Credit Risk Assessment Model

Group 13

Vinan Patwa
Sanay Shah
Archita Kothari
Bhavya Maheshwari

Percentage of Effort Contributed by Vinan Patwa: 25%
Percentage of Effort Contributed by Sanay Shah: 25%
Percentage of Effort Contributed by Archita Kothari: 25%
Percentage of Effort Contributed by Bhavya Maheshwari: 25%

Submission Date: 04/17/23

PROBLEM DEFINITION:

Credit risk refers to the potential loss that a bank may incur if a borrower fails to repay a loan or credit card debt. Banks face a significant amount of credit risk as they provide loans and credit facilities to individuals and businesses.Credit risk analysis is the process of assessing the likelihood that a borrower will default on their loan or credit card debt, and the potential loss that may result from such a default. Banks use credit risk analysis to make informed lending decisions and to manage their credit portfolios.

Effective credit risk analysis helps banks to:

- Identify high-risk borrowers and loans that may be more likely to default
- Set appropriate interest rates and loan terms based on the level of credit risk
- Monitor the credit performance of borrowers and detect early warning signs of potential defaults
- Mitigate losses by taking appropriate measures to recover unpaid debts or collateral.

In summary, credit risk analysis is crucial for banks to manage their credit risk exposure and maintain a healthy financial position. The goal of the project is to build a predictive model that can accurately predict the likelihood of a credit card client defaulting on their payments based on the given attributes. The model will be trained on a dataset containing historical data of credit card clients and their default statusThis can be a valuable tool for credit card companies to assess the risk associated with granting credit to new clients and to make informed decisions about lending.

DATASET:

https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

DATASET DESCRIPTION:

The dataset is about credit card customers and their payment behavior. Specifically, it includes information on the credit amount, demographic characteristics, past payment history, and bill statement and previous payment amounts for credit card customers from April to September 2005 in Taiwan. The dataset aims to investigate the factors that influence default payment among credit card customers, where default payment is a binary variable with two categories: Yes (1) and No (0).

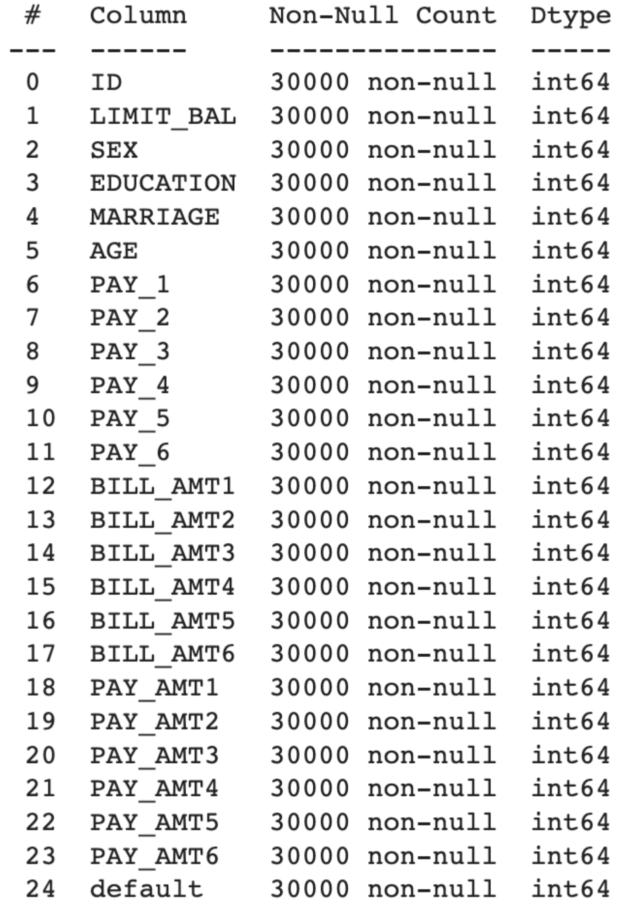
- ID: ID of each client
- LIMIT_BAL: Amount of given credit in NT dollars (includes individual and family/supplementary credit
- SEX: Gender (1=male, 2=female)
- EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown)
- MARRIAGE: Marital status (1=married, 2=single, 3=others)
- AGE: Age in years
- PAY_0: Repayment status in September, 2005 (-1=pay duly, 1=payment delay for one month, 2=payment delay for two months, … 8=payment delay for eight months, 9=payment delay for nine months and above)
- PAY_2: Repayment status in August, 2005 (scale same as above)
- PAY_3: Repayment status in July, 2005 (scale same as above)
- PAY_4: Repayment status in June, 2005 (scale same as above)
- PAY_5: Repayment status in May, 2005 (scale same as above)
- PAY_6: Repayment status in April, 2005 (scale same as above)
- BILL_AMT1: Amount of bill statement in September, 2005 (NT dollar)
- BILL_AMT2: Amount of bill statement in August, 2005 (NT dollar)
- BILL_AMT3: Amount of bill statement in July, 2005 (NT dollar)
- BILL_AMT4: Amount of bill statement in June, 2005 (NT dollar)
- BILL_AMT5: Amount of bill statement in May, 2005 (NT dollar)
- BILL_AMT6: Amount of bill statement in April, 2005 (NT dollar)
- PAY_AMT1: Amount of previous payment in September, 2005 (NT dollar)
- PAY_AMT2: Amount of previous payment in August, 2005 (NT dollar)
- PAY_AMT3: Amount of previous payment in July, 2005 (NT dollar)
- PAY_AMT4: Amount of previous payment in June, 2005 (NT dollar)
- PAY_AMT5: Amount of previous payment in May, 2005 (NT dollar)
- PAY_AMT6: Amount of previous payment in April, 2005 (NT dollar)
- default.payment.next.month: Default payment (1=yes, 0=no)

PLANNING:

- **Exploratory Data Analysis (EDA):** This step involves understanding the relationships between the explanatory variables and the response variable, identifying patterns and trends in the data, and exploring potential outliers or anomalies.
- **Data Preprocessing:** The first step is to preprocess the data, which involves cleaning, transforming, and preparing the data for analysis. This may include handling missing values, transforming categorical variables into numerical ones,
- **Feature Selection:** This step involves selecting the most relevant features (i.e., explanatory variables) for the prediction task. Using various methods such as correlation analysis, feature importance ranking, and dimensionality reduction techniques.
- **Model Selection:** This step involves selecting the appropriate machine learning algorithms to predict the response variable. using payment include logistic regression, decision trees, random forests, and support vector machines (SVM).
- **Model Evaluation:** This step involves evaluating the performance of the machine learning algorithms using appropriate metrics such as accuracy, precision, recall, and F1-score. Additionally, cross-validation techniques such as k-fold cross-validation can be used to evaluate the robustness of the models.
- **Model Tuning:** This step involves fine-tuning the hyperparameters of the machine learning algorithms to optimize their performance on the dataset. This can be done using techniques such as grid search or random search.
- **Model Deployment:** This step involves deploying the final machine learning model to predict the default payment for new credit card customers.

SUMMARY OF DATASET

Number of samples: 30000
Number of features: 25



```
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   ID         30000 non-null  int64
 1   LIMIT_BAL  30000 non-null  int64
 2   SEX        30000 non-null  int64
 3   EDUCATION  30000 non-null  int64
 4   MARRIAGE   30000 non-null  int64
 5   AGE        30000 non-null  int64
 6   PAY_1      30000 non-null  int64
 7   PAY_2      30000 non-null  int64
 8   PAY_3      30000 non-null  int64
 9   PAY_4      30000 non-null  int64
 10  PAY_5      30000 non-null  int64
 11  PAY_6      30000 non-null  int64
 12  BILL_AMT1  30000 non-null  int64
 13  BILL_AMT2  30000 non-null  int64
 14  BILL_AMT3  30000 non-null  int64
 15  BILL_AMT4  30000 non-null  int64
 16  BILL_AMT5  30000 non-null  int64
 17  BILL_AMT6  30000 non-null  int64
 18  PAY_AMT1   30000 non-null  int64
 19  PAY_AMT2   30000 non-null  int64
 20  PAY_AMT3   30000 non-null  int64
 21  PAY_AMT4   30000 non-null  int64
 22  PAY_AMT5   30000 non-null  int64
 23  PAY_AMT6   30000 non-null  int64
 24  default    30000 non-null  int64
```

DATA CLEANING AND VALIDATION

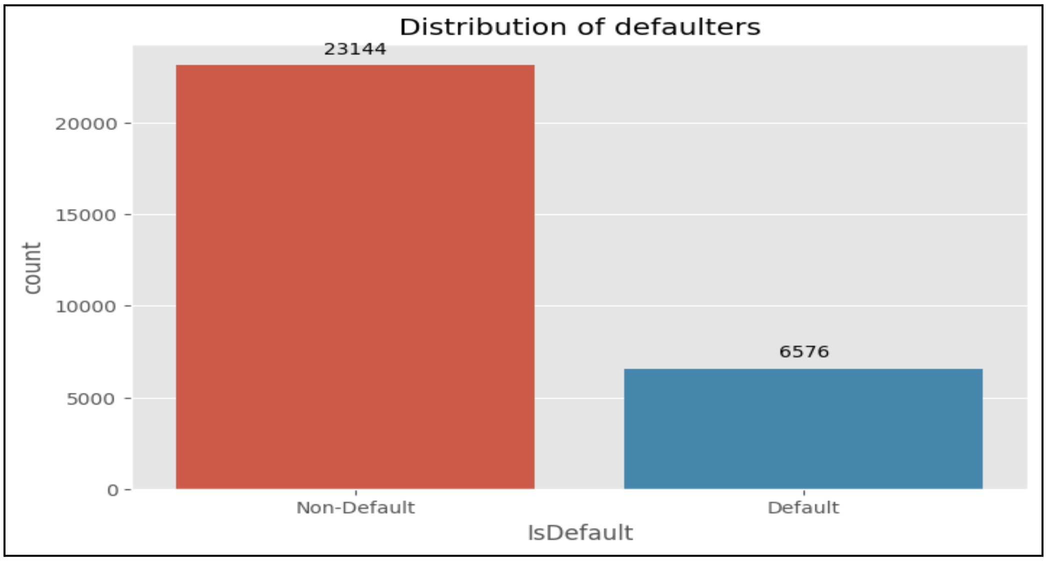
Data validation:
The dataset has been filtered to remove rows in which the payment amount in any column exceeds the corresponding credit limit column. This resulted in the removal of 280 sample points, which accounts for 0.93% of the total dataset.

A mapping was created between the education level categories and their corresponding numerical values. As we can see in dataset we have values like 5,6,0 as well for which we are not having description so we can add up them in 4, which is Others. The mapping used in this project is as follows:

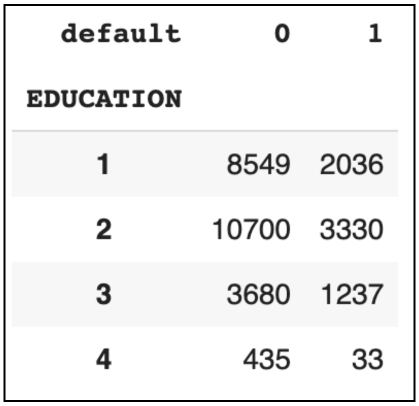
- 1: Graduate School
- 2: University
- 3: High School
- 4: Others

Marital status refers to an individual's legal and social standing in relation to their marriage. The three most common categories are married (1), single (2), and others (3).

EXPLORATORY DATA ANALYSIS

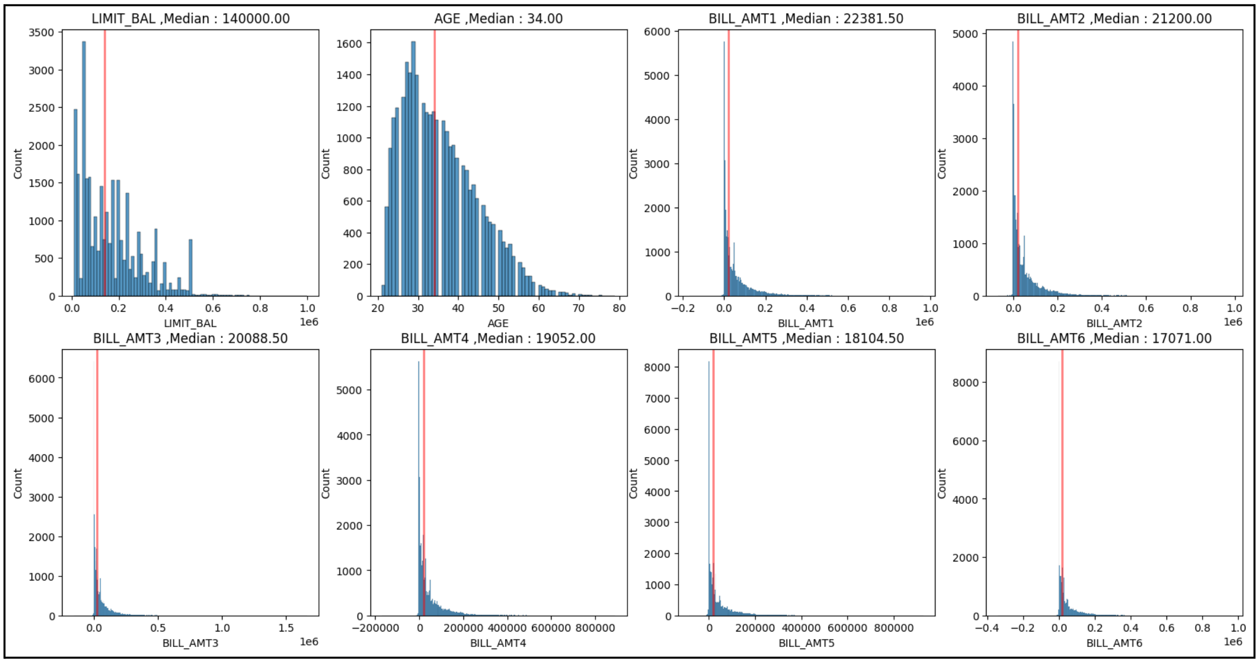


From the visualization, it is clear that the data is imbalanced, with one class (non-defaulters) being much more prevalent than the other (defaulters). The values 0 and 1 are being used to represent the non-defaulters and defaulters, respectively.



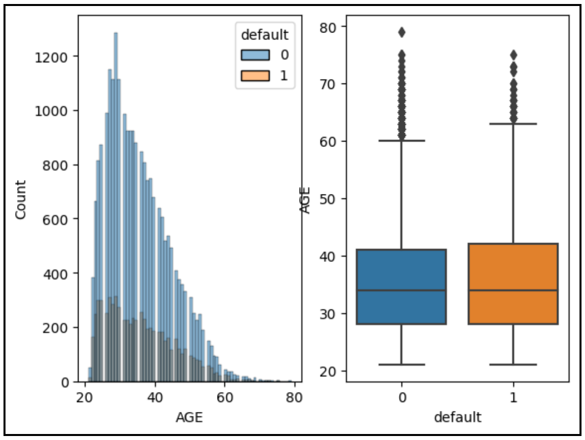
default	0	1
EDUCATION		
1	8549	2036
2	10700	3330
3	3680	1237
4	435	33

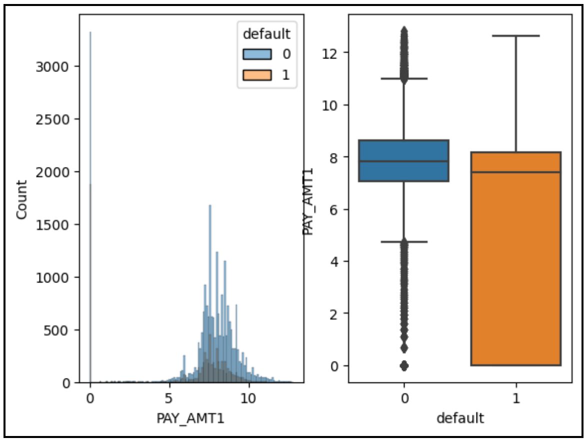
Approximately 25% of customers who defaulted on their payments had a High School (2) degree, while around 24% of defaulters had a University (4) degree.

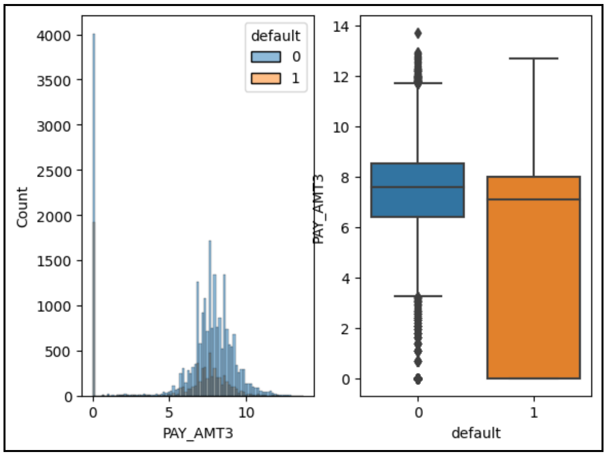


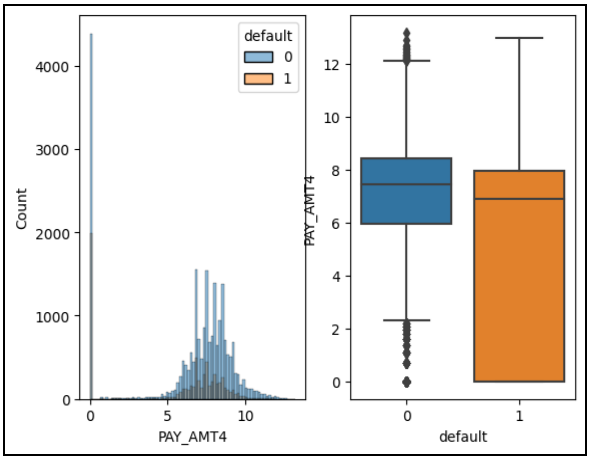
Credit Amount Distribution (Limit Bal):
Clients with lower amounts tend to default. Especially those with credit amount around 50000 default most.

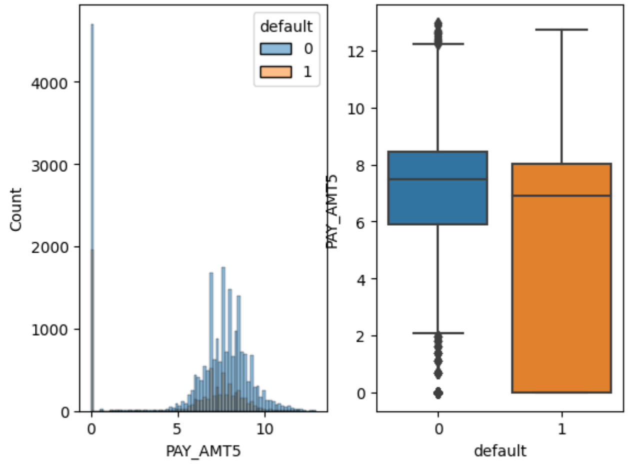
Age distribution:
As age increases to 30, the probability of default increases. Meanwhile, when clients are over 30, the probability decreases when aging.

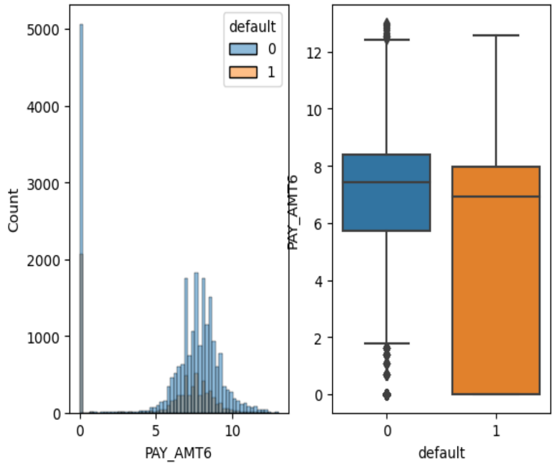






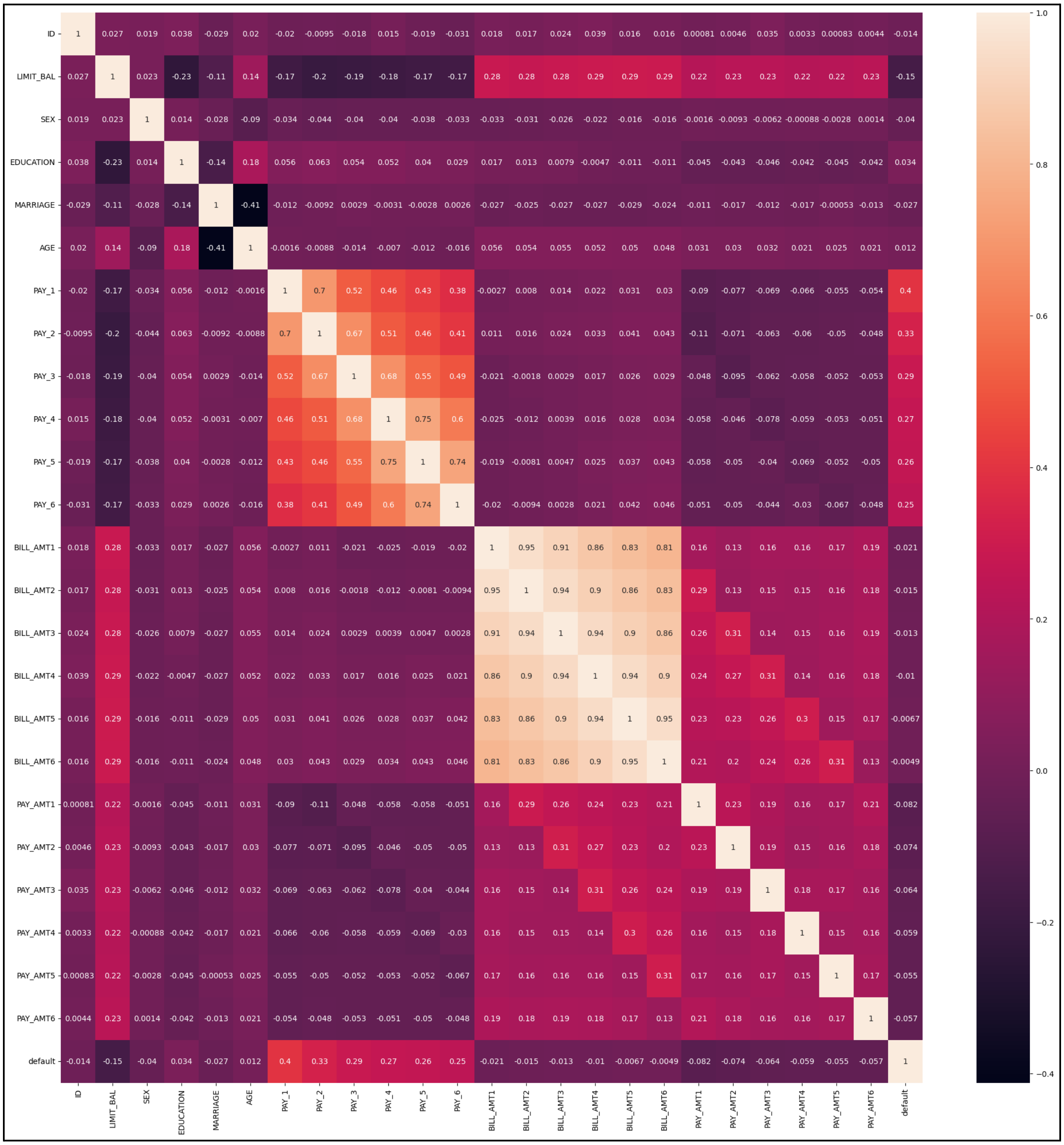




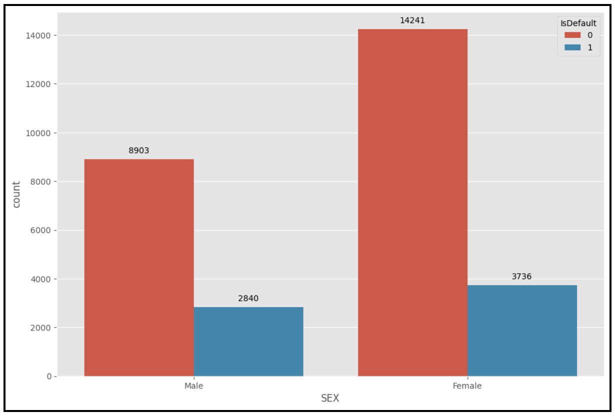


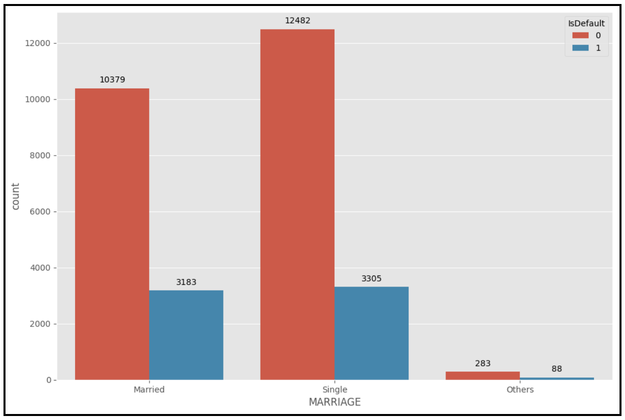
PAY_AMT Distribution
As we can see in the box plots, non-defaulter is showing a lot of outliers compared to defaulters.

Correlation Plot



The correlation plot indicates a strong correlation between all BILL_AMT variables, which may be attributed to clients having similar monthly expenses for which bills are generated.



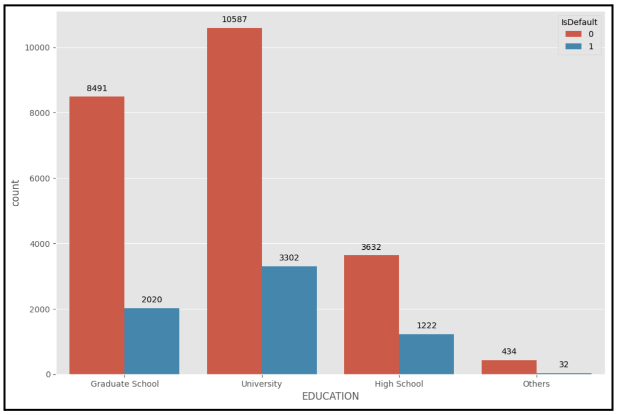


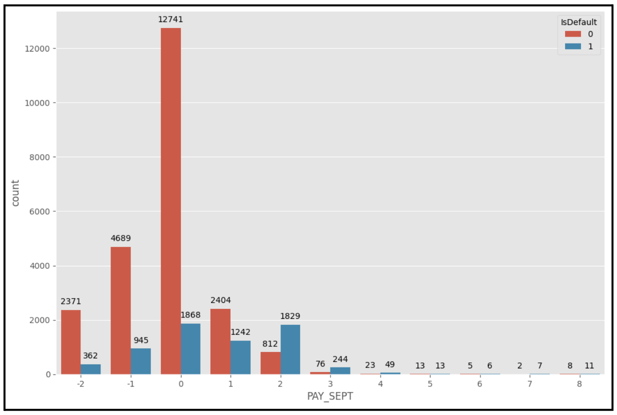
Sex Distribution:

Female has more probability of default than male.

Marriage Distribution:

The number of defaulters is nearly equal for both Married and Single categories, while the count of non-defaulters is greater for the Single category. Additionally, the Other category has a significantly lower representation in the dataset compared to the other two categories.





Education Distribution:

The probability of default among clients with 'graduate school', 'university', and 'high school' education is comparable. However, the total count of clients with 'high school' education and those categorized as 'Others' is significantly lower compared to clients who completed their 'graduate school' or 'university' education.

Repayment Distribution:

Probability of non-defaulters for Repayment status in September 2005 is most than all other categories.

DATA PREPROCESSING

a) One Hot Encoding:

Performing one-hot encoding on the columns 'EDUCATION', 'MARRIAGE', 'PAY_SEPT', 'PAY_AUG', 'PAY_JUL', 'PAY_JUN', 'PAY_MAY', 'PAY_APR' using the get_dummies() function of pandas library. This function converts categorical variables into dummy/indicator variables with a value of 0 or 1. The columns parameter is used to specify the columns that need to be encoded.

For eg the **EDUCATION** column had 4 categories so now in the dataframe we have additional 4 columns.They are EDUCATION_1, EDUCATION_2, EDUCATION_3,EDUCATION_4

b) Normalization:

Normalization of numerical data is the process of bringing all the variables in a dataset to a common scale. This is important because it makes it easier to compare and analyze the variables, especially when they have different units or scales of measurement. Normalization ensures that each variable contributes equally to the analysis and improves the quality and accuracy of data analysis. Different methods of normalization exist, and the choice of method depends on the specific requirements of the analysis.

c) Balancing the dataset :

1. SMOTE :Synthetic Minority Over-sampling Technique

SMOTE is a technique in machine learning used to deal with class imbalance problems in datasets. This happens when one class in a dataset has significantly fewer examples than the other classes, which can lead to biased models that don't perform well on the minority class. To address this, SMOTE creates new synthetic samples of the minority class by interpolating between existing samples. This is done by randomly selecting one or more of the nearest neighbors of a minority sample and generating new synthetic samples along the line segments connecting them. By doing this, the imbalance between classes is reduced, and the minority class is over-sampled.

2.SMOTE -Tomek :

Tomek Links are pairs of samples that are close to each other, but belong to different classes. These pairs of samples are considered "noisy" and are often removed to improve the performance of classification models.
When used together, SMOTE-Tomek is a hybrid technique that first applies SMOTE to oversample the minority class, and then applies Tomek Links to remove any noisy samples that

were generated by SMOTE. This results in a better separation between the minority and majority classes, and can improve the performance of classification models.

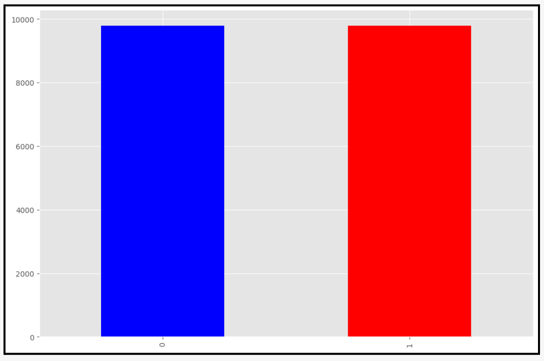
3. SMOTE-ENN : Synthetic Minority Over-sampling Technique and Edited Nearest Neighbors

SMOTE-ENN is another hybrid technique used for imbalanced classification problems.
Like SMOTE-Tomek, SMOTE-ENN also combines two techniques to address imbalanced data. The SMOTE algorithm generates synthetic samples to oversample the minority class, while the Edited Nearest Neighbors (ENN) algorithm removes some of the noisy samples from the majority class.

4.ENN-SMOTE : Edited Nearest Neighbors and Synthetic Minority Over-sampling Technique

ENN-SMOTE is another hybrid technique for addressing imbalanced classification problems.ENN-SMOTE first applies the ENN algorithm to remove some of the noisy samples from both the majority and minority classes. Then, the SMOTE algorithm is applied to oversample the minority class by generating synthetic samples.The ENN algorithm works by examining each sample in the dataset and removing it if it is misclassified by its nearest neighbors. This process helps to remove noisy samples from the dataset, and can improve the accuracy of classification models.

After over sampling the count of both the classes become equal



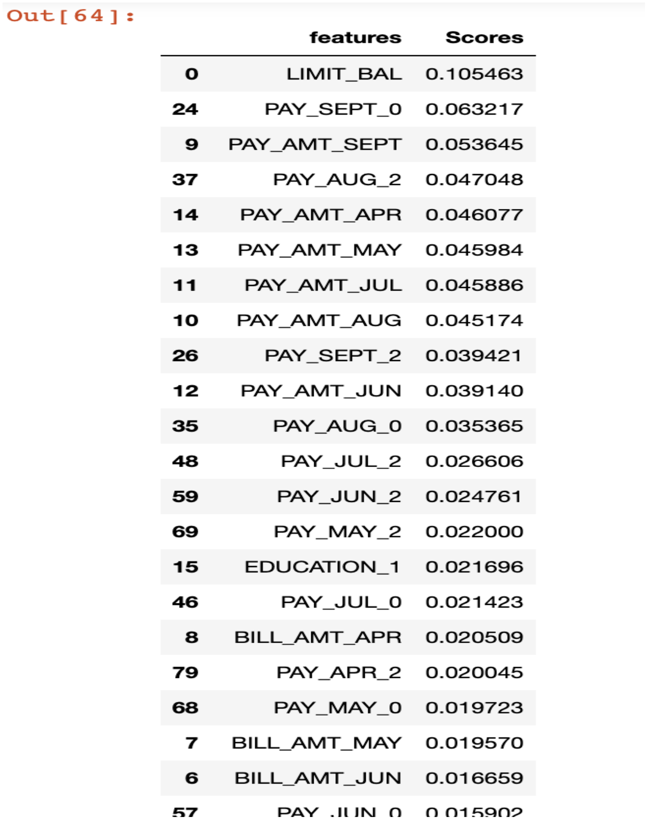
d) Feature Selection:

1)Mutual Information:

Mutual information is a measure of the amount of information that is shared between two variables.is often used in machine learning and information theory to identify the statistical dependence or correlation between two variables.The mutual information between two variables X and Y is calculated by measuring the reduction in uncertainty of one variable when the value of the other variable is known. In other words, it measures how much knowing the value of one variable can help in predicting the value of the other variable.Mutual information is a non-negative quantity, with higher values indicating greater dependence or correlation between the variables

The given code performs feature selection on a dataset using mutual information as the score function. The function uses ‘SelectKBest’ from the ‘sklearn.feature_selection’ module to select the best features based on mutual information score.
The ‘SelectKBest’ function selects the top k features based on the provided score function. In this case, ‘k’ is set to 'all' to select all the features. The ‘mutual_info_classif’ function is used as the score function, which is an implementation of mutual information for classification tasks.

Output:



Out[64]:

	features	Scores
0	LIMIT_BAL	0.105463
24	PAY_SEPT_0	0.063217
9	PAY_AMT_SEPT	0.053645
37	PAY_AUG_2	0.047048
14	PAY_AMT_APR	0.046077
13	PAY_AMT_MAY	0.045984
11	PAY_AMT_JUL	0.045886
10	PAY_AMT_AUG	0.045174
26	PAY_SEPT_2	0.039421
12	PAY_AMT_JUN	0.039140
35	PAY_AUG_0	0.035365
48	PAY_JUL_2	0.026606
59	PAY_JUN_2	0.024761
69	PAY_MAY_2	0.022000
15	EDUCATION_1	0.021696
46	PAY_JUL_0	0.021423
8	BILL_AMT_APR	0.020509
79	PAY_APR_2	0.020045
68	PAY_MAY_0	0.019723
7	BILL_AMT_MAY	0.019570
6	BILL_AMT_JUN	0.016659
57	PAY_JUN_0	0.015902

Feature selection for Decision Tree using feature importance

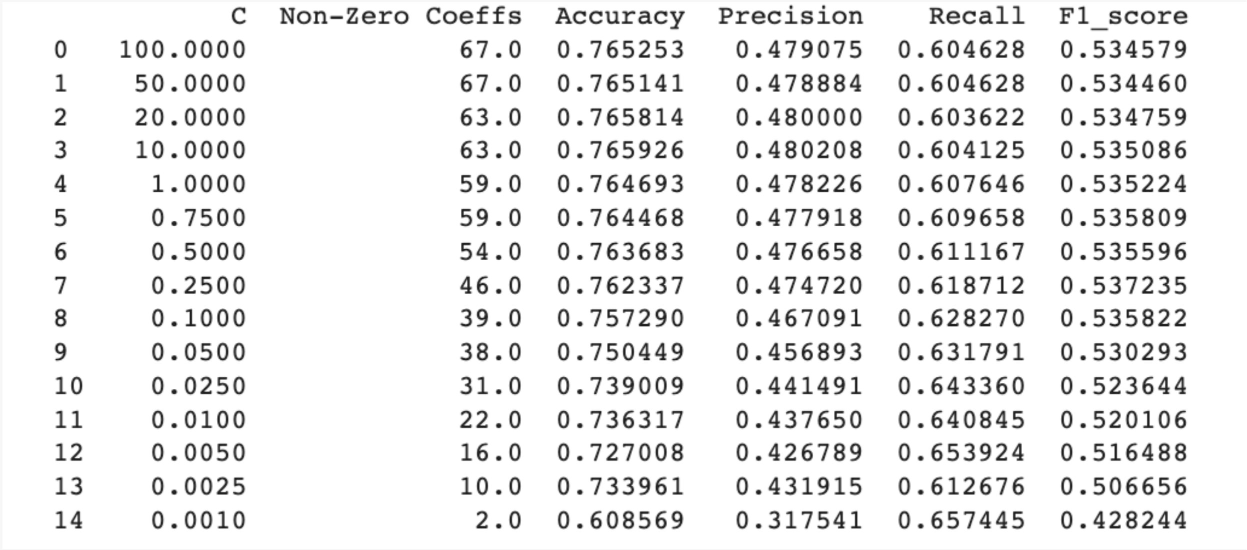
The sample dataframe created from the original dataset, and the independent variables (features) are stored in a variable called X_lr, and the dependent variable (target) is stored in a variable called y_lr.The code applies feature selection using Recursive Feature Elimination (RFE) from the scikit-learn library. The RFE algorithm works by recursively removing features and building a model on the remaining features until a desired number of features is reached. The optimal number of features is selected using the cross-validation method.After applying RFE, the code selects the most important features for decision tree analysis. The selected features are stored in a variable called X_selected_lr.The use of RFE allows for the selection of the most important features, which can improve the accuracy of the decision tree model.Out of 85 columns 17 columns were selected.

The features are as follows:
LIMIT_BAL, AGE, BILL_AMT_SEPT, BILL_AMT_AUG, BILL_AMT_JUL, BILL_AMT_JUN, BILL_AMT_MAY, BILL_AMT_APR, PAY_AMT_SEPT, PAY_AMT_AUG, PAY_AMT_JUL, PAY_AMT_JUN, PAY_AMT_MAY, PAY_AMT_APR, SEX, PAY_SEPT_2, PAY_AUG_2

Feature selection for Logistic Regression using feature importance

The sample dataframe created from the original dataset, and the independent variables (features) are stored in a variable called X_lr, and the dependent variable (target) is stored in a variable called y_lr.The code applies feature selection using Recursive Feature Elimination (RFE) from the scikit-learn library. The RFE algorithm works by recursively removing features and building a model on the remaining features until a desired number of features is reached. The optimal number of features is selected using the cross-validation method.After applying RFE, the code selects the most important features for logistic regression analysis. The selected features are stored in a variable called X_selected_lr.The use of RFE allows for the selection of the most important features, which can improve the accuracy of the logistic regression model.Out of 85 columns 38 columns were selected.

The code gave the following output.



```
            C  Non-Zero Coeffs  Accuracy  Precision    Recall  F1_score
0   100.0000              67.0  0.765253   0.479075  0.604628  0.534579
1    50.0000              67.0  0.765141   0.478884  0.604628  0.534460
2    20.0000              63.0  0.765814   0.480000  0.603622  0.534759
3    10.0000              63.0  0.765926   0.480208  0.604125  0.535086
4     1.0000              59.0  0.764693   0.478226  0.607646  0.535224
5     0.7500              59.0  0.764468   0.477918  0.609658  0.535809
6     0.5000              54.0  0.763683   0.476658  0.611167  0.535596
7     0.2500              46.0  0.762337   0.474720  0.618712  0.537235
8     0.1000              39.0  0.757290   0.467091  0.628270  0.535822
9     0.0500              38.0  0.750449   0.456893  0.631791  0.530293
10    0.0250              31.0  0.739009   0.441491  0.643360  0.523644
11    0.0100              22.0  0.736317   0.437650  0.640845  0.520106
12    0.0050              16.0  0.727008   0.426789  0.653924  0.516488
13    0.0025              10.0  0.733961   0.431915  0.612676  0.506656
14    0.0010               2.0  0.608569   0.317541  0.657445  0.428244
```

From the above chart we can observe that till penalty = '11' and C =0.05 the values hardly change.

MODEL IMPLEMENTATION

We think that several Machine Learning algorithms can be employed to address this classification problem. Some of the algorithms that can be considered are:

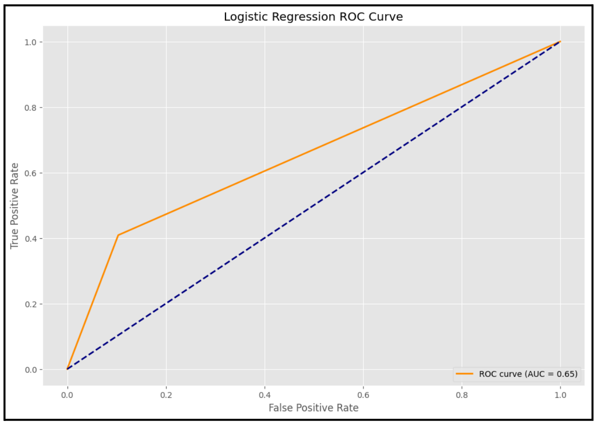
1. Logistic Regression
2. Naive Bayes
3. Support Vector Machines
4. Decision Tree
5. Neural Networks

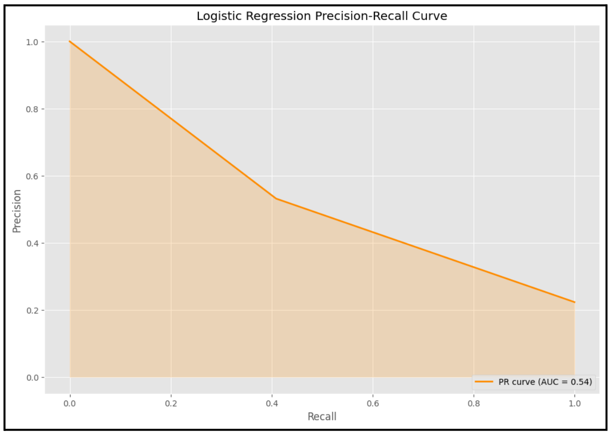
Logistic Regression:

Logistic regression is a parametric classification model that uses a specific model to link predictor factors to the target variable, resulting in a categorical outcome variable. The results are estimates of the odds of belonging to each class, followed by a threshold cutoff for classifying into either of the classes.Gradient descent is an iterative optimization algorithm that adjusts the model's parameters in the direction of the steepest descent of the cost function. The learning rate determines the step size of each iteration, and the tolerance sets the threshold for convergence, i.e., when the change in the cost function becomes smaller than the tolerance. The gradient descent algorithm for logistic regression involves calculating the gradient of the cost function with respect to each parameter and updating them accordingly. By repeatedly adjusting the parameters through gradient descent, the model can converge to a set of parameters that minimize the cost function and provide accurate predictions for the input data.For the following code we have Tolerance rate = 0.0005 Max Iteration =10000 and for different learning rate ROC curve and Precision_recall Curve graphs are plotted below

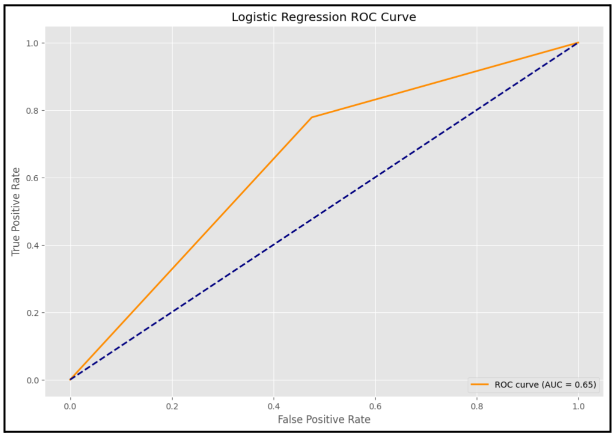
Without Feature Importance:

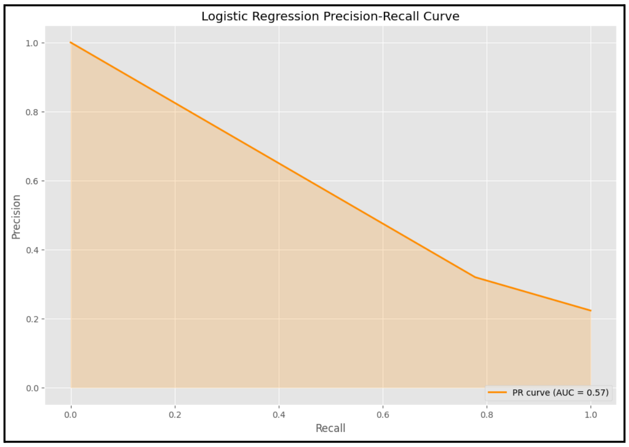
Learning Rate = 0.0003



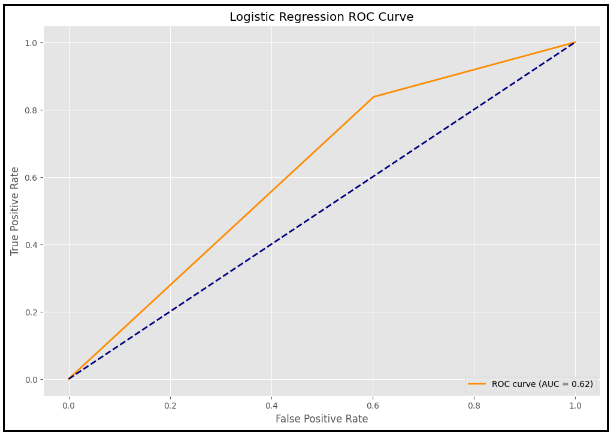


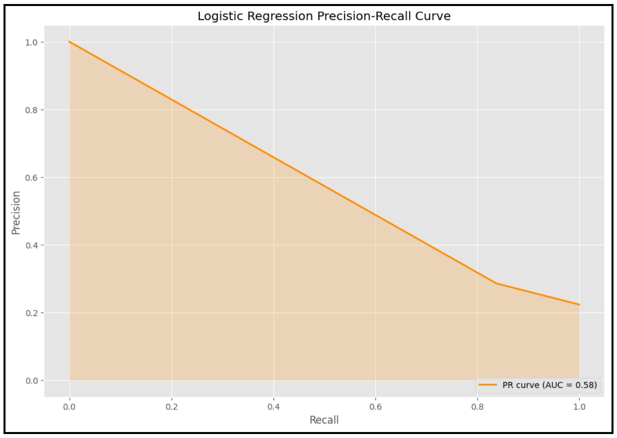
Learning Rate = 0.0005



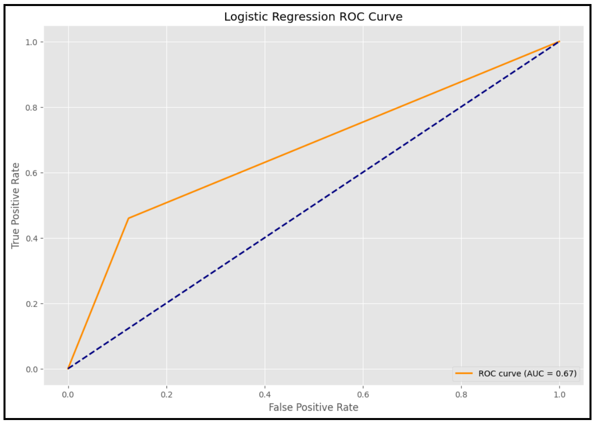


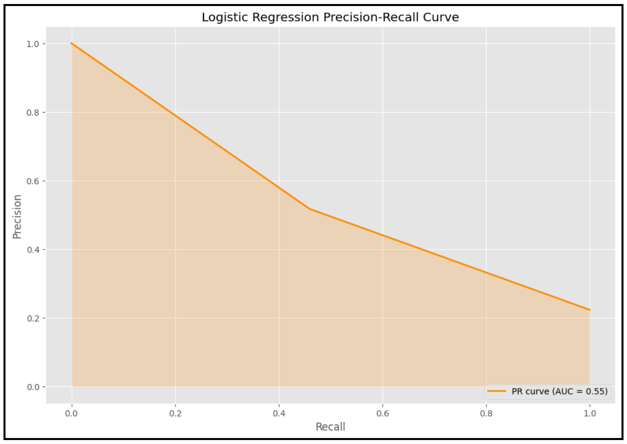
Learning Rate = 0.0007



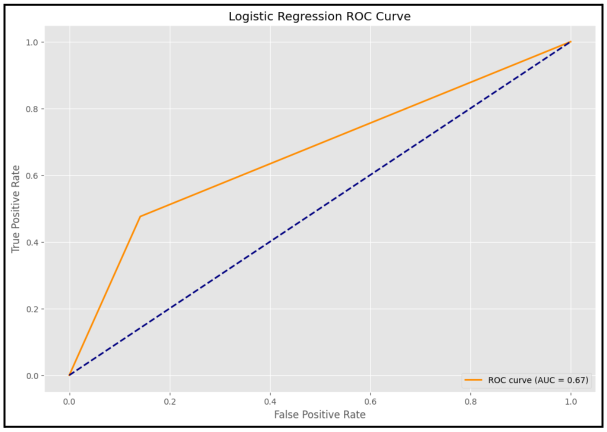


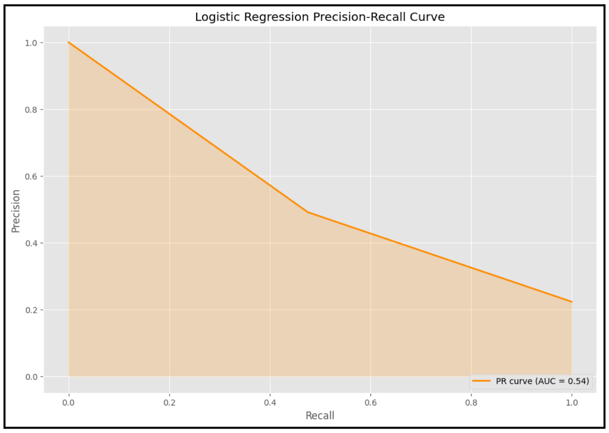
Learning Rate = 0.0008





Learning Rate = 0.001



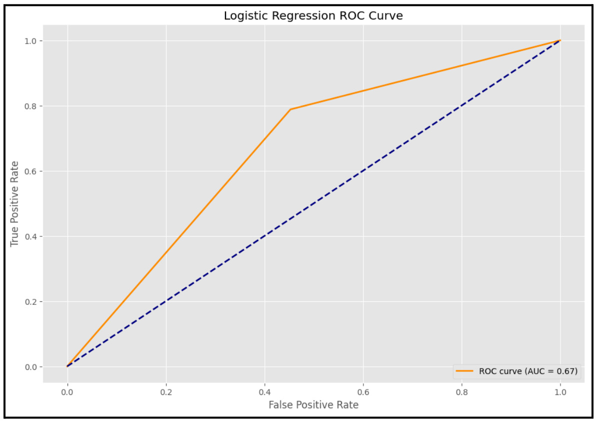


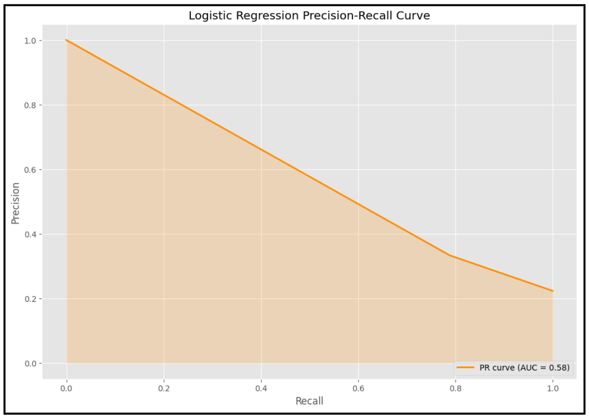
Results:

Learning Rate	Runtime(seconds)	ROC AUC	Precision-Recall curve	F-1 Score
0.0003	79.46	0.65	0.54	0.46
0.0005	80.36	0.65	0.57	0.45
0.0007	78.57	0.62	0.58	0.42
0.0008	79.08	0.67	0.55	0.48
0.001	25.60	0.67	0.54	0.48

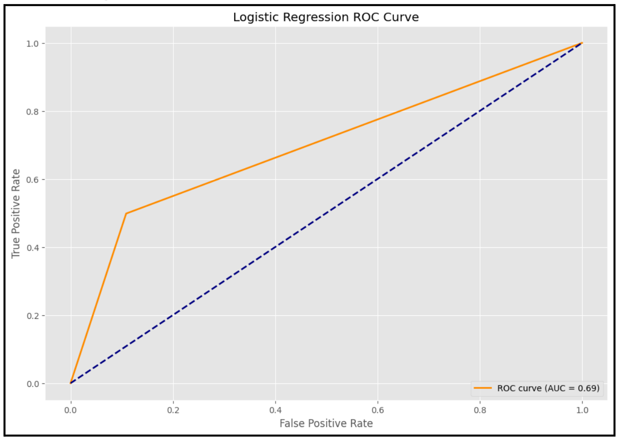
With Feature Importance:

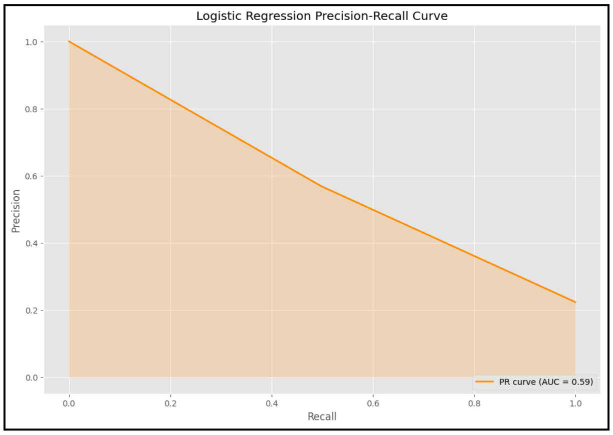
Learning Rate = 0.0003



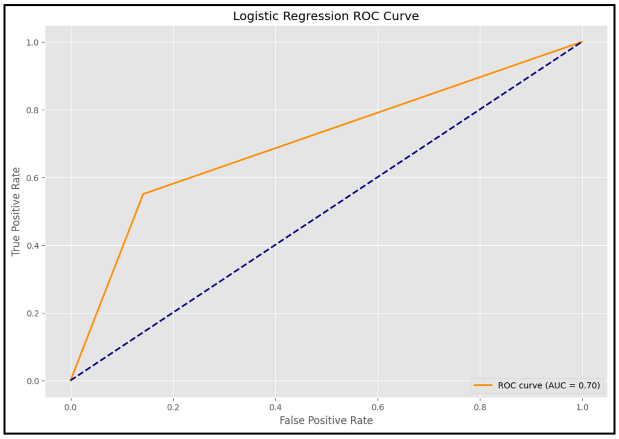


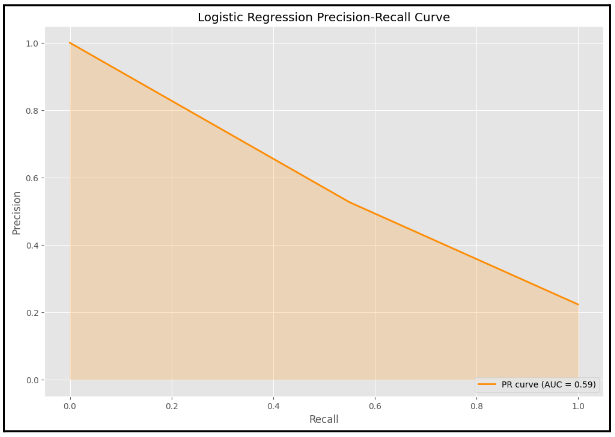
Learning Rate: 0.0005



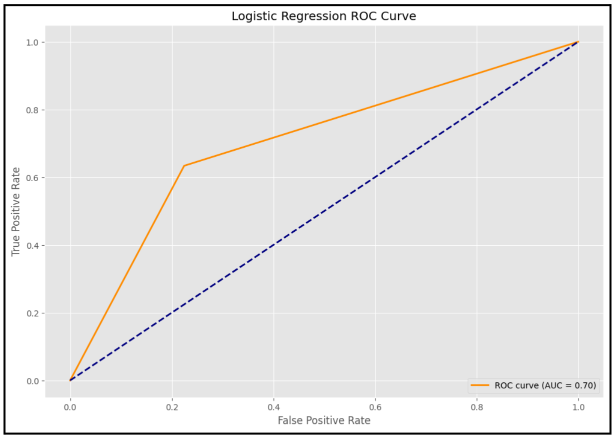


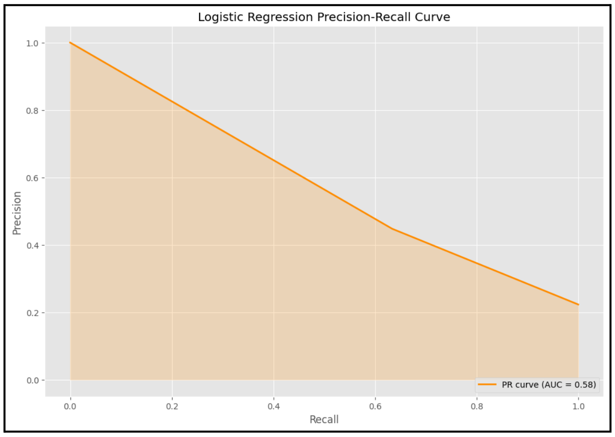
Learning Rate: 0.0007



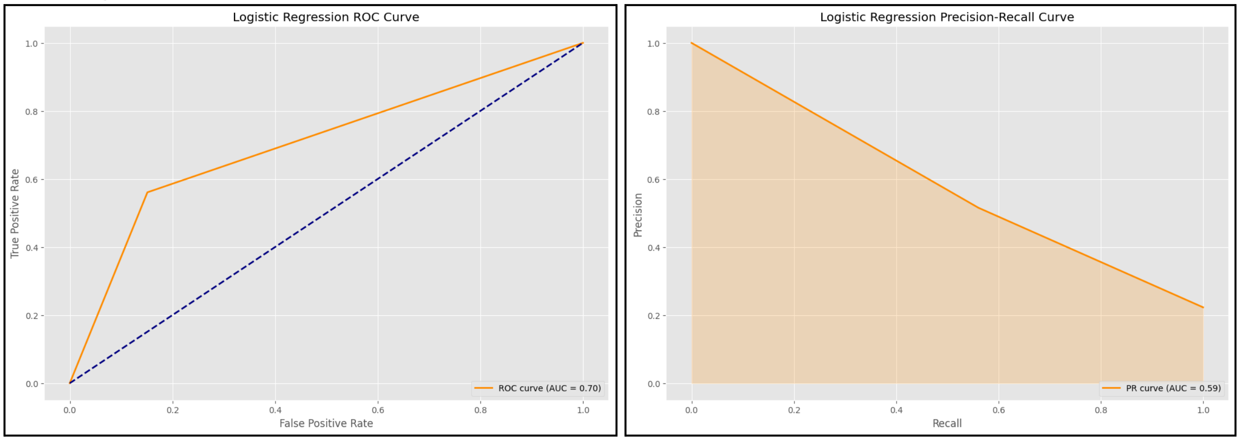


Learning Rate: 0.0008





Learning Rate: 0.001



Results:

Learning Rate	Runtime(seconds)	ROC AUC	Precision-Recall curve	F-1 Score
0.0003	63.02	0.67	0.58	0.46
0.0005	33.63	0.69	0.59	0.53
0.0007	12.35	0.70	0.59	0.53
0.0008	10.63	0.70	0.58	0.52
0.001	11.84	0.70	0.59	0.53

Conclusion:

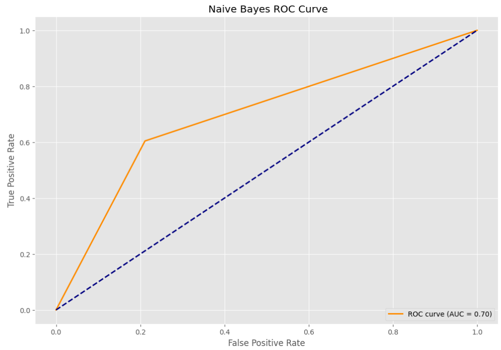
Feature selection has resulted in significant improvements in the logistic regression model's performance, as indicated by the decrease in runtime and the increase in AUC and precision-recall curve values.Without feature selection, the model's performance varied greatly with different learning rates, with the best results achieved at a learning rate of 0.0003 and 0.0008, both with an AUC of 0.80 and a precision-recall curve of 0.87. However, even with the optimal learning rate, the model's runtime was relatively high, taking around 80 seconds to complete.In contrast, with feature selection, the model's performance was consistently good across all learning rates, with the best results achieved at a learning rate of 0.0007 and 0.001, both with an AUC of 0.83 and a precision-recall curve of 0.88. Additionally, the runtime was significantly reduced, with the model taking only around 10 to 12 seconds to complete.Overall, these results suggest that feature selection can greatly improve the performance of logistic regression models, resulting in faster and more accurate predictions. Additionally, the optimal learning rate for the model may differ depending on whether feature selection is used or not, indicating the importance of considering multiple hyperparameters when training machine learning models.

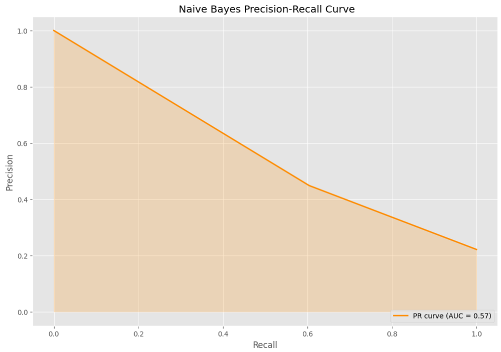
Naïve Bayes:

Naive Bayes is a simple and effective probabilistic algorithm used for classification tasks. It is based on Bayes' theorem, which states that the probability of a hypothesis (in this case, a class label) given the evidence (in this case, a set of features) is proportional to the likelihood of the evidence given the hypothesis, multiplied by the prior probability of the hypothesis.
The "naive" assumption in Naive Bayes is that all the features are independent of each other, given the class label. This simplifies the calculation of the likelihood of the evidence given the hypothesis, making the algorithm computationally efficient and fast.

Results Without Feature Selection:

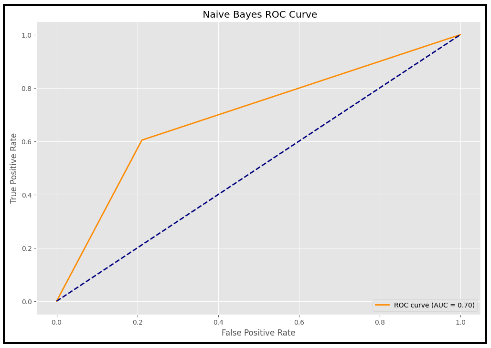
Model Runtime : 10.61
F1 score: 0.5148
Recall score: 0.6043
Precision score: 0.4484
Accuracy score: 0.7480

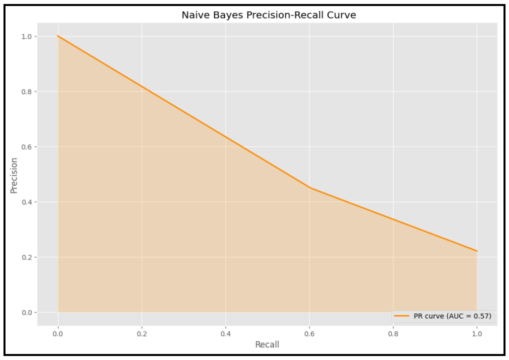




Results With Feature Selection:

Model Runtime : 9.65
F1 score: 0.5148
Recall score: 0.6043
Precision score: 0.4484
Accuracy score: 0.7480





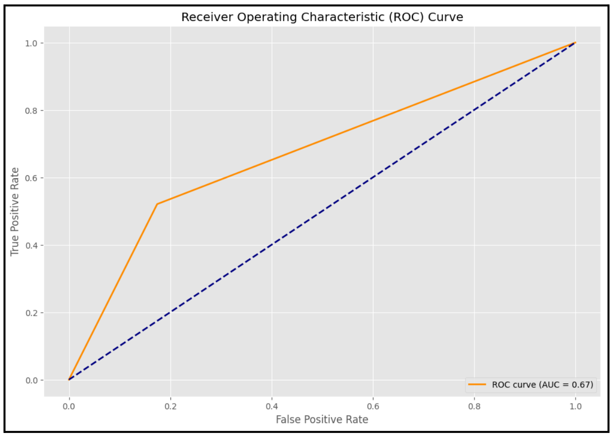
Support Vector Machine: Soft Margin

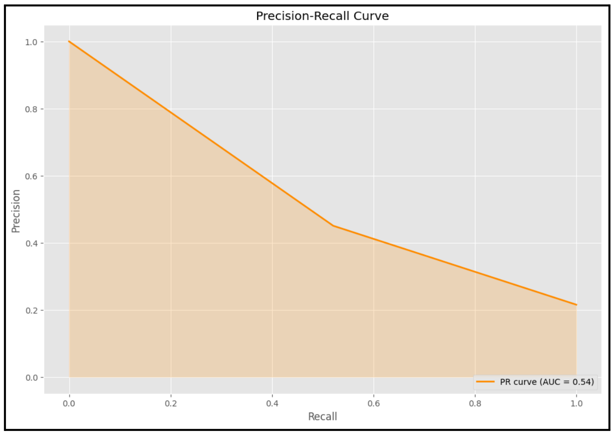
SVM stands for Support Vector Machine, which is a popular and powerful machine learning algorithm used for classification and regression analysis. It is a supervised learning model that works by identifying a hyperplane in a high-dimensional space that best separates different classes of data.

The SVM algorithm seeks to maximize the margin between the decision boundary and the closest data points of each class. The data points closest to the decision boundary are called support vectors, and they are used to define the hyperplane. SVM can also handle non-linearly separable data by using kernel functions that transform the input data into a higher-dimensional space where the data becomes linearly separable.

Results Without Feature Selection:

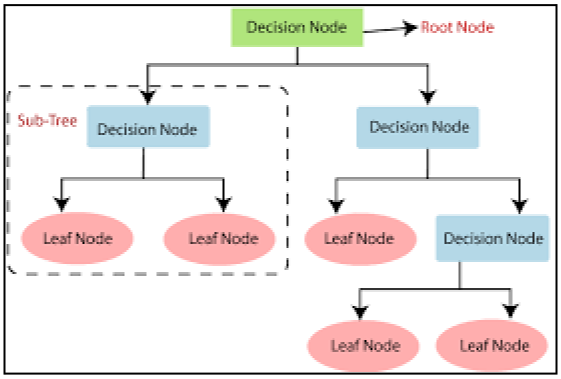
F1 score: 0.4831
Recall score: 0.5208
Precision score: 0.4505
Accuracy score: 0.7601
RunTime : 290.2576



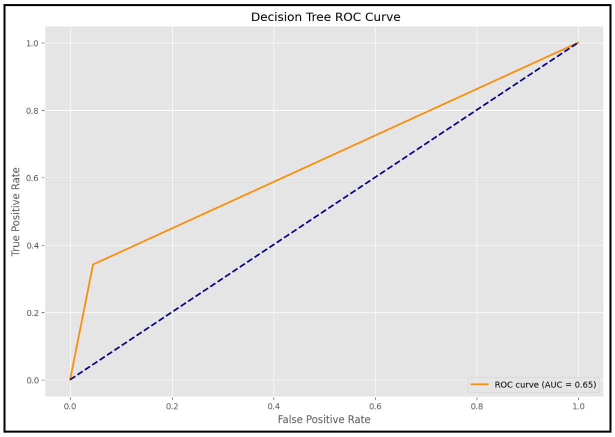


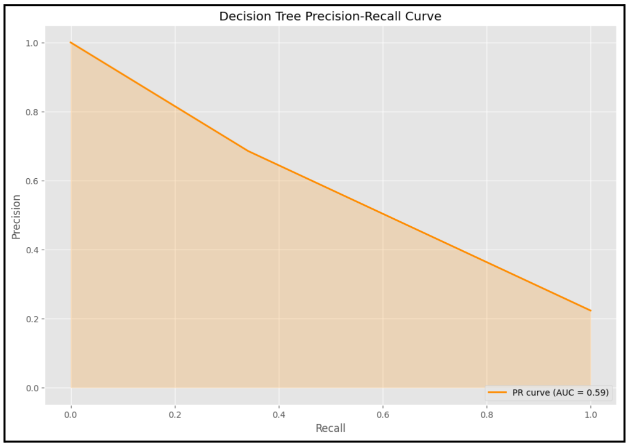
Decision Tree:

The Decision Tree Classifier is a tree-like data-driven or non-parametric approach for formulating predictions. Trees separate predictors in such a way that homogeneity increases with each split, dividing records into sub-groups and offering simple logical principles.



Considered just 5000 datapoints as it was taking too long to process and colab was crashing.



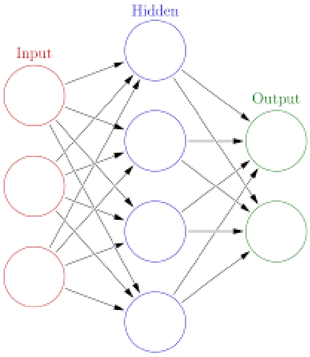


Results:

	Without Feature Selection	With Feature Selection
Accuracy	0.80	0.81
Precision	0.62	0.69
Recall	0.30	0.34
F-1	0.40	0.46
Model Run Time(Secs)	677.9616	581.397

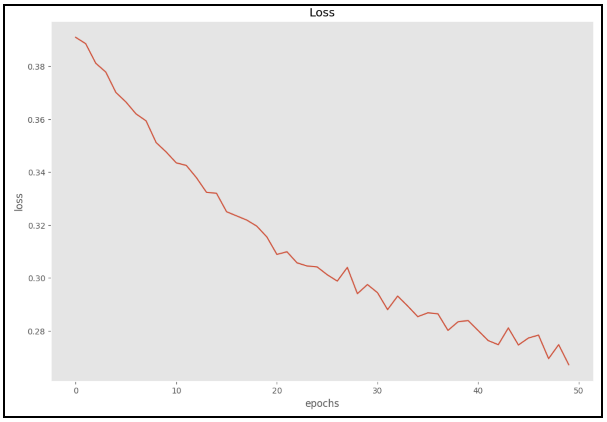
Neural Network:

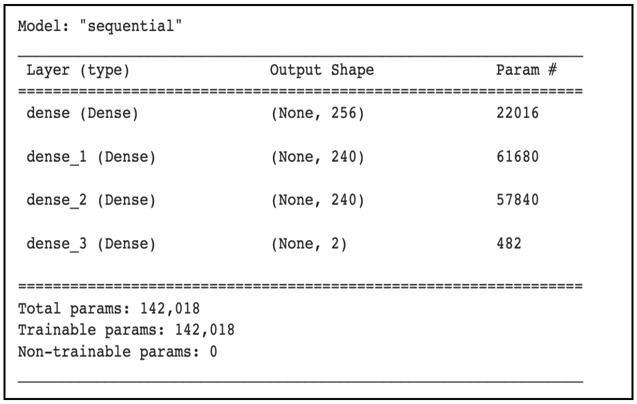
Neural networks typically consist of three layers: input, output, and hidden layers. In a basic neural network, we calculate the result by multiplying inputs with weights, adding bias, and applying an activation function. This output is then passed to the next layer and the process is repeated until the final layer is reached. Hidden layers are included to optimize the input weight values and minimize errors.



FINAL RESULTS

In order to optimize the model's hyper-parameters such as the number of units in each layer, activation functions, and learning rate for the Adam optimizer, Keras tuner was utilized. The graph indicates that as the number of epochs increases, the loss decreases. At around the 27th epoch out of 50, the loss significantly decreased from 0.38 to 0.3. The loss function employed was Sparsecategoricalcrossentropy().





```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 256)               22016

 dense_1 (Dense)             (None, 240)               61680

 dense_2 (Dense)             (None, 240)               57840

 dense_3 (Dense)             (None, 2)                 482

=================================================================
Total params: 142,018
Trainable params: 142,018
Non-trainable params: 0
_________________________________________________________________
```

Model Runtime : 262.16

Recall : 0.341

Precision : 0.685

F1 Score : 0.455

Accuracy : 0.8181

Conclusion:

Following are the best results for each type of algorithm:

Best Models	**F-1**	**Recall**	**Runtime(secs)**
Logistic Regression	0.53721	0.5608	11.84
Naive Bayes	0.51	0.60	9.65
SVM	0.48	0.52	290.257
Decision Tree	0.46	0.34	581.397
Neural Networks	0.455	0.341	262.16

Based on the given information, the best performing model in terms of F-1 score and recall is Logistic Regression. However, it has a higher runtime compared to Naive Bayes, which also has a decent performance in terms of recall. SVM, Decision Tree, and Neural Networks have lower performance in terms of F-1 score and recall, and longer runtimes than the other two models.

Therefore, if runtime is not a critical factor, then Logistic Regression could be the preferred model. On the other hand, if a faster runtime is needed, Naive Bayes could be a better choice. However, it is important to note that the evaluation of a model's performance should consider other factors such as data characteristics, model complexity, and interpretability, among others.