Credit Card Transaction

-Supervised Fraud Detection

Content

Executive Summary

This report examines the Card Transaction dataset to detect the fraud activity using supervised machine learning methods. Out team used R as the major tool, "Kolmogorov-Smirnov" as the primary feature selection method, and Logistic Regression, Neural Network, Extreme Gradient Boosting (XG-Boost), Support Vector Machine and Random Forest as featured algorithms to build the predictive model.

The steps we took were:

1. Variable construction and data cleaning
2. Feature selection using "Kolmogorov-Smirnov."
3. Fraud detection model building using Logistic Regression, Neural Network, Extreme Gradient Boosting (XG-Boost), Support Vector Machine and Random Forest

The original dataset contains 96,708 records from 1/1/2010 to 12/31/2010 with details about the transaction information including card number, date of the transaction, merchant number, merchant description, merchant state and zip code, transaction type and transaction amount. First, we built 102 expert variables by looking at similar characteristics or/and time windows. Then, we separated the dataset into training, testing and out-of-time dataset, and performed feature selection on the training set. We ranked all variables based on the "Kolmogorov-Smirnov" score and selected 63 most important variables. We used the training dataset to train our model first. Then, we tested the model on both training dataset and testing dataset. After that, we built the final model using both training and testing dataset and tested the model on the out-of-time dataset to calculate and compare the final fraud detection rate (FDR).

Using the 63 variables and supervised machine learning algorithms, we found that XG-Boost algorithm has the best performance, with FDR of 73.7% at 2% of the population. More detailed model performance information is demonstrated in the table below.

	FDR @ 2%		
Model	**Training**	**Testing**	**OOT**
Logistic Regression	50.6%	54.8%	53.6%
Neural Network	87.6%	83.0%	66.6%

XGBoost	100.0%	96.3%	73.7%
Support Vector Machine	89.6%	86.7%	65.4%
Random Forests	91.0%	93.3%	71.9%

Part I. Description of Data

Here we provide an overall description of the dataset we analyzed in this project.

Dataset name: Card Transactions Data

File Name: card transactions.xlsx

Dataset Overview:

- Category: Identify Fraud
- Dimensions: 96,708 records and 10 variables
    - Categorical Variables: 7
    - Numeric Variable: 1
    - Date Variable: 1
    - Text Variable: 1
- Features Names: Recordnum, Cardnum, Date, Merchantnum, Merch Description, Merchant State, Merchant Zip, Transtype, Amount
- Response: Fraud

Most important variables in consideration:

Variables	Data Type	Description

Recordnum	Categoric al	Unique identifier of each record
Cardnum	Categoric al	Unique identifier of each credit card
Date	Date	Date of the transaction
Merchantnum	Categoric al	Unique identifier of the merchant
Merch Description	Text	Description of the merchant
Merchant State	Categoric al	The state of the merchant
Merchant Zip	Categoric al	Zip code of the merchant
Transtype	Categoric al	Type of the transaction
Amount	Numeric al	The transaction amount
Fraud	Categoric al	Whether the application is fraud of not

Part II. Data Cleaning

We performed three steps of data cleaning on this dataset altogether:

1. Filtering out irrelevant data

According to the conversation with credit card fraud experts, we concluded that only transaction type P is relevant to our model prediction. Therefore, here we excluded all the records with transaction type A, D and Y, leaving 95353 records of transaction type P.

2. Removing the outlier

When performing exploratory analysis, we found a substantial outlier in the Amount variable, with a value higher than 3,000,000 dollars (details are shown in the data quality report appended to this report). After an investigation, we found that this outlier was due to a currency type error. The amount of this transaction was recorded in the unit of Mexican

Peso, while all the other amounts in this dataset were recorded in the unit of United States Dollar. Since this record is also not labeled as fraud, we choose to delete this outlier directly to avoid skewing the prediction results.

3. Dealing with missing data

When performing exploratory analysis, we found missing values in variables MerchantState, MerchantZip and Merchantnum. We come up with the following strategy to filling the missing values:

a. Filling "blank" into MerchantState and MerchantZip
b. Filling serial number from 1 to 3198 into the 3198 missing values in Merchantnum

Part III. Variable Creation

We created 102 variables in this project altogether. We considered what abnormal features a fraudulent transaction would have and derived expert variables using the following 5 methods:

1. Measure transaction frequency during a set period of time, both at the card level and the merchant level.

To address this aspect, we look back a certain period of time and count the number of previous records that have the same card/merchant number as the current record. For example, variable *"same_card _3"* means to check the transaction frequency of a specific card number within the last 3 days.

2.Capture the purchasing pattern during a set period of time, both at the card level and merchant level.

To address this aspect, we look back a certain period of time and count the number of unique attributes for transactions of the same card/merchant as the current record. For example, variable *"same_card_diff_merch_3"* means to check the number of different merchants associated with a card within the last 3 days.

3. Compare the current transaction amount to the historical amounts during a set period of time, both at the card level and merchant level.

To address this aspect, we look back a certain period of time and derive the ratio of the current transaction amount to some summary statistics of the historical amounts for the same card/merchant as the current record. For example, variable *"amount_avg_3"* is the ratio of the current amount of a specific card to its past 3-day average amount, excluding the past fraudulent transactions.

4. Measure the distance between merchants purchased by the same card.

To address this aspect, we calculate the difference between zip code of the current merchant and zip code of the last merchant purchased by the same card.

5. Capture the seasonality of the purchasing frequency.

To address this aspect, we look back on time and calculate the average number of transactions made for each day of the week. We then count the number of transactions within the day of the current record and divide it by the average number of transactions for this specific day.

The following chart provides a detailed description of all the 102 variables we created in this project:

No.	Variable Name	Variable Description
1	same_card_1	The transaction frequency for this specific card number within the same day of the current record. It only counts the records that appear before the current record.
2	same_card_3	The transaction frequency for this specific card number within the last 3 days. It only counts the records that appear before the current record.
3	same_card_7	The transaction frequency for this specific card number within the last 7 days. It only counts the records that appear before the current record.
4	same_card_14	The transaction frequency for this specific card number within the last 14 days. It only counts the records that appear before the current record.
5	same_card_28	The transaction frequency for this specific card number within the last 28 days. It only counts the records that appear before the current record.
6	same_card_diff_state_1	The number of unique merchant's states for transactions made by this specific card number within the same day of the current record. It only counts the records that appear before the current record.

7	same_card_diff_state_3	The number of unique merchant's states for transactions made by this specific card number within the last 3 days. It only counts the records that appear before the current record.
8	same_card_diff_state_7	The number of unique merchant's states for transactions made by this specific card number within the last 7 days. It only counts the records that appear before the current record.
9	same_card_diff_state_14	The number of unique merchant's states for transactions made by this specific card number within the last 14 days. It only counts the records that appear before the current record.
10	same_card_diff_state_28	The number of unique merchant's states for transactions made by this specific card number within the last 28 days. It only counts the records that appear before the current record.
11	same_card_diff_zip_1	The number of unique merchant's zips for transactions made by this specific card number within the same day of the current record. It only counts the records that appear before the current record.
12	same_card_diff_zip_3	The number of unique merchant's zips for transactions made by this specific card number within the last 3 days. It only counts the records that appear before the current record.
13	same_card_diff_zip_7	The number of unique merchant's zips for transactions made by this specific card number within the last 7 days. It only counts the records that appear before the current record.
14	same_card_diff_zip_14	The number of unique merchant's zips for transactions made by this specific card number within the last 14 days. It only counts the records that appear before the current record.

15	same_card_diff_zip_28	The number of unique merchant's zips for transactions made by this specific card number within the last 28 days. It only counts the records that appear before the current record.
16	same_card_diff_amount_1	The number of unique amounts for transactions made by this specific card number within the same day of the current record. It only counts the records that appear before the current record.
17	same_card_diff_amount_3	The number of unique amounts for transactions made by this specific card number within the last 3 days. It only counts the records that appear before the current record.
18	same_card_diff_amount_7	The number of unique amounts for transactions made by this specific card number within the last 7 days. It only counts the records that appear before the current record.
19	same_card_diff_amount_14	The number of unique amounts for transactions made by this specific card number within the last 14 days. It only counts the records that appear before the current record.
20	same_card_diff_amount_28	The number of unique amounts for transactions made by this specific card number within the last 28 days. It only counts the records that appear before the current record.
21	same_card_diff_merch_1	The number of unique merchants for transactions made by this specific card number within the same day of the current record. It only counts the records that appear before the current record.
22	same_card_diff_merch_3	The number of unique merchants for transactions made by this specific card number within the last 3 days. It only counts the records that appear before the current record.

23	same_card_diff_ merch_7	The number of unique merchants for transactions made by this specific card number within the last 7 days. It only counts the records that appear before the current record.
24	same_card_diff_ merch_14	The number of unique merchants for transactions made by this specific card number within the last 14 days. It only counts the records that appear before the current record.
25	same_card_diff_ merch_28	The number of unique merchants for transactions made by this specific card number within the last 28 days. It only counts the records that appear before the current record.
26	same_merch_1	The transaction frequency for this specific merchant number within the same day of the current record. It only counts the records that appear before the current record.
27	same_merch_3	The transaction frequency for this specific merchant number within the last 3 days. It only counts the records that appear before the current record.
28	same_merch_7	The transaction frequency for this specific merchant number within the last 7 days. It only counts the records that appear before the current record.
29	same_merch_14	The transaction frequency for this specific merchant number within the last 14 days. It only counts the records that appear before the current record.
30	same_merch_28	The transaction frequency for this specific merchant number within the last 28 days. It only counts the records that appear before the current record.
31	same_merch_diff _state_1	The number of unique merchant's states for transactions of this specific merchant number within the same day of the current record. It only counts the records that appear before the current record.

32	same_merch_diff _state_3	The number of unique merchant's states for transactions of this specific merchant number within the last 3 days. It only counts the records that appear before the current record.
33	same_merch_diff _state_7	The number of unique merchant's states for transactions of this specific merchant number within the last 7 days. It only counts the records that appear before the current record.
34	same_merch_diff _state_14	The number of unique merchant's states for transactions of this specific merchant number within the last 14 days. It only counts the records that appear before the current record.
35	same_merch_diff _state_28	The number of unique merchant's states for transactions of this specific merchant number within the last 28 days. It only counts the records that appear before the current record.
36	same_merch_diff _zip_1	The number of unique merchant's zips for transactions of this specific merchant number within the same day of the current record. It only counts the records that appear before the current record.
37	same_merch_diff _zip_3	The number of unique merchant's zips for transactions of this specific merchant number within the last 3 days. It only counts the records that appear before the current record.
38	same_merch_diff _zip_7	The number of unique merchant's zips for transactions of this specific merchant number within the last 7 days. It only counts the records that appear before the current record.
39	same_merch_diff _zip_14	The number of unique merchant's zips for transactions of this specific merchant number within the last 14 days. It only counts the records that appear before the current record.

40	same_merch_diff _zip_28	The number of unique merchant's zips for transactions of this specific merchant number within the last 28 days. It only counts the records that appear before the current record.
41	same_merch_diff _amount_1	The number of unique amounts for transactions of this specific merchant number within the same day of the current record. It only counts the records that appear before the current record.
42	same_merch_diff _amount_3	The number of unique amounts for transactions of this specific merchant number within the last 3 days. It only counts the records that appear before the current record.
43	same_merch_diff _amount_7	The number of unique amounts for transactions of this specific merchant number within the last 7 days. It only counts the records that appear before the current record.
44	same_merch_diff _amount_14	The number of unique amounts for transactions of this specific merchant number within the last 14 days. It only counts the records that appear before the current record.
45	same_merch_diff _amount_28	The number of unique amounts for transactions of this specific merchant number within the last 28 days. It only counts the records that appear before the current record.
46	same_merch_diff _card_1	The number of unique cards for transactions of this specific merchant number within the same day of the current record. It only counts the records that appear before the current record.
47	same_merch_diff _card_3	The number of unique cards for transactions of this specific merchant number within the last 3 days. It only counts the records that appear before the current record.

48	same_merch_diff _card_7	The number of unique cards for transactions of this specific merchant number within the last 7 days. It only counts the records that appear before the current record.
49	same_merch_diff _card_14	The number of unique cards for transactions of this specific merchant number within the last 14 days. It only counts the records that appear before the current record.
50	same_merch_diff _card_28	The number of unique cards for transactions of this specific merchant number within the last 28 days. It only counts the records that appear before the current record.
51	amount_avg_1	The ratio of transaction amount of this specific card number to its historical 1-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
52	amount_max_1	The ratio of transaction amount of this specific card number to its historical 1-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
53	amount_median_ 1	The ratio of transaction amount of this specific card number to its historical 1-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
54	amount_mode_1	The ratio of transaction amount of this specific card number to its historical 1-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
55	amount_sum_1	The ratio of transaction amount of this specific card number to its total amount within the same day, excluding the fraudulent amounts. It only counts the records that appear before the current record.

56	amount_avg_3	The ratio of transaction amount of this specific card number to its historical 3-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
57	amount_max_3	The ratio of transaction amount of this specific card number to its historical 3-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
58	amount_median_3	The ratio of transaction amount of this specific card number to its historical 3-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
59	amount_mode_3	The ratio of transaction amount of this specific card number to its historical 3-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
60	amount_sum_3	The ratio of transaction amount of this specific card number to its total amount within the 3 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
61	amount_avg_7	The ratio of transaction amount of this specific card number to its historical 7-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
62	amount_max_7	The ratio of transaction amount of this specific card number to its historical 7-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
63	amount_median_7	The ratio of transaction amount of this specific card number to its historical 7-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.

64	amount_mode_7	The ratio of transaction amount of this specific card number to its historical 7-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
65	amount_sum_7	The ratio of transaction amount of this specific card number to its total amount within the 7 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
66	amount_avg_14	The ratio of transaction amount of this specific card number to its historical 14-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
67	amount_max_14	The ratio of transaction amount of this specific card number to its historical 14-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
68	amount_median_14	The ratio of transaction amount of this specific card number to its historical 14-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
69	amount_mode_14	The ratio of transaction amount of this specific card number to its historical 14-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
70	amount_sum_14	The ratio of transaction amount of this specific card number to its total amount within the 14 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
71	amount_avg_28	The ratio of transaction amount of this specific card number to its historical 28-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.

72	amount_max_28	The ratio of transaction amount of this specific card number to its historical 28-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
73	amount_median_28	The ratio of transaction amount of this specific card number to its historical 28-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
74	amount_mode_28	The ratio of transaction amount of this specific card number to its historical 28-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
75	amount_sum_28	The ratio of transaction amount of this specific card number to its total amount within the 28 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
76	merch_amount_avg_1	The ratio of transaction amount of this specific merchant number to its historical 1-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
77	merch_amount_max_1	The ratio of transaction amount of this specific merchant number to its historical 1-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
78	merch_amount_median_1	The ratio of transaction amount of this specific merchant number to its historical 1-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
79	merch_amount_mode_1	The ratio of transaction amount of this specific merchant number to its historical 1-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.

80	merch_amount_sum_1	The ratio of transaction amount of this specific merchant number to its total amount within the same day, excluding the fraudulent amounts. It only counts the records that appear before the current record.
81	merch_amount_avg_3	The ratio of transaction amount of this specific merchant number to its historical 3-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
82	merch_amount_max_3	The ratio of transaction amount of this specific merchant number to its historical 3-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
83	merch_amount_median_3	The ratio of transaction amount of this specific merchant number to its historical 3-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
84	merch_amount_mode_3	The ratio of transaction amount of this specific merchant number to its historical 3-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
85	merch_amount_sum_3	The ratio of transaction amount of this specific merchant number to its total amount within the 3 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
86	merch_amount_avg_7	The ratio of transaction amount of this specific merchant number to its historical 7-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.

87	merch_amount_max_7	The ratio of transaction amount of this specific merchant number to its historical 7-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
88	merch_amount_median_7	The ratio of transaction amount of this specific merchant number to its historical 7-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
89	merch_amount_mode_7	The ratio of transaction amount of this specific merchant number to its historical 7-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
90	merch_amount_sum_7	The ratio of transaction amount of this specific merchant number to its total amount within the 7 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
91	merch_amount_avg_14	The ratio of transaction amount of this specific merchant number to its historical 14-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
92	merch_amount_max_14	The ratio of transaction amount of this specific merchant number to its historical 14-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
93	merch_amount_median_14	The ratio of transaction amount of this specific merchant number to its historical 14-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.

94	merch_amount_mode_14	The ratio of transaction amount of this specific merchant number to its historical 14-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
95	merch_amount_sum_14	The ratio of transaction amount of this specific merchant number to its total amount within the 14 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.
96	merch_amount_avg_28	The ratio of transaction amount of this specific merchant number to its historical 28-day average amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
97	merch_amount_max_28	The ratio of transaction amount of this specific merchant number to its historical 28-day maximum amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
98	merch_amount_median_28	The ratio of transaction amount of this specific merchant number to its historical 28-day median amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
99	merch_amount_mode_28	The ratio of transaction amount of this specific merchant number to its historical 28-day mode amount, excluding the fraudulent amounts. It only counts the records that appear before the current record.
100	merch_amount_sum_28	The ratio of transaction amount of this specific merchant number to its total amount within the 28 days, excluding the fraudulent amounts. It only counts the records that appear before the current record.

101	last_trans	The difference in merchant's zips between the current transaction of this specific card number and its last transaction.
102	day_of_week	How common it is for a transaction to happen on a certain day of week.

Part IV. Feature Selection

Before feature selection, we separated our dataset into training, testing, and out-of-time. The out-of-time dataset consists the last two months of transactions. We randomly sampled 80% of the first ten-month transactions as the training dataset, and the rest 20% transactions as the testing dataset.

We decided to use a "Filter" method before implementing our algorithms. The filter that we chose is to calculate the "Kolmogorov-Smirnov" score (KS Score) for each unique feature. The Kolmogorov-Smirnov test is to measure how separate is one distribution from the other distribution.

For applying to feature selection:

- Firstly, we derived two distributions from each variable: "Fraud=1" Distribution and "Fraud=0" Distribution.
- Then we used Kolmogorov-Smirnov test to see how well these two distributions are separated. The higher the KS score, the better performance of the variable in separating "goods" and "bads."

Based on the training set, we calculated the KS scores for all 102 variables we created and variable "Amount" in the original data. We picked the top 63 variables which have KS scores greater than 10 for our algorithms. The chart below lists the top 63 variables and their KS Score:

Variable Name	KS Score
Amount	50.62
amount_sum_28	47.17
amount_avg_28	44.03

amount_max_28	43.37
merch_amount_avg_28	43.31
merch_amount_median_28	42.98
amount_sum_14	42.74
merch_amount_median_14	41.13
merch_amount_avg_14	41.06
amount_max_14	40.43
same_card_7	40.18
amount_sum_7	39.71
amount_median_28	39.25
amount_sum_3	38.90
amount_avg_14	38.85
merch_amount_avg_7	38.49
same_card_3	38.46
merch_amount_median_7	37.23
merch_amount_mode_14	36.16
amount_max_7	36.14
amount_median_14	35.90
merch_amount_mode_7	35.83
merch_amount_max_28	35.71
merch_amount_mode_28	35.53
same_card_14	34.83
amount_avg_7	33.69
same_merch_3	33.24
amount_median_7	32.15

merch_amount_max_14	31.71
merch_amount_median_3	31.47
same_merch_7	31.18
amount_max_3	31.16
merch_amount_mode_3	31.01
merch_amount_sum_14	30.85
merch_amount_sum_28	30.43
merch_amount_avg_3	30.19
amount_mode_7	30.10
same_card_1	29.71
amount_mode_14	28.65
amount_avg_3	28.11
same_merch_14	28.09
amount_mode_28	27.14
same_card_28	25.95
same_merch_28	25.82
merch_amount_max_7	25.75
same_merch_1	25.29
merch_amount_sum_3	25.15
merch_amount_sum_1	25.12
merch_amount_sum_7	24.89
amount_sum_1	24.87
merch_amount_max_3	23.83
merch_amount_max_1	23.50
amount_median_3	22.70

merch_amount_avg_1	22.07
amount_mode_3	21.81
last_trans	16.14
amount_max_1	15.91
merch_amount_median_1	15.69
day_of_week	14.68
amount_avg_1	14.47
merch_amount_mode_1	12.35
amount_median_1	11.47
amount_mode_1	10.56

Part V. Algorithms

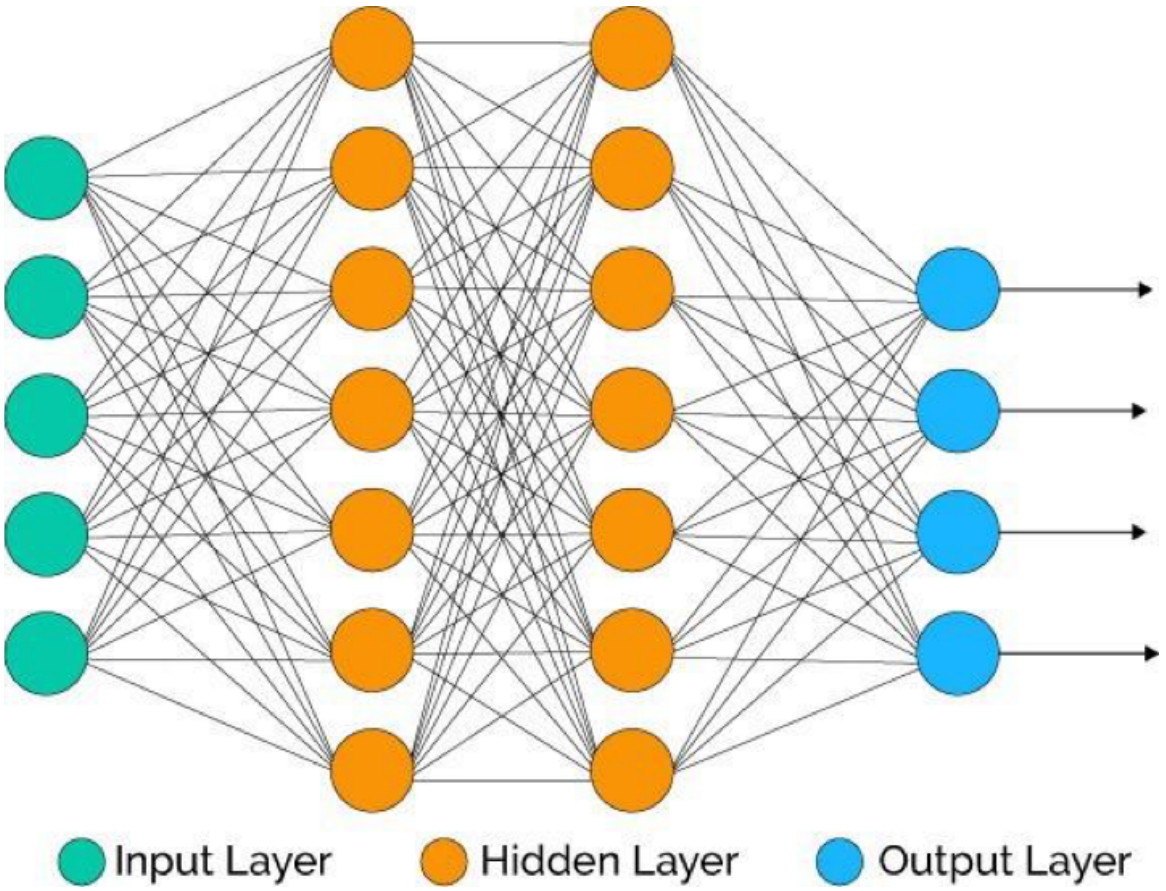
Logistic Regression

Logistic regression uses maximum likelihood to fit the logistic function and gives outputs of the probability of fraud. Mathematically, logistic regression model gives a linear fit to log-odds and estimates the coefficients to yield a number close to 1 for each individual who committed fraud, and a number close to 0 for each individual who did not commit fraud. To reduce dimensionality in the model and hence reduce the model variance, we used both-way stepwise logistic regression to select important variables in predicting fraud. The both-way stepwise logistic regression method starts modeling with nothing, then test the addition or reduction of each variable according to AIC, and add or drop the variables to achieve AIC improvement. It will repeat this process until none improves the model to a statistically significant extent. By implementing stepwise logistic regression, we selected 20 out of the 30 variables filtered by KS. Then we used these 20 variables to build logistic regression. The 20 variables selected include: 'amount_sum_28', 'Amount', 'same_card_3', 'amount_sum_3', 'same_card_7', 'same_merch_3', 'same_card_14', 'amount_sum_7', 'amount_avg_7', 'amount_max_7', 'amount_avg_14', 'merch_amount_mode_14', 'merch_amount_max_28', 'merch_amount_median_28', 'merch_amount_mode_28', 'merch_amount_avg_28',

'merch_amount_median_3', 'merch_amount_avg_7', 'merch_amount_avg_14', 'merch_amount_max_14'.
Using logistic regression, we got FDR@2% of 50.6% for the training dataset and FDR@2% of 54.8% for the testing dataset. For the OOT dataset, we got FDR@2% of 53.6%.

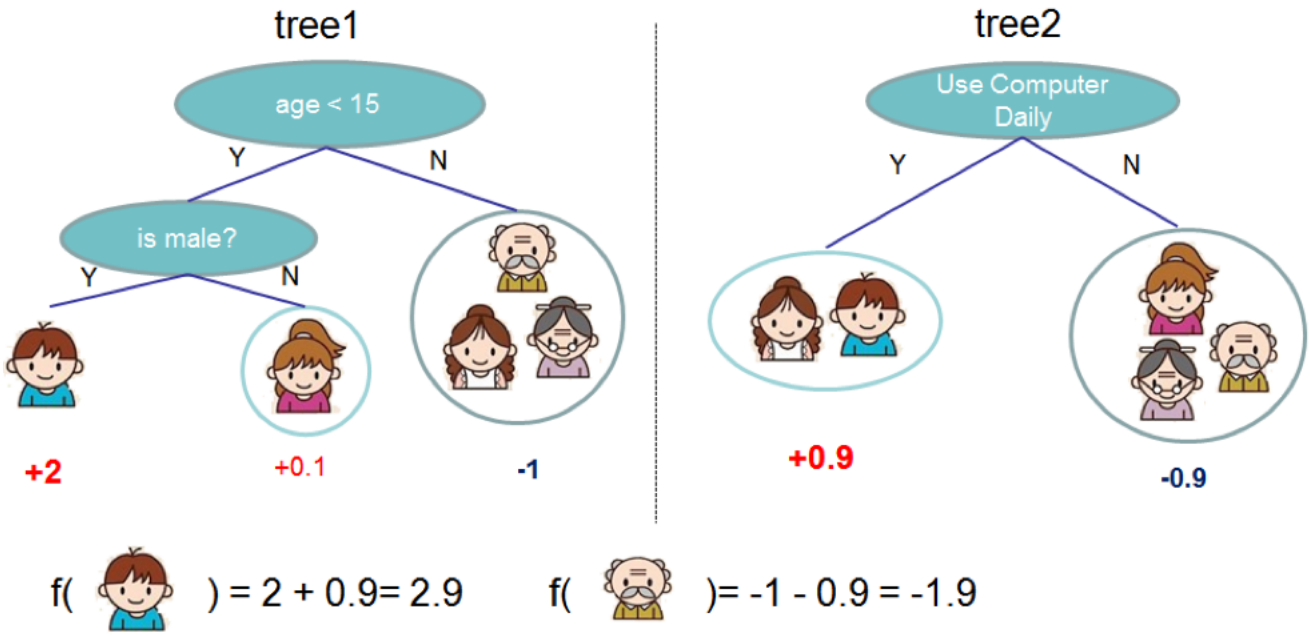
Neural Network

A neural network has an input layer, one or more hidden layers, and an output layer. For this model, each node in the input layer consists the information of all selected variables for each record. The nodes in the output layer are either 0 or 1; 0 represents non-fraudulent transaction, and 1 represents fraud. For the hidden layer, we chose 1 hidden layer with 5 nodes at the beginning. Then we slightly increase the size of hidden layers and hidden nodes to 2 layers with 5 nodes in each layer. The result is FDR@2% of 87.6% for the training dataset and FDR@2% of 83.0% for the testing dataset. For the OOT dataset, we got FDR@2% of 66.6%.



Extreme Gradient Boosting (XG-Boost)

XG-Boost is short for "Extreme Gradient Boosting", where the term "Gradient Boosting" is proposed in the paper *Greedy Function Approximation: A Gradient Boosting Machine,* by Friedman. XG-Boost is developed based on the gradient boosting model while taking into consideration of systems optimization. The goal of this model is to push the extreme of the computation limits of machines to provide a scalable, portable and accurate library.
XG-Boost's model is tree ensemble. The tree ensemble model is a set of classification and regression trees (CART). A CART is a bit different from decision trees, where the leaf only contains decision values. In CART, a real score is associated with each of the leaves, which gives us richer interpretations that go beyond classification. Usually, a single tree is not strong enough to be used in practice. What is actually used is the so-called tree ensemble model, which sums the prediction of multiple trees together.



Here is an example of a tree ensemble of two trees, or two weak learners, which are defined as ones whose performance are at least slightly better than random chance. The prediction scores of each individual tree are summed up to get the final score. You can tell from the example here that the two trees try to complement each other. This is also how XG-Boost works to improve prediction accuracy by slowly adding up multiple weak learners (trees) to minimize the loss function.
To sum up, XG-Boost involves four elements:

1.A loss function to be optimized.

2.A weak learner to make predictions.

3.An additive model to add weak learners to minimize the loss function.

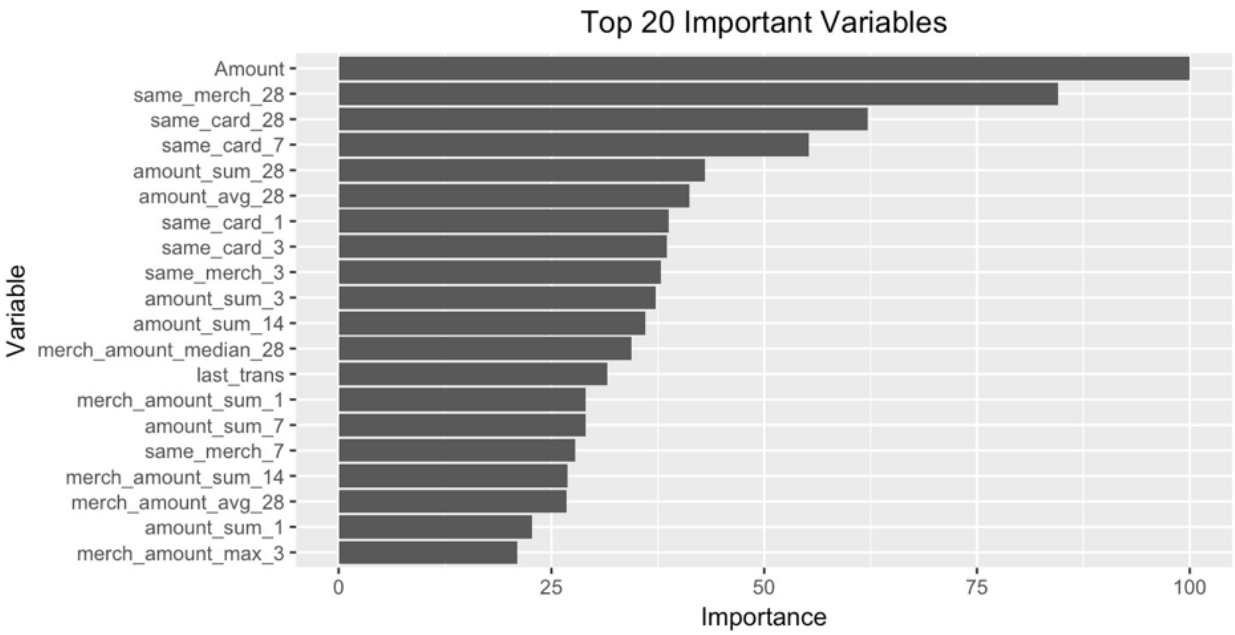
4.An improved algorithm that push the extreme of the computation limits of machines to outperform other tree-based algorithms in terms of computation time.

We used all 63 variables selected by the KS in the feature selection step when training the model and used the grid search to perform 5-fold cross-validation to tune the hyperparameters. Finally, we achieved the best performance on FDR@2% using the following parameters:

- nrounds = 90.
  - It measures the number of iterations and should be in a negative relationship with the value of eta.
- max_depth = 30.
  - It controls the maximum depth of a tree, same as in the gradient boosting model
  - It is mainly used to control overfitting.
- eta = 0.2.
  - It is analogous to learning rate in GBM and can make the model more robust by shrinking the weights on each step.
- gamma = 1.
  - A node is split only when the resulting split gives a positive reduction in the loss function. Gamma specifies the minimum loss reduction required to do a split.
- colsample_bytree = 0.9.
  - It is same as the max_features in gradient boosting model and denotes the fraction of columns to be randomly sampled for each tree.
- min_child_weight = 1.
  - It is defined as the minimum sum of weights of all observations required in a child.
  - It is used to control overfitting. Higher the value, less likely the model to overfit.
- subsample = 0.65.
  - It is the same as the subsample in the gradient boosting model and denotes the fraction of observations to be randomly samples for each tree.

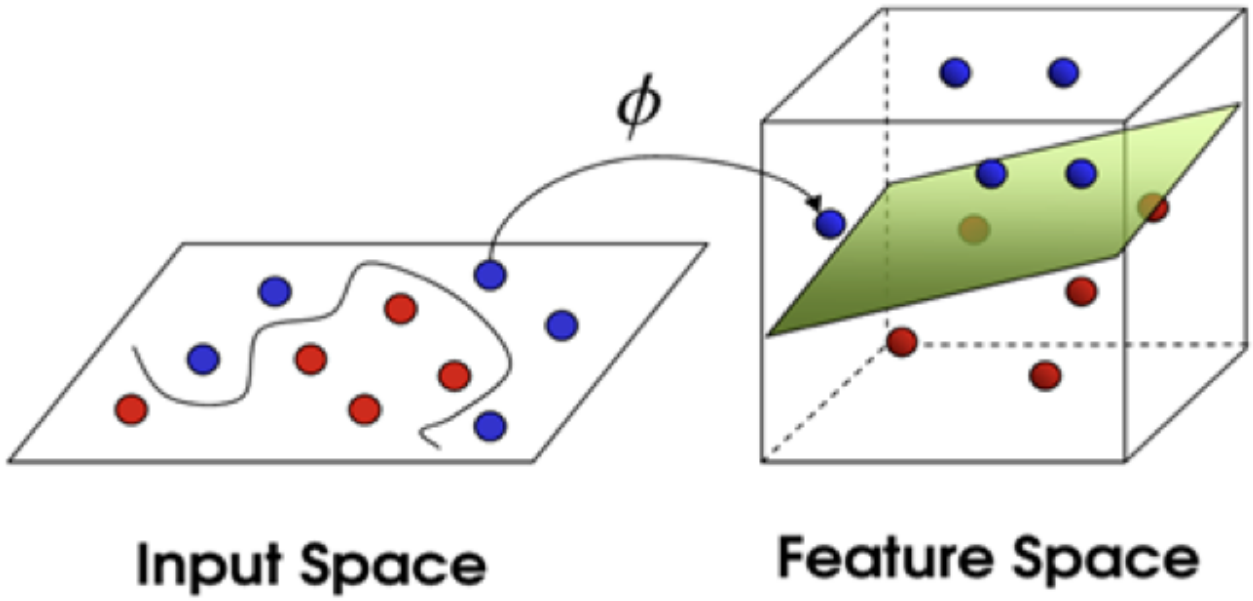
Using XG-Boost, we got FDR@2% of 100% for the training dataset and FDR@2% of 96.3% for the testing dataset. For the OOT dataset, we first trained the model on both training and testing datasets before testing on OOT dataset, and got FDR@2% of 73.7%, which has best

performance among all models. We also output the top 10 important variables selected by our XGBoost Model.



Support Vector Machine (SVM)

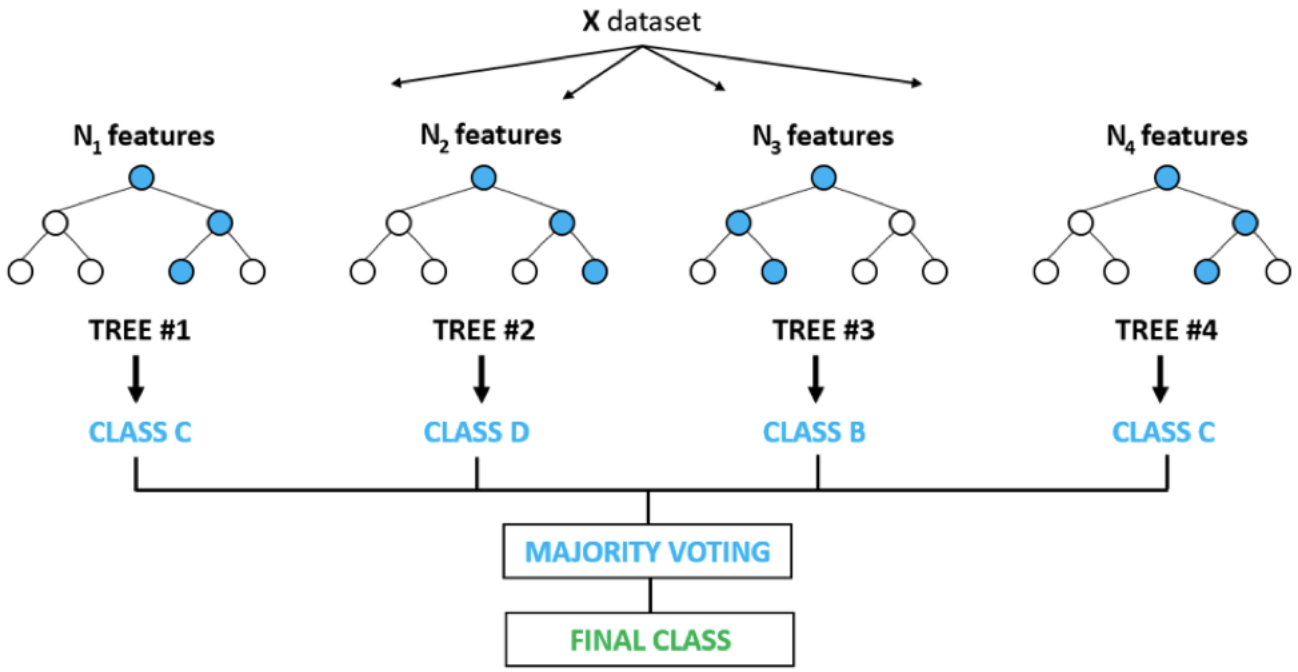
We chose to use SVM as one of our classification algorithms. SVM is a great supervised learning algorithm for both classification and regression problems. As a classifier, SVM can form a non-linear decision boundary(hyperplane) by projecting input observations to higher dimension spaces and split them as wide as possible.



Using SVM can bring the following advantages:

1. SVM can form various non-linear boundaries, which is specified by the kernel of the SVM.
2. SVM is robust to the noisy observations. Observations directly on the margin or on the wrong side of the margin for their class are known as support vectors since the observations do affect the shape of the support vector classifier. Only those support vectors will have an impact on the performance of the model, which is a very small fraction of the whole sample size.
3. SVM function in R can automatically scale the variables as needed for better performance. We used all 63 variables that we selected from feature selection stage to fit the SVM model, and we used the default parameters settings (Cost=1, gamma=0.015) that the function returned. SVM can give each observation a probability for each level, and we treat the probability to be classified as "1" (fraud) as the score to sort our populations. We got FDR@2% of 89.6% of the training dataset. We got FDR@2% of 86.7% for the testing dataset, which is slightly lower than the training FDR as expected. Then we re-trained a model on the combination of training and testing dataset and predicted on the OOT set. FDR@2% on the OOT set is 65.4%, which is good but still worse than the result of Random Forest. SVM is a powerful tool in theory, but the actual performance is sometimes not very ideal.

Random Forests

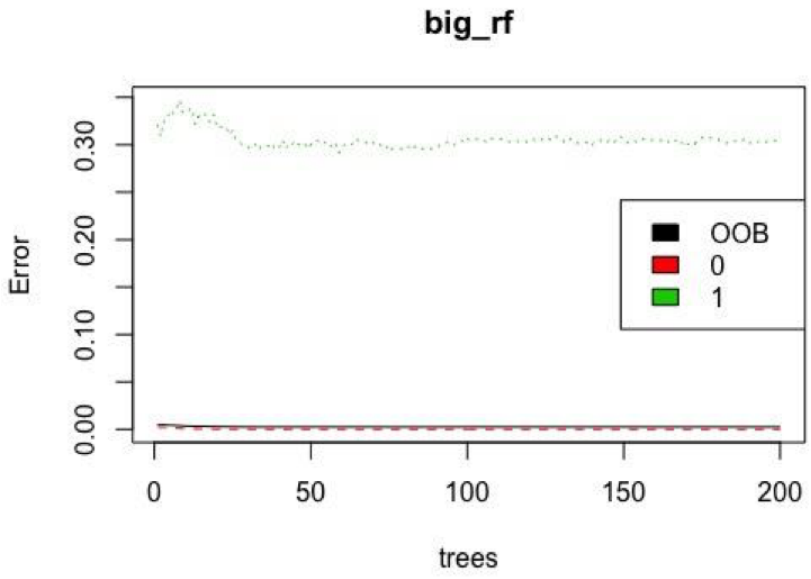


Random Forest is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees. Random forests are similar to bagged trees; however, random forests introduce a randomized process that helps decorrelate each tree. Random forests algorithm uses a resampling with replacement method (also regarded as Bootstrap Method) to generate multiple samples. For each sample, it will apply a classification decision tree model to predict each observation using a random subset of features space. Then, the final output is the combination of the result from each tree.
Random Forests have many advantages over other classifiers:

1. High Accuracy
2. Randomness will prevent the model from overfitting
3. Randomness will make the model robust, not suffering too much noise.
4. Random Forests can take in high dimension dataset and it can output the importance index for each variable.
5. Random Forests do not require the data to be standardized.
6. Random Forests do not need cross-validation for the performance measure since it will generate the OOB (Out of Bag) Estimation.
7. Random Forests do not have too many parameters to tune.

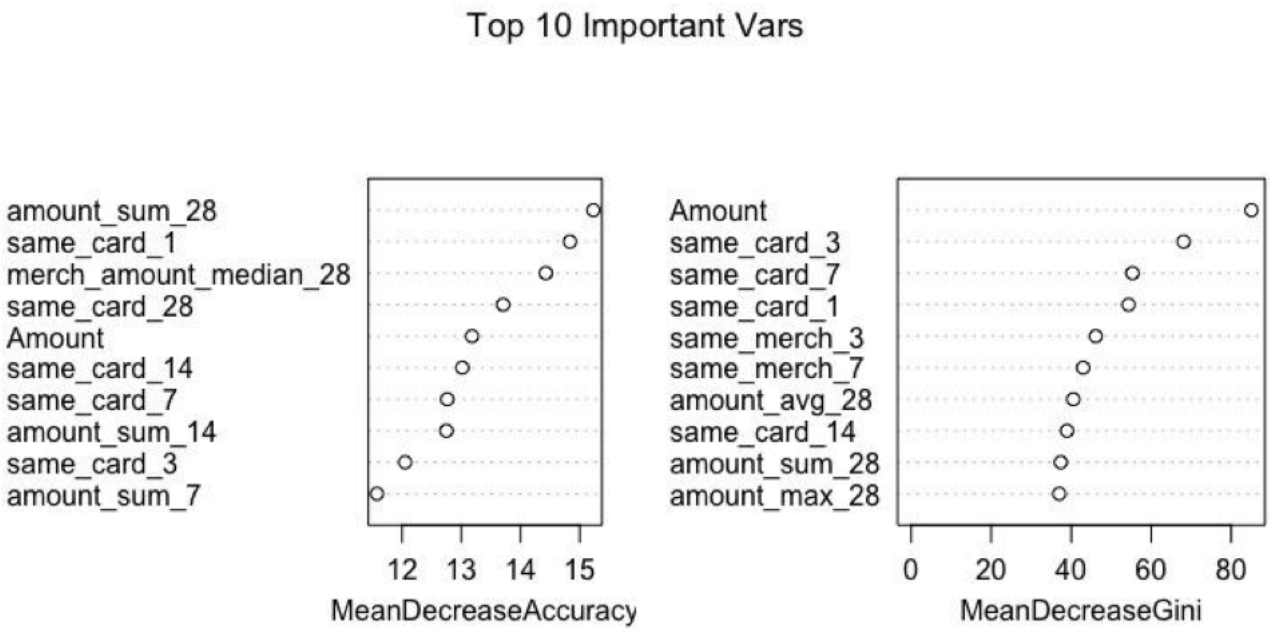
For this transactions dataset, Random Forests have good performance over most of our other algorithms. In R, Random Forests only have two parameters to tune:

- mtry: Number of variables randomly sampled as candidates at each split.
- ntree: Number of trees to grow.



For mtree, we can definitely tune this parameter, however, if we use a large enough mtree, the performance of the model will not have a significant change after a certain number of trees being built. We used 200 trees for the training dataset to train the model, as can be seen from the graph above, the error rate became stable after we built about 50 trees.

For the parameter mtry, we tuned this parameter and the optimal number of variables to be selected at each split is 14 with OOB error of 0.27%. Then we used mtree=50 and mtry=14 to run the Random Forests on the training and testing dataset. The FDR at 2% population for training is 91%. The FDR at 2% population for testing is 93.3%. We combined training and testing data together and built a model on this large dataset to get a model for the hold-out test dataset. On the hold-out test dataset, we got the FDR at 2% of 71.9%, which is pretty good. We also output the top 10 important variables for our Random Forests model.



The results of all above algorithms are shown as follows:

	FDR @ 2%		
Model	Training	Testing	OOT
Logistic Regression	50.6%	54.8%	53.6%

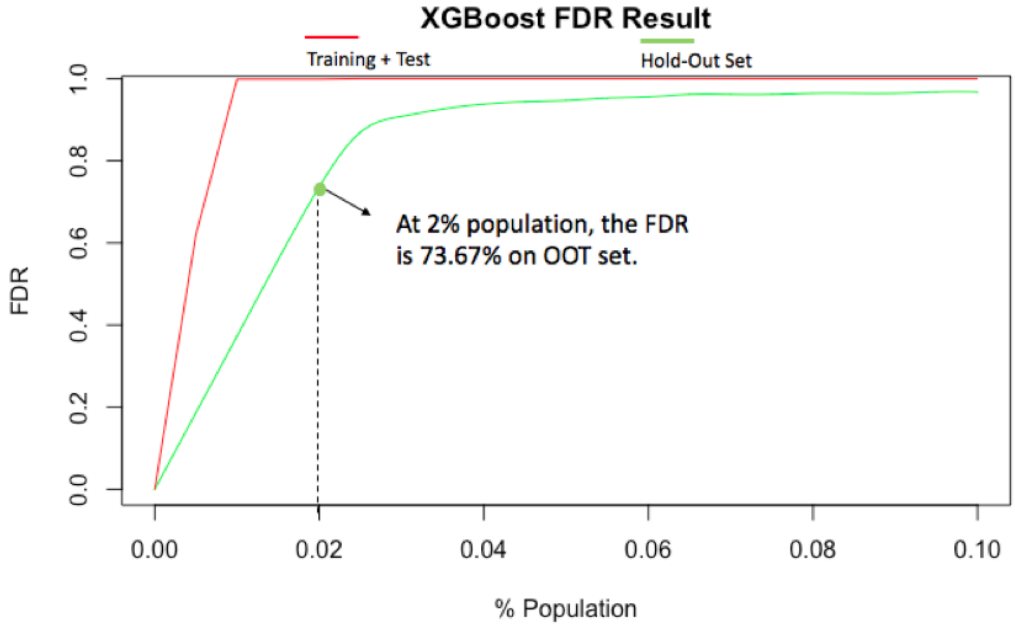
Neural Network	87.6%	83.0%	66.6%
XGBoost	100.0%	96.3%	73.7%
Support Vector Machine	89.6%	86.7%	65.4%
Random Forests	91.0%	93.3%	71.9%

Part VI. Results and Financial Analysis

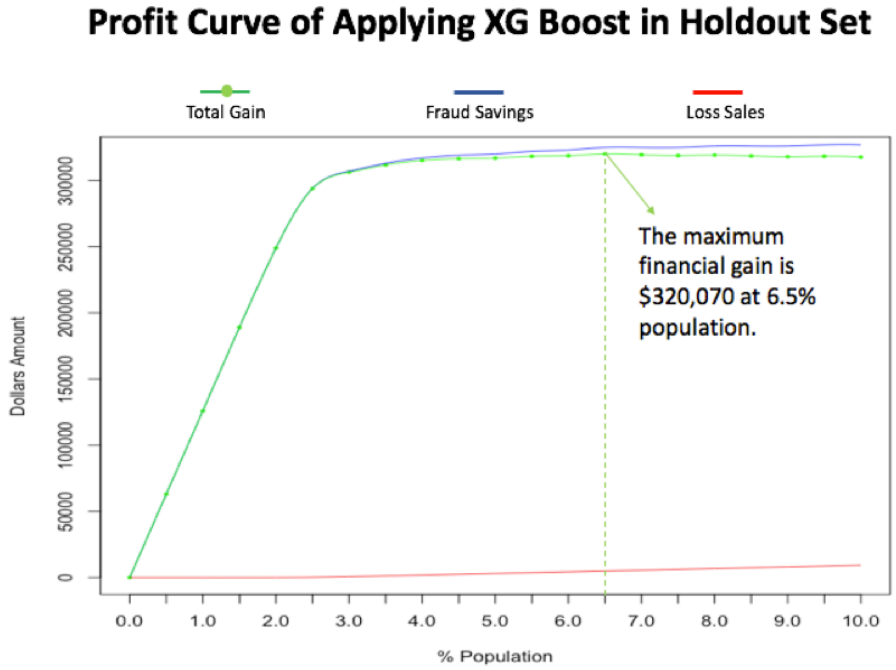
XG-Boost has the best performance among all algorithms with FDR at 2% population achieving 73.67%. At 2% population cut-off, the model only misclassified 3 good transactions as fraudulent transactions. The table below shows top 10% bins of the population. At 10% population cut-off, the FDR is 96.75%.

Overall Bad rate is 2.69%	Bin Statistics					Cumulative Statistics					
Population Bin%	Total # Record	#Good	#Bad	%Good	%Bad	Cml # Good	Cml # Bad	Cml % Good	Cml % Bad(FDR)	KS	False Positive Ratio
0.5	63	0	63	0.00	100.00	0	63	0.00	18.64	18.64	0.00
1	63	0	63	0.00	100.00	0	126	0.00	37.28	37.28	0.00
1.5	63	0	63	0.00	100.00	0	189	0.00	55.92	55.92	0.00
2	63	3	60	4.76	95.24	3	249	0.02	73.67	73.64	0.01
2.5	63	18	45	28.57	71.43	21	294	0.17	86.98	86.81	0.07
3	63	50	13	79.37	20.63	71	307	0.58	90.83	90.25	0.23
3.5	63	57	6	90.48	9.52	128	313	1.05	92.60	91.56	0.41
4	62	58	4	93.55	6.45	186	317	1.52	93.79	92.27	0.59
4.5	63	61	2	96.83	3.17	247	319	2.02	94.38	92.36	0.77
5	63	62	1	98.41	1.59	309	320	2.52	94.67	92.15	0.97
5.5	63	61	2	96.83	3.17	370	322	3.02	95.27	92.25	1.15
6	63	62	1	98.41	1.59	432	323	3.53	95.56	92.04	1.34
6.5	63	61	2	96.83	3.17	493	325	4.03	96.15	92.13	1.52
7	63	63	0	100.00	0.00	556	325	4.54	96.15	91.61	1.71
7.5	63	63	0	100.00	0.00	619	325	5.05	96.15	91.10	1.90
8	63	62	1	98.41	1.59	681	326	5.56	96.45	90.89	2.09
8.5	63	63	0	100.00	0.00	744	326	6.07	96.45	90.38	2.28
9	63	63	0	100.00	0.00	807	326	6.59	96.45	89.86	2.48
9.5	63	62	1	98.41	1.59	869	327	7.10	96.75	89.65	2.66
10	63	63	0	100.00	0.00	932	327	7.61	96.75	89.14	2.85

The line graph below further compares the FDR performance on the training+testing dataset and out-of-time dataset.



The following graph shows the financial indication of applying the XG-Boost model in the holdout set.When we analyze the financial implications, there are several important assumptions listed as below:



•Loss Sales = $10 * # false positive

Whenever we mistakenly flag a good transaction as a fraud, we anger the customer, and some of them leave. On average, we estimate a $10 penalty for every one of these false positives.

•Fraud Saving = $1000 * # true positive

Every true fraud that we catch stops on average a $1000 loss, so we save $1000 for every fraud we catch with our algorithm.

In the graph above, the blue curve represents the fraud savings which keeps increasing as the model catches more fraudulent records. The red curve represents the loss sales, which also keeps increasing since the number of cumulative good that the model catches is monotonously increasing. At 6.5% cut-off, the loss sale is $4,930, and the fraud saving is $325,000. Therefore, with 0.5% precision, the total gains (fraud saving - loss sales) of applying the XG-Boost model in hold-out set achieves the maximum ($320,070) at 6.5% population cut-off. After that, the profit starts decreasing.

In other words, at 6.5% population cut-off, we will achieve the highest profit which is $320,070.

Part VII. Conclusions

In this fraud detection project, we first created 102 new variables in the following 5 ways:

1. Measure transaction frequency during a set period of time, both at the card level and the merchant level;
2. Capture the purchasing pattern during a set period of time, both at the card level and merchant level;
3. Compare the current transaction amount to the historical amounts during a set period of time, both at the card level and merchant level
4. Measure the distance between merchants purchased by the same card
5. Capture the seasonality of the purchasing frequency.

Then, we eliminated all transaction types other than type P, corrected one outlier in “amount” variable which was due to currency type difference and filled missing values in variables: “Merchantstate”, “MerchantZip”, “Merchantnum”. Next, we separated the dataset into training, testing and out-of-time sets, and did feature selection on the training dataset using KS and picked the top 63 variables with the highest KS scores. Last, we used Logistic Regression, Neural Network, Extreme Gradient Boosting (XG-Boost), Support Vector

Machine and Random Forest to test individual algorithm's performance on the dataset. As a result, XG-Boost algorithm has the best performance with FDR @ 2% of 73.7% on the out-of-time dataset.

Overall, the model that we built has powerful performance, but there still remains some room for us to improve our methodology. With more time, we plan to:

- Calculate correlation among variables when doing the feature selection
- Create more features to characterize what a fraudster would do when committing fraud crime.
- In the dataset, there is about 99% of the transactions are in the "Not-Fraud" class, and 1% are in the "Fraud" class. To fix this problem, we can use an oversampling method such as SMOTE and ADASYN to create synthetic samples from the minor class.
- Try more feature selection methods and classification models.

Data Quality Report on Credit Card Transaction Data

Team 5

SUMMARY STATISTICS

Overall Description

There are altogether 96,708 observations and 10 variables in this dataset. It includes the transaction information of the credit card payment of the workers in federal government from January 1st to December 31st in 2010. Each transaction is labeled as fraud (represented by 1) or non-fraud (represented by 0). In total, there are 1014 fraudulent transactions in this dataset.

For all the 10 variables, we provide a summary table including the variable name, the data type, the number of unique values, the number of missing values and the percentage populated in the field to describe the data uniqueness and completeness in this dataset.

No.	Variable Name	Data Type	Number of Unique Values	Number of NAs	Data Completeness (%)
1	**Recordnum**	Categorical	96708	0	100.00
2	**Cardnum**	Categorical	1644	0	100.00
3	**Date**	Date	365	0	100.00
4	**Merchantnum**	Categorical	13091	3375	96.51
5	**Merch Description**	Categorical	13125	0	100.00
6	**Merchant State**	Categorical	228	1195	98.76
7	**Merchant Zip**	Categorical	4568	4656	95.19
8	**Transtype**	Categorical	4	0	100.00
9	**Amount**	Numeric	34876	0	100.00
10	**Fraud**	Categorical	2	0	100.00

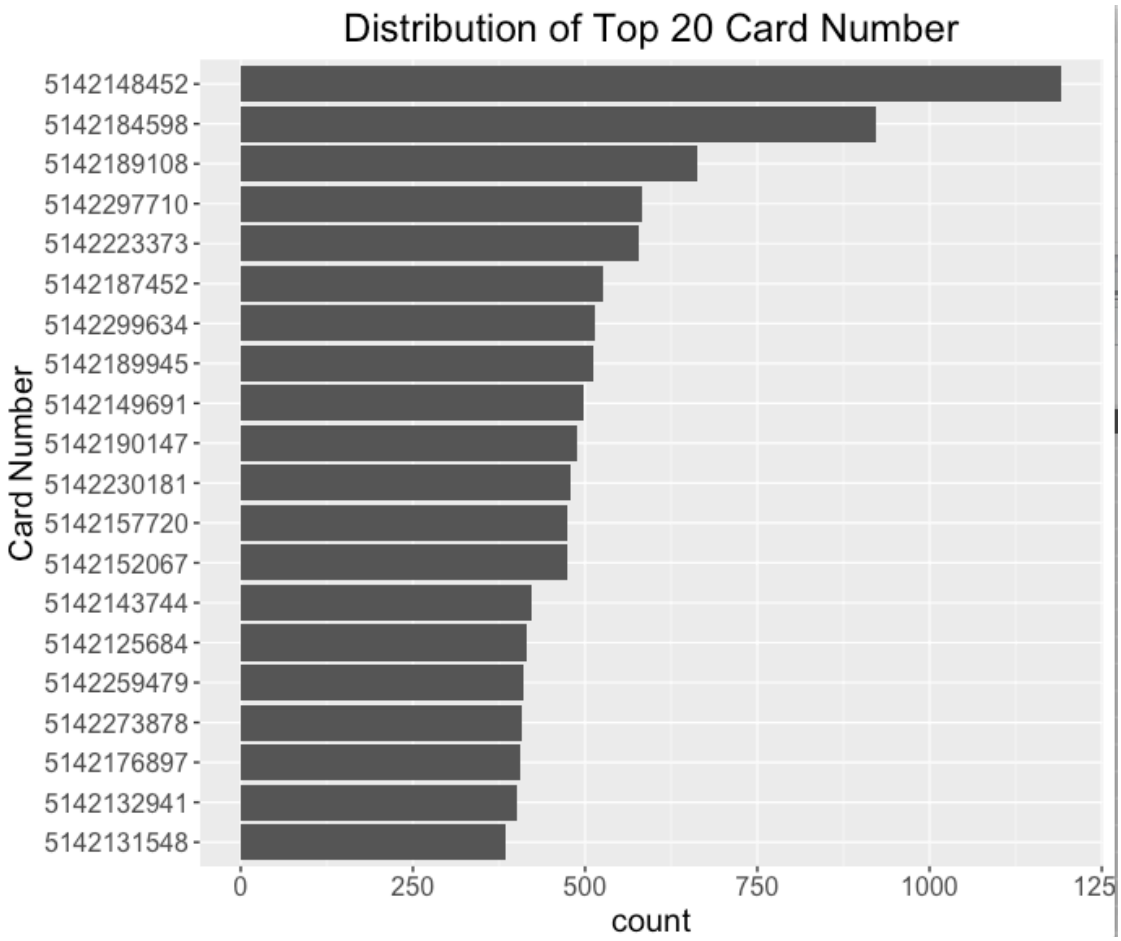
FIELD EXPLANATION

1. Recordnum

Recordnum is a categorical variable with 96708 unique values. It is the unique ordinal reference number for each transaction record.

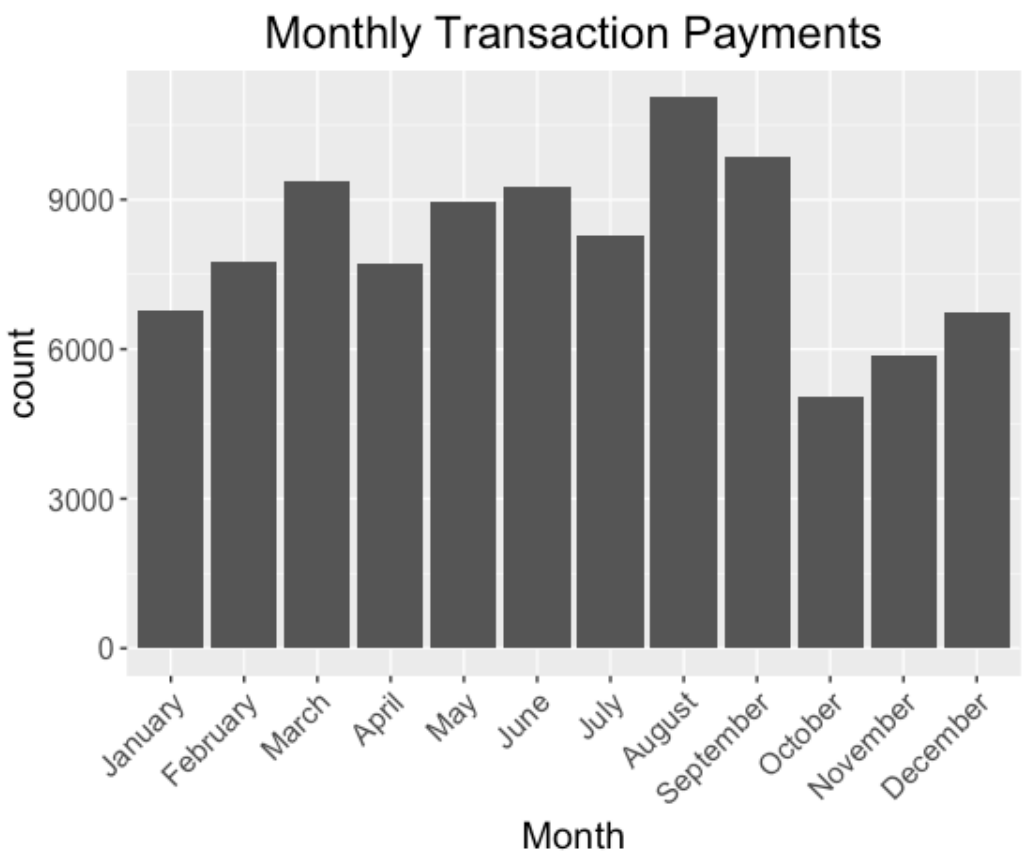
2. Cardnum

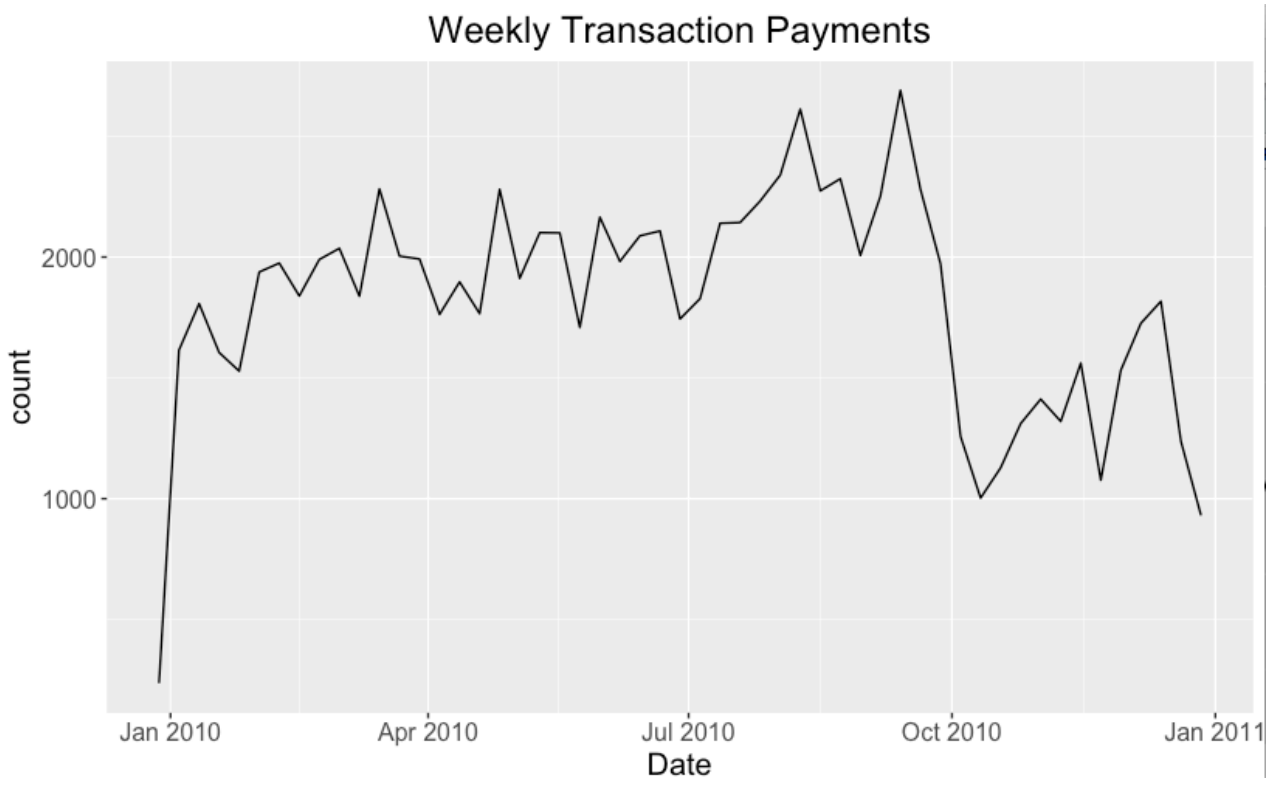
Cardnum is a categorical variable with 1644 unique values. It is the number of the credit card used in each transaction.

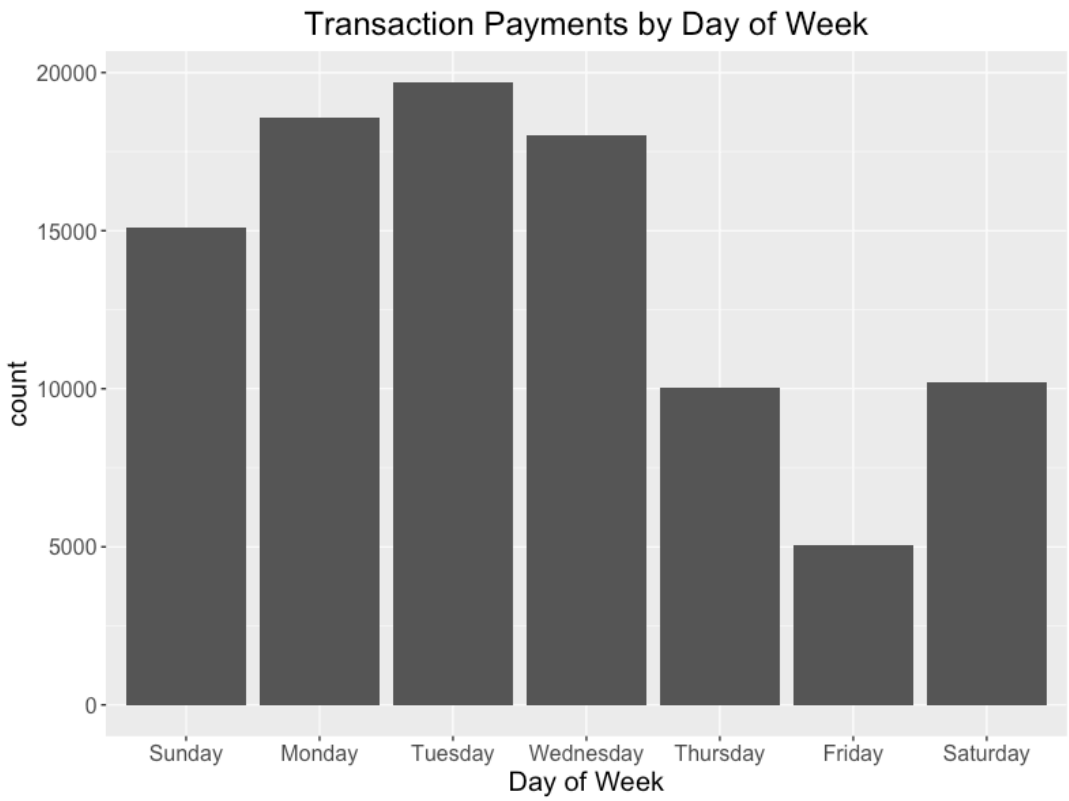


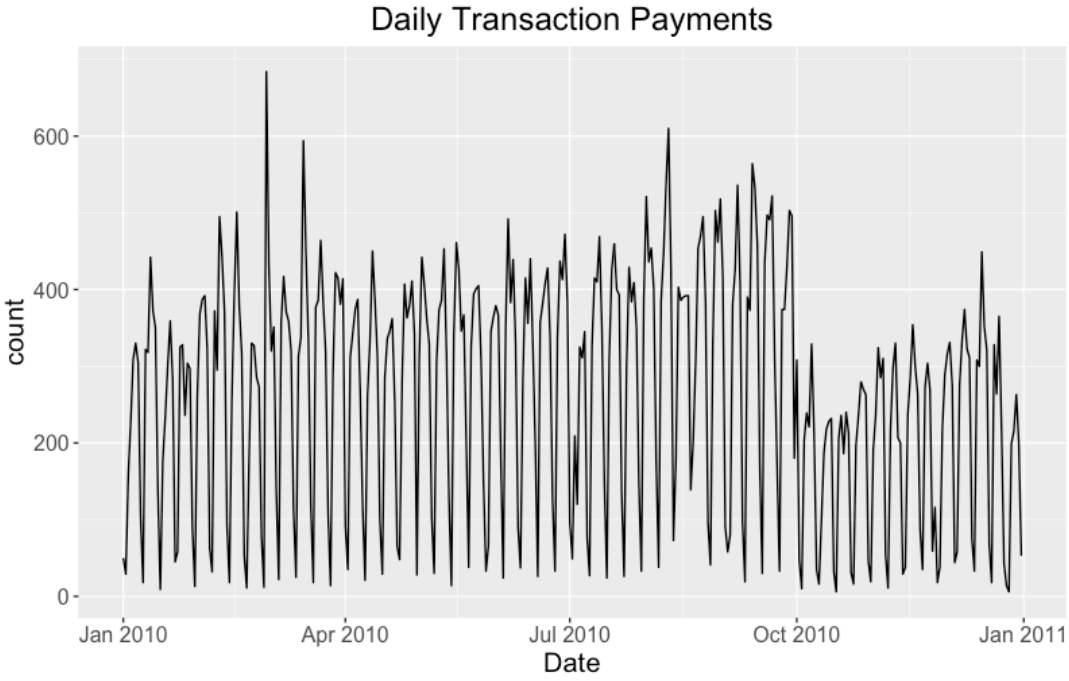
3. Date

Date is a date variable which takes the value ranging from 2010-01-01 to 2010-12-31. It is the date when the credit card transaction actually happened. We can tell from the graphs that there is a strong seasonality in the transaction payment trend throughout the year 2010.


Monthly Transaction Payments
count
9000
6000
3000
0
January
February
March
April
May
June
July
August
September
October
November
December
Month


Weekly Transaction Payments
count
2000
1000
Jan 2010
Apr 2010
Jul 2010
Oct 2010
Jan 2011
Date


Transaction Payments by Day of Week
20000
15000
10000
5000
0
count
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Day of Week

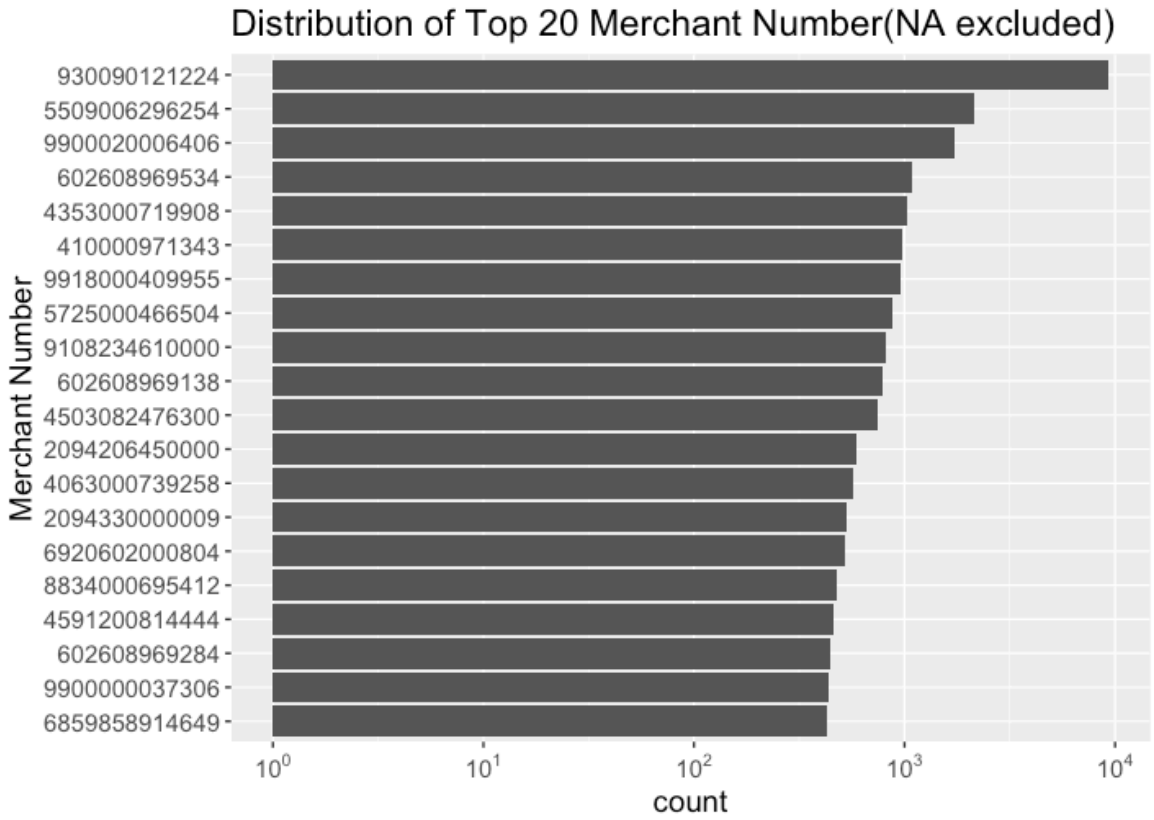

Daily Transaction Payments
600
400
200
0
count
Jan 2010
Apr 2010
Jul 2010
Oct 2010
Jan 2011
Date

According to the monthly transaction trend, it is clear from the graph that the number of transactions gradually increased from January to September with moderate fluctuations at the end of each season, and dropped sharply in October. We believe it was mainly due to the fact that the federal government reset the yearly budget for each department at the end of September.

According to the number of transactions by day of week, we can see that there is also an obvious cycle within weeks. The number of payments gradually decrease from Tuesday to Friday, and increase from weekend to the next Tuesday. Consequently, the number of transactions is lowest on Friday and highest on Tuesday.

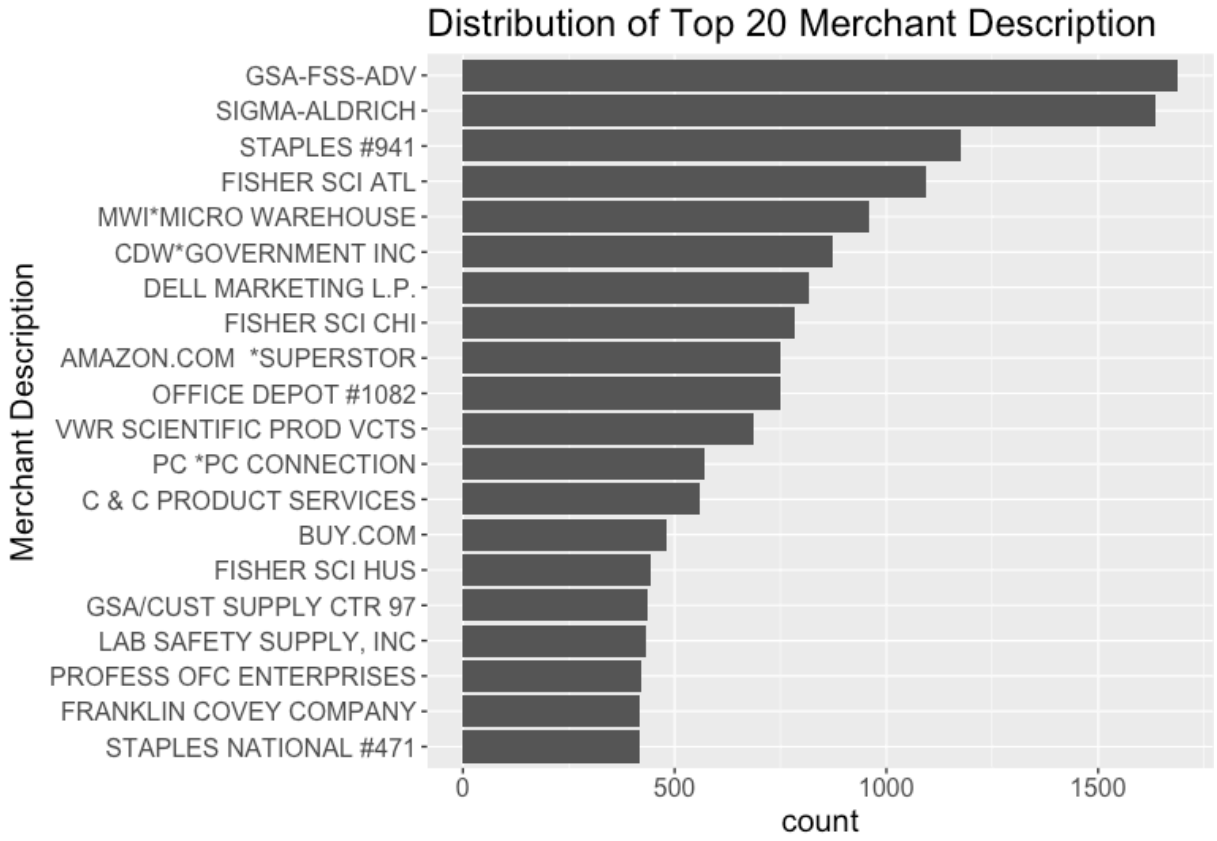
4. Merchantnum

Merchantnum is a categorical variable with 13091 unique values. It is the unique identifier for each merchant.



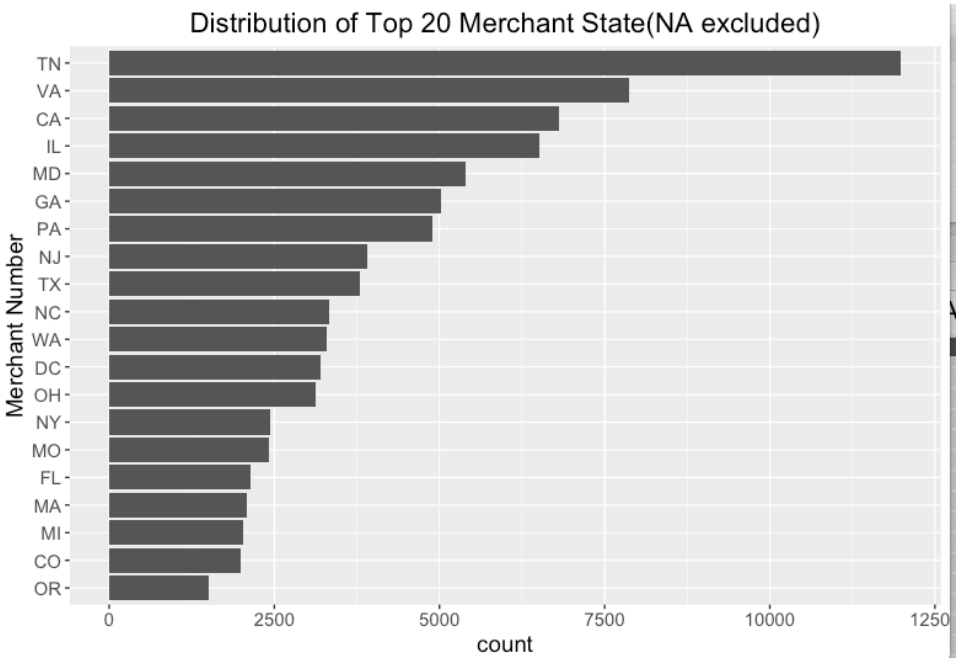
5. Merch Description

Merch Description is a categorical variable with 13125 unique values. It contains some detailed information about the merchant in each credit card transaction, such as the name of the merchant, etc.

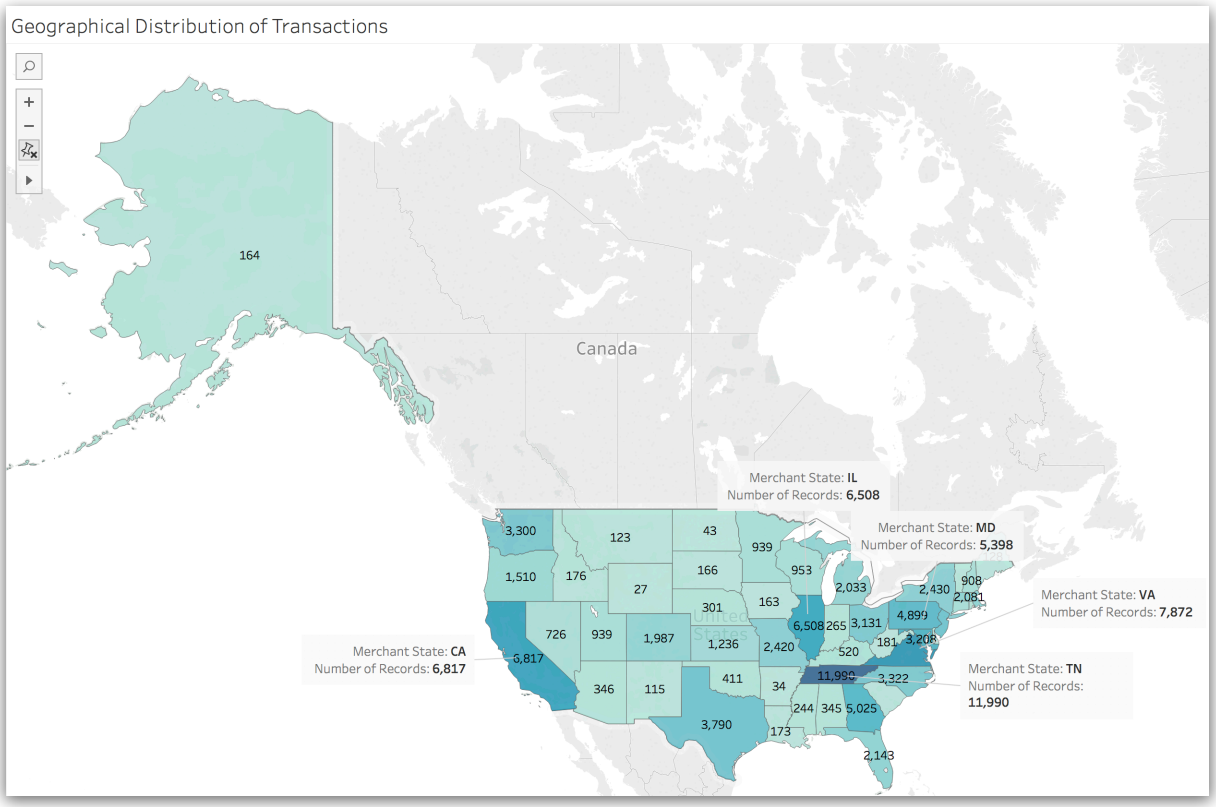


6. Merchant State

Merchant State is a categorical variable with 228 unique values. It consists of not only the two-letter abbreviation of American states, but also some digit combinations that indicate some specific geographical locations all around the globe.

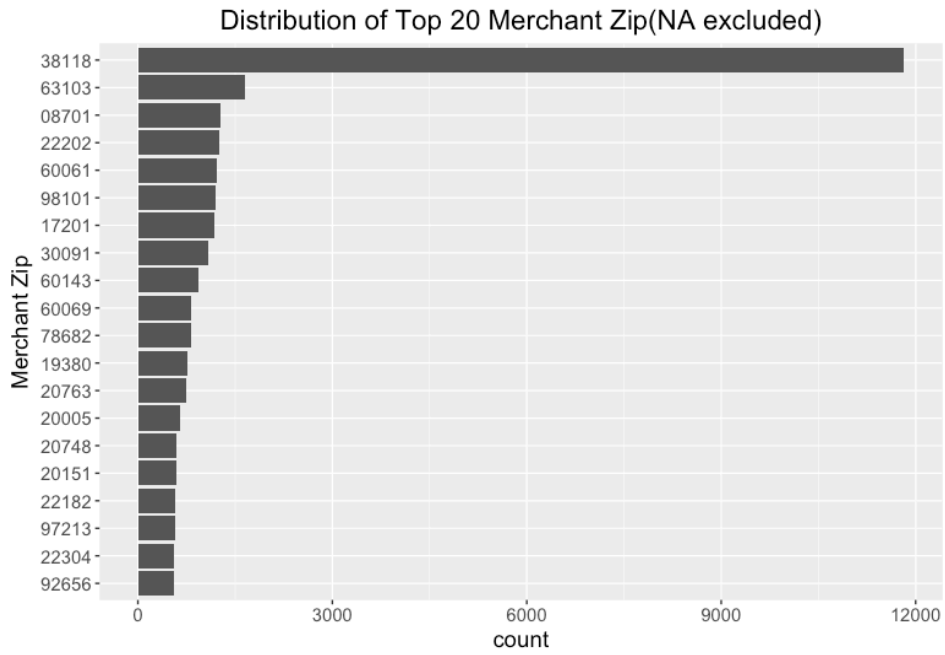


The following map shows the geographical distribution of the transaction records across the United States. The deeper the color, the more the number of transactions in the region. We can tell from the map that the top 5 states with the most credit card transactions are Tennessee, Virginia, California, Illinois and Maryland.



7. Merchant Zip

Merchant Zip is a categorical variable with 4568 unique values. It indicates the zip code of the merchant.



8. Transtype

Transtype is a categorical variable with 4 unique values: P, A, D and Y, among which P takes up the majority.

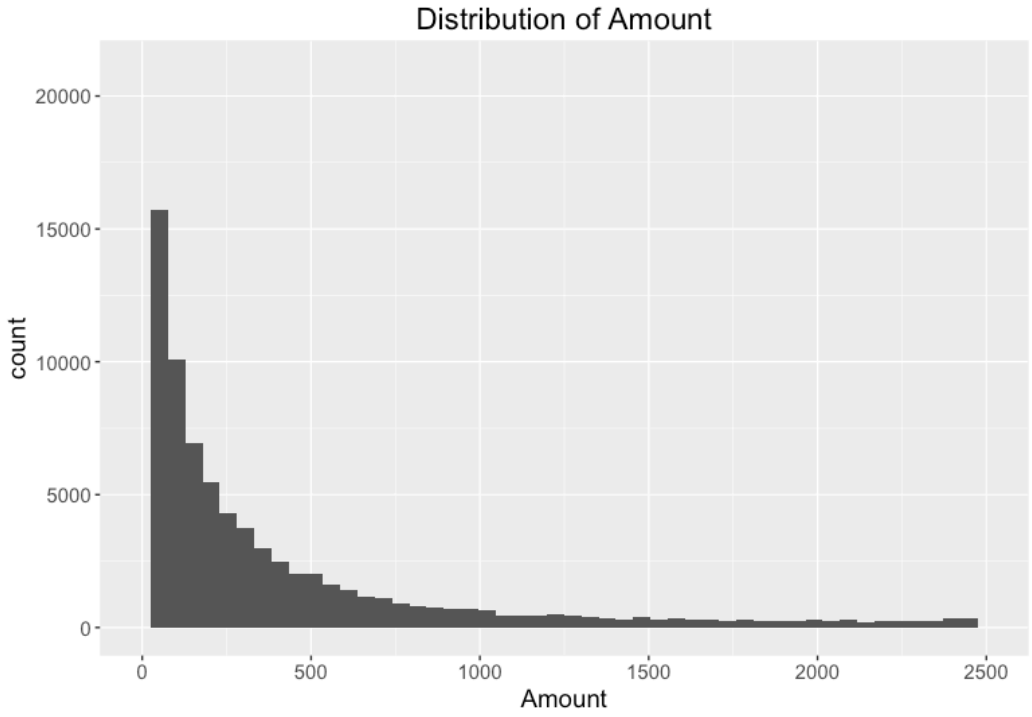
Transtype	Count
P	96353
A	181
D	173
Y	1

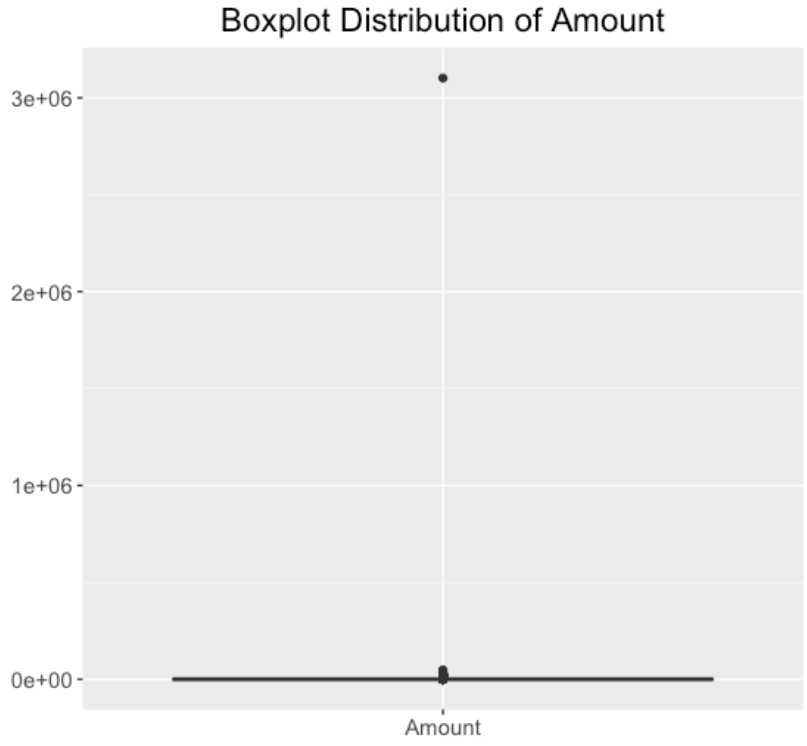
9. Amount

Amount is the only numeric variable in this dataset. Its summary statistics are shown as follows

Min	1st Quarter	Median	Mean	3rd Quarter	Max	Mode	SD
0.00	33.5	137.9	427.9	427.7	3102045.5	3.62	10008.5

The following two graphs show the distribution of ***Amount***. According to the distribution graphs, Amount is a right-skewed variable, with an obvious outlier higher than $3*10^6$.





10. Fraud

Fraud is a categorical variable with two unique values: 0 and 1, with 0 representing non-fraudulent records and 1 representing fraudulent record. It is a label that is manually added to the dataset for educational purpose.

Fraud	Count	Percentage
1	1014	1.05%
0	95694	98.95%

The following map shows the geographical distribution of fraudulent credit card transactions happened across the United States. The top 5 states with the most fraudulent transactions are Washington, Virginia, California, Pennsylvania and Oregon.