Dissertation on
Predicting Students' Dropout using ensemble machine learning methods and hyperparameter optimization

Thesis submitted towards partial fulfilment
of the requirements for the degree of
Master of Technology in IT (Courseware Engineering)

Submitted by
CHANDRA SARKAR

EXAMINATION ROLL NO. M4CWE23005
UNIVERSITY REGISTRATION NO. 160375 of 2021-22

Under the guidance of
Dr. SASWATI MUKHERJEE
School of Education Technology
Jadavpur University

Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata-700032
India
2023

Master of Technology in IT (Courseware Engineering)
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled **"Predicting Students' Dropout using ensemble machine learning methods and hyperparameter optimization**" is a bonafide work carried out by **Chandra Sarkar** under our supervision and guidance for partial fulfilment of the requirements for the degree of **Master of Technology in IT (Courseware Engineering) (Courseware Engineering) in School of Education Technology, 2021-2023.**

SUPERVISOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DIRECTOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DEAN – FISLM
Jadavpur University
Kolkata-700 032

Master of Technology in IT (Courseware Engineering)
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF APPROVAL **

This foregoing thesis is hereby approved as a credible study of an engineering subject carried out and presented in a manner satisfactory to warranty its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not endorse or approve any statement made or opinion expressed or conclusion drawn therein but approve the thesis only for purpose for which it has been submitted.

Committee of final examination for evaluation of Thesis ----------

---

** Only in case the thesis is approved.

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains a literature survey and original research work by the undersigned candidate, as part of his/her Master of Technology in IT (Courseware Engineering)/Master in Multimedia Development studies.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by this rule and conduct, I have fully cited and referenced all materials and results that are not original to this work.

NAME: CHANDRA SARKAR

EXAMINATION ROLL NUMBER: M4CWE23005

THESIS TITLE: Predicting Students' Dropout using ensemble machine learning methods and hyperparameter optimization

SIGNATURE: DATE: / /

Acknowledgement

I would like to express my special thanks of gratitude to Dr Saswati Mukherjee who guided me throughout the whole thesis.

This thesis also helped me to gain a lot of knowledge in machine learning and data mining which are under the umbrella of artificial intelligence.

I am also thankful to Dr Matangini Chattopadhyay and Mr Joydeep Mukherjee for their valuable suggestions throughout my thesis.

My thanks and appreciation to my classmates from M. Tech in IT (Courseware Engineering) and Master in Multimedia Development.

Finally, my thanks to my parents who have always been with me and for their constant support.

Chandra Sarkar
M.Tech in IT (Courseware Engineering)
School of Education Technology
Jadavpur University
Kolkata:700032

Date: / /

Contents

	Page no.
Executive Summary	1
1. Introduction 1.1. Problem Statement 1.2. Objective 1.3. Background Concept	2-9
2. Literature Survey	10-11
3. Proposed Approach 3.1. Dataset Description 3.2. Data Pre-processing 3.3. Methodology	12-18
4. Experimental Results	19-21
5. Comparative Analysis	22
6. Conclusion and Future Scope	23
Reference	24
Appendix	25-37

Executive Summary

Education is the backbone of any economy. For any developing country especially India, education is very important. Education makes a person stable and prosperous in his way of leading life. In the same way, the number of higher educated persons in a country can contribute to the development and progress of the country. However, this number decreases due to the dropout of learners without completing the academic course. The dropout rate refers to the percentage of students who leave an educational program before its completion. Various reasons contribute to the dropout percentage, such as academic difficulties, personal or family reasons, financial problems, and lack of interest in studies. Dropout rates vary by region, socio-economic status, gender, and level of education in India, also students from lower socio-economic backgrounds are more likely to drop out due to financial constraints or the need to work to support their families. A student completing the course acquires improved skills and knowledge in comparison to the students who are leaving the course before completion.

A machine learning-based predictive model is developed to classify students who are opting out the university course before its completion. A students' dataset is considered to build the predictive model that consists of essential information about the students. Data pre-processing followed by statistical-based feature selection is performed on the dataset. The machine learning algorithms such as Support Vector Machine/SVM, K-Nearest Neighbour/KNN, Random Forest/RF, Decision Tree/DT, XGBoost, Naïve Bayes/NB, and Logistic Regression/LR are applied to build the classifiers. After building the classifiers, hyperparameters are adjusted to achieve classification accuracy.

1. Introduction

It is frequently observed that university students leave their courses before their completion not only in India but also in other countries. This dropout rate is affected by a number of factors, including academic challenges, personal or family issues, money issues, and a lack of enthusiasm for studies. Dropout rates also differ by location, socioeconomic position, gender, and educational level. Students from lower socioeconomic backgrounds are more likely to drop out due to financial difficulties or the necessity to work to support their families. To address the issues of students' drop-out, an ensemble of various machine learning-based prediction models is implemented to classify students who leave the university course before its completion. In addition, the relevant characteristics data relating to their dropout are also taken into consideration while building the prediction model. The classifiers are implemented and a comparative evaluation is performed to assess the classification accuracy.

1.1. Problem Statement

Predicting students' dropout using ensemble machine learning methods and hyperparameter optimization.

1.2. Objective

a) To classify students dropping out from University courses and to identify the underlying reasons for their drop-out.
b) To select the correlated features with respect to the prediction outcome that would enhance the predictive quality of the machine learning models.
c) To adjust the hyperparameters of the predictive models for classification accuracy and optimal loss function.

1.3 Background concept

The following is the description of the supervised classification models considered in this work.

KNN- KNN stands for k-nearest neighbours used for classification and regression analysis given in Figure 1. In this algorithm, a new data point is classified based on its proximity to the other known data points in the dataset. The term "k" refers to the number of nearest neighbours used to make a classification decision.

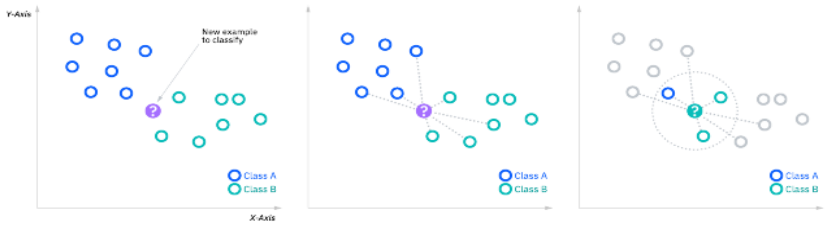

Figure1: K-Nearest Neighbour

Support Vector Machine (SVM): Support Vector Machine is used for classification and regression tasks. In SVM, the algorithm tries to find the best possible line i.e., hyperplane that separates the different classes of data points as shown in Figure 2. The data points that are closest to the line are called support vectors.

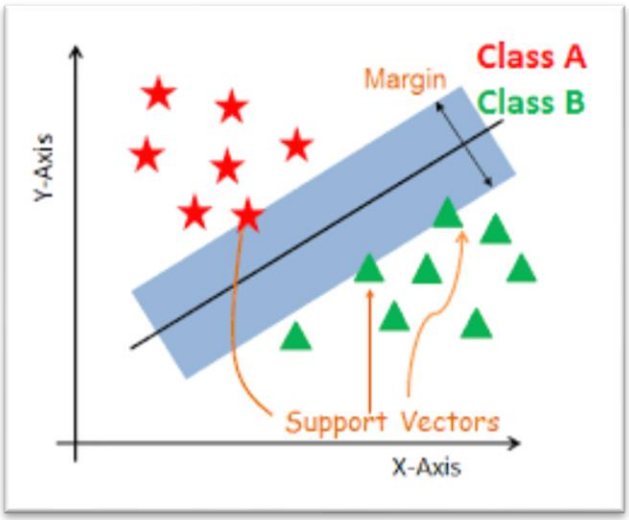

Figure 2: Support Vector Machine

Naïve Bayes (NB): Naive Bayes calculates the probability of a data point belonging to a certain class based on the occurrence of its features or attributes. The features are independent of each other which simplifies the calculations and makes the algorithm efficient as shown in Figure 3.

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

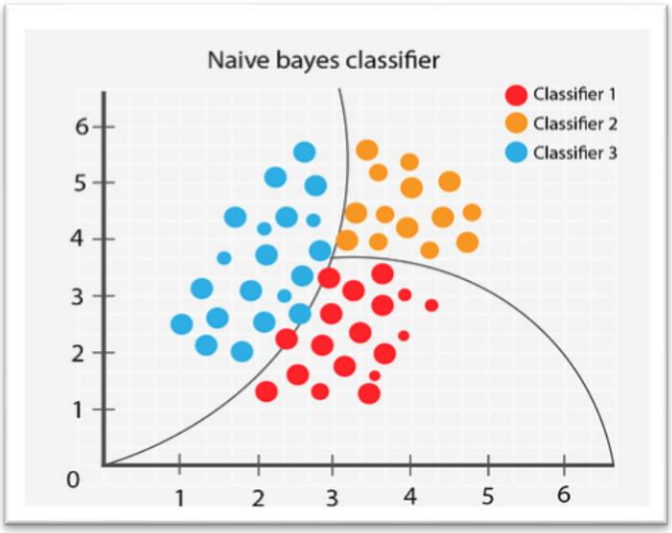

Figure 3: Naïve Bayes

Random Forest (RF): Random Forest is used for both classification and regression tasks. Figure 4 presents an ensemble learning method that combines multiple decision trees on randomly selected subsets of the training data, and then the average or majority vote of the predictions of all the trees is computed to make the final prediction. Each tree is built using a random subset of the features, which helps to reduce overfitting.

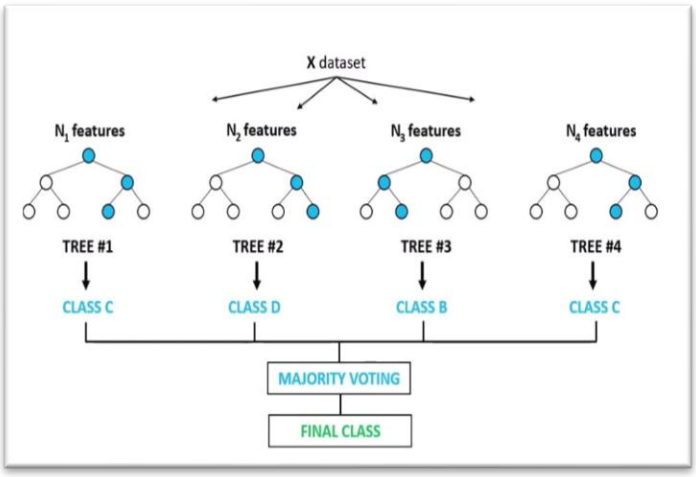

Figure 4: Random Forest

Decision Tree (DT): Decision Tree is used for both classification and regression tasks. It uses a set of *if-then* rules to make decisions. The basic idea is to divide the data into smaller subsets based on the features and each split is chosen by selecting the best feature that separates the data using certain criterion as shown in Figure 5.

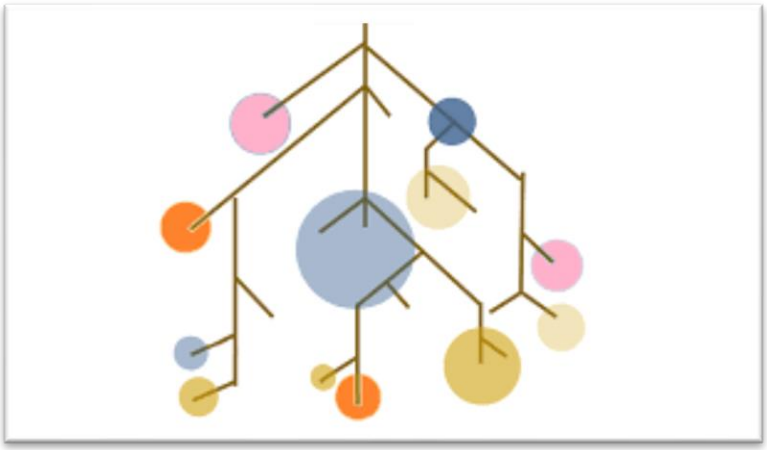

Figure 5: Decision Tree

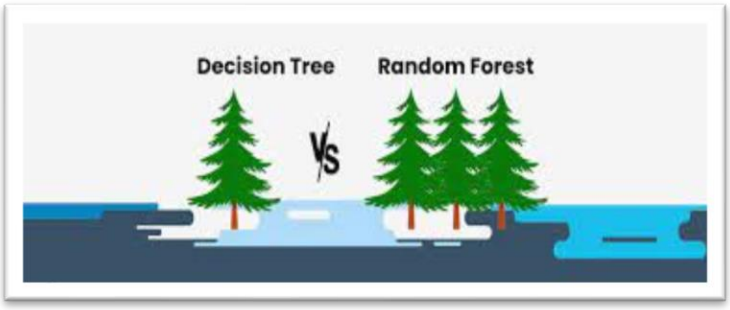

Figure 6: Decision Tree vs. Random Forest

Logistic Regression (LR): Logistic Regression is used for binary classification tasks, where the output variable is categorical with two possible values (e.g., 0 or 1). It considers the probability of the output variable as a function of the input variables between 0 and 1 presented in Figure 7. If the probability is greater than 0.5, the output is predicted as 1, otherwise it is predicted as 0.

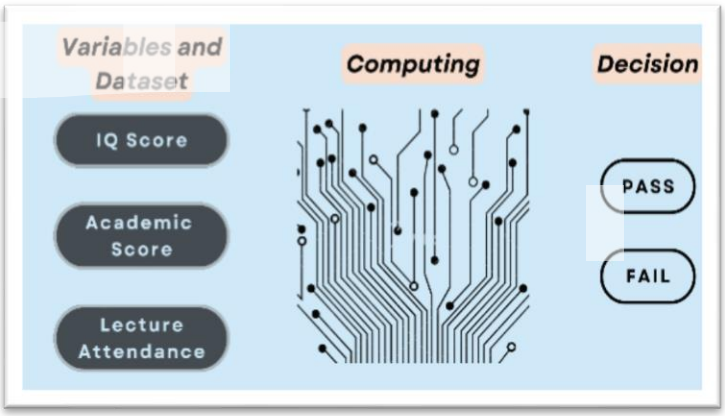

Figure 7: Logistic Regression

Extreme Gradient Boosting (XGBoost)- XGBoost is an ensemble learning method that combines the predictions of multiple weak models to produce a stronger prediction by handling large datasets. It trains weak models (decision trees) sequentially, where each new model focuses on correcting the errors made by the previous models as shown in Figure 8.

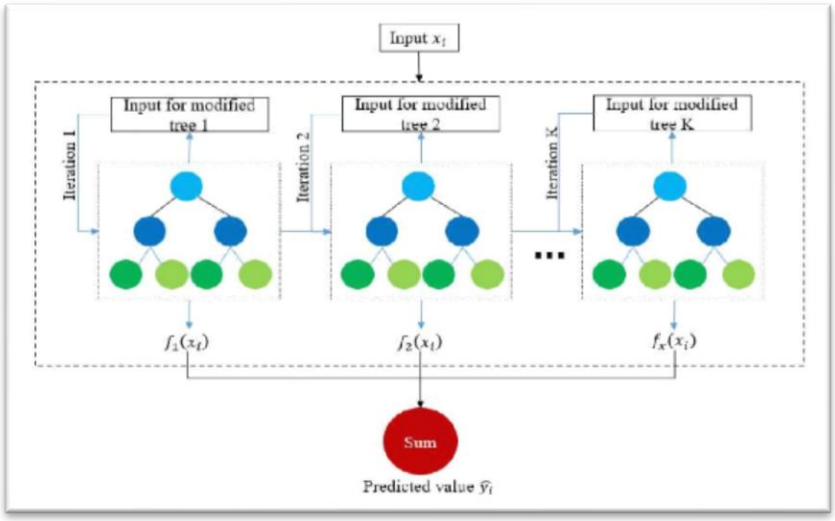

Figure 8: XGBoost

Statistical-based Feature Selection

The biggest challenge in machine learning is the selection of the best features to train the model. Statistical-based feature selection helps in sorting out this problem. It is the process of reducing the number of input features to minimize the cost of modelling and improve the model performance. Statistical-based feature selection methods involve evaluating the statistical relationship between each input and target variable and selecting those input variables that have the strongest relationship with the target variable.

Pearson Correlation Coefficient: This measures the linear relationship between two variables from -1 to 1, with a value of 0 indicating no correlation and values close to -1 or 1 indicating a strong negative or positive correlation. Correlation indicates the extent to which two or more variables fluctuate together within -1 to 1. The closer to 1 the more powerful the connection between the two variables, the lower the connection, the relationship is closer to 0.

$$r_{xy} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

- r_{xy}- the correlation coefficient of the linear relationship between the variables x and y
- x_i - the values of the x-variable in a sample
- $\bar{x}$ - the mean of the values of the x-variable
- y_i - the values of the y-variable in a sample
- $\bar{y}$ - the mean of the values of the y-variable

Mutual Information: This measures the amount of information that one attribute provides about the target variable. It is a statistical measure used to quantify the mutual dependence i.e. relationship between two random variables. In the context of feature selection, It ranges from

0 to 1, with a value of 0 indicating no information and a value of 1 indicating complete information.

$I(X; Y) = \Sigma\Sigma\ P(x, y) * \log(P(x, y) / (P(x) * P(y)))$

- $I(X; Y)$ - represents the mutual information between variables X and Y
- $P(x, y)$ - the joint probability mass function of variables X and Y
- $P(x)$ and $P(y)$ - the marginal probability mass functions of variables X and Y

Chi-squared Test: The chi-squared (χ^2) test is a statistical test used to determine the association between two categorical variables by measuring the dependence between each feature and the target variable. This measures the independence between two categorical variables. High Chi-Squared value indicates that the hypothesis of independence is incorrect.

$$\chi^2 = \Sigma\ [(O_i - E_i)^2 / E_i]$$

- χ^2 - the chi-squared statistic
- Σ -represents the summation over all categories
- O_i - the observed frequency (count) for each category
- E_i - the expected frequency (count) for each category

ANOVA Test: It tests whether the means of two or more groups are significantly different from each other. Features with low p-values are considered significant and are selected as relevant features. There are variations of the ANOVA test, such as one-way ANOVA, two-way ANOVA, and repeated measures ANOVA, which have different formulae and requirements depending on the specific analysis being performed.

$F = (SSB / df_B) / (SSW / df_W)$

- F - F-statistic measures the ratio of between-group variability to within-group variability
- SSB - Sum of squares which represents the variability between the group means
- df_B - Degrees of freedom for the between-group variability i.e., equal to the number of groups minus
- SSW - Sum of squares which represents the variability within each group
- df_W - Degrees of freedom for the within-group variability i.e., equal to the total number of observations minus the number of groups

Spearman's Rank Correlation Coefficient – It is a statistical measure that determines the monotonic relationship between two variables. If the relationship is not monotonic, Spearman's rank correlation may not be appropriate to use. It is often employed in feature selection to determine the importance of variables in a dataset. The formula for Spearman's rank correlation coefficient is as follows:

$$\rho = 1 - [(6 * \Sigma d^2) / (n * (n^2 - 1))]$$

ρ (rho) - Spearman's rank correlation coefficient.

- Σd^2 - sum of the squared differences between the ranks of corresponding pairs of variables
- n - number of observations

Kendall's Rank Correlation Coefficient or Kendall's Tau

It is a statistical measure determine the association between two variables. It is used in feature selection to determine the relevance of variables for a particular task. The value of Kendall's tau ranges between -1 and 1. A positive value indicates a positive association and a negative value indicates a negative association. The closer the value is to -1 or 1, the stronger the association whereas the value of 0 indicates no association between the variables.

The formula for Kendall's tau is as follows:

$\tau = (P - Q) / \text{sqrt}((P + Q + T) * (P + Q + U))$

- τ - Kendall's tau coefficient
- P - denotes the pairs where the order of values is the same in both variables
- Q - Represents the number of pairs where the order of values is different in the two variables
- T - Represents the number of ties in variable 1
- U - Represents the number of ties in variable 2

Evaluation Metrics measure the classification performance. The following are the evaluation metrics commonly applied in classification models.

Confusion Matrix is used to evaluate the performance of a classification model by providing a summary of the predicted and actual class labels for a set of data points. The matrix is particularly useful when dealing with binary classification problems (two classes), but it can also be extended to multi-class problems. It consists of four cells, representing different combinations of predicted and actual class labels as shown in Figure 9. The confusion matrix is a 2x2 matrix that has four possible outcomes as follows.

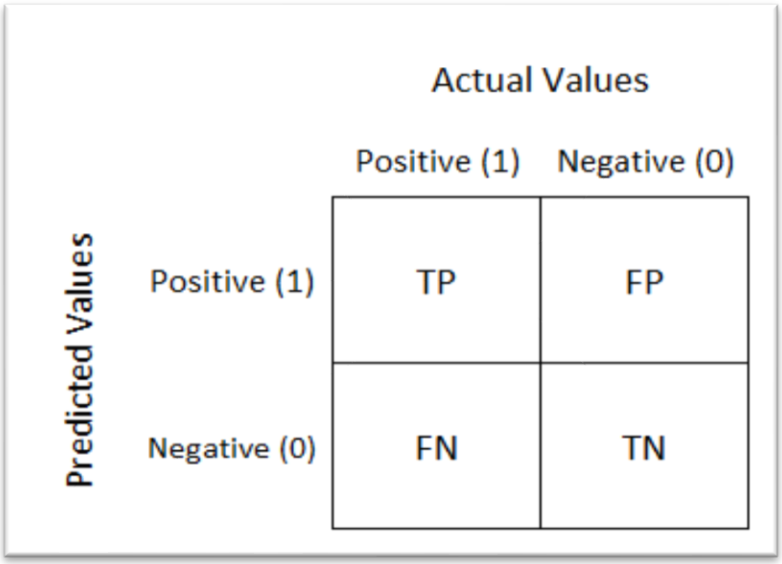

Figure 9: Confusion matrix

- True Positive (TP): The actual class is positive and the model predicted the class correctly as positive.
- False Positive (FP): The actual class is negative but the model predicted the class as positive.
- True Negative (TN): The actual class is negative and the model predicted the class correctly as negative.
- False Negative (FN): The actual class is positive but the model predicted the class as negative.

  Based on the TP, TN, FP and FN the following values are calculated.

F1-Score provides a more balanced evaluation of the model's ability to correctly classify both positive and negative instances. F1-score is most useful when the dataset is imbalanced which means the number of instances in each class is significantly different. F1 score = 2 * (precision * recall) / (precision + recall)

Recall is the number of true positives divided by the sum of true positives and false negatives. The recall is important in scenarios where the consequences of false negatives are significant. Recall = TP / (TP + FN)

Precision is the number of true positives divided by the sum of true positives and false positives. Precision is in situations where false positives have significant consequences. Precision = TP / (TP + FP)

Support represents the frequency of samples that are correctly classified as a specific class by a model by giving information about the distribution and occurrence of patterns in the dataset, which can be useful for understanding the data and identifying imbalances.

Accuracy is the ratio of correctly predicted observations to the total number of observations. Actually, it measures the model's ability to correctly classify both positive and negative instances. Accuracy = (TP + TN) / (TP + TN + FP + FN)

Hyperparameter Tuning refers to the process of finding the best values for the hyperparameters of a machine-learning model. Hyperparameters cannot be learned from the training data, but rather have to be set before training. Hyperparameter tuning is an iterative process that requires multiple experiments and evaluations to find the best hyperparameter values that maximize the model's performance on the specific task. Grid search and randomized search are two popular techniques for hyperparameter tuning in machine learning. Grid search is suitable when the hyperparameter space is small and discrete, or when we have prior knowledge about specific hyperparameter values that are likely to perform well but Randomized search is more efficient when the hyperparameter space is large and continuous, and we want to explore a wide range of values without testing every possible combination.

2. Literature Survey

Sravani et al. [1] presented an analysis of predicting student's academic performance by applying linear regression model. In this regard a student's dataset is collected to develop a model that can predict student's performance by considering some background attributes present in the student's dataset. Academic scores were taken as target variable(Y), and gender, age, mother job, parent education, test preparation, father job, parent status, travel time, absent days were taken as independent variables (X). Based on this a linear regression model was built which gave confidence of 95%.

Six different ensemble models containing Logistic Regression/LR, Support Vector Machine/SVM, Naive Bayes/NB, Decision Tree/DT, Neural Networks/NN, and Random Forest/RF algorithms was proposed[2]to predict students who are in danger of leaving institutions by recognizing each student's individual needs and dropout issues after which adjustments are made to include successful dropout prevention techniques. Each of the employed classification algorithms were quite effective in detecting positive class occurrences. NB gave precision of 97%, LR=96%, NB=92%, RF, SVM, DT gave precision as 94%. NB, SVM, DT, LR gave recall value 0f 98% whereas NN and RF had recall value as 96 and 92%. Random Forest, logistic regression, Decision tree and neural network classifiers were the most accurate models in this regard.

Gill et al. [3] proposed a way of predicting student dropout rate using indicators based on the academic and demographic data, running series of tests using the Weka Machine Learning Tool by applying different approaches of data mining algorithms. To identify the drop-out rate data mining approaches were used based on J4.8 and Naïve Bayes algorithms. These two classification algorithms were trained and tested using a ten-fold cross-validation approach. Naïve Bayes and J4.8 Bayes gave accuracy of 96% and 97% respectively. It is observed that J4.8 model is more accurate in predicting student dropout cases based on students' data collection.

A model based on Random Forest/RF algorithm was proposed[4] to identify students who are pursuing computer science and have a likelihood of dropping out without finishing the course. RF with 5-fold and 10-fold cross-validation is applied for predicting drop out cases. Academic performance in the first-year programming course was found to be the strongest predictor of drop out. Early identification of students' intention to drop out would enable universities to intervene with focused strategies at individual level to ensure that these students continue the course and attain successful grades. RF5 and RF10 gave accuracy of 86% and 85% respectively.

Ujkani et al. [5] presented an application of logistic regression technique to predict whether a student will retain or dropout the studies based on data such gender, age, city, enrolment date, level of study, number of exams to be taken, exam passed, average grade, current status i.e. whether student is active, inactive, graduated or dropped out. The data from six academic years were used to train and test the model, built on the logistic regression classifier which gave accuracy of 90%, F1 score of 0.85, indicating that the model was performing well and the predictions results are reliable.

Nasiri et al. [6] presented an analysis to predict GPA of graduated students as well as analysing their performance in e-learning centres with regression and C5.0 algorithms. These methods

are strongly dependent on the distribution of the data so even small variations in data can create different conclusions. Further, by applying these methods Regression Analysis gave -Sum of Squares=102994.643, Mean Square= 8582.887, F-score=530.961and C5.0 algorithms gave an accuracy of 88%.

Devasia et al. [7] explained different data mining techniques for discovering impressive patterns from enormous volumes of data. It is mainly deployed in educational data analysis, especially in higher education where final examination marks can be predicted by analysing the past and current performance in the internal exams. Data mining techniques include associations, classifications, sequential patterns & clustering are used which are helpful in making effective academic decisions regarding students.

A model based on the KNN algorithm was proposed [8] which helps in the prediction of students dropping out of secondary education based on various related features of students. KNN method was applied to classify students into dropout or no-dropout categories, as well as finding their welfare and performance score to check the reasons for their dropout. These results would further assist teachers to counsel students who are at risk of dropping out and benefit the student to overcome the problems and continue their studies. Euclidean distance was used on training set to classify students which gave Welfare score=16 and Performance score=6 with nearest neighbours in the training set as K=4.

An ensemble model is proposed containing three regression models[9] based on bagging methods for prediction of school dropout in Higher Education Institutions in Brazil. Demographic factors, academic scores , and socioeconomic status of a student present in the Census and Flow Indicators of Higher Education were used for this evaluation. These features were chosen by stepwise method and Pearson correlation methods. Box-plots and Relative gain (RG) charts were used to evaluate the performance of the models where PM1(Bagging with linear regression) performed better than PM2(Bagging with robust regression), PM3(Bagging with ridge regression), and LM4(linear regression, lasso regression, bagging (using decision tree), boosting, random forest, vector regression support and k-nearest neighbours). Using the Correlation method, PM1 got smaller errors than PM2, PM3and LM4 since the value of p-value was $2.47x10^{-7}$ and others had p-value near to $7.916x10^{-2}$.

A feature-based selection approach [10] was proposed that chooses the most relevant features for the predictive task for finding student ‘s dropout prediction by increasing the model’s efficiency. This was achieved by combining SMOTE with Random Forest/RF, Convolutional Neural Networks/CNN, and Deep neural network/DNN and Artificial Neural Networks/ANN models. The proposed method gave Random Forest/RF as accuracy of 93%, Convolutional Neural Networks/CNN of 88%, Artificial Neural Networks/ANN of 89% and Deep neural network/DNN of 99%. DNN was among the best-performing machine-learning models.

3. Proposed Approach

A machine learning based predictive model using ensemble methods is developed for classifying students who are dropping out from the course before its completion. The dropout rate refers to the percentage of students who leave an educational program before its completion. Various reasons contribute to the drop out percentage, such as academic difficulties, personal or family reasons, financial problems and lack of interest in studies. Dropout rates vary by region, socio-economic status, gender, and level of education in India, also students from lower socio-economic backgrounds are more likely to drop out due to financial constraints or the need to work to support their families.

A dataset containing information of students studying in universities can be further used to estimate overall student performance at the end of each semester by assessing curricular unit credited/enrolled/evaluated/approved as well as their respective grades. Features like unemployment rate, inflation rate and GDP of the region helps to identify which of the students are not continuing the course. So, to build the ensemble models that includes Support Vector Machine/SVM, K-Nearest Neighbour/KNN, Random Forest/RF, Decision Tree/DT, XGBoost, Naïve Bayes/NB, Logistic Regression/LR algorithms, a data set from Kaggle repository is considered that contains information such as, students enrolled in various undergraduate degrees in academic institution. Data pre-processing is applied on the dataset to remove any missing or null values. Moreso, normalization and scaling of feature variables are also implemented. The dataset is split into train and test set after identifying the correlated features present in the dataset. In addition, hyperparameter adjustment is done for optimized classification accuracy.

The following are the steps of developing the machine learning-based predictive model.

- Data collection
- Data pre-processing
- Feature selection
- Methodology

A block diagram of the predictive model is presented in Figure 10.

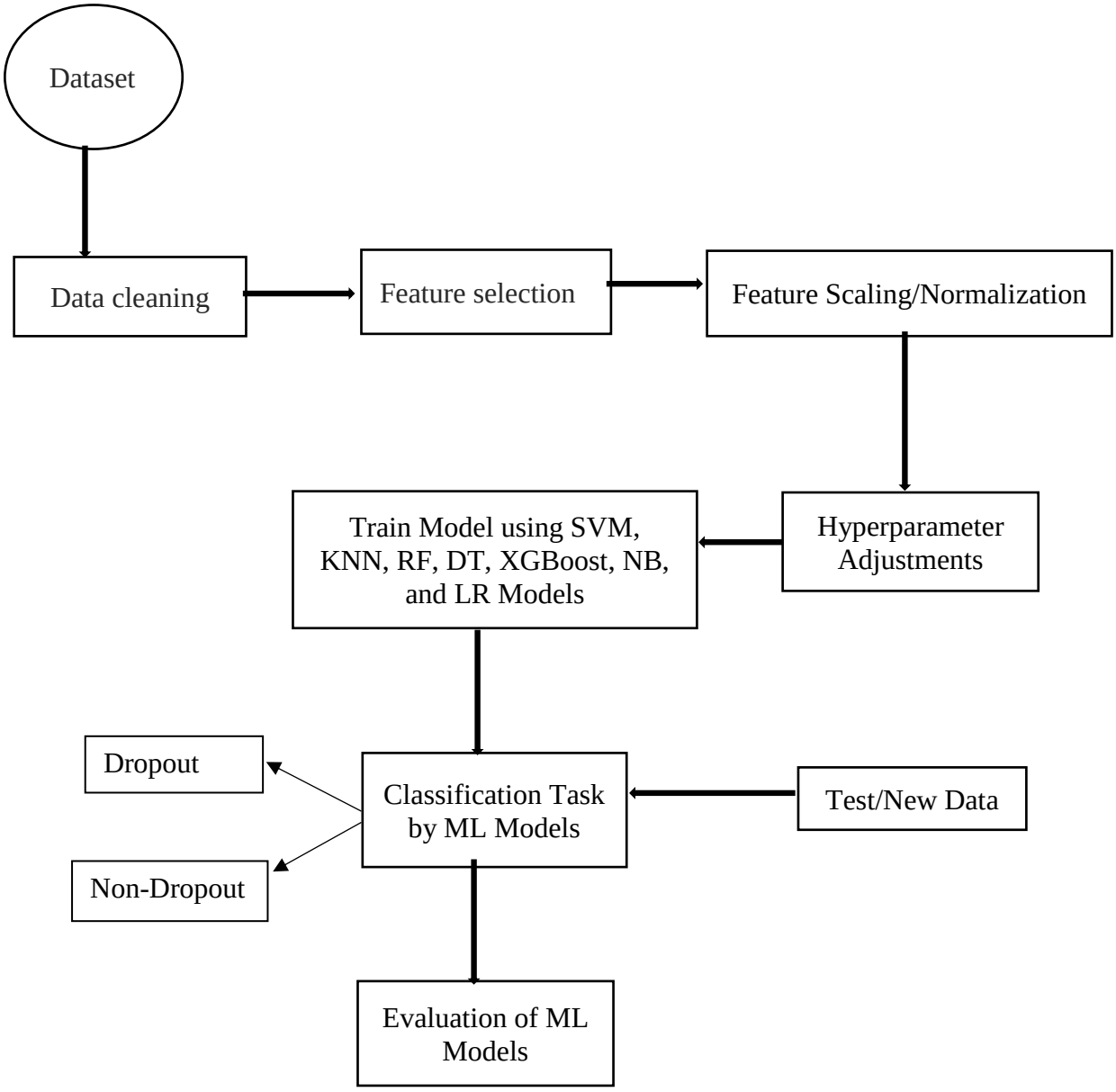

Figure 10: Steps for developing a predictive model for dropout and non-dropout classification

3.1 Dataset Description

The dataset contains 4424 students' records and 37 features collected from Kaggle repository. The predictor variables are application mode, marital status, course chosen etc. The dataset can be further used to estimate overall student performance at the end of each semester by assessing curricular unit credited/enrolled/evaluated/approved as well as their respective grades. Features like unemployment rate, inflation rate and GDP of the region also helps to identify which of the students are leaving the course. The features of the dataset are as follows.

Marital status, Application order, Daytime/evening attendance, Previous qualification (grade), Mother's qualification, Mother's occupation, Admission grade, Educational special needs, Tuition fees up to date, Scholarship holder, International, Application mode, Course, Previous qualification, Nationality, Father's qualification, Father's occupation, Displaced,

Debtor, Gender, Age at enrolment , Curricular units 1st sem (credited), Curricular units 1st sem (enrolled), Curricular units 1st sem (evaluations), Curricular units 1st sem (approved), Curricular units 1st sem (grade), Curricular units 1st sem (without evaluations), Curricular units 2nd sem (credited), Curricular units 2nd sem (enrolled), Curricular units 2nd sem (evaluations), Curricular units 2nd sem (approved), Curricular units 2nd sem (grade), Curricular units 2nd sem (without evaluations), Unemployment rate, Inflation rate, GDP, Target. The model predicts the target outcome variable which is categorical and the predictor/independent variables are numerical.

3.2 Data Pre-Processing

Data cleaning

Dataset is pre-processed to identify the missing values and outliers to ensure the data is of high quality and suitable for analysis and modelling. Data cleaning is followed by feature scaling to maintain a similar scale or range by preventing certain features from dominating others during model training.

Feature selection

The objective of the feature selection process is to select an appropriate subset of features that efficiently describes the input data by reducing the dimensionality of feature space, and removing redundant and irrelevant data. The choice of feature selection method depends on the specific problem, the characteristics of the dataset, and the machine learning algorithms used. Using statistical based-feature selection the relevant features are sorted out.

i. ANOVA test is performed to find out the correlated features having p-value less than 0.01 as shown in Figure 11.

ii. p-value is defined as the probability value of null hypothesis (A hypothesis that treats everything as equal and similar) being true which gives a measure of how much evidence is present to reject the null hypothesis. The smaller the p value, the higher the evidence against null hypothesis.

- p-value <=0.01-very strong evidence
- 0.01<= p-value <=0.05-strong evidence
- 0.05<= p-value <=0.1-mild evidence
- p-value >=0.1-no evidence

Features with p-value less than 0.01 are selected for prediction and classification. The following features are selected after applying the statistical measure (ANOVA test) as shown in Figure 12.

Marital status, Application mode, Application order, Daytime/evening attendance, Previous qualification, Previous qualification (grade), Mother's qualification, Father's qualification, Mother's occupation, Admission grade, Displaced, Debtor, Tuition fees up to date, Gender, Scholarship holder, Age at enrolment, Curricular units 1st sem (credited), Curricular units 1st sem (enrolled), Curricular units 1st sem (evaluations), Curricular units 1st sem (approved), Curricular units 1st sem (grade), Curricular units 1st sem (without evaluations), Curricular units 2nd sem (credited), Curricular units 2nd sem (enrolled), Curricular units 2nd sem (evaluations), Curricular units 2nd sem (approved), Curricular units 2nd sem (grade), Curricular units 2nd sem (without evaluations), Unemployment rate, GDP.

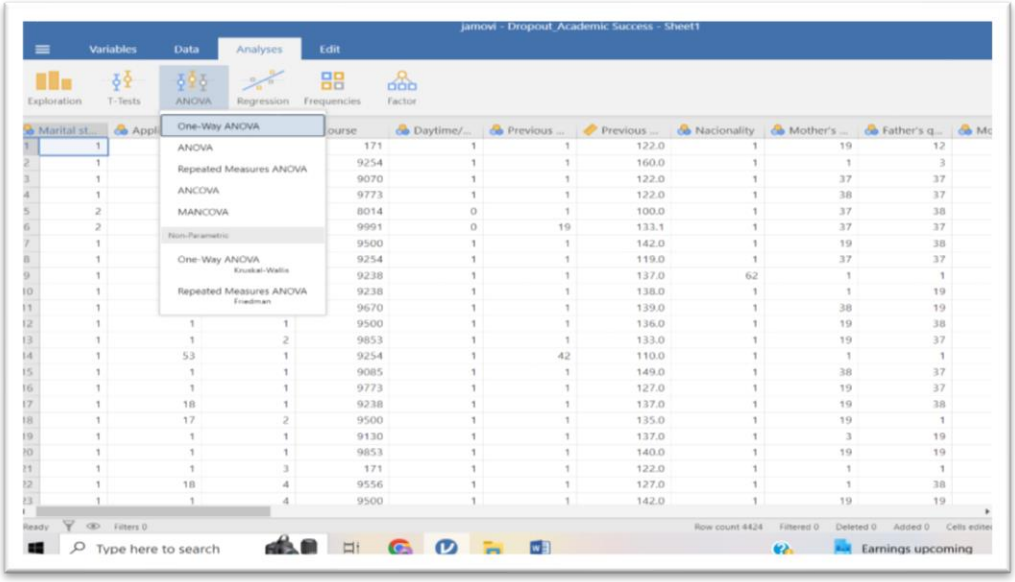

Figure 11: Selection of correlated features using ANOVA test

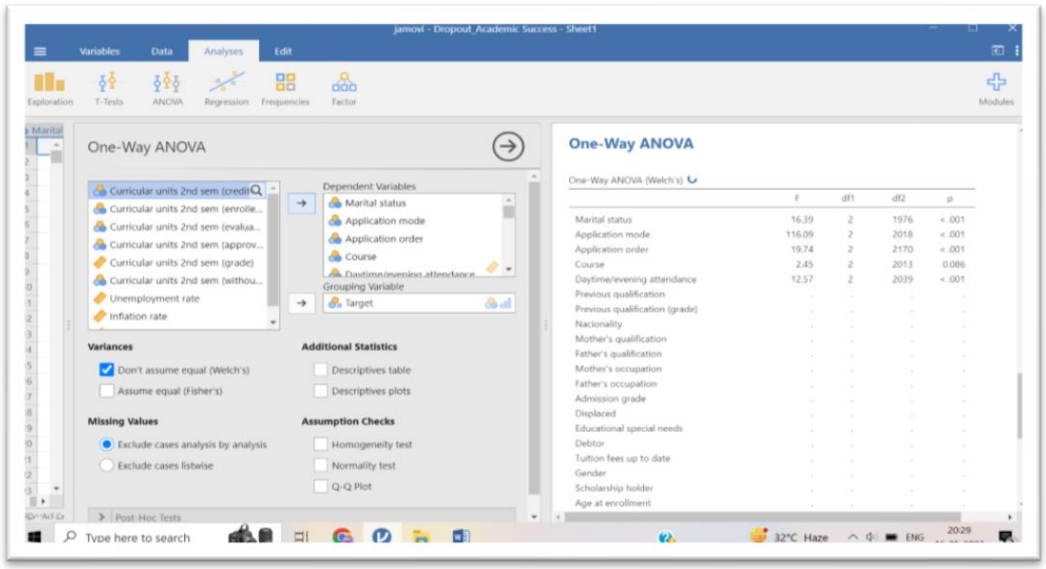

Figure 12: Correlated features for the binary classification of the predictive model

3.3 Methodology

The dataset is further split into train and test set in the ratio of 80% and 20%, after identifying correlated features. Scaling operation is performed to bring all the features of a dataset to a similar scale or range. Several machine learning models such as SVM, KNN, RF, DT, XGBoost, NB, LR classifiers are considered for predicting student's dropout. During model training, hyperparameters are adjusted for optimized classification accuracy.

The parameters in hyperparameter tuning for each model are depicted in Table 1, 2, 3, 4, 5, 6 and 7.

Table 1: Parameters for measuring metrics in K-nearest neighbour algorithm

n_neighbors	Specify the number of neighbours to consider when making predictions.
Weights	Determine the weight assigned to each neighbour in the prediction process.
metric	Define the distance metric used to measure the similarity between instances.
p	Used when the 'minkowski' metric is selected. The values provided are 1 and 2, corresponding to Manhattan distance and Euclidean distance and the respective formulae are given in Figures 13, 14 and 15.

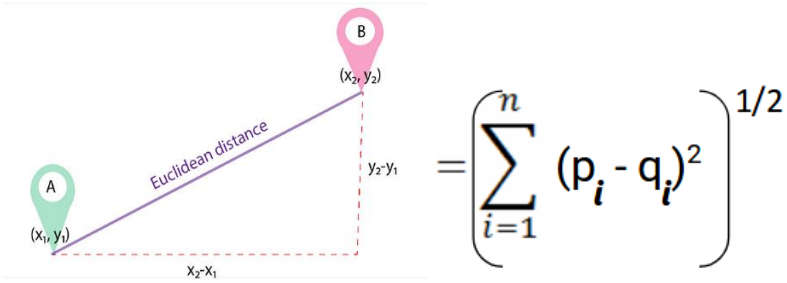

Figure 13: Euclidean distance

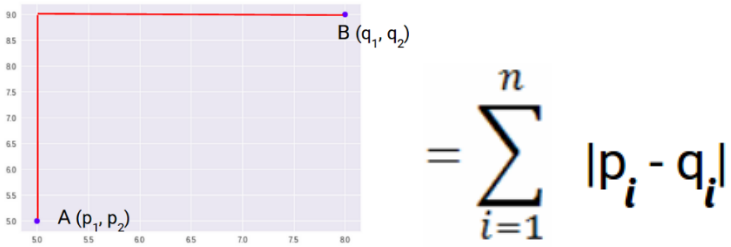

Figure 14: Manhattan distance

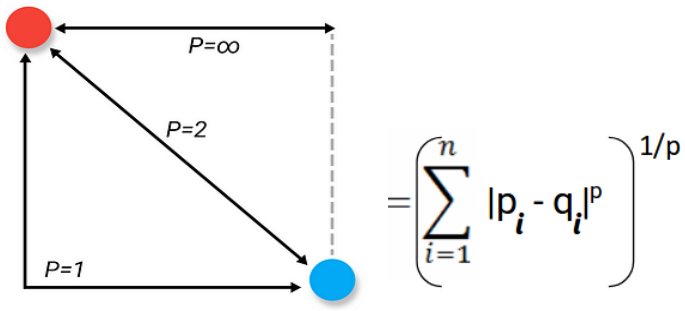

Figure 15: Minkowski distance

n = no of dimensions; pi, qi = data points; p= order parameter

Table 2: Parameters for Logistic Regression optimization

penalty	Prevents overfitting and improve generalization.
C	Influences the balance between model-fit and model-overfit as shown in Figure 16.
Solver	Solves the optimization problem.

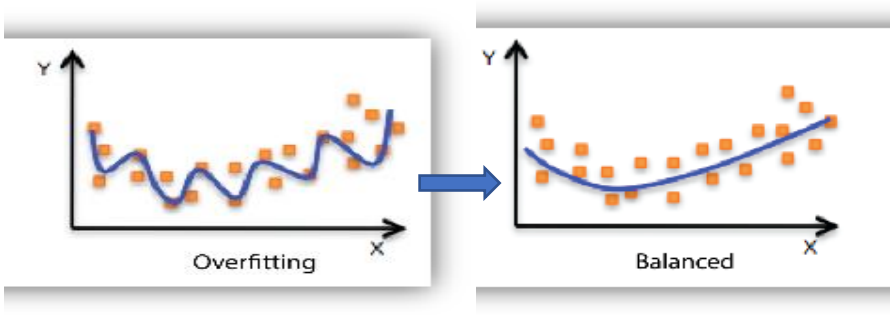

Figure 16: Overfit model leading to balanced model

Table 3: Parameters to reduce overfitting in Naïve Bayes algorithm

Var_smoothing	Prevent overfitting and improve generalization.

Table 4: Different criteria for selecting the best tree in Random Forest

'n_estimators'	Represent the number of decision trees to be included in the ensemble as shown in Figure 17.
'max_depth'	Maximum number of levels in each decision tree.
'min_samples_split'	Determine the minimum number of samples to split an internal node.
'min_samples_leaf'	Set the minimum number of samples to be present in the leaf node.

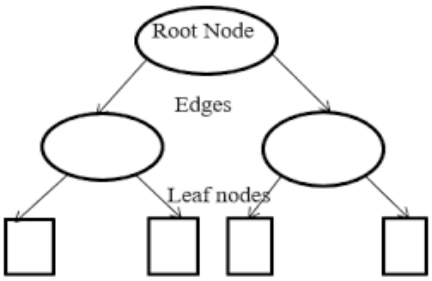

Figure 17: Tree

Table 5: Splitting criteria for Decision Tree algorithm

Criterion	Measure the quality of a split at each node of the tree.
'max_depth'	Maximum number of levels in each decision tree.
'min_samples_split'	Determine the minimum number of samples required to split an internal node.

'min_samples_leaf'	Set the minimum number of samples required to be at a leaf node.

Table 6: Criteria for hyperplane maximization in Support Vector Machine

C	Maximize the margin and minimize the classification error as depicted in Figure 18.
Gamma	Affect the shape and flexibility of the decision boundary.
Kernel	Kernel function is the distance between data points.

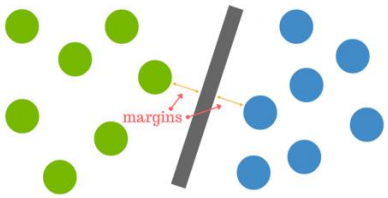

Figure 18: Margin between data points

Table 7: Different criteria to select the best tree in XGBoost

'n_estimators'	Refer to the number of decision trees to be built in the ensemble.
'learning_rate'	Determine the step size at each boosting iteration and controls the contribution of each tree in the ensemble.
'booster	Determine the type of model to be used.
'gamma'	Control the minimum loss reduction required to make a further partition on a leaf node.
'reg_alpha	Reduce overfitting through sparsity in the feature weights.
'reg_lambda	Prevent overfitting based on the sum of squared weights.
'base_score'	Act as a starting point for the boosting process.

Randomized search with cross-validation is performed to find the best combination of hyperparameters for the chosen classifier. Cross-validation is a technique where instead of splitting the dataset into two parts, one to train and another to test, the dataset is split into multiple portions, train on some of these and use the rest to test on and again use a different portion to train and test our model on which ensures that the model is training and testing on new data at every new step.

The dataset is trained using the fit() method. The fit() method typically takes the input features and the corresponding target values to fit the model to the training data by allowing it to learn patterns and make predictions. The model is then evaluated for prediction efficiency and minimum classification error.

4. Experimentations and Results

The dissertation consists of an ensemble approach of Machine Learning methods to assess the dropping out of students from universities. A sample dataset is taken from Kaggle repository. Performing ANOVA test in Jamovi software, relevant features are selected from the dataset followed by scaling operation and data pre-processing. Several machine learning classifiers such as Support Vector Machine/SVM, K-Nearest Neighbour/KNN, Random Forest/RF, Decision Tree/DT, XGBoost, Naïve Bayes/NB, Logistic Regression/LR models are built after splitting the dataset into train and test set. In addition, the hyperparameters adjustments are made for optimized classification accuracy. The metrics to assess performance of classification models are accuracy, precision, and recall. The evaluation results are depicted in Table 8, 9, 10, 11, 12, 13 and 14.

Table 8: Classification report of K-nearest neighbour model

Classification	Precision	Recall	F1-Score	Support
Dropout	0.87	0.68	0.76	316
Enrolled	0.42	0.21	0.28	151
Graduate	0.72	0.96	0.82	418

The accuracy of K-nearest neighbour model is 73.43% after randomized search tuning. Table 8 shows that the dropout classification has a precision score of 87%, which is substantially higher than the other two classification models. Recall value of classifying the graduate students is 96%, which is much higher in compare to the other two classification models. Graduate classification has the highest f1-score is 82%, generated by combining precision and recall.

Table 9: Classification report of Logistic Regression model

Classification	Precision	Recall	F1-Score	Support
Dropout	0.85	0.73	0.79	316
Enrolled	0.44	0.31	0.36	151
Graduate	0.76	0.92	0.83	418

The accuracy of Logistic Regression model is 75.93% after randomized search tuning. From Table 9 it is observed that the precision value of dropout classification is 85%, which is substantially higher than the values for the other two classifications. Recall value of graduate classification is 92%, which is substantially higher than the other two classification models.

Table 10: Classification report of Naïve Bayes model

Classification	Precision	Recall	F1-Score	Support
Dropout	0.82	0.69	0.75	316
Enrolled	0.41	0.33	0.37	151
Graduate	0.73	0.87	0.80	418

The accuracy of Naïve Bayes is 71.19% after randomized search tuning. From Table 10 it is observed that the precision value of dropout classification is 82%, which is substantially higher than the values for the other two classifications. Recall value of graduate classification is 87%, which is substantially higher than the values of the other two classifications. F1-score of graduate classification is 80%.

Table 11: Classification report of Support Vector Machine model

Classification	Precision	Recall	F1-Score	Support
Dropout	0.85	0.73	0.79	316
Enrolled	0.44	0.31	0.36	151
Graduate	0.76	0.92	0.83	418

The accuracy of Support Vector Machine model is 75.03% after randomized search tuning. From Table 11 it is observed, that the precision value of dropout classification is 85%, which is substantially higher than the values for the other two classifications. Recall value of graduate classification is 92%, which is substantially higher than the values of the other two classifications. F1-score of classifying the graduate students is 83% which is on the higher range.

Table 12: Classification report of Random Forest model

Classification	Precision	Recall	F1-Score	Support
Dropout	0.84	0.74	0.79	316
Enrolled	0.51	0.27	0.35	151
Graduate	0.76	0.95	0.84	418

The accuracy of Random Forest model is 76.93% after randomized search tuning. From Table 12, it is observed that the precision value of dropout classification is 84%, which is substantially higher than the values for the other two classifications. Recall value of graduate classification is 95%, which is substantially higher than the values of the other two classifications. Graduate classification's f1-score has obtained the highest value of 84%.

Table 13: Classification report of Decision Tree model

Classification	Precision	Recall	F1-Score	Support
Dropout	0.83	0.69	0.75	316
Enrolled	0.39	0.19	0.39	151
Graduate	0.79	0.89	0.84	418

The accuracy of Decision Tree model is 73.33% after randomized search tuning. From Table 13, it is observed that the precision value of dropout classification is 83%, which is substantially higher than the values than the other two classifications. Recall value of graduate classification is 89%, which is substantially higher than the other two classification models. F1-score of graduate classification has achieved the highest value of 84%.

Table 14: Classification report of XGBoost algorithm

Classification	precision	recall	f1-score	support
Dropout	0.85	0.75	0.80	316
Enrolled	0.50	0.37	0.43	151
Graduate	0.78	0.92	0.85	418

The accuracy of XGboost algorithm is 78.84% after randomized search tuning. From Table 14 it is observed that the precision value of dropout classification is 85%, which is substantially higher than the values of other two classifications. Recall value of graduate classification is 92% which is substantially higher than the other two classification models. F1-score of graduate classification has obtained the highest value of 85%.

5. Comparative Analysis

A comparative evaluation is presented in Table15 considering different ensemble models i.e., LR, KNN, SVM, NB, RF, DT and XGBoost before and after applying hyperparameter adjustments. Accuracy, precision, and recall measure the performance of classification models. Other than accuracy, precision and recall are also considered to assess the model-performance of respective models before applying hyperparameter tuning. After feature-based selection and hyperparameter tuning, the performance of various model is compared considering the base scores.

Table 15: Score comparison of various ML models with hyperparameter adjustments

Models	Accuracy (%)		Precision (%)		Recall (%)	
	Base score	New score	Base score	New score	Base score	New score
KNN	61	74	73	87	51	68
LR	72	76	78	85	71	73
NB	71	71	81	82	68	69
SVM	64	75	84	85	52	73
RF	76	77	84	85	76	75
DT	67	74	74	83	66	69
XGBoost	77	79	82	85	74	75

It is observed that all the ensemble models had an increase in accuracy, precision and recall after hyperparameter adjustment is made. It is observed that XGBoost, RF and LR have good accuracy scores i.e., 79%, 77%, and 76% respectively in comparison with other models. KNN, SVM, and XGBoost has good precision scores with 87% and 85% for dropout classification. The recall for RF and XGBoost is 75% for dropout classification. It is observed that overall performance of XGBoost model is more efficient in comparison to other models with precision of 85%, recall of 75% and accuracy 79%. So XGBoost is the most efficient algorithm among all the other models described and experimented.

6. Conclusion and Future Work

Ensemble classification methods like LR, KNN, SVM, NB, RF, DT and XGBoost are applied with hyperparameters adjustments to improve the optimized accuracy of the respective models. ANOVA test is used to choose relevant features from the dataset. This approach helped in reducing errors as well as identification of students who are dropping out without completing the courses in the University. The metrics to assess the performance of classification models are accuracy, precision, and recall and experimental analysis shows that the XGBoost algorithm is the most efficient among all the models.

This work can be further experimented and compared with neural networks, and unsupervised models to identify the efficacy of the model-prediction.

Reference

1. B. Sravani and M.Madhu Bala, "Prediction of Student Performance Using Linear Regression" , 2020 International Conference for Emerging Technology (INCET) Belgaum, India, 2020.
2. H. Dasi and S. Kanakala, "Student Dropout Prediction Using Machine Learning Techniques", International Journal of Intelligent Systems and Applications in Engineering, vol.10, no.4, pp. 408–414, 2022.
3. J. S. Gil, A.J P. Delima and R.N. Vilchez, "Predicting Students' Dropout Indicators in Public School using Data Mining Approaches", International Journal of Advanced Trends in Computer Science and Engineering, vol.9, no.1, pp.774-78, 2020.
4. M. Naseem, K. Chaudhary, B. Sharma and A.G Lal, "Using Ensemble Decision Tree Model to Predict Student Dropout in Computing Science", 2019 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), 2019.
5. B. Ujkani, D. Minkovska and L. Stoyanova, "Application of Logistic Regression Technique for Predicting Student Dropout", 2022 XXXI International Scientific Conference Electronics (ET), 2019.
6. M. Nasiri, B. Minaei and F. Vafaei , "Predicting GPA and Academic Dismissal in LMS Using Educational Data Mining: A Case Mining", 6th National and 3rd International conference of e-Learning and e-Teaching(ICELET2012), 2012.
7. Dr. R. Raju, Sri M.Vinayagar, Mrs. N. Kalaiselvi, Sri M.Vinayagar, A. Sulthana M, Sri M.Vinayagar, Divya I, Sri M.Madagadipet and Selvarani, "EDUCATIONAL DATA MINING: A COMPREHENSIVE STUDY", 2020 International Conference on System, Computation, Automation and Networking (ICSCAN), 2020.
8. M. Mardolkar and N. Kumaran, "Forecasting and Avoiding Student Dropout Using the K-Nearest Neighbor Approach", SN COMPUT. SCI. 1, 96 (2020), 2020.
9. Paulo M. da Silva, Marília N. C. A. Lima, Wedson L. Soares, Iago R. R. Silva, Roberta A. de A. Fagundes and Fernando F. de Souza, "Ensemble Regression Models Applied to Dropout in Higher Education", 2019 8th Brazilian Conference on Intelligent Systems (BRACIS), 2019.
10. E. Mulyani, I. Hidayah and S. Fauziati , "Dropout Prediction Optimization through SMOTE and Ensemble Learning", 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), 2019.

Appendix

I/P→

```
#importing all libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.metrics import
classification_report,confusion_matrix, accuracy_score,
f1_score, recall_score, precision_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import RandomizedSearchCV
```

```
#loading the dataset
df = pd.read_csv('/content/Dropout_Academic Success -
Sheet1.csv')
```

```
#check missing values
df.isnull().sum()
```

O/P→

```
Marital status                                  0
Application mode                                0
Application order                               0
Course                                          0
Daytime/evening attendance                      0
Previous qualification                          0
Previous qualification (grade)                  0
Nationality                                     0
Mother's qualification                          0
Father's qualification                          0
Mother's occupation                             0
Father's occupation                             0
Admission grade                                 0
Displaced                                       0
Educational special needs                       0
Debtor                                          0
Tuition fees up to date                         0
Gender                                          0
Scholarship holder                              0
Age at enrolment                                0
International                                   0
```

```
Curricular units 1st sem (credited)                0
Curricular units 1st sem (enrolled)                0
Curricular units 1st sem (evaluations)             0
Curricular units 1st sem (approved)                0
Curricular units 1st sem (grade)                   0
Curricular units 1st sem (without evaluations)     0
Curricular units 2nd sem (credited)                0
Curricular units 2nd sem (enrolled)                0
Curricular units 2nd sem (evaluations)             0
Curricular units 2nd sem (approved)                0
Curricular units 2nd sem (grade)                   0
Curricular units 2nd sem (without evaluations)     0
Unemployment rate                                  0
Inflation rate                                     0
GDP                                                0
Target                                             0
dtype: int64
```

I/P→

```
#displays no of attributes present in the dataset
df.columns
```

O/P→

```
Index(['Marital status', 'Application mode', 'Application order', 'Course', 'Daytime/evening attendance', 'Previous qualification', 'Previous qualification (grade)', 'Nationality', 'Mother's qualification', 'Father's qualification', 'Mother's occupation', 'Father's occupation', 'Admission grade', 'Displaced', 'Educational special needs', 'Debtor', 'Tuition fees up to date', 'Gender', 'Scholarship holder', 'Age at enrolment', 'International', 'Curricular units 1st sem (credited)', 'Curricular units 1st sem (enrolled)', 'Curricular units 1st sem (evaluations)', 'Curricular units 1st sem (approved)', 'Curricular units 1st sem (grade)', 'Curricular units 1st sem (without evaluations)', 'Curricular units 2nd sem (credited)', 'Curricular units 2nd sem (enrolled)', 'Curricular units 2nd sem (evaluations)', 'Curricular units 2nd sem (approved)', 'Curricular units 2nd sem (grade)', 'Curricular units 2nd sem (without evaluations)', 'Unemployment rate', 'Inflation rate', 'GDP', 'Target'], dtype='object')
```

I/P→

```
#  returns a specified number of rows from the top
df.head()
```

O/P→

	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Previous qualification (grade)	Nacionality	Mother's qualification	Father's qualification	...	Curricular units 2nd sem (credited)
0	1	17	5	171	1	1	122.0	1	19	12	...	0
1	1	15	1	9254	1	1	160.0	1	1	3	...	0
2	1	1	5	9070	1	1	122.0	1	37	37	...	0
3	1	17	2	9773	1	1	122.0	1	38	37	...	0
4	2	39	1	8014	0	1	100.0	1	37	38	...	0

5 rows × 37 columns

I/P→

```
#information regarding  description and types of attributes
df.info()
```

O/P→

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4424 entries, 0 to 4423
Data columns (total 37 columns):
 #   Column                                          Non-Null
Count  Dtype
---  ------                                          ---------
-----  -----
 0   Marital status                                  4424 non-
null   int64
 1   Application mode                                4424 non-
null   int64
 2   Application order                               4424 non-
null   int64
 3   Course                                          4424 non-
null   int64
 4   Daytime/evening attendance                      4424 non-
null   int64
 5   Previous qualification                          4424 non-
null   int64
 6   Previous qualification (grade)                  4424 non-
null   float64
 7   Nationality                                     4424 non-
null   int64
 8   Mother's qualification                          4424 non-
null   int64
```

```
 9   Father's qualification                          4424 non-
null   int64
 10  Mother's occupation                             4424 non-
null   int64
 11  Father's occupation                             4424 non-
null   int64
 12  Admission grade                                 4424 non-
null   float64
 13  Displaced                                       4424 non-
null   int64
 14  Educational special needs                       4424 non-
null   int64
 15  Debtor                                          4424 non-
null   int64
 16  Tuition fees up to date                         4424 non-
null   int64
 17  Gender                                          4424 non-
null   int64
 18  Scholarship holder                              4424 non-
null   int64
 19  Age at enrolment                               4424 non-
null   int64
 20  International                                   4424 non-
null   int64
 21  Curricular units 1st sem (credited)             4424 non-
null   int64
 22  Curricular units 1st sem (enrolled)             4424 non-
null   int64
 23  Curricular units 1st sem (evaluations)          4424 non-
null   int64
 24  Curricular units 1st sem (approved)             4424 non-
null   int64
 25  Curricular units 1st sem (grade)                4424 non-
null   float64
 26  Curricular units 1st sem (without evaluations)  4424 non-
null   int64
 27  Curricular units 2nd sem (credited)             4424 non-
null   int64
 28  Curricular units 2nd sem (enrolled)             4424 non-
null   int64
 29  Curricular units 2nd sem (evaluations)          4424 non-
null   int64
 30  Curricular units 2nd sem (approved)             4424 non-
null   int64
```

```
 31  Curricular units 2nd sem (grade)                4424 non-
null   float64
 32  Curricular units 2nd sem (without evaluations)  4424 non-
null   int64
 33  Unemployment rate                               4424 non-
null   float64
 34  Inflation rate                                  4424 non-
null   float64
 35  GDP                                             4424 non-
null   float64
 36  Target                                          4424 non-
null   object
dtypes: float64(7), int64(29), object(1)
memory usage: 1.2+ MB
```

**Then applying ANOVA test the relevant features are selected which are further used for train and test set for the models.

I/P→

```
X=df[['Marital status', 'Application mode', 'Application order', 'Course', 'Daytime/evening attendance', 'Previous qualification', 'Previous qualification (grade)', 'Nationality', "Mother's qualification", "Father's qualification","Mother's occupation", "Father's occupation", 'Admission grade','Displaced', 'Educational special needs', 'Debtor','Tuition fees up to date', 'Gender', 'Scholarship holder','Age at enrolment', 'International', 'Curricular units 1st sem (credited)','Curricular units 1st sem (enrolled)', 'Curricular units 1st sem (evaluations)','Curricular units 1st sem (approved)','Curricular units 1st sem (grade)','Curricular units 1st sem (without evaluations)','Curricular units 2nd sem (credited)', 'Curricular units 2nd sem (enrolled)','Curricular units 2nd sem (evaluations)','Curricular units 2nd sem (approved)','Curricular units 2nd sem (grade)','Curricular units 2nd sem (without evaluations)', 'Unemployment rate', 'Inflation rate', 'GDP']]

X_selected = X[['Marital status', 'Application mode', 'Application order', 'Daytime/evening attendance', 'Previous qualification','Previous qualification (grade)', "Mother's qualification", "Father's qualification", "Mother's occupation", 'Admission grade', 'Displaced', 'Debtor', 'Tuition
```

```
fees up to date', 'Gender', 'Scholarship holder','Age at enrolment', 'Curricular units 1st sem (credited)','Curricular units 1st sem (enrolled)', 'Curricular units 1st sem (evaluations)', 'Curricular units 1st sem (approved)','Curricular units 1st sem (grade)','Curricular units 1st sem (without evaluations)','Curricular units 2nd sem (credited)', 'Curricular units 2nd sem (enrolled)','Curricular units 2nd sem (evaluations)','Curricular units 2nd sem (approved)', 'Curricular units 2nd sem (grade)','Curricular units 2nd sem (without evaluations)', 'Unemployment rate', 'GDP']]
```

#train set and test set is built with the selected features

```
X_train, X_test, y_train, y_test = train_test_split(X_selected,df['Target'], test_size=0.2, random_state=42)
```

#scaling operation is performed

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

#importing different ML models

```
from sklearn.ensemble import RandomForestClassifier
from sklearn import svm
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
```

#building classifiers

```
dtc = DecisionTreeClassifier()
lr = LogisticRegression()
rfc = RandomForestClassifier()
nb_model = GaussianNB()
knn = KNeighborsClassifier()
svc=   svm.SVC()
```

**After building the classifiers hyperparameter adjustment is done.

I/P→

#Applying hyperparameter adjustment to NB model

```
param_grid2 = {
     'var_smoothing': [0, 1, 1e-8, 1e-7, 1e-6,1e-9]
```

```
}
random_search2 = RandomizedSearchCV(nb_model,
param_distributions=param_grid2, cv=5, n_iter=10)
random_search2.fit(X_train, y_train)
best_model2 = random_search2.best_estimator_
best_params2 = random_search2.best_params_
print("Best Hyperparameters for NB:", best_params2)
y_pred2 = best_model2.predict(X_test)
accuracy2 = accuracy_score(y_test, y_pred2)
print("Classification report of
NB\n",classification_report(y_test,y_pred2))
print("NB--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy2 * 100))
```

O/P→

```
Best Hyperparameters for NB: {'var_smoothing': 1e-08}
Classification report of NB
              precision    recall  f1-score   support

     Dropout       0.82      0.69      0.75       316
    Enrolled       0.41      0.33      0.37       151
    Graduate       0.73      0.87      0.80       418

    accuracy                           0.71       885
   macro avg       0.65      0.63      0.63       885
weighted avg       0.71      0.71      0.70       885

NB--Accuracy is : 71.19% after randomized search tuning
```

I/P→

#Applying hyperparameter adjustment to kNN model

```
param_grid =  {'n_neighbors': range(1, 30),
               'weights': ['uniform', 'distance'],
               'metric': [
'minkowski','euclidean','manhattan'],
                'p': [1, 2]}
random_search = RandomizedSearchCV(knn,
param_distributions=param_grid, n_iter=10, cv=5)
random_search.fit(X_train, y_train)
print("Best Parameters of knn: ", random_search.best_params_)
best_model = random_search.best_estimator_
y_pred = best_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
```

```
print("Classification report of
KNN\n",classification_report(y_test,y_pred))
print("KNN--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy * 100))
```

O/P→

```
Best Parameters of knn:  {'weights': 'distance', 'p': 1,
'n_neighbors': 17, 'metric': 'manhattan'}
Classification report of KNN
              precision    recall  f1-score   support

     Dropout       0.87      0.64      0.74       316
    Enrolled       0.48      0.22      0.30       151
    Graduate       0.69      0.98      0.81       418

    accuracy                           0.73       885
   macro avg       0.69      0.61      0.62       885
weighted avg       0.73      0.73      0.70       885

KNN--Accuracy is : 73.33% after randomized search tuning
```

I/P→

#Applying hyperparameter adjustment to SVM model

```
svm_grid = {
   'C': [0.1, 1, 10, 1000], 'gamma': [0.1, 1, 10, 1000],
'kernel': ['linear', 'rbf']
}
random_search6 = RandomizedSearchCV(svc, svm_grid, cv=5)
random_search6.fit(X_train, y_train)
best_params6 = random_search6.best_params_
print("Best Hyperparameters of SVM:", best_params6)
best_model6 = random_search6.best_estimator_
y_pred6 = best_model6.predict(X_test)
accuracy6 = accuracy_score(y_test, y_pred6)
print("Classification report of
SVM\n",classification_report(y_test,y_pred6))
print("SVM--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy6 * 100))
```

O/P→

```
Best Hyperparameters of SVM: {'kernel': 'linear', 'gamma':
1000, 'C': 10}
Classification report of SVM
              precision    recall  f1-score   support
```

```
     Dropout       0.85      0.73      0.79       316
    Enrolled       0.44      0.31      0.36       151
    Graduate       0.76      0.92      0.83       418

    accuracy                           0.75       885
   macro avg       0.68      0.66      0.66       885
weighted avg       0.74      0.75      0.74       885

SVM--Accuracy is : 75.03% after randomized search tuning
```

I/P→

```
#Applying hyperparameter adjustment to  LR model
param_grid1 = {
    'penalty': ['l1', 'l2'],
    'C': np.logspace(-4, 4, 20),
    'solver': ['liblinear']
}
random_search1 = RandomizedSearchCV(lr,
param_distributions=param_grid1, n_iter=10, cv=5,
random_state=42)
random_search1.fit(X_train, y_train)
print("Best Parameters of LR: ", random_search1.best_params_)
best_model1 = random_search1.best_estimator_
y_pred1 = best_model1.predict(X_test)
accuracy1 = accuracy_score(y_test, y_pred1)
print("Classification report of
LR\n",classification_report(y_test,y_pred6))
print("LR--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy1 * 100))
```

O/P→

```
Best Parameters of LR:  {'solver': 'liblinear', 'penalty':
'l2', 'C': 0.615848211066026}
Classification report of LR
              precision    recall  f1-score   support

     Dropout       0.85      0.73      0.79       316
    Enrolled       0.44      0.31      0.36       151
    Graduate       0.76      0.92      0.83       418

    accuracy                           0.76       885
   macro avg       0.68      0.66      0.66       885
weighted avg       0.74      0.75      0.74       885
```

```
LR--Accuracy is : 75.93% after randomized search tuning
```

I/P→

```
#Applying hyperparameter adjustment to DT model
dtc_params = {
      'criterion': ['gini', 'entropy'],
    'max_depth': [None,2,  10, 20],
    'min_samples_split': [12,34,67],
    'min_samples_leaf': [1,5,9]
    }
random_search5 = RandomizedSearchCV(estimator=dtc,
param_distributions=dtc_params, cv=5)
random_search5.fit(X_train, y_train)
best_params5 = random_search5.best_params_
print("Best Hyperparameters of DT:", best_params5)
best_model5 = random_search5.best_estimator_
y_pred5 = best_model5.predict(X_test)
accuracy5 = accuracy_score(y_test, y_pred5)
print("Classification report of
DT\n",classification_report(y_test,y_pred5))
print("DT--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy5 * 100))
```

O/P→

```
Best Hyperparameters of DT: {'min_samples_split': 67,
'min_samples_leaf': 5, 'max_depth': 20, 'criterion': 'gini'}
Classification report of DT
               precision    recall  f1-score   support

     Dropout       0.83      0.69      0.75       316
    Enrolled       0.38      0.36      0.37       151
    Graduate       0.78      0.88      0.83       418

    accuracy                           0.73       885
   macro avg       0.66      0.64      0.65       885
weighted avg       0.72      0.73      0.72       885

DT--Accuracy is : 72.54% after randomized search tuning
```

I/P→

```
#Applying hyperparameter adjustment to RF model
rfc_params={
    'n_estimators': [100, 200, 500],
```

```
    'criterion': ['gini', 'entropy'],
    'min_samples_split': [1,2,4,5,50],
    'max_depth': [ 5, 10],
    'min_samples_leaf': [1,2,4,5],
    'max_leaf_nodes': [4,10,20,50,None]
}
random_search4 = RandomizedSearchCV(estimator=rfc,
param_distributions=rfc_params, cv=5)
random_search4.fit(X_train, y_train)
best_params4 = random_search4.best_params_
print("Best Hyperparameters of RF:", best_params4)
best_model4 = random_search4.best_estimator_
best_model5 = random_search5.best_estimator_
y_pred4 = best_model4.predict(X_test)
accuracy4 = accuracy_score(y_test, y_pred4)
print("Classification report of
DT\n",classification_report(y_test,y_pred4))
print("RF--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy4 * 100))
```

O/P→

```
Best Hyperparameters of RF: {'n_estimators': 100,
'min_samples_split': 5, 'min_samples_leaf': 1,
'max_leaf_nodes': None, 'max_depth': 10, 'criterion':
'entropy'}
Classification report of RF
               precision    recall  f1-score   support

     Dropout       0.84      0.75      0.79       316
    Enrolled       0.51      0.25      0.33       151
    Graduate       0.75      0.95      0.84       418

    accuracy                           0.76       885
   macro avg       0.70      0.65      0.65       885
weighted avg       0.74      0.76      0.73       885

RF--Accuracy is : 75.82% after randomized search tuning
```

I/P→

#Applying hyperparameter adjustment to XGBoost model

```
import xgboost as xgb
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
X_train, X_test, y_train, y_test =
train_test_split(X_selected,le.fit_transform(df['Target']),
test_size=0.2, random_state=42)
xgb_grid = {
    'learning_rate': np.linspace(0.01, 0.1, num=10),
    'max_depth': range(3, 10),
    'n_estimators': range(100, 1000, 100),
    'subsample': np.arange(0.5, 1.0, 0.1),
    'colsample_bytree': np.arange(0.5, 1.0, 0.1),
    'reg_alpha': range(0,10),
    'reg_lambda': range(0,10)
}
xgb_model = xgb.XGBClassifier()
random_search7 = RandomizedSearchCV(
    xgb_model,
    param_distributions=xgb_grid,
    n_iter=10,
    cv=5,
    verbose=1,
    random_state=42
)
random_search7.fit(X_train, y_train)
best_model7 = random_search7.best_estimator_
best_params7 = random_search7.best_params_
print("Best Hyperparameters:", best_params7)
best_model7 = random_search7.best_estimator_
y_pred7 = best_model7.predict(X_test)
accuracy7 = accuracy_score(y_test, y_pred7)
print("Classification report of
XGBoost\n",classification_report(y_test,y_pred7))
print("XGboost--Accuracy is : {:.2f}% after randomized search
tuning".format(accuracy7 * 100))
```

O/P→

```
Best Hyperparameters: {'subsample': 0.8999999999999999,
'reg_lambda': 10, 'reg_alpha': 10, 'n_estimators': 700,
'max_depth': 8, 'learning_rate': 0.06000000000000001,
'colsample_bytree': 0.7999999999999999}
Classification report of XGBoost
              precision    recall  f1-score   support

           0       0.85      0.75      0.80       316
           1       0.50      0.37      0.43       151
           2       0.78      0.92      0.85       418
```

```
    accuracy                           0.79       885
   macro avg       0.71      0.68      0.69       885
weighted avg       0.76      0.77      0.76       885

XGboost--Accuracy is : 78.84% after randomized search tuning
```
