Finding Fraud in the Card Payment Data

A supervised learning process

Team 4 - Chong Li, Jie Chen, Raman Deep Singh, Xiaowen Zhang, Yu Dong

Table of Contents

1. Executive Summary

This report provides an analysis and evaluation of Card Payments Data for detecting fraud using supervised machine learning methods. The tools used include R, Python, and Tableau, and some of the algorithms used for analysis include Neural Network and Random Forest.

The original data set contains 95,007 records of card transactions with 10 variables of transaction details such as card number and date of transaction. The general process of analysis follows data cleaning and manipulation, building expert variables with different time windows, selecting important variables, applying fraud algorithm, calculating fraud score, identifying potential fraud, and evaluating results.

Two feature selection models were used - forward stepwise and filter feature selection based on mutual information. Six different models were tried using Python, and their performances were recorded. Among them, Neural Network model and Random Forest model worked best. Two sets of ensemble models were made by calculating average fraud score from Neural Network and Random Forest, and by picking the highest score between these two algorithms. Overall, the best model is Random Forest model with 9 variables selected using forward selection method.

Using the best model, Fraud Detection Rate is 99.7%, KS is 99, False Positive Rate is 66.82%, and ROI is $166,049 in the top 1% highest scored records. The report further finds that there are several expert variables that were strong predictors of fraud. Detailed examination of the important features selected from different algorithms indicates that fraudulent records typically have unusual geographical appearance.

2. Data Overview

Card payment data is a dataset containing 95,007 records of card transaction from 2010-01-01 to 2010-12-31. It includes information about card number, date, merchant's number, description, state, zip code, transaction type, and amount. Every record is also labeled as fraud or not. In total, there are 298 labeled fraudulent records.

10 variables in total – 1 numeric, 7 categorical, 1 text, 1 date
Numeric: amount
Categorical: recordnum, cardnum, merchnum, merch.state, merch.zip, transtype, and fraud
Text: merch.description
Date: date

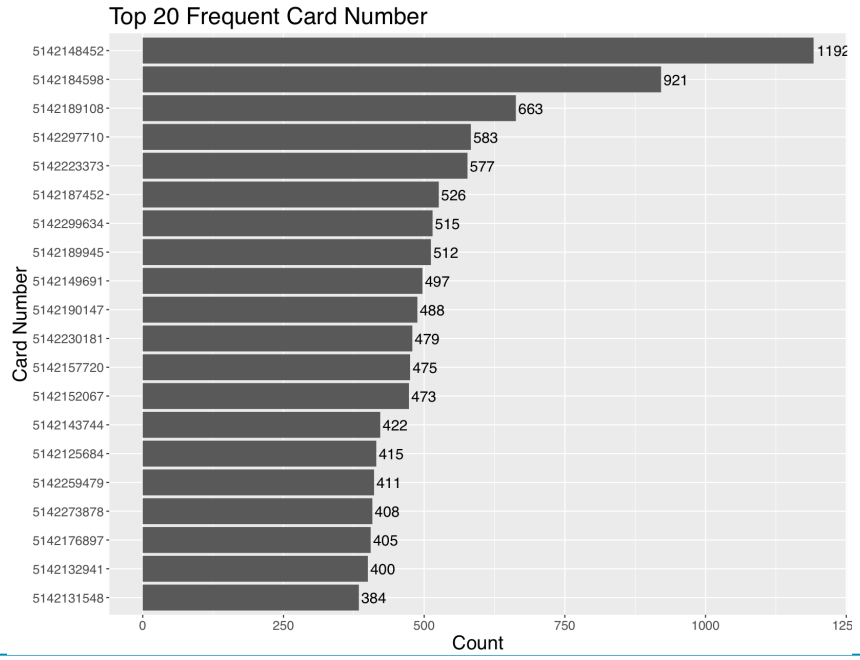
Following is the description of the variables we consider to be the most important. The complete Data Quality Report can be found in appendix.

2.1 Description of important variables

2.2.1 cardnum

cardnum is a categorical variable. It is the number of the card used for the payment.
Distribution
100% populated with 1,634 unique values. The following table shows the distribution of times of a card used, and the plot shows top 20 frequent card numbers.

Min	1st Qu	Median	Mean	3rd Qu	Max
1	10	30	58	72	1192



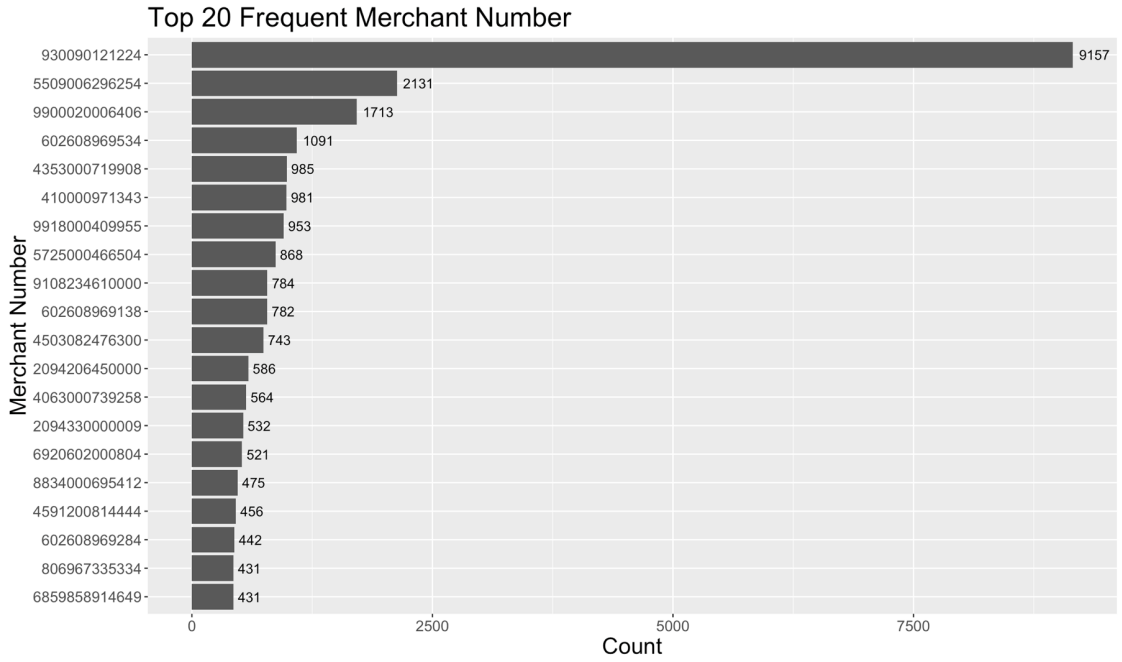
2.2.2 merchnum

merchnum is a categorical variable. It is the merchant number involved in the payment.

Distribution

96.6% populated with 13,089 unique values. There are 3,174 missing values, and 53 0's. The following table shows the distribution of valid merchant numbers, and the plot shows top 20 frequent merchant numbers.

Min	1st Qu	Median	Mean	3rd Qu	Max
1	1	1	7.013	3	9157



2.2.3 merch.state

merch.state is a categorical variable, indicating the abbreviations of states of the merchant.

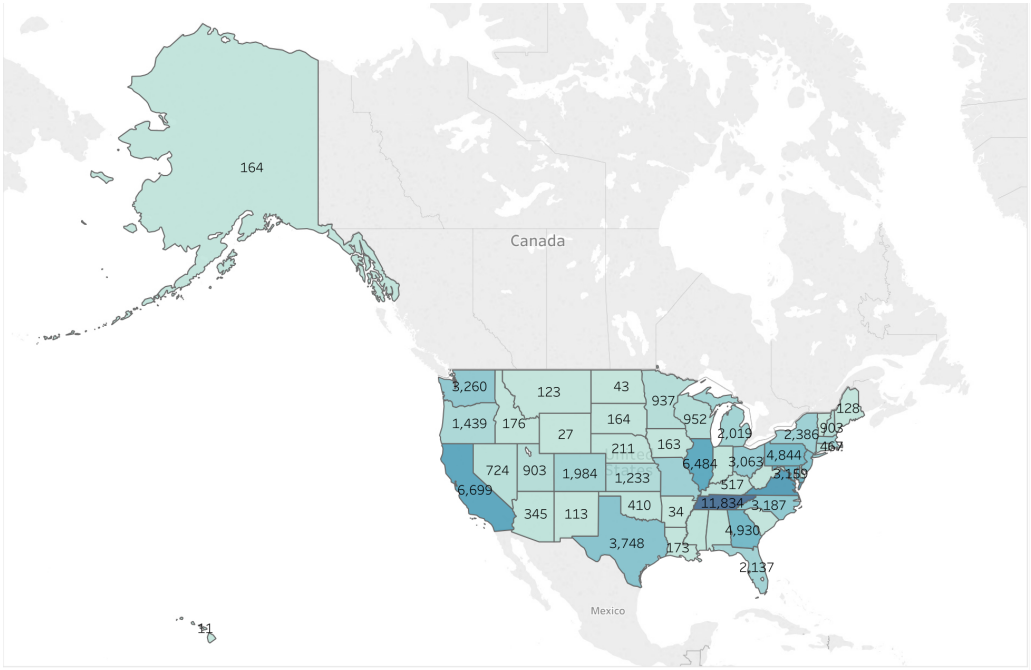
Distribution

98.93% populated with 60 unique values. 51 values stand for regions in the United States, 50 states and District of Columbia. 7 values stand for Canada. Besides that, we discovered are 1,016 missing values, and 1 invalid value US.

Distribution of states is showed in the following table. States in Canada are marked in blue. Invalid and missing values are marked in red. Regions in the United States are in black:

	AB	AK	AL	AR	AZ	BC	CA	CO	CT
1016	5	164	343	34	345	23	6699	1984	949
DC	DE	FL	GA	HI	IA	ID	IL	IN	KS
3159	70	2137	4930	11	163	176	6484	244	1233
KY	LA	MA	MB	MD	ME	MI	MN	MO	MS
517	173	2075	3	5344	128	2019	937	2415	244
MT	NC	ND	NE	NH	NJ	NM	NS	NV	NY
123	3187	43	211	903	3904	113	5	724	2386
OH	OK	ON	OR	PA	PQ	QC	RI	SC	SD
3063	410	137	1439	4844	14	4	467	153	164
TN	TX	US	UT	VA	VT	WA	WI	WV	WY
11834	3748	1	903	7698	85	3260	952	181	27

The following picture displays distribution by states in the United States. We can clearly see that TN has the largest number of records (11,834).



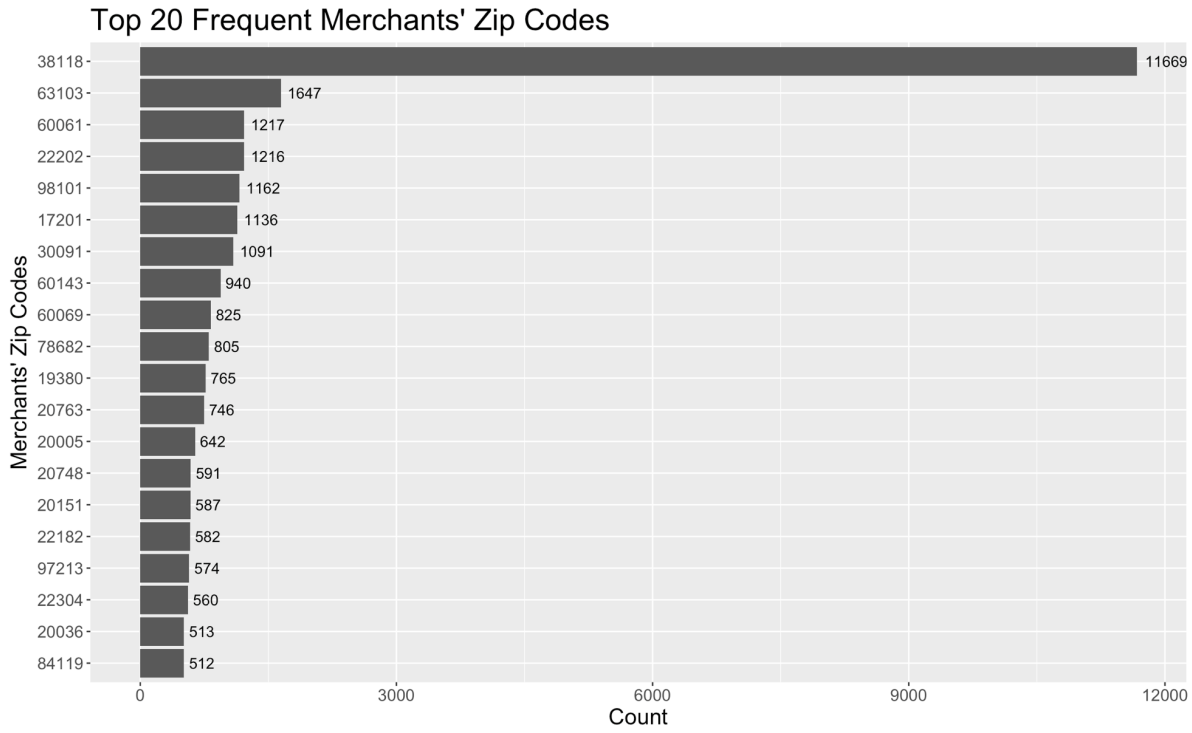
2.2.4 merch.zip

merch.zip is a categorical variable, indicating the zip code of the merchant.

Distribution

95.49% populated with 4584 unique values. 86.99% values have in valid 5-digit zip code. The following table shows the distribution of valid zip codes, and the plot shows top 20 zip codes. Although there is one zip code “38118” appears 10 times more than other values, it is in TN, which is the most frequent states; hence we do not treat it as a frivolous value.

Min	1st Qu	Median	Mean	3rd Qu	Max
1	1	3	20.71	10	11669



3. Handling Missing/Invalid Values

According to data quality checking, three variables -- merchnum, merch.state and merch.zip are not 100% populated, with missing values and 0s. We assigned values in these fields based on merch.description. We considered records with different merch.description as different merchants, and assumed that each merchant should have a unique merchant number, be in one state, and only have one zip code. Since all these three variables are categorical, we assigned unique values in these fields according to merch.description, and those values were designed to be quite different from other existing values, therefore easy to recognize.

Missing values and 0s in merchnum were replaced by strings from ‘M1’ to ‘M771’, missing values in merch.state were substituted by numbers from ‘1’ to ‘150’, and NAs in merch.zip were replaced by strings from ‘M1’ to ‘M644’.

4. Variable Construction

Since this analysis involves time, with limited data, we chose four different time windows, **3, 7, 14 and 28 days**. The rationale is to capture more (and maybe different types of) fraudulent records that might be detected in those time windows.

We then built four types of variables using R, totaling to 56 new variables:

1) **Type I variables** are intended to capture unusual amounts of transaction, both at the card level and the merchant level.

Example: **card_amount_to_avg_3** tells if a particular transaction amount is unusual compared to historical averages of the past 3 days.

2) **Type II variables** are intended to capture unusual transaction frequency during a set period of time, both at the card level and the merchant level.

Example: **card_frequency_3** describes the transaction frequency for each specific card number in the past 3 days

3) **Type III variables** are location related variables, which are intended to capture merchants with different zip codes and states in a set period of time.

Example: **zip_with_merchnum_3** tells how many zip codes are related with a particular merchant in the past 3 days.

4) **Type IV variables** are intended to catch card appearance pattern, either for a merchant or for a card holder.

Example: **merchnum_per_card_3** tells how many different merchants a card was used for transaction within the past 3 days.

Below are detailed descriptions of each of the variables.

4.1 Type I Variables:

card_amount_to_avg: ratio of transaction amount of a specific card number to its historical average amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific card number in a specific time window was replaced by a neutral value - 1 before building this set of variables.

merchant_amount_to_avg: ratio of transaction amount of a specific merchant number to its historical average amount by time window 3, 7, 14 and 28 days. To set a baseline, the first

value of of each specific merchant number in a specific time window was replaced by a neutral value - 1 before building this set of variables.

card_amount_to_max: ratio of transaction amount of a specific card number to its historical maximum amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific card number in a specific time window was replaced by a neutral value - 0 before building this set of variables.

merchant_amount_to_max: ratio of transaction amount of a specific merchant number to its historical maximum amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific merchant number in a specific time window was replaced by a neutral value - 0 before building this set of variables.

card_amount_to_median: ratio of transaction amount of a specific card number to historical median amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific card number in a specific time window was replaced by a neutral value - 1 before building this set of variables.

merchant_amount_to_median: ratio of transaction amount of a specific merchant number to its historical median amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific merchant number in a specific time window was replaced by a neutral value - 1 before building this set of variables.

card_amount_to_total: ratio of transaction amount of a specific card number to its historical total amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific card number in a specific time window was replaced by a neutral value - 0 before building this set of variables.

merchant_amount_to_total: ratio of transaction amount of a specific merchant number and historical total amount by time window 3, 7, 14 and 28 days. To set a baseline, the first value of of each specific merchant number in a specific time window was replaced by a neutral value - 0 before building this set of variables.

4.2 Type II Variables:

card_frequency: transaction frequency for each specific card number by time window 3, 7, 14 and 28 days.

merchant_frequency: transaction frequency for each specific merchant number by time window 3, 7, 14 and 28 days.

4.3 Type III Variables: Location

zip_with_merchnum: number of different zip codes related to a particular merchant in the time windows of 3, 7, 14 and 28 days.

state_with_merchnum: number of different states related to a particular merchant in the time windows of 3, 7, 14 and 28 days.

4.4 Type IV variables: Purchasing pattern

cardnum_per_merch: number of different cards associated with a particular merchant in time windows of 3, 7, 14 and 28 days.

merchnum_per_card: number of different merchants associated with a particular card in time windows of 3, 7, 14 and 28 days.

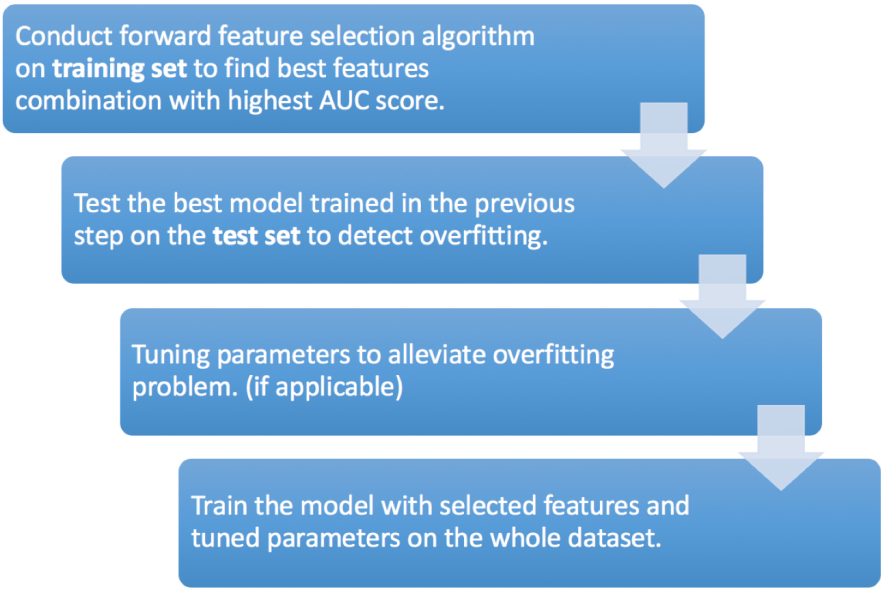
5. Supervised Fraud Algorithm

5.1 Training/Test Set Split

Before training all the models, we first extracted records after the first 28 days, Z-scaled all the features, randomly shuffled the observations, and then split the whole dataset into training and test set with the proportion of 2:1. As a result, we got these two sets with similar fraud rate:

Dataset	Number of Records	Number of Frauds	Fraud Rate
Training Set	59,080	197	0.333%
Test Set	29,541	98	0.332%
Total	88,621	295	0.333%

5.2 Forward Feature Selection



5.2.1 Forward Feature Selection Algorithm

We used a wrapper feature selection method called forward stepwise selection algorithm to select features during the modeling process. The algorithm we applied is described below.

Forward Stepwise Selection Algorithm
1. Initialization: (a) Let M_0 denote the null model, which contains no predictors. (b) Initialize maximum AUC score: $S_0 = 0$. (c) Initialize best features combination: F = [].
2. For k = 0, 1, 2, …, 55: (a) Consider all 56-k models that augment the predictors in M_k with one additional predictor. Calculate AUC score for each of them. (b) Choose the model with highest AUC score among these 56-k models, and call it M_{k+1}. (c) Compare the highest AUC score S found in (b) with S_0, if $S > S_0$, update S_0 with the value of S and update F with current feature selections, and enter the next loop step; if $S = S_0$, do nothing and enter the next loop step; if $S < S_0$, stop the loop.
3. Finalize: F is the set of features selected through this forward stepwise selection algorithm, and S_0 is the highest AUC score that ever got during the process.

This algorithm helped us to find the features combination that gives highest AUC score on the training set in a forward-selection manner. We used AUC scores as the comparison criterion, because AUC scores takes all classification cut-offs into consideration (i.e. considered all the cases that we predict probability >= p as fraud, where p could be any value between 0 and 1), reflecting the overall performance of the classifier in all fraud score quantiles.

5.2.2 Supervised Models

We have tried six supervised models combining the forward feature selection algorithm. The performance metrics of each model are shown below.

Models	AUC		Classification Accuracy*		True Positive Rate*	
	Training	Test	Training	Test	Training	Test
Gaussian Naïve Bayes	0.828	0.774	0.874	0.874	0.782	0.673
Logistic Regression	0.637	0.576	0.997	0.997	0.274	0.153
Decision Tree	0.997	0.503	1.000	0.992	0.995	0.010
Random Forest	0.815	0.719	0.999	0.998	0.629	0.439
Support Vector Machine	0.703	0.592	0.998	0.997	0.406	0.184
Neural Network	0.789	0.704	0.998	0.998	0.579	0.408

* Classification Accuracy and True Positive Rates are calculated when all the records with a probability >= 0.5 are predicted as frauds.

5.2.2.1 Gaussian Naive Bayes

Gaussian Naive Bayes Classifier is a simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

It was the most basic model we used, and was very fast to train. Accordingly, since its assumption of independence was not well satisfied, its performance was quite poor. Nine features were selected: zip_with_merchnum_7, card_amount_to_median_28, merchant_amount_to_median_28, state_with_merchnum_3, merchant_frequency_3, card_frequency_7, merchant_amount_to_avg_7, card_frequency_3, and merchant_amount_to_avg_3.

This model does not suffer from overfitting as much and has high AUC scores and True Positive Rates. However, its classification accuracy is only about 87%, which is unacceptable since there is only 0.3% frauds in the dataset. This low accuracy is caused by predicting a lot of non-frauds as frauds.

5.2.2.2 Logistic Regression

Logistic Regression model uses sigmoid function to estimate the probability, and predict probability of each record.

In our case, Logistic regression selected 45 features, including zip_with_merchnum_14, card_amount_to_median_7, card_amount_to_max_3, card_amount_to_avg_28, state_with_merchnum_7, merchnum_per_card_3, and etc.

Although it does not suffer a lot from overfitting, its True Positive Rate is relatively low, indicating many frauds are predicted with a low probability.

5.2.2.3 Decision Tree

Decision Tree partitions the feature spaces into multiple high-dimensional boxes, and give predictions according to the majority vote in each box.

A single decision tree is extremely prone to overfitting. As is shown in the summary table, although the tree selected only 2 features- merchant_amount_to_avg_28 and card_amount_to_avg_14, it achieved almost perfect performance on the training set by repetitively partitioning, and performed terrible on the test set.

5.2.2.4 Random Forest

Random Forest is a modified version of decision tree, which combines multiple small trees, and considers only small amount of features at each split. Therefore, it performs far better than decision tree on the overfitting problem.

It selected 9 features - zip_with_merchnum_14, merchant_amount_to_max_7, cardnum_per_merch_28, merchant_frequency_28, merchnum_per_card_28, card_frequency_3, card_amount_to_median_28, merchant_amount_to_max_28, and merchant_frequency_14. And the parameters after tuning are: 25 trees, max depth of each tree is 20, and minimal sample size on each terminal node is 5. It also generated good AUC score and classification accuracy. Although the True Positive Rate was not good enough on test data, better result could be achieved by lowering the prediction threshold, such as predicting all the observations with probability >= 0.3 as frauds.

5.2.2.5 Support Vector Machine

Support Vector Machine(SVM) tries to project observations to higher dimension to find a split boundary.

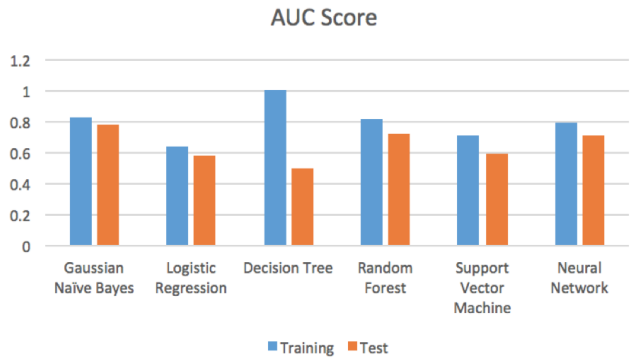
It selected 15 features - zip_with_merchnum_14, card_amount_to_median_7, card_frequency_7, merchnum_per_card_28, and card_amount_to_max_14, etc. Although it does not suffer from overfitting much, its true positive rates were not satisfactory. Since SVM does not predict probability but only gives binary prediction, this performance indicates that it missed more than 50% of the frauds forever.

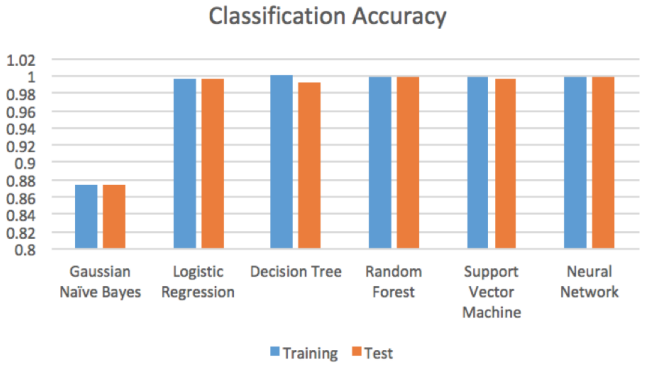
5.2.2.6 Neural Network

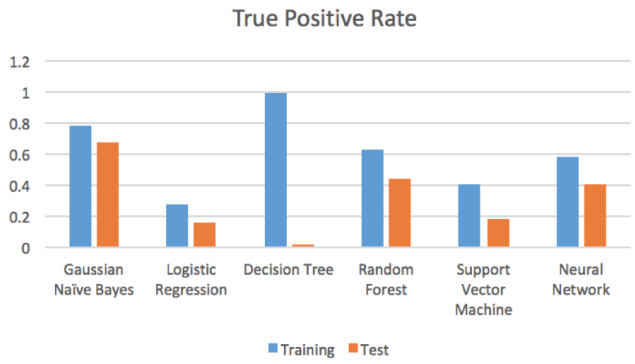
To build a neural network classifier, we used “Multi-Layer Perceptron Classifier” in Scikit-Learn package in Python. After tuning, the parameters we used are “relu” function as the activation function, L2 penalty as 0.0001, batch size is 200, learning rate is 0.001, and 100 nodes in each hidden layer. And 3 layers are built with 13 features selected: zip_with_merchnum_14, card_amount_to_median_3, merchant_amount_to_avg_28, card_amount_to_median_14, merchant_frequency_3, card_frequency_3, merchnum_per_card_7, cardnum_per_merch_14, merchnum_per_card_3, cardnum_per_merch_28, merchant_frequency_28, merchant_amount_to_median_3, card_amount_to_max_14.

The performance of Multi-Layer Perceptron Classifier is similar to random forest, and it improved on the overfitting issue.

5.2.3 Summary







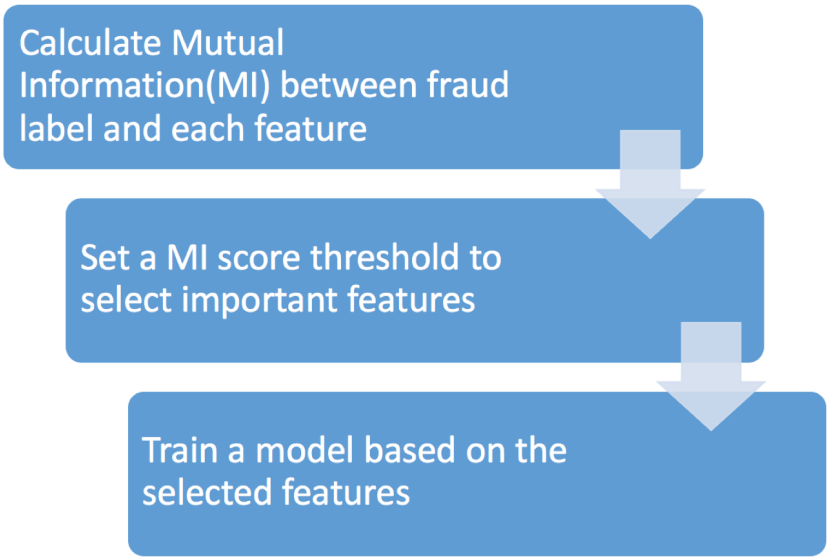
According to the above analysis and comparison, our conclusions are: Decision Tree model suffers a lot from overfitting; Gaussian Naive Bayes model has poor classification accuracy, predicting lots of non-frauds as fraud; Logistic Regression and Support Vector Machine model do not overfit much, but both cannot capture enough frauds; Random Forest and Neural Network models are the best performing models among all the six models.

	Gaussian Naïve Bayes	Logistic Regression	Random Forest	Support Vector Machine	Neural Network
1	zip_with_merchnum_7	zip_with_merchnum_14	zip_with_merchnum_14	zip_with_merchnum_14	zip_with_merchnum_14
2	card_amount_to_median_28	card_amount_to_median_7	merchant_amount_to_max_7	card_amount_to_median_7	card_amount_to_median_3
3	merchant_amount_to_median_28	card_amount_to_max_3	cardnum_per_merch_28	card_frequency_7	merchant_amount_to_avg_28
4	state_with_merchnum_3	card_amount_to_avg_28	merchant_frequency_28	merchnum_per_card_28	card_amount_to_median_14
5	merchant_frequency_3	state_with_merchnum_7	merchnum_per_card_28	card_amount_to_max_14	merchant_frequency_3
6	card_frequency_7	merchnum_per_card_3	card_frequency_3	merchant_amount_to_avg_14	card_frequency_3
7	merchant_amount_to_avg_7	merchnum_per_card_28	card_amount_to_median_28	merchant_amount_to_median_3	merchnum_per_card_7
8	card_frequency_3	card_amount_to_max_7	merchant_amount_to_max_28	merchant_amount_to_median_28	cardnum_per_merch_14
9	merchant_amount_to_avg_3	card_amount_to_avg_3	merchant_frequency_14	merchant_amount_to_median_14	merchnum_per_card_3
10		merchant_amount_to_avg_3		merchant_amount_to_avg_28	cardnum_per_merch_28

Type I variable
Type II variable
Type III variable
Type IV variable

When we look at the top 10 selected variables (the 10 variables that are first selected) for each model, a noticeable pattern is that all the models selected type III variable - the number of different zipcode related to a particular merchant number - at the very beginning, indicating that geographical pattern is a very useful feature to detect frauds. Besides, all the models selected several type I variable, since unusual transaction patterns can also play a key role in detecting frauds. Some of the Type II and Type IV variables were also selected as top 10 variables, but generally speaking, they are not as important as the other two types of variables.

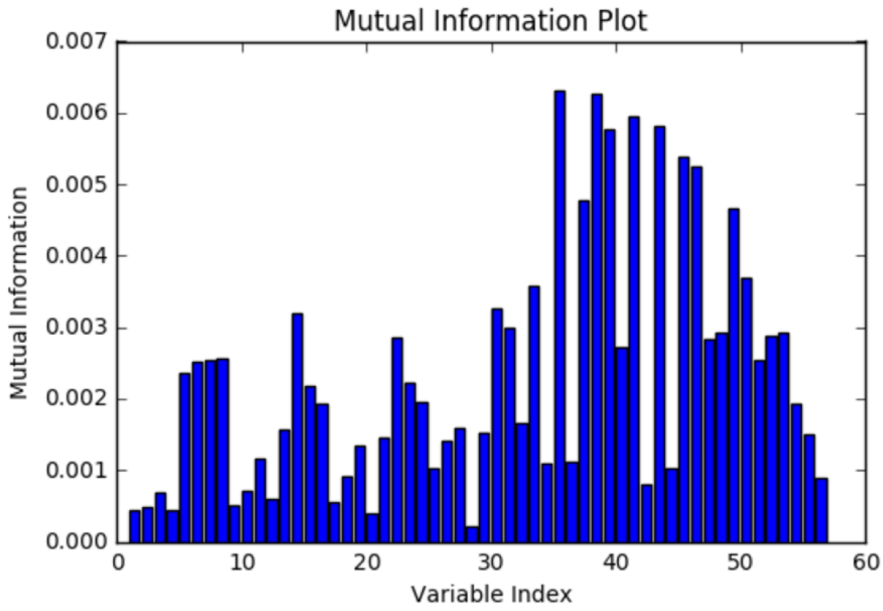
5.3 Feature Selection Based on Mutual Information



We have also tried to select features based on Mutual Information score, and trained models only on the selected important features.

Mutual Information criterion was applied to select significant features in fraud analysis. This Mutual Information (MI) method measured the mutual dependence between each variable we built and the dependent variable 'fraud'. More specifically, it quantified the 'amount of information' obtained from each variable through the variable 'fraud' and gave a score. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency.

Before calculating the mutual information, we first z-scaled all the features to get rid of the influence of different units. Then we used the "**mutual_info_classif**" function in the Scikit-Learn package in Python to calculate the mutual information score. The mutual information calculated are shown in the graph below.



As we can see from the graph, about one-fourth of the mutual information scores are higher than 0.003. To be precise, 13 out of 56 mutual information are higher than 0.003, 9 out of 56 mutual information are higher than 0.004, and 7 out of 58 mutual information are higher than 0.005.

To avoid overfitting by involving too many variables, we decided to set the threshold at 0.004. Therefore, the 9 features we selected were:

Expert Variable	Mutual Information
zip_with_merchnum_7	0.0063
cardnum_per_merch_7	0.0063
zip_with_merchnum_14	0.0060
zip_with_merchnum_28	0.0058
merchnum_per_card_3	0.0058
cardnum_per_merch_14	0.0054
cardnum_per_merch_28	0.0052
cardnum_per_merch_3	0.0048
merchant_frequency_3	0.0047

The performance of models trained with these features are shown as below. We can see the performance is not better than the features selected through forward selection method. This could be due to the fact that, when we select features based on mutual information, we only

considered the dependency between each feature and fraud, but did not take into consideration the effect of combining features. For example, we selected three type III variables - zip_with_merchnum_7, zip_with_merchnum_14, andzip_with_merchnum_28, but the information they provided could be highly overlapped. Besides, some features could work much better when they are used together.

5.4 Conclusion of the Algorithms

To conclude, we have tried two feature selection methods: wrapped forward stepwise feature selection and filter feature selection method based on mutual information. Forward stepwise feature selection gives us better classification results than feature selection based on mutual information. To strike a balance between prediction accuracy and overfitting, we think the best models are Random Forest Model and Neural Network Model with features selected by the forward feature selection.

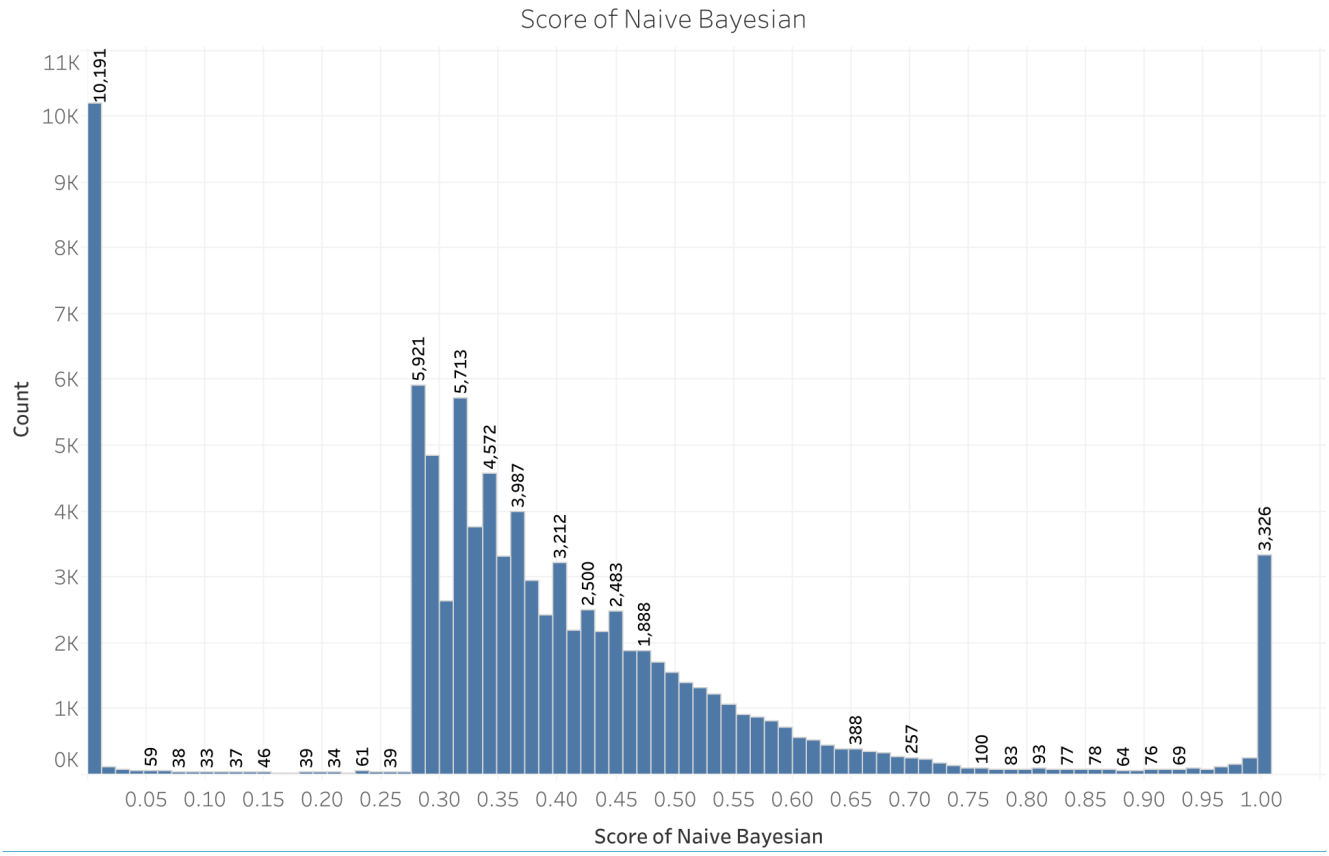
6. Results

As analyzed above, we found that the models built using forward stepwise feature selection have better performance than those built using features selected based on mutual information.In the following part, we will go through the fraud detection results from Gaussian Naive Bayes Model, Logistic Regression Model, Random Forest Model, Support Vector Machine Model, and Neural Network Model with forward stepwise feature selection method. Here we ignore Decision Tree Model, since it overfitted too much. After that, we will also show the result of ensemble modeling methods, which calculate fraud score by averaging the scores of Neural Network and Random Forest or pick the maximum score between Neural Network and Random Forest scores.

6.1 Results of 5 Individual Models

Let us start with the results of 5 different models that we built:

1) **Gaussian Naive Bayes:** Gaussian Naïve Bayes model did not work well and did not produce results which are good enough to use in the future.
The below diagram shows the histogram of fraud scores from Naïve Bayesian. The fraud scores are scattered from 0 to 1 and doesn't look like a normal fraud distribution.



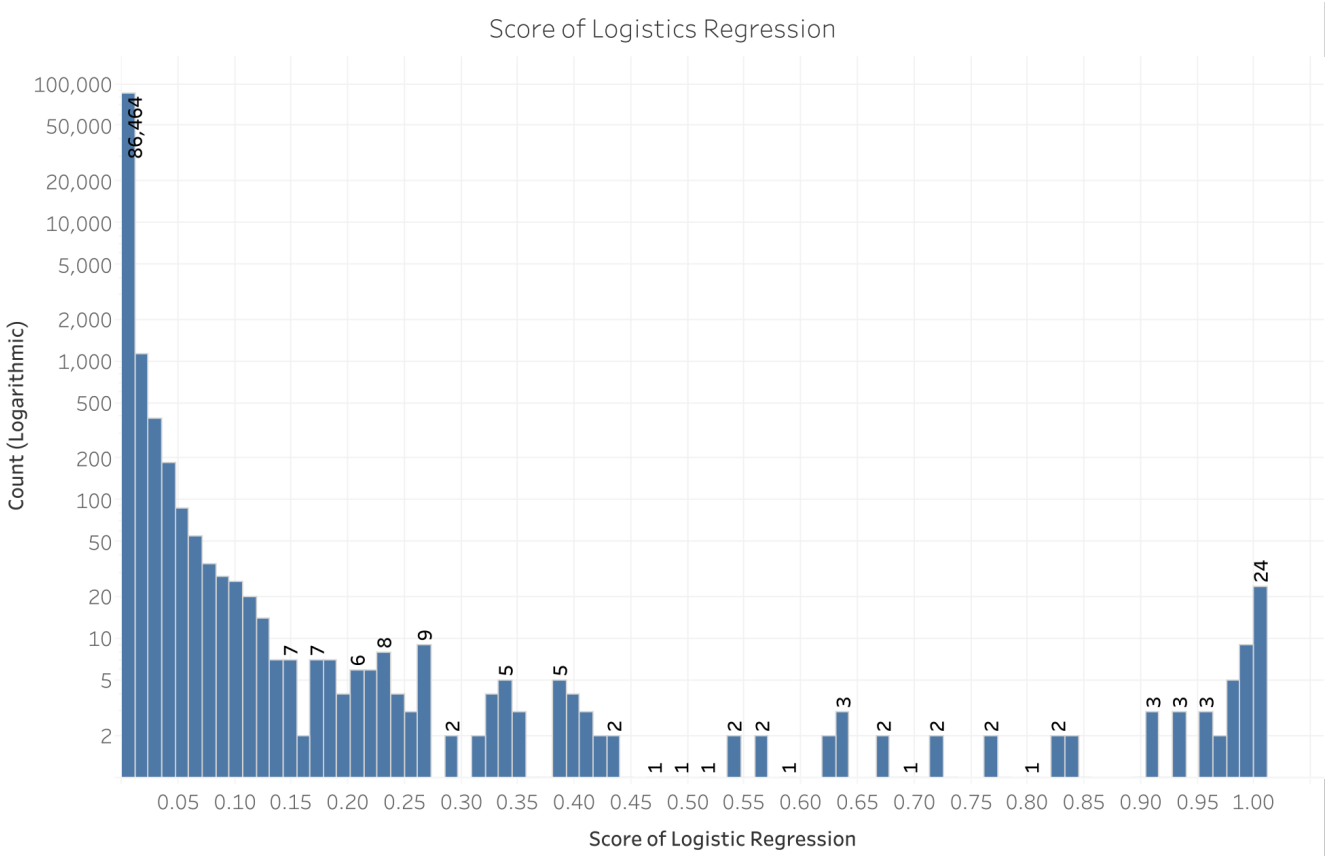
In terms of Fraud Detection Rate and ROI, the results are summarized below:

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	718	168	0.81	56.95	718	168	0.8	56.9	56.1	4.3	81.04
0.02	886	869	17	0.98	5.76	1,587	185	1.8	62.7	60.9	8.6	89.56
0.03	886	871	15	0.98	5.08	2,458	200	2.8	67.8	65.0	12.3	92.48
0.04	886	866	20	0.98	6.78	3,324	220	3.8	74.6	70.8	15.1	93.79
0.05	887	882	5	1.00	1.69	4,206	225	4.7	76.3	71.5	18.7	94.92
0.06	886	880	6	0.99	2.03	5,086	231	5.7	78.3	72.6	22.0	95.66
0.07	886	883	3	1.00	1.02	5,969	234	6.7	79.3	72.6	25.5	96.23
0.08	886	884	2	1.00	0.68	6,853	236	7.7	80.0	72.3	29.0	96.67

Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$100,800	$7,180	$89,189
0.02	$111,000	$15,870	$90,699
0.03	$120,000	$24,580	$90,989
0.04	$132,000	$33,240	$94,329
0.05	$135,000	$42,060	$88,509
0.06	$138,600	$50,860	$83,309
0.07	$140,400	$59,690	$76,279
0.08	$141,600	$68,530	$68,639

The plots below demonstrate the business values of Naive Bayesian Model. We zoom in to 0% to 0.1% of the plot on the left-hand side to get the plot on the right-hand side. We can see the maximum ROI is only about $50,000, and it soon decreased to 0 after including more bins.

2) **Logistic Regression:** For this case, Logistic Regression did not work well and did not produce results which are good enough to use in the future.
The below diagram shows the histogram of fraud score of Logistic Regression. As can be seen the fraud scores are scattered from 0 to 1 and doesn't look like a normal fraud distribution. **We have used logarithmic scale on y-axis for clarity.**



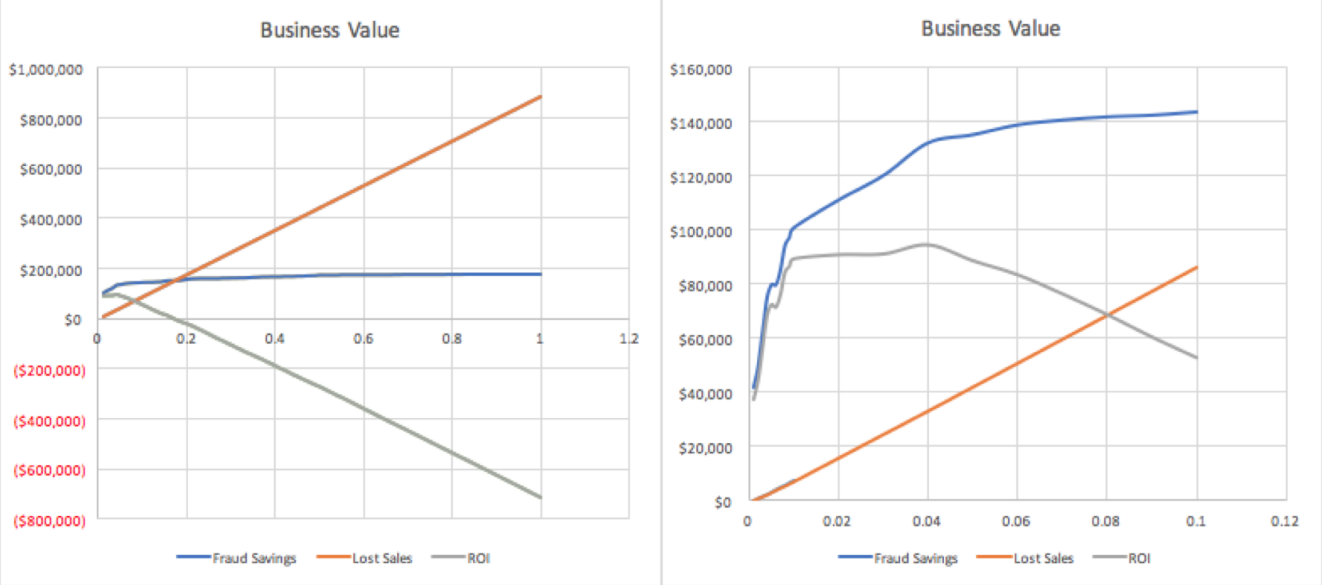
The trend of count of LR for LR (bin). The marks are labeled by count of LR.

In terms of Fraud Detection Rate and ROI, the results are summarized below:

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	718	168	0.81	56.95	718	168	0.8	56.9	56.1	4.3	81.04
0.02	886	869	17	0.98	5.76	1,587	185	1.8	62.7	60.9	8.6	89.56
0.03	886	871	15	0.98	5.08	2,458	200	2.8	67.8	65.0	12.3	92.48
0.04	886	866	20	0.98	6.78	3,324	220	3.8	74.6	70.8	15.1	93.79
0.05	887	882	5	1.00	1.69	4,206	225	4.7	76.3	71.5	18.7	94.92
0.06	886	880	6	0.99	2.03	5,086	231	5.7	78.3	72.6	22.0	95.66
0.07	886	883	3	1.00	1.02	5,969	234	6.7	79.3	72.6	25.5	96.23
0.08	886	884	2	1.00	0.68	6,853	236	7.7	80.0	72.3	29.0	96.67

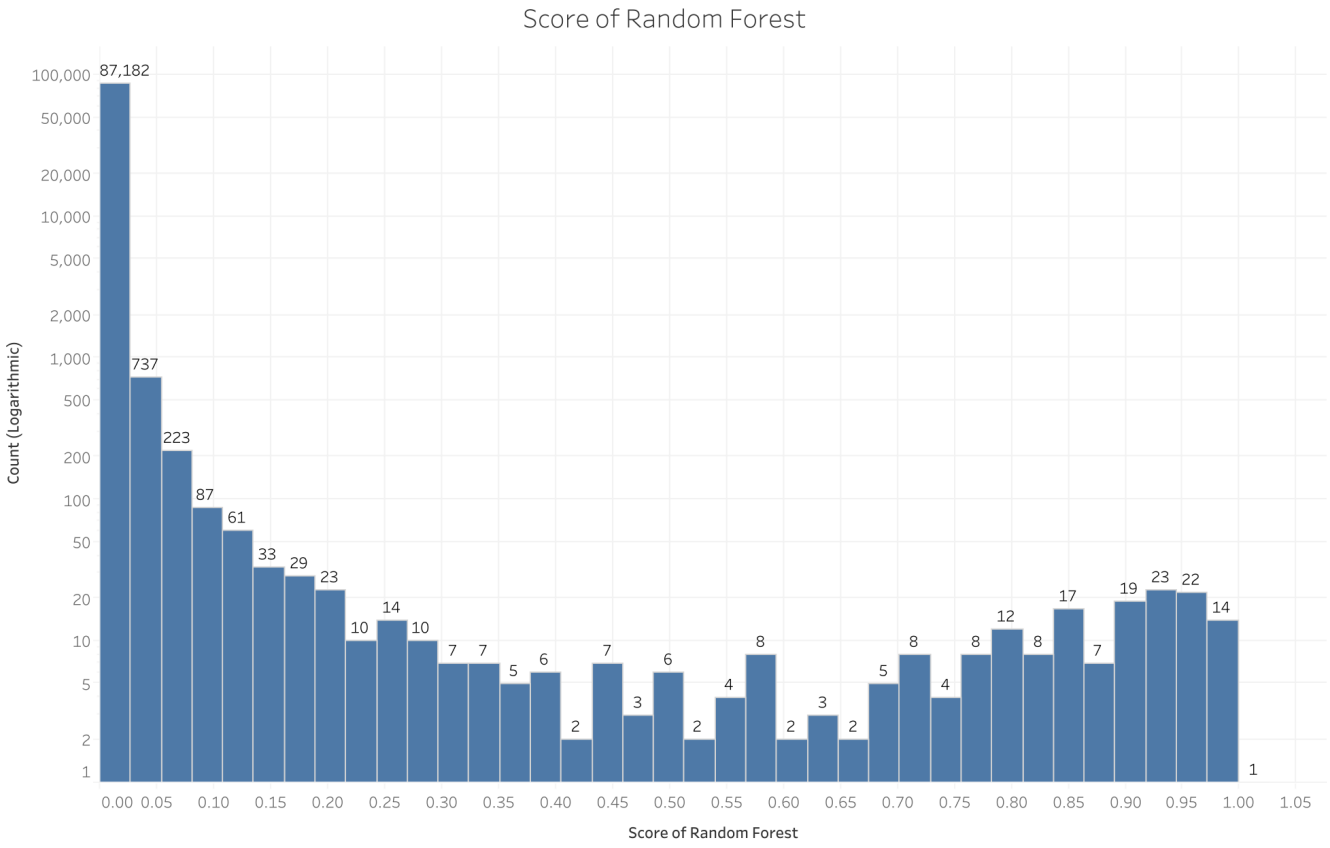
Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$100,800	$7,180	$89,189
0.02	$111,000	$15,870	$90,699
0.03	$120,000	$24,580	$90,989
0.04	$132,000	$33,240	$94,329
0.05	$135,000	$42,060	$88,509
0.06	$138,600	$50,860	$83,309
0.07	$140,400	$59,690	$76,279
0.08	$141,600	$68,530	$68,639

The plots below demonstrate the business values of Logistic Regression Model. We zoom in the plot on the left-hand side to 0% to 10% to get the plot on the right-hand side. We can see



that its performance is better than Gaussian Naive Bayes model, but still with a maximum ROI below $100,000.

3) **Random Forest:** For this case, Random Forest outperformed the other algorithms that we tried.
The below diagram shows the histogram of fraud scores from Random Forest. The fraud scores are scattered from 0 to 1 and there is a good amount of fraud scores on right i.e. records which are flagged as fraud. **We used logarithmic scale on y-axis for clarity.**



The trend of count of RF for RF (bin). The marks are labeled by count of RF.

In terms of Fraud Detection Rate and ROI, the results are summarized below. Its maximum ROI is $166,049 at the first 1% population bin.

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	592	294	0.67	99.66	592	294	0.7	99.7	99.0	2.0	66.82
0.02	886	886	0	1.00	0.00	1,478	294	1.7	99.7	98.0	5.0	83.41
0.03	886	886	0	1.00	0.00	2,364	294	2.7	99.7	97.0	8.0	88.94
0.04	886	886	0	1.00	0.00	3,250	294	3.7	99.7	96.0	11.1	91.70
0.05	887	887	0	1.00	0.00	4,137	294	4.7	99.7	95.0	14.1	93.36
0.06	886	886	0	1.00	0.00	5,023	294	5.7	99.7	94.0	17.1	94.47
0.07	886	886	0	1.00	0.00	5,909	294	6.7	99.7	93.0	20.1	95.26
0.08	886	886	0	1.00	0.00	6,795	294	7.7	99.7	92.0	23.1	95.85
0.09	886	885	1	1.00	0.34	7,680	295	8.7	100.0	91.3	26.0	96.30

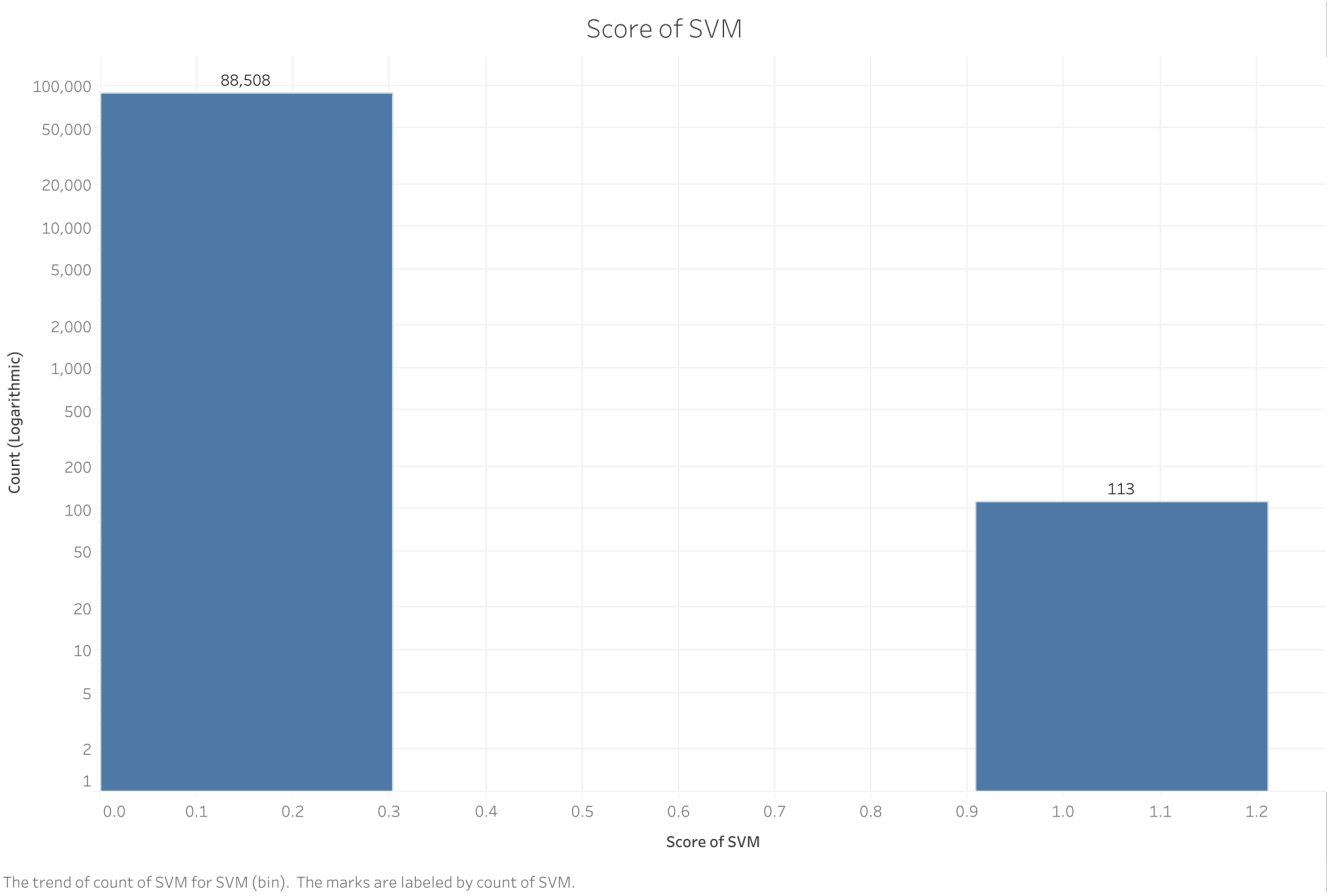
Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$176,400	$5,920	$166,049
0.02	$176,400	$14,780	$157,189
0.03	$176,400	$23,640	$148,329
0.04	$176,400	$32,500	$139,469
0.05	$176,400	$41,370	$130,599
0.06	$176,400	$50,230	$121,739
0.07	$176,400	$59,090	$112,879
0.08	$176,400	$67,950	$104,019
0.09	$177,000	$76,800	$95,769

The plot below demonstrates the business values of Random Forest Model. When we zoom in to the range between 0% to 1%, we find that the maximum of ROI is $168,409, achieved when population bin is 0.5%.

		Bin Statistics				Cumulative Statistics										
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	False Pos. Ratio*	False Pos. Rate**(%)	Population Bin	Fraud Savings	Lost Sales	ROI
0.001	88	0	88	0.00	29.83	-	88	0.0	29.8	29.8	0.0	-	0.001	$52,800	$0	$48,369
0.002	89	0	89	0.00	30.17	-	177	0.0	60.0	60.0	0.0	-	0.002	$106,200	$0	$101,769
0.003	88	18	70	0.02	23.73	18	247	0.0	83.7	83.7	0.1	6.79	0.003	$148,200	$180	$143,589
0.004	89	54	35	0.06	11.86	72	282	0.1	95.6	95.5	0.3	20.34	0.004	$169,200	$720	$164,049
0.005	89	80	9	0.09	3.05	152	291	0.2	98.6	98.5	0.5	34.31	0.005	$174,600	$1,520	$168,649
0.006	88	87	1	0.10	0.34	239	292	0.3	99.0	98.7	0.8	45.01	0.006	$175,200	$2,390	$168,379
0.007	89	88	1	0.10	0.34	327	293	0.4	99.3	99.0	1.1	52.74	0.007	$175,800	$3,270	$168,099
0.008	88	88	0	0.10	0.00	415	293	0.5	99.3	98.9	1.4	58.62	0.008	$175,800	$4,150	$167,219
0.009	89	88	1	0.10	0.34	503	294	0.6	99.7	99.1	1.7	63.11	0.009	$176,400	$5,030	$166,939
0.01	89	89	0	0.10	0.00	592	294	0.7	99.7	99.0	2.0	66.82	0.01	$176,400	$5,920	$166,049
0.02	886	886	0	1.00	0.00	1,478	294	1.7	99.7	98.0	5.0	83.41	0.02	$176,400	$14,780	$157,189
0.03	886	886	0	1.00	0.00	2,364	294	2.7	99.7	97.0	8.0	88.94	0.03	$176,400	$23,640	$148,329

4) **SVM:** Support Vector Machine didn't work well and did not produce results which are good enough to use in the future.

The below diagram shows the histogram of fraud score of Support Vector Machine. The fraud scores are only 0's and 1's, and this is because Support Vector Machine model does not predict probability, but instead predicts binary classification. It flagged only 113 records as likely fraud and hence did not even find half of the total fraudulent records. **We used logarithmic scale on y-axis for clarity.**

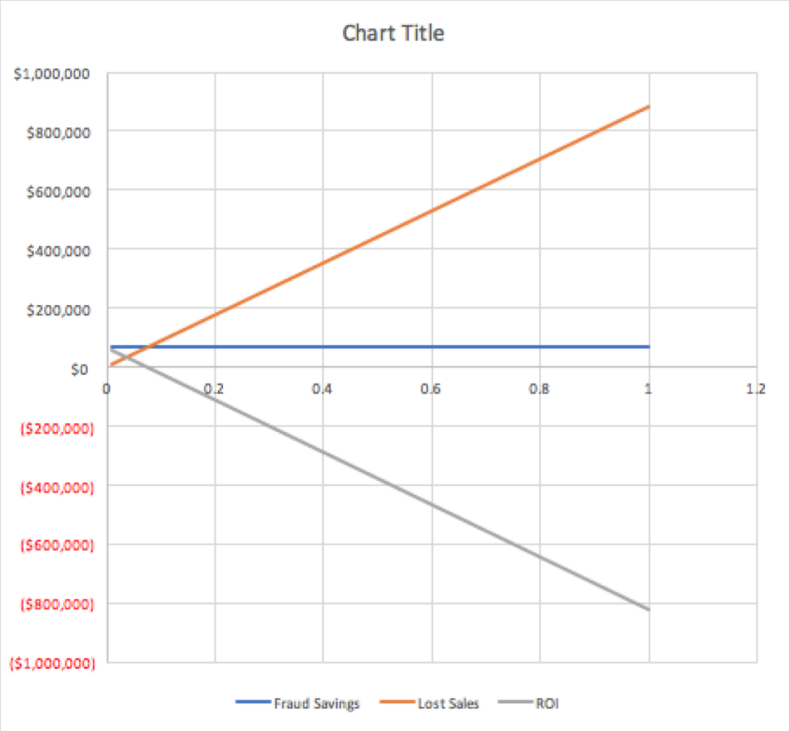


In terms of Fraud Detection Rate and ROI, the results are summarized below. Its maximum ROI is only $54,419.

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	775	111	0.87	37.63	775	111	0.9	37.6	36.8	7.0	87.47
0.02	886	885	1	1.00	0.34	1,660	112	1.9	38.0	36.1	14.8	93.68
0.03	886	886	0	1.00	0.00	2,546	112	2.9	38.0	35.1	22.7	95.79
0.04	886	885	1	1.00	0.34	3,431	113	3.9	38.3	34.4	30.4	96.81
0.05	887	887	0	1.00	0.00	4,318	113	4.9	38.3	33.4	38.2	97.45

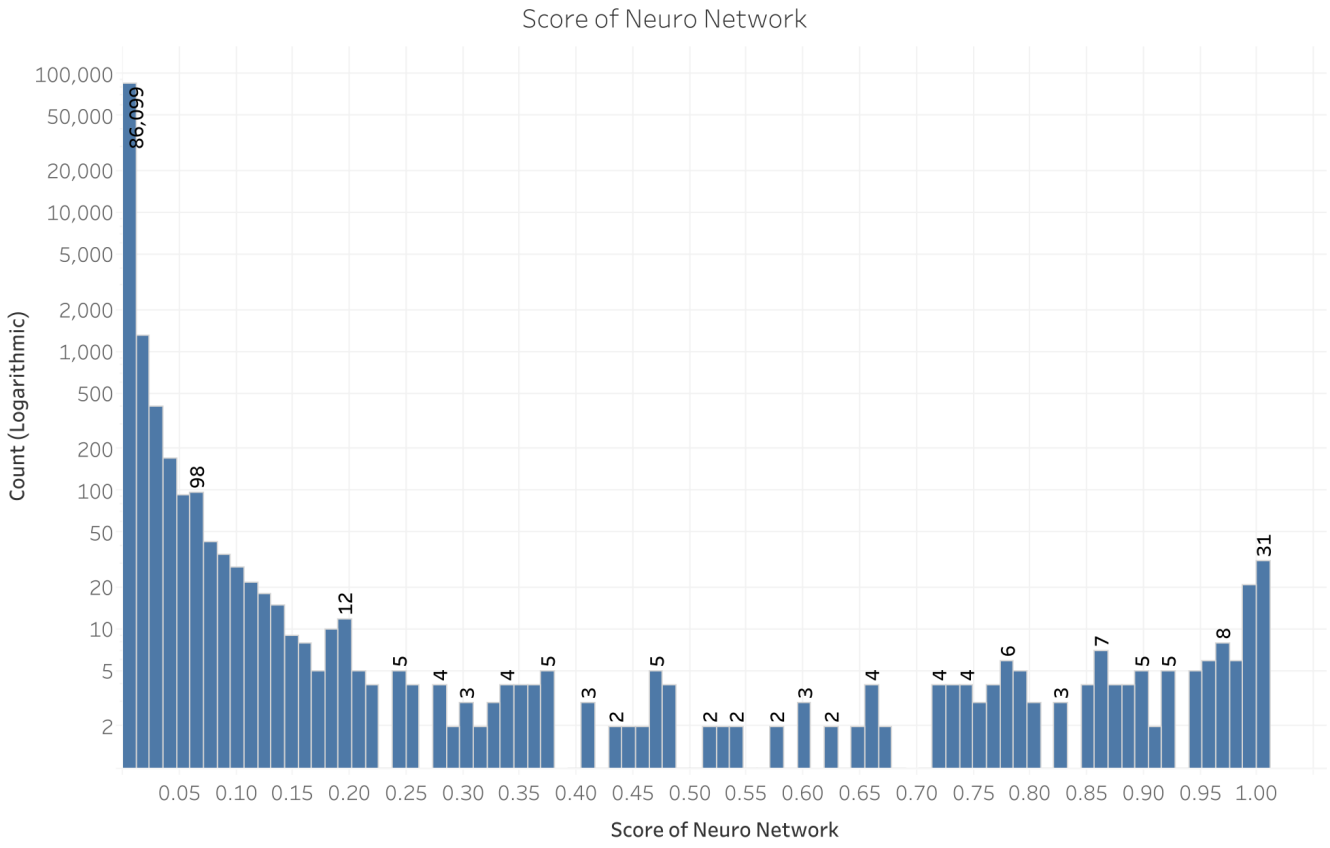
Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$66,600	$7,750	$54,419
0.02	$67,200	$16,600	$46,169
0.03	$67,200	$25,460	$37,309
0.04	$67,800	$34,310	$29,059
0.05	$67,800	$43,180	$20,189

The plot below demonstrates the business values of SVM Model.



5) **Neural Network:** For this case, Neural Network performed much better than many other algorithms that we tried.

The below diagram shows the histogram of fraud scores from Neural Network. The fraud scores are scattered from 0 to 1 and there is a good amount of fraud scores on right i.e. records which are flagged as fraud. **We used logarithmic scale on y-axis for clarity.**



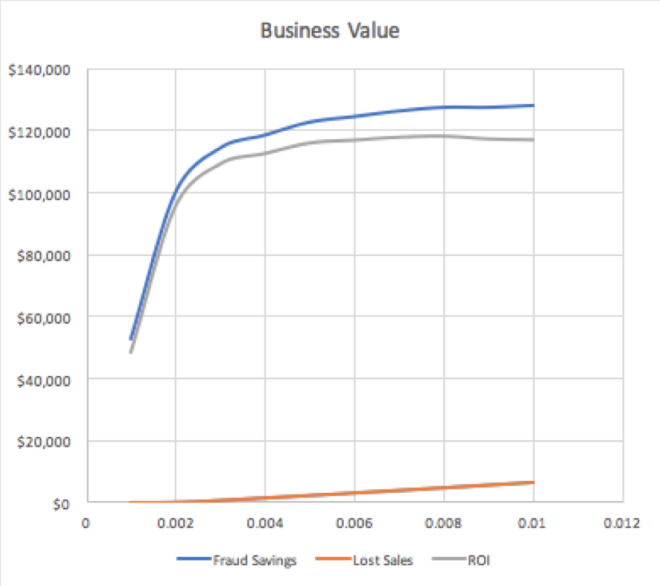
The trend of count of NN for NN (bin). The marks are labeled by count of NN.

In terms of Fraud Detection Rate and ROI, the results are summarized below. When regarding all the records in the 1% population bin as frauds, it captured 72.5% of the frauds, and generated a ROI of $117,249.

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	672	214	0.76	72.54	672	214	0.8	72.5	71.8	3.1	75.85
0.02	886	878	8	0.99	2.71	1,550	222	1.7	75.3	73.5	7.0	87.47
0.03	886	877	9	0.99	3.05	2,427	231	2.7	78.3	75.6	10.5	91.31
0.04	886	878	8	0.99	2.71	3,305	239	3.7	81.0	77.3	13.8	93.26
0.05	887	881	6	0.99	2.03	4,186	245	4.7	83.1	78.3	17.1	94.47
0.06	886	885	1	1.00	0.34	5,071	246	5.7	83.4	77.7	20.6	95.37
0.07	886	884	2	1.00	0.68	5,955	248	6.7	84.1	77.3	24.0	96.00
0.08	886	879	7	0.99	2.37	6,834	255	7.7	86.4	78.7	26.8	96.40
0.09	886	885	1	1.00	0.34	7,719	256	8.7	86.8	78.1	30.2	96.79

Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$128,400	$6,720	$117,249
0.02	$133,200	$15,500	$113,269
0.03	$138,600	$24,270	$109,899
0.04	$143,400	$33,050	$105,919
0.05	$147,000	$41,860	$100,709
0.06	$147,600	$50,710	$92,459
0.07	$148,800	$59,550	$84,819
0.08	$153,000	$68,340	$80,229
0.09	$153,600	$77,190	$71,979

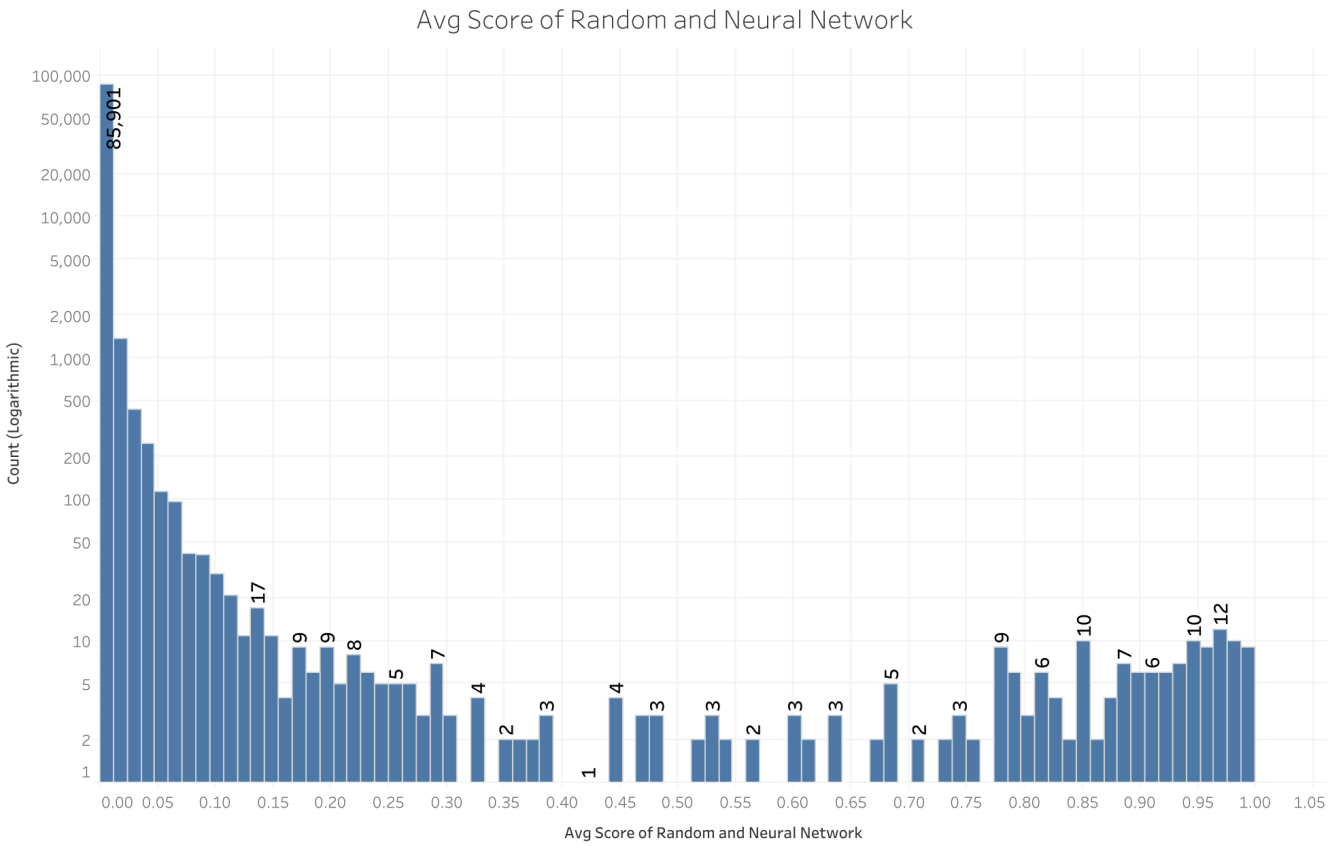
The plots below demonstrate the business values of Neural Network Model. We zoom in the plot on the left-hand side to 0% to 1% population bins to get the plot on the right-hand side. We can see that the maximum ROI is achieved when treating all the records in the 0.8% population bins as frauds, and the corresponding ROI is $118,419.



6.2 Results of Ensemble Models

After covering 5 different techniques that we used, we wanted to try the ensemble modeling approach. For this purpose, we selected two of our best models - Neural Network and Random Forest.

1) **Average of Neural Network Score and Random Forest Score:** In this approach we used the fraud scores from both Neural Network and Random Forest and then took an average of those scores. The newly obtained average scores are then used to find fraudulent records. The below diagram shows the histogram of fraud scores from this technique. The fraud scores are scattered from 0 to 1 and there is a good amount of fraud scores on right i.e. records which are flagged as fraud. **We used logarithmic scale on y-axis for clarity.**



The trend of count of Avg(RF+NN) for Avg(RF+NN) (bin). The marks are labeled by count of Avg(RF+NN).

In terms of Fraud Detection Rate and ROI, the results are summarized below:

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Population Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	592	294	0.67	99.66	592	294	0.7	99.7	99.0	2.0	66.82
0.02	886	886	0	1.00	0.00	1,478	294	1.7	99.7	98.0	5.0	83.41
0.03	886	886	0	1.00	0.00	2,364	294	2.7	99.7	97.0	8.0	88.94
0.04	886	886	0	1.00	0.00	3,250	294	3.7	99.7	96.0	11.1	91.70
0.05	887	887	0	1.00	0.00	4,137	294	4.7	99.7	95.0	14.1	93.36

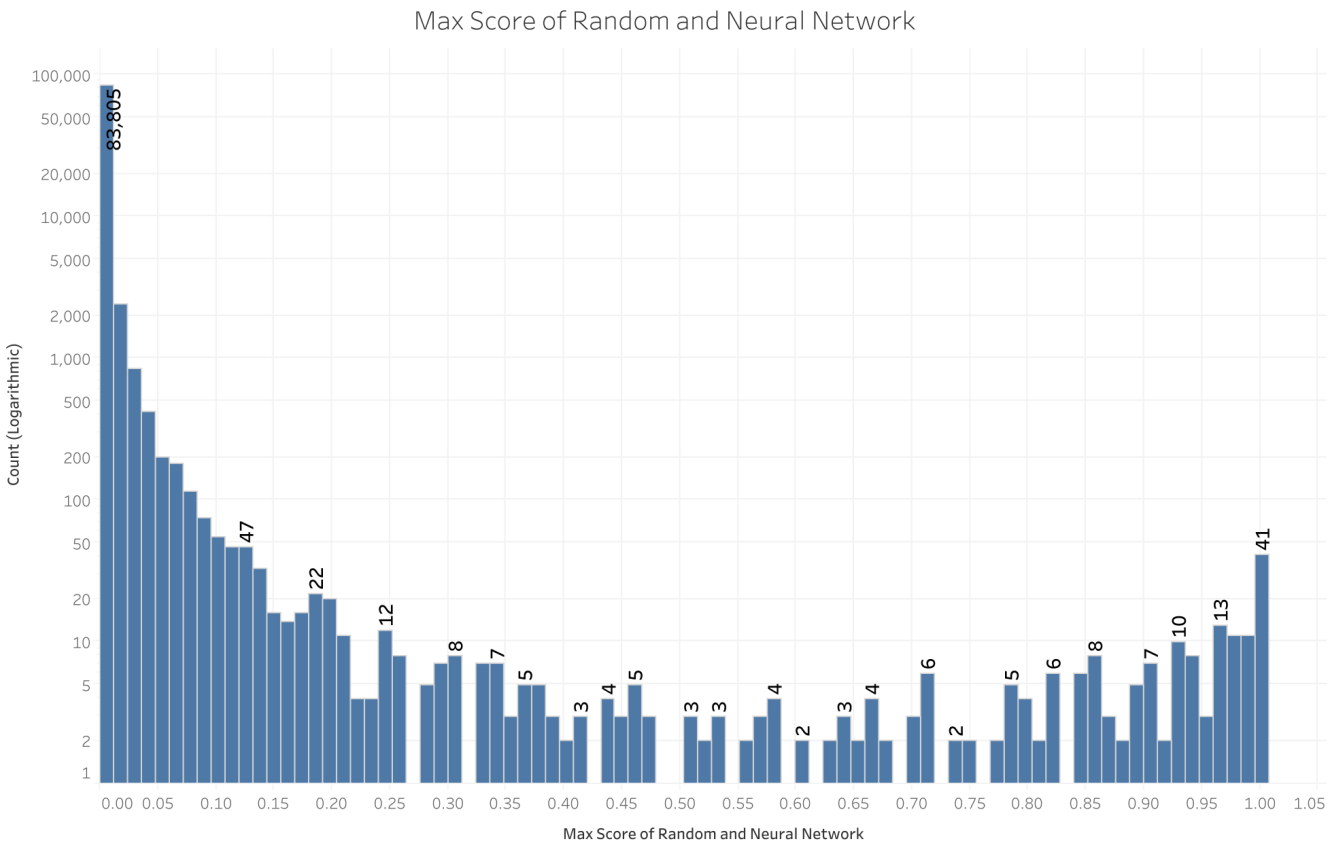
Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$176,400	$5,920	$166,049
0.02	$176,400	$14,780	$157,189
0.03	$176,400	$23,640	$148,329
0.04	$176,400	$32,500	$139,469
0.05	$176,400	$41,370	$130,599

The plot below demonstrates the business values of Average Model.

This result is almost the same as only using the Random Forest Model.

2) **Maximum of Neural Network Score and Random Forest Score:** In this approach we used the fraud scores from both Neural Network and Random Forest and then chose the highest score from these two models. The newly obtained maximum scores are then used to find fraudulent records.

The below diagram shows the histogram of fraud scores from this technique. The fraud scores are scattered from 0 to 1 and there is a good amount of fraud scores on right i.e. records which are flagged as fraud. **We used logarithmic scale on y-axis for clarity.**



The trend of count of Max(RF+NN) for Max(RF+NN) (bin). The marks are labeled by count of Max(RF+NN).

Overall Bad Rate is 0.3%		Bin Statistics				Cumulative Statistics						
Populatio n Bin	Total # Records	# Good	# Bad	% Good	% Bad	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	Pos. Ratio*	Pos. Rate**(%
0.01	886	593	293	0.67	99.32	593	293	0.7	99.3	98.7	2.0	66.93
0.02	886	885	1	1.00	0.34	1,478	294	1.7	99.7	98.0	5.0	83.41
0.03	886	886	0	1.00	0.00	2,364	294	2.7	99.7	97.0	8.0	88.94
0.04	886	886	0	1.00	0.00	3,250	294	3.7	99.7	96.0	11.1	91.70
0.05	887	887	0	1.00	0.00	4,137	294	4.7	99.7	95.0	14.1	93.36

Population Bin	Fraud Savings	Lost Sales	ROI
0.01	$175,800	$5,930	$165,439
0.02	$176,400	$14,780	$157,189
0.03	$176,400	$23,640	$148,329
0.04	$176,400	$32,500	$139,469
0.05	$176,400	$41,370	$130,599

The plot below demonstrates the business values of Maximum Model.

Again, this ensemble result does not seem to be better than using only the Random Forest Model.

6.3 Conclusion to the Results Part

From comparing all the above models and their performances, we could decide that Random Forest Model with the 9 variables selected during the forward feature selection process performed best. And to generate most business value, i.e. ROI, we should flag all the records with top 0.5% score as frauds. The estimated ROI using the given dataset is $168,409.

Appendix
Data Quality Report

Card Payment Data

Summary

File Description

Card payment data is a dataset containing 95,007 records of card transaction. It includes information about card number, date, merchant's number, description, state, and zip code, transaction type, and amount. It is also labeled as fraud or not. In total, there are 298 labeled fraudulent records.

File Name

Card payments.xlsx

Data Source

This dataset is based on real transaction records, while some records are manipulated to be fraudulent for education purpose.

Number of Records

95,007

Number of Fields

10 variables in total – 1 numeric, 7 categorical, 1 text, 1 date
Numeric: amount
Categorical: recordnum, cardnum, merchnum, merch.state, merch.zip, transtype, and fraud
Text: merch.description
Date: date

Time Frame

2010-01-01 to 2010-12-31

Field Explanation

1. recordnum

Description

recordnum is a categorical variable. It works as the ordinal reference number for each card payment record.

Distribution

100% populated with 95,007 unique values, ranging from 1 to 95,007.

2. cardnum

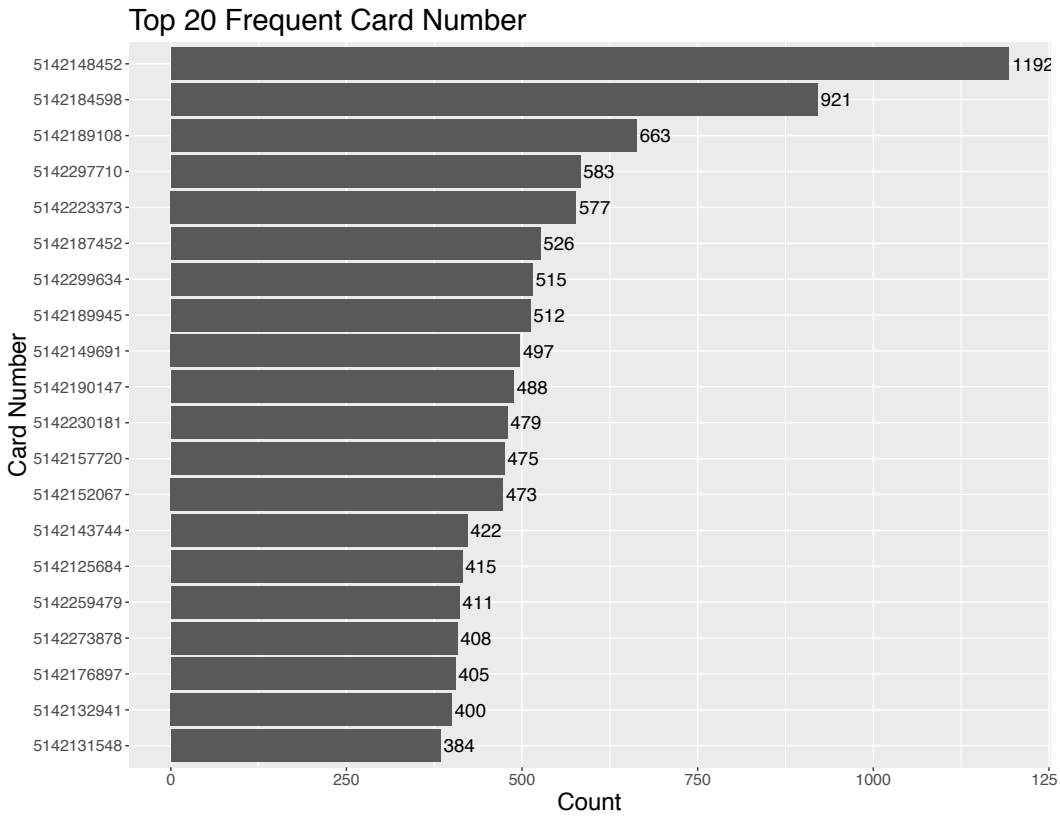
Description

cardnum is a categorical variable. It is the number of the card used for the payment.

Distribution

100% populated with 1,634 unique values. The following table shows the distribution of times of a card used, and the plot shows top 20 frequent card numbers.

Min	1st Qu	Median	Mean	3rd Qu	Max
1	10	30	58	72	1192



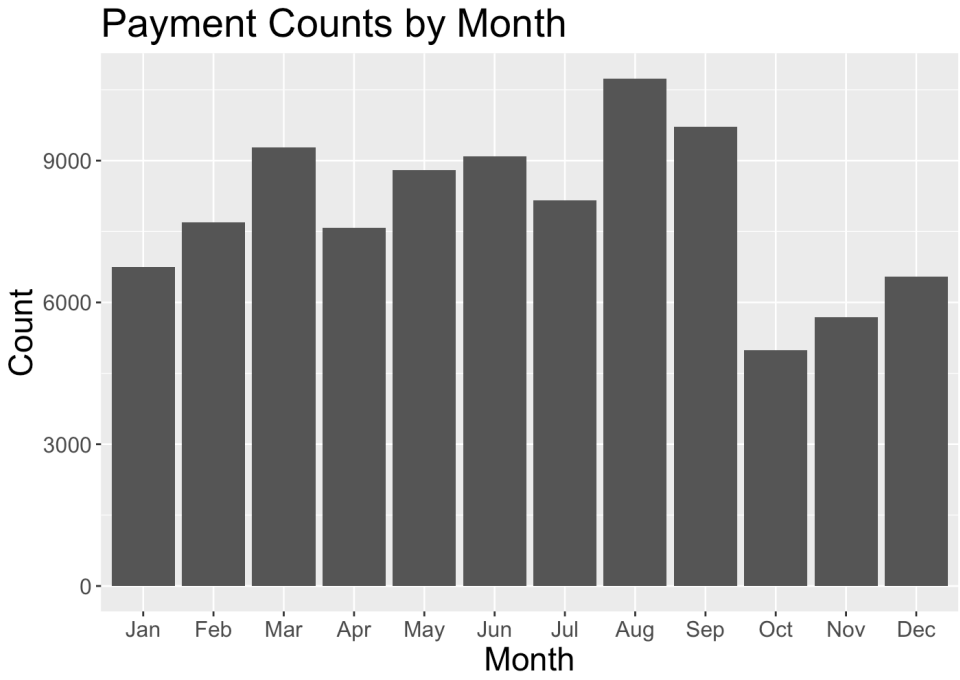
3. date

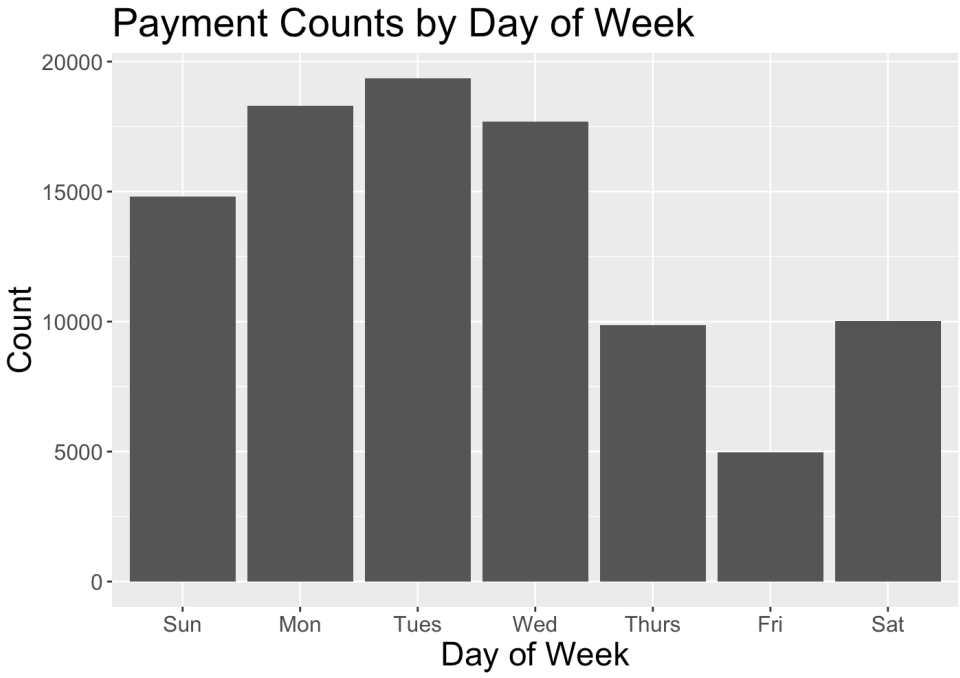
Description

date is a date variable, indicating the date that payment was made.

Distribution

100% populated with 365 unique values, i.e. every day in year 2010. Plot the distribution of the months and day of week as below, we can see that the number of payments decreases in winter and the lowest number appears in October and Friday.





4. merchnum

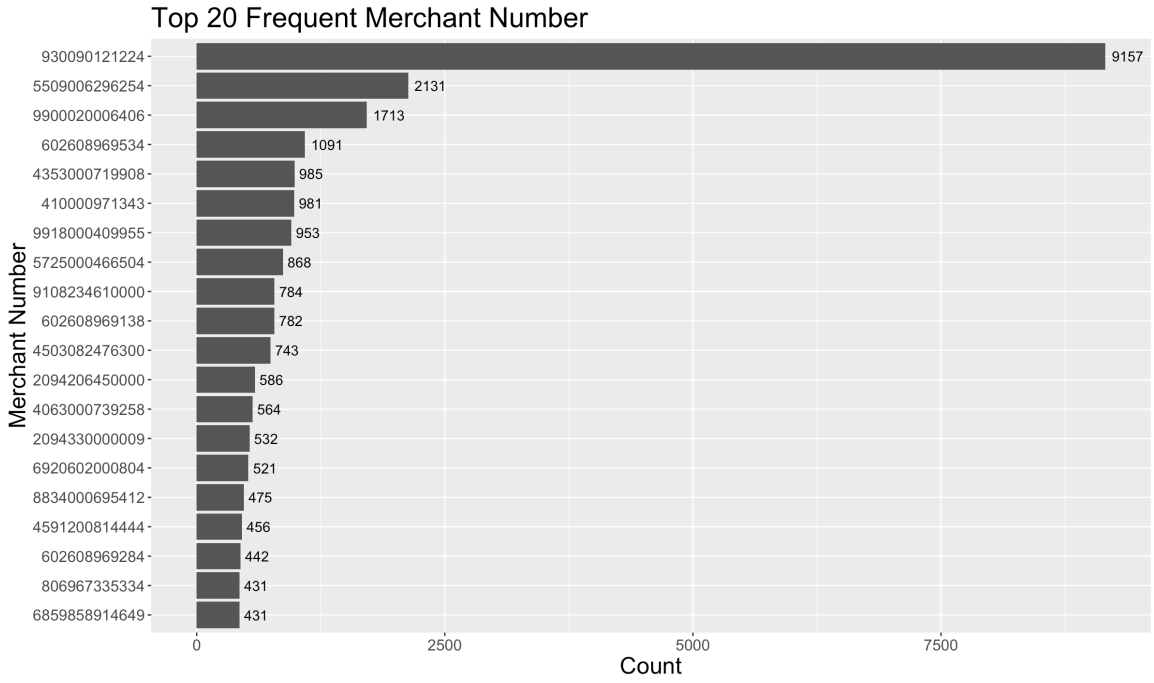
Description

merchnum is a categorical variable. It is the merchant number involved in the payment.

Distribution

96.6% populated with 13,089 unique values. There are 3,174 missing values, and 53 0's. The following table shows the distribution of valid merchant numbers, and the plot shows top 20 frequent merchant numbers.

Min	1st Qu	Median	Mean	3rd Qu	Max
1	1	1	7.013	3	9157



5. merch.description

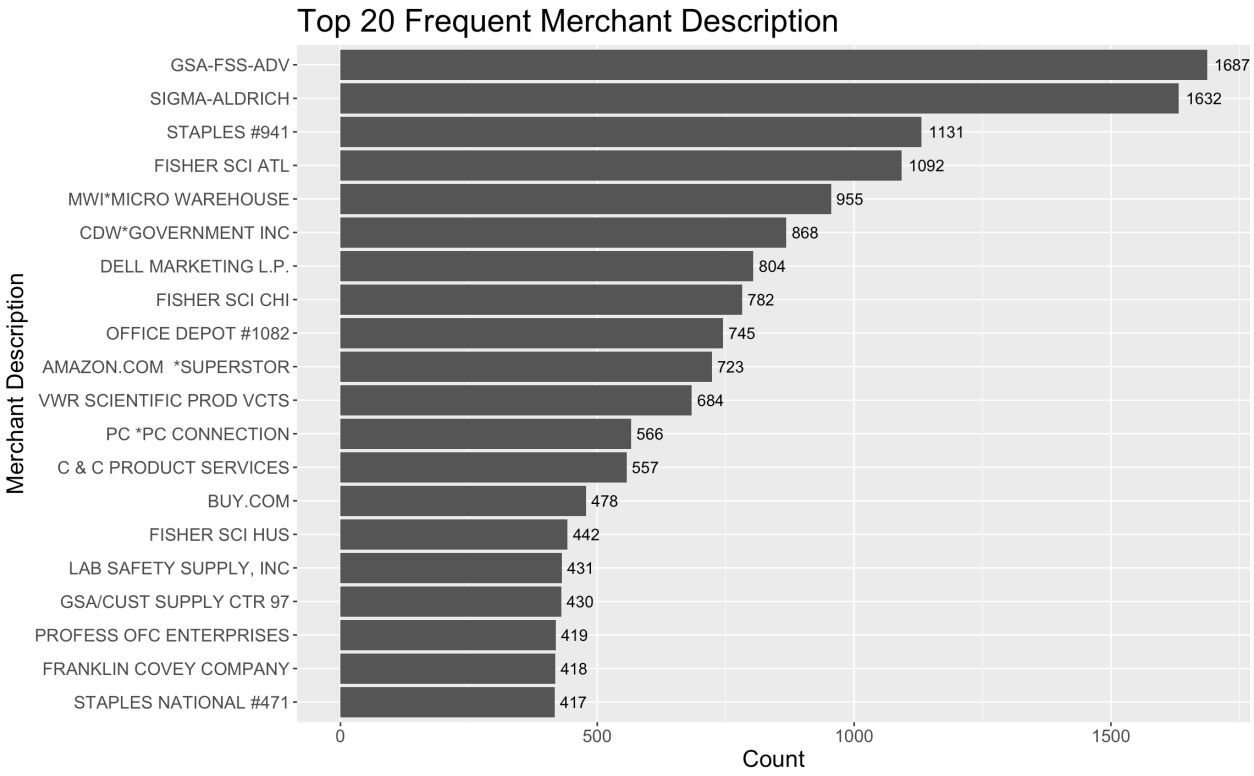
Description

merch.description is a text variable. It provides information about the merchant, and some may also include some transaction explanation, such as “FEDEX SHP 12/23/09 AB#”.

Distribution

100% populated with 12,964 unique values. The following table shows the distribution of valid merchant descriptions, and the plot shows top 20 frequent merchant descriptions

Min	1st Qu	Median	Mean	3rd Qu	Max
1	1	1	7.013	3	9157



6. merch.state

Description

merch.state is a categorical variable, indicating the abbreviations of states of the merchant.

Distribution

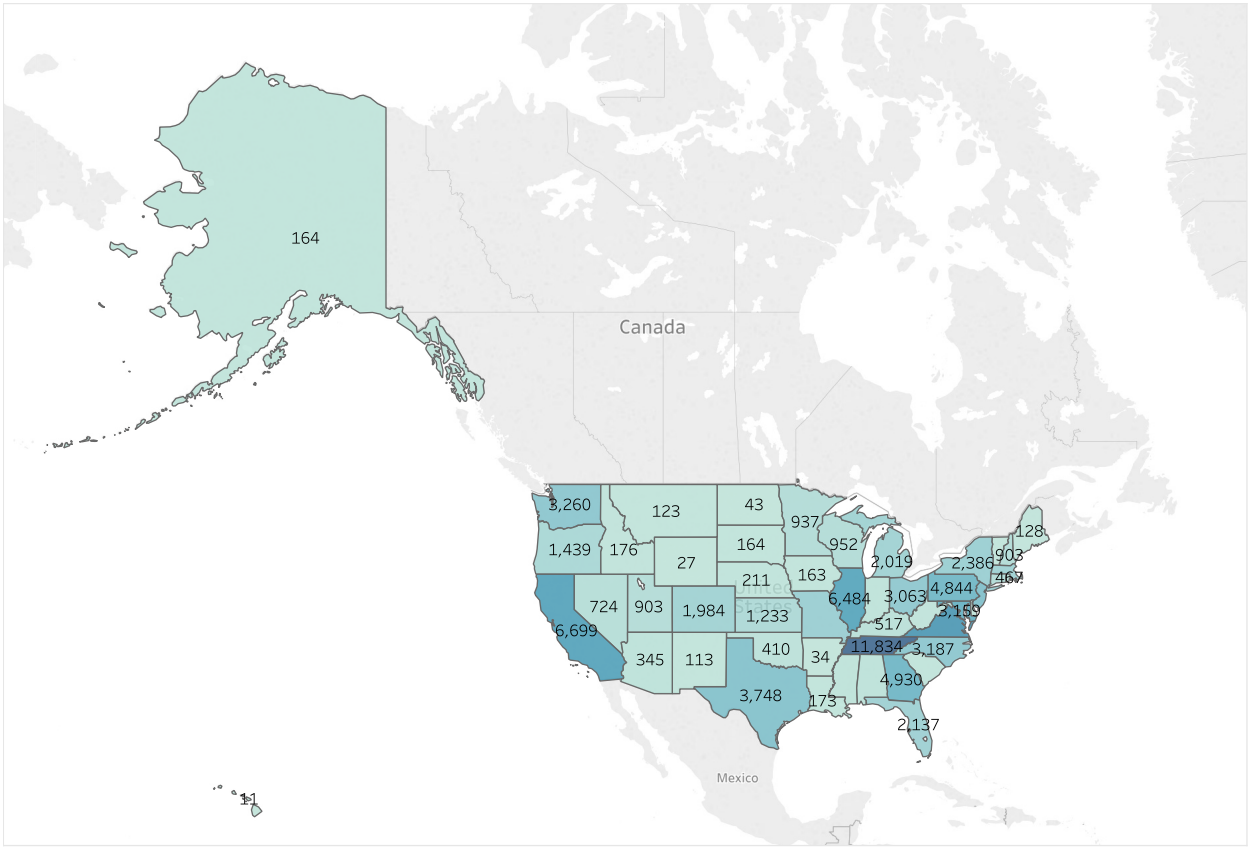
98.93% populated with 60 unique values. 51 values stand for regions in the United States, 50 states and District of Columbia. 7 values stand for Canada. Besides that, we discovered are 1,016 missing values, and 1 invalid value *US*.

Distribution of states is showed in the following table. States in Canada are marked in blue. Invalid and missing values are marked in red. Regions in the United States are in black

	AB	**AK**	**AL**	**AR**	**AZ**	BC	**CA**	**CO**	**CT**
1016	5	164	343	34	345	23	6699	1984	949
DC	DE	FL	GA	HI	IA	ID	IL	IN	KS
3159	70	2137	4930	11	163	176	6484	244	1233
KY	LA	MA	MB	MD	ME	MI	MN	MO	MS
517	173	2075	3	5344	128	2019	937	2415	244
MT	NC	ND	NE	NH	NJ	NM	NS	NV	NY
123	3187	43	211	903	3904	113	5	724	2386
OH	OK	ON	OR	PA	PQ	QC	RI	SC	SD
3063	410	137	1439	4844	14	4	467	153	164
TN	TX	**US**	UT	VA	VT	WA	WI	WV	WY
11834	3748	**1**	903	7698	85	3260	952	181	27

The following picture displays distribution by states in the United States. We can clearly see that TN has the largest number of records(11,834).

Distribution by States

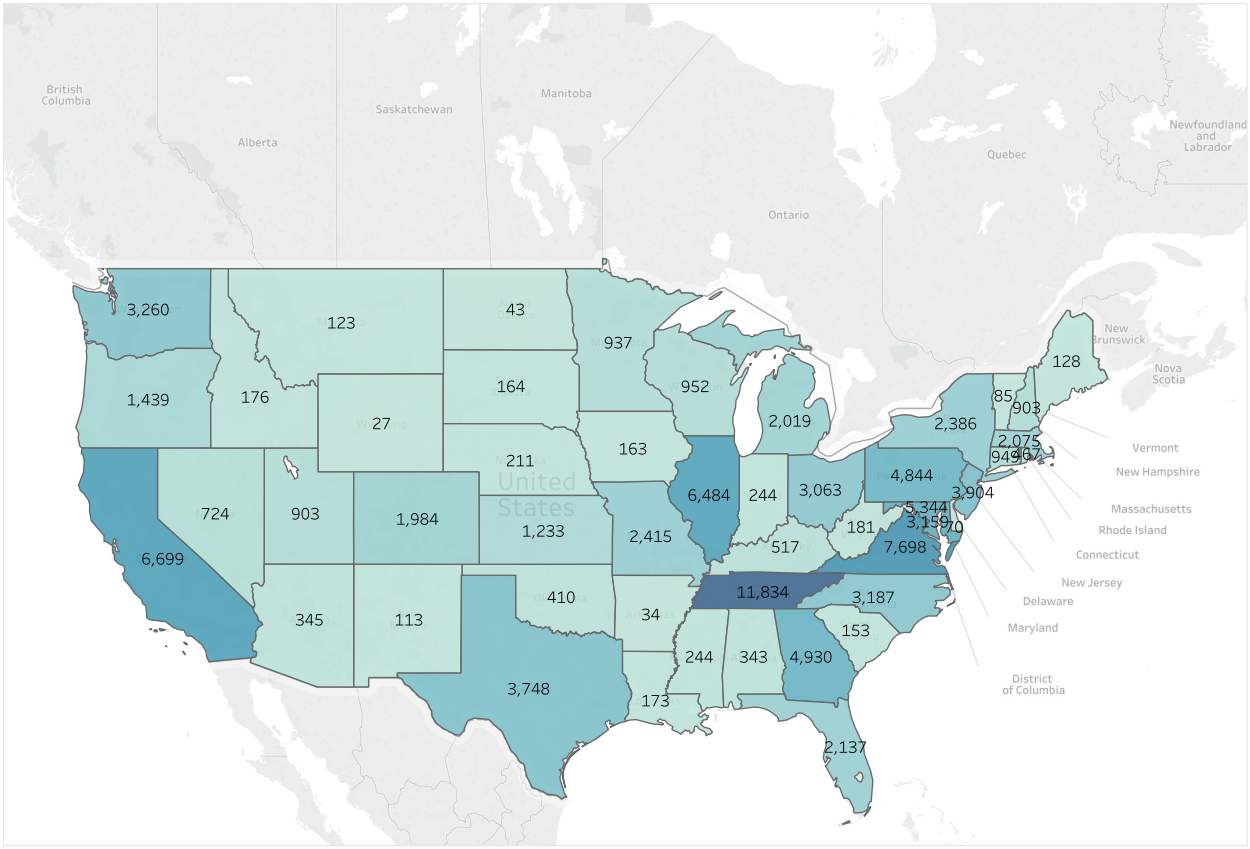


Map based on Longitude (generated) and Latitude (generated). Color shows sum of Number of Records. Details are shown for Merch.State.

Number of Records

1 11,834

Distribution by States



Map based on Longitude (generated) and Latitude (generated). Color shows sum of Number of Records. Details are shown for Merch.State.

Number of Records

1 11,834

7. merch.zip

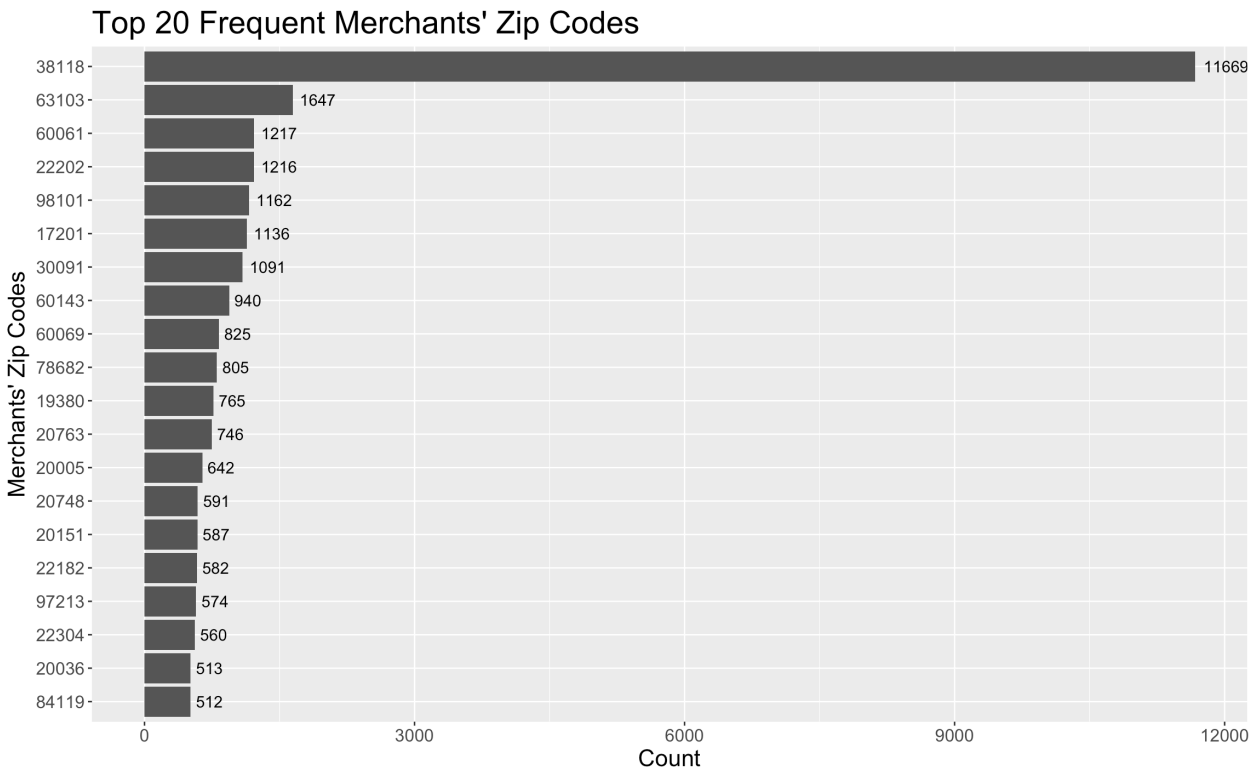
Description

merch.zip is a categorical variable, indicating the zip code of the merchant.

Distribution

95.49% populated with 4584 unique values. 86.99% values have in valid 5-digit zip code. The following table shows the distribution of valid zip codes, and the plot shows top 20 zip codes. Although there is one zip code “38118” appears 10 times more than other values, it is in TN, which is the most frequent states; hence we do not treat it as a frivolous value.

Min	1st Qu	Median	Mean	3rd Qu	Max
1	1	3	20.71	10	11669


Top 20 Frequent Merchants' Zip Codes
Merchants' Zip Codes
38118
63103
60061
22202
98101
17201
30091
60143
60069
78682
19380
20763
20005
20748
20151
22182
97213
22304
20036
84119
11669
1647
1217
1216
1162
1136
1091
940
825
805
765
746
642
591
587
582
574
560
513
512
0
3000
6000
9000
12000
Count

8. transtype

Description

transtype is a categorical variable, indicating the type of transaction.

Distribution

100% populated with 1 unique value – P.

9. amount

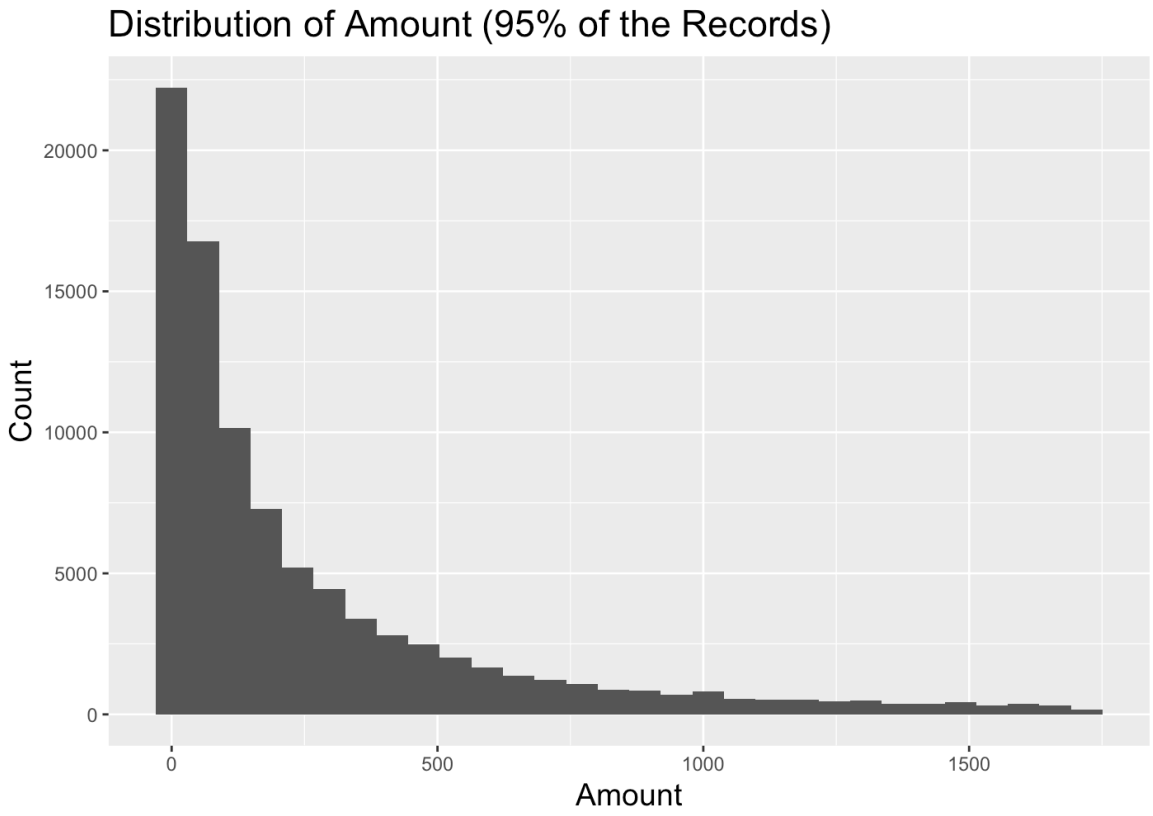
Description

“amount” is a numeric variable. It is the transaction amount of that record

Distribution

100% populated. The table shows the distribution, and the plot excludes the tail and shows 95% of the data.

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.01	33.25	136.5	381	420.8	47900



10. fraud

Description

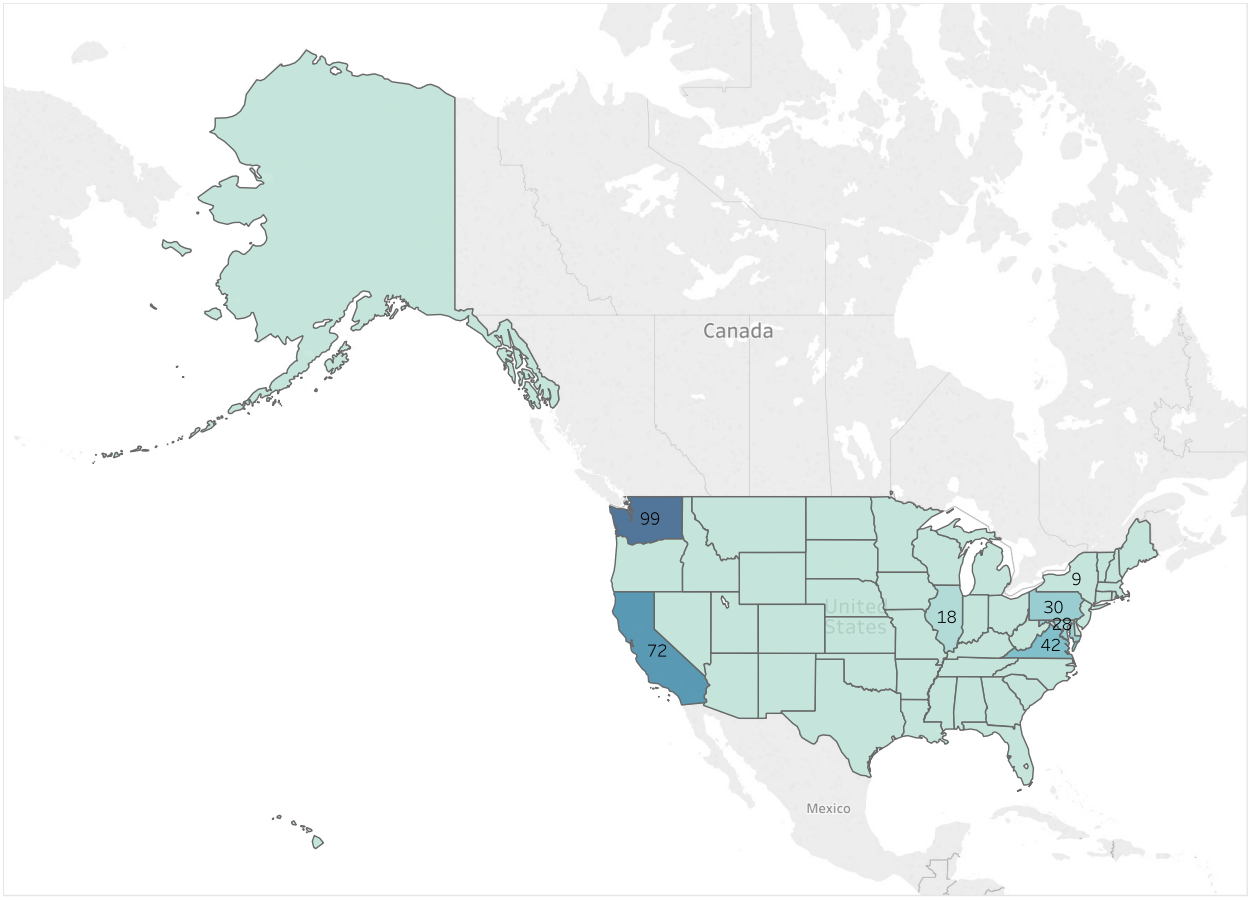
fraud is a categorical variable. It is the label of manually added fraudulent records for education purpose. If the value in this field is 1, it represents that record is fraudulent; if there is no value there, then it means that record is not fraudulent.

Distribution

0.3137% populated with 1 unique value – "1". In total, there are 298 fraudulent records in the dataset.

The following plot shows the fraud distribution by states. It seems that frauds only appeared in certain cities. These cities are WA, CA, VA, PA, MD, IL, and NY.

Fraud Distribution by States



Map based on Longitude (generated) and Latitude (generated). Color shows sum of Fraud. Details are shown for Merch.State.

Fraud

9 99