1. Introduction

In this summary report, my aim is to explain the frauds that can happen through the credit card transactions either Online or Physical swipe purchases using different factors like transaction amount, credit card limit, Type of transaction etc. The model is built using the past data that was downloaded from here.

2. Motivation & Intent

Credit card transactions are a huge problem in the current technologically advanced world. With **~10%** market share & **~90Bn USD** outstanding loan through Credit cards, Capital One is one of the leading banks in the USA for Credit cards. This means many transactions happen, which are difficult to analyze manually.

Credit card fraud detection is, by definition, a cost-sensitive problem in the sense that the cost due to a false positive is different than the cost of a false negative. When predicting a transaction as fraudulent, when in fact it is not a fraud, there is an administrative cost that the financial institution incurs. On the other hand, when failing to detect fraud, that transaction is lost. Moreover, it is not enough to assume a constant cost difference between false positives and false negatives, as the amount of the transactions varies significantly; therefore, its financial impact is not consistent but depends on each transaction. Here I tried to propose a new cost-based measure to evaluate credit card fraud detection models, taking into account the different financial costs incurred by the fraud detection process.

By analyzing the classification algorithms in the Machine Learning area, my idea is to:

- Reduce Human Effort in deciding Credit Card fraud transactions.
- Reduce Computations effort.
- Optimise the False Positive rates, which lead to customer churn.

Post-deployment, our systems should detect patterns and restrict the fraudulent transactions that might occur through our credit cards.

The objective is to study and analyze how machine learning models like Boosting (XGBooster, etc.), Ensemble (Random Forest, etc.), and data balancing techniques like RandomUnderSampler impact the fraud detection problem we have here.

3. Dataset Description

The dataset that I obtained is **786363 Rows** and **29 Features.** Preliminary analysis reveals several interesting features about the data. Figure 1 shows the split of fraudulent and legitimate transactions.

Only 12417 transactions, or less than **1.6%** , are fraudulent, highlighting a stark class imbalance. The data is seen to be from Jan 1, 2016, to Dec 31, 2016, a full one year of data of multiple customers and transactions.

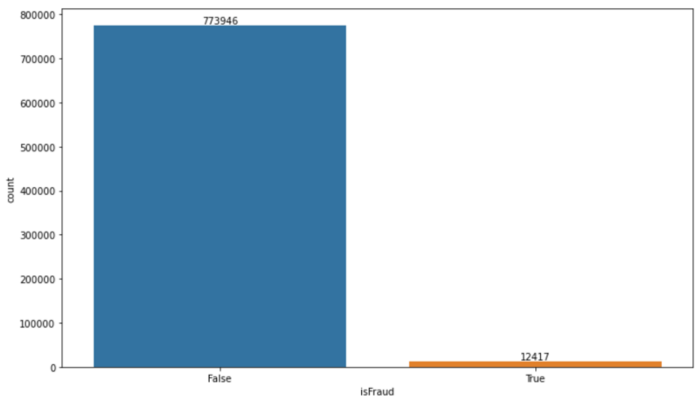


Table 1: Description key variables of the data:

Variable	Description
accountNumber	Account number of the customer
creditLimit	Total credit limit of customer
availableMoney	Current available limit on Credit Card
transactionDateTime	The time stamp of transaction
transactionAmount	Amount transferred to merchant
merchantName	Name of the seller / merchant
merchantCategoryCode	Category of merchant
cardCVV, enteredCVV	CVV Information
posEntryMode	Type of transaction made (Encoded with numbers)
posConditionCode	Condition of pos Machine
transactionType	Catgeorical data which represents Purchase, Reversal or Verification

Also, Table 2 shows a striking difference in the average value of the transaction between fraud and nonfraud types. When I looked at the distribution of the critical variables of fraud transactions, I discovered that data in those columns are tail *heavy / Left Skewed.* This might lead to a few operations like logarithmic function application on the columns for obtaining a more bell curve.

Figure: Distribution of key variables.

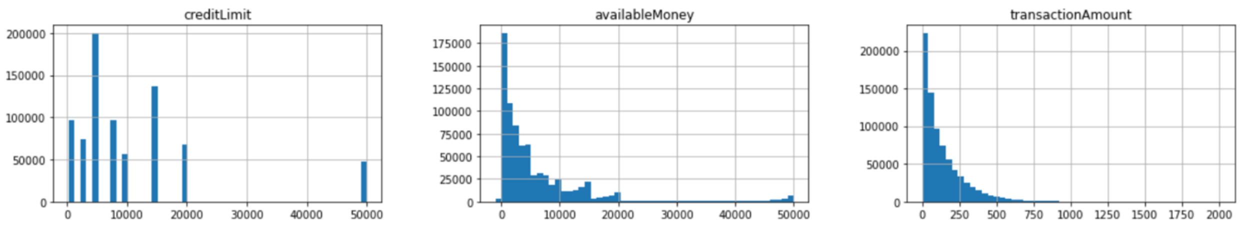
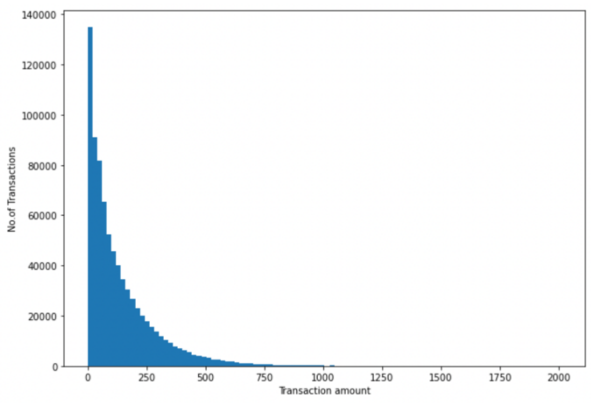


Table 2: Transaction value by class:

	Total Value in USD	Average Value in USD
All Transactions	107720557.53	136.98
Non Fraud	104924051.64	135.70
Fraud	2796505.89	225.21

One interesting thing, related to the average value of the transactions, I found that with transaction amount plot below, with bins = 100 that transactions peak from 0-100 usd with more that 50% transactions are under 100 USD & **>75%** transactions are **< 200USD**



4. Fraud model evaluation

A credit card fraud detection algorithm identifies those transactions with a high probability of fraud based on historical fraud patterns. The use of machine learning in fraud detection has been an exciting topic in recent years. As discussed, our goal is to minimize the False Positives and False Negatives given the financial and customer satisfaction point of view. Hence I will be using the following metrics for measuring the models:

- Accuracy Score = (TP+TN) /(TP+TN+FP+FN)
- Confusion Matrix
- Area under ROC (receiver operating characteristic) Curve: *The ROC curve is a plot for the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.*

True Positive (TP) : Fraud correctly identified as Fraud
True Negative (TN) : Non-fraud correctly identified as Non-fraud
False Positive (FP) : Fraud incorrectly identified as Non-fraud
False Negative (FN) : Non-fraud incorrectly identified as Fraud

5. Reversal & Multi Swipe Transactions

Credit cards are one of the most used mediums of transactions; this huge volume often results in mistaken transactions at times, even by the legal customer. An analysis of reversed transactions will help us understand more of the customer persona. An increased reversed transaction might lead to a bad customer experience. On the other hand, multiswipe transactions often lead to multiple deductions from the user account. If the analysis of this aspect can help us build an alert system on the multiswipe transactions, it can enhance secure payments through our bank. Duplicate transactions are searched in the 5**Mins bracket** after the first original transaction for the analysis.

In the current data I am working on, I tried to work on the two types of transactions as detailed in below Table:

Type of transaction	Total Transactions	Total value of transactions
Reversal Transactions	20303	2821792.5 USD
Multi-swipe transactions	13403	1933949.11 USD

Trying to answer this question –

Is there a relation between Multiswipe transactions & reversal transactions?

If we look in depth, **42.7%** of *Multiswipe transactions* are *Reversed transactions* which is a significant number. In the whole dataset, **129** transactions were fraudulent and were multiswiped, which is a very negligible amount.

6. Models

I used the Python-based Jupyter Notebook for all parts of the analysis. Four primary model categories were considered: logistic regression, Decision Tree Classifier, Random forests, and XGBoost Classifier. In fitting the models, I followed a step-by-step procedure where models were trained with a training dataset and calculated evaluation scores. The best model is the one that gives me the best accuracy score and better learning rate in the ROC curve. The best-performing model on the test set in each category was selected as a finalist.

6.1 *Trying to answer this question –*

But the accuracy score isn't the best metric for a huge data imbalance; how to go about that?

I know I can depend on accuracy score as one of my evaluation metrics because I will undersample my dataset and ensure we have equal amounts of data on Fraud and Non-Fraud. Such methodology also helps conserve computational power.

My Sampling method:

I used RandomUnderSampler() from imblearn package; this is good because this method gives me an equal number of fraud and non-fraud transactions randomly.

I have implemented 4 of the most famous classification algorithms for this dataset.

6.2 Logistic Regression

Logistic regression is a widely used discriminative learning algorithm that uses a hypothesis of the form h_θ (x)=1/(1+e^(-θ^T x)). In its simplistic form, logistic regression fits a linear decision boundary in the feature space to model a binary dependent variable.

6.3 Decision Tree Classifier

The Decision tree algorithm is a simple yet efficient supervised learning algorithm wherein the data points are continuously split according to specific parameters and the algorithm's problem. As we have inherently categorical data, this can be one good addition to the models to solve the current situation.

6.4 XGBoost Classifier

My thought behind using XGBoost Classifier is that XGBoost performs sequential bagging on illegal data, which is very relevant in the case of the given dataset. In XGBoost, we fit a model on the gradient of loss generated from the previous step. Hence it is a good model for a varied unbalanced dataset.

6.5 Random Forest Classifier

Tree-based models stratify the predictor space into simple regions that can classify the dependent variable. A simple tree model like a decision tree is not competitive with other machine learning techniques.

Hence, a model like Random forest, which combines many trees using bagging, boosting, and random forests, often results in significant improvements in prediction accuracy at the expense of some loss interpretation. I think that's not a bad tradeoff for the project at hand since the goal is to maximize predictive performance over interpretability.

7 Model Results

Table: Accuracy Score of Models

		Predicted False	Predicted True
Logistic Regression	Actual False	1907	889
	Actual True	1073	2092
	Accuracy Score		**0.67**
		Predicted False	Predicted True
Decision Tree	Actual False	1840	1202
	Actual True	1140	1779
	Accuracy Score		**0.60**

		Predicted False	Predicted True
XGBoost	Actual False	2069	881
	Actual True	911	2100
	Accuracy Score		**0.70**
		Predicted False	Predicted True
Random Forest	Actual False	2055	988
	Actual True	925	1993
	Accuracy Score		**0.68**

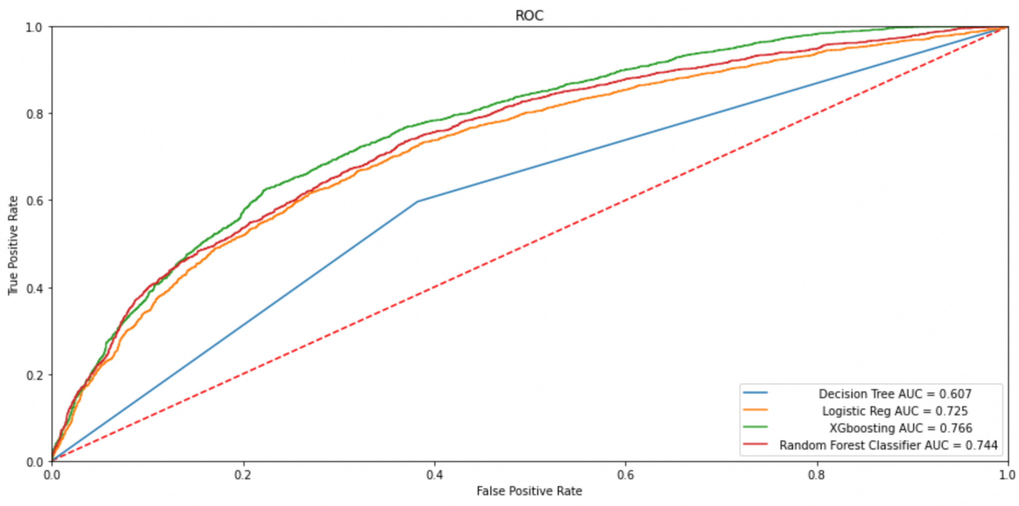
Amongst all the models, XGBoost Classifier has the best Area under Curve (Refer to the ROC Curve below). It, therefore, is the best model for analyzing and predicting fraud transactions.

7.1 So, which model should be used for deployment?

Hence, as discussed initially, XGBoost has the lowest **False Positive numbers, 13% better** than Random Forest False Positive numbers. Using a model with less False Positive is cost-effective.

Table: Accuracy scores and Figure: ROC curve shows the performance of the finalist models on the test set after they were modeled using the train data, which was balanced. The XG Boost Classification model was the best performing model across almost every objective performance metric, with the random forest being near the second-best performing model

Figure: ROC Curve of Models implemented



8 Conclusion & Future Work

Credit Card fraud is a significant and growing issue in the United States, but detecting it can be like finding a needle in a haystack. The model I built here can be made better by using other sampling methods like SMOTE and using variable tuning methods like hyperparameter tuning.

A key aspect for Fraud prediction modeling is analyzing the cost-effectiveness of the model; for future purposes, we need to build a cost function of the effect of these prediction models. Getting a hold on customer spending patterns through RFM(Recency, Frequency & Monetary) methods can help our machine build alert systems.

Example of Alert machines:
*One such finding in our dataset is that if the transaction is **15%** more than any customer's avg spent in the past, it is **75%** sure that it is a fraud transaction. If this data area is loaded with more points, this can potentially save real-life scenarios for fraud transactions.*

Given that payments fraud is constantly evolving, future work areas might include applying reinforcement learning to a real-time data stream. Although fraudsters will never retire, machine learning algorithms can grow into a more challenging entity to conduct frauds.

9 Resources:

Feature engineering strategies for credit card fraud detection, Alejandro Correa Bahnsen :
https://albahnsen.github.io/files/Feature%20Engineering%20Strategies%20for%20Credit%20Card%20Fraud%20Detection_published.pdf

https://trenton3983.github.io/files/projects/2019-07-19_fraud_detection_python/2019-07-19_fraud_detection_python.html

https://seaborn.pydata.org/generated/seaborn.scatterplot.html