Journal of
High School
Science

# A machine learning approach to detect fraudulent customers based on their financial transaction history

Xie L

Submitted: May 13, 2023, Revised: version 1, August 4, 2023
Accepted: August 13, 2023

## Abstract

Financial fraud is a global phenomenon that causes billions in financial losses every year and poses a major threat for organizations and customers. However, it has become increasingly difficult to detect using traditional auditing methods due to the growth of technology. To address this issue, we implemented a machine learning based approach to detect fraudulent agents by analyzing the history of transactions of customers. We designed a method and a system to detect fraudulent activity at the customer level instead of at individual transactions, since this approach could potentially be used to uncover entire fraudulent networks. Four families of features were designed that capture multiple aspects of a customer's behavior: (i) transaction volume, (ii) the difference in balance between the sender and receiver, (iii) the transaction volume in comparison to a customer's balance, and (iv) overall customer behavior that summarized the customer as a whole, which made up a total of 44 different features. The dataset that we analyzed included 9071212 customers (8169 fraudulent, 9063043 legitimate) through the history of 6362620 transactions. We suggested a classification algorithm based on the XGBoost model and found that our proposed model performed with an accuracy of 99.93% and a ROC AUC score of 0.979, classifying customers with a false positive rate of less than 0.001% out of 2722703 total predictions.

## Keywords

Financial fraud, Financial fraud detection, Data mining, XGBoost, Machine learning, Transactions, Classification algorithm, Malware, Customer, Cybersecurity

Larris Xie, Bayview Secondary School, 10077 Bayview Ave., Richmond Hill, ON L4C 2L4, Canada. larris.xie@gmail.com

## Introduction

The term “financial fraud” refers to the use of deceptive practices for financial gain. Financial fraud has been evolving to become a global threat affecting all countries and populations. While some instances may only have an influence on a few people, others may have far-reaching repercussions that impact numerous stakeholders, including consumers and even governments. Financial fraud is an active threat that needs to be addressed as victims can suffer significant financial losses and may cause people to lose their trust in the financial system, damaging the economy as a whole. Unfortunately, financial fraud can be difficult to detect and prosecute, which makes it a persistent problem.

Early methods that were used for detecting financial fraud included auditing and structured financial analysis to identify suspicious patterns through large volumes of financial data (1, 2). This approach included methods that involved reviewing financial statements and transactions, comparing a company’s financial performance to industry benchmarks, etc. However, fraudsters have been developing better methods (e.g. encryption based techniques) to conceal their fraudulent activity and evade traditional detection methods (3). As a result, in order to combat the constantly evolving state of financial fraud, cybersecurity professionals and the cybersecurity community as a whole, have been designing better detection methods, notably data mining and machine learning algorithms, to identify patterns and anomalies in large financial datasets. By automating the process of flagging suspicious transactions, it makes it more difficult for fraudsters to operate undetected.

The objective of this research was to develop a machine learning approach to identify fraudulent activity. More specifically, we focused on modeling the behavior of fraudulent customers and developed an algorithm to distinguish them from legitimate customers. Therefore, our model identified fraudulent customers instead of individual financial transactions, making it possible to stop a wide range of fraudulent activities as a fraudster can have multiple streams of revenue and ways to conduct fraud, saving potential victims from significant financial losses.

We designed multiple families of features to capture the behavior of fraudulent customers. In particular, we developed four features, namely, i) Transaction volume, ii) Sender and receiver account balance, iii) Transaction volume as a ratio over the customer’s balance, and iv) Overall customer behavior. By analyzing a total of 6362620 transactions composed of 9063043 legitimate and 8169 fraudulent customers, an example of a common behavior we found in fraudulent customers was carrying out a large fraudulent transaction, frequently depleting the victim’s account, and then dealing in several smaller transactions to cover up their fraudulent activity, on average carrying out 7 more transactions than legitimate customers. We implemented a model that was based on the XGBoost classification algorithm to differentiate between the legitimate and the fraudulent customers. Our algorithms performed at an operational point with an accuracy of 99.9315% and a false positive rate of 0.00099%.

## Case Study

In this section, we discuss the malware Zeus, an infamous real-world example of a large-scale attacks against banks. We discuss why an effective solution for detecting financial fraud is important, as financial fraud is a global issue with egregious capabilities and consequences.

*How the Zeus bot works*

Zeus is a Trojan horse malware package that penetrates computers to steal banking information through keystroke logging and form grabbing (3). Zeus functions as a Man-in-the-Browser that enables fraudsters to manipulate what the victim sees on their screen, a common example being injecting extra fields, such as PIN numbers, in a legitimate bank's login webpage (4). Additionally, the malware is able to copy itself to other computers through instant and email messages, where hackers can then control the infected devices. In each infection, the Zeus Trojan also applies various techniques to evade detection, notably involving encryption (5).

*The timeline of the attack*

Two years after the malware was first identified in July 2007 (3), the largest attack by the Zeus malware ensued in 2009. The malware managed to compromise over 74,000 FTP accounts from Fortune 500 companies like Bank of America, NASA, Oracle, Cisco, and Amazon (6). In the same year, Zeus sent over 1.5 million phishing messages on Facebook and upwards of 9 million phishing emails impersonating Verizon (7). Eventually, the full source code of the malware package was released to the public in 2011, enabling the development of malicious variants (3).

*Impact*

In the 2009 attack, the Zeus Trojan infected the Protected Storage (PStore) of computers to gather any Internet Explorer, FTP, or POP3 passwords, which enabled fraudsters to illegally wire transfer money from the accounts of victims (6). In this attack, Zeus infected more than 3 million PCs and 88% of Fortune 500 companies, causing damages worth hundreds of millions of dollars (5). Zeus became the largest and most successful botnet software in the world, accounting for around 44% of all banking malware infections in over 2500 organizations across 196 different countries (7).

Although the original Zeus has been largely neutralized, its components have been used in new variants of malware after the source code was released in 2011 (4). There have been attacks as recently as 2022 from variants like the Kronos malware (8), making it clear that a solution for effective financial fraud detection is necessary.

## Background

In this section, we review the landscape of fraudulent transactions through the common types of financial fraud and existing industry strategies for detecting financial fraud.

*Credit Card Fraud*

Credit card fraud occurs when someone makes unauthorized purchases using a stolen credit or debit card, often performed remotely. This can be done through email phishing, skimming, pick-pocketing, and social engineering. A common example is when fraudsters send mass emails pretending to be legitimate corporations to steal credit card information.

*Mobile Fraud*

Mobile fraud is defined as using mobile phishing methods to steal credentials for mobile payment methods. This can be done through techniques such as reverse engineering, SIM swapping, overlay attacks, and mobile phishing. For example, fraudsters could reverse engineer a banking app to develop malware such as an attacker-generated screen that appears on top of the legitimate app. Thus, the victim's sensitive information is inputted into a form controlled by the attacker instead of the bank.

*Identity Theft*

Identity theft is the fraudulent use of someone's sensitive information for financial gain. Identity theft is often carried out through social engineering, malware, phishing, and data breaches. This includes buying personal information on the dark market through compromised websites to spear phish for payment credentials.

*Insurance Fraud*

Insurance fraud takes place when deceiving an insurer through exaggeration or false information in order to profit. This can be achieved through deception and duplicate claims. Common examples include making fake claims for accidents that did not happen, staging an accident, and lying about the value of a car.

*Mortgage Fraud*

Mortgage fraud is the deliberate misrepresentation of information to obtain mortgage financing that would not have normally been granted. This can be committed by various people, including real estate agents, loan officers, mortgage brokers, and appraisers, through social engineering. For example, one who omits income and assets to maximize profits on a loan transaction is committing mortgage fraud.

*Debt Collection Fraud*

Loan/debt collection fraud occurs when scammers call consumers demanding payments of fake outstanding debts. This is most often done through social engineering, where the fraudster pressures the victim to pay with cash, money transfer, or a prepaid card. A realistic example is someone posing as a government official to demand debt through a phone call, withholding information like the name of the creditor and the specific amount of debt.

*Wire Transfer Fraud*

Wire fraud is defined as any bank fraud involving telecommunication or electronic processes instead of physical communication. This can be done through various techniques including phishing, hacking, and social engineering. For example, a fraudster could use malware to take control of a remote workstation to transfer a large amount of money to an account, then closing the account and being long gone before people notice.

*Cryptocurrency Fraud*

The fast-changing and independent state of the cryptocurrency market makes it a popular target for fraudsters. Instantaneous and international transactions make cryptocurrency a common tool for tax fraud, bribery, and money laundering. Common techniques include creating fake coins, pump and dump schemes, Ponzi Schemes, hacking digital wallets, and social engineering. An example of cryptocurrency fraud is creating and mass promoting a newly developed crypto coin and selling the holdings at an artificial peak.

*Financial Statement Fraud*

Financial statement fraud occurs when a company manipulates its financial statements to make itself appear profitable. This allows them to raise stock prices, and exploit tax obligations and loan applications. Financial statement fraud is carried out through deception, often withholding information about cash flow or giving exaggerated estimated profits to increase the company's value.

*Securities and Commodities Fraud*

Securities and commodities fraud is deceiving a victim into investing into a company based on false information. This can be done through market manipulation, Ponzi and Pyramid Schemes, embezzlement, and foreign exchange fraud. An example is a broker falsifying information to persuade investors into buying stock for a near-bankrupt company.

*Money Laundering*

Money laundering is defined as transferring illegally obtained money through criminal activity. The recent growth of online banking and cryptocurrency make it easier for criminals to handle transactions without detection. This is done through deception, trafficking, smurfing, and wire transfers. For example, a criminal might split large chunks of money from selling drugs in smaller deposits through multiple accounts.

## How the Industry Protects Consumers

This subsection provides two industry examples of software used to protect consumers from financial fraud.

PayPal’s IGOR is a program that identifies patterns in organization records that match fraudulent criteria (9). IGOR employs rule-based classification and a neural network based on characteristics such as the volume of payment amounts, account usage frequency, and any similarities in the information the fraudsters use when opening accounts (10). Suspicious accounts are then restricted for a human investigator to investigate. PayPal's fraud rate drastically decreased as a result of their machine learning techniques; it stood at 0.17% in 2019---over 90% lower than the industry average (11). However, organization backlash results from the program incorrectly classifying legitimate users as fraudulent (12), a factor that is important to consider in the real-world.

Sift Science is an online platform that uses a global network of data to detect fraudulent activity through deep learning (13). Founded in 2011 by Brandon Ballinger and Jason Tan, they were the first to use machine learning for fraud prevention (14). Sift uses a stack of algorithms built upon Logistic Regression, Decision Forest, and Naive Bayes, to build a model capable of learning new fraudulent behavior within a fraction of a second (13). They extract features that fall into three primary categories: identification, behavioral, and similarity, which enables them to find who the user is, their suspicious actions, and any similar users, such as those that use the same IP address. Sift's deployed neural networks process over 70 billion monthly events for 34K businesses across practically every industry, handling $1.5 billion in annual value for customers (15). Their services are currently used by businesses such as DoorDash, Indeed, and Airbnb, and prevent thousands of dollars in fraud losses on a daily basis (16).

## Related Work

As financial fraud is an extensive problem tackled by data scientists, financial institutions, and governments alike, prior research in financial fraud detection has been extensive. Detection methods are constantly evolving, from structured quantitative analysis pioneered by individuals such as Beaver (1) and Altman (2), to more technical practices in the areas of data mining and machine learning systems.

### *Machine Learning*

Dheepa et al. (17) approached the challenge of credit card fraud using support vector machines (SVM) in a behavior based classification approach. Similarly, Srivastava et al. (18) recognized the increasing use of credit card fraud and modeled a Hidden Markov Model (HMM) based on the normal behavior of a cardholder. In recent times, Singh et al. (19) proposed a new methodology for detecting credit card fraud in 2022. They hybridized She's Firefly optimization algorithm (20) and SVM to enhance classification accuracy and reduce misclassification costs. Pambudi et al. (21) also aimed to improve the performance of the SVM classifier through Random Under Sampling and a minimum error-based principal component analysis. This paper also uses Lopez-Rojas's PaySim dataset (22) where they tackle the imbalance of fraudulent and legitimate transactions present in many of the available financial datasets.

### *Time-series Analysis*

Seyedhossein et al. (23) approached credit card fraud detection by mining information based on its time-series for online financial transactions. This was achieved through extracting the patterns inherent in the time series of aggregated daily amounts spent on credit cards, looking at the sequences of transactions rather than individual transactions. Another instance of time-series analysis is Takahashi et al.'s (24) method of using generative adversarial networks, a deep neural network-based approach, for financial time-series modelling.

### *Graph-level Analysis*

Qiu et al. (25) presented a new deep learning approach involving graph-level anomaly detection for financial fraud detection. Their approach involved combining deep one-class classification and self-supervision. Mao et al. (26) realized related-party transactions were a common way to implement financial fraud, but traditional quantitative analysis methods would treat each firm as an independent individual. They approached this issue using a knowledge graph for RPTs to mine hidden knowledge from large-scale associated data and the intricate relation among the transactions of related parties. Anti-money laundering is another topic graph learning was applied in. Weber and al. (27) provided a first look at scalable graph learning convolutional neural networks for analysis of dense financial data, reviewing current and emergent AML methods.

There are also several papers dedicated to surveying literature in this field. West et al. (28) presented a comprehensive review of over fifty scientific literature featuring financial fraud detection research spanning from 2004 to 2014. The type of literature reviewed was focused on data mining methods and computational intelligence techniques. In 2021, Zhu et al. (29) reviewed the development of financial fraud detection in the post-pandemic era based on the data type and method, with an emphasis on graph neural network methods due to their capacity for data analysis.

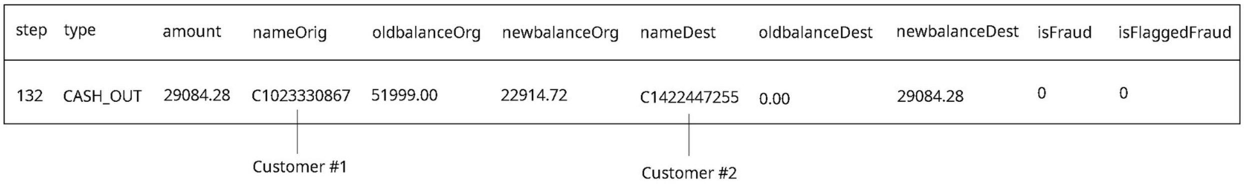
## Materials and Methods

The dataset we used to train and test our machine learning model for detecting financial customers through financial transactions was Lopez-Rojas's PaySim dataset (22) that can be found on Kaggle, an online database platform (https://www.kaggle.com/datasets/ealaxi/paysim1). PaySim is a simulation of mobile transactions based on one month of private financial logs from the African mobile financial company Ericsson (ericsson.com). This simulation covers the financial domain of mobile payments, with isolated cases of relatively straightforward fraud. The aim of the simulation was to explore the use of computer simulation for fraud detection and its applications in financial domains as currently, there is a lack of public available financial data which is extremely sensitive and difficult to obtain without breaking any non-disclose agreements. PaySim uses Multi Agent-Based Simulations (MABS) to add more realism to the simulation, where the behaviors of customers were calibrated using real data from financial transactions. MABS models and simulates the interactions and behaviors of multiple autonomous agents within a given environment, where rules of transactions and interaction were initially specified for each customer. PaySim also models fraudulent behavior using malicious agents that follow known criminal patterns. Statistics and social network analysis (SNA) were then used on real data to verify the relations between the simulated customers and statistically validate the synthetic datasets against the original source. Overall, simulations using techniques to mimic realism represent the highest level of realism achievable for us, since it is very challenging to obtain a public dataset on real financial transactions as financial institutions would have to release very private and sensitive data. PaySim has 6362620 rows which each represent a transaction, and 11 columns which represent various characteristics of the transaction. Table 1 shows the transaction characteristics.

Table 1. Characteristics of a transaction in PaySim

| **Column** | **Description** |
|---|---|
| step | hour when the transaction occurred, ranging from 1 to 744 |
| type | transaction type, either CASH-IN, CASH-OUT, DEBIT, PAYMENT or TRANSFER |
| amount | amount of money involved in the transaction in local currency |
| nameOrig | customer who initiated the transaction |
| oldbalanceOrg | customer's initial balance (before the transaction) |
| newbalanceOrig | customer's new balance (after the transaction) |
| nameDest | customer who is the recipient of the transaction |
| oldbalanceDest | recipient customer's initial balance (before the transaction) |
| newbalanceDest | recipient customer's new balance (after the transaction) |
| isFraud | 0 or 1, transaction made by a fraudulent agent marked as 1 |
| isFlaggedFraud | 0 or 1, attempted transactions flagged as illegal marked as 1 |

There are 9071212 unique customers (8169 fraudulent, 9063043 legitimate) across the 6362620 transactions. The average number of transactions a customer is involved in is 1.4028, since even though the number of customers exceeds the number of transactions, every transaction involves 2 customers as seen in Figure 1. Thus, every customer is involved in at least one transaction, with 1769 customers being involved with two or more transactions.

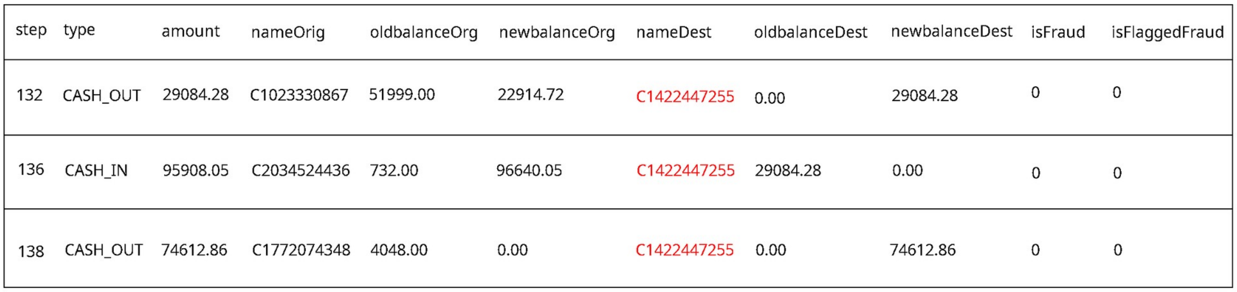


| step | type | amount | nameOrig | oldbalanceOrg | newbalanceOrg | nameDest | oldbalanceDest | newbalanceDest | isFraud | isFlaggedFraud |
|---|---|---|---|---|---|---|---|---|---|---|
| 132 | CASH_OUT | 29084.28 | C1023330867 | 51999.00 | 22914.72 | C1422447255 | 0.00 | 29084.28 | 0 | 0 |

Customer #1
Customer #2

Figure 1. Example of one row in the dataset, representing a single transaction involving 2 customers.

Our approach characterized and classified a customer based on their entire history of transactions rather than a single transaction. Figure 2 illustrates our approach of incorporating all the transactions of the customer highlighted in red into our features, rather than making predictions at the transaction level.



| step | type | amount | nameOrig | oldbalanceOrg | newbalanceOrg | nameDest | oldbalanceDest | newbalanceDest | isFraud | isFlaggedFraud |
|---|---|---|---|---|---|---|---|---|---|---|
| 132 | CASH_OUT | 29084.28 | C1023330867 | 51999.00 | 22914.72 | C1422447255 | 0.00 | 29084.28 | 0 | 0 |
| 136 | CASH_IN | 95908.05 | C2034524436 | 732.00 | 96640.05 | C1422447255 | 29084.28 | 0.00 | 0 | 0 |
| 138 | CASH_OUT | 74612.86 | C1772074348 | 4048.00 | 0.00 | C1422447255 | 0.00 | 74612.86 | 0 | 0 |

Figure 2. The model focuses on classifying a customer based on all their transactions compared to a single transaction like shown in Figure 1.

Features were extracted from these columns that help differentiate legitimate from fraudulent customers and were used to train and test a machine learning classification model based on XGBoost using Python. XGBoost (eXtreme Gradient Boosting) is an implementation of the gradient boosted trees algorithm, a supervised learning algorithm which combines predictions from a set of simpler, weaker models into one. XGBoost uses decision trees as its weak predictors and sequentially models each predictor based on the errors of its predecessor. This boosting technique allows each subsequent model to be trained to correct the mistakes made by the previous models. Unlike deep learning algorithms that use epochs to iterate over the training dataset multiple times, XGBoost does not rely on epochs. Instead, it builds a collection of decision trees in an iterative manner, with each iteration adding a new tree to the collection. Each tree is trained using a gradient boosting approach, which involves minimizing a loss function by optimizing the gradients of the loss based on the model's predictions. This training process in XGBoost continues until a stopping criterion is met, such as reaching a maximum number of iterations or when the improvement in the model's performance becomes negligible. These iterations are referred to as "boosting rounds" instead of epochs, where each boosting round adds a new tree to the collection, and the final prediction is the sum of predictions from all the individual trees. So, while XGBoost undergoes an iterative training process, it does not have the concept of epochs like in deep learning. Instead, it focuses on optimizing the model by gradually adding new trees based on the gradient boosting approach. The 44 features that were extracted over 9,075,669 unique customers fall into the following 4 families of features:

### *Transaction Volume*

This family of features represents the amount of money involved in each specific customer's transactions. We extracted this feature because the typical behavior of fraudulent customers is extracting large sums of money from the victim in a single transaction. This is done in an effort to maximize their profit while minimizing the danger of financial fraud by obtaining a large sum of money as quickly as possible. To characterize the transaction volume for each customer, we created a list of the amount of money in each transaction they dealt in. From there, various types of descriptive statistics were extracted, including the mean, median, standard deviation, and 25th, 75th, and 95th percentiles, for a total of 6 features in this family.

### *Sender and Receiver Account Balance*

This family of features summarizes the balances for the receiver and the sender of money before and after transactions. The characteristic oldBalanceRatio describes the ratio between the sender's and recipient's balances before the transaction. Similarly, newBalanceRatio refers to the ratio between the sender's and recipient's balances after the transaction. We used this family of features because it is frequently seen that fraudsters target wealthy customers to increase their profits, which results in a significant disparity between their balances. Similar to the previous family, we gathered a list of oldBalanceRatio and newBalanceRatio for each transaction a customer dealt in and extracted the previously mentioned descriptive statistics that added up to a total of 12 features in this family.

### *Transaction Volume as a Ratio over the Customer's Balance*

This family of features represents the volume of the transaction as a ratio over the balances of the two customers involved. The ratio between the sender's old balance and the overall amount of money involved is indicated by senderOldRatio. The ratio between their new balance and the total amount involved is specified by senderNewRatio. Likewise, the receiver's old balance to the amount involved is represented by receiverOldRatio, while their new balance to the amount is represented by receiverNewRatio. We extracted these features because legitimate customers rarely deal with a significant portion of their balance in a single transaction. On the other hand, fraudulent customers want to obtain money as quickly as possible, so they frequently make one big transaction to get all of their victim's money. In the vast majority of fraudulent transactions, the sender's initial balance before the transaction is equal to the volume of the transaction (Figure 3). This means that the fraudulent customer stole the full balance of their victim's account. As with the previous families, we gathered a list of the senderOldRatio, senderNewRatio, receiverOldRatio, and receiverNewRatio for each transaction of a customer and extracted the descriptive statistics that made up a total of 24 features in this family.

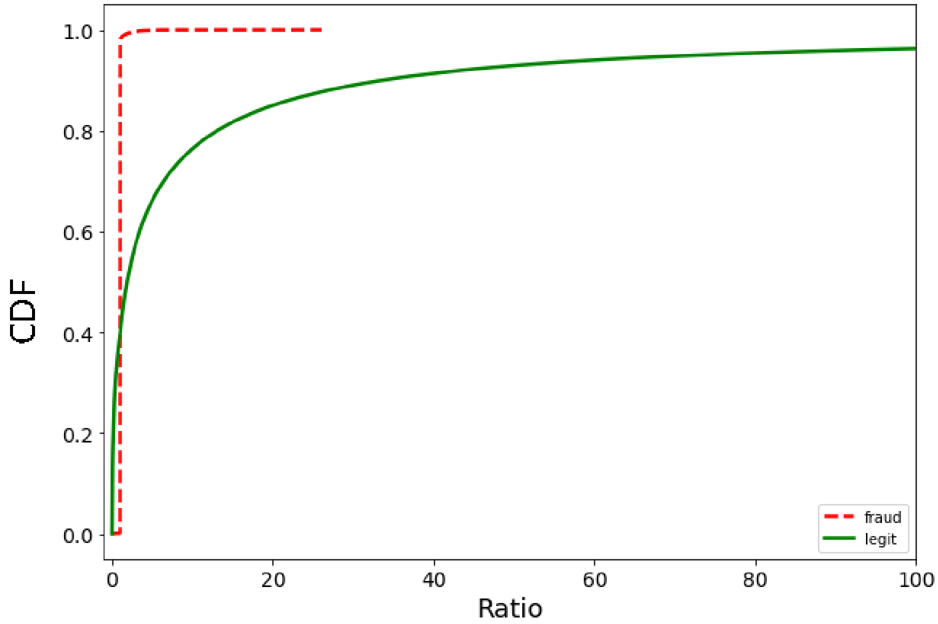


Figure 3. Empirical Cumulative Distribution Function (ECDF) graph of the ratio of the sender of money's initial balance to the transaction volume, comparing this ratio between fraudulent and legitimate transactions.

An Empirical Cumulative Distribution Function (ECDF) is an estimator of the Cumulative Distribution Function (CDF), which describes the probability distribution of a variable by specifying the probability that the value of the variable is less than or equal to a given value. The graph can be read in a similar way to the Gaussian (normal) distribution. For example, in Figure 1, the variable is the ratio described in the caption, which is represented by the x-axis, and the CDF is represented on the y-axis. Thus, if we wanted to find the percentage of legitimate customers (green curve) with the ratio less or equal to 20, we would find the y-value on the green curve when the x-value is 20, which is about 0.85. Therefore, 85% of the legitimate customers have the ratio less or equal to 20. ECDF plots are useful for comparing the distribution of different sets of data, which in our case, helps us differentiate the behavior of fraudulent and legitimate customers, and can be created through the Python library statmodels.

### *Overall Customer Behavior*

The final two features—numTransactions and rangeElapsed— are singular values that were assigned to each customer instead of features relating to individual transactions. The total number of transactions in which a customer took part is described by numTransactions. The number of hours that have passed between a customer's first and last transaction is represented by rangeElapsed. We extracted these features because analyzing a customer's behavior as a whole, rather than only focusing on individual transactions, could provide useful information for spotting suspicious activities. For instance, to avoid raising suspicion, fraudulent customers frequently disperse the money from their victims among several smaller transactions, averaging 8.038 transactions while legitimate customers average 1.406. (Figure 4). If we only considered single transactions, we would have overlooked this information.

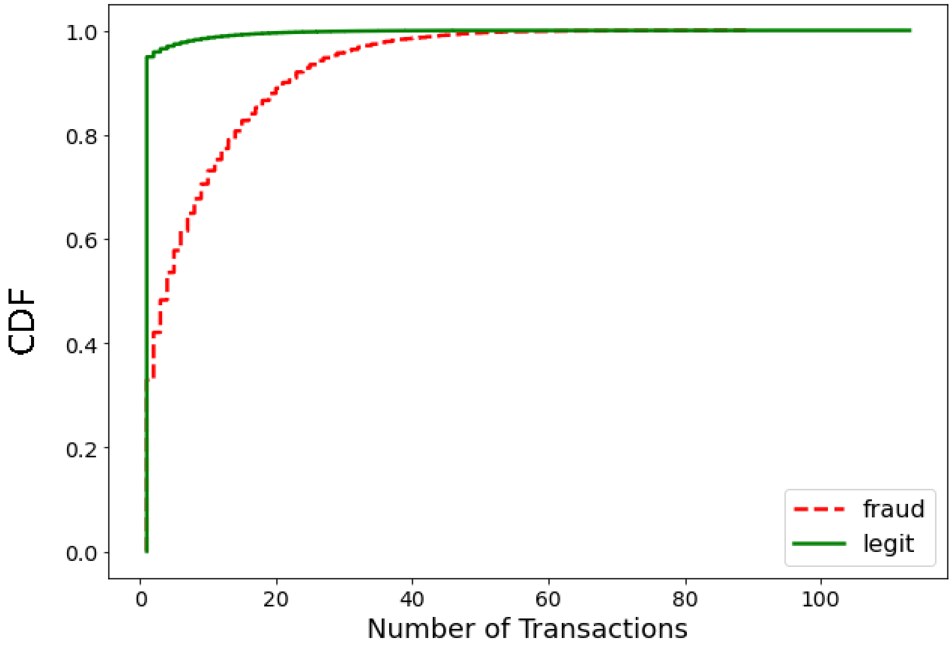


Figure 4. ECDF graph of the number of transactions involved with each customer, comparing fraudulent and legitimate customers.

*Goal*

The goal of this research was to classify fraudulent customers rather than fraudulent transactions. Compared to a customer's entire history of transactions, individual transactions are more difficult to classify because there is significant context to consider while analyzing a customer's behavior. For instance, looking at a customer's transaction history, data such as the number of transactions they dealt in was gathered, a characteristic used to model a customer's behavior.

*System*

In Figure 5, an overview of our machine learning pipeline is shown. First, the PaySim dataset was read, and the pre-processing steps were applied by adding the direction of the transaction and removing rows with incomplete values. Then, for each unique customer, features that distinguish a legitimate customer from a fraudulent customer were extracted into a dataset. A fraudulent agent is one that has engaged in at least one fraudulent transaction. The feature dataset was divided into training and testing using a standard 70-30 distribution. Having completed the preparations to train our model, grid search was performed to refine the parameters of our XGBoost classification model. Grid search exhaustively searched through a specified hyperparameter space of our targeted algorithm to select the model with the best accuracy, our chosen evaluation metric. Finally, our trained model was evaluated using the test dataset that contained labeled instances of fraudulent and legitimate customers.

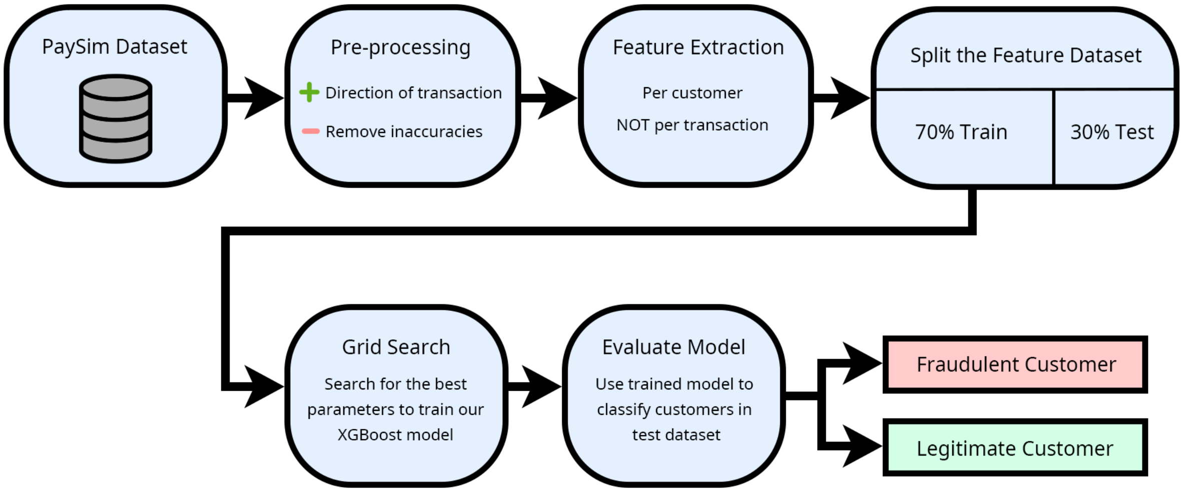


Figure 5. Machine learning pipeline of our approach, showing the system in which our XGBoost model was implemented

## Results

Our machine learning model executed with an accuracy of 99.9315%, with a total of 2720838 correct predictions and 1865 incorrect predictions on whether a customer was legitimate or fraudulent. After conducting k-fold cross-validation, a technique that estimates the accuracy of our model on unseen data, the resulting score was 72.76%. K-fold cross-validation divides the dataset into K subsets of approximately equal size. Our model is then trained and evaluated K times, each time using a different fold as the validation set and the remaining K-1 folds as the training set. We chose 10 as our k value, a commonly used value as it generally results in an estimate with low bias and modest variance.

### *ROC curve*

The receiver operating characteristic (ROC) curve is a graphical plot that illustrates the classifying capability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR). The true positive rate, also known as recall and sensitivity, is the number of true positive predictions made by the model, divided by the total number of positive examples in the dataset. The false positive rate is the number of false positive predictions made by the model, divided by the total number of negative examples in the dataset. As observed from our model's ROC curve (Figure 6), the ROC AUC (Area Under ROC Curve) score is 0.979, indicating that it is making accurate positive predictions for fraudulent customers with few false positive predictions.

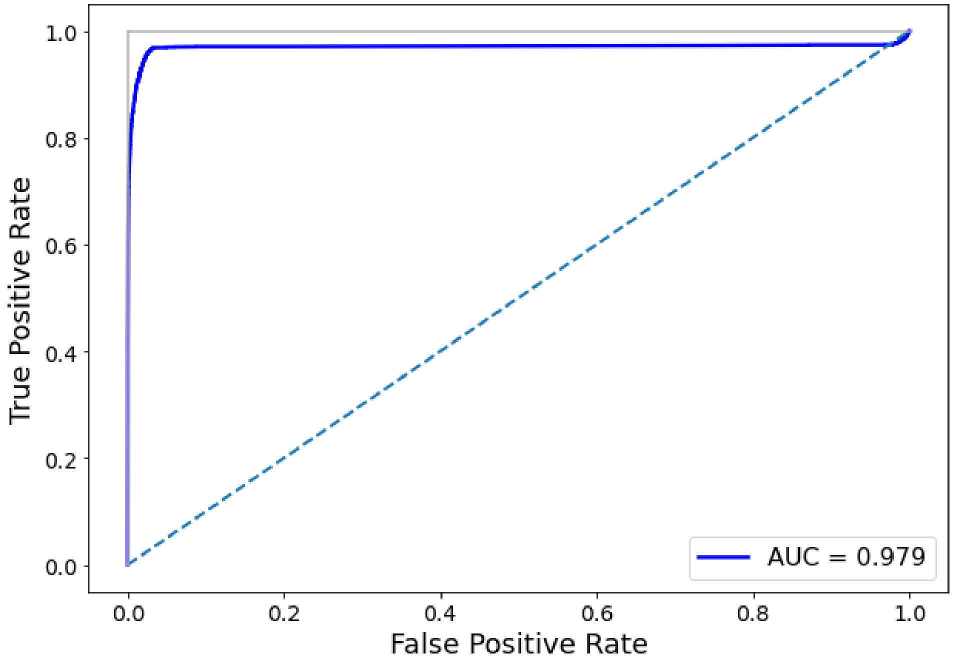


Figure 6. ROC curve of our XGBoost model, showing its performance through plotting its true positive rate against its false positive rate (ROC AUC score of 0.979).

*Confusion matrix*

The confusion matrix shows the number of correct and incorrect predictions made by the model, organized by the predicted and actual classes. For our binary classification problem where the classes are “positive” for fraudulent customers and “negative” for legitimate customers, our confusion matrix has four entries: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). An overwhelming majority of our predictions fell into the TN class, which is followed by the FP, TP, and FN classes respectively (Figure 7). This was evidently due to the imbalanced classes as the ratio of legitimate customers to fraudulent customers was over 1000:1. Through the confusion matrix, we were also able to recognize how our model performed with a 0.00099% false positive rate (27 FP out of 2722703 total predictions), meaning extremely few legitimate customers were mistakenly classified as fraudulent. This is important in a real-world context since this ensures minimal customer disruption and backlash while maintaining their security and safety.

*Feature importance*

Feature importance is a value that describes how useful each feature is for making predictions. The value given to each feature is normalized so that the total will always add up to 1. These values were important to consider in our model that had 44 features because they could be used to identify both important and redundant features. The median of newBalanceRatio was the feature in our model that was most beneficial for predicting fraudulent customers. This feature referred to the ratio between the balances of the sender and receiver of money after each transaction. Furthermore, the least impactful feature used to predict fraudulent customers in our model was the median amount of money in their transactions. This feature fell under the Transaction Volume family and represented the median of the amount of money involved in the customer's transactions. Figure 8 shows a graph of the feature importance of all 44 features.

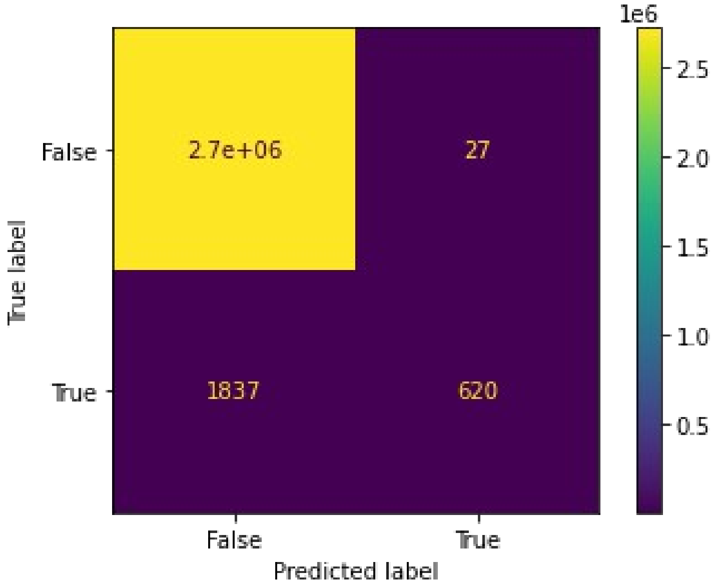


Figure 7. Confusion matrix of our XGBoost model, showing the distribution of positive and false predictions compared to the customers' actual classifications.

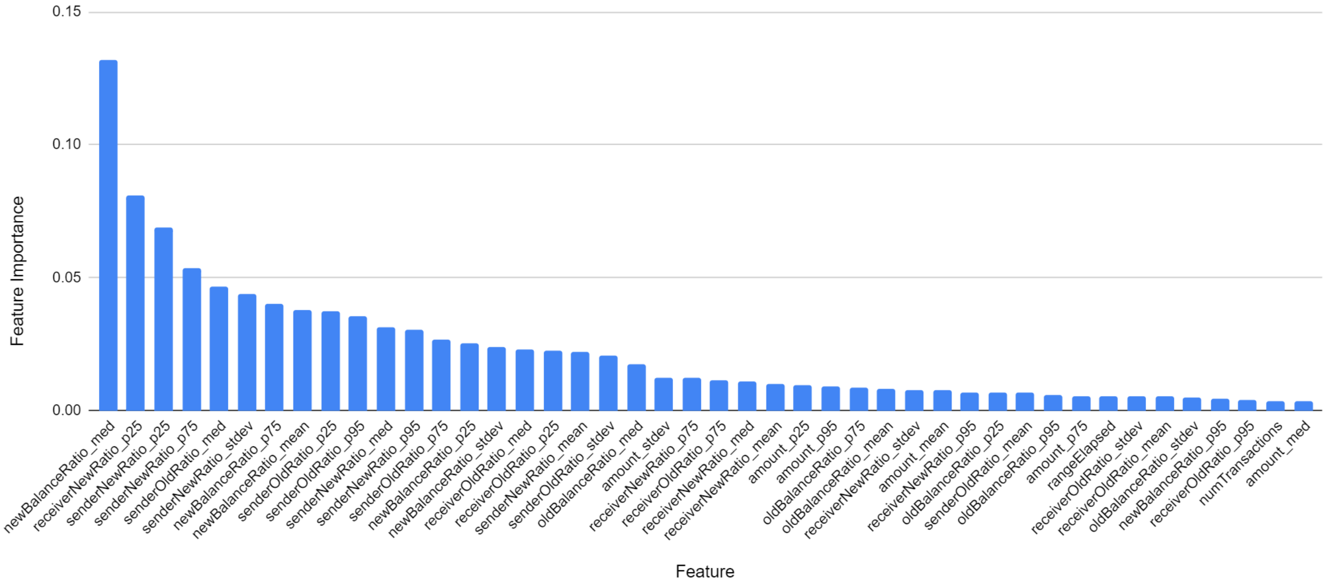


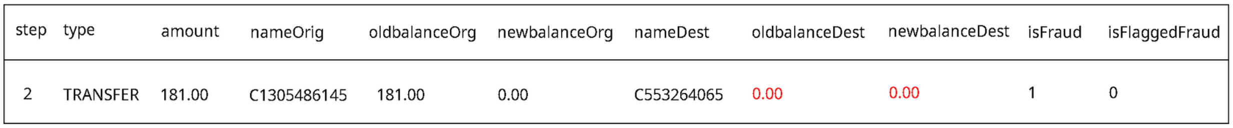
Figure 8. Graph of feature importance for all the features used in our XGBoost model, ranging from 0.131875 (median of newBalanceRatio) to 0.003438 (median of amount).

## Discussion

### *Dataset challenges and limitations*

There were challenges that we faced with our dataset that could have influenced the results. The most apparent problem with the dataset was the very unbalanced classes as of the 6362620 transactions, 6354407 (99.8709%) were legitimate and only 8213 (0.1291%) were fraudulent. Similarly, there were 8169 fraudulent customers compared to 9063043 legitimate customers, a huge disparity. These imbalanced class proportions could have led to inaccurate predictions due to bias towards the abundant legitimate transactions. Future strategies to deal with the unbalanced classes could be to undersample the class with more data (legitimate customers), but it could lead to the loss of potentially important information. This is especially true because with the massive difference in the size of the two classes, the majority class would have to be undersampled to 0.13% of its original size for both classes to have an equal number of instances. Oversampling the minority class (fraudulent customers) is impractical since we have no access to the simulated model, so we also cannot simulate data closer to reality.

Additionally, there were missing values in the balances of customers in 2317276 transactions. In these transactions, both the initial and final balances of a customer were $0.00 while the amount of the transaction was greater than $0.00 (Figure 9).



| step | type | amount | nameOrig | oldbalanceOrg | newbalanceOrg | nameDest | oldbalanceDest | newbalanceDest | isFraud | isFlaggedFraud |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | TRANSFER | 181.00 | C1305486145 | 181.00 | 0.00 | C553264065 | 0.00 | 0.00 | 1 | 0 |

Figure 9. A transaction with missing values for the balances of a customer.

Of the 2317276 transactions, 2151495 transactions were those involving merchants (customers represented by the prefix ’M’) who were all missing their balances. We ensured that empty balances (e.g. an entirely withdrawn bank account) were not misinterpreted as missing values by checking if the amount of the transaction made sense with the balances of the two customers.

There were also limitations in finding a dataset because there is a lack of publicly available financial data. The sensitivity of financial transaction data makes datasets hard to obtain without financial institutions breaking any non-disclosed agreements. Although public data is available through simulations, generating realistic synthetic data sets is difficult and thus, it is hard to validate how fraud detection models would perform in the real world. For example, despite PaySim’s use of MABS, it is important to realize that the dataset does not contain actual transaction data from real individuals and thus, it would differ in terms of specific transaction patterns, volumes, and characteristics.

### *Model challenges*

This section addresses common errors in our model’s incorrect classifications. Two cases where our model incorrectly classified a customer were examined. The first was a false positive case, where the model predicted the customer as legitimate when it was actually fraudulent. The second is a false negative case, where our model predicted the customer as fraudulent when it was actually legitimate.

It can be observed in the transaction history of the false negative case (Table 2) that the customer had an unusually low average amount of money involved in all transactions. The average was $248937.88, which was low when compared to the average for fraudulent transactions of $1467967.30. This difference was almost 80% less than the typical behavior of other fraudulent customers, which could lead our model to mistakenly classify it as legitimate behavior. Another possibility could have been the seemingly legitimate behavior of the customer after its fraudulent transaction. Although fraudulent behavior was clearly evident in their first transaction where they stole the victim's entire balance, the customer engaged in smaller transactions afterwards which seem more legitimate.

Table 2. Transaction history of a false negative case, where our model incorrectly predicted a fraudulent customer (C979594589) as legitimate.

| Amount ($) | Sender | Sender's balance ($) | | Receiver | Receiver's balance ($) | | Fraud |
|---|---|---|---|---|---|---|---|
| | | Initial | Final | | Initial | Final | |
| 548269.98 | C145966586 | 548269.98 | 0.00 | C979594589 | 0.00 | 548269.90 | 1 |
| 116532.17 | C979594589 | 21328.00 | 137860.17 | C623162276 | 548269.98 | 431737.81 | 0 |
| 208728.78 | C979594589 | 82530.22 | 291259.00 | C2084202420 | 431737.81 | 223009.03 | 0 |
| 199111.93 | C979594589 | 49870.00 | 248981.93 | C603437156 | 223009.03 | 23897.10 | 0 |
| 33336.76 | C979594589 | 0.00 | 0.00 | C960842494 | 76321.69 | 109658.45 | 0 |
| 445448.91 | C193132445 | 10596.00 | 0.00 | C979594589 | 23897.10 | 469346.01 | 0 |
| 137286.39 | C701720200 | 17290.22 | 0.00 | C979594589 | 469346.01 | 606632.40 | 0 |
| 165483.92 | C76714408 | 0.00 | 0.00 | C979594589 | 606632.40 | 772116.32 | 0 |
| 191408.96 | C979594589 | 5467250.12 | 5658659.07 | C868195233 | 772116.32 | 769735.70 | 0 |
| 189028.33 | C814845387 | 0.00 | 0.00 | C979594589 | 580707.37 | 769735.70 | 0 |
| 503680.58 | C979594589 | 4413761.77 | 4917442.35 | C758737061 | 769735.70 | 266055.12 | 0 |

Next, we examined the entire history of transactions of the false positive case in Table 3. In this case, a transaction in which the full balance of another customer was transferred to them was carried out by the legitimate customer. This scenario accounted for 97.55% of all fraudulent transactions so it was likely the main cause of our model's failure in this instance. Another possibility stems from the average amount of money involved in the customer's transactions, being $2388768.80 compared to the average of $178197.04 for legitimate customers, a difference of over 170%. This was largely due to the single transaction of $9202694.12, a clear outlier as the average without this transaction was $117460.36, which is much closer to the average for legitimate customers.

Table 3. Transaction history of a false positive case, where our model incorrectly predicted a legitimate customer (C1488784031) as fraudulent.

| **Amount ($)** | **Sender** | **Sender's balance ($)** | | **Receiver** | **Receiver's balance ($)** | | **Fraud** |
|---|---|---|---|---|---|---|---|
| | | **Initial** | **Final** | | **Initial** | **Final** | |
| 9910.02 | C844612280 | 15954.00 | 6043.98 | C1488784031 | 0.00 | 9910.02 | 0 |
| 9202694.12 | C2130293395 | 24869.00 | 0.00 | C1488784031 | 9910.02 | 9212604.14 | 0 |
| 150910.05 | C1488784031 | 8952373.98 | 9103284.03 | C1046970418 | 9212604.14 | 9061694.09 | 0 |
| 191561 | C1488784031 | 9127700.31 | 9319261.31 | C161044779 | 9061694.09 | 8870133.09 | 0 |

In the future, we plan to explore more classification models and compare their overall performance. This includes hyperparameter tuning and Bayesian hyperparameter optimization to find the parameters with the best performance scores while avoiding overfitting. This is important since our model's k-fold cross validation score was lower than the train-test validation accuracy mainly due to overfitting and class imbalance. Although we performed grid search to tune the parameters, overfitting in our XGBoost model could be further avoided by focusing specifically on the subsample and max_depth parameters. The subsample parameter represents the ratio of the training instances used and can reduce variance by averaging trees with high variance, while max_depth represents the maximum depth of a tree and by limiting tree complexity, it can improve generalization to unseen data. To decrease the 1865 incorrect predictions, we would also incorporate a threshold, where specific customers will not be included when training the model. These customers would mainly include outliers like the two previous examples above that do not represent the general patterns of fraudulent customers, and those in the majority class to help prevent the model from being too biased towards the class of legitimate customers. Furthermore, we would be stricter before training the model, ensuring our data is clean and properly preprocessed, as well as eliminating redundant features that do not contribute much to the predictive power of the model. Although removing inaccurate and missing values was done, scaling or normalizing the data would have helped make the data more consistent, and stricter feature selection would reduce noise and enhance the model's focus on essential patterns. Another potential next step is to introduce an additional criterion before classifying a customer, which involves implementing a two-step mechanism. The first step is simply using our current customer-based model to predict the customer as legitimate or fraudulent. The second step is applying a new transaction-based model to make a prediction, so both the customer as a whole and individual transactions are considered. Both values would be taken into account before making a prediction, aiming to enhance accuracy and reduce false predictions. These steps are summarized in Figure 10.

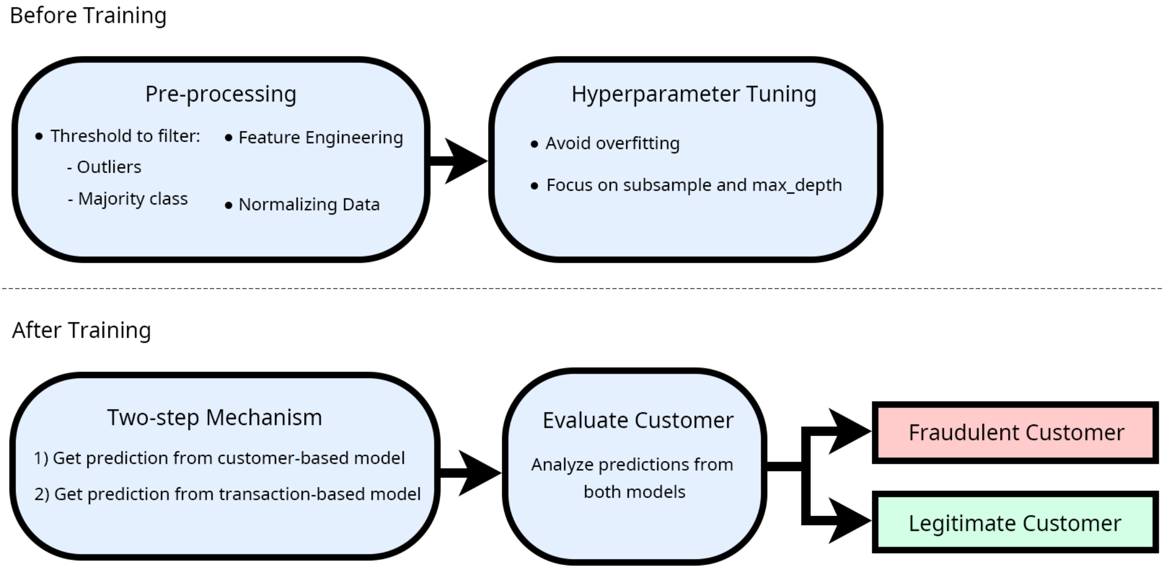


Figure 10. A diagram of potential future steps before and after training to improve the model’s performance.

Moreover, in order to make a solution more suitable to integrate into the real world, we aim to collect additional data, especially real industry data, if possible, to better model the true behavior of fraudulent customers. Extending further, if we could work with a financial institution to obtain a history of financial attacks, we would be able to design a larger system where any fraudulent activity would be stopped and marked for investigation. This would automatically detect the fraudster customers in the early stages of their activity, and it would help to prevent possible additional damage. However, as mentioned earlier with the challenge of obtaining real financial data, even more synthetic datasets could help the model make a more informed decision.

## Conclusion

Overall, we analyzed a dataset of financial transactions that included 9071212 customers with a total of 6362620 transactions. We captured 4 family of features that each contributed towards a different angle of capturing the behavior of a customer. For example, we examined the behavior of customers as a sender compared to a receiver of money through the family of Sender and Receiver Account Balance. This enabled us to observe notable differences in the balances between the two customers in a transaction. We also investigated Overall Customer behavior, a family of features that captures the behavior of customers as a whole instead of examining their individual transactions. By exploring these different angles, we tried to avoid any gaps in our modeling of the behavior. Our approach suggests a machine learning based system to model financial customer's overall behavior and detect financial fraud. The design and implementation of our system allowed the model to perform with high accuracy (99.9315%) and a low false positive rate (0.00099%). As our approach looks at customers instead of individual transactions, it could potentially be used to detect entire fraudulent networks and protect customers and consumers from financial fraud.

## References

1. Beaver, William H. “Financial Ratios As Predictors of Failure”. *Journal of Accounting Research*, vol. 4, [Accounting Research Center, Booth School of Business, University of Chicago, Wiley], 1966, pp. 71–111, http://doi.org/10.2307/2490171.

2. Altman, Edward I. “Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy”. *The Journal of Finance*, vol. 23, no. 4, [American Finance Association, Wiley], 1968, pp. 589–609, http://doi.org/10.2307/2978933.

3. Etaher, Najla, et al. “From ZeuS to Zitmo: Trends in Banking Malware”. *2015 IEEE*, vol. 1, 2015, pp. 1386–1391, http://doi.org/10.1109/Trustcom.2015.535.

4. Tajalizadehkhoob, Samaneh & Asghari, Hadi & H. Ganan, Carlos & Eeten, Michel. (2014). Why Them? Extracting Intelligence about Target Selection from Zeus Financial Malware.

5. Alazab, Mamoun & Venkatraman, Sitalakshmi & Watters, Paul & Alazab, Moutaz & Alazab, Ammar. (2011). Cybercrime: The Case of Obfuscated Malware. Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering. 99. http://doi.org/10.1007/978-3-642-33448-1_28.

6. Prasad, Balaji, and Nupur Maheshwari. “Botnets — Secret Puppetry with Computers .” University of Arizona, 2012, https://www2.cs.arizona.edu/~collberg/Teaching/466-566/2013/Resources/presentations/2012/topic11-final/report.pdf

7. Unisys Stealth Solution Team. Zeus Malware: Threat Banking Industry, May 2010, https://botnetlegalnotice.com/citadel/files/Guerrino_Decl_Ex1.pdf

8. Ryan, M. (2021). Ransomware Case Studies. In: Ransomware Revolution: The Rise of a Prodigious Cyber Threat. Advances in Information Security, vol 85. Springer, Cham. https://doi.org/10.1007/978-3-030-66583-8_5

9. Schwartz, Evan I. "Digital Cash Payoff." MIT Technology Review, Dec. 2001, www.technologyreview.com/2001/12/01/235339/digital-cash-payoff/.

10. Cox, Paul. "PayPal and FBI Team Up To Combat Wire Fraud." The Wall Street Journal, Dow Jones & Company, June 2001, www.wsj.com/articles/SB992639123888198275.

11. LexisNexis. LexisNexis® Risk Solutions 2019 TRUE COST OF FRAUDTM Study, 2019, https://risk.lexisnexis.com/about-us/press-room/press-release/20220802-true-cost-of-fraud-study

12. Allison, Chelsea. "PayPal's History of Fighting Fraud." Fin, Mar. 2019, https://fin.plaid.com/articles/paypals-history-of-fighting-fraud/

13. Carvalho, Ralf, and Alex Paino. "Deep Learning for Fraud Detection." Sift Engineering Blog, Mar. 2018, https://engineering.sift.com/deep-learning-fraud-detection/

14. Sift. "About Sift." *About Sift | Digital Fraud Detection and Protection*, 2022, https://sift.com/about

15. Sift. Fraud Detection Software for Secure Growth, 2023, https://sift.com/products/digital-trust-safety-platform

16. Sift. "DoorDash Case Study." Sift Resources, 2023, https://resources.sift.com/case-studies/doordash/

17. Dheepa, V., and R. Dhanapal. "Behavior Based Credit Card Fraud Detection Using Support Vector Machines". *Soft Computing Models in Industrial and Environmental Applications*, 2012, http://doi.org/10.21917/IJSC.2012.0061.

18. A. Srivastava, A. Kundu, S. Sural and A. Majumdar, "Credit Card Fraud Detection Using Hidden Markov Model," in IEEE Transactions on Dependable and Secure Computing, vol. 5, no. 1, pp. 37-48, Jan.-March 2008, http://doi.org/10.1109/TDSC.2007.70228.

19. Ajeet Singh, Anurag Jain, Seblewongel Esseynew Biable, "Financial Fraud Detection Approach Based on Firefly Optimization Algorithm and Support Vector Machine", Applied Computational Intelligence and Soft Computing, vol. 2022, Article ID 1468015, 10 pages, 2022. https://doi.org/10.1155/2022/1468015

20. Y. X. She, “Firefly algorithms for multimodal optimization,” in Proceedings of the 5th symposium on stochastic algorithms, foundations and applications, vol. 5792, pp. 169–178, Sapporo, Japan, 2009.

21. Pambudi, Bayu, et al. “A Minimum Error-Based PCA for Improving Classifier Performance in Detecting Financial Fraud”. *Jurnal Teknik Elektro*, vol. 14, no. 1, 2022, pp. 1–9, http://doi.org/10.15294/jte.v14i1.35787.

22. Lopez-Rojas, Edgar Alonso. “Applying Simulation to the Problem of Detecting Financial Fraud”. 2016, https://bth.diva-portal.org/smash/record.jsf?pid=diva2%3A955852&dswid=5931

23. Seyedhossein, Leila, and Mahmoud Reza Hashemi. “Mining Information from Credit Card Time Series for Timelier Fraud Detection”. *2010 5th International Symposium on Telecommunications*, 2010, pp. 619–624, http://doi.org/10.1109/ISTEL.2010.5734099.

24. Shuntaro Takahashi, Yu Chen, Kumiko Tanaka-Ishii, Modeling financial time-series with generative adversarial networks, Physica A: Statistical Mechanics and its Applications, Volume 527, 2019, 121261, ISSN 0378-4371, https://doi.org/10.1016/j.physa.2019.121261.

25. Qiu, Chen, et al. ‘Raising the Bar in Graph-Level Anomaly Detection’. ArXiv [Cs.LG], 2022, http://arxiv.org/abs/2205.13845.

26. Xuting Mao, Hao Sun, Xiaoqian Zhu, Jianping Li, Financial fraud detection using the related-party transaction knowledge graph, Procedia Computer Science, Volume 199, 2022, Pages 733-740, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.01.091.

27. Weber, Mark, et al. “Scalable Graph Learning for Anti-Money Laundering: A First Look”. arXiv, 2018, doi.org/10.48550/ARXIV.1812.00076.
28. Jarrod West, Maumita Bhattacharya, Intelligent financial fraud detection: A comprehensive review, Computers & Security, Volume 57, 2016, Pages 47-66, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2015.09.005.

29. Xiaoqian Zhu, Xiang Ao, Zidi Qin, Yanpeng Chang, Yang Liu, Qing He, Jianping Li, Intelligent financial fraud detection practices in post-pandemic era, The Innovation, Volume 2, Issue 4, 2021, 100176, ISSN 2666-6758, https://doi.org/10.1016/j.xinn.2021.100176.