NORTH TEXAS ISSA

Detecting Fraud Using Machine Learning Models

Aryan Pillai - Liberty High School Senior

https://www.linkedin.com/in/apillai2701/

Speaker

Aryan Pillai

Former Independent Study & Mentorship Student at Frisco ISD.

Moderator

Rachael Ridenour

Cybersecurity

- Cybersecurity encompasses technologies, processes, and practices designed to protect networks, devices, programs, and data from attack, damage, or unauthorized access.
- Key Concepts
  - Threats: Fraud, malware, phishing, ransomware, DDoS attacks, etc.
  - Vulnerabilities: Weak passwords, unpatched software, misconfigured systems, etc.
  - Defense Mechanisms: Firewalls, antivirus software, intrusion detection systems (IDS), encryption, etc.

Machine Learning

- Effective Techniques to enable human-like learning
  - Techniques such as logistic regression, decision trees, and random forests emerged as effective tools for detecting and combating cyber risks.
- Utilization in Final Project
  - Leveraged machine learning algorithms in constructing the final solution, a robust fraud detection system.

Supervised vs. Unsupervised Machine Learning

Machine Learning (ML) involves training algorithms to make predictions or decisions based on data.
Two main types of ML: Supervised and Unsupervised learning.

Supervised ML

- **Definition**: Learning from labeled data where the outcome is known.
- **Goal**: Train a model to predict the output for new, unseen data.
- **Examples**: Fraud detection, email spam classification, predicting stock prices.
- **Algorithms**: Logistic Regression, Random Forest, Support Vector Machines.

Unsupervised ML

- **Definition**: Learning from data without labeled outcomes; the model finds patterns on its own.
- **Goal**: Discover hidden patterns, groupings, or structures in data.
- **Examples**: Customer segmentation, anomaly detection, topic modeling.
- **Algorithms**: K-means Clustering, Principal Component Analysis (PCA), Hierarchical Clustering.

Supervised ML is the ideal choice for this, as fraud detection often requires predicting whether a transaction is fraudulent (yes/no).

- Labeled data (fraudulent vs. non-fraudulent) is used to train models that classify new transactions.
- Accuracy & Precision: Supervised models can be optimized to reduce false positives/negatives, crucial in fraud prevention.

We will proceed with the analysis with using **Supervised Machine Learning**.

Goals

1. **Analyze the Data Distribution**
   - Understand the characteristics and distribution of the provided dataset.
   - Identify key features and patterns within the data.
2. **Create a Balanced Dataset**
   - Use Random Under Sampling (RUS) to achieve a 50/50 ratio of "Fraud" and "Non-Fraud" transactions.
   - Ensure the dataset is balanced to enhance model training and performance.
3. **Address Data Imbalances and Anomalies**
   - Apply anomaly detection techniques to identify and remove extreme outliers.
4. **Optimize the Model Performance**
   - Enhance the accuracy and effectiveness of the fraud detection model.
   - Ensure that the data used for training reflects a realistic and balanced representation of fraudulent and non-fraudulent transactions.

Dataset Introduction

- The dataset contains credit card transactions made in September 2013 by European cardholders. This dataset contains transactions that occurred in two days, with 492 frauds out of 284,807 total. The dataset is **extremely unbalanced**, with the positive class (frauds) accounting for **0.172%** of total transactions. **Non-Fraud (99.83%).**

- Due to confidentiality, it contains only numeric input variables, which are variables that have values that can be described as numbers, and can be used to measure quantities, which are the result of a PCA transformation.

- We are not provided the original features and more background information about the data.

- Feature Class is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Dataset Overview:

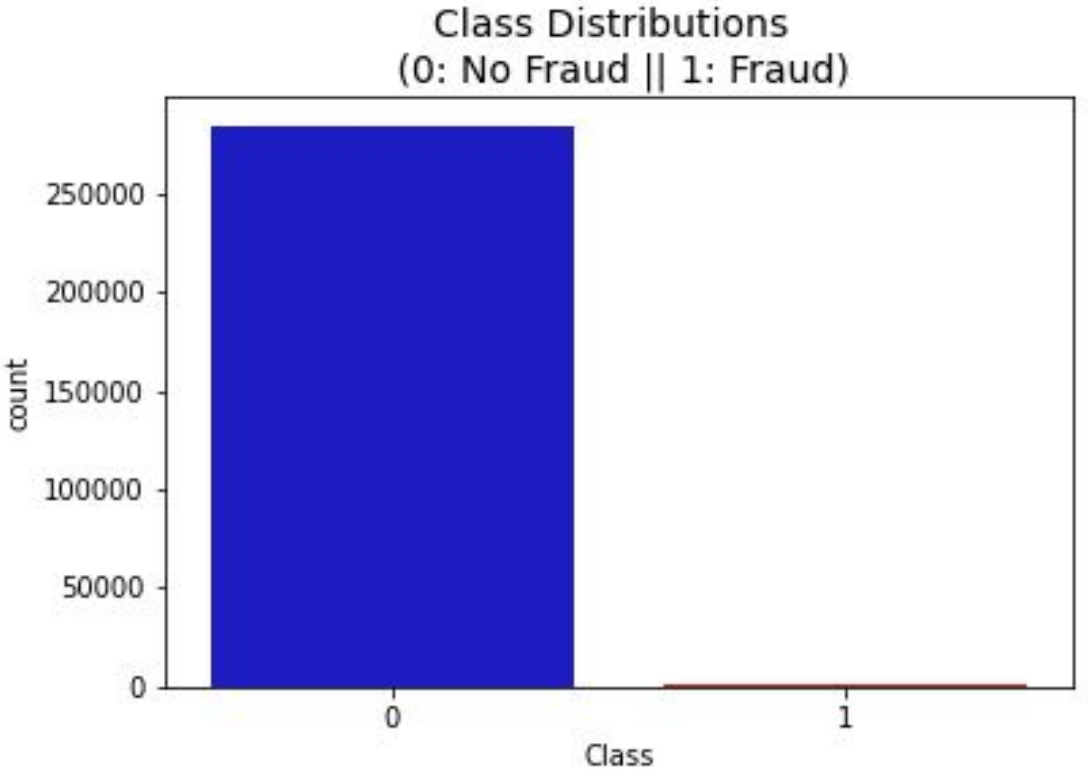
- The dataset contains transaction data labeled as either "Fraud" (1) or "No Fraud" (0).
- The data is highly imbalanced, with a significantly larger number of non-fraudulent transactions compared to fraudulent ones.

Distribution Insight:

- Approximately **99.83%** of the transactions are non-fraudulent, while only about **0.17%** are fraudulent.
- This imbalance highlights the challenge of detecting fraud due to the scarcity of fraudulent examples in the dataset.

Visualization Insight:

- Notice how imbalanced is our original dataset. Most of the transactions are non-fraud. If we use this dataframe as the base for our predictive models and analysis we might get a lot of errors and our algorithms will likely overfit since it will "assume" that most transactions are not fraud.



Class Distribution (Pre-RUS)

Creating a sub-sample

What is Sub-Sampling?

- **Definition:**
  - **Sub-Sampling** involves creating a smaller, balanced subset of the original dataset.
  - In this scenario, the sub-sample will contain an equal number of fraudulent and non-fraudulent transactions (50/50 ratio).
- **Purpose:**
  - **Balance:** Ensures an equal representation of both classes to improve model training.
  - **Enhanced Learning:** Helps the model learn patterns that distinguish between fraudulent and non-fraudulent transactions more effectively.

Why Create a Sub-Sample?

- **Addressing Imbalance:**
  - The original dataset is heavily skewed with far more non-fraudulent cases than fraudulent ones.
- **Avoiding Common Issues:**
  - **Overfitting:** Models might learn to predict non-fraud transactions more often, reducing their effectiveness in identifying fraud.
  - **Wrong Correlations:** Imbalance makes it difficult to understand the true relationships between features and the target class.

Creating the Sub-Sample:

- Fraud Cases: 492 transactions.
- Non-Fraud Cases: Randomly select 492 transactions.
- Concatenate both sets to form a balanced sub-sample.

Dataset Splitting Details:

To ensure effective training and validation of the model, the dataset is split into multiple training and testing sets:

- **Split 1:**
  - **Training Indices:** [30473, 30496, 31002, ..., 284804, 284805, 284806]
  - **Testing Indices:** [0, 1, 2, ..., 57017, 57018, 57019]
- **Split 2:**
  - **Training Indices:** [0, 1, 2, ..., 284804, 284805, 284806]
  - **Testing Indices:** [30473, 30496, 31002, ..., 113964, 113965, 113966]
- **Split 3:**
  - **Training Indices:** [0, 1, 2, ..., 284804, 284805, 284806]
  - **Testing Indices:** [81609, 82400, 83053, ..., 170946, 170947, 170948]
- **Split 4:**
  - **Training Indices:** [0, 1, 2, ..., 284804, 284805, 284806]
  - **Testing Indices:** [150654, 150660, 150661, ..., 227866, 227867, 227868]
- **Split 5:**
  - **Training Indices:** [0, 1, 2, ..., 227866, 227867, 227868]
  - **Testing Indices:** [212516, 212644, 213092, ..., 284804, 284805, 284806]

Training Set

- The training set is used to train the machine learning model. During this phase, the model learns patterns and relationships in the data by adjusting its parameters based on the input features and the known target labels.

Testing Set

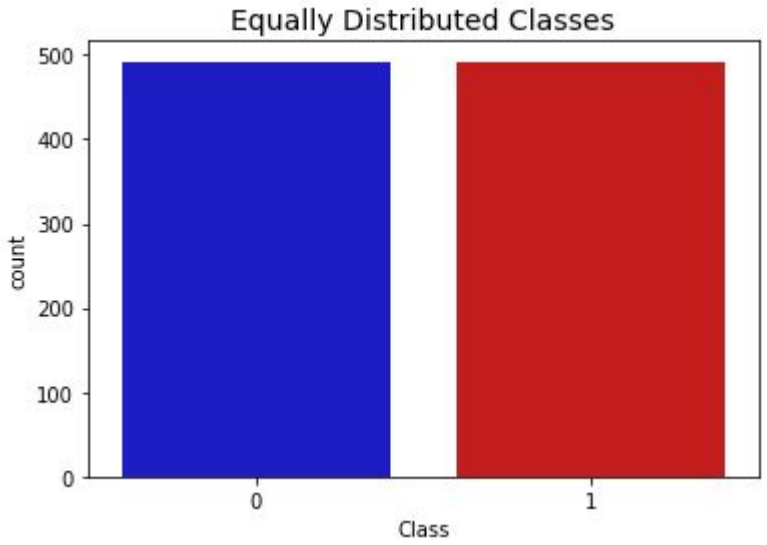
- **Purpose:** The testing set is used to evaluate the performance of the trained model. It assesses how well the model generalizes to new, unseen data. This set is not used during the training phase.

Label Distributions in Train/Test Splits:

- **Non-Balanced Distribution Consistency:**
  - Non-Fraud (0): ~99.83%
  - Fraud (1): ~0.17%

Balanced Distribution:

  - **Frauds:** 50% of the subsample
  - **Non-Frauds:** 50% of the subsample
- **Purpose:** In order to address class imbalance, ensuring that the model is equally exposed to both classes during training and evaluation.



Utilizing Random UnderSampling (Including Splitting, Train/Test)

Correlation Matrices

Role of Correlation Matrices

- **Essence of Data Analysis**:
  - Correlation matrices are fundamental tools for understanding the relationships between features in a dataset.
  - ***They help identify which features have a strong influence on whether a transaction is classified as fraud.***

Using a balanced subsample ensures that the analysis reflects a more accurate picture of feature importance, avoiding the distortions caused by class imbalance.

Correlation Insights

By examining the correlation matrix, we can determine how each feature correlates with the target variable, "Class".

- **Positive Correlations**: Features with higher values are associated with a higher likelihood of fraud. (Blue Coloring)
- **Negative Correlations**: Features with lower values are associated with a higher likelihood of fraud. (Red Coloring)

Visual Interpretation:

- **Positive Correlation**: Features with more intense coloring show a stronger association with an increased likelihood of fraud.
- **Negative Correlation**: Features with intense coloring in the opposite direction indicate a stronger association with a higher likelihood of fraud.

Negative Correlations

- **Observation:** Lower values of these features are associated with a higher probability of fraud.
- **Implication:** When these feature values are low, the likelihood of a transaction being fraudulent increases.

Positive Correlations

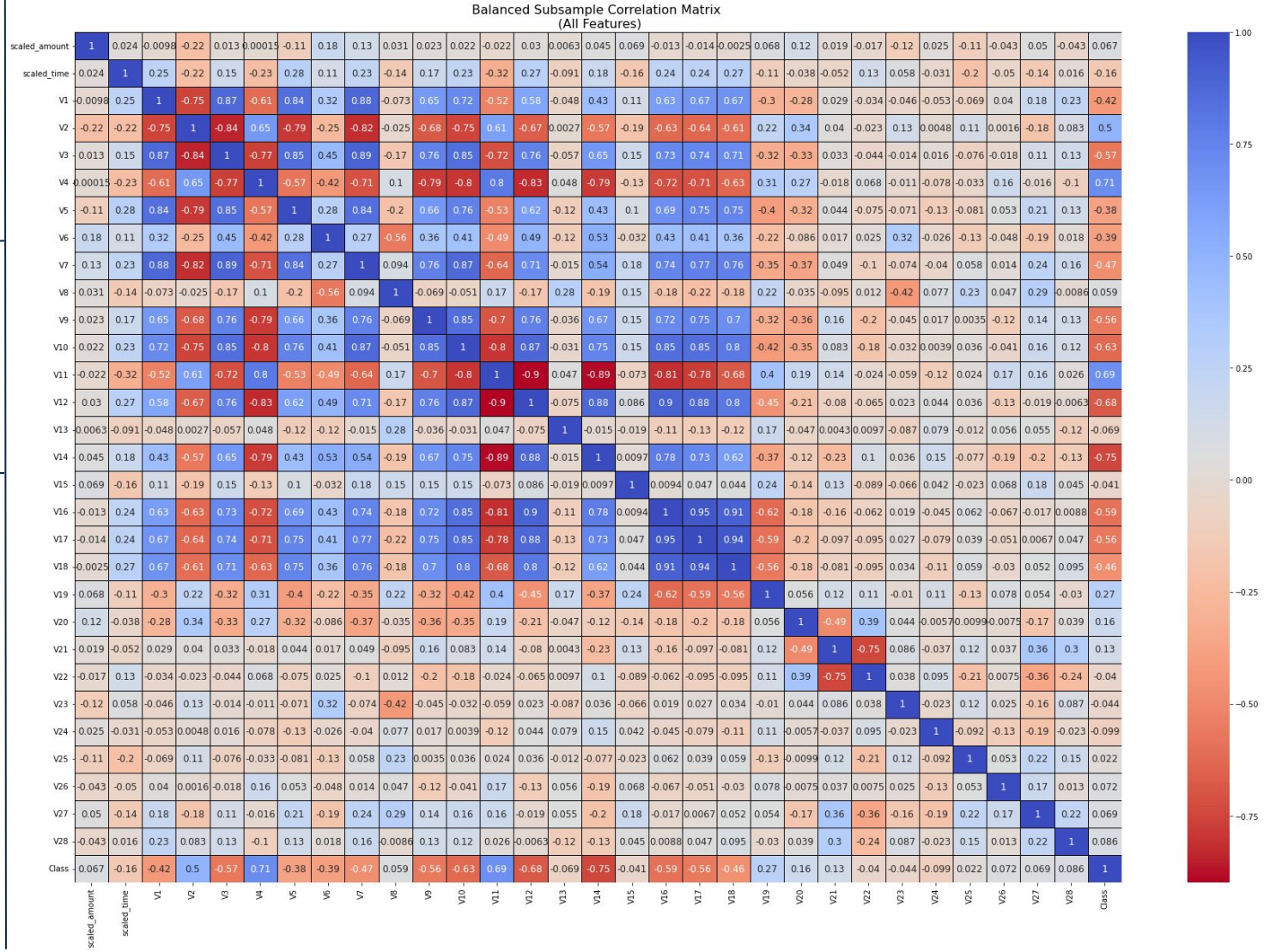
- **Observation:** Higher values of these features correlate with a higher probability of fraud.
- **Implication:** As these feature values increase, the likelihood of a transaction being fraudulent also rises.

Top 5 Positive Correlations:

1. **V4**: 0.71
2. **V11**: 0.69
3. **V2**: 0.50
4. **V19**: 0.27
5. **V28**: 0.086

Top 5 Negative Correlations:

1. **V14**: -0.75
2. **V12**: -0.68
3. **V10**: -0.63
4. **V16**: -0.59
5. **V3**: -0.57



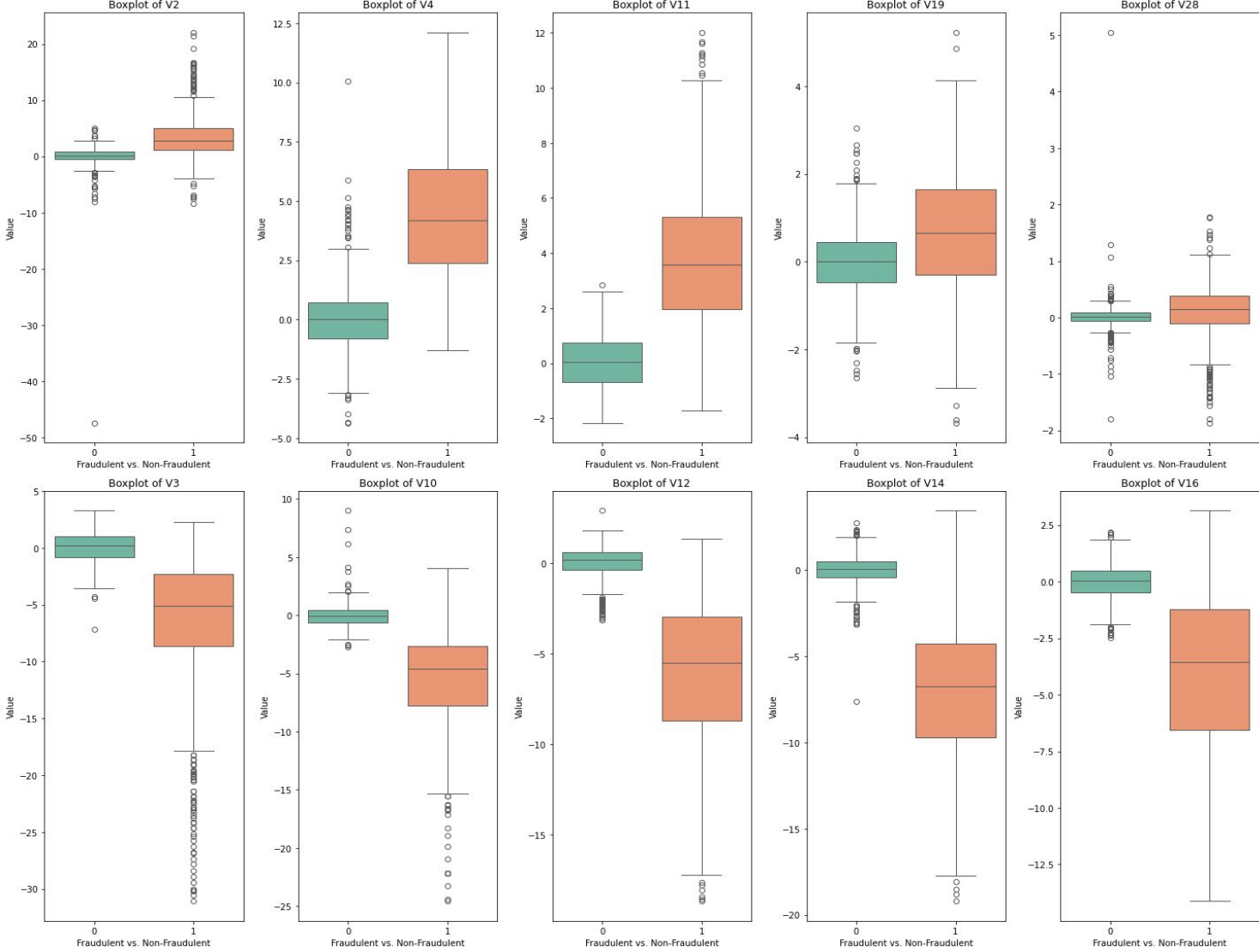
Sub-Sample Correlation Matrix

1. **Positive Correlations:**
   - **V2, V4, V11, V19, V28:**
     - Fraudulent transactions (orange) generally have higher values compared to non-fraudulent transactions (green).
     - V4 and V11, in particular, show clear separation, highlighting their strong positive correlation with fraud.

2. **Negative Correlations:**
   - **V3, V10, V12, V14, V16:**
     - Fraudulent transactions (orange) tend to have lower values compared to non-fraudulent ones (green).
     - V14 and V12 exhibit substantial separation, emphasizing their strong negative correlation with fraudulent activity.

Key Takeaways

- **Distribution Differences:** Noticeable differences in the distributions between fraudulent and non-fraudulent classes indicate that these features are valuable predictors for fraud detection.
- **Potential Indicators:** Features like ***V4, V11, V14, and V12*** are particularly distinguishable and might serve as strong indicators in the model.



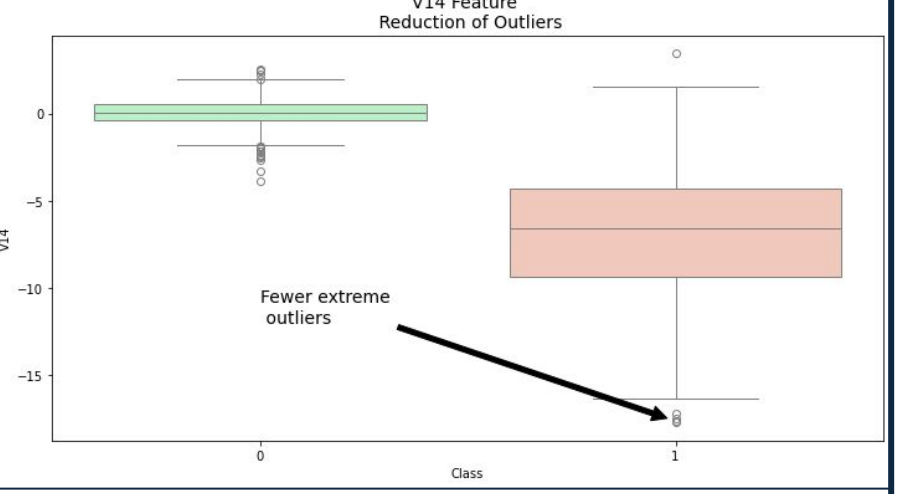
Visualizing Box Plots

Anomaly Detection with IQR

The main aim is to remove "extreme outliers" from features that have a high correlation with our target variable (fraud or non-fraud), which skew results, reduce overall accuracy.

Quartile 25 (Q1): This is the 25th percentile of the data, meaning 25% of the data falls below this value.
Quartile 75 (Q3): This is the 75th percentile of the data, meaning 75% of the data falls below this value.
Interquartile Range (IQR): This is the difference between Q3 and Q1, which measures the spread of the middle 50% of the data.
Thresholds for Outliers:

- **Lower Threshold:** Calculated as `Q1 - (1.5 * IQR)`. It extends 1.5 times the IQR below the 25th percentile.
- **Upper Threshold:** Calculated as `Q3 + (1.5 * IQR)`. It extends 1.5 times the IQR above the 75th percentile.

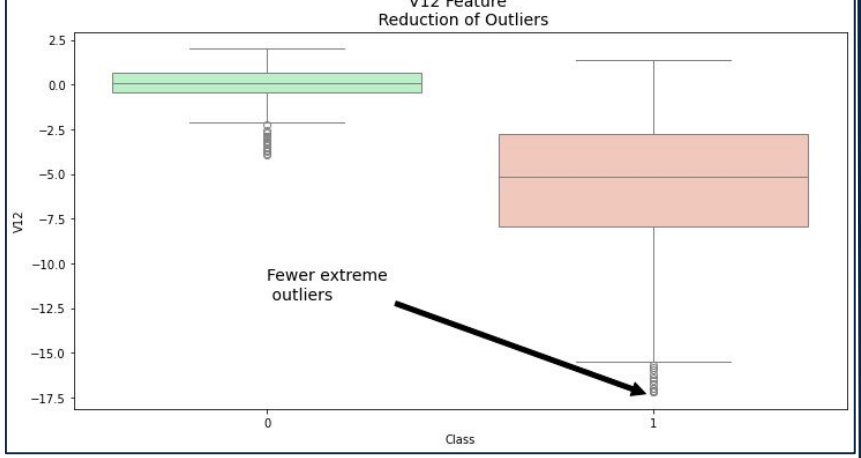


Quartile 25 (Q1): -9.692722964972386
Quartile 75 (Q3): -4.282820849486865
Interquartile Range (IQR): 5.409902115485521

Thresholds for Outliers:

- **Lower Threshold:** -17.807576138200666 (Q1 - 1.5 * IQR)
- **Upper Threshold:** 3.8320323237414167 (Q3 + 1.5 * IQR)

Feature V14 Outliers for Fraud Cases: 4
V14 Outliers: [-18.4937733551053, -18.0499976898594, -19.2143254902614, -18.8220867423816]

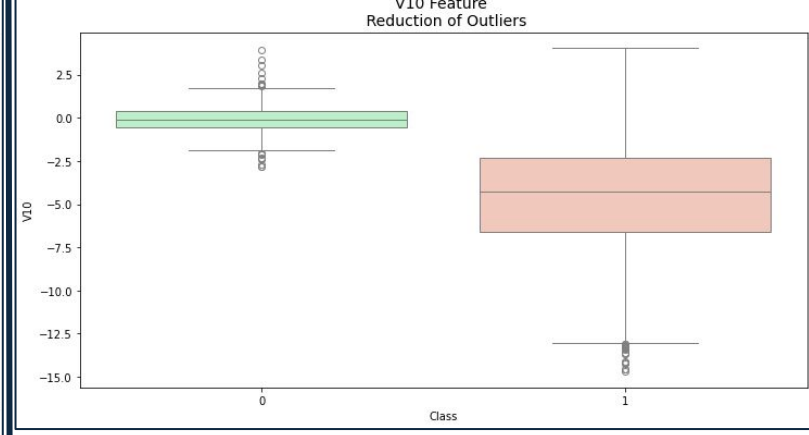


Quartile 25 (Q1): -8.824009765608226
Quartile 75 (Q3): -3.641665007459668
Interquartile Range (IQR): 5.182344758148558

Thresholds for Outliers:

- **Lower Threshold:** -17.3430371579634 (Q1 - 1.5 * IQR)
- **Upper Threshold:** 5.776973384895937 (Q3 + 1.5 * IQR)

Feature V12 Outliers for Fraud Cases: 4
V12 Outliers: [-18.0475965708216, -18.6837146333443, -18.4311310279993, -18.5536970096458]



Quartile 25 (Q1): -2.224870717084841
Quartile 75 (Q3): 5.863989908107994
Interquartile Range (IQR): 8.088860625192835

Thresholds for Outliers:

- **Lower Threshold:** -14.89885463232024 (Q1 - 1.5 * IQR)
- **Upper Threshold:** 4.92033495834214 (Q3 + 1.5 * IQR)

Feature V10 Outliers for Fraud Cases: 27
V10 Outliers: [-22.1870885620007, -15.1237521803455, -15.5637913387301, -14.9246547735487, -15.3460988468775, -24.4031849699728, -16.7460441053944, -17.1415136412892, -16.2556117491401, -19.836148851696, -16.3035376590131, -15.2318333653018, -18.9132433348732, -22.1870885620007, -24.5882624372475, -15.1241628144947, -15.5637913387301, -15.2399619587112, -15.2399619587112, -23.2282548357516, -14.9246547735487, -22.1870885620007, -16.6496281595399, -20.9491915543611, -22.1870885620007, -18.2711681738888, -16.6011969664137]

Feature V14 has a lower threshold of -17.81 and an upper threshold of 3.83, with four extreme outliers below the lower threshold, ranging around -18 to -19.

For V12, the thresholds were -17.34 and 5.78, also revealing four outliers below the lower boundary.

V10 showed a much higher concentration of outliers, with 27 values falling significantly below the lower threshold of -14.90, highlighting ***more frequent extreme deviations compared to the other features, meaning its strongly associated with fraudulent transactions.***

Removing Anomalies

Evaluating Best Classifier (Initialization)

Objective: Train and evaluate four types of classifiers to determine which is most effective in detecting fraud transactions.

1. **Data Splitting:**
   - **Separate Features and Labels:** Divide the dataset into features (X) and labels (y).
   - **Train-Test Split:** Split the data into training and testing sets to evaluate model performance.

2. **Classifier Evaluation:**
   - **Classifiers to be tested:**
     - Logistic Regression
     - K-Nearest Neighbors
     - Support Vector Classifier
     - Decision Tree Classifier

```
# Let's implement simple classifiers

classifiers = {
    "LogisiticRegression": LogisticRegression(),
    "KNearest": KNeighborsClassifier(),
    "Support Vector Classifier": SVC(),
    "DecisionTreeClassifier": DecisionTreeClassifier()
}
```

Best Performing Classifier: Logistic Regression with a 93.0% accuracy score.

Next Best: KNN and SVC, both with a 92.0% accuracy score.

Lowest Score: Decision Tree Classifier with an 89.0% accuracy score.

Logistic Regression shows the highest accuracy and should be preferred for further analysis and model selection.

```
Classifiers:  LogisticRegression Has a training score of 95.0 % accuracy score
Classifiers:  KNeighborsClassifier Has a training score of 93.0 % accuracy score
Classifiers:  SVC Has a training score of 93.0 % accuracy score
Classifiers:  DecisionTreeClassifier Has a training score of 91.0 % accuracy score
```

- GridSearchCV automates the process of hyperparameter tuning by exhaustively searching through a grid of specified parameters.
- It uses cross-validation to evaluate each combination, ensuring the best model parameters are selected.
- Cross-validation is a technique used to evaluate the performance of the classifiers by testing how well the models ***generalize*** ***(how well a model can digest new data and make correct predictions)*** to the dataset.

```
Logistic Regression Cross Validation Score:  93.92%
Kneears Neighbors Cross Validation Score 93.13%
Support Vector Classifier Cross Validation Score 93.26%
DecisionTree Classifier Cross Validation Score 90.88%
```

Logistic Regression performs slightly better than the other classifiers, with SVC and K-Nearest Neighbors closely following.

Classifier Accuracy Results & GridSearchCV

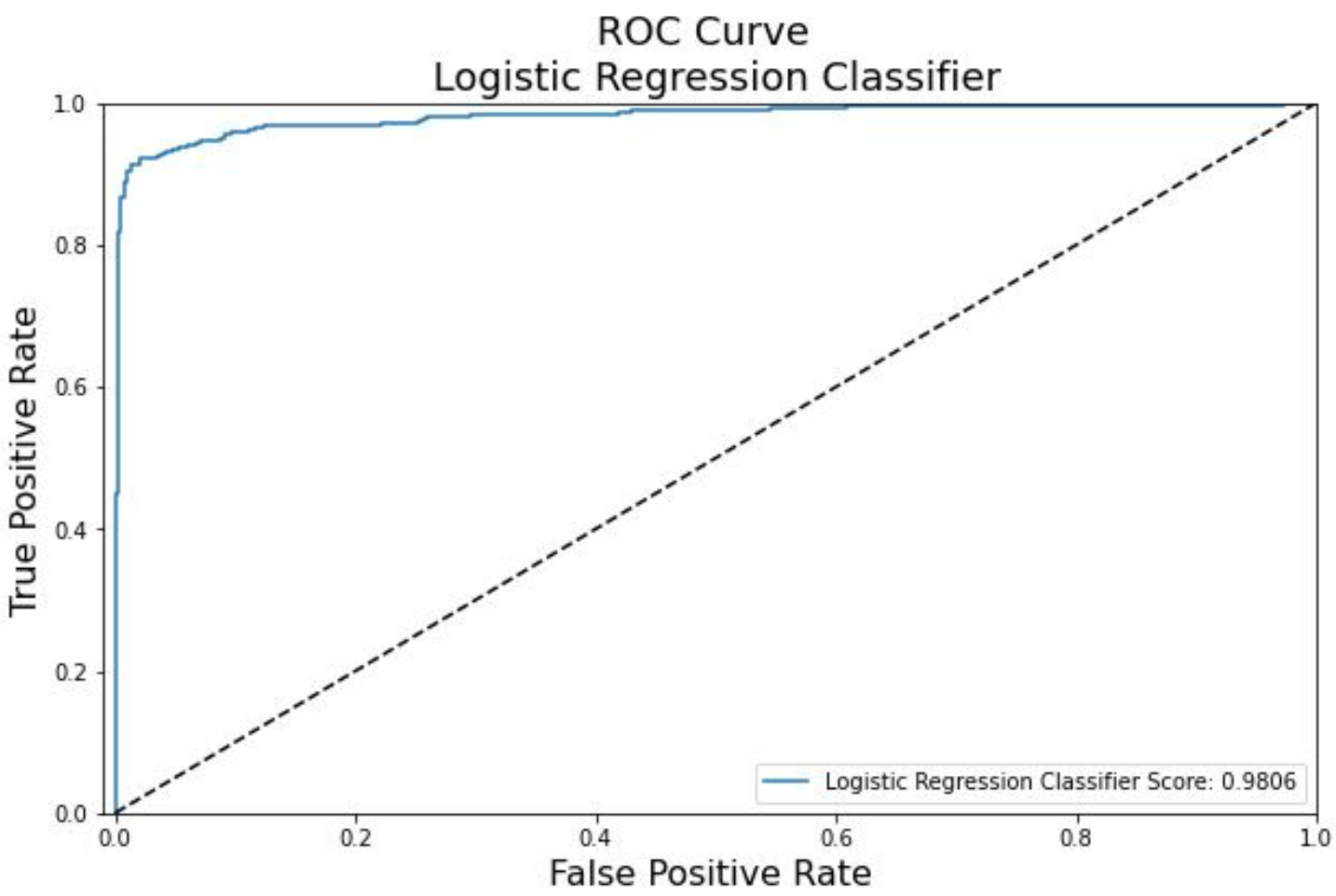
Logistic Regression Classifier Score: 0.9806

Key Insights:

- **ROC AUC Score**: ***0.9806 indicates exceptional performance.***
- **What It Means**:
  - **High Discrimination**: The model can effectively distinguish between fraudulent and non-fraudulent instances.
  - **True Positive Rate vs. False Positive Rate**: The score reflects that the Logistic Regression classifier has a high rate of correctly identifying fraud cases while minimizing false positives.

Implications:

- **Model Effectiveness**: The high AUC score suggests that Logistic Regression is very effective in this context and can correctly discern between fraud and non-fraud cases.
- **Generalization to New Data**: While the model performs well on the current dataset, testing it on additional or unseen data is crucial to confirm its ability to generalize and maintain high performance in different scenarios.



ROC Curve Score - Logistic Regression

Performance Metrics of Logistic Regression

1. **Sensitivity (True Positive Rate):** ***0.9451***
2. **Precision:** ***0.9556***
3. **Specificity:** ***0.9592***

```
Sensitivity (True Positive Rate): 0.9451
Precision: 0.9556
Specificity: 0.9592
```

Key Insights:

- **Sensitivity (True Positive Rate):** 92.86% - The model successfully identifies 94.51% of actual fraud cases.
- **Precision:** 98.91% - Of all instances predicted as fraud, 95.56% are actual fraud cases, indicating a low false positive rate.
- **Specificity:** 98.91% - The model accurately identifies 95.92% of non-fraud cases, showing strong performance in distinguishing between fraud and non-fraud cases.

Conclusion:

- The Logistic Regression model demonstrates high sensitivity, precision, specificity, and balanced accuracy, indicating robust performance in detecting fraud while accurately classifying both fraud and non-fraud cases.
- **Next Steps:**
  - **Validate with New Data:** Ensure the model generalizes well to unseen datasets.
  - **Monitor Over Time:** Continuously evaluate model performance to adapt to new patterns in fraud detection.

Questions?

Questions?