Synthetic Financial Datasets For Fraud Detection

Introduction/Problem statement:

Due to the private nature of financial data, there aren't many publicly accessible datasets for analysis. The dataset used in this project was generated using a simulator called PaySim, which is publicly available on Kaggle. A multinational mobile financial services company's private dataset was used to generate the dataset. Financial companies are looking for ways to detect fraudulent transactions so that customers are not charged for items they did not purchase. The purpose of this project is to implement machine learning models to detect fraudulent transactions.

Dataset source and information:

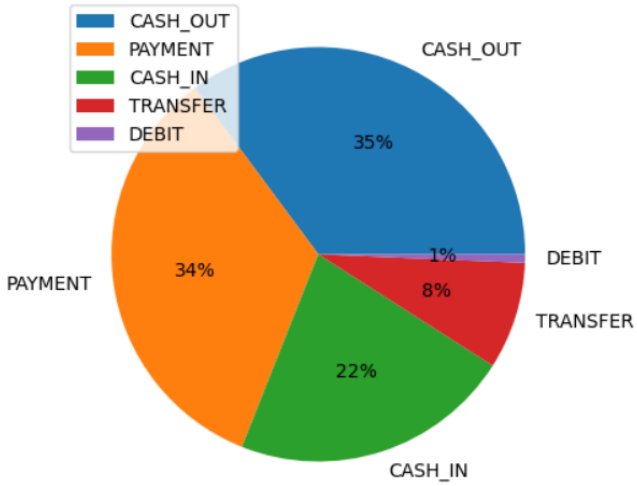
This synthetic dataset is generated by the PaySim mobile money simulator from Kaggle. The dataset has over 6 million transactions and 11 variables. There is a variable named 'isFraud' that indicates actual fraud status of the transaction. The columns in the dataset are described as follows:

- step - maps a unit of time in the real world. In this case 1 step is 1 hour of time. Total steps 744 (30 days simulation).
- type - CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.
- amount - amount of the transaction in local currency.
- nameOrig - customer who started the transaction
- oldbalanceOrg - initial balance before the transaction
- newbalanceOrig - new balance after the transaction
- nameDest - customer who is the recipient of the transaction
- oldbalanceDest - initial balance recipient before the transaction. Note that there is not information for customers that start with M (Merchants).
- newbalanceDest - new balance recipient after the transaction. Note that there is not information for customers that start with M (Merchants).
- isFraud - This is the transactions made by the fraudulent agents inside the simulation. In this specific dataset the fraudulent behavior of the agents aims to profit by taking control or customers accounts and try to empty the funds by transferring to another account and then cashing out of the system.
- isFlaggedFraud - The business model aims to control massive transfers from one account to another and flags illegal attempts. An illegal attempt in this dataset is an attempt to transfer more than 200.000 in a single transaction.

Data Wrangling and EDA

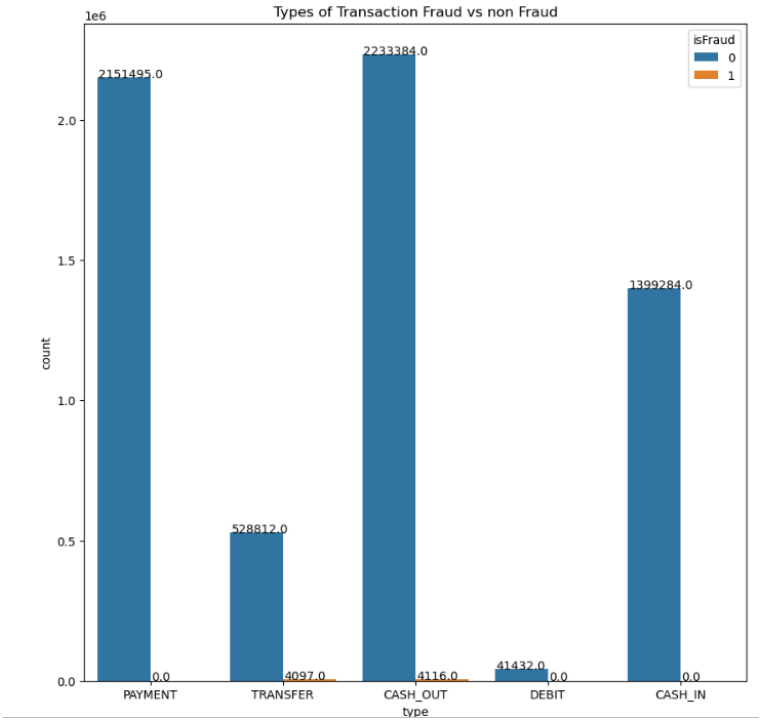
To begin with, I did an initial inspection of the dataset, and there aren't any duplicate or missing values. Surprisingly, there is only 16 fraudulent transactions within 6 million records under isFlaggedFraud column. It turns out that Fraud column has 8213 entries that are marked as yes. 16 out of 6362620 entries seems to be odd. Then I would like to

learn more about what types of transactions: there are 5 types of transactions, cash_out 35%, followed by payment 34%, then cash_in 22%, then transfer 8%, lastly, debit transaction 1%.



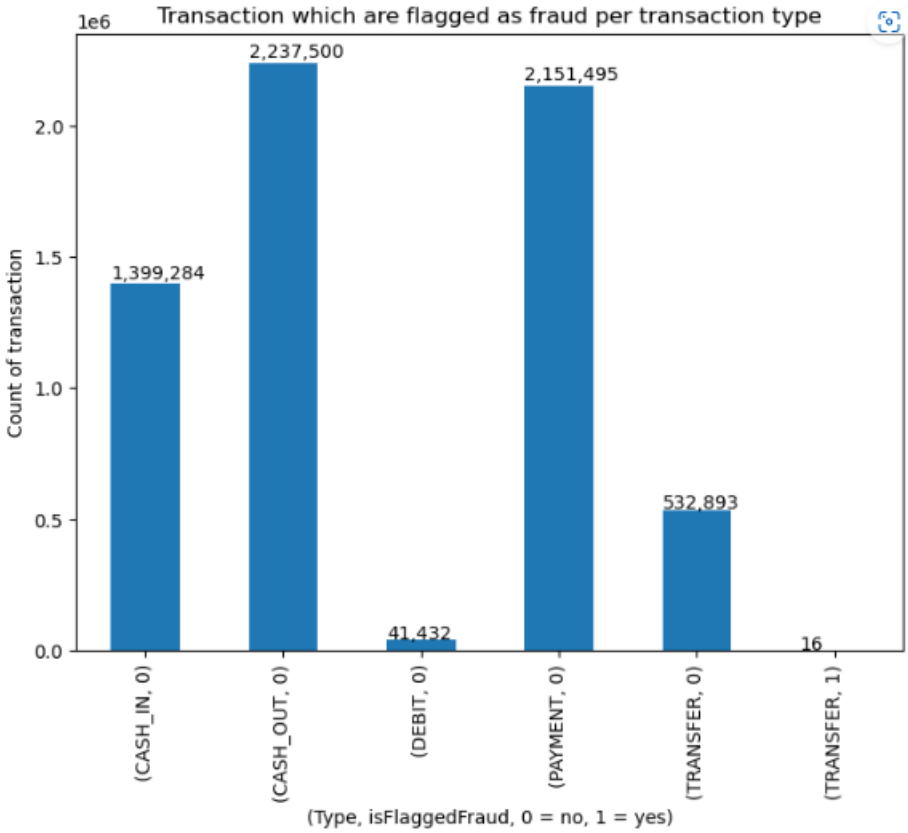
Type of transactions

I also plotted a graph to show the difference types of fraud transaction vs non fraud transaction. I found out fraud transaction only occur in transfer and cash_out transactions (see graph below).

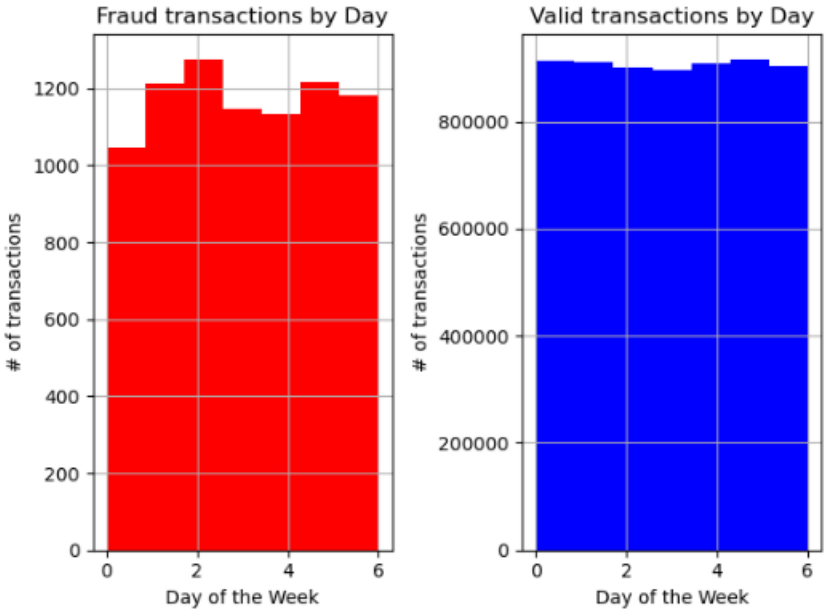


Furthermore, there are no specific accounts from which fraud transactions are carried out. For fraudulent transactions, the account that received funds during a transfer was not used at all for cashing out. It doesn't seem nameDest and NameOrig have any fraudulent transactions.

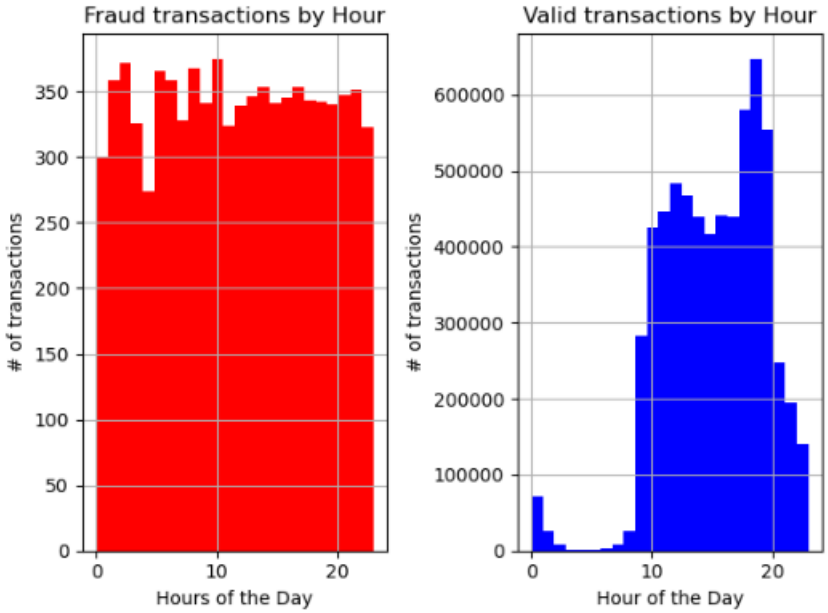
Another thing I found out is that we have 4097 Fraud Transfers and there are only 16 were flagged by the system. 16 out of 6 millions transaction were flagged by the system. It's safe to say that this feature may not be significant for us to do modeling (graph is provided below).



Another thing to take into account is the time, I did inspect different timings when the transactions occurred, it seems the fraud and valid transaction occur pretty much at the similar rate during the week.



I looked into details to inspect transaction in hours, the valid transactions very rarely occur from hours 0 to 9. Also, fraudulent transactions still occur at similar rates to any hour of the day even outside of hours 0 to 9. As a result, the hours may be a good feature to include in the modeling.



Feature Engineering & Machine Learning:

The target variable in this dataset is the fraud column, and I removed a couple of columns from modeling as they don't seem to be related to the target variable during EDA process. Then, I filtered out the transaction types because I saw only fraudulent

transactions in the transfer and cash_out transaction types. HourOfDay was added to the dataset set.

The following three classification models from scikit-learn were used to evaluate the best classification model: Logistic Regression, Random Forest Classifier, and XGBoost. The data was split into 70%/30% training/testing sets and stratified on the 'Fraud' feature to ensure an equal percentage of fraud samples in each set, respectively.

Evaluation

The table below is my initial modeling and it seems like the Logistic give us the better ROC AUC score, however, the f1 score is the lowest one. I will use ROC/AUC score and PR AUC as our chosen metric to evaluate the model.

Model	ROC/AUC score	F1 score
Logistic	0.9910	0.6114
Random Forest	0.9153	0.997
XGBoost	0.9886	0.9039

Since our dataset is imbalanced, I use synthetic minority oversampling technique (SMOTE) and under sampling from Random Under Sampler package.

Model	ROC/AUC score	F1 score	PR AUC
Logistic	0.9293	0.9322	0.9776
Random Forest	0.9993	0.9993	0.9999
XGBoost	0.9985	0.9985	0.9999

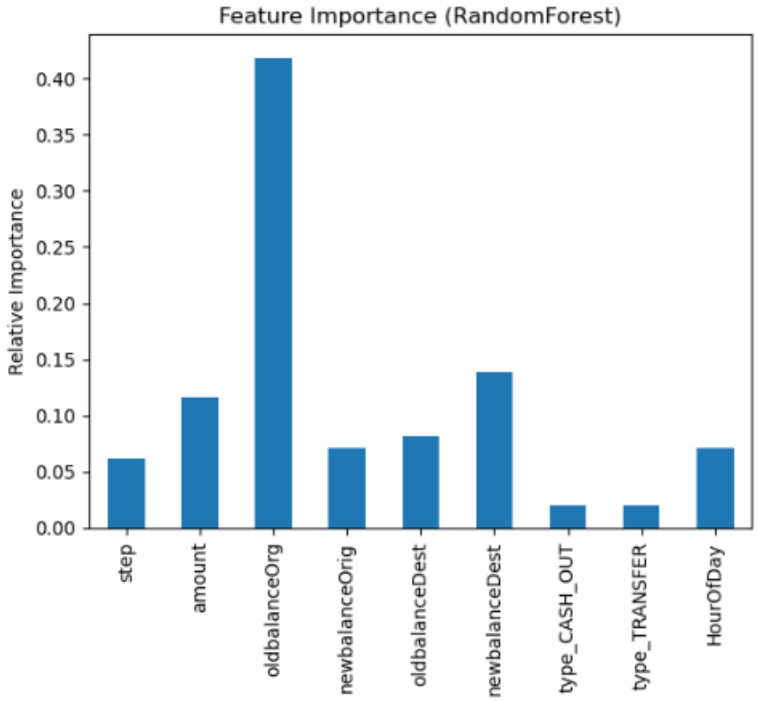
SMOTE over sampling to deal with imbalanced data

Model	ROC/AUC score	F1 score	PR AUC
Logistic	0.9383	0.9398	0.9788
Random Forest	0.9878	0.9878	0.9989
XGBoost	0.9919	0.9919	0.9994

Under sampling to deal with imbalanced data

I will pick the Random Forest Classifier from SMOTE oversampling because 1) ROC&AUC score is the highest . ROC AUC is especially good at ranking predictions. 2) PR AUC is also high like 99%. PR AUC is usually sensitive to positive class, especially since our dataset is imbalanced.

Below are the features that are relatively important to the final model. oldbalanceOrg feature (initial balance before the transaction) is ranked as the highest one, followed by the oldbalanceDest (initial balance recipient before the transaction), and amount (amount of the transaction in local currency).



Limitation/Future Research:

This analysis was a look into an imbalanced data problem. It was also an insightful exploration of some popular classification models. One of the limitations was that the dataset was generated using a simulator called PaySim; Hence, it had a high AUC score to begin with, and it might not be feasible to see in the actual business data. However, we can apply the same concept and approach when handling imbalanced data. It will also be interesting to look into more features such as location, online banking, and currency value in the future..