University of Hertfordshire UH

School of Physics, Engineering and Computer Science

MSc Data Science Project
7PAM2002-0901-2024
Department of Physics, Astronomy and Mathematics

DATA SCIENCE FINAL PROJECT REPORT

Project Title:

Detecting Online Payment Fraud: A Machine Learning Approach for Predictive Analytics

Student Name and SRN:

Devi Nandana Sandhya, 22014197

Supervisor: Dr Carolyn Devereux

Date Submitted: 6 January 2025

Word Count: 7,487

GitHub address:
https://github.com/devinandana01/DeviProject/blob/main/Devi_v18.ipynb

DECLARATION STATEMENT

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Data Science at the University of Hertfordshire.

I have read the guidance to students on academic integrity, misconduct and plagiarism information at Assessment Offences and Academic Misconduct and understand the University process of dealing with suspected cases of academic misconduct and the possible penalties, which could include failing the project module or course.

I certify that the work submitted is my own and that any material derived or quoted from published or unpublished work of other persons has been duly acknowledged. (Ref. UPR AS/C/6.1, section 7 and UPR AS/C/5, section 3.6).

I have not used chatGPT, or any other generative AI tool, to write the report or code (other than were declared or referenced).

I did not use human participants or undertake a survey in my MSc Project.

I hereby give permission for the report to be made available on module websites provided the source is acknowledged.

Student SRN number: 22014197

Student Name printed: Devi Nandana Sandhya

Student signature:

UNIVERSITY OF HERTFORDSHIRE

SCHOOL OF PHYSICS, ENGINEERING AND COMPUTER SCIENCE

Acknowledgements

I would like to express my deepest appreciation to everyone who helped and directed me through my final project, 'Detecting Online Payment Fraud: A Machine Learning Approach for Predictive Analytics'.

First and foremost, I am eternally grateful to the Almighty for the gifts and strength that have been vital in my path.

I am extremely grateful to my project supervisor, Dr Carolyn Devereux, for her helpful direction, persistent support, and insightful input. Her knowledge and efforts have helped shape the project's direction and improve its quality.

I would also want to thank the University of Hertfordshire faculty members for their lessons and guidance, which have helped me build a solid foundation in machine learning and data analysis.

I am grateful to my friends and peers for openly sharing their views and ideas, which helped to enrich the project's growth by providing varied viewpoints.

Finally, I would like to recognise the numerous online resources, research papers, and open-source tools that have helped me extend my understanding and apply various machine learning methods.

This project has been an exciting experience, made possible by the combined efforts of these persons and resources. Thank you everybody for your assistance in making this project a success.

Abstract

This project result in the detection mechanism for online payment fraud, therefore being effective and efficient. Based on this thought, the machine learning-based predictive model will be built using the IEEE Dataport provided "Financial Dataset". In drawing meaningful insights into such datasets or optimizing the model's performance, tasks such as data preparation, exploratory data analysis, and feature engineering need to be performed. To classify fraudulent transactions, the machine learning algorithms used here are Logistic Regression, Voting Classifier, K-nearest neighbors(KNN) and XGBoost. The performance of the model was tested using various metrics such as Receiver Operating Characteristic (ROC) curve and precision-recall analysis. The designed final model has a real-time-deployment capability. This scalable model provides an efficient solution to identify frauds and malicious activities, and therefore, secure online payment systems with reliability.

Table of Contents

Chapter 1 Introduction

The online payment systems have become ubiquitous in international business, thus affording greater comfort to the merchant and the buyer. Yet the ease has ushered an increase in fraud with serious risks threatening financial institutions, businesses, and consumers. There is continuous hacking of weaknesses found in such payment systems and other sophisticated ways by cybercrime hackers to facilitate such fraudulent transactions. Continuing from the already high level of online transactions, methods traditionally used in fraud detection, which rely on rules and heuristics, have proved insufficient in uncovering novel and changing fraud patterns. This creates an emergency situation because one needs methods that are much more sophisticated and driven by data to detect fraudulent activity in real time.

Actually, machine learning with its capability to deal with large data and its intrinsic ability to identify hidden patterns in them is promising, and predictive models from historical transaction data are more likely to differentiate between legitimate and fraudulent transactions and help with detection in more timely and accurate ways. These models are capable of evolution and adaptation and thereby provide a significant improvement over traditional methods that rely on static rules.

The study intends to design a framework for online payment fraud detection through machine learning techniques using the "Financial Fraud Detection Dataset" acquired from IEEE DataPort. It provides transactional data from multiple financial systems and their features such as transaction amount, balances, type of transaction, and identifiers related to the customers, which will be highly helpful in the authentication of transactions. The main aim of this study is to identify the effectiveness of various machine learning algorithms in fraud transaction prediction and to evaluate the performance of these models based on accuracy, precision, recall, and F1-score.

The paper addresses the following key objectives:

• **Data Preparation** is the process used to explore, clean, and even preprocess datasets, before training up a machine learning model.
• **Model Development:** The best model for fraud detection was chosen by selecting the best machine learning algorithm between Logistic Regression, Voting Classifier, K-nearest neighbors(KNN) and XGBoost.
•**Model Evaluation:** Various metrics are used in order to test the performance of the developed models, and empirical results determine which model will best perform.
• **Design:** The architecture of the implementation of the adopted model in the real-time online fraud detection system.

With these intentions, it aims to develop the practical aspect of machine learning in online fraud detection of electronic payments and subsequently contribute to advancing security measures through digital financial transactions. It relies on the force of machine learning techniques to drive improvements in both accuracy, efficiency, and scale for fraud-detection systems such that a relatively safer digital environment is created between businesses and customers.

Project Aim

The objective of this project is to design an evidence-gathering framework for the detection of fraudulent online payments using transaction data. This will analyse several machine learning models and find out which of them could possibly have the best performance in terms of accurately predicting fraudulent transactions. This will enhance real-time fraud detection systems, thus improving security in online payment environments.

Objective

The project's goal is to create a machine learning system that can effectively detect fraudulent internet payments. It all begins with data preparation, which includes exploratory data analysis (EDA), cleaning, and preprocessing to resolve missing values, outliers, and scale features for improved model performance. Several algorithms, including Logistic Regression, Voting Classifier, KNN, and XGBoost, are trained on a validation set to find the optimal model for fraud detection. The model's performance is measured by accuracy, precision, recall, F1-score, and ROC-AUC. The finished model will be integrated into a real-time fraud detection system, improving accuracy, efficiency, and scalability for identifying online fraud.

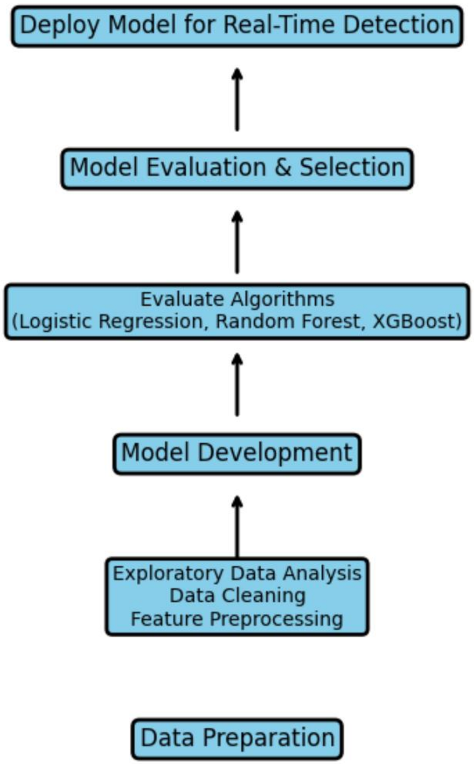

Deploy Model for Real-Time Detection
Model Evaluation & Selection
Evaluate Algorithms
(Logistic Regression, Random Forest, XGBoost)
Model Development
Exploratory Data Analysis
Data Cleaning
Feature Preprocessing
Data Preparation

Figure 1:"Workflow for Fraud Detection Model Development and Deployment

Scheme of Project Work

The project workflow can be divided into the following stages, each addressing a critical aspect of online payment fraud detection using machine learning:

1. Data Acquisition and Understanding

The workflow of the project can be divided into the following stages, which tackle the most important aspect of machine learning-based online payment fraud detection:

Objective: Acquire a quality dataset for training and testing machine learning models.

- **Key Actions:**

Source the "Financial Dataset" from IEEE Dataport.

- Understand the structure of the dataset, attributes, and importance of each feature.
- Determine the target variables and possible predictors to detect fraud.
- Output: Cleaned and structured dataset ready for analysis.

2. Exploratory Data Analysis (EDA)

•**Objective:** To analyze the data distributions and to discover patterns, correlations, and anomalies.

- **Key Actions:**
- Statistical analysis and data visualization of feature distributions and outliers.
- Class imbalance and other dataset challenges.
- Generate insights for feature engineering.
- Output: Summary report on characteristics of the data.

3. Data Preprocessing and Feature Engineering

- **Objective**: Make data ready so that a machine learning model could perform at its best.
- **Key Actions:**
- Handling missing values, normalization, encoding categorical variables
- Development of new features; for example, transaction velocity, customer activity history
- Class imbalance techniques: oversampling, undersampling or SMOTE to be used
- Output: Feature-rich preprocessed dataset

4.**Model Development**

- **Objective**: To train/build the fraud detection machine learning models
- **Key Actions:**
- Simple models such as Logistic Regression and Support Vector Machines and also Decision Trees
- Advanced models such as Random Forest, XGBoost, and Neural Networks
- Divide the data into training and testing sets to check out the performance of the model

- Output: A list of trained models along with the relevant performance metrics

5.Model Tuning and Comparison

- **Objective**: Task: Fine-tune the model for maximum accuracy and validate the reliability.
- **Key Steps:**
- Hyperparameter tuning using Grid Search or Random Search
- Evaluation metric such as accuracy, precision, recall, F1-score and ROC-AUC
- Compare the model so that the algorithm that performs the best can be advised.
- Output: Best fraud detection model by all evaluation reports.

6. Real-Time Deployment

- **Objective**: Take the best performing algorithm and deploy the same in the simulated real time environment.
- **Key activities:**
- Integrate the best performing model in fraud detection framework
- Simulate online transactions in order to assess its capability to detect
- Optimizing runtime performance for scalability
- Output: Deployed fraud detection system, ready to be used in the field.

Chapter 2 Literature Review

Online payment fraud has increased proportionally with digital transactions, which is why detection of fraud becomes critical for maintaining financial security. The traditional system of fraud detection usually depends on the rule-based methodologies, which lack the ability to capture complexity and variability found in real-world financial transactions. However, promising approaches for machine learning have demonstrated the potential of identifying possible fraud by analyzing the patterns across a large dataset.

The paper titled Fraud Detection in Online Payments using Machine Learning Techniques by Siddaiah, Anjaneyulu, Haritha, and Ramesh (Siddaiah, et al., 2023) is a comprehensive study of the application of machine learning in fraud detection for online payments, a growing need in today's digital economy. However, though online transactions are convenient and fast, they pose significant risks to users, such as fraud and privacy breaches. To counter these vulnerabilities, the researchers suggest a machine learning model that is specifically designed for the detection of risky transactions through a feature-engineered approach. Based on the XGBoost algorithm, which is an ensemble decision-tree-based method known for high efficiency and perfect accuracy, this model works. The paper highlights how XGBoost makes it possible for handling large datasets efficiently by combining multiple decision trees such that complex patterns in the transaction data may form, suggesting a potential fraud. Siddaiah et al. (2023) further argue that XGBoost, compared to other machine learning algorithms, is both faster and more accurate, thus suitable for real-time fraud detection applications. It is important in financial environments where detection of anomalies can prevent fraudulent activities from happening and improve the safety of online payment systems. By focusing on feature engineering and continuous data processing, the authors depict how this approach can make model stability and reliability even stronger, by contributing considerably to the advancement of machine learning applications in financial security.

The above paper, entitled Online Payment Fraud Detection Model Using Machine Learning Techniques by Almazroi, A.A., and Ayub, N. (ALMAZROI & AYUB, 2023), had conceptualized the advanced AI model called RXT-J specially made in combating the ever-growing problem of financial fraud. Published by IEEE Access, it answers how to process complicated real-time transactions with the incorporation of a ResNeXt-embedded Gated Recurrent Unit (GRU) model to integrate the AI techniques enhanced superb fraud detection. It used SMOTE for balancing the data and to tackle the data imbalance problem. The ensemble method that used autoencoders along with ResNet was chosen by the authors; they called this EARN. Using this, they could grasp the major patterns required in the data. Now, it was time for the classification process using the RXT model and fine-tuned using Jaya optimization algorithm to enhance its performance.

The authors test the model on three real-world financial transaction datasets and report a 10-18% improvement over traditional models in various metrics. This indicates that RXT-J is competitive in terms of both accuracy and computational efficiency. The robust architecture of the model improves detection accuracy while also providing resilience against interference from wireless communication, with the goal of strengthening data security, reliability, and availability against cyber threats in the financial sector.Almazroi and Ayub (2023) have pointed out that RXT-J is a significant advancement in the field, marking a step forward in securing financial transactions and enhancing operational efficiency.

The paper Fraud_Detection_ML: Machine Learning Based on Online Payment Fraud Detection by Maged Farouk, Nashwa Shaker Ragab, Diaa Salama, and Omnia Elrashidy, (Farouk, et al., 2024), proposes an efficient frame for the prediction of online payment fraud using machine learning algorithms. Looking closely into the work of the six different algorithms: KNN, Tree, Random Forest, Gradient Boosting, Support Vector Machine (SVM), Logistic Regression, Naive Bayes, AdaBoost, Neural Networks, and Stochastic Gradient Descent applied over three different datasets shows that their results indicate that the best algorithm is the gradient-boosting algorithm, with a 99.7% accuracy rate, which demonstrates that it is robust and can be used for the detection of fraudulent transactions in electronic transactions (Farouk et al., 2024). Such performance reveals that gradient boosting can be considered a strong option for e-commerce platforms that might prevent fraud by showing preemptive behavior. The depth of the current study emphasizes that the choice of algorithm is as important as detection, since it varies in efficacy among different algorithms.

It therefore has become clear to emphasize that gradient boosting algorithm performs surpassing peer benchmarks in accuracy measures and proves excellent performance at diversified testing, putting it up among the vital guarding assets to assure e-commerce transaction safety. Due to such analyses of complexness in the scenarios of fraud transactions online payments involve, data driven solution provision can provide contribution to a highlevel knowledge source relevant to such, and is actually a well-drafted robust safeguard mechanism towards their e-commerce system. Their efforts are crucial to stakeholders who are interested in increasing the security of online transactions during a time when cyber threats keep changing. The digital landscape keeps changing, and it has become one of the burning issues for both consumers and businesses as online payment fraud has gained momentum.

(Namani, et al., 2024), have proposed an integrated review of the detection system of online fraud in payments. This review has been titled as "Online Payment Fraud Detection: An Integrated Approach.". In that paper, the authors seem to share some techniques and tools by which they are currently classifying transaction data real time through the mechanisms of machine learning, pattern recognition, and anomaly detection. So, this study does portray how these systems play an important role in preventing financial loss and protecting the stakeholders from fraudulent activities by having a thorough analysis of the needs of these systems against the growing complexities of cyber threats (Namani et al., 2024). Again, the authors emphasize the requirement of secure online transactions to find, detect, and mitigate the theft, illegal transactions, or other fraudulent entities.

Their work highlights the need for an integrated approach to fraud detection in order to improve the security of online payment systems.This has led to lightened methodologies in the detection of fraud that Namani et al. (2024) The frameworks developed have contributed toward the knowledge arena, providing an avenue for stakeholder defense. This paper is a rich source of knowledge for understanding the intricacies involved in online payment fraud and mechanisms that can help in combating such fraud effectively. As technology is emerging at breakneck speed, online payment systems have become one of the fastest-growing sectors around the world.

In "Online Payment Fraud Detection Using Machine Learning," the authors (Venkatesh, 2024) talk about the current scenario of increasing online transaction fraud that has mushroomed with digital payment methods. The authors here also stress that improving online payment security is significant with the aid of machine learning models for the detection of frauds. They claim that such models can better and faster analyze large volumes of transactional data than the traditional manual inspection methods (Venkatesh et al., 2024). The paper discusses the advantages of online payments to consumers, such as convenience and time saving, but the risks involved in these transactions have been growing.

The authors apply the techniques of machine learning to produce a robust framework that is strong enough to effectively detect fraudulent activities so that both buyer and seller will be protected. This paper, therefore, adds to existing literature by demonstrating that machine learning works well in securing online transactions so that it should be an essential resource for the stakeholders who are interested in integrating advanced fraud detection solutions in their payment systems. This fast-evolving digital scenario demands advanced detection systems for the growing menace of online payment fraud to ensure secure financial transactions.

The existing literature, while taken together, points out that ML techniques contribute significantly to enhanced fraud detection capability. Researchers used a variety of algorithms, like XGBoost, gradient boosting, and advanced models RXT-J, and achieved higher precision and efficiency levels than the conventional rule-based system.These studies show feature engineering and preprocessing of data being critical to generate robust models in order to enable the analysis of complex transactional patterns and further facilitate the discovery of fraudulent behavior in time. The findings for the datasets are promising and establish a foundation through which solutions can counter the sophisticated attacks by fraudsters. It is only through the incorporation of machine learning techniques in online payment fraud detection that the consumer's trust can be protected and digital transactions secured. Innovation and adaptation, as per Farouk et al. (2024), Namani et al. (2024), and Venkatesh et al. (2024), are a must in dynamic cyber threats.

The findings of these studies promote the continuous evolution of detection methods, with real-time analytics and comprehensive frameworks capable of adapting to changing fraud patterns. As more financial transactions migrate to digital platforms, effective fraud detection systems will be critical to mitigating risk and ensuring that online payment systems are safe and reliable, leading to a secure e-commerce environment.

XGBoost would be a highly preferred choice of fraud detection on account of being highly accurate to pick out nuances in data for fraud detection in applications where accuracy with high recall in detecting fraudulent cases is the core objective.

Chapter 3 Dataset

This project makes use of IEEE Dataport "Financial Dataset," which simulates the realistic financial environment by combining the account profile, transaction generation, and predefined fraud scenarios. These elements allow one to produce minute, probabilistic data, which could closely replicate actual financial behavior, forming a strong foundation for training ML models regarding fraud prediction based on the attributes of transactions.

The IEEE dataset contains realistic account profiles, diverse transaction types, and fraud scenarios designed to identify anomalies through attributes such as transaction type, amount, frequency, and timing. This research intends to advance the techniques of online fraud detection through the realistic and dynamic attributes of the IEEE dataset, contributing valuable insights to cybersecurity and financial safety.

The dataset is the foundation of this project, and based on it, a robust fraud detection model would be developed. It offers an abundance of transactional data through which patterns and anomalies that show fraudulent behavior could be identified. In this project, a dataset was accessed from a CSV file called transactions_df.csv containing a mix of legitimate and fraudulent transaction records. The data offered insight into both user behavior and transaction characteristics and was thus important in training the machine learning models.

The dataset contains 284,807 transactions, all of which have been labeled as fraudulent or genuine. The dataset is highly imbalanced in terms of the two classes; fraudulent ones constitute a minuscule portion of the entire data set, and this happens in real-life too, since fraudulent activities occur very rarely, but are indeed crucial to identify. Each transaction is described using a set of numerical and categorical features, richly describing user behavior, patterns of transactions, and temporal trends.

The dataset consists of multiple features that provide detailed insights into each transaction. Some of the key features are, TX_AMOUNT, TX_FRAUD, TX_DURING_WEEKEND, TX_DURING_NIGHT, ACCOUNT_ID_NB_TX_1H_WINDOW, ACCOUNT_ID_AVG_AMOUNT_1H_WINDOW, Amount_Deviation etc.These features, combined with the target labels, form a robust dataset for training machine learning models to detect fraudulent transactions. The dataset's diversity in feature types enables both behavior-based and pattern-based detection approaches. A summary of the key features and their descriptions is provided in the following table,

Feature Name	**Description**	**Type**
TRANSACTION_ID	Unique identifier for each transaction.	Categorical
TX_DATETIME	Timestamp of the transaction.	Datetime

ACCOUNT_ID	Unique identifier for the account associated with the transaction.	Categorical
TRANSACTION_TYPE	Type of the transaction (e.g., withdrawal, deposit).	Categorical
TX_AMOUNT	The monetary value of the transaction.	Numerical
TX_TIME_SECONDS	Time of the transaction in seconds since the start of the day.	Numerical
TX_TIME_DAYS	Day of the transaction relative to a reference point.	Numerical
TX_FRAUD	Binary flag indicating whether the transaction is fraudulent (1 for fraud, 0 otherwise).	Categorical
TX_FRAUD_SCENARIO	Fraud scenario identifier for fraudulent transactions.	Categorical
TX_AMOUNT_STD	Standard deviation of the transaction amount for the account.	Numerical
TX_AMOUNT_MEAN	Mean transaction amount for the account.	Numerical
Amount_Deviation	Deviation of the transaction amount from the account's mean transaction amount.	Numerical

Amount_Threshold	Threshold amount for flagging suspicious transactions.	Numerical
Time_Seconds_Diff	Difference in seconds between consecutive transactions.	Numerical
Time_Diff	Difference in time between consecutive transactions.	Numerical
TX_DURING_WEEKEND	Binary flag indicating whether the transaction occurred during the weekend.	Categorical
TX_DURING_NIGHT	Binary flag indicating whether the transaction occurred during nighttime hours.	Categorical
TX_FRAUD_1H_SCENARIO	Fraud scenario within a one-hour window for the account.	Categorical
ACCOUNT_ID_NB_TX_1H_WINDOW	Number of transactions made by the account within a one-hour window.	Numerical
ACCOUNT_ID_AVG_AMOUNT_1H_WINDOW	Average transaction amount for the account within a one-hour window.	Numerical

Table 1:Columns of Dataset

One important stage in getting the dataset ready for analysis and modelling is data pre-processing. Even though some preprocessing was completed, further work was required to create machine learning models that worked. These actions customised the dataset for fraud detection and addressed frequent issues with data quality.

The initial phase was dealing with null or missing values, which, if ignored, might affect model performance. A thorough review revealed missing data, which was either eliminated if it was scarce or inconsequential or imputed using statistical techniques (mean or median) to maintain the dataset's accuracy and objectivity.

The focus turned to preparing input features because the target feature, TX_FRAUD, was already binary-encoded (1 for fraudulent, 0 for legal). Binary flags like TX_DURING_WEEKEND and TX_DURING_NIGHT were developed through feature engineering to account for temporal trends, while variables like Amount_Deviation were constructed to capture transaction anomalies. To avoid distortion in the training of the model, continuous variables such as TX_AMOUNT were examined for outliers. In order to prevent larger features, such as transaction amounts, from controlling the training process, feature scaling was used to normalise the variables.

In this context, exploratory data analysis was crucial in understanding the dataset and its features and in extracting meaningful patterns and relationships that are present. This would help in identifying major trends, imbalances, and behavioral indicators needed for designing the most effective fraud detection models. A number of interesting insights emerge from the analysis presented below.

Class imbalance: During EDA, one of the most prominent observations was about class imbalance in the dataset. Fraudulent transactions were less than 0.17% of the total records, which makes the occurrence of such cases very rare in real-world scenarios. This severe imbalance presented a challenge to the machine learning models as they tend to be biased to the majority class unless balanced. The low ratio of fraudulent transactions made the resampling method essential, especially using the Synthetic Minority Oversampling Technique (SMOTE), in order to allow both classes to be learned adequately without biasing the model.

Trend of transaction amount suggested that fraudulent transactions happened more at lower monetary amounts. This pattern was picked out by histograms and density plots that indicated the high concentration of fraudulent cases at the lower ends of the spectrum of transaction amount.Legitimate transactions were balanced across all sizes of transactions. This observation means that fraudsters may target low-value transactions because they will likely not be caught in a conventional fraud detection system. Knowing this trend provided an important feature that was used to distinguish fraudulent behavior.

Temporal patterns showed that fraudulent transactions were more common at night and on weekends, as fraudsters often target low-monitoring periods. Binary flags like TX_DURING_WEEKEND and TX_DURING_NIGHT helped capture these trends. Account-level behavior analysis also revealed suspicious activity, with features such as ACCOUNT_ID_NB_TX_1H_WINDOW and ACCOUNT_ID_AVG_AMOUNT_1H_WINDOW detecting bursts of abnormal transactions. High transaction volumes in short timeframes, with unusually high or low amounts, typically signaled fraud, with scatter plots highlighting clusters of fraudulent activity for model training.

For that, the dataset was divided into two subsets: a training set and a testing set to make sure the machine learning models were indeed trained appropriately. The training set consisted of 80% of the dataset and was used by the applied machine learning algorithms to learn patterns and relationships that are hidden in the transactional data. The rest 20% of the dataset was kept as the test set. It was used to measure the model's performance on data it hasn't seen yet. This approach ensures that models were evaluated in terms of generalization, a factor of high importance when assessing whether the model is going to perform well in the real world.

Another major issue encountered in this project was class imbalance. The fraudulent transactions accounted for only 0.17% of the total data, which meant they were vastly underrepresented compared to legitimate transactions. This posed a risk of bias in the machine learning models because they might favor the majority class, that is, legitimate transactions, while neglecting the minority class, which is fraudulent transactions. The Synthetic Minority Oversampling Technique (SMOTE) was used during the training phase to address this problem.

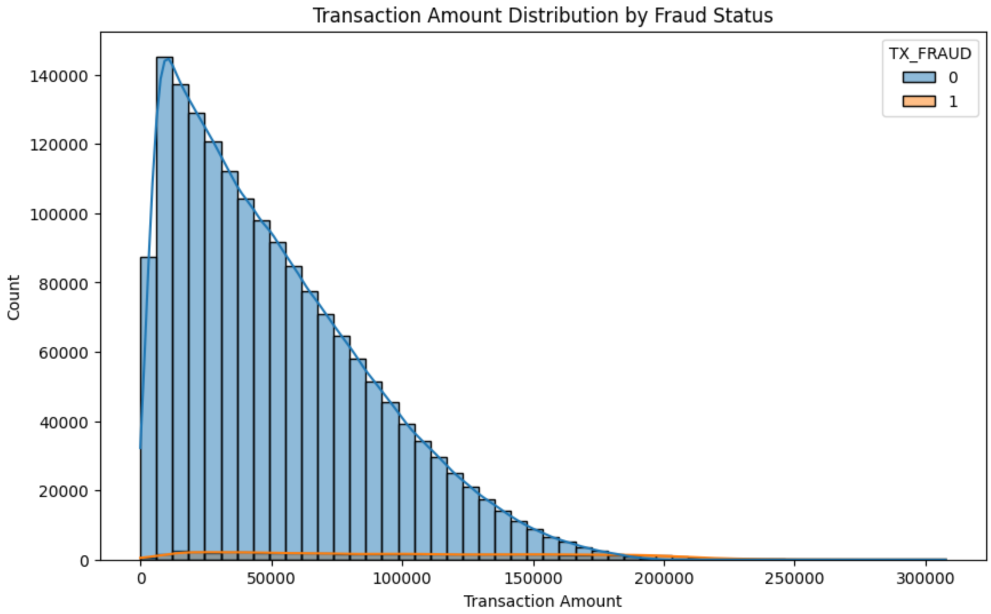
SMOTE is one of the common oversampling methods developed for handling an imbalanced dataset by generating artificial examples of the minority class. Instead of just duplicating current existing instances of fraudulent transactions, SMOTE creates new synthetic data points through interpolation of the existing example. This is achieved by identifying the nearest neighbors of a minority class instance in the feature space and generating synthetic examples along the line segments connecting them. It improves the representation of the minority class and introduces variationality to the data set, increasing the diversity of the training set.

Through the use of SMOTE, the training set had a balanced distribution with equal fractions of fraudulent and legitimate transactions. The overall practice of this helped in adequate training of models because the models learned to pick fraudulent transactions rather than being overwhelmed by the majority class. In the absence of it, the models might result in high general accuracy but would fail to pick frauds, which is a crucial limitation in practical applications.

The testing set, on the other hand, was not balanced to mirror the real-world distribution of transactions. This would make the evaluation of the models realistic to then avail for an exact measure of how good these models could potentially catch fraud transactions, given that the scenario was really highly imbalanced. The balanced training data together with realistic testing conditions formed a good framework within which to develop and test the fraud detection models.

Exploratory Data Analysis (EDA)

(Anon., n.d.)Exploratory Data Analysis (EDA) aimed to understand the dataset's structure and reveal insights to help model development. The investigation began by looking at label distribution, which revealed an imbalance between fraudulent and non-fraudulent transactions. Correlation heatmaps were utilised to determine links between features, indicating strongly linked pairs for possible removal. Transaction patterns were analysed over time, weekends, and evenings to identify behavioural distinctions between fraud and genuine activities. The distribution plots and boxplots indicated substantial differences in transaction quantities and deviations for fraudulent transactions. Scatterplots were used to investigate account-level activity across 1-hour time periods, revealing trends in transaction frequency and average amounts. Descriptive statistics, missing value checks, and association with the target variable highlighted important predictors such as 'Amount_Deviation' and 'TX_AMOUNT'. These insights, together with SMOTE's ability to handle class imbalance, offered a solid platform for feature selection and machine learning modelling, resulting in improved fraud detection accuracy.



\

Figure 2:Transaction amount distribution by fraud status

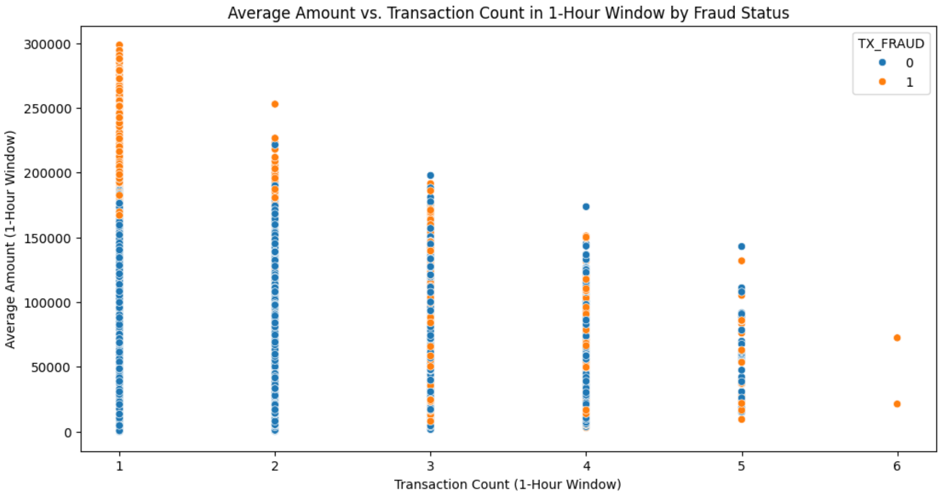


Figure 3:Average amount and transaction count within 1-hour window

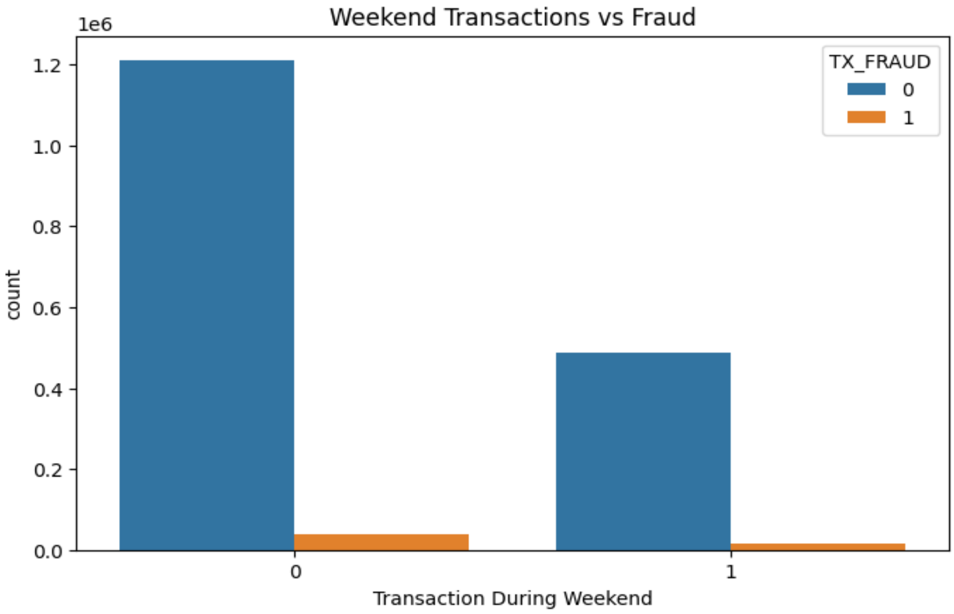


Figure 4:Transactions on weekends vs. fraud rate

Correlation map depicts the relationship between variables using values ranging from -1 to +1, with +1 being a perfect positive correlation and -1 representing a perfect negative correlation. A heatmap graphically illustrates this matrix by employing colours to express correlation strength, with warm colours representing high positive correlations and cold colours representing strong negative correlations. Heatmaps assist analysts in quickly identifying patterns and multicollinearity, emphasising essential aspects or those that may require change owing to strong association with others.

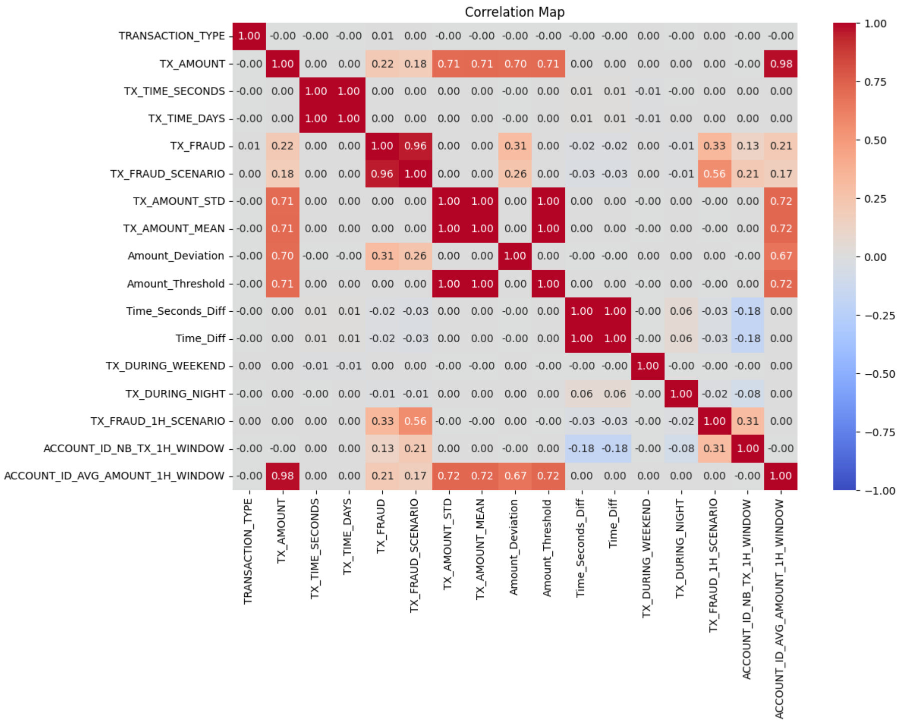


Figure 5:Correlation Map

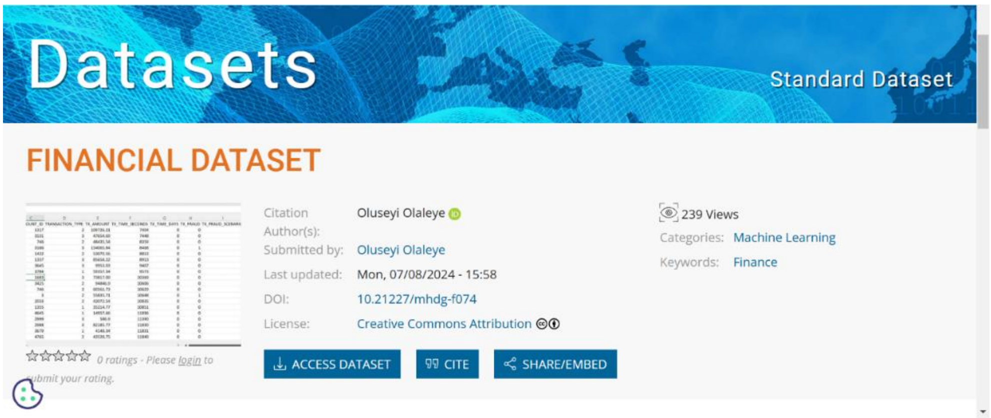
Ethical Issues

The IEEE DataPort repository is a public and curated source of academic and research datasets, which is the source of the dataset for this project. The financial dataset was prepared by Oluseyi Olaleye in 2020 and is hosted on IEEE DataPort. Since it is a public source, the dataset is bound by data privacy and ethical standards, so no personally identifiable information (PII) or sensitive information about an individual exists. All transaction records in the dataset have anonymized information, implying that information that can directly or indirectly lead to identification of the individuals or the organizations has been removed and obscured. This means that the data, thus, cannot raise privacy concerns or ethical violations and can thus be used for research and other academic projects.

The major characteristic of the dataset is the presence of fraud labels, indicating whether a transaction was fraudulent or legitimate. Advanced fraud detection mechanisms had assigned such labels before releasing the dataset. Pre-labeling allows researchers to concentrate on model development and evaluation without handling sensitive data during the labeling process.Further, the structure and content of the dataset are developed to balance the realism with privacy, so the patterns and behaviors in the data reflect real transaction dynamics while ensuring confidentiality of the users.

By using this data set for only academic purposes, this project remains within the boundary of ethical practices in research. No exploitation and misuse of the data were present in the analyses and modeling. Instead, it adds to the larger purpose: the enhancement of fraud detection systems for the benefit of the organizations and people through security enhanced financial transactions.

Further, the dataset given does not carry any re-identification risk. Although the dataset contains some of the highly sensitive attributes that are about transaction amounts, time stamps, and behavioral flags that could be correlated back to specific individuals or accounts, all the features are fully depersonalized. Non-sensitive anonymized data shields privacy, along with other considerations of the code of ethics surrounding the design and implementation of a machine learning model in cyber fraud detection.



Datasets

Standard Dataset

FINANCIAL DATASET

Citation Author(s): Oluseyi Olaleye

Submitted by: Oluseyi Olaleye

Last updated: Mon, 07/08/2024 - 15:58

DOI: 10.21227/mhdg-f074

License: Creative Commons Attribution

239 Views

Categories: Machine Learning

Keywords: Finance

☆☆☆☆☆ 0 ratings - Please login to submit your rating.

Figure 6:Proof of License (Olaleye, 2024)

\

Chapter 4 Methodology

It uses the development and testing of many machine learning models that could distinguish fraudulent online payments from a provided financial dataset. First, this paper involves preprocessing, which has everything to do with cleaning a given dataset, management of missing values, and features normalization in guaranteeing consistency and quality.In addition, feature engineering is used to extract some important features including transaction amount, time-based attributes, and behavior of the accounts, which would be critical to distinguish fraudulent transactions. Feature selection is done through the techniques of SelectKBest and chi-square. These assist in determining relevant features to use in training a model, therefore improving the process of learning because it reduces unnecessary or redundant information.

This research methodology outlines the process of developing and evaluating machine learning models for detecting fraudulent online payments. Adopting this approach entails data preprocessing, feature engineering, model development, evaluation, and performance analysis in order to ensure that the chosen model can predict fraud in online transactions with reliability.

1. Data Preprocessing

Importing and preprocessing the dataset is the first step of the methodology. The dataset "transactions_df.csv" contains multiple features, like transaction details: amount, time, type, and fraud labels. Preprocessing focuses on cleaning the dataset: missing values handling, encoding categorical variables, and normalization of continuous features. Features related to transaction behavior and fraud patterns are also identified. The preprocessing steps take care of prepping the data for training by making sure there is clean data available for making models.

2. Model Development

Several fraud detection models using machine learning are developed and compared in this project. The model selection is made based on its ability to process the complexity of the dataset, as well as its known classification performance.

• **K-Nearest Neighbors (KNN):** KNN is a non-parametric classification algorithm that depends on proximity in the feature space. This is because it is easy to apply and can perform on small to medium-sized data sets. For the purpose of this project, the trained KNN classifier has been built with a particular k-neighbors, in this case k = 32, to test their precision in prediction against fraud.

• **XGBoost: XGBoost** (Nalluri, et al., 2020) is one of the more efficient gradient boosted algorithms that provides the best-in-class performance especially for classification-type tasks. Based on its accuracy, it suits for imbalanced datasets and complexities of decision boundary. The Hyperparameter tuning will be done through number of tree, learning rate, and also max depth by XG Boost model.

•**Logistic Regression**: (Peng, et al., 2010)This is a linear model that is designed for binary classification. In this case, logistic regression is used because it can distinguish between the fraudulent and legitimate transactions. Regularization techniques allow for better results and less overfitting by the model.

- **Voting Classifiers**: Voting classifiers use the strengths of several models together. A voting classifier is quite simple; it simply aggregates predictions from multiple models, such as Logistic Regression or Naive Bayes, via soft voting. These are ensemble methods which aim to exploit diversity in the outputs of different models to obtain higher accuracy and robustness.

4. Model Evaluation

The models are measured for their performances using different metrics: accuracy, precision, recall, F1-score, and confusion matrices. Accuracy is a measure of total correctness of predictions. In addition, precision and recall are measured to understand how well fraudulent versus non-fraudulent transactions are classified by each model. The F1-score is used to balance precision and recall, especially when dealing with imbalanced datasets. Additionally, confusion matrices are used to visually represent the true positives, false positives, true negatives, and false negatives.

5. Model Optimization

To further enhance the models' performance, optimization techniques such as resampling (e.g., SMOTE) and hyperparameter tuning are applied. Resampling helps address class imbalance by generating synthetic instances of the minority class (fraudulent transactions), improving the model's ability to detect fraud. Hyperparameter tuning involves adjusting the parameters of each model to find the optimal configuration that maximizes performance on the validation set.

6. ROC Curve and AUC Score

In addition to the above evaluation metrics, we create Receiver Operating Characteristic (ROC) curves and compute the Area Under the Curve (AUC) score for models that show a notable performance. The ROC curve depicts the balance between the true positive rate and the false positive rate for different threshold settings, while the AUC score is a measure of the overall capacity of the model to differentiate between fraudulent and non-fraudulent transactions.

7. Final Model Selection

The best one is selected out of the models tested based on its performance metrics after evaluation. One model that gains the highest accuracy, precision, recall, and F1-score balance is recommended for online fraud payment detection, and further scrutiny and validation check whether it actually generalizes to unseen data as well.

Chapter 5 Results

This section discusses the results and analysis of our machine learning models applied to the Online Payments Fraud Detection Dataset sourced from IEEE Dataport. The dataset includes a rich set of features that are very important in identifying fraudulent transactions.

The dataset was diversified and in-depth, therefore enabling the exhaustive testing of most of the models in machine learning. These models included K-Nearest Neighbors (KNN), Logistic Regression, Voting Classifier, and Xg Boost. In doing so, there was a sensible attempt to make evaluation fair and regulate the effects of the class-imbalanced phenomenon through the use of resampling methods.

This section compares and analyzes the performance of these models based on accuracy, confusion matrices, and overall predictive power to predict fraudulent transactions. The best model selected in this comparison will be applied in real-world fraud detection applications.

K-Nearest Neighbors (KNN) Classifier
(Anon., n.d.)The K-Nearest Neighbors (KNN) model achieved an accuracy of 89.81% in predicting fraud transactions. The confusion matrix for this model is presented in the figure below.

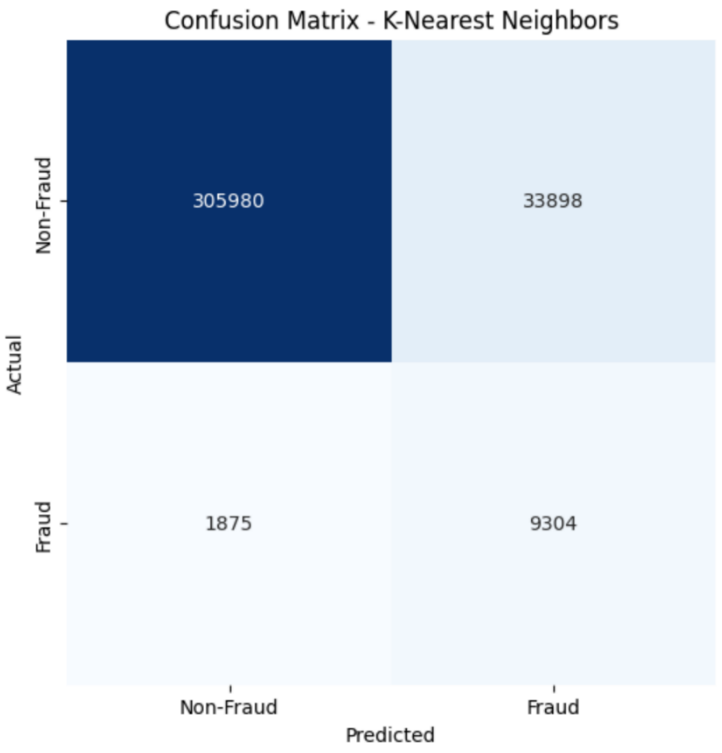


Figure 7:Confusion Matrix-KNN

The performance metrics of the model are defined as follows:

Accuracy = 89.81%, which means the high proportion of correct predictions

Precision = 99.39%: Most of the fraud transactions predicted by the model were found to be true

Recall = 90.03%: The model was successful in identifying 99.39% of all transactions

F1 Score = 94.48%, balancing both precision and recall.

This model is highly accurate and precise, though its false positive may be reduced by fine tuning the hyperparameters of the model. The high recall also reflects the ability to identify most fraudulent transactions.

Logistic Regression

The Logistic Regression model achieved an accuracy of 85.39% in predicting fraud transactions. The confusion matrix for this model is shown in the figure below.

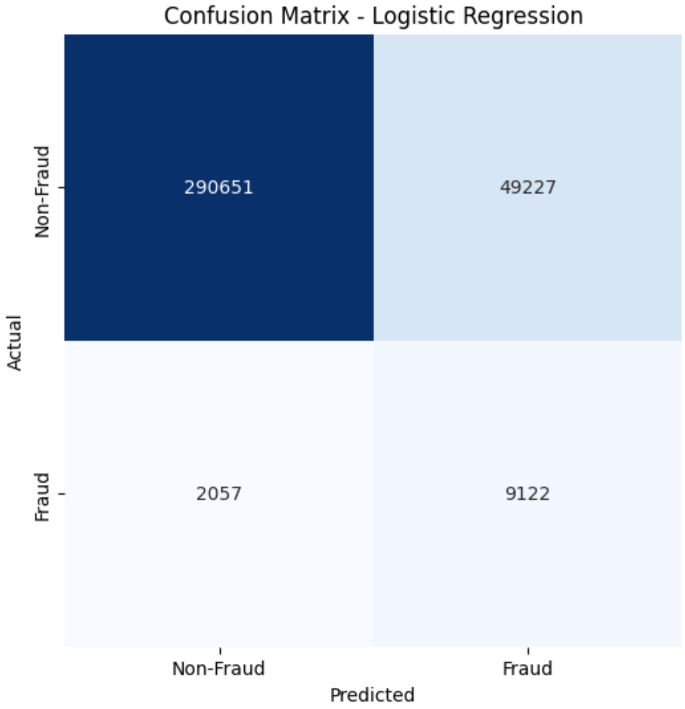


Figure 8:Confusion Matrix:Logistic Regression

The performance metrics of the model are as follows:

- Accuracy: 85.39% – most of the fraud predictions made by the model are correct.
- Recall: It is supposed to recall 85.52% fraud transactions in accurate manners.
- F1 Score: 91.89% – demonstrating a strong balance between precision and recall.

Although Logistic Regression gives good precision and recall, the false positives suggest further improvement by tuning parameters and selecting advanced features.

Voting Classifier

(Anon., n.d.)The Voting Classifier model, combining the Logistic Regression and Naive Bayes classifiers, achieved an accuracy of 94.36% in predicting fraud transactions. The confusion matrix for this model is shown in the figure below.

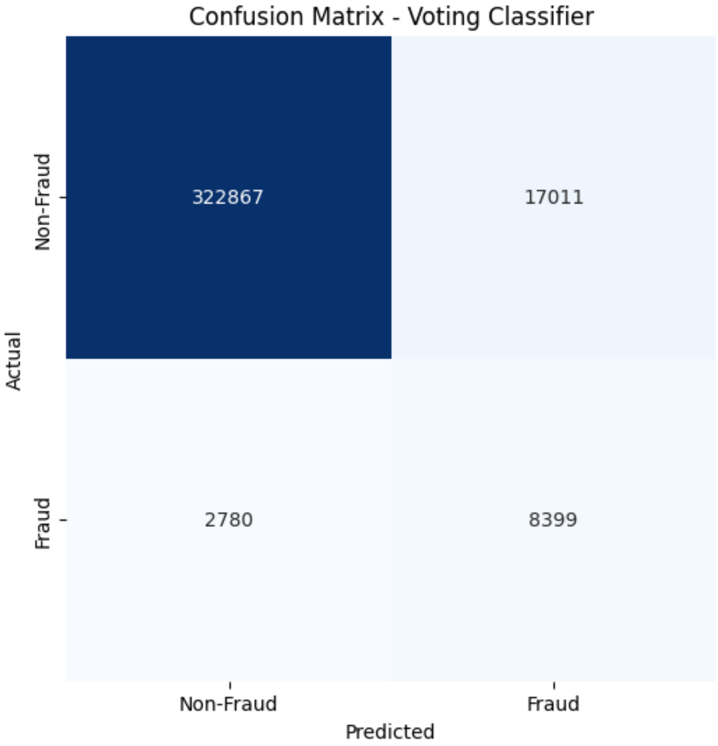


Figure 9:Confusion Matrix-Voting Classifier

Performance Metrics of the model are as follows:
Accuracy 94.36% indicates good prediction accuracy
Precision 99.15% most of the fraud predictions were correct.
Recall 94.99% that is the model correctly classified of fraudulent transactions
F1 Score 97.03% it has a balance of both precision and recall
Thus, though it has provided fairly good accuracy in predictions, still the performance might be improved further by fine-tuning the classifiers in the voting ensemble.

XG-Boost Model

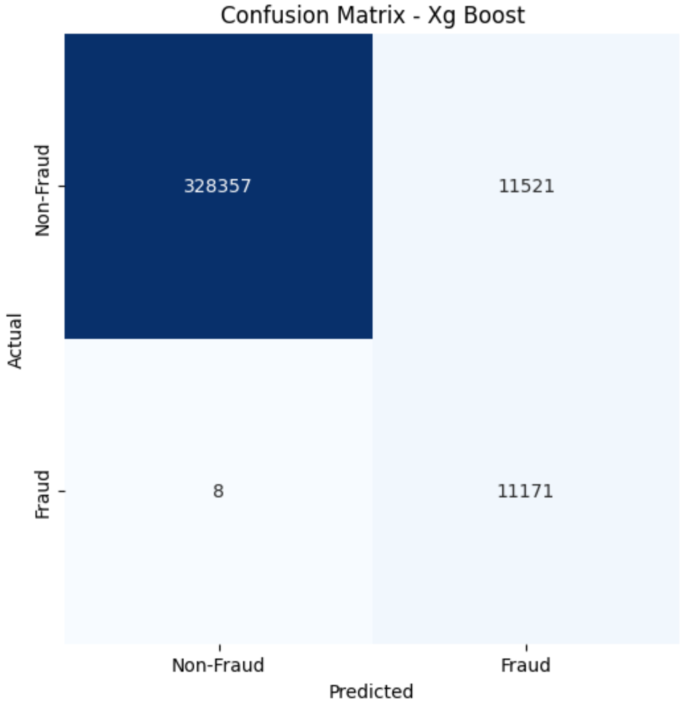


Figure 10:Confusion Matrix-XG Boost

This is evident from the above results, in which the model XGBoost performed exceptionally by classifying fraud transactions with accuracy of 96.72%. The confusion matrix for this model is given in the figure.

Model Metrics
Performance measures for the XGBoost algorithm are as below:
• **Accuracy: 96.72%** Highly accurate predictions in fraud and non-fraud cases.
•**Precision: 1.00%** – the fraud predictions of the model are highly reliable.
•**Recall: 96.61%** – the model is nearly perfect in detecting fraudulent transactions.
•**F1 Score: 98.27%** – an excellent balance between precision and recall.

The XGBoost model boasts high precision and recall and is therefore less likely to miss fraudulent activities. Misclassifications are also low, with the number of false positives and false negatives remaining low. Therefore, this shows that the model achieves a good balance between detection accuracy and minimal disturbance to legitimate transactions. Hyperparameter tuning can also be explored in further improvements so that performance could be optimized in production environments.

Model Comparison and Best Model Selection

When comparing the performance of the models, XGBoost clearly stands out as the best performer on all metrics. It has achieved an excellent accuracy of 96.72%, which is much higher than other models. Its precision of 1.00% shows that almost all the fraud predictions done by

XGBoost are accurate, and its recall of 96.61% shows its near-perfect ability to identify fraudulent transactions. Also, the F1 score is 98.27%, showing that XGBoost has the perfect balance of precision and recall; hence, XGBoost would be the most reliable model in fraud detection. These results point out the capability of XGBoost in minimizing false positives as well as false negatives, thereby not causing disruptions to legitimate transactions.

The KNN classifier is also strong with an accuracy of 89.81% but not as good as XGBoost. Its precision of 99.39% indicates that it is reliable in making correct fraud predictions, and its recall of 90.03% ensures that it captures most fraudulent transactions. The F1 score of 94.48% further emphasizes KNN's ability to balance precision and recall effectively. The fact that the false positives are more than XGBoost indicates that there is a scope for improvement in its performance by further hyperparameter tuning.

Logistic Regression as well as the Voting Classifier both are showing comparable performances with accuracies at 85.39% and 94.36%, respectively. The precision of the Logistic Regression model is 99.30% with a recall of 85.52%, and for the Voting Classifier, a slight better precision was achieved with 99.15% but lesser recall at 94.99%. These models, as represented by the F1 scores for Logistic Regression at 91.89% and Voting Classifier at 97.03%, represent a reasonable balance between precision and recall but trail both XGBoost and KNN in overall effectiveness. The feature selection of these models is not that great and there are lots of false positives. Therefore, feature selection and tuning of parameters will enhance the performance of these models.

XGBoost is best for fraud detection because it scores the highest precision, recall, and F1 score. Based on the output, the performance reliability of KNN classifier came out to be strong. Since both Logistic Regression and Voting Classifier results are reasonable and lower than those from XGBoost and KNN, further optimisation is in high demand, so that its usage in a production environment should become operational.

ROC Curve

(Hoo, et al., 2017)The Receiver Operating Characteristic (ROC) curve is a graphical representation of the performance of a classification model at various thresholds. It plots the True Positive Rate (sensitivity) versus the False Positive Rate (1-specificity) and shows how much one would have to compromise on the former for the latter. The Area Under the Curve (AUC) measures the general performance, which is closer to 1 better at discrimination. For XGBoost, the ROC curve is a very insightful quantity because the algorithm uses gradient boosting to minimize its classification errors, and it works to optimize its probabilities. So, the ROC curve usually is strong for AUC, which depicts the ability to distinguish between classes, making this algorithm a trustworthy one for purposes such as fraud detection or anomalies.

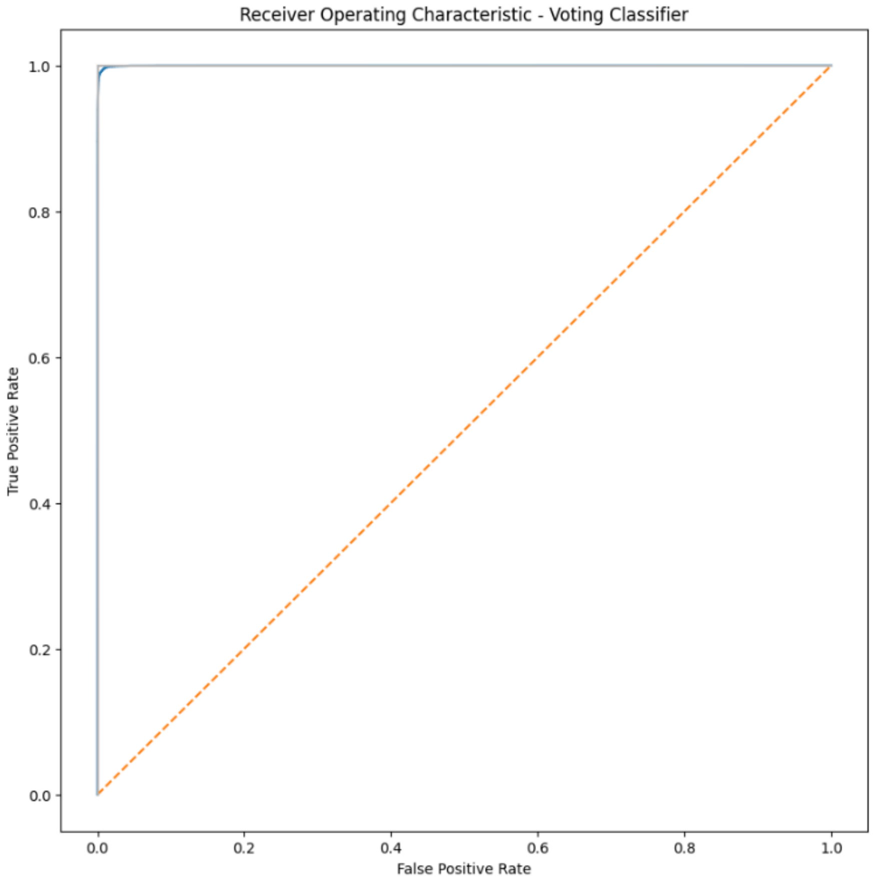


Figure 11:ROC CURVE-XG Boost

Chapter 6 ANALYSIS AND DISCUSSION

In this section, we report and discuss the performance metrics of the six machine learning models applied for the detection of online payment fraud. These are Naive Bayes, KNN Classifier, Logistic Regression, Voting Classifier, Stacking Classifier, and Decision Tree. It uses metrics such as True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN), Sensitivity (TPR), Specificity (SPC), Precision (PPV), Negative Predictive Value (NPV), False Positive Rate (FPR), False Discovery Rate (FDR), False Negative Rate (FNR), Accuracy (ACC), F1 Score, and Matthews Correlation Coefficient (MCC) for analysis.

Model Performance Metrics

Metric	**XGBoost**	**KNN Classifier**	**Logistic Regression**	**Voting Classifier**
True Positive (TP)	328,357	305,980	290,651	322,867
False Positive (FP)	8	1,875	2,057	2,780
True Negative (TN)	11,171	9,304	9,122	8,399
False Negative (FN)	11,521	33,898	49,227	17,011
Sensitivity (TPR)	0.9661	0.9003	0.8552	0.9499
Specificity (SPC)	0.9993	0.8323	0.8160	0.7513
Precision (PPV)	1.0000	0.9939	0.9930	0.9915
Negative Predictive Value	0.4923	0.2154	0.1563	0.3305
False Positive Rate (FPR)	0.0007	0.1677	0.1840	0.2487
False Discovery Rate (FDR)	0.0000	0.0061	0.0070	0.0085
False Negative Rate (FNR)	0.0339	0.0997	0.1448	0.0501
Accuracy (ACC)	0.9672	0.8981	0.8539	0.9436
F1 Score	0.9827	0.9448	0.9189	0.9703
Matthews Correlation Coefficient (MCC)	0.6894	0.3915	0.3166	0.4752

Table 2: Model Performance Metrics

Conclusion

The XGBoost model performs well in fraud detection, with a perfect Precision (1.0000), a high F1 Score (0.9827), and a low False Positive Rate, making it extremely dependable for distinguishing fraudulent situations. Its excellent blend of sensitivity, specificity, and accuracy demonstrates its strong performance. In comparison, the KNN Classifier, while reaching a respectable F1 Score (0.9448), has lower sensitivity (0.9003) and specificity (0.8323), as well as a larger False Positive Rate, resulting in a somewhat higher rate of misclassifications than XGBoost.

Logistic Regression has sufficient precision and accuracy but falls short in sensitivity and specificity. Its low Negative Predictive Value (0.1563) and greater False Negative Rate (0.1448) limit its ability to identify non-fraud instances. The Voting Classifier, with a balanced Sensitivity and Specificity, provides a dependable alternative, however its F1 Score (0.9703) is somewhat lower than XGBoost's. Overall, XGBoost is the most effective model, while the others provide decent, but less ideal, possibilities.

Future Work

1. **Real-time Fraud Detection**: Develop a fraud-detecting system that takes advantage of integrating trained models with an operational processing live transactions. Detection in real-time is a way to mitigate loss associated with fraud because, sooner than later, the financial loss will be contained by timely action against suspect behavior.
2. **Model Interpretability and Explainability**: General modelling of complex ensemble models is treated like "black boxes". So, such model explainability frameworks as SHAP or LIME may come in handy, providing an explanation of the system output and thus more trustworthy stakeholders when, for example, fraud detection results need to be explained to parties or regulatory entities.
3. **Integration with Security Systems**: The fraud detection model can be integrated with IDS or SIEM platforms along with other security tools to build a more holistic security. This will help in the development of concerted responses against fraud attempts, thereby increasing general security posture.
4. **Continuous Learning and Model Updates**: Frauds are in a flux mode; hence, to make the model contemporary, continuous retraining that includes new data may be quite crucial. A continuous pipeline of learning or an online system can help a model morph into newer fraud plans as they emerge.
5. **Evaluation and Benchmarking**: More evaluation on real-world datasets other than those already used for model training or even in production-like environments would better determine the efficacy of the models. It would then compare how it would work on different distributions of data or environments, which means it would work well in a variety of fraud types.
6. **Predictive Analytics for Future Fraud**: for Future Fraud Develop predictive analytics capabilities that can predict potential fraud events through the use of historical data and trends, moving detection from reactive to proactive types with organization readiness to take preventive actions against emerging fraud threats.

These could make the fraud detection system powerful, adaptive and scalable enough to better protect the online payment systems from financial frauds.

Bibliography

ALMAZROI, A. A. & AYUB, N., 2023. IEEE. *Online Payment Fraud Detection Model Using Machine Learning Techniques.Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10341223*

Anon., n.d. *K-Nearest Neighbor(KNN) Algorithm for Machine Learning.* [Online] Available at: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Anon., n.d. *Majority Voting Algorithm in Machine Learning.* [Online] Available at: https://www.javatpoint.com/majority-voting-algorithm-in-machine-learning

Anon., n.d. *What is exploratory data analysis (EDA)?.* [Online] Available at: https://www.ibm.com/think/topics/exploratory-data-analysis

Farouk, M. et al., 2024. Fraud_Detection_ML: Machine Learning Based on Online Payment Fraud Detection. *Journal of Computing and Communication,* 3(1), pp. 116-131.

Hoo, Z. H., Candlish, J. & Teare, D., 2017. What is an ROC curve?. *Emergency Medicine Journal,* Volume 34, pp. 357-359..Available at: https://emj.bmj.com/content/emermed/34/6/357.full.pdf

Nalluri, M., Pentela, M. & Eluri, N. R., 2020. A Scalable Tree Boosting System: XG Boost. *International Journal of Research Studies in Science, Engineering and Technology,* 7(12), pp. 36-51.Available at: https://ijrsset.org/pdfs/v7-i12/4.pdf

Namani, S., Mordharia, H., Gajare, N. & Bemila, T., 2024. ONLINE PAYMENT FRAUD DETECTION: AN INTEGRATED APPROACH. *International Research Journal of Modernization in Engineering Technology and Science.Available at: https://www.irjmets.com/uploadedfiles/paper//issue_4_april_2024/53881/final/fin_irjmets1713887125.pdf*

Olaleye, O., 2024. *Financial dataset.* [Online] Available at: https://ieee-dataport.org/documents/financial-dataset#files

Peng, C. Y. J., Lee, K. L. & Ingersoll, G. M., 2010. An Introduction to Logistic Regression Analysis and Reporting. *The Journal of Educational Research,* pp. 3-14.Available at: https://doi.org/10.1080/00220670209598786

Siddaiah, U., Anjaneyulu, P., Haritha, Y. & M, R., 2023. Fraud Detection in Online Payments using Machine Learning Techniques. *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS),* 8 June.Available at: https://ieeexplore.ieee.org/document/10142404

Venkatesh, M. B. B. K. B. B. M. C. &. M. D. (. O. P. F. D. U. M. L. I. 1. 8.-8. I.-2.-4. R. f. h., 2024. ONLINE PAYMENT FRAUD DETECTION. *JARIIE-ISSN(O),* 10(2), pp. 2395-4396.Available at: https://ijariie.com/AdminUploadPdf/ONLINE_PAYMENT_FRAUD_DETECTION_USING_MACHINE_LEARNING_ijariie22820.pdf

\

APPENDIX

```
!pip install xgboost


!pip install imbalanced-learn


## IMPORTING LIBRARIES


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
import seaborn as sns
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.metrics import confusion_matrix,accuracy_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import VotingClassifier
from numpy import argmax
from sklearn.metrics import classification_report
from sklearn.metrics import roc_curve, roc_auc_score
import random
import pandas as pd
```

```
from sklearn.preprocessing import LabelEncoder
import xgboost as xgb
from imblearn.over_sampling import SMOTE
from collections import Counter

## IMPORTING DATASET

df=pd.read_csv("transactions_df.csv")
df

## PRINTING COLUMNS

df.columns

## VISUALISING LABEL COUNT

# Count the occurrences of each label
label_counts = df['TX_FRAUD'].value_counts()
# Plotting the pie chart
plt.figure(figsize=(8, 8))
plt.pie(label_counts, labels=label_counts.index, autopct='%1.1f%%', startangle=140)
plt.title('Label Distribution')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle
plt.show()
```

```
## CORRELATION MAPPING

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Load your dataset
df1=df.drop(columns=['TRANSACTION_ID','TX_DATETIME', 'ACCOUNT_ID'])
# df = pd.read_csv('your_dataset.csv')

# Step 2: Generate a correlation matrix
correlation_matrix = df1.corr()

# Step 3: Visualize the correlation matrix as a heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm', vmin=-1, vmax=1)
plt.title("Correlation Map")
plt.show()

# Step 4: Identify highly correlated features
# Consider a threshold for high correlation (e.g., 0.8)
threshold = 0.8
high_correlation_pairs = [(i, j, correlation_matrix.loc[i, j])
                          for i in correlation_matrix.columns
                          for j in correlation_matrix.columns
```

```
                        if i != j and abs(correlation_matrix.loc[i, j]) > threshold]

print("Highly Correlated Feature Pairs (above threshold):")
for pair in high_correlation_pairs:
    print(f"{pair[0]} - {pair[1]}: {pair[2]}")

# Step 5: Remove two highly correlated features
# Manually decide which features to drop or automate the process
# Example: Dropping the first two features with high correlation
features_to_drop = set(pair[0] for pair in high_correlation_pairs[:2])
# df_reduced = df1.drop(columns=features_to_drop)

print(f"Removed features: {features_to_drop}")



## EDA

# 1. Display the first few rows
print("First few rows of the dataset:")
df.head()

# 2. Check for missing values
print("\nMissing values per column:")
print(df.isnull().sum())
```

```
# 3. Basic statistical summary of numerical columns
print("\nStatistical summary of numerical columns:")
print(df.describe())

# 4. Distribution of transaction types
print("\nTransaction type counts:")
print(df['TRANSACTION_TYPE'].value_counts())

# 5. Distribution of fraud vs. non-fraud transactions
print("\nFraud vs. Non-fraud transactions:")
print(df['TX_FRAUD'].value_counts())

# 6. Visualizing fraud and non-fraud transactions
sns.countplot(data=df, x='TX_FRAUD')
plt.title("Distribution of Fraud vs. Non-fraud Transactions")
plt.show()

# 7. Transaction amount distribution (Overall)
plt.figure(figsize=(10, 6))
sns.histplot(df['TX_AMOUNT'], bins=50, kde=True)
plt.title("Distribution of Transaction Amounts")
plt.xlabel("Transaction Amount")
plt.show()

# 8. Transaction amount distribution by fraud status
```

```
plt.figure(figsize=(10, 6))

sns.histplot(data=df, x='TX_AMOUNT', hue='TX_FRAUD', bins=50, kde=True)

plt.title("Transaction Amount Distribution by Fraud Status")

plt.xlabel("Transaction Amount")

plt.show()


# 9. Mean transaction amount per fraud status

mean_amount_by_fraud = df.groupby('TX_FRAUD')['TX_AMOUNT'].mean()

print("\nMean transaction amount by fraud status:")

print(mean_amount_by_fraud)


# 10. Analysis of Amount Deviation by Fraud Status

plt.figure(figsize=(10, 6))

sns.boxplot(data=df, x='TX_FRAUD', y='Amount_Deviation')

plt.title("Amount Deviation by Fraud Status")

plt.show()


# 11. Transaction count by time of day

plt.figure(figsize=(10, 6))

sns.histplot(df['TX_TIME_SECONDS'], bins=24, kde=True)

plt.title("Distribution of Transactions by Time of Day (Seconds)")

plt.xlabel("Time in Seconds")

plt.show()


# 12. Transactions on weekends vs. fraud rate
```

```
df['TX_DURING_NIGHT'] = df['TX_DURING_NIGHT'].astype(str)
df['TX_FRAUD'] = df['TX_FRAUD'].astype(str)
plt.figure(figsize=(8, 5))
sns.countplot(data=df, x='TX_DURING_WEEKEND', hue='TX_FRAUD')
plt.title("Weekend Transactions vs Fraud")
plt.xlabel("Transaction During Weekend")
plt.show()

# 13. Transactions at night vs fraud rate
# Convert 'TX_DURING_NIGHT' and 'TX_FRAUD' to strings to ensure proper interpretation
df['TX_DURING_NIGHT'] = df['TX_DURING_NIGHT'].astype(str)
df['TX_FRAUD'] = df['TX_FRAUD'].astype(str)
plt.figure(figsize=(8, 5))
sns.countplot(data=df, x='TX_DURING_NIGHT', hue='TX_FRAUD')
plt.title("Night Transactions vs Fraud")
plt.xlabel("Transaction During Night")
plt.show()

# 14. Average amount and transaction count within 1-hour window
plt.figure(figsize=(12, 6))
sns.scatterplot(data=df, x='ACCOUNT_ID_NB_TX_1H_WINDOW', y='ACCOUNT_ID_AVG_AMOUNT_1H_WINDOW', hue='TX_FRAUD')
plt.title("Average Amount vs. Transaction Count in 1-Hour Window by Fraud Status")
plt.xlabel("Transaction Count (1-Hour Window)")
plt.ylabel("Average Amount (1-Hour Window)")
plt.show()
```

```
# 15. Correlation matrix heatmap

x1=df.drop(columns=['TX_DATETIME'])

plt.figure(figsize=(14, 10))

sns.heatmap(x1.corr(), annot=True, cmap='coolwarm', fmt='.2f')

plt.title("Correlation Matrix of Features")

plt.show()
```

## ASSIGNING VALUES FOR DATA AND TARGET

```
x=df.drop(columns=['TRANSACTION_ID','TX_DATETIME', 'ACCOUNT_ID','TX_FRAUD','TX_FRAUD_SCENARIO'])

y=df['TX_FRAUD']
```

```
x
```

```
y
```

## FITTING AND TRAINING THE MODEL

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(x, y,test_size=.2,stratify=y ,random_state=56)
```

## USING SMOT

```
print("Before SMOTE:", Counter(y))


smote = SMOTE(random_state=42)

X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)


print("After SMOTE:", Counter(y_train_resampled))


## ML MODELS


# KNN


KNN1=KNeighborsClassifier(n_neighbors=32, metric='minkowski', p=2 )

KNN1.fit(X_train_resampled, y_train_resampled)

y_predKNN=KNN1.predict(X_test)

y_actKNN=y_test

acc=accuracy_score(y_actKNN,y_predKNN)

cm=confusion_matrix(y_actKNN,y_predKNN)

print(cm)

print("accuracy:",acc)


## CROSS VALIDATION


from sklearn.model_selection import cross_val_score

# Perform 5-fold cross-validation

cv_scores = cross_val_score(KNN1, X_train_resampled, y_train_resampled, cv=5, scoring='accuracy')
```

```
# Output results
print(f"Cross-validation scores: {cv_scores}")
print(f"Average accuracy: {cv_scores.mean()}")
print(f"Standard deviation: {cv_scores.std()}")

## AUC ROC CURVE

y_pred_proba = KNN1.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test.astype(int), y_pred_proba)
auc_score = roc_auc_score(y_test, y_pred_proba)

plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - KNN')
plt.plot(fpr, tpr,label='ROC curve (AUC = %0.2f)' % auc_score)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()

## LOGISTIC REGRESSION

LR= LogisticRegression(
    penalty='l2',           # Regularization type
    C=1.0,                  # Inverse of regularization strength
    solver='lbfgs',        # Optimization algorithm

    max_iter=100,          # Maximum number of iterations

    class_weight='balanced',  # Handles class imbalance

    random_state=42        # For reproducibility

)

LR.fit(X_train_resampled, y_train_resampled)

y_predLR=LR.predict(X_test)

y_actLR=y_test

print(confusion_matrix(y_actLR,y_predLR))

print("accuracy_score:",accuracy_score(y_actLR,y_predLR))


## CROSS VALIDATION


from sklearn.model_selection import cross_val_score

# Perform 5-fold cross-validation

cv_scores = cross_val_score(LR, X_train_resampled, y_train_resampled, cv=5, scoring='accuracy')

# Output results

print(f"Cross-validation scores: {cv_scores}")

print(f"Average accuracy: {cv_scores.mean()}")

print(f"Standard deviation: {cv_scores.std()}")


## AUC ROC CURVE


y_pred_proba = LR.predict_proba(X_test)[:, 1]

fpr, tpr, thresholds = roc_curve(y_test.astype(int), y_pred_proba)
```

```
auc_score = roc_auc_score(y_test, y_pred_proba)

plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - LR')
plt.plot(fpr, tpr,label='ROC curve (AUC = %0.2f)' % auc_score)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
print(auc_score)




## VOTING CLASSIFIER

estimators = [  ('rf', LogisticRegression()),
            ('dt',KNeighborsClassifier() )]
vot_hard = VotingClassifier(estimators = estimators, voting ='soft')
vot_hard.fit(X_train_resampled, y_train_resampled)
y_pred_VOT=vot_hard.predict(X_test)
y_act_VOT=y_test
acc=accuracy_score(y_act_VOT,y_pred_VOT)
print(confusion_matrix(y_act_VOT,y_pred_VOT))
print("accuracy:",acc)


## CROSS VALIDATION


from sklearn.model_selection import cross_val_score

# Perform 5-fold cross-validation

cv_scores = cross_val_score(vot_hard, X_train_resampled, y_train_resampled, cv=5, scoring='accuracy')

# Output results

print(f"Cross-validation scores: {cv_scores}")

print(f"Average accuracy: {cv_scores.mean()}")

print(f"Standard deviation: {cv_scores.std()}")


## ROC CURVE


y_pred_proba = vot_hard.predict_proba(X_test)[:, 1]

fpr, tpr, thresholds = roc_curve(y_test.astype(int), y_pred_proba)

auc_score = roc_auc_score(y_test, y_pred_proba)


plt.subplots(1, figsize=(10,10))

plt.title('Receiver Operating Characteristic - Voting Classifier')

plt.plot(fpr, tpr,label='ROC curve (AUC = %0.2f)' % auc_score)

plt.plot([0, 1], ls="--")

plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")

plt.ylabel('True Positive Rate')

plt.xlabel('False Positive Rate')
plt.show()


## XG BOOST


# Initialize the XGBoost model for classification
X_BoostModel = xgb.XGBClassifier(
    n_estimators=100,        # Number of trees
    max_depth=6,             # Maximum depth of a tree
    learning_rate=0.1,       # Step size shrinkage
    subsample=0.8,           # Subsample ratio of the training data
    colsample_bytree=0.8,    # Subsample ratio of columns
    random_state=42          # Random seed
)
y_train_resampled = y_train_resampled.astype(int)
y_test = y_test.astype(int)
X_test = X_test.astype(int)
X_train_resampled=X_train_resampled.astype(int)


# Fit the model
X_BoostModel.fit(X_train_resampled, y_train_resampled)
# Predict on test data
y_act=y_test
y_pred = X_BoostModel.predict(X_test)
acc=accuracy_score(y_act,y_pred)
cm=confusion_matrix(y_act,y_pred)
```

```
print(cm)

print("accuracy:",acc)



## ROC CURVE


y_pred_proba = X_BoostModel.predict_proba(X_test)[:, 1]

fpr, tpr, thresholds = roc_curve(y_test.astype(int), y_pred_proba)

auc_score = roc_auc_score(y_test, y_pred_proba)


plt.subplots(1, figsize=(10,10))

plt.title('Receiver Operating Characteristic - XG Boost')

plt.plot(fpr, tpr,label='ROC curve (AUC = %0.2f)' % auc_score)

plt.plot([0, 1], ls="--")

plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")

plt.ylabel('True Positive Rate')

plt.xlabel('False Positive Rate')

plt.show()


## FOR VISUALIZING CM


import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

from sklearn.metrics import confusion_matrix
```

```
# Confusion matrices for each model (These would be derived from your earlier outputs)
cm_knn = np.array([[305980, 33898], [1875, 9304]])  # K-Nearest Neighbors
cm_lr = np.array([[290651, 49227], [2057, 9122]])  # Logistic Regression
cm_voting = np.array([[322867, 17011], [2780, 8399]])  # Voting Classifier
cm_xg = np.array([[328357, 11521],[8, 11171]])


# Function to plot confusion matrix
def plot_confusion_matrix(cm, model_name):
    plt.figure(figsize=(6, 6))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", cbar=False, xticklabels=["Non-Fraud", "Fraud"], yticklabels=["Non-Fraud", "Fraud"])
    plt.title(f"Confusion Matrix - {model_name}")
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()


# Plot for each model
plot_confusion_matrix(cm_knn, "K-Nearest Neighbors")
plot_confusion_matrix(cm_lr, "Logistic Regression")
plot_confusion_matrix(cm_voting, "Voting Classifier")
plot_confusion_matrix(cm_xg, "Xg Boost")
```