Financial Fraud Detection using Machine Learning Techniques

By

Harpreet Shoker

Abstract

Payments related fraud is a key aspect of cyber-crime agencies and recent research has shown that machine learning techniques can be applied successfully to detect fraudulent transactions in large amounts of payments data. Such techniques have the ability to detect fraudulent transactions that human auditors may not be able to catch and also do this on a real time basis.

In this project, we apply multiple supervised machine learning techniques to the problem of fraud detection using a dataset available at kaggle about simulated payment transactions data. We aim to demonstrate how supervised ML techniques can be used to classify data with high class imbalance with high accuracy.

We demonstrate that exploratory analysis can be used to separate fraudulent and non- fraudulent transactions. We also demonstrate that for a well separated dataset, algorithms like Random Forest and XGBoost work much better than Logistic Regression.

Table of Contents

Chapter 1

1.1 Introduction

Digital payments of various forms are rapidly increasing across the world. Payments companies are experiencing rapid growth in their transactions volume. For example, PayPal processed ~$578 billion in total payments in 2018. Along with this transformation, there is also a rapid increase in financial fraud that happens in these payment systems.

Preventing online financial fraud is a vital part of the work done by cybersecurity and cyber-crime teams. Most banks and financial institutions have dedicated teams of dozens of analysts building automated systems to analyze transactions taking place through their products and flag potentially fraudulent ones. Therefore, it is essential to explore the approach to solving the problem of detecting fraudulent entries/transactions in large amounts of data in order to be better prepared to solve cyber-crime cases.

1.2 Aims and Objectives

This project is an effort to develop a framework of fraud detection in financial transactions. We hope the outcome of the project will help streamline the analysis and detection of fraudulent transactions.
Overall, there are three main objectives of the project :

- To study the literature on financial fraud detection and understand the different aspects of the problem.
- To solve the problem of financial fraud detection on a publicly available sample dataset using supervised machine learning techniques.
- To compare different classification techniques to understand which is best suitable for this application.

  Ultimately, the creation of a framework and codes that incorporate analytics and machine learning concepts studied in the program is the goal. The success of the project is predicated on the accuracy of the classification results and the extent of analysis conducted. We hope the final report will serve as a benchmark for further development on this topic and as a knowledge base for students to understand the nuances of fraud detection.

1.3 Research Methodology

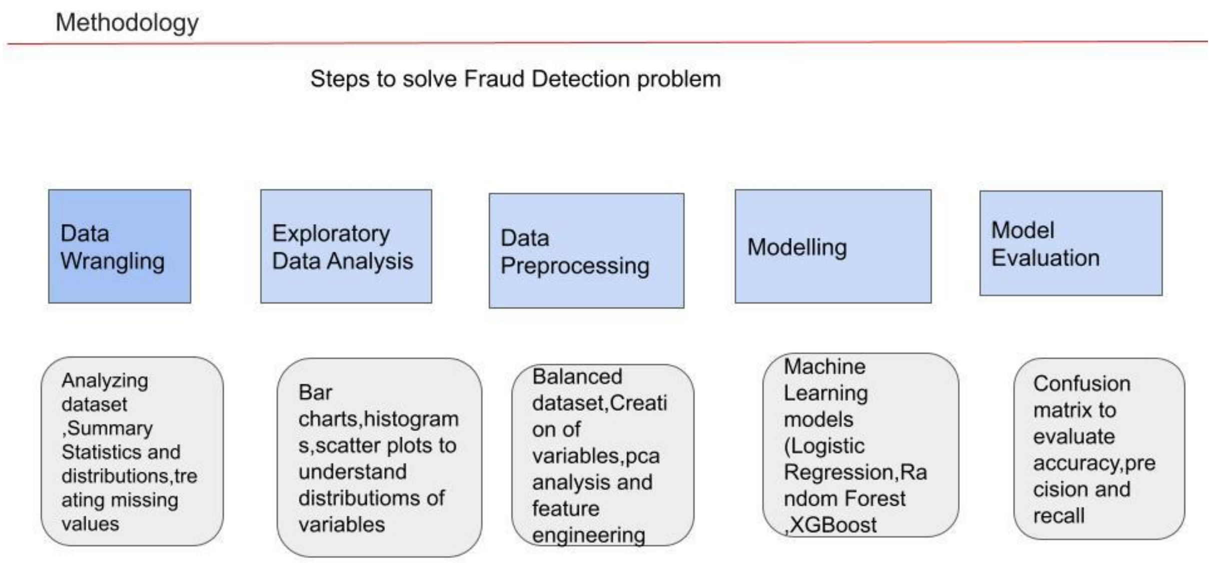
The typical machine learning approach was followed in this project. The identified dataset has labelled class variable, which was used as the prediction variable in machine learning models.

- Through exploratory analysis, we analyzed the data set in detail and identified possible predictors of fraud.
- Through various visualization techniques, we observed the separation between fraud and non-fraud transactions.
- To solve the fraud detection problem, we experimented with supervised machine learning techniques – Logistic Regression,Random Forest and XGBoost. Additionally, we also tried under-sampling to address the class imbalance in the dataset.
- The models were developed with cross-validation to avoid overfitting and obtain consist of performance.
- Performance measures, like Confusion Matrix and Area Under Curve (AUC), was used to compare the performance of the models.

This analysis was conducted using Python through Jupyter notebook. In-built libraries and methods were used to run the machine learning models. When needed, functions were defined to simplify specific analyses or visualizations.

The below diagram shows in detail the full process that was followed in the project.

Figure 1: Project Methodology


Methodology
Steps to solve Fraud Detection problem
Data Wrangling
Exploratory Data Analysis
Data Preprocessing
Modelling
Model Evaluation
Analyzing dataset ,Summary Statistics and distributions,treating missing values
Bar charts,histograms,scatter plots to understand distributioms of variables
Balanced dataset,Creation of variables,pca analysis and feature engineering
Machine Learning models (Logistic Regression,Random Forest ,XGBoost
Confusion matrix to evaluate accuracy,precision and recall

1.4 Limitations of the Study

In this study, we evaluated the effectiveness of using specific supervised machine learning techniques to solve the problem of fraud detection in financial transactions. The limitations of the methods applied in this study are as follows:

- We used a pre-labeled dataset to train the algorithms. However, usually, it is difficult to find labelled data and thus applying supervised machine learning techniques may not be feasible. In such cases, we should evaluate unsupervised techniques which were beyond the scope of this study. This study considers digital transactions data that includes amount transacted, the balance of recipient and originator, and time of transaction.
- These variables that helped in detecting fraud may not apply to types of financial transactions, such as credit card fraud. We evaluated two machine learning algorithms – Logistic Regression,Random Forest and XGBoost. Although the result of the study using these algorithms is good, it is necessary to evaluate other techniques to determine which algorithm works best for this application.
- Due to the large size of data, we were limited by computation capacity to explore different techniques such as grid search for parameter tuning,

Chapter 2

2.1 Methodology

This methodology served as the deliverables of the project. It describes the results of each phase that was tried out and do a comparison between them to identify which is the best technique to address the fraud detection problem.

Each phase of the project has an output that describes the findings in that phase. These deliverables were used in this final project are explained below –

Table 2: Project Deliverables

Methodology Phases	**Project Deliverables**
Data Wrangling	Report on the summary of the data set and each variable it contains along with necessary visualisations
Exploratory Data Analysis	● Report on analysis conducted and critical findings with a full description of data slices considered ● Hypothesis about the separation between fraud and non- fraud transactions. ● Visualisations and charts that show the differences between fraud and non-fraud transactions
Data Preprocessing	Preparing the data for modelling.Applied Principal Component Analysis,Feature Engineering and split data into training and test data.
Modelling	Report on the results of the different techniques tried out, iterations that were experimented with, data transformations and the detailed modeling approach.Python code used to build machine

	learning models .
Evaluation and Report	Final report summarizing the work done over the course of the project, highlighting the key findings, comparing different models and identifying best model for financial fraud detection

2.2 Tools Used

This project was entirely done using Python, and the analysis was documented in a Jupyter notebook. Standard python libraries were used to conduct different analyses. These libraries are described below –

sklearn – used for machine learning tasks
seaborn – used to generate charts and visualizations
pandas – used for reading and transforming the data

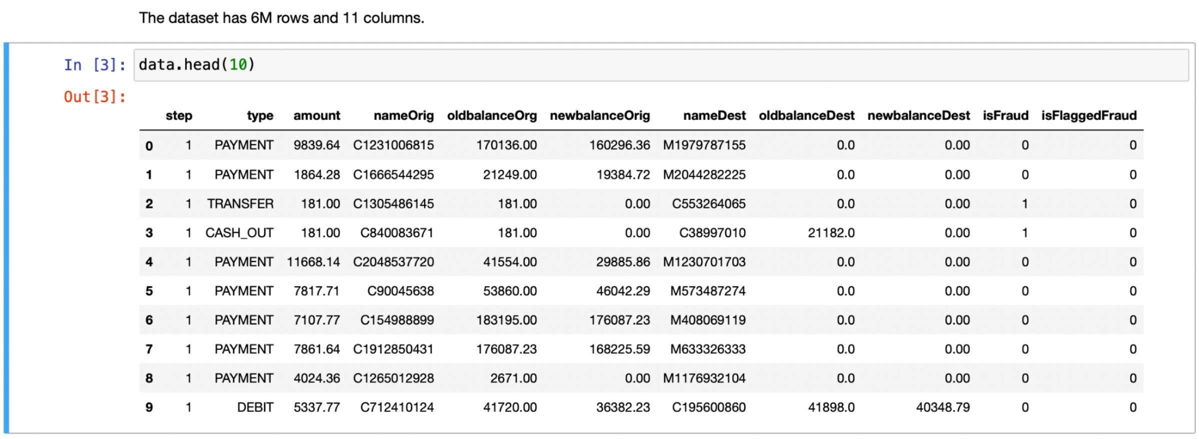
2.3 Data Sources

Due to the private nature of financial data, there is a lack of publicly available datasets that can be used for analysis. In this project, a synthetic dataset, publicly available on Kaggle, generated using a simulator called PaySim is used. The dataset was generated using aggregated metrics from the private dataset of a multinational mobile financial services company, and then malicious entries were injected. (TESTIMON @ NTNU, Kaggle).

The dataset contains 11 columns of information for ~6 million rows of data. The key columns available are –

Type of transactions
Amount transacted
Customer ID and Recipient ID
Old and New balance of Customer and Recipient
Time step of the transaction
Whether the transaction was fraudulent or not

In the following figure, a snapshot of the first few lines of the dataset is presented.



The dataset has 6M rows and 11 columns.

In [3]:
```
data.head(10)
```

Out[3]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	PAYMENT	9839.64	C1231006815	170136.00	160296.36	M1979787155	0.0	0.00	0	0
1	1	PAYMENT	1864.28	C1666544295	21249.00	19384.72	M2044282225	0.0	0.00	0	0
2	1	TRANSFER	181.00	C1305486145	181.00	0.00	C553264065	0.0	0.00	1	0
3	1	CASH_OUT	181.00	C840083671	181.00	0.00	C38997010	21182.0	0.00	1	0
4	1	PAYMENT	11668.14	C2048537720	41554.00	29885.86	M1230701703	0.0	0.00	0	0
5	1	PAYMENT	7817.71	C90045638	53860.00	46042.29	M573487274	0.0	0.00	0	0
6	1	PAYMENT	7107.77	C154988899	183195.00	176087.23	M408069119	0.0	0.00	0	0
7	1	PAYMENT	7861.64	C1912850431	176087.23	168225.59	M633326333	0.0	0.00	0	0
8	1	PAYMENT	4024.36	C1265012928	2671.00	0.00	M1176932104	0.0	0.00	0	0
9	1	DEBIT	5337.77	C712410124	41720.00	36382.23	C195600860	41898.0	40348.79	0	0

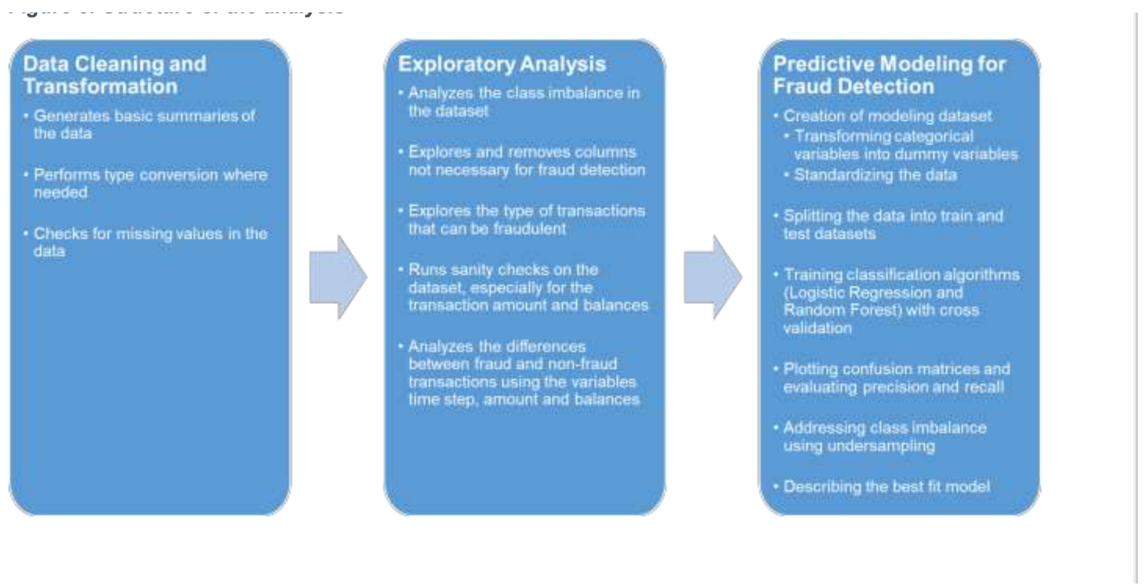
Chapter 3

3.1 Data Analysis

This section describes each step of the analysis conducted in detail. All analysis is documented in Jupyter notebook format, and the code is presented along with the outputs.

The analysis is split into three main sections. These are described in the diagram below.

Figure 3: Structure of the analysis


Data Cleaning and Transformation
• Generates basic summaries of the data
• Performs type conversion where needed
• Checks for missing values in the data
Exploratory Analysis
• Analyzes the class imbalance in the dataset
• Explores and removes columns not necessary for fraud detection
• Explores the type of transactions that can be fraudulent
• Runs sanity checks on the dataset, especially for the transaction amount and balances
• Analyzes the differences between fraud and non-fraud transactions using the variables time step, amount and balances
Predictive Modeling for Fraud Detection
• Creation of modeling dataset
• Transforming categorical variables into dummy variables
• Standardizing the data
• Splitting the data into train and test datasets
• Training classification algorithms (Logistic Regression and Random Forest) with cross validation
• Plotting confusion matrices and evaluating precision and recall
• Addressing class imbalance using undersampling
• Describing the best fit model

3.2 Detailed Analysis

The following pages show the step by step process followed in executing the mentioned analysis structure. Relevant code snippets and graphics included are based on Python programming language.

3.2.1 Data Cleaning

This section describes the data exploration conducted to understand the data and the differences between fraudulent and non-fraudulent transactions.

3.2.1.1 Data Description

The data used for this analysis is a synthetically generated digital transactions dataset using a simulator called PaySim. PaySim simulates mobile money transactions based on a sample of real transactions extracted from one month of financial logs from a mobile money service implemented in an African country. It aggregates anonymized data from the private dataset to generate a synthetic dataset and then injects fraudulent transactions.

The dataset has over 6 million transactions and 11 variables. There is a variable named 'isFraud' that indicates actual fraud status of the transaction. This is the class variable for our analysis.

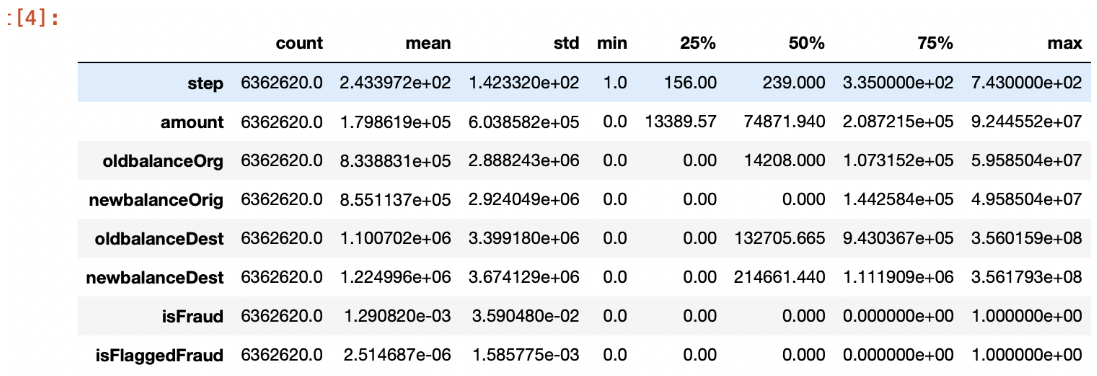
The columns in the dataset are described as follows:

step	maps a unit of time in the real world. In this case 1 step is 1 hour of time
	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
newbalanceOrig	initial balance recipient before the transaction.
newbalanceDest	new balance recipient after the transaction
isFraud	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

3.2.1.2 Summary Statistics

Before proceeding with the analysis, we present the summary statistics of the variables. In case of numeric variables, we evaluate the mean, standard deviation and the range of values at different percentiles. In case of categorical variables, we evaluate only the number of unique categories, the most frequent category and its frequency.

Figure 5: Summary of Statistics of Numeric Variables



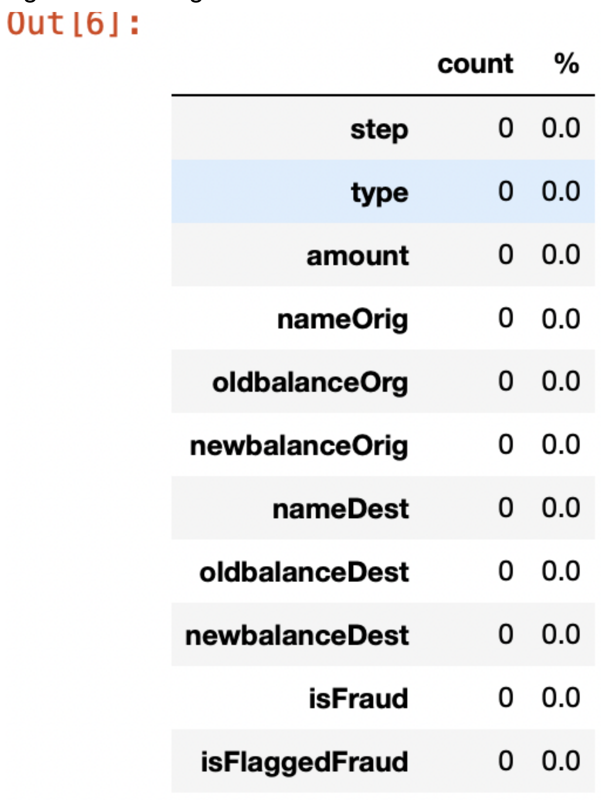
:[4]:

	count	mean	std	min	25%	50%	75%	max
step	6362620.0	2.433972e+02	1.423320e+02	1.0	156.00	239.000	3.350000e+02	7.430000e+02
amount	6362620.0	1.798619e+05	6.038582e+05	0.0	13389.57	74871.940	2.087215e+05	9.244552e+07
oldbalanceOrg	6362620.0	8.338831e+05	2.888243e+06	0.0	0.00	14208.000	1.073152e+05	5.958504e+07
newbalanceOrig	6362620.0	8.551137e+05	2.924049e+06	0.0	0.00	0.000	1.442584e+05	4.958504e+07
oldbalanceDest	6362620.0	1.100702e+06	3.399180e+06	0.0	0.00	132705.665	9.430367e+05	3.560159e+08
newbalanceDest	6362620.0	1.224996e+06	3.674129e+06	0.0	0.00	214661.440	1.111909e+06	3.561793e+08
isFraud	6362620.0	1.290820e-03	3.590480e-02	0.0	0.00	0.000	0.000000e+00	1.000000e+00
isFlaggedFraud	6362620.0	2.514687e-06	1.585775e-03	0.0	0.00	0.000	0.000000e+00	1.000000e+00

3.2.1.3 Missing Values Check

In this phase, we also check if there are any missing values in the dataset. The following code and output indicate the total number of missing / NA values in all columns, which is zero.

Figure 6: Missing Values Check



Out[6]:

	count	%
step	0	0.0
type	0	0.0
amount	0	0.0
nameOrig	0	0.0
oldbalanceOrg	0	0.0
newbalanceOrig	0	0.0
nameDest	0	0.0
oldbalanceDest	0	0.0
newbalanceDest	0	0.0
isFraud	0	0.0
isFlaggedFraud	0	0.0

As per the count per column, we have no null values.

Chapter 4

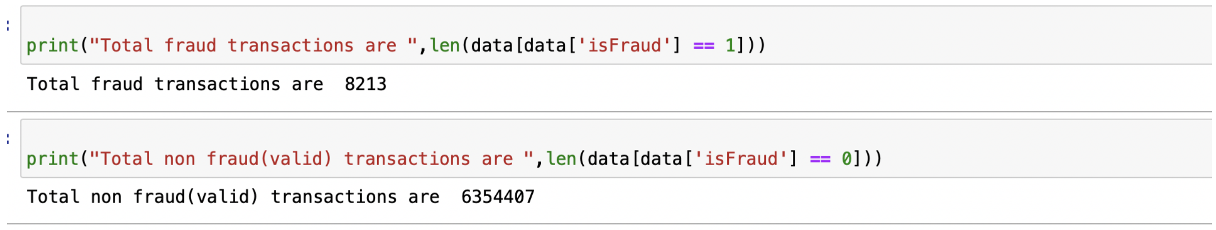
4.1 Exploratory Analysis

4.1.1 Class Imbalance

In this exploratory analysis, we assess the class imbalance in the dataset. The class imbalance is defined as a percentage of the total number of transactions presented in the isFraud column.

The percentage frequency output for the isFraud class variable is shown below:

Figure 7: Class Imbalance



```
print("Total fraud transactions are ",len(data[data['isFraud'] == 1]))
```

```
Total fraud transactions are  8213
```

```
print("Total non fraud(valid) transactions are ",len(data[data['isFraud'] == 0]))
```

```
Total non fraud(valid) transactions are  6354407
```

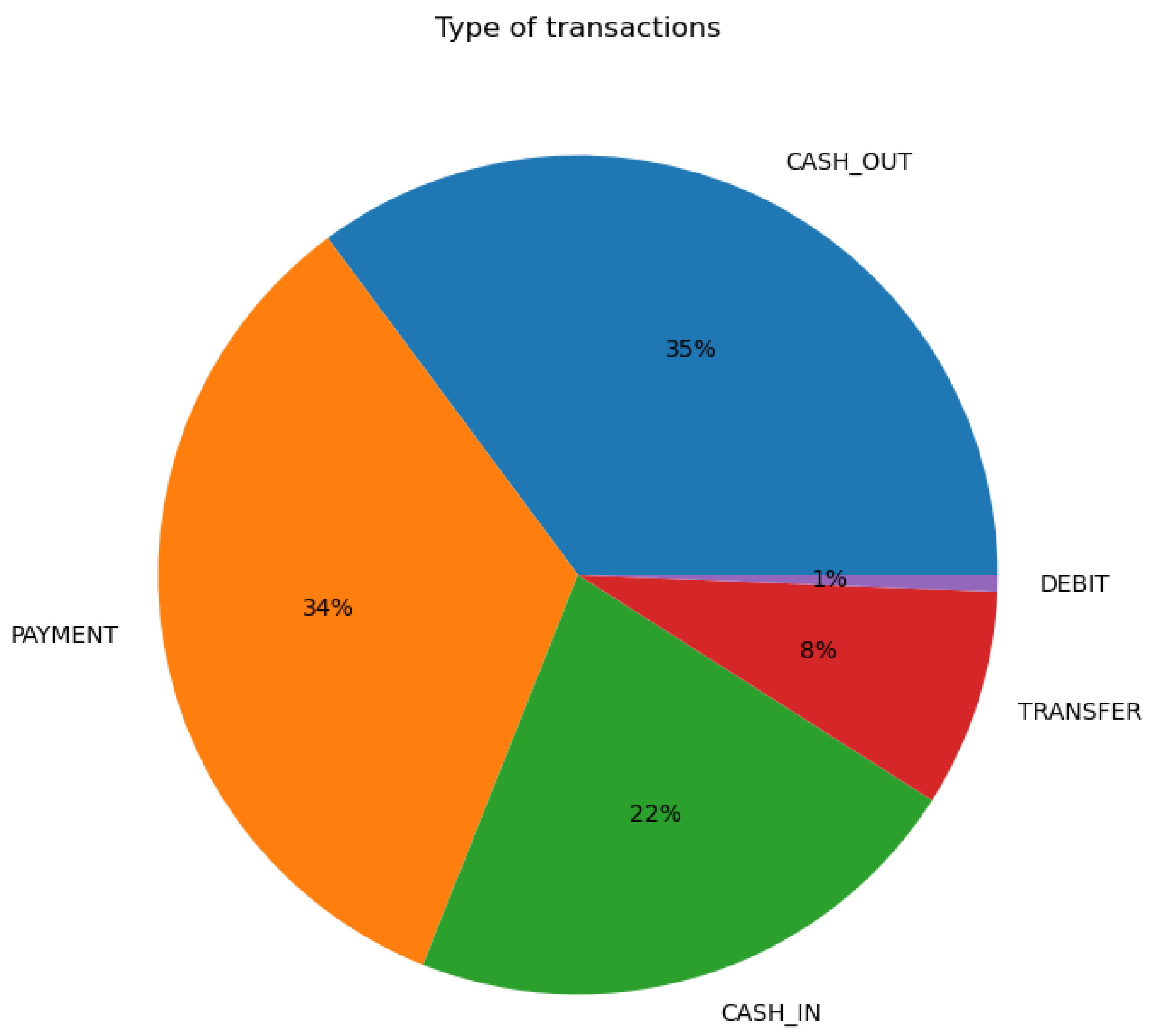
Only 0.13% (8,213) transactions in the dataset are fraudulent indicating high-class imbalance in the dataset. This is important because if we build a machine learning model on this highly skewed data, the non-fraudulent transactions will influence the training of the model almost entirely, thus affecting the results.

4.1.2 Types of Transactions

In this section, we are exploring the dataset by examining the 'type' variable. We present what the different 'types' of transactions are and which of these types can be fraudulent.

The following plot shows the frequencies of the different transaction types.

Figure 11: Frequencies of Transaction Types

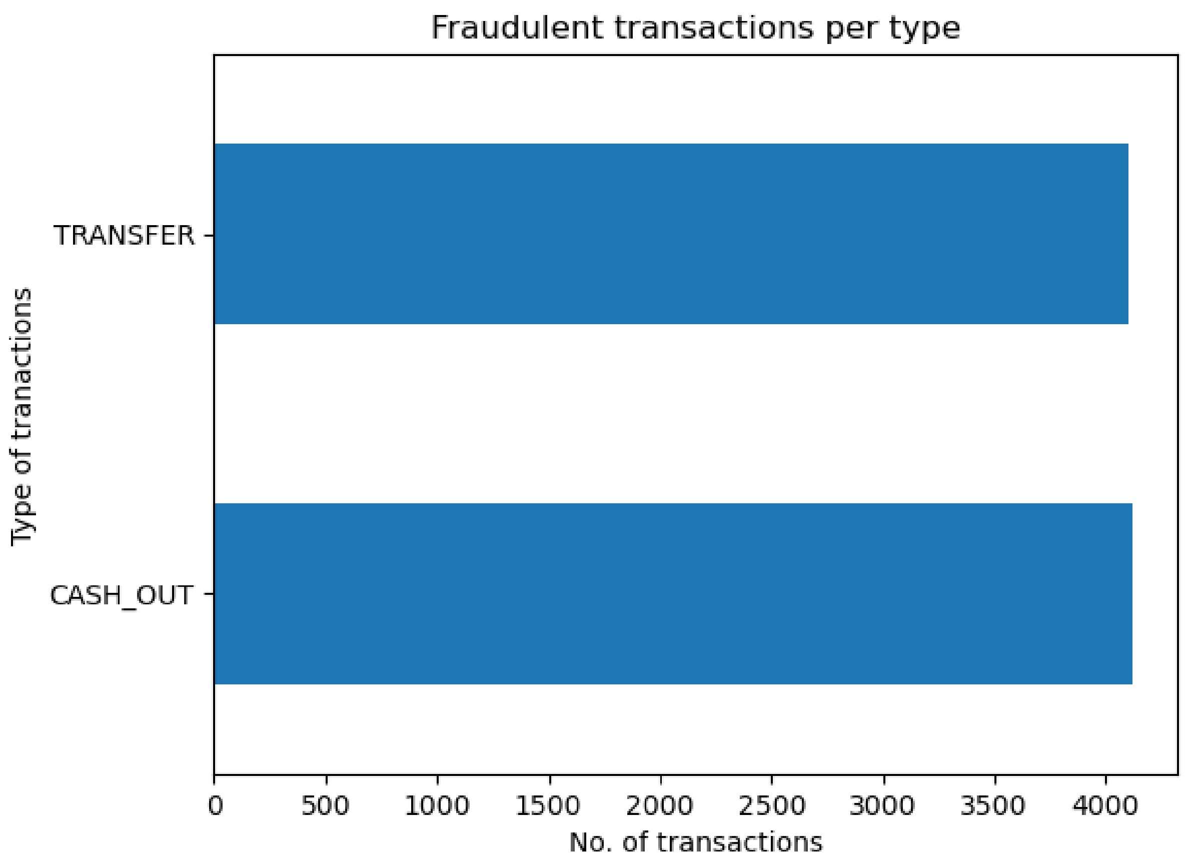


The most frequent transaction types are CASH-OUT and PAYMENT.

From the above possible types of transactions, only cash-out and transfer are considered as fraudulent transactions.

Only CASH-OUT and TRANSFER transactions can be fraudulent. So, it makes sense to retain only these two types of transactions in our dataset.
From figure.12 the fraudulent transactions are splitted in an equal percentage.

Figure 12: Split of Fraud Transactions by Transaction Type



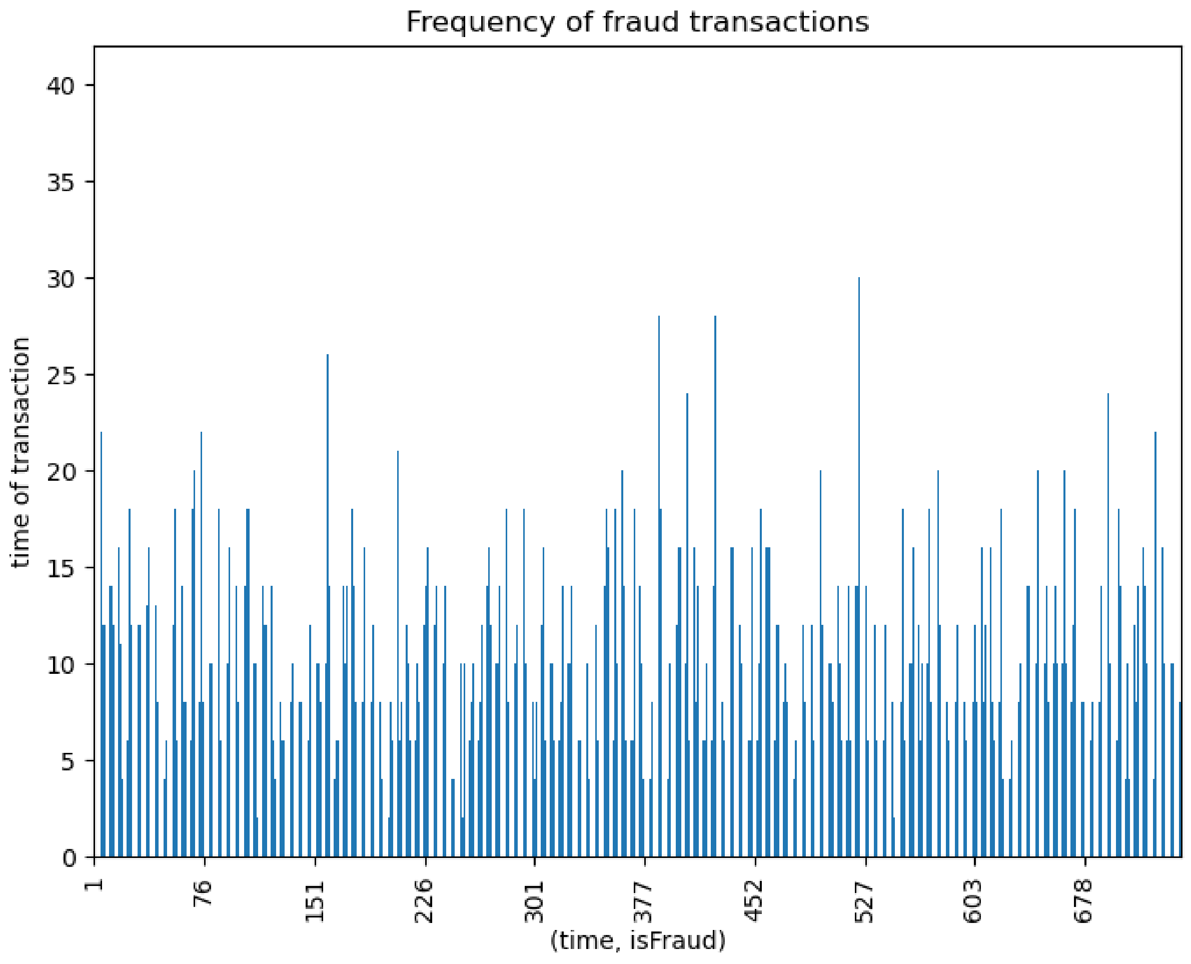
4.1.3 Fraud Transactions Analysis

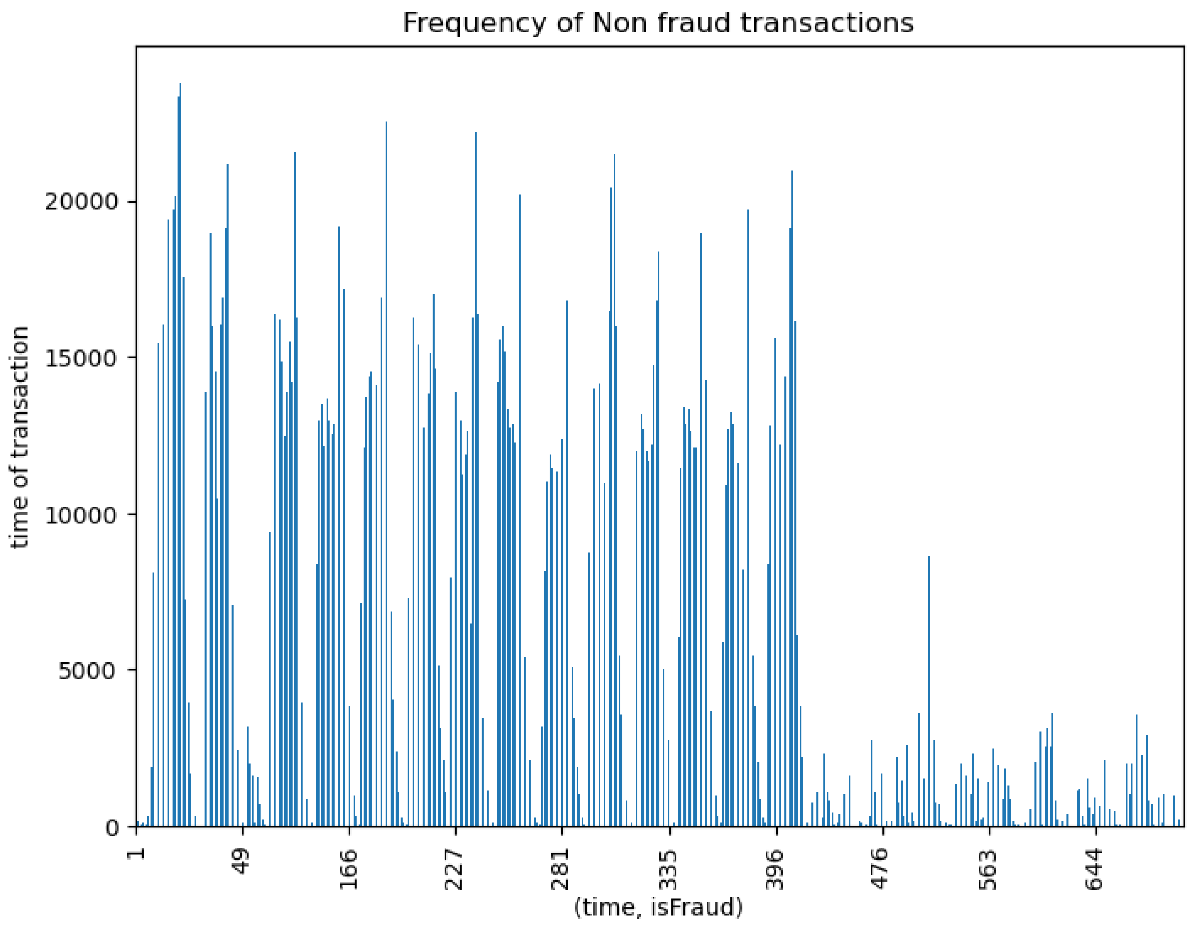
In this section, an additional exploratory analysis is performed to identify if any of the variables can predict a fraud.

Time Step:

We start by analyzing the time step variable. The number of transactions in each time step by fraud status was measured in order to identify if there are any particular time steps where fraudulent transactions are more common than others. From the data description, we know that each time step is an hour.

Figure 13: Fraud and Non-Fraud Transactions Count by Time Step





From Figure.13 show that the fraud transactions are almost uniformly spread out across time steps, whereas non-fraudulent transactions are more concentrated in specific time steps. This could be a differentiator between the two categories and can help in the training of the classification models.

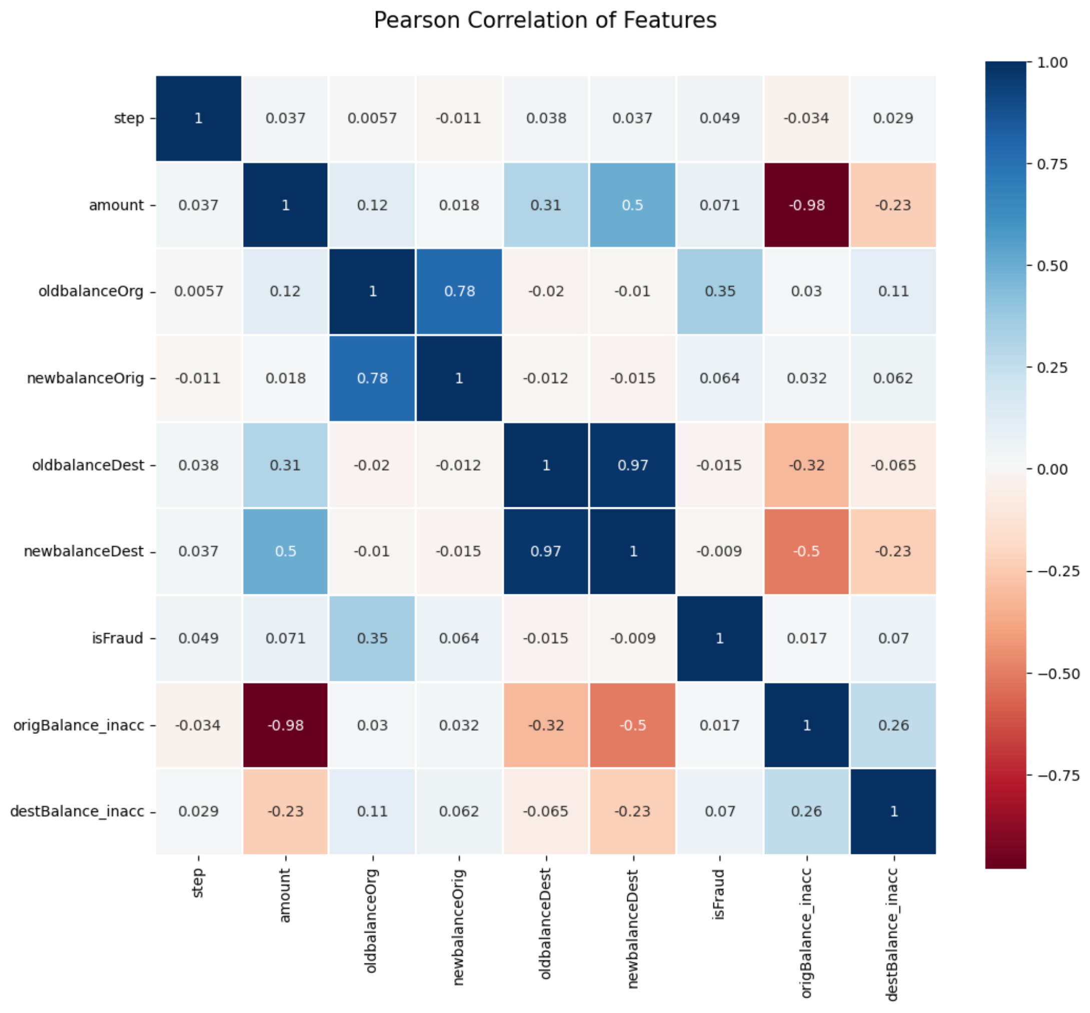
4.2 Predictive Modeling for Fraud Detection

In the previous sections, we identified dimensions that make fraudulent transactions detectable. Based on these results, we build supervised classification models.

4.2.1 Modeling Dataset Creation

First making a correlation plot to check highly correlated variables

Figure 14: Correlation plot of variables

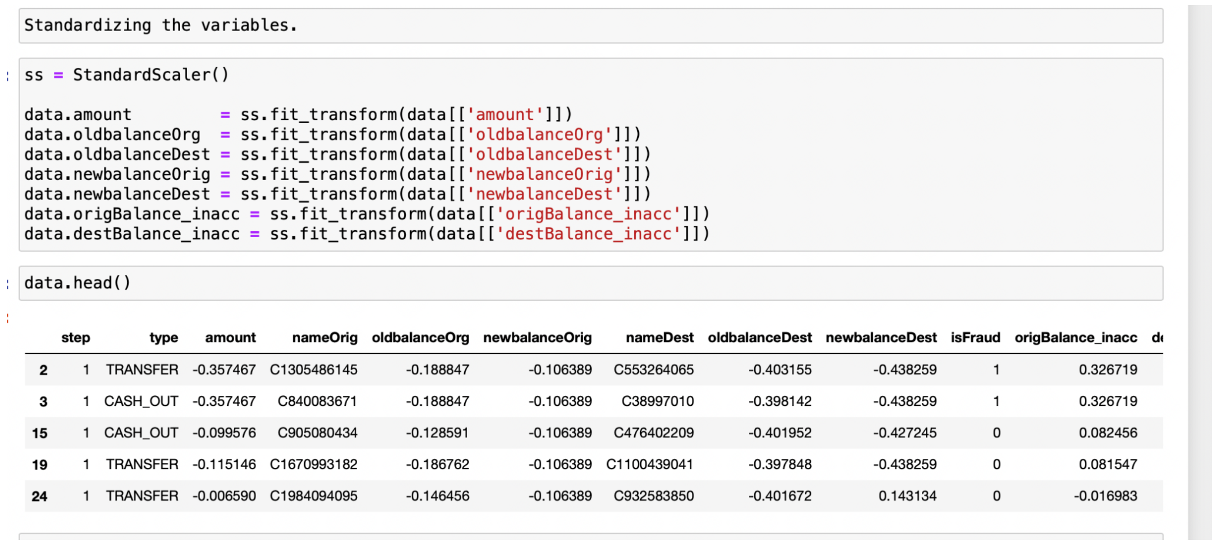


	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	origBalance_inacc	destBalance_inacc
step	1	0.037	0.0057	-0.011	0.038	0.037	0.049	-0.034	0.029
amount	0.037	1	0.12	0.018	0.31	0.5	0.071	-0.98	-0.23
oldbalanceOrg	0.0057	0.12	1	0.78	-0.02	-0.01	0.35	0.03	0.11
newbalanceOrig	-0.011	0.018	0.78	1	-0.012	-0.015	0.064	0.032	0.062
oldbalanceDest	0.038	0.31	-0.02	-0.012	1	0.97	-0.015	-0.32	-0.065
newbalanceDest	0.037	0.5	-0.01	-0.015	0.97	1	-0.009	-0.5	-0.23
isFraud	0.049	0.071	0.35	0.064	-0.015	-0.009	1	0.017	0.07
origBalance_inacc	-0.034	-0.98	0.03	0.032	-0.32	-0.5	0.017	1	0.26
destBalance_inacc	0.029	-0.23	0.11	0.062	-0.065	-0.23	0.07	0.26	1

Input features oldbalanceDest and newbalanceDest are correlated with each other so we had to drop one of the feature. To decide which one to drop amongst them we have to check their correlation with dependent feature (isFraud), oldbalanceDest is highly negatively correlated (-0.0059) with target feature than newbalanceDest (0.00054).Also oldbalanceOrig and newbalanceOrig are also correlated with each other and newbalanceOrig is highly negatively correlated (-0.0081) with target feature than oldbalanceOrg (0.01). We will be considering dropping oldbalanceOrg and newbalanceDest feature.

4.2.2 Standardizing the data

In this transformation, we convert all columns in the data to have the same range. This is done through the standard scaler feature available in python. The following code snippet is used to perform this transformation.



```
Standardizing the variables.
```

```
ss = StandardScaler()

data.amount         = ss.fit_transform(data[['amount']])
data.oldbalanceOrg  = ss.fit_transform(data[['oldbalanceOrg']])
data.oldbalanceDest = ss.fit_transform(data[['oldbalanceDest']])
data.newbalanceOrig = ss.fit_transform(data[['newbalanceOrig']])
data.newbalanceDest = ss.fit_transform(data[['newbalanceDest']])
data.origBalance_inacc = ss.fit_transform(data[['origBalance_inacc']])
data.destBalance_inacc = ss.fit_transform(data[['destBalance_inacc']])
```

```
data.head()
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	origBalance_inacc	d
2	1	TRANSFER	-0.357467	C1305486145	-0.188847	-0.106389	C553264065	-0.403155	-0.438259	1	0.326719	
3	1	CASH_OUT	-0.357467	C840083671	-0.188847	-0.106389	C38997010	-0.398142	-0.438259	1	0.326719	
15	1	CASH_OUT	-0.099576	C905080434	-0.128591	-0.106389	C476402209	-0.401952	-0.427245	0	0.082456	
19	1	TRANSFER	-0.115146	C1670993182	-0.186762	-0.106389	C1100439041	-0.397848	-0.438259	0	0.081547	
24	1	TRANSFER	-0.006590	C1984094095	-0.146456	-0.106389	C932583850	-0.401672	0.143134	0	-0.016983	

4.2.3 Create train and test datasets

We split the scaled dataset into training and testing datasets. We decide to use 70% of the original data for training and the remaining 30% for testing.

Chapter 5

5.1 Classification Models for Fraud detection

We define 3 models to perform the classification: Logistic Regression,Random Forest and XGBoost.

To measure the performance of the models, Recall is a useful metric. High-class imbalance datasets typically result in poor Recall, although accuracy may be high. Precision will also be a consideration because reduced precision implies that the company that is trying to detect fraud will incur more cost in screening the transactions. In fraud detection problems, though, accurately identifying fraudulent transactions is more critical than incorrectly classifying legitimate transactions as fraudulent.

Alternatively, we could also go with Area Under Curve (AUC) of the ROC curve. However, this will not adequately capture if the model is correctly identifying most of the fraudulent transactions. Therefore, we use this as a validation of the model performance.

We also need to do cross-validation to ensure the models do not overfit the training data. For this, we use Stratified 5-fold since we need to ensure that the class imbalance is retained in the validation sets.

5.1.1 Logistic Regression Model

In this section, we train the logistic regression model and calculate the mean recall score. This parameter will serve as a benchmark for further experiments.

The following output indicates how the Logistic Regression model performs on the training dataset.

Figure 15

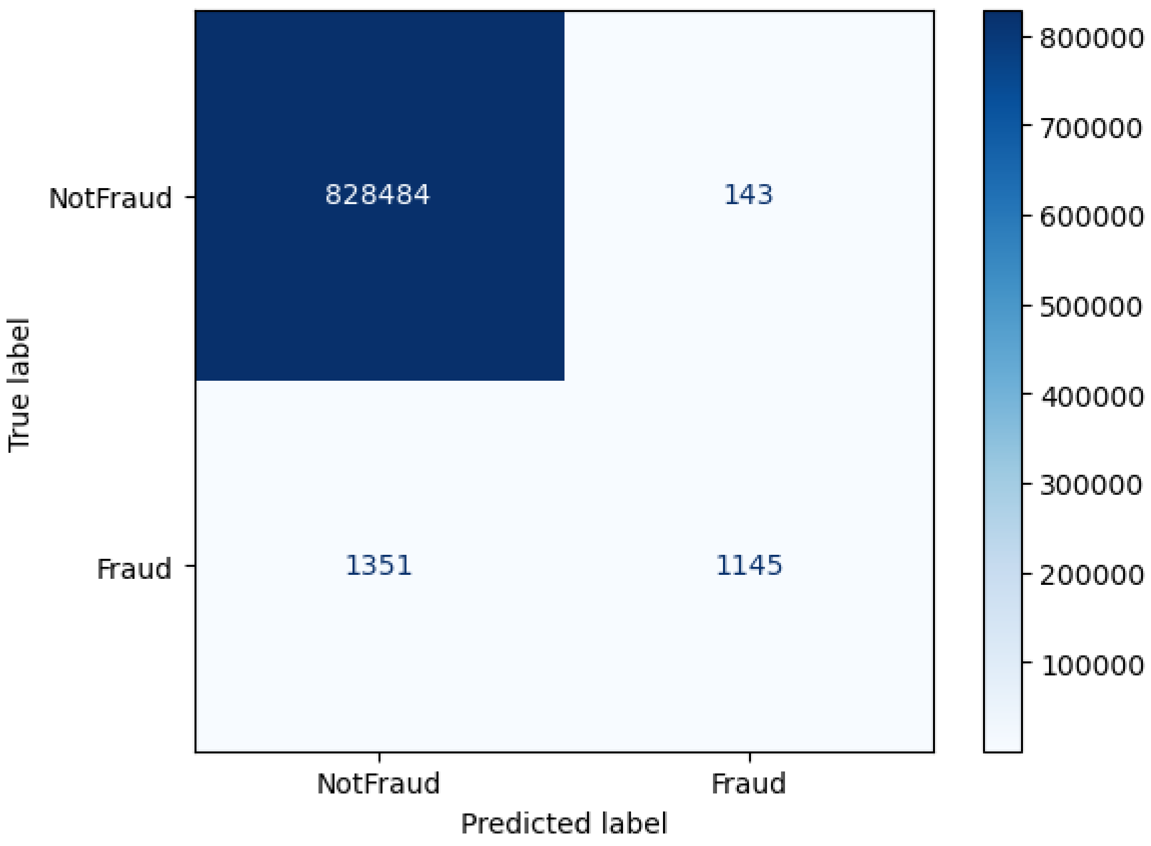
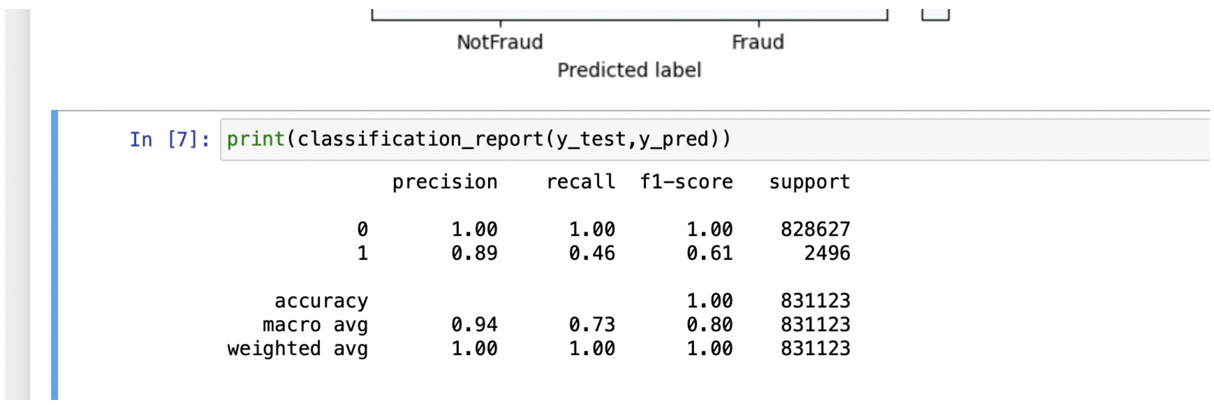


Fig 16



```
In [7]: print(classification_report(y_test,y_pred))
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    828627
           1       0.89      0.46      0.61      2496

    accuracy                           1.00    831123
   macro avg       0.94      0.73      0.80    831123
weighted avg       1.00      1.00      1.00    831123
```

5.1.2 Addressing Class Imbalance

There are many techniques to address high-class imbalanced datasets. A few examples are as follows –

1. Undersampling: In this method, random samples from the majority class are deleted so that the class imbalance is more manageable.

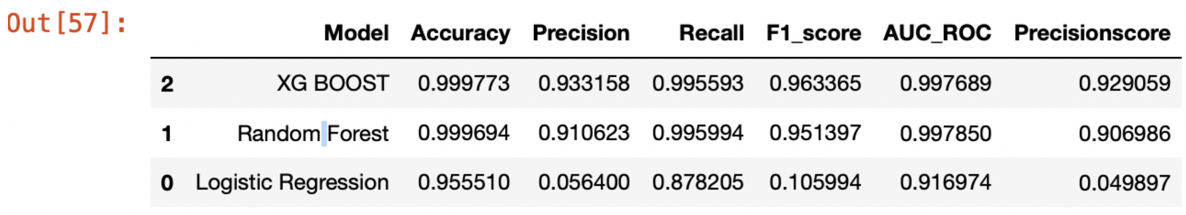
2. Oversampling: In this method, observations of the minority class are resampled with repetition to increase their presence in the data

3. SMOTE: This is a type of oversampling, but instead of repeating the observations, it synthesizes new plausible observations of the minority class

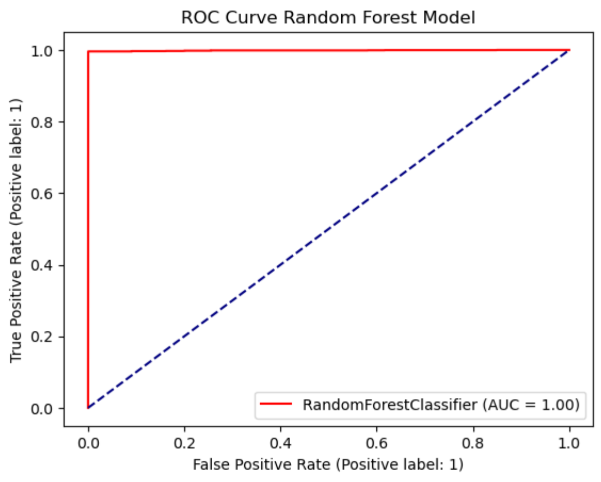
We use undersampling as it is less computation-intensive. We also do this for all our models.
We retain all the fraud cases and randomly select an equal number of non-fraud cases to create an undersampled training dataset.

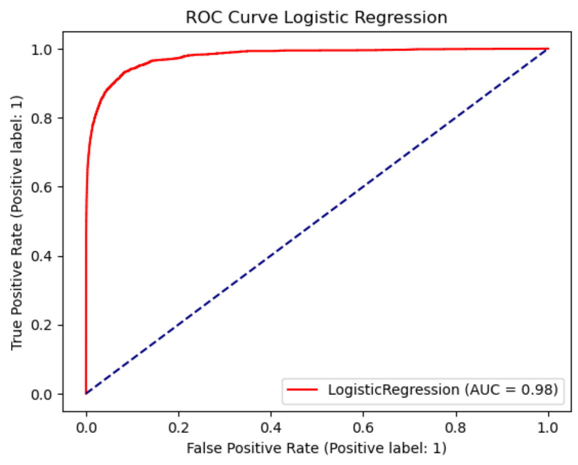
5.2 Best Fit Model Details

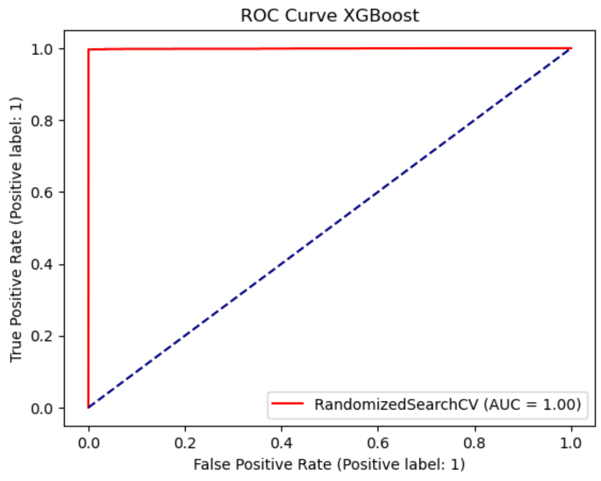
We tried all three models after undersampling the data. Random forest and XGBoost results were better than Logistic Regression.So we performed hyper parameter tuning and cross validation on both the later models.We saw the XGboost is performing the best.

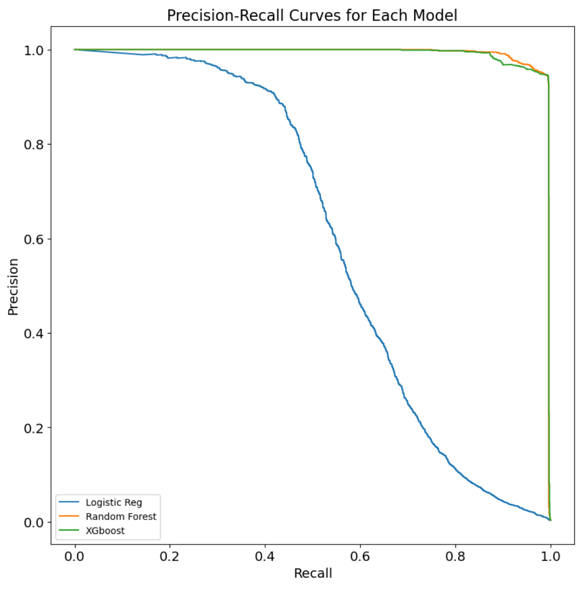

Out[57]:

	Model	Accuracy	Precision	Recall	F1_score	AUC_ROC	Precisionscore
2	XG BOOST	0.999773	0.933158	0.995593	0.963365	0.997689	0.929059
1	Random Forest	0.999694	0.910623	0.995994	0.951397	0.997850	0.906986
0	Logistic Regression	0.955510	0.056400	0.878205	0.105994	0.916974	0.049897

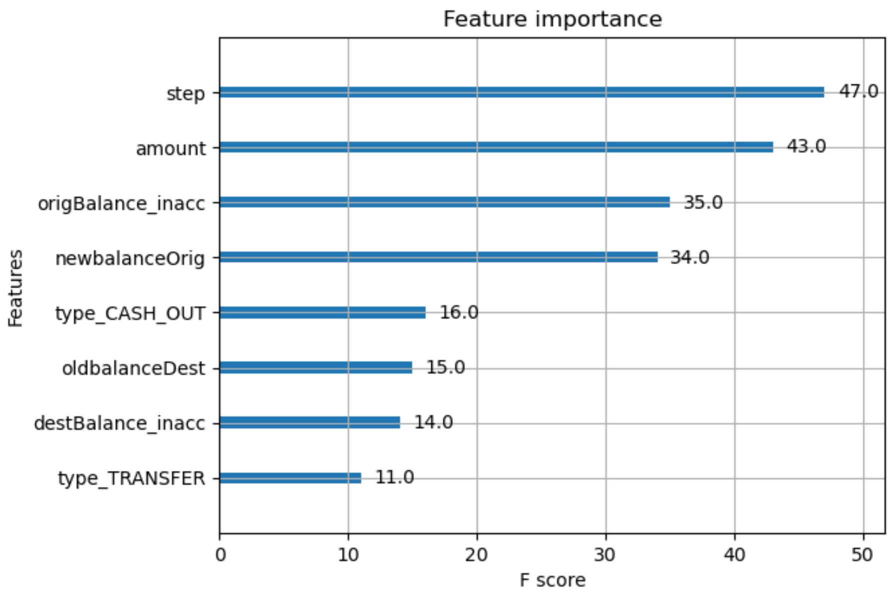








In the following figure, we present the relative feature importance of the XGboost model. The fig. shows which variables are contributing more to fraud prediction.



Chapter 6

6.1 Conclusion

In conclusion, we successfully developed a framework for detecting fraudulent transactions in financial data. This framework will help understand the nuances of fraud detection such as the creation of derived variables that may help separate the classes, addressing class imbalance and choosing the right machine learning algorithm.

We experimented with 3 machine learning algorithms – Logistic Regression,Random Forest and XGBoost. The XGboost algorithm gave results closer to Random forest but far better results than Logistic Regression indicating tree-based algorithms work well for transactions data with well- differentiated classes. This also emphasizes the usefulness of conducting rigorous exploratory analysis to understand the data in detail before developing machine learning models. Through this exploratory analysis, we derived a few features that differentiated the classes better than the raw data.

6.2 Recommendations

Through this project, we demonstrated that it is possible to identify fraudulent transactions in financial transactions data with very high accuracy despite the high-class imbalance. We provide the following recommendations from this exercise

Fraud detection in transactions data where transaction amount and balances of the recipient and originator are available can be best performed using tree-based algorithms like Random Forest and XGBoost .Using dispersion and scatter plots to visualize the separation between fraud and non-fraud transactions is essential to choose the right features. To address the high-class imbalance typical in fraud detection problems, sampling techniques like undersampling, oversampling, SMOTE can be used. However, there are limitations in terms of computation requirements with these approaches, especially when dealing with big data sets.

To measure the performance of fraud detection systems, we need to be careful about choosing the right measure. The recall parameter is a good measure as it captures whether a good number of fraudulent transactions are correctly classified or not. We should not rely only on accuracy as it can be misleading.

References

1. E. Ngai et.al., The Application of Data Mining Techniques in Financial Fraud Detection: A Classification Framework and an Academic Review of Literature, Decision Support Systems. 50, 2011, 559–569

2. Albashrawi et.al., Detecting Financial Fraud Using Data Mining Techniques: A Decade Review from 2004 to 2015, Journal of Data Science 14(2016), 553-570

3. TESTIMON @ NTNU, Synthetic Financial Datasets for Fraud Detection, Kaggle, retrieved from https://www.kaggle.com/ntnu-testimon/paysim1

4. Jayakumar et.al., A New Procedure of Clustering based on Multivariate Outlier Detection. Journal of Data Science 2013; 11: 69-84

5. Jans et.al, A Business Process Mining Application for Internal Transaction Fraud Mitigation, Expert Systems with Applications 2011; 38: 13351–13359

6. Phua et.al., Minority Report in Fraud Detection: Classification of Skewed Data. ACM SIGKDD Explorations Newsletter 2004; 6: 50-59.

7. Dharwa et.al., A Data Mining with Hybrid Approach Based Transaction Risk Score Generation Model (TRSGM) for Fraud Detection of Online Financial Transaction, International Journal of Computer Applications 2011; 16: 18-25.

8. Sahin et.al., A Cost-Sensitive Decision Tree Approach for Fraud Detection, Expert Systems with Applications 2013; 40: 5916–5923.

9. Sorournejad et.al., A Survey of Credit Card Fraud Detection Techniques: Data and Technique Oriented Perspective, 2016