

Students Dropout Prediction Using Machine Learning Algorithms

Machine Learning Project Report
May 3rd, 2022

TABLE OF CONTENT

S. No	TOPICS	Page No.
1	INTRODUCTION	03
2	DATASET	03
3	EXPLORATORY DATA ANALYSIS	04
4	DATA PREPROCESSING	07
4	FEATURE SELECTION	09
5	METHODOLOGY	10
6	MODEL EVALUATION AND OPTIMIZATION	10
7	MODEL IMPLEMENTATION	14
8	RESULT AND CONCLUSION	16

INTRODUCTION

The problem of student dropout is widespread in the educational system and has serious detrimental effects on both the lives of the individuals involved and society at large. Machine learning approaches have been utilized more and more to forecast and prevent student dropouts as well as to uncover aspects that contribute to student performance to address this problem. In this research, we seek to create a machine learning model that can precisely predict student success and dropout using a variety of pertinent factors, including demographic data, academic achievement, and socioeconomic status. Our objective is to design a tool that may help educators and decision-makers create efficient interventions to increase student success and retention.

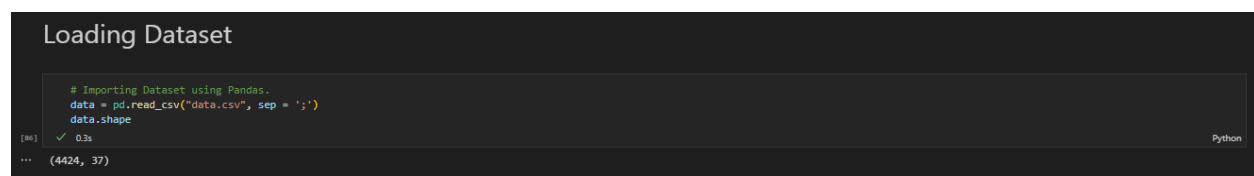
To develop an accurate model for predicting student dropout and success, we will be implementing various machine learning techniques. These will include both traditional models, such as logistic regression and decision trees, as well as more advanced methods such as support vector machines and random forest. We will also be performing feature selection to identify the most relevant variables for predicting student outcomes. This will involve using techniques such as K-fold cross validation and scaling techniques such as standardization and normalization to ensure that our model is robust to differences in the scale and distribution of our input variables. Finally, we will perform hyperparameter tuning to optimize the performance of our model by selecting the best combinations of model parameters.

OBJECTIVE

The main objective of this project is to develop a machine learning model that can accurately predict student dropout and success and to identify the most important factors that contribute to student dropout and success by performing feature selection on a range of relevant variables.

DATASET

The dataset we are using in this project is taken from UC Irvine repository. The dataset contains 4424 people records and 37 features. We are using “Target” as our target variable. It is a multi-class classification problem, where our target variable contains three classes.



```
Loading Dataset

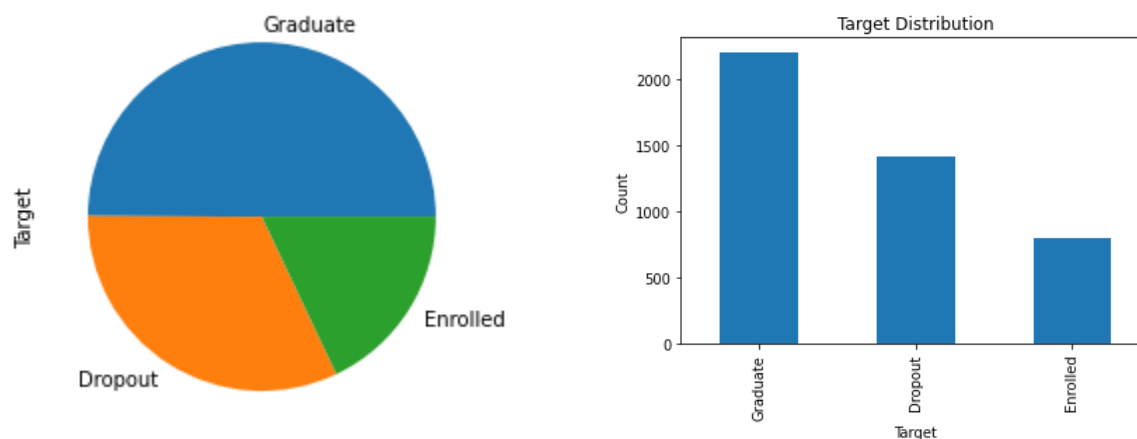
# Importing Dataset using Pandas.
data = pd.read_csv("data.csv", sep = ';')
data.shape

[0]: ✓ 0.3s
... (4424, 37)
```

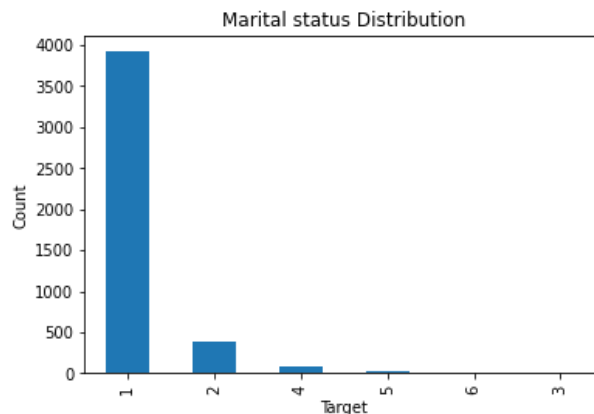
EXPLORATORY DATA ANALYSIS

In this project, before performing data preprocessing, we are conducting EDA to get deep insights from our data. We are using different visualizations libraries such as seaborn, matplotlib, and pandas etc. for this purpose.

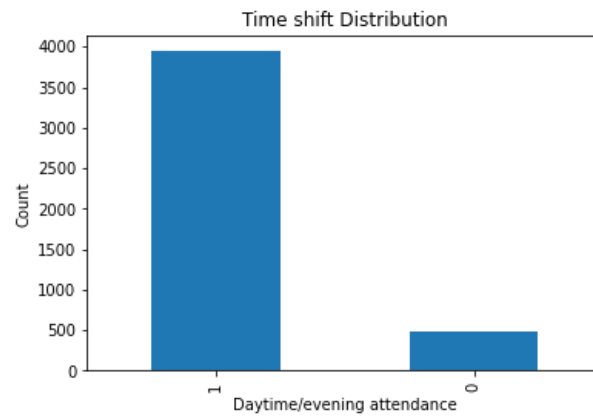
The first plot we are plotting in our exploratory data analysis is the distribution of our target variable status. From the below charts, we can clearly see that Graduate is the most common class in the dataset. In Pie chart, we can observe that around 50% are graduates and remaining are Dropout and Enrolled.



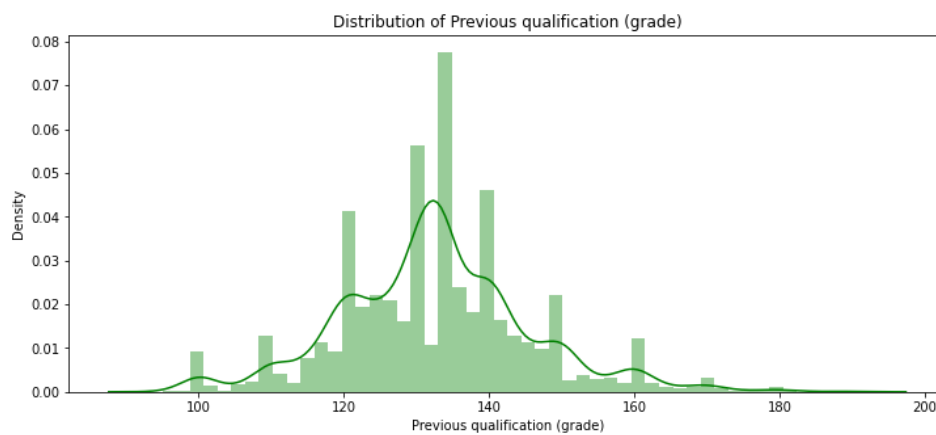
In our data, most people are single and some of them are married and engaged.



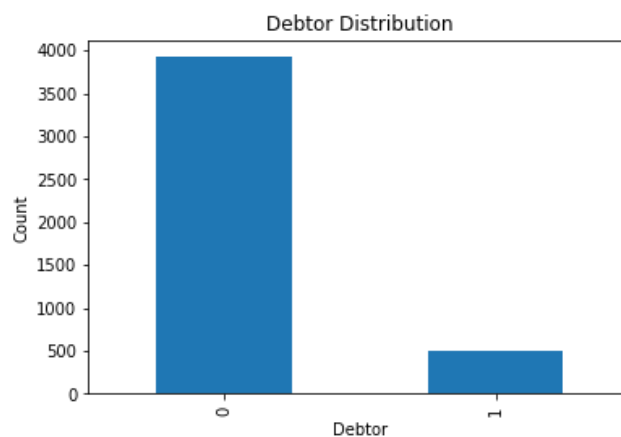
Most of the people are in Morning classes students as compared to the evening class. Which can be observed from the bar chart below.



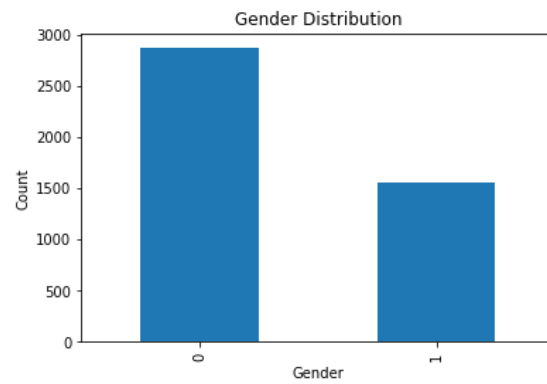
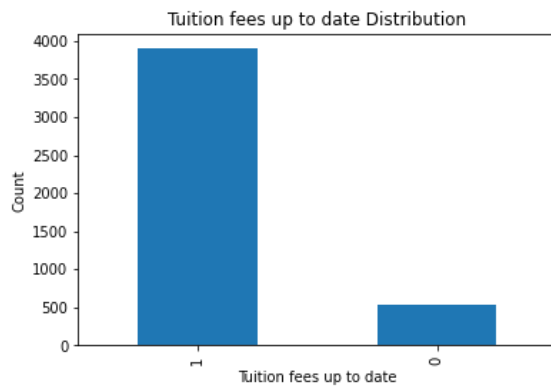
Average Grade points of student's previous grade are around 135 as can be seen in the distribution below.



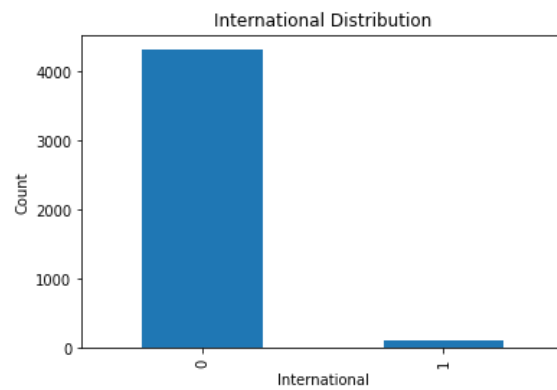
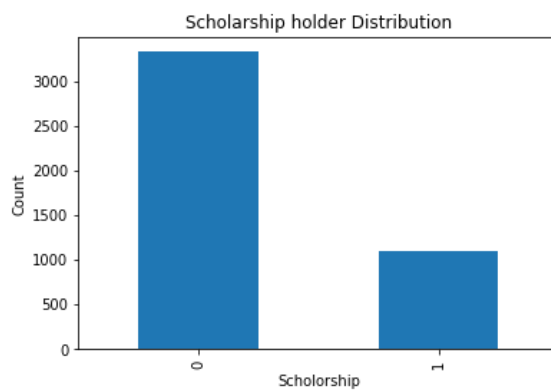
Very few students are debtors and most of the students do not have any debts. Can be seen in the bar chart below.



Most of the people are Females in the dataset as compared to the males and most of them are paying their tuition fee on time.



Around 1000 students are scholarship holders and most of the students are local. The quantity of international students is quite low.



DATA PREPROCESSING

In machine learning projects, preprocessing is the process of preparing raw data for machine learning algorithms by cleaning and converting it. It consists of numerous phases, including handling missing data, encoding category variables, scaling numerical features, and deleting outliers. Data preprocessing aims to increase the precision and effectiveness of the generated models by preparing the data for machine learning modelling. Data preprocessing guarantees that machine learning algorithms can accurately analyze the data and find significant patterns and correlations among the features. Preprocessing the data allows us to take out duplicate or irrelevant information, minimize noise and inconsistencies, and generally raise the quality of the data. This step is critical to the success of any machine learning project, as the accuracy and reliability of the resulting models are heavily dependent on the quality of the data used to train them.

- TREATING NULL VALUES

Treating null or missing values is an essential step in data preprocessing for machine learning projects. Null or missing values can cause errors in machine learning algorithms and reduce the accuracy of the resulting models. The choice of method depends on the type and amount of missing data, the impact of missing data on the model's accuracy, and the specific requirements of the machine learning project. It is essential to handle missing values appropriately to ensure that the resulting machine learning models are accurate and reliable. In our project, null values are present among the features.

- SCALING DATASET

In machine learning, scaling is a common preprocessing step that involves converting the values of the dataset's feature values to a standard scale. This is done to make sure that every feature, regardless of scale or measurement units used in the past, contributes equally to the model. The ability to manage data of different magnitudes is improved by scaling, which also ensures that no characteristic dominates others due to its bigger scale. This reduces bias towards traits during model training, resulting in more accurate and balanced predictions. Scaling also facilitates effective and quick model convergence during training. The optimization techniques used in 10 training converge more rapidly when the feature scales are similar, which lowers the computing cost and shortens the training process. Third, scaling makes feature importance comparisons more meaningful, which improves model interpretability. The most important aspects for model predictions can be determined by directly comparing features of the same scale.

Scaling can help the model perform better on unknown input and be more general. Since they are less susceptible to fluctuations in feature magnitudes, models trained on scaled

data typically perform better in terms of generalization, producing predictions on real-world data that are more solid and dependable.

```
Feature Scaling

from sklearn.preprocessing import MinMaxScaler
scaling = MinMaxScaler()

scaling.fit(data)
data_scaled = scaling.transform(data)
```

[111] ✓ 0.1s Python

- LABEL ENCODING

Label encoding is a process of converting categorical variables or labels into numerical representations. In machine learning, algorithms work better with numerical inputs, and therefore, it is often necessary to encode categorical variables into numerical values for machine learning models to process them. Label encoding assigns a unique numerical value to each category in a categorical variable.

In our project, the dataset contains both string and numerical data, so we must convert our string features into numerical format.

```
Label Encoding

Converting Categorical variable into numerical format.

data['Target'].unique()
array(['Dropout', 'Graduate', 'Enrolled'], dtype=object)

data['Target'] = data['Target'].map({'Dropout': 0, 'Graduate': 1, 'Enrolled': 2})
data['Target'] = pd.to_numeric(data['Target'])
```

[186] ✓ 0.1s Python

[187] ✓ 0.1s Python

- DROPPING UNNECESSARY COLUMNS

We drop unnecessary columns such as ID from data before machine learning because these columns do not contribute to the prediction of the target variable and including them can introduce noise and unnecessary complexity to the machine learning models.

```
Now all our features are in Numerical format.

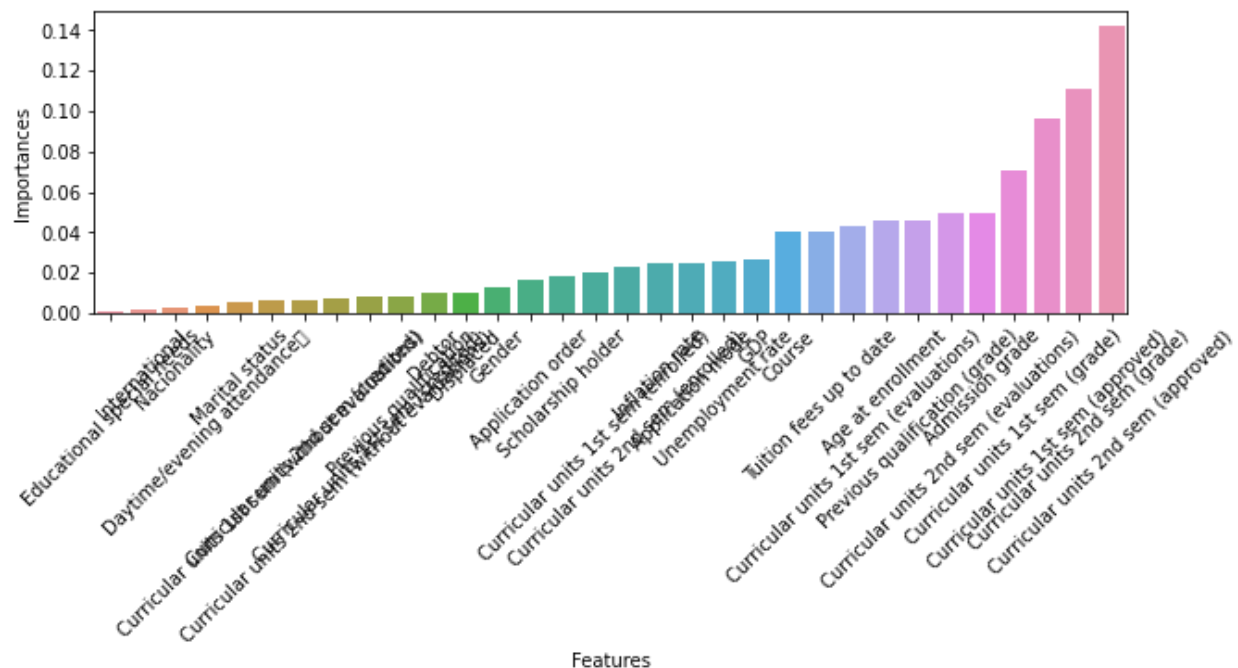
# Dropping Unnecessary Column.
data.drop("Mother's qualification",axis=1,inplace=True)
data.drop("Father's qualification",axis=1,inplace=True)
data.drop("Mother's occupation",axis=1,inplace=True)
data.drop("Father's occupation",axis=1,inplace=True)
```

[118] ✓ 0.1s Python

FEATURE SELECTION

One of the key steps in developing an accurate machine learning model for predicting student dropout and success is feature selection. This involves identifying the most relevant variables from a large set of potential predictors. We are using sequential forward selection (SFS) as our feature selection method. SFS is a wrapper method that involves iteratively adding features to the model and selecting the subset of features that produces the best performance. At each iteration, the algorithm evaluates the performance of the model using the selected features and the remaining unused features. It then adds the feature that provides the best performance and repeats the process until the desired number of features is selected. By using SFS, we aim to identify a smaller set of the most important predictors of student dropout and success, which can lead to a more efficient and effective predictive model.

We are selecting the 15 most impactful features out of the features from the data.



TRAIN TEST SPLIT

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. Here we are splitting

our dataset into train dataset and test dataset with the ratio of 70-30% where 70% is our train dataset and 20 is our test dataset.

```
trainX, testX, trainy, testy = train_test_split(X, y, test_size=0.3, random_state=2)
print(trainX.shape)
print(trainy.shape)
print(testX.shape)
print(testy.shape)
```

[121] ✓ 0.0s Python

... (3096, 15)
(3096, 1)
(1328, 15)
(1328, 1)

METHODOLOGY

In this project, we use various machine learning algorithms to predict the target variable and compare the outcomes of each approach to see which model fits and performs best on the training data set. To boost the model's performance, we are also using various model enhancement strategies. These methodologies include K-fold cross validation and grid search for hyperparameter tuning of models. Our only goal is to increase the model's performance with highest accuracy.

We are applying machine learning models after selecting the most impactful features.

Model Evaluation: -

We are evaluating the performance of our model's using accuracy, AUC graph, precision score, sensitivity, specificity, miss rate, precision, and recall values. Before we move forward to model implementation, it is necessary to know the following terms.

1) Precision: -

Precision is the ratio between the True Positives and all the Positives. For our problem statement, that would be the measure of cases that we correctly identify having a disease out of all the cases having it.

2) Recall: -

The recall is the measure of our model correctly identifying True Positives. Thus, for all the cases who have disease, recall tells us how many we correctly identified as having a disease.

3) Sensitivity: -

How well a machine learning model can identify positive examples is measured by its sensitivity. The true positive rate (TPR) or recall are other names for it. Sensitivity helps us to determine how many cases the model was able to accurately recognize, which is why it is utilized to assess model performance.

4) Specificity: -

Specificity is the proportion of true negatives that are correctly predicted by the model.

5) Miss Rate: -

The false negative rate – also called the miss rate – is the probability that a true positive will be missed by the test. It's calculated as $FN / (FN + TP)$, where FN is the number of false negatives and TP is the number of true positives (FN+TP being the total number of positives).

6) False Positive Rate: -

The false positive rate is calculated as $FP / (FP + TN)$, where FP is the number of false positives and TN is the number of true negatives (FP+TN being the total number of negatives). It's the probability that a false alarm will be raised: that a positive result will be given when the true value is negative.

7) Confusion Matrix: -

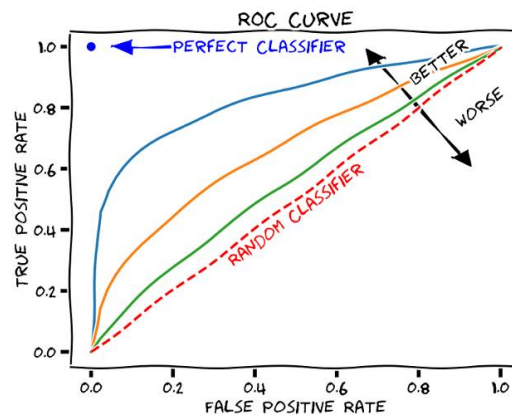
A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix.

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

8) ROC – AUC Curve: -

The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.

The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes. When $AUC = 1$, then the classifier can perfectly distinguish between all the Positive and the Negative class points correctly. If, however, the AUC had been 0, then the classifier would be predicting all Negatives as Positives, and all Positives as Negatives.

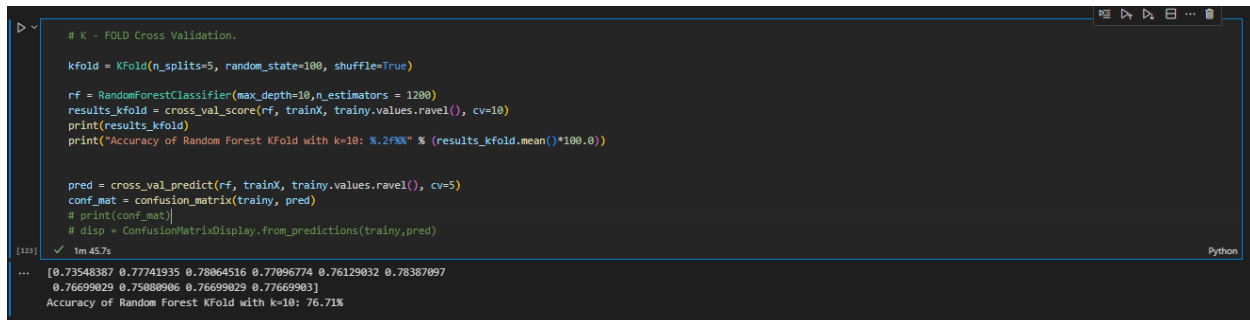


K fold Cross Validation

K-fold cross-validation is a technique used to evaluate the performance of a machine learning model by dividing the dataset into K subsets (or folds) of roughly equal size. The model is trained and evaluated K times, with each fold used as the test set once and the remaining K-1 folds used as the training set.

K-fold cross-validation is a commonly used technique for model evaluation because it provides a more accurate estimate of model performance than a single train-test split. By using multiple train-test splits, K-fold cross-validation reduces the risk of overfitting and provides a more reliable estimate of how the model will perform on new, unseen data.

In our project, we are using K fold cross validation with 10 folds. It tells about how our model will perform on the actual test data.



```
# K - FOLD Cross Validation.

kfold = KFold(n_splits=5, random_state=100, shuffle=True)

rf = RandomForestClassifier(max_depth=10, n_estimators = 1200)
results_kfold = cross_val_score(rf, trainX, trainy.values.ravel(), cv=10)
print(results_kfold)
print("Accuracy of Random Forest KFold with k=10: %.2f%%" % (results_kfold.mean()*100.0))

pred = cross_val_predict(rf, trainX, trainy.values.ravel(), cv=5)
conf_mat = confusion_matrix(trainy, pred)
# print(conf_mat)
# disp = ConfusionMatrixDisplay.from_predictions(trainy, pred)

[122] ✓ 1m 45.7s
... [0.73548387 0.77741935 0.78864516 0.77896774 0.76129832 0.78387097
0.76699029 0.75880906 0.76699029 0.77669903]
Accuracy of Random Forest KFold with k=10: 76.71%
```

Grid Search Hyper Parameter Tuning

To determine the ideal set of hyperparameters for a particular model, we implemented the grid search hyperparameter tuning technique. Hyperparameters are model parameters that are selected by the user prior to training rather than being learned from the data. In our project, we are using Grid search technique to get the best possible hyper parameters for the model to increase its efficiency.

L2 Regularization

L2 Regularization, also known as ridge regression, is quite famous for reducing overfitting by adding penalty terms in the model. We are implementing it in logistic regression model. L2 regularization can help a logistic regression model by preventing overfitting, improving

generalization, and increasing the model's robustness. When a logistic regression model is trained on a dataset, it may end up fitting too closely to the training data, resulting in overfitting.

MODEL IMPLEMENTATION

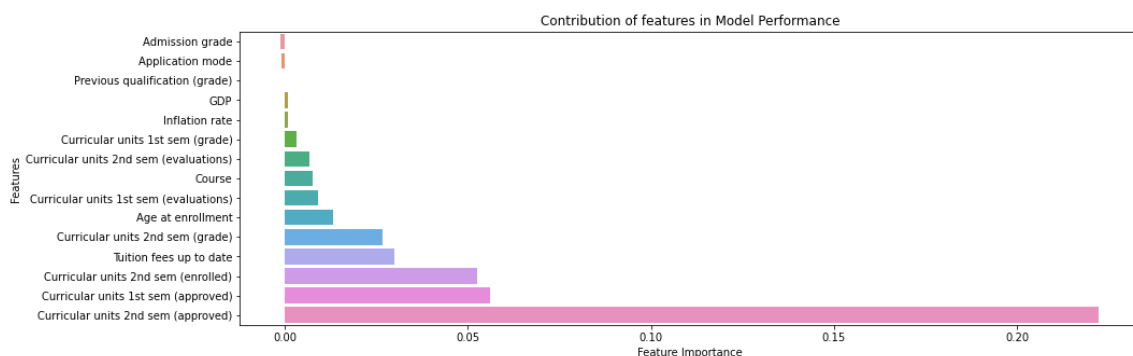
We are implementing different machine learning models in this project to predict our target variable and using different performance metrics to evaluate the performance of the models.

1- Logistics Regression

A statistical technique called logistic regression is used to examine data in which one or more independent variables affect how something turns out. It is frequently applied to situations involving binary classification, where the objective is to forecast a binary result based on one or more independent variables.

The logistic function, also referred to as the sigmoid function, is used in the logistic regression model, a sort of generalized linear model, to describe the likelihood of a binary outcome.

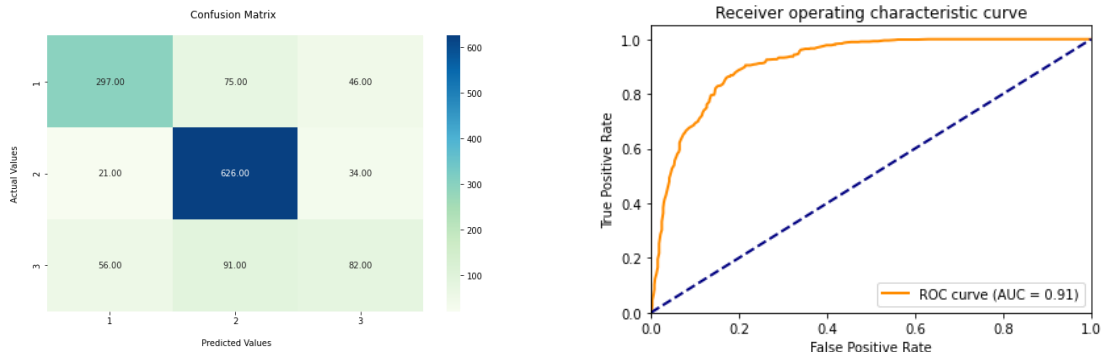
In our project, logistic regression gives accuracy of 75% and after adding penalty term its model performance decreases.



2- Random Forest Classifier

A random forest classifier is an ensemble learning method that combines multiple decision trees to improve the accuracy and robustness of the classification model. The algorithm builds multiple decision trees on random subsets of the input features and the training data, and then combines their predictions to make a final classification decision.

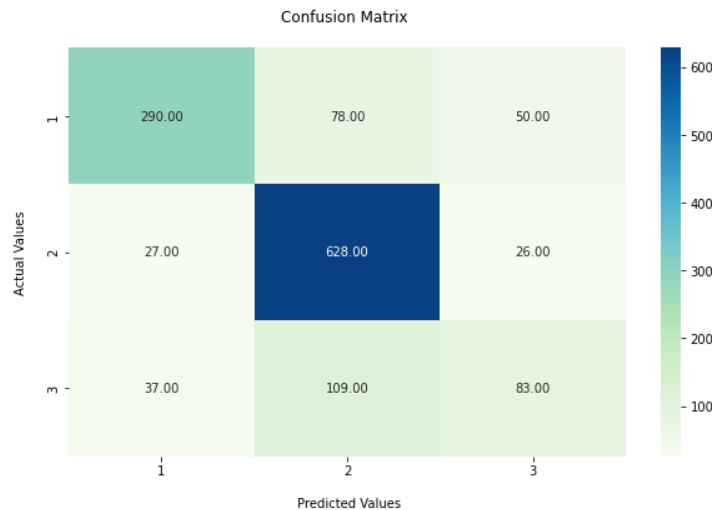
For our project, random forest works better than logistic regression and support vector machine with 76% accuracy and 0.91 ROC score which means that the model is performing reasonably well in classifying the data.



3- Support Vector Machine

SVMs work by finding the optimal hyperplane that separates the data into different classes while maximizing the margin between the classes. In classification tasks, SVMs aim to find the hyperplane that correctly separates the different classes of data, while in regression tasks, the goal is to find the hyperplane that best fits the data.

In our project, the support vector machine is working fine and giving accuracy of 75% same as logistic regression.



RESULTS

Model	Random Forest	Logistic Regression	SVM
Accuracy	76.23%	75.6%	75.36%
Specificity	0.798	0.75	0.788
Sensitivity	0.967	0.96	0.95
Recall	0.66	0.63	0.66
Precision	0.765	0.70	0.70
Miss Rate (FNR)	0.03	0.03	0.04
Miss Rate (FPR)	0.201	0.21	0.211

Here we can see that random forests are getting the highest accuracy score as compared to the other model maybe because random forest is ensemble technique and specially designed to deal with overfitting issue. Other than that random forest model is also good in dealing with high dimensional data. Recall and precision of random forest are also better than other two models.

Logistic regression and SVM are also performing good in terms of accuracy but their precision scores are not up to the mark which means model is predicting too many false negatives maybe because of imbalance dataset.

CONCLUSION

In conclusion, four alternative models were implemented for our study on applying machine learning to predict student dropout. The random forest model outperformed the other models in terms of predicted accuracy and performance measures, according to our thorough experimentation and review. In terms of accuracy, precision, recall, and confusion metrics, random forests outscored other models, demonstrating their superior performance and robustness in target prediction. By reducing overfitting and enhancing prediction, and the use of ensemble techniques in random forest, which aggregates decision trees, contributed to its higher performance. According to our research, random forests are a suitable and trustworthy model for predicting student dropout, and it might be successfully applied in real-world settings.

LIMITATIONS

Although the techniques used in this project helped to improve the performance of the models, there are some limitations that should be considered.

- First, the quality of the results depends on the quality of the data. If the data is low in quantity, or biased, the performance of the models may be limited.
- For this project only 3 algorithms were used, there might be other supervised algorithms such as Gradient boosting, Naïve Bayes etc. that can also be used to check if they are performing better or low.
- We can also use ensemble techniques such as voting classifier model to create more robust model.
- Other limitation is, instead of splitting the dataset, a different dataset could be used to evaluate the model performance.