JOIN (Jurnal Online Informatika)
p-ISSN: 2528-1682, e-ISSN: 2527-9165
Volume 10 Number 2 | December 2025: 270-285
DOI: 10.15575/join.v10i2.1590

Optimizing Machine Learning Models for Graduation on Time Prediction: A Comparative Study with Resampling and Hyperparameter Tuning

Rizal Bakri[1,2], Syamsu Alam[1], Niken Probondani Astuti[2], Muhammad Ilham Bakhtiar[3]
[1]Department of Digital Business, Makassar State University, Indonesia
[2]Statistics Research Group, STIEM Bongaya, Makassar, Indonesia
[3]Department of Guidance and Counseling Education, Makassar State University, Indonesia

Article Info

Article history:

Received Mar 13, 2025
Revised April 10, 2025
Accepted May 22, 2025
Published August 17, 2025

Keywords:

Educational Data Mining
Graduation on Time
Hyperparameter Tuning
Machine Learning
Resampling Methods

ABSTRACT

Timely graduation prediction is a crucial issue in higher education, especially when academic, demographic, and behavioral factors interact in complex ways. However, many previous studies rely on default machine learning (ML) parameters and fail to consider the class imbalance problem, leading to suboptimal predictions. This study aims to build a comprehensive framework to evaluate the effectiveness of seven ML algorithms, which are AdaBoost, K-Nearest Neighbors, Naïve Bayes, Neural Network, Random Forest, SVM-RBF, and XGBoost, for predicting graduation on time by incorporating five resampling techniques and hyperparameter tuning. Resampling methods include Random Undersampling (RUS), Random Oversampling (ROS), SMOTENC, and two hybrid approaches (RUS-ROS and SMOTENC-RUS). Hyperparameter tuning was conducted using Grid Search, and model performance was evaluated through cross-validation and hold-out methods. The results show that Random Forest combined with RUS-ROS achieved the best performance, with an average metric score of 0.948. Statistical analysis using PERMANOVA ($p = 0.009$) and Bonferroni's post-hoc pairwise tests confirmed significant differences between certain models. This study contributes to the educational data mining literature by demonstrating that combining resampling and hyperparameter tuning improves classification performance in imbalanced educational datasets.

Corresponding Author:

Rizal Bakri,
Department of Digital Business, Makassar State University, Indonesia
Jl. Pendidikan I No.27, Tidung, Kec. Rappocini, Kota Makassar, Sulawesi Selatan 90222
Email: rizal.bakri@unm.ac.id

1. INTRODUCTION

A main challenge for universities is to deeply analyze their performance, identify institutional uniqueness, and build development strategies to achieve future academic targets [1]. In an increasingly complex and competitive educational environment, student academic success, particularly Graduation on Time (GOT), has become a key metric for evaluating institutional performance [2]. However, predicting GOT is inherently difficult due to the complex interplay between academic, demographic, and behavioral factors. Traditional statistical models often lack the flexibility to capture these interactions,

prompting a shift toward machine learning (ML) approaches that can process large and diverse educational datasets [3], [4].

Machine learning has shown promise in educational contexts, especially for predicting student outcomes such as academic achievement, dropout risk, and graduation likelihood [5]. As more data becomes available from student information systems, learning management platforms, and administrative records, the ability to develop accurate and proactive prediction models is increasingly valuable for higher education institutions. Yet despite widespread ML adoption in Educational Data Mining (EDM), several methodological limitations persist in the literature.

Many prior studies rely on default ML parameters without conducting proper hyperparameter tuning, which limits the models' potential performance. In addition, class imbalance is a recurring issue in GOT datasets, where the number of on-time graduates typically far exceeds that of delayed graduates. Models trained on such data often become biased toward the majority class, resulting in poor predictive performance on minority cases. These issues are further compounded by the limited use of robust evaluation methods; most studies rely on a single metric, such as accuracy, and few apply multivariate statistical significance testing to validate model differences.

To address these limitations, this study proposes a comprehensive evaluation framework for GOT prediction by combining multiple machine learning algorithms, five data resampling techniques, and systematic hyperparameter tuning using grid search. Unlike previous research, this study evaluates model performance across five key metrics, which are accuracy, precision, recall, F1-score, and AUC, using both cross-validation and hold-out validation. It also introduces a statistical significance testing procedure through Permutational Multivariate Analysis of Variance (PERMANOVA) and Bonferroni's post-hoc pairwise comparisons test to determine whether performance differences are statistically significant.

By integrating these components, this study contributes a holistic and statistically grounded framework to the EDM literature. The findings not only identify the most effective combination of algorithm and resampling strategy for GOT classification but also offer practical guidance for educational institutions in developing reliable, data-driven approaches to support timely student graduation.

2. RELATED WORK

The application of machine learning (ML) in predicting Graduation on Time (GOT) has received growing attention in the field of Educational Data Mining (EDM). For instance, a study by [6] in 2019 introduced an Artificial Neural Network (ANN), which demonstrated promising performance for GOT classification. Other researchers have explored the use of Naïve Bayes (NB), achieving an accuracy of 86.63%, making it a technique that remains relevant to this day [7], [8], [9], [10], [11], [12], [13], [14]. In 2020, [15] employed K-Nearest Neighbor (KNN) with k-fold cross-validation and reported improvements in prediction accuracy. More advanced models, such as Support Vector Machine (SVM) [16] and Random Forest (RF) [17], have also shown high accuracy in predicting GOT, while ensemble methods like AdaBoost [18] have exhibited potential for further enhancing model performance.

In addition to evaluating individual models, several studies have compared ML algorithms for GOT prediction across different performance metrics. For example, [19] compared NB and ANN, revealing the superiority of ANN with an accuracy of 77.04%. In a subsequent study, [20] reported that SVM outperformed both RF and NB, highlighting the importance of selecting an optimal number of k-folds, typically between 5 and 20. Similarly, [21] found that ANN achieved higher accuracy than NB (81.82%). More recently, studies from 2021 to 2024 [22], [23], [24] have consistently demonstrated that RF outperforms other algorithms, including SVM and NB, in terms of overall predictive accuracy and Area Under the Curve (AUC). Meanwhile, [25] and [26] confirmed that SVM remained competitive, achieving up to 96.34% accuracy in 2023. On the other hand, [27] and [13] offered further support for RF, concluding that it outperforms both SVM and NB in predicting on-time graduation.

Despite these promising results, existing studies still report conflicting findings regarding the most effective algorithm for GOT prediction. Moreover, many studies rely on default ML settings without performing hyperparameter tuning, potentially constraining model performance. In fact, hyperparameter optimization plays a critical role in improving model generalizability [28]. For instance, [29] demonstrated that tuning parameters in RF, NB, and SVM significantly enhances prediction accuracy. Similarly, [30] reported improved accuracy after applying hyperparameter tuning to Extreme Gradient Boosting (XGBTree). The importance of this process was further highlighted by [31], who

p-ISSN: 2528-1682
e-ISSN: 2527-9165

showed that dynamic tuning improves performance in GOT prediction. However, the application of hyperparameter tuning within EDM remains underexplored.

Another critical limitation in prior research is the treatment of imbalanced datasets. GOT prediction data is often skewed, with a significantly larger proportion of students graduating on time compared to those who do not. As a result, ML models are frequently biased toward the majority class, leading to poor sensitivity in detecting minority outcomes. Addressing this issue requires the application of effective resampling techniques [32], [33]. Several studies have evaluated such techniques in the context of GOT prediction. For example, [34] found that applying SMOTE to ANN improved performance over imbalanced baselines. Likewise, [35] and [36] reported that SMOTE enhances the predictive accuracy of SVM and RF models, respectively. A comparative analysis by [37] further demonstrated that different resampling methods each offer unique benefits when used with RF. Despite these advances, the role of resampling in EDM remains underrepresented in the literature.

A final methodological gap lies in the evaluation process itself. Most prior studies assess model performance using only a single metric, typically accuracy, without conducting multivariate significance testing across multiple evaluation criteria. For example, [32] employed ANOVA to compare resampling strategies but did not incorporate metrics such as precision, recall, F1-score, or AUC, which may lead to incomplete or biased conclusions.

To bridge these methodological gaps, the present study proposes a comprehensive and rigorous evaluation framework that integrates seven ML algorithms, five resampling strategies, systematic hyperparameter tuning, and multivariate performance validation.

Table 1. Review of research works in the field of educational data mining for predicting students' Graduation on Time.

Article	Machine Learning Techniques							Resampling Methods					Evaluation						Validation		Hyperparameters Tuning
	Random Forest	SVM	XGBTree	Naïve Bayes	K-Nearest Neighbors	Neural Network	AdaBoost	RUS	ROS	RUS+ROS	SMOTE	SMOTE+RUS	PRECISION	RECALL	F1-SCORE	ACCURACY	ROC-AUC	MULTIVARIATE ANOVA	Hold-Out	K-Fold Cross-validation	
[6]						✓										✓			✓		
[7]				✓												✓			✓	✓	
[8]				✓									✓	✓		✓			✓		
[9]				✓												✓					
[10]				✓												✓			✓	✓	
[11]				✓												✓			✓	✓	
[12]				✓									✓	✓	✓	✓			✓		
[13]	✓			✓	✓	✓							✓	✓	✓	✓	✓		✓	✓	
[14]				✓									✓	✓	✓	✓	✓		✓	✓	
[15]					✓											✓			✓	✓	
[16]		✓														✓				✓	
[17]	✓															✓					
[18]							✓									✓					
[19]				✓		✓							✓	✓	✓	✓	✓		✓	✓	
[20]	✓	✓		✓									✓	✓	✓	✓				✓	
[21]				✓		✓										✓			✓	✓	
[22]	✓	✓				✓										✓			✓		
[23]	✓	✓		✓	✓											✓			✓	✓	
[24]	✓			✓	✓								✓	✓		✓			✓	✓	
[25]		✓				✓										✓			✓		
[26]		✓					✓						✓	✓	✓	✓			✓		
[27]	✓	✓		✓									✓	✓	✓	✓	✓		✓	✓	

Article	Machine Learning Techniques							Resampling Methods					Evaluation					Validation			Hyperparameters Tuning
	Random Forest	SVM	XGBTree	Naïve Bayes	K-Nearest Neighbors	Neural Network	AdaBoost	RUS	ROS	RUS+ROS	SMOTE	SMOTE+RUS	PRECISION	RECALL	F1-SCORE	ACCURACY	ROC-AUC	MULTIVARIATE ANOVA	Hold-Out	K-Fold Cross-validation	
[29]	✓	✓		✓									✓	✓		✓	✓		✓	✓	✓
[30]			✓												✓	✓	✓		✓		✓
[31]	✓	✓	✓	✓	✓	✓	✓									✓			✓	✓	✓
[34]	✓										✓					✓	✓			✓	
[35]		✓									✓		✓	✓		✓			✓	✓	
[36]	✓	✓			✓						✓		✓	✓	✓	✓			✓	✓	
[37]	✓							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	
Present Work	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

3. MATERIAL AND METHOD

3.1. Dataset Information and data preprocessing

This study uses a previously collected dataset focusing on the on-time graduation (GOT) of undergraduate students at STIEM Bongaya University, as referenced in [31], [37]. The dataset was obtained directly from the institution's academic information system and contains 4,093 records with 15 predictor variables, consisting of six continuous and nine categorical features, as shown in Table 2. The target variable is binary: a value of 1 indicates that a student graduated on time, while 0 represents a delayed graduation.

The class distribution is highly imbalanced, with 3,071 students classified as on-time graduates (majority class) and 1,022 as delayed graduates (minority class). Although this imbalance may influence model bias, no specific observation-level bias was detected. Standard preprocessing procedures were applied to prepare the data for modeling. Categorical variables were encoded using label encoding, while continuous variables were scaled to ensure consistent feature ranges across input values. Outlier detection was also considered during preprocessing; however, no significant outliers were identified that warranted removal or transformation.

This study employed two common model validation techniques: random hold-out validation and shuffled 10-fold cross-validation. In the holdout method, 80% of the data was allocated for training and 20% for testing.

Table 2. The main features of students' dataset of STIEM Bongaya University.

Feature	Value	Description	Type
NCP	44 - 175	Student's Number Credit Passed	Continuous
SMT4	0 - 4	Student's GPA Semester 4	Continuous
SMT3	0 - 4	Student's GPA Semester 3	Continuous
SMT2	0 - 4	Student's GPA Semester 2	Continuous
SMT1	0 - 4	Student's GPA Semester 1	Continuous
AA	16 - 46	Student's Age Admission	Continuous
FS	Accounting, Financial, Marketing, Human Resource	Student's Focus Study	Categorical
FI	IDR 1 - IDR 499.999 IDR 500.000 - IDR 999.999 IDR 1.000.000 - IDR 1.999.999 IDR 2.000.000 - IDR 4.999.999 IDR 5.000.000 - IDR 20.000.000 More than IDR 20.000.000 Nil Income	Student's Father Income	Categorical Categorical

Feature	Value	Description	Type
MI	IDR 1 - IDR 500.000 IDR 500.000 - IDR 999.999 IDR 1.000.000 - IDR 1.999.999 IDR 2.000.000 - IDR 4.999.999 IDR 5.000.000 - IDR 20.000.000 More than IDR 20.000.000 Nil Income	Student's Mother Income	Categorical
SEX	Male, Female	Student's Sex	Categorical
RE	with Parents, with Guardian, Boarding House, Dormitory, Others	Student's Residence	Categorical
TR	Public transportation, Private Car, Private Motorcycle Walk to campus	Student's Transportation	Categorical
DEP	Management, Accounting	Department taken by student	Categorical
CT	Regular Class, Executive Class	Class type taken by student	Categorical
GOT Status	1, 0	Student's graduation status (1 is GOT and 0 is not GOT)	Categorical

3.2. Handling Imbalance dataset with resampling methods

To address the class imbalance observed in the dataset, this study applied five widely adopted resampling strategies from recent EDM literature: Random Undersampling (RUS), Random Oversampling (ROS), a hybrid of RUS and ROS, SMOTE for Nominal and Continuous data (SMOTE-NC), and a hybrid of SMOTE-NC and RUS. These methods were selected based on their popularity, proven effectiveness in prior studies, and suitability to the data characteristics [33], [37], [38], [39].

SMOTE-NC was chosen over standard SMOTE because the dataset contains both categorical and continuous features, which SMOTE-NC is specifically designed to handle. However, each resampling method presents potential limitations: RUS may discard valuable information by removing majority-class samples, ROS can increase the risk of overfitting by duplicating minority-class samples, and SMOTE-NC may generate noisy or unrealistic synthetic instances. To mitigate these limitations, hybrid techniques were employed to achieve class balance while reducing the risk of overfitting and minimizing information loss or synthetic bias.

In this study, SMOTE-NC was configured using five nearest neighbors (k = 5) and a resampling ratio of 0.8, following from previous research [33], [37]. All resampling procedures were applied exclusively to the training set to prevent data leakage and to ensure valid model evaluation.

All resampling methods were systematically evaluated using a consistent set of evaluation metrics (accuracy, precision, recall, F1-score, and AUC) and validation protocols (hold-out and cross-validation) and were applied uniformly across seven machine learning models. This provided a robust comparative framework. The implementation was conducted using the ROSE [40] and themis [41] packages in R.

3.3. Machine learning models and Hyperparameters tuning

This study evaluated the performance of seven machine learning (ML) algorithms for predicting on-time graduation: Random Forest (RF), Support Vector Machine with Radial Basis Function kernel (SVM-RBF), Extreme Gradient Boosting (XGBTree), k-Nearest Neighbors (KNN), Naïve Bayes (NB), Artificial Neural Network (ANN), and AdaBoost. These models were selected based on their theoretical diversity and frequent use in educational data mining (EDM). They represent a range of learning paradigms: tree-based methods (RF, XGB), probabilistic classifiers (NB), kernel-based models (SVM), neural models (ANN), instance-based learning (KNN), and ensemble techniques (AdaBoost), allowing comprehensive algorithmic comparison under varied conditions.

To maximize each model's predictive potential, hyperparameter tuning was conducted using grid search combined with 10-fold cross-validation on the training data. This approach ensures that tuning decisions do not leak information from the test set, thereby reducing the risk of overfitting. The tuning process was performed separately for each resampling scenario to account for changes in data distribution. The choice of hyperparameter ranges was based on commonly accepted defaults in prior

studies [31] and empirical guidelines. For instance, Random Forest used mtry = c(2, 3, 4, 5) to explore how many features to consider at each split, appropriate for a dataset with 15 predictors. SVM-RBF was tuned over sigma = seq(0.1, 0.9, 0.1) and C = c(0.01, 0.1, 1), enabling control over the kernel width and regularization strength. XGBTree's learning rate (eta = c(0.025, 0.05, 0.1)) and tree depth (max_depth = c(3, 4, 5)) were tuned to balance training speed and model complexity. KNN used k = c(3, 5, 7, 9) to test different neighborhood sizes, avoiding even numbers to prevent tie votes. For NB, kernel density estimation (usekernel) and smoothing parameters (adjust, fL) were tuned. ANN configurations included variations in network size (size = c(1, 3, 5, 7)) and regularization (decay = seq(0.01, 0.1, 0.01)). AdaBoost used different boosting iterations (nIter = seq(10, 100, 10)) and base method settings.

Two model validation strategies were employed: (1) 80/20 hold-out validation, which provides a realistic estimate of model performance on unseen data, and (2) shuffle 10-fold cross-validation, which increases reliability by ensuring each data point contributes to both training and evaluation. The use of both methods allowed for comparison between real-world generalization (hold-out) and internal consistency (cross-validation). To further prevent overfitting, particularly in small or imbalanced subsets, the tuning process was constrained to moderate parameter ranges, and regularization components were activated where applicable (e.g., decay in ANN, gamma in XGBTree). Model performance under each configuration was compared using a consistent set of evaluation metrics and resampling methods, ensuring fair and robust comparison across all experimental conditions. All models were implemented using R, primarily utilizing the caret package [42], and the randomForest package [43]. These packages provide robust functions for training, tuning, and evaluating ML models efficiently. This systematic approach enables a fair and reproducible comparison of classification performance across different algorithm-resampling-tuning combinations, thereby strengthening the empirical validity of the findings.

Table 3. Machine learning techniques with hyperparameters settings.

Methods	Hyperparameters tuning
Random Forest	mtry = c(2, 3, 4, 5)
SVM-RBF	sigma = seq(0.1,0.9, by=0.1); C= c(0.01, 0.1, 1)

XGBTree	max_depth = c(3, 4, 5); nrounds = seq(from = 25, to = 95, by = 10);
	eta = c(0.025, 0.05, 0.1); gamma = seq(from = 1, to = 5, by = 1);
	colsample_bytree = c(0.6,0.7,0.8); min_child_weight = 1;
	subsample = c(0.7, 0.8, 0.9, 1)

NB	usekernel = c(T,F); adjust=c(0.01,0.1,1); fL=c(0.01,0.1,1)
KNN	k = c(1,3,5,7,9)
ANN	size = seq(from = 1, to = 7, by = 2); decay = seq(from = 0, to = 0.1, by = 0.01)
AdaBoost	nIter = seq(10, 100, by=10); method = c("Adaboost.MI", "Real Adaboost")

3.4. Evaluation Methods

Evaluating classifier performance is essential for identifying the most effective model, particularly in imbalanced classification problems. This study uses five key evaluation metrics: accuracy, sensitivity (recall), precision, F1-score, and area under the curve (AUC) to provide a comprehensive and balanced assessment of model effectiveness. Relying solely on a single metric such as accuracy can be misleading, especially when class distributions are skewed. To ensure a fair comparison across all performance dimensions, the average of these five metrics is also computed for each model.

In addition to descriptive metrics, statistical testing is conducted to determine whether observed differences between machine learning models are statistically significant. While nonparametric tests like the Friedman test and Wilcoxon signed-rank test are frequently used in ML research, they are typically restricted to a single metric. In contrast, this study compares model performance across multiple metrics simultaneously, necessitating the use of multivariate methods such as MANOVA. However, MANOVA requires multivariate normality assumptions.

To verify this assumption, several multivariate normality tests were conducted, including Mardia's test, Henze-Zirkler's test, Royston's test, Doornik-Hansen's test, and the E-statistic, using the MVN package in R [44]. As reported in the Results section, all tests indicated significant deviations from

normality, invalidating the use of MANOVA for this dataset. As an alternative, Permutational Multivariate Analysis of Variance (PERMANOVA) was used via the vegan package in R [45]. PERMANOVA computes Euclidean distances between observations and uses random permutations to test whether the performance differences among models are greater than would be expected by chance, without requiring distributional assumptions [46]. To explore which model pairs differ significantly, Bonferroni-adjusted post-hoc comparisons were conducted [47]. The results are presented in the Results section. Additionally, a boxplot of average metric scores is provided to visually compare performance and highlight the most consistent and best-performing model.

4. RESULT AND DISCUSSION

This study aims to examine the impact of class imbalance on machine learning performance by applying various resampling techniques. Additionally, hyperparameter tuning is conducted to optimize model performance across different resampling strategies. All models and analyses were implemented in R, a high-level programming language widely used for statistical computing and machine learning. The following sections present the experimental results, starting with dataset distribution, followed by hyperparameter tuning, model performance evaluation, statistical significance testing, and discussion section.

4.1. Results

4.1.1. Resampling results

As the first step in the modeling process, this study examines the impact of class imbalance on machine learning performance through the application of various resampling strategies. Table 4 presents the class distribution of the dataset before and after resampling, including the breakdown of training and testing sets. Initially, the dataset was highly imbalanced, with 3,071 instances in the majority class (students graduating on time) and 1,022 in the minority class (students not graduating on time). This imbalance can lead to biased classification outcomes, where machine learning models disproportionately favor the majority class, often at the expense of correctly identifying minority class instances.

To address this issue, five resampling techniques were applied, each offering distinct advantages and trade-offs. Random Undersampling (RUS) reduces the number of majority class instances to match the minority class, resulting in a fully balanced dataset (1,022:1,022). While this improves class balance, it does so by removing potentially informative samples, which may lead to information loss. In contrast, Random Oversampling (ROS) duplicates instances from the minority class until it matches the majority (3,071:3,071). This method retains all original data but may introduce redundancy and increase the likelihood of overfitting.

SMOTE-NC (Synthetic Minority Over-sampling Technique for Nominal and Continuous variables) was also employed to generate synthetic examples of the minority class. This technique produced a distribution of 3,071:2,456, allowing for increased diversity in training data while minimizing the overfitting risks associated with simple duplication. Furthermore, two hybrid approaches were utilized to combine the strengths of individual resampling methods. The first hybrid, combining RUS and ROS, generated a relatively balanced dataset (1,978:2,115), aiming to reduce data loss while avoiding excessive duplication. The second hybrid, which combines SMOTE-NC with RUS, achieved a perfect balance (2,456:2,456) while preserving variation and reducing noise.

Although all datasets preserved an 80:20 split between training and testing partitions, the actual number of instances in each split varied depending on the resampling strategy. Undersampling techniques led to smaller training sets, while oversampling expanded the dataset size. These differences can affect both the learning process and the generalization ability of the models.

In summary, the use of diverse resampling strategies highlights the importance of systematically evaluating how class distribution affects model performance. The next section

investigates how these differences interact with hyperparameter optimization and contribute to overall classification outcomes.

Table 4. Class size of full dataset, Training, and testing for each resampling method

Dataset	Baseline	Undersampling (RUS)	Oversampling (ROS)	Hybrid 1 (RUS+ROS)	SMOTE-NC	Hybrid 2 (SMOTE-NC+RUS)
Full Dataset	1:3071 0:1022	1:1022 0:1022	1:3071 0:3071	1:2115 0:1978	1:3071 0:2456	1:2456 0:2456
Training (80%)	1:2457 0:818	1:818 0:818	1:2457 0:2457	1:1692 0:1583	1:2457 0:1965	1:1965 0:1965
Testing (20%)	1:614 0:204	1:212 0:212	1:614 0:614	1:423 0:395	1:614 0:491	1:491 0:491

4.1.2 The best hyperparameter tuning

After balancing the data, the next step was to optimize each model's performance through hyperparameter tuning. This process was conducted independently for each resampling technique to account for distributional differences. Table 5 presents the best hyperparameter configurations for each machine learning algorithm across different resampling methods, highlighting how data distribution influences model performance. Random Forest consistently selects mtry = 5, except under RUS, where a lower value (mtry = 4) is preferred due to the reduced dataset size. SVM-Radial maintains C = 1 across all methods, but sigma varies, with higher values observed in ROS, indicating sensitivity to oversampling. XGBTree exhibits variations in nrounds, gamma, and colsample_bytree, where deeper trees (max_depth = 5) and higher gamma are favored in oversampling techniques, while RUS prefers simpler configurations (nrounds = 75). Naïve Bayes generally benefits from kernel density estimation (usekernel = TRUE), except in the baseline model. KNN performs best with k = 1 under most resampling methods, suggesting that synthetic data enhances nearest-neighbor classification. Neural networks prefer larger architectures (size = 7), with slight variations in decay values depending on the resampling approach. Meanwhile, AdaBoost shows fluctuations in nIter, with oversampling favoring more iterations (100), while RUS achieves optimal performance with fewer iterations (20).

These findings demonstrate that resampling techniques significantly impact hyperparameter tuning, affecting model complexity and generalization. Although computational time was not explicitly recorded, the hyperparameter search space was designed with efficiency. For instance, simpler configurations such as Random Forest with lower mtry values or AdaBoost with fewer boosting iterations offered faster training times, making them more suitable for practical deployment in resource-constrained educational environments. The next section will further examine how these optimized models perform in classification tasks.

Table 5. Best hyperparameters tuning of machine learning techniques for each resampling method

Methods	Best Hyperparameters tuning					
	Baseline	RUS	ROS	Both	Smotenc	Smotenc-Rus
Random Forest	mtry = 5.	mtry = 4.	mtry = 5.	mtry = 5.	mtry = 5.	mtry = 5.
SVM Radial	sigma = 0.1, C = 1	sigma = 0.1, C = 1	sigma = 0.9, C = 1	sigma = 0.4, C = 1	sigma = 0.1, C = 1	sigma = 0.1, C = 1
XGBTree	nrounds = 95, max_depth = 4, eta = 0.1, gamma = 1, colsample_bytree = 0.6, min_child_weight = 1, subsample = 0.9.	nrounds = 75, max_depth = 4, eta= 0.1, gamma = 2, colsample_bytree = 0.7, min_child_weight = 1, subsample = 0.7.	nrounds = 95, max_depth = 5, eta = 0.1, gamma = 1, colsample_bytree = 0.8, min_child_weight = 1, subsample = 0.7.	nrounds = 95, max_depth = 5, eta = 0.1, gamma = 2, colsample_bytree = 0.8, min_child_weight = 1, subsample = 0.8.	nrounds = 95, max_depth = 5, eta = 0.1, gamma = 1, colsample_bytree = 0.6, min_child_weight = 1, subsample = 0.9.	nrounds = 95, max_depth = 5, eta = 0.1, gamma = 2, colsample_bytree = 0.8, min_child_weight = 1, subsample = 0.9.

p-ISSN: 2528-1682
e-ISSN: 2527-9165

Methods	Best Hyperparameters tuning					
	Baseline	RUS	ROS	Both	Smotenc	Smotenc-Rus
Naïve Bayes	fL = 0.01, usekernel = FALSE, adjust = 0.01.	fL = 0.01, usekernel = TRUE, adjust = 0.1.	fL = 0.01, usekernel = TRUE, adjust = 0.1.	fL = 0.01, usekernel = TRUE, adjust = 1.	fL = 0.01, usekernel = TRUE, adjust = 0.1.	fL = 0.01, usekernel = TRUE, adjust = 0.1.
KNN	k = 9.	k = 7.	k = 1.	k = 1.	k = 1.	k = 1.
Neural Network	size = 3, decay = 0.06.	size = 1, decay = 0.09	size = 7, decay = 0.04.	size = 7, decay = 0.06.	size = 7, decay = 0.05.	size = 7, decay = 0.04.
AdaBoost	nIter = 90, method = Adaboost.MI.	nIter = 20, method = Adaboost.MI.	nIter = 100, method = Adaboost.MI.	nIter = 70, method = Adaboost.MI	nIter = 100, method = Adaboost.MI.	nIter = 100, method = Adaboost.MI

4.1.3. Machine learning performance

This section presents the comparative performance of machine learning models after applying resampling strategies and hyperparameter tuning. Table 6 summarizes the precision, recall, F1-score, accuracy, AUC, and average performance metrics for each model under different resampling conditions. To ensure comprehensive evaluation, five key metrics are used: precision, recall, F1-score, accuracy, and AUC. Relying on a single metric, such as accuracy, can be misleading, especially in imbalanced datasets, because it may mask poor performance on the minority class. By averaging multiple metrics, the evaluation becomes more balanced, reflecting overall model effectiveness.

In the context of this study, the F1-score is particularly important as it evaluates the model's ability to correctly identify students who graduate on time (the positive class), balancing the trade-off between precision and recall. A high F1-score means the model is both precise in its predictions and capable of identifying most students in the positive class. While precision emphasizes correct positive predictions and recall emphasizes complete positive coverage, the F1-score integrates both perspectives into a single harmonic mean, which is useful in high-stakes educational decision-making.

Table 6. Performance statistics of machine learning techniques with resampling methods

Algorithm	Resample	Precision	Recall	F1	Accuracy	AUC	Average
SVM-Radial	Baseline	0.843	0.967	0.901	0.840	0.837	0.878
	RUS	0.718	0.711	0.714	0.716	0.794	0.731
	ROS	0.855	0.919	0.885	0.881	0.931	0.894
	RUS-ROS	0.870	0.839	0.854	0.852	0.930	0.869
	SMOTENC	0.829	0.879	0.854	0.833	0.896	0.858
	SMOTENC-RUS	0.785	0.774	0.779	0.781	0.869	0.798
XGBTree	Baseline	0.867	0.964	0.913	0.862	0.878	0.897
	RUS	0.746	0.833	0.787	0.775	0.850	0.798
	ROS	0.836	0.897	0.866	0.861	0.940	0.880
	RUS-ROS	0.856	0.898	0.877	0.869	0.941	0.888
	SMOTENC	0.863	0.936	0.898	0.882	0.951	0.906
	SMOTENC-RUS	0.842	0.900	0.870	0.866	0.939	0.883
RandomForest	Baseline	0.876	0.967	0.920	0.873	0.890	0.905
	RUS	0.749	0.833	0.789	0.777	0.848	0.799
	ROS	0.943	0.910	0.926	0.928	0.980	0.937
	RUS-ROS	0.950	0.934	0.942	0.940	0.977	0.948
	SMOTENC	0.870	0.896	0.883	0.868	0.946	0.892
	SMOTENC-RUS	0.860	0.849	0.855	0.855	0.934	0.871
KNN	Baseline	0.828	0.974	0.895	0.829	0.846	0.874
	RUS	0.639	0.765	0.696	0.667	0.763	0.706
	ROS	0.954	0.886	0.919	0.922	0.922	0.921
	RUS-ROS	0.927	0.903	0.915	0.913	0.914	0.914
	SMOTENC	0.888	0.818	0.852	0.842	0.845	0.849
	SMOTENC-RUS	0.854	0.776	0.813	0.822	0.822	0.817

Algorithm	Resample	Precision	Recall	F1	Accuracy	AUC	Average
Naïve Bayes	Baseline	0.868	0.946	0.906	0.852	0.847	0.884
	RUS	0.635	0.946	0.760	0.701	0.829	0.774
	ROS	0.726	0.917	0.810	0.785	0.871	0.822
	RUS-ROS	0.686	0.965	0.802	0.753	0.865	0.814
	SMOTENC	0.820	0.940	0.876	0.852	0.928	0.883
	SMOTENC-RUS	0.795	0.876	0.833	0.825	0.911	0.848
Neural Network	Baseline	0.877	0.953	0.913	0.864	0.856	0.893
	RUS	0.728	0.838	0.779	0.762	0.833	0.788
	ROS	0.807	0.832	0.820	0.817	0.900	0.835
	RUS-ROS	0.822	0.842	0.832	0.824	0.902	0.844
	SMOTENC	0.850	0.865	0.857	0.840	0.904	0.863
	SMOTENC-RUS	0.774	0.800	0.787	0.783	0.871	0.803
adaBoost	Baseline	0.839	0.860	0.850	0.771	0.611	0.786
	RUS	0.724	0.784	0.753	0.743	0.739	0.749
	ROS	0.876	0.853	0.865	0.866	0.794	0.851
	RUS-ROS	0.849	0.865	0.857	0.851	0.774	0.839
	SMOTENC	0.832	0.840	0.836	0.817	0.748	0.815
	SMOTENC-RUS	0.793	0.825	0.808	0.805	0.741	0.794

As shown in Table 6, Random Forest consistently yielded the highest F1-scores across most resampling techniques. Its best performance was recorded with the RUS-ROS method (F1 = 0.942), followed by ROS (F1 = 0.926). Compared to its baseline performance (F1 = 0.920), this represents a measurable improvement, highlighting the benefit of resampling. While the difference may appear numerically small (0.022), in practical terms it implies more reliable identification of students likely to graduate on time, a valuable outcome for early academic interventions. K-Nearest Neighbors (KNN) also demonstrated strong results, particularly under oversampling techniques. With F1-scores of 0.919 (ROS) and 0.915 (RUS-ROS), KNN benefited from increased minority class representation, which is crucial for instance-based learning algorithms. Similarly, XGBTree showed competitive F1-scores ranging from 0.787 (RUS) to 0.898 (SMOTENC), indicating its robustness across varying class distributions. Neural networks achieved solid performance under most resampling strategies, peaking at 0.913 (baseline) and 0.857 (SMOTENC). However, performance dropped under RUS (F1 = 0.779), suggesting sensitivity to reduced training data. Naïve Bayes achieved F1-scores of 0.906 (baseline) and 0.876 (SMOTENC) but was less stable under RUS and hybrid resampling. AdaBoost consistently performed the weakest among all models, with F1-scores ranging from 0.753 (RUS) to 0.865 (ROS), revealing its limited effectiveness in handling imbalanced data even with tuning.

While differences in F1-score between models may seem small (e.g., 0.01 to 0.02), they can translate to meaningful practical implications. For example, in a dataset of over 4,000 students, a 1% improvement in F1-score could affect the classification of dozens of students. This may influence how resources are allocated for advising, scholarship eligibility, or early warning systems. Therefore, small metric improvements, especially in F1, should not be overlooked in educational contexts.

Overall, the combination of Random Forest and RUS-ROS resampling emerges as the most effective configuration, achieving both high average metric scores (0.948) and the highest F1-score. This suggests that hybrid resampling paired with a robust tree-based model provides optimal performance in predicting on-time graduation. The next section will examine the statistical significance of these observed differences using PERMANOVA and pairwise post-hoc testing.

4.1.4. Statistical analysis results

The subsequent step involves statistically evaluating performance differences among the machine learning models. This study employs PERMANOVA due to the violation of the multivariate normality assumption required by multivariate analysis of variance (MANOVA). Table 7 presents the results of several multivariate normality tests, including the Mardia, Henze-Zirkler, Royston, and Doornik-Hansen tests and the E-statistic. All tests produced p-values less than 0.05, indicating significant deviations from multivariate normality. Consequently, the assumptions for applying MANOVA are not met, necessitating a non-parametric alternative.

As a solution, this study adopts Permutational Multivariate Analysis of Variance (PERMANOVA), a non-parametric method based on permutation tests and dissimilarity matrices. Unlike MANOVA, PERMANOVA does not assume multivariate normality. It calculates pseudo-F statistics by partitioning total variance across groups using distance matrices, with p-values obtained through repeated permutations of group labels to determine whether observed differences exceed those expected by chance.

Table 7. Multivariate normality test of MANOVA assumption

Method	Statistics	p value	MVN
Mardia's test (Kurtosis)	4.296	1.736e-05	NO
Henze-Zirkler's test	2.346	0.000	NO
Royston's test	10.627	0.018	NO
Doornik-Hansen's test	66.740	1.881e-10	NO
E-statistic	3.051	0.000	NO

Table 8. Permutational multivariate analysis of variance (PERMANOVA) results

Source	df	SS	R^2	F	p value
Model	6	0.287	0.311	2.634	0.009**
Residual	35	0.635	0.689		
Total	41	0.922	1		

Table 8 displays the PERMANOVA results. The model yielded an F value of 2.634 and a p-value of 0.009, indicating statistically significant differences between at least some machine learning model pairs. The R^2 value of 0.311 suggests that approximately 31.1% of the variation in model performance is attributable to algorithmic differences, while the remaining 68.9% may result from model-resampling interactions or random variation. Although this R^2 value is moderate, it is not uncommon in complex modeling contexts, particularly when multiple algorithms interact with heterogeneous resampling strategies and feature structures.

To further explore these differences, Bonferroni's post-hoc pairwise comparison tests were conducted, as presented in Table 9. Several algorithm pairs exhibited statistically significant performance differences. For instance, Random Forest significantly outperformed AdaBoost ($p = 0.006$, $R^2 = 0.519$), and XGBTree also showed superiority over AdaBoost ($p = 0.012$, $R^2 = 0.541$). Likewise, Naïve Bayes and Neural Network yielded significantly better results than AdaBoost, with p-values of 0.004 and 0.008, respectively, and R^2 values above 0.36. Additionally, Random Forest significantly outperformed Naïve Bayes ($p = 0.032$, $R^2 = 0.329$). These findings confirm AdaBoost's consistently weaker performance relative to the stronger and more stable outcomes produced by Random Forest and XGBTree.

However, not all comparisons revealed statistically significant differences. For example, comparisons between SVM-Radial and Random Forest ($p = 0.155$), SVM-Radial and XGBTree ($p = 0.243$), and XGBTree and Random Forest ($p = 0.530$) suggest these models perform similarly in practical terms. This is further supported by small effect sizes and overlapping distributions observed in performance visualizations. While statistical tests offer formal validation, practical relevance and effect sizes should also guide educational decision-making. In many cases, even small but consistent gains in classification performance, especially for predicting delayed graduation, can meaningfully inform institutional interventions.

Table 9. The Bonferronis' post-hoc results

Pairs	df	ss	F Model	R^2	p value
SVM-Radial vs XGBTree	1	0.023	1.423	0.125	0.243
SVM-Radial vs RandomForest	1	0.044	2.176	0.179	0.155
SVM-Radial vs KNN	1	0.007	0.211	0.021	0.781

Pairs	df	ss	F Model	R^2	p value
SVM-Radial vs Naïve Bayes	1	0.034	1.684	0.144	0.207
SVM-Radial vs Neural Network	1	0.000	0.018	0.002	0.994
SVM-Radial vs adaBoost	1	0.061	3.450	0.256	0.047
XGBTree vs RandomForest	1	0.008	0.573	0.054	0.530
XGBTree vs KNN	1	0.023	0.984	0.090	0.430
XGBTree vs Naïve Bayes	1	0.041	3.165	0.240	0.078
XGBTree vs Neural Network	1	0.022	2.284	0.186	0.155
XGBTree vs adaBoost	1	0.124	11.809	0.541	0.012
RandomForest vs KNN	1	0.035	1.267	0.112	0.304
RandomForest vs Naïve Bayes	1	0.083	4.911	0.329	0.032
RandomForest vs Neural Network	1	0.045	3.307	0.248	0.090
RandomForest vs adaBoost	1	0.156	10.799	0.519	0.006
KNN vs Naïve Bayes	1	0.051	1.871	0.158	0.176
KNN vs Neural Network	1	0.008	0.344	0.033	0.662
KNN vs adaBoost	1	0.048	1.895	0.159	0.170
Naïve Bayes vs Neural Network	1	0.028	2.075	0.172	0.145
Naïve Bayes vs adaBoost	1	0.098	6.940	0.410	0.004
Neural Network vs adaBoost	1	0.063	5.686	0.362	0.008

To complement the statistical analyses, two visualizations were created to provide a clearer overview of model performance. The first is a boxplot illustrating the average metrics for each machine learning technique, calculated as the mean of five key performance indicators: precision, recall, F1-score, accuracy, and AUC. As shown in Figure 1, Random Forest and XGBTree achieve the highest median values with tight interquartile ranges, indicating consistent performance across multiple resampling iterations. In contrast, AdaBoost exhibits the lowest median and broad variability, reinforcing its weaker and less stable performance. KNN and Naïve Bayes display relatively high average scores, albeit with greater variability compared to the top performers.

The second boxplot, presented in Figure 2, highlights the distribution of F1 scores across all models. This metric is particularly emphasized in this study due to its relevance in predicting timely graduation. The visualization reveals that Random Forest and XGBTree not only achieve high F1-scores but also maintain narrow spreads, signifying both strong and stable classification performance. Conversely, AdaBoost again records lower medians and wider variability, while SVM-Radial shows greater dispersion, suggesting less consistent outcomes across resampling iterations. These visualizations reinforce the statistical findings and offer practical insight into the consistency and robustness of each evaluated model.

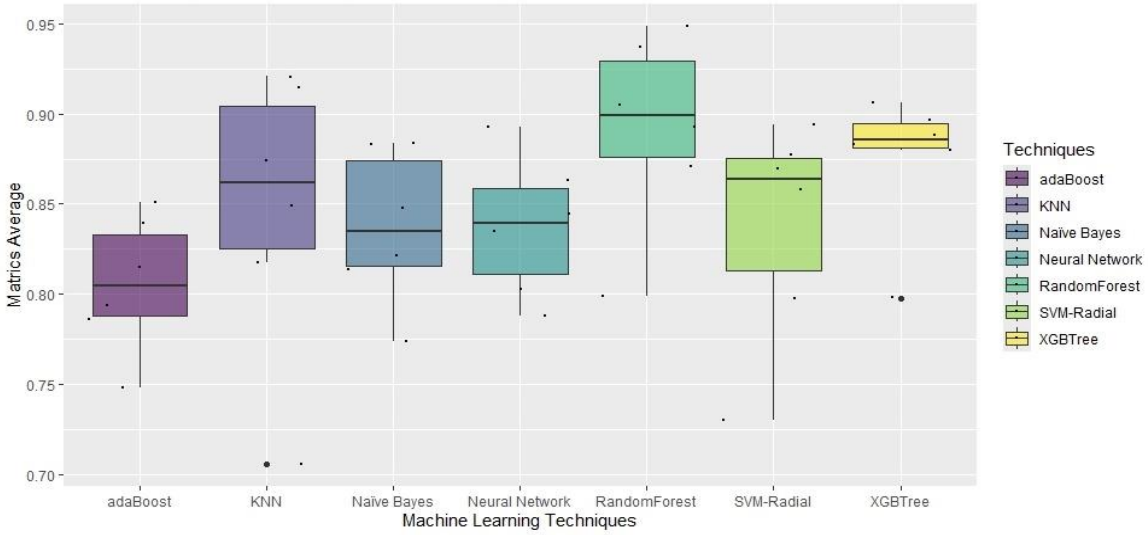

Figure 1. Boxplot of Machine learning performance based on average metric

p-ISSN: 2528-1682
e-ISSN: 2527-9165

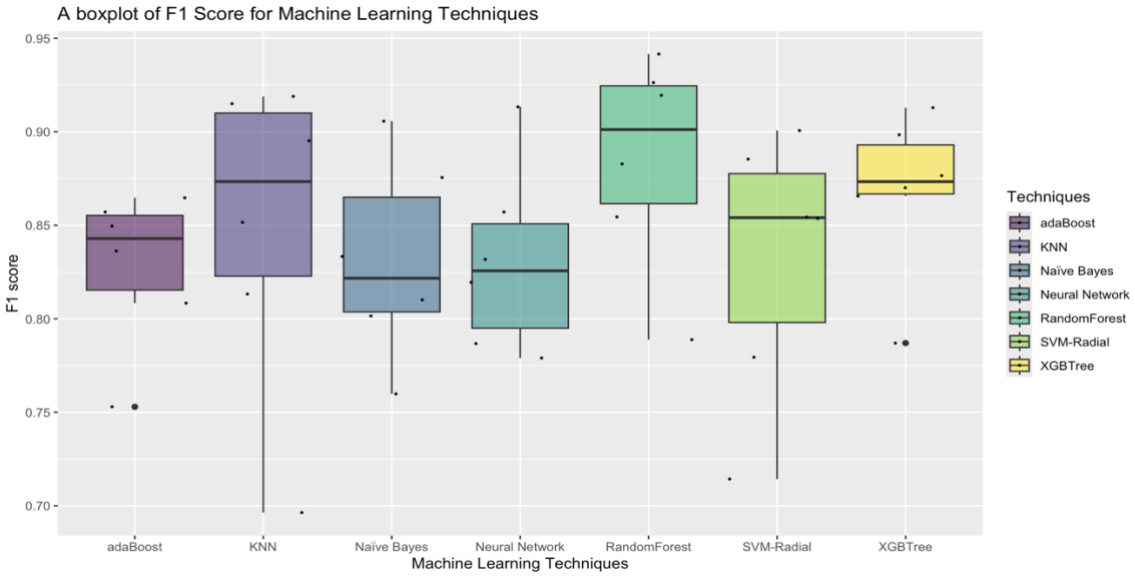

Figure 2. Boxplot of Machine learning performance based on F1 score

In summary, the combination of PERMANOVA, Bonferroni post-hoc analysis, and detailed performance visualizations provides a statistically grounded and practically relevant understanding of model differences. These insights ensure that the selected models are not only statistically sound but also meaningful when applied to real-world challenges in predicting timely student graduation.

4.2. Discussion

This study investigated the influence of resampling strategies and hyperparameter tuning on the performance of various machine learning algorithms for predicting student graduation on time. The results confirm that model performance significantly improves when class imbalance is properly addressed and hyperparameters are carefully optimized. Compared to baseline models, resampled datasets, particularly those using hybrid methods, demonstrated better classification accuracy and model robustness.

The findings are consistent with previous studies, such as [31], which identified Random Forest as a strong classifier for predicting on-time graduation using the same dataset. However, unlike [31], which did not apply any resampling strategies to address data imbalance, this study incorporates five different resampling techniques and hyperparameter tuning. This enhancement allows for a more realistic and fair model comparison, especially in contexts where class imbalance can lead to biased results. Similarly, while [37] introduced the hybrid RUS-ROS approach as a promising solution for imbalance problems, their study did not assess its performance across multiple machine learning algorithms nor validate its effectiveness through multivariate statistical testing. In contrast, this study demonstrates that combining RUS-ROS with Random Forest yields the highest average performance score (0.948), making it a highly effective configuration for imbalanced educational datasets.

A key insight from the performance evaluation is the role of the F1-score, which balances precision and recall and is especially important in predicting students who graduate on time, the positive class in this study. The results showed that Random Forest achieved the highest F1-score of 0.942 under the RUS-ROS combination, indicating its strong ability to accurately identify on-time graduates without sacrificing precision or recall.

This study's contribution also lies in its methodological rigor. Unlike previous works that relied on single-metric evaluations (e.g., [32] using accuracy), this research adopts a multivariate perspective by applying PERMANOVA across five key metrics: precision, recall, F1-score, accuracy, and AUC. The use of PERMANOVA, complemented by Bonferroni's post-hoc tests and performance visualizations, ensures a more robust.

Visual analysis using boxplots further supported the statistical findings. Random Forest and XGBoost not only achieved the highest median performance but also showed low variability across resampling methods. In contrast, AdaBoost exhibited both lower scores and greater inconsistency, reinforcing its weaker suitability for the task.

In summary, this research advances the field of educational data mining by proposing an integrated framework that combines advanced resampling, hyperparameter optimization, multivariate evaluation, and visual analysis. These innovations enhance the reliability and interpretability of predictive models for supporting data-driven academic decision-making, especially in identifying students at risk of delayed graduation.

5. CONCLUSION

This study directly addresses two critical challenges in predicting on-time graduation: the presence of class imbalance in educational data and the lack of hyperparameter optimization in prior research. By integrating five resampling strategies and seven machine learning algorithms, this work systematically evaluates how these factors influence model performance. The results demonstrate that the combination of Random Forest and the hybrid RUS-ROS resampling technique yields the highest average performance score (0.948), including a notably high F1-score of 0.942, indicating strong predictive accuracy for identifying students who graduate on time. These findings are supported by rigorous evaluation using both cross-validation and hold-out validation methods. Furthermore, this study introduces a multivariate statistical testing framework using PERMANOVA and Bonferroni's post-hoc pairwise comparisons, which confirms the significance of performance differences across models. These findings contribute to the field of educational data mining by offering a robust methodology for handling imbalanced data and optimizing predictive model performance in academic classification tasks.

While the results are encouraging, they are not without limitations. The reliability of the findings is supported by statistical validation, although the R^2 value of 31.1% suggests that a substantial portion of performance variability remains unexplained. Future research could explore more advanced resampling methods such as Borderline-SMOTE, ADASYN, or adaptive synthetic techniques to improve class balance. The adoption of deep learning architectures, model stacking, or AutoML frameworks may also enhance predictive accuracy and scalability in future implementations.

Finally, the proposed framework may be extended to other educational data mining problems, such as dropout prediction, academic risk detection, or early warning systems for student disengagement. Applying this methodology to more diverse datasets, particularly those that integrate both cognitive and non-cognitive variables, could further strengthen data-driven decision-making in higher education institutions.

REFERENCES

[1] E. Fernandes, M. Holanda, M. Victorino, V. Borges, R. Carvalho, and G. Van Erven, "Educational data mining: Predictive analysis of academic performance of public school students in the capital of Brazil," J Bus Res, vol. 94, pp. 335–343, Jan. 2019, doi: 10.1016/J.JBUSRES.2018.02.012.

[2] C. Romero and S. Ventura, "Educational data mining and learning analytics: An updated survey," Wiley Interdiscip Rev Data Min Knowl Discov, vol. 10, no. 3, p. e1355, May 2020, doi: 10.1002/WIDM.1355.

[3] R. S. Baker and P. S. Inventado, "Educational Data Mining and Learning Analytics," Learning Analytics: From Research to Practice, pp. 61–75, Jan. 2014, doi: 10.1007/978-1-4614-3305-7_4.

[4] A. Peña-Ayala, "Educational data mining: A survey and a data mining-based analysis of recent works," Expert Syst Appl, vol. 41, no. 4, pp. 1432–1462, Mar. 2014, doi: 10.1016/J.ESWA.2013.08.042.

[5] A. M. Shahiri, W. Husain, and N. A. Rashid, "A Review on Predicting Student's Performance Using Data Mining Techniques," Procedia Comput Sci, vol. 72, pp. 414–422, Jan. 2015, doi: 10.1016/J.PROCS.2015.12.157.

[6] J. Stephen Bassi, E. Gbenga Dada, A. Abdulkadir Hamidu, M. Dauda Elijah, and C. Author, "Students Graduation on Time Prediction Model Using Artificial Neural Network," IOSR Journal of Computer Engineering , vol. 21, no. 3, pp. 28–35, 2019, doi: 10.9790/0661-2103012835.

[7] A. C. Lagman et al., "Embedding naïve bayes algorithm data model in predicting student graduation," ACM International Conference Proceeding Series, pp. 51–56, Nov. 2019, doi: 10.1145/3369555.3369570.

[8] A. Meiriza, E. Lestari, P. Putra, A. Monaputri, and D. A. Lestari, "Prediction Graduate Student Use Naive Bayes Classifier," vol. 172, pp. 370–375, May 2020, doi: 10.2991/AISR.K.200424.056.

[9] F. Nuraeni, Y. H. Agustin, S. Rahayu, D. Kurniadi, Y. Septiana, and S. M. Lestari, "Student Study Timeline Prediction Model Using Naïve Bayes Based Forward Selection Feature," 8th International Conference on ICT for Smart Society: Digital Twin for Smart Society, ICISS 2021 - Proceeding, Aug. 2021, doi: 10.1109/ICISS53185.2021.9532502.

[10] C. P. Kuncoro, "Analysis Of UMN Student Graduation Timeliness Using Supervised Learning Method," IJNMT (International Journal of New Media Technology), vol. 8, no. 2, pp. 89–95, Feb. 2021, doi: 10.31937/IJNMT.V8I2.2366.
[11] Gunawan, F. Halim, and Djoni, "Students' Timely Graduation Attributes Prediction Using Feature Selection Techniques, Case Study: Informatics Engineering Bachelor Study Program," ICOSNIKOM 2022 - 2022 IEEE International Conference of Computer Science and Information Technology: Boundary Free: Preparing Indonesia for Metaverse Society, 2022, doi: 10.1109/ICOSNIKOM56551.2022.10034873.
[12] D. Dikriani, A. Tahta, and I. Karim, "Comparison of C4.5 and Naive Bayes Algorithm Methods in Prediction of Student Graduation on Time (Case Study: Information Systems Study Program)," Journal of Dinda : Data Science, Information Technology, and Data Analytics, vol. 3, no. 1, pp. 40–44, Feb. 2023, doi: 10.20895/DINDA.V3I1.782.
[13] A. Santoso, H. Retnawati, Kartianom, E. Apino, I. Rafi, and M. N. Rosyada, "Predicting Time to Graduation of Open University Students: An Educational Data Mining Study," Open Education Studies, vol. 6, no. 1, Jan. 2024, doi: 10.1515/EDU-2022-0220/MACHINEREADABLECITATION/RIS.
[14] B. Jia et al., "Prediction for Student Academic Performance Using SMNaive Bayes Model," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 11888 LNAI, pp. 712–725, 2019, doi: 10.1007/978-3-030-35231-8_52.
[15] A. P. Salim, K. A. Laksitowening, and I. Asror, "Time Series Prediction on College Graduation Using KNN Algorithm," 2020 8th International Conference on Information and Communication Technology, ICoICT 2020, Jun. 2020, doi: 10.1109/ICOICT49345.2020.9166238.
[16] K. T. Chui, D. C. L. Fung, M. D. Lytras, and T. M. Lam, "Predicting at-risk university students in a virtual learning environment via a machine learning algorithm," Comput Human Behav, vol. 107, p. 105584, Jun. 2020, doi: 10.1016/J.CHB.2018.06.032.
[17] S. Noviaristanti, G. Ramantoko, A. T. Hadi, and A. Inayati, "Predictive Model of Student Academic Performance in Private Higher Education Institution (Case in Undergraduate Management Program)," 2022 International Conference on Data Science and Its Applications, ICoDSA 2022, pp. 262–267, 2022, doi: 10.1109/ICODSA55874.2022.9862822.
[18] Y. Crismayella, N. Satyahadewi, and H. Perdana, "Comparison of Adaboost Application to C4.5 and C5.0 Algorithms in Student Graduation Classification," Pattimura International Journal of Mathematics (PIJMath), vol. 2, no. 1, pp. 07–16, Apr. 2023, doi: 10.30598/PIJMATHVOL2ISS1PP07-16.
[19] H. Altabrawee, O. Abdul, J. Ali, and Q. Ajmi, "Predicting Students' Performance Using Machine Learning Techniques," JOURNAL OF UNIVERSITY OF BABYLON for Pure and Applied Sciences, vol. 27, no. 1, pp. 194–205, Apr. 2019, doi: 10.29196/JUBPAS.V27I1.2108.
[20] N. Mohammad Suhaimi, S. Abdul-Rahman, S. Mutalib, N. H. Abdul Hamid, and A. Md Ab Malik, "Predictive Model of Graduate-On-Time Using Machine Learning Algorithms," Communications in Computer and Information Science, vol. 1100, pp. 130–141, 2019, doi: 10.1007/978-981-15-0399-3_11/COVER.
[21] M. Windarti and P. T. Prasetyaninrum, "Prediction Analysis Student Graduate Using Multilayer Perceptron," pp. 53–57, May 2020, doi: 10.2991/ASSEHR.K.200521.011.
[22] N. Suresh, V. Hashiyana, G. T. Nhinda, I. Stephanus, and P. Kautwima, "Graduates' Prediction System Using Artificial Intelligence," ACM International Conference Proceeding Series, pp. 317–327, Aug. 2021, doi: 10.1145/3484824.3484873.
[23] D. Ruete et al., "Early Detection of Delayed Graduation in Master's Students," ASEE Annual Conference and Exposition, Conference Proceedings, Jul. 2021, doi: 10.18260/1-2--36999.
[24] G. Gunawan, H. Hanes, and C. Catherine, "C4.5, K-Nearest Neighbor, Naïve Bayes, and Random Forest Algorithms Comparison to Predict Students' On Time Graduation," Indonesian Journal of Artificial Intelligence and Data Mining, vol. 4, no. 2, pp. 62–71, Nov. 2021, doi: 10.24014/IJAIDM.V4I2.10833.
[25] J. Mantik, Y. Yennimar, M. R. Faturrahman, S. Nesen, M. A. Guci, and S. R. Pasaribu, "Implementation of artificial neural network and support vector machine algorithm on student graduation prediction model on time," Jurnal Mantik, vol. 7, no. 2, pp. 925–934, Aug. 2023, doi: 10.35335/MANTIK.V7I2.3992.
[26] A. Desfiandi and B. Soewito, "STUDENT GRADUATION TIME PREDICTION USING LOGISTIC REGRESSION, DECISION TREE, SUPPORT VECTOR MACHINE, AND ADABOOST ENSEMBLE LEARNING," IJISCS (International Journal of Information System and Computer Science), vol. 7, no. 3, pp. 195–199, Oct. 2023, doi: 10.56327/IJISCS.V7I2.1579.
[27] A. Sadqui, M. Ertel, H. Sadiki, and S. Amali, "Evaluating Machine Learning Models for Predicting Graduation Timelines in Moroccan Universities," International Journal of Advanced Computer Science and Applications, vol. 14, no. 7, pp. 304–310, 2023, doi: 10.14569/IJACSA.2023.0140734.
[28] H. J. P. Weerts, A. C. Mueller, and J. Vanschoren, "Importance of Tuning Hyperparameters of Machine Learning Algorithms," Jul. 2020, doi: 10.48550/arxiv.2007.07588.
[29] Z. Bitar and A. Al-Mousa, "Prediction of Graduate Admission using Multiple Supervised Machine Learning Models," Conference Proceedings - IEEE SOUTHEASTCON, vol. 2020-March, Mar. 2020, doi: 10.1109/SOUTHEASTCON44009.2020.9249747.
[30] J. M. Aiken, R. de Bin, M. Hjorth-Jensen, and M. D. Caballero, "Predicting time to graduation at a large enrollment American university," PLoS One, vol. 15, no. 11, p. e0242334, Nov. 2020, doi: 10.1371/JOURNAL.PONE.0242334.
[31] R. Bakri, N. P. Astuti, and A. S. Ahmar, "Machine Learning Algorithms with Parameter Tuning to Predict Students' Graduation-on-time: A Case Study in Higher Education," Journal of Applied Science, Engineering, Technology, and Education, vol. 4, no. 2, pp. 259–265, Dec. 2022, doi: 10.35877/454RI.ASCI1581.
[32] R. Ghorbani and R. Ghousi, "Comparing Different Resampling Methods in Predicting Students' Performance Using Machine Learning Techniques," IEEE Access, vol. 8, pp. 67899–67911, 2020, doi: 10.1109/ACCESS.2020.2986809.
[33] T. Wongvorachan, S. He, and O. Bulut, "A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining," Information 2023, Vol. 14, Page 54, vol. 14, no. 1, p. 54, Jan. 2023, doi: 10.3390/INFO14010054.

[34] H. Brdesee, W. Alsaggaf, N. Aljohani, and S. U. Hassan, "Predictive Model Using a Machine Learning Approach for Enhancing the Retention Rate of Students At-Risk," https://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/IJSWIS.299859, vol. 18, no. 1, pp. 1–21, Jan. 2020, doi: 10.4018/IJSWIS.299859.

[35] A. Anggrawan, H. Hairani, and C. Satria, "Improving SVM Classification Performance on Unbalanced Student Graduation Time Data Using SMOTE," International Journal of Information and Education Technology, vol. 13, no. 2, pp. 289–295, Feb. 2023, doi: 10.18178/IJIET.2023.13.2.1806.

[36] H. S. Bako, F. U. Ambursa, B. S. Galadanci, and M. Garba, "PREDICTING TIMELY GRADUATION OF POSTGRADUATE STUDENTS USING RANDOM FORESTS ENSEMBLE METHOD," FUDMA JOURNAL OF SCIENCES, vol. 7, no. 3, pp. 177–185, Jul. 2023, doi: 10.33003/fjs-2023-0703-1773.

[37] R. Bakri, N. P. Astuti, & Ansari, and S. Ahmar, "Evaluating Random Forest Algorithm in Educational Data Mining: Optimizing Graduation on-time prediction using Imbalance Methods," ARRUS Journal of Social Sciences and Humanities, vol. 4, no. 1, pp. 108–116, Feb. 2024, doi: 10.35877/SOSHUM2449.

[38] H. Hassan, N. B. Ahmad, and S. Anuar, "Improved students' performance prediction for multi-class imbalanced problems using hybrid and ensemble approach in educational data mining," J Phys Conf Ser, vol. 1529, no. 5, p. 052041, May 2020, doi: 10.1088/1742-6596/1529/5/052041.

[39] M. Mukherjee and M. Khushi, "SMOTE-ENC: A Novel SMOTE-Based Method to Generate Synthetic Data for Nominal and Continuous Features," Applied System Innovation 2021, Vol. 4, Page 18, vol. 4, no. 1, p. 18, Mar. 2021, doi: 10.3390/ASI4010018.

[40] N. Lunardon, G. Menardi, and N. Torelli, "ROSE: a Package for Binary Imbalanced Learning," R Journal, vol. 6, no. 1, pp. 82–92, 2014.

[41] E. Hvitfeldt, "themis: Extra Recipes Steps for Dealing with Unbalanced Data." [Online]. Available: https://cran.r-project.org/package=themis

[42] M. Kuhn, "caret: Classification and Regression Training," 2020. [Online]. Available: https://github.com/topepo/caret/

[43] A. Liaw and M. Wiener, "Classification and Regression by randomForest," R News, vol. 2, no. 3, pp. 18–22, 2022, [Online]. Available: https://cran.r-project.org/package=randomForest

[44] Maciej Serda et al., "MVN: An R Package for Assessing Multivariate Normality," R JOURNAL, vol. 6, no. 2, pp. 343–354, 2014, doi: 10.2/JQUERY.MIN.JS.

[45] J. Oksanen et al., "Community Ecology Package [R package vegan version 2.6-10]," CRAN: Contributed Packages, Jan. 2025, doi: 10.32614/CRAN.PACKAGE.VEGAN.

[46] M. J. Anderson, "A new method for non-parametric multivariate analysis of variance," Austral Ecol, vol. 26, no. 1, pp. 32–46, Feb. 2001, doi: 10.1111/J.1442-9993.2001.01070.PP.X.

[47] F. A. Al-Abdullatif, M. A. Al-Abdullatif, and G. Brooks, "MANOVA Post Hoc Techniques Used in Published Articles: A Systematic Review," General Linear Model Journal, vol. 45, no. 1, pp. 4–11, Mar. 2019, doi: 10.31523/GLMJ.045001.002.