Fraud Detection

On Credit Card Data

Yash Kumar
Masters' in Data Science
Indiana University, Bloomington

Contents

Overview

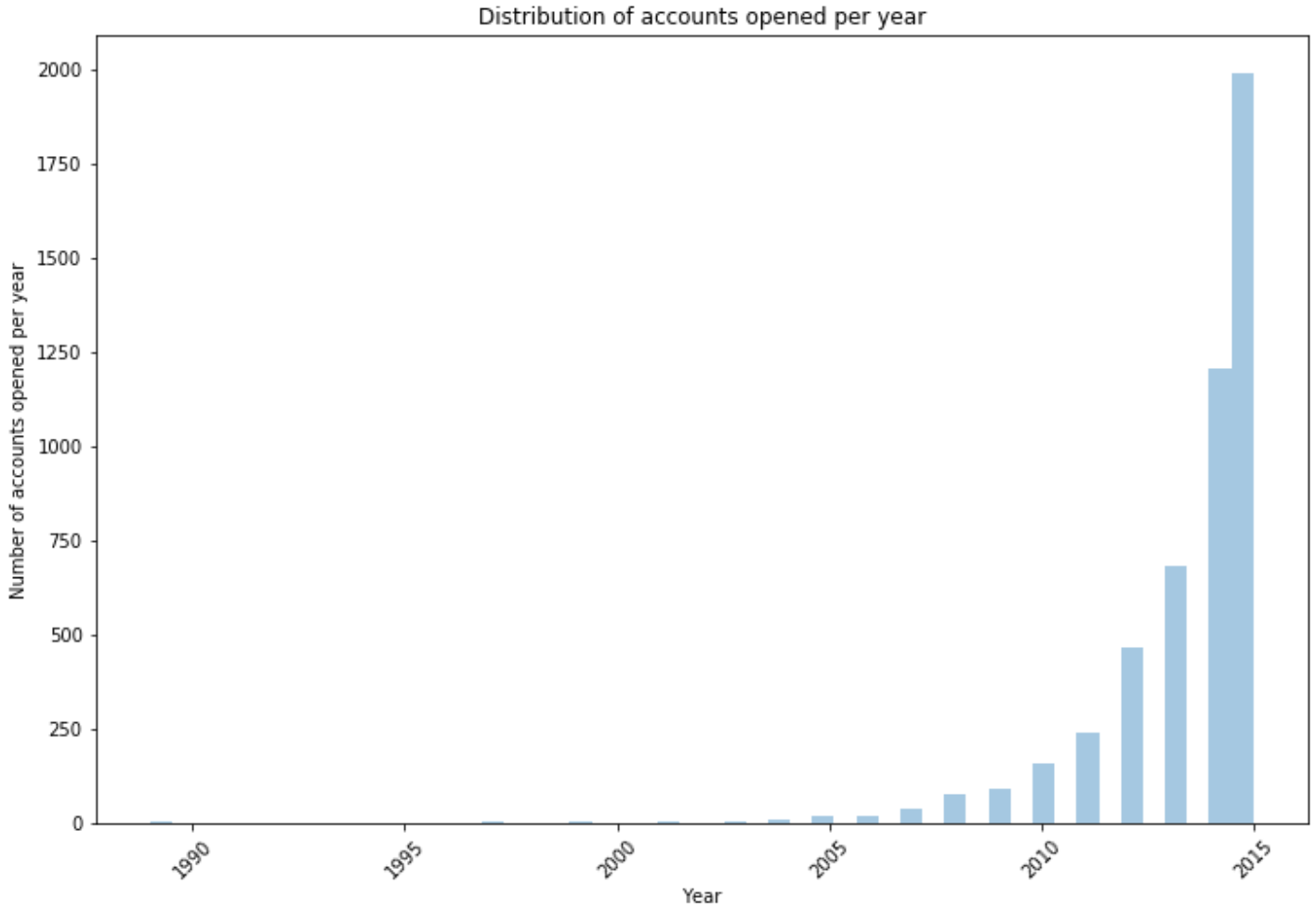
- According to Barclays, 47% of global credit card frauds occur here in the United States
- In 2014, the average loss in credit card fraudulent transaction was $7,761
- The client wanted to flag all fraudulent transactions of credit card customers using predictive modelling
- The rule-based legacy system was not effective

Data Information

- Contains transaction level data of credit card customers
- Dataset contains 786,563 rows with 29 columns
- 20% of the columns contain all null values which have been dropped for analysis
- 5,000 unique customers with 5,245 unique credit cards
- Highly skewed data where only 1.5% of all transactions are fraud

Data Distribution

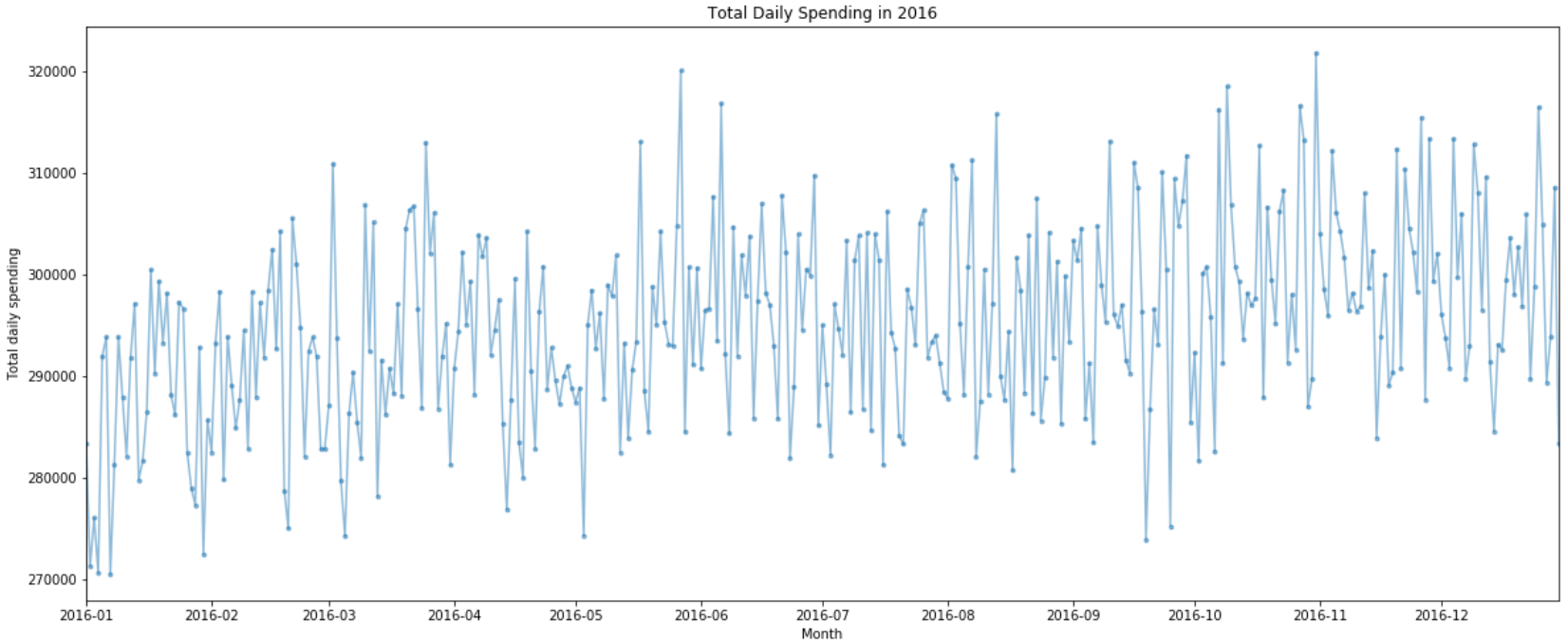
- Number of accounts opened per year



- The number of accounts opened per years has clearly increased exponentially post 2010
- Increase in credit card users in the 20th century

Data Distribution

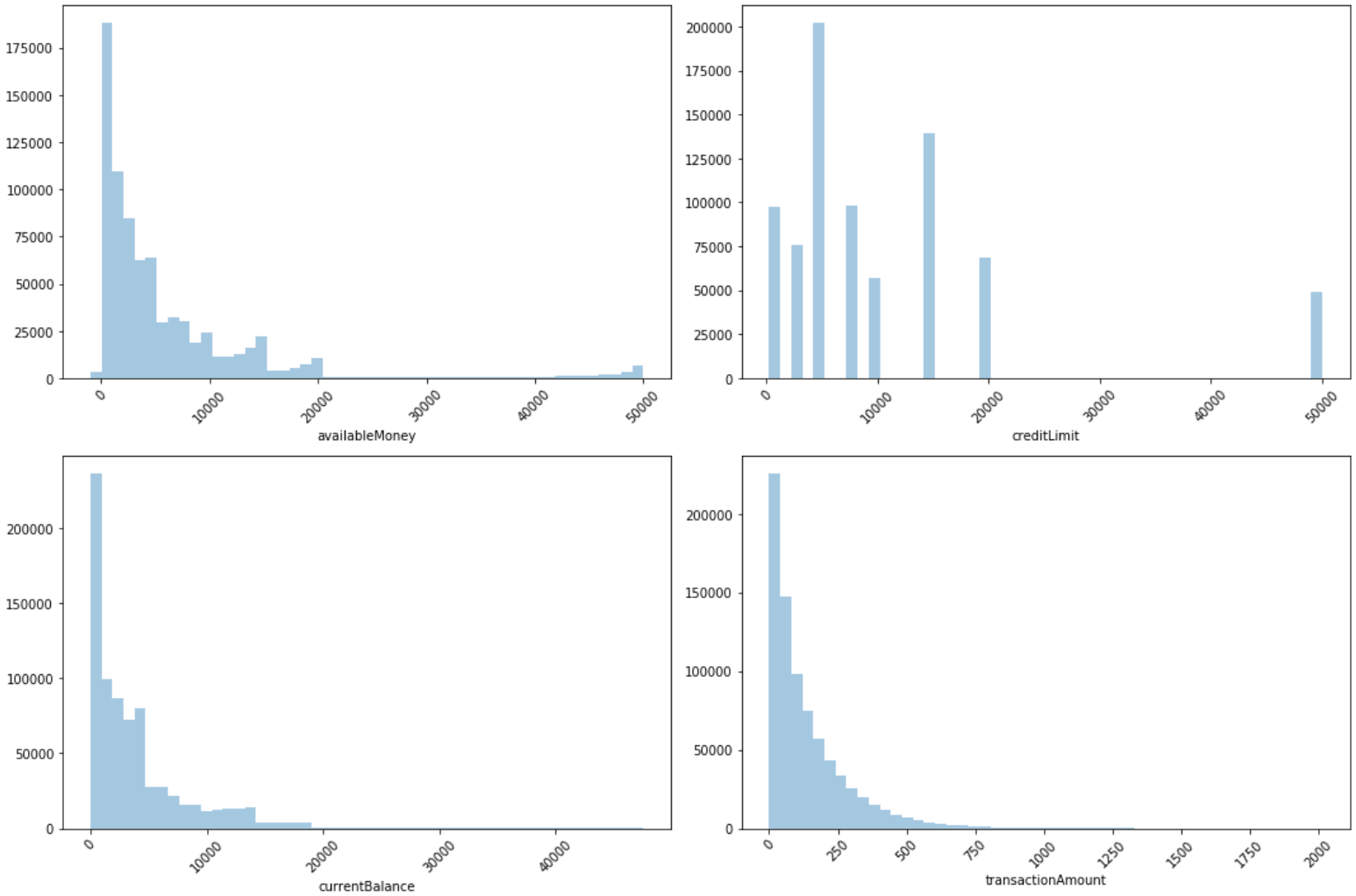
- Variation of transaction amount with month



- Increasing trend from January 2016 to December 2016
- Spikes in transaction amount during Valentine's day (mid-February), Easter (March end), Mother's day (mid-May), Father's day (early June), Halloween (October end) and Thanksgiving (November end)

Data Distribution

- Analysis of continuous variables

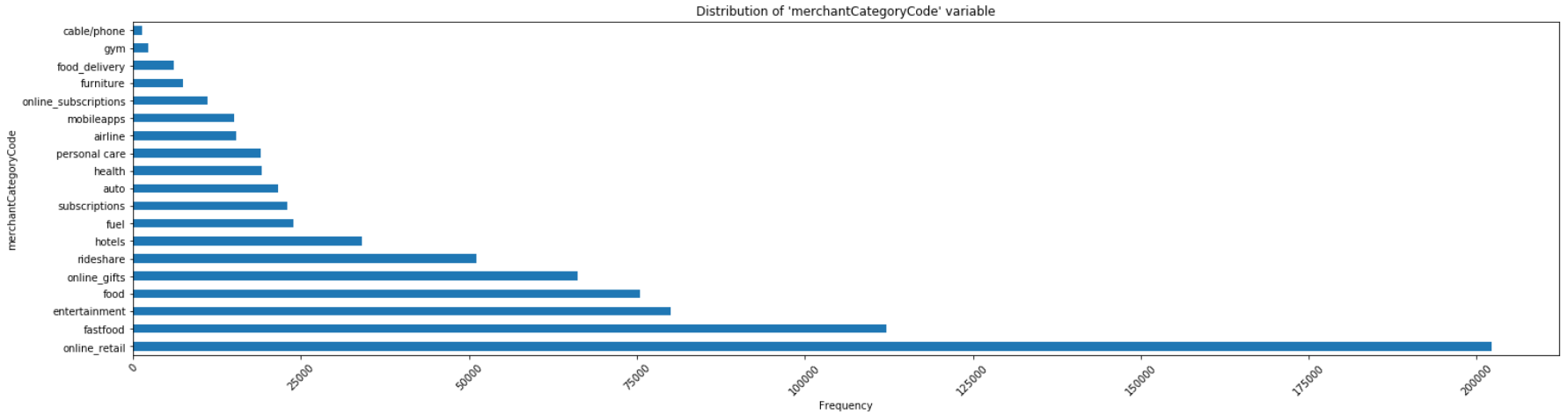


There are 4 continuous variables which are highly skewed to the right :

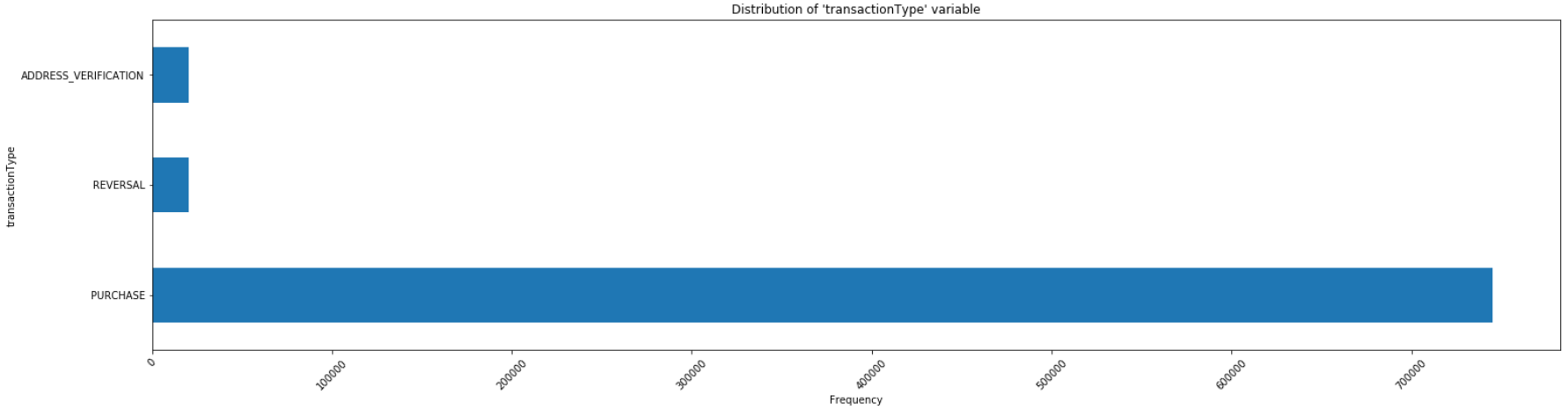
- Variables ‘availablemoney’, ‘currentBalance’ and ‘transactionAmount’ seem to follow exponential distribution
- Variable ‘creditLimit’ is multimodal with distinct peaks

Data Distribution

- Analysis of categorical variables



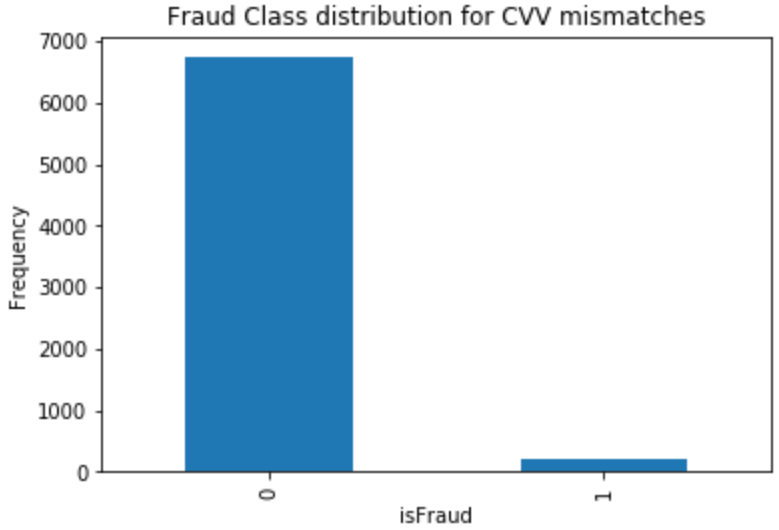
- Majority of the transactions are from online retail stores, fast foods chains and entertainment



- Majority of the transactions have the type ‘purchase’

Feature Engineering
- Card mismatch

- There are two CVV variables ‘cardCVV’ and ‘enteredCVV’ which on their own do not provide any interesting information
- We can check cases where there is a mismatch between CVV fields
- 7,015 out of total 786,563 transactions have card mismatch
- However, there are merchants who process transactions even when there is a card mismatch (using information such as billing address to authorize the transaction)
- Number of fraud cases of transactions which has card mismatch is shown below :

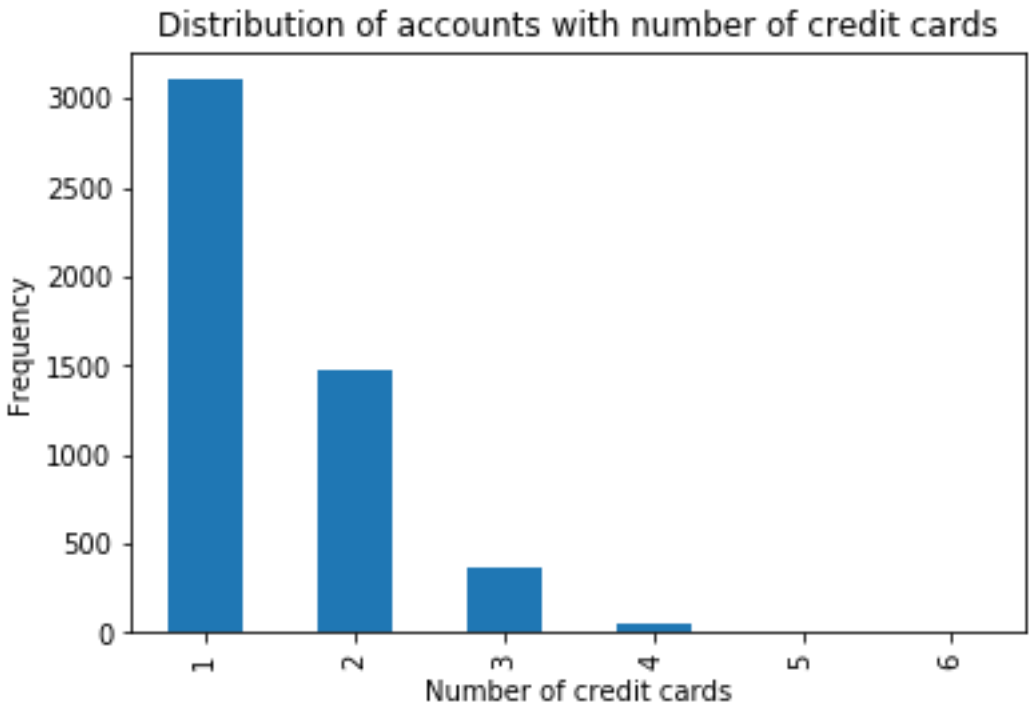


- Majority of the transactions are legitimate
- However, the event rate is 0.5% higher for transactions where the card is mis-matched

Feature Engineering

- Multiple card users

- Aggregated the data on account level to get the counts of number of cards a customer has
- Flags has been given to accounts which has more than one credit card
- Distribution of accounts with number of credit cards is shown below :



- About 40% of customers own more than one credit card

Feature Engineering

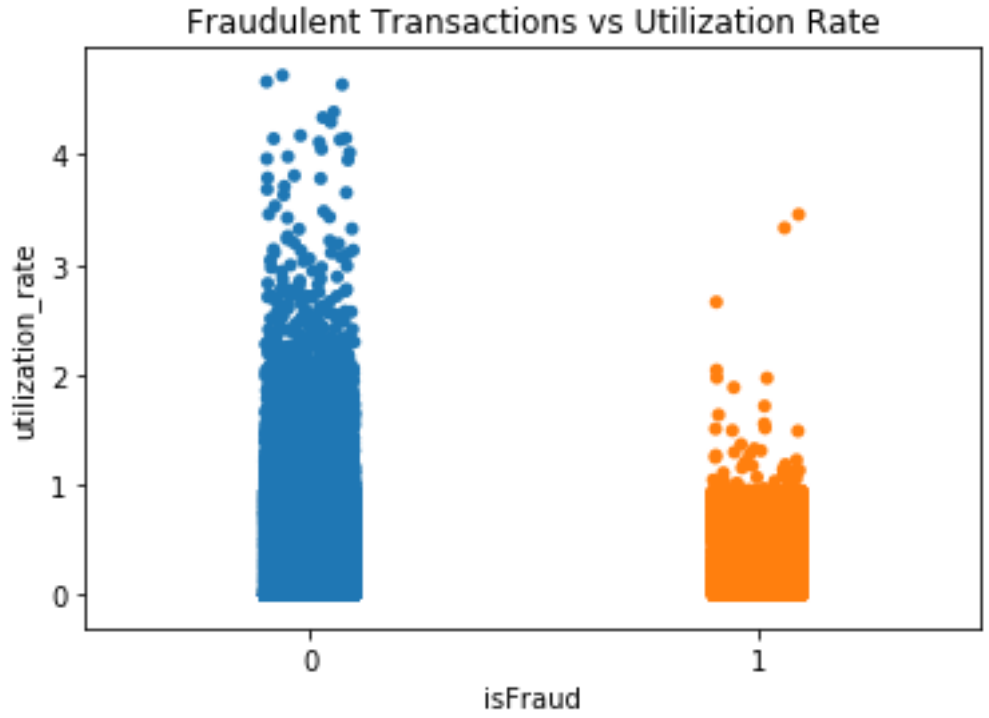
- Multi swipe transactions

- Multi swipe transactions happen in the case of Purchase ‘transactiontype’
- To identify duplicate transactions, subset of the data is created with ‘accountNumber’, ‘merchantname’, ‘merchantCountryCode’ and ‘transactionAmount’ to check repeated transactions for an account number which has the same transaction amount for the same merchant in the same country.
- There are 7,249 transactions corresponding to multi-swipe transactions with amount worth $ 1,075,506
- Flag 1 has been given to transactions which are multi-swiped and rest 0
- The highest transaction value for multi-swipe transactions is contributed by rideshare services like Uber and Lyft (arising from booking multiple rides) and e-commerce websites like gap.com, Alibaba.com, etc. (arising usually from credit card preauthorization holds). The amount is highest for these merchants due to sheer volume of people utilizing their services.

Feature Engineering

- Utilization rate

- Utilization rate is calculated using the ratio of current balance with credit limit
- Distribution of utilization rate with fraudulent transactions has been given below:



- Majority of the transactions have utilization rate in between 0-2
- The trend is similar for both fraudulent and legitimate transactions

Model Building
- Data Preparation

Preprocessing

Data preprocessing is done to transform raw features to representations that are better suited to work with machine learning algorithms. Following techniques are being used :

- Drop features which has all null values
- Standardize continuous variables by removing their means and scaling by dividing with the standard deviation. This process is called Z-score normalization so that the variable has the properties of standard normal distribution with 0 mean and unit standard deviation
- One-hot encoding of categorical variables

Train-Test split

Train using 75% of the dataset and test the model using the rest 25% of the dataset

Oversampling

Up sample the train samples to ensure 50-50 ratio of target variable class as the event rate is very low (1.5%)

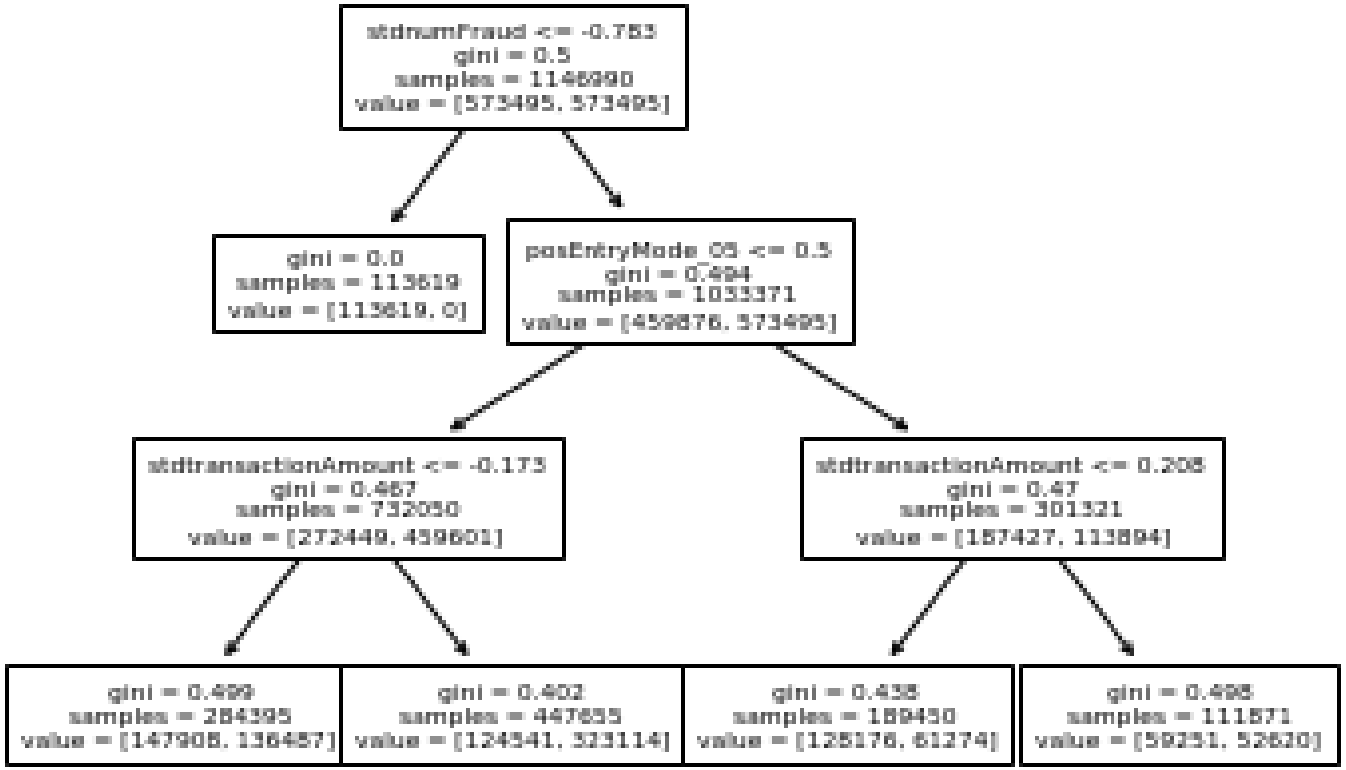
Model Building
- Random Forest Classifier

- Random Forest classifier is an ensemble model which grows multiple decision tree classifiers and chooses the mode decisions of all trees in the forest

- It uses bagging approach which takes subset of data with replacement and subset of features for growing individual trees

- Avoids overfitting since it grows multiple decision trees with bagging approach and takes mode decision of all trees

Model Building
- Decision Tree Result



- The sample decision tree model is plotted for depth 3
- Tree split is done using gini index which is calculated as :

Average Gini = L x (1 - $\sum p^2$) + R x (1 - $\sum p^2$)

where L and R are left and right proportions of samples
$\sum p^2$ is sum of square probability of classes

Model Building

- Logistic Regression

Logistic Regression

- It is the most common supervised learning algorithm used in classification problems. Logistic function, also called sigmoid function which has a range from 0 to 1. The equation of logistic regression is given below :

$$\mathrm{p(c|}\varkappa) = \sigma(\omega.\varkappa) = \frac{1}{1+e^{-\omega.\varkappa}}$$

where ω *is weights and* $\varkappa$ *is predictor variable and* $p(c|\varkappa)$ *is the probability of class* given predictor variable

- Cost function is evaluated using the equation given below :

$$J(\omega) = -\sum_{1}^{n} ylog(\sigma(\omega.\varkappa)) + (1-\mathrm{y})\log(1-\sigma(\omega.\varkappa))$$

where $J(\omega)$ *is cost function and* y *is response variable*

- Gradient descent optimization is used to update weights using the equation given below :

$$\omega \coloneqq \omega - \alpha \frac{dJ(\omega)}{d\omega}$$

where α *is the learning rate and* $\frac{dJ(\omega)}{d\omega}$ is partial differential of $\mathrm{J}(\omega)$ *with respect to* ω

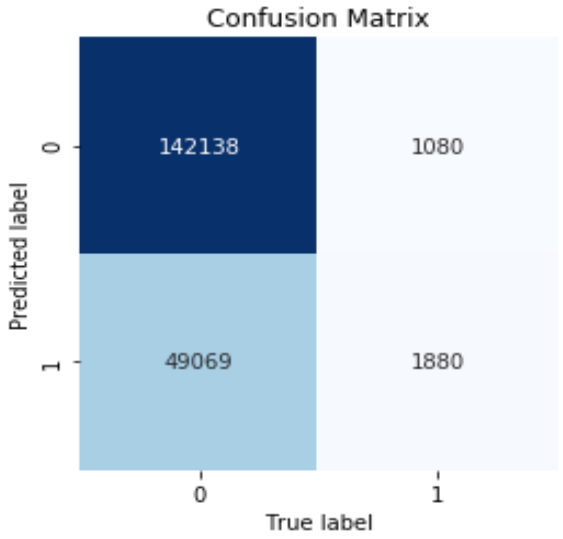
Model Comparison

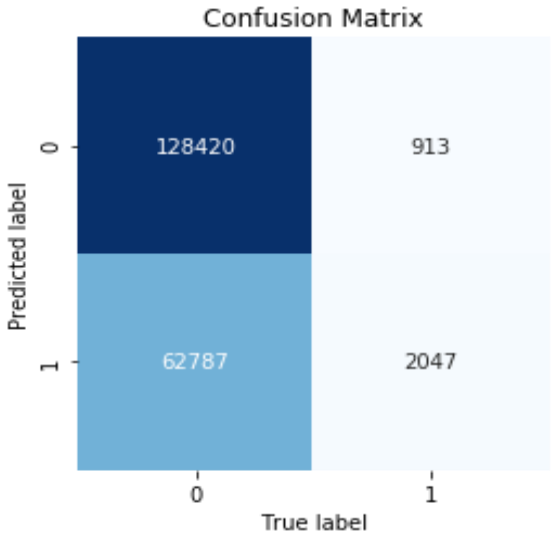
- Evaluation metrics

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

$$Recall = \frac{TP}{TP + FN}$$

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



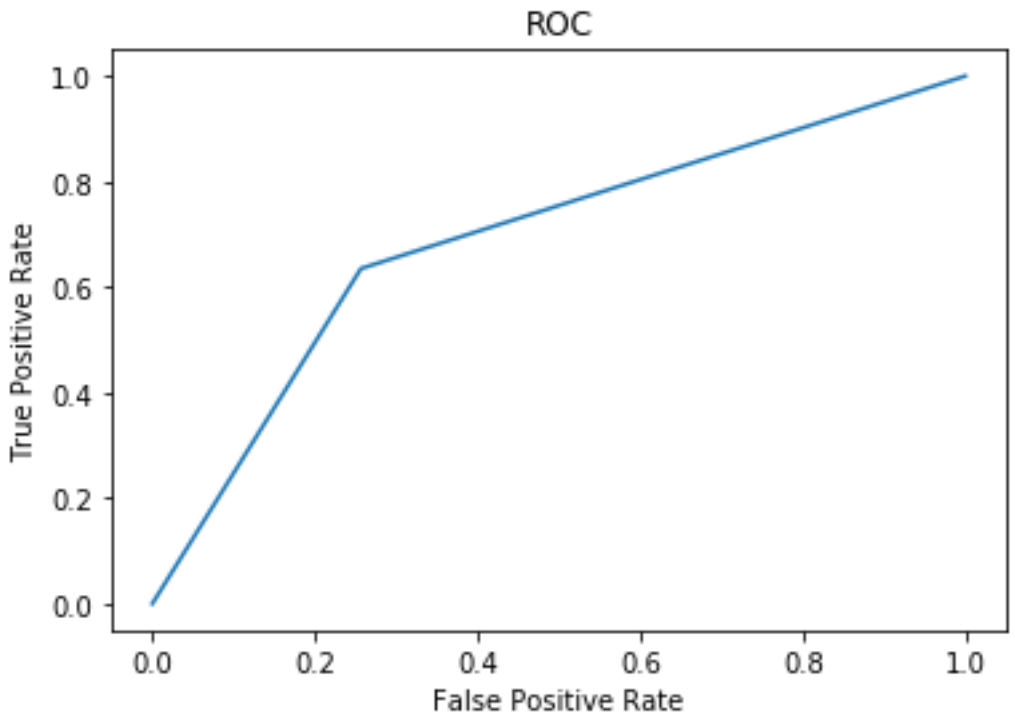


Evaluation Metrics	Random Forest	Logistic Regression
Accuracy	74%	68%
Recall	64%	70%
Precision	4%	3%

Model Evaluation

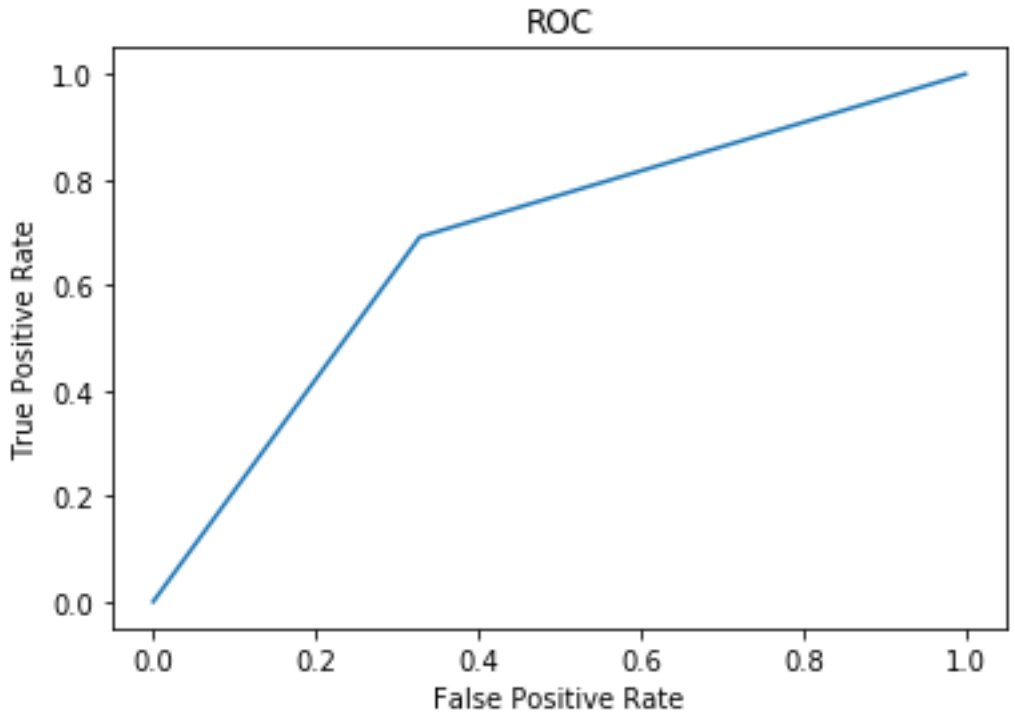
- Area under curve (AUC)

Random Forest



Area under curve : 70%

Logistic Regression



Area under curve : 70%

References

1. Pandas Official Documentation: https://pandas.pydata.org/docs/

2. Scikit Learn User Guide: https://scikit-learn.org/stable/user_guide.html

3. Seaborn Official tutorial: https://seaborn.pydata.org/tutorial.html

4. Dr. Sebastian Raschka's blog: https://sebastianraschka.com/blog/index.html

5. Dr. Jason Brownlee's blog: https://machinelearningmastery.com/blog/

6. Dr. Leo Breiman's notes on Random Forests: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm