Fraud Detection in E-commerce using Machine Learning Techniques

Hari Chandana Kotnani

Introduction:

Our project aims to create a machine learning-based fraud detection system to address the rise of fraudulent transactions in the e-commerce sector. By analyzing customer transaction data from an online retailer of electronic goods, we will use supervised learning algorithms such as logistic regression, decision trees, and ensemble techniques like Gradient boosting, Adaboost, and Random Forests to identify patterns that suggest fraudulent activity, automating the process of identifying potentially fraudulent transactions. The purpose of this project is to provide an efficient and effective fraud detection system for the e-commerce sector, leading to significant cost savings and enhanced customer satisfaction.

Our results will provide businesses with insights into fraudulent transaction patterns in the e-commerce sector, enabling them to reduce expenses and enhance the overall customer experience. By implementing a machine learning-based fraud detection system, businesses in the e-commerce sector can save money and improve customer satisfaction. Our analysis has the potential to significantly impact the e-commerce sector by improving the efficiency and effectiveness of fraud detection.
Slides: Presentation Slides Link Github: GitHub Link

Dataset:

The dataset used in the project is a combination of three sources, including user and transaction information, a list of country names with IP address bounds, and a list of blacklisted IPs associated with fraudulent activities.
Dataset 1: The first dataset contains information about users and their transactions. This dataset includes important data attributes, such as the user's age, gender, and transaction details such as the purchase amount and date.
Dataset 2: The second dataset contains a list of country names with the lower and upper bounds of IP addresses. This dataset is crucial for fraud detection as it helps to map IP addresses to their respective countries, which is a crucial step in identifying fraudulent activities that often involve transactions from multiple countries. The first two datasets were downloaded from Kaggle.
Dataset 3: We have also added a third dataset, which contains a list of IP addresses that are known to be associated with fraudulent activities in the real world. This dataset will help us to identify potentially fraudulent transactions by checking if the IP address in the previous dataset matches with any of the blacklisted IPs. This dataset was taken from a MYIP.MS website that provides information on blacklisted IPs, and was converted from text to CSV format.
Preprocessing: Our dataset has undergone several preprocessing steps, including data cleaning, such as checking for missing values, one-hot encoding, and standardization, to ensure that it is

ready for analysis. Additionally, we have converted IP addresses to quad-dot format and checked them for blacklisting.
The dataset contains 151,112 records, with only 14,151 records classified as fraud, resulting in an imbalanced dataset. Our dataset is suitable for fraud detection, and we made an effort to understand its features, attributes, and potential challenges in order to create accurate machine learning models.

Analysis Technique:

After performing preprocessing and exploratory data analysis (EDA) on our dataset, we identified certain features such as source, user id, sex, browser, country, and age that can be considered as identification information and can be transformed for machine learning models. However, features like device-id, purchase time, signup time, and IP address are activity-based and cannot be directly transformed. Instead, we used feature engineering techniques to generate new feature variables that can be used for machine learning and to identify patterns in our dataset. For example, we added new attributes like time difference between sign-up time and purchase-time, purchase_week, purchase_year, IP users, and total-purchase. By analyzing patterns in the data, we were able to identify potential fraudulent activities, such as multiple users using the same IP address or device-id. These new features were beneficial in improving our analysis and helped us to identify potentially fraudulent transactions [shown with plots in results section]. We dropped some features, such as user_id and device_id, which were not useful for our analysis and used the resulting subset of features as our original dataset.
We split our original dataset into a training set (70%) and a testing set (30%). We then applied various machine learning models to our original dataset, such as decision tree classifier, MLP classifier, K-Neighbors classifier, Random Forest Classifier, and Logistic regression classifier. However, since our dataset is imbalanced, with the non-fraud class having more values (majority class) and the fraud class having fewer values (minority class), we obtained moderate scores. To improve the performance of our machine learning models on the imbalanced dataset, we used ensemble models such as Light Gradient Boosting Machine (LightGBM), Extreme Gradient Boosting (XGboost), Gradient Boosting, Extra trees and AdaBoost classifiers.

To address the issue of data imbalance, we used various sampling techniques like oversampling and undersampling to optimize the performance of machine learning models. To perform undersampling, we used the random sampler technique, which randomly selects a few samples from the majority class to remove and balance the dataset. For oversampling, we used two techniques: SMOTE (Synthetic Minority Over-sampling Technique) and the Adaptive Synthetic (ADASYN) algorithm, which randomly duplicates examples from the minority class to balance the dataset. After balancing the dataset, we applied all the models we used for the original dataset, including ensemble models, and evaluated their performance on the testing set and compared their performance using metrics such as accuracy, precision, recall, and F1-score, as well as a confusion matrix. To obtain the best hyperparameters for all our models, we applied grid searchCV. Finally, we plotted the ROC curve to visualize the performance of our models.

Results:

As mentioned in the analysis technique, after adding new features and conducting exploratory data analysis, we analyzed the dataset using a combination of the new features and existing features and found interesting patterns.

Analysis:

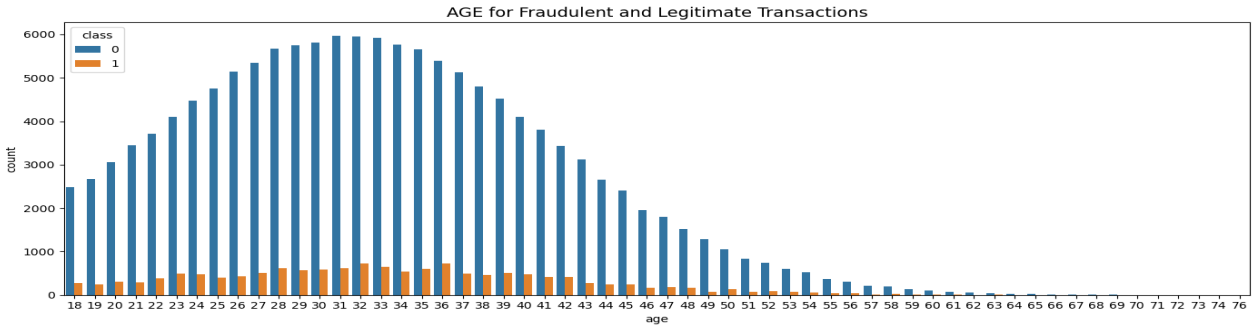


Figure 1: Age for Fradulent and Legitimate Transactions

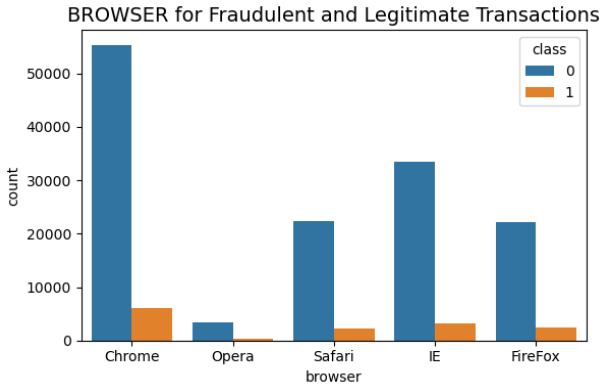


Figure 2: Browser

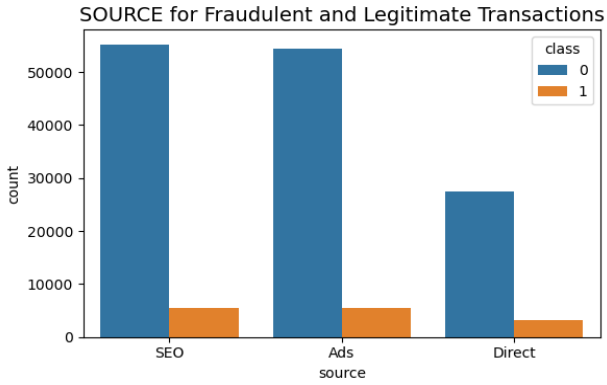


Figure 3: Source

Based on our analysis of the dataset, we have observed certain trends in the distribution of variables for fraudulent and legitimate transactions. In particular, we have visualized the distribution of variables such as age, browser, and source. We have found that individuals in the age group of 22-40 are more likely to engage in fraudulent transactions due to various factors such as financial pressure and risk-taking tendencies as per figure 1. Figure 2 shows, the most commonly used browser for transactions is Chrome, which could potentially make it easier for fraudsters to deceive users using tactics such as popup-ads and extensions. Figure 3 shows the analysis of transaction sources has revealed that fraudsters can exploit SEO and ads through techniques such as click fraud, fake websites, false promises, and botnets. It is important to take measures to prevent and detect fraudulent transactions, especially among individuals in the age group of 25-35, transactions made through popular browsers such as Chrome, and common sources such as SEO and ads.

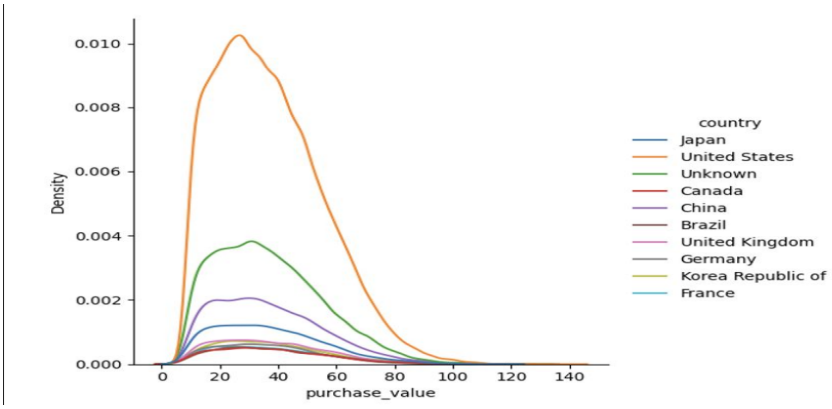


Figure 4: Country with highest fraudulent transactions

As part of our analysis, we also examined the distribution of fraudulent transactions by country. Our findings suggest that the United States has a higher incidence of fraudulent transactions compared to other countries, which could be attributed to factors such as its economic power and technological advancement.

Feature Engineering analysis Findings:

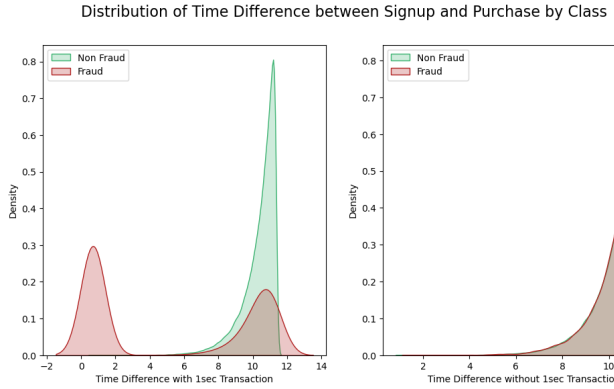


Figure 5 : Time Difference

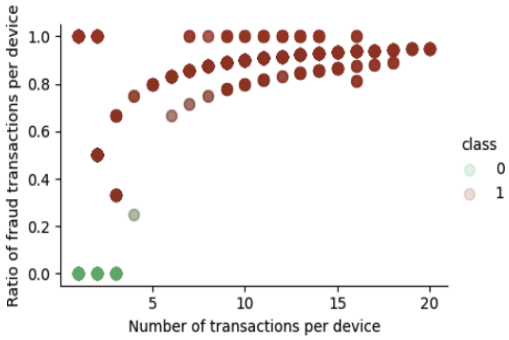


Figure 6: Ratio of fraudulent transactions

We added a new column to the dataset that captures the time difference between the 'signup' and 'purchase' time for each transaction. Upon analyzing this column, we discovered that all transactions with a time difference of 1 second were fraudulent as shown in Figure 5. This finding suggests that transactions with such a small time difference between signup and purchase are likely to be fraudulent and should be closely monitored.

We also observed that repeated IP addresses with different user IDs are more likely to conclude as fraudulent. This indicates that there may be fraudulent activity associated with certain IP addresses, which can be an important factor to consider when evaluating transactions.

On further investigation, we analyzed the ratio of fraudulent transactions and found that if repeated transactions are made from a single device with different user IDs, there is a higher incidence of fraudulent activity among those transactions, shown in Figure 6. This finding highlights the importance of considering not only the IP address but also the device used to make a transaction when assessing the risk of fraud.

Models:

We applied ten different machine-learning algorithms to classify fraudulent transactions. We compared the performance of these algorithms, which included the Decision Tree Classifier, MLP Classifier, K-Neighbors Classifier, and Logistic Regression Classifier, as well as six ensemble classifiers: Light Gradient Boosting Machine (LightGBM), Extreme Gradient Boosting (XGboost), Gradient Boosting, Extra Trees, Random Forest Classifier, and AdaBoost classifiers. We evaluated the performance of each algorithm on three different datasets, which included the original dataset, an undersampled dataset using a random sampler, and an oversampled dataset using SMOTE and ADASYN techniques. Our analysis aimed to determine which algorithms worked best for each dataset, as well as which technique performed the best overall. By comparing the results, we can gain insights into which machine learning models and techniques are most effective for detecting fraudulent transactions.

Original Dataset:

In the table below, we have summarized the performance of various machine learning models for detecting fraudulent transactions. The table shows the accuracy, precision, recall, and F1-score for each model, as well as the best hyperparameters that were used to train the models.

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Hyperparameters**
Decision Tree Classifier	0.95	0.95	0.77	0.83	‘max_depth’: 6
MLP Classifier	0.91	0.95	0.50	0.48	‘hidden_layer_sizes ’=(3), (20, 10, 5)
K Neighbors Classifier	0.88	0.51	0.50	0.50	n_neighbors: 3
Random Forest Classifier	0.95	0.95	0.77	0.83	'max_depth': 8, 'n_estimators': 100
Light Gradient Boosting Machine Classifier	0.95	0.95	0.77	0.83	'learning_rate': 0.1, 'max_depth': 10, 'num_leaves': 10
AdaBoost Classifier	0.95	0.95	0.77	0.83	'learning_rate': 0.01, 'n_estimators': 100

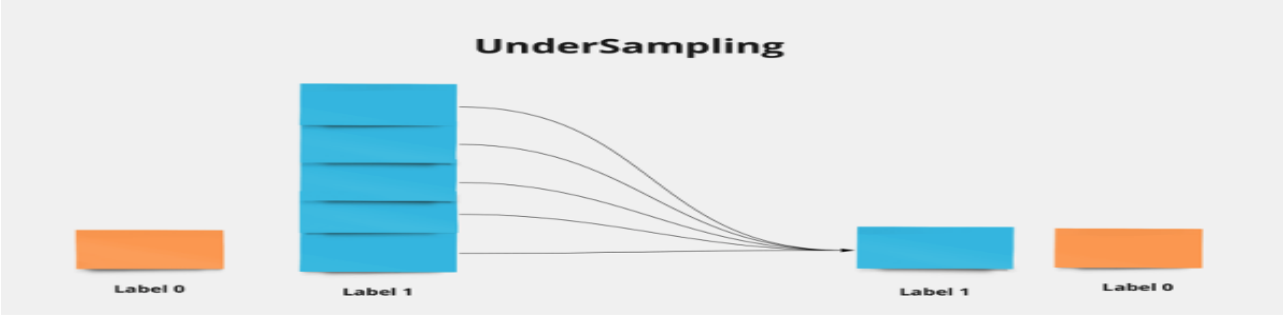
Gradient Boosting Classifier	0.95	0.95	0.77	0.83	'learning_rate': 0.1, 'max_depth': 6, 'n_estimators': 50
Logistic Regression	0.95	0.93	0.76	0.82	'C': 100, 'class_weight': {0: 1, 1: 2}, 'penalty': 'l1'
XGBoost Classifier	0.95	0.95	0.77	0.83	'colsample_bytree': 0.8, 'learning_rate': 0.01, 'max_depth': 7, 'subsample': 0.8
Extra Trees Classifier	0.96	0.96	0.77	0.84	'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 50

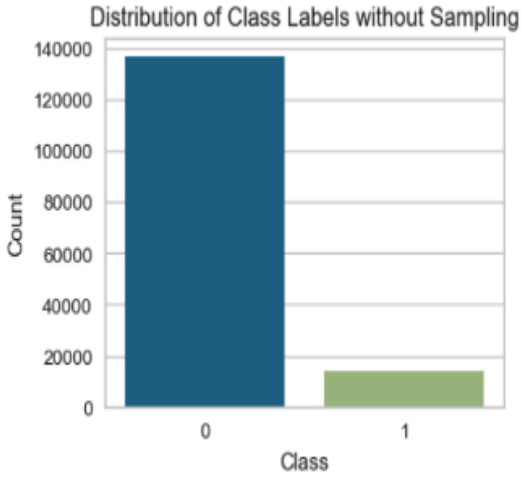
Table 1: Models performance on Original Dataset

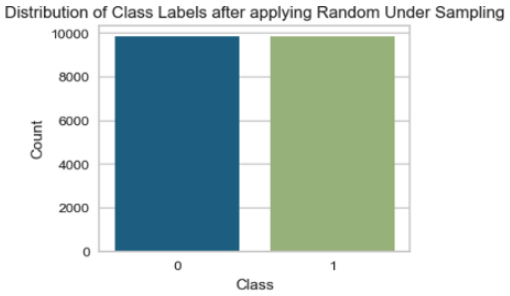
While accuracy is a commonly used metric to evaluate the performance of machine learning models, it may not always be the most appropriate measure, particularly for imbalanced datasets such as those involving fraudulent transactions. In such cases, precision, recall, and F1-score may provide a better indication of how well the models are performing. Based on our analysis of the different models and techniques, we found that the Extra Trees Classifier performed the best in achieving a high F1-score of 0.84. This model demonstrated a high level of precision and recall, indicating that it was able to accurately identify fraudulent transactions while minimizing false positives. Other models, such as the Decision Tree Classifier, Random Forest Classifier, Light Gradient Boosting Machine Classifier, AdaBoost Classifier, and Gradient Boosting Classifier, also performed well, achieving F1-scores of 0.83. While the MLP Classifier and K Neighbors Classifier had lower F1-scores of 0.48 and 0.50, respectively, they may still be useful in specific situations.

Undersampling Dataset:

We employed undersampling using the Random Sampler Technique to address the imbalanced data issue by reducing the number of samples from the majority class to match the number of samples in the minority class. This technique randomly selects samples from the majority class to remove and balance the dataset. This method can result in a loss of information, but it can be an effective way to balance the dataset when the class imbalance is severe.







Original - Imbalance dataset **Undersampled Dataset**

Below table shows the performance of various machine learning models for detecting fraudulent transaction using Undersampling technique.

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Hyperparameters**
Decision Tree Classifier	0.92	0.76	0.83	0.78	‘max_depth’: 4
MLP Classifier	0.90	0.72	0.81	0.75	‘hidden_layer_sizes’=(3), (20, 10, 5)
K Neighbors Classifier	0.90	0.72	0.79	0.75	n_neighbors: 9
Random Forest Classifier	0.91	0.75	0.82	0.78	'max_depth':8, 'n_estimators': 100
Light Gradient Boosting Machine Classifier	0.92	0.76	0.83	0.79	'learning_rate':0.01, 'max_depth':5, 'num_leaves': 5

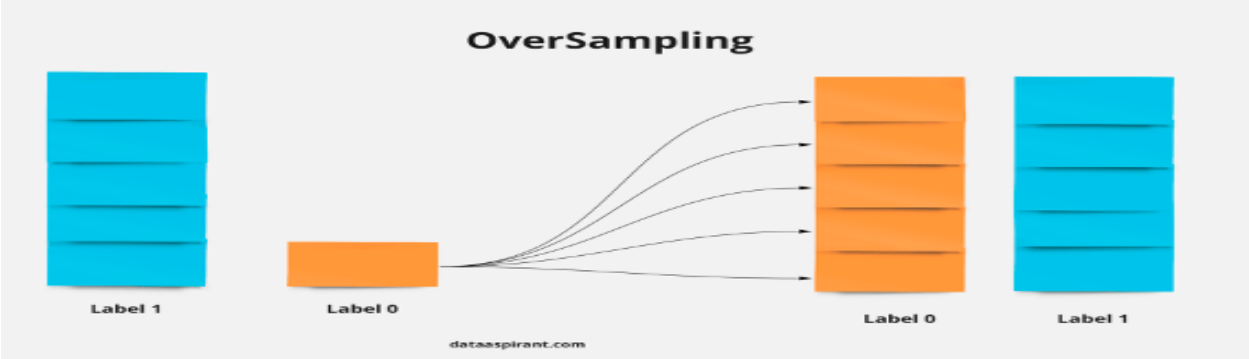
AdaBoost Classifier	0.91	0.75	0.82	0.78	'learning_rate':0.01, 'n_estimators': 50
GradientBoosting Classifier	0.92	0.76	0.83	0.78	'learning_rate':0.01, 'max_depth':4, 'n_estimators': 100
Logistic Regression	0.92	0.76	0.83	0.79	'C': 0.1, 'class_weight': {0: 2, 1: 1}, 'penalty': 'l1'
XGBoost Classifier	0.92	0.76	0.83	0.79	'colsample_bytree':0.6, 'learning_rate':0.01, 'max_depth':7, 'subsample': 0.6
Extra Trees Classifier	0.92	0.75	0.83	0.78	'max_depth':20, 'min_samples_leaf':2, 'min_samples_split':2, 'n_estimators': 100

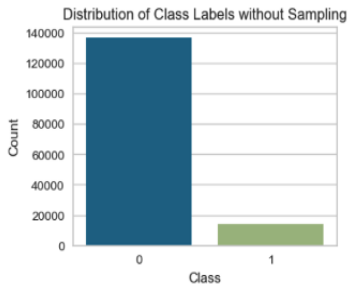
Table 2: Models performance on Undersampled Dataset

Based on the results of the models, we can see that the Extra Trees Classifier had the best F1-score of 0.78 and performs the highest recall. The best hyperparameters for this model include a maximum depth of 20, a minimum number of samples per leaf of 2, a minimum number of samples required to split an internal node of 2, and 100 estimators, followed closely by the Decision Tree Classifier and the Light Gradient Boosting Machine Classifier with an F1-score of 0.78 and 0.79, respectively.

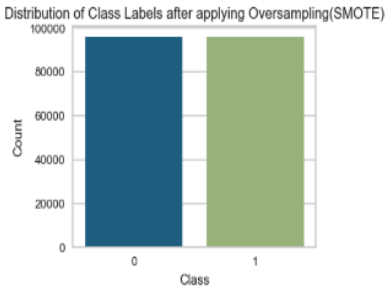
Oversampling Dataset:

We utilized a common oversampling technique called SMOTE (Synthetic Minority Over-sampling Technique) to balance the data. SMOTE creates more synthetic samples of the minority class that are similar to the existing minority class. This process is repeated until the classes are balanced. Another oversampling technique we employed was the Adaptive Synthetic (ADASYN) algorithm. This method is similar to SMOTE but generates a different number of samples depending on an estimate of the local distribution of the class to be oversampled. These techniques can help improve the performance of the model by providing it with a balanced dataset to train on.

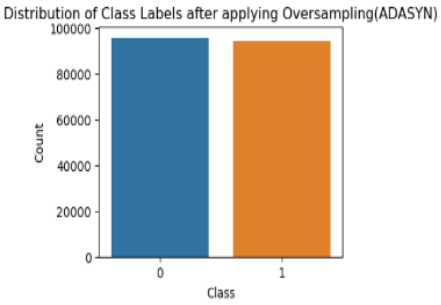




Imbalance Dataset



Oversampling-SMOTE



Oversampling-ADASYN

SMOTE:

Scores and Hyperparameters on Oversampled (SMOTE) Dataset:

Model	Accuracy	Precision	Recall	F1-Score	Best Hyperparameters
Decision Tree Classifier	0.91	0.74	0.77	0.75	'max_depth': 100
MLP Classifier	0.96	0.97	0.77	0.84	'hidden_layer_sizes'=(3), (20, 10, 5)
K Neighbors Classifier	0.81	0.63	0.75	0.65	n_neighbors: 3
Random Forest Classifier	0.92	0.76	0.82	0.79	'max_depth':8, 'n_estimators': 200
Light Gradient Boosting Machine Classifier	0.95	0.94	0.77	0.83	'learning_rate':1, 'max_depth':5, 'num_leaves': 10
AdaBoost Classifier	0.95	0.88	0.78	0.82	'learning_rate':1, 'n_estimators': 100
GradientBoosting Classifier	0.95	0.89	0.78	0.82	'learning_rate':1, 'max_depth':6, 'n_estimators': 100

Logistic Regression	0.96	0.97	0.77	0.84	'C': 100, 'class_weight': 'balanced', 'penalty': 'l1'
XGBoost Classifier	0.95	0.91	0.78	0.83	'colsample_bytree': 0.8, 'learning_rate':0.1, 'max_depth':7, 'subsample': 0.6
Extra Trees Classifier	0.94	0.82	0.80	0.81	'max_depth':20, 'min_samples_leaf':1, 'min_samples_split':5, 'n_estimators': 50

Table 3:Models performance on Oversampled (SMOTE) Dataset

After applying SMOTE oversampling technique to balance the data, we observe significant improvement in the precision and recall scores for most of the models. The MLP Classifier and Logistic Regression models showed the highest precision, recall, and F1-score scores across all models with the SMOTE dataset. This indicates that these models were able to identify more true positives and reduce the number of false negatives, which is important for identifying fraudulent transactions. However, the K Neighbors Classifier model showed a significant decrease in performance on the SMOTE dataset, particularly in terms of precision and F1-score. This suggests that oversampling techniques may not always be effective in improving the performance of all models.

ADASYN:

Scores and Hyperparameters on Oversampled (ADASYN) Dataset:

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Hyperparamaters**
Decision Tree Classifier	0.91	0.73	0.77	0.75	'max_depth': 50
MLP Classifier	0.96	0.96	0.77	0.84	'hidden_layer_sizes'=(3), (20, 10, 5)
K Neighbors Classifier	0.77	0.61	0.74	0.61	n_neighbors: 3
Random Forest Classifier	0.91	0.75	0.82	0.78	'max_depth':8, 'n_estimators': 100
Light Gradient Boosting Machine Classifier	0.95	0.94	0.77	0.83	'learning_rate':1, 'max_depth':10, 'num_leaves': 10

AdaBoost Classifier	0.95	0.88	0.78	0.82	'learning_rate':1, 'n_estimators': 100
GradientBoosting Classifier	0.95	0.90	0.78	0.83	'learning_rate':1.0, 'max_depth':4, 'n_estimators': 100
Logistic Regression	0.96	0.97	0.77	0.84	'C': 100, 'class_weight': {0: 2, 1: 1}, 'penalty': '12'
XGBoost Classifier	0.93	0.81	0.81	0.81	'colsample_bytree':0.8, 'learning_rate':0.1, 'max_depth':5, 'subsample': 0.6
Extra Trees Classifier	0.93	0.78	0.81	0.87	'max_depth':20, 'min_samples_leaf':1, 'min_samples_split':2, 'n_estimators': 100

Table 4: Models performance on Oversampled (ADASYN) Dataset

Based on the precision, recall, and F1-score metrics, we can conclude that the best performing models on the ADASYN oversampled dataset are: MLP Classifier with a precision of 0.96, recall of 0.77, and F1-score of 0.84Light Gradient Boosting Machine Classifier with a precision of 0.94, recall of 0.77, and F1-score of 0.83.Logistic Regression with a precision of 0.97, recall of 0.77, and F1-score of 0.84.

Comparision:

Based on the evaluation of three datasets (original, undersampled, and oversampled), the Extra Trees Classifier appears to be the best-performing model with F1-scores ranging from 0.78 to 0.84 across all three datasets. However, the Random Forest Classifier and the Light Gradient Boosting Machine Classifier also performed well with F1-scores ranging from 0.78 to 0.83. It is important to note that the performance of each model varied depending on the dataset used. For example, the MLP Classifier did not perform well on the original dataset with an F1-score of 0.48, but performed well on the undersampled or oversampled datasets with F1-scores of 0.75 and 0.84, respectively. Therefore, the choice of dataset can have a significant impact on model performance. When considering the F1-score, which is a more appropriate metric for imbalanced datasets, the oversampled datasets using SMOTE and ADASYN performed better than the original and undersampled datasets. Therefore, it can be concluded that oversampling techniques are effective for improving model performance on imbalanced datasets.

Technical:

Data Preparation:

In order to prepare our dataset for analysis, we first had to clean and format the raw data. This involved removing duplicates, handling missing values, and converting categorical variables to numerical ones. We also conducted some exploratory data analysis to identify any potential outliers or anomalies that needed to be addressed. This process ensured that our dataset was suitable for analysis and that we could draw meaningful conclusions from our results.

Analysis:

For our analysis, we explored multiple machine learning algorithms to determine the best fit for our dataset and goals. After trying out several models, we found that both Extra Trees and Random Forest algorithms were able to effectively capture the complex relationships between the features and the outcome variable. We chose these algorithms due to their ability to handle missing data and not requiring assumptions about the underlying distribution of the data. Random Forest algorithm and Extra Trees is well-suited for handling high-dimensional data, reducing overfitting, and improving the generalizability of the results through its ensemble approach. Therefore, we recommend these for fraud detection as it gave the best results among the machine learning algorithms that we tried in our analysis.

Analysis process:

In order to conduct our analysis, we initially split the dataset into training and testing sets to evaluate model performance. We also experimented with different hyperparameters to optimize our model's performance and prevent overfitting. However, we faced several challenges, including imbalanced classes and noisy data, which required us to try various data augmentation techniques and ensemble methods. As the dataset was large, oversampling and hyperparameter tuning took a considerable amount of time to run, which prolonged the overall analysis process. Despite these challenges, we were able to achieve a high level of predictive accuracy, but we acknowledge that alternative approaches such as incorporating anomaly detection methods such as isolation forests or one-class support vector machines alongside traditional classification techniques could yield similar or even better results in detecting fraud.

The changes we made from our proposal include adding a third dataset containing blacklisted IP addresses and incorporating an additional oversampling technique, ADASYN and ensemble models. Other than that, we followed the plan outlined in our proposal. Also, We followed the "Novel data" approach and created an interesting dataset by combining multiple datasets and manual collection to identify fraud. Our findings showed effective machine learning models for detecting fraudulent transactions. Stakeholders, such as financial institutions and retailers, can benefit from our insights and techniques, as well as potential cost savings through accurate fraud detection. We also provided recommendations for improving fraud detection techniques.