Abstract

Fraud detection is an important aspect of any business or organization. It helps in identifying and preventing fraudulent activities that can cause financial losses. With the advent of technology, fraudsters have become more sophisticated in their methods, making it difficult for traditional methods of fraud detection to be effective. This is where data science comes in. By using machine learning algorithms, data scientists can develop models that can accurately predict fraudulent activities.

Introduction

The threat of fraudulent activities grows large in today's business operations landscape, posing potential financial risks to companies. Traditional detection methods become less effective as fraudulent methodologies become more complex with the advancement of technology. The combination of data science and machine learning has become a key strategy for enhancing fraud detection capabilities in response to this changing environment. Using a dataset designed especially for credit card fraud detection, this study focuses on the critical task of fraud detection using machine learning algorithms.

Creating reliable prediction models that can distinguish between real and fraudulent transactions is the main goal of this research. The train and test datasets are used to assess the models' performance, and the dataset is the basis for training different machine learning classifiers. Important metrics like accuracy, confusion matrix analysis, area under the receiver operating characteristic curve (AUC), and F1 score are used to assess how effective the models are.

The study aims to identify and utilise the most relevant information for fraud detection by incorporating feature engineering and selection techniques to improve the predictive power of the models. Logistic regression, Support Vector Machine (SVM), K-nearest neighbour (KNN), Decision Tree, Random Forest, Naive Bayes, XGBoost, and Light GBM are some of the classifiers that were selected for this study. By conducting a thorough investigation of these approaches, this study aims to provide light on how well each classifier addresses the complex problems associated with credit card fraud detection.

1 EDA, Feature Engineering and Selection

1.1 Data Overview

The dataset under consideration comprises 1,296,675 transactions with 23 columns. The target variable, `is_fraud`, is binary, with 7,506 instances of fraud (1) and 1,289,169 instances of non-fraud (0).

1.2 Data Cleaning and Preprocessing

1.2.1 Unnecessary Columns

Several columns deemed unnecessary for the analysis were dropped: `Unnamed: 0`, `street`, `state`, `zip`, `first`, `last`, `trans_num`, and `unix_time`.

1.2.2 Time

The `trans_date_trans_time` column was converted to a datetime format. Additionally, a new column, `trans_date`, was created to represent the date without the timestamp. The `age` of each transaction was calculated based on the difference between `trans_date` and `dob`. Separate columns for transaction month (`trans_month`) and year (`trans_year`) were also created.

1.2.3 Gender

The `gender` column was transformed into binary values (1 for 'M' and 0 for 'F').

1.2.4 Distance

The geographical distance between the transaction and the merchant was calculated and stored in `lat_dis` and `long_dis` columns.

1.2.5 Merchant

The `merchant` column was cleaned by removing the 'fraud_' prefix. Label encoding was applied to categorical columns, including `merchant`, `category`, `job`, and `city`.

1.2.6 Correlation Analysis

A correlation matrix was computed, and a heatmap was generated to visualize the correlation between different features.

1.3 Label Encoding

Label encoding was applied to categorical columns using the `LabelEncoder` from scikit-learn.

1.4 Train-Test Split

The dataset was split into training and testing sets with an 80-20 ratio. The resulting shapes of the sets were as follows:

X_train shape: (1037340, 14)
X_test shape: (259335, 14)
y_train shape: (1037340,)
y_test shape: (259335,)

1.5 Standardization

Standard scaling was applied using `StandardScaler` from scikit-learn to normalize the feature values in both the training and testing sets.

1.6 Conclusion

The foundation for developing and assessing machine learning models for credit card fraud detection is laid by this thorough preprocessing and feature engineering pipeline. The chosen classifiers can now be trained and tested using the cleaned and transformed dataset and standardised features.

2 Model Training and Evaluation Overview

In order to create and evaluate our credit card fraud detection models, we used a standard approach for all classifiers. An 80-20 split was made in the training dataset, with 80% going towards training the model and the remaining 20% going towards testing. For every classifier, a methodical pipeline covering the whole gamut from feature engineering and preprocessing to model training and assessment was put into place.

Logistic regression, Support Vector Machine (SVM), K-nearest neighbour (KNN), Decision Tree, Random Forest, Naive Bayes, XGBoost, and Light GBM are some of the models that were taken into consideration for this investigation. We specifically applied undersampling to the training data for SVM and KNN classifiers due to their computationally demanding nature, in order to maximise computational resources. The remaining classifiers, on the other hand, were trained using the complete dataset.

For the evaluation of model performance, we utilized key metrics including the classification report, area under the receiver operating characteristic curve (AUC-ROC), confusion matrix analysis, and the F1 score. These metrics provide a comprehensive view of each model's ability to discriminate between legitimate and fraudulent transactions. By presenting results for each method individually, we aim to offer insights into the strengths and limitations of various machine learning approaches for credit card fraud detection.

2.1 Logistic Regression

2.1.1 Training Results

We used the training dataset to start the training process for the Logistic Regression model. The Logistic Regression algorithm was utilised to fit the model, and predictions were generated for the test set. 99.55% accuracy was attained during the training phase. But after looking over the classification report, it's clear that the model had trouble spotting fraud cases (class 1), as evidenced by the fraud class's 0.0 F1-score, precision, and recall.

Class	Precision	Recall	F1-Score	Support
0	0.99	1.00	1.00	257,780
1	0.00	0.00	0.00	1,555

Table 1: Classification Report for Logistic Regression (Training)

The macro-averaged F1-score and weighted F1-score for the Logistic Regression training phase were both 0.0.

2.1.2 Test Results

After that, the test dataset was subjected to the Logistic Regression pipeline, yielding an accuracy of 99.55%. However, as shown by precision, recall, and F1-score values of 0.0 for class 1, the model had difficulty correctly classifying instances of fraud, just as it had during the training phase.

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	553,574
1	0.00	0.00	0.00	2,145

Table 2: Classification Report for Logistic Regression (Test)

For the Logistic Regression test phase, both the weighted and macro-averaged F1-scores were 0.0. For a thorough grasp of the model's performance, additional examination of the confusion matrix and ROC plot is provided.

2.1.3 Evaluation Metrics and Visualizations

To gain a more comprehensive understanding of the Logistic Regression model's performance, we examine both training and test results in terms of Receiver Operating Characteristic (ROC) curves and confusion matrices.

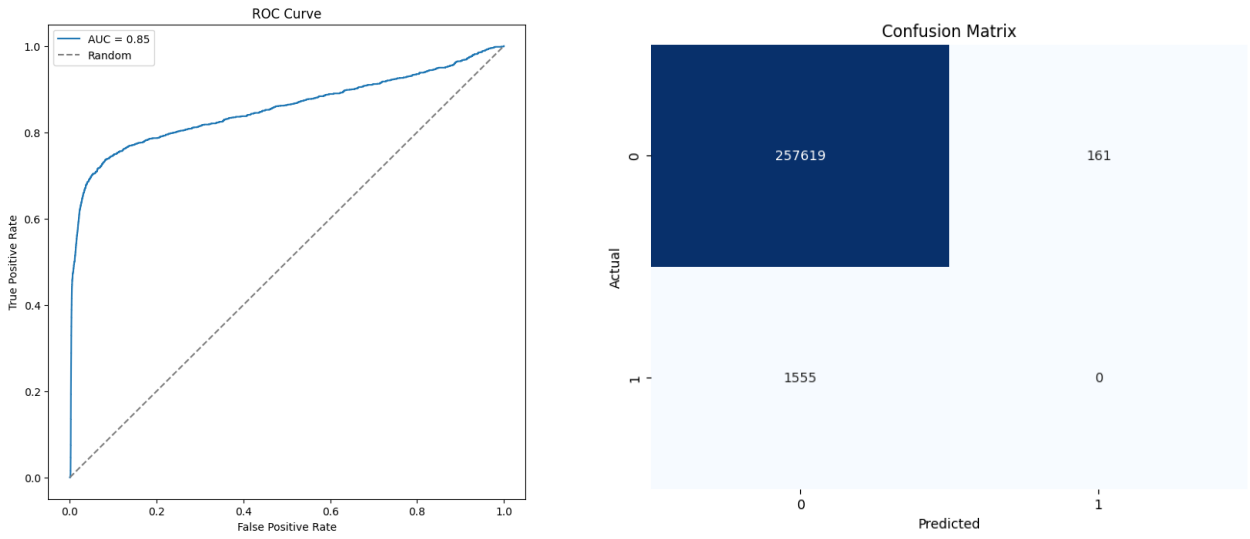

Figure 1: ROC Plot and Confusion Matrix for Logistic Regression (Training)

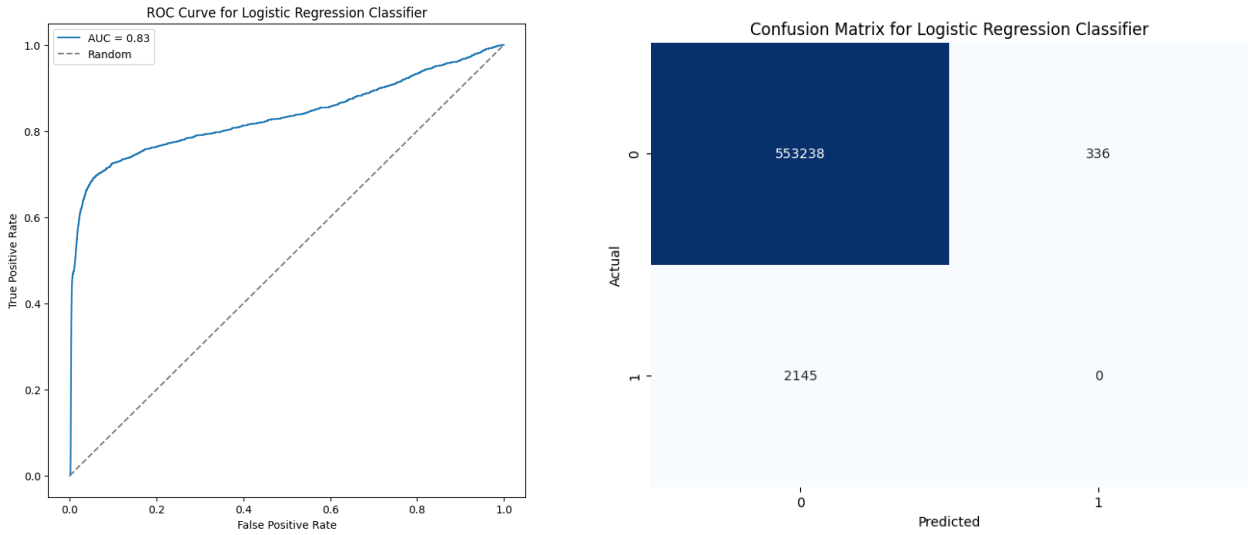

Figure 2: ROC Plot and Confusion Matrix for Logistic Regression (Test)

The trade-off between true positive rate and false positive rate for different thresholds is visually represented by the ROC plots. Furthermore, the confusion matrices provides insight into the model's accuracy in classifying instances as well as possible misclassifications. These visuals provide a more nuanced view of the performance of the Logistic Regression model, enhancing the numerical metrics that were previously presented.

2.2 Decision Tree Classifier

2.2.1 Training Results

With the dataset provided, the Decision Tree Classifier was trained, achieving an accuracy of 99.55%. According to the classification report, the model achieved perfect precision, recall, and F1-score for class 0, which indicates that it performed exceptionally well in correctly identifying non-fraudulent transactions. However, the model performed moderately for fraudulent transactions (class 1), with an F1-score of 0.64, recall of 0.67, and precision of 0.62.

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	257,780
1	0.62	0.67	0.64	1,555

Table 3: Classification Report for Decision Tree Classifier (Training)

The macro-averaged F1-score and weighted F1-score for the Decision Tree Classifier training phase were 0.643 and 1.00, respectively.

2.2.2 Test Results

An accuracy of 99.55% was obtained by testing the Decision Tree Classifier on the test dataset. The model performed exceptionally well in identifying non-fraudulent transactions (class 0), consistent with the training results. With an F1-score of 0.56, recall of 0.60, and precision of 0.53, it performed marginally worse on fraudulent transactions (class 1) when compared to the training phase.

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	553,574
1	0.53	0.60	0.56	2,145

Table 4: Classification Report for Decision Tree Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the Decision Tree Classifier test phase were 0.564 and 1.00, respectively.

2.2.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

These visualizations complement the numerical metrics, providing a holistic view of the Decision Tree Classifier's performance in detecting fraudulent transactions.

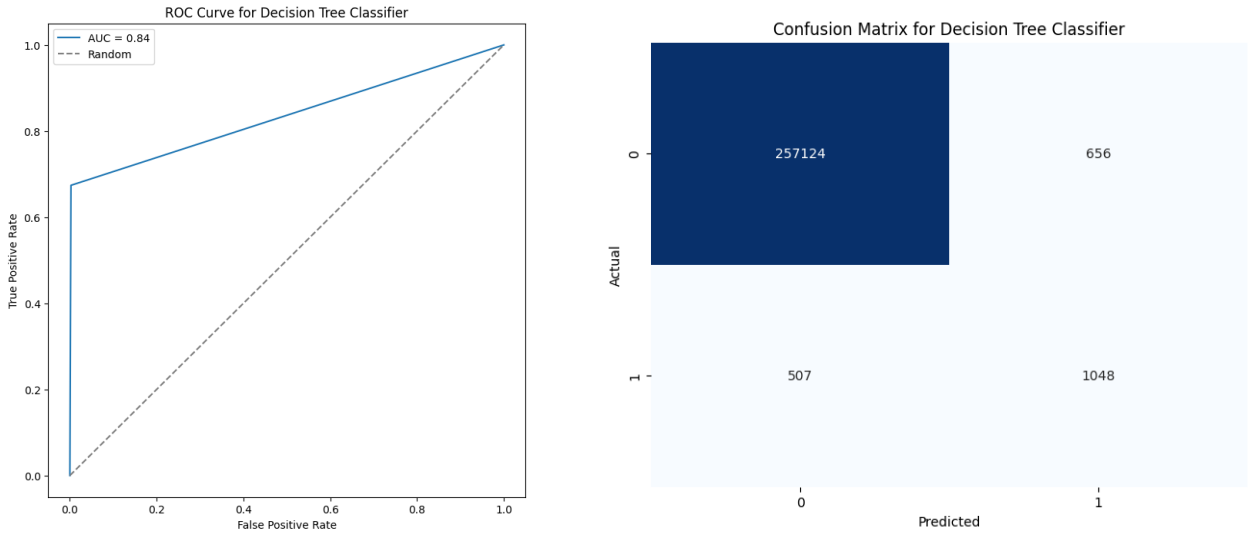

Figure 3: ROC Plot and Confusion Matrix for Decision Tree (Training)

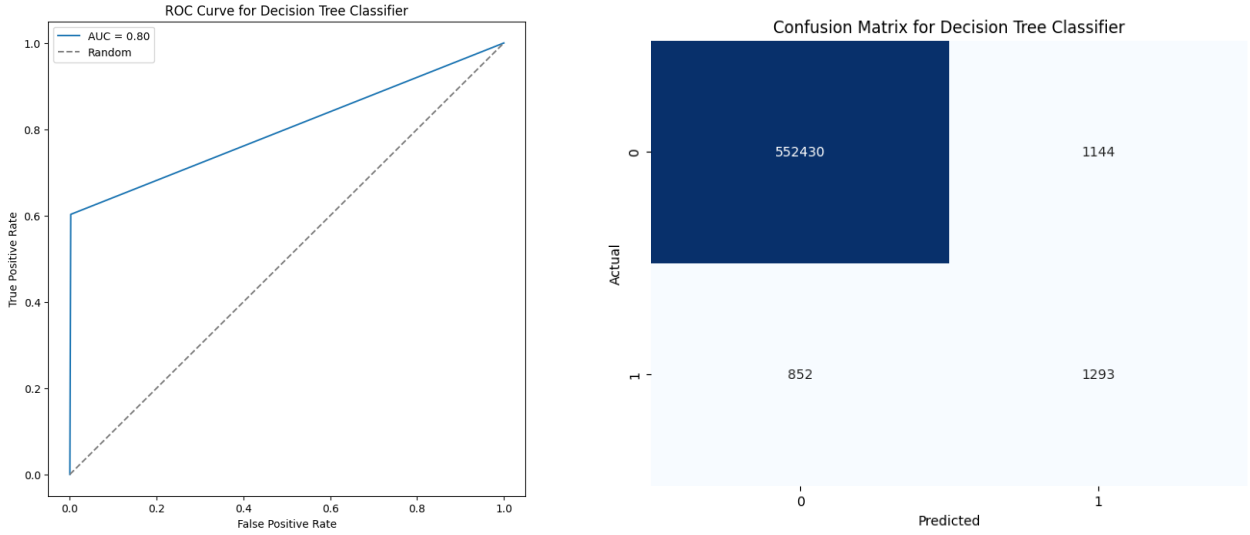

Figure 4: ROC Plot and Confusion Matrix for Decision Tree (Test)

2.3 Naive Bayes Classifier

2.3.1 Training Results

With the dataset provided, the Naive Bayes Classifier was trained and reached 99.01% accuracy. The model performed relatively poorly at identifying fraudulent transactions (class 1), even though it showed high precision and recall for non-fraudulent transactions (class 0). For class 1, the corresponding values for precision, recall, and F1-score were 0.30, 0.48, and 0.37.

The macro-averaged F1-score and weighted F1-score for the Naive Bayes Classifier training phase were 0.370 and 0.99, respectively.

2.3.2 Test Results

After that, the test dataset was used to evaluate the Naive Bayes Classifier, which produced an accuracy of 99.19%. In line with the training phase, the model performed better

Class	Precision	Recall	F1-Score	Support
0	1.00	0.99	1.00	257,780
1	0.30	0.48	0.37	1,555

Table 5: Classification Report for Naive Bayes Classifier (Training)

in terms of recall and precision for class 0 than for class 1. For class 1, the corresponding values for precision, recall, and F1-score were 0.23, 0.47, and 0.31.

Class	Precision	Recall	F1-Score	Support
0	1.00	0.99	1.00	553,574
1	0.23	0.47	0.31	2,145

Table 6: Classification Report for Naive Bayes Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the Naive Bayes Classifier test phase were 0.309 and 0.99, respectively.

2.3.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

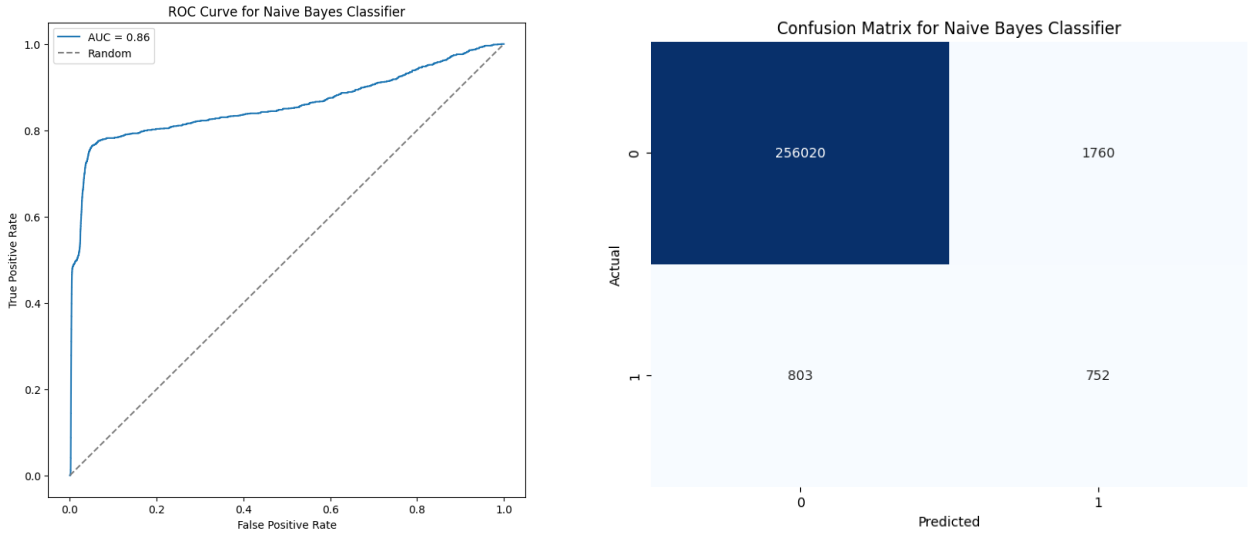

Figure 5: ROC Plot and Confusion Matrix for Naive Bayes (Training)

These visualizations supplement the numerical metrics, offering a more comprehensive view of the Naive Bayes Classifier's performance in detecting fraudulent transactions.

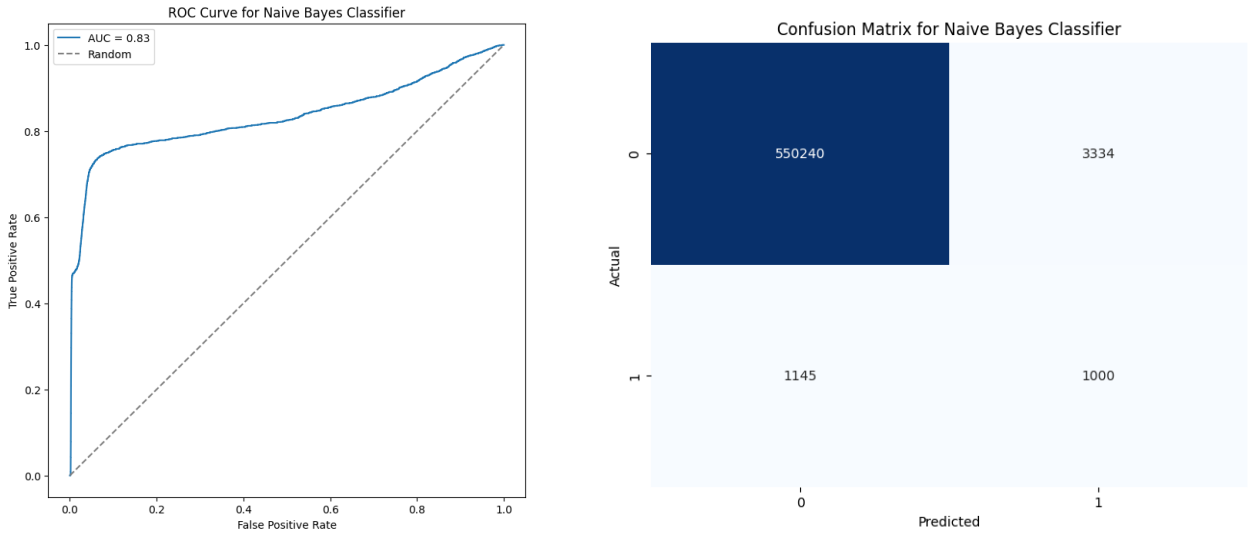

Figure 6: ROC Plot and Confusion Matrix for Naive Bayes (Test)

2.4 Random Forest Classifier

2.4.1 Training Results

With the dataset provided, the Random Forest Classifier was trained and reached 99.75% accuracy. With an F1-score of 1.00, precision, recall, and excellent performance in identifying non-fraudulent transactions (class 0), the model was shown. The model performed well for fraudulent transactions (class 1), with an F1-score of 0.76, recall of 0.67, and precision of 0.88.

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	257,780
1	0.88	0.67	0.76	1,555

Table 7: Classification Report for Random Forest Classifier (Training)

The macro-averaged F1-score and weighted F1-score for the Random Forest Classifier training phase were 0.759 and 1.00, respectively.

2.4.2 Test Results

The Random Forest Classifier was then evaluated using the test dataset, and it obtained a 99.77% accuracy rate. Like in the training phase, the model performed exceptionally well in classifying transactions as non-fraudulent (class 0). The model demonstrated strong precision (0.82), recall (0.53), and an F1-score of 0.65 for fraudulent transactions (class 1).

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	553,574
1	0.82	0.53	0.65	2,145

Table 8: Classification Report for Random Forest Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the Random Forest Classifier test phase were 0.645 and 1.00, respectively.

2.4.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

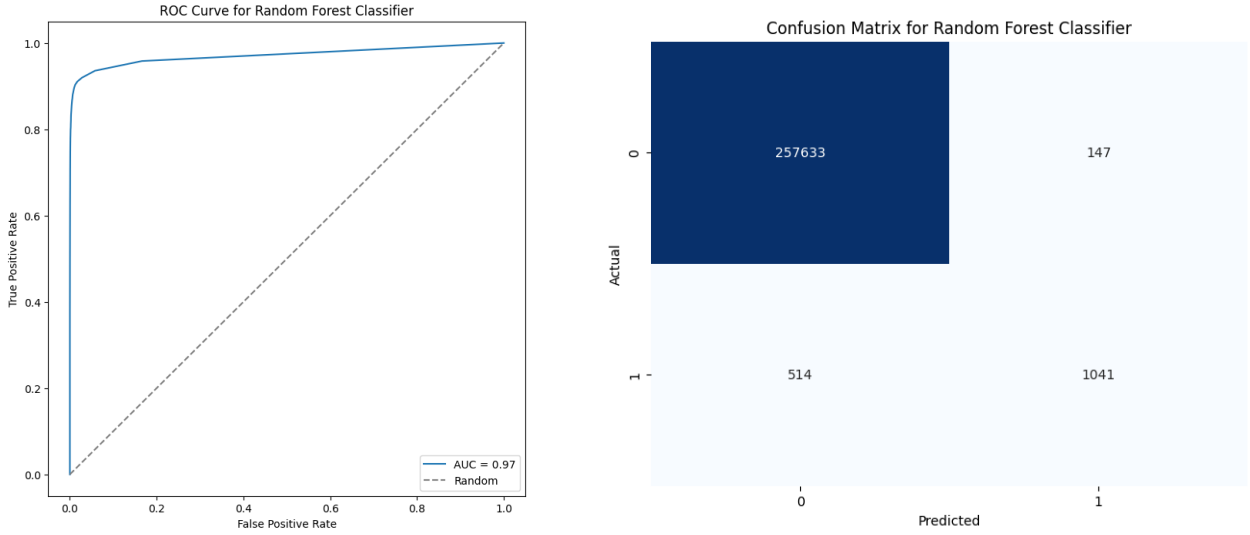

Figure 7: ROC Plot and Confusion Matrix for Random Forest (Training)

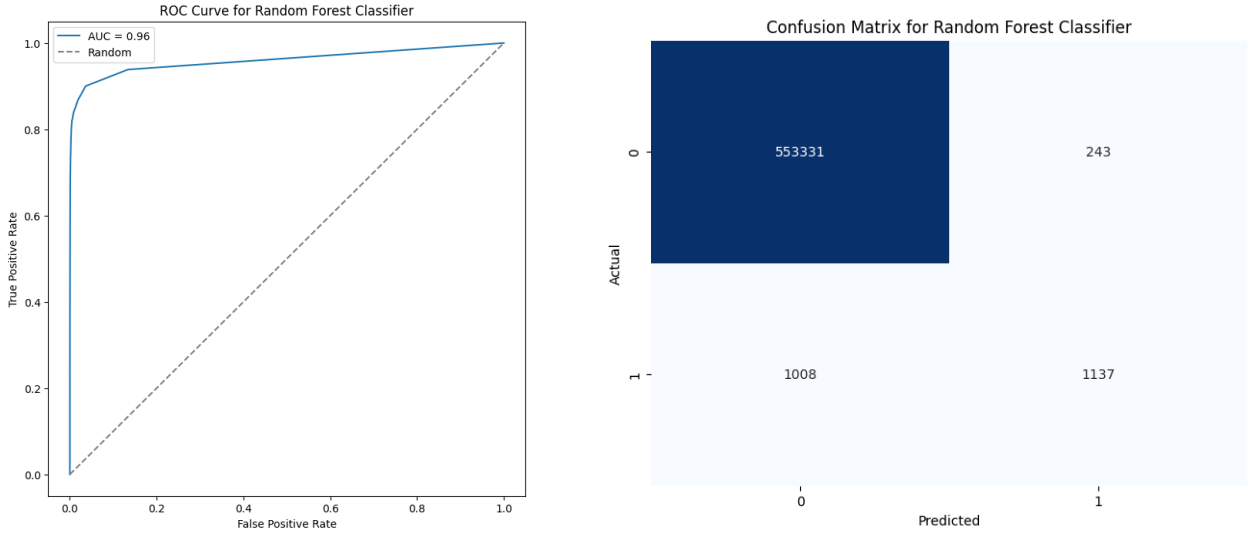

Figure 8: ROC Plot and Confusion Matrix for Random Forest (Test)

These visualizations offer additional insights into the Random Forest Classifier's ability to discriminate between legitimate and fraudulent transactions.

2.5 XGBoost Classifier

2.5.1 Training Results

With the dataset provided, the XGBoost Classifier was trained and reached 99.41% accuracy. For non-fraudulent transactions (class 0), the model showed high precision, recall, and F1-score, with all metrics reaching 1.00. On the other hand, the model performed worse for fraudulent transactions (class 1), with an F1-score of 0.53, recall of 0.56, and precision of 0.51.

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	257,780
1	0.51	0.56	0.53	1,555

Table 9: Classification Report for XGBoost Classifier (Training)

The macro-averaged F1-score and weighted F1-score for the XGBoost Classifier training phase were 0.532 and 0.99, respectively.

2.5.2 Test Results

After that, the test dataset was used to evaluate the XGBoost Classifier, which produced an accuracy of 99.74%. The model demonstrated remarkable performance in class 0 (non-fraudulent transaction identification) with an F1-score of 1.00, recall, and precision all exceeding 1.00. The model demonstrated a precision of 0.81, recall of 0.42, and an F1-score of 0.55 for fraudulent transactions (class 1).

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	553,574
1	0.81	0.42	0.55	2,145

Table 10: Classification Report for XGBoost Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the XGBoost Classifier test phase were 0.552 and 1.00, respectively.

2.5.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

These visualizations offer additional insights into the XGBoost Classifier's ability to distinguish between legitimate and fraudulent transactions.

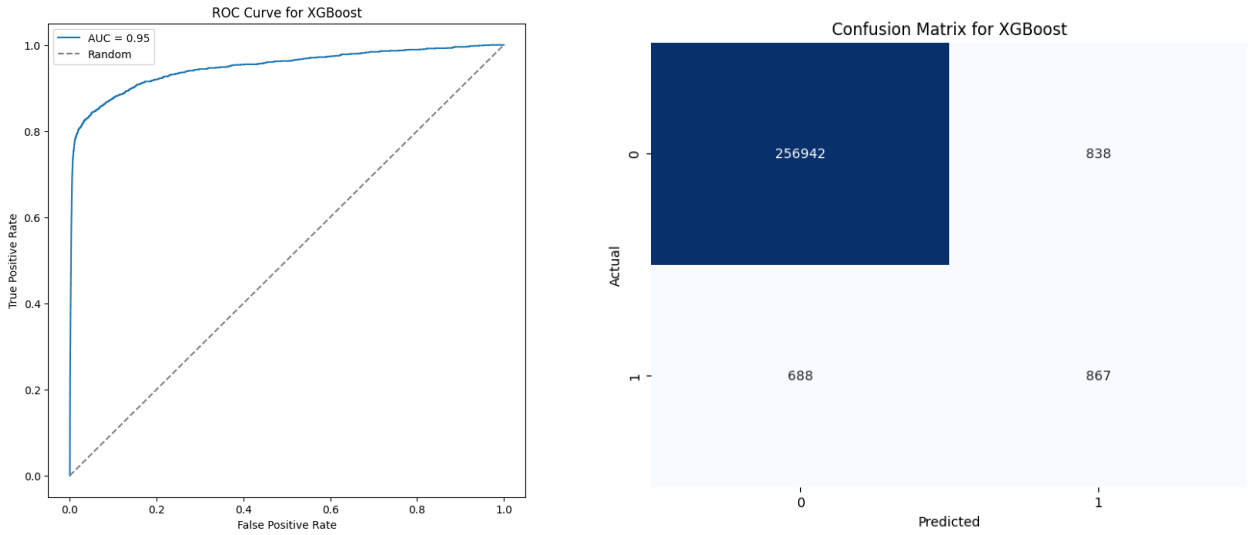

Figure 9: ROC Plot and Confusion Matrix for XGBoost (Training)

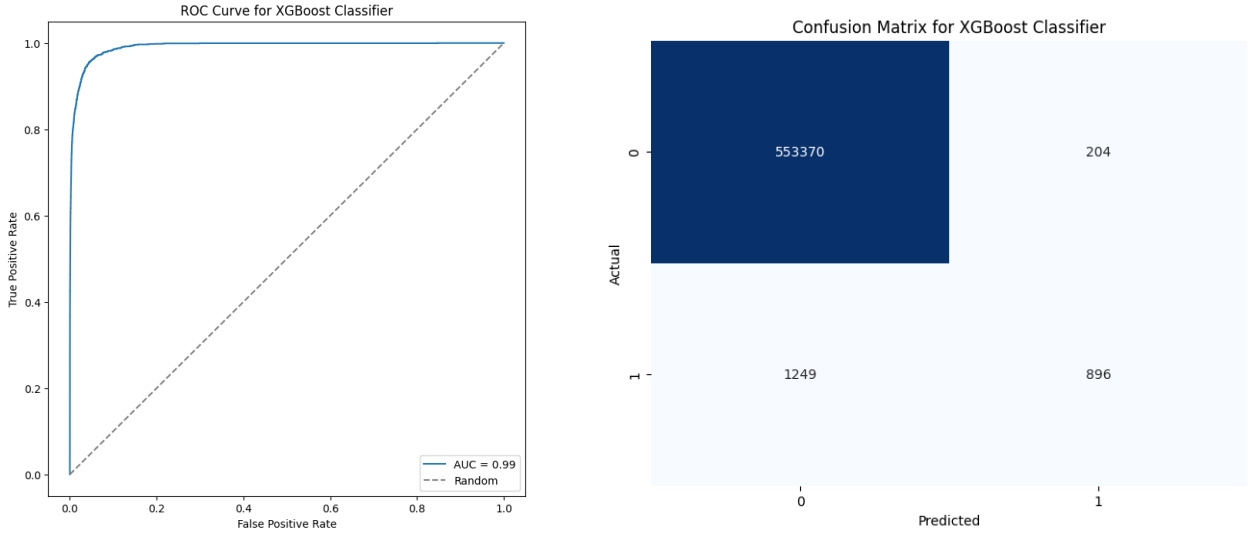

Figure 10: ROC Plot and Confusion Matrix for XGBoost (Test)

2.6 LightGBM Classifier

2.6.1 Training Results

After training on the given dataset, the LightGBM Classifier achieved 99.60% accuracy. The model performed well in accurately classifying transactions that were not fraudulent (class 0), with an F1-score of 1.00, recall, and precision all reaching 1.00. On the other hand, the model performed moderately well for fraudulent transactions (class 1), with an F1-score of 0.56, recall of 0.60, and precision of 0.53.

The macro-averaged F1-score and weighted F1-score for the LightGBM Classifier training phase were 0.562 and 0.99, respectively.

2.6.2 Test Results

After that, the test dataset was used to evaluate the LightGBM Classifier, which produced an accuracy of 99.60%. The model's accuracy, recall, and F1-score all reached 1.00

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	257,780
1	0.53	0.60	0.56	1,555

Table 11: Classification Report for LightGBM Classifier (Training)

when it came to classifying non-fraudulent transactions (class 0). The model demonstrated a precision of 0.48, recall of 0.56, and an F1-score of 0.52 for fraudulent transactions (class 1).

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	553,574
1	0.48	0.56	0.52	2,145

Table 12: Classification Report for LightGBM Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the LightGBM Classifier test phase were 0.520 and 0.99, respectively.

2.6.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

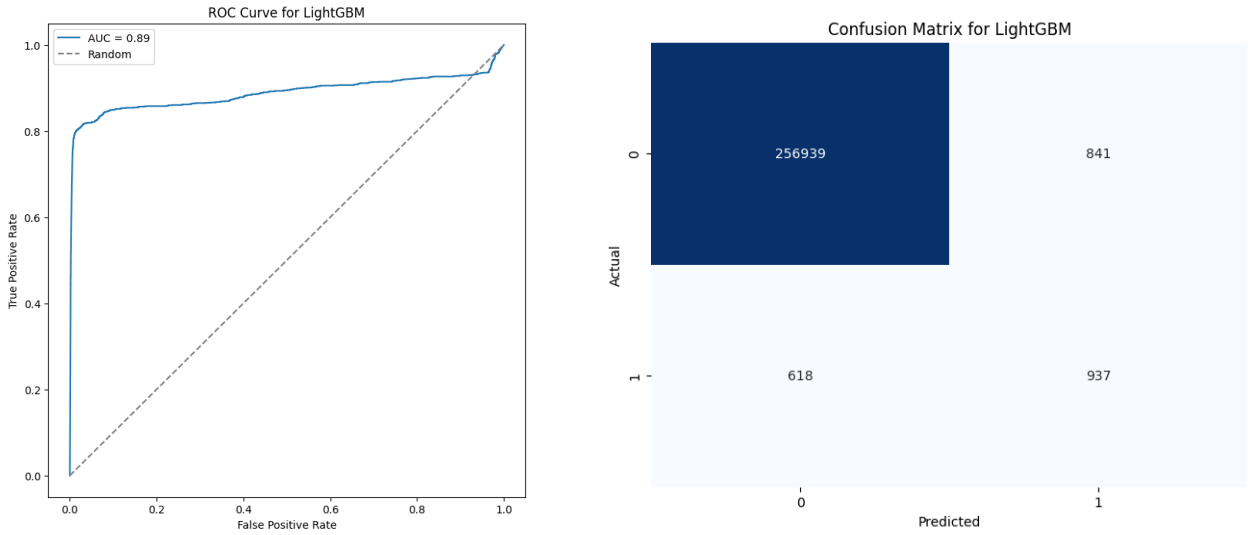

Figure 11: ROC Plot and Confusion Matrix for LightGBM (Training)

These visualizations provide a comprehensive overview of the LightGBM Classifier's performance in distinguishing between legitimate and fraudulent transactions.

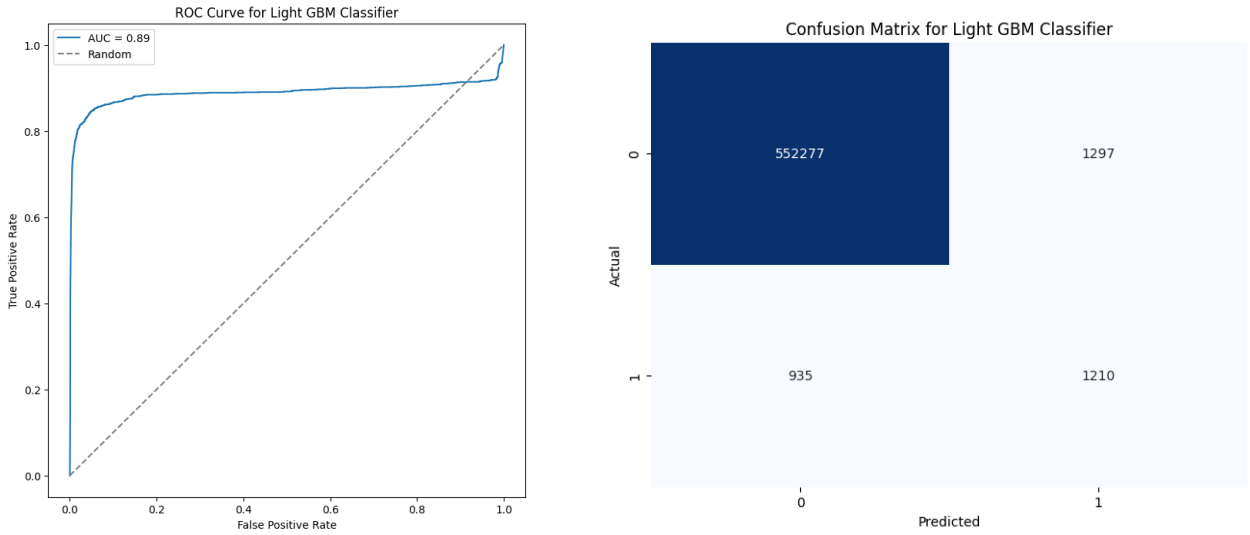

Figure 12: ROC Plot and Confusion Matrix for LightGBM (Test)

2.7 Support Vector Machine (SVM) Classifier

2.7.1 Training Results

After training on the given dataset, the SVM Classifier achieved an accuracy of 86.35%. Good precision, recall, and F1-score were shown by the model for both fraudulent (class 1) and non-fraudulent (class 0) transactions. For class 0, the corresponding values for precision, recall, and F1-score were 0.80, 0.97, and 0.88. The model produced an F1-score of 0.85, recall of 0.76, and precision of 0.96 for class 1.

Class	Precision	Recall	F1-Score	Support
0	0.80	0.97	0.88	1,501
1	0.96	0.76	0.85	1,502

Table 13: Classification Report for SVM Classifier (Training)

The macro-averaged F1-score and weighted F1-score for the SVM Classifier training phase were 0.848 and 0.86, respectively.

2.7.2 Test Results

After that, the SVM Classifier was assessed using the test dataset, and it obtained an accuracy of 96.13%. For transactions that were not fraudulent (class 0), the model demonstrated high precision, recall, and F1-score; these metrics reached 1.00, 0.96, and 0.98, respectively. Nevertheless, the model performed worse for fraudulent transactions (class 1), with an F1-score of 0.13, a recall of 0.73, and a precision of 0.07.

Class	Precision	Recall	F1-Score	Support
0	1.00	0.96	0.98	553,574
1	0.07	0.73	0.13	2,145

Table 14: Classification Report for SVM Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the SVM Classifier test phase were 0.126 and 0.96, respectively.

2.7.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

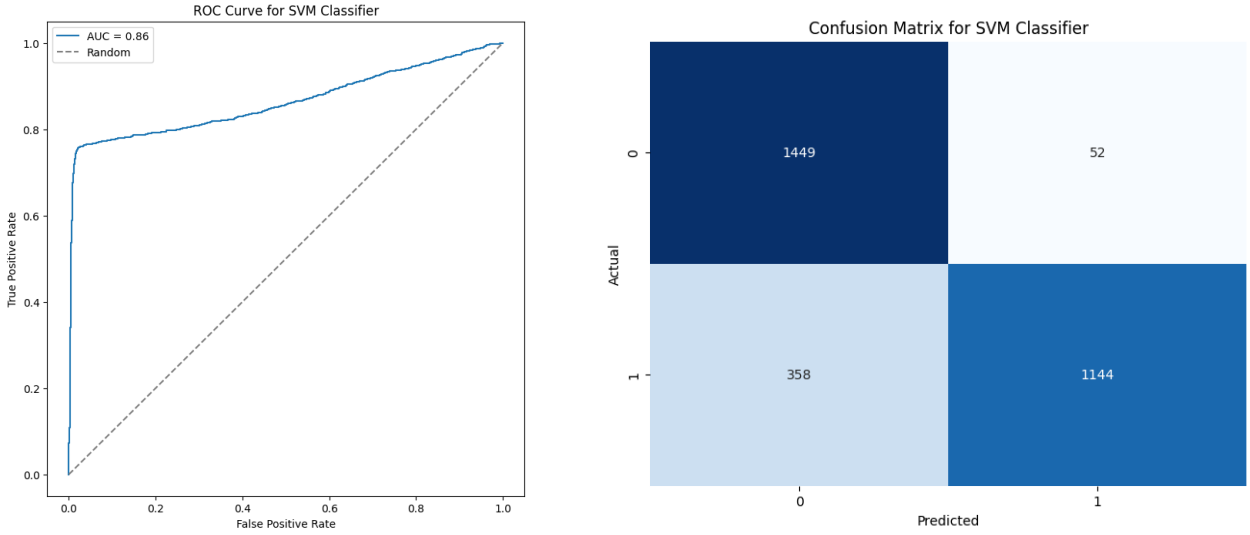

Figure 13: ROC Plot and Confusion Matrix for SVM (Training)

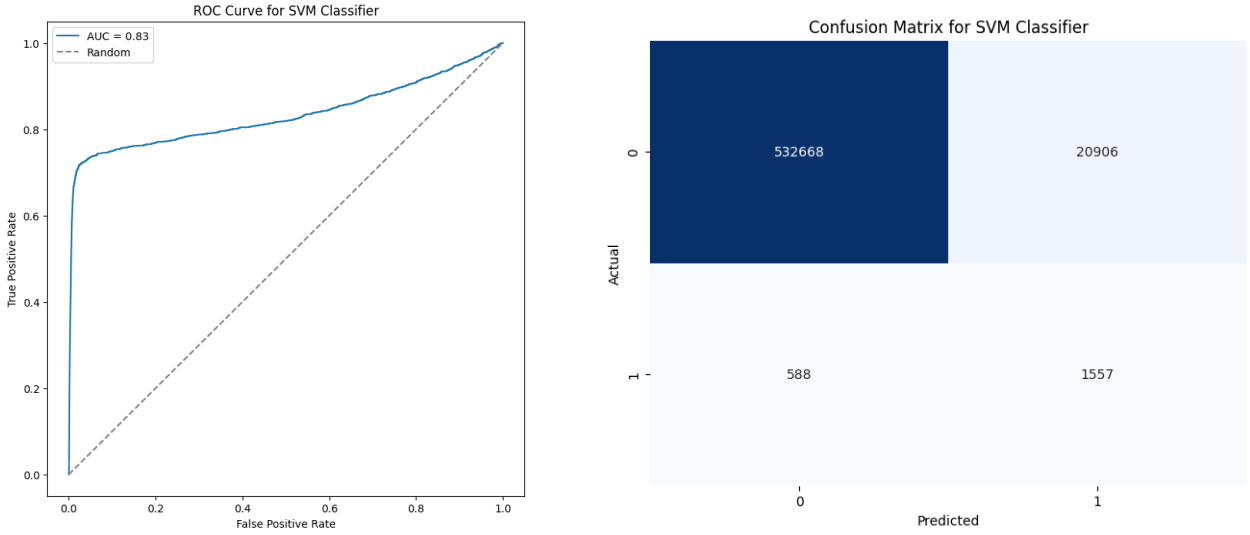

Figure 14: ROC Plot and Confusion Matrix for SVM (Test)

These visualizations offer insights into the SVM Classifier's ability to distinguish between legitimate and fraudulent transactions.

2.8 K-Nearest Neighbors (KNN) Classifier

2.8.1 Training Results

After training on the given dataset, the KNN Classifier achieved 79.22% accuracy. For both non-fraudulent (class 0) and fraudulent (class 1) transactions, the model showed balanced performance. For class 0, the corresponding values for precision, recall, and F1-score were 0.77, 0.82, and 0.80. The model produced an F1-score of 0.79, recall of 0.76, and precision of 0.81 for class 1.

Class	Precision	Recall	F1-Score	Support
0	0.77	0.82	0.80	1,501
1	0.81	0.76	0.79	1,502

Table 15: Classification Report for KNN Classifier (Training)

The macro-averaged F1-score and weighted F1-score for the KNN Classifier training phase were 0.786 and 0.79, respectively.

2.8.2 Test Results

After that, the KNN Classifier was assessed using the test dataset, and it obtained an accuracy of 78.23%. For non-fraudulent transactions (class 0), the model demonstrated high precision, recall, and F1-score: these metrics reached 1.00, 0.78, and 0.88, respectively. On the other hand, the model performed worse for fraudulent transactions (class 1), with an F1-score of 0.02 and a precision of 0.01.

Class	Precision	Recall	F1-Score	Support
0	1.00	0.78	0.88	553,574
1	0.01	0.62	0.02	2,145

Table 16: Classification Report for KNN Classifier (Test)

The macro-averaged F1-score and weighted F1-score for the KNN Classifier test phase were 0.022 and 0.78, respectively.

2.8.3 Evaluation Metrics and Visualizations

Visualizations, including ROC plots and confusion matrices, are provided below for both the training and test phases.

These visualizations provide insights into the KNN Classifier's ability to distinguish between legitimate and fraudulent transactions.

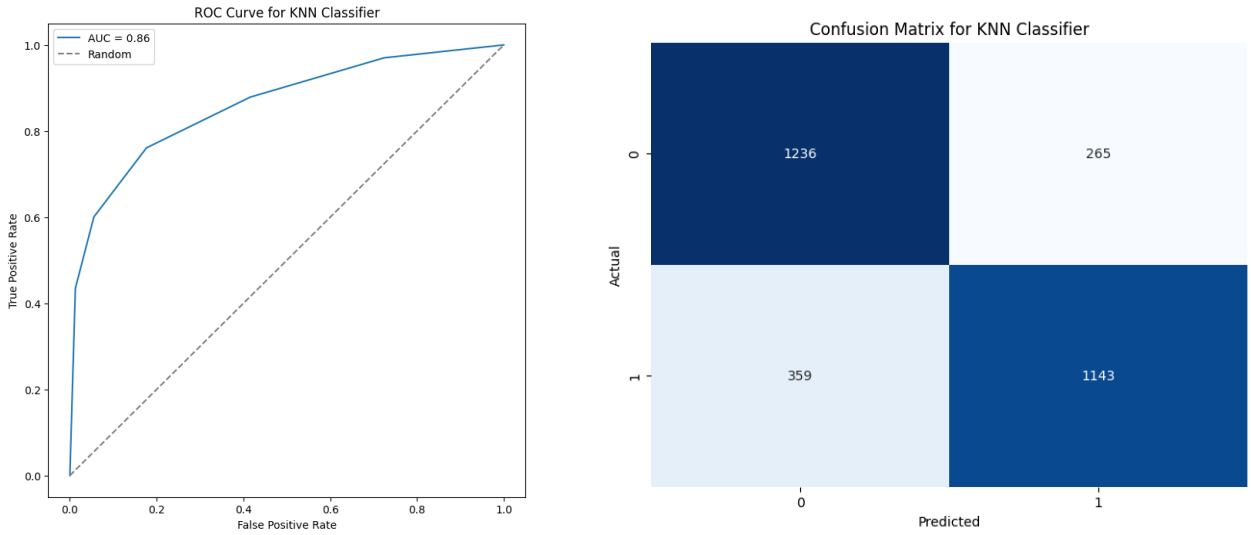

Figure 15: ROC Plot and Confusion Matrix for K-Nearest Neighbors (Training)

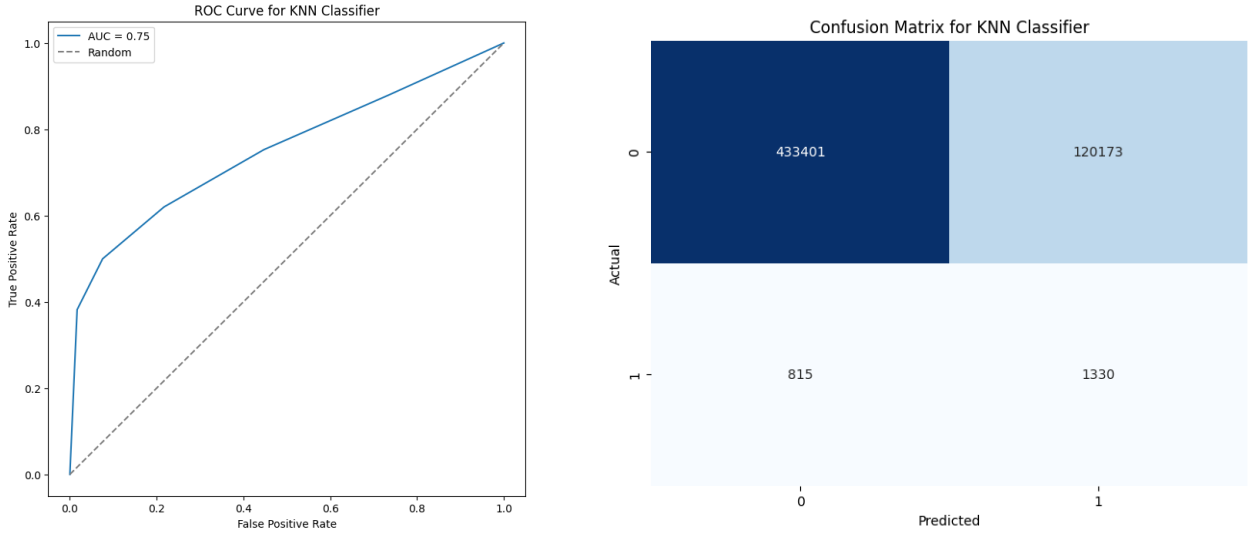

Figure 16: ROC Plot and Confusion Matrix for K-Nearest Neighbors (Test)

3 Conclusion

Trough this report we investigated the performance of eight different classifiers in this thorough examination of machine learning models for credit card fraud detection: Logistic Regression, Support Vector Machine (SVM), K-Nearest Neighbours (KNN), Decision Tree, Random Forest, Naive Bayes, XGBoost, and LightGBM. Creating reliable models that can reliably determine whether a transaction is fraudulent or not was our main objective.

The Logistic Regression model demonstrated remarkable accuracy, attaining nearly 99% accuracy on both the training and test datasets. With an F1-score of 0.0, its predictive ability to detect fraudulent transactions was, however, quite low.

With an F1-score of 0.56, the Decision Tree model performed significantly worse on the test set than it did on the training set, where it showed a high accuracy of 99.55% in identifying fraudulent transactions.

Even though Naive Bayes achieved an impressive 99% accuracy during training, it had trouble generalising to the test set, as evidenced by its F1-score of 0.31 for fraud detection.

During training, Random Forest demonstrated strong performance, achieving an accuracy of 99.75%. With an F1-score of 0.65 for fraud identification, its ability to generalise to the test set was also called into question.

During training, XGBoost and LightGBM showed competitive accuracy levels of about 99.41% and 99.60%, respectively. With XGBoost scoring 0.55 and LightGBM scoring 0.52 for fraud detection on the test set, both models, however, had difficulty reaching high F1-scores.

SVM encountered severe difficulties in generalisation, even though it achieved an accuracy of 86.35% on the training set. As a result, it only received an F1-score of 0.13 for fraud identification on the test set.

KNN performed well in training, achieving an accuracy of 79.22%; however, it had difficulty in generalising, resulting in an F1-score of 0.02 for fraud detection on the test set.

In conclusion, even though some models showed excellent accuracy during training, they performed differently on the test set, particularly when it came to spotting fraudulent transactions. Even with its high accuracy, Logistic Regression did not work well for fraud detection. Although it had encouraging results, Random Forest had problems with generalisation. Other models, including Decision Tree, Naive Bayes, XGBoost, LightGBM, SVM, and KNN, performed poorly and required improvement in order to improve their ability to predict future data.

Based on their unique needs and priorities, stakeholders can use the insightful information provided by this analysis to choose the best model for detecting credit card fraud.