ISBN: 978-628-95207-8-1. ISSN: 2414-6390. Digital Object Identifier: 10.18687/LACCEI2024.1.1.1412

# Predicting Student Performance in a Non-linear Self-Paced Moodle Course

Jhon Mercado*, Carlos Mendoza-Cardenas†, Natalia Gaviria-Gomez*,
Juan F. Botero*, and Luis Fletscher*
**ETE Department*, *Universidad de Antioquia*, Medellín, Colombia
†*Twitch Interactive*, San Francisco, United States
Emails: {jfredy.mercado, natalia.gaviria, juanf.botero, luis.fletscher}@udea.edu.co

***Abstract*—Although student performance prediction in online courses has been extensively studied before, previous research efforts have focused mostly on courses with a linear structure, where the student is expected to take the lessons and assessments in a sequential (linear) and unique order. There are, however, non-linear courses where the student can take the lessons and assessments in any order they wish, which makes the performance prediction more challenging, as the cumulative assessment percentage might vary widely between students at a given point in time. Here, we present a data-driven method to predict student performance in non-linear courses. In particular, our method predicts whether or not a student will reach a final g rade a bove a fi xed th reshold fo r di fferent percentages of assessment completion. We use data from a Moodle course designed to prepare high-school students for the entrance exam of a public university. We show that for cumulative assessment percentages $\geq$60%, our method has a mean F1-score above 70%. Finally, we also assess the importance of each feature used in the prediction, illustrating the effect of our data preprocessing and feature selection approach on the model performance. Our future research efforts will focus on applying our predictive models in real-world scenarios, particularly within educational settings where learning resources dynamically adapt based on student performance.**

***Index Terms*—Learning Management System (LMS), Student Performance Prediction, Machine Learning.**

## I. Introduction

The use of information and communication technologies in educational institutions has grown consistently. Of these technologies, the Learning Management System (LMS) has become increasingly important [1]. These systems act as dynamic platforms that facilitate the delivery, documentation, monitoring, and administration of educational courses in virtual environments [2]. They are instrumental in reshaping virtual and face-to-face courses by providing tools for content management, communication, assignment submission and evaluation, online quizzes, and student grading [3].

Despite their general use, virtual platforms, especially LMSs, face a significant c hallenge: l ow c ompletion rates. Research indicates that completion rates often fall below 13%, with only 2-10% of students achieving course objectives and earning certificates [4]. Student dropout has attracted considerable research attention, giving rise to various explanations and proposed solutions. Some attribute it to factors such as lack of motivation, commitment, and initial intention among students [5]. In contrast, others point to the need for high autonomy, which may lead to feelings of isolation among participants [6]. Various adaptive, personalized, or recommendation approaches have been suggested to mitigate dropout rates, considering factors such as learning path [7], prior knowledge [8], skills [9], and learning style [10].

The early prediction of student performance in an online course can be of great help for teachers in identifying students at risk of dropping out and implementing retention and academic support strategies. Although several student performance prediction methods have been proposed in the literature [11]–[19], the data used in those works usually comes from courses where the students are expected to take the lessons and assessments in a sequential manner from the beginning to the end of the course. In contrast, our work focuses on non-linear courses, which are courses where the students can take the lessons and assessments in any order they want [20]. Since not all the course assessments might have the same weight, these non-linear courses bring a new challenge in the early prediction of student performance, as the percentage of assessment completed at any given point in time could vary widely between students.

We present here a methodology that includes feature engineering, automatic feature selection, and the training of a binary classifier to predict if a student will reach a certificate-granting grade. We use data from the course "Prepárate para la Vida Universitaria" (PPVU), a course hosted in the LMS Moodle and designed by the Universidad de Antioquia to prepare high-school students for its entrance exam. PPVU is a non-linear course that empowers students with complete freedom to navigate its content at their own pace and in the desired order. The data corresponds to Moodle logs from 580 students and contains interactions of the students with the course, together with their grades. As we will see later, the nature of this course demands a tailored predictive model that captures student engagement and performance.

## II. Student Performance Prediction

The problem of predicting a student's performance in an LMS course has several aspects that need to be considered. First, one needs to define whether the problem one wants to solve is either a regression or classification task. In regression, the predictions are real numbers, like the student's final grade. In classification, the predictions are two or more student statuses, such as at-risk or passing. In this paper, we solve a classification problem, predicting whether a student will be granted a certificate of participation or not based on a final grade threshold.

The second aspect is the features used to predict, which can be classified as demographic, performance, or engagement. Demographic features pertain to population characteristics, including gender, age, ethnicity, nationality, and socioeconomic status (e.g., parental occupations or household income). Performance features encompass study-related performance measures such as grades, passes/fails, and assignment percentages. Engagement features reflect a student's level of engagement with their studies. They may include participation during class, hours spent on online learning resources, participation in forums, time spent on evaluative activities, and the number of attempts. In our work, we consider only performance and engagement features. We decided not to consider demographic features because they might introduce biases to the model based on categories like gender or age and have not shown substantial predictive power [15].

The third aspect of this problem relates to how interpretable the prediction is. Although there is not a universal consensus of what the term *interpretability* means [21], [22], here we define it as the ability to measure how important is each input feature in the prediction. We believe this is an important property of the model, as it could help teachers identify patterns of student behavior that are more correlated with low predicted performance and thus make more effective academic interventions. With that definition of interpretability in mind, we included linear regression and support vector machines in the set of models we evaluated. We used the value of the model parameters (weights) as a direct way to measure the importance of each feature.

A fourth aspect of student performance prediction is the method used to define *when* to make the prediction. We call that time the *prediction time point* (PTP). This aspect is important in cases where, for example, the teacher wants to apply an intervention that could help students at risk of dropping out, and therefore, an early prediction is required. As we will see below, some authors define the PTP through a percentage of the course length or a cumulative number of assessments taken by the student, among others. As we will detail in section III, we define the PTP through something that we call the cumulative weight of assessments (CWA), which is the sum of the percentages (weights) of all the assessments that a student has taken up to a given point in time.

Finally, a fifth and usually less considered aspect of this problem is the nature of the course. The usual type of course considered in the literature is one where the course is designed to be followed in a linear order, like traditional college courses. We are concerned with courses where students can take the lessons and assessments in any order they like, something we call a "non-linear" course.

## III. Related Works

The use of demographic features, in addition to inducing potential biases in the model, might not provide significant predictive value. Tomasevic et al. used demographic, performance, and engagement features [13]. They solved a regression and a classification problem using data from two courses of the Open University Learning Analytics Dataset (OULAD) [14] and different machine learning methods like K-Nearest Neighbors, Support Vector Machine, and Logistic Regression. In the regression problem, they created models for predicting student test scores. For the classification problem, they predict whether a student will fail or pass the course. They tested the models during mid-term and final assessments, and the F1-score increased gradually over time. The F1-score for the first assessment was 78%, while the sixth assessment had an F1-score of 94.9%. The neural network models produced the best results, with an accuracy of 96.6% before the final exam. They highlight that the usage of demographic data did not significantly influence the precision of predictions. A recent study [15] also used the OULAD and other datasets and found strong evidence that including demographic features does not lead to better-performing models as long as some study-related features exist, such as performance or activity data.

To perform early student performance prediction, researchers have used different ways to define the PTP, i.e., the point in time where the prediction is made. Riestra-González et al. predict students' performance at 10%, 25%, 33%, and 50% of the course length [16]. Their objective is to detect at-risk, failed, and excellent students in the early stage of the course, creating different classification models. In contrast, in [13], [17], they define the PTP based on the cumulative number of student assignments.

To give interpretability to the results, [16] used feature agglomeration and cluster analysis to detect six different clusters of how the students interact with the LMS. They found that those interaction patterns of each cluster are repeated in all the initial stages of the course, finally showing how four of those six patterns of student interaction with the LMS have a strong correlation with student performance. In contrast, [17], [18] uses the SHapley Additive exPlanations (SHAP) method [23] to select the most important features in predicting students' performance. In [17], they classify students into four groups based on their performance. These groups include experts with high scores and minimal incorrect submissions, learning students with many incorrect submissions and low average scores, struggling students with low scores and minimal incorrect submissions, and outliers with high scores and a high number of incorrect submissions.

Although previous studies proved effective in student performance prediction tasks, we identified different challenges associated with these studies. Static features, like demographics, fail to capture the dynamic behavior of students during their learning process. The studies carried out with the early prediction of student performance are linear courses, where it is possible to know on which dates to make the prediction based on the course's duration or the mid-term assessment, unlike the non-linear course used in this studio. Furthermore, neural network models are becoming popular with time; however, it is challenging to achieve good performance with these models when trained on small datasets [19], and their prediction is not easily interpretable. Finally, we acknowledge that, as is also the case in previous works, the applicability of the specific performance and engagement features used in this paper is limited, as those features are not generalizable in different courses since they differ in the course outline and interim exams can vary from one version of the course to the next one.

## IV. Methodology

Fig. 1 shows the high-level flow of the data in our method, from the interaction of the students to the prediction of their performance. Here, interactions are specifically captured through clickstream data within the Moodle platform and include events like clicks on course materials, forum participation, assignment submissions, and other platform-specific activities.

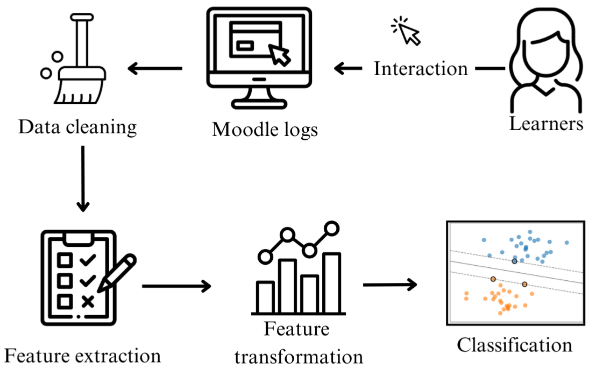

Fig. 1. Overview of the proposed method.

### A. Data cleaning

We first start by extracting only the Moodle logs that are related to student interactions and student grades and removing Moodle columns that are not relevant (e.g., the `objecttable` column in the `logstore_standard_log` table), ensuring that only data essential to our analysis is retained [24].

### B. Feature extraction

In this stage, we extract and compute features from the Moodle logs that are known to be relevant to the student performance [13]. Since we are interested in early prediction, we compute each feature by aggregating the activity of the students at different PTPs. Since the course allows students to progress through activities at their own pace, repeat evaluative tasks as desired, and take the lessons and assessments in any order they wish, not all the students will have the same CWA. For this reason, we define a PTP as a point in time where the maximum CWA across all students is less than or equal to a given threshold. To better illustrate this procedure, Fig. 2 presents an example wherein the CWA of various students is depicted. At the end of the course, some students complete the 100% of the assessments, while others do not. In the example, the CWA threshold used to compute the features is 50%. The selected PTP ($t_3$) corresponds to the point where a student (*Student 3*) achieved the highest CWA that is less than or equal to 50%. If, for example, we were to aggregate the weight of the next assessment that *Student 4* took, their CWA will go from 40% to a value that is over 50%.

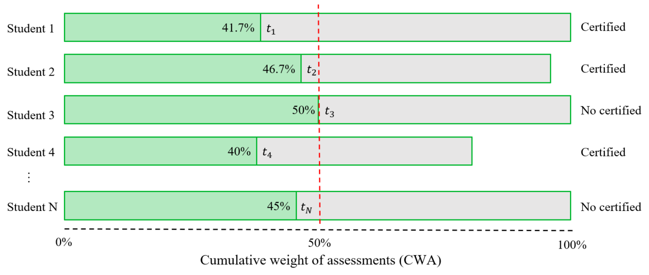

Fig. 2. Example of adaptive feature update strategy: student progression, cumulative weight of assessments, and certification status.

### C. Feature transformation and filtering

After the features have been computed for a given CWA, they are normalized using z-score normalization to have zero mean and unit variance. This normalization helps create a more symmetric loss surface and accelerate training [25]. Finally, we use two feature selection filters: Variance Threshold (0.01) to remove low-variance features and Correlation Threshold (0.9) to eliminate highly correlated features, ensuring a more informative and diverse feature set [26].

### D. Prediction

Let $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ be a dataset of $n$ students, with $\mathbf{x}_i \in \mathbb{R}^d$ being the vector of $d$ normalized features of the $i$-th student, and $y_i \in \{0, 1\}$ a label that is 1 when, in a scale of 0 to 5, the student's final grade is less than to 3.5, and 0 otherwise. Our goal is to learn a binary classifier $f_\theta : \mathbb{R}^d \to \{0, 1\}$ such that $\hat{y} = f_\theta(\mathbf{x})$ is the predicted label for $\mathbf{x}$ [27].

## V. Data

The data comes from the Moodle tables of students who took the course on logic reasoning of PPVU. The course is estimated to last 48 hours, spread across eight weeks, with a recommended study time of 6 hours per week. The course has five units and 12 evaluative assessments in total. Each unit has videos explaining specific topics and modules[1]

[1]These modules come in two different presentation formats: (a) a lesson, which is a type of Moodle activity, and (b) a multimedia built using the Genially platform. These two formats are considered separately in the definition of the features (see Table I).

explaining solved problems. The 12 evaluative assessments include 6 workshops that evaluate the knowledge acquired. Each workshop contributes $5\%$ to the overall evaluation, and by 6 simulations that imitate the structure and time restrictions of the university entrance exam, each simulation contributes $11.67\%$ to the overall evaluation.

The course also has two diagnostic exams that are meant to assess the student's prior knowledge and do not weigh in the final grade. Those exams have questions about the different topics offered by the course, and their results are used to suggest the units that the student should study.

One of the particularities of the course is the freedom that students have to take the course in the order they want and in the time they want, being able to repeat the different evaluative activities as many times as they like, in the order that they like, having as the final grade for each evaluative assessment the best grade of all attempts. The course delivers a certificate of participation to those students who achieve a final grade higher or equal to 3.5, being 0 the minimum grade and 5 the maximum grade.

Figure 3 shows the number of students who completed at least a given percentage of the overall assessment of the course, where the total enrolled students was 17108. There is a fast decay at the beginning[2] of the course, with more than half of the students that achieved at least a CWA of 5% dropping out, relative to those that achieved at least a CWA of $20\%$. This shows the importance of developing predictive models that can accurately identify at-risk students in those early stages of the course. To have enough logged interactions for training, we use the data from the 580 students who completed at least 65% of the overall assessment (highlighted with the dotted red line in Fig. 3).

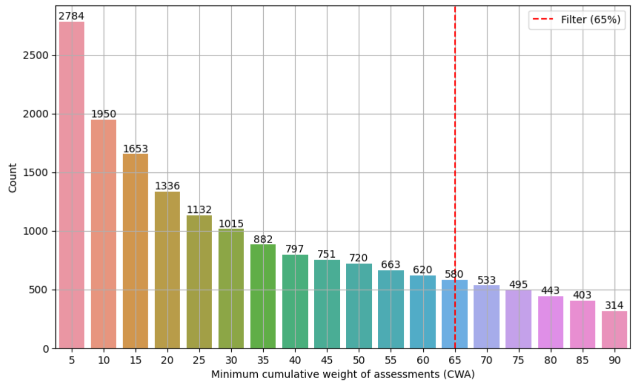

Fig. 3. Distribution of students by percentage assessment completion

## A. Features

As was discussed in sections IV-A and IV-B, we clean the raw data present in Moodle tables and extract the features that will be used to train our classifier. Table I shows the 41 engagement and performance features that we extracted, with the performance features being the last 17 rows in the second column of features of the table. Recall from section IV-B that the features are computed by aggregating the student activity up to the point where their CWA is less than or equal to a given threshold. Consequently, the distribution of each feature will change depending on the CWA threshold. As an example, we show in figure 4 the distribution of the feature `mean_multimedia_views`, for two different CWA thresholds. As expected, as the threshold increases from $20\%$ to $60\%$, the distribution of the feature becomes less concentrated around zero, and the variance increases.

[2]"Beginning" in this context means the first evaluative activities taken, as the course can be followed in any order the student wants.

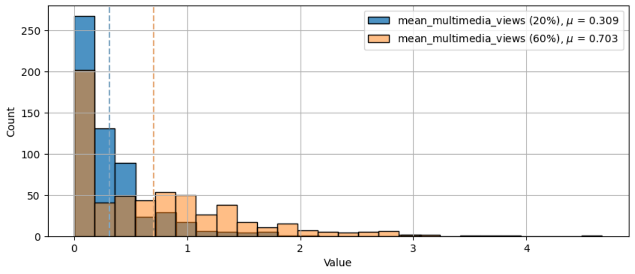

Fig. 4. Distribution of the mean number of multimedia views for CWA thresholds of 20% and 50%.

## B. Target variable

The course gives a certificate of participation to students who achieve a final grade of 3.5 or higher on a scale of 0 to 5. The final grade is computed using the highest grade achieved on each evaluative activity and then computing the weighted average. As mentioned in section IV-D, the target variable, $y$, takes the value 1 if the student did not achieve a certificate of participation and 0 otherwise. Of the 580 students in our dataset, 233 students did not achieve a certificate of participation ($y = 1$), and 347 students did ($y = 0$).

## VI. Experiments and Results

We performed two experiments. In our first experiment, we trained and evaluated a binary classifier for a given CWA threshold. In our second experiment, we trained the classifier for one CWA threshold and then used it to make predictions at other CWA thresholds for which the model was not trained. We start by defining some elements of our experimental design that are common to both experiments and then present the details of each experiment and the results we found, including an analysis of feature importance.

## A. Heuristic baseline

We believe it can be informative to consider a naive non-machine-learning baseline that uses only the grades to predict student performance. As such, we use the weighted average grade (WAG) that the student has at a given CWA threshold, predicting that the student does not receive a certificate of participation ($\hat{y} = 1$) if their WAG is less than 3.5, and that it does ($\hat{y} = 0$) otherwise.

TABLE I
FEATURES EXTRACTED FROM MOODLE.

| Feature | Description | Feature | Description |
|---|---|---|---|
| course_navigation | Number of accesses to the course units | std_lessons_attempts | Std of lessons attempts |
| forum_viewed | Number of access to the forum | mean_lessons_time | Mean of time spent in lessons |
| post_created | Number of post created | std_lessons_time | Std of time spent in lessons |
| discussion_viewed | Number of access to the discussion | mean_prior_knowledge_grade | Mean prior knowledge grade |
| discussion_created | Number of discussions created | std_prior_knowledge_grade | Std prior knowledge grade |
| forum_searched | Number of searches in the forum | mean_prior_knowledge_time | Mean time spent in prior knowledge |
| outline_viewed | Number of accesses to the course outline | std_prior_knowledge_time | Std time spent in prior knowledge |
| week | Number of interactions per week | mean_workshops_grade | Mean workshop grade |
| weekend | Number of interactions per weekend | std_workshops_grade | Std workshop grade |
| days | Number of days on the platform | mean_workshops_time | Mean time spent in workshops |
| mean_interactions | Mean of interactions generated per day | std_workshops_time | Std of time spent in workshops |
| std_interactions | Std of interactions generated per day | mean_workshops_attempts | Mean of workshops attempts |
| mean_multimedia_views | Mean of multimedia views | std_workshops_attempts | Std of workshops attempts |
| std_multimedia_views | Std of multimedia views | mean_simulations_grade | Mean of simulations grade |
| mean_multimedia_time | Mean of time spent in multimedia | std_simulations_grade | Std of simulations grade |
| std_multimedia_time | Std of time spent in multimedia | mean_simulations_time | Mean of time spent in simulations |
| mean_video_views | Mean of video views | std_simulations_time | Std of time spent in simulations |
| std_video_views | Std of multimedia views | mean_simulations_attempts | Mean of simulations attempts |
| mean_video_time | Mean of time spent in videos | std_simulations_attempts | Std of simulations attempts |
| std_video_time | Std of time spent in videos | weighted_average | Weighted average |
| mean_lessons_attempts | Mean of lessons attempts | | |

### *B. Machine learning models and training procedure*

We evaluated the following machine learning algorithms: Gradient Boosting (GB) [28], K-Nearest Neighbors (KNN) [29], Logistic Regression (LR) [29], Random Forest (RF) [29], and Support Vector Machine (SVM) [29]. For the training and evaluation of the models, we use a nested cross-validation, with a 4-fold cross-validation for the inner loop and a stratified 10-fold split for the outer loop. In the inner loop, we search for the best hyperparameters of the model, while in the outer loop, we estimate the generalization error of our training methodology [30]. Table II shows the set of values used for each hyperparameter and each algorithm.

TABLE II
HYPERPARAMETERS OPTIMIZED USING THE GRID-SEARCH TECHNIQUE

| Classifier | Hyperparameters | Values’ interval |
|---|---|---|
| **GradientBoosting** | n_estimators | [50, 100, 150, 200, 250, 300, 350, 400] |
| | learning_rate | [0.01, 0.1, 0.2] |
| | max_depth | [3, 4, 5, 6] |
| | min_samples_split | [2, 3, 4] |
| | min_samples_leaf | [1, 2, 3] |
| | max_features | [’sqrt’, ’log2’, None] |
| **KNN** | n_neighbors | [3, 5, 7, 9, 11, 13] |
| | weights | [’uniform’, ’distance’] |
| | p | [1, 2] |
| | leaf_size | [10, 20, 30] |
| | algorithm | [’auto’, ’ball_tree’, ’kd_tree’, ’brute’] |
| | metric | [’minkowski’, ’euclidean’, ’manhattan’] |
| **LogisticRegression** | penalty | [None, ’l1’, ’l2’] |
| | C | [0.01, 0.1, 1, 10, 100] |
| | class_weight | [None, ’balanced’] |
| | solver | [’liblinear’, ’lbfgs’, ’newton-cg’, ’sag’, ’saga’] |
| | max_iter | [100, 200, 300] |
| **SVM** | C | [0.01, 0.1, 1, 10, 100] |
| | kernel | [’linear’, ’rbf’] |
| | gamma | [0.01, 0.1, 1, 10, 100] |
| | class_weight | [None, ’balanced’] |
| **RandomForest** | n_estimators | [100, 150, 200, 250, 300] |
| | max_depth | [None, 10, 20, 30] |
| | min_samples_split | [2-20] |
| | min_samples_leaf | [1, 2, 3, 4] |
| | max_features | [’auto’, ’sqrt’, ’log2’, None] |
| | class_weight | [None, ’balanced’, ’balanced_subsample’] |

### *C. Performance measures*

Using machine learning terminology, we define a *positive* example as the student that does not achieve a certificate of participation ($y = 1$) and a negative example as the student that does ($y = 0$). Since we care more about minimizing the prediction errors for positive examples, we use recall, precision, and the F1-score to measure the classifier’s performance.

Precision is the fraction of examples that were correctly classified as positive (true positives) out of all the examples that were classified as positive (true and false positives); it measures how accurate the model is when it predicts that a student will not get a certificate. Recall is the fraction of positive examples that were correctly classified as positive; it measures how good the model is at identifying (recovering) the students who are at risk of not getting a certificate. Finally, the F1-score is the harmonic mean of precision and recall and provides a good balance of what the two metrics are measuring.

### *D. First experiment: Individual models*

For our first experiment, the aim is to train different models at different CWA to see the evolution of the models at the moment of predicting the performance of students, we used the 20%, 40%, 60%, and 80% CWA thresholds to train and evaluate the different classifier algorithms. For each model, the training and test data are both computed at the same CWA threshold (see section IV-B). Table III shows that all the models, with the exception of KNN, outpeform the baseline in terms of the F1-score. Although Logistic Regression and Random Forest rival as the top-performing models, the results from Logistic Regression are quite remarkable, given that its computational complexity is much lower than Random Forest[3], and its interpretability is much higher.

TABLE III
SUMMARY OF RESULTS AT DIFFERENT ASSESSMENT PERCENTAGES OF THE COURSE

| Model | Assessment Percentage | Recall ($\mu \pm \sigma$) | Precision ($\mu \pm \sigma$) | F1-score ($\mu \pm \sigma$) |
|---|---|---|---|---|
| **Baseline** | 20% | 0.57 ± 0.12 | 0.65 ± 0.09 | 0.60 ± 0.08 |
| | 40% | 0.60 ± 0.13 | 0.79 ± 0.06 | 0.68 ± 0.10 |
| | 60% | 0.60 ± 0.14 | 0.87 ± 0.08 | 0.70 ± 0.11 |
| | 80% | 0.61 ± 0.12 | 0.92 ± 0.07 | 0.73 ± 0.09 |
| **GradientBoosting** | 20% | 0.58 ± 0.10 | 0.69 ± 0.10 | 0.63 ± 0.08 |
| | 40% | 0.64 ± 0.15 | 0.73 ± 0.05 | 0.67 ± 0.10 |
| | 60% | 0.69 ± 0.14 | 0.80 ± 0.06 | 0.73 ± 0.07 |
| | 80% | 0.74 ± 0.15 | 0.81 ± 0.07 | 0.77 ± 0.11 |
| **KNN** | 20% | 0.51 ± 0.11 | 0.66 ± 0.12 | 0.57 ± 0.10 |
| | 40% | 0.56 ± 0.10 | 0.75 ± 0.09 | 0.64 ± 0.09 |
| | 60% | 0.62 ± 0.10 | 0.78 ± 0.08 | 0.69 ± 0.08 |
| | 80% | 0.62 ± 0.13 | 0.78 ± 0.09 | 0.69 ± 0.10 |
| **LogisticRegression** | 20% | 0.63 ± 0.13 | 0.65 ± 0.08 | 0.64 ± 0.09 |
| | 40% | 0.72 ± 0.14 | 0.75 ± 0.06 | 0.72 ± 0.08 |
| | 60% | 0.76 ± 0.18 | 0.78 ± 0.08 | 0.76 ± 0.11 |
| | 80% | 0.76 ± 0.13 | 0.81 ± 0.07 | 0.78 ± 0.10 |
| **SVM** | 20% | 0.63 ± 0.12 | 0.64 ± 0.07 | 0.63 ± 0.07 |
| | 40% | 0.66 ± 0.14 | 0.73 ± 0.03 | 0.69 ± 0.08 |
| | 60% | 0.72 ± 0.13 | 0.77 ± 0.08 | 0.74 ± 0.07 |
| | 80% | 0.76 ± 0.13 | 0.79 ± 0.06 | 0.77 ± 0.07 |
| **RandomForest** | 20% | 0.64 ± 0.10 | 0.68 ± 0.08 | 0.66 ± 0.08 |
| | 40% | 0.70 ± 0.12 | 0.74 ± 0.06 | 0.71 ± 0.07 |
| | 60% | 0.73 ± 0.11 | 0.79 ± 0.08 | 0.75 ± 0.06 |
| | 80% | 0.78 ± 0.11 | 0.82 ± 0.04 | 0.80 ± 0.07 |

Although most machine learning models consistently outperformed the baselines, KNN behaved differently. The poor performance of KNN could be due to the large number of features we used, as the difference in the distances between data points becomes smaller as the dimensionality of the data increases [31].

Recall that the data we used comes from students who completed at least 65% of the overall assessment. We trained and tested the classifier at other WAG thresholds not shown in Table III. Interestingly, during these evaluations, we observed a notable decline in baseline performance for thresholds greater than 65%. Despite achieving a WAG that would warrant a certification at 65%, our analysis revealed that some students abandon the course after reaching that percentage. Consequently, this leads to a WAG that falls short of the certification threshold when the course ends.

## E. Second experiment: Single model

In our second experiment, as having different models for training has higher computational costs a difference from just using a single model [32], we aim to investigate the outcome of training a single model at a given CWA threshold and then using it to make predictions at different CWA thresholds. For this purpose, we trained a Logistic Regression classifier using features computed at 60% CWA. Let us refer to this model as *SingleModel@60*. We are interested in comparing the performance of SingleModel@60 with the baseline and with a model (also a Logistic Regression classifier) that was trained for the specific CWA threshold we want to predict at. Let us denote by *IndividualModel@k* a model that was trained at k% CWA, and it is used to predict at that same CWA threshold. Note that SingleModel@60 is the same model as IndividualModel@60.

[3]We observed that the training time for Logistic Regression was in the order of minutes, while for Random Forest was in the order of hours.

Since we use nested cross-validation to train and validate our models, we need to pick a fold in the 10-fold outer loop to compare the performance of all three models on the same data. We select the fold where the SingleModel@60 has the highest performance. During inference, we compute the features for the students that are in that fold at the 20%, 40%, 60%, and 80% CWA thresholds and use SingleModel@60 to predict on that data. For IndividualModel@k, we train one model for each CWA threshold and make predictions on the same threshold, using the same fold (data split) as SingleModel@60. Finally, we apply the baseline in the same test data used by SingleModel@60 and IndividualModel@k across the four CWA thresholds.

Table IV shows the performance of each approach. As expected, using a single model for different CWAs leads to an overall decrease in performance, measured by the F1-score. Furthermore, there is a significant decrease in precision at early stages of the course (20% and 40% CWA).

TABLE IV
COMPARATIVE RESULTS OF DIFFERENT APPROACHES AT DIFFERENT ASSESSMENT PERCENTAGES OF THE COURSE.

| Approach | Assessment Percentage | Recall | Precision | F1-score |
|---|---|---|---|---|
| **Baseline** | 20% | 0.69 | 0.76 | 0.72 |
| | 40% | 0.65 | 0.88 | 0.75 |
| | 60% | 0.65 | 1.00 | 0.79 |
| | 80% | 0.57 | 1.00 | 0.72 |
| **SingleModel@60** | 20% | 0.83 | 0.45 | 0.58 |
| | 40% | 0.83 | 0.63 | 0.72 |
| | 60% | 0.70 | 0.84 | 0.76 |
| | 80% | 0.70 | 0.84 | 0.76 |
| **IndividualModel@k** | 20% | 0.70 | 0.73 | 0.71 |
| | 40% | 0.78 | 0.78 | 0.78 |
| | 60% | 0.70 | 0.84 | 0.76 |
| | 80% | 0.83 | 0.76 | 0.79 |

## F. Feature Selection

As the values of the features change in each CWA, we studied how different CWA thresholds affect feature selection when analyzing student behavior. Specifically, we considered four CWA thresholds: 20%, 40%, 60%, and 80%. Our feature selection process, described in Section IV-C, involved removing low variance and highly correlated features. Table V presents the results of the features eliminated at each CWA threshold.

At the 20% CWA threshold, we removed features related to the standard deviation of simulation grade, time, and attempts. This happens because, at that threshold, students could take at most one simulation-type assessment, which contributes to their low variance.

At the 40% CWA threshold, we removed the feature `discussion_viewed` because it is highly correlated with `forum_viewed`. Similarly, at the 60% and 80%

CWA thresholds, we removed `discussion_viewed` and `mean_video_views`. At these thresholds, the feature `mean_video_views` is highly correlated with `mean_multimedia_views`.

TABLE V
FEATURES ELIMINATED AT DIFFERENT CWA THRESHOLDS

| Assessment Percentage | Features |
|---|---|
| **20%** | std_simulations_grade |
| | std_simulations_time |
| | std_simulations_attempt |
| **40%** | discussion_viewed |
| **60%** | discussion_viewed |
| | mean_video_views |
| **80%** | discussion_viewed |
| | mean_video_views |

## G. Feature Importance Analysis

We use Logistic Regression to study the importance of each of the features in the final prediction because the weights or coefficients of the model provide to us a direct measure of such importance. The coefficients signify both the strength and direction of the relationship between individual features and the final probabilities of each class. To visualize the relative importance of each feature in the prediction, we captured the absolute values of the coefficients.

In Table VI, we present the top and bottom 5 feature importance coefficients. As expected, `weighted_average` is the feature with the highest importance across all CWA thresholds, as intuitively the cumulative grade is highly correlated with the final grade that the student needs to achieve the participation certificate. It is also interesting to note that `mean_workshops_time` is the second most influential feature in the prediction in 3 of the 4 CWA thresholds. Since this feature relates to the *behavior* of the student during an evaluative activity, this results shows the potential that this kind of feature importance analysis could have in providing valuable insights to teachers about the relationship between student behaviour and performance.

Table VII reveals engagement's top and bottom 5 feature importance coefficients. In contrast to features related to performance, the coefficients are relatively lower, indicating a lower impact on the prediction. However, it is informative to note that `mean_multimedia_views` or `mean_video_views` appear consistently in the top 5 features across all the CWA thresholds. The importance of the student engagement with video and multimedia for performance prediction highlights the pedagogical value that such type of content can provide.

Conversely, the bottom 5 features exhibit notably smaller coefficient values, with features related to forum participation showing minimal impact on the prediction. This implies that student engagement in forum activities, as captured by `discussion_created`, `post_created`, and `forum_searched`, has limited value in predicting student performance.

TABLE VI
TOP AND BOTTOM 5 PERFORMANCE FEATURES: IMPORTANCE ACROSS DIFFERENT CWA THRESHOLDS

| Assessment Percentage | Top 5 features | Coef | Bottom 5 features | Coef |
|---|---|---|---|---|
| **20%** | weighted_average | 0.731 | std_workshops_grade | 0.002 |
| | mean_workshops_time | 0.313 | mean_prior_knowledge_grade | 0.003 |
| | mean_simulations_grade | 0.057 | std_prior_knowledge_grade | 0.003 |
| | mean_workshops_grade | 0.044 | mean_prior_knowledge_time | 0.007 |
| | mean_simulations_time | 0.037 | std_prior_knowledge_time | 0.010 |
| **40%** | weighted_average | 1.154 | std_simulations_grade | 0.002 |
| | mean_workshops_time | 0.421 | mean_prior_knowledge_time | 0.004 |
| | std_prior_knowledge_time | 0.081 | std_prior_knowledge_grade | 0.004 |
| | mean_simulations_time | 0.058 | std_workshops_time | 0.004 |
| | mean_workshops_grade | 0.052 | mean_simulations_grade | 0.024 |
| **60%** | weighted_average | 1.543 | mean_prior_knowledge_time | 0.007 |
| | mean_workshops_time | 0.467 | std_prior_knowledge_grade | 0.015 |
| | std_workshops_grade | 0.216 | std_simulations_grade | 0.016 |
| | std_prior_knowledge_time | 0.205 | std_simulations_time | 0.027 |
| | mean_workshops_grade | 0.118 | mean_simulations_grade | 0.028 |
| **80%** | weighted_average | 0.952 | std_simulations_grade | 0.004 |
| | mean_workshops_grade | 0.204 | std_prior_knowledge_grade | 0.010 |
| | mean_workshops_time | 0.160 | mean_simulations_time | 0.014 |
| | mean_simulations_grade | 0.095 | std_workshops_time | 0.018 |
| | std_prior_knowledge_time | 0.072 | mean_prior_knowledge_time | 0.018 |

TABLE VII
TOP AND BOTTOM 5 ENGAGEMENT FEATURES: IMPORTANCE ACROSS DIFFERENT CWA THRESHOLDS

| Assessment Percentage | Top 5 features | Coef | Bottom 5 features | Coef |
|---|---|---|---|---|
| **20%** | std_video_views | 0.055 | std_interactions | 0.002 |
| | mean_video_time | 0.053 | mean_lessons_attempts | 0.002 |
| | outline_viewed | 0.042 | discussion_created | 0.003 |
| | std_video_time | 0.036 | forum_searched | 0.004 |
| | mean_multimedia_views | 0.029 | post_created | 0.004 |
| **40%** | std_multimedia_views | 0.120 | mean_lessons_attempts | 0.0001 |
| | mean_multimedia_views | 0.016 | mean_lessons_time | 0.0002 |
| | mean_video_time | 0.014 | forum_viewed | 0.001 |
| | std_interactions | 0.014 | course_navigation | 0.001 |
| | std_video_views | 0.013 | forum_searched | 0.002 |
| **60%** | mean_lessons_attempts | 0.224 | std_interactions | 0.012 |
| | weekend | 0.174 | discussion_created | 0.014 |
| | mean_multimedia_views | 0.165 | post_created | 0.015 |
| | forum_searched | 0.138 | week | 0.020 |
| | days | 0.135 | outline_viewed | 0.021 |
| **80%** | mean_multimedia_views | 0.070 | forum_searched | 0.001 |
| | mean_lessons_attempts | 0.064 | std_interactions | 0.006 |
| | days | 0.061 | course_navigation | 0.007 |
| | weekend | 0.051 | std_video_views | 0.008 |
| | std_lessons_attempts | 0.036 | outline_viewed | 0.010 |

## VII. DISCUSSIONS AND CONCLUSIONS

We explored the use of machine learning techniques to predict student performance in a Moodle course. We focused on the logic reasoning PPVU course at the Universidad de Antioquia, which has a unique non-linear structure and flexible learning path. This made it important to update the features at the moment of predictions to perform the predictive models.

To implement our approach, we followed a three-phase methodology: (a) collected and preprocessed data from the PPVU course, (b) created features related to student engagement and performance, and (c) used a nested cross-validation strategy to evaluate several machine learning algorithms, such as logistic regression, SVM, and random forest.

Our results showed that logistic regression performs well in all the CWA thresholds, compared with the other models and the baseline, demonstrating high precision, recall, and F1-score. Although random forest might provide a better performance for some CWA thresholds, logistic regression provides a better trade-off between performance, computational complexity, and interpretability.

In our analysis, we examined the data of students who completed at least 65% of the course and found that the models showed improved performance compared to a baseline. A feature importance analysis revealed that certain features related to student performance, such as weighted average, mean simulations grade, mean workshops grade, and the time spent in the different evaluative assessments, significantly impacted the predictions. Engagement features had less impact on the prediction, but we found that student participation with educative resources like multimedia and video had a positive impact.

Our study contributes to understanding how machine learning can be applied to predict student performance in Moodle courses and emphasizes the importance of considering the non-linear structure and flexible learning path of courses like PPVU. Our proposed methodology and the insights gained from our analysis can provide a foundation for further research and for developing intervention strategies to improve engagement and academic outcomes in virtual learning environments.

## References

[1] R. Li, J. Singh, and J. Bunk, "Technology tools in distance education: A review of faculty adoption," *EdMedia+ Innovate Learning*, pp. 1982–1987, 2018.

[2] R. K. Ellis, "Learning management systems," *Alexandria, VI: American Society for Training & Development (ASTD)*, 2009.

[3] M. Llamas, M. Caeiro, M. Castro, I. Plaza, and E. Tovar, "Use of lms functionalities in engineering education," in *2011 Frontiers in Education Conference (FIE)*, IEEE, 2011, S1G–1.

[4] M. Kloft, F. Stiehler, Z. Zheng, and N. Pinkwart, "Predicting mooc dropout over weeks using machine learning methods," in *Proceedings of the EMNLP 2014 workshop on analysis of large scale social interaction in MOOCs*, 2014, pp. 60–65.

[5] J.-T. Seo, B.-N. Park, Y.-g. Kim, and K.-W. Yeon, "Analysis of lms data of distance lifelong learning center learners and drop-out prediction," *Journal of Human-centric Science and Technology Innovation*, vol. 1, no. 3, pp. 23–32, 2021.

[6] N. Nordin, H. Norman, and M. A. Embi, "Technology acceptance of massive open online courses in malaysia.," *Malaysian Journal of Distance Education*, vol. 17, no. 2, 2015.

[7] A. H. Nabizadeh, D. Goncalves, S. Gama, J. Jorge, and H. N. Rafsanjani, "Adaptive learning path recommender approach using auxiliary learning objects," *Computers & Education*, vol. 147, p. 103 777, 2020.

[8] E. Alsadoon, "The impact of an adaptive e-course on students' achievements based on the students' prior knowledge," *Education and information technologies*, vol. 25, no. 5, pp. 3541–3551, 2020.

[9] Y. Chen, X. Li, J. Liu, and Z. Ying, "Recommendation system for adaptive learning," *Applied psychological measurement*, vol. 42, no. 1, pp. 24–41, 2018.

[10] J. Mercado, C. H. Mendoza, D. A. Ramirez-Salazar, *et al.*, "Work in progress: A didactic strategy based on machine learning for adaptive learning in virtual environments," in *2023 IEEE World Engineering Education Conference (EDUNINE)*, IEEE, 2023, pp. 1–4.

[11] 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. 1, pp. 17–29, 2016.

[12] M. Adnan, A. Habib, J. Ashraf, *et al.*, "Predicting at-risk students at different percentages of course length for early intervention using machine learning models," *Ieee Access*, vol. 9, pp. 7519–7539, 2021.

[13] N. Tomasevic, N. Gvozdenovic, and S. Vranes, "An overview and comparison of supervised data mining techniques for student exam performance prediction," *Computers & education*, vol. 143, p. 103 676, 2020.

[14] J. Kuzilek, M. Hlosta, and Z. Zdrahal, "Open university learning analytics dataset," *Scientific data*, vol. 4, no. 1, pp. 1–8, 2017.

[15] L. Cohausz, A. Tschalzev, C. Bartelt, and H. Stuckenschmidt, "Investigating the importance of demographic features for edm-predictions.," *International Educational Data Mining Society*, 2023.

[16] M. Riestra-González, M. del Puerto Paule-Ruíz, and F. Ortin, "Massive lms log data analysis for the early prediction of course-agnostic student performance," *Computers & Education*, vol. 163, p. 104 108, 2021.

[17] M. Hoq, P. Brusilovsky, and B. Akram, "Analysis of an explainable student performance prediction model in an introductory programming course.," *International Educational Data Mining Society*, 2023.

[18] N. Rohani, K. Gal, M. Gallagher, and A. Manataki, "Early prediction of student performance in a health data science mooc," in *Proceedings of the 16th International Conference on Educational Data Mining*, International Educational Data Mining Society, 2023.

[19] Y. Mao, F. Khoshnevisan, T. Price, T. Barnes, and M. Chi, "Cross-lingual adversarial domain adaptation for novice programming," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 7682–7690.

[20] T. Anderson, D. Annand, and N. Wark, "The search for learning community in learner paced distance education: Or,'having your cake and eating it, too!'" *Australasian Journal of Educational Technology*, vol. 21, no. 2, 2005.

[21] R. Marcinkevičs and J. E. Vogt, "Interpretable and explainable machine learning: A methods-centric overview with concrete examples," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, e1493, 2023.

[22] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.," *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[23] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[24] P. J. Guo, J. Kim, and R. Rubin, "How video production affects student engagement: An empirical study of mooc videos," in *Proceedings of the first ACM conference on Learning@ scale conference*, 2014, pp. 41–50.

[25] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*, Springer, 2002, pp. 9–50.

[26] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, Ieee, 2015, pp. 1200–1205.

[27] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.

[28] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artificial Intelligence Review*, vol. 54, pp. 1937–1967, 2021.

[29] P. Joshi, *Python machine learning cookbook*. Packt Publishing Ltd, 2016.

[30] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance eval-

uation," *The Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.

[31] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" In *Database Theory—ICDT'99: 7th International Conference Jerusalem, Israel, January 10–12, 1999 Proceedings 7*, Springer, 1999, pp. 217–235.

[32] Y. A. Alsariera, Y. Baashar, G. Alkawsi, A. Mustafa, A. A. Alkahtani, and N. Ali, "Assessment and evaluation of different machine learning algorithms for predicting student performance," *Computational Intelligence and Neuroscience*, vol. 2022, 2022.