approach for power distribution systems fault cause identification. *IEEE Transactions on Power Systems* 21, 1 (2006), 53–60.

[63] Albina Zavgorodniaia, Raj Shrestha, Juho Leinonen, Arto Hellas, and John Edwards. 2021. Morning or Evening? An Examination of Circadian Rhythms of CS1 Students. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. IEEE, 261–272.