International Journal on Advanced Science Engineering Information Technology

Vol.13 (2023) No. 3
ISSN: 2088-5334

Tree-Based Ensemble Methods and Their Applications for Predicting Students' Academic Performance

Indri Dayanah Ayulani [a], Agatha Melinda Yunawan [a], Tamara Prihutaminingsih [a], Devvi Sarwinda [a], Gianinna Ardaneswari [a], Bevina Desjwiandra Handari [a,*]

[a] *Department of Mathematics, Universitas Indonesia, Kampus UI Depok, Depok, 16424, Indonesia*
Corresponding author: [*]bevina@sci.ui.ac.id

Abstract— **Students' academic performance is a key aspect of online learning success. Online learning applications known as Learning Management Systems (LMS) store various online learning activities. In this research, students' academic performances in online course X are predicted such that teachers could identify students who are at risk much sooner. The prediction uses tree-based ensemble methods such as Random Forest, XGBoost (Extreme Gradient Boosting), and LightGBM (Light Gradient Boosting Machine). Random Forest is a bagging method, whereas XGBoost and LightGBM are boosting methods. The data recorded in LMS UI, or EMAS (e-Learning Management Systems) is collected. The data consists of activity data for 232 students (219 passed, 13 failed) in course X. This data is divided into three proportions (80:20, 70:30, and 60:40) and three periods (the first, first two, and first three months of the study period). Data is pre-processed using the SMOTE method to handle imbalanced data and implemented in all categories, with and without feature selection. The prediction results are compared to determine the best time for predicting students' academic performance and how well each model can predict the number of unsuccessful students. The implementation results show that students' academic performance can be predicted at the end of the second month, with best prediction rates of 86.8%, 80%, and 75% for the LightGBM, Random Forest, and XGBoost models, respectively, with feature selection. Therefore, with this prediction, students who could fail still have time to improve their academic performance.**

Keywords— **Students' academic performance; online learning; learning management systems; tree-based ensemble methods; machine learning; Random Forest; XGBoost; LightGBM; features selection; learning analytics.**

Manuscript received 17 Dec. 2021; revised 5 Aug. 2022; accepted 30 Jan. 2023. Date of publication 30 Jun. 2023.

I. INTRODUCTION

Currently, the Learning Management System (LMS) application as the online learning medium is increasingly popular in universities. LMS is an online learning medium providing various teaching and learning facilities such as modules, online discussion forums, online quizzes, assignment collection media, and student recordings [1]. Using LMS, the teacher could give students feedback and create appropriate learning. In addition, the student activities in the LMS are recorded and saved by the system so that student activity data can be analyzed by institutions, administrators, and teachers to obtain information about student behavior during online learning [1]. Hence the teacher can determine the factors that affect the student's academic performance [2]. The performance of academics results from acquisition based on the evaluation of a particular final period. The evaluation in question is the scores obtained by students from assignments, mid-term exams, and final exams [3]. The result of academic performance can be represented in the form of exam results, GPA, and statements of if a student has passed a course or not [4]. In this research, academic performance depends on if a student has passed a course or not.

Prediction of student academic performance is an essential topic in the university environment. Prediction of student performance is intended to identify early student academic performance and to determine indicators that affect student learning success [5]. Students' academic performance can be predicted by using Machine Learning [2]. According to the research that Jayaprakash, Krishnan, and Jaiganesh [6] and Akçapınar, Altun, and Aşkar [2] have done, students' academic performance can be predicted based on the student activities in LMS by using various Machine Learning methods.

Research by Oliveira et al. [7] identifying students at risk of dropping out is based on learning activity data on LMS using several forms of machine learning such as k-Nearest Neighbors, C-Support Vector Classification, Logistics Regression, Random Forest, Adaptive Boosting, Gradient Boosting, and Extremely Randomized Trees. In addition, Helal et al. [8] used Naive Bayes (NB), Support Vector Machine (SVM), and Decision Tree (DT) to predict students who are at risk of not graduating in a course based on data on online learning activities and social factors. However, according to Zhao et al. [9], some tree-based ensemble methods were proven to produce better and more accurate prediction performance than some other models. Therefore, this research uses the tree-based ensemble method to predict students' academic performance based on their activities in course X at Universitas Indonesia LMS.

The tree-based ensemble methods used are Random Forest, XGBoost, and LightGBM. The Random Forest method uses the principle of bagging, which is a method that is based on resampling that is accompanied by duplication to build a decision tree [10]. While XGBoost and LightGBM use the boosting principle, combining multiple decision tree models [11]. Before we developed prediction models, we oversampled the imbalanced datasets on training data using SMOTE as a pre-processing step. It then continues with feature selection. From this, we can determine activities on LMS that influence students' academic performance. We also compared the obtained results with models without feature selection.

Each method is implemented on three different proportions of training to testing data: 80:20, 70:30, and 60:40. In each of these proportions. The implementation is also based on three data sets. These are data from the first, second, and third months of the study period in course X. The output of this research is that we can determine the best time to predict students' academic performance, including information on each model's prediction results, i.e., how well they can predict unsuccessful students and which model is most suitable for this research. With this prediction, teachers can identify students who have the potential to fail in course X earlier to take preventive action to enable students who can fail to improve their academic performance.

II. Materials and Method

A. Data

The data used in this research is student activity data in the form of activity features on LMS UI in online course X, which happened on Sept. 14, 2020 - Jan. 6, 2021. The data is divided into three periods: from the beginning of the semester to the end of the first month, from the beginning to the end of the second month, and from the beginning to the end of the third month. The data consists of activity data of 232 students (219 successful students, 13 unsuccessful students), where successful students were the majority class, and unsuccessful students were the minority class. Table I shows features used in predicting student academic performance.

TABLE I
List of Features and Their Descriptions

No	Feature	Description
1	A submission has been submitted	The number of submitted document files
2	Course activity completion updated	The number of activities that have been ticked
3	File viewed	The total number of visits for the document file page
4	URL viewed	The total number of visits for the URL page
5	Course viewed	The total number of visits for the course page
6	Discussion viewed	The total number of visits for the discussion page
7	Some content has been posted.	The total number of posts for the discussion page
8	The status of the submission has been viewed	The total number of visits for viewing the status of the quiz and assignment

A. Methods

A decision tree is one of the most well-known classification methods. This method produces a model as a branched tree structure consisting of a root node, an internal node, and a leaf node. The root node is the top node in the tree, an internal node is a branch of a node with more branches from it, and the leaf node is the end of the tree [12]. The decision tree method, namely the tree-based ensemble method, is a machine learning technique that combines multiple decision tree models to produce a more optimal predictive model. In the next sub-chapter, three tree-based ensemble methods are discussed, which are the Random Forest, Extreme Gradient Boosting (XGBoost), and Light Gradient Boosting Machine (LightGBM) models

1) Random Forest Method: Random Forest is a model for classification, resulting from the development of the CART (Classification and Regression Tree) method [13]. Random Forest consists of a collection of decision trees [14]. The Random Forest method uses bootstrap on the data sample to randomly draw B datasets with resampling replacements from the original data. Each dataset has the same size as the original data [15]. The random forest classification method uses the Gini index for splitting D nodes [16]. The basic idea of splitting nodes is to make each sub-node as homogeneous as possible after splitting. Given data D and A_i feature $i = 1,2,\ldots,n$. A binary split on A_i feature divides D into D_1 and D_2, with the Gini index value of A_i feature can be calculated as follows [12].

$$Gini_{A_i}(D) = \frac{|D_1|}{|D|} I_G(D_1) + \frac{|D_2|}{|D|} I_G(D_2) \qquad (1)$$

where $|D|$ is the number of data or frequency in data D, $|D_i|$ is the number of data or frequency in sub-data D_i, $i = 1,2$, and

$$I_G(D_j) = 1 - \sum_{a_i \in A_i} p_{a_i}^2, j = 1,2 \qquad (2)$$

where p_{a_i} is the proportion of values of A_i features. The feature that has the smallest Gini index value will be the root node in the decision tree. Obtaining the Random Forest model requires the majority vote or mode from the predicted results of each decision tree [15]. Random Forest utilizes a random selection feature in forming trees [13]. Algorithm 1 illustrates a Random Forest algorithm [17].

2) XGBoost Method: XGBoost is a scalable machine-learning model for tree boosting [18]. The XGBoost model consists of a collection of trees. This collection uses the cumulative sum of the predicted values of a sample in each tree as the prediction of the sample. In the XGBoost model, regularization terms directly use the first and second derivative values of the loss function to prevent overfitting [19]. The XGBoost model will optimize the following objective function:

$$\tilde{L}^{(k)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(k-1)} + f_k(x_i)\right) + \Omega^{(f_k)} \tag{3}$$

```
**Algorithm 1:** Random Forest algorithm
**Input**: Dataset ($D$), number of trees ($B$)
**for** $b = 1, 2, \ldots, B$ **do**
    1. Make a bootstrap dataset $D_b$ of size $n$ from $D$
    2. While $m < p$
        i. Select $m$ features at random from $p$ available features.
        ii. Find the best binary split of $m$ features using equations (1) and (2)
        iii. Split the node into two descendant nodes using the result from step ii.
    3. Combine the prediction results from all decision trees based on the majority vote as the final prediction result
**end**
**Output**: Classification results from predictions
```

where k stands for the k-th tree, $l\left(y_i, \hat{y}_i^{(k-1)} + f_k(x_i)\right)$ is a differentiable loss function that measures the difference between the prediction value $\hat{y}_i$ and the target value y_i, and $\Omega^{(f_k)}$ is a regularization function to control the model complexity and prevent overfitting [18]. Use the objective function to find the optimal tree structure [20].

Use the greedy algorithm to find the splitting node of each feature as a parent node. For each split node, calculate the gain.

$$Gain = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \tag{4}$$

where g_i and h_i denote the first and second derivatives of the loss function, respectively, I_L and I_R denote the set of the observation on the left and right branches of a tree, respectively, λ is the leaf weight penalty regularization term, and γ is the leaf tree penalty regularization term. Splitting the tree becomes possible when the value of the gain is larger than zero. Algorithm 2 illustrates an XGBoost algorithm.

3) LightGBM Method: Light Gradient Boosting Machine (LightGBM) is a tree-based ensemble method, launched by Microsoft in 2017 [17]. LightGBM uses histogram-based algorithms and a leaf-wise growth strategy to increase the training speed and reduce memory consumption [21]. LightGBM also uses Exclusive Feature Bundling (EFB), and Gradient-based One-Side Sampling (GOSS). EFB can reduce the feature dimension further. It bundles mutually exclusive features into a single feature [22].

```
**Algorithm 2**: XGBoost algorithm
**Input**: Dataset, the number of trees ($K$), the learning rate ($\eta$), the number of terminal nodes ($T$)
Initialize $\hat{y}_i^{(0)}$
**for** $k = 1, 2, \ldots, K$ **do**
    Calculate $g_i^{(k)} = \partial_{\hat{y}_i^{(k-1)}} l\left(y_i, \hat{y}_i^{(k-1)}\right)$
    Calculate $h_i^{(k)} = \partial^2_{\hat{y}_i^{(k-1)}} l\left(y_i, \hat{y}_i^{(k-1)}\right)$
    Determine the structure $\{\hat{R}_{jk}\}_{j=1}^{T}$ by choosing split node with maximized $Gain$ in equation (4)
    Determine the leaf weights $\{\omega_{jk}\}_{j=1}^{T}$, $\omega_{jk} = -\frac{\sum_{i\in I_j} g_i^{(k)}}{\sum_{i\in I_j} h_i^{(k)} + \lambda}$
    Determine the decision tree $f_k(x_i) = \eta \sum_{j=1}^{T} \omega_j I[x \in \hat{R}_{jk}]$
        Add trees $\hat{y}_i^{(k)} = \hat{y}_i^{(k-1)} + f_k(x_i)$
**end**
**Output**: Prediction value $\hat{y}_i = \hat{y}_i^{(K)} = \sum_{k=1}^{K} f_k(x_i)$
```

GOSS is used to split the optimal node by calculating variance gain. Firstly, rank the dataset to their gradient absolute values in descending order. Second, the top $a \times 100\%$ data with larger gradients are selected. This is referred to as set A. Then, subset B with size $b \times |A^c|$ is randomly selected from set A^c consisting of $(1-a) \times 100\%$. Finally, split the dataset using the variance gain $V_j(d)$ as follows [23]:

$$V_j(d) = \frac{1}{n}\left(\frac{\left(\sum_{x_i\epsilon A_l} g_i + \frac{1-a}{b}\sum_{x_i\epsilon B_l} g_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_i\epsilon A_r} g_i + \frac{1-a}{b}\sum_{x_i\epsilon B_r} g_i\right)^2}{n_r^j(d)}\right) \tag{5}$$

where d is the split point, x_{ij} is the i^{th} observation of the j^{th} feature, $A_l = \{x_i \in A : x_{ij} \leq d\}$, $A_r = \{x_i \in A : x_{ij} > d\}$, $B_l = \{x_i \in B : x_{ij} \leq d\}$, $B_r = \{x_i \in A : x_{ij} > d\}$, g_i is the

```
**Algorithm 3**: LightGBM algorithm
**Input**: Dataset ($D$), the number of trees ($M$), big gradient data sampling ratio ($a$), slight gradient data sampling ratio ($b$) the number of observations ($N$), the learning rate ($\lambda$), the model parameter ($\gamma$)
Initialize $F_0(x_i) = \underset{\gamma}{arg\ min} \sum_{i=1}^{N} l(y_i, \gamma)$
**for** $m = 1, 2, \ldots, M$ **do**
    Calculate gradient values
        $r_{im} = -\left[\frac{\partial l(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x_i)=F_{m-1}(x_i)}, i = 1, \ldots, N$
    Resample dataset using GOSS process:
        $topN = a \times len(D); randN = b \times len(D)$
        $sorted = GetSortedIndices(abs(r_{im}))$
        $A = sorted[1 : topN]$
        $B = RandomPick(sorted[topN : len(D)], randN)$
    Determine the structure $\{R_{k,m}\}_{k=1}^{K}$ by calculate the maximum variance gain with equation (5)
    Determine the leaf weights
        $\gamma_{k,m} = \underset{\gamma}{\arg min} \sum_{x_i \in R_{k,m}} l(y_i, F_{m-1}(x_i) + \gamma)$
    Determine the decision tree
        $F_m(x_i)' = \lambda \sum_{k=1}^{K} \gamma_{k,m} 1_{R_{k,m}}(x_i)$ with
        $1_{R_{k,m}}(x_i) = \begin{cases} 1 & if\ x_i \in R_{k,m} \\ 0 & if\ x_i \notin R_{k,m} \end{cases}$
    Add trees $F_m(x_i) = F_{m-1}(x_i) + F_m(x_i)'$
**end**
**Output** Prediction value $F_M(x_i)$
```

gradient of loss function $l(y_i, F(x_i))$, where y_i is target value and $F(x_i)$ is the prediction value, n is the number of observations, $n_l^j(d)$ is the number of observations in sets A_l and B_l, $n_r^j(d)$ is the number of observations in sets A_r and B_r, and $\frac{1-a}{b}$ is employed to normalize the sum of the gradients over B back to the size of A^c [23]. Algorithm 3 shows the LightGBM algorithm [24].

The initial stage of research is the preparation of data, in which student activity data in the form of activity features recorded on LMS UI in online course X is collected and serves as the input data. The next stage in data preparation is data pre-processing. This process involves splitting data into training and testing data. Since these data sets are imbalanced, the Synthetic Minority Oversampling Technique (SMOTE) is implemented on the training data, and the testing data will be used in the modeling process. The implementation stage consists of the feature selection and modeling processes. The feature selection process on Random Forest, XGBoost, and LightGBM models utilizes Gini index, Gain, and Variance Gain, respectively, and is implemented on the training data resulting from SMOTE. The features in question are student activities recorded in LMS, such as online discussion forums, quizzes, etc. The next stage is modeling all three algorithms using data resulting from feature selection and data without feature selection. Each algorithm is implemented on three proportions of training to testing data: 80:20, 70:30, and 60:40. In each of these proportions, the implementation is also divided into three periods. These are data from the first, second, and third months of the study period in course X. Later, the evaluation stage of these three algorithms is determined by the accuracy, precision minority, and recall minority values that can be calculated based on the corresponding confusion matrices. Based on the evaluation results, the interpretation will be configured. Finally, the output stage confirms the best time to predict students' academic performance, including information on each model's prediction results. This stage of research is described in Fig 1.

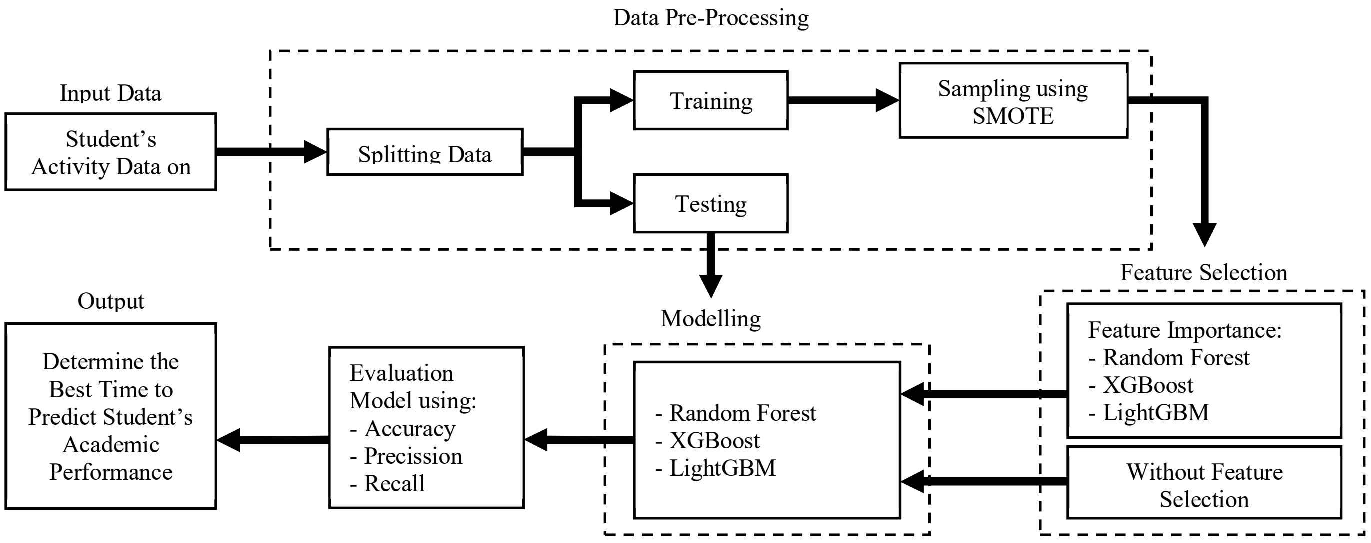
Data Pre-Processing
Input Data
Student's Activity Data on
Splitting Data
Training
Testing
Sampling using SMOTE
Feature Selection
Feature Importance: - Random Forest - XGBoost - LightGBM
Without Feature Selection
Modelling
- Random Forest - XGBoost - LightGBM
Evaluation Model using: - Accuracy - Precission - Recall
Output
Determine the Best Time to Predict Student's Academic Performance

Fig. 1 Stages of research

SMOTE is an important approach by oversampling the minority class to generate a balanced data set [25]. It is a method that uses oversampling to deal with data imbalances by transforming a minority class becomes a class whose proportions in the sample are greater than those in the original data. SMOTE creates synthetic data from the minority class using the k-nearest neighbor approach. The k-nearest neighbors value is determined using the Euclidean distances between two minority data. A chosen minority data is the one with the smallest value of such Euclidean distance, which is calculated as follows [26].

$$d(i,j) = \sqrt{\sum_{n=1}^{p}\left(x_{i,n} - x_{j,n}\right)^2} \quad (6)$$

where $d(i,j)$ is the distance between observations of i and j, $x_{i,n}$ is the ith observed value of the nth variable, $x_{j,n}$ is the jth observed value of the nth variable, and p is the number of features. Synthetic data is created using the following equation:

$$x_{syn} = x_0 + (x^* - x_0)\gamma \quad (7)$$

where x_{syn} is the synthetic data, x_0 is the primary observation data on the minority class, x^* is the observation data randomly selected from k data observations closest to x_0, and γ is a random number between 0 and 1. Table II shows data proportions before and after implementation of SMOTE.

TABLE II
DATA PROPORTIONS BEFORE AND AFTER IMPLEMENTATION OF SMOTE

Train-test Proportion		**Training Data**		**Testing Data**	
		Pass	**Fail**	**Pass**	**Fail**
80:20	Before	175	10	44	3
	After	175	175	44	3
70:30	Before	153	9	66	4
	After	153	153	66	4
60:40	Before	131	8	88	5
	After	131	131	88	5

Table II shows imbalanced data in the training data in each proportion. Specifically, minority class data (representing unsuccessful students) is far less than the majority class data (representing successful students). For example, with a 60:40 proportion, there is 131 majority class data and only 8 minority class data. After using SMOTE, the number of minority class data and majority class data becomes equal. for example, with a 60:40 proportion, the amount of data in both classes is equal at 131. In the testing data, there is no change in the number of minority class data where SMOTE is not used.

The performance of each model in the evaluation stage is determined based on the corresponding confusion matrix, where the positive class represents students that failed, and the negative class represents students that passed. The confusion matrix can be seen below in Table III [2], [27].

TABLE III
CONFUSION MATRIX

		Prediction	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

In Table III, True Positive (TP) is a correct prediction of failed students, True Negative (TN) is a correct prediction of passed students, False Positive (FP) is an incorrect prediction of failed students, and False Positive (FN) is an incorrect prediction of passed students.

The model's performances rely on the accuracy, precision minority, and recall minority values that can be calculated based on the confusion matrix in Table III. The following equations can determine each value, respectively [28]. Accuracy: The accuracy of a model when predicting failed or passed students is as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

Precision minority: The accuracy of a given model incorrectly predicting failed students compared with the overall prediction of failed students is as follows:

$$Precision\ minority = \frac{TP}{TP+FP} \tag{9}$$

Recall minority: The accuracy of a given model in correctly predicting the number of failed students is as follows:

$$Recall\ minority = \frac{TP}{TP+FN} \tag{10}$$

The evaluation stage determines the average accuracy, precision minority, recall minority, precision majority, and recall the majority of a particular model. The minority class represents the class of unsuccessful students, and the majority class represents the class of successful students. Based on equation (9), the precision majority is the ratio of students predicted to pass correctly compared to the number of students initially predicted to pass. Based on equation (10), the recall majority is how accurately the model predicts successful students. The results are in the form of average values since there are ten experiments in each category.

III. Results and Discussion

Each model is implemented both with and without feature selection. In each case, this is applied to both training and testing data. Implementations on training data build the model, while implementations on testing data test its performance. Each implementation uses three different data proportions to determine how they affect each model's performance.

Implementation of both training and testing data without feature selection is meant to show if any of the models show signs of overfitting. The results of the implementation of each model on training data with and without feature selection are then compared to show if feature selection affects the accuracy of each model on training data.

Implementation of testing data, both with and without feature selection, is meant to show if the data sample size (first, second, or third months) affects the accuracy of each model. The model's accuracy shows how well that model can predict passed or failed students. In addition, this implementation also shows if any of the models' performance is affected by feature selection. Their performance is determined by recall minority, i.e., the accuracy of a given model in correctly predicting the number of failed students.

A. Models' Performance without Features Selection

Tables IV, V, and VI, respectively, state the performance of the Random Forest, XGBoost, and LightGBM models on training data without feature selection in three train-test proportions: 80:20, 70:30, and 60:40, using data up to the end of each of the first, second and third months. The implementation results in the form of model performance include the average accuracy, precision minority, recall minority, precision majority, and recall majority.

TABLE IV
THE AVERAGE OF RANDOM FOREST MODEL PERFORMANCE WITHOUT FEATURE SELECTION USING TRAINING DATA

Train-test Proportion	**End of month #**	**Accuracy**	**Precision Minority**	**Recall Minority**	**Precision Majority**	**Recall Majority**
	1	1	1	1	1	1
80:20	2	1	1	1	1	1
	3	1	1	1	1	1
	1	1	1	1	1	1
70:30	2	1	1	1	1	1
	3	1	1	1	1	1
	1	1	1	1	1	1
60:40	2	1	1	1	1	1
	3	1	1	1	1	1

Table IV shows that the Random Forest model created using training data without feature selection accurately predicted students' academic performance in each experiment as indicated by the performance model with a value of one on all performance metrics. The XGBoost model created using training data without feature selection yielded a different performance in each experiment, as shown in Table V.

TABLE V
THE AVERAGE OF XGBOOST MODEL PERFORMANCE WITHOUT FEATURE SELECTION USING TRAINING DATA

Train-test Proportion	**End of month #**	**Accuracy**	**Precision Minority**	**Recall Minority**	**Precision Majority**	**Recall Majority**
	1	0.983	0.974	0.99	0.99	0.974
80:20	2	0.994	0.992	0.995	0.995	0.992
	3	0.997	0.993	0.996	0.996	0.993
	1	0.985	0.976	0.991	0.991	0.975
70:30	2	0.998	0.997	0.998	0.998	0.997
	3	0.998	0.995	0.997	0.997	0.995
	1	0.991	0.985	0.995	0.995	0.985
60:40	2	0.998	0.994	0.997	0.997	0.994
	3	0.999	0.996	0.998	0.998	0.996

However, this model predicted students' academic performance in each experiment sufficiently since all the parameter values are more than 97% in both the majority and minority classes. The performance of the LightGBM model created using training data without feature selection, as seen in Table VI, also shows that its parameters have a value of 1. Therefore, the LightGBM model accurately predicted students' academic performance in each experiment.

TABLE VI
THE AVERAGE OF LIGHTGBM MODEL PERFORMANCE WITHOUT FEATURE SELECTION USING TRAINING DATA

Train-test Proportion	End of month #	Accuracy	Precision Minority	Recall Minority	Precision Majority	Recall Majority
80:20	1	1	1	1	1	1
	2	1	1	1	1	1
	3	1	1	1	1	1
70:30	1	1	1	1	1	1
	2	1	1	1	1	1
	3	1	1	1	1	1
60:40	1	1	1	1	1	1
	2	1	1	1	1	1
	3	1	1	1	1	1

Tables VII through IX show the results of the evaluation of each model on testing data without feature selection, with predictions at the end of each of the first three months. Based on Tables VII through IX, each model with the respective train-test proportions has generally improved performance for each period based on the average accuracy, precision minority, and recall minority values, except for the LightGBM model in Table IX. The LightGBM model, with a train-test proportion of 80:20, has the highest value of average recall minority and precision majority using data up to the second month.

TABLE VII
THE AVERAGE OF RANDOM FOREST MODEL PERFORMANCE WITHOUT FEATURE SELECTION USING TESTING DATA

Train-test Proportion	End of month #	Accuracy	Precision Minority	Recall Minority	Precision Majority	Recall Majority
80:20	1	0.87	0.338	0.432	0.952	0.906
	2	0.927	0.48	0.67	0.972	0.945
	3	0.937	0.569	0.766	0.98	0.953
70:30	1	0.874	0.178	0.375	0.96	0.903
	2	0.929	0.488	0.725	0.983	0.941
	3	0.938	0.548	0.775	0.986	0.947
60:40	1	0.889	0.196	0.38	0.954	0.915
	2	0.923	0.373	0.68	0.984	0.899
	3	0.952	0.508	0.7	0.985	0.957

Table VII shows that the Random Forest model created using testing data without feature selection for each train-test proportion displayed an increasing trend for all parameters. Using data up to the end of the first month, the Random Forest model can predict at least 87% of successful and unsuccessful students correctly. Meanwhile, at the end of the second and third months, the model can correctly predict more than 92% of successful and unsuccessful students.

Table VIII shows the increasing trend for all parameters also applies to the XGBoost model created using testing data without feature selection for each train-test proportion in most periods. Using data from the first month, the XGBoost model has a prediction rate of at least 82% for both passed and failed students. With data from both the second and third months, the prediction rate improves to at least 90%.

In Table IX, using data until the end of the first month, the LightGBM model can predict at least more than 83% of successful and unsuccessful students correctly. By including data until the end of the second and third months, its successful prediction rate increases to over 90%.

TABLE VIII
THE AVERAGE OF XGBOOST MODEL PERFORMANCE WITHOUT FEATURE SELECTION USING TESTING DATA

Train-test Proportion	End of month #	Accuracy	Precision Minority	Recall Minority	Precision Majority	Recall Majority
80:20	1	0.819	0.152	0.399	0.953	0.849
	2	0.904	0.431	0.734	0.981	0.915
	3	0.924	0.488	0.734	0.981	0.935
70:30	1	0.824	0.123	0.4	0.96	0.849
	2	0.923	0.437	0.75	0.983	0.933
	3	0.932	0.502	0.8	0.986	0.937
60:40	1	0.852	0.16	0.4	0.961	0.878
	2	0.902	0.318	0.68	0.983	0.916
	3	0.921	0.4	0.68	0.983	0.936

The results from Tables VII through IX show that the Random Forest and LightGBM models yielded the best prediction without feature selection, with an average recall minority of 77.5% on a 70:30 proportion by using data until the third month. With the same data and proportion, the XGBoost model yielded an average recall minority of 80%. For the LightGBM model, there is a best average recall minority difference between the 70:30 and 80:20 proportions no greater than 0.8%. Therefore, data from the end of the second month can be used to predict the failure rate for students.

TABLE IX
THE AVERAGE OF LIGHTGBM MODEL PERFORMANCE WITHOUT FEATURE SELECTION USING TESTING DATA

Train-test Proportion	End of month #	Accuracy	Precision Minority	Recall Minority	Precision Majority	Recall Majority
80:20	1	0.836	0.14	0.332	0.95	0.871
	2	0.906	0.441	0.767	0.984	0.914
	3	0.928	0.597	0.7	0.981	0.943
70:30	1	0.844	0.131	0.375	0.96	0.871
	2	0.926	0.497	0.725	0.982	0.937
	3	0.936	0.532	0.775	0.986	0.949
60:40	1	0.854	0.156	0.36	0.959	0.884
	2	0.906	0.339	0.68	0.983	0.92
	3	0.925	0.419	0.74	0.987	0.936

By comparing the results of Tables IV through VI to those of Tables VII through IX, overfitting is absent in all three models, as shown by comparing the performance of all three models on training data with their performance on testing data in the majority class. The average accuracies, precision majorities, and recall majorities for the training data are very similar to those of the testing data. The significant difference between the average precision minorities and recall minorities in training and testing data results from the small number of students that failed (minority class) in testing data.

B. Models' Performance with Features Selection

The following implementation uses feature selection based on the feature importance of every model constructed. Table X shows each model's top 5 features' importance and a slice of the list of features for all three models. The features include discussion viewed, and some content has been posted. Features from the selection results of feature importance include student activity in discussion forums, accessing learning materials, and checking the results of quizzes and assignments.

TABLE X
TOP 5 FEATURE IMPORTANCE OF RANDOM FOREST, XGBOOST, AND LIGHTGBM MODELS

No	**Random Forest**	**XGBoost**	**LightGBM**
1.	File viewed	The status of the submission has been viewed	Course activity completion updated
2.	Course viewed	Some content has been posted.	File viewed
3.	Discussion viewed	Discussion viewed	URL viewed
4.	Some content has been posted.	URL viewed	Discussion viewed
5.	The status of the submission has been viewed	A submission has been submitted	Some content has been posted

The performance of the Random Forest model on training data with feature selection using data at the end of each of the first three months is the same as that obtained in Table IV. Similarly, the performance of the LightGBM model is identical to that shown in Table VI, except that with a 60:40 train-test proportion, the average accuracy, precision minority, and recall majority was 99.7%, 99.9%, and 99.7%, respectively. Therefore, the Random Forest and LightGBM models accurately predicted students' academic performance in each experiment. Meanwhile, Table XI shows the performance of the XGBoost model with feature selection on training data.

Comparing Tables V and XI shows a slight decrease in parameter values for the XGBoost model. Nevertheless, this model can still sufficiently predict students' academic performance in each experiment, given that all the parameter values are more than 95% in both the majority and minority classes. This discussion shows that feature selection on all three models' training data does not affect predictions of students' academic performance.

TABLE XI
THE AVERAGE OF XGBOOST MODEL PERFORMANCE WITH FEATURE SELECTION USING TRAINING DATA

Train-test Proportion	**End of month #**	**Accuracy**	**Precision Minority**	**Recall Minority**	**Precision Majority**	**Recall Majority**
80:20	1	0.963	0.957	0.969	0.968	0.957
	2	0.977	0.973	0.981	0.981	0.972
	3	0.988	0.983	0.988	0.988	0.983
70:30	1	0.967	0.964	0.972	0.972	0.962
	2	0.985	0.982	0.989	0.989	0.982
	3	0.986	0.984	0.986	0.986	0.983
60:40	1	0.972	0.964	0.976	0.976	0.963
	2	0.991	0.988	0.991	0.991	0.988
	3	0.992	0.985	0.995	0.995	0.985

Tables XII through XIV show the results of the implementation of each model on testing data with feature selection. Table XII, the average XGBoost model performance with feature selection using training data, shows that the Random Forest model created using testing data with feature selection for each train-test proportion displayed an increasing trend for all parameters. The Random Forest model with a data proportion of 80:20 can predict at least 86% of successful and unsuccessful students correctly. This improves to 95.2% with the use of data up to the end of the third month.

TABLE XII
THE AVERAGE OF RANDOM FOREST MODEL PERFORMANCE WITH FEATURE SELECTION USING TESTING DATA

Train-test Proportion	**End of month #**	**Accuracy**	**Precision Minority**	**Recall Minority**	**Precision Majority**	**Recall Majority**
80:20	1	0.86	0.19	0.4	0.955	0.915
	2	0.923	0.373	0.68	0.984	0.899
	3	0.952	0.508	0.7	0.985	0.957
70:30	1	0.862	0.177	0.4	0.963	0.888
	2	0.904	0.448	0.8	0.988	0.908
	3	0.925	0.469	0.825	0.99	0.927
60:40	1	0.859	0.164	0.42	0.963	0.884
	2	0.916	0.351	0.7	0.985	0.928
	3	0.935	0.429	0.74	0.987	0.943

Table XIII, the average XGBoost model performance with feature selection using testing data, shows that the model with a data proportion of 80:20 can predict at least 84% of successful and unsuccessful students correctly. The prediction improves to 91.2% with data up to the end of the second month. Interestingly, with a 70:30 proportion, using data up to the end of the second month yields the highest value of all parameters. In addition, the most significant difference between each parameter in the proportion 80:20 and 70:30 is only 9.8%. Therefore, the model with a 70:30 proportion and data up to the end of the second month can predict students' academic performance.

TABLE XIII
THE AVERAGE OF XGBOOST MODEL PERFORMANCE WITH FEATURE SELECTION USING TESTING DATA

Train-test Proportion	**End of month #**	**Accuracy**	**Precision Minority**	**Recall Minority**	**Precision Majority**	**Recall Majority**
80:20	1	0.84	0.159	0.4	0.955	0.869
	2	0.912	0.451	0.666	0.977	0.929
	3	0.911	0.442	0.699	0.979	0.926
70:30	1	0.858	0.164	0.425	0.963	0.881
	2	0.901	0.353	0.75	0.983	0.907
	3	0.893	0.335	0.725	0.983	0.904
60:40	1	0.85	0.134	0.38	0.961	0.878
	2	0.89	0.277	0.66	0.98	0.903
	3	0.893	0.304	0.68	0.982	0.906

Table XIV shows that using a data proportion of 80:20 yields a correct prediction rate of 83.1% for passed and failed students, and adding data from the end of the third month increases this rate to 95.7%. Interestingly, with feature selection, the most significant difference between parameter values using data from the end of the second and third months, with an 80:20 train-test data proportion, is at most only 15.4%.

Therefore, with feature selection, the model can make an accurate prediction using an 80:20 distribution and data from the end of the second month. Based on the results of Tables XIII and XIV, the XGBoost and LightGBM models can predict students' academic performance using data after the second month. The XGBoost model with a data proportion of 70:30 has a recall minority of 75% (less than the 80% figure recorded without feature selection), whereas the LightGBM model requires an 80:20 proportion with a maximum recall minority of 86.8% (higher than the 76.7% figure recorded without feature selection).

TABLE XIV
THE AVERAGE OF LIGHTGBM MODEL PERFORMANCE WITH FEATURE SELECTION USING TESTING DATA

Train-test Proportion	End of month #	Accuracy	Precision Minority	Recall Minority	Precision Majority	Recall Majority
80:20	1	0.831	0.143	0.366	0.94	0.869
	2	0.926	0.483	0.868	0.992	0.928
	3	0.957	0.637	0.834	0.99	0.964
70:30	1	0.834	0.145	0.4	0.96	0.859
	2	0.921	0.44	0.85	0.989	0.924
	3	0.939	0.555	0.8	0.988	0.944
60:40	1	0.836	0.153	0.46	0.965	0.857
	2	0.914	0.389	0.76	0.988	0.922
	3	0.929	0.471	0.8	0.99	0.938

In Table XII, the Random Forest model with a 70:30 proportion and data taken from the end of the third month yielded a peak average recall minority of 82.5%. However, this model's average peak recall minority was 80% by the end of the second month (greater than the pre-feature selection figure of 76.6%). Therefore, feature selection can improve the performance of the Random Forest and LightGBM models, but not the XGBoost model. To determine if the Random Forest model with a data proportion of 70:30 using data taken after the third month yields results that are not significantly different from the same model using data taken from after the second month, use a statistic test called McNemar's Test [29], [30]. The hypotheses of McNemar's Test are as follows:

- H_0 = The predictions from the models built using data until the end of the second and third months show no significant differences.
- H_1 = The predictions from the models built using data until the end of the second and third months show significant differences.

with $\alpha = 0.01$. Reject H_0 if p-value $\leq \alpha$. Based on McNemar's Test, the p-value of the accuracy of both models is 0.062 and the p-value of the recall minority is 1. Since the p-values of both the accuracy and recall minority are larger than α, we can accept H_0. Therefore, the Random Forest model with a data proportion of 70:30 using data taken after the third-month yields results that are not significantly different from the same model using data from the end of the second month. Thus, all three models (Random Forest, XGBoost, and LightGBM) can be used to predict unsuccessful students earlier, i.e., after the second month.

IV. Conclusion

This research aims to predict students' academic performance, particularly unsuccessful students, based on academic activity features recorded by LMS UI using three tree-based ensemble methods: the Random Forest, XGBoost, and LightGBM models. Implementing training and testing data without feature selection shows no signs of overfitting on any of these models. Implementing training data for all three models shows that feature selection has little effect on their performance.

The implementation of testing data on these models shows that data proportions, data size and feature selection affect their performance. The Random Forest and XGBoost models can predict 77.5% and 80% of unsuccessful students based on data after the third month without feature selection on a 70:30 proportion. The prediction rate of the LightGBM model using data up to the second month is 76.7% on an 80:20 proportion. Meanwhile, for models created using feature selection and data up to the end of the second month, the LightGBM, Random Forest, and XGBoost models can predict 86.6%, 80%, and 75% of unsuccessful students with 80:20, 70:30, and 60:40 proportions, respectively. Based on these results, the XGBoost model performs best without feature selection with three months of data, but the other two models perform best with feature selection with two months of data.

To predict unsuccessful students, prediction results should be obtained sooner. In this research, the LightGBM and Random Forest models should be used to successfully predict unsuccessful students, since, compared to the XGBoost model, they can yield a better prediction using data up to the end of the second month.

Acknowledgment

Universitas Indonesia funded this research, PUTI grant scheme No.: NKB-1977/UN2.RST/HKP.05.00/2020.

References

[1] F. Chen and Y. Cui, "Utilizing student time series behaviour in learning management systems for early prediction of course performance," *J. Learn. Anal.*, vol. 7, no. 2, pp. 1–17, 2020, doi: 10.18608/JLA.2020.72.1.

[2] G. Akçapınar, A. Altun, and P. Aşkar, "Using learning analytics to develop early-warning system for at-risk students," *Int. J. Educ. Technol. High. Educ.*, vol. 16, no. 1, 2019, doi: 10.1186/s41239-019-0172-z.

[3] E. Latif and S. Miles, "The Impact of Assignments and Quizzes on Exam Grades: A Difference-in-Difference Approach," *J. Stat. Educ.*, vol. 28, no. 3, 2020, doi: 10.1080/10691898.2020.1807429.

[4] E. Alyahyan and D. Düştegör, "Predicting academic success in higher education: literature review and best practices," *International Journal of Educational Technology in Higher Education*, vol. 17, no. 1. 2020, doi: 10.1186/s41239-020-0177-7.

[5] E. Popescu and F. Leon, "Predicting Academic Performance Based on Learner Traces in a Social Learning Environment," *IEEE Access*, vol. 6, 2018, doi: 10.1109/ACCESS.2018.2882297.

[6] S. Jayaprakash, S. Krishnan, and J. Jaiganesh, "Predicting Students Academic Performance using an Improved Random Forest Classifier," *2020 Int. Conf. Emerg. Smart Comput. Informatics, ESCI 2020*, pp. 238–243, 2020, doi: 10.1109/ESCI48226.2020.9167547.

[7] M. M. De Oliveira, R. Barwaldt, M. R. Pias, and D. B. Espindola, "Understanding the Student Dropout in Distance Learning," in *Proceedings - Frontiers in Education Conference, FIE*, 2019, vol. 2019-October, doi: 10.1109/FIE43999.2019.9028433.

[8] S. Helal *et al.*, "Predicting academic performance by considering student heterogeneity," *Knowledge-Based Syst.*, vol. 161, 2018, doi: 10.1016/j.knosys.2018.07.042.

[9] Y. Zhao *et al.*, “Ensemble learning predicts multiple sclerosis disease course in the SUMMIT study,” *npj Digit. Med.*, vol. 3, no. 1, 2020, doi: 10.1038/s41746-020-00338-8.

[10] T. H. Lee, A. Ullah, and R. Wang, “Bootstrap Aggregating and Random Forest,” *Adv. Stud. Theor. Appl. Econom.*, vol. 52, pp. 389–429, 2020, doi: 10.1007/978-3-030-31150-6_13.

[11] S. Rahman, M. Irfan, M. Raza, K. M. Ghori, S. Yaqoob, and M. Awais, “Performance analysis of boosting classifiers in recognizing activities of daily living,” *Int. J. Environ. Res. Public Health*, vol. 17, no. 3, 2020, doi: 10.3390/ijerph17031082.

[12] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. 2012.

[13] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, 2001, doi: 10.1023/A:1010933404324.

[14] D. Denisko and M. M. Hoffman, “Classification and interaction in random forests,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 115, no. 8. 2018, doi: 10.1073/pnas.1800256115.

[15] K. Lin, Y. Hu, and G. Kong, “Predicting in-hospital mortality of patients with acute kidney injury in the ICU using random forest model,” *Int. J. Med. Inform.*, vol. 125, 2019, doi: 10.1016/j.ijmedinf.2019.02.002.

[16] L. E. Raileanu and K. Stoffel, “Theoretical comparison between the Gini Index and Information Gain criteria,” *Ann. Math. Artif. Intell.*, vol. 41, no. 1, 2004, doi: 10.1023/B:AMAI.0000018580.96245.c6.

[17] K. C. Dewi, H. Murfi, and S. Abdullah, “Analysis Accuracy of Random Forest Model for Big Data - A Case Study of Claim Severity Prediction in Car Insurance,” 2019, doi: 10.1109/ICSITech46713.2019.8987520.

[18] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-August-2016, doi: 10.1145/2939672.2939785.

[19] D. Zhang and Y. Gong, “The Comparison of LightGBM and XGBoost Coupling Factor Analysis and Prediagnosis of Acute Liver Failure,” *IEEE Access*, 2020, doi: 10.1109/ACCESS.2020.3042848.

[20] S. Wang *et al.*, “A new method of diesel fuel brands identification: SMOTE oversampling combined with XGBoost ensemble learning,” *Fuel*, vol. 282, 2020, doi: 10.1016/j.fuel.2020.118848.

[21] W. Liang, S. Luo, G. Zhao, and H. Wu, “Predicting hard rock pillar stability using GBDT, XGBoost, and LightGBM algorithms,” *Mathematics*, vol. 8, no. 5, 2020, doi: 10.3390/MATH8050765.

[22] C. Chen, Q. Zhang, Q. Ma, and B. Yu, “LightGBM-PPI: Predicting protein-protein interactions through LightGBM with multi-information fusion,” *Chemom. Intell. Lab. Syst.*, vol. 191, 2019, doi: 10.1016/j.chemolab.2019.06.003.

[23] G. Ke *et al.*, “LightGBM: A highly efficient gradient boosting decision tree,” in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-December.

[24] A. A. Taha and S. J. Malebary, “An Intelligent Approach to Credit Card Fraud Detection Using an Optimized Light Gradient Boosting Machine,” *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2020.2971354.

[25] A. S. Hussein, T. Li, C. W. Yohannese, and K. Bashir, “A-SMOTE: A new pre-processing approach for highly imbalanced datasets by improving SMOTE,” *Int. J. Comput. Intell. Syst.*, vol. 12, no. 2, 2019, doi: 10.2991/ijcis.d.191114.002.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “snopes.com: Two-Striped Telamonia Spider,” *J. Artif. Intell. Res.*, vol. 16, no. Sept. 28, pp. 321–357, 2002, [Online]. Available: https://arxiv.org/pdf/1106.1813.pdf%0Ahttp://www.snopes.com/horrors/insects/telamonia.asp.

[27] I. Düntsch and G. Gediga, “Confusion Matrices and Rough Set Data Analysis,” in *Journal of Physics: Conference Series*, 2019, vol. 1229, no. 1, doi: 10.1088/1742-6596/1229/1/012055.

[28] J. D. Novaković, A. Veljović, S. S. Ilić, Ž. Papić, and T. Milica, “Evaluation of Classification Models in Machine Learning,” *Theory Appl. Math. Comput. Sci.*, vol. 7, no. 1, 2017.

[29] Q. Wu, F. Nasoz, J. Jung, B. Bhattarai, and M. V. Han, “Machine Learning Approaches for Fracture Risk Assessment: A Comparative Analysis of Genomic and Phenotypic Data in 5130 Older Men,” *Calcif. Tissue Int.*, vol. 107, no. 4, 2020, doi: 10.1007/s00223-020-00734-y.

[30] M. Q. R. Pembury Smith and G. D. Ruxton, “Effective use of the McNemar test,” *Behav. Ecol. Sociobiol.*, vol. 74, no. 11, 2020, doi: 10.1007/s00265-020-02916-y.