1 Introduction

Financial frauds are a wide-ranging term for theft including involving a payment card, such as a credit card or debit card or loans. This project shows that increasing financial fraud a an important risk and it is hard to get real data due to privacy rules and laws. In the United States, the ratio of the financial fraud happening is limited to %0.1 which means that 99.9% of the transactions are safe. This still causes huge financial losses in the USA and all around the world. This losses can be scaled up to tens of thousands of million dollars annually.

In this project a fraud prediction model will be developed. This project may potentially benefit financial institutions saving thousands of US dollars and each individuals who are victims of incidents.

Data set is acquired from Kaggle that is about "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. " (Kaggle,2019).

As mentioned above the fraudulent transaction lower than 0.1%. This broughts a technical difficulty to label non-fraudulent transaction (true positive)   as non-fraudulent (true positive) and fraudulent (false negative) transactions as fraudulent (false negative). This kind of data sets are called imbalanced data. There are 6362620 records of that only less than 1% is fraudulent.

2. Data Collection and Upload

Data set is acquired from Kaggle. Synthetic Financial Datasets For Fraud Detection PaySim simulation of mobile money transactions based on a sample of real transactions.

Exploratory data analysis and modelling will be performed on this dataset. Only for the evaluation purposes, best models will be tested on two datasets.

There are 6362620 records and 11 features (columns) in the Synthetic Financial Datasets For Fraud Detection dataset. Dataset is relatively clean. The details of the dataset can be seen at the jupyter notebook here Capstone 1 - Preparing Data.ipynb.

3 Data Exploration

3.1 Data Wrangling

Below is the head of data at Table1. Data mainly represents the money transfers between two individual form origin (sender) including sender's balance before and after transaction to destination (receiver) including receiver's balance before and after the transaction.

These transactions are also labelled as fraud, not fraud and flagged as fraud or not fraud.

STEP	TYPE	AMOUNT	ORIGIN	OLD BAL ORI	NEW BAL ORG	DESTINATION	OLD BAL DEST	NEW BAL DEST	Is Fraud	Is Flagged Fraud
1	PAYMENT	9839.64	C1231006815	170136	160296.36	M1979787155	0	0	0	0
1	PAYMENT	1864.28	C1666544295	21249	19384.72	M2044282225	0	0	0	0
1	TRANSFER	181	C1305486145	181	0	C553264065	0	0	1	0

Table 1. Dataset

Data set has no null values by default each column is checked if there was a null value.

Feature	isNull()
type	100% Not Null
amount	100% Not Null
nameOrig	100% Not Null
oldBalanceOrig	100% Not Null
newBalanceOrig	100% Not Null
nameDest	100% Not Null
oldBalanceDest	100% Not Null
newBalanceDest	100% Not Null
isFraud	100% Not Null
isFlaggedFraud	100% Not Null

Table 2. Null Value Ratio

3.2 Exploratory Data Analysis

3.2.1.Transaction Types

There are five transaction types presented in the dataset which are debit, transfer, cash-out, payment, cash-in. Cash-out is the most common transaction type and using debit for the transaction is the least common type.

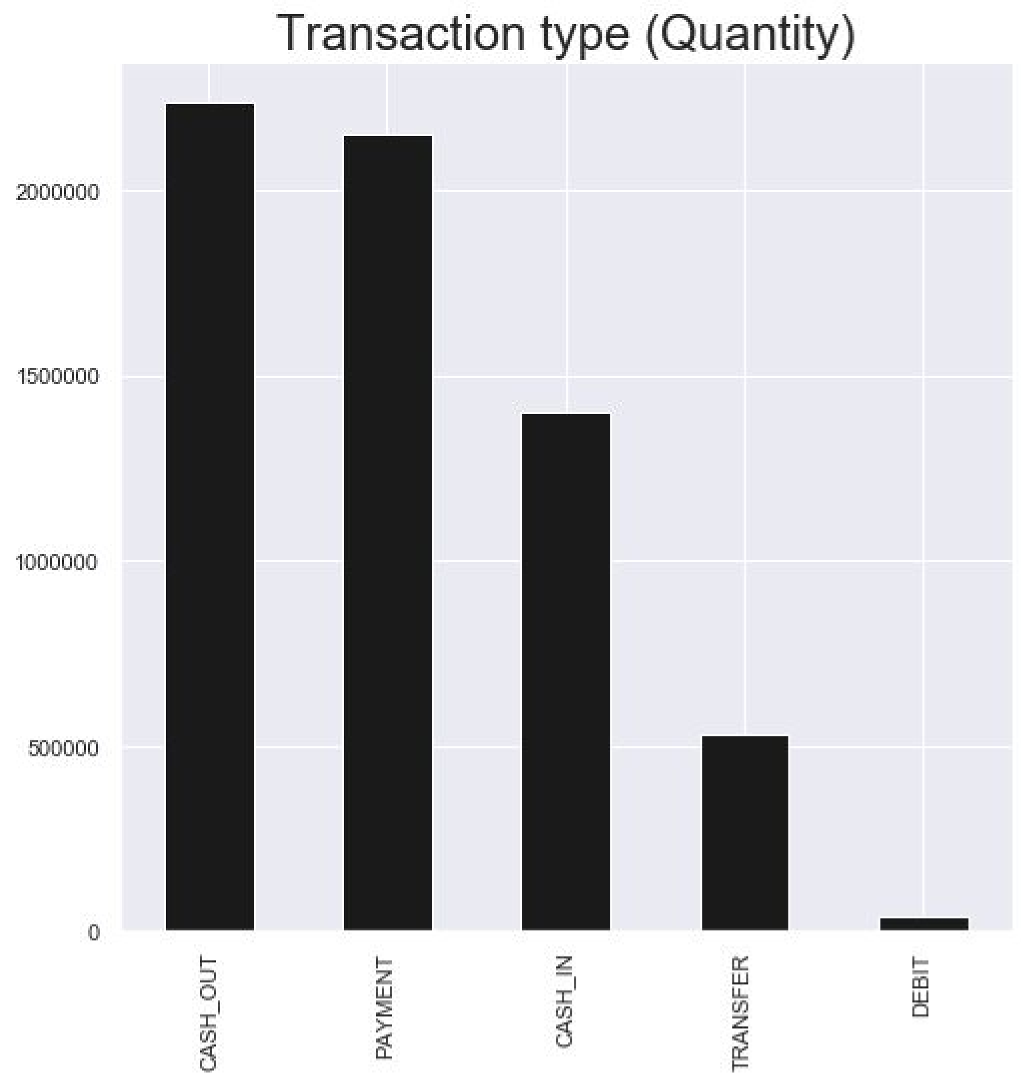


Figure 1. All Transaction Types

The amount of transaction per each type is provided in Figure 2 below. The largest amount of money transaction types are the TRANSFER and CASH-OUT transactions. As seen below debit transactions are the least ones in the total amount of $227199221.27. It is shown as zero in Figure 2 because it is too small in comparison to other transactions.

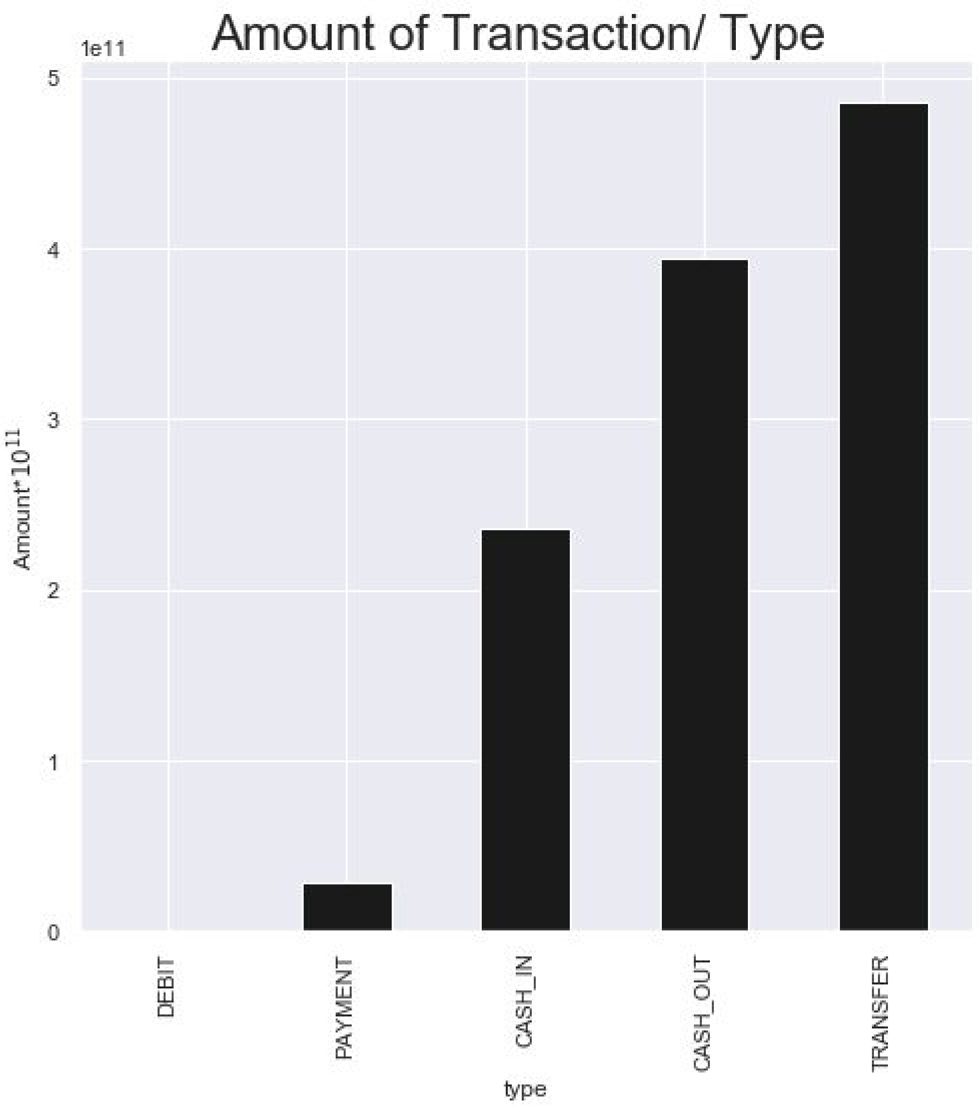


Figure 2. Amount of Transaction per Type

3.2.3. Fraud Analysis

Ratio fraudulent transaction is very small. There are 6354407 transactions of which only 8213 were identified as a fraudulent transaction. The ratio of fraudulent transaction is %0.13.

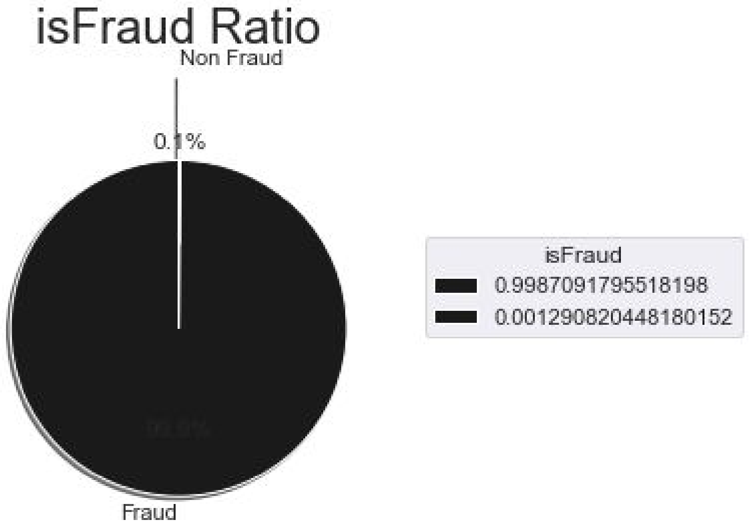


Figure 3. isFraud Ratio Pie Chart

isFraud	Quantity	Percentage
False	6354407	99.87%
True	8213	0.13%

Table 3. isFraud Ratio

3.2.3. Fraudulent Transaction Types

When fraudulent transaction types grouped by isFraud == True, interestingly only two types of transactions involved with fraudulent activity.

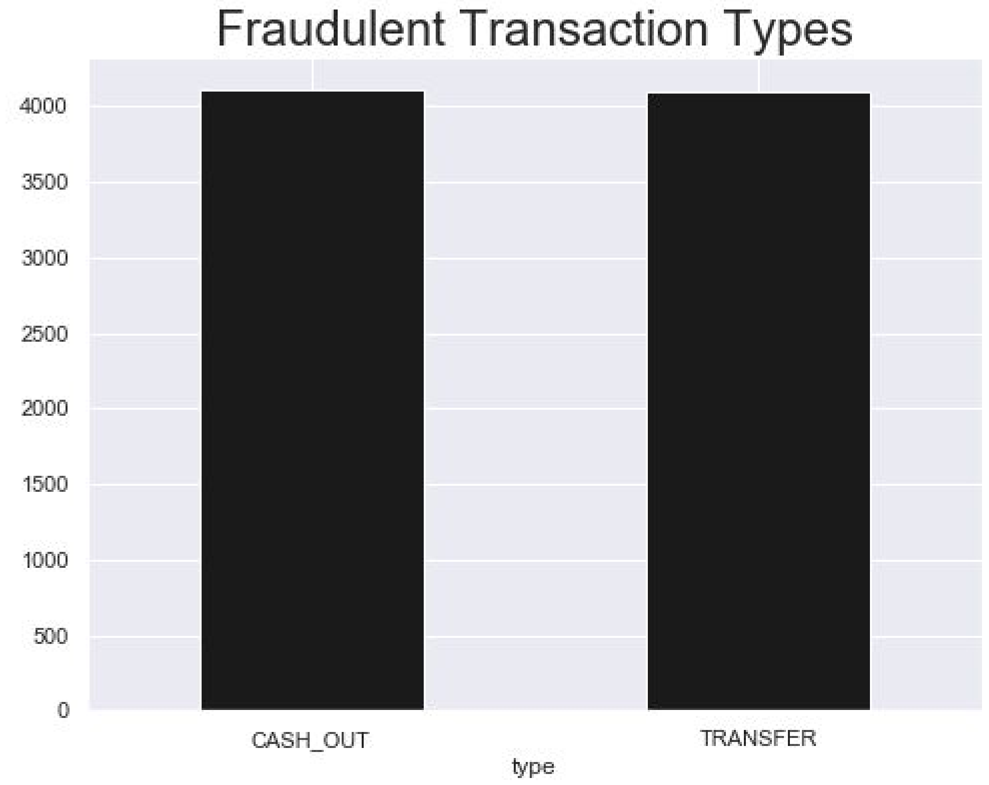


Figure 4. Fraudulent Transaction Types

When money TRANSFERRED fraudulently, sender floors the balance 96.12% of the time and the receiver wont gets the fraudulently TRANSFERRED into their account for the 99.29% of the time. The total money lost is 7564595045.72 dollar in fraudulent TRANSFERS.

When money CASH-OUT fraudulently sender sender floors the balance 99% percent of the time but interestingly receiver put that money into account 99.44% of the time. This shows that fraudulent transactions share common properties like wiping out one of the balances completely.

The amount of money that is lost during CASH -OUT fraudulent transaction is about 5984124999.99 dollars.

There are 532909 TRANSFER transactions of which only 4097 TRANSFERS are fraudulent. This is about 0.08 % of the TRANSFER transaction, 50% of all fraudulent transactions and 0.06% of the whole dataset.

Similarly, There are 2237500 CASH-OUT transactions of which only 4097 TRANSFERS are fraudulent. This is about 0.18 % of the CASH-OUT transaction, 50% of all fraudulent transactions and 0.06% of the whole dataset.

To sum up, CASH-OUT fraudulent transactions are more frequent than TRANSFERS fraudulent transactions.

3.2.4 Amounts

There are 8213 transactions that are labelled as being fraudulent. Majority of the fraudulent transaction between $1M to $100K. Second largest group that has the higher amount of fraudulent transactions are the ones higher than $1M.

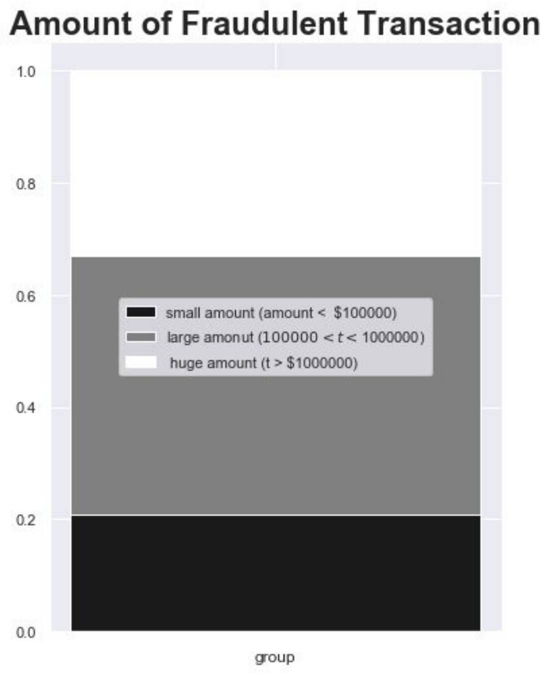


Figure 5. Fraudulent Transaction Amounts Count

This is interesting that the count of fraudulent transaction are not that many in comparison to numbers of fraudulent transaction in larger amounts.

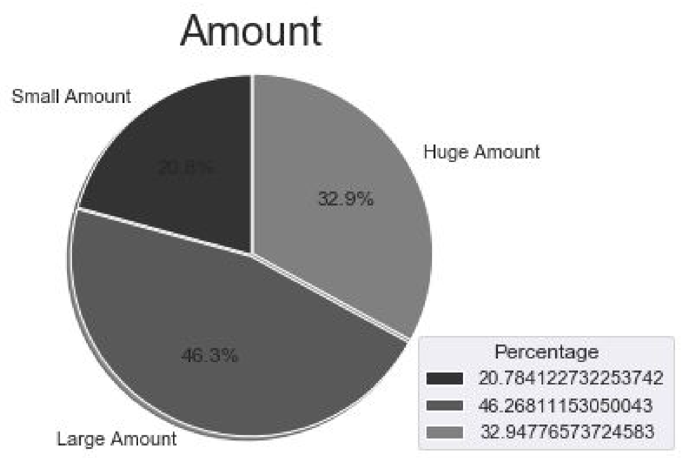


Figure 5. Fraudulent Transaction Amounts Count

Amount	Quantity	Total	Ratio
Small amount	1710	8213	%20.78
Large amount	3800	8213	%46.27
Huge amount	2706	8213	%32.95

Table 4. Fraudulent Transaction Amounts Count

There are total 6362620 records of which 8213 records are found as fraudulent transaction and 6354407 records are not fraudulent transaction. The ratio of is ridiculously small, however it in fact affects 8213 individuals with 11968599360.44 $ in total which is about 12 Million dollars. (Calculation is made when the fraud is true, original balance before transaction minus original balance after transaction)

3.2.5 Balance Analysis

3.2.5 Flagging Fraudulent Transaction

Since two fraudulent transactions are mainly from two categories. How could these transactions possibly labelled with a flag that merchant can put more attention on those transactions to prevent them. By the definition of the flagging transaction as fraudulent is that " 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."

When the flagged transaction are analyzed there are common futures or methods might be used to categorize them.

Method 1

When there is a record of transaction that balance of the sender has not changed, these transactions are flagged as being fraudulent.

Method 2

When the type of transaction is TRANSFER and no money exchange has happened. These transactions are also flagged as being fraudulent.

Method 3

If the transactions comply with two methods above and higher than $200K, those are also flagged as being fraudulent.

According to the description of the dataset on flagged fraudulent transactions, there are only 16 matching case that has been flagged. In a data set has records more than 6M this flagged fraudulent data is insignificant and discarded. Is flagged feature (columns) is also dropped.

3.2.6 Time Series

Step feature of the data set is representing that it maps a unit of time in the real world. In this case 1 step is 1 hour of time. This 1 hour time slots are converted to a month time series so that some time series analysis can be performed on it.

Data set includes 743 hours that maps consecutive 30 days. There is a peak in the data regarding the amount of the transaction in the first half of the month. There is also a high volume of transactions at the mid of the month. These times seems suspicious and will be examined if any fraud activity is happening around the same time. Second, half of the month is pretty much stable, and has no too big fluctuations.

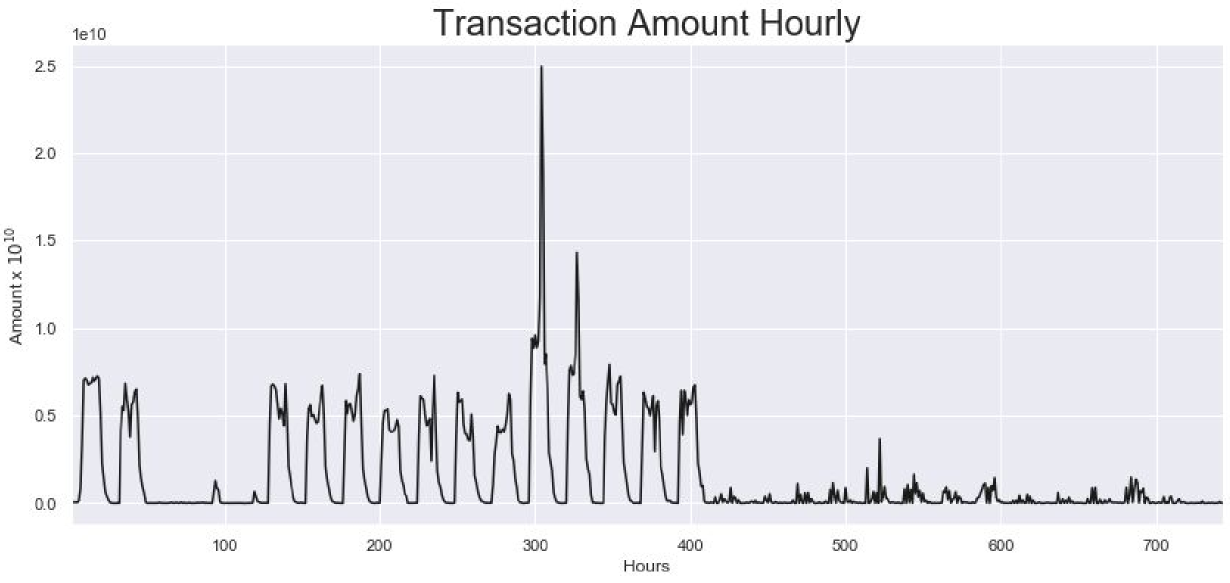


Figure 7. Transaction Amount Hourly

The mean amount of transaction is about 180K that represented with a red line in the plotted graph. This shows that there are some transactions are way higher than the rest. Median of the amounts is also around 150K. Maximum transaction is about $100M.

Fraudulent transactions are very quite often and happening everyday and at very large amounts.

Maximum amount of fraudulent transaction is $10000000 and minimum amount is $63.80. Mode of the fraudulent transaction is $10000000 which repeats 287 times. Mean and median of the transaction are $1467967.30 and $441423.44.

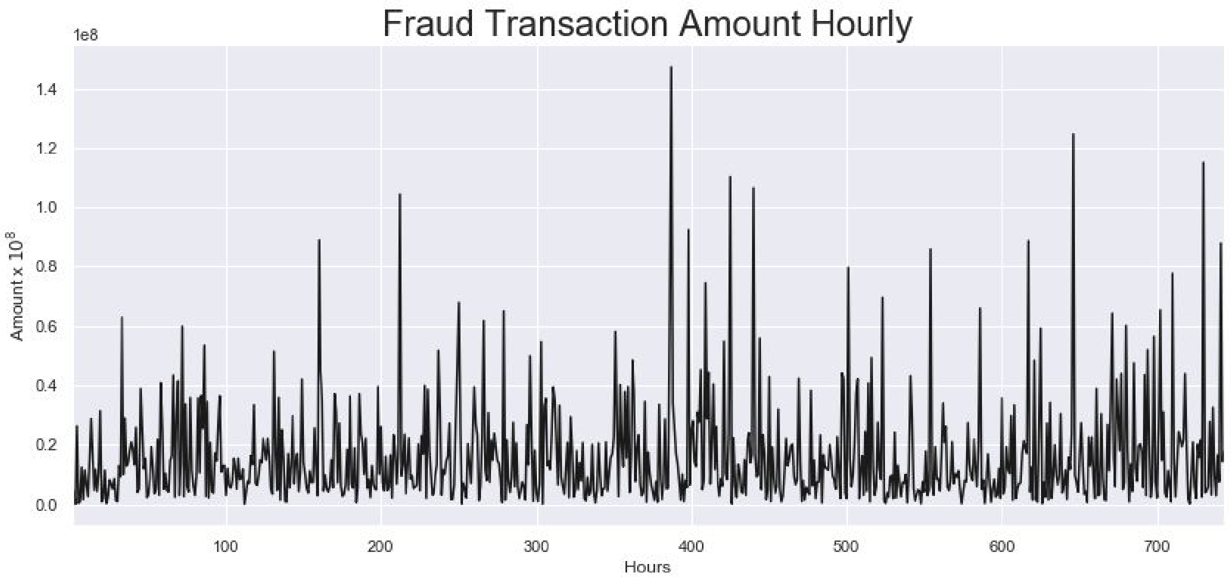


Figure 8.Fraud Transaction Amount Hourly

Daily average of the transactions are computed from the data set. Mode of 24 is calculated on STEP feature of the data set. As a result, daily and hourly data set is gathered (sum is used as aggregated method).

Data shows that total transaction per our is not consistent. Starting time of the data set is not defined, however after the mid day, there is usually a high volume of transactions.These transaction are very high in comparison to the first part of the day. It is also possible that the first part of the day might be night time.

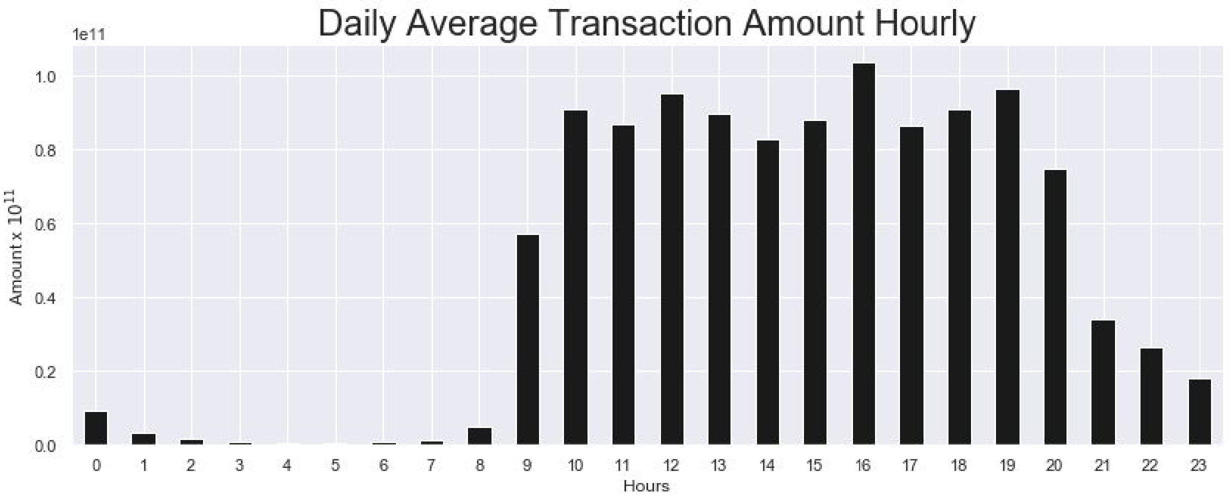


Figure 9.Daily Average Transaction Amount Hourly

Again, fraud transactions are happening in large amount but there is no high fluctuations in the amounts. This means that fraud transactions are happening any time of the day.

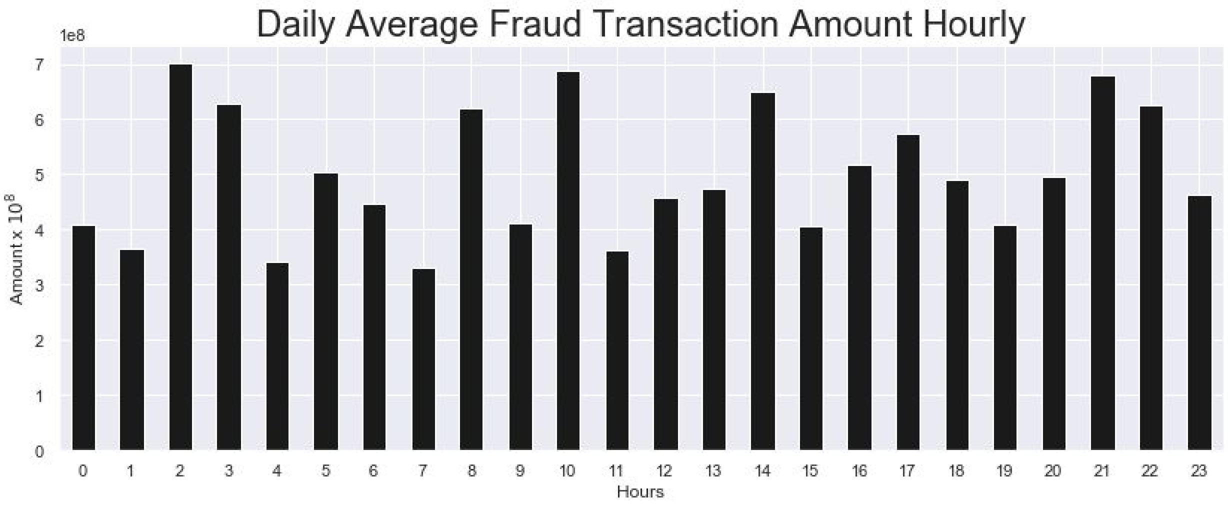


Figure 10.Daily Average Fraud Transaction Amount Hourly

3.2.7 Merchant - Customer Transaction

By the definition of the data sets, account holders are labelled as C and M. C stands for customer and at the same time M stands for merchant. Number of transaction between two parties are shown in Figure 6. Customer to customer transaction is much more than customer to merchant. There no other types of transaction between these two groups like merchant to customer or merchant to merchant.

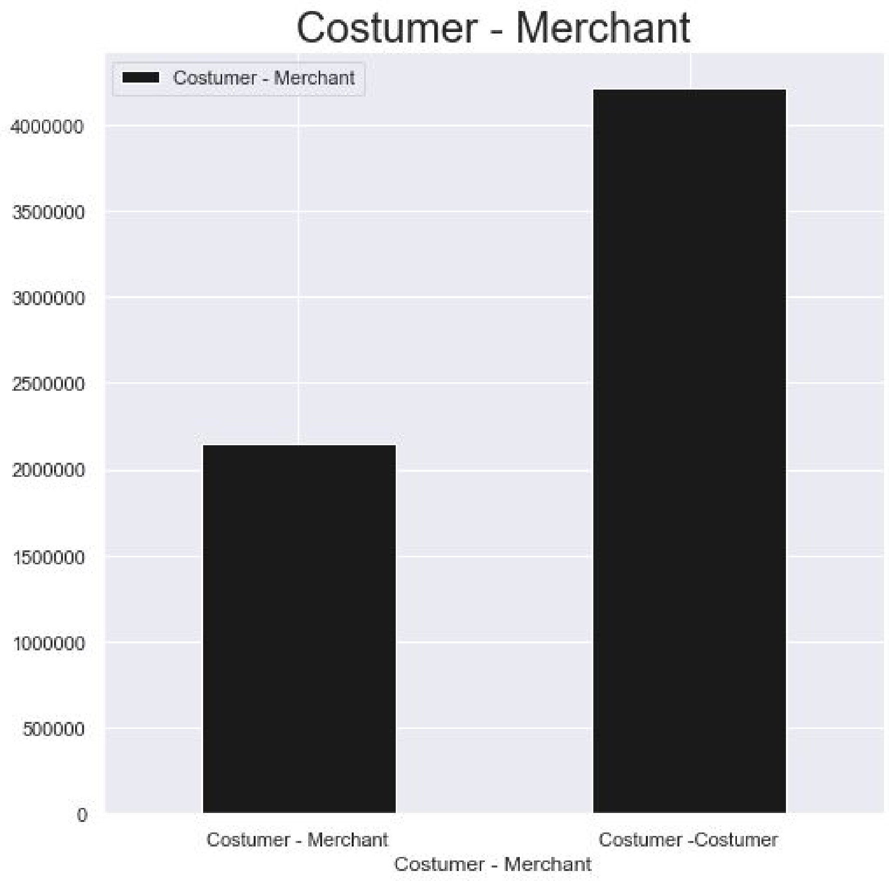


Figure 11. Customer - Merchant Transactions

Among these two groups the fraudulent transactions are showing that only fraudulent transactions took place between customer to customer transactions. This will be encoded for the machine learning.

Graph on the top shows genuine transactions and the one on the bottom shows fraudulent transactions. Transaction between customers makes up the whole fraudulent transaction 100%.

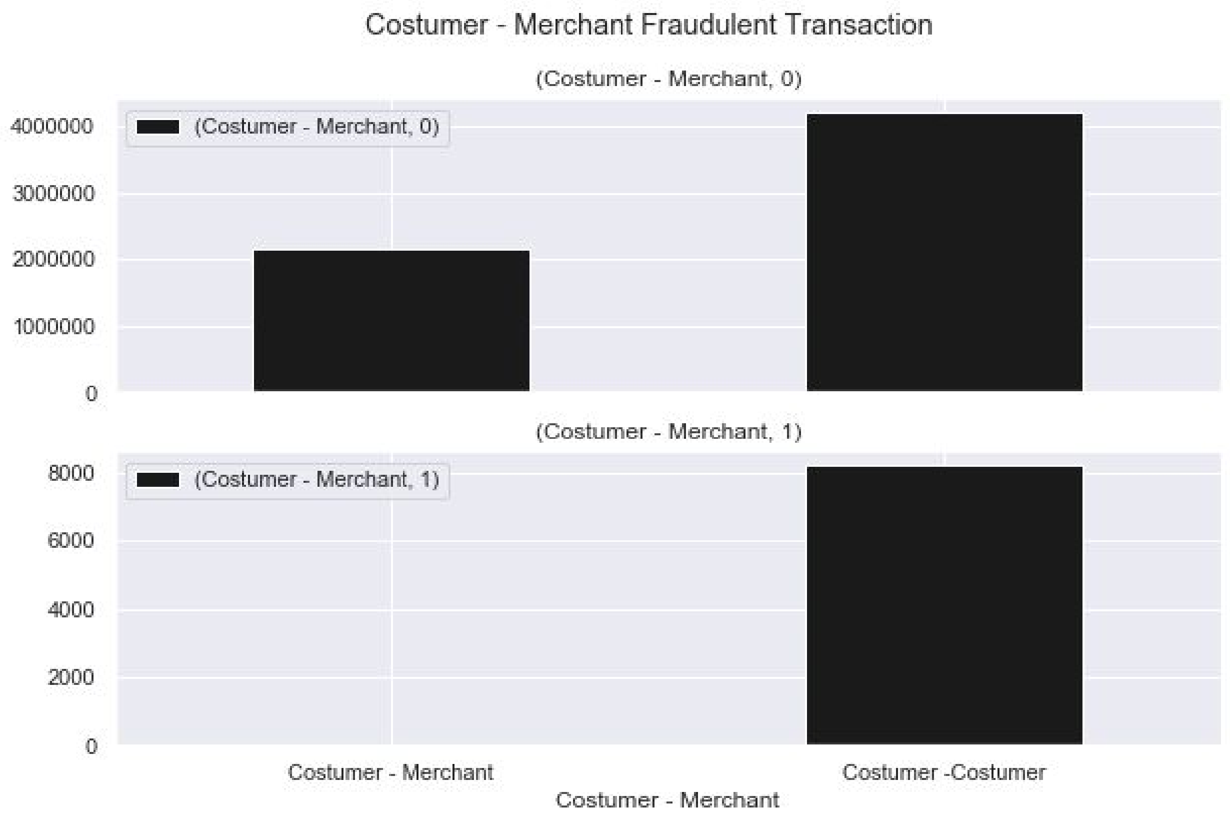


Figure 12. Fraudulent Transaction *Customer to Customer*

4.Modelling

4.1 Part I - Dealing with Imbalanced Dataset

4.1.1 Problem with Imbalanced Datasets

Most supervised machine learning algorithms perform better with balanced data set where size target values are equal. When the data set is so imbalanced like the one used in this project which has non fraudulent transaction about %99.97 and fraudulent transaction is about %0.13. By default some algorithms work fine with imbalance data set like Random Forest, etc., however, in order to maximize accuracy and reduce error, imbalanced data techniques will be used on this dataset.

4.1.2 Problem with Reading Accuracy

Reading accuracy score is not going to help a lot and misleading for most of the time. In order to demonstrate this DummyClassifier() is used. Without even training the data the accuracy score is %99.97. Unique predicted label(s) is [0] that it is non-fraudulent transactions. This is possible because dataset has non fraudulent transaction about %99.97 already. This model did 0% accuracy on predicting non-fraudulent data. Modelling without training can be found on this jupyter notebook.

Elements	
Unique predicted labels	[0] Non Fraudulent
Accuracy Score	%99.97

Table 5. Problem with Reading Accuracy

4.1.3 Changing the Performance Metric

Using accuracy is not very accurate on checking model performance, especially when it comes to evaluating the performance of a model on imbalanced datasets.

In this case using different performance will give better insights

4.1.3.a Confusion Matrix

Confusion matrix is a table that identifies true positives (correct predictions) and false negatives (incorrect predictions) as shown in Figure 9 below. As seen below, label 1 which are fraudulent transactions are not labelled correctly. Confusion matrix helps to visualize how well the data fitted and predicted. According the confusion matrix all data points have been identified as label 0 and 2083 of them labelled incorrectly. 2083 records are probably belongs to label 1.

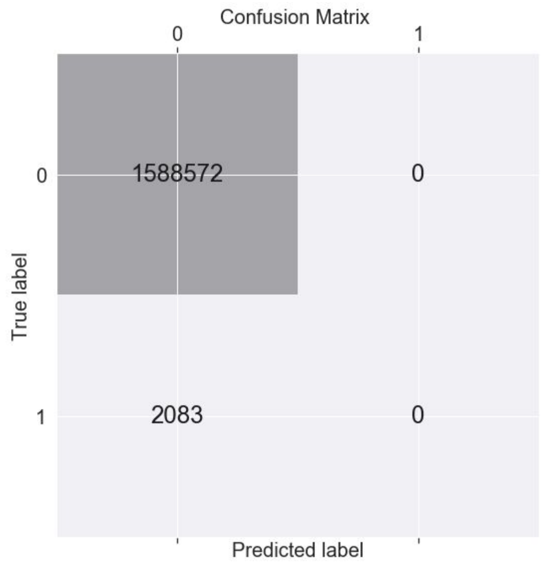


Figure 13. Confusion Matrix

4.1.3.b Precision and Recall

Precision is defined as the number of true positives divided by all positive predictions. Higher precision means more true positives are predicted or labelled correctly. Low precision means the otherwise. This is also called Positive Predictive Value.

Recall is defined as the of true positives divided by the number of positive values in the test data.

4.1.3.c F1: Score:

F1 Score is the weighted average of precision and recall. In this case F1 score is 0. Sample below shows untrained model performs on imbalanced dataset.

	Precision	Recall	F1 Score	Support
Label 0	1	1	1	1588572
Label 1	0	0	0	83

Table 6. Performance Metrics

F1 Performance Metrics	
F1 Score	0

Table 7. Performance Metrics F1 Score

4.1.4.c ROC Curves

ROC curve can be used to select a threshold for a classifier which maximises the true positives, while minimising the false positives.

4.1.3 Class weight

One of the simplest ways to address the class imbalance is to simply provide a weight for each class which places more emphasis on the minority classes such that the end result is a classifier which can learn equally from all classes.

4.1.3 Resampling Techniques

There are 3 types of resampling techniques will be used to test on this unbalanced dataset. Resampling techniques are oversampling minority class, undersampling the majority class, and generating synthetic samples.

4.1.3.a Oversampling Minority Class

Making more copies from the minority class and equalizing the number of each class is called oversampling. For this Scikit-Learn resample module will be used. Scikit-Learn resample is randomly making more copies of minority class under the hood.

4.1.3.a Random Sampling

The most naive method of oversampling is to randomly sample the minority classes and simply duplicate the sampled observations.(https://www.jeremyjordan.me/imbalanced-data/)

4.1.3.b ADASYN

Adaptive Synthetic (ADASYN) sampling works in a similar manner as SMOTE, however, the number of samples generated for a given Xi is proportional to the number of nearby samples which do not belong to the same class as xi. Thus, ADASYN tends to focus solely on outliers when generating new synthetic training examples.

4.1.3.a Undersampling Majority Class

Despite oversampling, undersampling takes out records from majority class to balance training data. This method removes lots of records from the data set, this is something that must be carefully handled.

4.1.3.c Generating Synthetic Samples

Generating Synthetic Samples is creating new records by using an algorithm. This brings training data equalized, same number for both classes fraudulent transactions and non-fraudulent transactions.For generating synthetic samples, Synthetic Minority Oversampling Technique (SMOTE) method from imblearn will be used..

4.1.3.d Tomek's link

Tomek's link exists if two observations of different classes are the nearest neighbors of each other.

4.1.3 Changing Algorithm

Variety of modeling algorithms will be used including KNN neighbors, Logistic Regression, Gradient Boosting, and Random Forest. After the best resampling method and algorithm is chosen this method will be used on a variety of algorithms to check performance metrics Precision, Recall and F1 Score.

4.2 Part II - Resampling, Modelling Algorithms and Model Performances

In the figure 9. PCA of the original data given. This illustrates how imbalanced the dataset is.

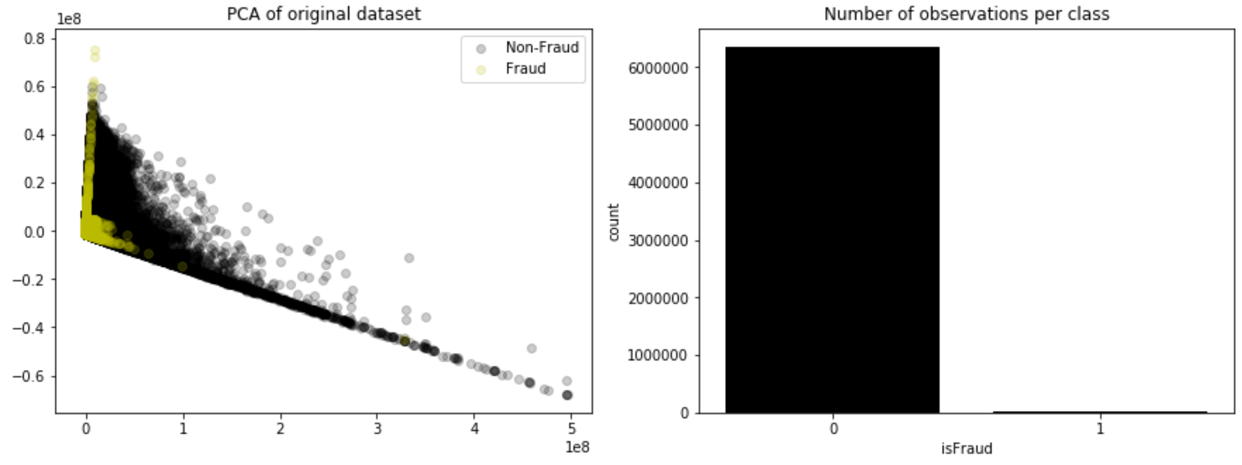


Figure 14. PCA of Original Dataset

4.2.1.a Methodology

Modelling algorithms are performed in an order described below:

Ordinary Techniques

- First each modelling algorithm is tested with its default setting
- Second, each modelling algorithm is tested with its class weight techniques to handle imbalanced dataset.

Oversampling Techniques

- Modelling algorithm is tested with RandomOverSampler from imblear.
- Each modelling algorithm is tested with the help of Synthetic Minority Oversampling Technique (SMOTE) method from imblearn to handle imbalanced dataset.
- Each modelling algorithm is tested with Adaptive Synthetic (ADASYN) sampling that works in a similar manner as SMOTE.

Undersampling Techniques

- Modelling algorithm is tested with RandomUnderSampler from imblear.
- Last, Tomek's link is used for testing the modelling algorithm

Modelling Evaluation Metrics

For each technique listed above per modelling algorithm following metrics are evaluated:

- F1 Score
- Cohen Kappa
- Brier
- Area Under the Curve
- Confusing Matrix

After comparing each matrix and the performance final sampling technique and modelling algorithm will be used.

Final modeling algorithm and sampling technique will be also tested on similar datasets to predict fraud and evaluate modeling performance.

4.2.2.a Decision Tree

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

	F1 Score	Cohen Kappa	Brier	AUC	Precision TP/(TP+FP)	Recall TP/(TP+FN)
Ordinary	0.83	0.83	0.0004	0.90	0.86	0.80
Class Weight	0.79	0.79	0.005	0.87	0.83	0.75
Oversampling	0.79	0.79	0.005	0.87	0.85	0.74
SMOTE	0.71	0.71	0.009	0.94	0.59	0.89
ADASYN	0.71	0.71	0.009	0.94	0.59	0.87
RandomUnder Sampler	0.10	0.10	0.02	0.98	0.05	0.98
TomekLinks	0.77	.077	0.005	0.87	0.82	0.73
EditedNearest Neighbours	.79	.79	0.005	0.88	0.82	0.75

Table 8. Decision Tree Metrics

According to the Performance metrics ordinary (defaut) setting of the decision tree worked the best consistently. Decision tree is working fine with imbalanced data set by default. Other techniques worked well are class weight, RandomOverSampling, RandomUnderSampler EditedNearestNeighbours. SMOTE and ADASYN oversampling techniques did not worked very well. They probably caused an overfitting.

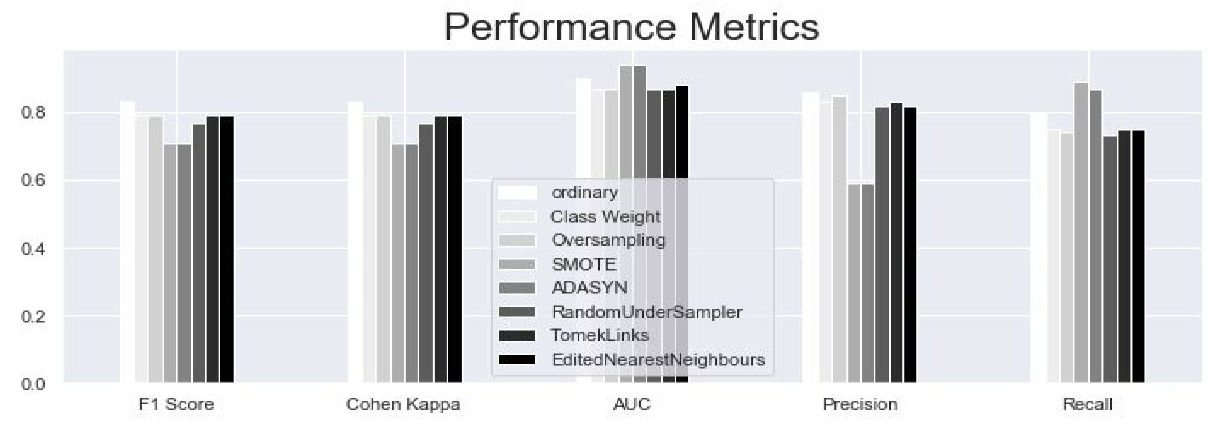


Figure 15. Decision Tree PerformanceMetrics

4.2.2.b Logistic Regression

Logistic Regression is used when the dependent variable(target) is categorical. In this case our categorical target variable is fraudulent transactions where one being fraudulent transactions and zero being non-fraudulent transactions.

	F1 Score	Cohen Kappa	Brier	AUC	Precision TP/(TP+FP)	Recall TP/(TP+FN)
Ordinary	0.39	0.39	0.002	0.72	0.36	0.45
Class Weight	0.05	0.05	0.001	0.95	0.03	0.95
Oversampling	0.03	0.03	0.08	0.91	0.01	0.90
SMOTE	0.03	0.03	0.06	0.93	0.02	0.92
ADASYN	0.02	0.02	0.08	0.94	0.01	0.97
RandomUnderSampler	0.02	0.02	0.08	0.91	0.01	0.90
TomekLinks	0.37	.37	0.001	0.71	0.33	0.42
EditedNearestNeighbours	0.37	.37	0.001	0.71	0.33	0.42

Table 9. Logistic Regression Metrics

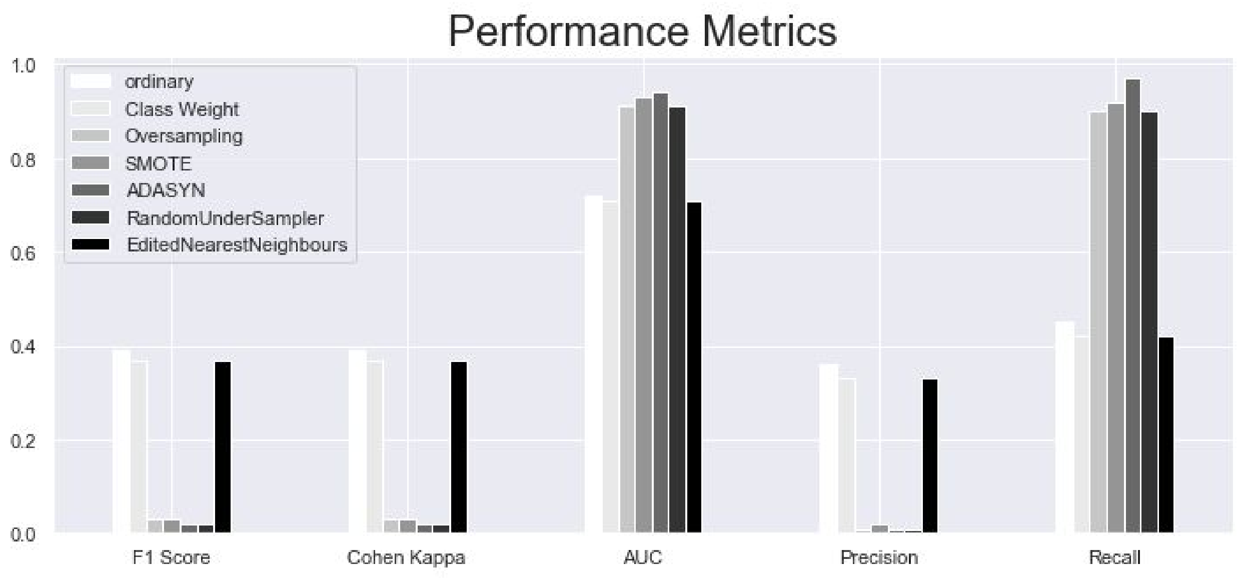


Table 16. Logistic Regression Metrics

Logistic Regression did not provide sufficient results for this dataset. Even though ordinary (default settings), RandomUnderSampler, Edited Nearest Neighbours worked slightly better, decision tree provided better results. Using Logistic Regression for this dataset will not be suitable at this time.

4.2.2.c RandomForestClassifier

Random Forest Classifier is ensemble algorithm.Ensembled algorithms are those which combines more than one algorithms of same or different kind for classifying objects.By default they handle imbalanced data set pretty well.

	F1 Score	Cohen Kappa	Brier	AUC	Precision TP/(TP+FP)	Recall TP/(TP+FN)
Ordinary	0.83	0.83	0.003	0.86	0.98	0.72
Class Weight	0.82	.82	0.004	0.85	.98	.71
Oversampling	0.83	0.83	0.003	0.87	.95	.74
SMOTE	0.74	0.74	0.008	0.93	0.65	0.86

	F1 Score	Cohen Kappa	Brier	AUC	Precision TP/(TP+FP)	Recall TP/(TP+FN)
ADASYN	0.73	0.73	0.008	0.94	0.63	0.87
RandomUnderSampler	0.1	0.1	0.02	.98	0.05	0.99
TomekLinks	0.80	.80	0.004	.84	.97	.64
EditedNearestNeighbours	0.82	.82	0.004	.85	.98	70

Table 10. Random Forest Metrics

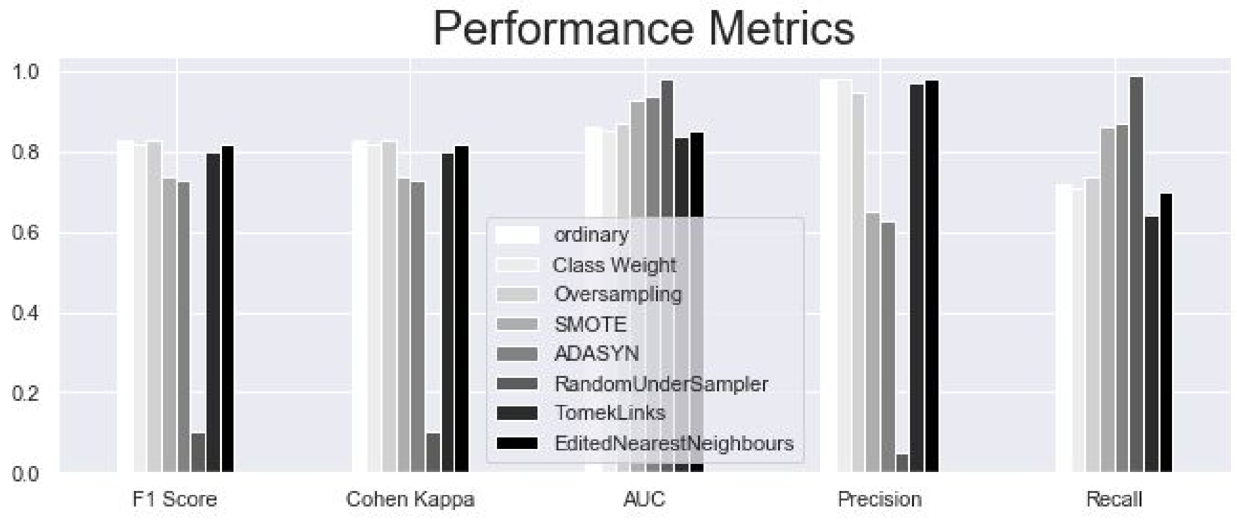


Table 17. Random Forest Metrics

Ordinary (default setting) worked very well constantly. This shows that Random Forest handle imbalanced dataset efficiently. Oversampling techniques provided higher AUC meaning with lower F1 scores that they probably caused overfitting. Undersampling techniques worked best for the Random Forest.

4.2.2.d KNeighborsClassifier

"In KNeighborsClassifier, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.
The basic idea is that if point A is very distant from point B, and point B is very close to point C, then we know that points A and C are very distant, without having to explicitly calculate their distance." (Machine Learning 101)

	F1 Score	Cohen Kappa	Brier	AUC	Precision TP/(TP+FP)	Recall TP/(TP+FN)
Ordinary	0.66	0.66	0.006	0.76	0.91	0.52
Class Weight	0.70	0.70	0.006	0.79	0.91	0.59
Oversampling	0.59	0.59	0.001	0.84	0.51	0.68
SMOTE	0.18	0.18	0.008	0.88	0.10	0.77
ADASYN	0.18	0.17	0.08	0.88	0.10	0.77
RandomUnderSampler	0.02	0.02	0.09	0.88	0.01	0.85
TomekLinks	0.66	.66	0.006	0.76	0.90	0.52
EditedNearestNeighbours	0.66	.66	0.006	0.77	0.85	0.53

Table 11. KNeighborsClassifier Metrics

According to the metrics above Ordinary (default settings), Class Weight, TomekLinks, EditedNearestNeighbours were the best techniques for KNeighborsClassifier. Oversampling methods did not do well for KNeighborsClassifier as well. They provide higher AUC scores, however other metrics shows that they are misleading.

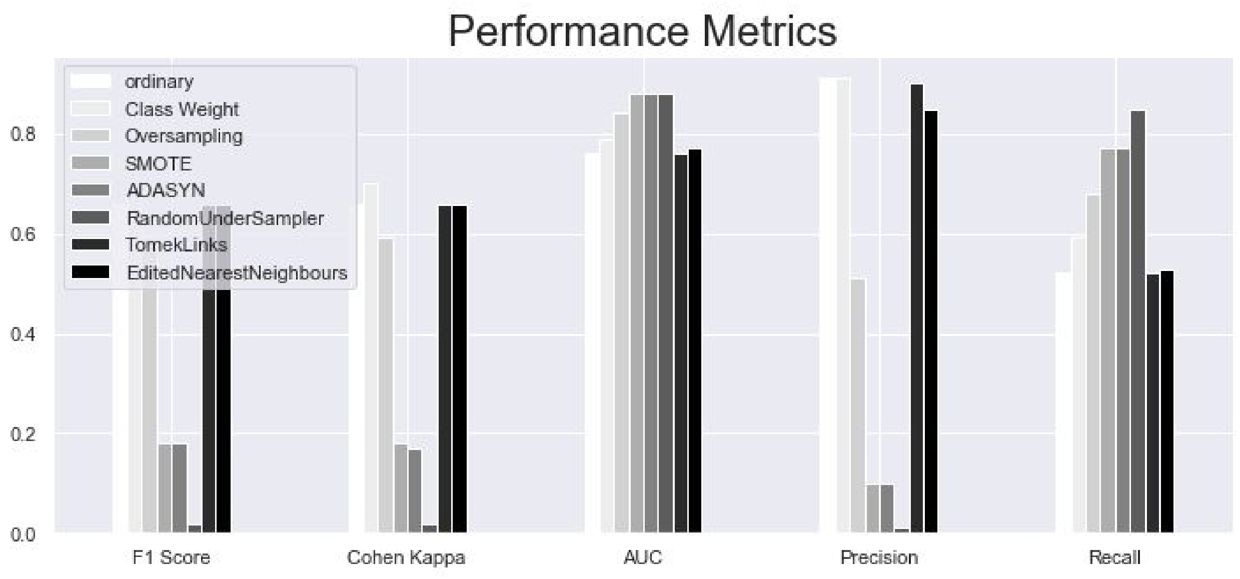


Figure 18. KNeighborsClassifier Metrics

Figure 10 provides an easy visual representation of best metrics to use. In this case class weight makes more sense. In this modelling class weight is set to weight = ‘distance’ that means it weights the points by the inverse of their distance. In this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.

4.2.2.d Model Comparison

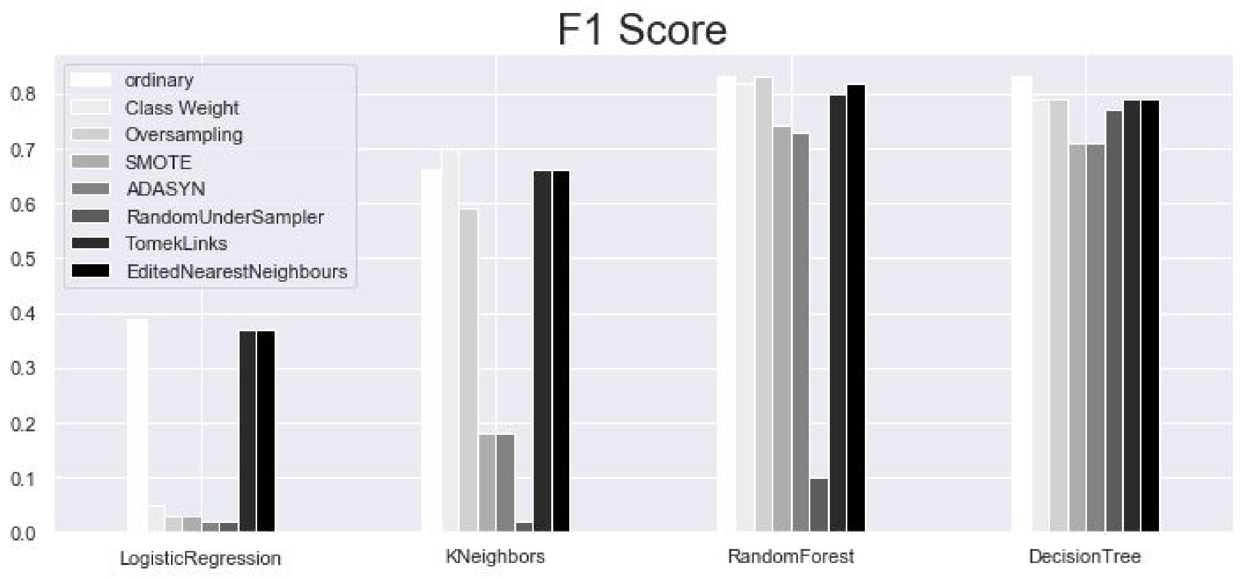


Figure 19. F1 Score Comparison

Random Forest and Decision Tree has the highest F1 scores. Both oversampling and undersampling provided higher F1 scores, however, default parameter setting Random Forest and Decision Tree provided better scores.

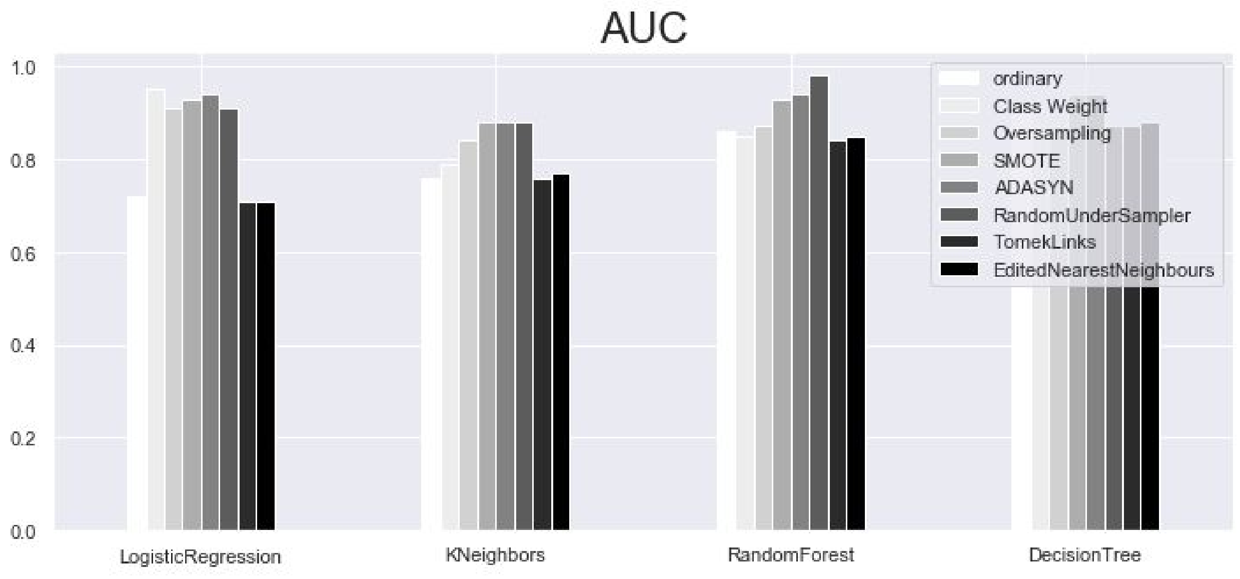


Figure 20. AUC Comparison

Area under the curve (AUC) seems very high for almost all of the modelling algorithms. AUC is not the only metrics to determine the overall performance of the modelling algorithms. Again AUC is also showing that Random Forest is the best model. AUC looks very high for the oversampling techniques which provided lower F1 scores. Oversampling techniques may have caused overfitting the model.

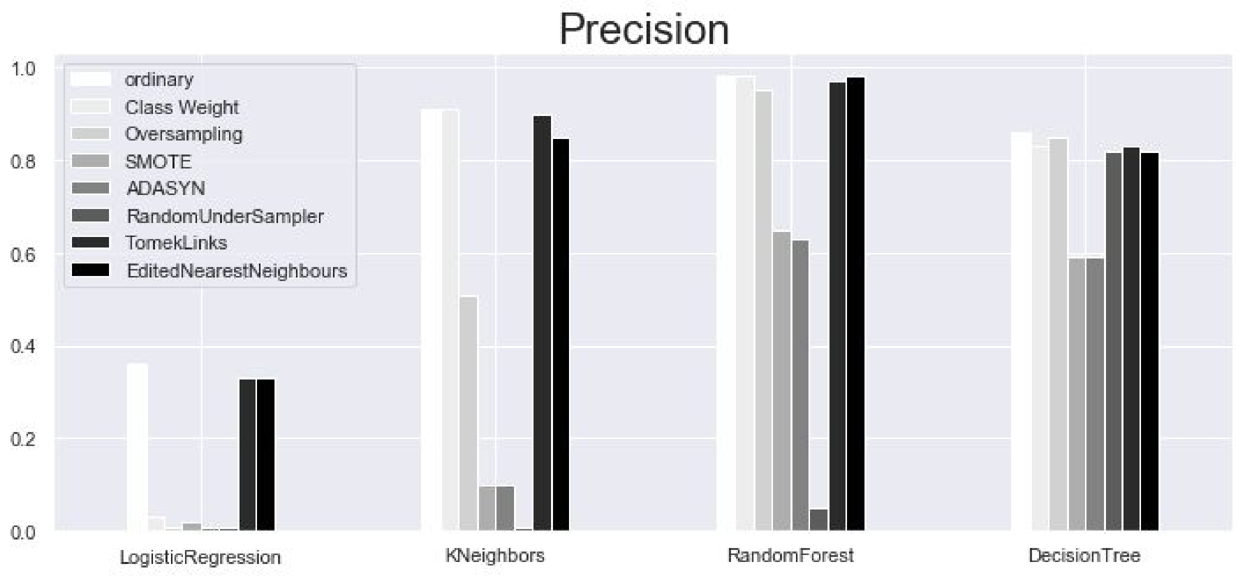


Figure 21. Precision Comparison

According to the precision metrics Random Forest is the best modelling algorithm for this dataset. As a result, after evaluating F1 scores, AUC and precision metrics Random Forest will be the modelling algorithm for the final stage.

Recall is the ability of a model to find all the relevant cases within a dataset. The precise definition of recall is the number of true positives divided by the number of true positives plus the number of false negatives.

According to the *Figure 12* SMOTE and ADASYN oversampling techniques provided better recall scores with higher precision.

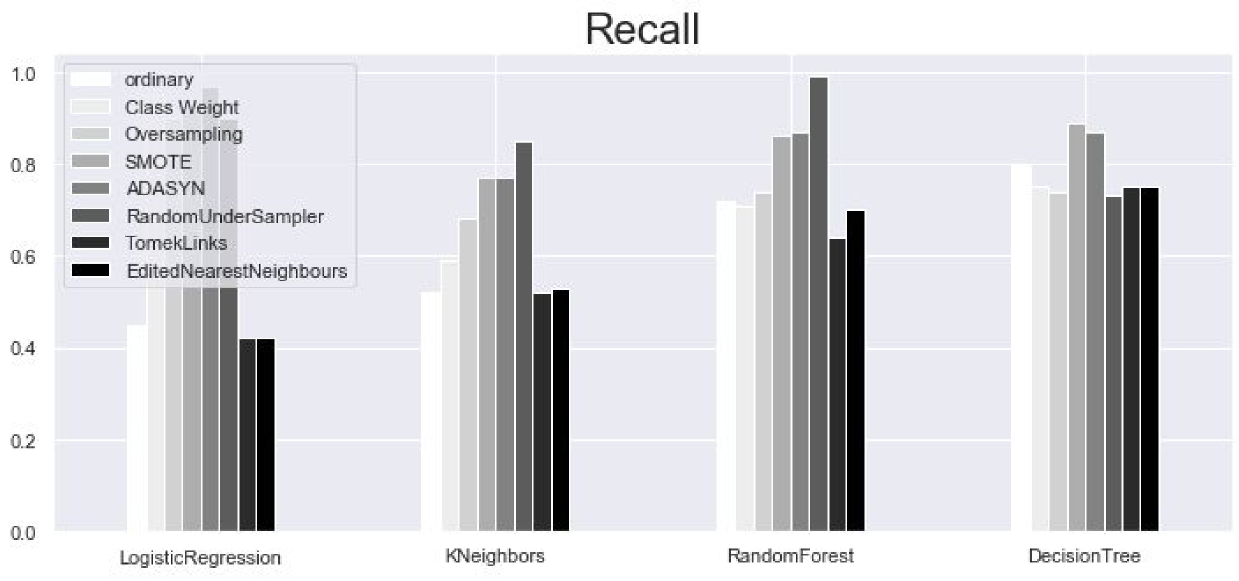


Figure 22. Recall Comparison

4.2.2.e Trade-Off

Choosing the right metrics for classification tasks is very important for modelling, especially for the imbalanced datasets. This data set consists of 99% non fraudulent transactions.Does this mean that this model is 99% accurate for detecting fraudulent transactions. In fact, this model would have done a terrible job identifying fraudulent transactions, if right metrics are not used. Accuracy, AUC and F1 scores are very useful however they are not enough to identify all number of true positive class.

In order to maximize recall, or the ability of a model to find all the relevant cases within a dataset. True positives are data points classified as positive by the model that actually are positive (meaning they are correct), and false negatives are data points the model identifies as negative that actually are positive (incorrect).[1]

$$recall = \frac{true\ positives}{true\ positives\ + false\ negatives}$$

In this case, when data set is imbalanced, there will be trade off in order to maximize to find all the relevant cases. Increasing recall means decreasing precision or the ability of a classification model to identify only the relevant data points[2]

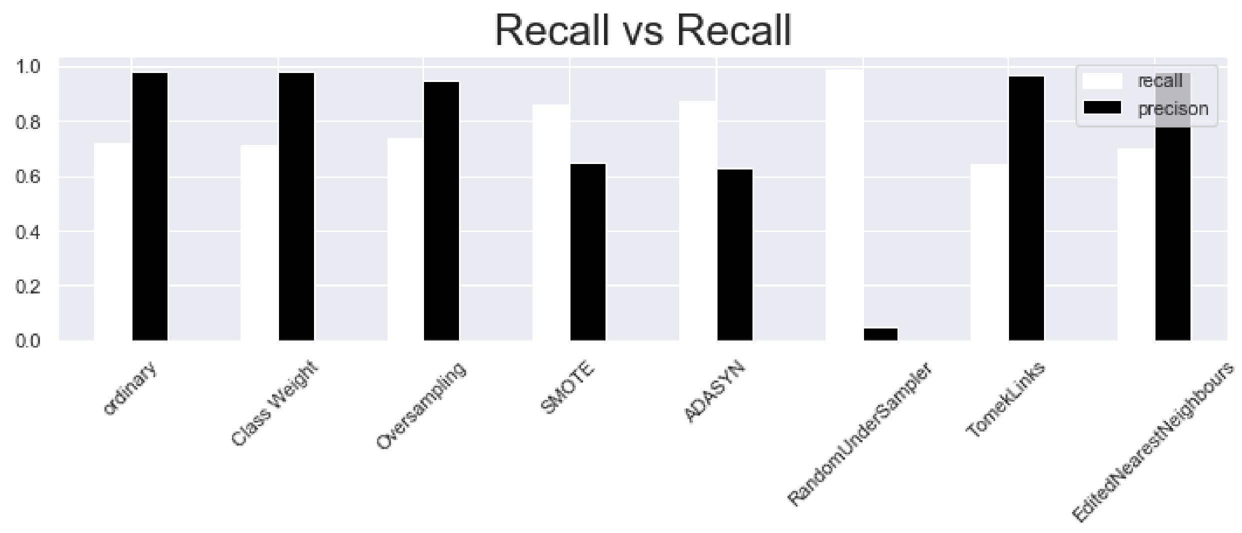


Figure 23. Recall vs Precision

[1] "Beyond Accuracy: Precision and Recall - Towards Data Science." 3 Mar. 2018, https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c. Accessed 21 Jul. 2019.
[2] "Beyond Accuracy: Precision and Recall - Towards Data Science." 3 Mar. 2018, https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c. Accessed 21 Jul. 2019.

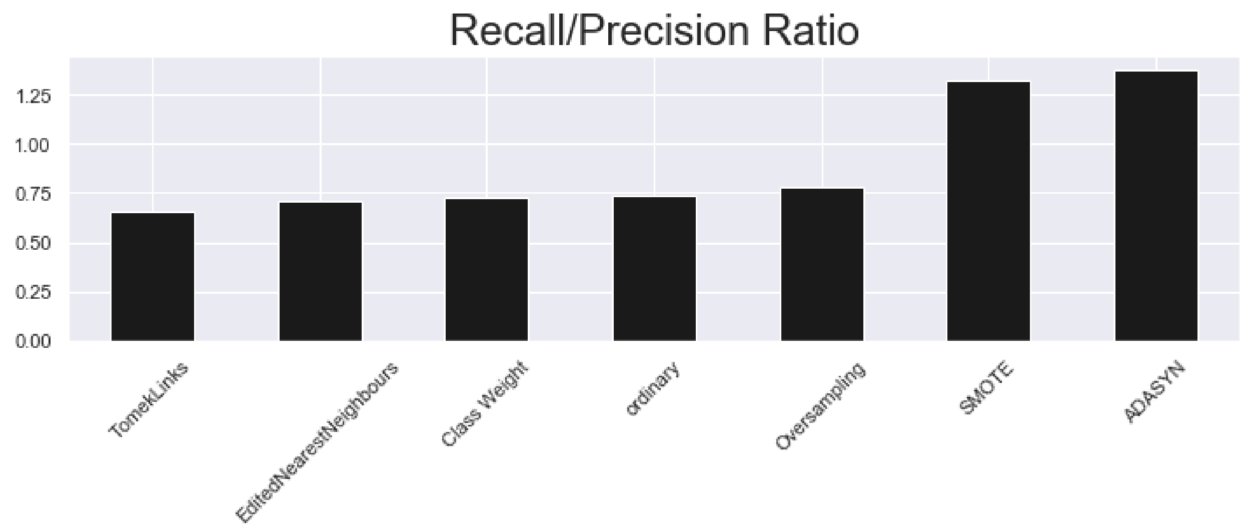


Figure 24. Trade Off

Based on the Figure 14. SMOTE and ADASYN seems to be the best techniques that improves model algorithm to classify the relevant data points as the trade off.

4.3 Model Hyperparameter Tuning

Random Forest is chosen to be final modelling algorithm. Now it is time to adjust hyperparameter to improve the performance of the model. Here are some of the parameters will be adjusted:

- n_estimators is the number of trees will be constructed. Default number is 10.
- max_depth is the maximum depth of each tree this continues until each tree is pure
- min_samples_split is number of samples to be split at each leaf node
- min_samples_leaf number of samples at each leaf node

4.3.a Before Trade Off

Here is the best parameters without applying trade off between (applying the oversampling technique)

Mean validation score: 1.000 (std: 0.000)

Parameters: {'bootstrap': True,
'criterion': 'gini',
'max_depth': None,
'max_features': 7,
'min_samples_split': 7}

Target	F1 Score	Precision	Recall	AUC
Fraudulent Transaction	0.86	0.97	0.77	0.99

Table 12. Performance Metrics

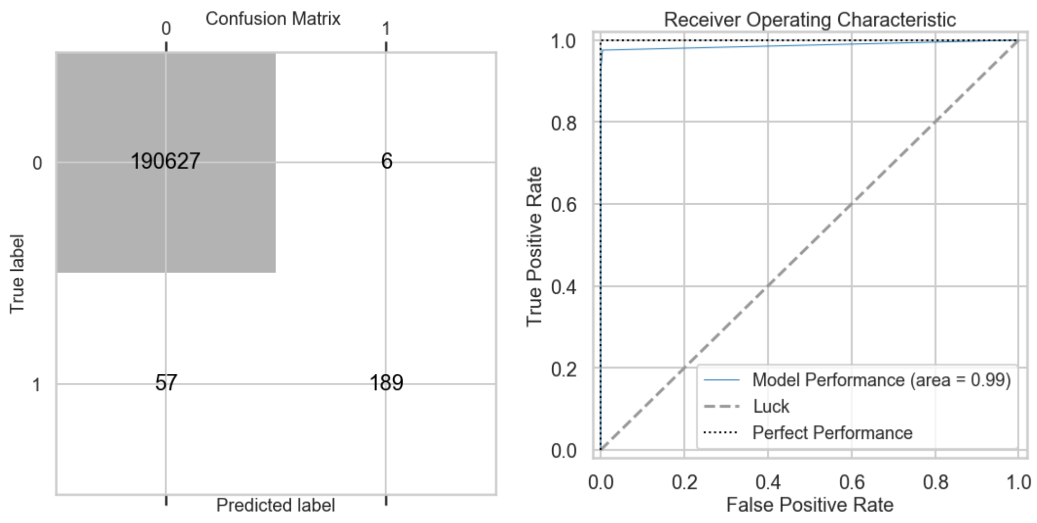


Figure 25. Performance Metrics

57 out of 246 target samples are misclassified. Precision of the model algorithm look fine however recall needs to be improved. There is a trade off required here.

4.3.b After Trade Off

When SMOTE is applied as a resampling technique, here are the best parameters:

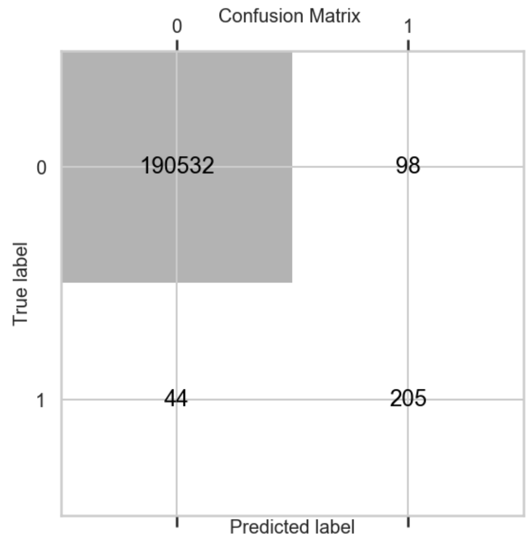
Mean validation score: 1.000 (std: 0.000)

```
Parameters: {'bootstrap': False,
             'criterion': 'gini',
            'max_depth': None,
             'max_features': 2,
            'min_samples_split': 2}
```

As expected precision has decreased and recall has increased. The ability of a model to find all the relevant cases within a dataset has slightly increased. It classified more true positives as true positives and sacrificed some of true negatives as true positive.

Target	F1 Score	Precision	Recall	AUC
Fraudulent Transaction	0.74	0.68	0.82	0.91

Table 12. Performance Metrics



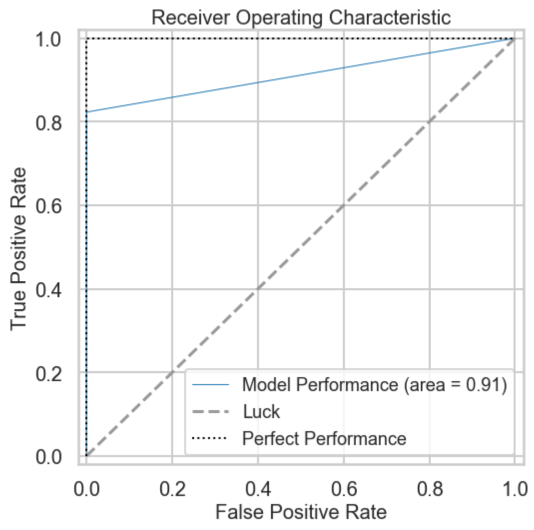


Table 26. Performance Metrics

5. Future Connection

User history is also very important for detecting fraudulent transactions. There are lots of determining factors in terms of fraudulent transactions. This dataset only includes transfer types and transfer amounts. Following items will be considered to accurately predict fraudulent activities, including the items below but not limited to:

- Shipping address does not match the billing address
- Purchaser attempts to circumvent your usual payment process (e.g. sending credit card information via email rather than entering it on your website)
- Order is for an unusually large amount of items
- Purchaser wants items rushed or shipped next day shipping (this is not necessarily suspicious on its own but is very suspicious along with other red flags)
- Order is from another country – particularly if you sell items that could be easily obtained in any country
- Customer tries different expiration dates after initial decline
- Customer purchases large amount of the same item
- Multiple orders come in with the same shipping address but different cards

6.Conclusion

6.1 Exploratory Data Analysis

In data exploration part of this project, each column and its relationship with the fraudulent transactions were analysed. There were five transaction types presented in the dataset which were debit, transfer, cash-out, payment, cash-in. Cash-out was the most common transaction type and using debit for the transaction is the least common type. There were 6354407 transactions of which only 8213 were identified as a fraudulent transaction. The ratio of fraudulent transaction is %0.13. Fraudulent transactions were happened only during CASH OUT and TRANSFER transactions. Majority of the fraudulent transaction between $1M to $100K and higher than higher than $1M.All fraudulent transaction took place between Customer to Customer transactions. Transaction between customers makes up the whole fraudulent transaction 100%.

6.2 Modelling

Supervised machine learning and classification models are used for the flight cancellation like Decision Tree, Random Forest, KNeighbor, and Logistic Regression. Data was split by 70% to train a predictive model and 30% to test the data.

After testing and tuning hyperparameters, Random Forest was the best model to classify a set of data about probability of being fraudulent transaction. Model performance metrics evaluated are accuracy, F1 score, precision, recall and AUC. In order to maximize the recall, or the ability of a model to find all the relevant cases within a dataset trade off techniques were used. SMOTE (oversampling) technique worked the best.