

Credit Card Approval Prediction System

Project Report

Group 14

Shreyans Thesia

Aryak Bodkhe

609-791-2693(Tel of Shreyans Thesia)

857-991-7323 (Tel of Aryak Bodkhe)

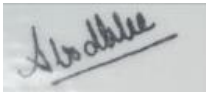
thesia.s@northeastern.edu

bodkhe.a@northeastern.edu

Percentage of Effort Contributed by Student 1: 50%

Percentage of Effort Contributed by Student 2: 50%

Signature of Student 1:  _____

Signature of Student 2:  _____

Submission Date: 04/21/2023

Table of Contents

Problem Setting	4
Problem Definition	4
Data Source	4
Data Description	4
Table 1. Data Columns and Information	4
Data Visualization	5
Fig 1. Pie Chart of Bivariate Analysis	5
Fig 2. Bar Chart of Average Income	5
Fig 3. Donut chart comparing customers	6
Fig 4. Bubble Chart comparing Income	6
Fig 5. Correlation Heatmap	7
Data Preprocessing	8
1. Checking for Null Values.....	8
Fig 5. Count of Null Values	8
Fig 6. Heatmap of Null Values	8
2. Outlier Detection and Removal.....	9
Fig 7. Data with Outliers	9
Fig 8. Data without Outliers	9
3. Statistical Analysis	9
Fig 9. Stat. Snapshot	10
4. Correlation Analysis.....	10
Fig 10. Correlation Heatmap	10
5. Encoding Categorical Variables.....	11
6. Balancing the Dataset	11
7. Feature Selection	11
Data Preparation	12
Data Partitioning	12

Data Mining Models	12
1. Logistic Regression	12
2. KNN	13
3. Decision Tree	14
4. XG Boost.....	14
5. Random Forest	15
6. SVM.....	15
Performance Evaluation	17
1. Logistic Regression	17
Fig 11. Linear Regression ROC Curve and Classification Report	17
2. Decision Tree	18
Fig 12. Decision Tree ROC Curve and Classification Report	18
3. Random Forest	19
Fig 13. Random Forest ROC Curve and Classification Report	19
4. SVM	20
Fig 14.SVM ROC Curve and Classification Report	20
5. KNN	21
Fig 15.KNN ROC Curve and Classification Report	21
6. XG Boost.....	22
Fig 16. XG Boost ROC Curve and Classification Report	22
Summary of all Models	23
Table 2. Model classification Report	23
Result	24
Impact of project Outcomes	21
References	24

Problem Setting:

Credit cards applications are increasing everyday as people come to know the advantages of having a credit card. But for doing so each time there is a hard enquiry on your credit, and it affects your score negatively. Therefore, a prediction system would be beneficial to find out if you will be approved or not without affecting your credit score. By doing so, it will be beneficial for both you as well as the issuer as less time and resources would be affected.

Problem Definition:

A credit card approval prediction problem entails training a model on historical credit card application data to predict whether a new applicant will be approved for a credit card. The model will take a set of input features like the applicant's income, credit score, and employment status to predict whether the applicant will be approved or not using various supervised machine learning algorithms. The goal is to accurately predict which applicants are likely to be approved or not and whether the applicant should move forward with his application to the issuer.

Data Source:

This project's dataset comes from Kaggle. This project's dataset URL:

<https://www.kaggle.com/rikdifos/credit-card-approval-prediction>

Data Description:

The dataset deals with 400k+ records and 18 attributes. This data deals in predicting if an applicant is 'good' or 'bad' client, different from other tasks, the definition of 'good' or 'bad' is not given. The data attributes are classified into following categories:

Table 1. Data Columns and Information

ID	Client number	NAME_FAMILY_STATUS	Marital status
CODE_GENDER	Gender	NAME_HOUSING_TYPE	Way of living
FLAG_OWN_CAR	Is there a car	DAYS_BIRTH	Birthday
FLAG_OWN_REALTY	Is there a property	DAYS_EMPLOYED	Start date of employment
CNT_CHILDREN	Number of children	FLAG_MOBILE	Is there a mobile phone
AMT_INCOME_TOTAL	Annual income	FLAG_WORK_PHONE	Is there a work phone
NAME_INCOME_TYPE	Income category	FLAG_PHONE	Is there a phone
NAME_EDUCATION_TYPE	Education level	FLAG_EMAIL	Is there an email
		OCCUPATION_TYPE	Occupation

Data Visualization:

1. Bivariate analysis using Pie Chart.

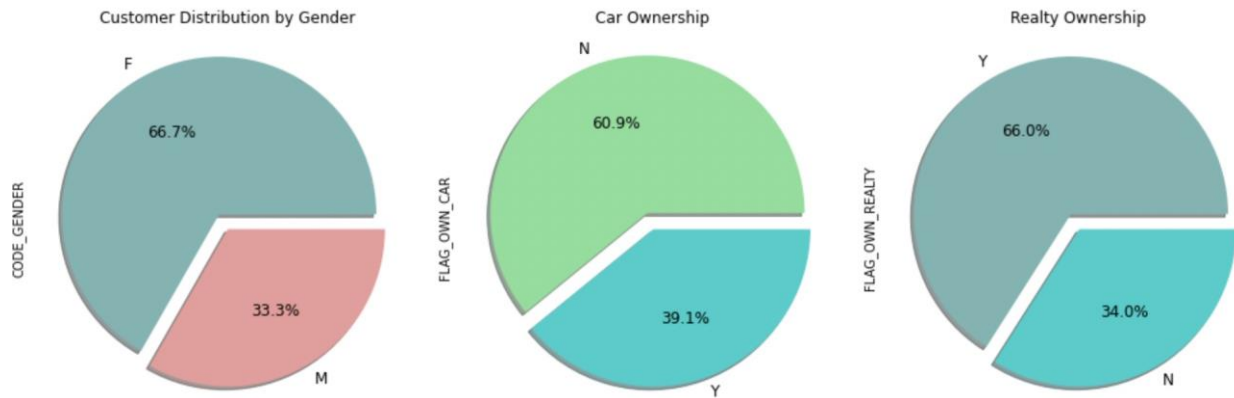


Fig 1. Pie Chart of Bivariate Analysis

Here we have visualized the categorical columns of the dataset and here we can see that

- The applicants have a gender distribution of 66.7% female and 33.3% male.
- The percentage of applicants who have their vehicle is 39.1%
- The percentage of applicants own realty vehicle is 66.0%

2. Average income by education level using horizontal bar chart

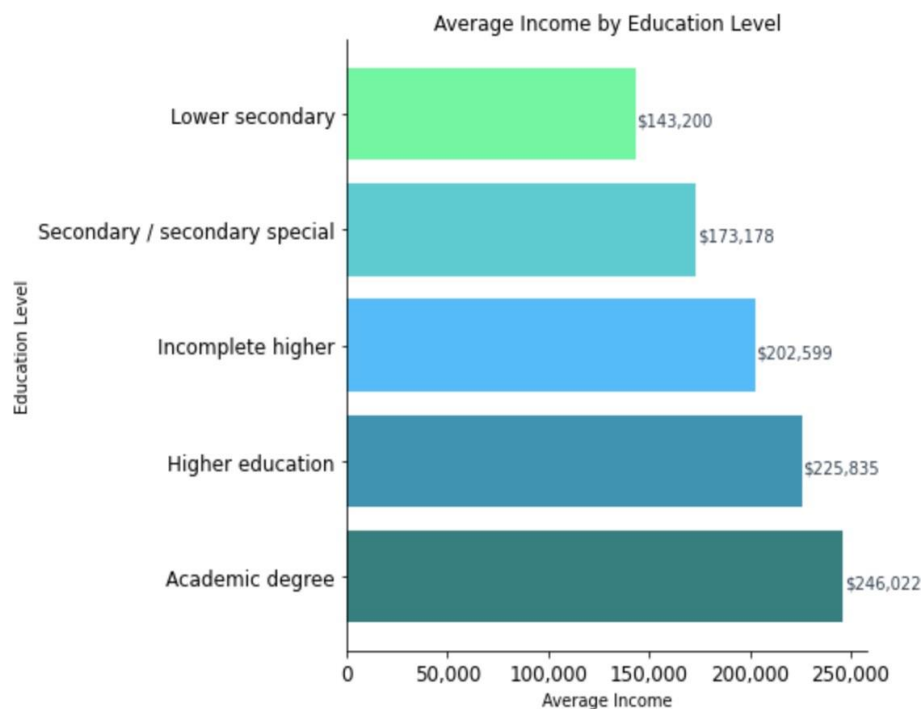


Fig 2. Bar Chart of Average Income

Here we tried visualizing, by taking two variables at a time and from this plot we concluded that the average income increases with the education level.

3. Donut chart comparing the income type of different customer working class.

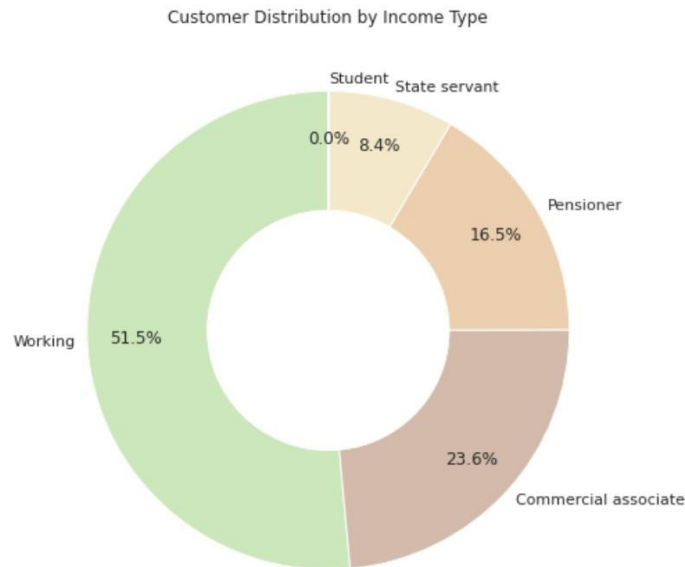


Fig 3. Donut chart comparing customer distribution.

Here, we created a donut chart to show the customer distribution of the bank by their income type. From this chart we can see that working class people are more likely to apply for credit cards and are more likely to get approved. There are also many company based associates who applied for credit cards and some retired folks also.

4. Bubble chart for income range of people

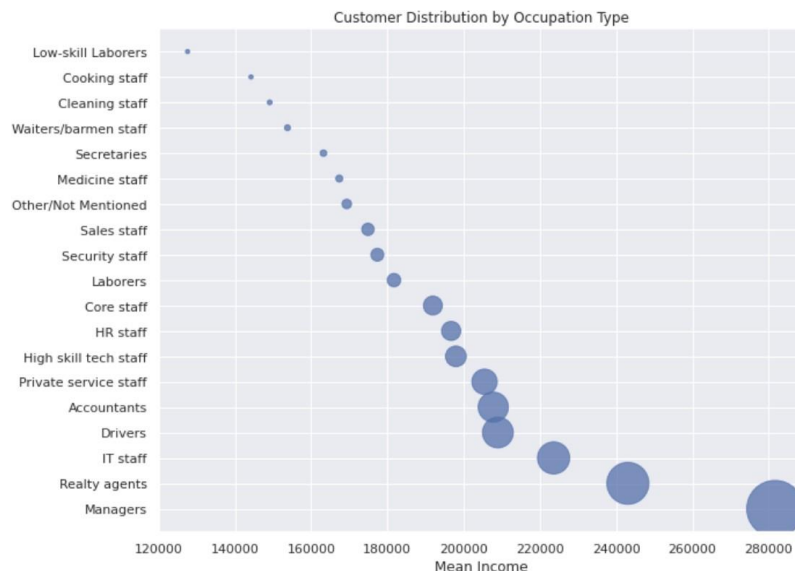


Fig 4. Bubble Chart comparing Income

From the above bubble chart we can infer that managers have high income and the people who do regular jobs such as company staff, drivers and they also have high probability of getting approved for credit cards.

5. Pivot chart matrix to analyze the percentage of defaulters.

We tried analyzing the percentage/count of bad customer/defaulters. By analyzing the entire dataset and we came up with the following strategy: a person is considered a bank defaulter if he has a payment outstanding for more than 60 days. As a result, all customers with STATUS ≥ 2 are considered defaulters or bad customers.

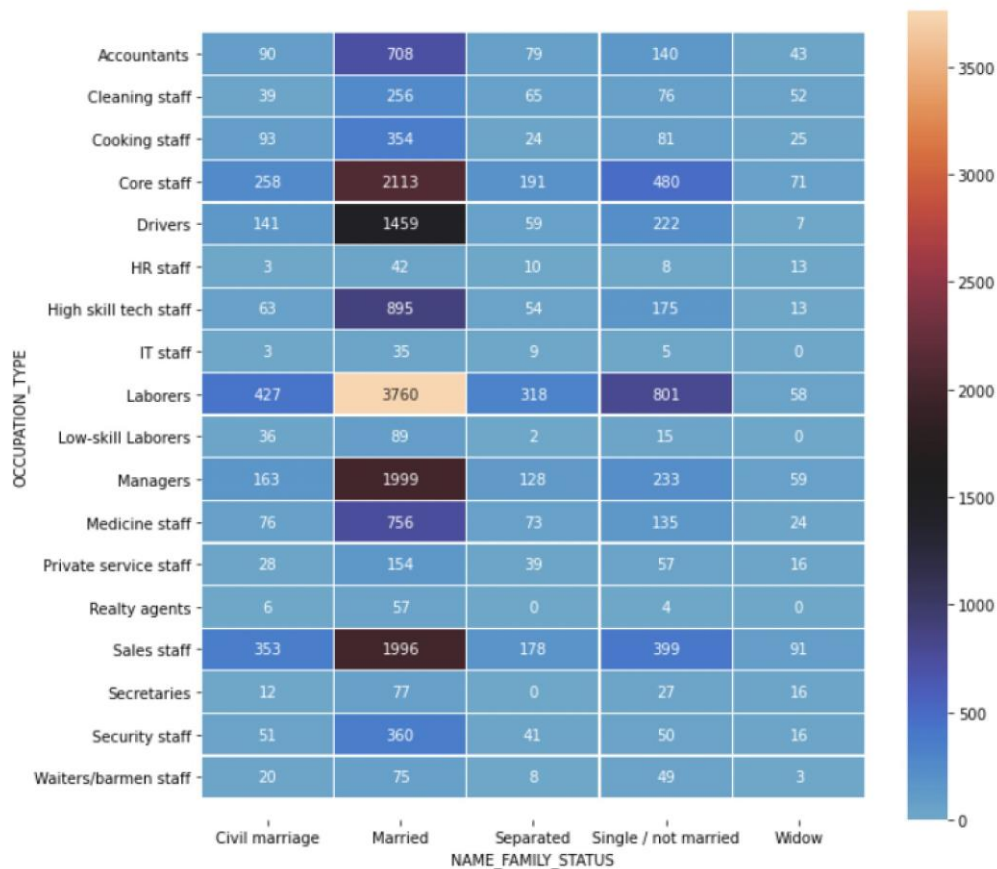


Fig 5. Correlation Heatmap

The above visualization is a pivot table which has been plotted in the form of a heatmap. The above visualization takes into account the two columns here OCCUPATION_TYPE and NAME_FAMILY_STATUS. We conclude that Laborers who are married have highest number of defaulters here at the count of 3760.

Data Preprocessing:

1. Checking for Null Values

- We found out that OCCUPATION_TYPE column has null values.
- The count of Null values to be was 240048.

```
ID 0
MONTHS_BALANCE 0
STATUS 0
CODE_GENDER 0
FLAG_OWN_CAR 0
FLAG_OWN_REALTY 0
CNT_CHILDREN 0
AMT_INCOME_TOTAL 0
NAME_INCOME_TYPE 0
NAME_EDUCATION_TYPE 0
NAME_FAMILY_STATUS 0
NAME_HOUSING_TYPE 0
DAYS_BIRTH 0
DAYS_EMPLOYED 0
FLAG_MOBIL 0
FLAG_WORK_PHONE 0
FLAG_PHONE 0
FLAG_EMAIL 0
OCCUPATION_TYPE 240048
CNT_FAM_MEMBERS 0
dtype: int64
```

Fig 5. Count of Null Values

- We even tried visualizing the missing values of the OCCUPATION_TYPE column using a matrix.

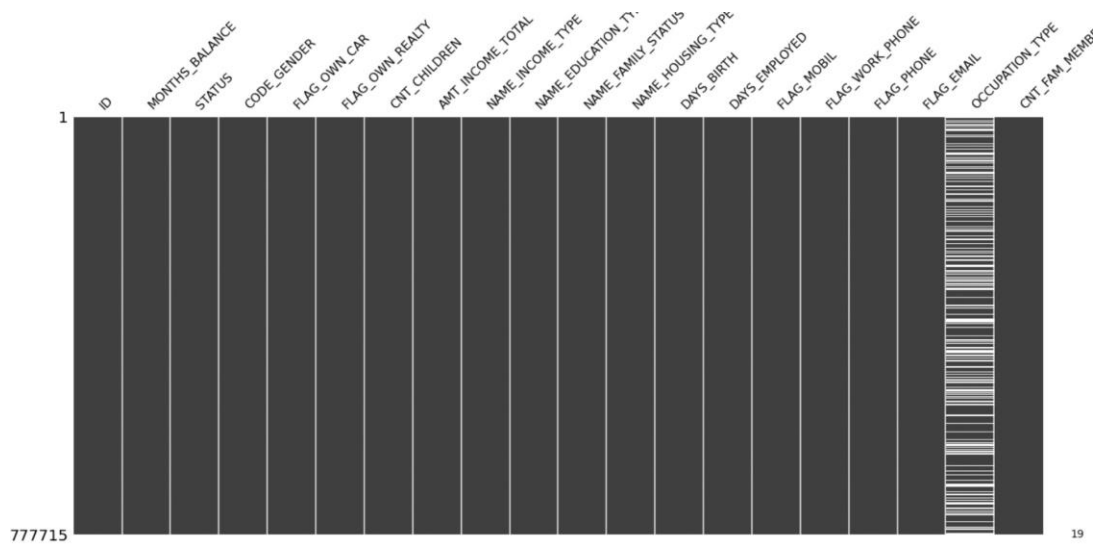


Fig 6. Heatmap of Null Values

2. Outlier Detection and Removal:

- We checked for Outliers in this dataset and found that the CNT_CHILDREN, DAYS_EMPLOYED and FLAG_PHONE had outliers in them.
- We even visualized the outliers and did the comparison of the same using a scatter plot before and after treating the outliers.

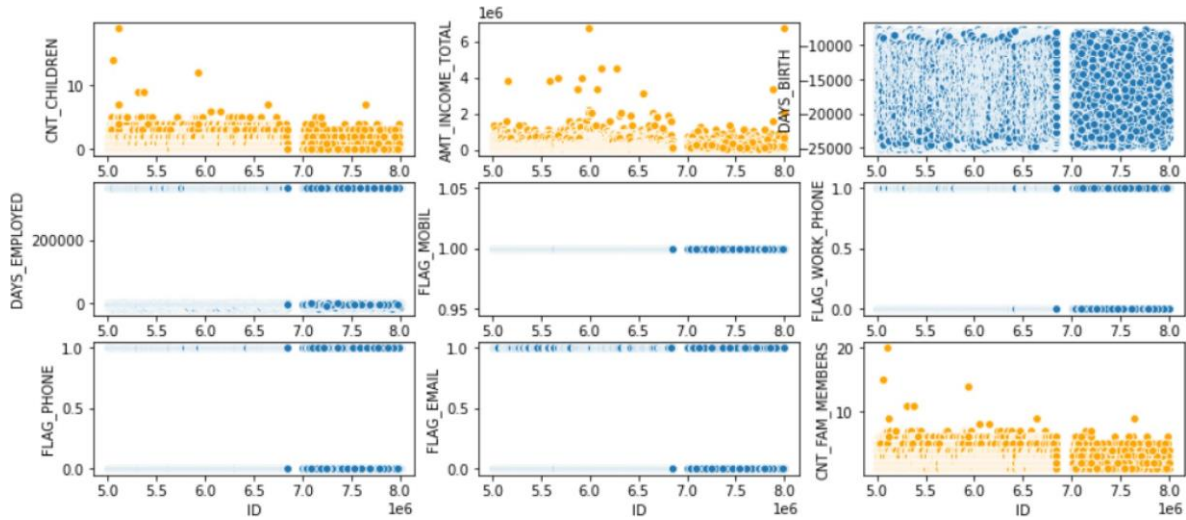


Fig 7. Data with Outliers

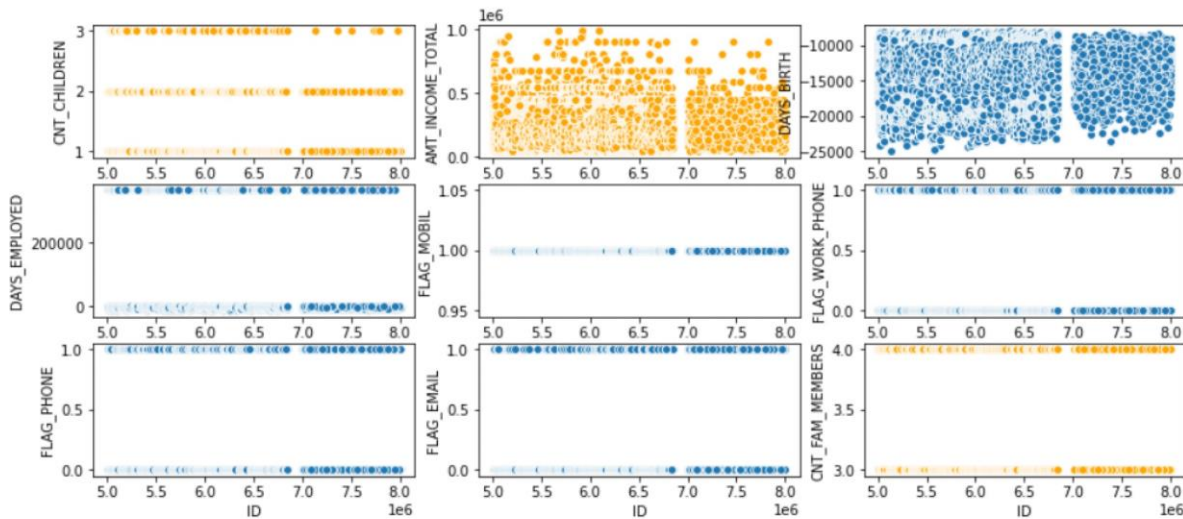


Fig 8. Data without Outliers

3. Statistical Analysis:

We performed statistical analysis on the dataset and some of our findings from the dataset were as follows:

- Average monthly balance of the applicants is -19.37.
- Maximum no. of Family members for an applicant is as high as 20.
- FLAG_MOBILE because it has one value

	ID	MONTHS_BALANCE	CNT_CHILDREN	AMT_INCOME_TOTAL	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EMAIL	CNT_FAM_MEMBERS
count	7.777150e+05	777715.000000	777715.000000	7.777150e+05	777715.000000	777715.000000	777715.0	777715.000000	777715.000000	777715.000000	777715.000000
mean	5.078743e+06	-19.373564	0.428082	1.885348e+05	-16124.937046	57775.825016	1.0	0.231818	0.300965	0.091675	2.208837
std	4.180442e+04	14.082208	0.745755	1.016225e+05	4104.304018	136471.735391	0.0	0.421993	0.458678	0.288567	0.907380
min	5.008904e+06	-60.000000	0.000000	2.700000e+04	-25152.000000	-15713.000000	1.0	0.000000	0.000000	0.000000	1.000000
25%	5.044568e+06	-29.000000	0.000000	1.215000e+05	-19453.000000	-3292.000000	1.0	0.000000	0.000000	0.000000	2.000000
50%	5.069530e+06	-17.000000	0.000000	1.620000e+05	-15760.000000	-1682.000000	1.0	0.000000	0.000000	0.000000	2.000000
75%	5.115551e+06	-8.000000	1.000000	2.250000e+05	-12716.000000	-431.000000	1.0	0.000000	1.000000	0.000000	3.000000
max	5.150487e+06	0.000000	19.000000	1.575000e+06	-7489.000000	365243.000000	1.0	1.000000	1.000000	1.000000	20.000000

Fig 9. Stat. Snapshot

4. Correlation Analysis:

Looking at the correlation analysis heatmap, we see how strongly the variables are related, i.e, if they have a positive or negative relationship. From the visualization, we can conclude that the count of family members and count of children is highly positively correlated. There is a high positive correlation between the gender and own car column. Now the other thing we can conclude here is that even though there is not high correlation between the outcome variable i.e, Status with any one single variable but the combination of the other dependent variables can help us predict the status.

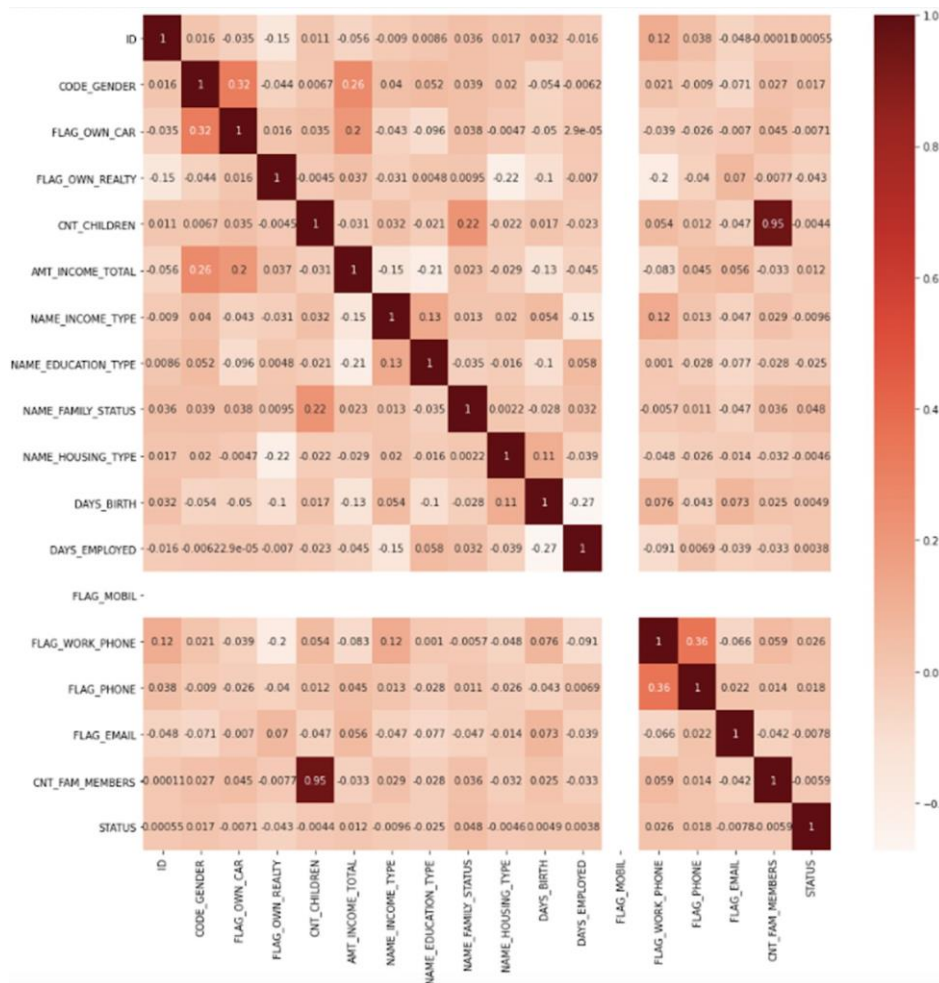


Fig 10. Correlation Heatmap

5. Encoded Categorical Variables:

Here, we encode categorical variables to numerical values. For this we created a custom function that would add a dummy variable as shown in the output below.

```
CODE_GENDER --> {'F': 0, 'M': 1}
FLAG_OWN_CAR --> {'N': 0, 'Y': 1}
FLAG_OWN_REALTY --> {'N': 0, 'Y': 1}
NAME_INCOME_TYPE --> {'Commercial associate': 0, 'Pensioner': 1, 'State servant': 2, 'Student': 3, 'Working': 4}
NAME_FAMILY_STATUS --> {'Civil marriage': 0, 'Married': 1, 'Separated': 2, 'Single / not married': 3, 'Widow': 4}
OCCUPATION_TYPE --> {'Accountants': 0, 'Cleaning staff': 1, 'Cooking staff': 2, 'Core staff': 3, 'Drivers': 4, 'HR staff': 5, 'High skill tech staff': 6, 'IT staff': 7}
NAME_HOUSING_TYPE --> {'Co-op apartment': 0, 'House / apartment': 1, 'Municipal apartment': 2, 'Office apartment': 3, 'Rented apartment': 4, 'With parents': 5}
NAME_EDUCATION_TYPE --> {'Academic degree': 0, 'Higher education': 1, 'Incomplete higher': 2, 'Lower secondary': 3, 'Secondary / secondary special': 4}
```

Fig 10. Encoded Categorical Features

In a machine learning model, we transformed categorical information into numerical characteristics. This is crucial because machine learning algorithms need numerical input, and transforming category information into numerical features facilitates data analysis and precise prediction.

A numerical value has been assigned to each classified feature. As an illustration, the feature 'CODE_GENDER' has been mapped to 'F': 0, 'M': 1, where 'F' and 'M' stand for 0 and 1, respectively. Each characteristic has also been assigned a numerical value that corresponds to it, making it simpler to enter the data into a machine learning model.

The fundamental meaning of the characteristic is preserved during the mapping of categorical data to numerical features.

6. Balancing the Dataset

Too few examples of the minority class exist for a model to successfully learn the decision boundary, which is a drawback of imbalanced categorization.

Oversampling the instances from the minority class is one technique to address this issue. Before fitting a model, this can be accomplished by simply copying samples from the minority class in the training dataset. Although the class distribution can be balanced, the model does not receive any new data as a result. Synthesizing fresh minority-class instances is a better alternative to copying existing minority-class examples.

SMOTE picks examples in the feature space that are near together, draws a line connecting the examples, and then creates a new sample at a point.

7. Feature Selection

A feature selection technique called recursive feature elimination (RFE) eliminates the weakest feature (or features) from a model until the required number of features is reached. RFE aims to remove any potential dependencies and collinearity from the model by recursively deleting a limited number of features every loop according to the model's coefficient_ or feature importance attributes.

Data Preparation

Another benefit of standardization is that it can help to improve the efficiency and accuracy of certain machine learning algorithms. For example, algorithms that rely on distance measures between data points, such as K-nearest neighbors or clustering algorithms, can be sensitive to differences in scale between variables. Standardizing the data can help to address this issue and improve the accuracy and stability of these algorithms.

It is also worth noting that standardization should be performed on the training set and then applied to the test set using the same scaling parameters. This ensures that the test data is scaled in the same way as the training data and prevents any information leakage between the two sets.

Overall, scaling techniques such as standardization are an important step in data preprocessing and can have a significant impact on the results of data analysis and machine learning models. It is important to carefully consider the nature of the data and the goals of the analysis when choosing a scaling technique, and to ensure that the scaling is applied consistently across the entire dataset.

To address the scaling difference between the predictor variables, the dataset was standardized using the `StandardScaler()` library from the scikit-learn package. This was done to ensure that all variables were equally important in terms of their variability, regardless of their initial scale.

Data Partitioning

There are 19 variables in this dataset collectively represented as 'X' that are used to predict the target variable 'STATUS', represented as 'y'. The sklearn.model selection libraries `train_test_split` was used to divide the data into training and testing sets in a 70:30 ratio. As a result, 70% of the dataset was used to train classification models, with the remaining 30% used to evaluate the performance of these models. The training set, X train, had 544,400 records with 19 variables, and the test set, X test, had 233,315 records with 19 variables. The target variable was also divided into training and testing sets, with 544,400 records in y train and 233,315 records in y test.

Data Mining models/methods:

There were data mining classification models that were built using the training data mentioned above.

1. Logistic regression

Logistic regression is a suitable choice for a predictive model of credit card approval system based on different parameters because it allows you to predict the probability of an event occurring based on multiple predictor variables. In this case, the event could be the approval or rejection of a credit card application, and the predictor variables could be whether the applicant has a car or not, has a home or not, what their income is, and so on. Logistic regression is particularly useful when the outcome is binary, meaning it can only take on two possible values (such as approved or rejected).

Advantages of using logistic regression for a credit card approval model include:

- Interpretable results: Logistic regression provides easy-to-interpret results in terms of odds ratios, which can help you understand the relative importance of different predictor variables.
- Handles categorical variables: Logistic regression can handle both categorical and continuous predictor variables, making it suitable for credit card approval models that include variables like car ownership and home ownership.

- Robust to outliers: Logistic regression is less sensitive to outliers than some other modelling techniques, making it a good choice for data with potential outliers.

Disadvantages of using logistic regression for a credit card approval model include:

- Assumes linearity: Logistic regression assumes that the relationship between predictor variables and the log odds of the outcome is linear. This may not be true in all cases, which can lead to inaccurate predictions.
- Assumes independence: Logistic regression assumes that the predictor variables are independent of each other. Some predictor variables may be correlated, which can lead to multicollinearity and inaccurate predictions.
- Limited flexibility: Logistic regression is a relatively simple modeling technique and may not capture the full complexity of credit card approval decisions, which can involve a wide range of factors and decision-making processes.

2. K Nearest Neighbors

K nearest neighbor (KNN) is a non-parametric classification algorithm that is also commonly used for predictive modelling in credit card approval systems. The basic idea behind KNN is to classify a new observation based on the class of its k nearest neighbors in the feature space.

Advantages of KNN for a credit card approval system:

- Non-parametric: KNN is a non-parametric method that does not make any assumptions about the underlying distribution of the data, which can be beneficial in situations where the data is complex or difficult to model.
- Can handle noisy data: KNN is robust to noisy data, which can be useful in a credit card approval system where there may be missing or inaccurate data.
- Flexibility: KNN is a flexible algorithm that can handle different types of data and can be adapted to different credit card approval scenarios.

Disadvantages of KNN for a credit card approval system:

- Sensitivity to feature scaling: KNN is sensitive to the scaling of the features, which can affect the distance calculations between observations. This can be a problem in a credit card approval system where some features may have very different scales.
- Slow for large datasets: KNN can be computationally expensive for large datasets, which can be a disadvantage in a credit card approval system with a large number of applicants.
- Requires careful tuning of hyperparameters: KNN's performance depends on the choice of hyperparameters, such as the number of neighbours (k) and the distance metric used. Tuning these hyperparameters can be time-consuming and require domain expertise.

Overall, KNN can be a good choice for credit card approval prediction models when the relationship between the predictor variables and the outcome is complex or unknown, there is a high degree of local variation, and/or real-time predictions are required. However, the computational intensity of the algorithm, the need for parameter selection, and potential issues with imbalanced data should also be taken into consideration.

3. Decision Tree Classification

Decision tree classification is another popular algorithm. Decision trees work by recursively partitioning the feature space into regions, with each region corresponding to a different class label.

Advantages of using decision tree classification for a credit card approval model include:

- Easy to understand: Decision trees are easy to understand and visualize, as they can be represented as a tree structure with nodes corresponding to decisions and branches corresponding to possible outcomes.
- Can handle both categorical and numerical features: Decision trees can handle both categorical and numerical features, making them a good choice for credit card approval systems that include a mix of different types of features.
- Can capture non-linear relationships: Decision trees can capture non-linear relationships between the features and the target variable, making them more flexible than linear models like logistic regression.

Disadvantages of using decision tree classification for a credit card approval model include:

- Prone to overfitting: Decision trees can be prone to overfitting, especially when the tree is deep and complex. This can lead to poor generalization performance on new data.
- Sensitive to small changes in the data: Decision trees can be sensitive to small changes in the data, which can lead to different tree structures and different predictions.
- Can be biased towards features with many levels: Decision trees can be biased towards features with many levels or categories, as they tend to create more splits for such features. This can lead to over-representation of some features and under-representation of others.

4. XGBoost

XGBoost (Extreme Gradient Boosting) is a popular algorithm that uses gradient boosting to build an ensemble of decision trees for classification tasks. It is often used in predictive modelling due to its high accuracy and speed.

Advantages of using XGBoost for a credit card approval model include:

- High accuracy: XGBoost has been shown to achieve state-of-the-art performance on many classification tasks, including credit card approval systems.
- Fast and scalable: XGBoost is designed to be fast and scalable, making it suitable for large datasets and real-time applications.
- Regularization techniques: XGBoost includes a variety of regularization techniques, such as L1 and L2 regularization and dropout, that can help prevent overfitting and improve generalization performance.

Disadvantages of using XGBoost for a credit card approval model include:

- Requires careful tuning of hyperparameters: XGBoost has many hyperparameters that need to be tuned carefully to achieve optimal performance. This can be time-consuming and requires some expertise.
- Can be sensitive to outliers: XGBoost can be sensitive to outliers, especially when using the default loss function (binary cross-entropy). It may be necessary to use a different loss function or pre-processing techniques to deal with outliers.

- Can be difficult to interpret: XGBoost models can be difficult to interpret, especially when they are deep and complex. This can make it challenging to understand how the model is making its predictions and identify areas for improvement.

5. Random Forest

Random forest is another popular algorithm that is commonly used in predictive modelling for credit card approval systems. Random forest is an ensemble learning method that combines multiple decision trees to improve the accuracy and robustness of the model.

Advantages of using random forest for a credit card approval model include:

- High accuracy: Random Forest has been shown to achieve high accuracy on many classification tasks, including credit card approval systems.
- Robust to outliers and noise: Random Forest is robust to outliers and noise, as it averages the predictions of multiple decision trees.
- Feature importance: Random Forest can provide information on the importance of each feature in the classification task, making it easier to understand the factors that are driving the model's predictions.

Disadvantages of using random forest for a credit card approval model include:

- Difficult to interpret: Like other ensemble methods, random forest can be difficult to interpret, especially when the model is complex and includes many decision trees.
- Slow training time: Random Forest can be slower to train than some other algorithms, especially when the number of decision trees and the depth of each tree is large.
- Overfitting: Random Forest can overfit the training data if the number of trees is too large or if the trees are too deep. Regularization techniques, such as pruning or limiting the depth of each tree, can help prevent overfitting.

6. Support Vector Machine (SVM)

Support Vector Machines (SVM) are a popular algorithm. SVM is a linear classifier that tries to find the hyperplane that maximally separates the data points into different classes.

Advantages of using SVM for a credit card approval model include:

- High accuracy: SVM has been shown to achieve high accuracy on many classification tasks, including credit card approval systems.
- Robust to overfitting: SVM can be robust to overfitting, especially when using a kernel trick that maps the data to a higher-dimensional feature space.
- Can handle both linear and non-linear data: SVM can handle both linearly separable and non-linearly separable data using kernel functions.

Disadvantages of using SVM for a credit card approval model include:

- Computationally intensive: SVM can be computationally intensive, especially when using a kernel function or when the dataset is large.
- Sensitivity to hyperparameters: SVM performance can be sensitive to the choice of hyperparameters, such as the regularization parameter or the kernel function. Careful tuning is needed to achieve optimal performance.
- Can be difficult to interpret: SVM can be difficult to interpret, especially when using a non-linear kernel function that maps the data to a higher-dimensional feature space. This can make it challenging to understand how the model is making its predictions and identify areas for improvement.

Performance Evaluation:

1. Logistic Regression

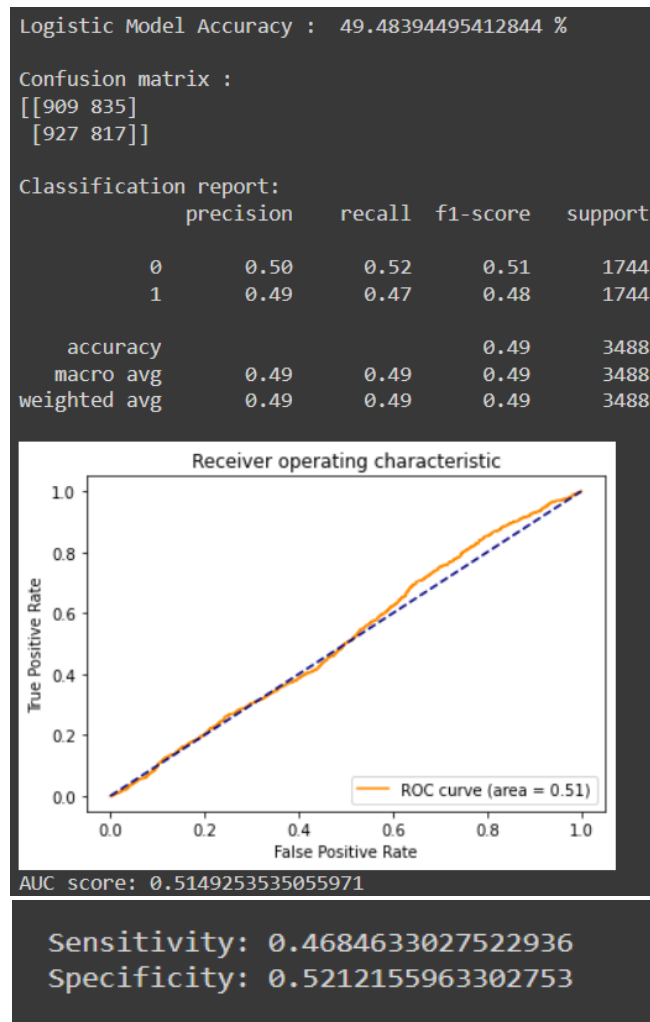


Fig 11. Linear Regression ROC Curve and Classification Report

Logistic model accuracy is 49.48%, correctly predicting target variable for about half of the samples. Confusion matrix shows true positives, false positives, true negatives, and false negatives: 909 true positives, 835 false positives, 927 false negatives, and 817 true negatives. Classification report displays precision, recall, and F1-score for each class. Precision, recall, and F1-score are slightly higher for class 0 than class 1. Weighted average of precision, recall, and F1-score is 0.49 for both classes.

The ROC/AUC score is 0.51, which indicates that the model's ability to distinguish between positive and negative classes is not much better than random guessing.

In summary, the logistic model has poor accuracy, precision, recall, and f1-score, and its ability to differentiate between positive and negative classes is only slightly better than random guessing. The model may require further tuning or feature engineering to improve its performance.

2. Decision Tree

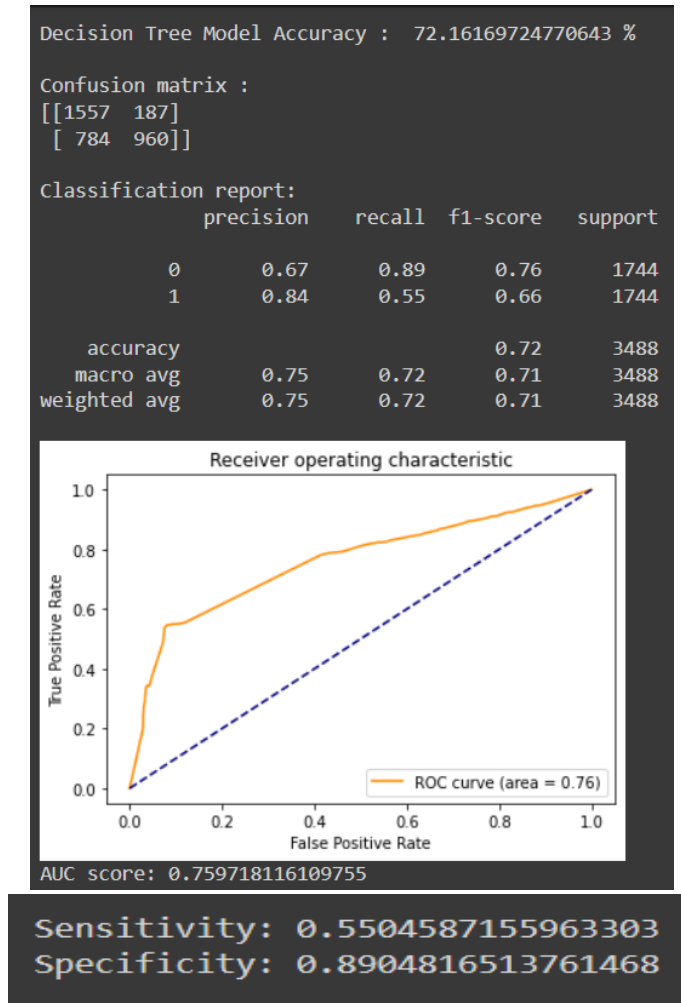


Fig 12. Decision Tree ROC Curve and Classification Report

The decision tree model has an accuracy of 72.16%, which is a significant improvement over the logistic model. The confusion matrix shows that the model has correctly predicted 1557 true positives (samples correctly classified as 0) and 960 true negatives (samples correctly classified as 1), but has misclassified 187 false positives (samples wrongly classified as 1) and 784 false negatives (samples wrongly classified as 0).

The classification report shows precision, recall, and f1-score for each class. Precision is the proportion of true positives among all positive predictions, while recall is the proportion of true positives among all actual positive samples. F1-score is the harmonic mean of precision and recall. The report shows that precision and recall are both higher for class 1 than class 0, indicating that the model is better at identifying positive samples. The weighted average of precision, recall, and f1-score is 0.75 for both classes.

The ROC/AUC score is 0.76, which indicates that the model's ability to distinguish between positive and negative classes is better than the logistic model.

In summary, the decision tree model has significantly better accuracy, precision, recall, and f1-score than the logistic model, and its ability to differentiate between positive and negative classes is higher. However, it still misclassifies a significant number of samples, and further improvements may be possible through hyperparameter tuning or other techniques such as ensemble methods.

3. Random Forests

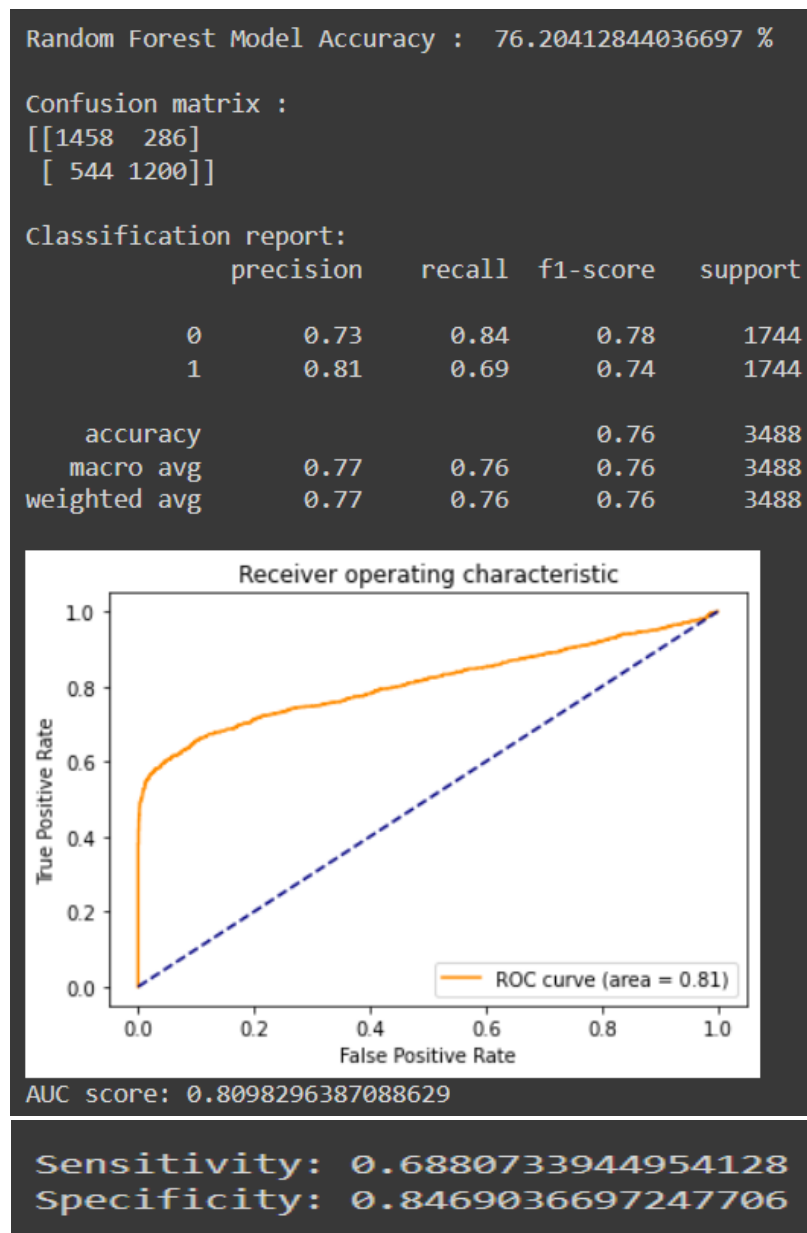


Fig 13. Random Forest ROC Curve and Classification Report

The random forest model has an accuracy of 76.20%, which is an improvement over both the logistic and decision tree models. The confusion matrix shows that the model has correctly predicted 1458 true positives (samples correctly classified as 0) and 1200 true negatives (samples correctly classified as 1), but has misclassified 286 false positives (samples wrongly classified as 1) and 544 false negatives (samples wrongly classified as 0).

The classification report shows precision, recall, and f1-score for each class. Precision is the proportion of true positives among all positive predictions, while recall is the proportion of true positives among all actual positive samples. F1-score is the harmonic mean of precision and recall. The report shows that precision and recall are both higher for class 1 than class 0, indicating that the model is better at identifying positive samples. The weighted average of precision, recall, and f1-score is 0.77 for both classes.

The AUC score is 0.81, which indicates that the model's ability to distinguish between positive and negative classes is good.

In summary, the random forest model has the highest accuracy, precision, recall, and f1-score among the three models evaluated, and its ability to differentiate between positive and negative classes is good. However, it still misclassifies a non-negligible number of samples, and further improvements may be possible through hyperparameter tuning or other techniques such as boosting or stacking.

4. Support Vectors Classifier

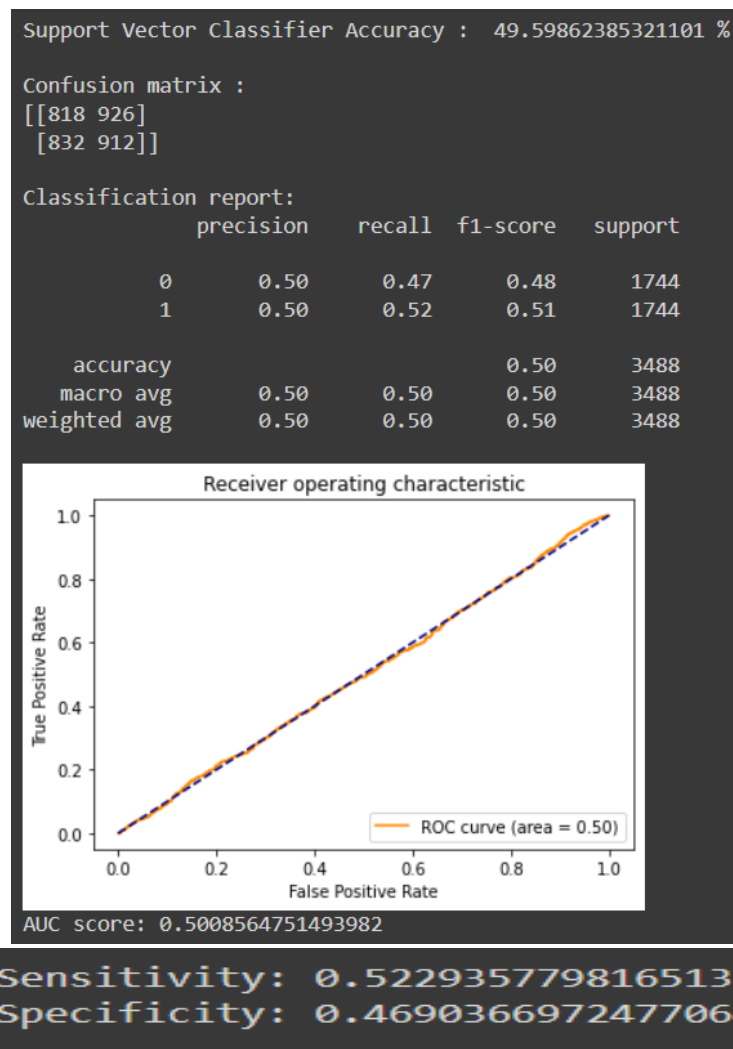


Fig 14.SVM ROC Curve and Classification Report

The support vector classifier (SVC) model has an accuracy of 49.60%, which is not much better than random guessing. The confusion matrix shows that the model has misclassified almost as many samples as it has classified correctly, with 818 true negatives and 912 true positives, but 926 false negatives and 832 false positives.

The classification report shows that the model has low precision, recall, and f1-score for both classes, indicating that it is not performing well at identifying positive or negative samples. The weighted average of precision, recall, and f1-score is also low at 0.50.

The AUC/ROC score is 0.50, which indicates that the model is performing no better than random guessing at distinguishing between positive and negative classes.

In summary, the SVC model is not performing well on this dataset and may require additional feature engineering, hyperparameter tuning, or a different modeling approach to improve performance.

5. KNN Classifier

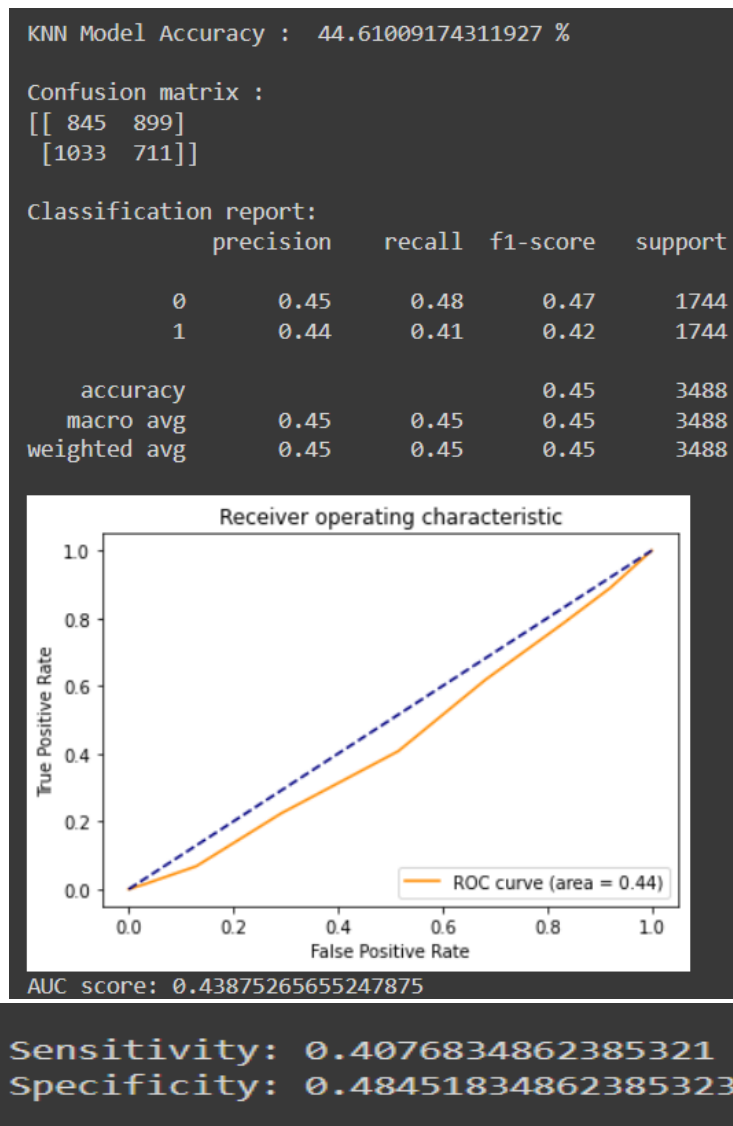


Fig 15.KNN ROC Curve and Classification Report

The KNN model has an accuracy of 44.61%, which is even worse than random guessing. The confusion matrix shows that the model has misclassified a large number of samples, with 845 true negatives and 711 true positives, but 899 false negatives and 1033 false positives.

The classification report shows that the model has low precision, recall, and f1-score for both classes, indicating that it is not performing well at identifying positive or negative samples. The weighted average of precision, recall, and f1-score is also low at 0.45.

In summary, the KNN model is not performing well on this dataset and may require additional feature engineering, hyperparameter tuning, or a different modeling approach to improve performance.

6. XG Boost classifier.

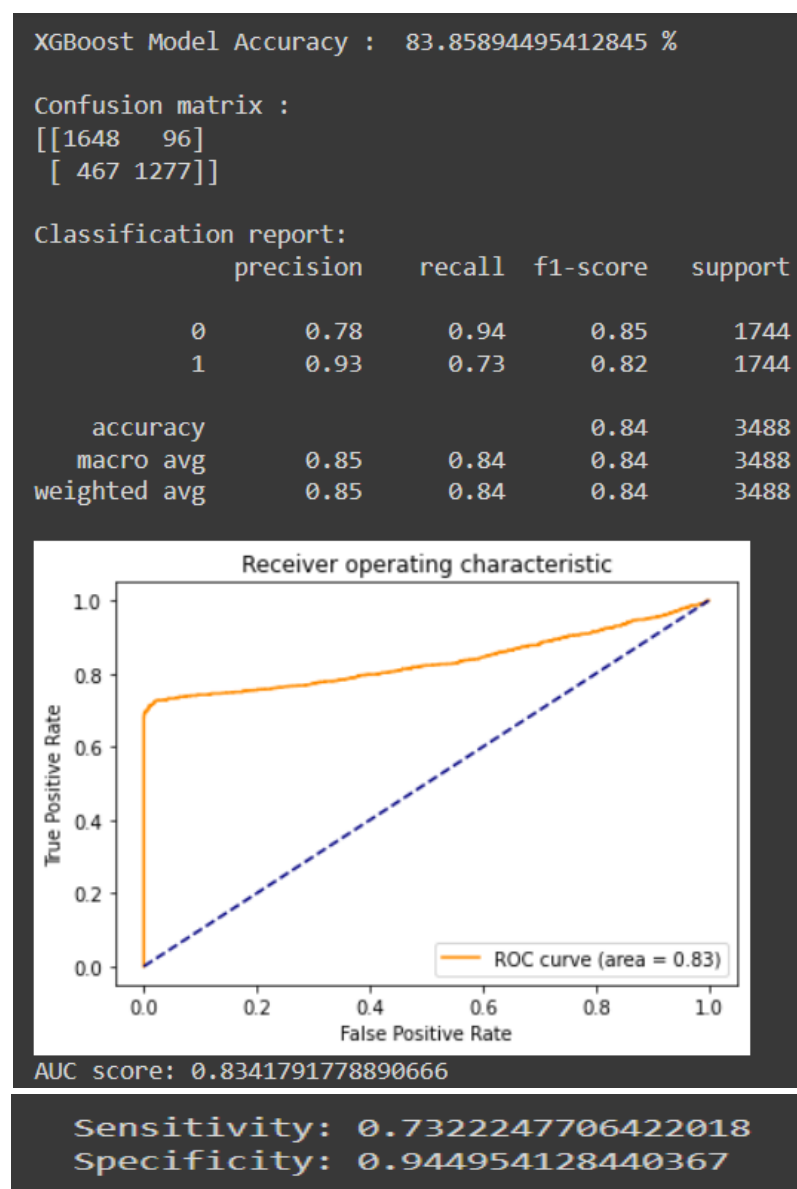


Fig 16. XG Boost ROC Curve and Classification Report

The XGBoost model has an accuracy of 83.86%, which is the highest among all the models evaluated. The confusion matrix shows that the model has correctly classified a large number of samples, with 1648 true negatives and 1277 true positives, with only 96 false negatives and 467 false positives.

The classification report shows that the model has high precision, recall, and f1-score for both classes, indicating that it is performing well at identifying positive or negative samples. The weighted average of precision, recall, and f1-score is also high at 0.85.

In summary, the XGBoost model is performing very well on this dataset and can be considered as the best model among the ones evaluated. However, it is important to keep in mind that the model's performance may vary on other datasets and may require hyperparameter tuning or feature engineering to improve its performance further.

Summary of all the above performance Evaluation Metrics

Models	Precision	Recall	Accuracy	F1	Sensitivity	Specificity
Logistic Regression	0.49	0.52	0.49	0.49	0.46	0.52
Decision Tree	0.75	0.72	0.72	0.71	0.55	0.89
Random Forests	0.77	0.76	0.76	0.76	0.68	0.84
K Nearest Neighbors	0.45	0.45	0.44	0.45	0.40	0.48
Support Vectors	0.50	0.50	0.49	0.50	0.52	0.46
XG Boost	0.85	0.84	0.83	0.84	0.73	0.94

Table 2. Model classification Report

The table shows the performance metrics of six different models, evaluated using Precision, Recall, Accuracy, F1-score, Sensitivity, and Specificity. Here's a brief summary of the models' performance:

- Logistic Regression:** It has the lowest performance metrics across all the evaluation metrics. It has a low Precision, Recall, F1-score, Sensitivity, and Specificity, indicating that it's not a suitable model for this dataset.
- Decision Tree:** This model has higher scores than Logistic Regression for all metrics except specificity, making it a better option. The high sensitivity (0.55) indicates that the model can correctly identify most of the positive cases.
- Random Forests:** This model outperforms both Logistic Regression and Decision Tree models. It has high scores for all metrics except sensitivity, indicating that it may miss a few positive cases.
- K Nearest Neighbors:** This model has low scores for all metrics except specificity. It's not a suitable model for this dataset.
- Support Vectors:** This model has low scores for all metrics except sensitivity. It's not a suitable model for this dataset.

6. **XG Boost:** This model outperforms all other models in terms of all metrics, with high scores for precision, recall, F1-score, sensitivity, and specificity.

Based on the above analysis, XG Boost is the best model for this dataset, as it has the highest scores for all metrics. It's worth noting that the Decision Tree and Random Forest models also perform relatively well, but XG Boost outperforms them.

Results:

In this project, we looked at the dataset for credit card approval prediction and built a machine learning model to forecast credit card acceptance based on numerous parameters. We began by performing data preprocessing, which included scaling the data, removing missing values, and transforming categorical attributes into numerical ones. Then, in order to learn more about the dataset and identify any patterns or correlations, we conducted exploratory data analysis. We also created and trained a number of machine learning models, including decision trees, random forests, and logistic regression, and we assessed their performance using a variety of metrics.

The model that performed the best, XG Boost, had a test set accuracy of 83%. Based on their data, this model might be used to forecast whether new applicants will be approved for credit cards.

But, by gathering more data, including more features, or experimenting with various machine learning algorithms, significant advancements could be made.

Impact of the Outcomes:

Both the applicants and the credit card firms may be significantly impacted by the credit card acceptance prediction project. The applicants can more fully understand their prospects of acceptance and make educated decisions about applying for credit by precisely predicting credit card approval. The prediction model can also help credit card firms make better decisions and lower the likelihood of defaults and late payments. Better profitability and financial stability for the credit card businesses may follow from this.

Also, the project can act as a springboard for additional investigation and the creation of more complex credit risk models that contain more advanced features and methodologies. The credit card acceptance prediction project could, in the end, help the financial sector advance while also benefiting consumers and credit card firms.

References:

[1] <https://www.kaggle.com/datasets/rikdifos/credit-card-approval-prediction>