Credit Card Fraud Detection Using Supervised Learning Techniques

1. Introduction

In this report, we implement and evaluate different supervised learning techniques to detect fraudulent credit card transactions. When a fraudulent transaction occurs, it is the responsibility of the financial institution to cover those losses. Thus, financial institutions must minimize losses and protect customers. This project aims to use various machine learning models to classify transactions as “fraud” or “non-fraud” accurately. This project will look at various steps, including Data Acquisition, Exploratory Data Analysis, data preprocessing, and model implementation. The final analysis highlights the most effective model and sampling strategies based on metrics such as precision, recall, and F1-score for detecting fraud cases.

2. Data Acquisition

The dataset used is sourced from Kaggle’s Credit Card Fraud Detection dataset. The dataset contains transactions from September 2013 by European cardholders. This dataset contains 284,807 credit card transactions, with each transaction labeled as either “fraud” (positive class) or “non-fraud” (negative class). The dataset is highly imbalanced, with only 0.172% of transactions classified as fraud, which presents unique challenges for model training and evaluation. The dataset features a single csv file with all the transactions.

3. Exploratory Data Analysis (EDA)

The Exploratory Data Analysis phase provides insights into the dataset's structure, distribution, and potential patterns that might assist in fraud detection.

Each transaction has 30 features:

- **V1 to V28**: Principal Component Analysis (PCA)-derived features to protect sensitive information. These are features related to the transaction that are anonymized due to the sensitivity of the financial data.
- **Time**: Seconds elapsed since the first transaction in the dataset.
- **Amount**: Transaction amount.
- **Class**: Binary label indicating fraud (1) or non-fraud (0).
- There are no null values in the dataset

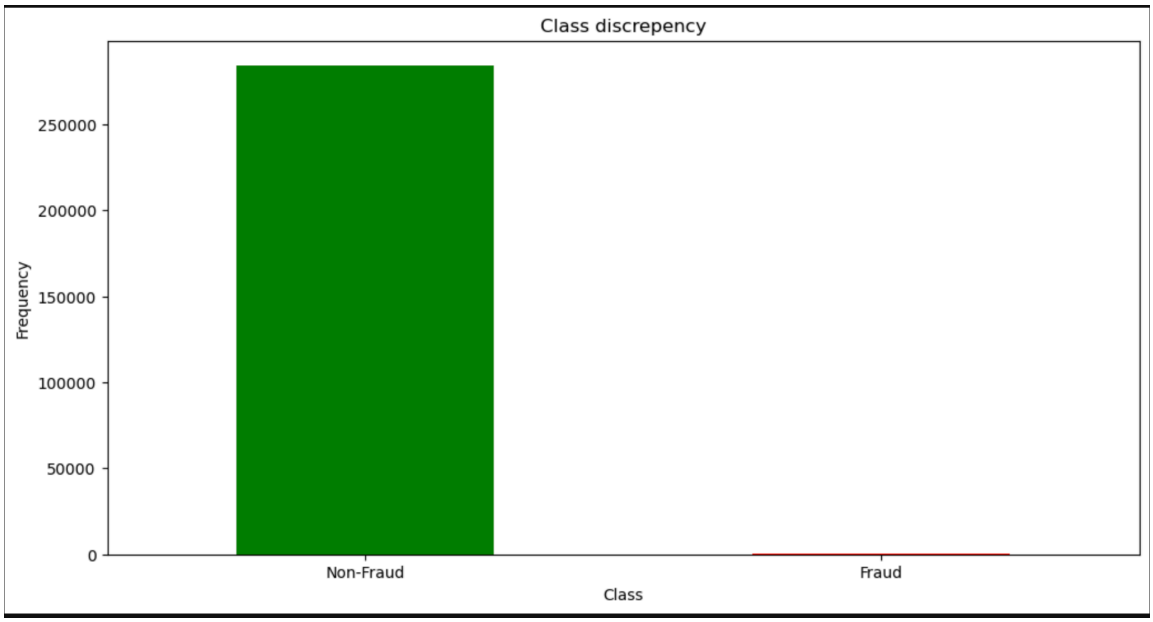
3.1 Data Distribution

• The dataset is highly imbalanced, with only a small fraction of transactions classified as fraud. This imbalance poses a challenge for model training, as most algorithms might naturally favor the majority class (non-fraud transactions).

•**Class Distribution**: The plot of class distribution reveals that approximately 99.83% of transactions are non-fraud, while only 0.17% are fraud.

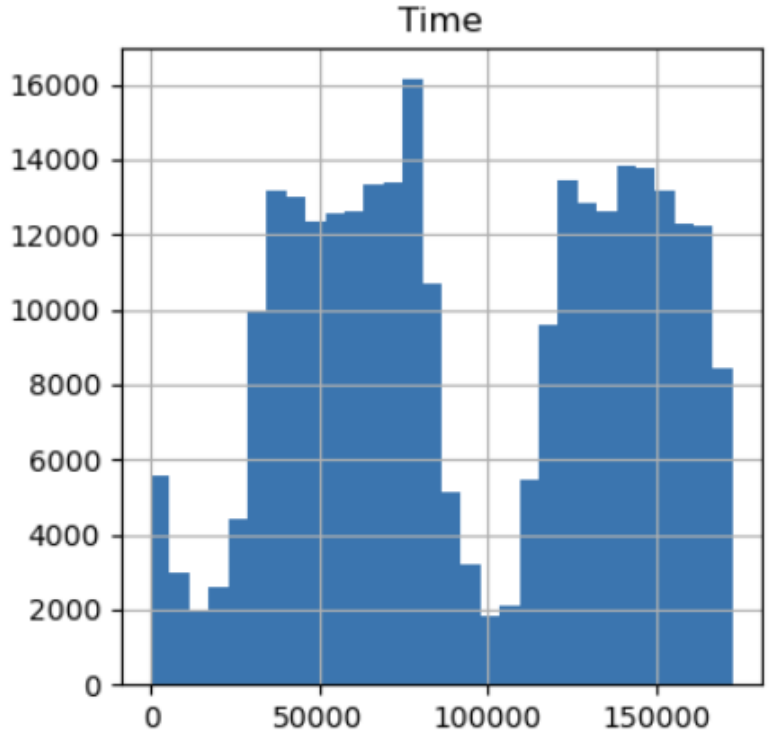
```
[28]: df['Class'].value_counts()

[28]: Class
      0    284315
      1       492
```

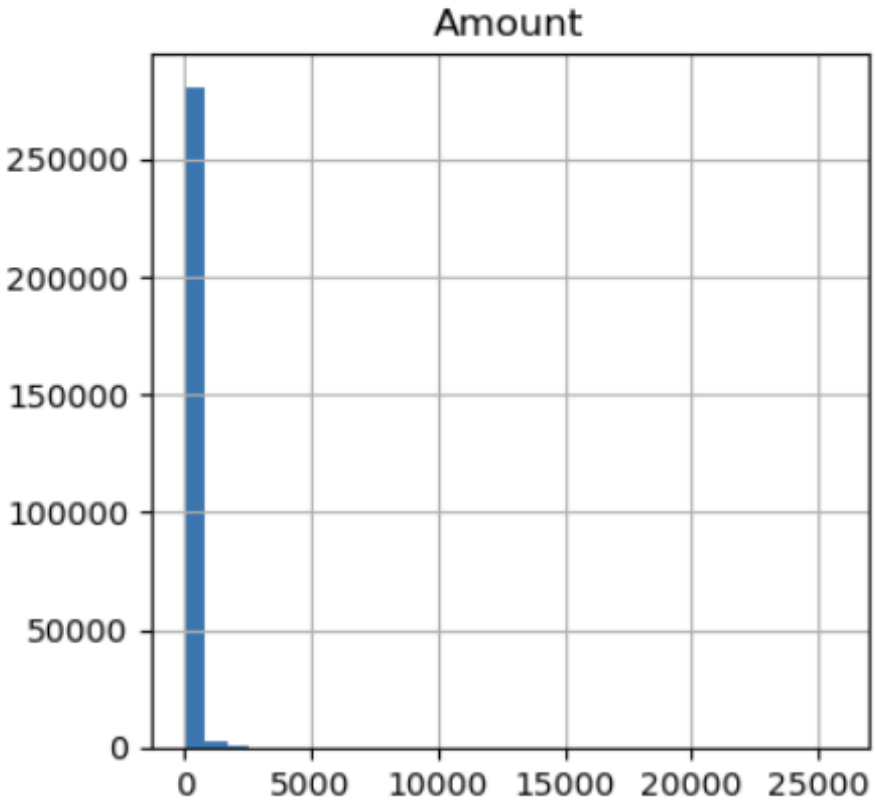


3.2 Feature Analysis

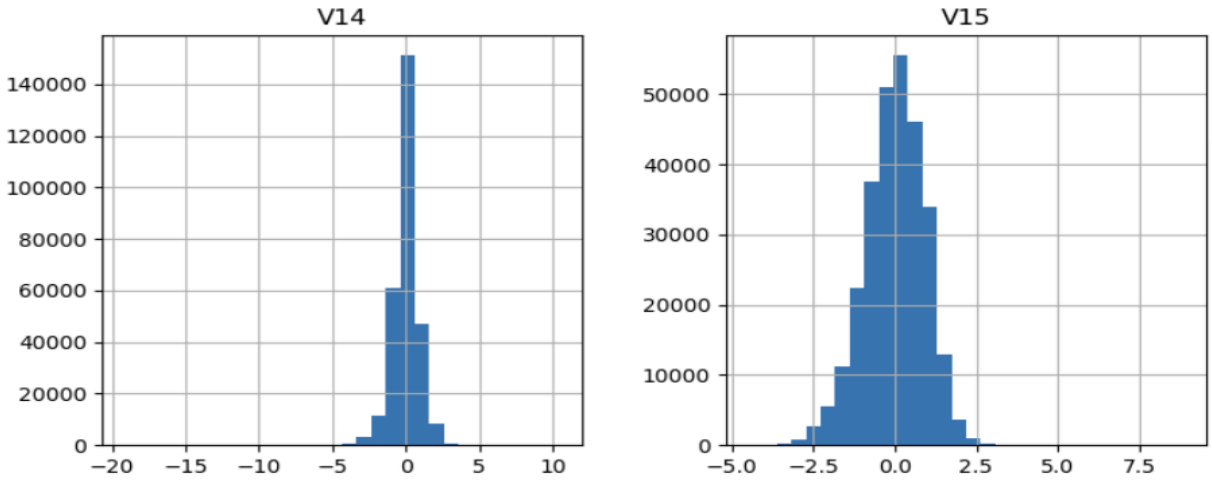
- **Time and Amount Distributions:**
- The Time feature, representing the number of seconds elapsed since the first transaction, does not show any clear trend related to fraud.



- The Amount feature varies widely, with most transactions having lower amounts. Fraudulent transactions don't show a strong association with high or low amounts, as fraud can occur at any transaction value.

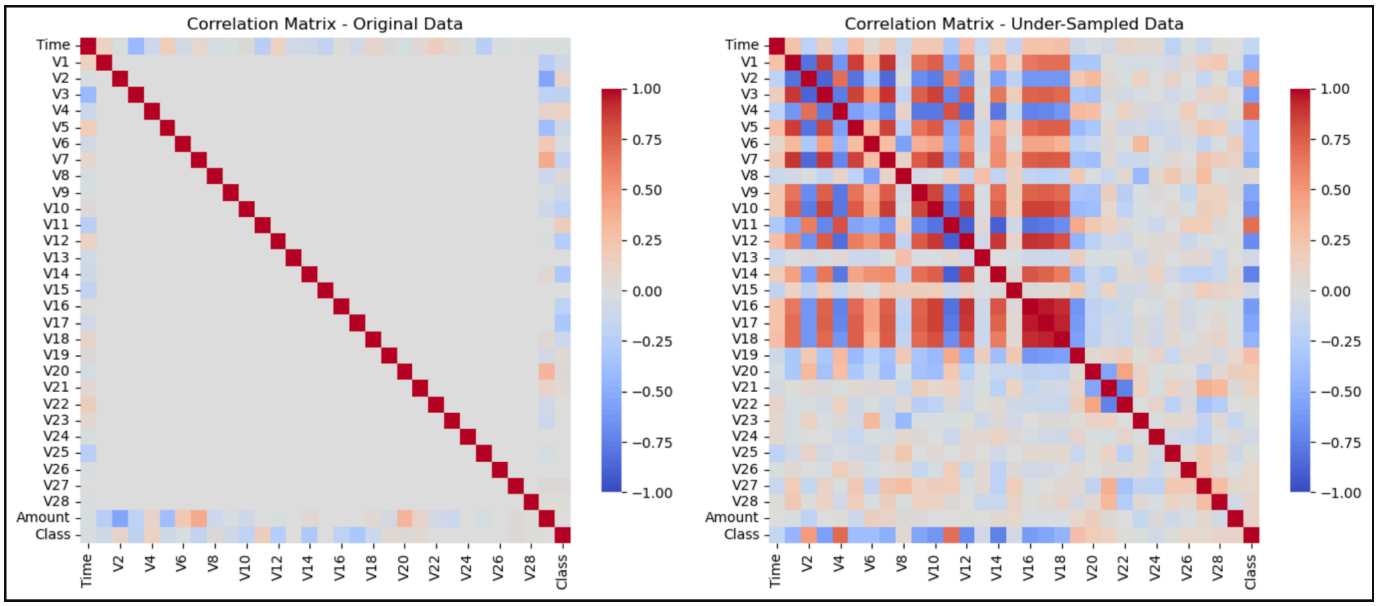


- **PCA Features (V1 - V28)**: The dataset's principal components (V1 to V28) were generated using PCA for anonymization. Fraud and non-fraud transactions do not exhibit immediately discernible differences within these features in isolation, though some subtle patterns may exist in certain combinations. This indicates that multivariate analysis could be beneficial for pattern recognition.



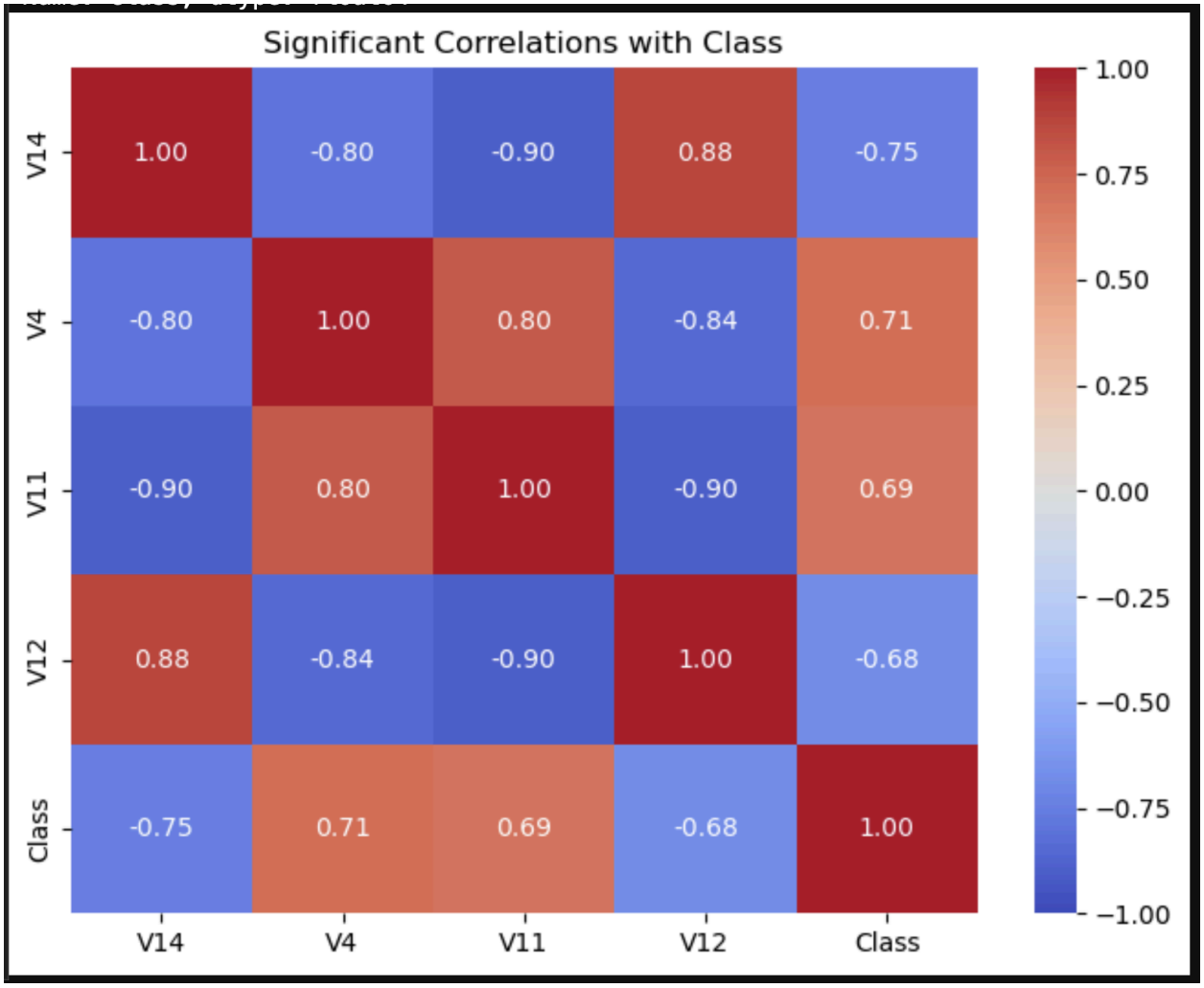
3.3 Correlation Analysis

- **Correlation Heatmap**: The generated heatmap shows how different components affect the classes. Since the features are taken after the PCA, each of the features tells something important about the transaction.



Significant Correlations -

V14	- 0.749228
V4	+ 0.712610
V11	+ 0.685056
V12	- 0.682039

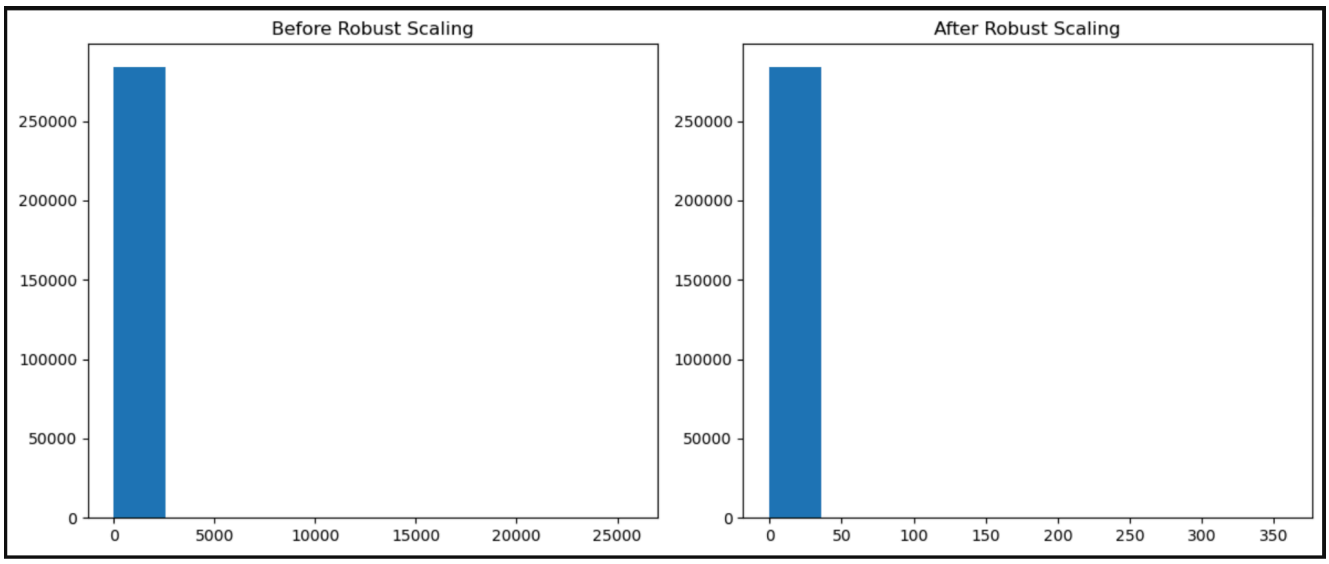

Significant Correlations with Class
V14
V4
V11
V12
Class
1.00
-0.80
-0.90
0.88
-0.75
-0.80
1.00
0.80
-0.84
0.71
-0.90
0.80
1.00
-0.90
0.69
0.88
-0.84
-0.90
1.00
-0.68
-0.75
0.71
0.69
-0.68
1.00
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

4. Data Preprocessing

The data preprocessing stage prepares the dataset for modeling, focusing on handling the class imbalance and scaling features to enhance model performance.

4.1 Scaling Amount Column

- The dataset contains significant outliers, and most of the transactions are very small in amount. This can negatively affect our prediction models. To prevent this, we can scale our data using a robust scaler to reduce the impact of the outliers. This will bring down the standard deviation significantly.

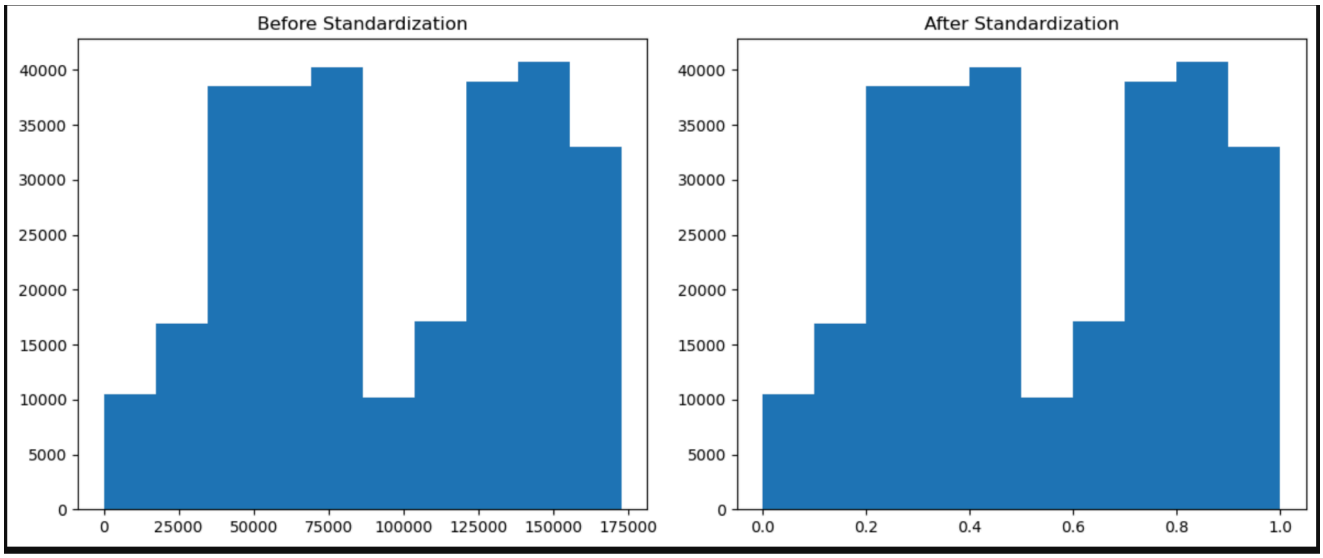


Before scaling	
count	284807.000000
mean	88.349619
std	250.120109
min	0.000000
25%	5.600000
50%	22.000000
75%	77.165000
max	25691.160000

After scaling	
count	284807.000000
mean	0.927124
std	3.495006
min	-0.307413
25%	-0.229162
50%	0.000000
75%	0.770838
max	358.683155

4.2 Standardizing time data

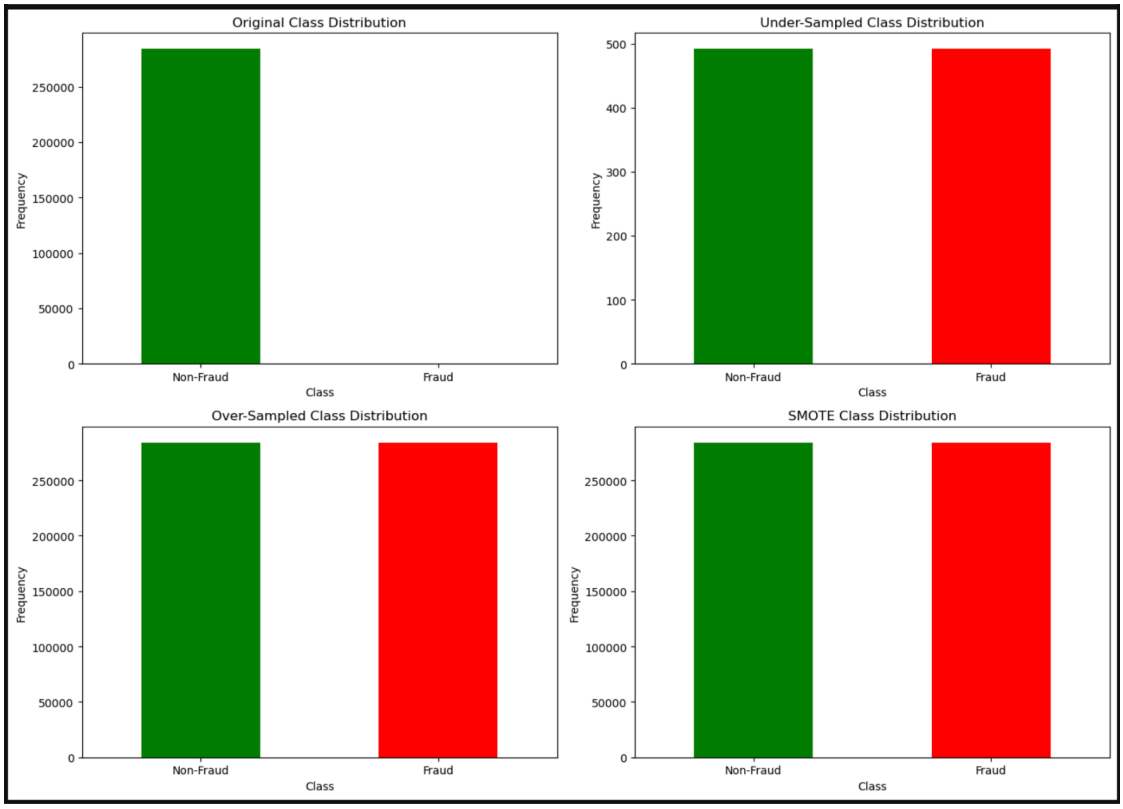
- The “Time” column has values with a very large range. We can standardize this and make all the values between 0 and 1 according to the time from the first transaction.

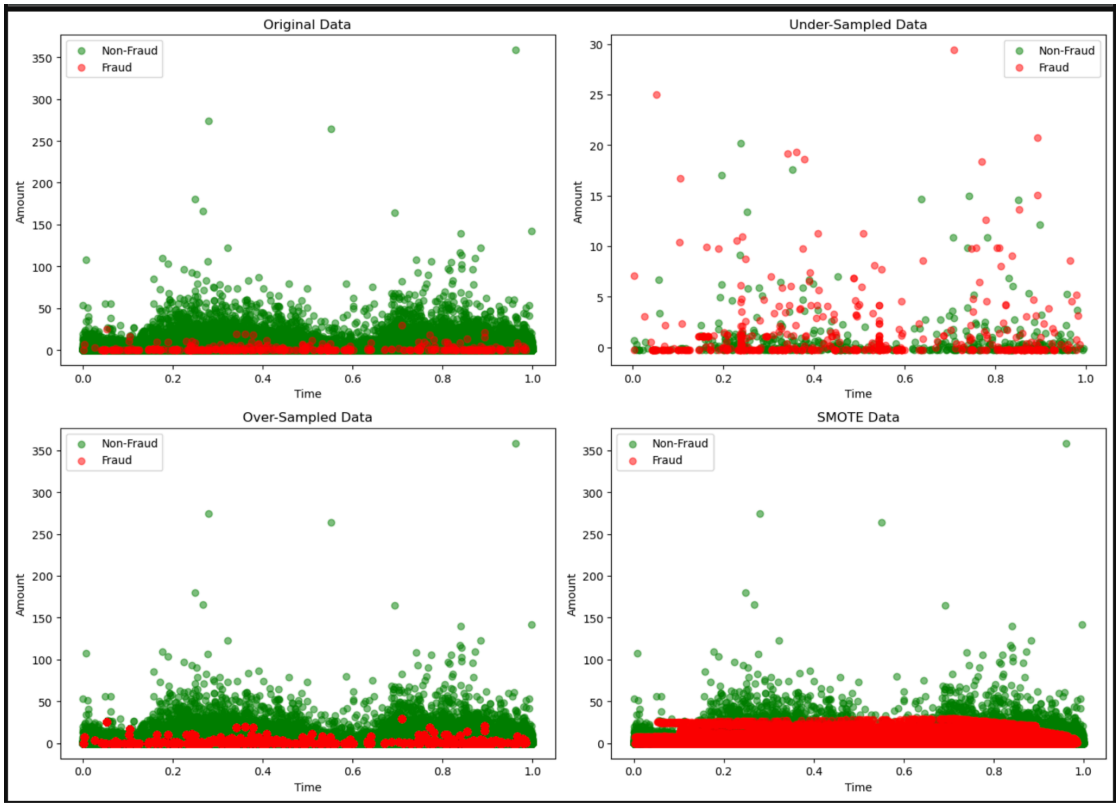


4.3 Handling Class Imbalance

Given the substantial class imbalance, multiple resampling techniques were employed to create balanced datasets, which can improve model sensitivity to the minority (fraud) class. The following techniques were applied:

- **Under-sampling**: Reduced the majority class (non-fraud) by random sampling to match the fraud cases. While under-sampling can balance classes, it risks discarding valuable information from the majority class, potentially limiting the model’s accuracy.
- **Over-sampling**: Increased the minority class (fraud) by replicating instances, allowing the dataset to retain the full majority class. This can prevent the model from ignoring the minority class, though it risks overfitting to duplicated data.
- **Synthetic Minority Over-sampling Technique (SMOTE)**: Created synthetic samples of the minority class by interpolating between real fraud cases. SMOTE allows the model to train on a more diverse set of fraud instances, which can improve model generalizability. However, these aren’t real points and are synthetically created, which can lead to poor model performance.


Original Class Distribution
Under-Sampled Class Distribution
Over-Sampled Class Distribution
SMOTE Class Distribution
Frequency
Class
Non-Fraud
Fraud


Original Data
Under-Sampled Data
Over-Sampled Data
SMOTE Data
Non-Fraud
Fraud
Amount
Time

5. Model Training and Evaluation

In this section, multiple machine learning models were trained and evaluated across different resampling strategies to identify the best-performing model for credit card fraud detection. The following models were evaluated:

1. K-Nearest Neighbors (KNN)
2. Decision Tree
3. Support Vector Machine (SVM)
4. Logistic Regression

5.1 Evaluation Metrics

Accuracy is a bad measure for the unsampled data. This is because the majority of the data is the negative class. So, even if we train a model that only predicts the negative class, it will have a very high accuracy.

To assess model performance, the following metrics were used:

- **Precision (Fraud)**: The precision metric (Fraud) indicates the proportion of predicted fraud cases that were truly fraudulent. A high precision score (closer to 1) signifies that most predicted fraud cases were accurate, while a low score points to a significant number of false positives.
- **Recall (Fraud)**: Recall measures how effectively the model identifies actual fraudulent transactions. A high recall suggests that most fraud cases are detected, while a low recall implies many false negatives. High recall is critical for credit card companies as it minimizes potential losses from undetected fraud, which could otherwise have significant financial implications.
- **F1-Score (Fraud)**: The F1-Score, representing the harmonic mean of precision and recall, provides a balanced measure of model performance.

5.2 Findings

1. **Logistic regression**

Final Logistic Regression Results:

	Model	Sampling	Accuracy	Precision (Non-Fraud)	Recall (Non-Fraud)	F1-Score (Non-Fraud)	Precision (Fraud)	Recall (Fraud)	F1-Score (Fraud)
0	Logistic Regression	Original	0.999	0.999	1.000	1.000	0.884	0.633	0.738
1	Logistic Regression	Under-Sampled	0.939	0.913	0.966	0.939	0.967	0.913	0.939
2	Logistic Regression	Over-Sampled	0.949	0.924	0.977	0.950	0.976	0.920	0.947
3	Logistic Regression	SMOTE	0.949	0.926	0.976	0.950	0.974	0.922	0.947

2. **KNN**

Model	Sampling	Neighbors	Accuracy	Precision (Non-Fraud)	Recall (Non-Fraud)	F1-Score (Non-Fraud)	Precision (Fraud)	Recall (Fraud)	F1-Score (Fraud)
KNN	Original	3	1.000	1.000	1.000	1.000	0.923	0.80	0.857
KNN	Under-Sampled	3	0.943	0.895	1.000	0.944	1.000	0.89	0.942
KNN	Over-Sampled	3	1.000	1.000	0.999	1.000	0.999	1.00	1.000
KNN	SMOTE	3	0.999	1.000	0.999	0.999	0.999	1.00	0.999

3. **Decision Trees**

Model	Sampling	Max Depth	Accuracy	Precision (Non-Fraud)	Recall (Non-Fraud)	F1-Score (Non-Fraud)	Precision (Fraud)	Recall (Fraud)	F1-Score (Fraud)
Decision Tree	Original	5	0.999	1.000	1.000	1.000	0.881	0.800	0.838
Decision Tree	Under-Sampled	3	0.923	0.873	0.983	0.925	0.982	0.866	0.921
Decision Tree	Over-Sampled	7	0.979	0.981	0.976	0.978	0.976	0.981	0.979
Decision Tree	SMOTE	7	0.972	0.968	0.977	0.972	0.976	0.968	0.972

4. **Support Vector Machines**

Model	Sampling	Kernel	Accuracy	Precision (Non-Fraud)	Recall (Non-Fraud)	F1-Score (Non-Fraud)	Precision (Fraud)	Recall (Fraud)	F1-Score (Fraud)
SVM	Original	linear	0.999	1.000	1.000	1.000	0.829	0.808	0.819
SVM	Under-Sampled	linear	0.943	0.913	0.975	0.943	0.975	0.913	0.943

5.3 Summary of Findings:

1. **Precision (Fraud)**

	Model	Sampling	Precision (Fraud)	Recall (Fraud)
1	KNN	Under-Sampled	1.000	0.890
5	Decision Tree	Under-Sampled	0.982	0.866
6	Decision Tree	Over-Sampled	0.976	0.981
7	Decision Tree	SMOTE	0.976	0.968
12	Logistic Regression	Over-Sampled	0.976	0.920
9	SVM	Under-Sampled	0.975	0.913
13	Logistic Regression	SMOTE	0.974	0.922
11	Logistic Regression	Under-Sampled	0.967	0.913
0	KNN	Original	0.923	0.800
10	Logistic Regression	Original	0.884	0.633
4	Decision Tree	Original	0.881	0.800
8	SVM	Original	0.829	0.808

- The KNN model with undersampling achieved the highest precision of 1, indicating no false positives. However, it exhibited low recall.
- The Decision Tree models consistently performed well, following KNN, while the oversampled Logistic Regression and undersampled SVMs also performed admirably.
- The unsampled SVM model had the lowest precision, indicating a high number of false positives, which can adversely affect credit card companies by annoying customers and potentially leading them to switch providers.

2. **Recall (Fraud)**

	Model	Sampling	Recall (Fraud)	Precision (Fraud)
6	Decision Tree	Over-Sampled	0.981	0.976
7	Decision Tree	SMOTE	0.968	0.976
13	Logistic Regression	SMOTE	0.922	0.974
12	Logistic Regression	Over-Sampled	0.920	0.976
9	SVM	Under-Sampled	0.913	0.975
11	Logistic Regression	Under-Sampled	0.913	0.967
1	KNN	Under-Sampled	0.890	1.000
5	Decision Tree	Under-Sampled	0.866	0.982
8	SVM	Original	0.808	0.829
0	KNN	Original	0.800	0.923
4	Decision Tree	Original	0.800	0.881
10	Logistic Regression	Original	0.633	0.884

- The oversampled Decision Tree had the highest recall at 0.98, making it particularly effective.
- The SMOTE-sampled Decision Tree and Logistic Regression models closely followed it.
- The original unsampled Logistic Regression had the lowest recall of 0.63, indicating a significant number of fraudulent transactions went undetected.

3. **F1-Score (Fraud)**

	Model	Sampling	F1-Score (Fraud)	Accuracy
6	Decision Tree	Over-Sampled	0.979	0.979
7	Decision Tree	SMOTE	0.972	0.972
12	Logistic Regression	Over-Sampled	0.947	0.949
13	Logistic Regression	SMOTE	0.947	0.949
9	SVM	Under-Sampled	0.943	0.943
1	KNN	Under-Sampled	0.942	0.943
11	Logistic Regression	Under-Sampled	0.939	0.939
5	Decision Tree	Under-Sampled	0.921	0.923
0	KNN	Original	0.857	1.000
4	Decision Tree	Original	0.838	0.999
8	SVM	Original	0.819	0.999
10	Logistic Regression	Original	0.738	0.999

- The oversampled Decision Tree attained the highest F1-Score, making it the top model for fraud prediction among all tested methods.
- The top-performing techniques were predominantly from oversampling and SMOTE methods.

However, it's essential to recognize that these techniques can introduce synthetic data, potentially leading to misleading performance metrics.

5.4 Exclusion of KNN Oversampled and SMOTE

The KNN models trained on oversampled and SMOTE data were excluded for several reasons:

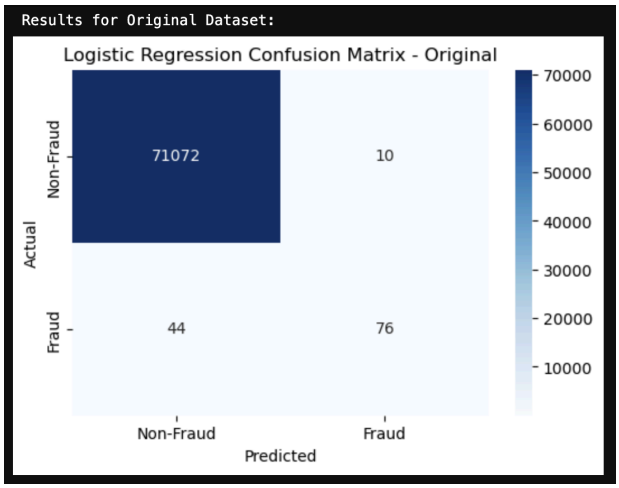
- **Artificial Data Points**: Oversampling and SMOTE create synthetic samples that can distort the natural data distribution, leading to overfitting in KNN, which relies heavily on local neighborhoods.

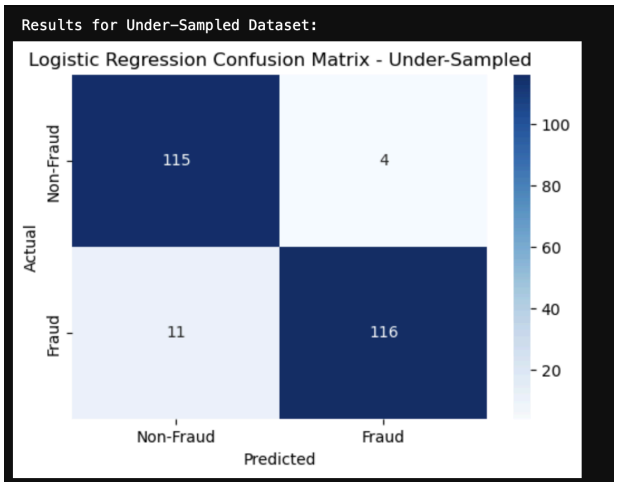
- **Redundant Application of SMOTE**: KNN on synthetic data generated by SMOTE is counterproductive, as it doesn't add value to the learning process.
- **Skewed Performance Metrics**: Models trained on artificially augmented data may present misleadingly high metrics, giving a false sense of performance reliability.

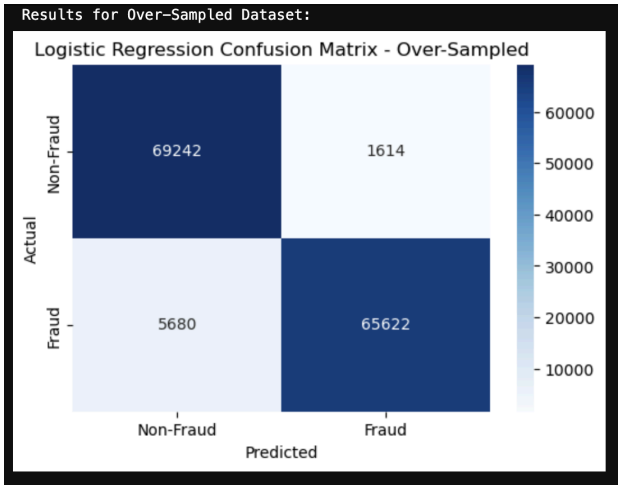
Lastly, the SVM model using oversampled and SMOTE data was not evaluated due to extended processing times, emphasizing the need for practical considerations in model evaluation.

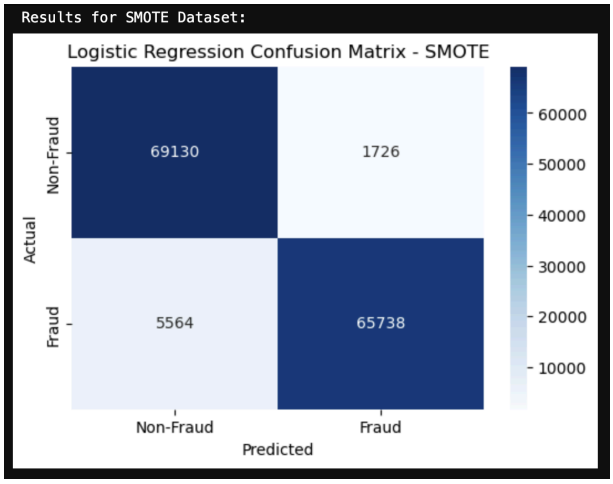
5.5 Confusion Matrices

1. Logistic regression

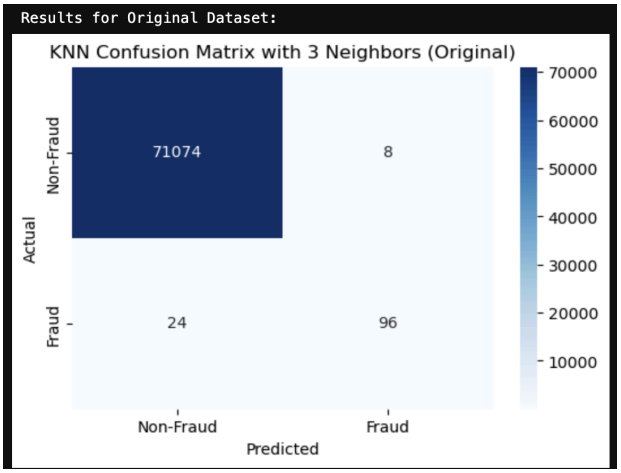


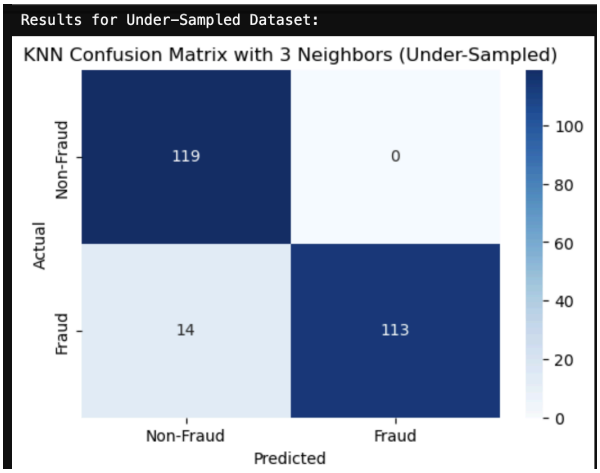


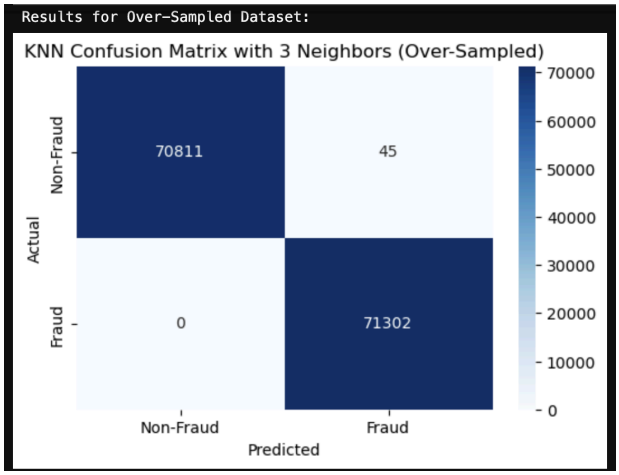


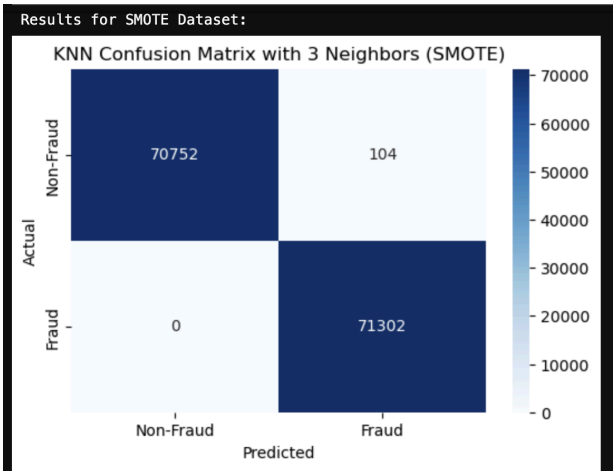


2. KNN

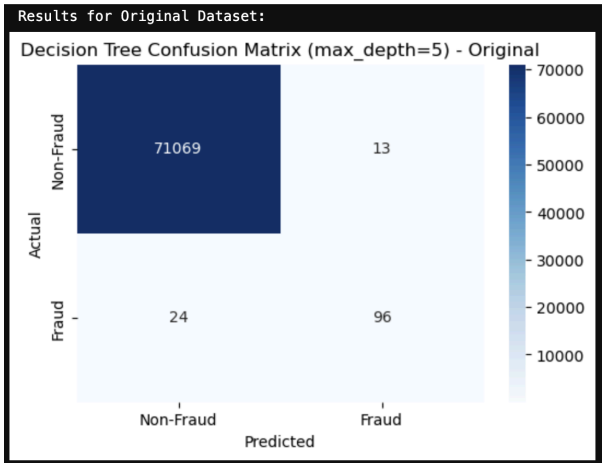


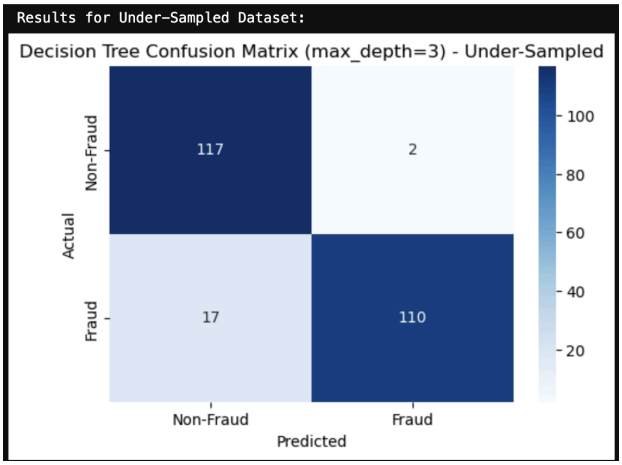


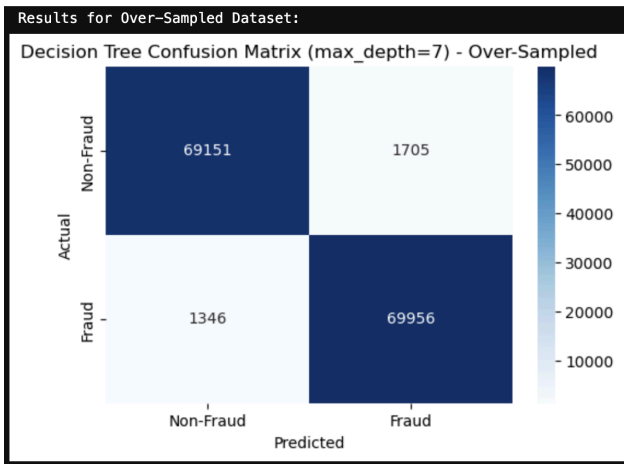


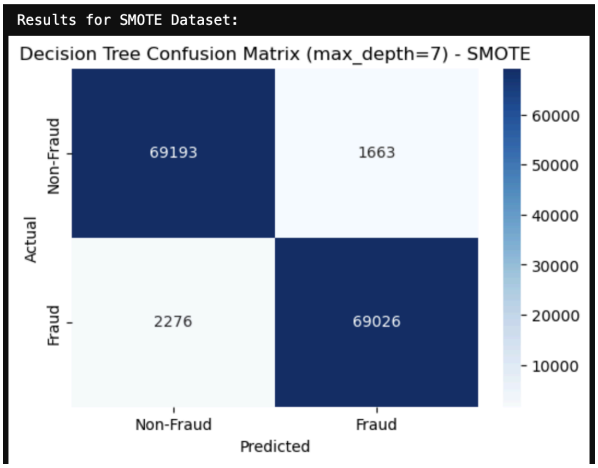


3. Decision Trees

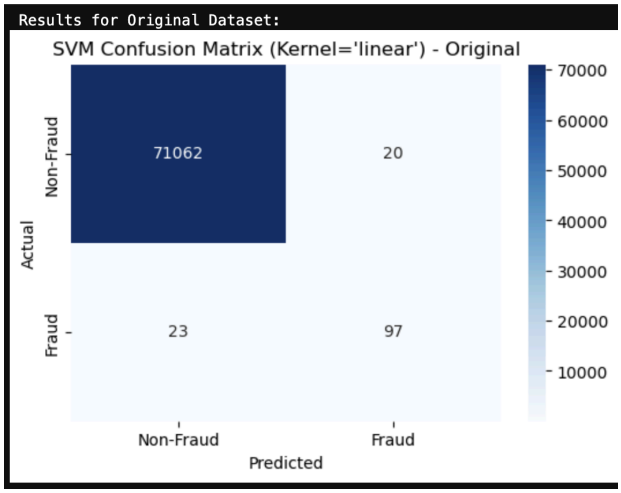


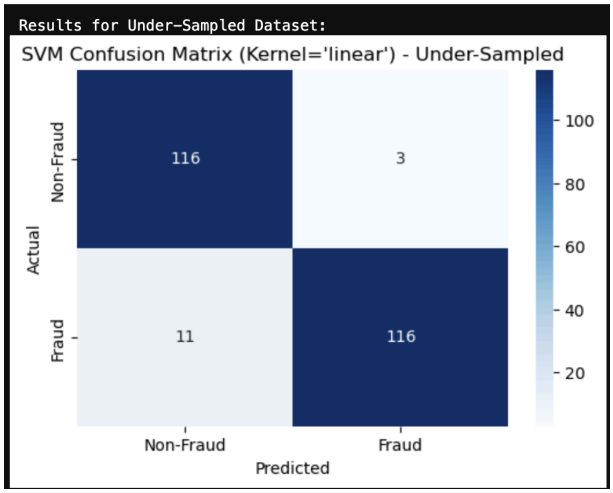






4. Support Vector Machines (SVM)





6. Conclusion

In this analysis, various resampling techniques and machine learning models were explored to develop an effective fraud detection model. The results highlight key findings for each model and sampling approach, as well as the strengths and limitations in detecting fraudulent transactions. Here is a summary of the conclusions:

Key Takeaways and Recommendations

- **Importance of Resampling**: This analysis underscores the value of resampling techniques, particularly SMOTE and oversampling, in improving model performance on imbalanced datasets. Models trained with these techniques generally exhibited higher recall, which is critical for fraud detection.
- **Model Selection for Fraud Detection**: Decision Tree models paired with SMOTE and oversampling emerged as top-performing combinations for fraud detection, combining high recall with balanced metrics.
- **Implications for Real-World Application**: Precision and recall are critical in fraud detection, as high precision reduces false positives, avoiding disruptions for genuine customers, while high recall minimizes undetected fraud cases, reducing potential losses for credit card companies.
- Over-sampling techniques like SMOTE are valuable for fraud detection tasks; however, models like KNN may be less suitable for synthetic data, where artificial points can distort natural distributions.