Project Report

CSE422: Artificial Intelligence

Section: 05

Group No: 07

Submitted by:

Shahriar Khandoker

ID: 24141122

shahriar.khandoker@g.bracu.ac.bd

Muftasim Fuad Mahee

ID: 22201317

muftasim.fuad.mahee@g.bracu.ac.bd

Submission Date: 07 January 2025

TABLE OF CONTENTS

Introduction

E-commerce fraud has been a major concern in the digital marketplace. It poses significant financial risks to businesses and breaking customer trust. Frauds exploit vulnerabilities through techniques such as stolen credit card usage, account takeovers and in many other ways. Addressing these issues is critical to safeguarding the integrity of e-commerce platforms and ensuring a secure shopping experience for consumers. We developed this project to create a fraudulent e-commerce transaction detection system using machine learning techniques. By analyzing customer data, transaction details and patterns, the system seeks to identify suspicious activities with high accuracy. This not only minimizes financial losses, but also enhances operational efficiency, and strengthens customer trust in e-commerce platforms. Our motivation for undertaking this project stems from the increasing prevalence of e-commerce fraud and the potential of machine learning to offer scalable and robust solutions. By leveraging advanced algorithms and data analysis, we hope to contribute to a safer and more reliable e-commerce ecosystem.

Dataset Description

The "Fraudulent E-Commerce Transactions" synthetic dataset is created to mimic transaction data from an e-commerce platform, emphasizing fraud detection. It includes various features typically present in transactional data, along with extra attributes specifically designed to aid in the development and testing of fraud detection algorithms.

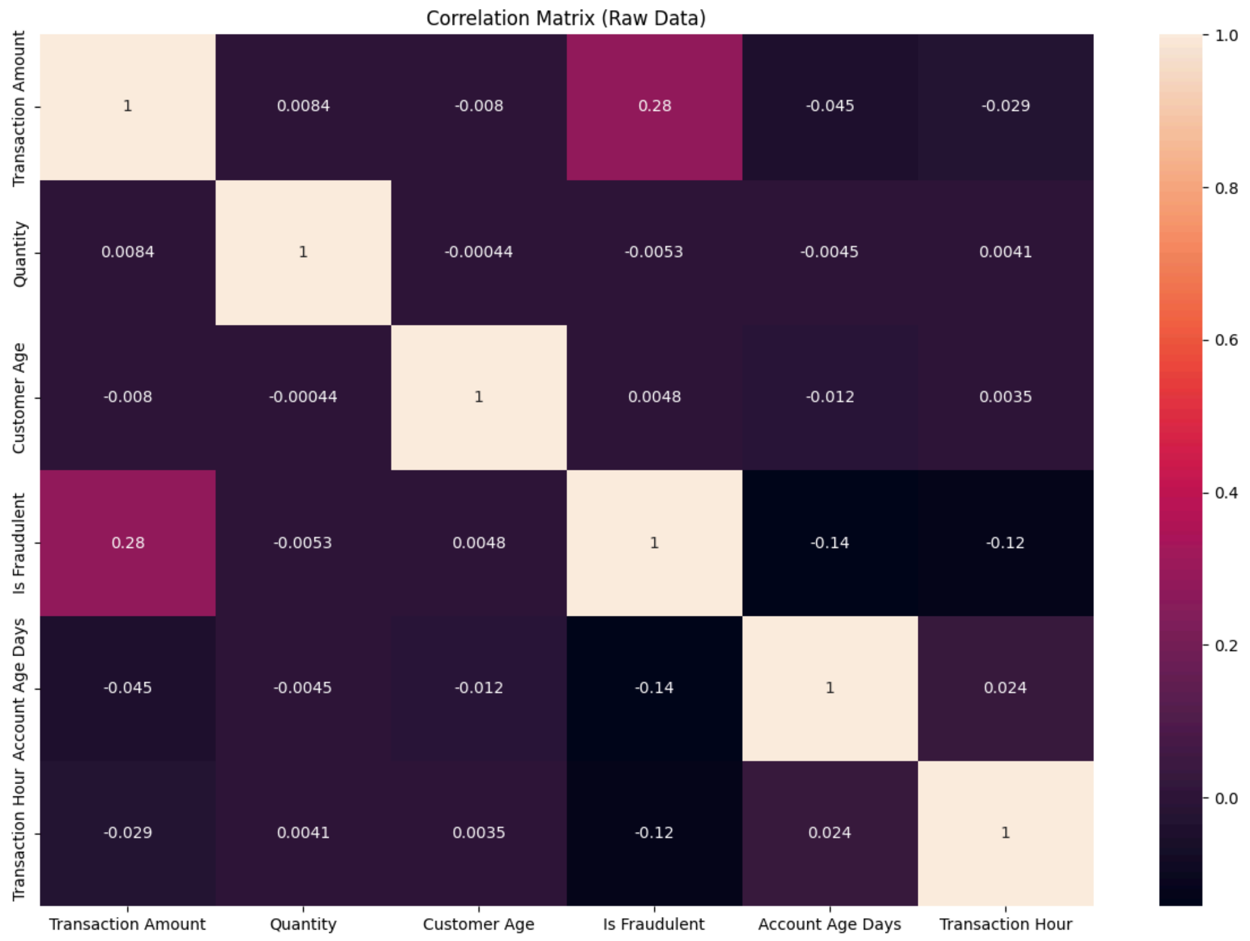
Source: 🚨 Fraudulent E-Commerce Transactions 💳
Reference: *Fraudulent E-Commerce transactions.* (2024, April 7). Kaggle. https://www.kaggle.com/datasets/shriyashjagtap/fraudulent-e-commerce-transactions?select=Fraudulent_E-Commerce_Transaction_Data_2.csv

Dataset Characteristics:

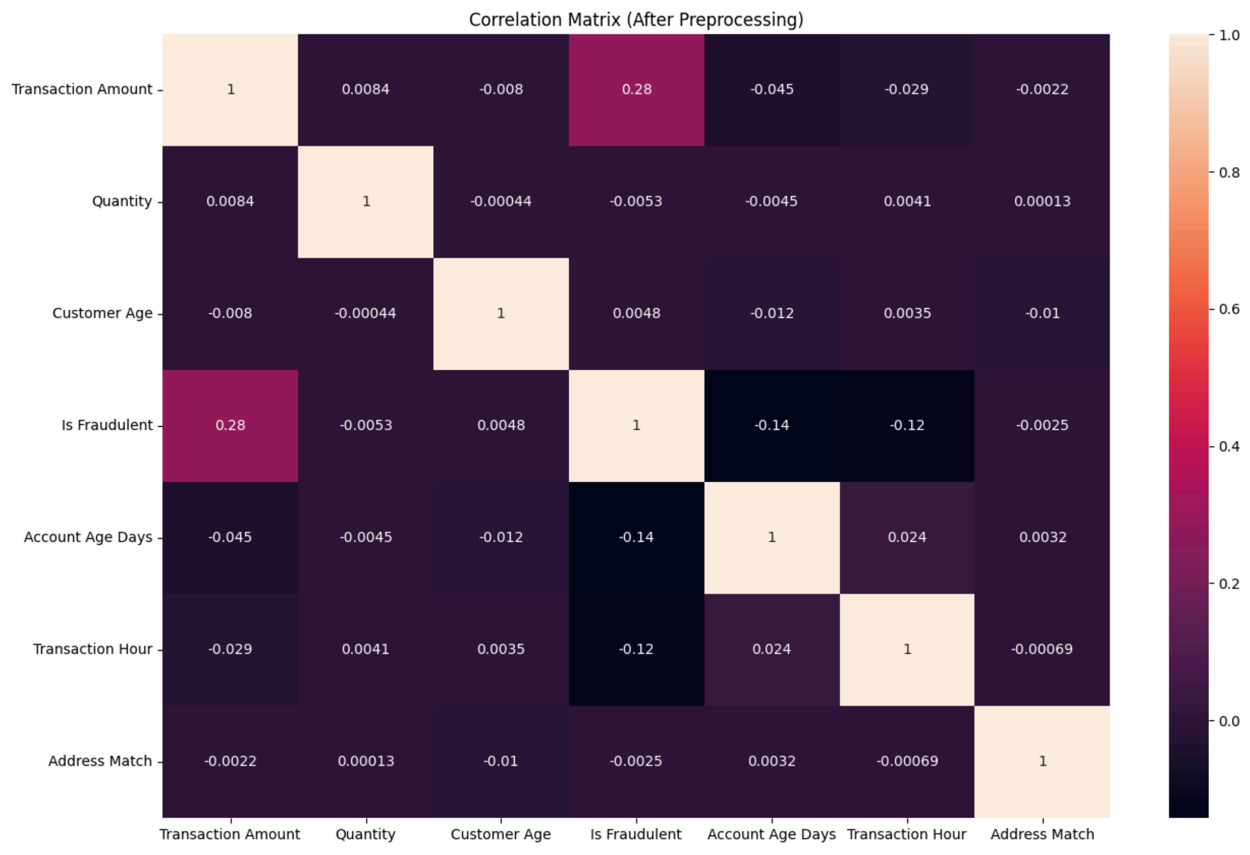
- **Number of features:** 16
- **Problem type:** Classification problem. The target variable "Is Fraudulent" has binary categories such as 0 (Not Fraudulent) and 1 (Fraudulent).
- **Number of Data Points:** There are 23634 instances, each representing a transaction.
- **Types of features:** The dataset consists of both quantitative and categorical features.
    - **Quantitative:** 'Transaction Amount', 'Quantity', 'Customer Age', 'Is Fraudulent', 'Account Age Days', 'Transaction Hour'
    - **Categorical:** 'Transaction ID', 'Customer ID', 'Transaction Date', 'Payment Method', 'Product Category', 'Customer Location', 'Device Used', 'IP Address',

'Shipping Address', 'Billing Address'

- **Correlation Analysis:** A heatmap was created using the Seaborn library to examine the correlation among all features, including both input and output attributes. This visualization aids in comprehending the relationships and potential collinearity between various variables, which is crucial for the model selection and feature engineering stages.

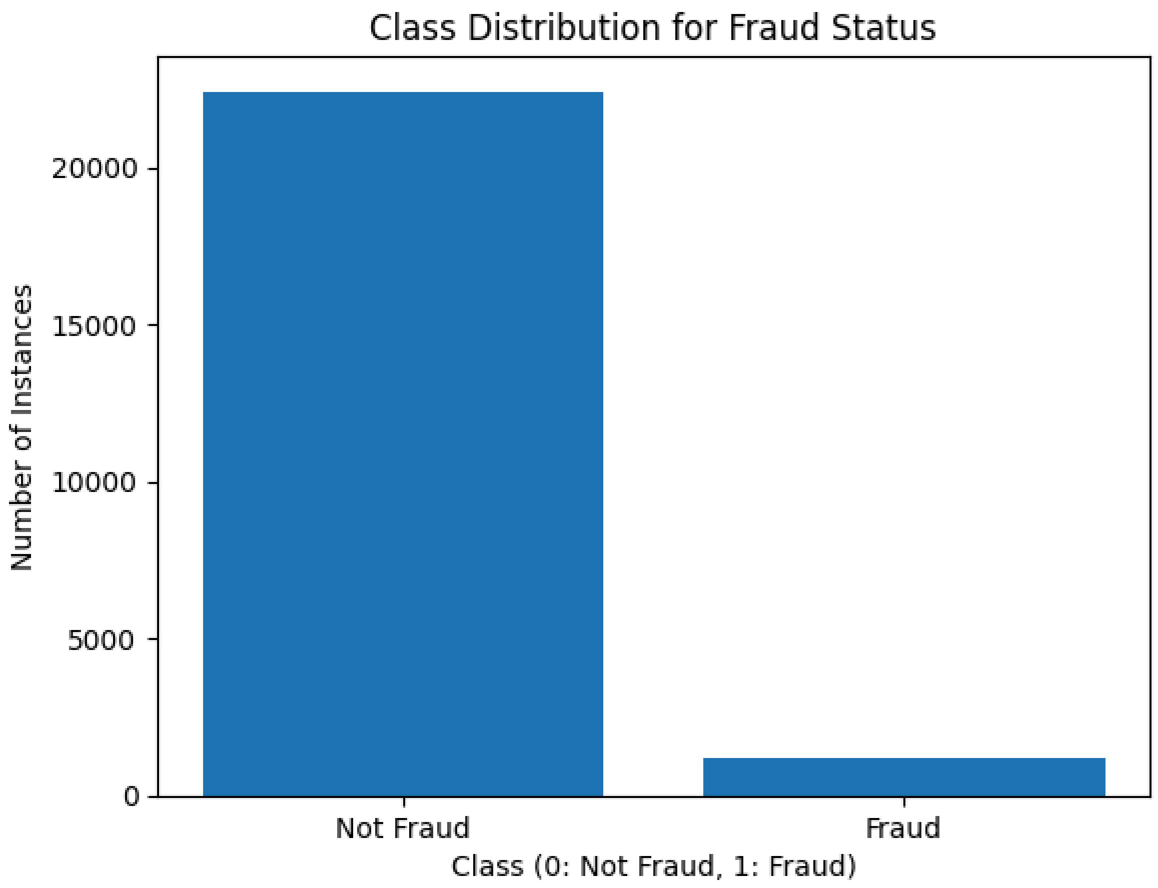


Another correlation matrix was created after preprocessing the dataset to visualize the influence of preprocessed data on the features:



	Transaction Amount	Quantity	Customer Age	Is Fraudulent	Account Age Days	Transaction Hour	Address Match
Transaction Amount	1	0.0084	-0.008	0.28	-0.045	-0.029	-0.0022
Quantity	0.0084	1	-0.00044	-0.0053	-0.0045	0.0041	0.00013
Customer Age	-0.008	-0.00044	1	0.0048	-0.012	0.0035	-0.01
Is Fraudulent	0.28	-0.0053	0.0048	1	-0.14	-0.12	-0.0025
Account Age Days	-0.045	-0.0045	-0.012	-0.14	1	0.024	0.0032
Transaction Hour	-0.029	0.0041	0.0035	-0.12	0.024	1	-0.00069
Address Match	-0.0022	0.00013	-0.01	-0.0025	0.0032	-0.00069	1

- **Imbalanced Dataset:** The “Is Fraudulent” classes are imbalanced. A bar chart was plotted to show the distribution of instances across categories:

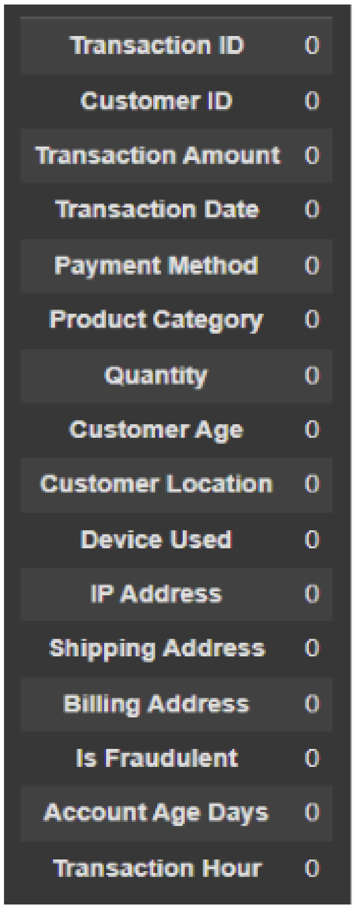


Dataset Preprocessing

Preparing the dataset is an essential phase in data analysis and predictive modeling. Ensuring the data's integrity and quality is vital for obtaining reliable results. During this stage, we addressed missing values, duplicate data points, and categorical data

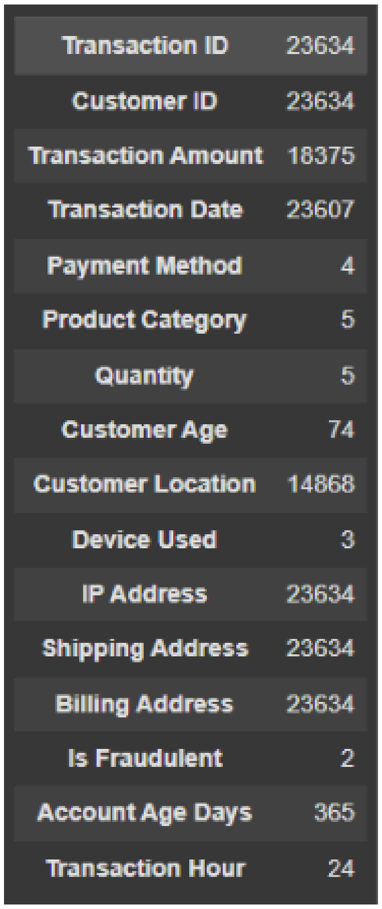
Null Values and Duplicate Data: After examining our dataset, we found no duplicate entries or missing values. Therefore, we did not need to perform any cleaning or imputation in this section.

No Null Values

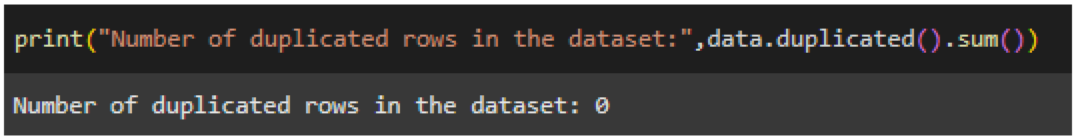


Transaction ID	0
Customer ID	0
Transaction Amount	0
Transaction Date	0
Payment Method	0
Product Category	0
Quantity	0
Customer Age	0
Customer Location	0
Device Used	0
IP Address	0
Shipping Address	0
Billing Address	0
Is Fraudulent	0
Account Age Days	0
Transaction Hour	0

No Duplicate Values



Transaction ID	23634
Customer ID	23634
Transaction Amount	18375
Transaction Date	23607
Payment Method	4
Product Category	5
Quantity	5
Customer Age	74
Customer Location	14868
Device Used	3
IP Address	23634
Shipping Address	23634
Billing Address	23634
Is Fraudulent	2
Account Age Days	365
Transaction Hour	24



```
print("Number of duplicated rows in the dataset:",data.duplicated().sum())
```

```
Number of duplicated rows in the dataset: 0
```

Dropping irrelevant features:

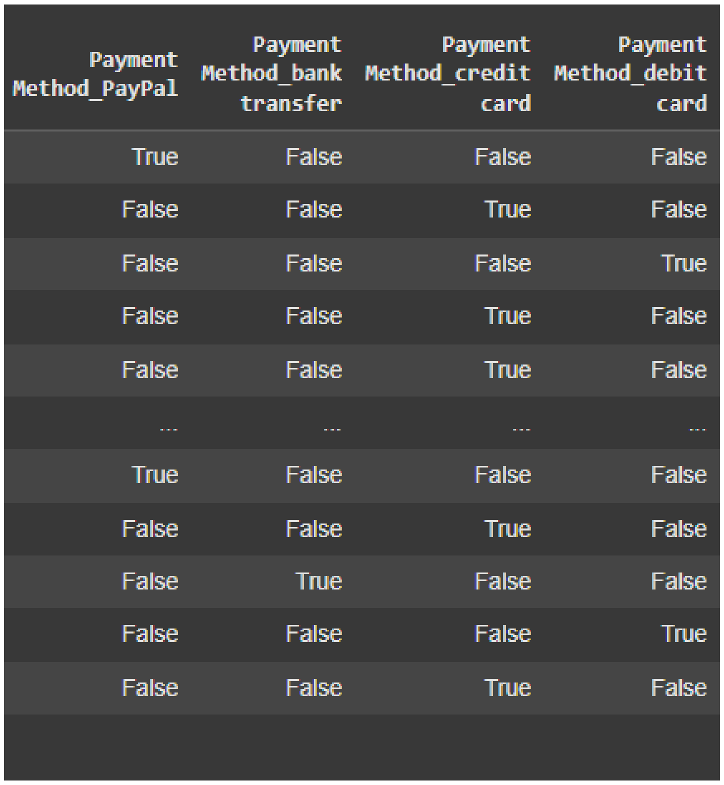
- **Problem:** The dataset contained several irrelevant features that needed to be removed to enhance the efficiency and accuracy of the analysis. These features included "Transaction ID," "Customer ID," "Customer Location," "Transaction Date," "IP Address," "Shipping Address," and "Billing Address."

- **Solution:**The dataset was refined by eliminating unnecessary features and downcasting the data types to minimize its size. The features like “Transaction ID", "Customer ID", "Customer Location", "Transaction Date", "IP Address", "Shipping Address" and "Billing Address" were dropped.

Handling Categorical Values:

- **Problem:** During the dataset preprocessing, we encountered categorical variables. These variables present a challenge for machine learning models, as they require numerical input for computations. The features 'Payment Method', 'Product Category' and 'Device Used’ consist of categorical variables. These values need to be encoded into numerical variables.

- **Solution:** One hot encoding was used to solve the issue. This method is appropriate for categorical values as it transforms them into a straightforward boolean logic.



Payment Method_PayPal	Payment Method_bank transfer	Payment Method_credit card	Payment Method_debit card
True	False	False	False
False	False	True	False
False	False	False	True
False	False	True	False
False	False	True	False
...	...	...	...
True	False	False	False
False	False	True	False
False	True	False	False
False	False	False	True
False	False	True	False

Here we can see the Payment Method was split into 5 columns, each containing a boolean logic of a unique value of Payment Method.

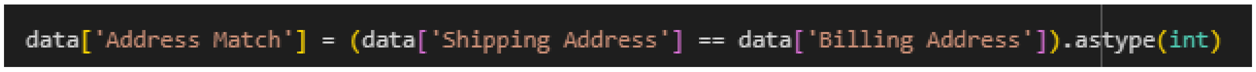
Handling Imbalance:

- **Problem:** One of the major challenges with our dataset is our target class is massively imbalanced. We have 22,412 examples of not fraudulent datapoint, but only 1222 for fraudulent target class. This poses significant problems for our models such as: a bias towards majority class, misleading metrics, poor generalization for minority class etc

- **Solution:** To address this problem, we use a very common technique called SMOTE. SMOTE is a data augmentation technique for addressing class imbalance. Using this, we first oversample the minority class by generating synthetic data. Then, we also undersample the majority class by randomly choosing values. We evaluate our models on both these samples and found out that SMOTE significantly outperforms its counterpart.

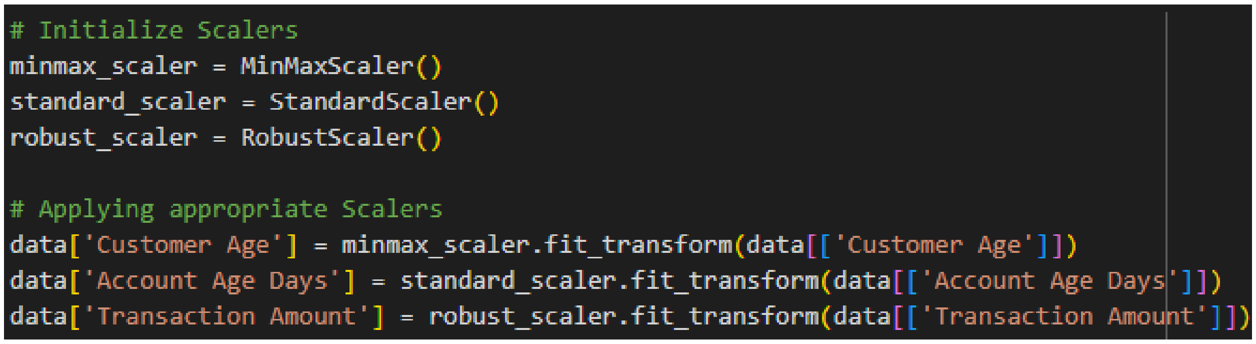
Feature Scaling

Feature Engineering: The feature engineering technique involves creating a new feature, “Address Match”, to capture whether the shipping and billing address are the same or not. In fraud detection cases, mismatched addresses can be an indication of suspicious activity which is why this technique was considered in this case. It’s basically done by encoding a binary feature (1 for Match, otherwise 0).



```
data['Address Match'] = (data['Shipping Address'] == data['Billing Address']).astype(int)
```

Scaling: The process of feature scaling is crucial, particularly for models sensitive to the size of input values. The aim is to make sure that numerical features are equally weighted, preventing the model from being biased towards features with larger values. For this dataset, feature scaling was conducted on the features 'Transaction Amount', 'Customer Age', 'Account Age Days' and 'Transaction Hour'. The StandardScaler, MinMaxScalar and Robust Scalar were employed to standardize features.



```
# Initialize Scalers
minmax_scaler = MinMaxScaler()
standard_scaler = StandardScaler()
robust_scaler = RobustScaler()

# Applying appropriate Scalers
data['Customer Age'] = minmax_scaler.fit_transform(data[['Customer Age']])
data['Account Age Days'] = standard_scaler.fit_transform(data[['Account Age Days']])
data['Transaction Amount'] = robust_scaler.fit_transform(data[['Transaction Amount']])
```

Dataset Splitting

The dataset was split into training and testing subsets using a stratified split to ensure that the class distribution in the target variable (Y) is preserved across the training and testing sets.

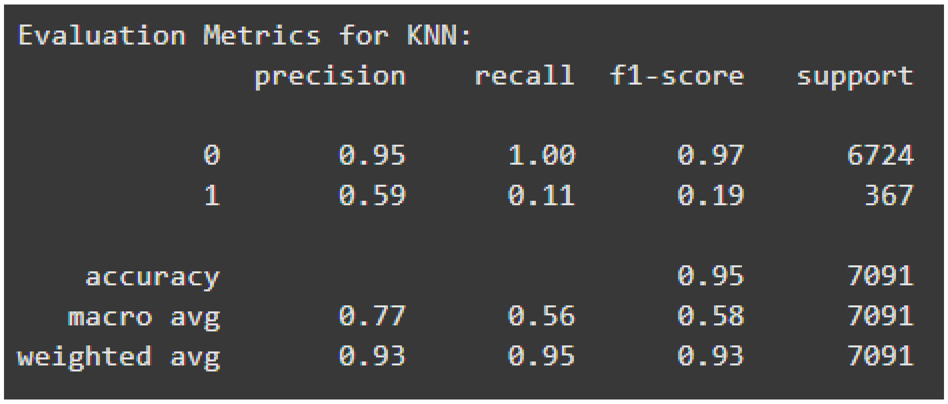
- **Training Set**: 70%
- **Test Set**: 30%

We divided the data with a 70:30 ratio, dedicating 70% for training the model and 30% for validation testing. This approach ensures proper learning the models while keeping enough data to effectively evaluate the models. The shapes of the training and testing subsets were initially (16543, 9) and (7091, 9). Although, while applying undersampling and oversampling these values were changed respectively.

Model Training and Testing

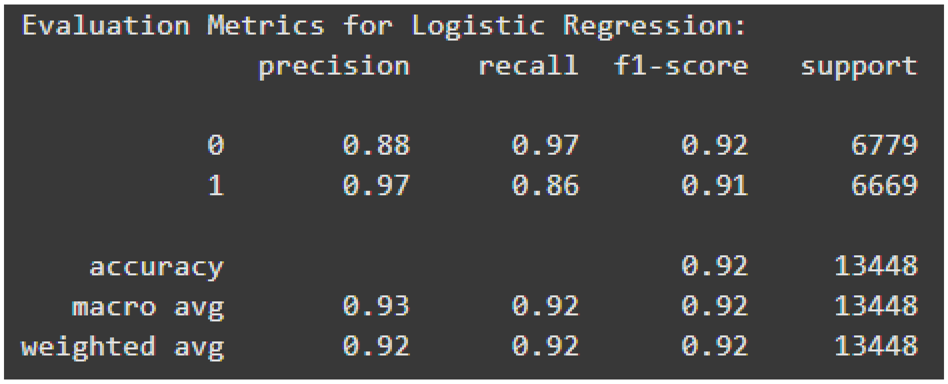
Once data preparation was complete, we utilized five distinct models for predicting the fraud in transactions.

- **K-Nearest Neighbors (KNN)**: KNN is a non-parametric, instance-based learning algorithm that classifies data by finding the majority label among the k closest neighbors. Its simplicity and reliance on distance metrics make it effective for smaller datasets but computationally expensive for large-scale problems.



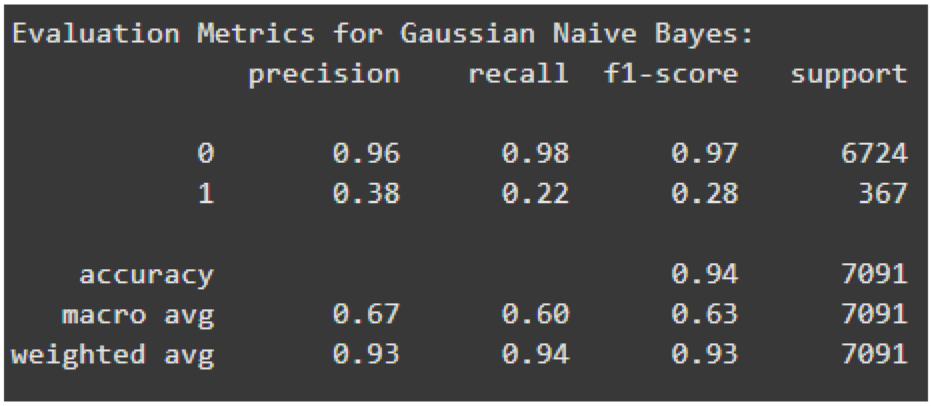
```
Evaluation Metrics for KNN:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97      6724
           1       0.59      0.11      0.19       367

    accuracy                           0.95      7091
   macro avg       0.77      0.56      0.58      7091
weighted avg       0.93      0.95      0.93      7091
```

- **Logistic Regression:** Logistic Regression is a linear model for binary or multi-class classification. It uses the sigmoid function to model probabilities, balancing simplicity with interpretability. It is often a strong baseline due to its efficiency and ease of implementation.



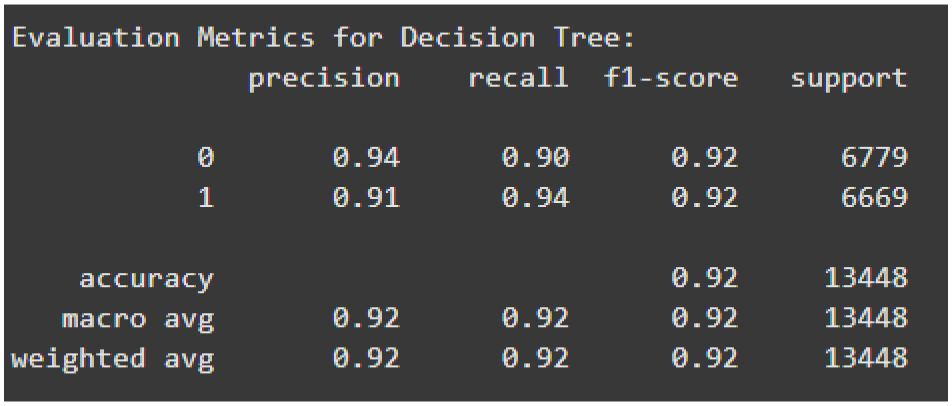
```
Evaluation Metrics for Logistic Regression:
              precision    recall  f1-score   support

           0       0.88      0.97      0.92      6779
           1       0.97      0.86      0.91      6669

    accuracy                           0.92     13448
   macro avg       0.93      0.92      0.92     13448
weighted avg       0.92      0.92      0.92     13448
```

- **Gaussian Naive Bayes:** Gaussian Naive Bayes assumes feature independence and a Gaussian distribution for continuous features. Despite its simplicity, it performs well on small or noisy datasets, making it a quick and effective probabilistic classifier.



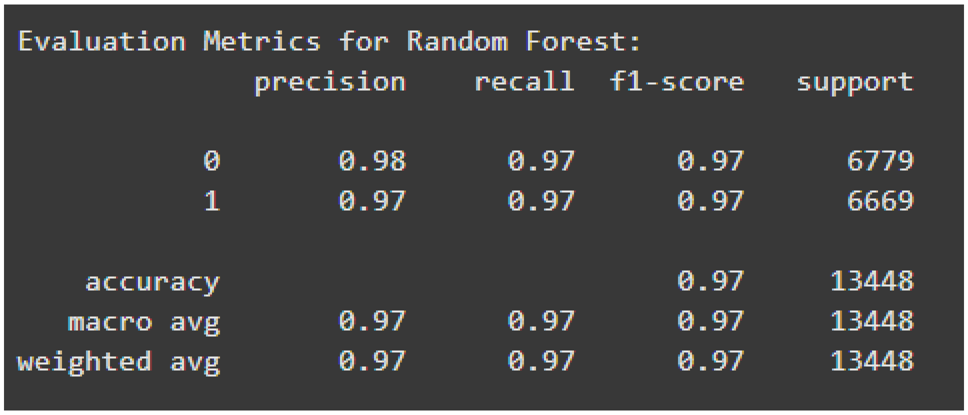
```
Evaluation Metrics for Gaussian Naive Bayes:
              precision    recall  f1-score   support

           0       0.96      0.98      0.97      6724
           1       0.38      0.22      0.28       367

    accuracy                           0.94      7091
   macro avg       0.67      0.60      0.63      7091
weighted avg       0.93      0.94      0.93      7091
```

- **Decision Tree:** Decision Trees are hierarchical models that split data based on feature thresholds to create branches. They are interpretable and effective for non-linear patterns but prone to overfitting on complex datasets.



```
Evaluation Metrics for Decision Tree:
              precision    recall  f1-score   support

           0       0.94      0.90      0.92      6779
           1       0.91      0.94      0.92      6669

    accuracy                           0.92     13448
   macro avg       0.92      0.92      0.92     13448
weighted avg       0.92      0.92      0.92     13448
```

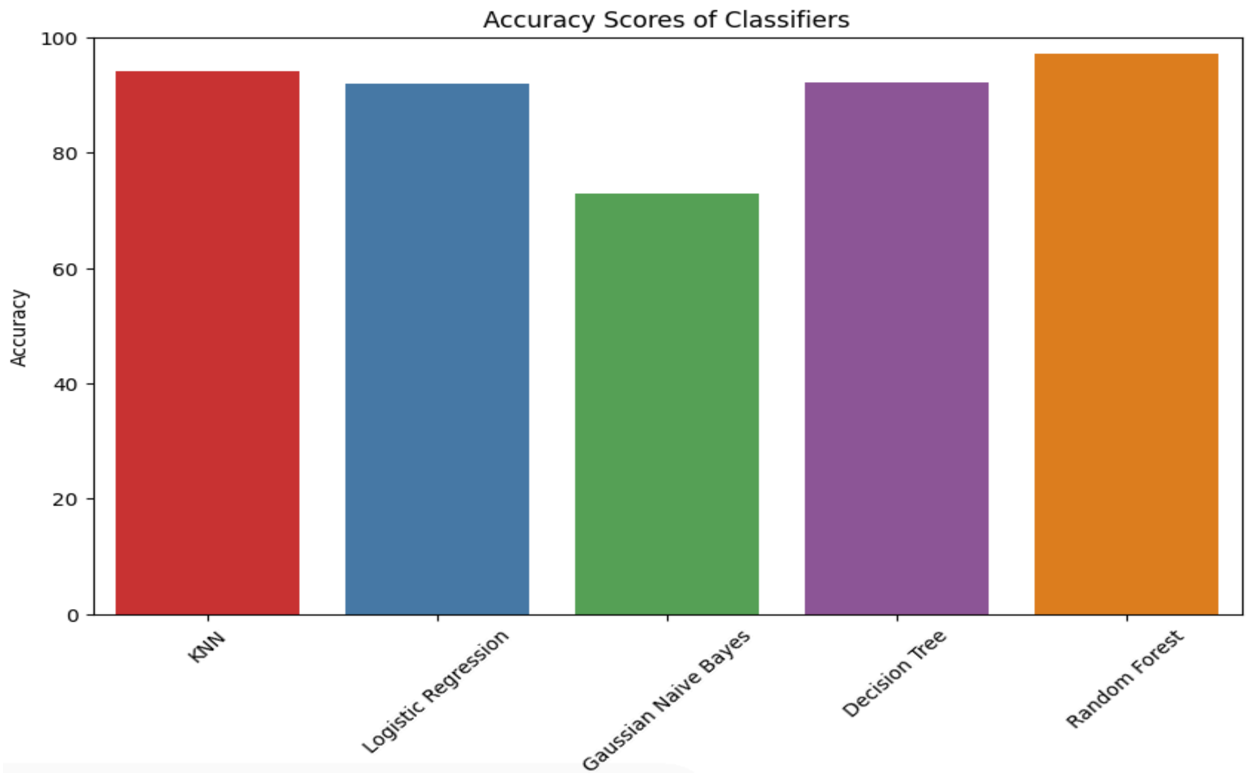
- **Random Forest:** Random Forest combines multiple decision trees trained on random subsets of data and features. This ensemble approach enhances generalization, reduces overfitting, and is robust for both classification and regression tasks.



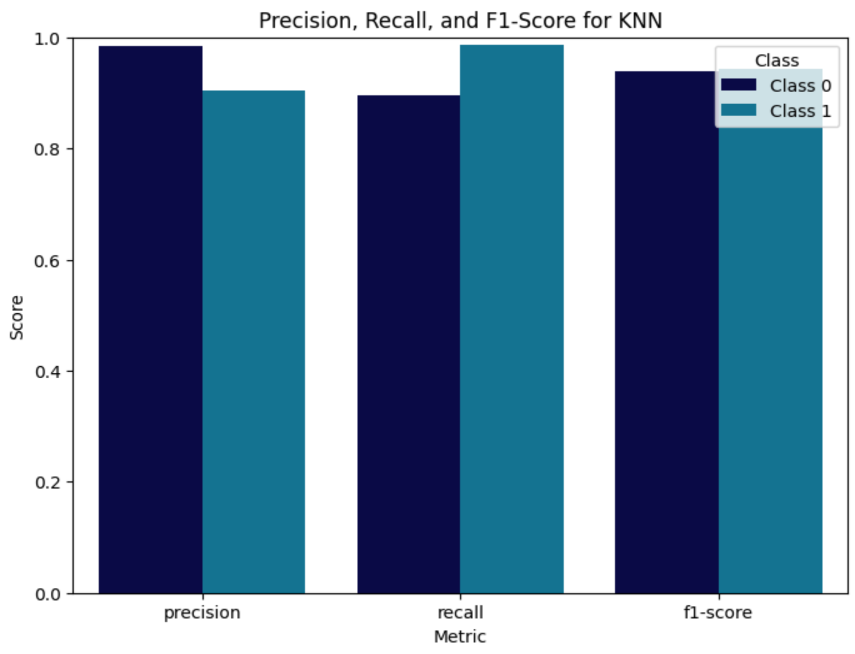
```
Evaluation Metrics for Random Forest:
              precision    recall  f1-score   support

           0       0.98      0.97      0.97      6779
           1       0.97      0.97      0.97      6669

    accuracy                           0.97     13448
   macro avg       0.97      0.97      0.97     13448
weighted avg       0.97      0.97      0.97     13448
```

Model Selection/Comparison Analysis

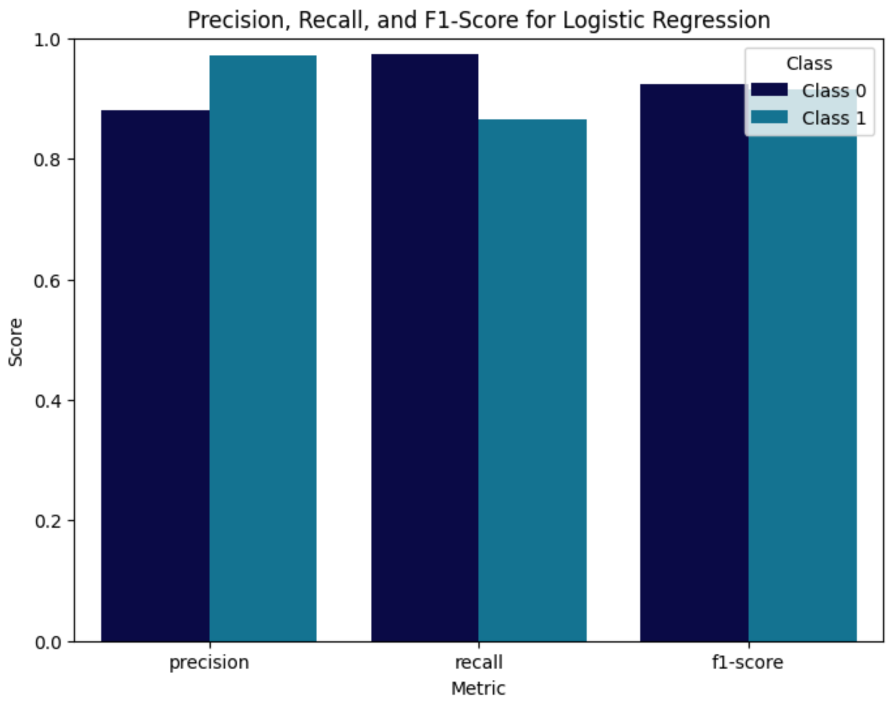
Barchart showcasing prediction accuracy across all models:



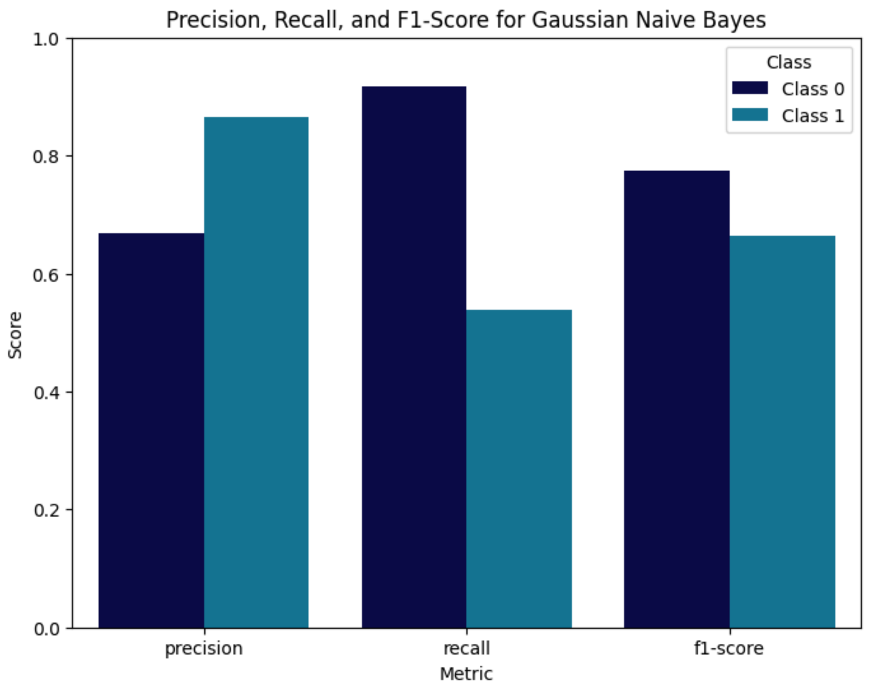
Bar Chart showcasing Precision, Recall, and F1 Score across all models:



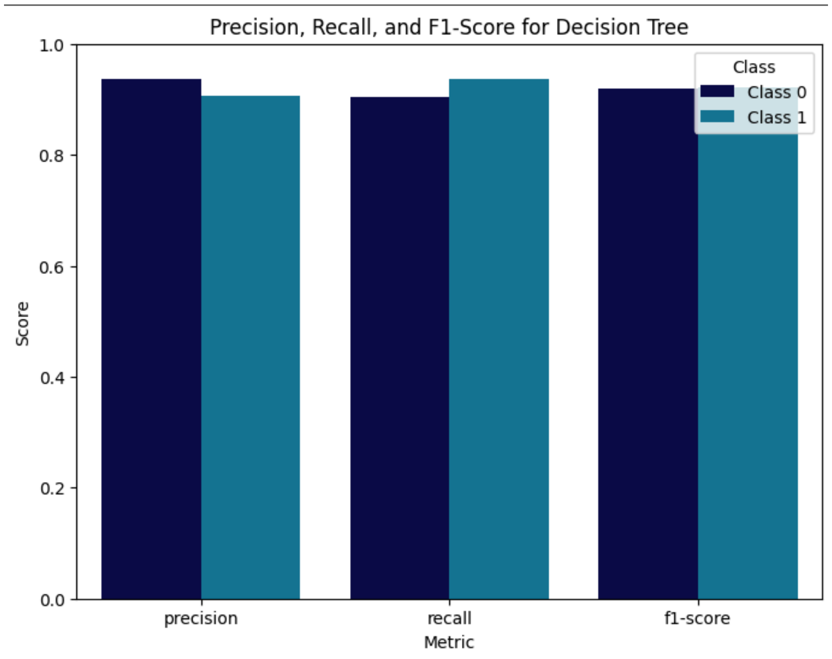
KNN



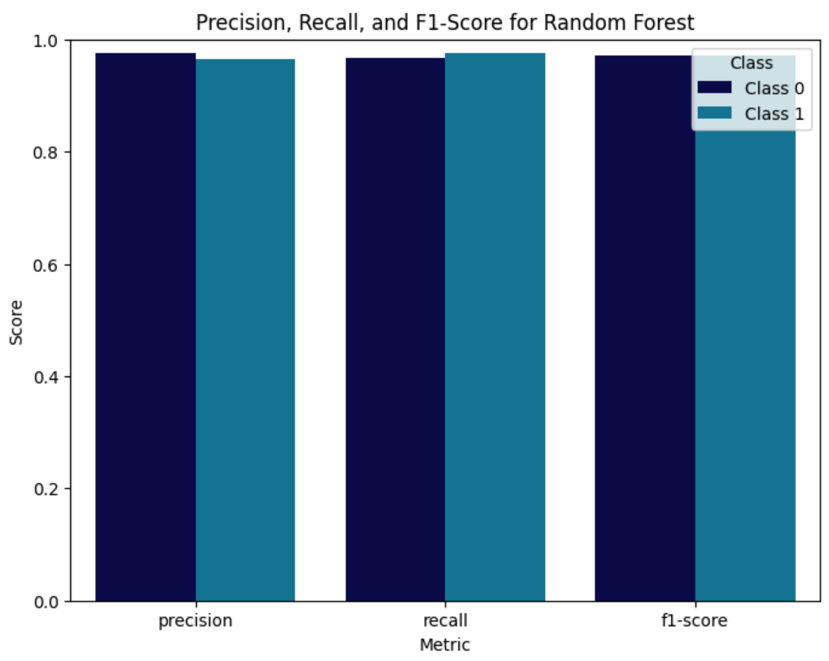
Logistic Regression



Gaussian Naive Bayes



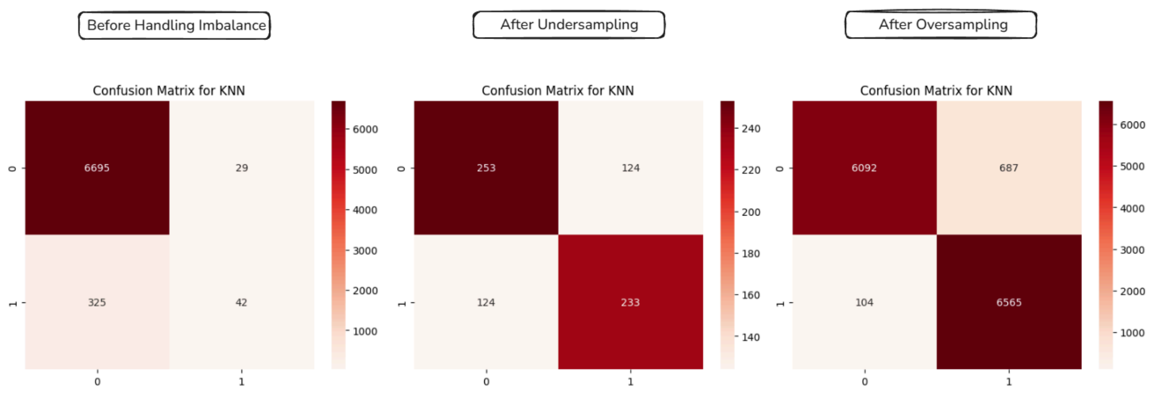
Decision Tree



Random Forest

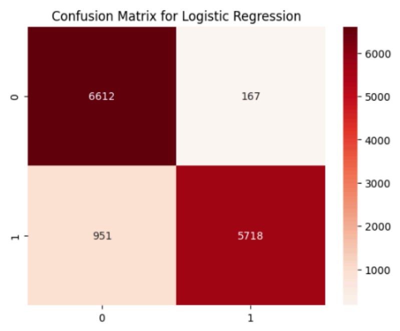
Confusion Matrix:

- **K-Nearest Neighbors (KNN)**:

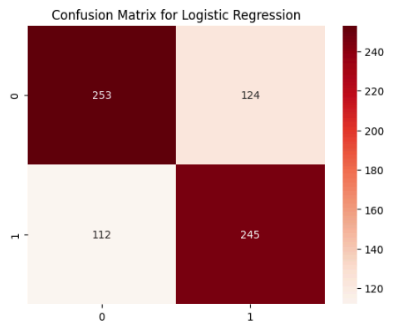


- **Logistic Regression:**

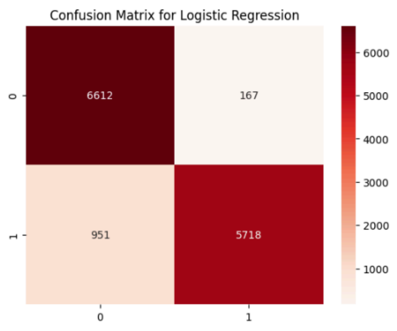
Before Handling Imbalance



After Undersampling

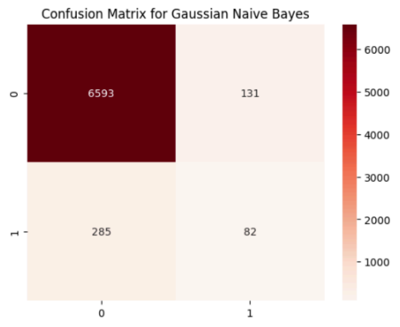


After Oversampling

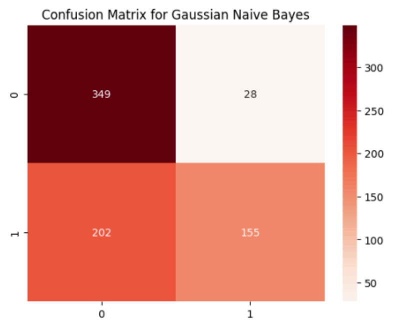


- **Gaussian Naive Bayes:**

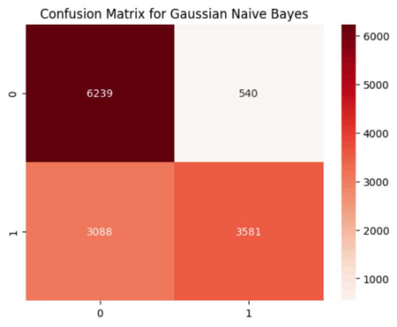
Before Handling Imbalance



After Undersampling

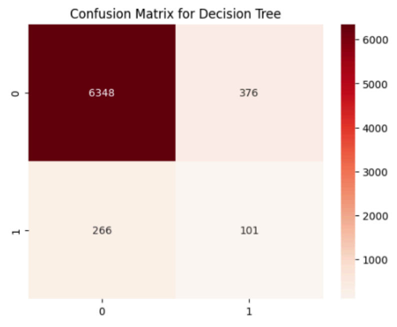


After Oversampling

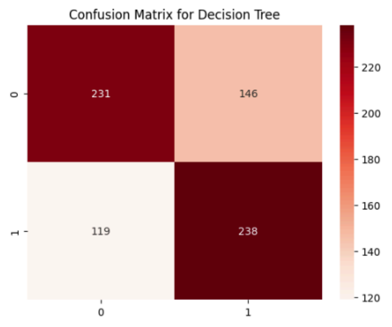


- **Decision Tree:**

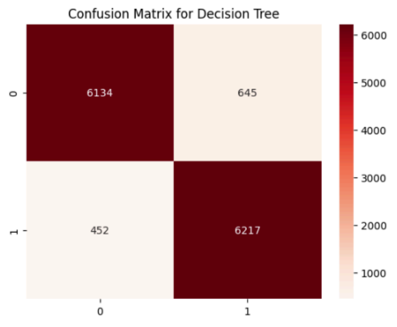
Before Handling Imbalance



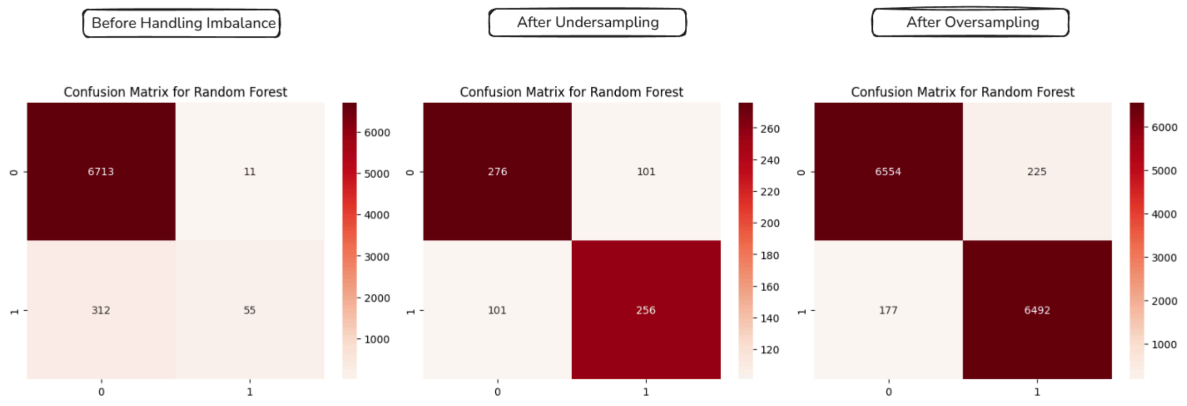
After Undersampling



After Oversampling



- **Random Forest:**



But the result was biased towards 0 which is Not Fraud as the cases of fraud was way much lesser than not fraud. That's why we oversampled the dataset and solved the problem. Now we can see the confusion matrix f

Conclusion

All models demonstrated competitive performance on this challenging dataset. However, their success heavily depended on the preprocessing steps applied. The dataset's massive class imbalance posed a significant challenge, making it difficult to prevent overfitting on the minority class. Additionally, the small sample size hindered the models' ability to generalize effectively.

Despite these obstacles, by employing state-of-the-art preprocessing techniques, we achieved desirable metrics across all models.