Udacity Machine Learning Nanodegree
Mobile Payments Fraud Detection

Venkat Maddi

July 9th, 2018

I. Definition

Project Overview

Nowadays, people are extensively using mobile devices to handle financial transactions. Banks and Financial industry experts are predicting that customers will utilize mobile devices to initialize payments extensively this year [1]. Financial institutions are constantly working on various methods to improve the customer experience, execute the payments faster and safer [2]. Banks have introduced Reward Points to encourage customers to complete the payments electronically.

Accenture Consulting Study indicates that the Gen Z, new generation adults and young people today, will make up to 40 percent of USA population by 2020. The Gen Z customers are more comfortable with executing transactions from mobile devices [3]. The mobile based payment transactions will grow exponentially in the next few years.

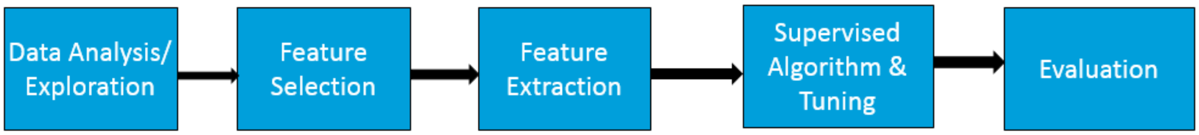
Problem statement

It is very important to detect fraudulent transactions while processing mobile payments. It is not possible to detect the fraudulent transactions manually because of huge volume of transactions banks handle hourly and daily. Researches and Data scientists are creating new algorithms and introducing new processes to detect the fraud as soon as fraudulent transaction hits the financial institutes.

Normally, the financial institutions do not publish mobile money transactions. Kaggle platform [4] has provided a synthetic dataset generated using the simulator called PaySim as an approach to detect the fraudulent transactions. [5]. PaySim uses aggregated data from the "private dataset" to generate a synthetic dataset that resembles the normal operation of transactions and injects malicious behavior to later evaluate the performance of fraud detection methods. The private dataset is based on real transactions from a mobile money services implemented in African country.

I have decided to work on Machine Learning algorithm to detect the mobile payments' fraud. My project will implement Supervised Learning Binary Classification techniques to detect the fraudulent transactions. Also, I will utilize fraud detection dataset available on Kaggle website at https://www.kaggle.com/ntnu-testimon/paysim1.

The high-level design activities and workflow while implementing the Supervised Learning models to predict fraudulent mobile payments are as follows:



Workflow Diagram

1) **Data Analysis/Exploration**: Data Analysis is a very important and key activity of the Machine Learning model creation. The activities within this phase are as follows:
    - Review and understand Data
    - Identify Data thresholds like minimum, maximum and etc
    - Determine Data mean, standard deviation
    - Load required libraries and data files

2) **Feature Selection:** The dataset contains multiple data fields/columns. The data field is considered as a feature.
    - Review all features included in the datafile.
    - Identify Feature Dependency. Some of the features are critical to determine the prediction. If a feature is dependent on another primary feature, then the primary feature must exist in the model.
    - Reduce features to a reasonable number. Eliminate least important features which do not cause major impact to the prediction.
    - Select Best Features from dataset.

3) **Feature Extraction**
    - Review the feature's data distribution and ranges. If the data range (difference between min and maximum values) is wide, then Normalize the data using Logarithm transformation.
    - Normally, Numeric values tend to tune models more effectively. Kaggle Paysim dataset contains features with non-numeric values.
    - Split data into training and testing
    - Assign a portion of training data to the Validation activity using cross-validation technique

4) **Supervised Learning Classification Algorithms & Tuning**
    - Create Accuracy and F-score bench marks using Naïve Bayes model.
    - Implement at least five (5) Supervised models.
    - Train the Model with the Training data
    - Tune the Algorithm by modifying Hyper parameters

5) **Evaluation**
    - Execute Model using the Testing data
    - Analyze the results and Model performance
    - Identify a best supervised Model which provides the maximum performance results, high accuracy, and high F-score.

Metrics

Each Supervised Model performance is calculated using the statistical concepts, Classification Accuracy, Recall and ROC AUC. The following table provides confusion matrix definitions.

	Predicted as Fraud	**Predicted as Genuine**
Fraud Transaction	True Positive (TP)	False Negative (FN)
Genuine Transaction	False Positive (FP)	True Negative (TN)

Table 1: Confusion Matrix

True Positive (TP): Transaction is Fraud and Model has predicted as Fraud accurately.
False Negative (FN): Transaction is Fraud and Model has predicted as Genuine incorrectly
False Positive (FP): Transaction is Genuine, and Model has predicted as Fraud incorrectly
True Negative (TN): Transaction is Genuine, and Model has predicted as Genuine accurately

The **Accuracy** measures how often the Model makes the correct prediction. It's the ratio of the number of correct predictions to the total number of data points. The dataset is imbalance, number of fraud transactions are far less compare to the total number of records. The Accuracy is not a suitable Metrics for Bank Fraud Modeling.

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP}$$

The **Precision** tells us what proportion of transactions Model predicted as fraud, actually were fraud. It is a ratio of True Positives to True Positives Plus False Positives.

$$\textbf{Precision} = \frac{TP}{TP+FP}$$

The **Recall(sensitivity)** or **True Positive Rate (TPR)** indicates what proportion of actual fraud transactions is predicted by the Model as fraud. It is a ratio of True Positives to True Positives Plus False Negatives.

$$\textbf{True Positive Rate (TPR) = Recall} = \frac{TP}{TP+FN}$$

False Positive Rate (FPR) indicates what portion of Genuine records are predicted incorrectly.

$$\textbf{False Positive Rate (FPR)} = \frac{FP}{FP+TN}$$

ROC (Receiver Operating Characteristic) Curve is a graph displaying the performance of a classification model at various thresholds. This curve plots two parameters, True Positive Rate and False Positive Rate.

AUC (Area Under the ROC Curve) is a two-dimensional area underneath the entire ROC curve. ROC and AUC metrics are suitable for imbalance data like fraud transactions [10]. Therefore, I will calculate ROC AUC to determine the Model efficiency.

II. Analysis

Data Exploration

The input dataset financial mobile based transactions provided by Kaggle platform. The dataset contains input attributes (AKA features) and the Fraud attribute (Target). The file contains more than six million records. Each record consists of both input attributes (features) and output variable. The classification goal is to predict whether mobile payment is a fraudulent or not.

Input Variables (Features):

Data Attribute #	Attribute Name	Description
1	Step	It maps a unit of time in the real world. In this case, step 1 represents First hour of transactions
2	Type	Transaction Type, CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER
3	Amount	Transaction Amount in local currency
4	nameOrig	The customer who initiated the transaction
5	oldbalanceOrg	The initial balance before the transaction
6	newbalanceOrig	The new balance after processing the transaction.
7	nameDest	The customer who is the recipient of the payment
8	oldbalanceDest	The initial balance in the recipient account before the transaction. Note that there is not information for customers that start with M (Merchants).
9	newbalanceDest	The new balance in the recipient account after processing the transaction. Note that there is not information for customers that start with M (Merchants).
11	isFlaggedFraud	If a transfer amount is more than 200,000 then single transaction flags as illegal attempt. The business model flags the transaction as "illegal Attempt" for higher denominations.

Table 2: Feature Details

Output Variable (Target)

The 10^{th} attribute, *isFraud*, is an output variable. The output variable valid values are either zero (0) or one (1). If the output variable value is zero, then the data record is categorized as a genuine transaction. If the output variable value is one, then the data record is categorized as a Fraudulent transaction.

Data Attribute #	Attribute Name	Description
10	isFraud	Value values are either 0 or 1. The value 1 indicates that this transaction was created by the fraudulent agent inside the simulator

Table 3: Target Column Details

Missing Attributes: The recipient account's old balance and new balance attributes do not have values for all records. If the recipient (destination customer) name starts with M(Merchants), then destination account old balance and destination new balance attributes are zero.

Categorical Features: The second column in the datafile is Type and it explains the transaction category. There are six types of transactions exist in the dataset. These transaction Types are CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.

The dataset input file contains 6,362,620 rows and 11 columns. It represents 10 features (input variable) and one target column (output). The number of Fraud records count in the dataset is 8,219. It translates 0.1291% of records are fraud payments. The percentage of fraud records are less than one percent.

Few sample records from the dataset are as follows:

type	amount	nameOrig	oldbalance Org	nameDest	oldbalance Dest	isFraud	isFlagged Fraud
PAYMENT	9839.64	C1231006815	170136.00	M1979787155	0.0	0	0
PAYMENT	1864.28	C1666544295	21249.00	M2044282225	0.0	0	0
TRANSFER	181.00	C1305486145	181.00	C553264065	0.0	1	0
CASH_OUT	181.00	C840083671	181.00	C38997010	21182.0	1	0
PAYMENT	11668.14	C2048537720	41554.00	M1230701703	0.0	0	0
PAYMENT	7817.71	C90045638	53860.00	M573487274	0.0	0	0
PAYMENT	7107.77	C154988899	183195.00	M408069119	0.0	0	0
PAYMENT	4024.36	C1265012928	2671.00	M1176932104	0.0	0	0
DEBIT	5337.77	C712410124	41720.00	C195600860	41898.0	0	0

Table 4: Sample Records from Dataset

The following table explains the Statistical info. The amount and balance features contain a higher standard deviation.

step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	isFraud
count	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06
mean	2.433972e+02	1.798619e+05	8.338831e+05	8.551137e+05	1.224996e+06
std	1.423320e+02	6.038582e+05	2.888243e+06	2.924049e+06	3.674129e+06
min	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	1.560000e+02	1.338957e+04	0.000000e+00	0.000000e+00	0.000000e+00
50%	2.390000e+02	7.487194e+04	1.420800e+04	0.000000e+00	2.146614e+05
75%	3.350000e+02	2.087215e+05	1.073152e+05	1.442584e+05	1.111909e+06
max	7.430000e+02	9.244552e+07	5.958504e+07	4.958504e+07	3.561793e+08

Table 5: Dataset Statistical Info

Exploratory Visualization

I have created various graphs to visualize and identify dependency across the features. Figure 1 shows the record count by transaction type. The number of CASH_OUT and PAYMENT records counts is more than 2 Million each. The CASH_IN records are around 1.5 Million. The TRANSFER records around 500,000. However, The DEBIT records count is much lower less than 45,000 records.

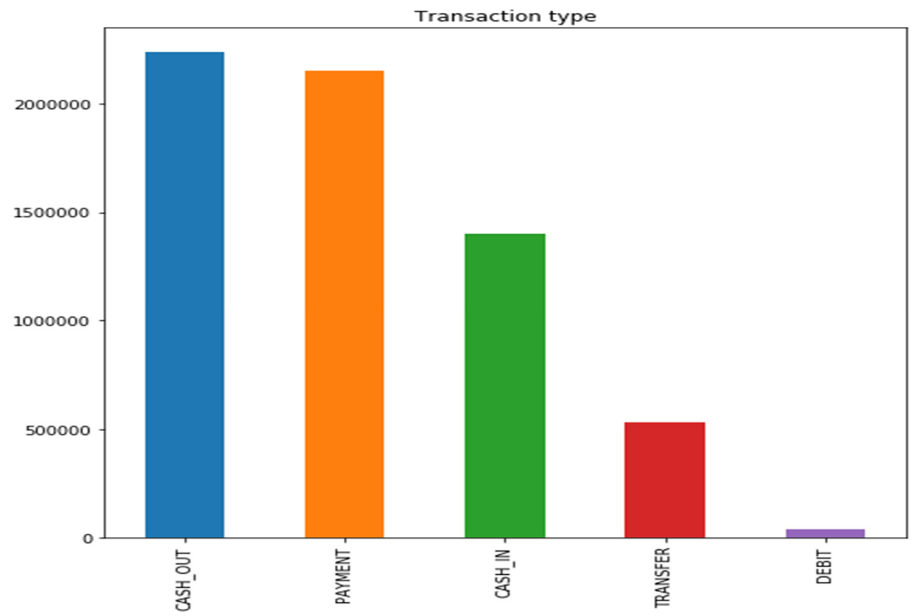


Fig 1: Summary by Transaction Type

I have stared reviewing Fraud transaction types. It seems, the Fraud records have been identified in two types of records, CASH_OUT and TRANSFER. The remaining three types do not have fraud records. Figure 2 shows the Fraud record count by transaction type:

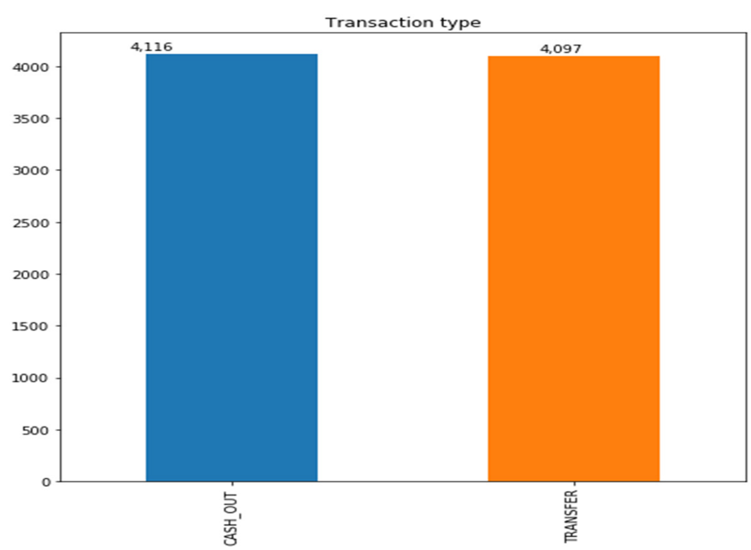


Fig 2: Fraud Transactions Count

The scatter matrix graph indicates the data relation between each pair of columns. Fig 4 shows the Amount, Old balance, New Balance and Fraud columns data relation. Most of the data elements are concentrated on left without having a normal distribution. The data points for Target Column (isFraud) are concentrated on either left side or complete right side.

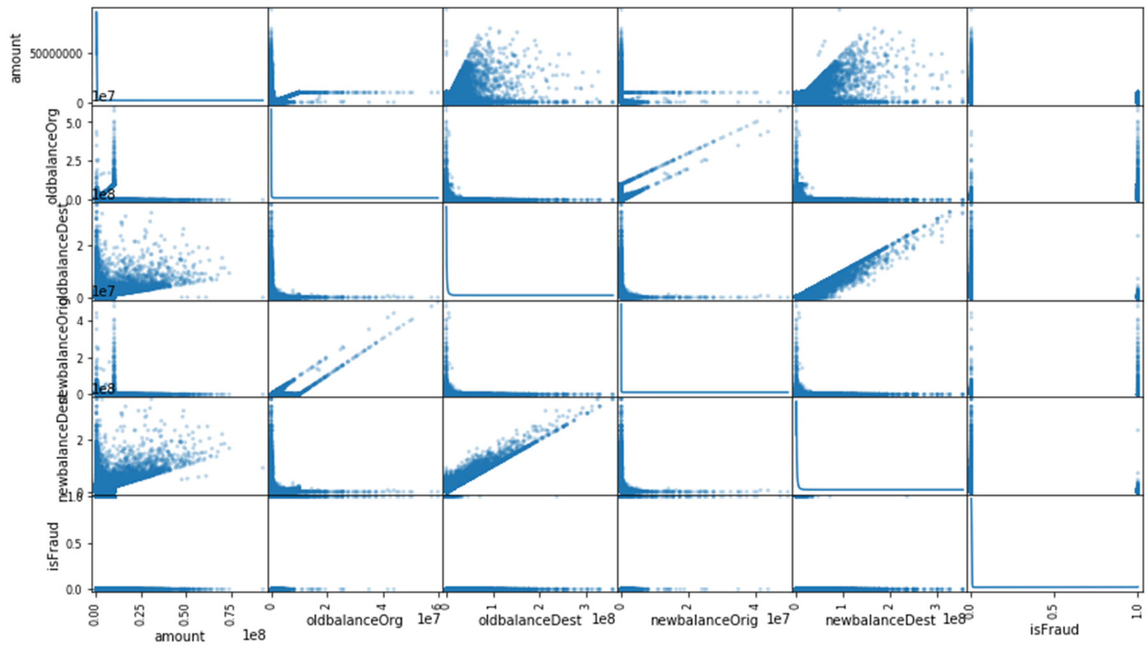


Fig 4: Amount distribution

Figure 5 shows that there is no correlation between the features except new balance features. The old Destination account balance feature is related to the new Destination account balance. Also, Old Origin account balance is related to the new Origin Account balance:

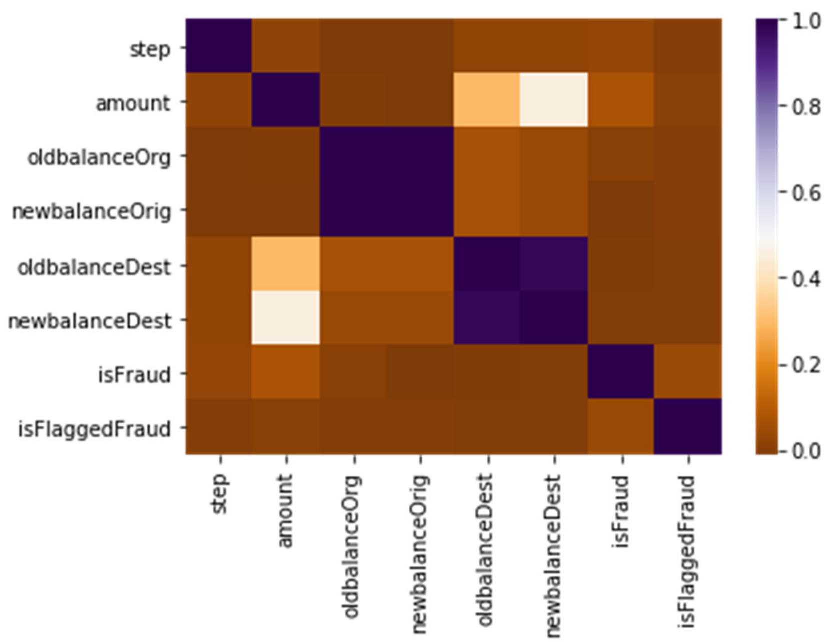


Fig 5: Features Correlation

Algorithms & Techniques

The Decision Tree algorithms are vastly utilized for Supervised Learning classifiers. The dataset contains millions of records, and each record consist of input variables (features) and output variable (target). I will execute multiple Supervised Learning classifiers with various hyper parameters to predict the fraudulent transactions. Here are the proposed classifier details:

- **Decision Tree Classifier**: The decision tree classifiers organize a series of test questions and conditions in a tree structure. In the decision tree, the root and internal nodes contain attribute test conditions to separate the records that have different characteristics. All the terminal node is assigned a class label either Yes or No. The following figure shows identify person's credit rating after verifying the age.

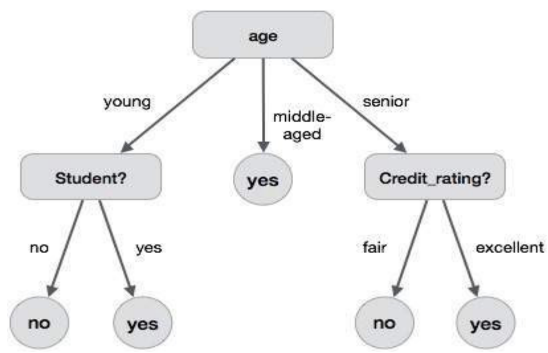


Fig 7: Decision Tree representation

- **Random Forest Classifier:** This model creates set of small decision trees from a randomly selected subset of training data. Then, it aggregates them into "Forest of Trees". Each tree provides a weak predictor because the tree is handling the subset of data. Combining each weaker predictor will potentially generate a stronger predictor model. Here is diagram [8] illustrating Random Forest example, each Tree generated a predictor class (Class-A, Class-B,... Class-N) and combining all classes generates an aggregated Final predictor class.

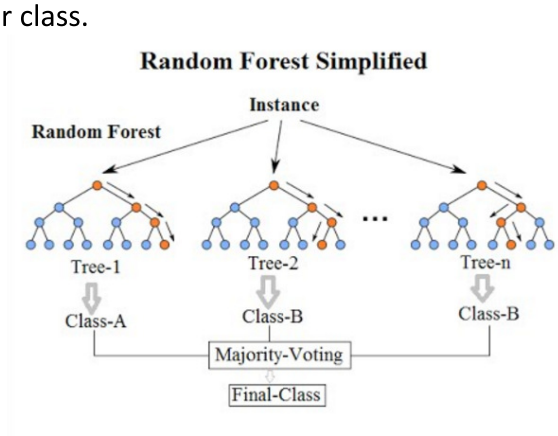


Fig 8: Random Forest Classifier Illustration

- **Ensemble – Voting Classifier:** Ensemble methods are techniques which create multiple classifiers/models and combine them to produce better results. The Voting classifier is one of the Ensemble methods widely used in the Supervised Learning. The major difference between Random Forest and Voting Classifier is, number of models utilized. In the Random Forest Classifier, we create multiple Decision Tree classifiers and generate a combined class. In the Voting Classifier, we create multiple supervised learning models and generate a combined class. The Ensemble technique would provide more accurate results compare to the individual model. Here is diagram [9] illustrating Voting Classifier Ensemble method.

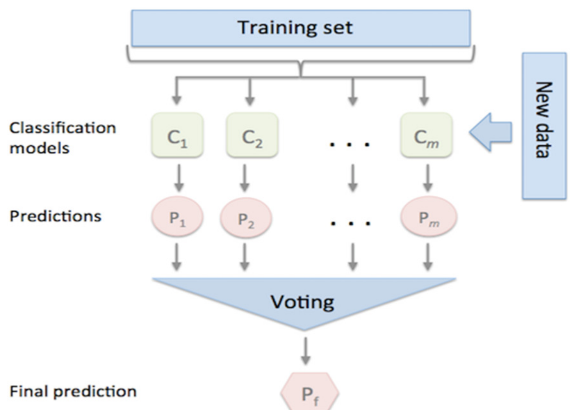


Fig 9: Ensemble Voting Classifier

Benchmark

I will implement there Supervised Learning model algorithms, Decision Tree, Random Forest and Ensemble Voting classifiers. I will first implement these three models with default parameters and slowly tune the hyper parameters to improve the score. The Decision Tree classifier default parameter metric is a benchmark for my project:

Model Name	**ROC- AOC**
Decision Tree	0.9206

Table 6: Decision Tree Classifier Benchmark

III. Methodology

Data Processing

I have executed various pre-processing steps to normalize data and to delete less-impacted columns. The Data preprocessing details are as follows:

- **Delete Three Types of Records**: The data analysis and bar chart graphs (Figure 2) clearly indicate that two types of data records (TRANSFER and CASH_OUT) contain Fraud transactions. The remaining three types (PAYMENT, CASH_IN and Debit) of records are

Genuine transactions. Therefore, we can safely delete the remaining three types of records from the dataset and keep the first two types in the dataset for data processing.

- **Convert String to Integer:** After deleting record from previous step, the second column, Type, contains two types of string values either TRANSFER and CASH_OUT. Therefore, we can convert column from string to integer, either 1 or 0. I have created a new column (c_type) to store the converted value. If the transaction type is "TRANSFER", then assign 1 to c_type. If the transaction type is "CASH_OUT", then assign 0 to c_type.
- **Delete Records with amount more than 10 Million:** The data analysis indicates that there are 5,650 records exist in the dataset with the amount more than 10 Million. None of these records are classified as Fraud. It means, Amount with more than 10 Million are Genuine transactions. Therefore, we can delete records with amount more than 10 Million from the dataset.
- **Logarithmic Normalization:** The amount and balance column values are distributed between zero and millions. These columns should be normalized for accurate and effective prediction. Therefore, I have applied Logarithmic technique to the amount and balance columns. The Log of zero is infinite. I have changed amount and balance values from zero to 0.01 before applying Logarithmic function.
- **Drop Columns:** The source account name and destination name columns do not add much value to the Model. Therefore, I have decided to drop Name columns along with the pre-normalized data columns.

Here are amount and balance histogram diagram after normalizing the data. The skewness after normalization is much lower compare to the post-normalization

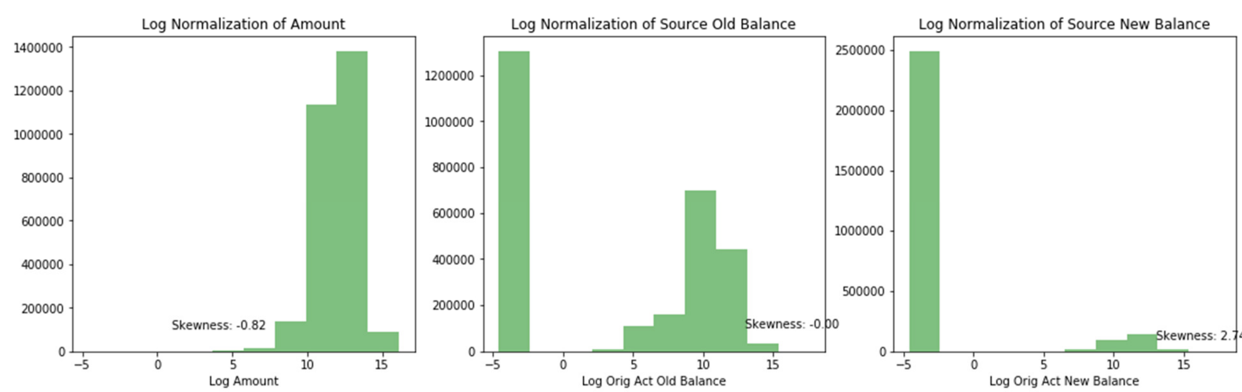


Fig 10: Amount, Original account old balance and new balance histograms

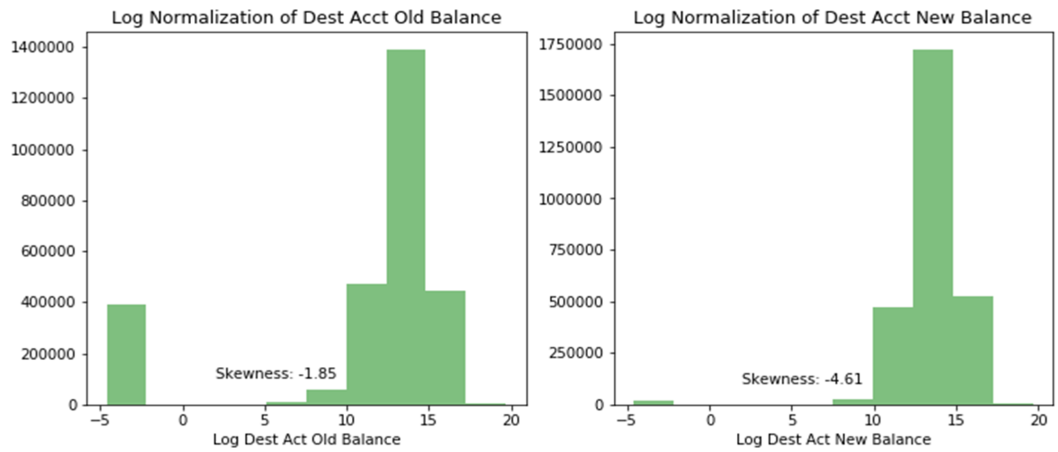


Fig 11: Dest account old balance and new balance histograms after normalization

Here are few sample records after feature scaling, feature selection and normalization:

step	c_type	amount_norm	oldbalanceOrg_norm	newbalanceOrig_norm	oldbalanceDest_norm	newbalanceDest_norm
1	1	5.198497	5.198497	-4.60517	-4.605170	-4.605170
1	0	5.198497	5.198497	-4.60517	9.960907	-4.605170
1	0	12.342062	9.637241	-4.60517	8.533657	10.849598
1	1	12.279836	6.558198	-4.60517	10.017932	-4.605170
1	1	12.649751	9.290537	-4.60517	8.743053	14.815838
1	0	11.611999	10.197850	-4.60517	12.573490	7.789521
1	0	10.949997	7.571484	-4.60517	11.159858	11.068296
1	0	8.584270	-4.605170	-4.60517	13.388776	15.680122
1	0	10.054546	9.923855	-4.60517	10.155879	-4.605170
1	1	11.044693	11.278645	9.71131	6.248043	9.033996

Table 7: Sample Records from dataset after normalization

Implementation

The next task is Model implementation. The features and target data have been split into training and testing datasets. I have allocated 80% of datasets randomly to the training data and remaining 20% of datasets. The training datasets are being utilized to train and tune the algorithm. The testing datasets are being utilized to evaluate the Model accuracy.

I have created a common function called "***Calculate_roc_aoc***" that executes a given Model and applies hyper (tuning) parameters. The function evaluates and prints the Metrics, Recall and ROC-AOC Score. I have implemented cross validation ShuffleSplit technique to set aside 20% of training datasets to the validation. All supervised learning classifiers have invoked this Common function. The function's input attributes are as follows:

- **Classifier**: the classifier model on which prediction is calculated
- **Parameters**: the hyper parameters applied to the Model
- **Score_type**: score that will be calculated, for example roc_auc
- **X_train**: Features training set
- **y_train**: Target data (fraud) training set

- **X_test**: Features testing set
- **y_test**: Target Data(fraud) testing set

I have executed two standard classifiers and one Ensemble Voting Classifier. The results are indicating that the Ensemble Voting classifier has returned slightly higher ROC-AOC score compare to the standard classifiers.

Refinement

I have applied various combinations of hyper parameters to each algorithm/Model to determine the parameters which returned the higher ROC- AUC. I have also implemented GridSearchCV logic to determine best tuning parameters from hyperparameters list. Here are few test scenarios:

Decision Tree Classifier

Hyper Parameter	Description	Values Tested
Max Depth	The maximum depth of the tree.	10,50, 75, 100, 1000, 10000
Min Samples Split	The minimum number of samples required to split an internal node:	100, 1000
Min Samples Leaf	The minimum number of samples required to be at a leaf node:	100, 1000
Criterion	The function to measure the quality of a split.	Gini, Entropy

Table 8: Decision Tree Classifier Hyperparameters

Random Forest Tree Classifier

Hyper Parameter	Description	Values Tested
Max Depth	The maximum depth of the tree.	10,50, 75, 100, 1000, 10000
Min Samples Split	The minimum number of samples required to split an internal node:	100, 1000
Min Samples Leaf	The minimum number of samples required to be at a leaf node:	100, 1000
Criterion	The function to measure the quality of a split.	Gini, Entropy

Table 9: Random Forest Tree Classifier Hyperparameters

Ensemble - Voting Classifier

Hyper Parameter	Description	Values Tested
Voting	If 'hard', uses predicted class labels for majority rule voting. Else if 'soft', predicts the class label based on the argmax of the sums of the predicted probabilities, which is recommended for an	Soft

	ensemble of well-calibrated classifiers.	
Weights	Weights assigned to predictor class	5,2,4 5,1,4 5,3,1

Table 10: Ensemble Voting Classifier Hyperparameters

The test simulation and results can be found at https://github.com/venkat998899/machine-learning/blob/master/projects/capstone/payments_fraud.ipynb

IV. Results

Model Evaluation and Validation

I have executed the Voting Classifier with different set of hyper parameters. The ROC-AOC from Voting Classifier with tuned hyper parameters is higher compare to the default values. Here are Voting Classifier test iteration results:

Class 1	Class 2	Class 3	voting	Weights	ROC SCore
RandomForest	Decision Tree	Logistic Regresion	Soft	5,4,2	0.9977
RandomForest	Decision Tree	Logistic Regresion	Soft	5,4,1	0.9977
RandomForest	Decision Tree	KNeighbors	Soft	5,4,2	0.9975
RandomForest	Decision Tree	KNeighbors	Soft	5,3,1	0.9975
RandomForest	Decision Tree	KNeighbors	Soft	5,4,1	0.9975

Table 11: Ensemble Voting Classifier Execution iterations

Stratified K-Fold

The K-Fold algorithm is standard technique to evaluate the Machine Learning Model performance and robustness. The dataset has been split into K-parts (Ex: 10). Each part of the dataset is called a “Fold”. The algorithm is Trained on k-1 Folds and a remaining Fold is designated as “Test” dataset. The same process is repeated K times to execute each Fold as a Test dataset. After executing algorithm K times, it provided K different scores (ROC-AOC numbers). Then, I have determined the mean and standard deviation of ROC-AOC scores. The dataset is a huge with millions of records. I have implemented Stratified K-Fold cross validation to ensure that the class distribution. Here are the results of Stratified K-Fold algorithm results:

Stratified K-Fold Iteration#	ROC AOC Score
1	0.98
2	0.95
3	0.98
4	0.84
5	0.95
6	0.99
7	1.00
8	1.00
9	0.99
10	0.99

Table 12: Stratified K-Fold Iteration Results

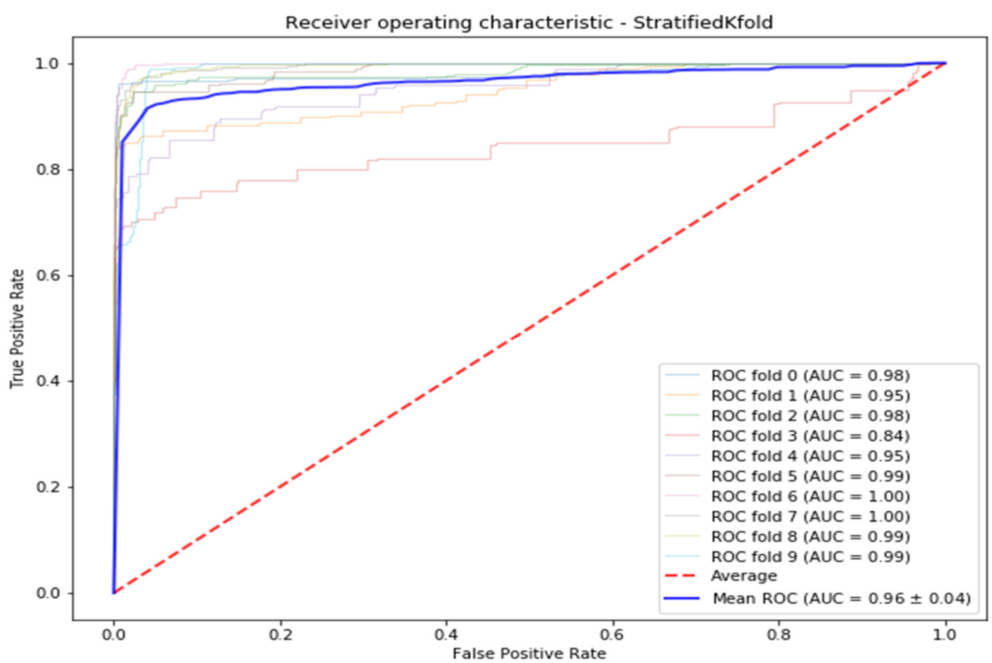


Fig 13: Stratified K-Fold iteration results.

Above table 12 and graph 13 indicate that the Mean AOC score from Stratified K-fold score is 0.96 with a small standard deviation. The K-Fold evaluation ROC-AOC is score is very close to the Voting Classifier ROC-AOC score.

Justification

The following table (#11) provides summary of default and post tuning ROC-AOC values for each Algorithm. ROC-AOC has improved from Decision Tree classifier to the Ensemble Voting Classifier. There is an improvement from Benchmark metrics (Decision Tree) to the Voting Classifier. I have taken three models that have slightly higher ROC-AOC and utilized in the Voting Classifier. Basically, Voting Classifier was built upon three other Models (Random Forest, Decision Tree and Logistic Regression).

Model Name	Best Hyper Parameters	Default ROC-AOC	Tuned ROC-AOC
Decision Tree	MaxDepth=10 Min Samples Split=100 Min Samples Leaf=100 Criterion = Gini	0.9206	0.9955
Random Forest	MaxDepth=10 Min Samples Split=100 Min Samples Leaf=100 Criterion = Gini n_iterations =10	0.9691	0.9973
Ensemble Voting Classifier	Voting: soft Weights: 5,4,1	0.9955	0.9977
Ensemble Voting Classifier – Stratified K-Fold Evaluation	Mean ROC-AOC	-	0.96

Table 13: Metrics comparison for all classifiers

The above table indicates that Ensemble Voting Classifier results are slightly better than the remaining two Models. Also, the Ensemble Voting classifier ROC-AOC score is closely matching with the Stratified K-Fold Evaluation.

V. Conclusion

Free-Form Visualization

The amount values are not evenly distributed. Fig 12 shows that most of the records have amount less value. Fig 13 shows the Source account (Origin) old and new balance distribution. Fig 14 shows the Destination account old and new balance distribution. There are very few records that have more amount more than 1,000,000. The Amount and Balance values are skewed towards to the left. The skewness factor is high for features before normalization.

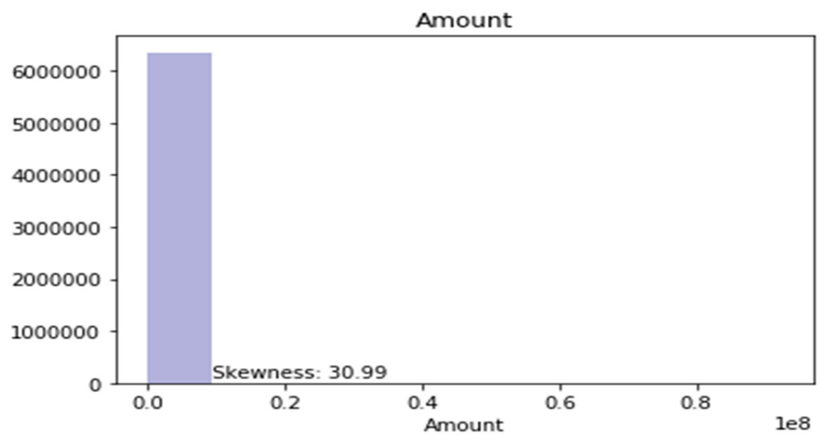


Fig 12: Amount distribution

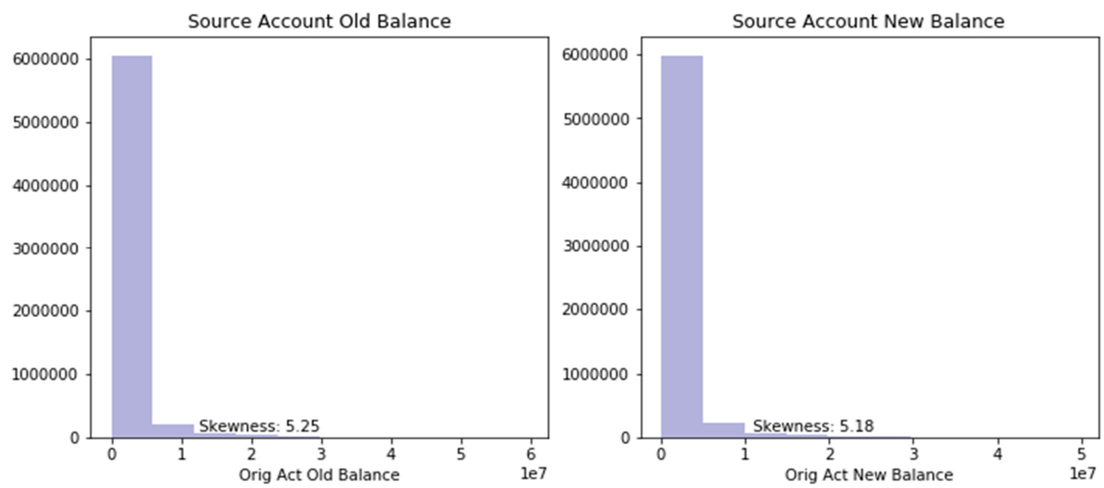


Fig 13: Original account old and new balance distribution.

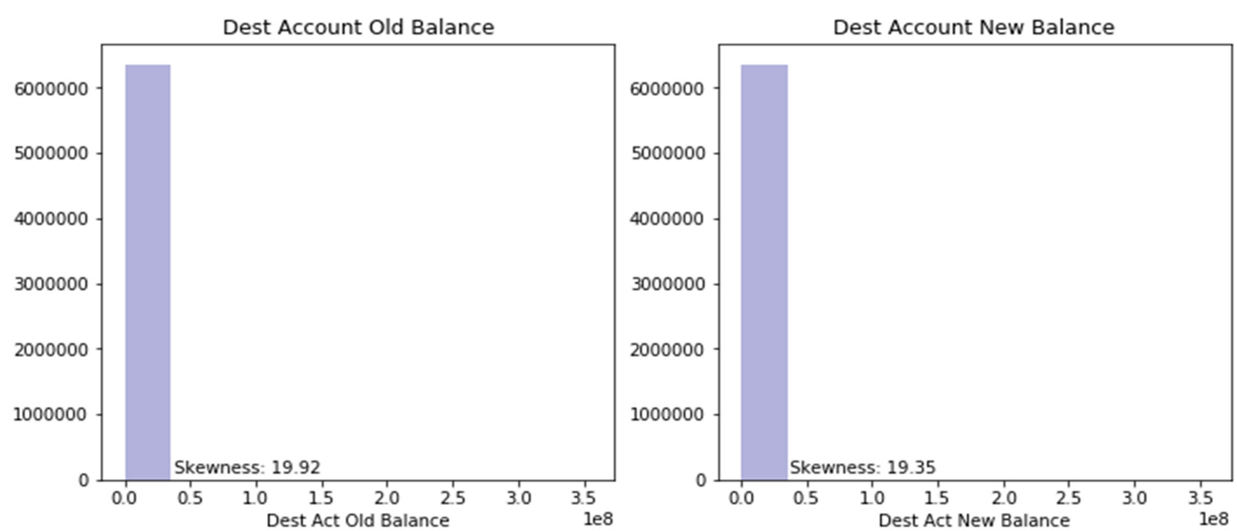


Fig 14: Destination account old and new balance distribution.

Reflection

The capstone project provided opportunity to demonstrate the skills and knowledge achieved from Machine Learning Engineering nanodegree. The project tasks and activities can be summarized as given below:

1. Identified project objective, scope, and public dataset.
2. Downloaded dataset from Kaggle website
3. Decided to implement Supervised Learning Classifier algorithms
4. A benchmark was created for the classifier
5. Analyzed the Dataset which contains more than 6 million records.
6. Created visual diagrams like Histograms and bar charts.
7. Selected the features which have high impact
8. Converted string to integrate and Normalized the data
9. Executed Five standalone algorithms and one ensemble algorithm
10. Tuned the algorithms by modifying hyperparameters.
11. Identified the best suitable algorithm after comparing the Accuracy and F-score.

I found step 5 and 7 are most tedious and time-consuming activities. I had to analyze the fraud activity records and identified the fraud pattern. In my analysis, Fraudulent activity was executed with two types of transactions, PAYMENT and CASH_IN.

The interesting aspects of project is, Ensemble Voting Classifier which takes multiple other classifiers and combines into a hybrid classifier. The Voting Classifier ROC-AOC is slightly higher compere to other classifiers. Also, the Voting Classifier ROC-AOC is closely matching with Stratified K-Fold ROC-AOC score.

The common function from the program can be reused for other projects easily. The common function was written as independent function can be utilized for other projects.

Improvements

There is a potential possibility of improving ROC-AOC by implementing other Ensemble method like "Stacking Classifier". The ROC-AOC and Recall can be improved with other Ensemble methods. Another improvement could be to normalize amount and balance columns with Box-Cox technique.

References

[1] Mobile Payment Trends in 2018: https://www.paymentvision.com/blog/2017/12/26/7-trends-that-prove-mobile-payments-are-here-to-stay-in-2018

[2] Mobile Payments Safer and Faster: https://www.mobilepaymentstoday.com/blogs/3-trends-for-2018-safer-data-faster-payments-better-experiences/

[3] Banking Future Payments- Accenture Consulting Study : https://www.accenture.com/us-en/insight-banking-future-payments-ten-trends

[4] Kaggle Financial Fraud Detection dataset: https://www.kaggle.com/ntnu-testimon/paysim1

[5] PaySim Simulator: E. A. Lopez-Rojas , A. Elmir, and S. Axelsson. "PaySim: A financial mobile money simulator for fraud detection". In: The 28th European Modeling and Simulation Symposium-EMSS, Larnaca, Cyprus. 2016

[6] How to fix Skewed dataset in Machine Learning: https://becominghuman.ai/how-to-deal-with-skewed-dataset-in-machine-learning-afd2928011cc

[7] Naïve Bayes theorem : https://www.python-course.eu/naive_bayes_classifier_introduction.php

[8] Random Forest Diagram: https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d

[9] Voting Classifier sample diagram: https://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier/

[10] Imbalance Data Metrics: https://towardsdatascience.com/what-metrics-should-we-use-on-imbalanced-data-set-precision-recall-roc-e2e79252aeba