Mobile Payment Fraud Detection

Table of Contents

Executive Summary

In this project, we applied logistic regression, random forest, and XGBoost on simulated transaction data in order to improve the PaySim's ability to detect fraudulent transactions. All three models yield high accuracy in detecting fraud payment. We considered logistic regression as the best model for the simulated data given its lower time cost. Real-world transaction data are required to improve model performance and choose the best algorithm.

Introduction

Our client PaySim is a company specializing in mobile payments. With its intelligent technology for instant authentication and authorization, online payment is quickly infiltrating people's daily lives. People no longer have to fill out lengthy forms or go through complicated log-in procedures for every transaction. The risks are accompanied by benefits. PaySim, as a member of the finance industry, is inevitably vulnerable to payment fraud. Because of the ease with which transactions can be completed, it has become an even more popular target for fraudulent activity. Therefore, Paysim has been dedicated to fraud detection in order to assuage customers' safety concerns and shoulder the responsibility of building a secure financial system. PaySim currently detects fraud payment using a simple cut-off of the transaction amount – any transactions with a transaction amount of more than 200,000 are flagged as illegal attempts. PaySim approached us, a fraud detection agency, for help in better detecting fraudulent activities.

Problem Formulation

In this project, we define fraudulent behavior as illegal attempts by agents to profit by seizing control of customers' accounts and trying to empty the funds by transferring them to

another account and then cashing out of the system. Our team used machine learning techniques such as simple regression, random forest, and XGBoost, and ultimately came up with a detection algorithm that outperforms the existing method.

Data Description

PaySim did not provide us with real transaction data at this early stage of coordination due to privacy concerns. Instead, it simulates the mobile money transactions based on a sample of real transactions extracted from one month of financial logs from its service implemented in an African country.

The dataset contains 6,362,619 transaction records, and the following 12 features:

- *step*: 1 step represents 1 hour of time. There are 744 steps recorded in total, Since the dataset contains one-month (31 days) data.
- *type* - types of transaction, including CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.
- *amount* - amount of the transaction in local currency.
- *nameOrig* - name of the applicant, customer who started the transaction.
- *oldbalanceOrg* - initial balance of the applicant before the transaction
- *newbalanceOrig* - new balance of the applicant after the transaction.
- *nameDest* - name of the recipient, customer who received the money
- *oldbalanceDest* - initial balance of the recipient before the transaction. Note that there is not information for customers that start with M (Merchants).
- *newbalanceDest* - new balance of the recipient after the transaction. Note that there is not information for customers that start with M (Merchants).

- *isFraud* - label 1 when the transaction was made by the fraudulent agents inside the simulation, and label 0 when the transaction was legal.
- isFlaggedFraud - An illegal attempt in this dataset is an attempt to transfer more than 200.000 in a single transaction. Suspected illegal transactions will be labelled 1 otherwise 0.

There are several interesting findings as we conducted the exploratory analysis of the dataset before modeling. First of all, the current approach of using a *cut-off of $200,000* in transaction amount to identify fraud payment has low performance. It catches only 16 fraud and fails to capture 8200 fraud cases.

Secondly, the simulation data are clean and ready for analysis. There are no missing values in the dataset and the data types of all the variables are reasonable.

Thirdly, among the 6,362,619 observations, only 8,216 of them, which is approximately 0.13% are fraudulent transactions, making it challenging for improving model accuracy.

Fourthly, *type* is one attribute that is worth attention. There are only two types of transactions falling under the fraud category -- Transfer and Cash-out which are listed in *Figure 1*.

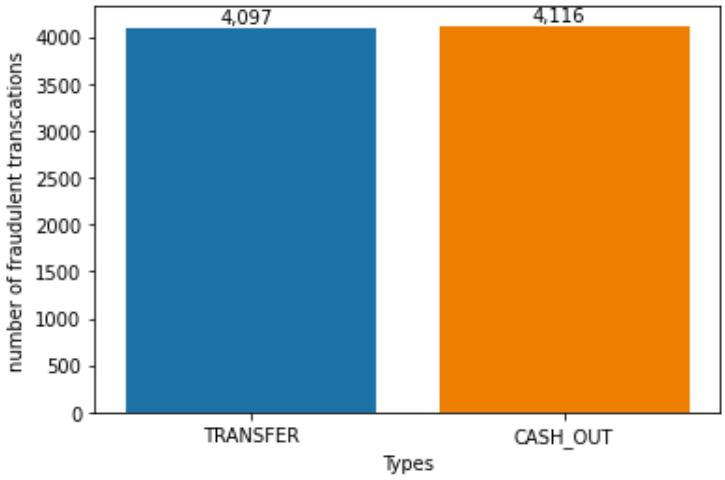

Figure 1: Types of Fraudulent Transactions

In addition, *amount*, *oldbalanceOrg*, and *oldbalanceDest* are three other critical features for fraud detection. Opportunities and challenges exist at the same time in these variables. From the boxplots shown in *Figure 2*, there are a lot of outliners for these three variables. Actions should be taken to tackle this problem. We cannot simply delete the outliers as we may do in other cases, because they are significant signals of fraudulent attempts. We can apply log transformation, specifically, log(x+1), since there are some 0 presented in the variables. Log transformation can effectively spread the data which originally clustering on the bottom.

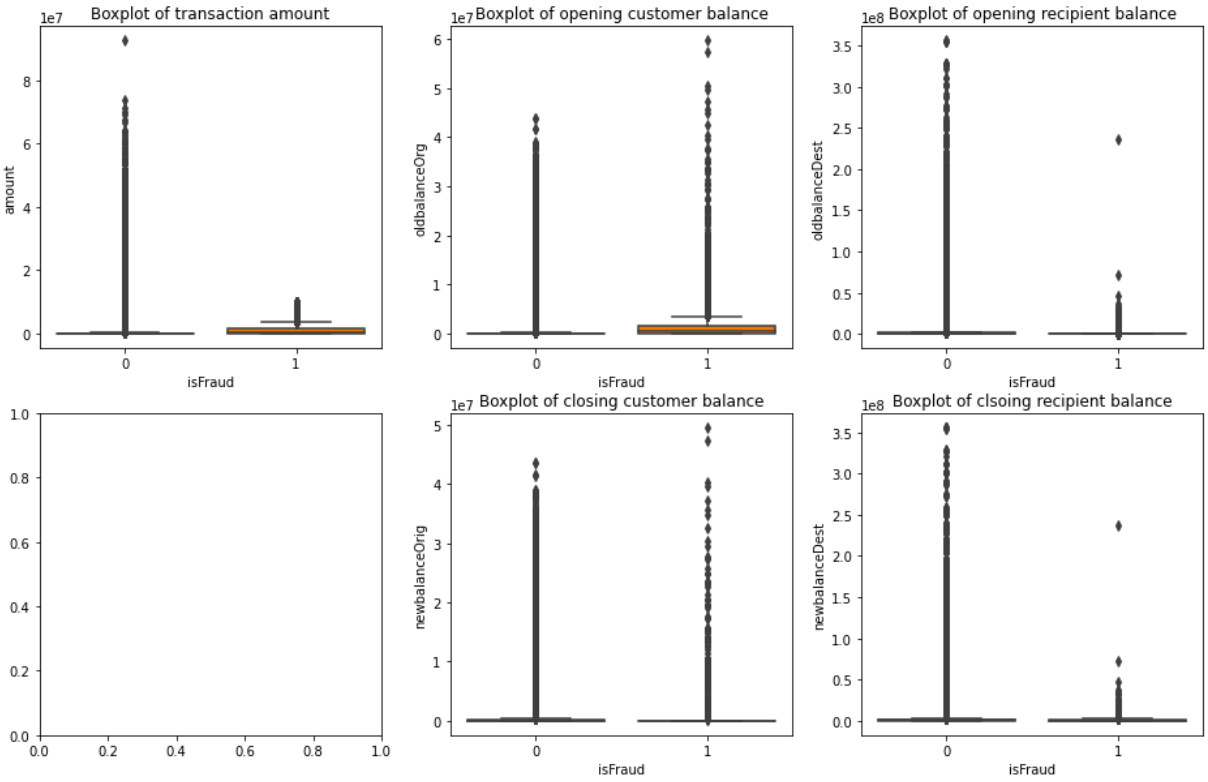

Figure 2: Boxplots of Transaction Amount and Balance

Furthermore, we drilled down transaction amount, customer opending balance, and recipient openning balance by two transaction types – CASH OUT and TRANSFER, as shown in *Figure 3*. We chose median as the measurement rather than mean to circumvent the effect of

outliers. We noticed that transaction amount and customer opening balance of CASH OUT are abnormally high for fraud transactions compared to non-fraud transactions, while that for recipient opening balance is the opposite.

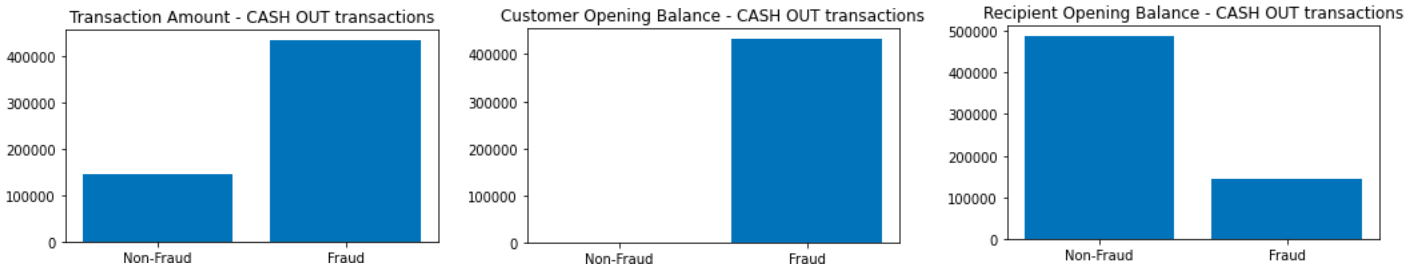

Figure 3: Transaction Amount and Opening Balance of CASH-OUT - Non-Fraud vs Fraud

Lastly, we observed some patterns of fraudulent activities. In *Figure 4*, we presented the number of fraud transactions over hours of the whole month. It is easy to notice that all the peaks sit close to each other, indicating a high frequency of fraudulent transactions.

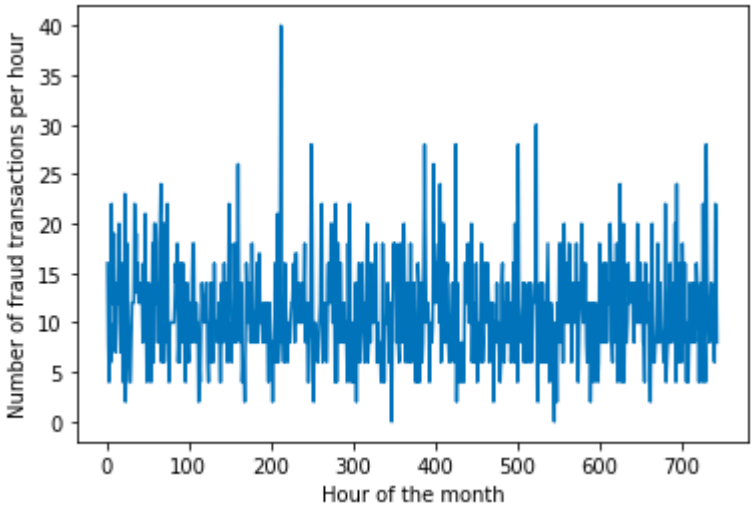

Figure 4: Number of Fraud Transactions Per Hour

We derived the *day* variable using the *step* variable. *Figure 5* below shows the fraud rate at different hours of day. Fraud transactions happened most often at late night. More than 20% of the transactions that happen between 4 am and 5 am are fraud transactions.

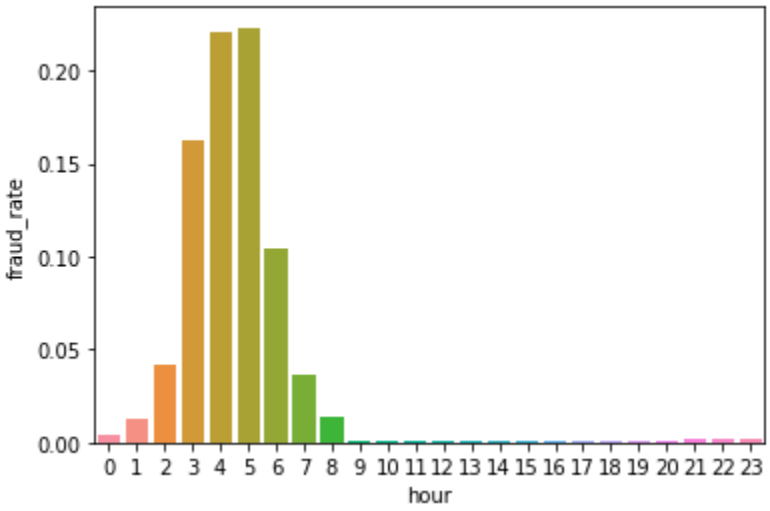

Figure 5: Fraud Rate Per Hour

Model Development, Estimation, and Results

We implemented three different classification models to detect if a transaction is fraudulent or not. We split our dataset into 80% as the training and 20% as the test dataset. We used the 20% test as the out-of-sample model testing dataset. Firstly, we started with a simple binary logistic regression model. To evaluate the model, we used the True Positive Rate (TPR) because we care about the ability to detect fraudulent transactions and the cost of failing to detect the transaction can cause huge problems. Next, we implemented the Random Forest classifier to train our model. In the end, we ran the XGBoost (Extreme Gradient Boost) to train and test the results. For logistic regression, we got TPR=0.79 approx, 0.78 for random forest model and 0.68 for XGBoost. We found that there are 1691 fraud transactions in the testing dataset of 1,272,523

observations. Logistic regression successfully identified 1298 true positives. Using random forest, we got 1291 and in XGBoost, we got 1121 fraudulent transactions. All three approaches perform much better than the existing approach which detected only 16 fraudulent payments out of 8,000 fraudulent transactions in the entire dataset. Logistic regression win over random forest and XGBoost by a small margin in TPR. It is highly likely that there is some overfitting happening in the complex models such as Random Forest and XGBoost. Though the TPRs of logistic regression and random forest are almost the same, we still prefer because the computation power required for logistic regression is much lower. The running time for logistic regression is 43 seconds while that for the random forest is 22 minutes. The computation time of the three algorithms is shown in *Table 1*.

Table 1: Computation Time of Three Classification Algorithms

Algorithm	Time taken (Hour:Minutes:Seconds:Microseconds)
Logistic Regression	0:00:43.644460
Random Forest	0:22:52.167307
XGBoost	0:34:14.542411

Recommendation and Managerial Implications

PaySim should switch to a data-driven approach for more accurate fraud detection. The three popular classification algorithms, logistic regression, random forest, XGBoost performed much better than the current model. Even though transaction amount is an important attribute in terms of detecting fraudulent activities, 200,000 may not be a good threshold. Except for transaction amount, we should also take applicant and recipient opening balance, and step into consideration.

However, all three models give similar TPR in simulation data, making it difficult to choose the best model. For the simulation dataset, we recommend logistic regression as the best model given less computation cost. We need to get the real transaction data to test and train the model in real-world settings. We will follow the rule of starting with the simplest logistic regression and keep adding complexity into the model from there.

Reference

[1] Synthetic Financial Datasets For Fraud Detection dataset from Kaggle

https://www.kaggle.com/ealaxi/paysim1