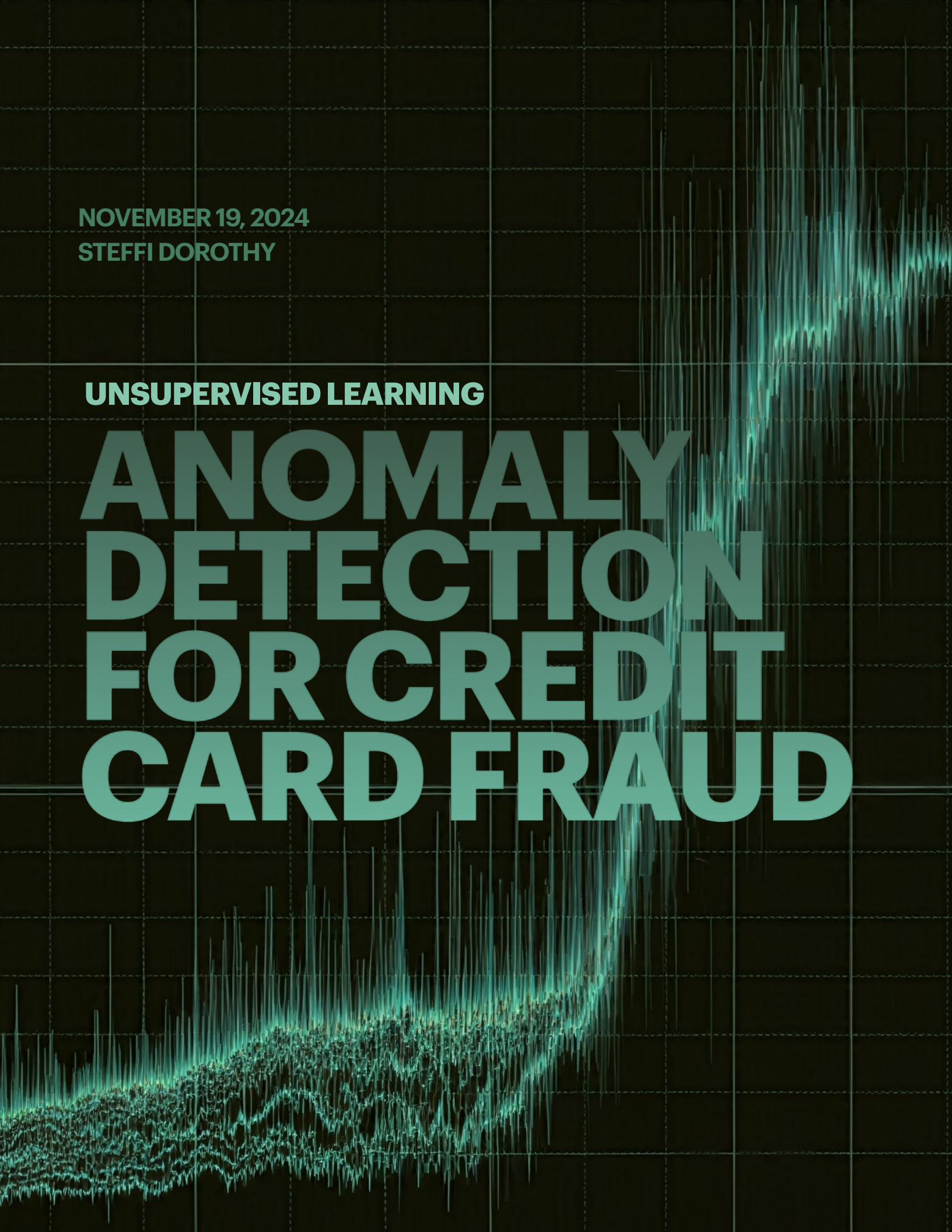


NOVEMBER 19, 2024

STEFFI DOROTHY

UNSUPERVISED LEARNING

ANOMALY DETECTION FOR CREDIT CARD FRAUD

TASK1

INTRODUCTION

This report provides background on credit card fraud detection, including the problem's scope, current solutions, and the potential of machine learning.

SECTION 1.1 - BACKGROUND

Credit card fraud is responsible for worldwide financial losses amounting to tens of billions of dollars. According to the Statistic Brain Research Institute, more than one in ten Americans has fallen victim to credit card fraud, with an average loss of $399. In Europe, the European Central Bank (ECB) reported that card fraud losses in 2018 reached €1.8 billion within the Single European Payments Area (SEPA).

Fraudsters can exploit a wide range of tactics to carry out fraudulent credit card transactions. Although there is no standardized classification of fraud types, certain patterns are more commonly observed. Fraud detection, however, is a dynamic and evolving process. As fraud prevention technologies advance, fraudsters adapt their techniques to exploit new vulnerabilities, often moving from older, well-known targets to new, emerging technologies. They continuously adjust to changes in transaction volumes and characteristics, making detection increasingly challenging.

Many fraud detection systems today rely on supervised learning methods, which require balanced datasets and the availability of labeled data. However, real-world business data often lacks these labels, and unsupervised learning plays a significant role in identifying anomalies. Unlike supervised algorithms, which classify known data points, unsupervised algorithms can detect anomalies in unlabeled datasets. In this project, I have taken an alternative approach by avoiding data balancing and focusing on unsupervised anomaly detection techniques.

SECTION 1.2 - MOTIVATION

In real-world scenarios, unsupervised models are often employed to generate labeled data for further analysis or to inform risk-based decision-making when predefined rules are unavailable. For instance, when detecting anomalies in network traffic, rules such as login times and geographic locations can help flag suspicious activity. Similarly, in employee fraud detection, behavioral data can reveal potential anomalies, though establishing such rules requires extensive domain expertise, which may not always be accessible.

This is where unsupervised learning proves particularly useful. By using algorithms with minimal domain knowledge, we can uncover potential anomalies, which can then be reviewed in an auditing process to assign true labels. As enough labeled data accumulates over time, the problem can evolve into a supervised learning task.

Since this project aims to compare the performance of various unsupervised anomaly detection algorithms, I used a labeled dataset for validation. However, the models do not use the labels during training. The labels are only employed to assess the model's performance and generate relevant evaluation metrics.

SECTION 1.3 - TECHNIQUES EMPLOYED

- Cosine Similarity to detect patterns in transactions.

- 3D PCA Visualization of fraud detection results.

- 3D t-SNE Visualization for better understanding of fraud detection.

- Feature Engineering.

- Class Imbalance Detection and handling strategies.

- Z-Score and IQR Methods for outlier detection.

- Summary Statistics and analysis.

- Transaction Time and Amount Distribution

- Correlation Matrix using Heat map

- Scatter Plots to illustrate key relationships

- Kernel Density Estimation (KDE) for visualizing probability distributions

In this report, I aim to provide a practical survey of anomaly detection techniques, offering intuitive explanations of how each method works and highlighting their advantages and disadvantages. I will begin by discussing the definition and types of anomalies, followed by an overview of the challenges posed by anomaly detection. Then, I will introduce the dataset used in this project.

SECTION 1.4 - ANOMALIES AND THEIR TYPES

Anomalies are patterns in data that significantly deviate from what is considered normal behavior. There are three primary types of anomalies:

1. Point Anomalies: These occur when an individual data point is notably different from the rest of the dataset. For example, in credit card transactions, a single large transaction, relative to an individual's usual spending habits, could be flagged as an anomaly.

2. Contextual Anomalies: These anomalies are identified based on the context surrounding the data point. For instance, a high transaction amount during the holiday season may be normal, but the same amount spent in the middle of spring might raise suspicion. Contextual anomalies often take into account factors such as time or location.

3. Collective Anomalies: This type involves a group of related data points that are anomalous as a whole, even if individual points may not stand out. For example, a retailer's stock levels may naturally fluctuate, but if the stock stays low for an extended period, this could indicate an anomaly.

In this project, I focus primarily on point anomalies, as they are most relevant to detecting fraud in credit card transactions. Although time is included as a feature in the dataset, the main emphasis is on point anomaly detection, with some contextual information also being considered.

SECTION 1.5 -CHALLENGES IN ANOMALY DETECTION

Anomaly detection presents several challenges:

- Defining the Normal Region: It is inherently difficult to determine what qualifies as "normal" behavior, and the distinction between normal and anomalous data is often unclear.

- Evolving Malicious Behavior: Fraudsters continuously refine their methods, causing previously detected anomalies to blend in with normal behavior.

- Shifting Normal Behavior: What is considered "normal" today may change over time, making it harder to identify anomalies in the future.

- Data Complexity and Domain Variability: Anomalies differ across various domains, and no single algorithm can handle all types of anomalies effectively.

- Data Noise: High levels of noise can make it challenging to separate true anomalies from random variations in the data.

- Class Imbalance: Fraudulent transactions represent only a small portion of total transactions, creating an imbalance that makes anomaly detection difficult. Most algorithms struggle with this imbalance, requiring techniques like resampling or loss weighting to address the issue.

SECTION 1.6 - DATASET OVERVIEW

This project uses the **Credit Card Fraud Detection** dataset, which includes transactions made by European cardholders in September 2013. The dataset

contains 492 fraudulent transactions out of 284,807 total transactions, making it highly imbalanced (fraud accounts for just 0.172% of all transactions).

The dataset consists solely of numerical input variables resulting from PCA transformations (features V1 to V28), along with two non-transformed features: Time(elapsed seconds since the first transaction) and Amount (transaction amount). The target variable, Class, indicates whether the transaction was fraudulent (1) or not (0).

SECTION 1.7 - MACHINE LEARNING IN CREDIT CARD FRAUD DETECTION (CCFD)

Credit card fraud detection (CCFD) is an increasingly important problem due to the vast amounts of transaction data involved. The complexity of fraudulent behavior and the volume of transactions make manual detection methods inefficient. Machine learning (ML) has become a powerful tool in CCFD, helping to identify patterns in large datasets and improving the effectiveness of fraud detection systems. ML algorithms have significantly enhanced the ability of fraud investigators to detect fraudulent transactions.

Although ML for CCFD is an active research area, caution is necessary when interpreting results, as there are currently no standardized benchmarks or methodologies for comparing the effectiveness of different techniques.

TASK2

EXPLORATORY DATA ANALYSIS

SECTION 2.1 - SUMMARY STATISTICS

MEASURES OF CENTRAL TENDENCY:

- **Mean**: The average value of the dataset= 88
- **Median**: The middle value = 22

MEASURES OF DISPERSION:

- **Range**: The difference between the maximum and minimum values =. 25691
- **Standard Deviation**: The square root of the variance, showing how much data points deviate from the mean= 250

Summary of the Dataset:

- No missing values are present.

- Fraudulent transactions constitute only 0.17% of all transactions in the dataset.

SECTION 2.2 - CLASS IMBALANCE DETECTION

```
Class
0    284315
1       492
Name: count, dtype: int64
```

A distribution of 284315 non-fraud and 492 fraud labels indicate a highly imbalanced data set. This imbalance should not be a problem for this project.

KEY OBSERVATIONS

Data Distribution:

- The data points are scattered across a wide range of Time values, suggesting a diverse distribution of transaction times.
- The Amount values also span a broad range, indicating varying transaction sizes.

Class Distribution:

- Two distinct classes are represented by different colors.
- Class 0 appears to be the majority class, with more data points than Class 1.

Relationship between Time and Amount:

- There doesn't seem to be a strong correlation between Time and Amount.
- The data points are dispersed without a clear pattern, suggesting that the time of a transaction doesn't necessarily dictate its size.

Class Segregation:

- The two classes appear to overlap to a certain extent, meaning that based on Time and Amount alone, it might be challenging to perfectly distinguish between the two classes.
- However, there might be subtle patterns or clusters within each class that could be further explored using more advanced techniques like PCA, t-SNE, KMeans Clustering and Kernel Density Estimation.

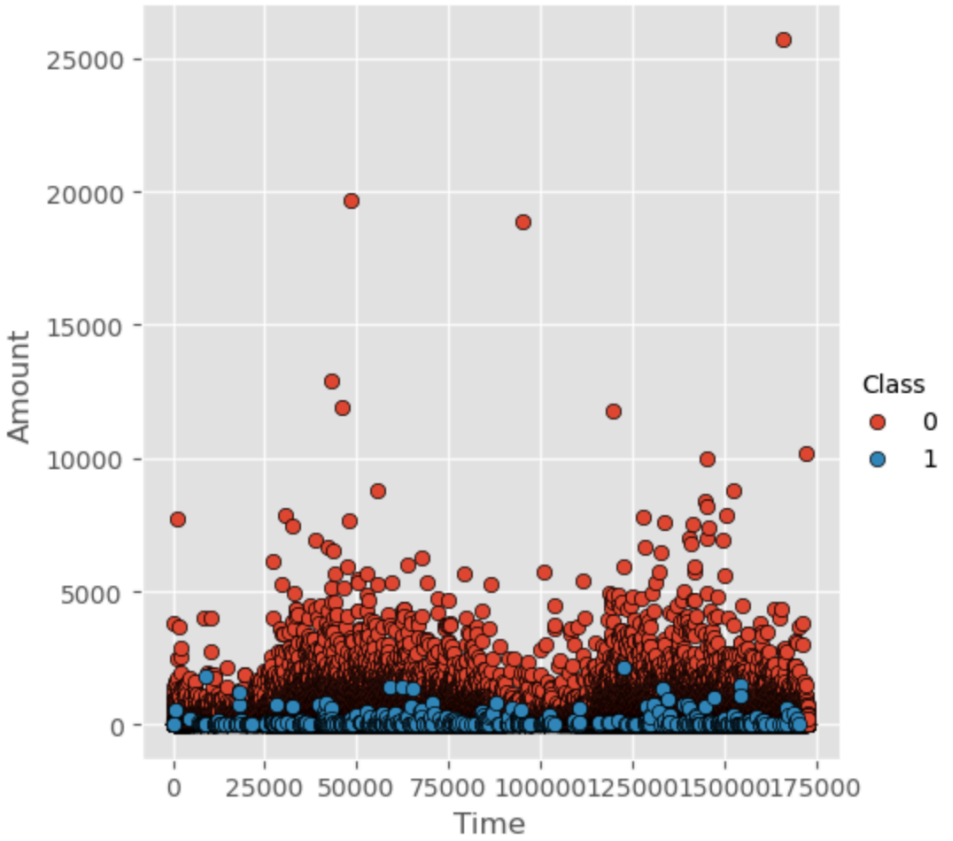
POTENTIAL INSIGHTS AND FURTHER ANALYSIS

Outliers:

- There are a few data points with exceptionally high Amount values, which could be considered outliers.
- Investigating these outliers using statistical methods like Z-Score, Inter Quartile Range and anomaly detection algorithms might provide valuable insights into unusual transaction patterns.

Class-Specific Analysis:

- Analyzing the distribution of Time and Amount within each class separately could reveal class-specific trends. This is done using displot in the further steps.



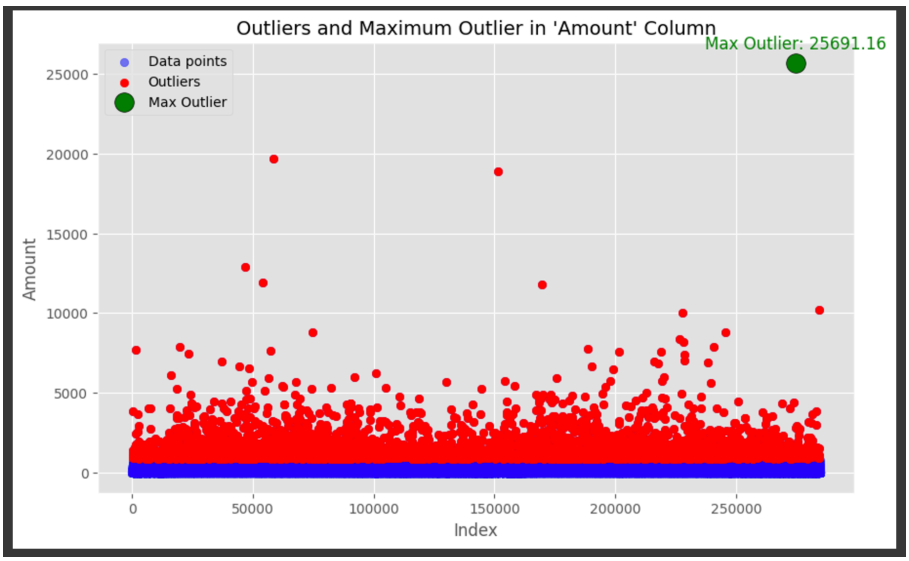
✤This might help identify characteristics that differentiate the two classes.

Feature Engineering:

- ✤ Creating new features based on Time (e.g., time of day, day of week) or Amount (e.g., log-transformed Amount) could potentially improve the separation between classes.

The plot shows that there are frauds only on the transactions which have transaction amount less than 2500 (approx). However, the frauds in the transactions are evenly distributed throughout all times.

SECTION 2.3 - Z SCORE METHOD FOR OUTLIER DETECTION



Data Summary

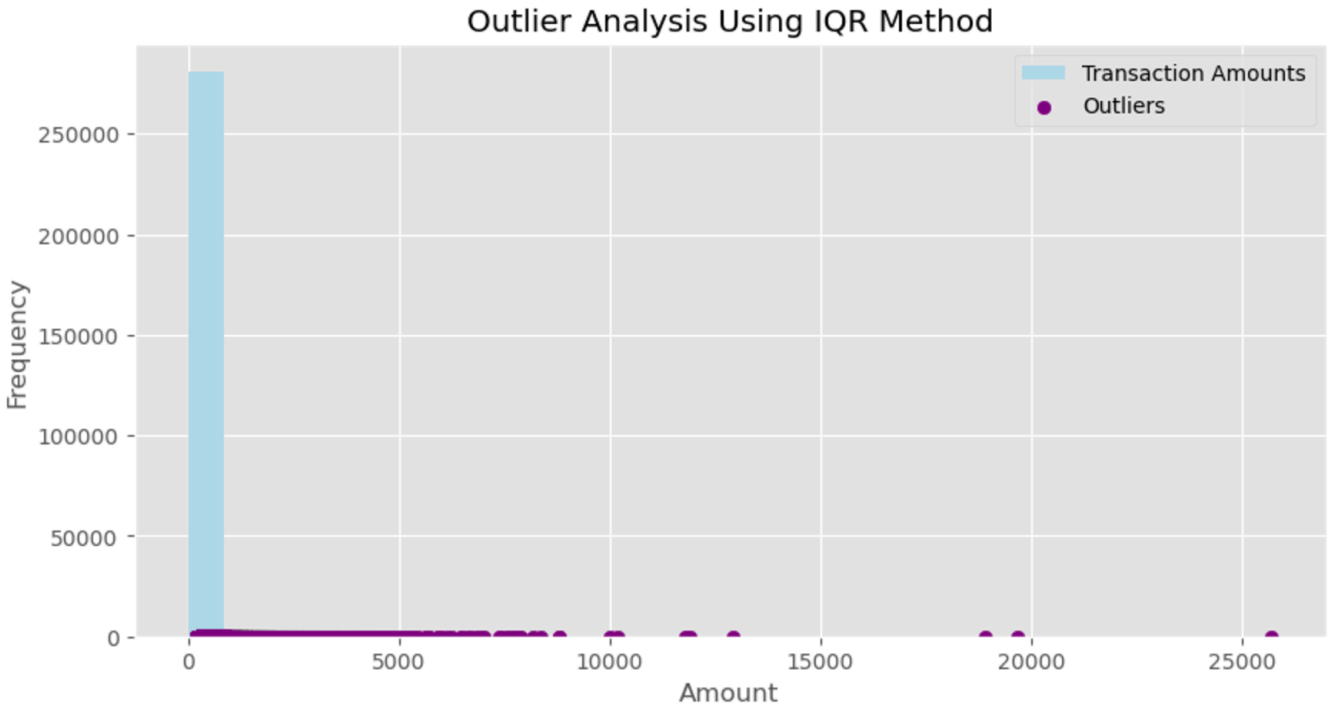
- **Number of Data Points:** 250,000
- **Maximum Outlier:** 25,691.16

Visual Analysis

The scatter plot visually represents the data points and outliers in the 'Amount' column.

- **Data Points:** The majority of data points are clustered around lower values, with some extending to higher values.
- **Outliers:** Several outliers are visible as isolated points scattered above the main cluster. These points deviate significantly from the general trend of the data.

SECTION 2.4 - IQR METHOD FOR OUTLIER DETECTION



The plot clearly shows that the transaction amounts are not uniformly scattered. The majority of transactions are concentrated in the lower range of amounts, with

a significant drop-off as the amount increases. This indicates a high frequency of smaller transactions.

On the other hand, the outliers are spread out across a wider range of higher amounts. While these outliers are present, they are less frequent compared to the majority of transactions.

This distribution pattern suggests that the dataset might be right-skewed, with a long tail towards higher values. This skewness is expected as it is a financial data, where a few large transactions can significantly impact the overall distribution.

SECTION 2.5 - FINDING SIMILARITY IN TRANSACTIONS USING COSINE SIMILARITY

Key Findings

- **High Similarity Scores:** The top 5 suspected fraud transactions for each of the five reference transactions have a similarity score of 1.0, suggesting a very strong similarity.
- **Consistent Class Labels:** All identified suspected fraud transactions have a class label of 1.0, which presumably indicates a classification as fraudulent.

Implications

The high similarity scores and consistent class labels of the suspected fraud transactions suggest that the similarity metric used is effective in identifying potential fraudulent activity. However, it's important to note that similarity alone is not sufficient for definitive fraud detection. Further investigation and analysis are necessary to confirm the fraudulent nature of these transactions.

```
Top 5 suspected fraud transactions having highest similarity with Transaction ID= 183484
     Similarity  Class  Transaction ID
435         1.0    1.0          239499
471         1.0    1.0          261056
450         1.0    1.0          247995
491         1.0    1.0          281674
364         1.0    1.0          181966


Top 5 suspected fraud transactions having highest similarity with Transaction ID= 255448
     Similarity  Class  Transaction ID
404         1.0    1.0          219025
476         1.0    1.0          263080
447         1.0    1.0          245347
410         1.0    1.0          222419
399         1.0    1.0          214775


Top 5 suspected fraud transactions having highest similarity with Transaction ID= 244749
     Similarity  Class  Transaction ID
240         1.0    1.0          112840
375         1.0    1.0          191544
420         1.0    1.0          230476
392         1.0    1.0          204503
463         1.0    1.0          251904


Top 5 suspected fraud transactions having highest similarity with Transaction ID= 63919
     Similarity  Class  Transaction ID
322         1.0    1.0          151196
238         1.0    1.0          108708
355         1.0    1.0          157918
267         1.0    1.0          141259
268         1.0    1.0          141260


Top 5 suspected fraud transactions having highest similarity with Transaction ID= 11475
     Similarity  Class  Transaction ID
231         1.0    1.0          102782
321         1.0    1.0          151103
491         1.0    1.0          281674
483         1.0    1.0          274475
338         1.0    1.0          154633
```

SECTION 2.6 - DISTRIBUTION OF TRANSACTION TIME AND AMOUNT

Transaction Time Distribution

- **Overall Shape:** The distribution of transaction time appears to be multimodal, with multiple peaks. This suggests that there are specific time periods during which transactions are more frequent.

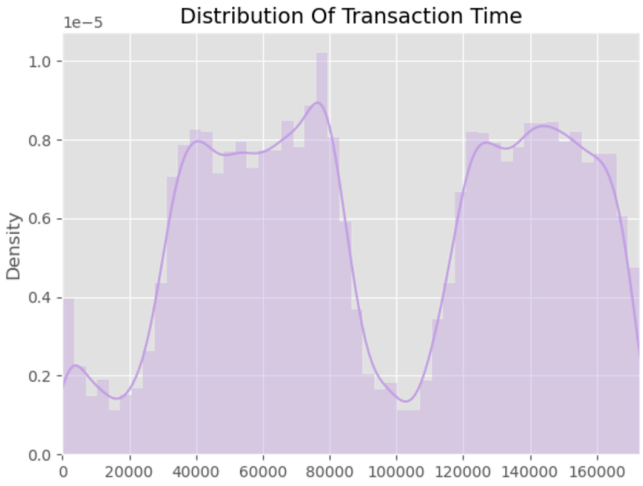
- **Potential Time Clusters:** The peaks in the distribution likely correspond to specific times of the day or week when transactions are more common, such as during business hours or weekends.

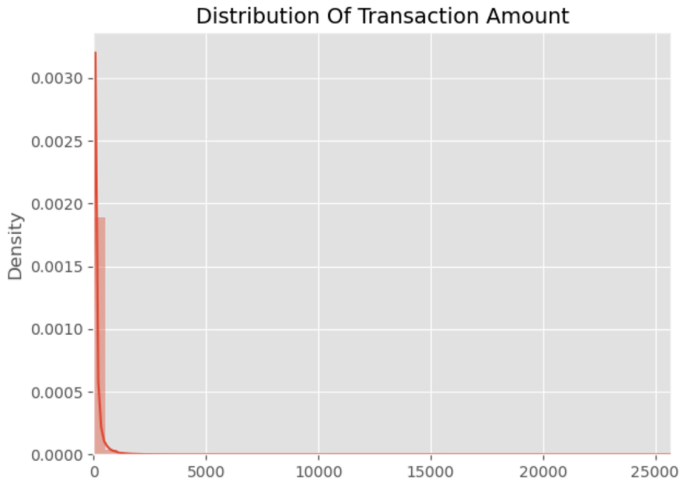
Transaction Amount Distribution

- **Right-Skewed:** The distribution of transaction amounts is heavily right-skewed, meaning that most transactions are relatively small, while a few transactions are significantly larger.
- **Outliers:** The long tail of the distribution indicates the presence of outliers, which are transactions with exceptionally high amounts.

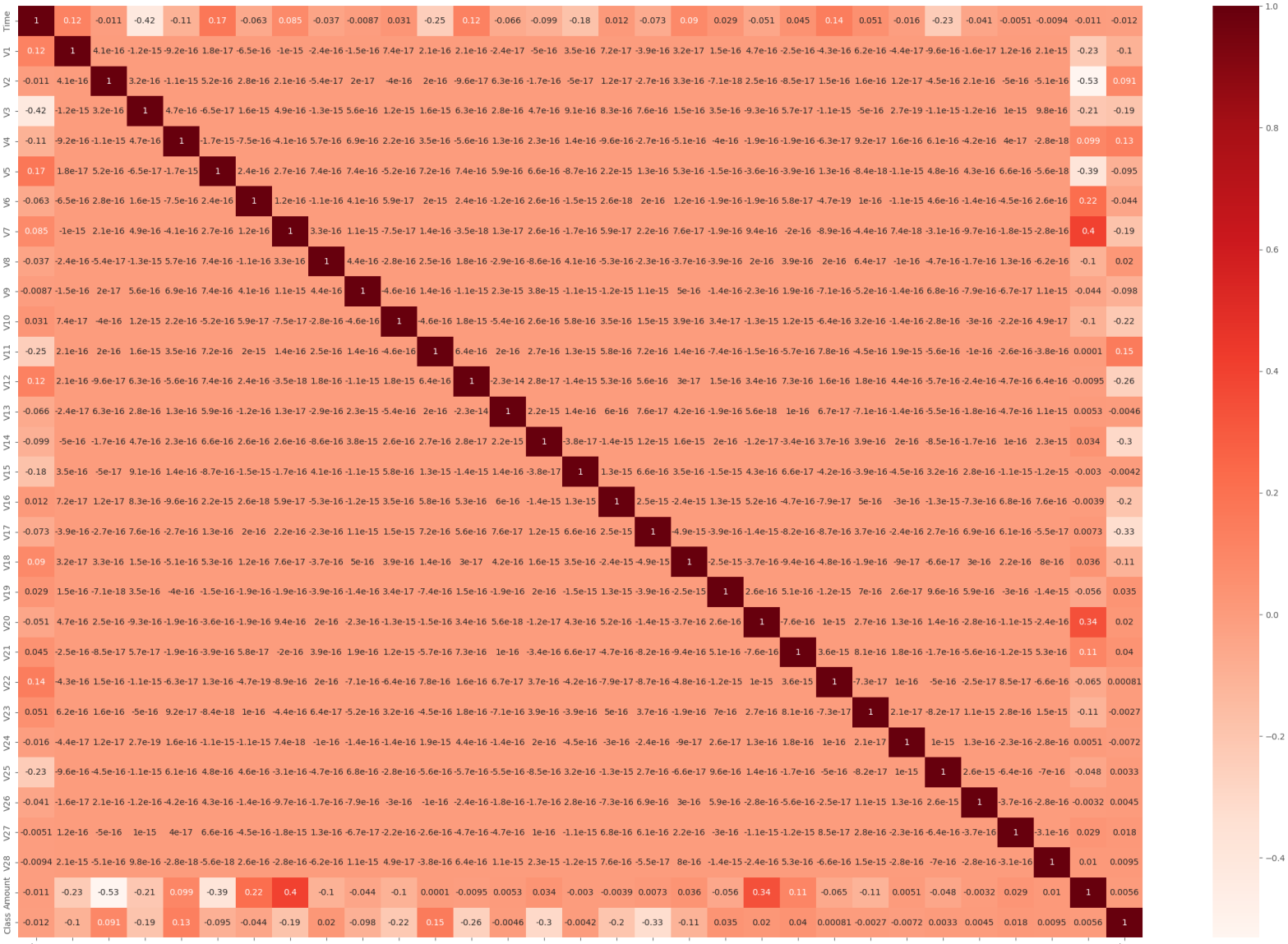
Implications

- **Transaction Time:** The multimodal nature of the transaction time distribution suggests that there are specific patterns in transaction behavior. Understanding these patterns can be useful for fraud detection, as unusual transaction times might indicate suspicious activity.
- **Transaction Amount:** The right-skewed distribution of transaction amounts is a common characteristic of financial data. Outliers in this distribution can significantly impact statistical analysis and modeling.





SECTION 2.7 - CORRELATION MATRIX USING HEAT MAP



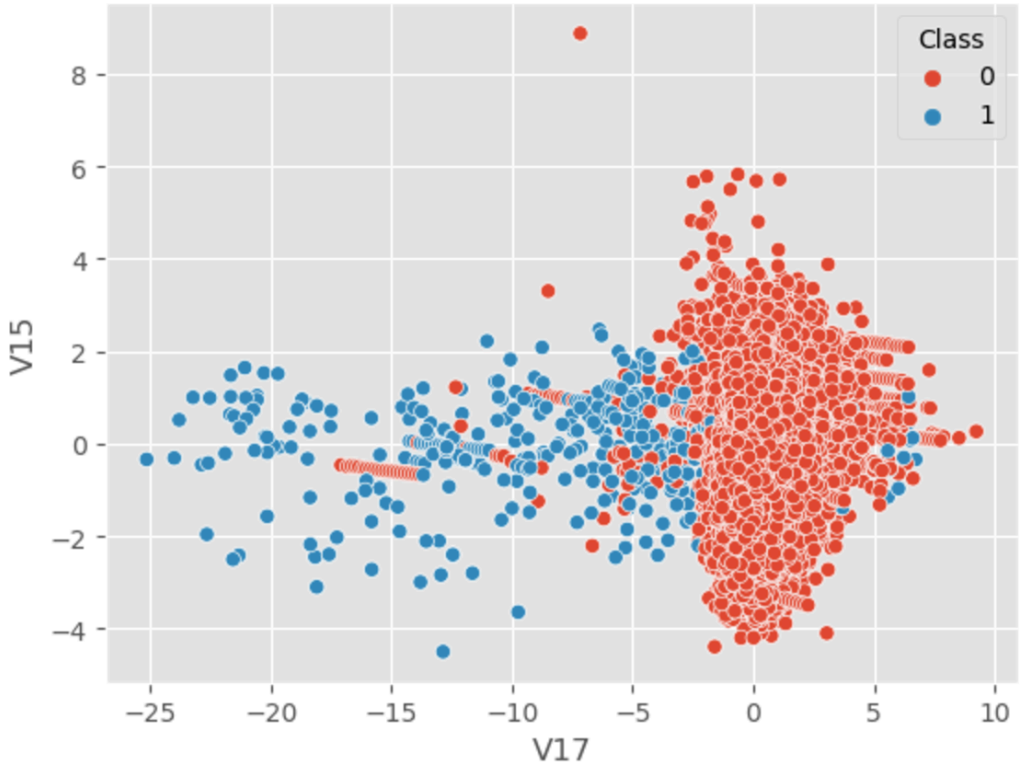
To look into the relation between the independent variables and the dependent variable Class, and see which variables are highly correlated with target variable Class, I plotted a correlation matrix.

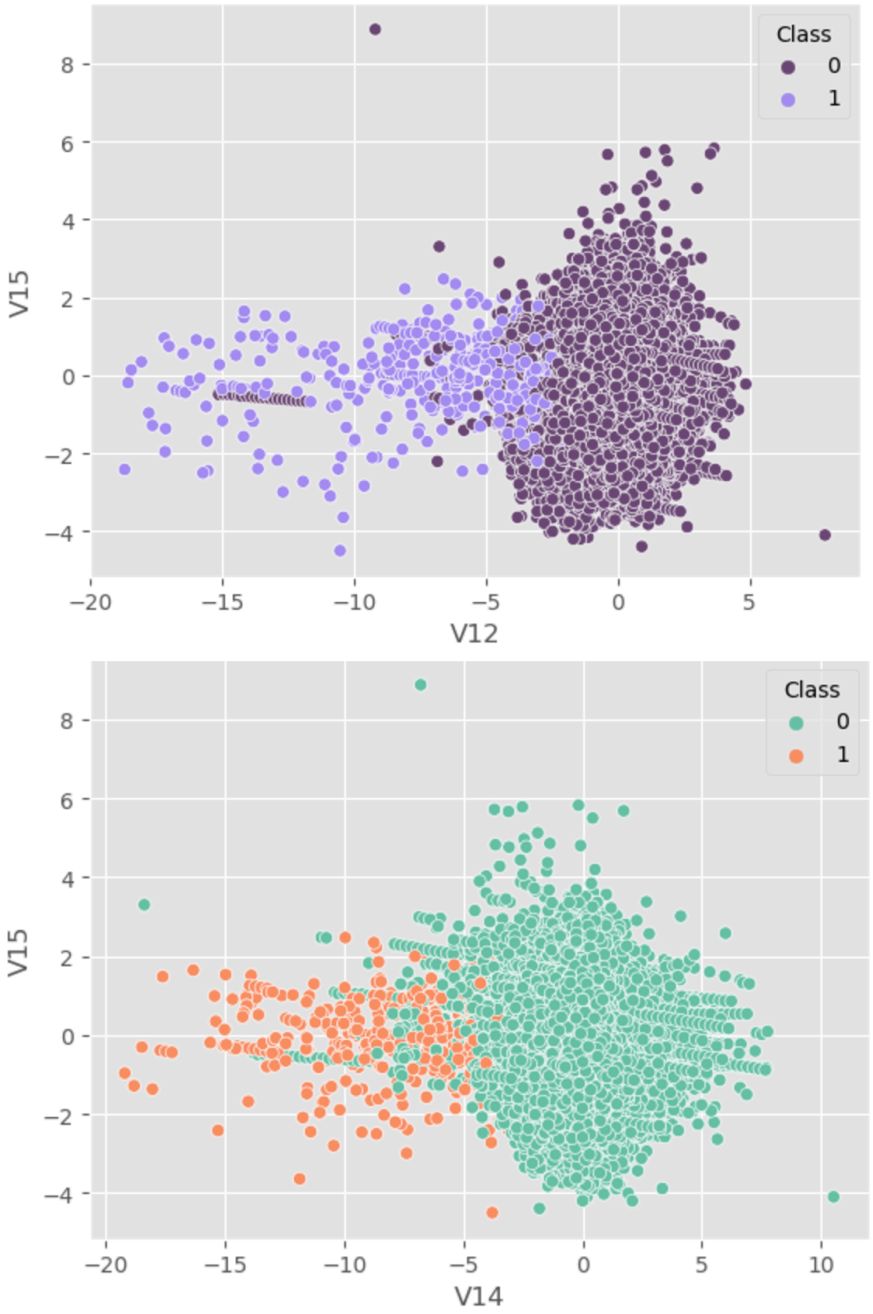
- Many features show weak or no correlation with each other (values close to 0). These features might be independent or only weakly related.
- A few features exhibit negative correlations (values close to -1). This indicates that as one feature increases, the other decreases.

- The top 3 independent variables {V17, V14, V12} that should contains important information about whether a giving credit card transaction is fraudulent or not.

SECTION 2.8 - SCATTER PLOTS

Visualized the interesting relations that I obtained from the correlation matrix using scatter plots. Plotted each one of the interesting variables against a very boring variable V15. In all three plots we can draw a vertical line that classifies almost all data points into their correct Class value. If we look at the x-axis that corresponds to one of the interesting variables {V17, V14, V12}, then we can see that most fraudulent data points are located below the value of -5 and normal data points are located above.
In contrast, if we look at the y-axis where the boring variable is, then both fraudulent and normal data points are almost equally distributed between -4 and 4, which means that we can't draw a horizontal line to separate the two groups.

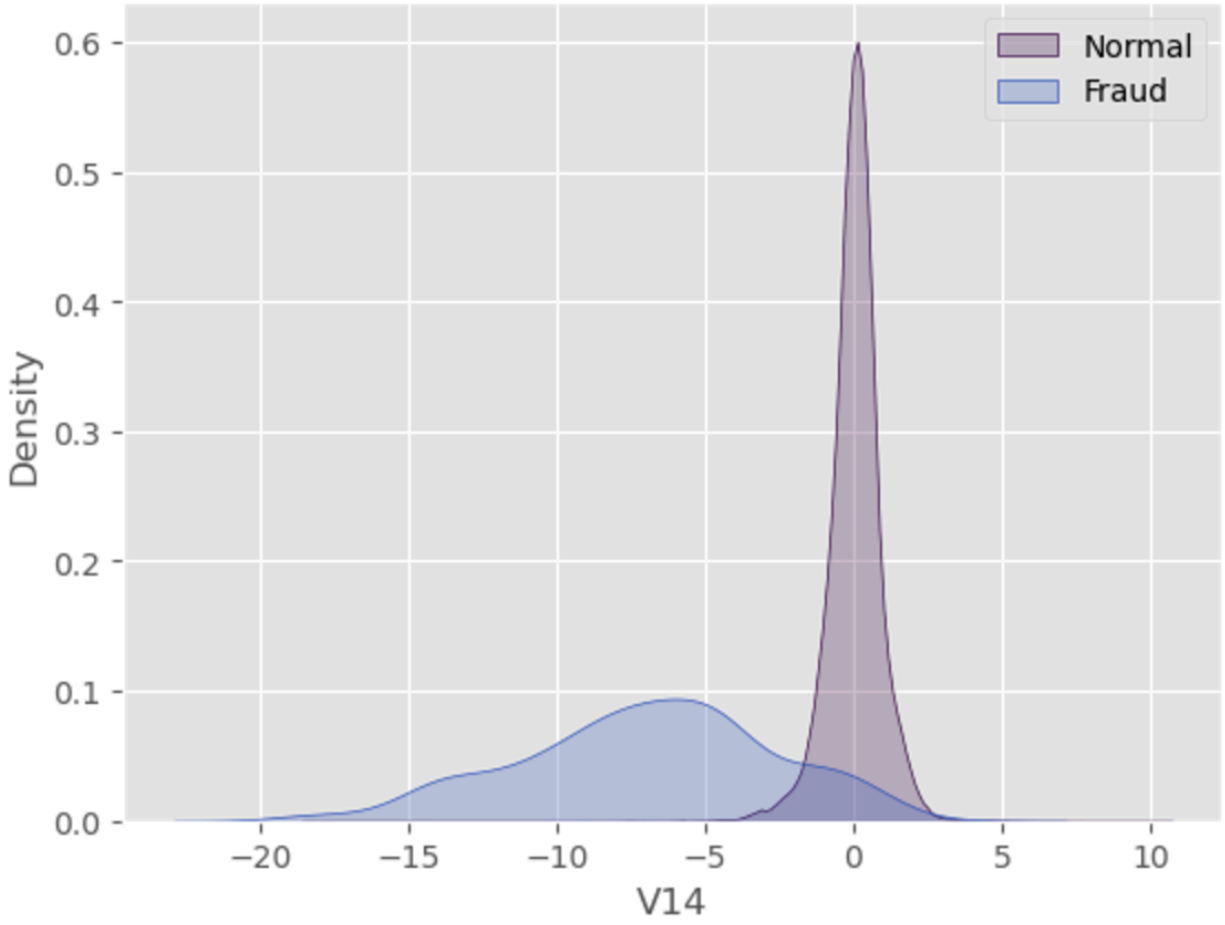


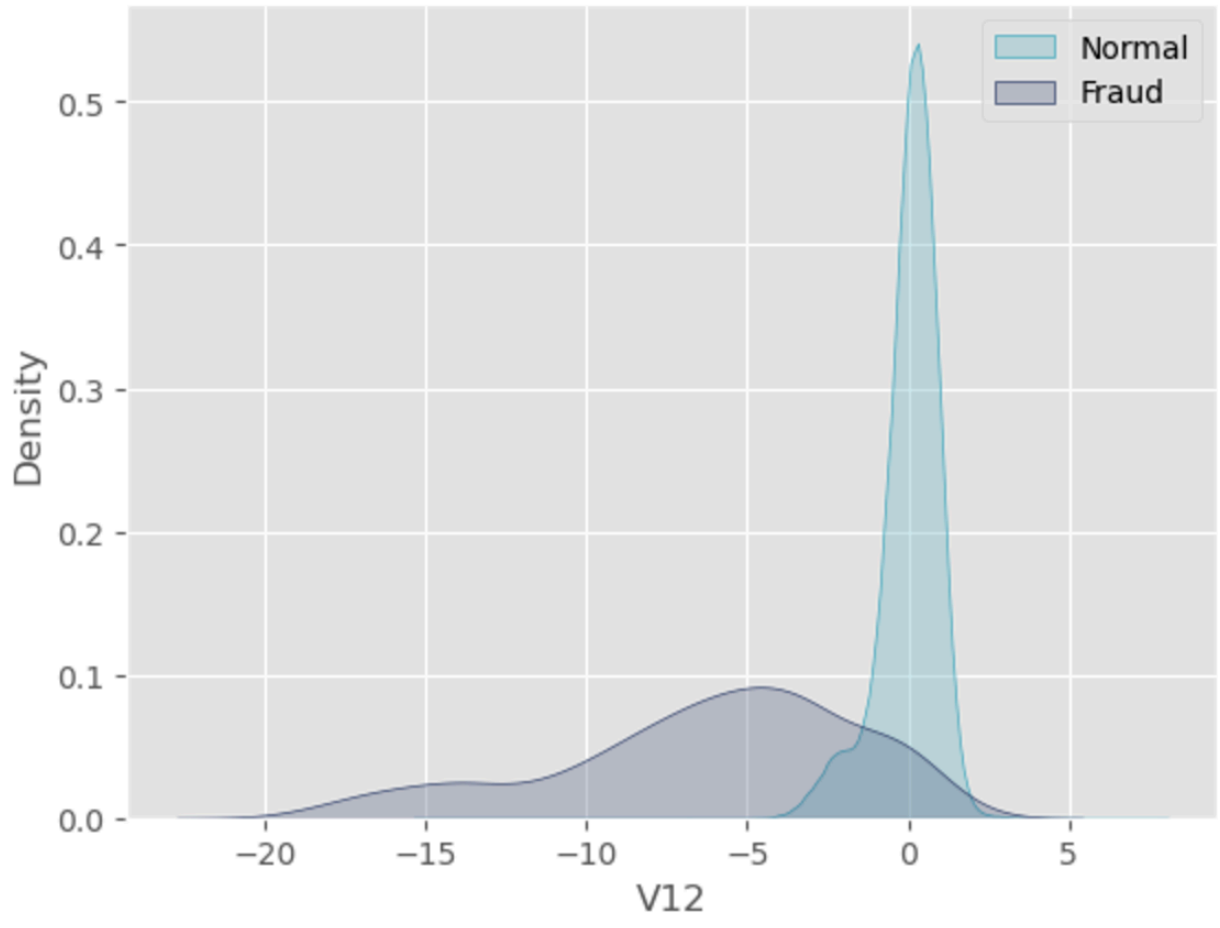


Although scatter plots gave us a general idea of what is going on, but they are not precise. So, I am going to use KDE plots.

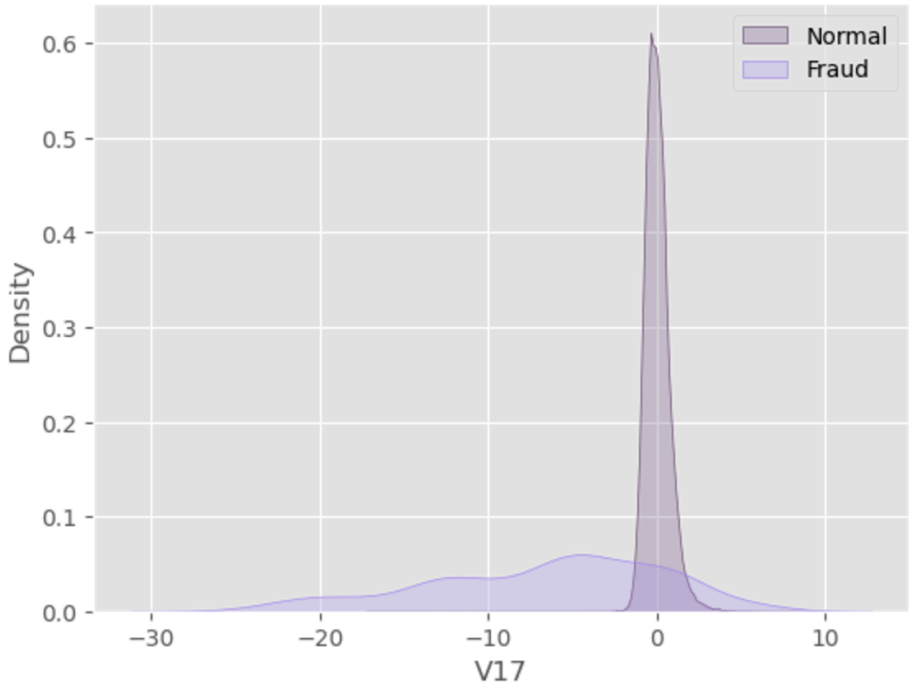
SECTION 2.9- KERNEL DENSITY ESTIMATION PLOTS

I am taking each one of the interesting variables and approximate the underlying probability density function for each Class value (Frauds Vs. Normal) using kernel density estimation. This should give a clear idea of how fraudulent and normal datapoints (credit card transactions) are distributed along each variable.

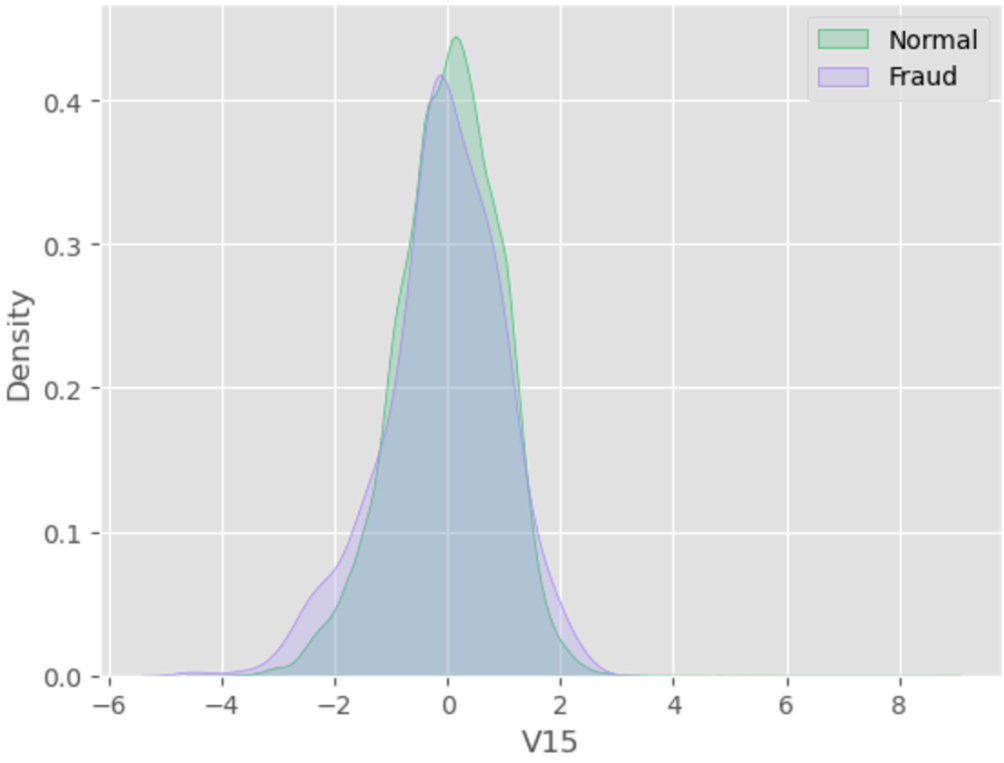

Normal
Fraud
Density
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−20
−15
−10
−5
0
5
10
V14


Normal
Fraud
Density
0.5
0.4
0.3
0.2
0.1
0.0
−20
−15
−10
−5
0
5
V12

The distribution for fraudulent transactions shows a higher concentration of values in the higher range, with some outliers extending further to the right. This suggests that fraudulent transactions might involve larger or unusual values for V12, V14, V17.



In contrast, plotted one of the boring variables.



Observations

- **Similar Distribution:** The distributions of V15 for both normal and fraudulent transactions appear to be quite similar. Both distributions are right-skewed, with a peak around 0. This suggests that V15 might not be a highly discriminative feature for distinguishing between the two classes.
- **Slight Overlap:** There is some overlap between the two distributions, especially in the lower range of values. This indicates that there might be some ambiguity in classifying transactions based solely on V15.

Summarizing my findings in the following points: In every interesting variable, the distribution of the normal transaction takes a shape that is very close to the standard normal distribution. In every interesting variable, the distribution of the fraudulent transaction takes a shape that is very close to a normal distribution with high standard deviation (highly spread). In the boring variable, both fraudulent and normal transactions have the same distribution, close to standard normal distribution.

STATISTICAL MEASUREMENTS TO SUPPORT MY FINDINGS

Values of the interesting variable V14 that belongs to Class 0 (Normal).

```
count    284315.000000
mean          0.012064
std           0.897007
min         -18.392091
25%          -0.422453
50%           0.051947
75%           0.494104
max          10.526766
Name: V14, dtype: float64
```

Values of the interesting variable V14 that belongs to Class 1 (Fraud).

```
count    492.000000
mean      -6.971723
std        4.278940
min      -19.214325
25%       -9.692723
50%       -6.729720
75%       -4.282821
max        3.442422
Name: V14, dtype: float64
```

Values of the boring variable V15 that belongs to Class 0 (Normal).

```
count    284315.000000
mean          0.000161
std           0.915060
min          -4.391307
25%          -0.582812
50%           0.048294
75%           0.648842
max           8.877742
Name: V15, dtype: float64
```

Values of the boring variable V15 that belongs to Class 1 (Fraud).

```
count    492.000000
mean      -0.092929
std        1.049915
min       -4.498945
25%       -0.643539
50%       -0.057227
75%        0.609189
max        2.471358
Name: V15, dtype: float64
```

TASK3

PROBLEM DEFINITION

Fraudulent activities have become increasingly sophisticated and prevalent in the digital age, posing significant financial risks to businesses and individuals. Traditional fraud detection methods often rely on rule-based systems or supervised machine learning models, which may not be effective in detecting novel fraud patterns. To address this challenge, this project explores unsupervised anomaly detection techniques to identify unusual transactions that deviate from normal behavior.

MOTIVATION:
The motivation for this project stems from the limitations of traditional supervised learning approaches, which require labeled data to train the model. In real-world scenarios, obtaining labeled data for fraudulent transactions can be challenging and time-consuming. Unsupervised learning techniques offer a promising alternative by identifying anomalies without prior knowledge of fraudulent behavior.

DATA AND METHODOLOGY:
The project utilizes a real-world credit card transaction dataset. Exploratory Data Analysis (EDA) revealed a significant class imbalance, with normal transactions vastly outnumbering fraudulent ones. To address this challenge, I employed unsupervised anomaly detection techniques that are not reliant on labeled data.

KEY OBSERVATIONS FROM EDA:

- **Class Imbalance:** The dataset exhibited a significant imbalance between normal and fraudulent transactions.
- **Feature Distribution:** Certain features, such as Amount and Time, showed distinct patterns for fraudulent transactions compared to normal ones.
- **Outlier Presence:** Outliers were observed in several features, potentially indicating anomalous behavior.

PROPOSED APPROACH:
Unsupervised Anomaly Detection:

- **Isolation Forest:** This algorithm isolates anomalies by randomly selecting features and splitting data points.
- **Local Outlier Factor:** It identifies anomalies based on the density of data points in the feature space.
- **K-means clustering(with distance to centroids)**: This method identifies data points that fall significantly far away from the cluster centroids, effectively treating outliers as anomalies.

Model Evaluation:

- Use appropriate metrics for anomaly detection, such as precision, recall, F1-score, and ROC-AUC curve.
- Evaluate the models on a held-out test set.

TASK4

DATA PREPROCESSING

Section 4.1 Data Cleaning

Data cleaning is a crucial step in any data analysis or machine learning project. It involves identifying and correcting errors and inconsistencies in the data to improve its quality and reliability. Here are the common steps involved in data cleaning in this project:

- Shape

```
df.shape
```

```
(284807, 31)
```

- Duplicate Data

```
duplicate_rows = df[df.duplicated()]
print(duplicate_rows)
```

```
Empty DataFrame
Columns: [Time, V1, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13, V14, V15, V16, V17, V18, V19, V20, V21, V22, V23, V24, V25, V26, V27, V28, Amount, Class, Hour, Log Amount]
Index: []

[0 rows x 33 columns]
```

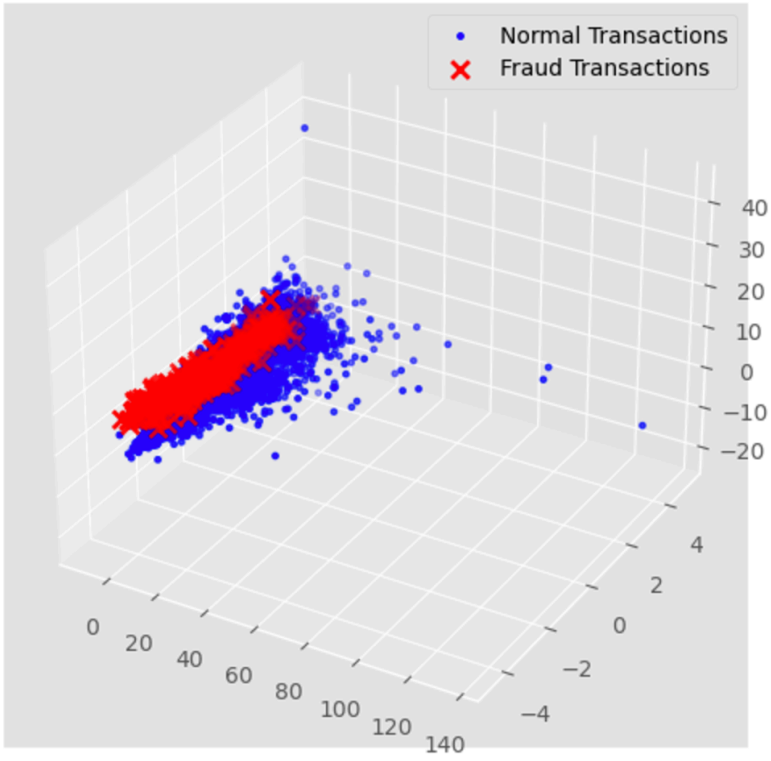
- Missing Values

```
df.isnull().sum()
```

```
Time      0
V1        0
V2        0
V3        0
V4        0
V5        0
V6        0
V7        0
V8        0
V9        0
V10       0
V11       0
V12       0
V13       0
V14       0
V15       0
V16       0
V17       0
V18       0
V19       0
V20       0
V21       0
V22       0
V23       0
V24       0
V25       0
V26       0
V27       0
V28       0
Amount    0
Class     0
dtype: int64
```

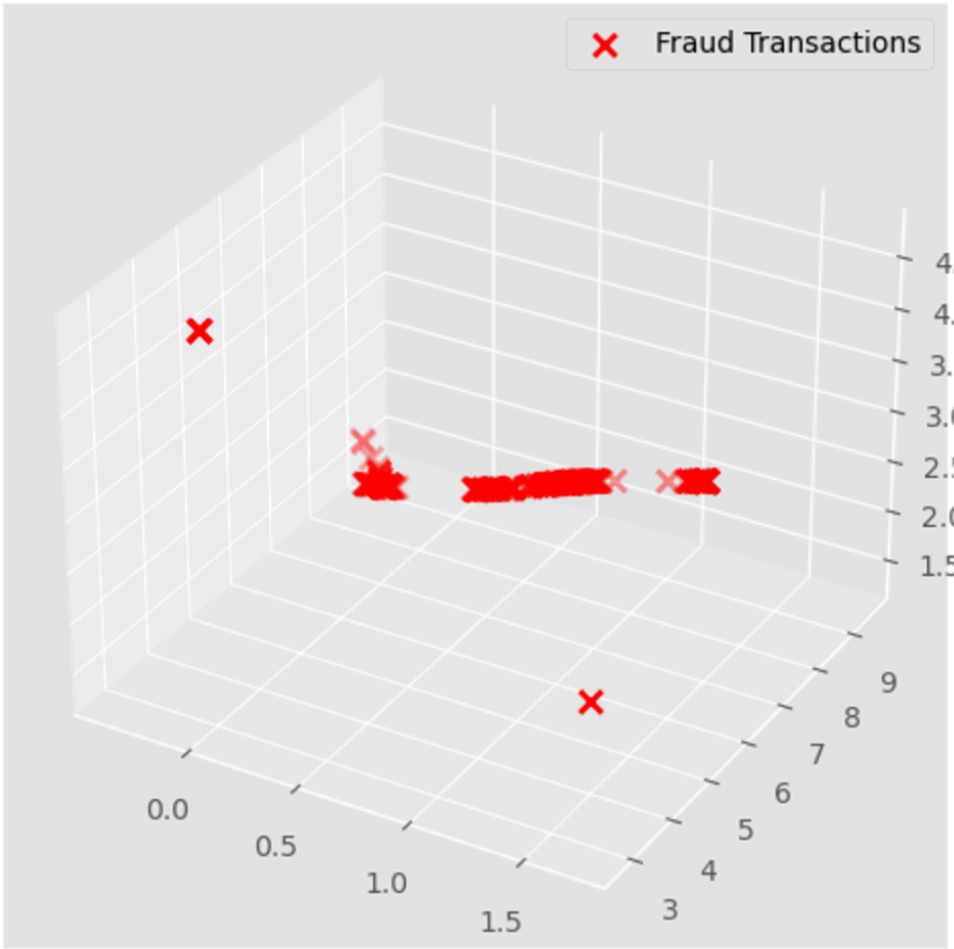
Section 4.2 Data Transformation

SECTION 4.2.1 VISUALIZING FRAUDS IN 3D WITH PCA



PCA (Principal Component Analysis) is an unsupervised linear dimensionality reduction technique. It tries to preserve the global structure of data. PCA is a deterministic method. I have reduced the entire data set into three principal components and then plotted the figure to visualize anomalies. The red cross points are outliers and the blue points are inliers.

SECTION 4.2.2 VISUALIZING FRAUDS IN 3D USING T-SNE



t-SNE (t-distributed Stochastic Neighborhood Embedding) is an unsupervised non-linear dimensionality reduction technique. It tries to preserve the local structure of data. I have reduced entire data set into three t-SNE components

and then plotted the figure to visualize anomalies. Unlike PCA, it is a non-deterministic or probabilistic method. In simple heuristic terms, we can say that PCA preserves large distances between data points, whereas t-SNE preserves data points which are close to each other.

Cluster Formation: The fraudulent transactions appear to form distinct clusters in the 3D space. This suggests that there might be underlying substructures within the fraudulent data, potentially indicating different types of fraudulent activities.

SECTION 4.2.3 FEATURE ENGINEERING

Temporal Feature

Transaction Time of Day: Extracting the hour from the Time column to create a feature indicating the time of day the transaction occurred.

Amount Based Feature

Log Transformation of Amount: Applying a log transformation to the Amount feature to handle skewness.

SECTION 4.2.4 ROBUST Z SCORE

I am adding the robust Z-score as a new feature to my existing data and using it to train the model.

Reasons To Add the Robust Z-Score:

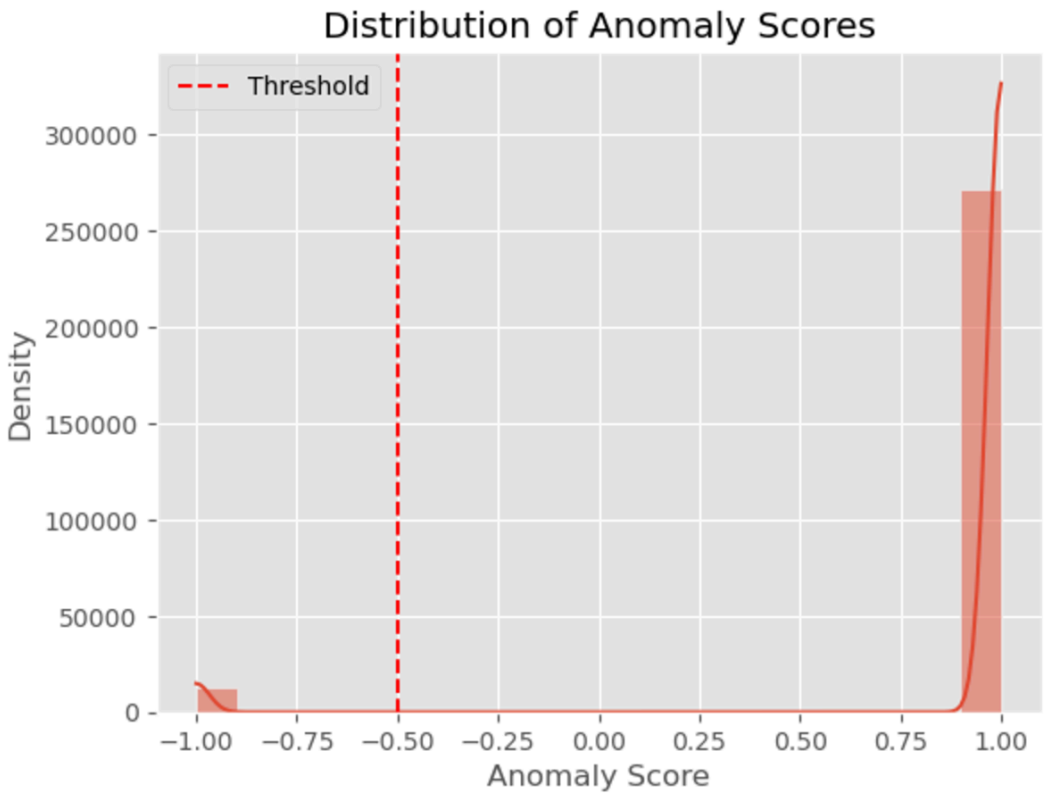
- **Enhanced Feature Engineering:** The robust Z-score can serve as a valuable feature, capturing information about the relative position of a data point within the distribution.
- **Improved Anomaly Detection:** By incorporating the robust Z-score, the model can better distinguish between normal and anomalous data points, especially in the presence of outliers.
- **Better Model Performance:** In many cases, adding relevant features can improve the overall performance of the model.

I have chosen anomaly detection algorithms that are well-suited for handling numerical data and can effectively leverage the information provided by the robust Z-score.

TASK5

MODEL TRAINING OPTIMIZATION EVALUATION & EXPLAINABILITY

1. UNSUPERVISED VANILLA MODEL USING ISOLATION FOREST



Since I have ground truth labels (Class), I am using the following metrics for evaluating the model.

Methodology:

1. **Model Training:** An Isolation Forest model was trained on the dataset.
2. **Anomaly Score Calculation:** The model assigned an anomaly score to each data point, indicating its degree of abnormality.
3. **Visualization:** The anomaly scores were visualized using a histogram with a density plot to understand their distribution.

Results:

The histogram and density plot reveal the following insights:

- **Normal Data:** A significant portion of the data points have anomaly scores close to 0, indicating normal behavior.
- **Anomalies:** A smaller number of data points exhibit higher anomaly scores, suggesting potential anomalies.

```
Confusion Matrix
[[272205  11048]
 [    74    399]]

Classification Report
              precision    recall  f1-score   support

           0       1.00      0.96      0.98    283253
           1       0.03      0.84      0.07       473

    accuracy                           0.96    283726
   macro avg       0.52      0.90      0.52    283726
weighted avg       1.00      0.96      0.98    283726

ROC AUC Score:
0.902273898541267
```

INTERPRETATION

True Negatives (TN): 272,205 (normal instances correctly identified)
False Positives (FP): 11,048 (normal instances incorrectly identified as anomalies)
False Negatives (FN): 74 (anomalies incorrectly identified as normal instances)
True Positives (TP): 399 (anomalies correctly identified)

Class 0 (Normal Instances):

- **Precision**: 1.00 (high precision indicates almost no false positives for normal instances)
- **Recall**: 0.96 (high recall indicates most normal instances are correctly identified)
- **F1-Score**: 0.98 (harmonic mean of precision and recall)
- **Support**: 283,253 (number of actual occurrences in the dataset)

Class 1 (Anomalies):

- **Precision**: 0.03 (low precision indicates a high number of false positives)
- **Recall**: 0.84 (high recall indicates most anomalies are detected)
- **F1-Score**: 0.07 (low F1 score due to the imbalance between precision and recall)
- **Support**: 473 (number of actual occurrences in the dataset)

ROC AUC Score

0.902273898541267

- This score indicates the model's ability to distinguish between the classes. A score of 0.90 shows good performance.

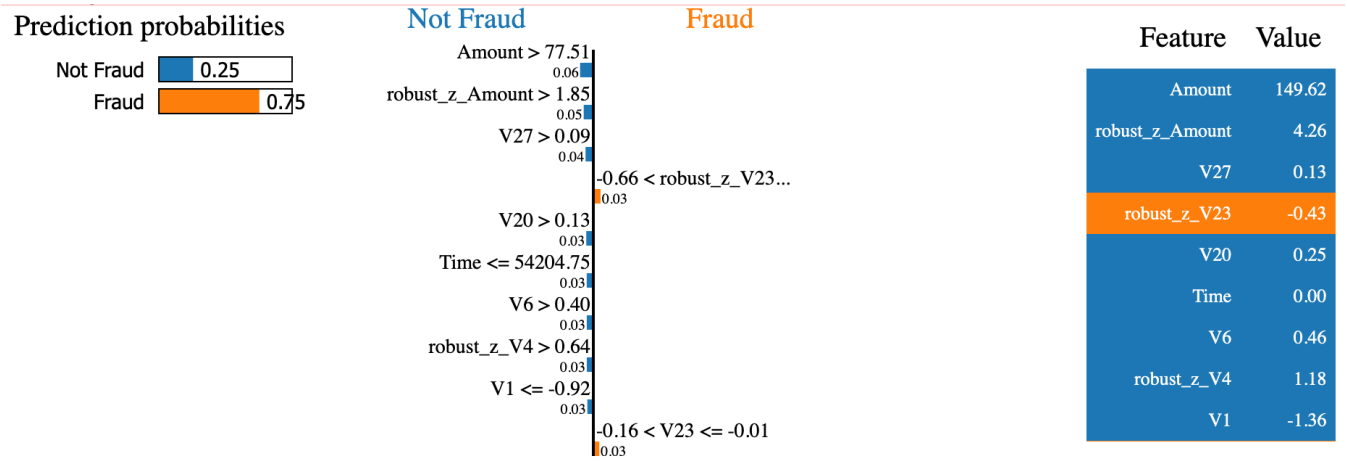
For anomalies (class 1), the model has a high recall (0.84), meaning it detects the majority of actual anomalies.

MODEL EXPLAINABILITY USING LIME

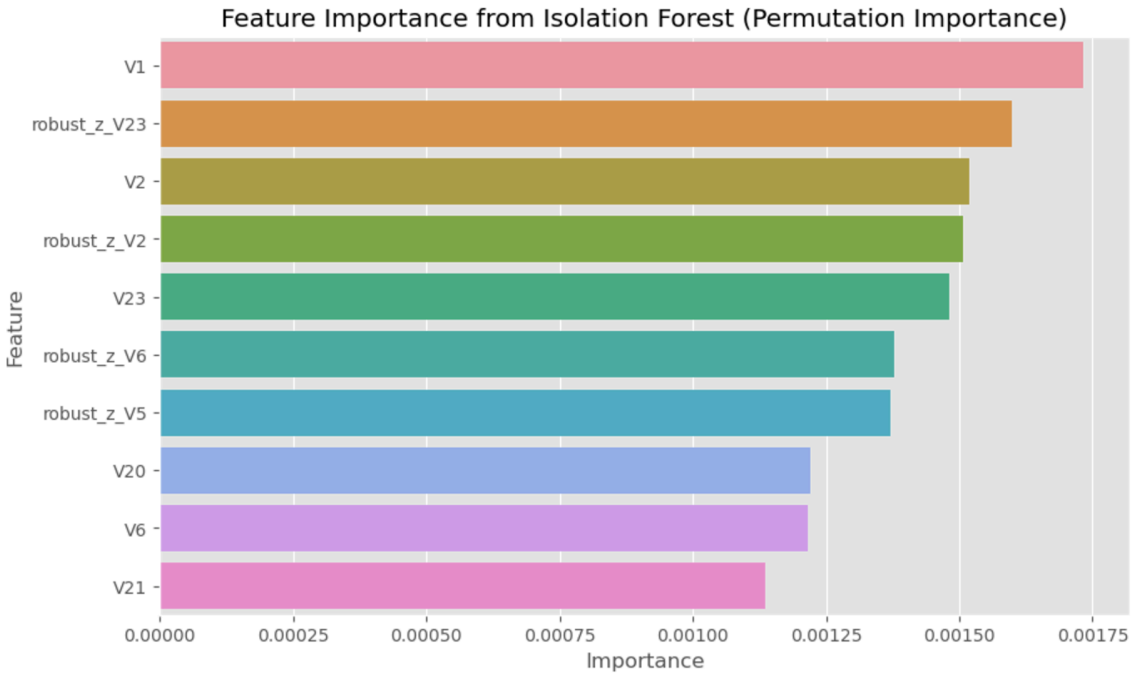
Key Insights

- **Prediction:** The model predicts a high probability of fraud for the given transaction.

- **Key Features:** The features Amount, robust_z_Amount, V27, robust_z_V23, V20, Time, V6, robust_z_V4, and V1 are the most influential in this prediction.
- **Decision Rules:** The model's decision is based on a combination of these features and their thresholds. For example, if the Amount is greater than 77.51 and the robust_z_Amount is greater than 1.85, the model is more likely to predict fraud.



MODEL EXPLAINABILITY USING PERMUTATION IMPORTANCE



Understanding the Visualization

The bar chart visualizes the feature importance scores obtained from a permutation importance analysis on an Isolation Forest model. It ranks the features based on their contribution to the model's ability to identify anomalies.

Key Insights from the Visualization:

- **Top Features:**
  V1 and robust_z_V23 are the most important features in predicting anomalies. They likely capture significant variations or outliers in the data.
- **Feature Groups:**
  Several features with similar names (e.g., V2, robust_z_V2) appear to be related, suggesting that certain groups of features might be more informative than others.
- **Model Sensitivity:** The model is sensitive to variations in these top-ranked features, and changes in their values can significantly impact the anomaly score.

1.1 OPTIMIZATION 1 USING OPTUNA

```
Confusion Matrix
[[263236  20017]
 [    53    420]]

Classification Report
              precision    recall  f1-score   support

           0       1.00      0.93      0.96    283253
           1       0.02      0.89      0.04       473

    accuracy                           0.93    283726
   macro avg       0.51      0.91      0.50    283726
weighted avg       1.00      0.93      0.96    283726

ROC AUC Score:
0.9086404941073118
```

True Negatives (TN): 263,236 (normal instances correctly identified)
False Positives (FP): 20,017 (normal instances incorrectly identified as anomalies)
False Negatives (FN): 53 (anomalies incorrectly identified as normal instances)
True Positives (TP): 420 (anomalies correctly identified)

Class 0 (Normal Instances):

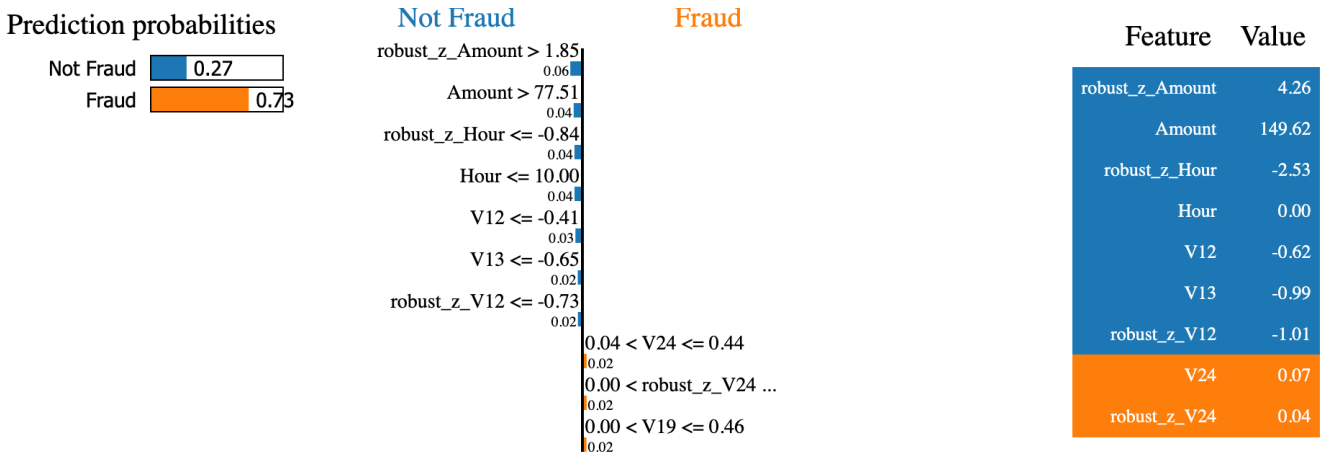
- **Precision**: 1.00 (high precision indicates almost no false positives for normal instances)
- **Recall**: 0.93 (high recall indicates most normal instances are correctly identified)
- **F1-Score**: 0.96 (harmonic mean of precision and recall)
- **Support**: 283,253 (number of actual occurrences in the dataset)

Class 1 (Anomalies):

- **Precision**: 0.02 (very low precision indicates a high number of false positives)
- **Recall**: 0.89 (high recall indicates most anomalies are detected)
- **F1-Score**: 0.04 (low F1 score due to the imbalance between precision and recall)
- **Support**: 473 (number of actual occurrences in the dataset)

The new model has a slightly higher recall for anomalies (0.89 vs. 0.84) but a lower precision (0.02 vs. 0.03), resulting in a similar F1-score.

MODEL EXPLAINABILITY USING LIME



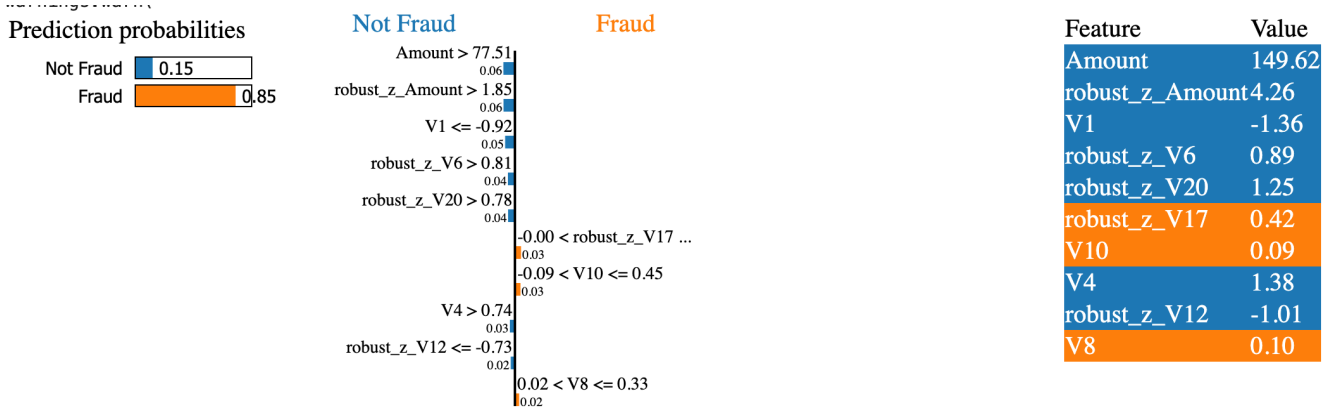
Key Features: The features robust_z_Amount, Amount, Hour, V12, robust_z_Hour are the most influential in this prediction.

1.2 OPTIMIZATION 2 - TUNING MODEL HYPER-PARAMETERS

```
Confusion Matrix:
[[84211   777]
 [   55    75]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.99      1.00     84988
           1       0.09      0.58      0.15       130

    accuracy                           0.99     85118
   macro avg       0.54      0.78      0.57     85118
weighted avg       1.00      0.99      0.99     85118

ROC AUC Score: 0.7838903048756204
```

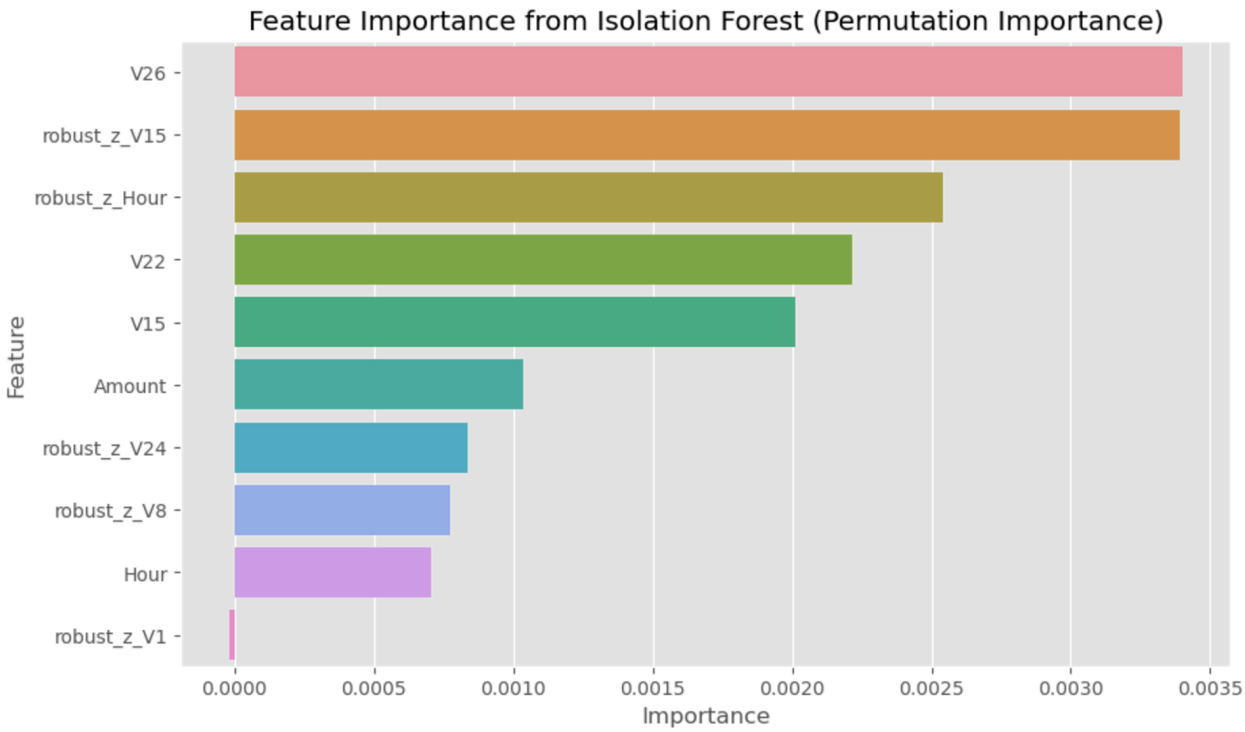
The low recall (0.58) for class 1 indicates that the model struggles to identify a significant portion of the anomalies.

MODEL EXPLAINABILITY USING LIME



Key Features: The features Amount, robust_z_Amount, V1, V6, V10, V4, V12, V8, and V20 are the most influential in this prediction.

MODEL EXPLAINABILITY USING PERMUTATION IMPORTANCE

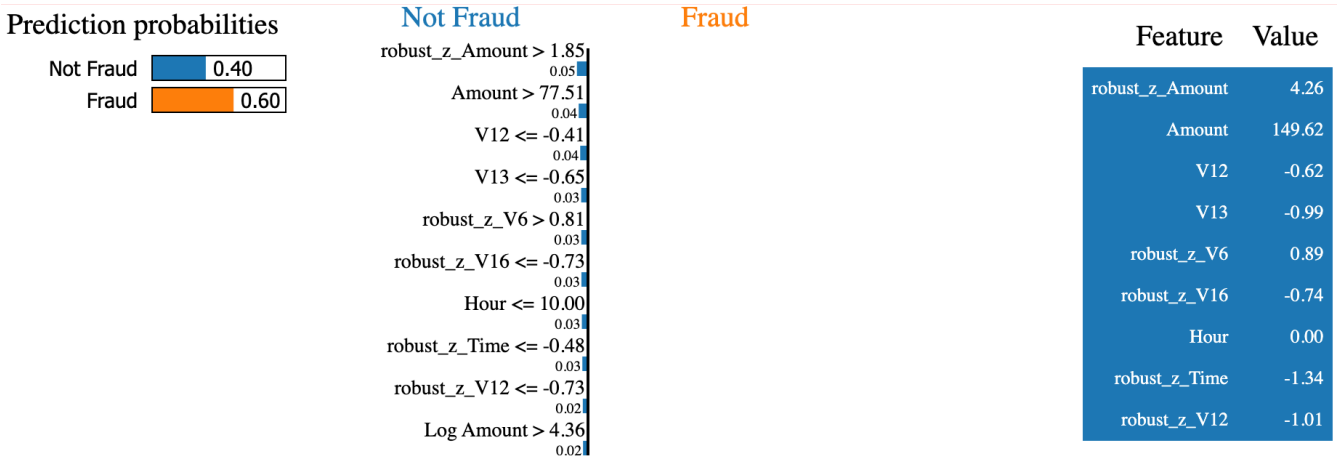


Top Features: V26 and robust_z_V15 are the most important features in predicting anomalies. They likely capture significant variations or outliers in the data.

1.3 OPTIMIZATION 3 - PARAMETERGRID

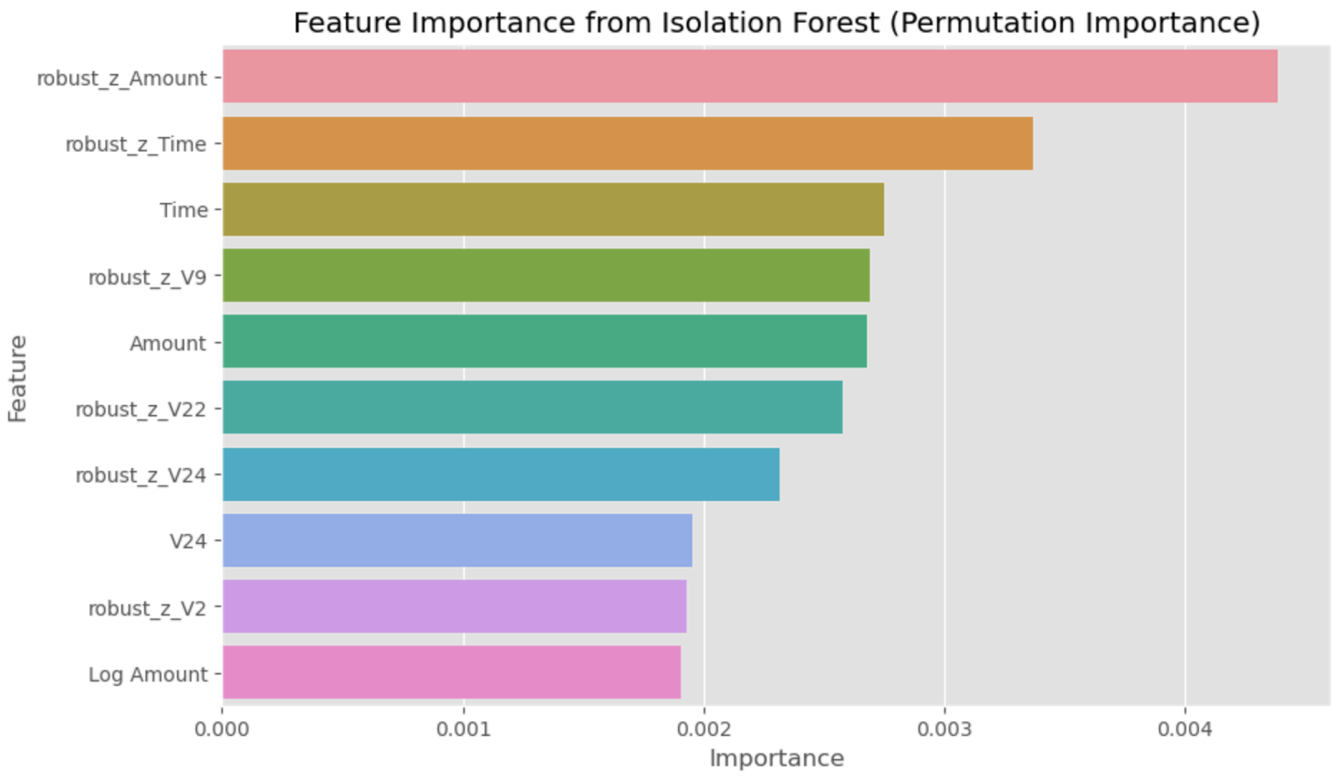
```
Confusion Matrix:
[[80841  4147]
 [   21   109]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.95      0.97     84988
           1       0.03      0.84      0.05       130

    accuracy                           0.95     85118
   macro avg       0.51      0.89      0.51     85118
weighted avg       1.00      0.95      0.97     85118

ROC AUC Score: 0.894833207222015
```

MODEL EXPLAINABILITY USING LIME



Key Features: The features robust_z_Amount, Amount, Hour, robust_z_V12, and V12 are the most influential in this prediction.

MODEL EXPLAINABILITY USING PERMUTATION IMPORTANCE

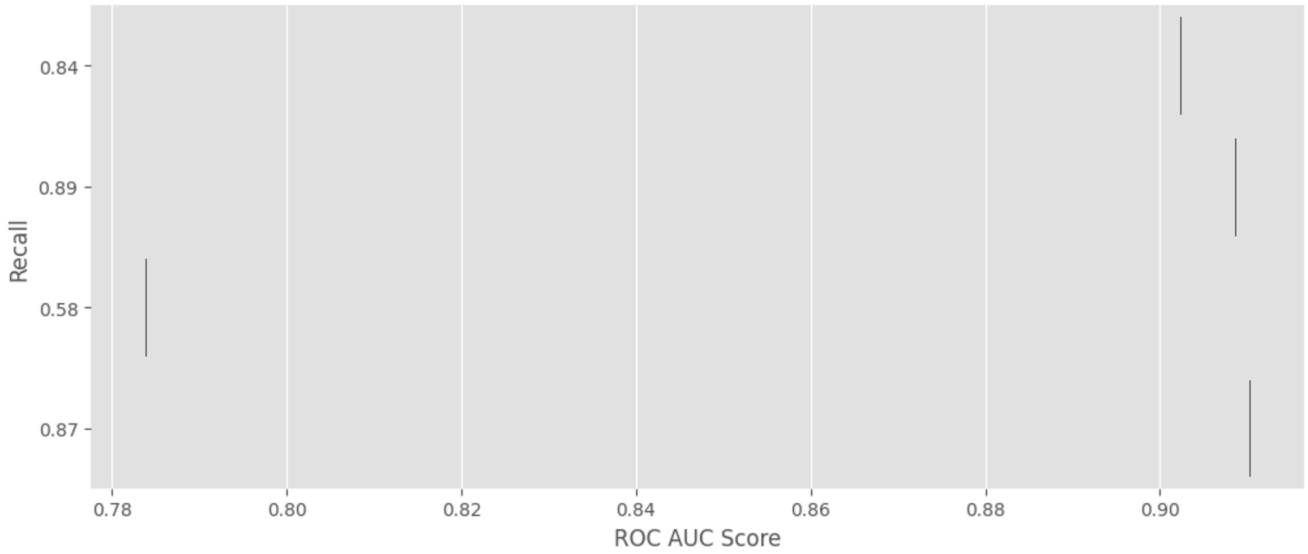


Top Features: robust_z_Amount and robust_z_Time are the most important features in predicting anomalies. They likely capture significant variations or outliers in the data related to transaction amount and time.

1.4 RESULTS COMPARISON

	Model	Confusion Matrix	ROC AUC Score	Precision	Recall	F1 Score
0	Vanilla Isolation Forest	[[272205, 11048], [74, 399]]	0.902274	0.03	0.84	0.07
1	Optimization - Optuna	[[263236, 20017], [53, 420]]	0.908640	0.02	0.89	0.04
2	Optimized Model - Tuning Hyperparameters	[[84211, 777], [55, 75]]	0.783890	0.09	0.58	0.15
3	ParameterGrid Optimization	[[80841, 4147], [21, 109]]	0.894833	0.03	0.84	0.05

- **Vanilla Isolation Forest**:
  - ✓ High recall (0.84) but low precision (0.03) and F1 score (0.07), indicating it catches most anomalies but with many false positives.
  - ✓ ROC AUC score is good at 0.902274.
- **Optimization - Optuna**:
  - ✓ Improved recall (0.89) compared to the vanilla model, but precision remains low (0.02).
  - ✓ Slightly higher ROC AUC score (0.908640), indicating a marginally better balance between sensitivity and specificity.
- **Optimized Model - Tuning Hyper-parameters**:
  - ✓ Significantly lower recall (0.58) but higher precision (0.09) and F1 score (0.15).
  - ✓ Lower ROC AUC score (0.783890) suggests it is not as good at distinguishing between normal instances and anomalies as the other models.
- **ParameterGrid Optimization**:
  - ✓ Similar recall (0.84) to the vanilla model but with a slightly lower precision (0.03) and F1 score (0.05).
  - ✓ ROC AUC score (0.894833) is slightly lower than the vanilla and Optuna optimized models but still high.



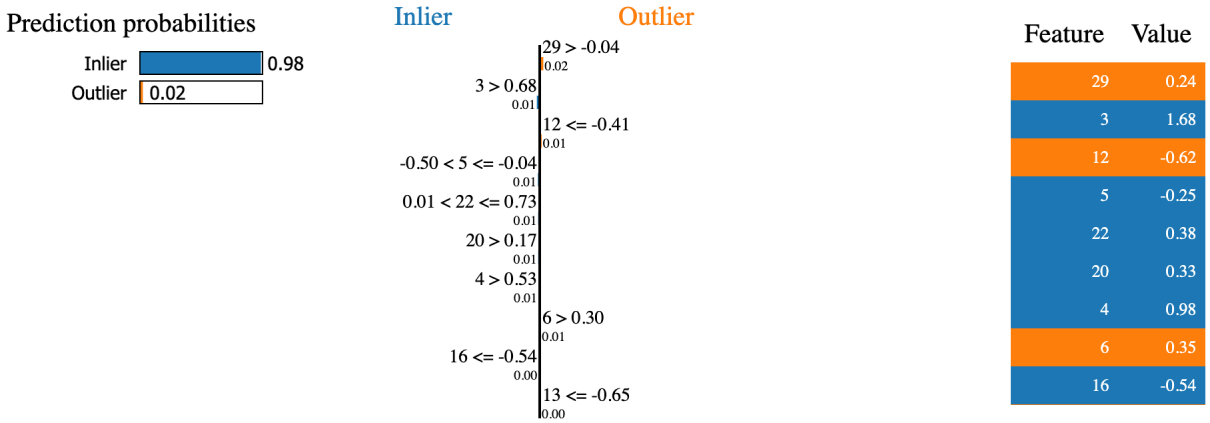
- **Choosing the Best Model**:
  The goal is to maximize anomaly detection (high recall): the Optuna-optimized model is preferable due to its high recall (0.89) and slightly better ROC AUC score (0.908640).

2.UNSUPERVISED VANILLA MODEL USING LOCAL OUTLIER FACTOR

```
Confusion Matrix
[[271183  12070]
 [   428     45]]

Classification Report
              precision    recall  f1-score   support

           0       1.00      0.96      0.98    283253
           1       0.00      0.10      0.01       473

    accuracy                           0.96    283726
   macro avg       0.50      0.53      0.49    283726
weighted avg       1.00      0.96      0.98    283726

ROC AUC Score:
0.5262626694701676
```

True Negatives (TN): 271183
False Positives (FP): 12070
False Negatives (FN): 428
True Positives (TP): 45
The recall for outliers is also very low (0.10), meaning the model is missing most of the actual outliers.
The ROC AUC score of 0.5263 indicates that the model's ability to distinguish between inliers and outliers is barely better than random guessing.

MODEL EXPLAINABILITY USING LIME

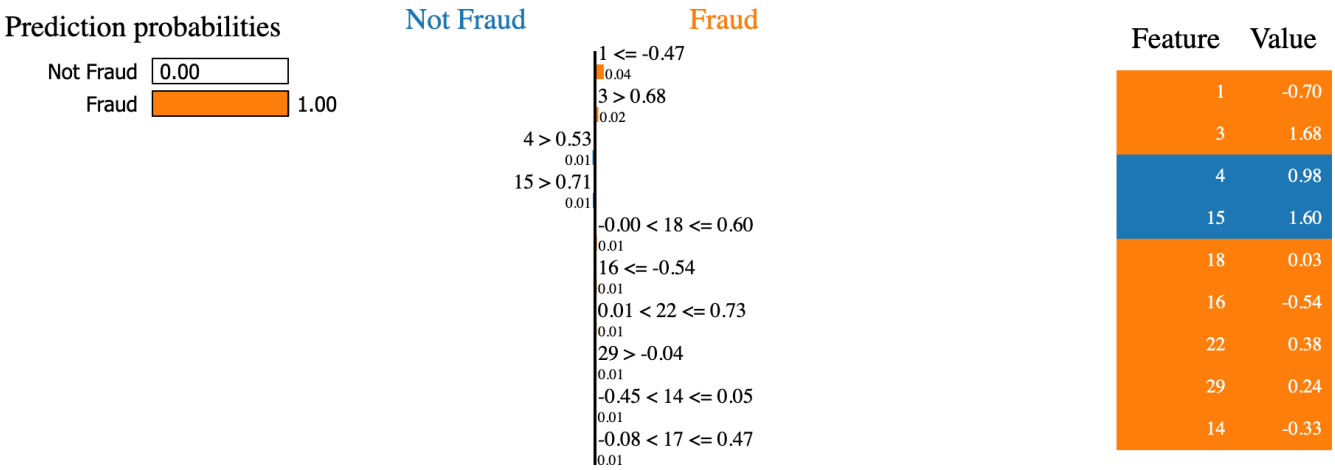


2.1 OPTIMIZATION 1- MANUAL SEARCH

```
Confusion Matrix
[[256986  26267]
 [   402     71]]

Classification Report
              precision    recall  f1-score   support

           0       1.00      0.91      0.95    283253
           1       0.00      0.15      0.01       473

    accuracy                           0.91    283726
   macro avg       0.50      0.53      0.48    283726
weighted avg       1.00      0.91      0.95    283726

ROC AUC Score:
0.5286861783945621
```

The ROC AUC score further confirms the model's poor performance.

MODEL EXPLAINABILITY USING LIME



2.2 OPTIMIZATION 2-USING OPTUNA

```
Confusion Matrix
[[83699  1289]
 [   24   106]]

Classification Report
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     84988
           1       0.08      0.82      0.14       130

    accuracy                           0.98     85118
   macro avg       0.54      0.90      0.57     85118
weighted avg       1.00      0.98      0.99     85118

ROC AUC Score:
0.9001088841501605
```

True Negatives (TN): 83,699
False Positives (FP): 1,289
False Negatives (FN): 24
True Positives (TP): 106
Class 0 (majority class)

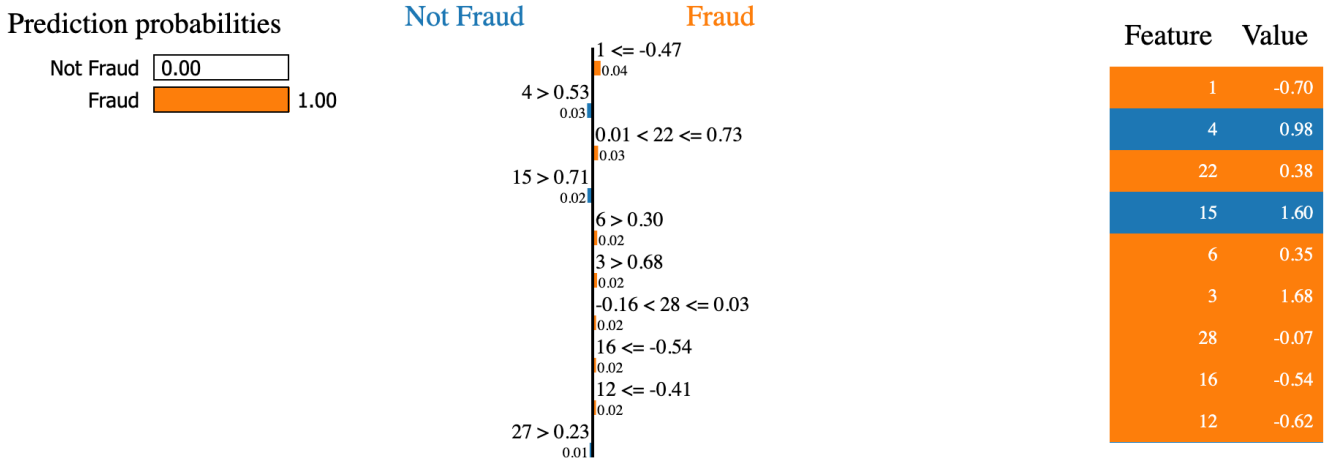
- **Recall:** 0.98 (98% of actual negatives are correctly identified)
- **F1-score:** 0.99 (High balance between precision and recall)
- **Support:** 84,988 (Number of actual instances for class 0)

Class 1 (minority class)

- **Recall:** 0.82 (82% of actual positives are correctly identified)

A ROC AUC score of 0.90 indicates that the model has a good ability to distinguish between the positive and negative classes, despite the imbalanced nature of the dataset.

MODEL EXPLAINABILITY USING LIME



The "Fraud" class has a probability of 1.00, suggesting that the model is highly confident in classifying this data point as fraudulent.

2.3 MODEL OPTIMIZATION - HYPERPARAMETER TUNING

```
Confusion Matrix
[[269147  14106]
 [   386     87]]

Classification Report
              precision    recall  f1-score   support

           0       1.00      0.95      0.97    283253
           1       0.01      0.18      0.01       473

    accuracy                           0.95    283726
   macro avg       0.50      0.57      0.49    283726
weighted avg       1.00      0.95      0.97    283726

ROC AUC Score:
0.5670661723023984
```

True Positives (TP): 269,147 (Correctly predicted as positive)
False Positives (FP): 14,106 (Incorrectly predicted as positive)
False Negatives (FN): 386 (Incorrectly predicted as negative)
True Negatives (TN): 87 (Correctly predicted as negative)
Recall: Ratio of correctly predicted positive observations to the all observations in the actual class.

- For class 0: 0.95 (Good, but not perfect)
- For class 1: 0.18 (Very low, indicating poor performance)

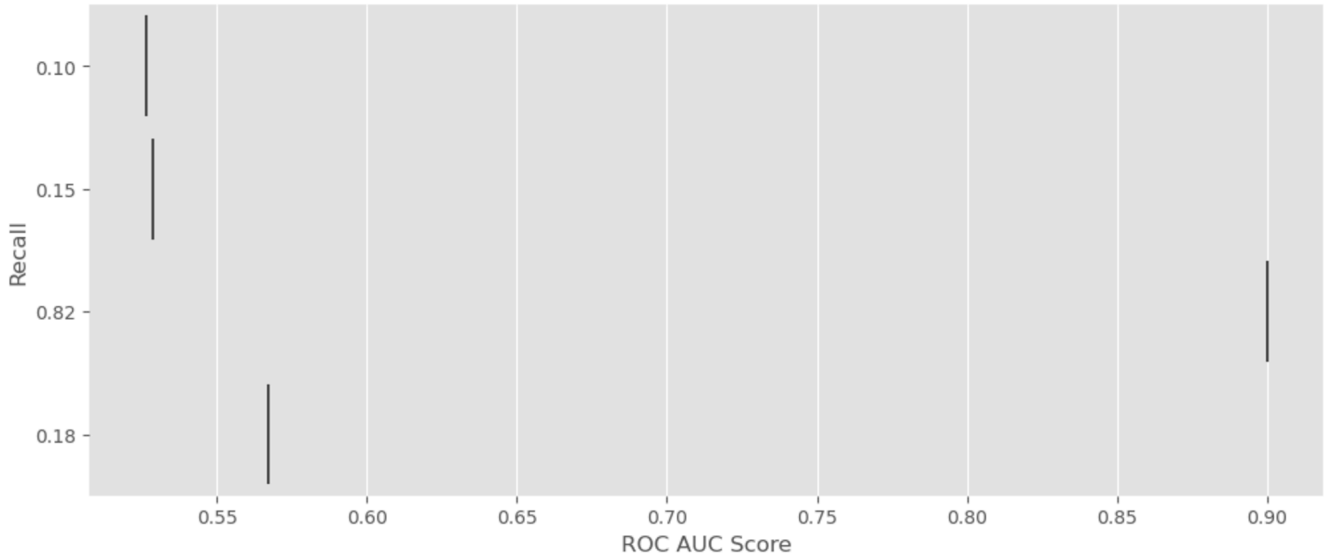
ROC AUC Score: 0.5671 (This score indicates a model that is barely better than random guessing)

2.4 COMPARISON OF RESULTS

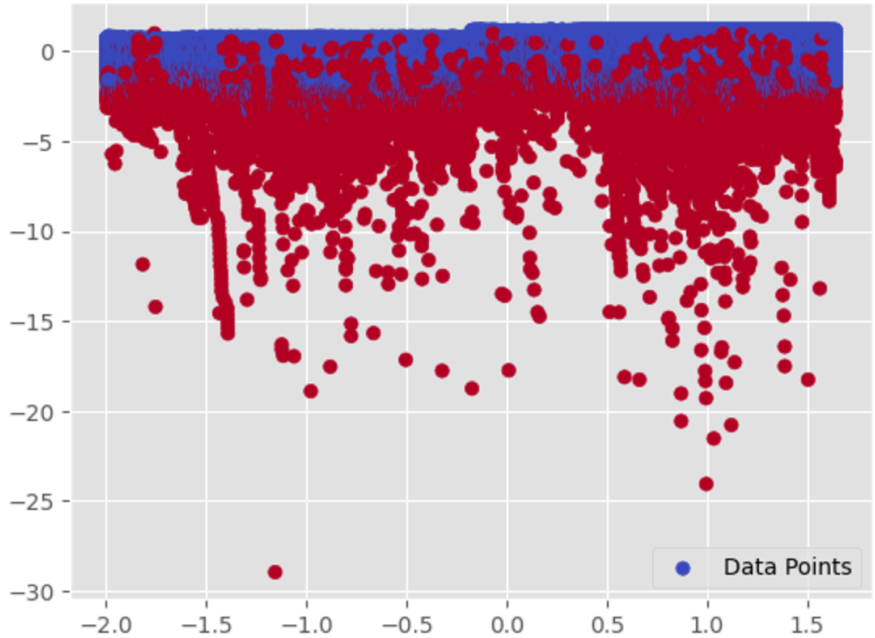
	Model	Confusion Matrix	ROC AUC Score	Precision	Recall	F1 Score
0	Vanilla LOF	[[271183, 12070], [428, 45]]	0.526263	0.00	0.10	0.01
1	Manual Search Optimization	[[256986, 26267], [402, 71]]	0.528686	0.00	0.15	0.01
2	Optuna Optimization	[[83699, 1289], [24, 106]]	0.900109	0.08	0.82	0.14
3	Optimized Model - Tuning Hyperparameters	[[269147, 14106], [386, 87]]	0.567066	0.01	0.18	0.01

The **Optuna Optimization** model outperforms the other models in all aspects, with a significantly higher ROC AUC score (0.900109) and much better precision (0.08), recall (0.82), and F1 score (0.14). It demonstrates a much better balance between identifying true positives and minimizing false positives compared to the other models.

The other models struggle with precision and F1 scores close to 0, indicating poor performance in correctly identifying the minority class (positives). Although there are minor improvements with Manual Search Optimization and Hyperparameter Tuning, they are not sufficient to outperform the Optuna Optimization model.



3. UNSUPERVISED VANILLA MODEL USING K-MEANS CLUSTERING

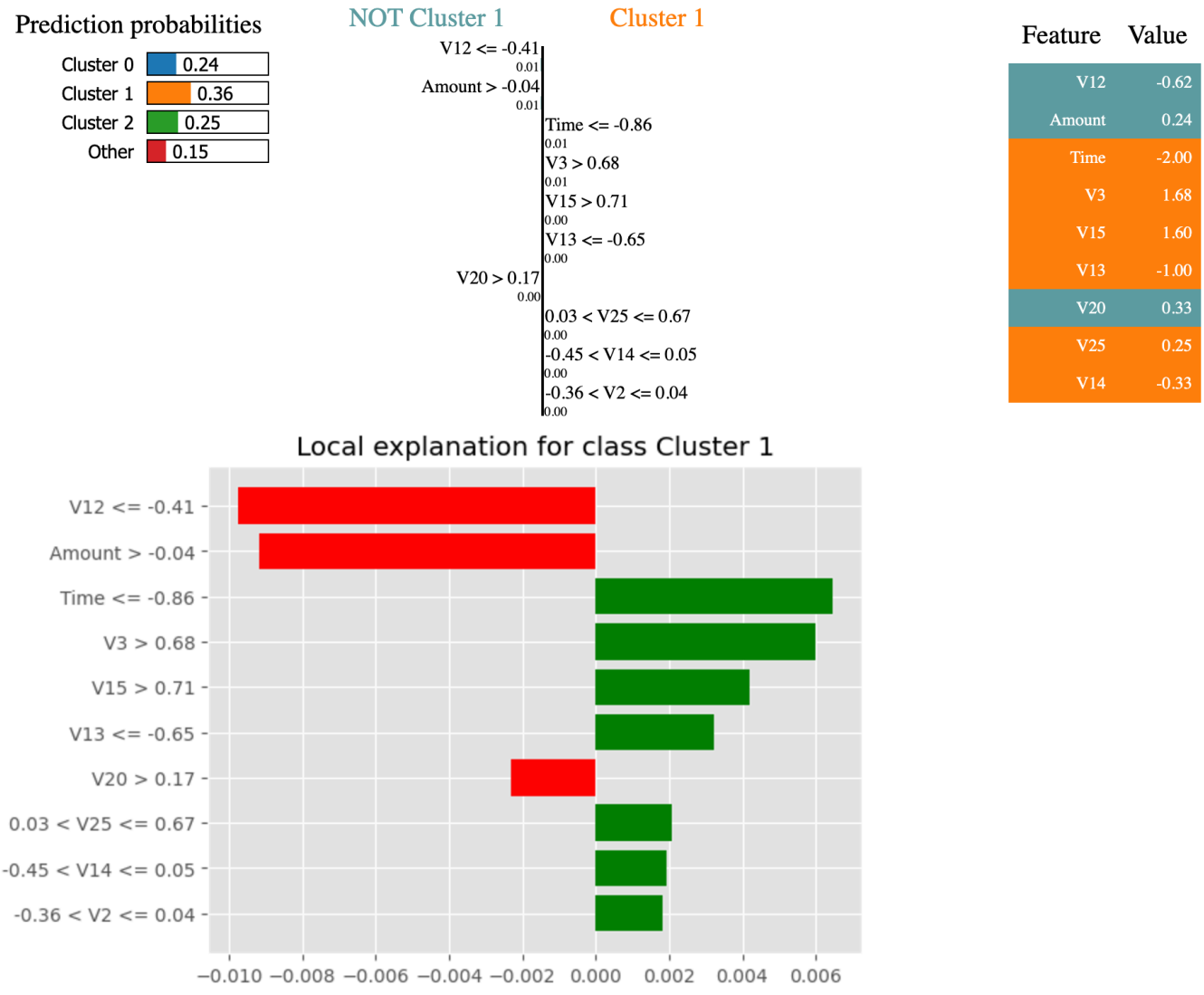



```
Anomalies:[     7     18     49 ... 283722 283723 283724]
```

```
Confusion Matrix
[[269478  13775]
 [    61    412]]
Classification Report
              precision    recall  f1-score   support

           0       1.00      0.95      0.97    283253
           1       0.03      0.87      0.06       473

    accuracy                           0.95    283726
   macro avg       0.51      0.91      0.52    283726
weighted avg       1.00      0.95      0.97    283726

ROC AUC Score
0.9112022526511291
```

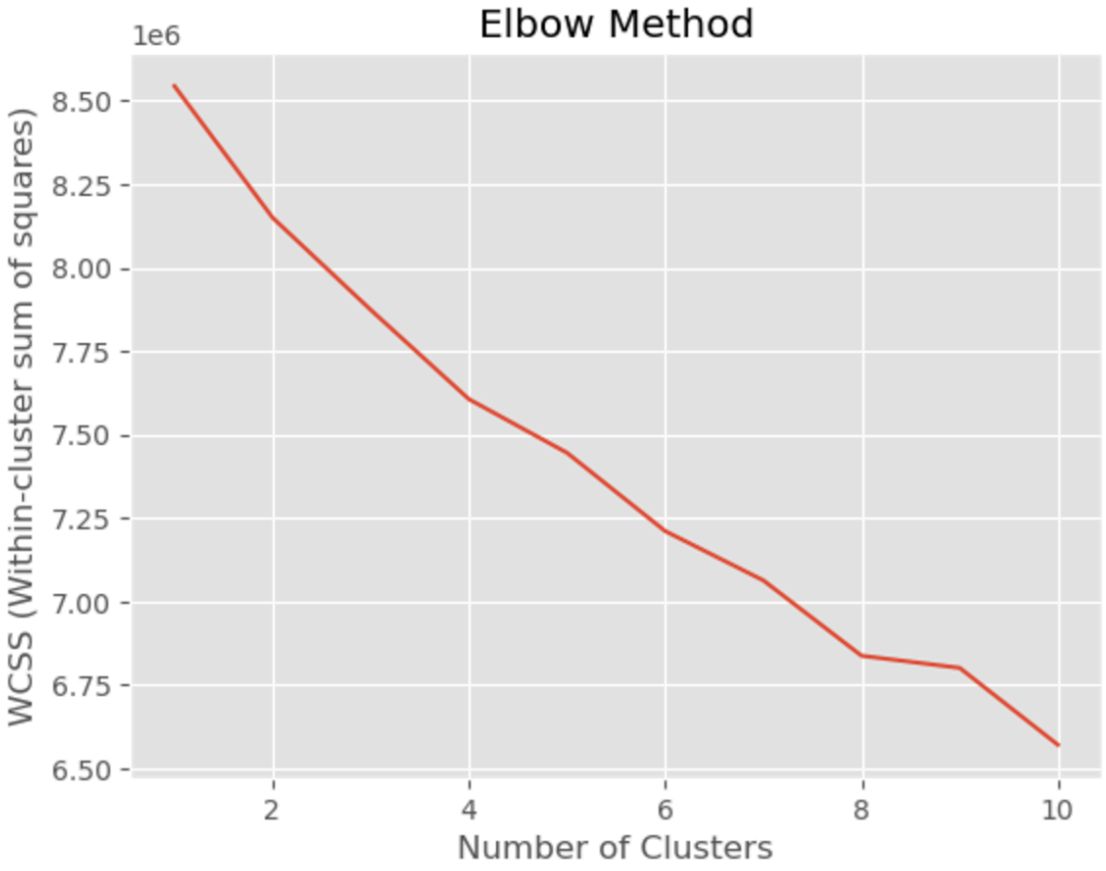
The recall of 0.87 suggests that the model can identify a significant portion of the actual anomalies.

MODEL EXPLAINABILITY USING LIME



V12: A value of -0.62 for this feature, combined with the condition V12 <= -0.41, positively contributes to the assignment to Cluster 1.
Amount: A value of 0.24 for this feature, combined with the condition Amount > -0.04, positively contributes to the assignment to Cluster 1.
Time: A value of -2.00 for this feature, combined with the condition Time <= -0.86, positively contributes to the assignment to Cluster 1.
V3: A value of 1.68 for this feature, combined with the condition V3 > 0.68, positively contributes to the assignment to Cluster 1.
V15: A value of 1.60 for this feature, combined with the condition V15 > 0.71, positively contributes to the assignment to Cluster 1.

3.1 OPTIMIZATION 1 - DETERMINING THE OPTIMAL NUMBER OF CLUSTERS USING ELBOW METHOD



The WCSS is the sum of the squared distances of each data point to its cluster centroid.

Interpreting the Plot

In the provided plot, we observe a decreasing trend in WCSS as the number of clusters increases. This is expected, as more clusters generally lead to a better fit to the data.

Identifying the "Elbow Point"

The "elbow point" is the point on the curve where the rate of decrease in WCSS begins to slow down significantly. This point suggests that adding more clusters beyond this point would result in diminishing returns. In this plot, the elbow point appears to be around **4 clusters**.

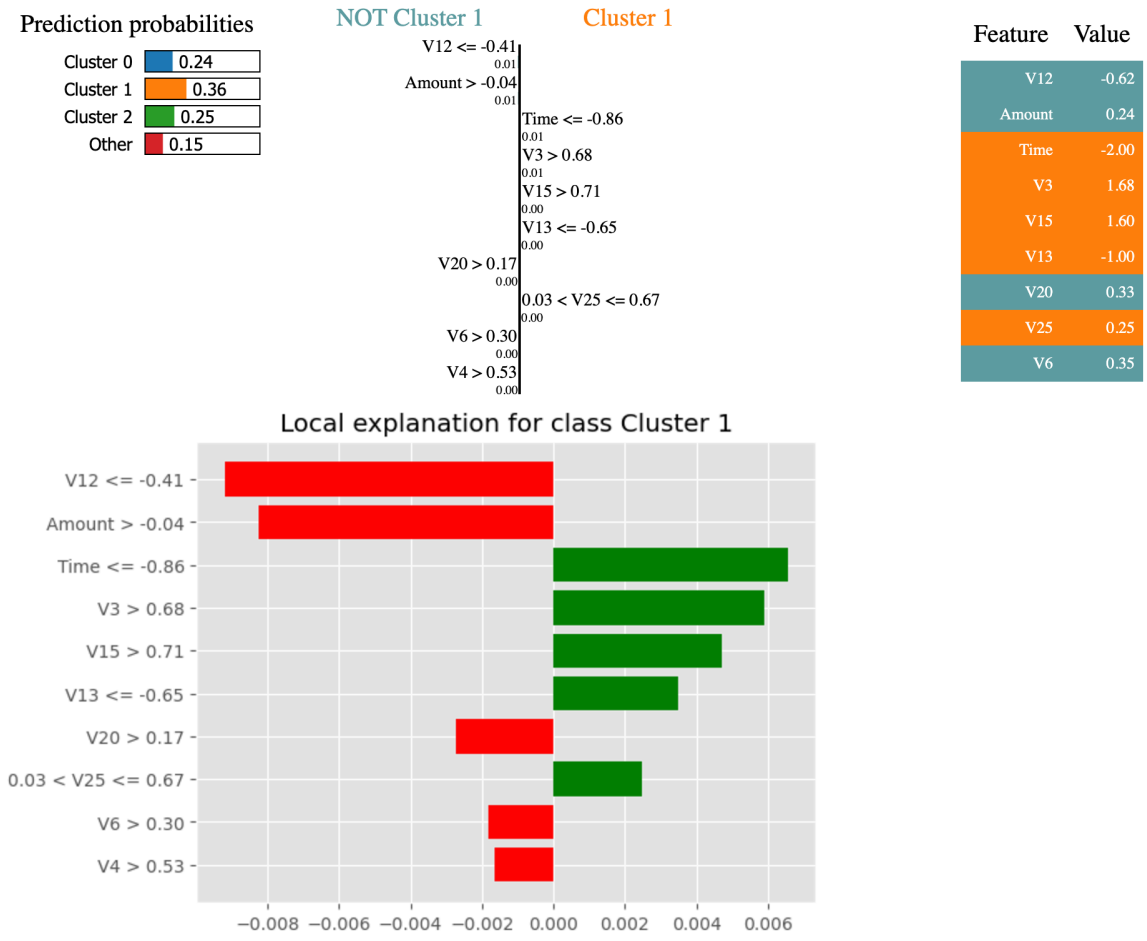
Choosing the Optimal Number of Clusters

Based on the elbow point, it's recommended to choose 4 clusters for this dataset. This number strikes a balance between maximizing the explained variance and minimizing the number of clusters.

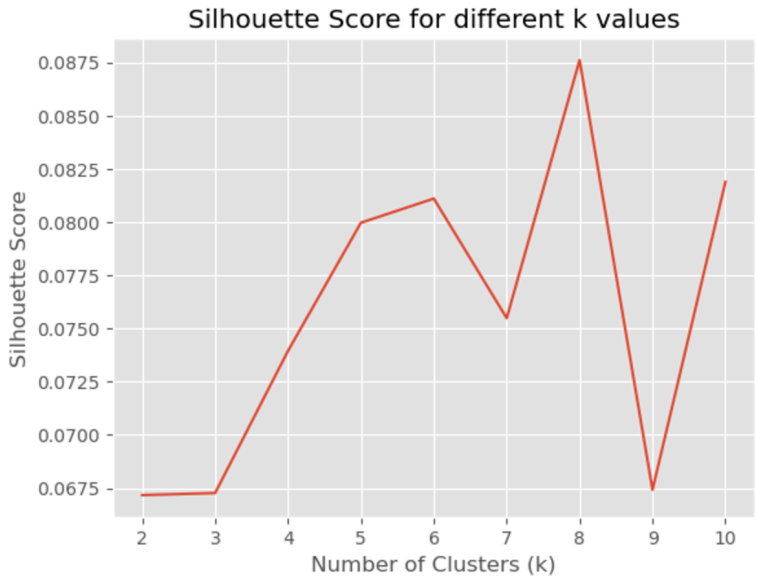
```
Confusion Matrix
[[269476  13777]
 [    63    410]]
Classification Report
              precision    recall  f1-score   support

           0       1.00      0.95      0.97    283253
           1       0.03      0.87      0.06       473

    accuracy                           0.95    283726
   macro avg       0.51      0.91      0.52    283726
weighted avg       1.00      0.95      0.97    283726

ROC AUC Score
0.909084557333526
```

Analysis: Performance similar to Vanilla KMeans, with high recall and ROC AUC but low precision.
True Positives (TP): 269476 (Correctly predicted as class 0)
False Positives (FP): 13777 (Incorrectly predicted as class 0)
False Negatives (FN): 63 (Incorrectly predicted as class 1)
True Negatives (TN): 410 (Correctly predicted as class 1)

MODEL EXPLAINABILITY USING LIME



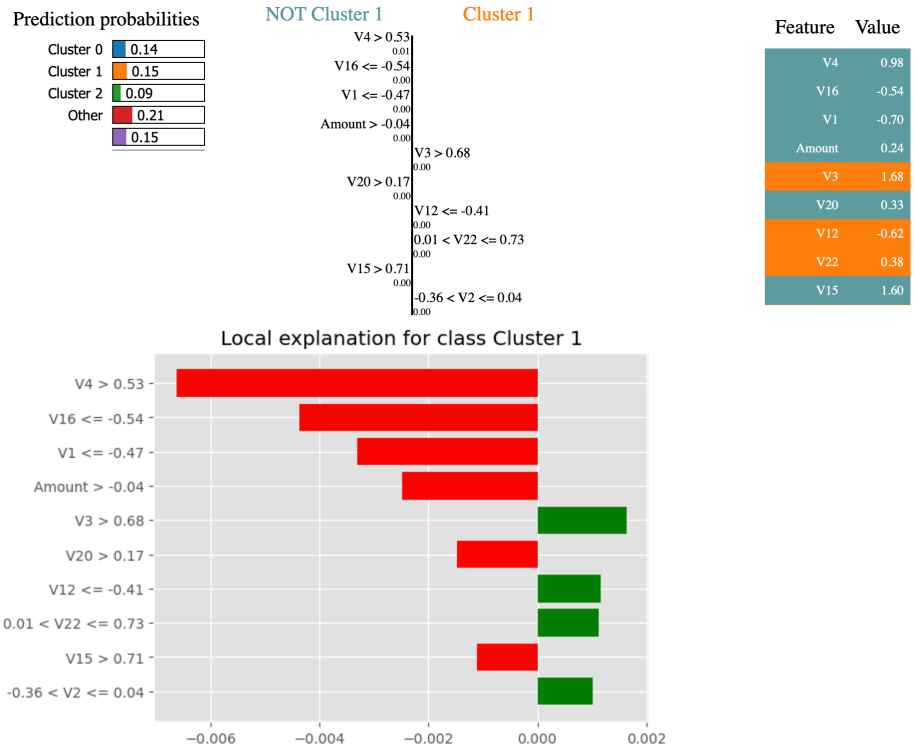
3.2OPTIMIZATION 2 - DETERMINING THE OPTIMAL NUMBER OF CLUSTERS USING SILHOUETTE SCORE



The silhouette score is a metric used to evaluate the quality of clustering.
Peak at k=8: The plot shows a peak at k=8, indicating that this might be the optimal number of clusters. At this point, the data points are, on average, most similar to their own cluster members and least similar to members of other clusters.
Decreasing Trend: After k=8, the silhouette score starts to decrease. This suggests that adding more clusters might not improve the clustering quality.

```
Confusion Matrix
[[277168   6085]
 [   110    363]]
Classification Report
              precision    recall  f1-score   support

           0       1.00      0.98      0.99    283253
           1       0.06      0.77      0.10       473

    accuracy                           0.98    283726
   macro avg       0.53      0.87      0.55    283726
weighted avg       1.00      0.98      0.99    283726

ROC AUC Score
0.8729796494694242
```

MODEL EXPLAINABILITY USING LIME



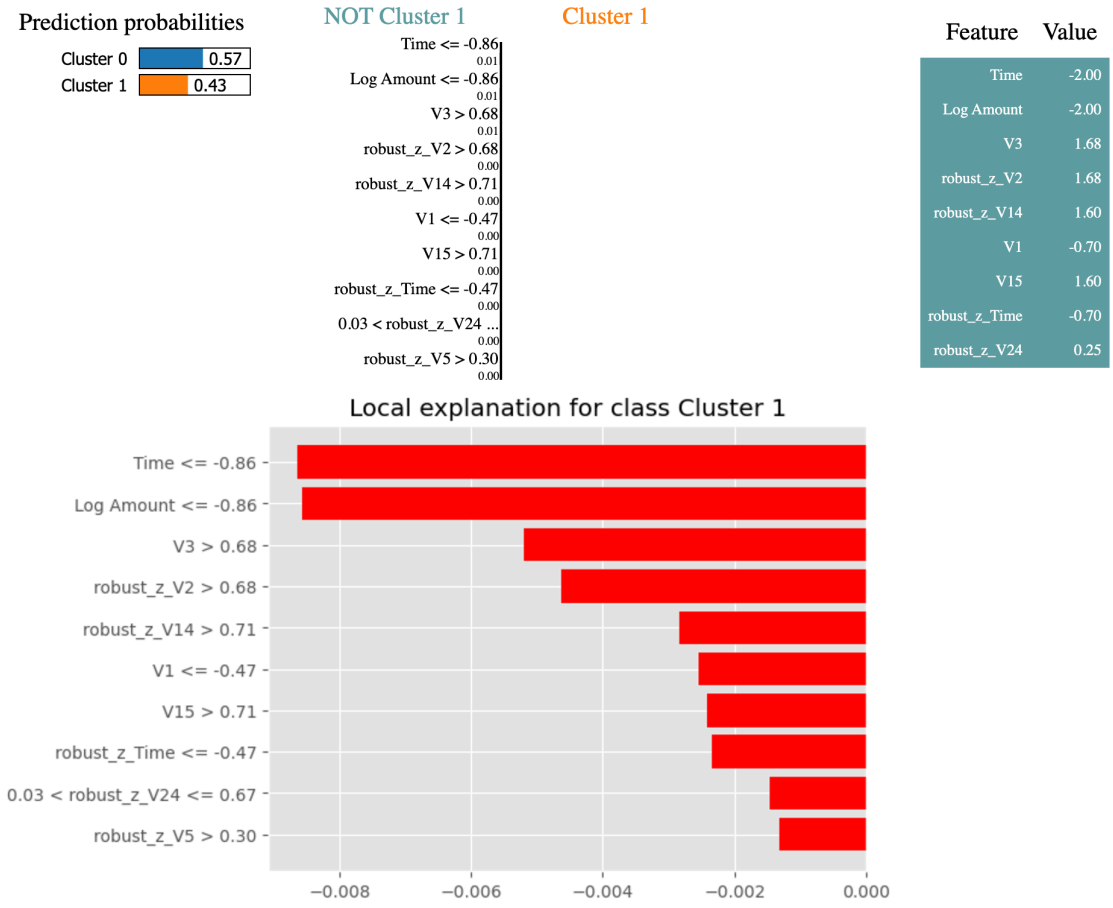
V4: A value of 0.98 for this feature, combined with the condition V4 > 0.53, positively contributes to the assignment to Cluster 1.
V16: A value of -0.54 for this feature, combined with the condition V16 <= -0.54, positively contributes to the assignment to Cluster 1.
V1: A value of -0.70 for this feature, combined with the condition V1 <= -0.47, positively contributes to the assignment to Cluster 1.
Amount: A value of 0.24 for this feature, combined with the condition Amount > -0.04, positively contributes to the assignment to Cluster 1.
V3: A value of 1.68 for this feature, combined with the condition V3 > 0.68, positively contributes to the assignment to Cluster 1.

3.3 OPTIMIZATION 3- IMPROVE INITIALIZATION

```
Confusion Matrix
[[276571   6682]
 [    91    382]]
Classification Report
              precision    recall  f1-score   support

           0       1.00      0.98      0.99    283253
           1       0.05      0.81      0.10       473

    accuracy                           0.98    283726
   macro avg       0.53      0.89      0.54    283726
weighted avg       1.00      0.98      0.99    283726

ROC AUC Score
0.8920103878625635
```

Analysis: Similar to the Silhouette Score optimization but with slightly lower recall.

MODEL EXPLAINABILITY USING LIME



Conditions such as Time <= -0.86 or V3 > 0.68 show how the model evaluates specific feature values to arrive at a decision.
The decision rules highlight which features are crucial in making predictions. Features like Time, Log Amount, V3, robust_z_V2, robust_z_V14, etc., are identified as key variables that influence the outcome.

3.4 RESULTS COMPARISON

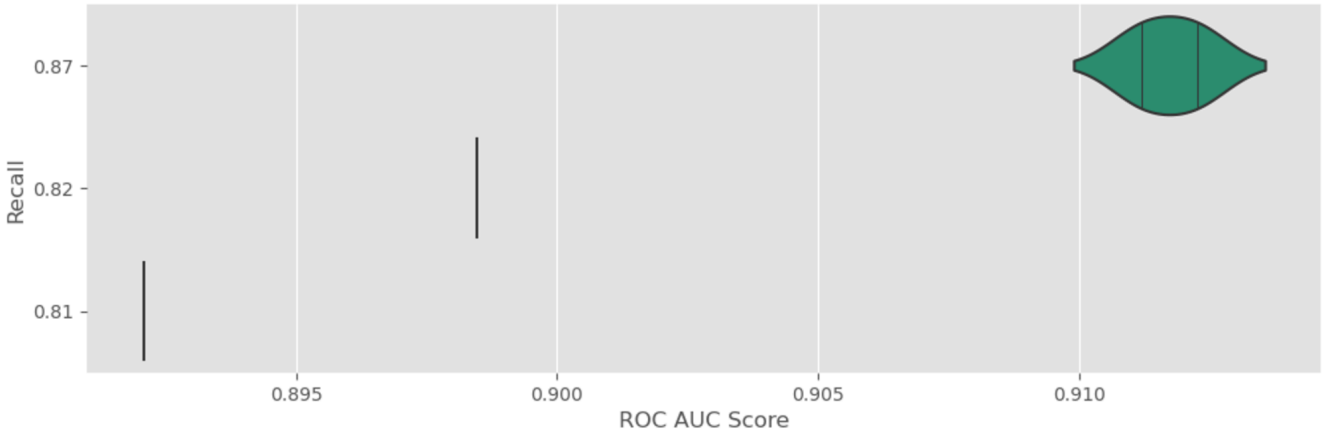
- **ROC AUC Score**:
  - ✓ The **Vanilla KMeans Model** and **Optimization - Elbow Method** perform the best with ROC AUC scores of 0.9112 and 0.9091 respectively. This indicates that these models have the best ability to distinguish between anomalies and normal data.
  - ✓ **Optimization - Silhouette Score** has a slightly lower ROC AUC score of 0.8730, indicating a less accurate ability to separate the two classes.
  - ✓ **Improve Initialization Optimization** has a score of 0.8920, indicating moderate performance in separating the two classes.

- **Recall**:

✓ **Recall** is higher for most models, particularly in **Vanilla KMeans Model** and **Optimization - Elbow Method**, where recall is 0.87. This means that these models are good at identifying the true anomalies (i.e., they catch most of the actual anomalies).

✓ **Optimization - Silhouette Score** and **Improve Initialization Optimization** also show good recall (0.77 to 0.81), meaning they correctly identify a large portion of the anomalies.

	Model	Confusion Matrix	ROC AUC Score	Precision	Recall	F1 Score
0	Vanilla KMeans Model	[[269478, 13775], [61, 412]]	0.911202	0.03	0.87	0.06
1	Optimization - Elbow Method	[[269476, 13777], [63, 410]]	0.909085	0.03	0.87	0.06
2	Optimization - Silhouette Score	[[277168, 6085], [110, 363]]	0.872980	0.06	0.77	0.10
3	Improve Inialization Optimization	[[276571, 6682], [91, 382]]	0.892010	0.05	0.81	0.10



4. CONCLUSION - EVALUATION

Metrics used:

Recall: To ensure as many fraudulent transactions as possible are detected.

Precision: To ensure that the fraudulent transactions detected by the model are actually fraud, minimizing false positives.

F1 Score: If a balance between precision and recall is required, the F1 score is a good metric, especially in a scenario where both false negatives (missed fraud) and false positives (mislabeling legitimate transactions) are important.

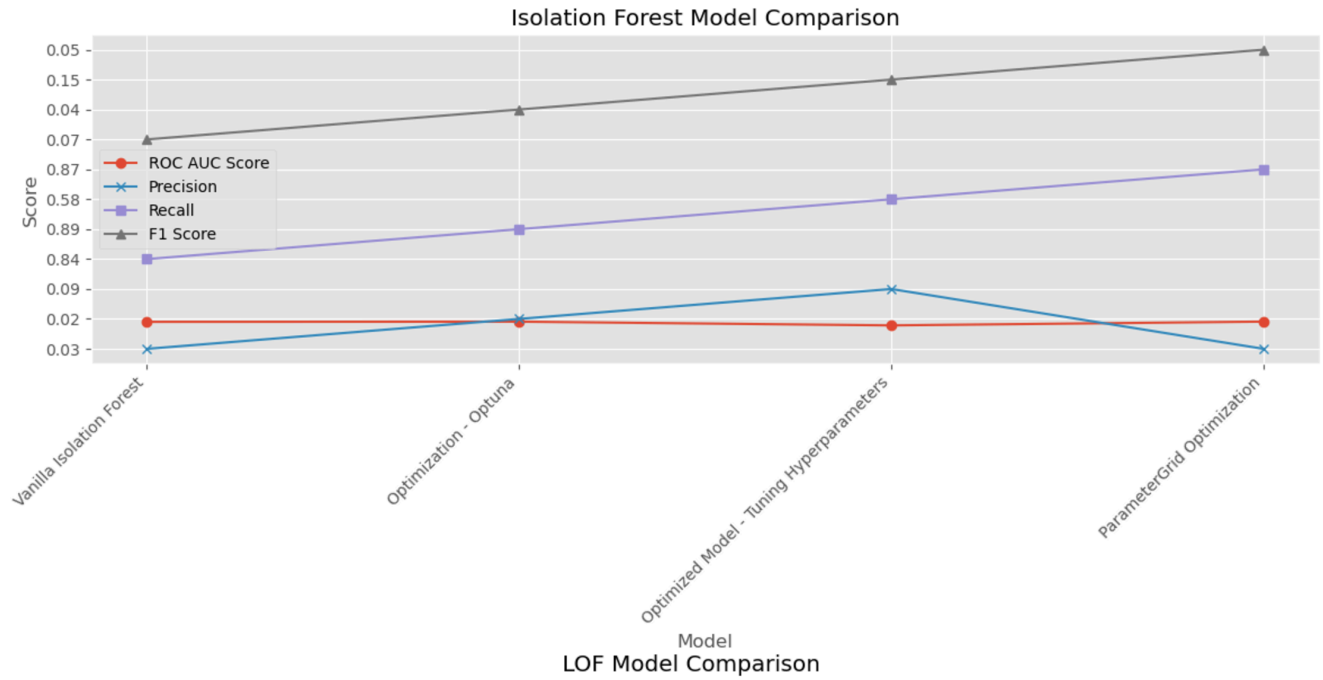
ROC AUC Score: For a general assessment of the model's ability to distinguish between fraudulent and legitimate transactions across all thresholds. A high ROC AUC is an indicator that the model can discriminate well between classes.

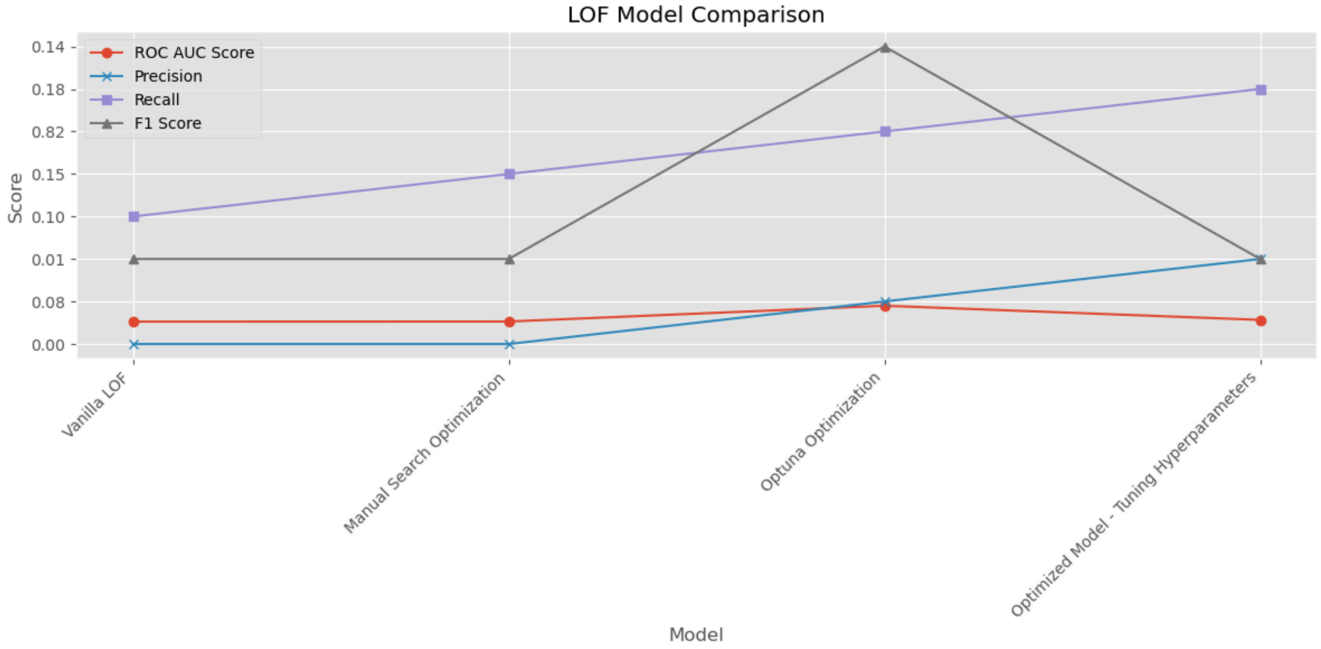
For credit card fraud detection, **recall** (also known as **sensitivity** or the **true positive rate**) is typically the most important metric.

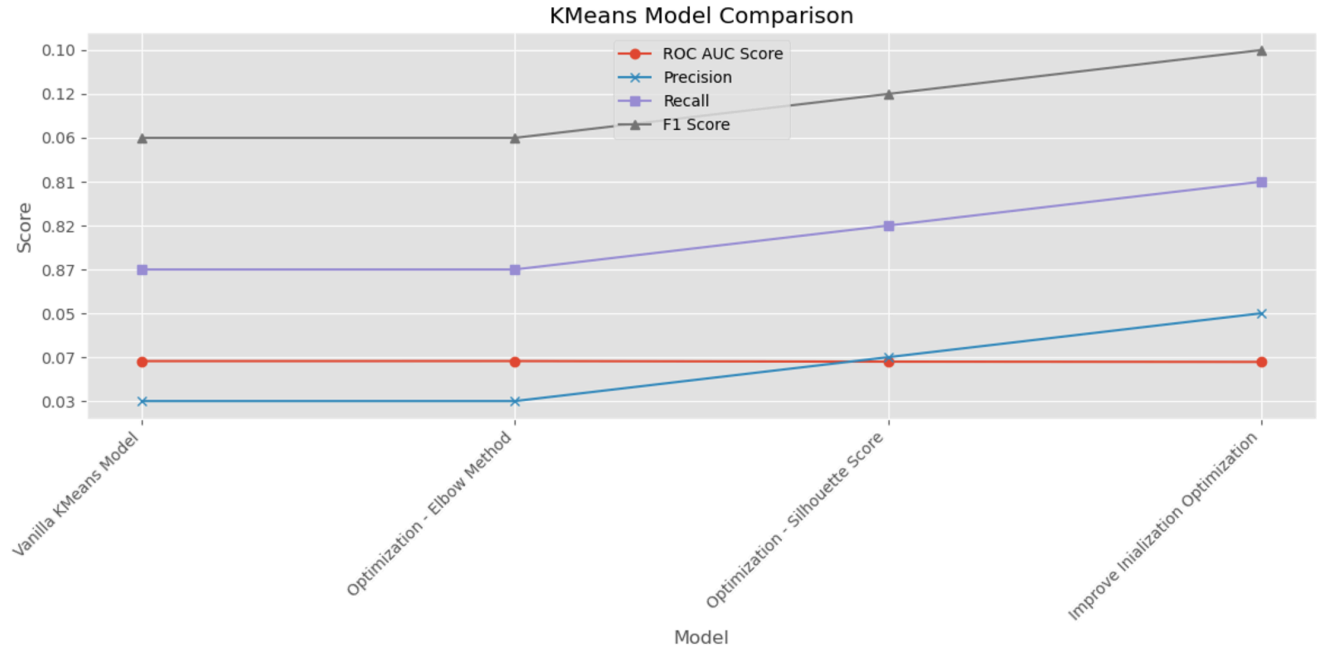
Reasons Why Recall is Critical for Credit Card Fraud Detection:

- **Cost of False Negatives (Missed Fraud)**: In fraud detection, missing a fraudulent transaction (false negative) can be very costly for both the cardholder and the financial institution. Recall measures the proportion of actual fraud cases that are successfully detected, so prioritizing recall ensures that you capture as many fraudulent transactions as possible.
- **False Positives (False Alarms)**: While high recall is important, credit card fraud detection also needs to manage false positives (incorrectly flagged legitimate transactions). False positives can lead to customer frustration, transaction disruptions, and the cost of investigating false alarms. Therefore, some **precision** is also important to minimize unnecessary alerts.

PROJECT 4


Isolation Forest Model Comparison
ROC AUC Score
Precision
Recall
F1 Score
Score
0.05
0.15
0.04
0.07
0.87
0.58
0.89
0.84
0.09
0.02
0.03
Vanilla Isolation Forest
Optimization - Optuna
Optimized Model - Tuning Hyperparameters
ParameterGrid Optimization
Model


LOF Model Comparison
ROC AUC Score
Precision
Recall
F1 Score
Score
0.14
0.18
0.82
0.15
0.10
0.01
0.08
0.00
Vanilla LOF
Manual Search Optimization
Optuna Optimization
Optimized Model - Tuning Hyperparameters
Model


KMeans Model Comparison
ROC AUC Score
Precision
Recall
F1 Score
Score
0.10
0.12
0.06
0.81
0.82
0.87
0.05
0.07
0.03
Vanilla KMeans Model
Optimization - Elbow Method
Optimization - Silhouette Score
Improve Inialization Optimization
Model

- **Best Performers**:
  - ✓ Isolation Forest : Optimization - Optuna: Highest recall and good ROC AUC score but very low precision.
  - ✓ KMeans Clustering : Optimization - Elbow Method: The better performer due to its higher recall score
- **Poor Performers**:
  - ✓ **Vanilla LOF** and **Manual Search Optimization**: Very low performance across all metrics.
  - ✓ **Optimized Model - Tuning Hyperparameters**: Consistently low precision, recall, and F1 score.
- **Balanced Performers**:
  - ✓ **Vanilla Isolation Forest**: High recall with moderate precision.
  - ✓ **Isolation Forest : ParameterGrid Optimization**: Good balance with high recall and ROC AUC.

	Model	Confusion Matrix	ROC AUC Score	Precision	Recall	F1 Score
0	Vanilla Isolation Forest	[[272205, 11048], [74, 399]]	0.902274	0.03	0.84	0.07
1	Optimization - Optuna	[[263236, 20017], [53, 420]]	0.908640	0.02	0.89	0.04
2	Optimized Model - Tuning Hyperparameters	[[84211, 777], [55, 75]]	0.783890	0.09	0.58	0.15
3	ParameterGrid Optimization	[[80845, 4143], [17, 113]]	0.910241	0.03	0.87	0.05
4	Vanilla LOF	[[271183, 12070], [428, 45]]	0.526263	0.00	0.10	0.01
5	Manual Search Optimization	[[256986, 26267], [402, 71]]	0.528686	0.00	0.15	0.01
6	Optuna Optimization	[[83699, 1289], [24, 106]]	0.900109	0.08	0.82	0.14
7	Optimized Model - Tuning Hyperparameters	[[269147, 14106], [386, 87]]	0.567066	0.01	0.18	0.01
8	Vanilla KMeans Model	[[269478, 13775], [61, 412]]	0.911202	0.03	0.87	0.06
9	Optimization - Elbow Method	[[269479, 13774], [60, 413]]	0.912261	0.03	0.87	0.06
10	Optimization - Silhouette Score	[[277837, 5416], [87, 386]]	0.898473	0.07	0.82	0.12
11	Improve Inialization Optimization	[[276604, 6649], [91, 382]]	0.892069	0.05	0.81	0.10