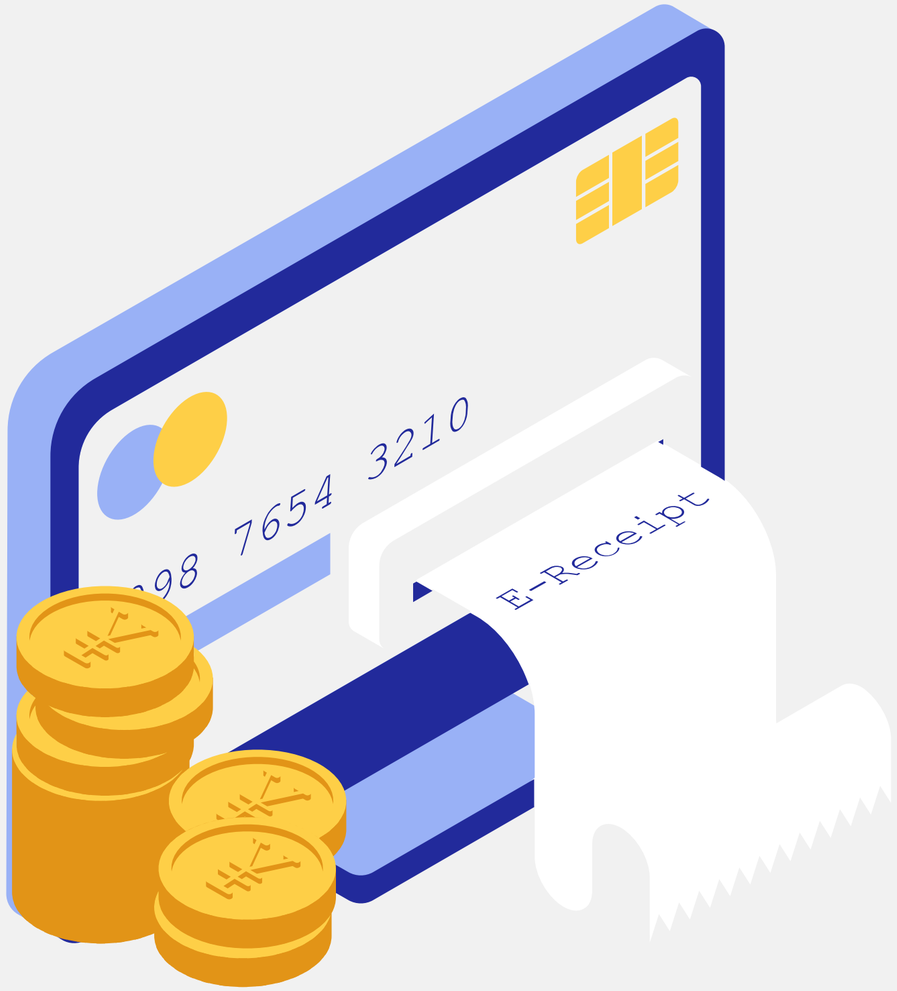


AI

Fraud Detection

Our Team:

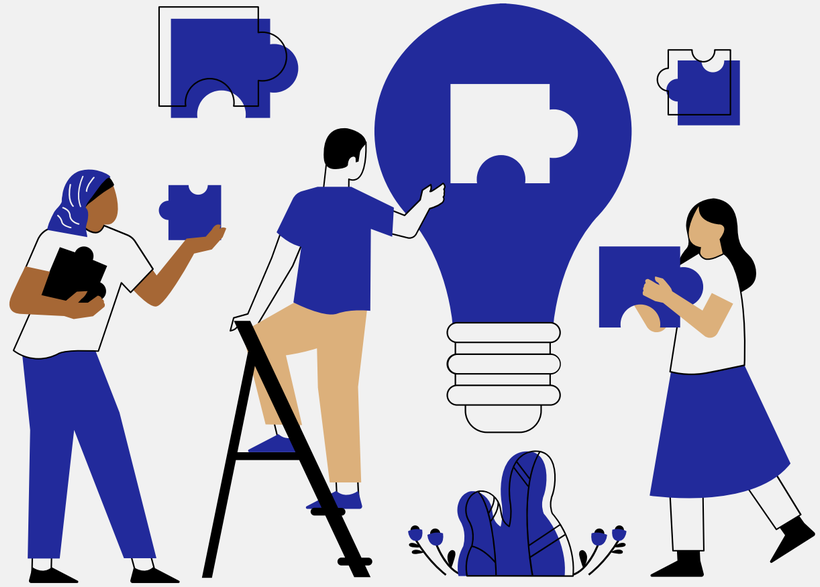
- Omar Wael Elsharkawy
- Khaled Ahmed Eldefry
- Omar Eid Awad
- Muhamed hamada Muhamed
- SarahMohamed Elghazali

Introduction

Credit card fraud is a growing concern in the banking and financial sectors, posing significant threats to individuals, businesses, and the overall trust in digital payment systems. Fraudulent activities result in substantial financial losses and disrupt the smooth functioning of payment infrastructures. Detecting fraud in real-time has become essential to prevent financial damages and maintain customer confidence. However, the inherent challenge lies in the highly imbalanced nature of the data, where fraudulent transactions represent only a small fraction of the total transactions. This imbalance complicates the process of accurate fraud detection.

Problem Statement

- Financial losses and loss of trust caused by undetected fraudulent transactions.
- The challenge of detecting fraud due to the imbalance between fraudulent and legitimate transactions in datasets.
- Inefficiency and inaccuracies in traditional fraud detection methods, especially in real-time scenarios.
- The need for scalable, automated solutions to handle the growing volume of digital payment transactions.



Objectives

The primary objectives of this project are:

1. Data Exploration and Preprocessing:
    - Understand and preprocess the dataset, handling missing values, scaling numerical features, and addressing class imbalances.
2. Model Development:
    - Develop a predictive model using machine learning algorithms to classify credit card transactions as fraud or not fraud.
    - Apply various resampling techniques, like RandomUnderSampler, to address class imbalance.
3. Model Evaluation:
    - Evaluate the model using relevant metrics such as accuracy, precision, recall, F1-score, and ROC-AUC, considering the class imbalance.
    - Visualize results through confusion matrices, ROC curves, and classification reports for deeper analysis.
4. Optimization:
    - Optimize the model's hyperparameters using GridSearchCV to achieve the best performance.
5. Deployment (optional for real-time systems):
    - Discuss and prototype how the model can be deployed in a real-time system to identify fraudulent transactions instantly.

Imported Libraries

We are importing various Python libraries that help us with data analysis, machine learning, and model evaluation. Here's what each one does in simpler terms:

1. pandas (import pandas as pd): This library helps us handle and analyze data, especially structured data like tables (e.g., CSV files).
2. numpy (import numpy as np): Used for numerical operations and handling arrays or large datasets efficiently.
3. matplotlib and seaborn (import matplotlib.pyplot as plt, import seaborn as sns): These libraries are used for visualizing data in charts and graphs, making it easier to understand patterns and relationships.
4. sklearn.model_selection (train_test_split, GridSearchCV): Helps in splitting data into training and testing sets and tuning models for optimal performance.
5. sklearn.linear_model, sklearn.neighbors, sklearn.naive_bayes, sklearn.svm, sklearn.tree, sklearn.ensemble: These import different machine learning algorithms (models), such as:
    - Logistic Regression (for binary outcomes),
    - K-Nearest Neighbors (KNN) (classifying based on nearby data points),
    - Naive Bayes (based on probabilities),
    - Support Vector Classifier (SVC) (used for classification by finding the best boundary),
    - Decision Trees (modeling decisions and their possible consequences),
    - Random Forests (using multiple decision trees for better prediction).
6. sklearn.preprocessing (MinMaxScaler, StandardScaler): These help scale or normalize data so that all features (e.g., amounts, time) have the same importance when feeding into models.
7. sklearn.feature_selection (SelectKBest, chi2): Helps choose the most important features (variables) that contribute to detecting fraud, improving model efficiency.
8. sklearn.metrics (accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confusion_matrix): These metrics help us evaluate how well our models are performing after they've been trained, focusing on things like accuracy, precision, and recall.
9. imblearn.over_sampling (SMOTE): This is a technique to balance data, especially when there are fewer fraudulent transactions than normal ones, making it easier for the model to learn.
10. collections.Counter: Helps count occurrences of items in a dataset, like how often fraud happens compared to normal transactions.

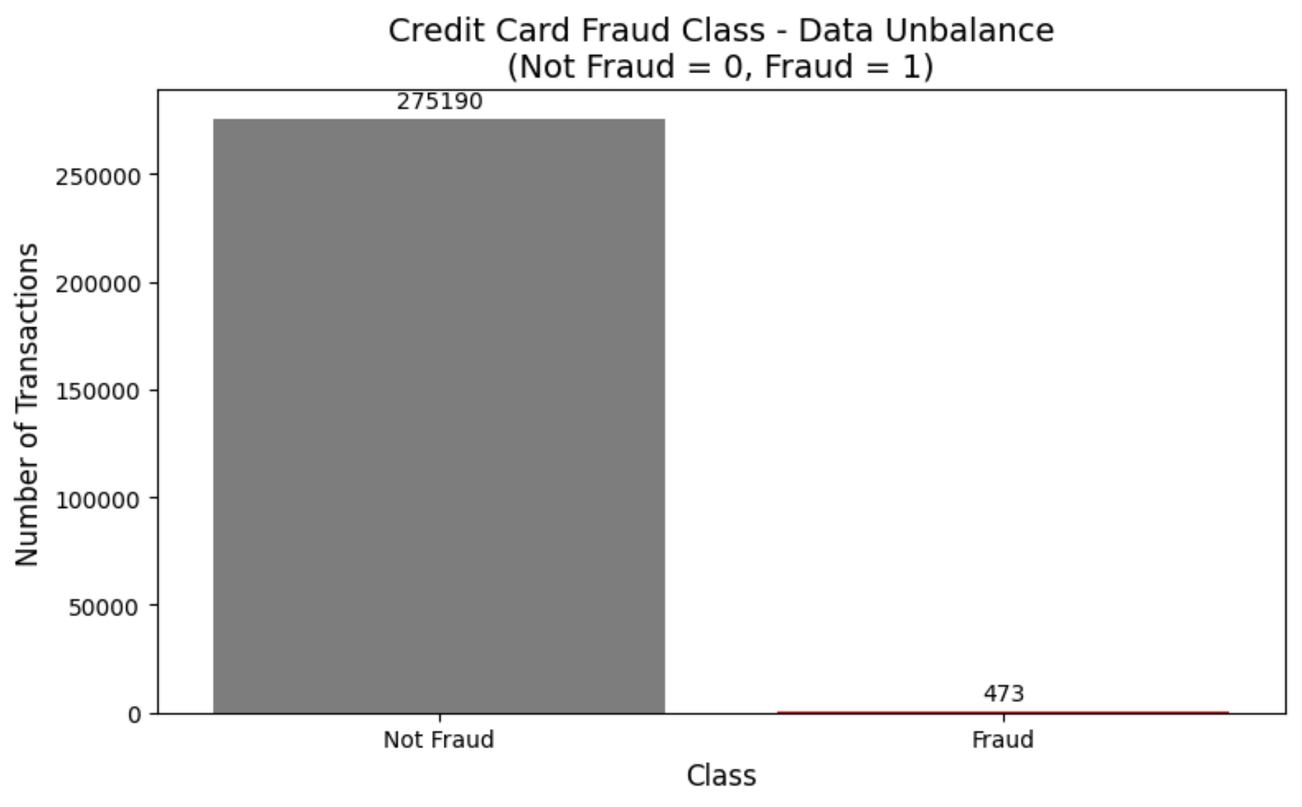
Understanding the Data

The table contains transaction information used to detect fraud:

1. Time: The time of the transaction.
2. V1 to V28: Various features representing different characteristics of the transaction, created through data processing.
3. Amount: The amount of money in the transaction.
4. Class: The target label:
    - 0 means the transaction is legitimate.
    - 1 means the transaction is fraudulent.

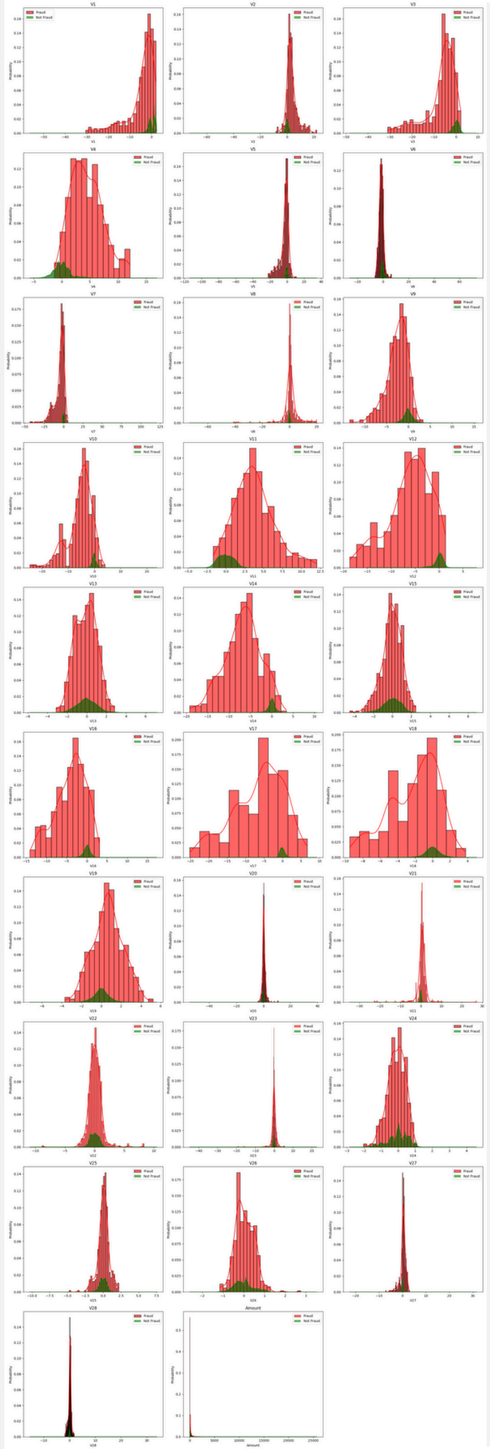
In short, this data helps us identify patterns in transactions and predict if a new transaction is fraudulent or not.

Visualizing Class Distribution:



This code generates a bar plot to visualize the imbalance between fraudulent and non-fraudulent transactions in the dataset:

1. Data Preparation:
    - It counts how many fraudulent (Class = 1) and non-fraudulent (Class = 0) transactions are present.
    - The count is stored in a DataFrame for easy plotting.
2. Plotting:
    - A bar plot is created using Seaborn where:
        - "Not Fraud" (Class = 0) is displayed in gray.
        - "Fraud" (Class = 1) is displayed in red.
    - The plot title explains the context, and the axes are labeled for clarity.
3. Display Values:
    - The exact count of transactions is displayed above each bar for a clearer understanding of the distribution.



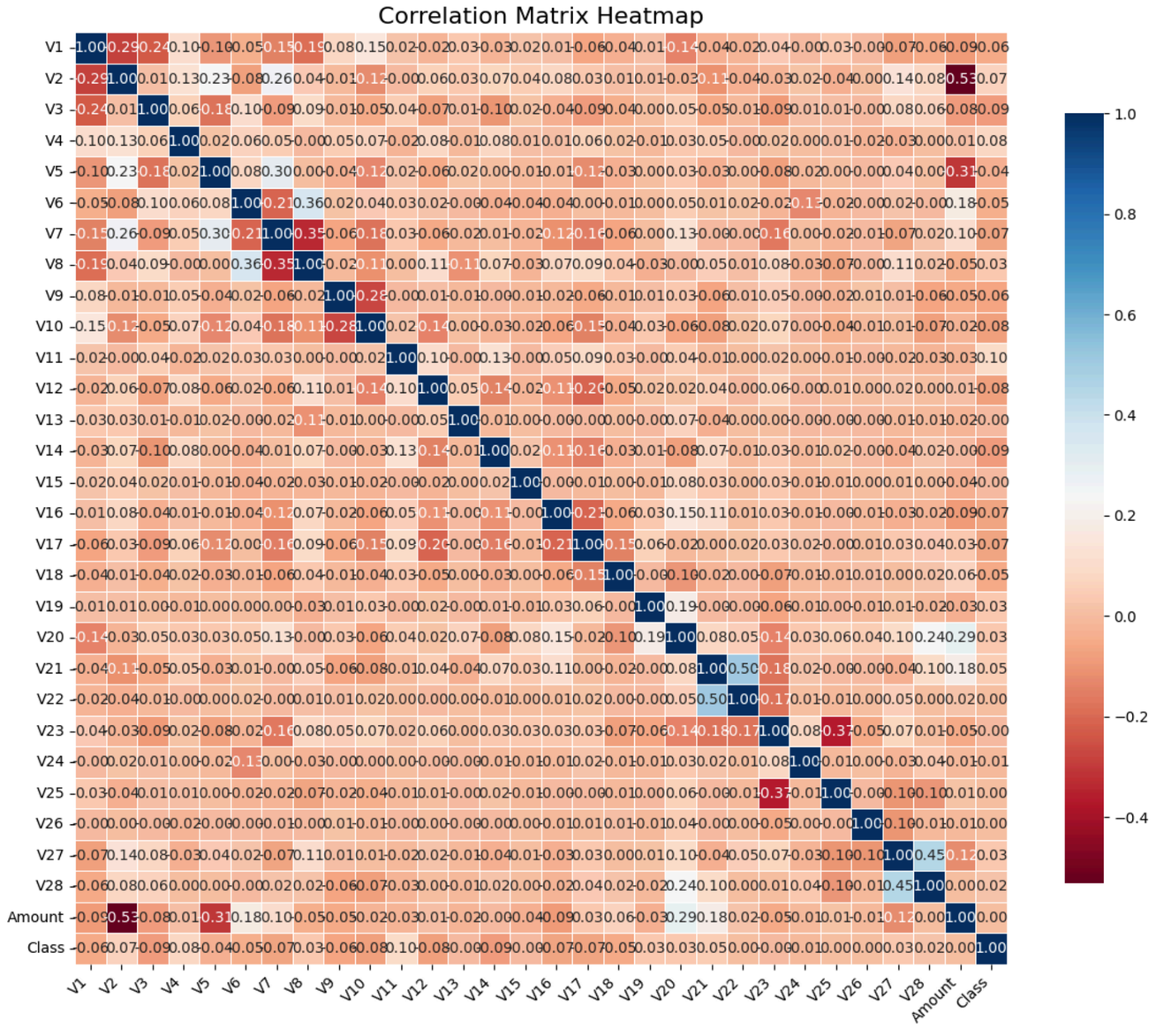
Visualizing Features Distribution for Fraud and Non-Fraud Transactions:

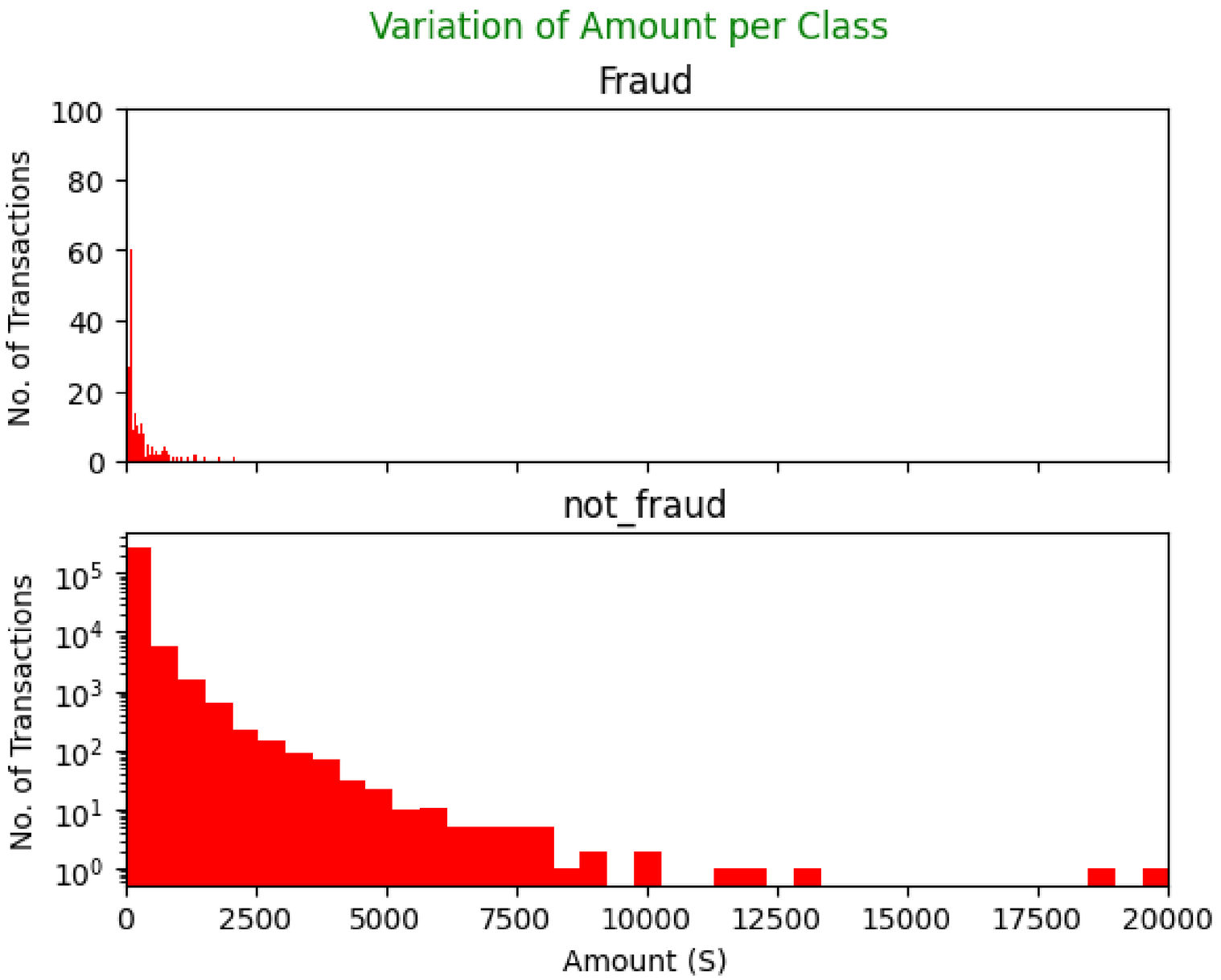
This code generates histograms for each feature in the dataset, comparing the distribution of values for fraudulent (Class = 1) and non-fraudulent (Class = 0) transactions.

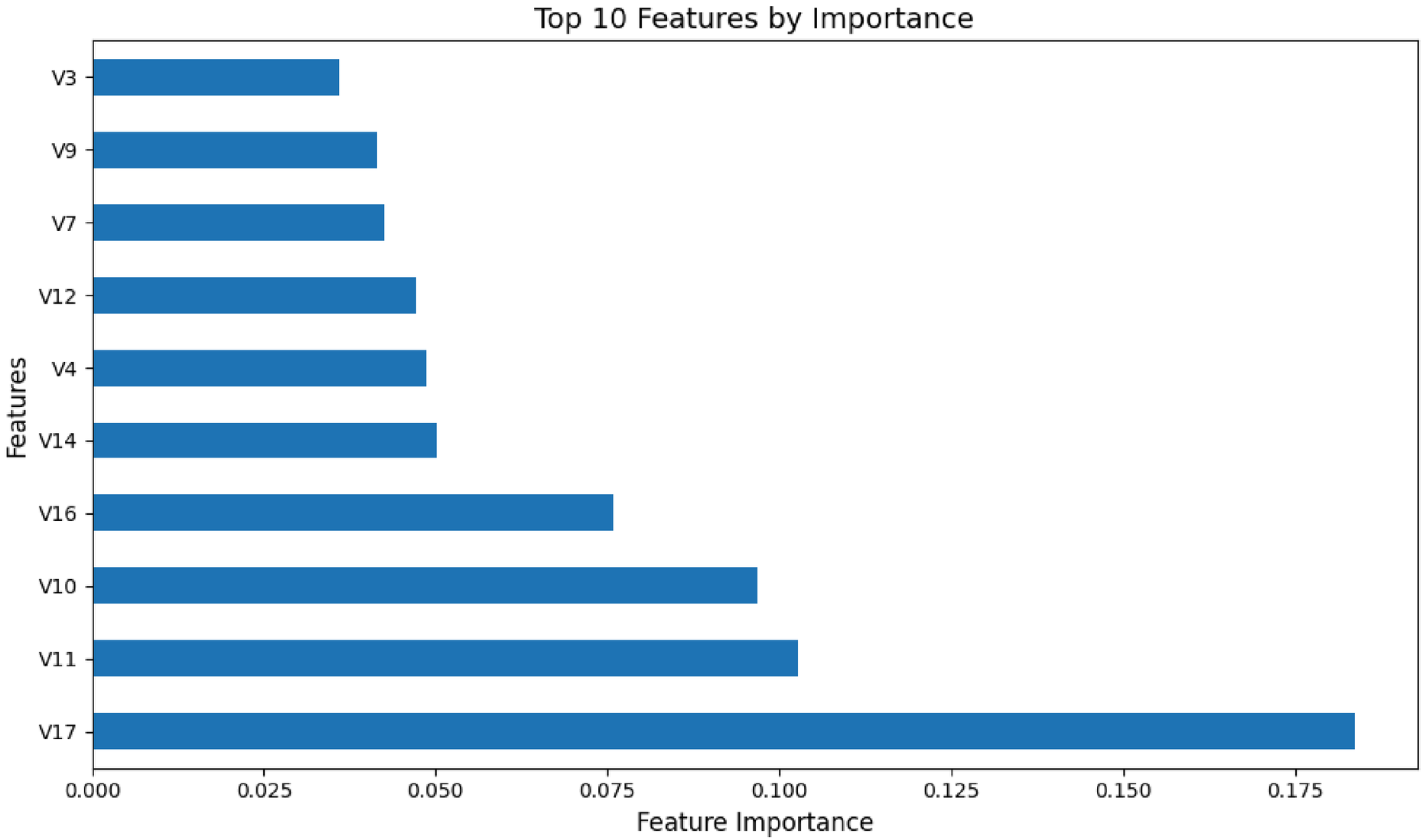
1. Separate Fraud and Non-Fraud Data:
    - The data is divided into two subsets: fraud (where Class = 1) and not_fraud (where Class = 0).
2. Plotting:
    - For each feature (except the Class column), histograms are plotted side by side for both fraudulent and non-fraudulent transactions:
        - Fraud: Red color.
        - Non-Fraud: Green color.
    - The kernel density estimation (KDE) is also plotted on top of the histogram to give a smoother distribution curve.
3. Subplots:
    - A grid of 10 rows and 3 columns is used to display the histograms for each feature, making the visualization clearer and easier to compare.

Correlation Heatmap redundancy.

Purpose: Identify relationships between features to avoid redundancy.



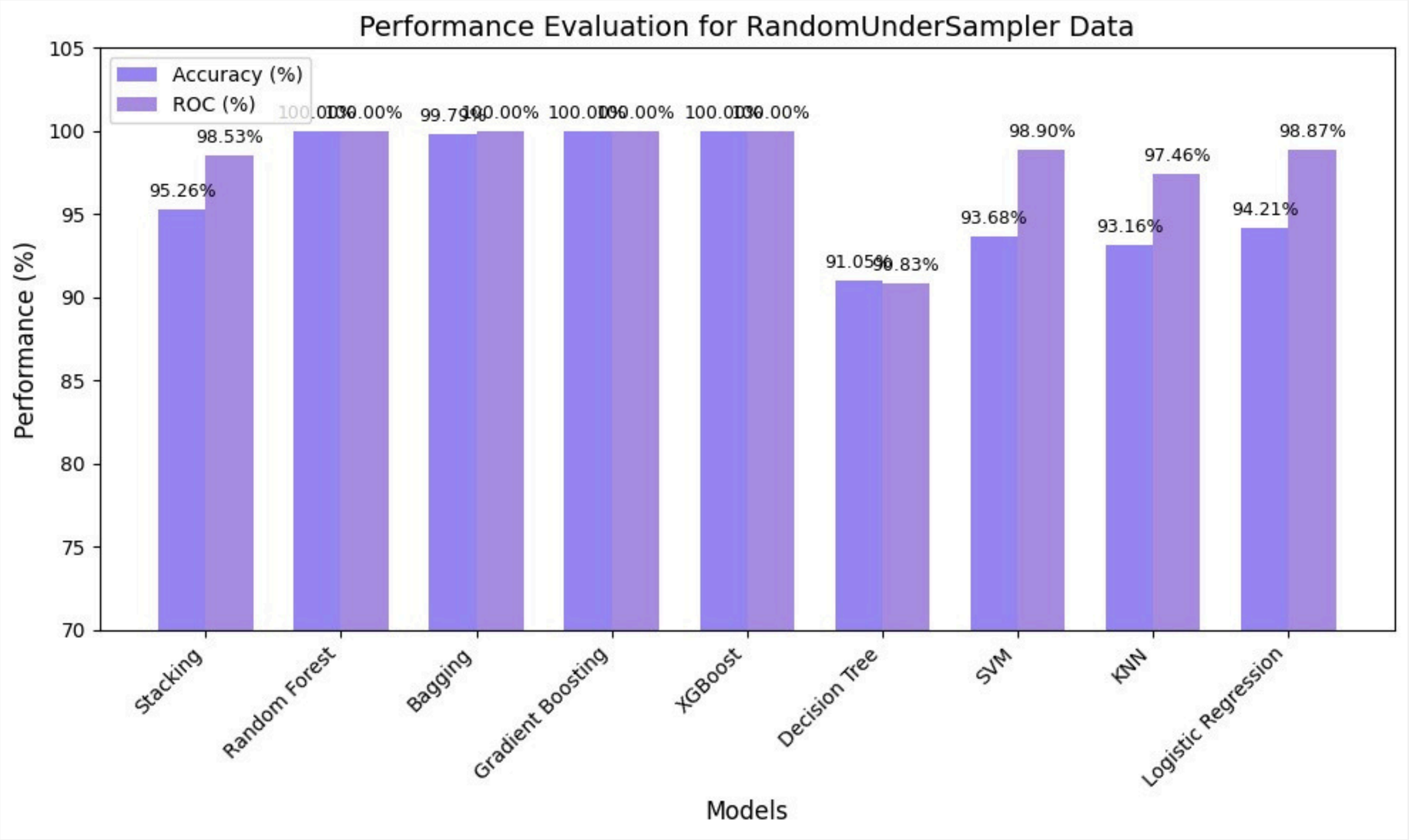

Variation of Amount per Class
Fraud
No. of Transactions
100
80
60
40
20
0
not_fraud
No. of Transactions
10^5
10^4
10^3
10^2
10^1
10^0
0
2500
5000
7500
10000
12500
15000
17500
20000
Amount (S)


Top 10 Features by Importance
Features
V3
V9
V7
V12
V4
V14
V16
V10
V11
V17
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
Feature Importance

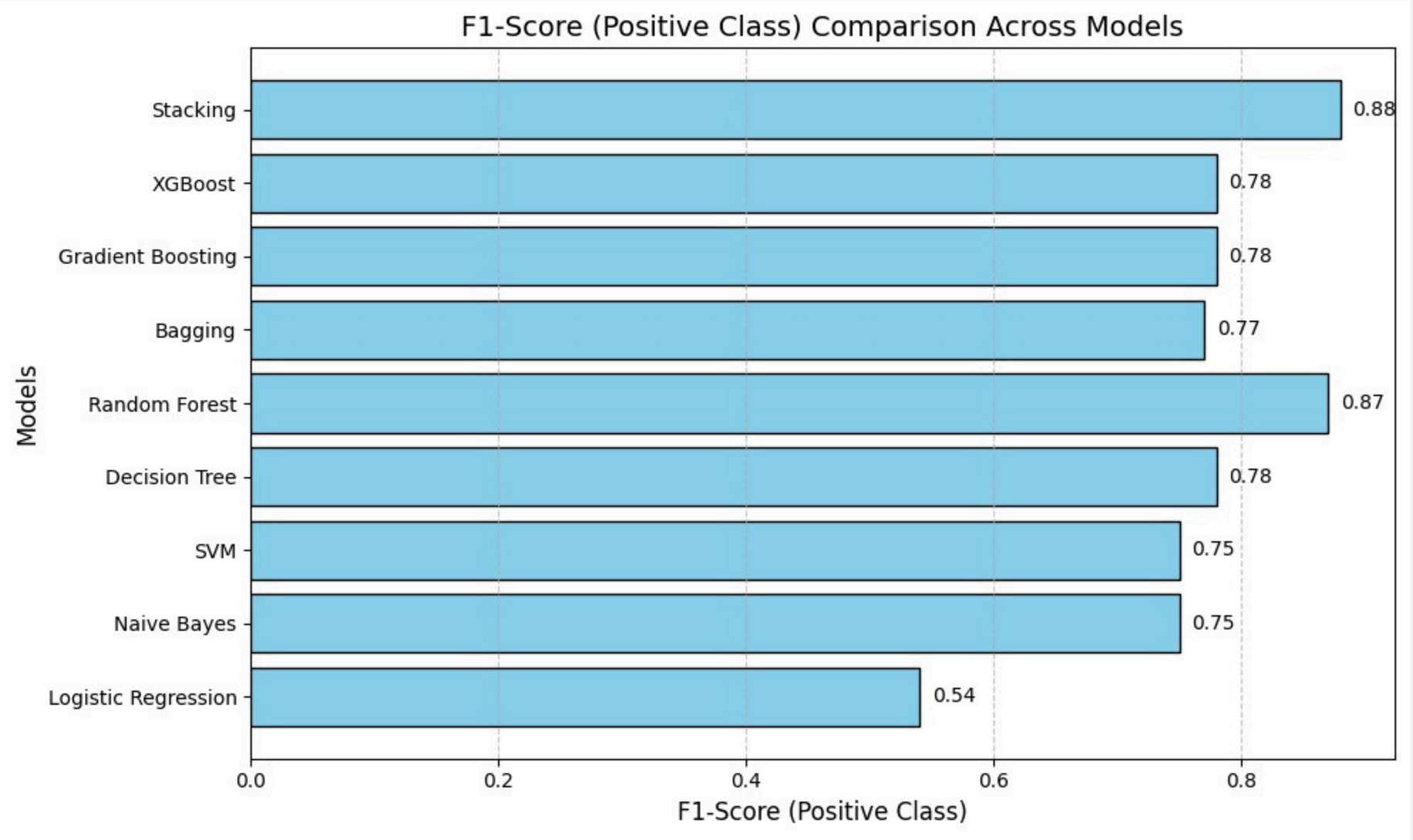
Machine Learning algorithms

- Stacking
- Random Forest Classifier
- Bagging
- Gradient Boosting Classifier
- XGBoost
- Decision Tree
- SVM
- KNN
- Logistic Regression Classifier

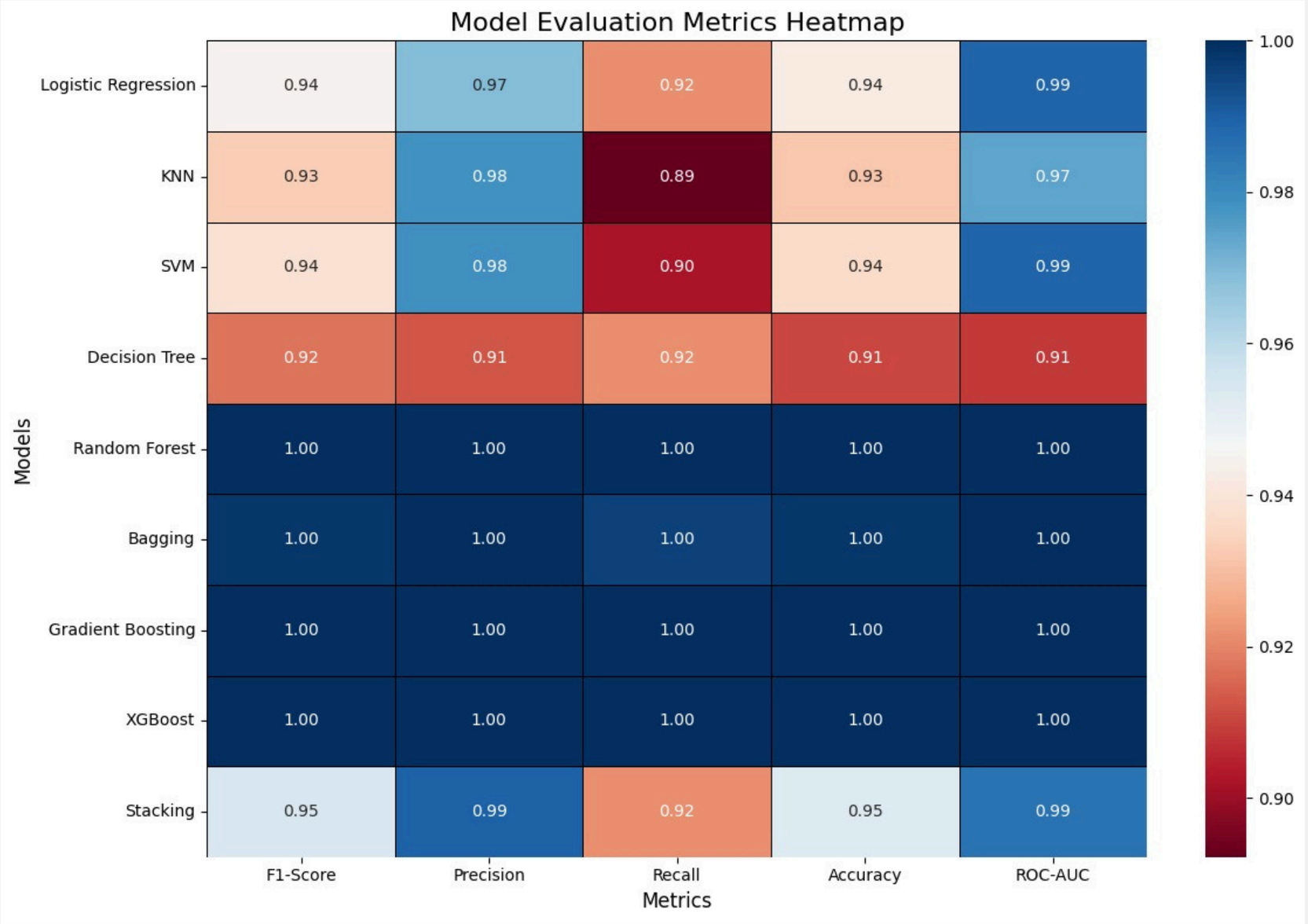
Results


Performance Evaluation for RandomUnderSampler Data
Accuracy (%)
ROC (%)
Performance (%)
105
100
95
90
85
80
75
70
95.26%
98.53%
100.00%
100.00%
99.79%
100.00%
100.00%
100.00%
100.00%
100.00%
91.05%
90.83%
93.68%
98.90%
93.16%
97.46%
94.21%
98.87%
Stacking
Random Forest
Bagging
Gradient Boosting
XGBoost
Decision Tree
SVM
KNN
Logistic Regression
Models

Results


F1-Score (Positive Class) Comparison Across Models
Stacking
XGBoost
Gradient Boosting
Bagging
Random Forest
Decision Tree
SVM
Naive Bayes
Logistic Regression
0.88
0.78
0.78
0.77
0.87
0.78
0.75
0.75
0.54
Models
0.0
0.2
0.4
0.6
0.8
F1-Score (Positive Class)

Results


Model Evaluation Metrics Heatmap
Logistic Regression
KNN
SVM
Decision Tree
Random Forest
Bagging
Gradient Boosting
XGBoost
Stacking
Models
0.94 0.97 0.92 0.94 0.99
0.93 0.98 0.89 0.93 0.97
0.94 0.98 0.90 0.94 0.99
0.92 0.91 0.92 0.91 0.91
1.00 1.00 1.00 1.00 1.00
1.00 1.00 1.00 1.00 1.00
1.00 1.00 1.00 1.00 1.00
1.00 1.00 1.00 1.00 1.00
0.95 0.99 0.92 0.95 0.99
F1-Score
Precision
Recall
Accuracy
ROC-AUC
Metrics
1.00
0.98
0.96
0.94
0.92
0.90

Related Works:

- □Here are some notable Kaggle projects and datasets related to credit card fraud detection:□
- Credit Card Fraud Detection Dataset
- □This dataset contains anonymized credit card transactions labeled as fraudulent or genuine, providing a foundation for building and evaluating fraud detection models.□
- View Dataset
- Fraud Detection with Naive Bayes Classifier
- □This notebook demonstrates the application of the Naive Bayes algorithm to detect fraudulent transactions, offering insights into its effectiveness for this task.□
- Explore Notebook
- Credit Card Transactions Fraud Detection Dataset
- □A simulated dataset of credit card transactions generated using Sparkov, useful for testing fraud detection algorithms.□
- Access Dataset
- Credit Card Fraud Detection - Using Neural Network
- □This notebook explores the use of neural networks for detecting fraudulent transactions, providing a practical example of deep learning application in fraud detection.□
- View Notebook
- Credit Card Fraud Detection (EDA Case Study)
- □An exploratory data analysis case study focusing on understanding the dataset and identifying patterns related to fraudulent transactions.□
- Explore Case Study
- □These resources offer valuable insights and practical examples for those interested in credit card fraud detection using machine learning techniques.□

CONCLUSION

- Machine learning offers significant potential for detecting credit card fraud, enabling real-time identification of fraudulent transactions.
- The proposed model demonstrates strong performance, making it a practical solution for financial institutions to combat fraud effectively.
- Future work could include testing the model on larger and more diverse datasets, addressing data imbalance, and exploring advanced techniques such as deep learning for further improvements.