Fraud Detection in E-commerce Using Machine Learning

Satyanarayana Vinay Achanta (A02395874)

Introduction:

The growth of e-commerce websites like Amazon has led to a huge increase in fraudulent transactions, causing financial losses and negatively impacting customer satisfaction. In the past, fraudulent transaction detection in the e-commerce sector was manual and prone to human error, making it time-consuming and expensive. With advancements in machine learning, it is now possible to automate the process of identifying potentially fraudulent transactions accurately. Our project aims to create a machine learning-based fraud detection system by analyzing transaction data obtained from an online retailer of electronic goods.

To achieve our goal of providing an efficient and effective fraud detection system for the e-commerce sector, we will utilize machine learning algorithms like decision trees, logistic regression, K-Neighbors, and ensemble techniques like Gradient boosting, Adaboost, XGBoost, and Random Forests. We will use transaction data that includes essential columns such as user ID, signup time, purchase time, purchase value, device ID, source, browser, sex, age, IP address, and class, along with IP data and blacklisted IP data to enhance the accuracy of the fraud detection system.

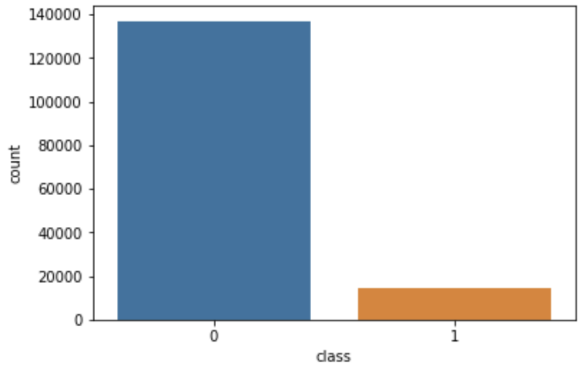
Our analysis of these datasets will identify patterns indicating fraudulent activity, automating the identification of potentially fraudulent transactions. These results will enable businesses to reduce expenses and enhance the overall customer experience by providing insights into fraudulent transaction patterns in the e-commerce sector. By implementing such a system, e-commerce companies can significantly reduce financial losses and improve the security and trustworthiness of online transactions for customers.

Presentation Slides Link, GitHub Link

Dataset:

In this project, we are using a Novel Data approach by combining multiple datasets. We have used a dataset that contains transactional data of an online retailer of electronic goods. The dataset 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. The first two datasets were obtained from Kaggle, while the third dataset was taken from MYIP.MS website that provides information on real-time blacklisted IPs. The dataset contains 151,112 rows, with 14,151 transaction details classified as fraud and the rest classified as nonfraud.

The dataset is suitable for fraud detection, and its various attributes, such as customer behavior and transaction patterns, make it an important asset for developing an accurate machine learning-based fraud detection system. In addition, the dataset's inclusion of a list of country names with IP address bounds is crucial for mapping IP addresses to their respective countries, which is a crucial step in identifying fraudulent activities that often involve transactions from multiple countries. We have performed several preprocessing steps, including data cleaning, one-hot encoding, label encoding, standardization, and IP address conversion to quad-dot format, to ensure that the dataset is ready for analysis. A potential problem we may face is imbalanced data, with few positive cases (fraud), leading to higher false positive rates. To address this, we'll use techniques like oversampling (SMOTE, ADASYN), and undersampling to balance the dataset.



	user_id	signup_time	purchase_time	purchase_value	device_id	source	browser	sex	age	ip_address	class
0	22058	2015-02-24 22:55:49	2015-04-18 02:47:11	34	QVPSPJUOCKZAR	SEO	Chrome	M	39	7.327584e+08	0
1	333320	2015-06-07 20:39:50	2015-06-08 01:38:54	16	EOGFQPIZPYXFZ	Ads	Chrome	F	53	3.503114e+08	0

	lower_bound_ip_address	upper_bound_ip_address	country
0	16777216.0	16777471	Australia
1	16777472.0	16777727	China

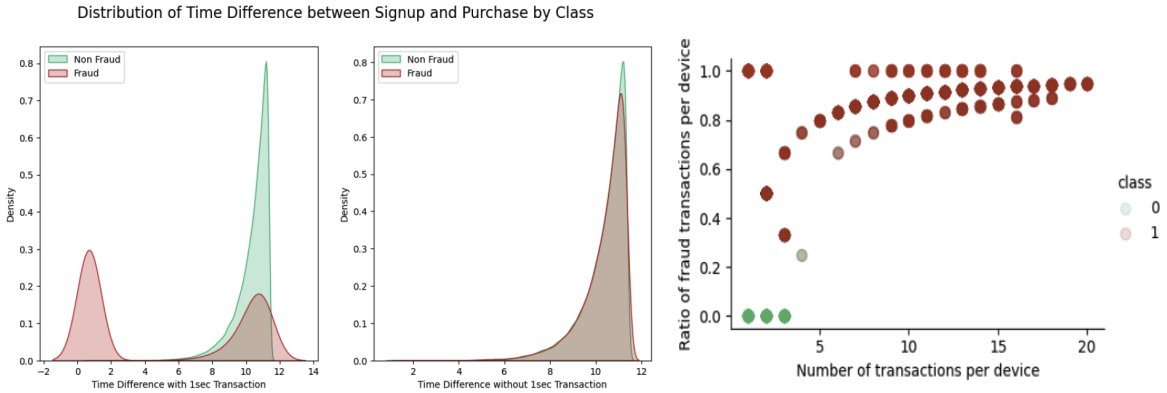
	Blacklisted IP_Address
0	1.0.136.29
1	1.0.136.215

Analysis Technique:

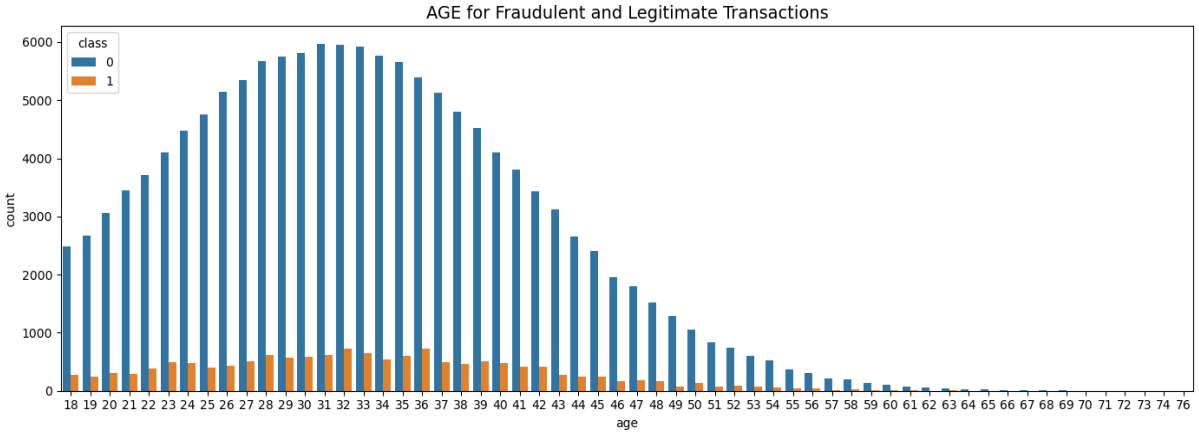
The analysis technique used in this project involves feature engineering, data visualization, and machine learning modeling techniques to detect fraudulent transactions. Feature engineering is a process of creating new features from existing ones to improve the accuracy of machine learning models. The feature engineering process involved converting activity-based features like signup time and purchase time into date format and dividing them into individual hours, seconds, minutes, days, and dates to find any patterns and also help out models in improving scores. New features such as the time difference between signup time and purchase time, purchase_week, purchase_year, the transaction is from a unique device id or not, the transaction is from a unique IP address or not, and total

purchase from one device were also added. These new features helped to identify patterns in the data that could be used to detect fraudulent transactions.

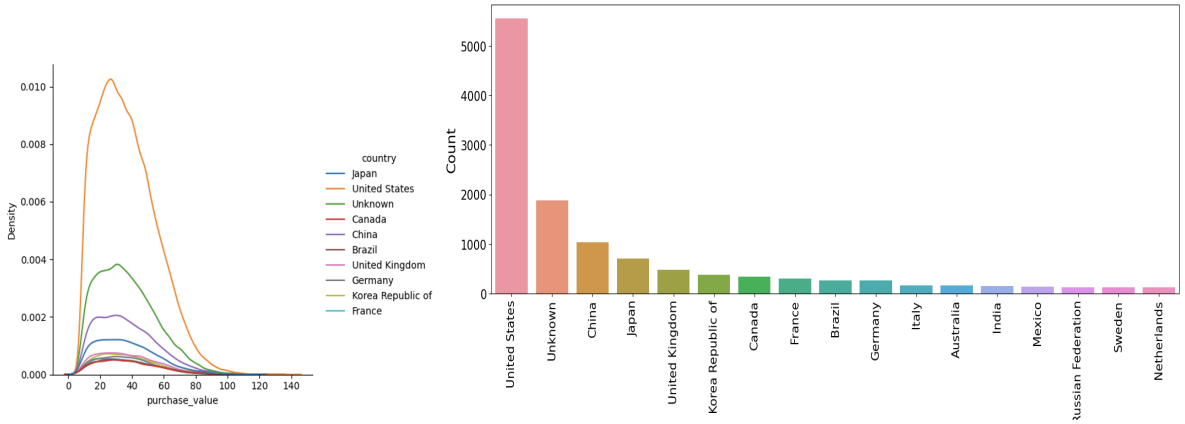
Analyzing patterns in the data, we were able to identify the patterns, such as if the time difference between sign up and purchase of one user is equal to one second then the transaction is definitely a fraud transaction. Scammers can use automated scripts and bots to achieve creating new user IDs and make transactions within seconds using a bot and automated code scripts. Additionally, if the transaction has multiple user IDs and happens from the same IP address, it tends to be more fraudulent. Similarly, if the transaction has multiple user IDs and happens from the same device ID, it also tends to be more fraudulent. These new features were beneficial in improving our analysis and helped us to identify potentially fraudulent transactions.



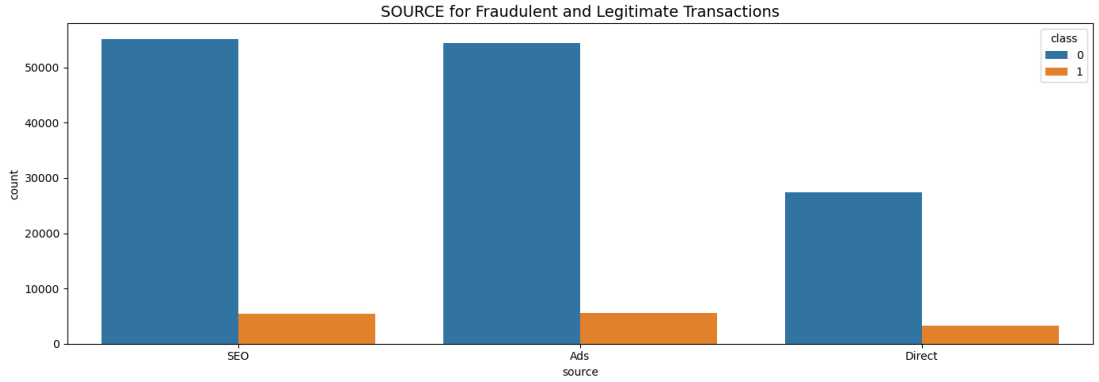
In addition to the trends mentioned, the data visualization plots also revealed some other interesting insights about the dataset. The plot analyzing the trend of age and fraud count showed that the highest number of fraud cases occurred in the age group of 22 to 42 years old. One possible reason for this trend is that individuals in this age group may be more prone to financial pressures and may be more likely to participate in fraudulent activities to reduce these pressures.



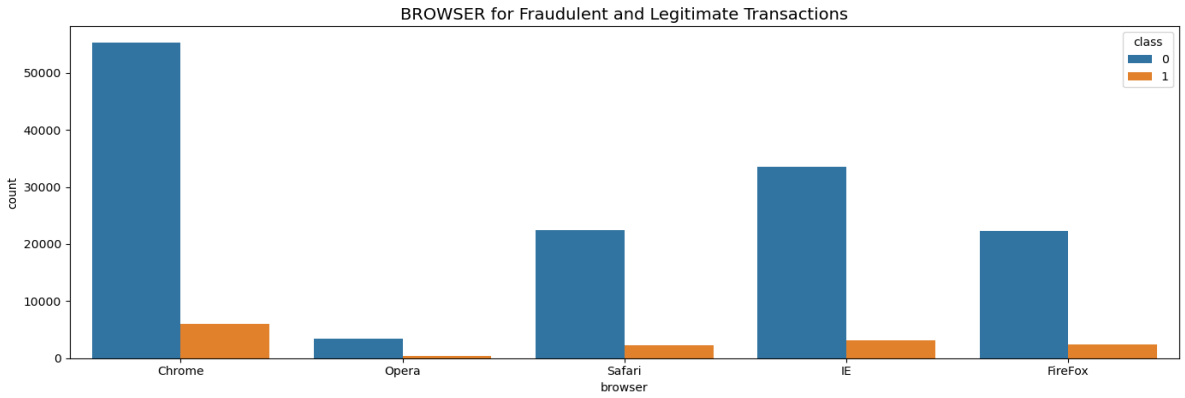
The countries with the most fraud plot showed that the United States had the highest number of fraud cases, followed by China and Japan. This reason could be these countries have a larger population and a higher number of transactions, making them more attractive targets for fraudsters. It could also be a result of the different levels of security measures implemented in these countries.



The plot showing fraudulent activities through SEO (Search Engine Optimization) and ads showed that more fraud occurred through ads compared to SEO. This could be due to the fact that ads provide a more direct and immediate means of communication with potential victims, and fraudsters can use ads to easily attract people into fraudulent schemes. Also, SEO requires more effort and patience to achieve results for fraudsters, making it a less attractive method for scammers.



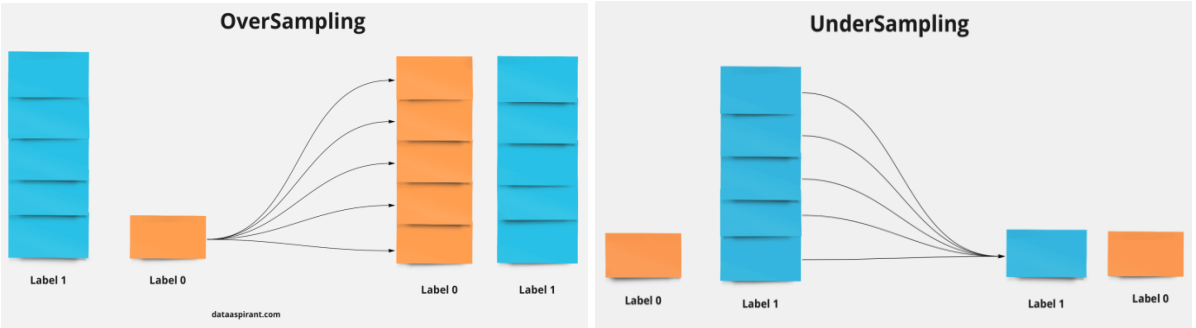
The plot analyzing browsers and sources for fraudulent transactions showed that the majority of fraud occurred through the Chrome browser as it is the most commonly used browser, making it an attractive target for scammers. These insights can help fraud-detecting institutions and government agencies better understand fraudsters and the ways they make fraud happen and develop strategies to prevent and detect these activities.

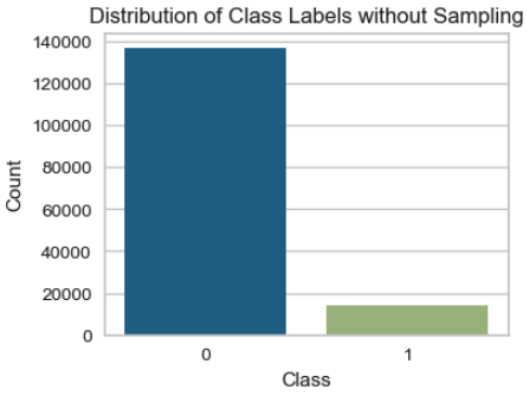


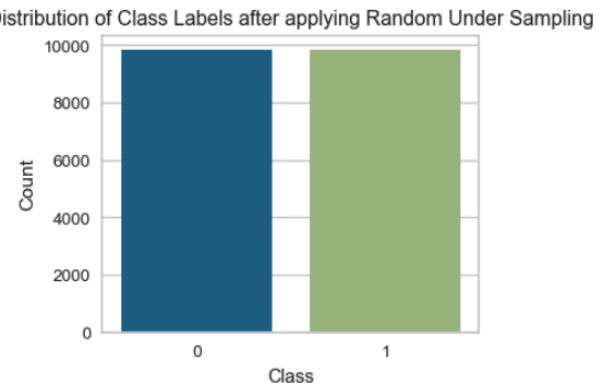
To build the machine learning models, various classifiers such as the decision tree classifier, MLP classifier, K-Neighbors classifier, Random Forest Classifier, Logistic regression classifier, and ensemble classifiers were used. However, since the dataset was imbalanced, with the non-fraud class having more values (majority class) and the fraud class having fewer values (minority class), the machine learning model's performance was moderate. To improve the performance of the models on the imbalanced dataset, various sampling techniques were used, such as oversampling SMOTE and ADASYN, and undersampling.

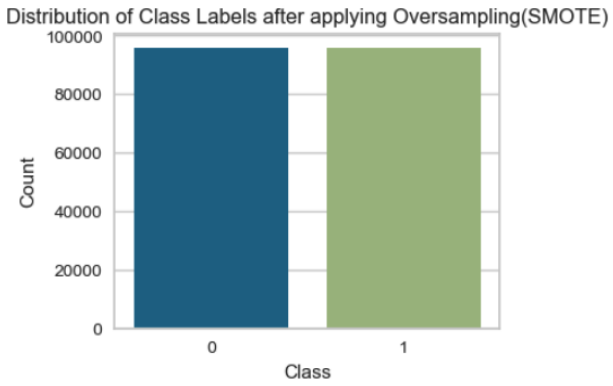
Oversampling techniques like SMOTE (Synthetic Minority Over-sampling Technique) and ADASYN (Adaptive Synthetic Sampling) are used to balance an imbalanced dataset. SMOTE works by creating synthetic samples of the minority class by using the k-nearest neighbors algorithm. The algorithm selects k nearest neighbors of the minority class and creates synthetic samples by interpolating between the minority class sample and its nearest neighbors. This technique increases the number of samples in the minority class, making it more balanced with the majority class. ADASYN is an extension of SMOTE that adaptively generates synthetic samples by considering the density distribution of the minority class. This technique generates more synthetic samples for the minority class samples that are difficult to learn by the model.

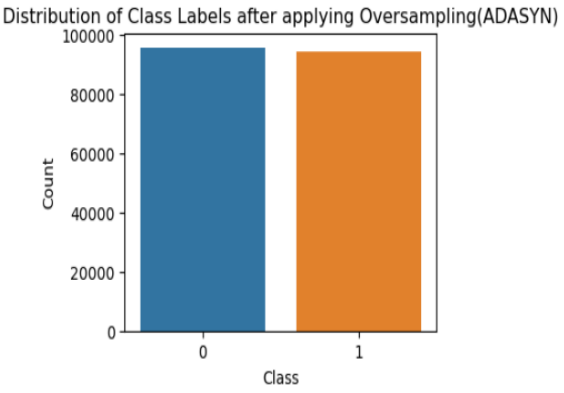
Undersampling techniques involve removing samples from the majority class to balance the dataset. This technique can lead to a loss of information and may result in underfitting. Random undersampling selects samples from the majority class randomly and removes them until the classes are balanced.











By balancing the dataset, the machine learning model's performance was improved, and the best hyperparameters for all the models were obtained using grid searchCV. Finally, the metrics used for evaluation are F1-score, precision, recall, and accuracy (not a great metric for this fraud detection project) as well as a confusion matrix. Overall, the analysis technique used in this project was suitable for the purpose of detecting fraudulent transactions, and the results obtained were promising.

Results:

The below tables show the performance of different classification models on the original, undersampled, and oversampled using SMOTE and oversampled using ADASYN datasets. We will focus on the recall and F1-score for each model since we want to optimize the detection of fraudulent transactions while minimizing the number of false negatives.

On the original dataset, the model with the best recall and F1 scores is the Extra Tree Classifier with a precision of 0.96, recall of 0.77, and an F1 score of 0.84. Other models such as Decision Tree, Random Forest, Light Gradient Boosting Machine, AdaBoost, Gradient Boosting, Logistic Regression, and XGBoost classifiers have also performed better. with a recall of 0.77 and an F1 score of 0.83.

On the undersampled dataset, the models with the best recall and F1 scores are the Light Gradient Boosting Machine, Logistic Regression, and XGBoost with a recall of 0.83 or higher

and an F1 score of 0.78 or higher. It's important to note that while the undersampling approach may improve the performance of the model in identifying fraud cases, it may not be the best choice in all situations and changes according to the requirement. This is because undersampling reduces the size of the majority class, resulting in a loss of information about non-fraudulent transactions. This may cause the precision of the model to decrease, as it may incorrectly identify non-fraudulent transactions as fraudulent. It's important to choose the trade-offs between precision and recall when choosing a sampling strategy.

On the oversampled dataset with SMOTE and ADASYN, the model with the best recall and F1 scores are the Extra Tree Classifier with a precision of 0.93, recall of 0.78, and an F1 score of 0.87 on the ADASYN dataset. Other models such as MLP Classifier, Light Gradient Boosting Machine, Gradient Boosting Classifier, Logistic Regression, and XGBoost classifiers have also performed better. with a recall of 0.77 and an F1 score of 0.83 or higher.

Based on these results, we can conclude that the Extra Trees Classifier is the most consistent model on our datasets with the highest F1 score of 0.87. Other than this, the MLP Classifier, Light Gradient Boosting Machine, Gradient Boosting Classifier, Logistic Regression, and XGBoost classifiers all performed similarly well on the original dataset with an F1-score of 0.83, the Extra Trees Classifier performed better than all of these classifiers on the undersampled and oversampled datasets.

These results show that the choice of dataset and appropriate sampling technique can have a great impact on the performance of classification models. It is important to evaluate the performance of models on multiple datasets and choose the most best one for the given problem. An alternative approach includes additional feature engineering and hyperparameter tuning to improve the performance of the classifiers.

Scores on Original Dataset:

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

Scores on Undersampled Dataset:

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
Gradient Boosting 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

Scores 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
Gradient Boosting 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

Scores on Oversampled (ADASYN) Dataset:

Model	Accuracy	Precision	Recall	F1-Score	Best Hyperparameters
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
Gradient Boosting 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

Conclusion:

The project aimed to develop a machine learning-based fraud detection system for the e-commerce sector, utilizing various algorithms and techniques such as decision trees, logistic regression, K-Neighbors, and ensemble techniques like Gradient boosting, Adaboost, XGBoost, and Random Forests. The analysis revealed patterns of fraudulent activity, and data visualization shows interesting insights like the highest number of fraud cases occurred in the age group of 22 to 42 years old, and the United States, China, and Japan having the highest number of fraud cases. Feature engineering techniques were used to improve the performance of machine learning models, and various techniques were used to solve the problem of imbalanced data. Ensemble models were found to give better results than individual models, with an F1-score of 87%. The project's machine learning-based fraud detection system can provide valuable insights to e-commerce companies about fraudulent transaction patterns in the e-commerce sector, significantly reducing financial losses, and improving the security and trustworthiness of online transactions for customers.

In addition, the Extra Tree Classifier algorithm was also used in the project and was found to have the best performance when combined with the ADASYN oversampling technique among all the other algorithms. The Extra Tree Classifier is a type of decision tree algorithm that builds multiple randomized decision trees and combines their predictions to produce the final output. This algorithm has the advantage of reducing overfitting and can handle noisy data well. Therefore, we would highly recommend using the Extra Tree Classifier algorithm for fraud detection in the e-commerce sector.

We made some changes to our proposal, including the addition of a third dataset consisting of blacklisted IP addresses and the implementation of an additional oversampling technique known as ADASYN. We incorporated various ensemble models, including the Adaboost Classifier, Gradient Boost, XG Boost, and Extra Tree Classifiers, to further enhance the performance of our fraud detection system.