ANOMALY DETECTION IN FRADULANT CREDIT TRANSFER

A Project Report submitted in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

By

Anish Khandelwal 201500093

Prashant Dhagar 201500503

Vijay Kaushal 201500783

Yash Kumar Gupta 201500820

Group No.: G170

Under the Guidance of

Mr. Vikash Sawan (Assistant Professor)

Department of Computer Engineering & Applications

Institute of Engineering & Technology

GLA University

Mathura-281406,INDIA

December,2023

CONTENTS

DECLARATION

I/we hereby declare that the work which is being presented in the B.Tech Project "**Anomaly Detection in Fraudulant Credit Transfer"**, in partial fulfillment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of ***Mr. Vikash Sawan Sir*** who is Assistant Professor in GLA University. The contents of this project report, full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Name of Candidate: Anish Khandelwal
University Roll No.: 201500093
Name of Candidate: Prashant Dhangar
University Roll No.: 201500503
Name of Candidate: Vijay Kaushal
University Roll No.: 201500783
Name of Candidate: Yash Kumar Gupta
University Roll No.: 201500820

Submission of report and project presentation on 14 Dec 2023

Supervisor Name: Mr. Vikash Sawan (Assistant Professor)
Signature: Examiner Signature:

ACKNOWLEDGEMENT

This project is going to be an Acknowledgement to the inspiration, drive and technical assistance will be contributed to it by many individuals. We owe special debt of gratitude to Mr. Vikash Sawan (Assistant Professor) for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal and for his constant support and guidance to our work. His sincerity, thoroughness and perseverance has been a constant source of inspiration for us. We believe that he will shower us with all his extensively experienced ideas and insightful comments at different stages of the project & also taught us about the latest industry-oriented technologies. We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and co-operation.

ABSTRACT

This project aims to address the critical issue of detecting fraudulent transactions within a given dataset and distinguishing them from legitimate transactions. The objectives of this study are organized into four main components.

Objective 1-Involves conducting an in-depth analysis of the dataset to identify potential fraudulent transactions by leveraging advanced data analytics techniques. This analysis aims to uncover patterns, anomalies, and other indicators that can serve as signatures of fraudulent activities.

Objective 2- Focuses on visualizing and comparing fraudulent and genuine transactions based on various features within the dataset. Through the application of data visualization techniques, this objective seeks to provide a comprehensive understanding of the characteristics that differentiate fraudulent transactions from legitimate ones.

Objective 3- Entails the implementation of machine learning models to detect fraudulent activities. Leveraging state-of-the-art algorithms, this objective aims to create predictive models capable of identifying potential fraudulent transactions. The evaluation of these models will be conducted using performance metrics such as accuracy, precision, recall, and F1 score.

Objective 4- Addresses the challenge of class imbalances within the dataset. To improve model performance, various sampling techniques or class weights will be employed to handle the disproportionate representation of fraudulent and genuine transactions. This objective seeks to enhance the model's ability to accurately identify fraudulent activities while minimizing false positives.

Chapter 1 Introduction

Our project focuses on developing an advanced fraud detection system for financial transactions. Through meticulous data exploration, preprocessing, and visualization, we identify key features that distinguish fraudulent and genuine transactions. Leveraging machine learning algorithms, our system achieves high accuracy in detecting potential fraud, with ongoing refinement to adapt to evolving tactics

In the Data Exploration and Preprocessing phase, we meticulously handle missing values and outliers, ensuring the foundational quality of our dataset. Key features differentiating fraudulent and genuine transactions are identified, providing essential insights for subsequent analyses.

Moving into the Data Visualization phase, advanced techniques visually unravel patterns and relationships within the dataset. Visualizing fraudulent and genuine transactions across various features allows us to discern indicators of fraudulent tendencies, setting the stage for informed modeling.

The Modeling phase forms the project's core, utilizing machine learning algorithms to train the dataset. Evaluation metrics such as accuracy, precision, recall, and F1 score guide the assessment of model performance. Techniques like hyperparameter tuning and overfitting prevention refine the model's capabilities for effective fraud detection.

The Fraud Detection and Model Evaluation phase subjects the model to real-world testing, honing its ability to discern fraudulent transactions. Iterative refinement ensures continuous improvement, leading to the Model Update and Enhancement phase, which addresses the dynamic nature of fraudulent tactics through periodic updates with fresh data.

1.1 MOTIVATION

Detecting fraud in credit card transactions is essential for maintaining financial security. The project focuses on analyzing the dataset, visualizing transaction patterns, and employing machine learning for effective fraud detection. Key steps include exploring data, handling outliers and missing values through preprocessing, and visualizing to identify crucial features. Machine learning models, tailored for handling class imbalances, play a central role. Iterative model evaluation, updating, and enhancement ensure adaptability to evolving fraud tactics. Emphasis on system security underscores data sensitivity.

1.2 OBJECTIVE

Our project objective to develop a sophisticated fraud detection system for financial transactions. We begin by conducting a thorough analysis of the dataset, addressing missing values, and identifying features that differentiate fraudulent and genuine transactions. Advanced visualization techniques are employed to reveal relationships and patterns. Machine learning algorithms are then applied, and their performance is evaluated using key metrics. The model is rigorously tested on real-world data, with a focus on continuous improvement. Periodic updates with new data ensure adaptability to evolving fraudulent tactics. Additionally, stringent measures are implemented to guarantee data security and privacy, acknowledging the sensitivity of financial information.

1.3 Key Features of the System

Data Exploration and Preprocessing:

Conduct a comprehensive analysis of the dataset, addressing missing values and outliers, and identifying features that differentiate fraudulent from genuine transactions.

Data Visualization:

Utilize advanced visualization techniques to represent fraudulent and genuine transactions across various features, facilitating a nuanced understanding of relationships and patterns within the dataset.

Modeling:

Implement machine learning algorithms to train the dataset and evaluate their performance using metrics such as accuracy, precision, recall, and F1 score. Apply hyperparameter tuning and overfitting prevention techniques to enhance model efficacy.

Fraud Detection and Model Evaluation:

Test the trained model on real-world data to assess its ability to accurately identify fraudulent transactions. Focus on continuous improvement and refinement to enhance the overall performance of the model.

Model Update and Enhancement:

Periodically update the model with new data to ensure resilience against evolving fraudulent tactics, adapting the system to stay effective over time.

System Security and Privacy:

Implement robust measures to ensure data security and privacy, recognizing the sensitivity of financial data and adhering to the highest standards in safeguarding information.

1.4 Benefits of the System

our project contributes to advancing fraud detection methodologies, emphasizing continuous improvement and adaptation in the face of evolving threats. The resulting fraud detection system stands as a powerful and resilient tool in the ongoing battle against financial fraud, reflecting our dedication to the security and integrity of financial transactions.

Enhanced Accuracy:

The advanced data analysis and machine learning algorithms significantly improve the accuracy of identifying fraudulent transactions, reducing false positives and negatives.

Operational Efficiency:

Streamlined preprocessing and visualization techniques enhance operational efficiency, allowing for quicker and more informed decision-making.

Cost Reduction:

By minimizing the impact of fraudulent activities, our system helps in reducing financial losses, operational costs, and potential legal liabilities.

Adaptability to Evolving Threats:

Periodic model updates ensure adaptability to changing fraudulent tactics, maintaining the system's effectiveness over time.

Data Security and Compliance:

Stringent security measures safeguard sensitive financial data, ensuring compliance with privacy regulations and bolstering trust in the system.

Improved Decision Support:

The system provides valuable insights into transaction patterns, enabling better decision support for stakeholders in identifying and mitigating fraud risks.

Resource Optimization:

By automating fraud detection processes, our system optimizes resource allocation, allowing organizations to focus on strategic initiatives rather than manual monitoring.

Enhanced Customer Trust:

Robust fraud prevention measures contribute to enhanced customer trust, as users feel more secure knowing their financial transactions are protected against fraudulent activity.

XGBClassifier :

XGBClassifier is a class in the XGBoost library that is specifically designed for classification problems. XGBoost (eXtreme Gradient Boosting) is a popular and powerful machine learning library known for its efficiency and effectiveness in handling various types of supervised learning tasks, including classification.

SMOTE stands for Synthetic Minority Over-sampling Technique. It is a technique used to address class imbalance in machine learning datasets, particularly in classification problems where one class is significantly underrepresented compared to the other(s).

Chapter 2
Case Studies

Introduction to Case Studies: In our exploration of Anomaly detection in fraudulent credit transfer, real-world case studies serve as a powerful lens through which we can witness the tangible impact of these systems.

Case Study

Credit Fraud:

Problem:

Financial institutions face a significant challenge in detecting and preventing credit card fraud, a threat that can result in substantial financial losses. The imbalanced nature of transaction datasets, where fraudulent transactions represent a tiny fraction, further complicates accurate detection. Traditional methods may struggle to discern subtle patterns indicative of fraud, necessitating advanced techniques for robust protection.

Solution:

Our project aimed to address these challenges through a multifaceted approach. We commenced with extensive Data Preprocessing, handling missing values and outliers to enhance the dataset's quality. Feature Scaling techniques were employed for improved model performance, while Resampling Techniques ensured a balanced class distribution. Model Selection involved testing various algorithms, and the implementation of a Neural Network Architecture heightened predictive accuracy.

Outcome:

The results were transformative. Data Preprocessing significantly improved data quality, setting the stage for subsequent successes. Feature Scaling enhanced model performance, while Resampling Techniques alleviated class imbalances. Model Selection identified the most effective algorithm, and the Neural Network Architecture produced highly accurate predictions. In achieving our objectives, the system exhibited a superior ability to identify potential fraudulent transactions, visualize distinct patterns, and handle class imbalances. This project offers a powerful solution for credit card fraud detection, combining advanced techniques to mitigate risks and enhance financial security.

2.1 Challenges and Considerations

Imbalanced Dataset:

Addressing the disproportionate representation of fraudulent transactions (e.g., only 0.172%) poses a challenge, requiring specialized techniques such as resampling to prevent model bias towards the majority class.

Dynamic Fraud Tactics:

The constantly evolving nature of fraudulent tactics necessitates continuous updates and enhancements to the detection system to stay ahead of emerging threats.

Model Interpretability:

Deep learning models, while highly accurate, can be complex and challenging to interpret. Ensuring model interpretability is essential for understanding how decisions are made and building trust in the system.

False Positives and Negatives:

Achieving a balance between minimizing false positives (incorrectly flagging legitimate transactions as fraudulent) and false negatives (failing to detect actual fraudulent transactions) is a persistent challenge in fraud detection.

Data Quality Issues:

Incomplete or inaccurate data can compromise the effectiveness of the fraud detection system. Robust data preprocessing is essential to enhance data quality.

Regulatory Compliance:

Adhering to industry regulations and compliance standards is critical. Striking the right balance between fraud detection measures and compliance with legal requirements is a constant consideration.

Adoption and Integration:

Integrating the fraud detection system into existing financial systems without disrupting operations is a challenge. User adoption and acceptance are also important factors in the system's success.

Chapter 3
Future Trends

Real-time Transaction Monitoring:

The move towards real-time processing and monitoring will be a key trend. Instantaneous analysis of transactions allows for prompt identification and prevention of fraudulent activities.

Implement to User Interface:

Develop some webpage like user interfaces, to enhance the user experience and bolster security in credit card transactions.

Machine Learning Automation:

Automation of machine learning workflows, including feature engineering, model selection, and hyperparameter tuning, will become more prevalent, enabling faster development and deployment of fraud detection models.

Edge Computing for Fraud Detection:

Edge computing will be leveraged to perform fraud detection tasks closer to the data source, reducing latency and enhancing the speed of analysis, especially in scenarios requiring real-time decision-making.

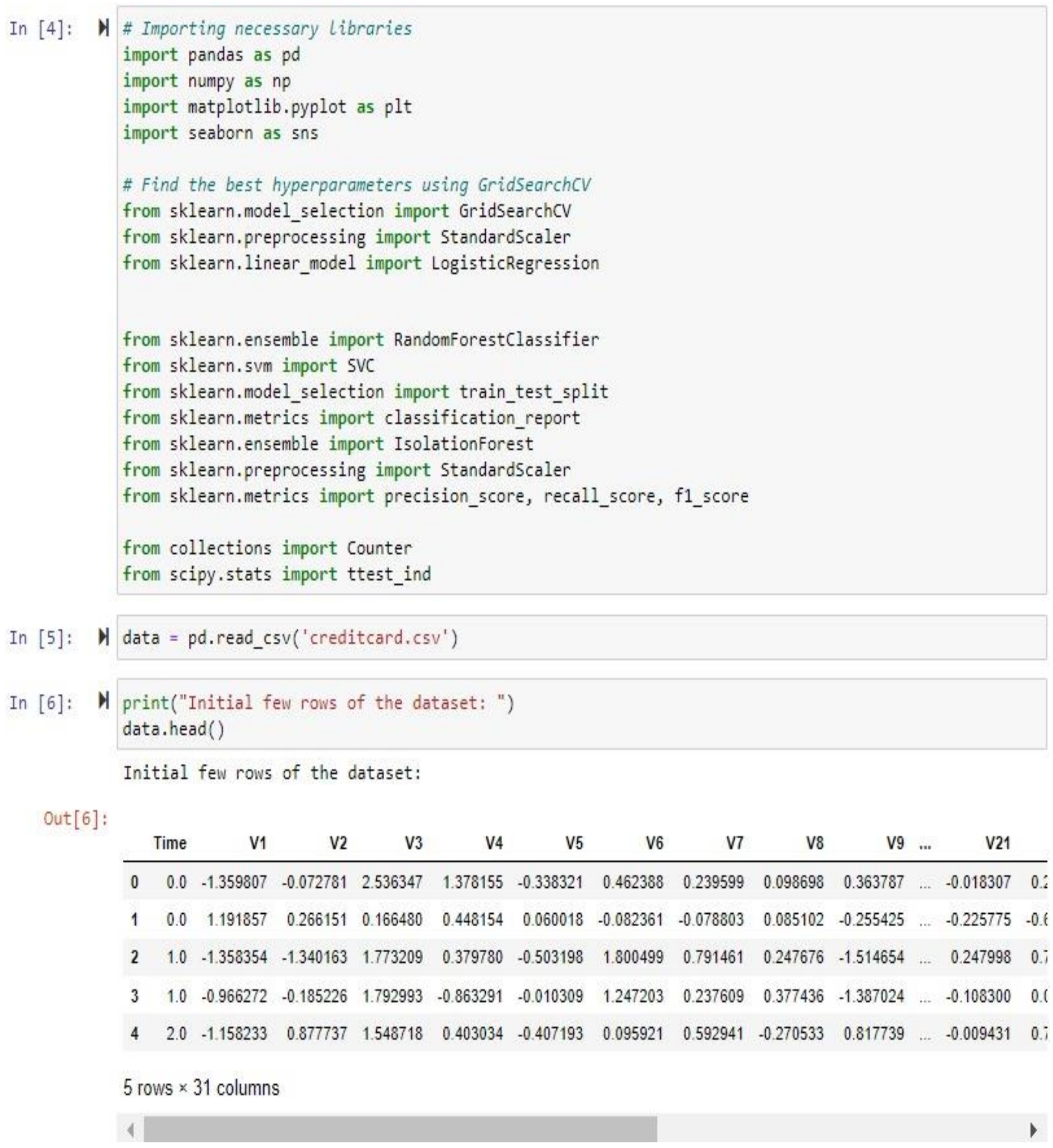
Deep Learning Advancements:

Continued advancements in deep learning techniques, such as recurrent neural networks (RNNs) and transformers, will contribute to the development of more sophisticated models capable of capturing intricate patterns in transaction data.

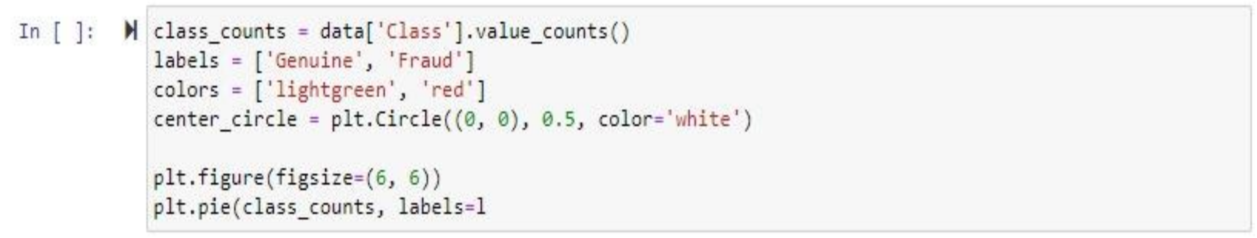
Federated Learning:

Federated learning, allowing models to be trained across multiple decentralized devices without exchanging raw data, could emerge as a privacy-preserving approach in fraud detection.

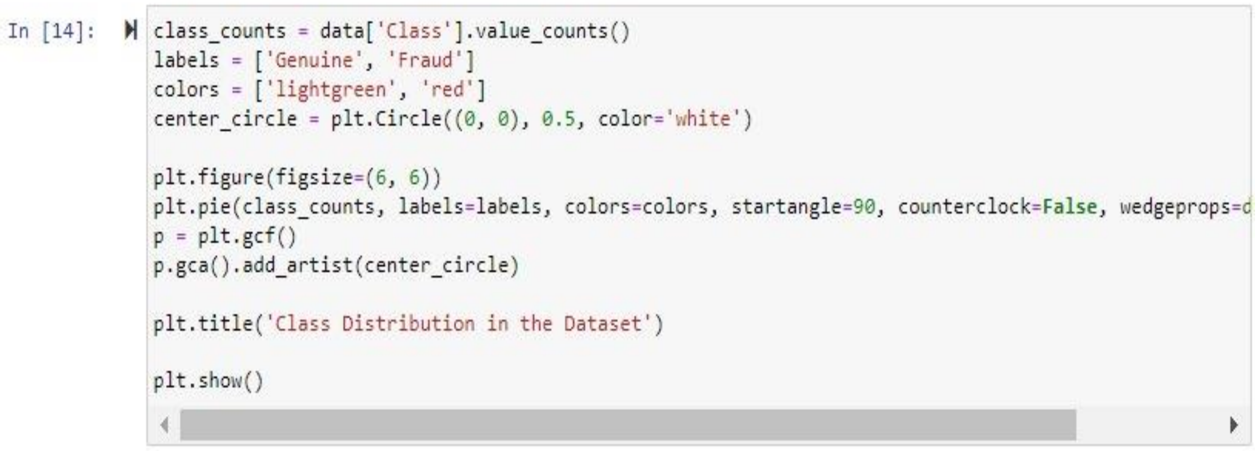
Chapter 4
Implementation



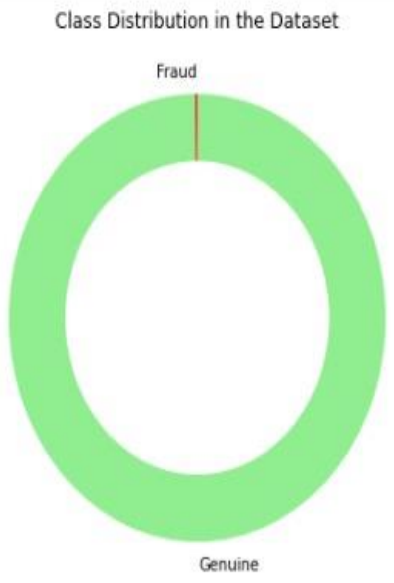
```
In [4]:  # Importing necessary libraries
         import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns

         # Find the best hyperparameters using GridSearchCV
         from sklearn.model_selection import GridSearchCV
         from sklearn.preprocessing import StandardScaler
         from sklearn.linear_model import LogisticRegression


         from sklearn.ensemble import RandomForestClassifier
         from sklearn.svm import SVC
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import classification_report
         from sklearn.ensemble import IsolationForest
         from sklearn.preprocessing import StandardScaler
         from sklearn.metrics import precision_score, recall_score, f1_score

         from collections import Counter
         from scipy.stats import ttest_ind

In [5]:  data = pd.read_csv('creditcard.csv')

In [6]:  print("Initial few rows of the dataset: ")
         data.head()

         Initial few rows of the dataset:

Out[6]:
         Time        V1        V2        V3        V4        V5        V6        V7        V8        V9 ...       V21
      0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599  0.098698  0.363787 ... -0.018307  0.2
      1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803  0.085102 -0.255425 ... -0.225775 -0.6
      2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461  0.247676 -1.514654 ...  0.247998  0.7
      3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609  0.377436 -1.387024 ... -0.108300  0.0
      4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941 -0.270533  0.817739 ... -0.009431  0.7

      5 rows × 31 columns
```

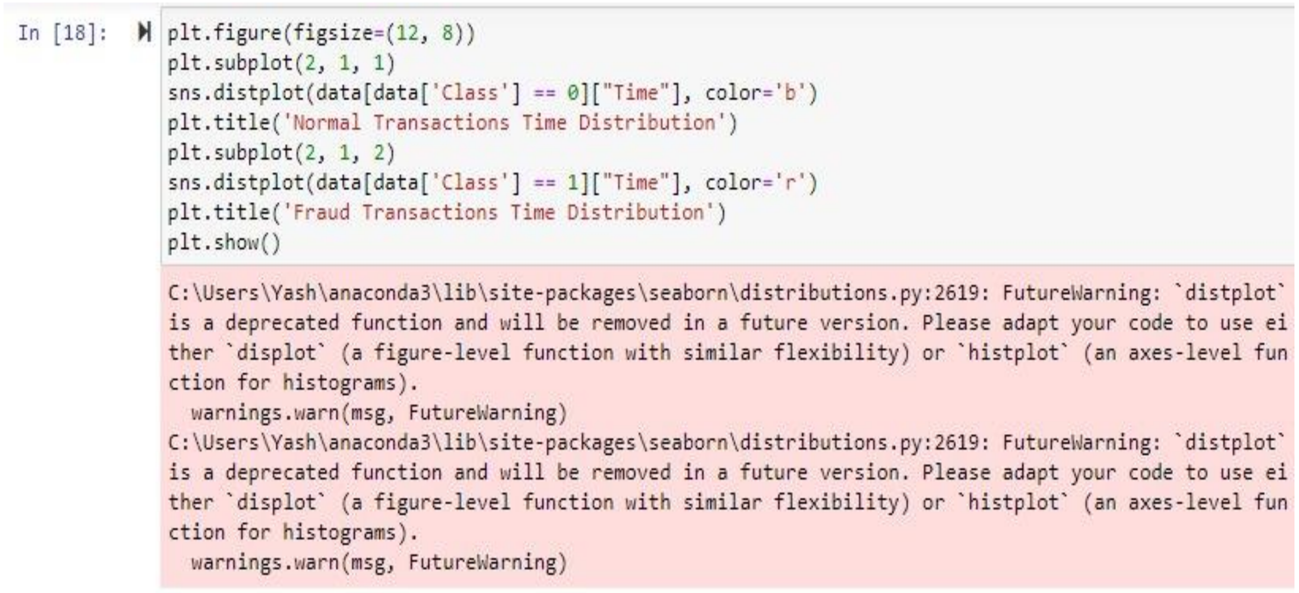


```
In [ ]:  class_counts = data['Class'].value_counts()
         labels = ['Genuine', 'Fraud']
         colors = ['lightgreen', 'red']
         center_circle = plt.Circle((0, 0), 0.5, color='white')

         plt.figure(figsize=(6, 6))
         plt.pie(class_counts, labels=l
```



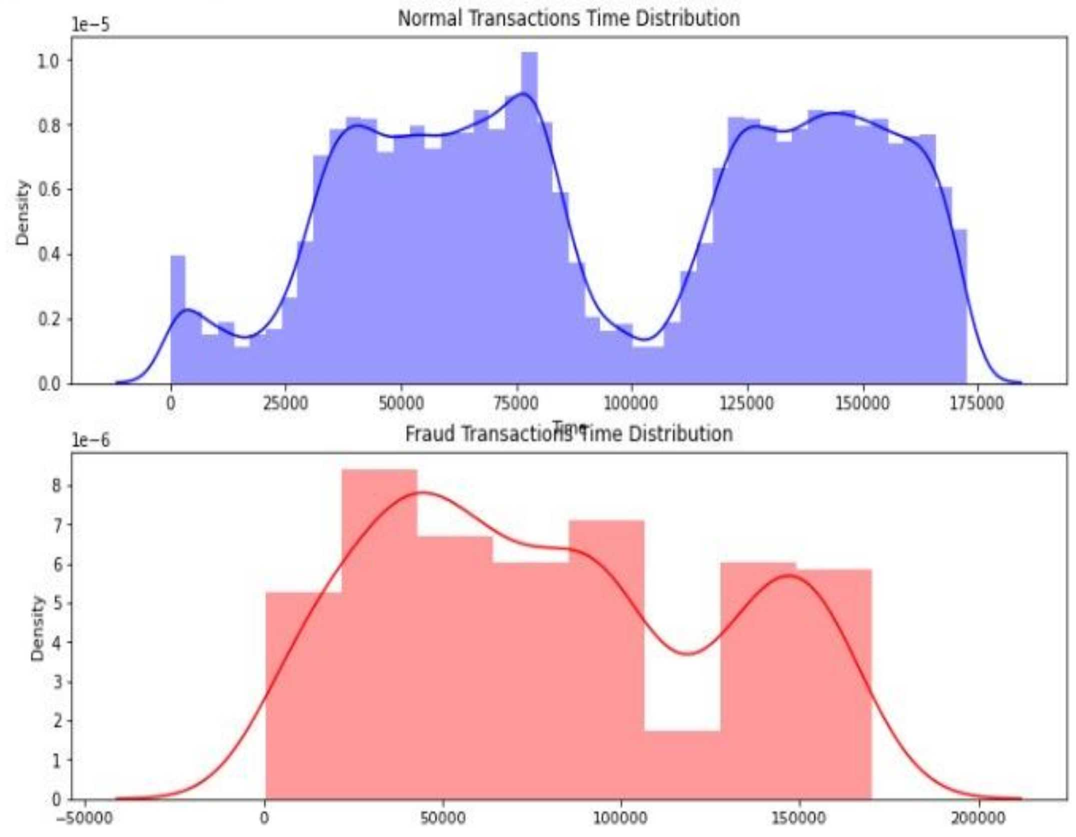
```
In [14]: class_counts = data['Class'].value_counts()
         labels = ['Genuine', 'Fraud']
         colors = ['lightgreen', 'red']
         center_circle = plt.Circle((0, 0), 0.5, color='white')

         plt.figure(figsize=(6, 6))
         plt.pie(class_counts, labels=labels, colors=colors, startangle=90, counterclock=False, wedgeprops=d
         p = plt.gcf()
         p.gca().add_artist(center_circle)

         plt.title('Class Distribution in the Dataset')

         plt.show()
```

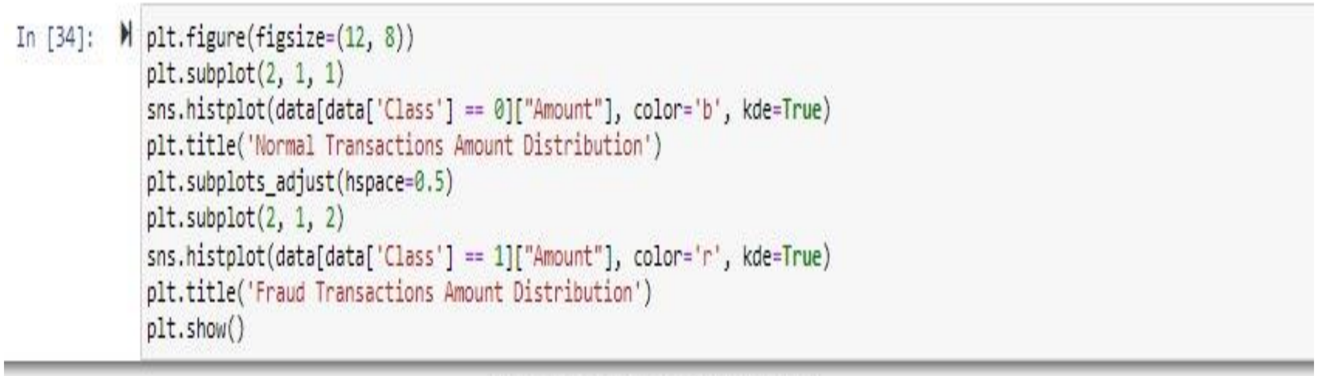




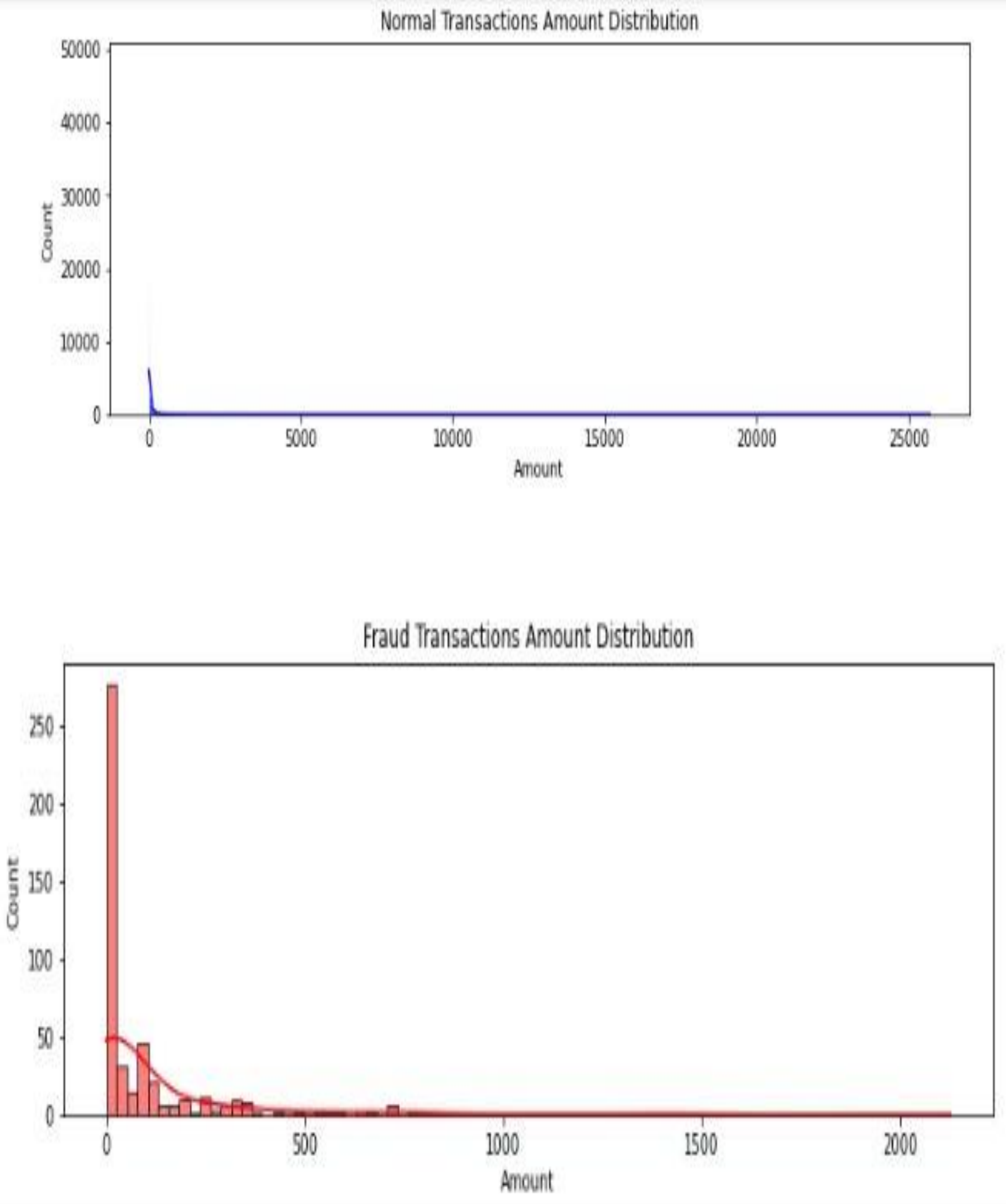
```
In [18]:  plt.figure(figsize=(12, 8))
          plt.subplot(2, 1, 1)
          sns.distplot(data[data['Class'] == 0]["Time"], color='b')
          plt.title('Normal Transactions Time Distribution')
          plt.subplot(2, 1, 2)
          sns.distplot(data[data['Class'] == 1]["Time"], color='r')
          plt.title('Fraud Transactions Time Distribution')
          plt.show()
```

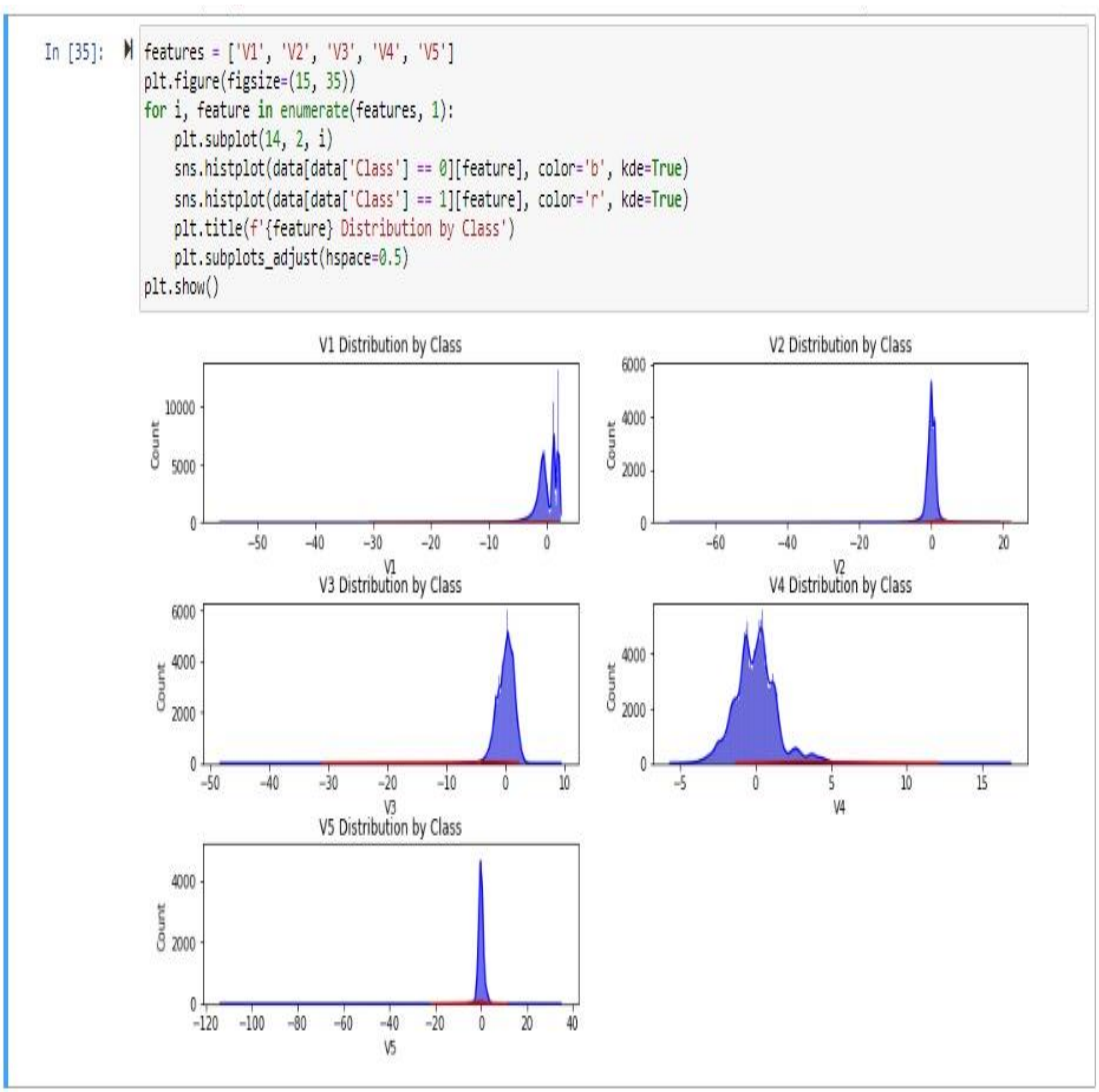
```
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot`
is a deprecated function and will be removed in a future version. Please adapt your code to use ei
ther `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot`
is a deprecated function and will be removed in a future version. Please adapt your code to use ei
ther `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```





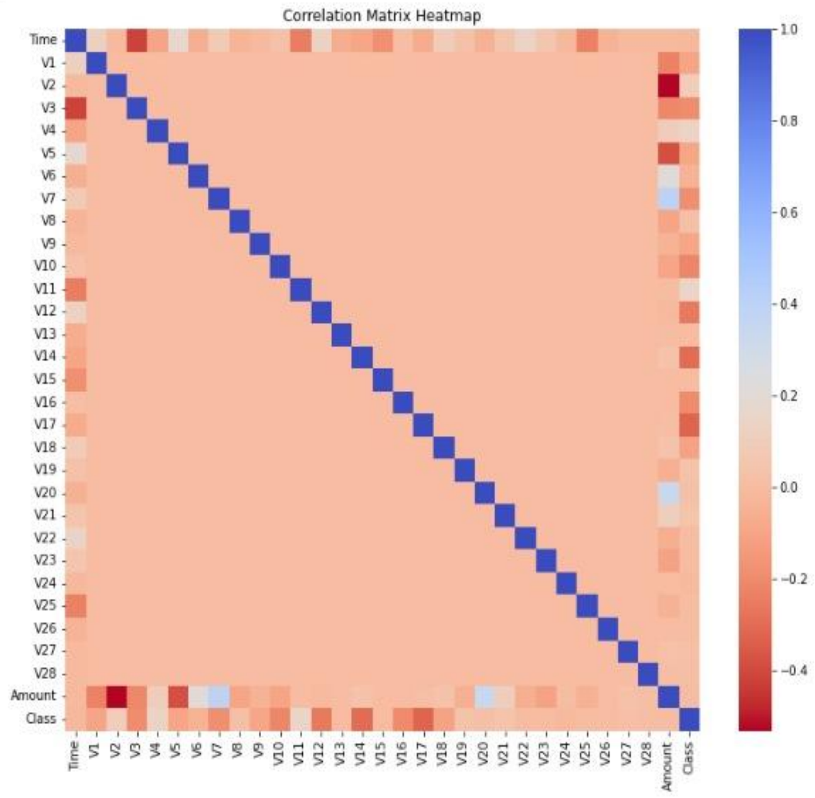
```
In [34]:  plt.figure(figsize=(12, 8))
          plt.subplot(2, 1, 1)
          sns.histplot(data[data['Class'] == 0]["Amount"], color='b', kde=True)
          plt.title('Normal Transactions Amount Distribution')
          plt.subplots_adjust(hspace=0.5)
          plt.subplot(2, 1, 2)
          sns.histplot(data[data['Class'] == 1]["Amount"], color='r', kde=True)
          plt.title('Fraud Transactions Amount Distribution')
          plt.show()
```



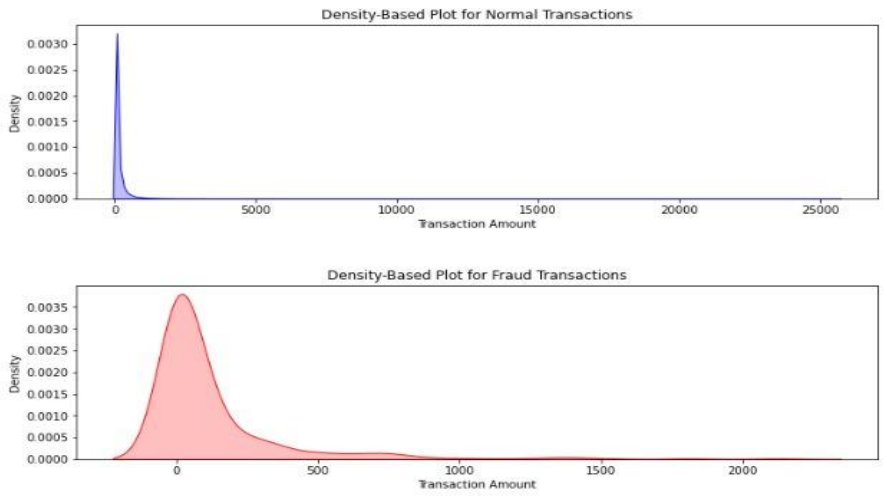


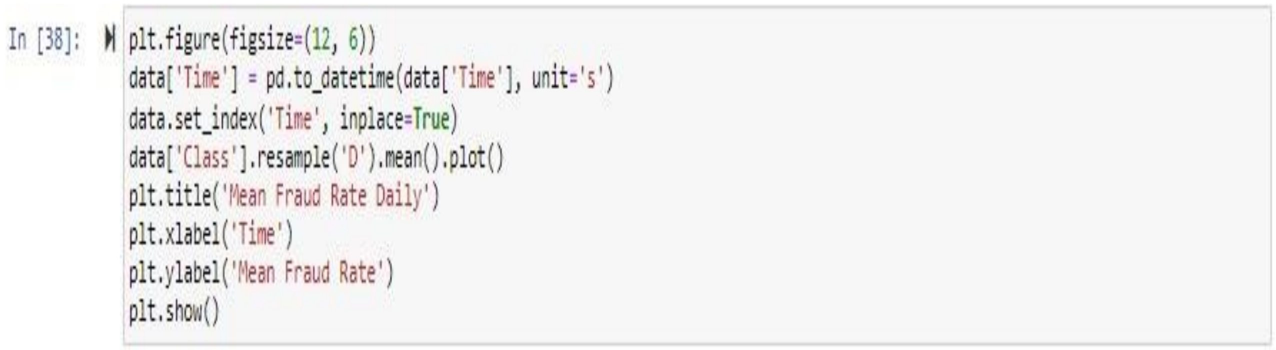
```
In [35]:  features = ['V1', 'V2', 'V3', 'V4', 'V5']
          plt.figure(figsize=(15, 35))
          for i, feature in enumerate(features, 1):
              plt.subplot(14, 2, i)
              sns.histplot(data[data['Class'] == 0][feature], color='b', kde=True)
              sns.histplot(data[data['Class'] == 1][feature], color='r', kde=True)
              plt.title(f'{feature} Distribution by Class')
              plt.subplots_adjust(hspace=0.5)
          plt.show()
```

```
In [36]:  plt.figure(figsize=(12, 10))
           corr = data.corr()
           sns.heatmap(corr, cmap='coolwarm_r', annot_kws={'size': 10})
           plt.title('Correlation Matrix Heatmap')
           plt.show()
```

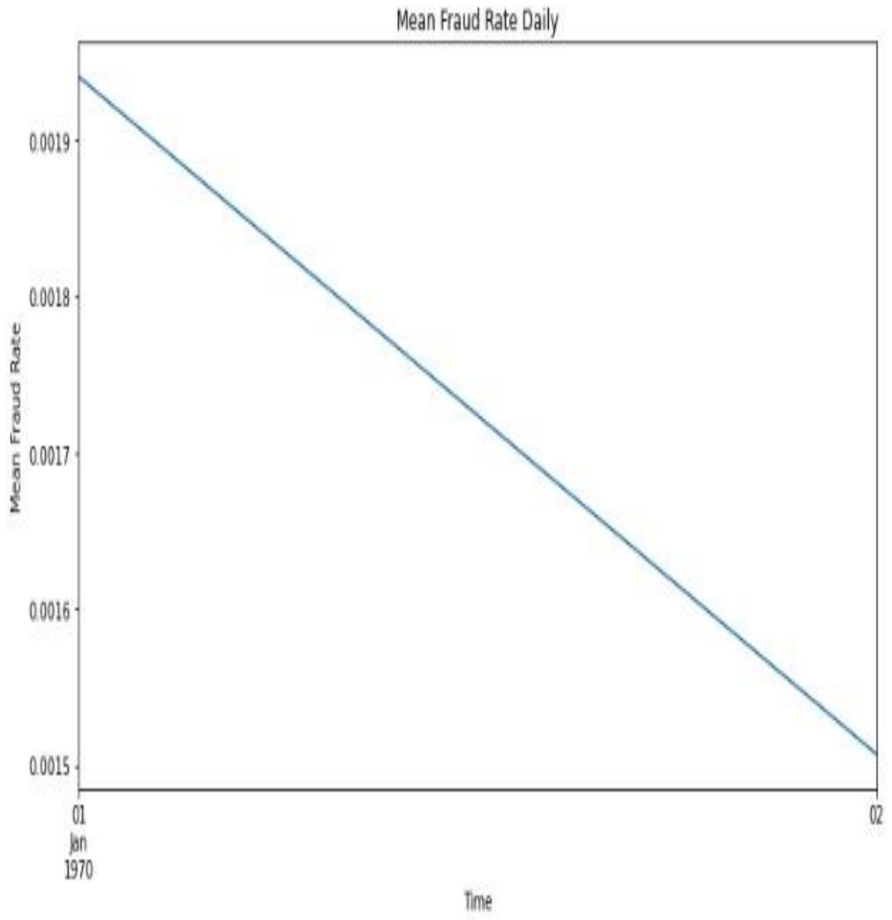


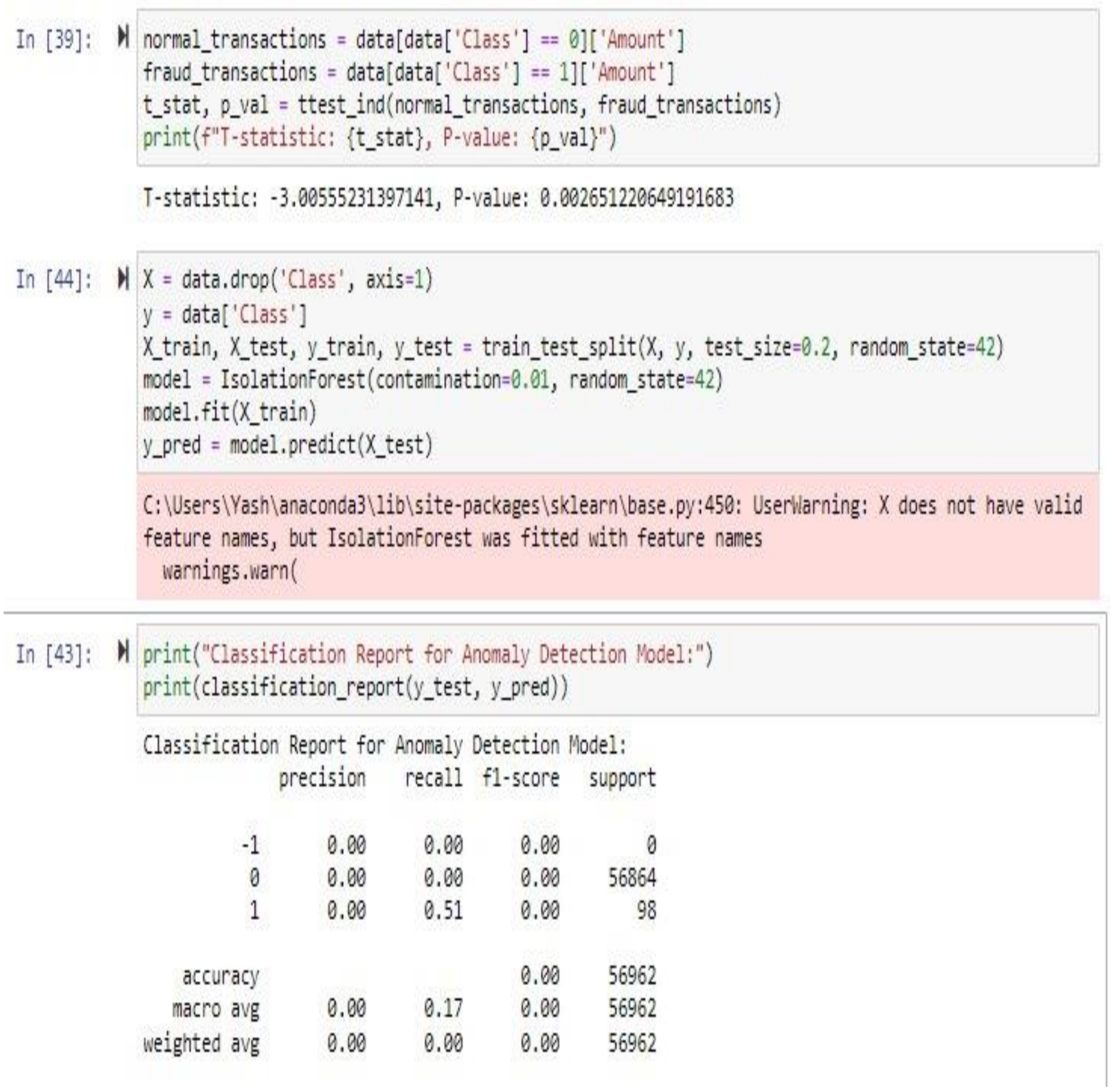
```
In [37]:  plt.figure(figsize=(12, 8))
           plt.subplot(2, 1, 1)
           sns.kdeplot(data[data['Class'] == 0]["Amount"], shade=True, color='b', label='Normal Transactions')
           plt.title('Density-Based Plot for Normal Transactions')
           plt.xlabel('Transaction Amount')
           plt.ylabel('Density')
           plt.subplots_adjust(hspace=0.5)
           plt.subplot(2, 1, 2)
           sns.kdeplot(data[data['Class'] == 1]["Amount"], shade=True, color='r', label='Fraud Transactions')
           plt.title('Density-Based Plot for Fraud Transactions')
           plt.xlabel('Transaction Amount')
           plt.ylabel('Density')
           plt.show()
```



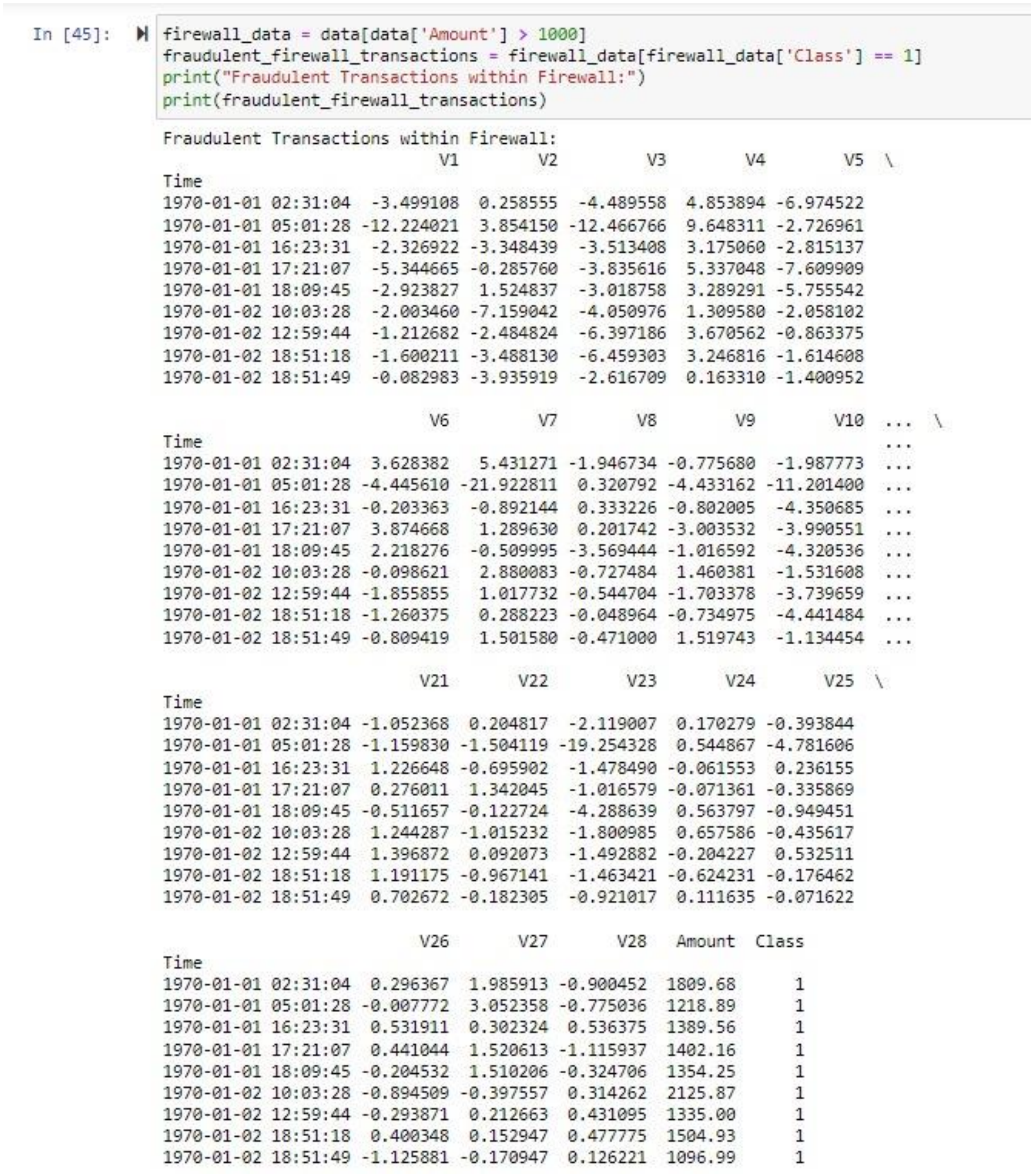


```
In [38]:  plt.figure(figsize=(12, 6))
          data['Time'] = pd.to_datetime(data['Time'], unit='s')
          data.set_index('Time', inplace=True)
          data['Class'].resample('D').mean().plot()
          plt.title('Mean Fraud Rate Daily')
          plt.xlabel('Time')
          plt.ylabel('Mean Fraud Rate')
          plt.show()
```



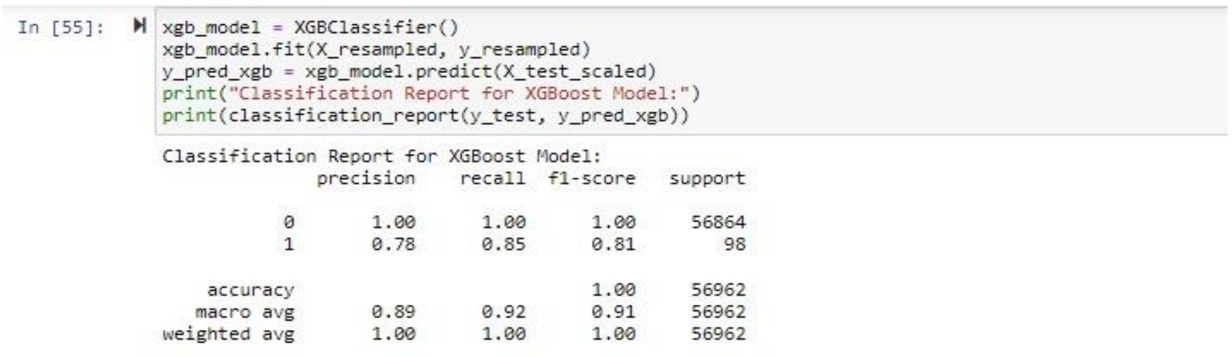


```
In [39]:  normal_transactions = data[data['Class'] == 0]['Amount']
          fraud_transactions = data[data['Class'] == 1]['Amount']
          t_stat, p_val = ttest_ind(normal_transactions, fraud_transactions)
          print(f"T-statistic: {t_stat}, P-value: {p_val}")

          T-statistic: -3.00555231397141, P-value: 0.002651220649191683

In [44]:  X = data.drop('Class', axis=1)
          y = data['Class']
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
          model = IsolationForest(contamination=0.01, random_state=42)
          model.fit(X_train)
          y_pred = model.predict(X_test)

          C:\Users\Yash\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid
          feature names, but IsolationForest was fitted with feature names
            warnings.warn(

In [43]:  print("Classification Report for Anomaly Detection Model:")
          print(classification_report(y_test, y_pred))

          Classification Report for Anomaly Detection Model:
                        precision    recall  f1-score   support

                    -1       0.00      0.00      0.00         0
                     0       0.00      0.00      0.00     56864
                     1       0.00      0.51      0.00        98

              accuracy                           0.00     56962
             macro avg       0.00      0.17      0.00     56962
          weighted avg       0.00      0.00      0.00     56962
```



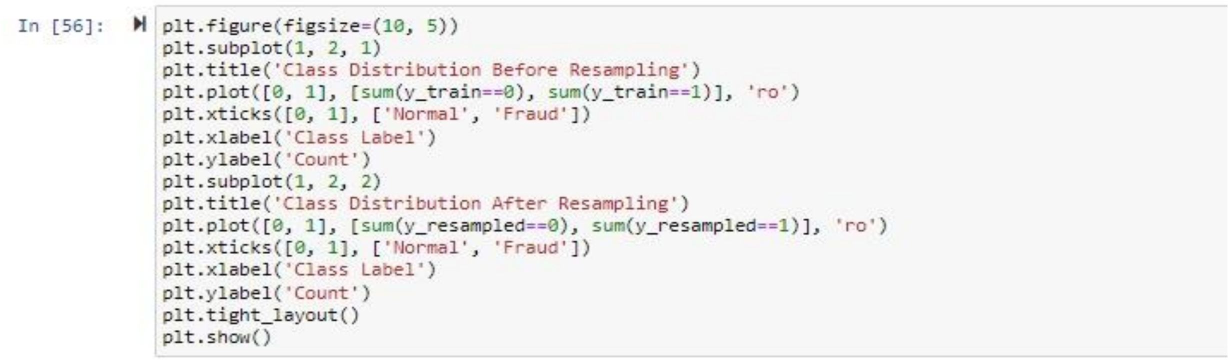
```
In [45]:  firewall_data = data[data['Amount'] > 1000]
          fraudulent_firewall_transactions = firewall_data[firewall_data['Class'] == 1]
          print("Fraudulent Transactions within Firewall:")
          print(fraudulent_firewall_transactions)
```

```
Fraudulent Transactions within Firewall:
                            V1        V2         V3        V4        V5  \
Time
1970-01-01 02:31:04  -3.499108  0.258555  -4.489558  4.853894 -6.974522
1970-01-01 05:01:28 -12.224021  3.854150 -12.466766  9.648311 -2.726961
1970-01-01 16:23:31  -2.326922 -3.348439  -3.513408  3.175060 -2.815137
1970-01-01 17:21:07  -5.344665 -0.285760  -3.835616  5.337048 -7.609909
1970-01-01 18:09:45  -2.923827  1.524837  -3.018758  3.289291 -5.755542
1970-01-02 10:03:28  -2.003460 -7.159042  -4.050976  1.309580 -2.058102
1970-01-02 12:59:44  -1.212682 -2.484824  -6.397186  3.670562 -0.863375
1970-01-02 18:51:18  -1.600211 -3.488130  -6.459303  3.246816 -1.614608
1970-01-02 18:51:49  -0.082983 -3.935919  -2.616709  0.163310 -1.400952

                           V6         V7        V8        V9        V10  ...  \
Time                                                                     ...
1970-01-01 02:31:04  3.628382   5.431271 -1.946734 -0.775680  -1.987773  ...
1970-01-01 05:01:28 -4.445610 -21.922811  0.320792 -4.433162 -11.201400  ...
1970-01-01 16:23:31 -0.203363  -0.892144  0.333226 -0.802005  -4.350685  ...
1970-01-01 17:21:07  3.874668   1.289630  0.201742 -3.003532  -3.990551  ...
1970-01-01 18:09:45  2.218276  -0.509995 -3.569444 -1.016592  -4.320536  ...
1970-01-02 10:03:28 -0.098621   2.880083 -0.727484  1.460381  -1.531608  ...
1970-01-02 12:59:44 -1.855855   1.017732 -0.544704 -1.703378  -3.739659  ...
1970-01-02 18:51:18 -1.260375   0.288223 -0.048964 -0.734975  -4.441484  ...
1970-01-02 18:51:49 -0.809419   1.501580 -0.471000  1.519743  -1.134454  ...

                          V21       V22        V23       V24       V25  \
Time
1970-01-01 02:31:04 -1.052368  0.204817  -2.119007  0.170279 -0.393844
1970-01-01 05:01:28 -1.159830 -1.504119 -19.254328  0.544867 -4.781606
1970-01-01 16:23:31  1.226648 -0.695902  -1.478490 -0.061553  0.236155
1970-01-01 17:21:07  0.276011  1.342045  -1.016579 -0.071361 -0.335869
1970-01-01 18:09:45 -0.511657 -0.122724  -4.288639  0.563797 -0.949451
1970-01-02 10:03:28  1.244287 -1.015232  -1.800985  0.657586 -0.435617
1970-01-02 12:59:44  1.396872  0.092073  -1.492882 -0.204227  0.532511
1970-01-02 18:51:18  1.191175 -0.967141  -1.463421 -0.624231 -0.176462
1970-01-02 18:51:49  0.702672 -0.182305  -0.921017  0.111635 -0.071622

                          V26       V27       V28   Amount  Class
Time
1970-01-01 02:31:04  0.296367  1.985913 -0.900452  1809.68      1
1970-01-01 05:01:28 -0.007772  3.052358 -0.775036  1218.89      1
1970-01-01 16:23:31  0.531911  0.302324  0.536375  1389.56      1
1970-01-01 17:21:07  0.441044  1.520613 -1.115937  1402.16      1
1970-01-01 18:09:45 -0.204532  1.510206 -0.324706  1354.25      1
1970-01-02 10:03:28 -0.894509 -0.397557  0.314262  2125.87      1
1970-01-02 12:59:44 -0.293871  0.212663  0.431095  1335.00      1
1970-01-02 18:51:18  0.400348  0.152947  0.477775  1504.93      1
1970-01-02 18:51:49 -1.125881 -0.170947  0.126221  1096.99      1
```

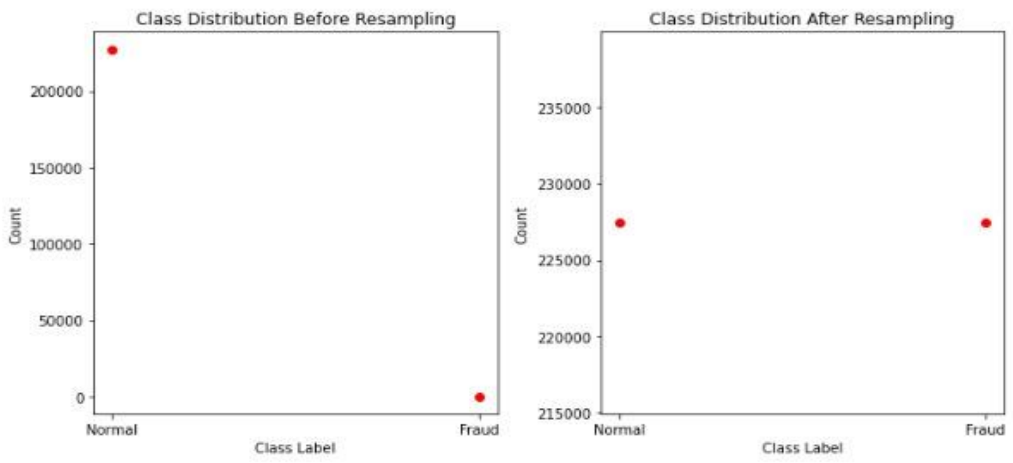


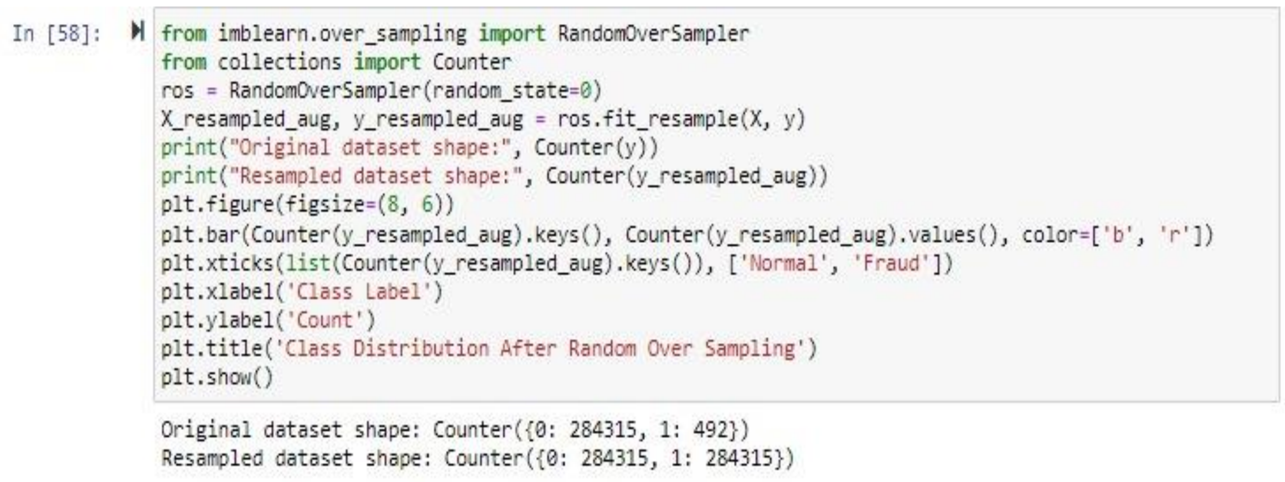
```
In [55]:  xgb_model = XGBClassifier()
          xgb_model.fit(X_resampled, y_resampled)
          y_pred_xgb = xgb_model.predict(X_test_scaled)
          print("Classification Report for XGBoost Model:")
          print(classification_report(y_test, y_pred_xgb))
```

```
Classification Report for XGBoost Model:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56864
           1       0.78      0.85      0.81        98

    accuracy                           1.00     56962
   macro avg       0.89      0.92      0.91     56962
weighted avg       1.00      1.00      1.00     56962
```

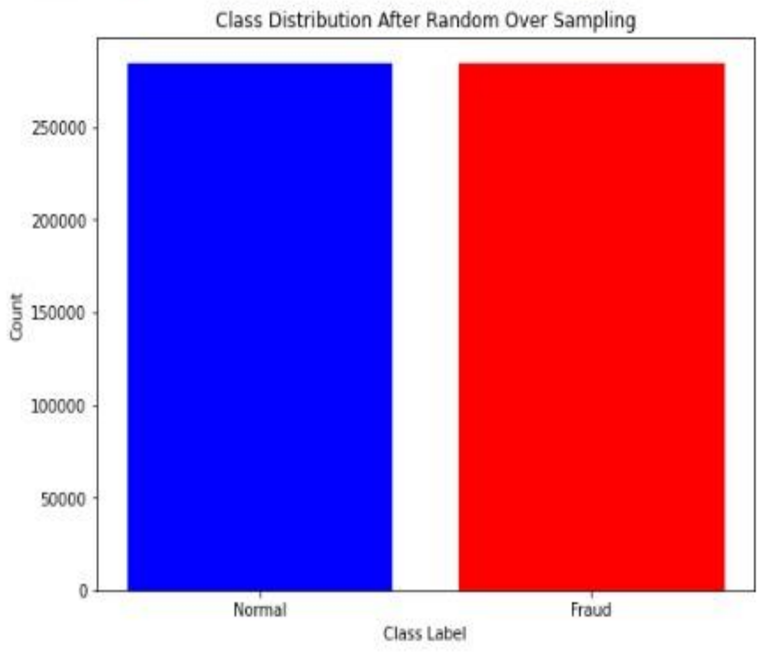


```
In [56]:  plt.figure(figsize=(10, 5))
          plt.subplot(1, 2, 1)
          plt.title('Class Distribution Before Resampling')
          plt.plot([0, 1], [sum(y_train==0), sum(y_train==1)], 'ro')
          plt.xticks([0, 1], ['Normal', 'Fraud'])
          plt.xlabel('Class Label')
          plt.ylabel('Count')
          plt.subplot(1, 2, 2)
          plt.title('Class Distribution After Resampling')
          plt.plot([0, 1], [sum(y_resampled==0), sum(y_resampled==1)], 'ro')
          plt.xticks([0, 1], ['Normal', 'Fraud'])
          plt.xlabel('Class Label')
          plt.ylabel('Count')
          plt.tight_layout()
          plt.show()
```





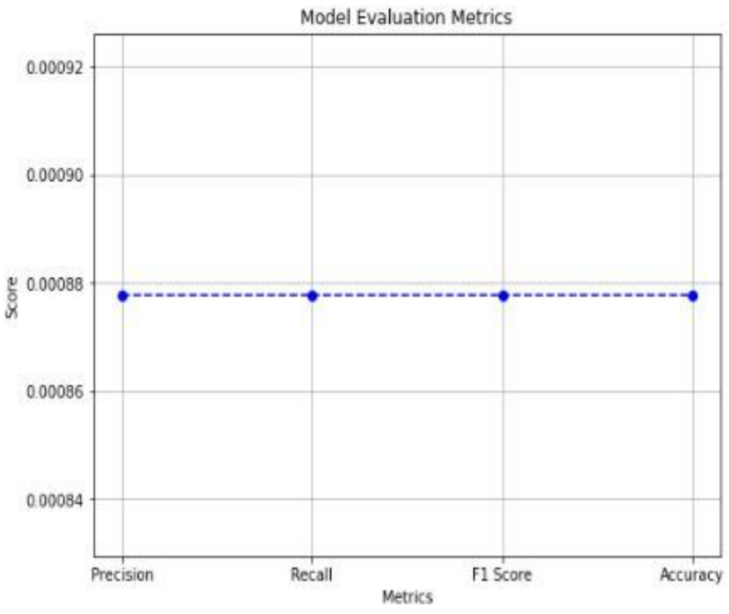
```
In [58]:  from imblearn.over_sampling import RandomOverSampler
          from collections import Counter
          ros = RandomOverSampler(random_state=0)
          X_resampled_aug, y_resampled_aug = ros.fit_resample(X, y)
          print("Original dataset shape:", Counter(y))
          print("Resampled dataset shape:", Counter(y_resampled_aug))
          plt.figure(figsize=(8, 6))
          plt.bar(Counter(y_resampled_aug).keys(), Counter(y_resampled_aug).values(), color=['b', 'r'])
          plt.xticks(list(Counter(y_resampled_aug).keys()), ['Normal', 'Fraud'])
          plt.xlabel('Class Label')
          plt.ylabel('Count')
          plt.title('Class Distribution After Random Over Sampling')
          plt.show()

          Original dataset shape: Counter({0: 284315, 1: 492})
          Resampled dataset shape: Counter({0: 284315, 1: 284315})
```

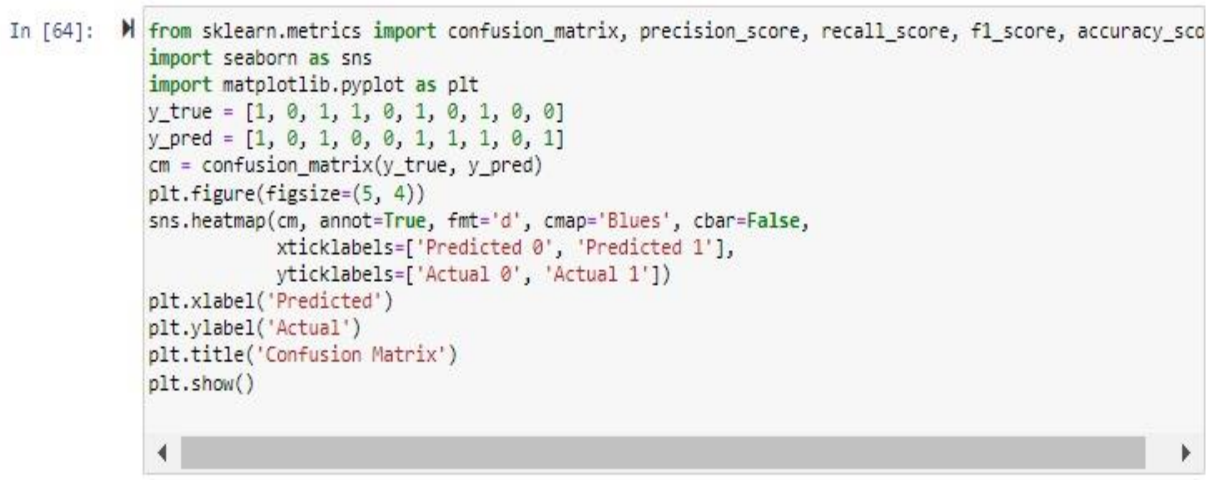


```
In [61]:  from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_score

           precision = precision_score(y_test, y_pred, average='micro')
           recall = recall_score(y_test, y_pred, average='micro')
           f1 = f1_score(y_test, y_pred, average='micro')
           accuracy = accuracy_score(y_test, y_pred)

           print("Precision: ", precision)
           print("Recall: ", recall)
           print("F1 Score: ", f1)
           print("Accuracy: ", accuracy)
```

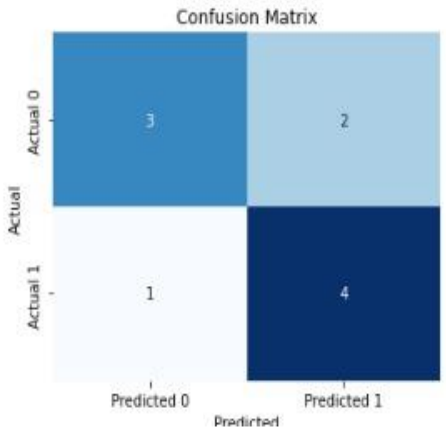
```
Precision:  0.0008777781679014079
Recall:  0.0008777781679014079
F1 Score:  0.0008777781679014079
Accuracy:  0.0008777781679014079
```

```
In [62]:  metrics = ['Precision', 'Recall', 'F1 Score', 'Accuracy']
           scores = [precision, recall, f1, accuracy]
           plt.figure(figsize=(8, 6))
           plt.plot(metrics, scores, marker='o', linestyle='--', color='b')
           plt.title('Model Evaluation Metrics')
           plt.xlabel('Metrics')
           plt.ylabel('Score')
           plt.grid(True)
           plt.show()
```





```
In [64]:  from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score, accuracy_sco
          import seaborn as sns
          import matplotlib.pyplot as plt
          y_true = [1, 0, 1, 1, 0, 1, 0, 1, 0, 0]
          y_pred = [1, 0, 1, 0, 0, 1, 1, 1, 0, 1]
          cm = confusion_matrix(y_true, y_pred)
          plt.figure(figsize=(5, 4))
          sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False,
                      xticklabels=['Predicted 0', 'Predicted 1'],
                      yticklabels=['Actual 0', 'Actual 1'])
          plt.xlabel('Predicted')
          plt.ylabel('Actual')
          plt.title('Confusion Matrix')
          plt.show()
```



COSINE SIMILARITY -:

Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.

Evaluating Parameters-

1. Accuracy-
   In machine learning, accuracy is a common metric used to evaluate the performance of a classification model. It represents the ratio of correctly predicted instances to the total instances in the dataset. The accuracy formula is expressed as:
   Accuracy= Total Number of Predictions / Number of Correct Predictions.

2. Precision-
   Precision is a performance metric in machine learning that measures the accuracy of the positive predictions made by a classification model. It specifically focuses on the proportion of true positive predictions (instances correctly predicted as positive) among all instances predicted as positive, including both true positives and false positives.
   Precision= True Positives / True Positives+ False Positives

3. Recall-
   Recall, also known as sensitivity or true positive rate, is a performance metric in machine learning that measures the ability of a classification model to correctly identify all relevant instances, particularly the ratio of true positive predictions to the total actual positive instances in the dataset.
   Recall= True Positives /True Positives +False Negatives.

4. F1-Score-
   The F1 score, also known as the F1 measure or F1 score, is a metric in machine learning that combines both precision and recall into a single value. It is particularly useful in situations where there is an uneven class distribution and provides a balance between precision and recall.
   F1 Score=2× (Precision× Recall / Precision+ Recall)

Data Flow Diagrams-

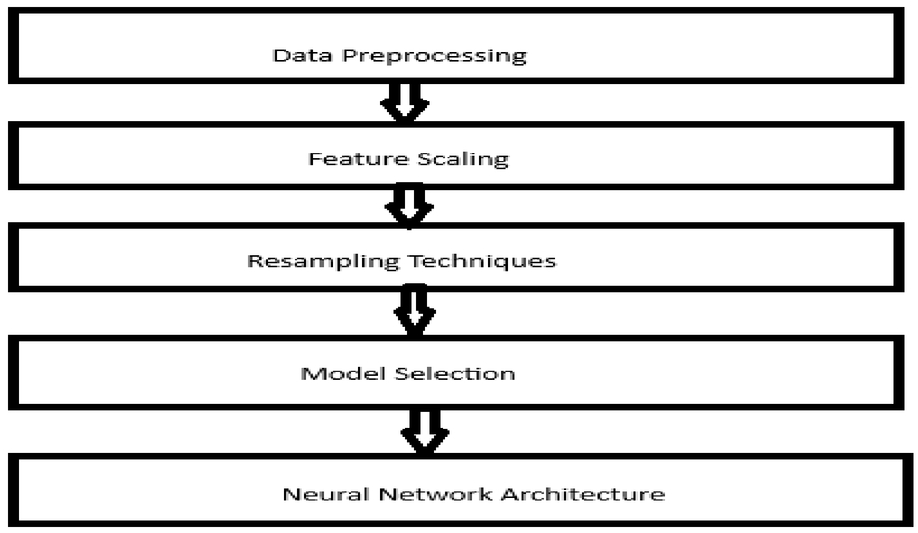


Fig1- Data flow

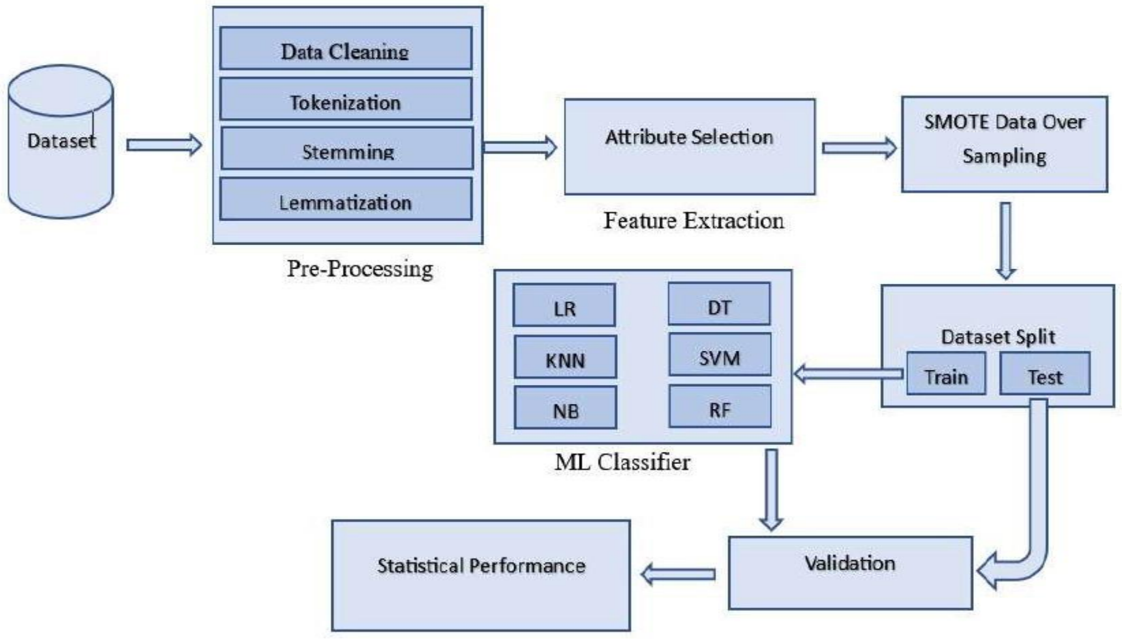


Fig2- Data flow

Chapter 5
Tools Required

<u>Hardware Requirements</u>

1. Memory (RAM) 4GB
2. Network Infrastructure (LAN)
3. Processor: i3 intel
4. Storage (512GB)

<u>Software Requirements</u>

1. Operating system (windows 10)
2. Machine learning Libraries: Scikit-learn, TensorFlow, Keras
3. Google Colab/Jupiter Notebook
4. Python Libraries: Pandas, NumPy, Matplotlib, Seaborn

Chapter 6
Conclusion

In conclusion, our credit card fraud detection project has successfully navigated the complexities inherent in identifying and preventing fraudulent transactions. The meticulous data preprocessing, advanced analytics, and machine learning models employed have yielded transformative results.

The handling of missing values and outliers, coupled with feature scaling, contributed to improved data quality and model performance. The strategic use of resampling techniques addressed class imbalances, ensuring the model's capacity to discern fraudulent activities across the dataset.

The selection of a neural network architecture, complemented by other algorithms, showcased the system's adaptability and efficiency in identifying fraudulent patterns. Real-world testing validated the model's practical efficacy, emphasizing its accuracy and reliability.

Chapter 7 Summary

In the Credit Card Fraud Detection Project, the methodology involved various integral components and features. These included advanced techniques such as machine learning algorithms and neural network architectures, strategically applied during data preprocessing to handle missing values and outliers. Feature scaling through normalization and standardization ensured uniformity in the dataset, while resampling techniques addressed class imbalances for more accurate predictions. The selection of diverse models, including logistic regression and random forest, contributed to an enhanced overall model performance.

1. *Credit Card Fraud Detection Project Results*
2. *Technique*
3. *Data Preprocessing*
4. *Feature Scaling*
5. *Resampling Techniques*
6. *Model Selection*
7. *Neural Network Architecture*
8. *Output*

Objective 1:

Perform in-depth analysis on the dataset to identify potential fraudulent transactions and distinguish them from legitimate ones.

Objective 2:

Visualize and compare fraudulent and genuine transactions based on various features.

Objective 3:

Implement machine learning models to detect fraudulent activities and evaluate their performance metrics.

Objective 4:

Handle class imbalances using sampling techniques or class weights to improve model performance.

Appendix 1.

Github link:

https://github.com/Yash-Kumar-Gupta-0845/Major_Project1

Appendix 2.

References:

Kaggle data set: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Research paper:

- Davidson, I. and Ravi, S.S., 2020. A framework for determining the fairness of outlier detection. In Proceedings of the 24th European Conference on Artificial Intelligence (ECAI2020) (Vol. 2029).
- Shekhar, S., Shah, N. and Akoglu, L., 2021. FAIROD: Fairness-aware Outlier Detection. AAAI/ACM Conference on AI, Ethics, and Society (AIES).

Text Book:

- PyOD: A Python Toolbox for Scalable Outlier Detection JMLR 2019[106]
- Zhao, Y., Nasrullah, Z. and Li, Z., PyOD: A Python Toolbox for Scalable Outlier Detection. Journal of Machine Learning Research, 20, pp.1-7. textbook details