Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025 ASEE
Paper ID #48329

Identifying Struggling Students Using LMS Data

Dr. Abdulmalek Al-Gahmi, Weber State University

Dr. Abdulmalek Al-Gahmi is an associate professor at the School of Computing Department of Weber State University. His teaching experience involves courses on object-oriented programming, full-stack web development, computer graphics, algorithms and data structures, and machine learning. He holds a Ph.D. in Computer Science from New Mexico State University, M.S. in Computer Science, M.A. in Extension Education, and B.S. in Electrical Engineering.

Identifying Struggling Students Using LMS Data

Abstract

Identifying struggling students has long been a key objective for educators and institutions. It allows for timely interventions that can improve student retention and graduation rates—two critical components of most institutional missions. This paper reports on our experimentation with developing and training machine learning models to identify struggling students using Learning Management System (LMS) data. Our findings indicate that while these models do not perform as well when making predictions over long-term periods, such as an entire semester, they perform significantly better over shorter timespans.

The focus on LMS data stems from its ubiquity and the fact that instructors using LMS platforms for instruction have direct access to the collected data without needing to submit additional applications or navigate multiple levels of approval to acquire such access. However, this task is challenging for several reasons. First, course-related data is inherently imbalanced. Second, many factors can adversely affect student performance, and these can arise at any point during the semester. Most of these factors lie outside the LMS and, therefore, are not captured by its data. Finally, it often takes time for signs of student struggles to become evident in the data.

This paper proposes a process through which struggling students within courses can be accurately and regularly identified. The process is evaluated using data spanning a three-year period from a public, four-year university. The data includes 274 lower-division Computer Science courses delivered in various formats (face-to-face, online, and virtual), involving 2,656 students and 37 instructors.

Introduction

Identifying struggling students has long been a key objective for educators and institutions. It enables timely interventions that can improve student retention and graduation rates—two critical components of most institutional missions. Various methods and predictive models have been proposed to achieve this goal, with varying degrees of success. Some methods focus on the exam level, others on the course level, and still others on the degree level. These efforts draw from multiple data sources, including pre-college information, student information systems (SIS), and Learning Management Systems (LMS).

This paper concludes a research project focused on the sole use of Learning Management System (LMS) data for identifying struggling students. While we recognize the importance of other data sources, there are three main reasons for concentrating on LMS data in this project.

First, Learning Management Systems are ubiquitous and widely used by higher-education institutions across colleges and disciplines, to the point of becoming a standard component of classroom technology [15]. They provide a convenient and effective way to deliver learning materials to students.

Second, although studies suggest that LMS platforms are underutilized [14], they remain central to much of the course-related activity, including discussions, student-instructor interactions, and assessments. This project uses data from these graded activities to capture the journeys of

students in classes and determine whether they are struggling or doing well. Conveniently, LMS platforms maintain an extensive record of such activities and make this data accessible through dedicated API services.

Third, LMS platforms offer direct access to collected data, allowing instructors to evaluate student performance without needing to navigate multiple levels of approval. This accessibility provides instructors with timely insights into their students' progress.

The use of LMS platforms is not without challenges. First, many classroom-related activities that could take place within these systems often do not, resulting in the collected data being less comprehensive and accurate in reflecting students' progress through courses than it could be. Second, there is significant inconsistency and variance in how these systems are used by instructors. Even sections of the same course at the same department/college can look vastly different depending on the instructor, the modality, and other factors. Such inconsistencies affect the quality of the data and can undermine its effectiveness.

That said, a body of research has explored the use of LMS data to predict student performance at the course level. A frequently posed question in these studies is whether student performance can be accurately predicted early enough to intervene and provide struggling students with the help they need.

The central question of this study is whether student performance can be predicted reliably and early enough to enable corrective action. To explore this question, the study employs a time-based data analysis approach. This approach represents student progress in courses as sequences or time series, where each time step corresponds to a student's activities in the course, and each sequence reflects an individual student's progress. At the end of each course, the result is a collection of time series, each capturing the progress of a single student in that course.

This task, however, is not straightforward. As we will see, student performance data is often imbalanced, with fewer struggling students compared to those performing well. Such an imbalance complicates the use and interpretation of traditional performance metrics, such as accuracy. Additionally, many factors can adversely affect student performance, and these factors can happen at any point during the semester. Most of these factors lie outside the scope of the LMS and are therefore not captured in its data. Lastly, it often takes time for signs of student struggles to become evident in the available data.

The rest of this paper is organized as follows. The next section briefly reviews the background and related work relevant to this study. The following section outlines the research question and details the approach taken regarding data acquisition, cleanup, preprocessing, feature engineering, and modeling. The subsequent section examines the preprocessed data and its diversity. Afterward, the paper presents and discusses the results of all experiments conducted. Finally, the concluding section explores potential future work and offers final remarks.

Background and Related Work

Various approaches to predicting student performance have been explored. Some studies involve designing specific randomized experiments [1], [2], [4], [6], while others, like this one, focus on leveraging data collected by ubiquitous Learning Management Systems (LMSs) based on student

activities and interactions with course materials [3], [8], [9]. Additionally, some research evaluates the efficacy of specific teaching methodologies [4], [5], while others aim to identify struggling students early in the semester to enable timely interventions [6], [7], [9]. Like many of these studies, this paper emphasizes the early prediction of student performance, utilizing machine learning (ML) algorithms trained on LMS data.

The use of machine learning (ML) and Learning Management Systems (LMS) to predict student performance is not new. For example, Umer et al. [1] employed several ML algorithms to predict student outcomes by mining LMS activity log data. They emphasized the importance of this data for making such predictions but found that it does not necessarily lead to improved predictive accuracy. Similarly, Van Goidsenhoven et al. [2] analyzed LMS activity log data to predict student success, specifically including courses with blended learning environments. They discovered that predicting student success based on activity streams is more challenging in these types of courses. Both studies utilized a variety of ML algorithms, including random forest and logistic regression, and concluded that while counting activities is helpful for predictions, it alone is not sufficient.

Shayan et al. [3] explored predicting student performance based on their behavior in an LMS. However, their focus was on student performance in formative assessments rather than summative ones. Conijn et al. [4] investigated predicting student performance by comparing 17 blended courses. Their primary focus was on examining the portability of predictive models across multiple courses and the timeliness of these predictions. In doing so, they replicated a study by Gašević et al. [5] on the impact of instructional conditions on predicting success, but with a larger sample size and using predictors available for all courses. They noted the significant diversity in the number of variables used as predictors and highlighted the inconsistency of findings (and non-robustness) when the same or similar predictors are employed. They emphasized the need to expand the empirical base regarding portability, especially since some studies suggest that prediction accuracy improves over time.

Two previous papers in this project have demonstrated that machine learning models can be used to predict and identify at-risk students [8], [9]. The issues of portability and robustness have also been explored in another paper from this research project [13]. In this context, portability refers to the adaptability and effectiveness of a predictive model when applied to different educational settings or teaching styles. A portable model should maintain its predictive accuracy and generalizability when trained on various datasets, such as courses with different modalities, different semesters, or taught by different instructors. Robustness, on the other hand, refers to the ability of a predictive model to sustain its performance despite variations, uncertainties, or changes in data distribution or input conditions. A robust model should not be overly sensitive to minor changes in input data, such as variations in data quality or shifts in the student population. It should provide reliable predictions under a range of conditions.

To address the issue of small sample sizes prevalent in previous studies, Gonzalez et al. [6] conducted an analysis of massive LMS log data with the goal of achieving early, course-agnostic predictions of student performance. They employed several ML models in a course-agnostic manner to classify students into "fail," "at-risk," and "excellent" groups at various intervals (10%, 25%, 33%, and 50%) throughout the course. Data from all courses within a single university over the course of one year were utilized.

Furthermore, Dias et al. [7] introduced DeepLMS, a deep learning predictive model designed to support online learning, particularly during the Covid-19 era. They employed deep learning (DL) techniques to forecast the quality of interaction (QoI) with the LMS using Long Short-Term Memory (LSTM) networks, with RMSE errors used as evaluation metrics. By leveraging online learning to overcome the temporal and spatial limitations of traditional courses, they emphasized that a student's QoI serves as a strong indicator of the effectiveness of course design.

In addition to the use of LMS data, a growing number of studies have leveraged data from various other sources, including student records, institutional surveys, and external data sources, to provide a more comprehensive understanding of student performance. These studies have emphasized the importance of considering temporal relationships when developing models for student performance prediction. Liu et al. [16] explored how clickstream data can be used to predict students' learning behaviors, identify at-risk students, and inspire potential improvements in teaching and learning. Gamulin et al. [17] investigated the use of student access time series to predict final learning outcomes in blended learning courses. They applied discrete Fourier transforms and principal component analysis to improve and compress time series data, then built classification models based on naïve Bayes and support vector machines to predict student performance. Liu et al. [18] proposed a student course result prediction model based on historical course results and basic course information. Their model incorporates numeric and non-numeric feature vector embedding, along with model optimization through data augmentation and integration. Mitrovic et al. [19] used a feed-forward neural network to predict the number of errors a student will make in database courses based on all actions performed by the student in class.

In summary, there is great interest in developing methods and models that help educators and institutions better understand the journeys students undergo during their course enrollments. The central question in many of these studies has been whether student performance can be accurately predicted early enough to allow for timely intervention and support. Various data sources have been used to make such predictions, with many previous studies relying on fine-grained interaction and activity logs. However, these methods often suffer from issues of portability and robustness.

Approach

This study views a student's journey through a course as a sequence of activities captured as a time series, rather than as a single data point. This perspective provides a more detailed understanding of the journey and allows for better utilization of the readily available data. The main question of this study is:

How can struggling (at-risk-of-failing) students be accurately and reliably identified?

This paper addresses this question in three ways:

1. By extracting key features from the sequences of activities captured in the time series and using them as predictors in traditional machine learning models.
2. By using time series data to build forecasting models that predict student performance at any point during the course.

3. By proposing a process through which struggling students within courses can be accurately and regularly identified.

The detection of at-risk students in this study is framed as a binary classification problem with two distinct classes: POSITIVE (coded as 1) for struggling students, and NEGATIVE (coded as 0) for all others. This classification is based on a threshold of C- (a cumulative score of <74%) or below.

Data Acquisition

All data used in this study were extracted from the Canvas Learning Management System via RESTful and GraphQL APIs. The dataset includes information from 274 lower-division Computer Science (CS) courses, involving 2,656 students and 37 instructors. These courses are offered in various modalities, including face-to-face, online, and virtual formats, at a public four-year university. The courses exhibit diversity in terms of topics, class sizes, academic levels (1st and 2nd year), modalities (face-to-face, virtual, and online), semesters (spring, summer, and fall), and instructors.

The data analyzed in this paper comprises 10 required lower-division CS courses taught over a span of three years, from Spring 2019 to Summer 2022, catering to students pursuing their associate CS degrees during the same period. Across these courses, there were 274 sections, averaging 27.6 sections per course. As mandatory CS courses, they typically accommodate more students and are offered in various modalities compared to other courses. Many of these courses are taught multiple times in different modalities by various instructors within the same semester.

Primarily relying on the Learning Management System (LMS) as the main platform for instruction, these courses use the LMS for posting learning materials, facilitating discussions, and collecting assignments and other graded activities. The LMS meticulously records all activities and events within its interface. In addition to basic student information, it captures data related to assignments, quizzes, and other graded activities, including submission attempts, scores, and due dates. The LMS also maintains activity logs that document student interactions with resources such as pages, modules, or assignments, including details on what was accessed, when, and how frequently. This paper specifically focuses on the LMS data associated with assignments and other graded activities.

Data Cleanup and Preparation

The LMS data underwent two preparatory steps: anonymization and normalization. During the anonymization step, randomly assigned IDs replaced identifying names for course sections, instructors, students, assignments, and assignment groups. each course added up to 100%. This was essential to prevent discrepancies between scores on different assessments. For example, a score of 90% on a quiz worth 5% of the final grade should not be treated the same as a score of 90% on an exam worth 30%. Normalizing these scores was complex, as each course was structured differently. All calculations were carefully verified by comparing the cumulative scores at the end of each course with the actual final scores obtained from the LMS.

Courses without student activity in the LMS were removed from the dataset. Additionally, students whose cumulative normalized scores did not match their final scores were excluded.

The resulting dataset consists of time series sequences indexed by student IDs, course IDs, and timestamps. It includes columns for normalized scores, possible scores, and cumulative normalized and possible scores.

Finally, to standardize the representation of each student in every course, a fixed length of 100 was applied to each time series sequence. This choice allows us to interpret each data point as representing the student's status at a specific percentage point in the course. For longer time series, where two or more data points needed to be combined, a new time point was created by summing the normalized and possible scores. This process maintains the order of events (except for combined time points) and preserves the relative distances between data points.

Feature Engineering

In addition to compressing and fixing the length of the time series sequences, seven additional quantities were calculated at each point in time, t. Here are the first five features.

- **Missed opportunity**: This represents the amount of coursework that the student has missed up to that point in time and is calculated as:

$$\text{missed\_opportunity}_t = \text{possible\_score}_t - \text{actual\_score}_t$$

- **Relative achievement**: This indicates how much of what is possible for a student to achieve has been accomplished and is calculated as:

$$\text{relative\_achievement}_t = (\text{actual\_score}_t / \text{possible\_score}_t) \times 100$$

- **Number of missed assessments**: A missed assessment is defined as one for which the student achieved a zero score out of a non-zero possible score they could have achieved.

- **Number of late assessments**: A late assessment is defined as one that does not align timewise with its corresponding possible score, indicating it was not submitted on time.

- **Number of failed assessments**: This represents the count of assessments that the student submitted but failed to achieve a passing score.

During the normalization step, all these features are calculated cumulatively. This means that at a given time point *t*, the cumulative value of any of these features will be the sum of all the individual values of these features up to and including that point. This approach allows us to view the journey as a whole, rather than as individual data points scattered across the course timeline.

Furthermore, calculating relative achievement cumulatively has the advantageous property of aligning with the final total grade at the end of the course (or at time point 100). The focus on relative achievement stems from two main reasons:

- It is a strong predictor of student performance [9].
- The way it changes can provide insight into how students progress in courses. These changes are not linear; they fluctuate, depending on various factors/events in students' lives not totally captured in the data. From this, we extract the remaining two features.

These two additional features are:

- **Speed or the rate at which cumulative relative achievement changes**: This can be captured by fitting a line to the relative achievement points up to a given time point and calculating its slope. A low slope value indicates a near-constant speed, suggesting smooth progress through the course. A high absolute slope, on the other hand, indicates either an upward trend (e.g., doing well after a period of struggling) or a downward trend.
- **Acceleration or deceleration**: Since the rate at which cumulative relative achievement changes is not constant, we also need to consider acceleration or deceleration. These measures capture how the rate of change in cumulative relative achievement slows down or speeds up. This can be assessed by fitting a line to the speed feature above and taking the slope of that line.

All seven of these features will be used as input for the predictive models that we will build and train using the scikit-learn library [10].

Results

Before making predictions, we explore the data to understand its structure, relationships, and underlying patterns.

Data Exploration

The normalized dataset consists of 100 data-point time series per student per course, with each data point representing a cumulative snapshot of the student's progress at a specific percentage point in the course. Table 1 highlights the diversity of the dataset. The first three courses are at the first-year level, while the remaining courses are at the second-year level. All courses are offered in Spring, Summer, and Fall. Notably, the first three courses attract a larger number of students, including many non-CS students from engineering and other majors. This could lead to a different student population compared to that of second-year courses.

Table 1: Courses' sections, modalities, instructors and students.

#	Course	Modality	# of Sections	# of Instructors	# of Students
1	Computing Foundations	F2F, ONL, VTL	47	10	1210
2	Programming I	F2F, ONL	35	13	773
3	Object-Oriented Programming	F2F, ONL, VTL	20	10	607
4	Computational Structures	F2F, ONL, VTL	26	7	562
5	Client Side Web Development	F2F, ONL, VTL	25	7	565
6	Data Structures & Algorithms	F2F, ONL, VTL	19	5	532
7	Software Engineering I	F2F, ONL, VTL	23	5	497

8	Database Design & SQL	F2F, ONL	40	8	843
9	Network Fundamentals	F2F, ONL	20	3	509
10	Computer Architecture	F2F, ONL	21	5	534

Figure 1 (top) shows what these time series look like for two randomly selected students. It displays the cumulative normalized and possible scores, as well as the relative achievement of both students. The progression of a student's time in the course is depicted as an upward stair-like pattern. The width of the horizontal steps is determined by the number and distribution of graded activities throughout the course, while the height reflects the weights of these assignments and activities.

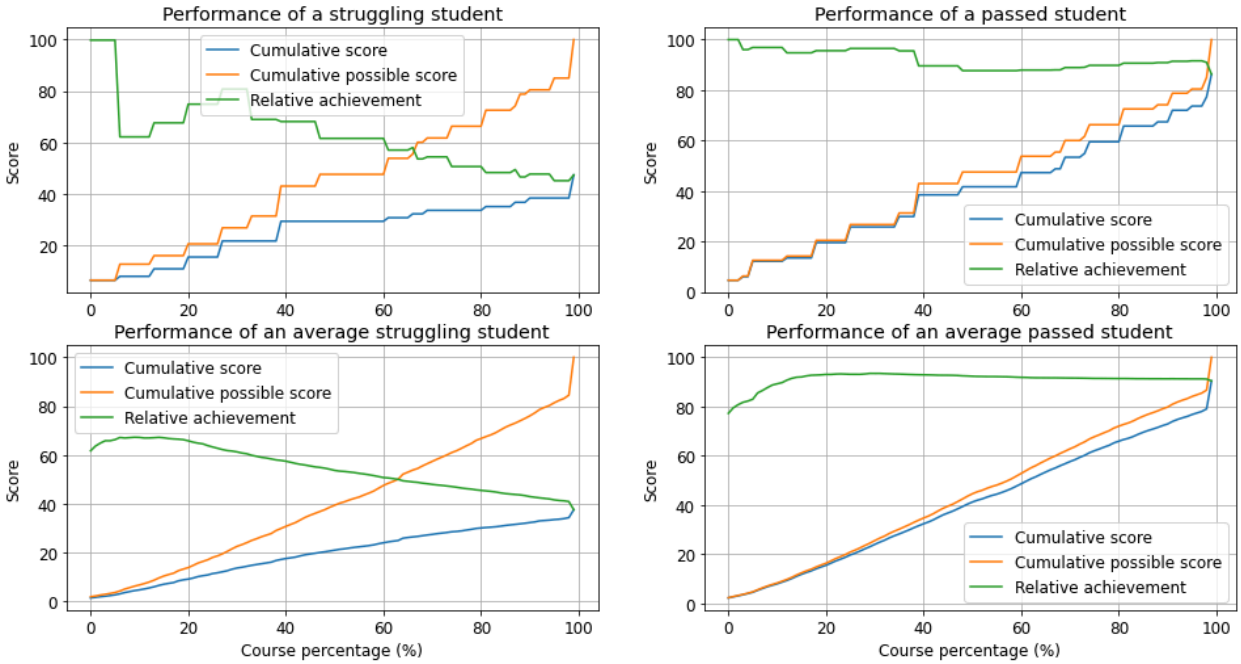

Figure 1: The progression of two struggling and passing students.

A student's struggle in a course can be visualized by the difference between their cumulative normalized and possible score curves. This difference tends to increase over time, particularly for at-risk students, indicating a steeper decline in performance. Additionally, this data allows for comparisons between the progress of an average at-risk student and an average passing student. Figure 1 (bottom) displays such progressions, averaged across the entire training dataset, side by side. Averaging results in smoother, almost linear curves, while maintaining similar gaps between actual and possible scores. Notice that by the end of the course, the relative achievement meets the cumulative normalized score.

As mentioned earlier, student progress through courses is not linear. This can be captured by the ups and downs of the relative achievement feature. Figure 2 (top) shows the relative achievement speed (velocity) and acceleration/deceleration for the same randomly selected students from Figure 1. This figure illustrates that the speed of these changes is not always constant, especially at the beginning of the course. It also shows that the changes smooth out in the latter part of the course, with both speed and acceleration becoming nearly constant. This trend can also be see in the averaged curves of Figure 2 (bottom).

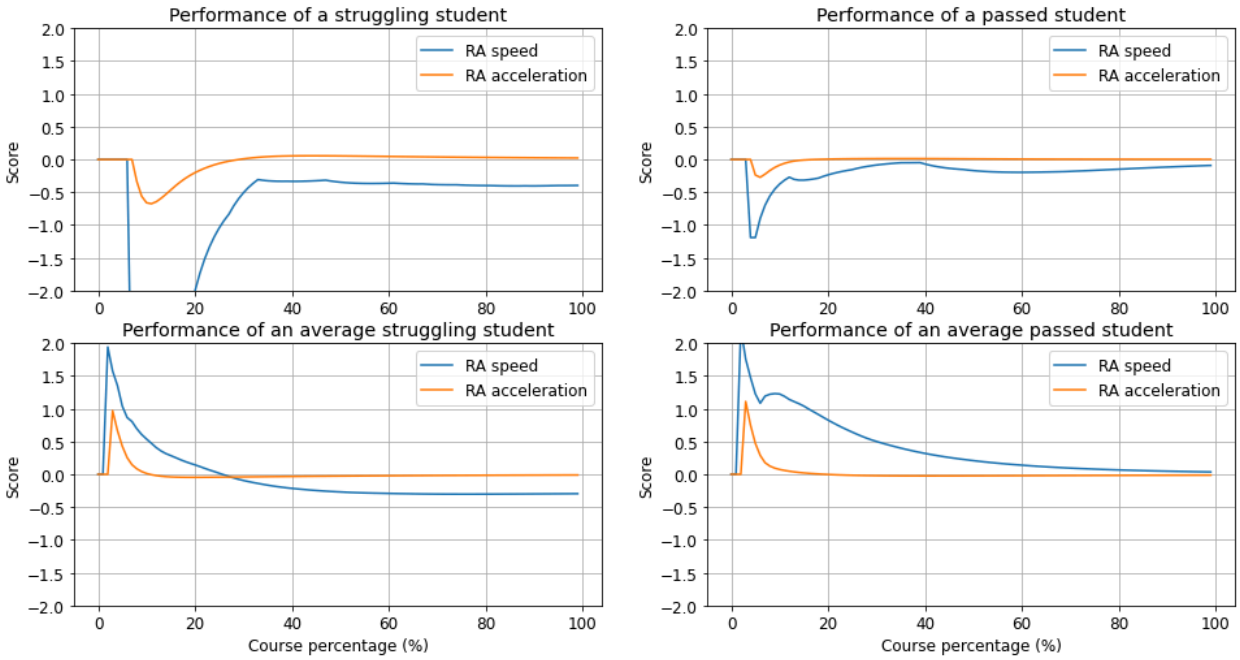

Figure 2 : The speed and acceleration of the relative achievement for the same two students.

Looking more closely at struggling students, one might ask how early they can be identified in a course. Figure 3 shows that cumulative relative achievement can be used to identify struggling students. It indicates that while most struggling students can be identified by the first third of the course, new struggling students continue to emerge until the end of the semester. This suggests that remedial services are needed throughout the semester. However, those identified earlier tend to have a better chance of turning things around and achieving better outcomes than those identified later. The histogram, along with the distribution curve superimposed on it, shows an increase in detected struggling students toward the end of the semester. These are students who performed well for most of the course but began to struggle toward the end, possibly due to personal events or circumstances not captured in the data. These students might require different types of remedial services.

Moreover, since new struggling students continue to emerge throughout the course, the process of detecting them should be ongoing throughout the semester as well.

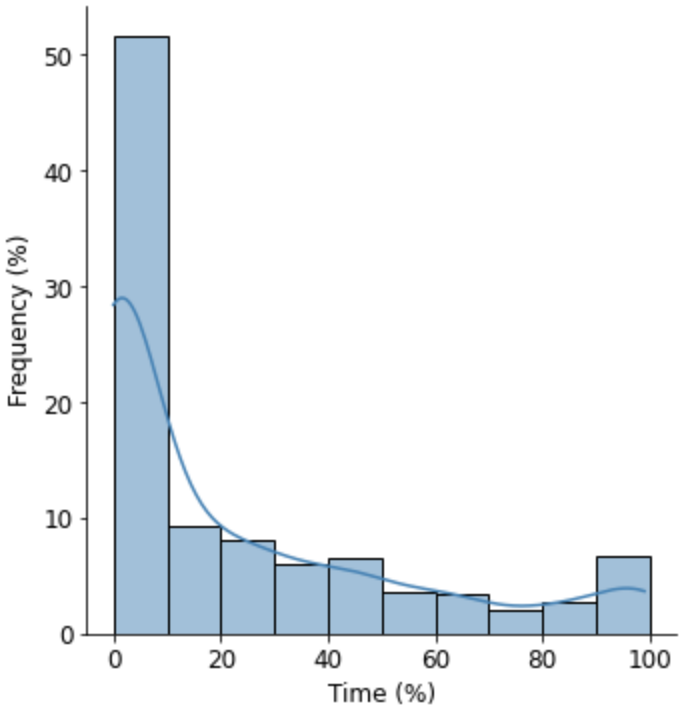

Figure 3 : How early can students be identified?

We can also visualize how distinguishable struggling students are from others. Figure 4 uses t-distributed stochastic neighbor embedding (t-SNE) to reduce the dimensionality of the training dataset from 7 dimensions to 2, allowing data visualization at four different points of course completion. Each circle represents a student in a course. As shown in the figure, early in the course, while most struggling and passing students are separable, there is still considerable overlap between the two classes. As the semester progresses, the overlap decreases, and the classes become much more separable, especially toward the end of the course. Although the shapes of the data distributions should not be overinterpreted, this visualization supports the idea that struggling students are not all detected at once, and emphasizes the need for a continuous, semester-long monitoring process.

Lastly, it is important to recognize that this dataset like this are unbalanced, with a significantly larger number of passing students compared to failing ones. This imbalance must be carefully considered during the training and evaluation of predictive models, as it can substantially affect the interpretation and usefulness of performance metrics.

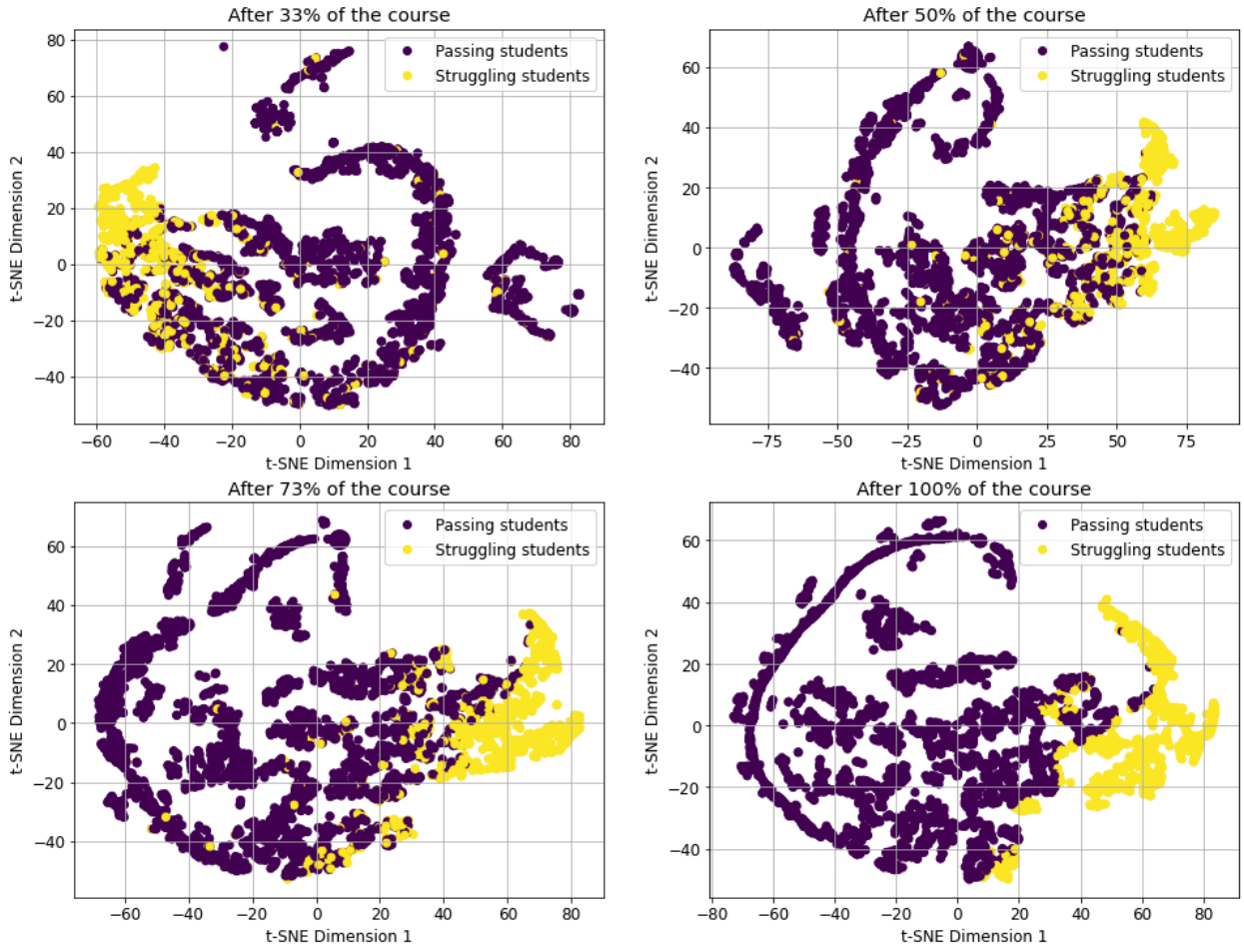

Figure 4 : Visualization of struggling and passing students.

Making Predictions

Previous research has shown that predictive models struggle to forecast student performance at the end of the semester [2]. However, they perform much better when, instead of predicting who will fail by the semester's end, they focus on identifying students who will be struggling 2 to 3 weeks from a given time point in the semester. To achieve this, the normalized time series representing students' journeys through courses are transformed into a dataset suitable for supervised machine learning modeling. Using two parameters,"starting-from" and "look-ahead", a new dataset is constructed by reading the aforementioned seven input features at the starting-from time point, while the relative achievement at the look-ahead time point is used to calculate the output class.

The "starting-from" and "look-ahead" parameters define the left and right boundaries of a time window, which is then slid one time step to the right to calculate the next data point. This process continues until the sliding window reaches the end of the semester. To ensure predictions are not

made prematurely, the sliding window process begins after a warm-up period, such as after the second week of the semester.

Using a starting-from value of 14 and a look-ahead value of 21, the resulting dataset consists of 455,975 data points, which are then split into training, validation, and testing subsets in a 60/20/20% ratio. Four models were trained using this data, and Table 2 presents these models along with their performance.

Table 2: Trained models' performance.

Model	**Accuracy**	**Recall**	**Precision**	**F1**
Random Forest	93.639	75.324	89.600	81.844
Logistic Regression	93.871	74.892	91.345	82.304
Multilayer Perceptron	93.752	77.409	88.319	82.505
Decision Tree	91.022	75.488	76.912	76.193

While these models are similar in their performance, the top-performing models are the Random Forest, Logistic Regression and Multilayer Perceptron, with accuracy values exceeding 93% and F1 scores above 81%. Although a higher accuracy value is important, one must be cautious not to overstate its significance, given the imbalance of the data. However, the table shows that predictive models can indeed be used to detect struggling students with good accuracy. Notably, all these models show higher precision values than recall. Ideally, we would prefer to see higher recall values, as higher recall indicates fewer false negatives—meaning fewer struggling students are misclassified as doing well.

The remainder of this section will focus on the Random Forest model for several reasons:

- A simple t-test using the F1 scores reveals that the performance of Random Forest is not statistically significantly different from the other models, with a p-value of 0.555.
- Random Forests do not require standardizing the input features before training, unlike Multilayer Perceptron. This reduces the number of steps to manage when deploying the model in production.
- Random Forests are more explainable than Multilayer Perceptrons, providing insight into which features are most important in making decisions. Figure 5 shows the feature importance produced by the Random Forest model in Table 2, with cumulative relative achievement and missed opportunity being the most important features.

The results in Table 2 and Figure 4 are based on a dataset generated with a look-ahead value of 21 (approximately 3.36 weeks for a 16-week semester). But how does the dataset and the performance of the models change when using different look-ahead values? Figure 6 answers this question by generating datasets with varying look-ahead values and training Random Forest models on these datasets. The figure shows that all performance metrics decline as the look-ahead value increases. This indicates that predictive models lose performance as they

attempt to predict further into the future, and that they perform significantly better when making short-term predictions (with lower look-ahead values).

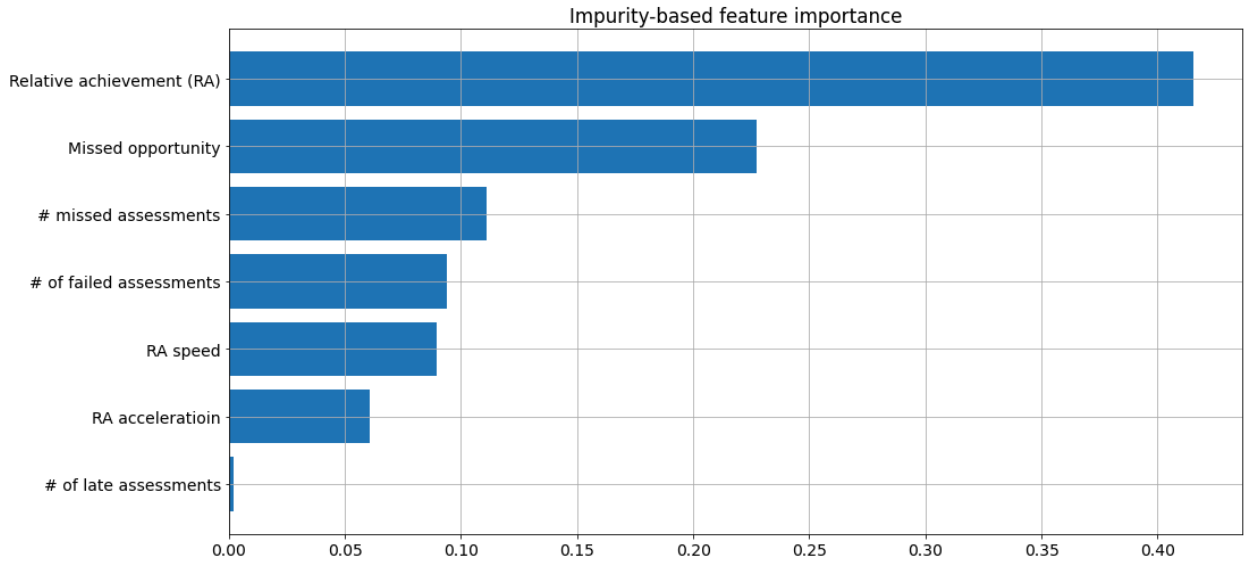

Figure 5 : Random forest feature importance.

Apart from the limitation in long-term predictions, these models have been shown to be portable and robust [13].

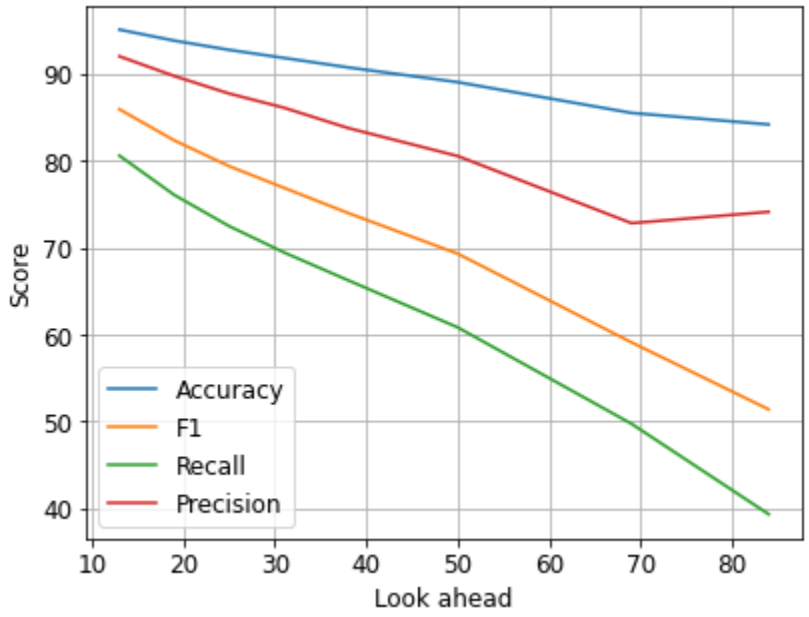

Figure 6 : The effect of the look ahead parameter.

Discussion

The results above suggest that having a one-time process for detecting struggling students is not as effective as having a process that runs regularly throughout the semester. The former process requires the models to make long-term predictions, whereas the latter relies on short-term predictions, for which the models are better suited.

We now outline a simple process by which these trained models can be used for short-term detection of struggling students. The process is as follows:

1. Starting 2 or 3 weeks into the semester (warm-up), download the data from the LMS using the appropriate API calls.
2. Convert the downloaded data into time series and normalize it by converting actual timestamps to percentages, with the entire course duration corresponding to 100%. The steps outlined under the Data Cleanup and Preparation subsection were chosen to be applicable to data from both in-progress and completed courses.
3. For every student in the course, calculate the cumulative input data and feed it into the model to get predictions about whether the student will be struggling or doing well at the look-ahead value of the model into the future.
4. Report the identified students to available remedial services.
5. Repeat steps 1-4 regularly (weekly, biweekly, etc.) until the end of the semester.

This process leverages the strengths of these models by running them regularly throughout the semester. It can be implemented by individual instructors or as part of a coordinated effort by a department, college, or university.

Concluding Remarks and Future Work

As shown in the previous sections, LMS data alone can be useful in identifying struggling students throughout the semester. Once properly processed, this data can be used to train machine learning models that make short-term predictions. However, the usefulness of these models should not be overstated. In other words, false negatives will still occur due to both the limitations of the models and, more importantly, the incompleteness of the data they were trained on. Despite its value, LMS data cannot account for all factors that may adversely affect student performance. Many such factors can arise at any point during the semester and, because they lie outside the LMS, are not captured by its data.

While these models can be improved with more complete data drawn from multiple sources beyond the LMS, access and practicality should always be considered before making such improvements. Since there is no perfect predictive model, the goal should be to improve performance, but 'better' should not be allowed to become the enemy of 'good.' In other words, if a better solution is harder to implement or out of reach for many instructors, it may not necessarily be the best option.

Little hyperparameter tuning was done on the models in this paper. Further efforts in this area should improve their performance.

While the results of this paper are based on lower-division CS courses, it is reasonable to assume that they may also apply to other CS and non-CS courses. However, it remains an open question whether and how these findings will generalize to other non-CS courses.

In summary, this paper demonstrated how time series sequences of graded activities can provide insights into student progression through courses. It evaluated various machine learning models for detecting struggling students. These models are better suited for making short-term predictions rather than long-term ones, and a process for utilizing these models throughout the entire semester in a production setting is outlined. Such models and processes can be crucial for higher education institutions in providing timely support to struggling students, thereby improving learning outcomes and student retention.

Bibliography

[1] R. Umer, A. Mathrani, T. Susnjak and S. Lim, "Mining Activity Log Data to Predict Student's Outcome in a Course," in Proceedings of the 2019 International Conference on Big Data and Education, New York, NY, USA, 2019.

[2] S. V. Goidsenhoven, D. Bogdanova, G. Deeva, S. v. Broucke, J. D. Weerdt and M. Snoeck, "Predicting Student Success in a Blended Learning Environment," New York, NY, USA: Association for Computing Machinery, 2020.

[3] P. Shayan and M. v. Zaanen, "Predicting Student Performance from Their Behavior in Learning Management Systems," International Journal of Information and Education Technology, vol. 9, no. 01, pp. 337-341, 2019.

[4] 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 Transactions on Learning Technologies, vol. 10, no. 01, pp. 17-29, 2017.

[5] 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," The Internet and Higher Education, vol. 28, pp. 68-84, 2016.

[6] M. Riestra-Gonza´lez, M. d. P. Paule-Ruíz and F. Ortin, "Massive LMS log data analysis for the early prediction of course-agnostic student performance," Comput. Educ., vol. 163, pp. 104-108, 2021.

[7] S. B. Dias, S. J. Hadjileontiadou, J. Diniz and L. J. Hadjileontiadis, "DeepLMS: a deep learning predictive model for supporting online learning in the Covid-19 era," Scientific Reports, vol. 10, no. 1, p. 19888, 2020.

[8] A. Al-Gahmi, K. D. Feuz, Y. Zhang, "On Time-based Exploration of LMS Data and Prediction of Student Performance,". 2022 ASEE Annual Conference & Exposition, Minneapolis, MN, June 2022, 10.18260/1-2--40852

[9] A. Al-Gahmi, K. D. Feuz, & Y. Zhang, "On Time-based Exploration of Student Performance Prediction", 2023 ASEE Annual Conference & Exposition, Baltimore , Maryland, June 2023, 10.18260/1-2--43772

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg and J. Vanderplas , "Scikit-learn: Machine Learning in Python," The Journal of Machine Learning Research, vol. 12, p. 2825–2830, 2011.

[11] S. Lundberg, S. Lee,"A Unified Approach to Interpreting Model Predictions". Advances in Neural Information Processing Systems 30, NIPS 2017.

[12] T. Dietterich, "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms," Neural Comput 10, p. 1895–1923, 1998.

[13] A. Al-Gahmi, "On the Portability and Robustness of Early Student Performance Predictions", 2024 ASEE Annual Conference & Exposition, Portland , Oregon, June 2024, 10.18260/1-2--47811

[14] P. Simon, J. Jiang. L. Fryer, R. King, and C. Frondozo. "An Assessment of Learning Management System Use in Higher Education: Perspectives from a Comprehensive Sample of Teachers and Students. Technology, Knowledge and Learning," 10.1007/s10758-024-09734-5, 2024.

[15] I. Bouchrika. "51 LMS statistics: 2025 data, trends & predictions," Research.com. Retrieved January 10, 2025, from https://research.com/education/lms-statistics, 2025.

[16] Y. Liu, S. Fan, S. Xu, A. Sajjanhar, S. Yeom and Y. Wei, "Predicting Student Performance Using Clickstream Data and Machine Learning," Education Sciences, 13, 1 (2022), 17.

[17] J. Gamulin, O. Gamulin and D. Kermek, "Using Fourier coefficients in time series analysis for student performance prediction in blended learning environments," Expert systems, 33, 2 (2016), 189-200.

[18] J. Liu, C. Yin, K. Wang, M. Guan, X. Wang and H. Zhou, "Students' Course Results Prediction Based on Data Processing and Machine Learning Methods," Journal of Signal Processing Systems, 94, 11 (2022), 1199-1211.

[19] A. Mitrovic and T. Wang, "Using Neural Networks to Predict Student's Performance," in Computers in Education, International Conference on, Auckland, New Zealand, 2002 pp. 969.