DSO 562 Project 3

Card Payment Fraud Analysis Report

Team 7 - May 1, 2017

Table of Content

1. Executive Summary

Card payment fraud involves unauthorized activities that individuals or merchants use someone else's card information to charge purchases or to withdraw funds from the card. Under this type of fraud, cardholders' property is invaded. Also, both financial institutions and merchants are victims who assume responsibility for most of the money lost as a result of fraud. Therefore, accurate prediction to detect card payment fraud can lead to great savings.

In this project, we have applied multiple machine learning techniques to build supervised models for fraud detection. Four major steps have been conducted in this analysis:

- Data Preparation: data cleaning and creating expert variables
- Building fraud algorithms:
    1. LASSO
    2. Random Forest
    3. Support Vector Machines (SVM)
    4. Neural Network
    5. XGBoost
- Model Selection
- Fraud score calculation

We created 80 expert variables depending on 2 base variables (cardnum and merchnum) and 5 time windows (1, 3, 7, 14, 31 days). As a procedure, the data has been randomly split into training set (80%) and testing set (20%).

Regarding the algorithms, 5 distinct prediction metrics have been selected build models on training dataset and predict on testing dataset. For each model, we have examined the confusion matrix and calculated the true positive rate as well as the AUC (Area Under Curve) for evaluation. We further compared the performance of those models by calculating their Fraud Detection Rate (FDR) and Return On Investment (ROI) and finally chose XGBoost as out best-performed model.

By adopting our best-performed model, 97.7% of fraud cases are captured at the top 1% of the population with $163,360 loss prevention.

To serve as a guide walking through the entire process of the fraud detection, this report is divided into 5 main sections, data description, data preparations, fraud algorithm assessment, model selection and result, and future work.

2. Data Description

2.1 Data Summary

File Name: Card payments.xlsx
Data Size: 95,007 records
Fields: 10 (including 9 categorical variables and 1 numeric variable)
Field Details: recordnum, cardnum, date, merchnum, merch description, merch state, merch zip, transtype, amount, fraud
Time Frame: 01/01/2010 - 12/31/2010

This dataset provides transaction information about card payment in 2010, and 298 records are labeled as fraud. The variable details are shown below.

2.2 Variable Details

Table 1 Variables

Variable Name	Type	Description
recordnum	Categorical	Serial number of each payment
cardnum	Categorical	Card number associated with each payment
date	Categorical	Date that each payment was made
merchnum	Categorical	Merchant number associated with each payment
merch description	Categorical	Merchant description for each payment
merch state	Categorical	State where the merchant was loacted
merch zip	Categorical	Zip code for each merchant
transtype	Categorical	Transaction type for each payment
amount	Numeric	Transaction amount for each payment
fraud	Categorical	Fraudulent transaction (shown as 1)

2.2.1 Base Variables

For this project, we decided to use cardnum and merchnum as base variables to investigate potential card payment fraud.

Table 2 Base Variables

Variable Name	Type	Description
cardnum	Categorical	Card number associated with each payment
merchnum	Categorical	Merchant number associated with each payment

cardnum

This field provides the card number for each transaction, and there are 1,634 unique values with no missing information. The card number "5142148452" appears the most with 1,192 times, and the distribution of top 20 card numbers is shown in the graph below.

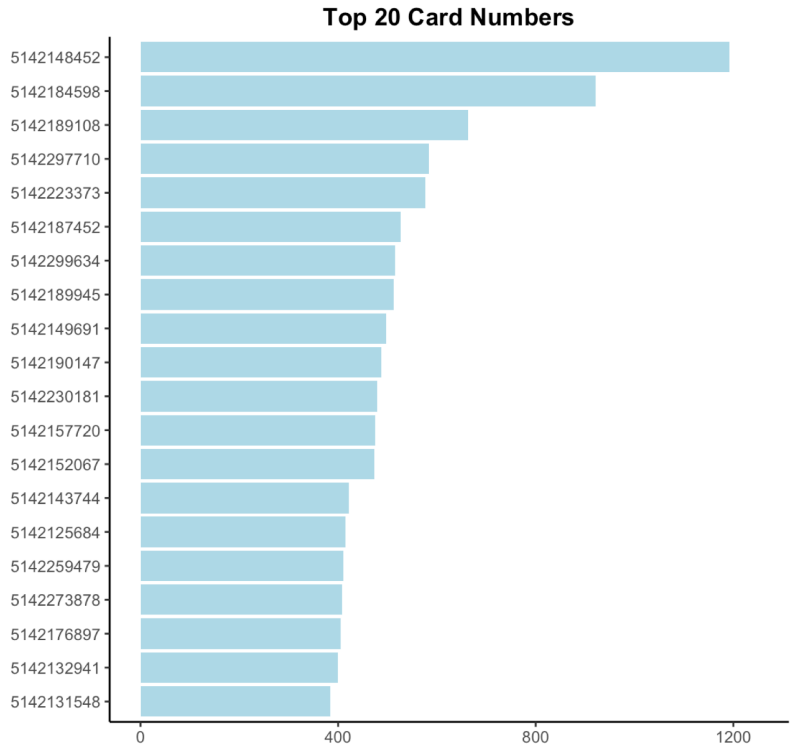


Figure 1: Top 20 Card Numbers

merchnum

This field provides the merchant number for each transaction, and there are 13,088 unique numbers with 3,174 missing values. The merchant number “930090121224” appears the most with 9,157 times, and the distribution of top 20 merchant numbers is shown in the graph below.

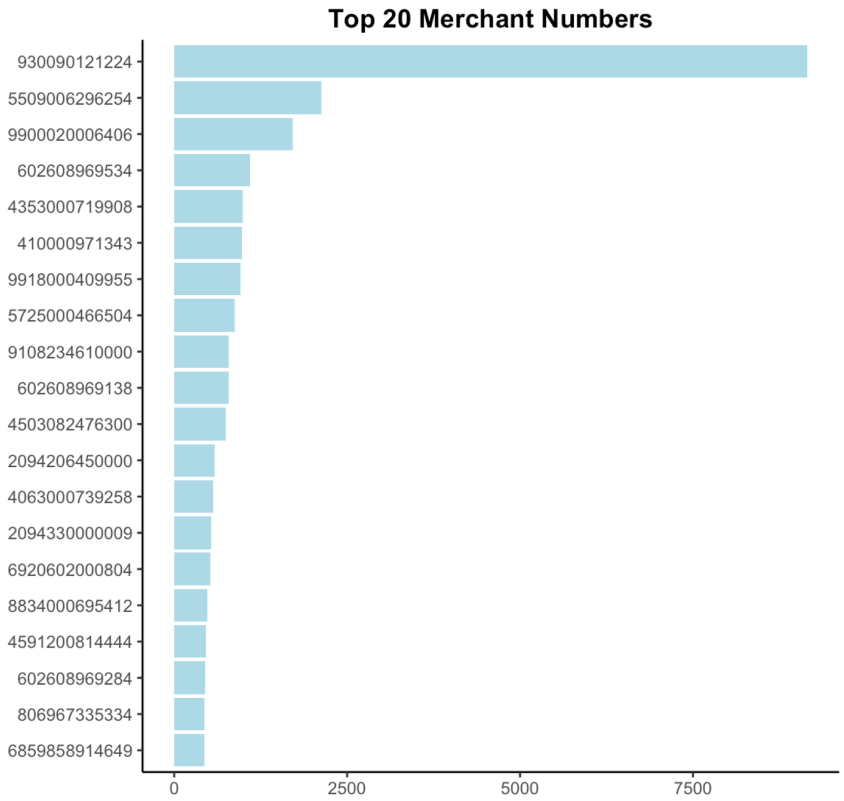


Figure2: Top 20 Merchant Numbers

3. Data Preparation

3.1 Create Expert Variables

3.1.1 Card Level

3.1.1.1 Frequency

To detect potential fraud transaction with each card number we observed, we counted the total number of unique merchants, based on the merchant number associated with each card, the total number of state and zip code, and the total number of transactions with different amounts in the past 1, 3, 7, 14, and 31 days.

Table 3 Frequency Variables with Card Level

Frequency		
ID	**Field**	**Description**
1	past1day_cnt_merchant	How many merchants were this card associated with in the past 1 day
2	past1day_cnt_state	How many States were this card associated with in the past 1 day
3	past1day_cnt_zip	How many Zip code was this card associated with in the past 1 day
4	past1day_transaction_cnt	How many transactions were this card associated with in the past 1 day
5	past3day_cnt_merchant	How many merchants were this card associated with in the past 3 days
6	past3day_cnt_state	How many States were this card associated with in the past 3 days
7	past3day_cnt_zip	How many Zip code was this card associated with in the past 3 days
8	past3day_transaction_cnt	How many transactions that had different amounts were this card associated with in the past 3 days
9	past7day_cnt_merchant	How many merchants were this card associated with in the past 7 days
10	past7day_cnt_state	How many States were this card associated with in the past 7 days
11	past7day_cnt_zip	How many Zip code was this card associated with in the past 7 days
12	past7day_transaction_cnt	How many transactions were this card associated with in the past 7 days
13	past14day_cnt_merchant	How many merchants were this card associated with in the past 14 days
14	past14day_cnt_state	How many States were this card associated with in the past 14 days
15	past14day_cnt_zip	How many Zip code was this card associated with in the past 14 days
16	past14day_transaction_cnt	How many transactions were this card associated with in the past 14 days
17	past31day_cnt_merchant	How many merchants were this card associated with in the past 31 days

Frequency		
ID	Field	Description
18	past31day_cnt_state	How many States were this card associated with in the past 31 days
19	past31day_cnt_zip	How many Zip code was this card associated with in the past 31 days
20	past31day_transaction_cnt	How many transactions were this card associated with in the past 31 days

3.1.1.2 Amount

It would be suspicious that the amount charged on a transaction was way different than the card normal transaction pattern. Therefore, the total, minimum, maximum, median, and average transaction amounts for each card number in the past 1, 3, 7, 14, and 31 days were calculated to identify potential fraud.

Table 4 Amount Variables with Card Level

Amount		
ID	Field	Description
1	past1day_amount	The total transaction amounts were charged on the card in the past 1 day
2	min_card_1damount	The minimum transaction amount was charged on the card in the past 1 day
3	max_card_1damount	The maximum transaction amount was charged on the card in the past 1 day
4	median_card_1damount	The median amount was charged on the card in the past 1 day
5	avg_card_1damount	The average amount was charged on the card in the past 1 day
6	past3day_amount	The total transaction amounts were charged on the card in the past 3 days
7	min_card_3damount	The minimum transaction amount was charged on the card in the past 3 days
8	max_card_3damount	The maximum transaction amount was charged on the card in the past 3 days
9	median_card_3damount	The median amount was charged on the card in the past 3 days
10	avg_card_3damount	The average amount was charged on the card in the past 3 days
11	past7day_amount	The total transaction amounts were charged on the card in the past 7 days
12	min_card_7damount	The minimum transaction amount was charged on the card in the past 7 days
13	max_card_7damount	The maximum transaction amount was charged on the card in the past 7 days

Amount		
ID	Field	Description
14	median_card_7damount	The median amount was charged on the card in the past 7 days
15	avg_card_7damount	The average amount was charged on the card in the past 7 days
16	past14day_amount	The total transaction amounts were charged on the card in the past 14 days
17	min_card_14damount	The minimum transaction amount was charged on the card in the past 14 days
18	max_card_14damount	The maximum transaction amount was charged on the card in the past 14 days
19	median_card_14damount	The median amount was charged on the card in the past 14 days
20	avg_card_14damount	The average amount was charged on the card in the past 14 days
21	past31day_amount	The total transaction amounts were charged on the card in the past 31 days
22	min_card_31damount	The minimum transaction amount was charged on the card in the past 31 days
23	max_card_31damount	The maximum transaction amount was charged on the card in the past 31 days
24	median_card_31damount	The median amount was charged on the card in the past 31 days
25	avg_card_31damount	The average amount was charged on the card in the past 31 days

3.1.2 Merchant Level

3.1.2.1 Frequency

To detect potential fraud transaction with each merchant we observed, we counted the total transactions associated with each merchant, the total transactions of unique cardnum associated with each merchant, and the the total number of transactions with different amounts in the past 1, 3, 7, 14, and 31 days.

Table 5 Frequency Variables with Merchant Level

Frequency		
ID	Field	Description
1	past1day_cnt	How many transactions with this merchant in the past 1 day
2	past1day_diff_card_cnt	How many transactions with this merchant associated with different card in the past 1 day

Frequency		
ID	Field	Description
3	past1day_diff_amount	How many transactions with this merchant associated with different amount in the past 1 day
4	past3day_cnt	How many transactions with this merchant in the past 3 day
5	past3day_diff_card_cnt	How many transactions with this merchant associated with different card in the past 3 day
6	past3day_diff_amount	How many transactions with this merchant associated with different amount in the past 3 day
7	past7day_cnt	How many transactions with this merchant in the past 7 day
8	past7day_diff_card_cnt	How many transactions with this merchant associated with different card in the past 7 day
9	past7day_diff_amount	How many transactions with this merchant associated with different amount in the past 7 day
10	past14day_cnt	How many transactions with this merchant in the past 14 day
11	past14day_diff_card_cnt	How many transactions with this merchant associated with different card in the past 14 day
12	past14day_diff_amount	How many transactions with this merchant associated with different amount in the past 14 day
13	past31day_cnt	How many transactions with this merchant in the past 31 day
14	past31day_diff_card_cnt	How many transactions with this merchant associated with different card in the past 31 day
15	past31day_diff_amount	How many transactions with this merchant associated with different amount in the past 31 day

3.1.2.2 Amount

It would be suspicious that the amount charged on a transaction was way different than the merchant normal transaction pattern. Therefore, the total, minimum, maximum, median, and average transaction amounts for each card number in the past 1, 3, 7, 14, and 31 days were calculated to identify potential fraud.

Table 6 Amount Variables with Merchant Level

Amount		
ID	**Field**	**Description**
1	**past1day_amount**	The total transaction amounts were charged on the merchant in the past 1 day
2	**min_merch_1damount**	The minimum transaction amount was charged on the merchant in the past 1 day
3	**max_merch_1damount**	The maximum transaction amount was charged on the merchant in the past 1 day
4	**median_merch_1damount**	The median amount was charged on the merchant in the past 3 day
5	**past3day_amount**	The total transaction amounts were charged on the merchant in the past 3 day
6	**min_merch_3damount**	The minimum transaction amount was charged on the merchant in the past 3 day
7	**max_merch_3damount**	The maximum transaction amount was charged on the merchant in the past 3 day
8	**median_merch_3damount**	The median amount was charged on the merchant in the past 3 day
9	**past7day_amount**	The total transaction amounts were charged on the merchant in the past 7 day
10	**min_merch_7damount**	The minimum transaction amount was charged on the merchant in the past 7 day
11	**max_merch_7damount**	The maximum transaction amount was charged on the merchant in the past 7 day
12	**median_merch_7damount**	The median amount was charged on the merchant in the past 7 day
13	**past14day_amount**	The total transaction amounts were charged on the merchant in the past 14 day
14	**min_merch_14damount**	The minimum transaction amount was charged on the merchant in the past 14 day
15	**max_merch_14damount**	The maximum transaction amount was charged on the merchant in the past 14 day
16	**median_merch_14damount**	The median amount was charged on the merchant in the past 14 day
17	**past31day_amount**	The total transaction amounts were charged on the merchant in the past 31 day
18	**min_merch_31damount**	The minimum transaction amount was charged on the merchant in the past 31 day

Amount		
ID	Field	Description
19	max_merch_31damount	The maximum transaction amount was charged on the merchant in the past 31 day
20	median_merch_31damount	The median amount was charged on the merchant in the past 31 day

4. Fraud Algorithm Assessment

4.1 Method 1: LASSO

The lasso is a regression method that performs both variable selection and regularization in order to enhance prediction accuracy and interpretability. The estimated coefficients are shrunken towards zero relative to the least squares estimates in linear regression or to the maximum likelihood estimates in logistic regression. This shrinkage has the effect of reducing variance.

4.1.1 Working Principle

For logistic regression, the objective function for the penalized logistic regression uses the negative binomial log-likelihood, and is

$$\min_{(\beta_0,\beta)\in\mathbb{R}^{p+1}} -\left[\frac{1}{N}\sum_{i=1}^{N} y_i \cdot (\beta_0 + x_i^T\beta) - \log(1+e^{(\beta_0+x_i^T\beta)})\right] + \lambda\left[(1-\alpha)||\beta||_2^2/2 + \alpha||\beta||_1\right]$$

After fitting the lasso model on our training data set, we could examine how the coefficients were associated with λ values in our model by looking at the Figure 3. It shows the path of coefficients as λ varies and each curve corresponds to a variable. The axis above indicates the number of nonzero coefficients at the current λ.

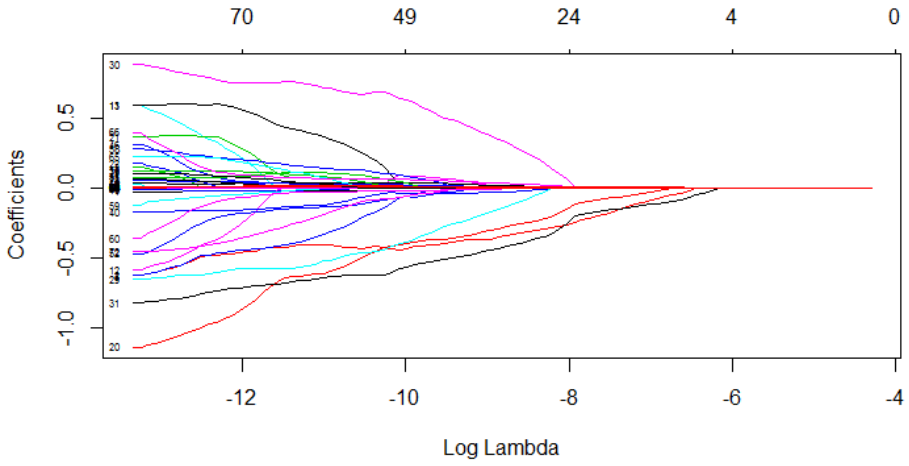


Figure 3: Misclassification Error by Log Lambda

In our project, we first used 5-fold cross validation to choose the best tuning parameter that minimized the cross-validation error. The relationship between the choice of λ and the misclassification error is plotted in Figure 4. The red dotted line is the cross-validation curve with upper and lower standard deviation curves along the λ sequence. The first vertical dashed line indicates the value at which the minimal misclassification error is achieved. And the second vertical dashed line indicates the most regularized model whose misclassification error is within one standard error of the minimal value.

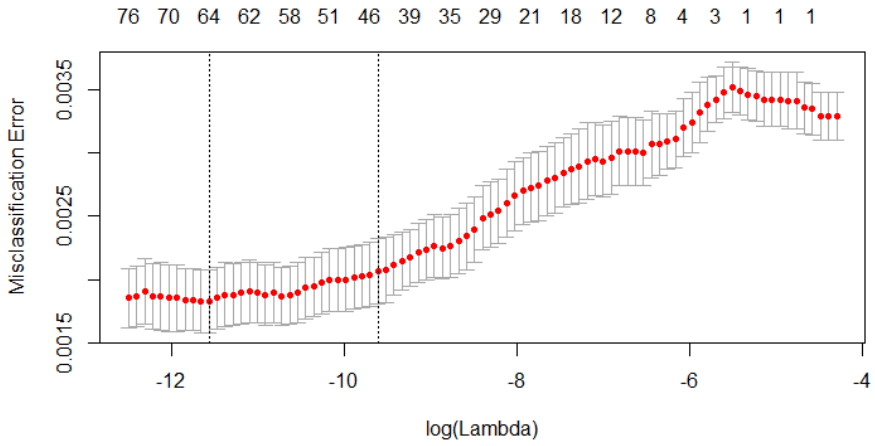


Figure 4: Choose best lambda with cross-validation

In our model, the best λ turned out to be 9.5257e-06, which shrunk the coefficient estimates of 16 variables to 0. The remaining variables and their coefficient estimates are listed in Figure 5.

Variable	Coefficient Estimate	Variable	Coefficient Estimate	Variable	Coefficient Estimate
(Intercept)	-8.6497	past14day_cnt_state	-0.6881	past3day_diff_amount	-0.0281
past1day_transaction_cnt	-0.0243	past14day_amount.x	-0.0003	past7day_cnt	-0.0437
past1day_cnt_merchant	-0.4270	avg_card_14damount	0.0008	max_merch_7damount	0.0001
past1day_cnt_state	-0.4226	min_card_14damount	0.0011	past7day_amount.y	1.2381E-05
past1day_amount.x	0.0006	max_card_14damount	0.0006	avg_merch_7damount	3.1948E-06
avg_card_1damount	-0.0006	past31day_transaction_cnt	-0.0011	median_merch_7damount	-2.544E-05
median_card_1damount	-0.0012	past31day_cnt_zip	0.1010	past7day_diff_card_cnt	0.1369
min_card_1damount	0.0038	past31day_cnt_state	-0.1459	past7day_diff_amount	0.0742
past3day_transaction_cnt	-0.0207	median_card_31damount	-0.0001	past14day_cnt	-0.0014
past3day_cnt_zip	-0.0108	min_card_31damount	-0.0043	past14day_amount.y	4.566E-06
past3day_cnt_state	0.4511	max_card_31damount	-0.0004	avg_merch_14damount	-3.52E-06
min_card_3damount	0.0006	past1day_cnt	0.0110	median_merch_14damount	2.3784E-05
max_card_3damount	-0.0002	max_merch_1damount	-8.981E-06	past14day_diff_card_cnt	-0.2927
past7day_transaction_cnt	0.0348	past1day_amount.y	-1.822E-05	past14day_diff_amount	0.0088
past7day_cnt_merchant	-0.6696	avg_merch_1damount	1.041E-05	past31day_cnt	0.0023
past7day_cnt_zip	0.1006	median_merch_1damount	-2.281E-05	max_merch_31damount	0.0001
past7day_amount.x	0.0003	past1day_diff_card_cnt	0.0994	past31day_amount.y	-4.171E-06
avg_card_7damount	-0.0005	past1day_diff_amount	-0.1580	avg_merch_31damount	-6.534E-06
median_card_7damount	0.0002	past3day_amount.y	-8.64E-06	median_merch_31damount	0.0001
past14day_transaction_cnt	0.0324	avg_merch_3damount	-1.398E-05	past31day_diff_card_cnt	0.0520
past14day_cnt_merchant	-0.5775	median_merch_3damount	3.004E-05	past31day_diff_amount	0.0159
past14day_cnt_zip	0.7537	past3day_diff_card_cnt	0.1726		

Figure 5: Coefficient estimates for remaining variables

With the remaining variables determined by the best λ, we applied our model on test data. For prediction, we also decided to set a threshold of 1% to determine the potential class of each record. If the predicted probability is greater than or equal to 1%, the class of "1" - fraud- would be assigned. Otherwise, the record would not be predicted as a fraud.

4.1.2 Model Performance

1) Confusion Matrix

The confusion matrix of the prediction result is shown below. The true positive rate of the model is 79.17%(38/(38+10)).

		Predicted	
		0	1
Actual	0	18444	515
	1	10	38

2) **AUC**

The area under the ROC curve for the model is 88.23%, which is much larger than random classifier.

4.2 Method 2: Random Forest

Random Forest is a supervised learning method for both classification and regression analysis that operates by generating multiple training decision trees while reducing the correlations among decision trees. When building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. A new sample of m predictors is taken at each split, and the number of predictors considered at each split is approximately equal to the square root of the total number of predictors; thereby making the average of the resulting trees less variable and hence more reliable.

4.2.1 Working Principle

The training algorithm of Random Forest applies the general technique of bootstrap and bagging aggregation to select training set. Given a training set $X = x_1, ..., x_n$ with responses = $y_1, ..., y_n$, random forest repeatedly selects a random sample with replacement of the training set and fits trees to these samples by selecting random subset of features.
In this project, we set 200 trees, the maximum number of different trees is allowed to build, and 5-fold cross validation as the default tree setting to train the model. The stop round of 20 and stopping metrics of "Auto" were defined to trigger the stopping points of the metrics among 200 trees so that the results would be collected at the same test performance level with shorter period of time. More specifically, each split of a decision tree was determined by achieving the best reduction in the classification error; therefore, this setting enabled the metric to stop modifying a model if the value achieved from all the measuring metrics such as AUC and logloss specified in stopping metrics did not decrease over a period of 20 steps. To determine whether an observation would be a potential fraud, we used the threshold of 0.279 to label a record where the probability of 0.279 or above would be considered as a fraud. Particularly, threshold value was calculated by the Random Forest metrics to achieve the maximum F1 score, a measure of a test's accuracy considered both the precision p and recall r of the test outcome.

$$F_1 = 2 \cdot \frac{1}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Precision is the number of actual positive results divided by the number of all positive results, and recall is the number of correct predicted positive results divided by the number of actual positive results.

4.2.2 Model Performance

1) Confusion Matrix

The true positive rate of the model is 79.69% (51/(13+51)).

		Predicted	
		0	1
Actual	0	18941	6
	1	13	51

2) AUC

The area under the ROC curve for the model is 0.98, which indicates the overall performance of the classifier produced by Random Forest

4.3 Method 3: Support Vector Machines

Support Vector Machines (SVM) is a supervised machine learning algorithm which can be used for both classification and regression analysis. In this project, SVM performs classification task by constructing an optimal separating hyperplane in a multidimensional space to maximize the margin (minimum distance) of the two classes in the training data. Particularly, the maximum margin in a hyperplane offers the best generalization ability. It allows the best classification performance on the training data and leaves much room for the correct classification of the future data.

4.3.1 Working Principle

In a p-dimensional space, a hyperplane is a flat affine subspace of dimension $p - 1$. For instance, a hyperplane is a line in two dimensions (Figure 6). In general, the training examples that are closest to the hyperplane are called support vectors. The goal of SVM is to find a linear or nonlinear separating hyperplane with the maximal margin in the high dimensional space by using different kernel types. In this project, we focused on the linear kernel and radial basis function (RBF) kernel. There are two parameters for a RBF kernel: cost (C) is the penalty parameter of the error term and gamma (γ) is a tolerance term where two points can be considered as similar even if the distance is large after applying gamma (γ).

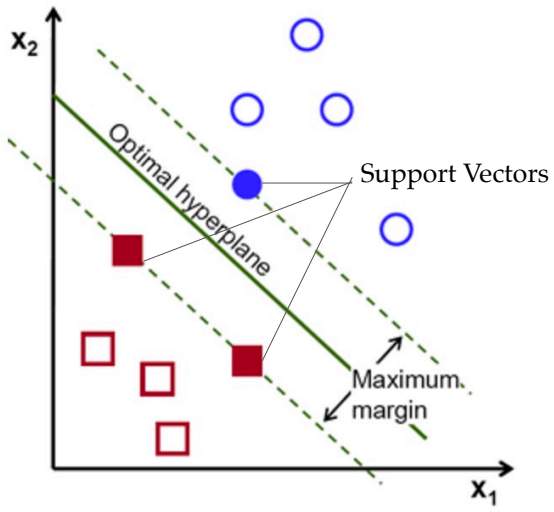


Figure 6: Example of support vectors and hyperplane

For our data set, we applied SVM with linear kernel and RBF kernel respectively to determine which model works better with a lower test misclassification error rate. Before applying SVM, we scaled the 20 features selected from the random forest (section 4.2) to have zero mean and unit variance. The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller ranges. Also, we performed a five-fold cross-validation to select the best choice of parameters with the smallest error rate so that the classifier could accurately predict unknown data. Then we applied on the test data set by using the two models with the best parameters separately, and as a result, the SVM with the RBF kernel performed better with the parameters of cost =10 and gamma = 0.5.

4.3.2 Model Performance

1) Confusion Matrix

The true positive rate of the model is 58.3% (28/(20+28)).

		Predicted	
		0	1
Actual	0	18956	3
	1	20	28

2) AUC

The area under the ROC curve for the model is 0.97, which indicates the overall performance of the classifier produced by non-linear SVM.

4.4 Method 4: Neural Network

In general, neural networks has the power of realizing an arbitrary mapping of one vector space onto another vector space. The main advantage of neural networks is the ability to use some priori unknown hidden information in the dataset. Trained with a back-propagated learning algorithm, Multiple-layer feed-forward (MLF) neural networks are the most popular neural networks, which can be applied to a wide variety of problems.

4.4.1 Working Principle

Neural Networks based fraud detection is based on the human brain working principle. We trained our model to learn through past experience and use its experience in making the decision in the credit card payment fraud detection. The model was trained with transaction patterns from both the perspectives of the merchant number and the card number (variables mentioned in 2. Data Description). Based on the pattern of using credit cards, neural networks model made use of prediction algorithm on these data pattern to classify whether a particular payment transaction was fraudulent or authentic. In our model, we set two hidden layers with 50 neurons for each of the layer.

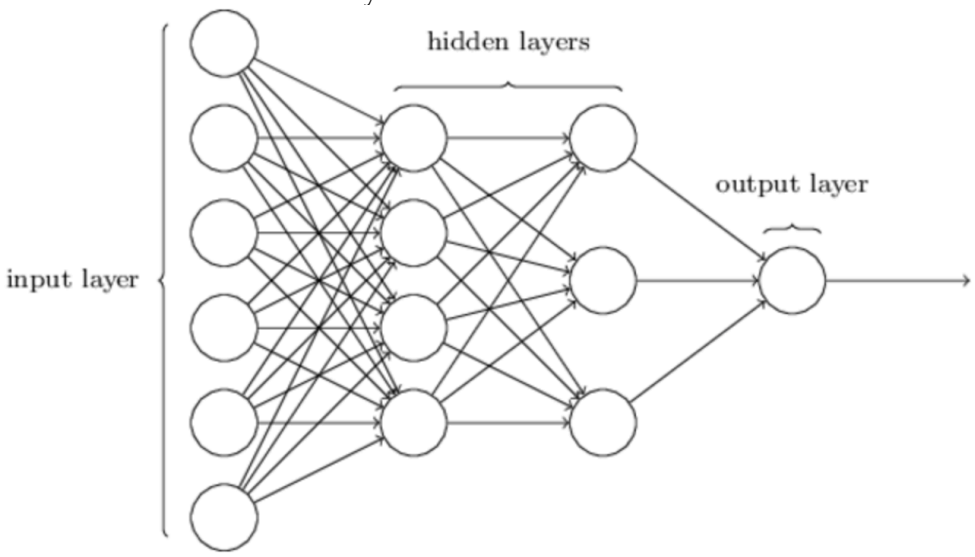


Figure 7: Neural Network

To avoid overfitting with training data, we adopted an early stopping method of setting stopping rounds at 3 and stopping metric to be "misclassification" with tolerance of 0.0005, which enabled training to stop as soon as the moving average of the cross validation misclassification did not improve by at least 0.05% for 3 consecutive scoring events.

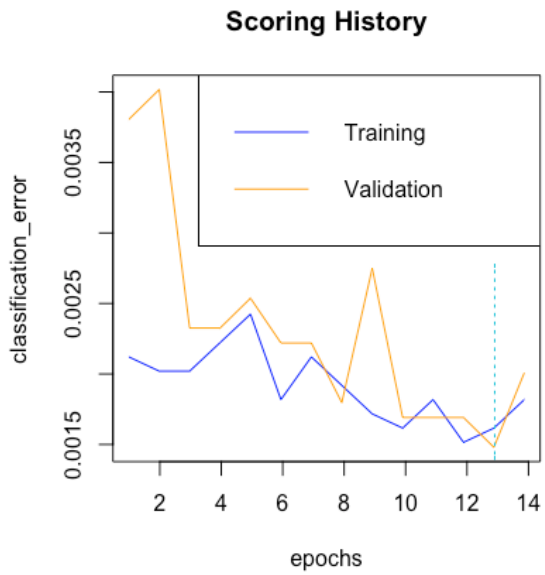


Figure 8: Classification Error by Epochs

The epoch stopped at the 15th iteration, and the best epoch is: 12.87.

4.4.2 Model Performance

1) Confusion Matrix

The confusion matrix of the model (setting the threshold as 0.02) shows as below. The true positive of the model is 70.3%(45/(45+19)).

		Predicted	
		0	1
Actual	0	18955	7
	1	19	45

2) AUC

The area under the ROC curve for the model is 0.93, which is much larger than random classifier.

4.5 Method 5: XGBoost

XGBoost (Extreme Gradient Boosting) is an optimized distributed gradient boosting algorithm, and it belongs to a family of boosting algorithms that convert weak learners into strong learners. A weak learner is one which is slightly better than random guessing. One important reason to choose XGBoost is the objective function including training loss and regularization. Regularization is a technique used to avoid overfitting in linear and tree-based models.

4.5.1 Working Principle

XGBoost can solve both regression and classification problems. It is enabled with separate methods to solve respective problems. In this project, XGBoost is applied to solve the fraud classifications. Specifically, XGBoost uses gbtree parameter so that each decision tree is built on the results obtained from previous trees to reduce misclassification rate. In other words, this strategy fixes what we have learned, and add one new tree at a time. Comparing with the boosting algorithms, the XGBoost capitalizes on the misclassification of previous model and tries to reduce the misclassification rate.
As we mentioned above, regularization is the key in this model and every parameter is crucial on the model's performance. Listed are the most important parameters.

XGBoost parameters can be divided into three categories:

- **General Parameters:** Controls the booster type in the model which eventually drives overall functioning
- **Booster Parameters:** Controls the performance of the selected booster
- **Learning Task Parameters:** Sets and evaluates the learning process of the booster from the given data

In our project, we followed the most common but effective steps in the parameter tuning:

- First, we built the XGBoost model using default parameters.
- Then, we fix **eta = 0.1**, leave the rest of the parameters at default value, using xgb.cv function get best **n_rounds**. Now, we build a model with these parameters and check the accuracy.
- Besides, we perform a grid search on rest of the parameters (**max_depth, gamma, subsample, colsample_bytree** etc) by fixing eta and nrounds.
- Last, using the best parameters from grid search, tune the regularization parameters(alpha, lambda) if required.

Although we have adjusted parameters by cross validation, we also implement some parameter like early_stopping_round to ensure the precision of our model and reduce overfitting.

4.5.2 Model Performance

1) Important Variables

Just like the random forest, XGBoost can also find out some important variables. The plot following shows the top 20 important variables in this project:

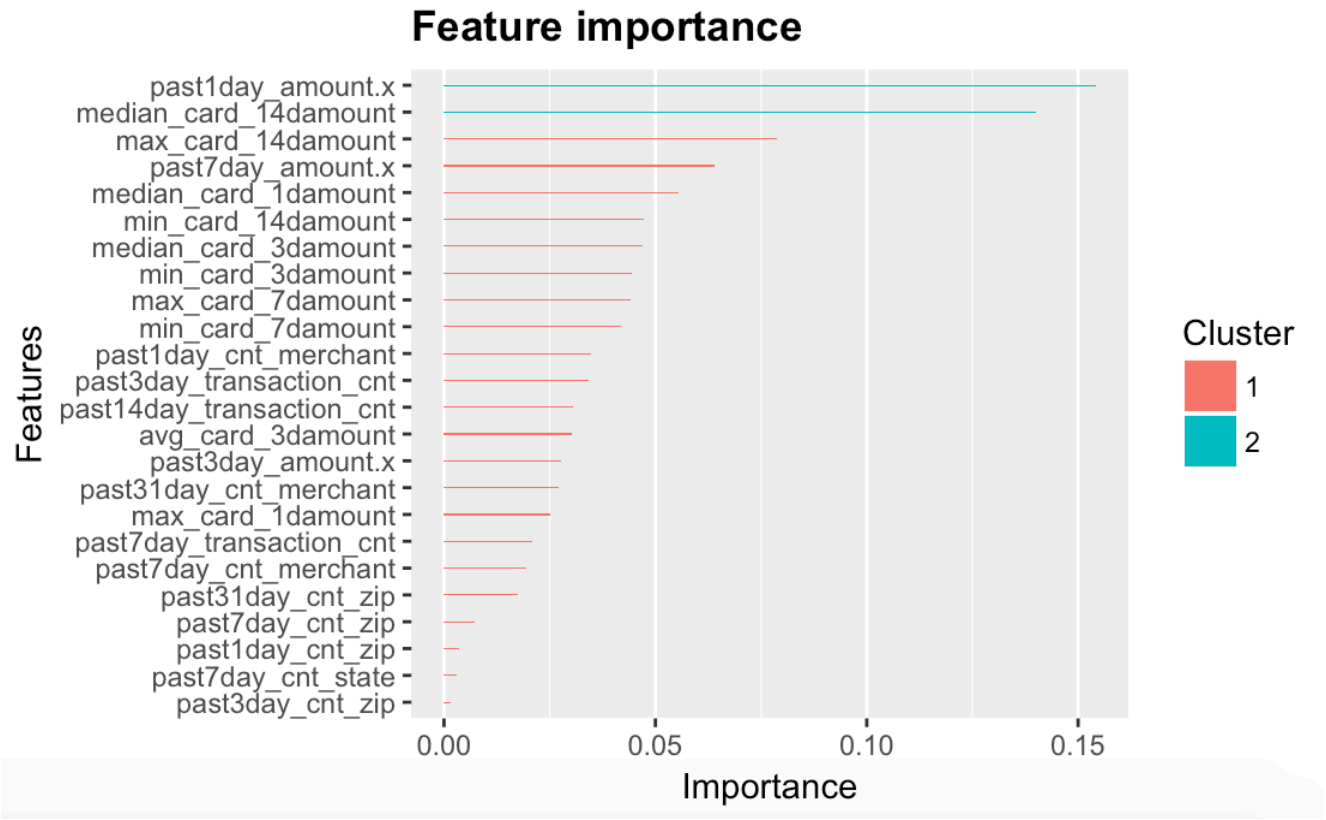


Figure 9: Top 20 important variables detected by XGBoost

2) Confusion Matrix

The true positive rate of the model is 81.25% (39/(39+9)).

		Predicted	
		0	1
Actual	0	18956	46
	1	9	39

5. Model Selection & Result

5.1 Model Performance Criterion

The model performance can be evaluated using different measures. For our project, we used Fraud Detection Rate, and Return on Investment (ROI) to measure our model performance.

5.1.1 Fraud Detection Rate

The fraud detection rate is the percentage of fraud cases detected by the model.
More specifically, it can be defined as:

$$\text{FDR} = \frac{\sum_{i=0.01}^{cutoff-bin} Fraud\ Cases}{Total\ Number\ of\ Fraud\ Cases} \times 100\%$$

where the cutoff-bin represents the top percentage of the population that labeled as fraud and rejected by the model. The possible values of cut-off bins are 0.01, 0.02, …., 0.99, 1. Fraud Cases represent number of cases that are true fraud within the boundary (cases that are predicted to be fraud).

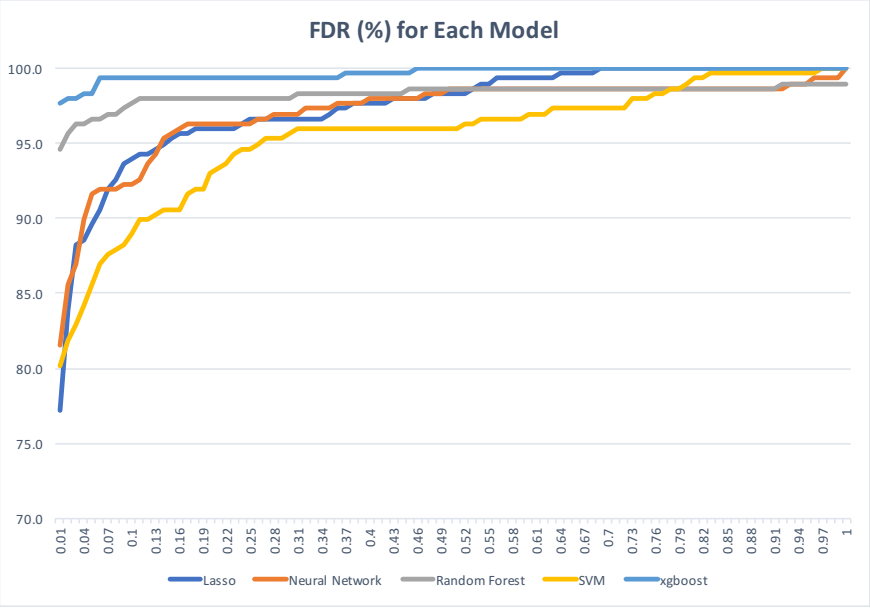


Figure 10: FDR by cut-off bin for each model

Among all the models , XGBoost yields much larger fraud detection rate. 97.7% of fraud can be caught at the top 1% of all the transactions in this data set.

5.1.2 ROI

Another metrics of the model performance evaluation is the Return On Investment (ROI). ROI measures the amount of return on an investment relative to the cost. ROI, in our case, is defined as:

ROI = fraud savings - lost sales - cost of model
= bad cases caught * lost for a fraud - good cases wrongly classified * penalty for a flagged good - cost per score * number of total cases

Likewise, bad cases and good cases refer to the cumulative cases within the cutoff boundary.

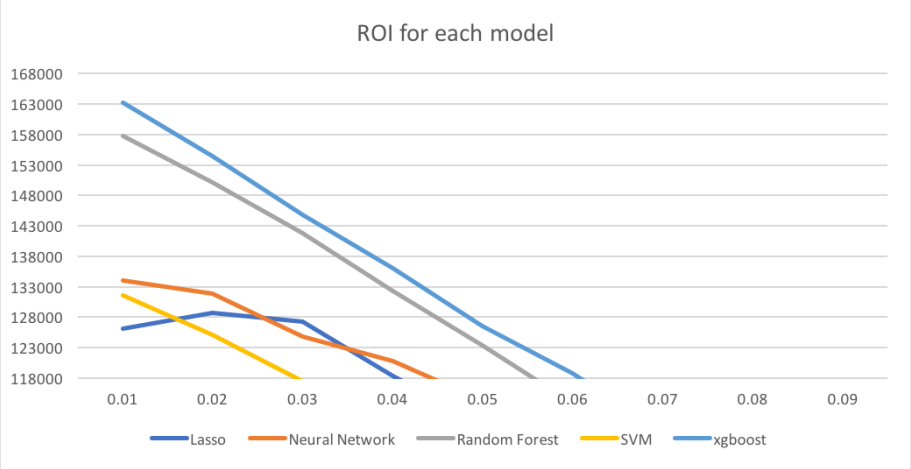


Figure 11: ROI by cut-off bin for each model

From the line in Figure 11, the XGBoost model (light blue line) outperforms other models in terms of ROI assessment.

5.2 Results

In conclusion, XGBoost performs better with high values of FDR and ROI. We choose to label top 1% suspicious cases as fraud and reject those transactions. Specifically, a higher fraud score (100 * the probability of the case is fraudulent regarding to XGBoost model) indicates a more suspicious case. The histogram below shows the fraud score distribution, the top 1% cases are within the rightmost bin.

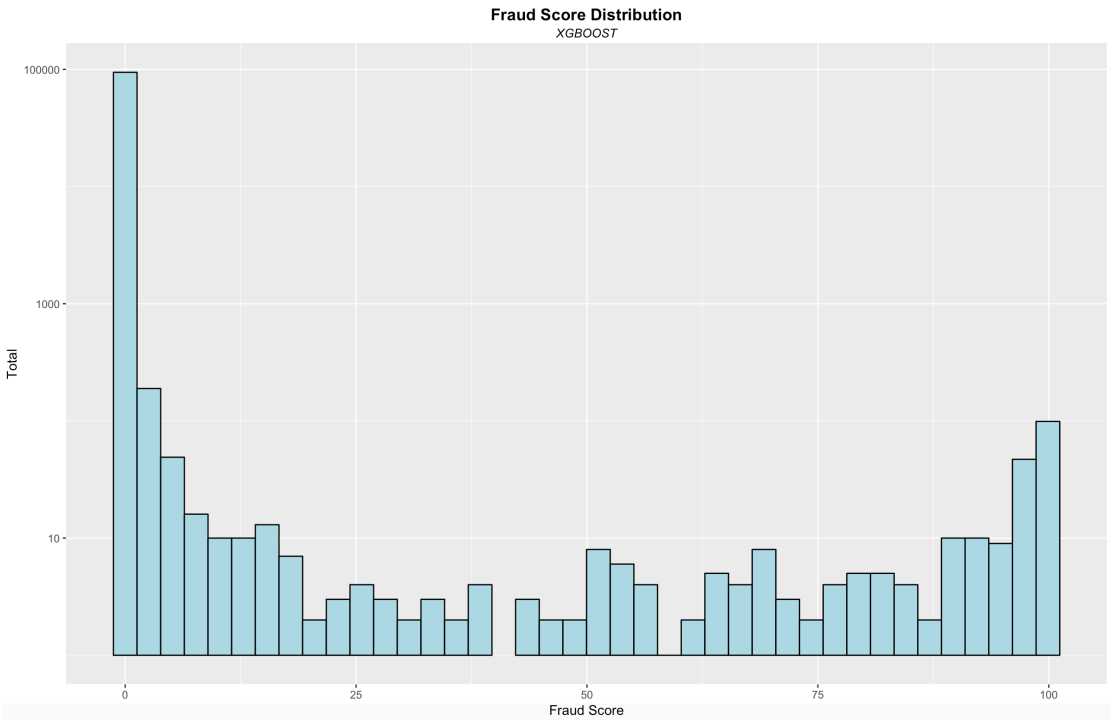


Figure 11: Fraud Score distribution for XGBoost

Table 7 XGBoost Performance Table

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	False Pos. Ratio*	False Pos. Rate**(%)	Population Bin	Fraud Savings	Lost Sales	ROI
0.01	950	659	291	0.70	97.65	659	291	0.7	97.7	97.0	2.3	69.37	0.01	$174,600	$6,590	$163,260
0.02	950	949	1	1.00	0.34	1,608	292	1.7	98.0	96.3	5.5	84.63	0.02	$175,200	$16,080	$154,370
0.03	950	950	0	1.00	0.00	2,558	292	2.7	98.0	95.3	8.8	89.75	0.03	$175,200	$25,580	$144,870
0.04	950	949	1	1.00	0.34	3,507	293	3.7	98.3	94.6	12.0	92.29	0.04	$175,800	$35,070	$135,980
0.05	950	950	0	1.00	0.00	4,457	293	4.7	98.3	93.6	15.2	93.83	0.05	$175,800	$44,570	$126,480
0.06	950	947	3	1.00	1.01	5,404	296	5.7	99.3	93.6	18.3	94.81	0.06	$177,600	$54,040	$118,810
0.07	950	950	0	1.00	0.00	6,354	296	6.7	99.3	92.6	21.5	95.55	0.07	$177,600	$63,540	$109,310
0.08	950	950	0	1.00	0.00	7,304	296	7.7	99.3	91.6	24.7	96.11	0.08	$177,600	$73,040	$99,810
0.09	950	950	0	1.00	0.00	8,254	296	8.7	99.3	90.6	27.9	96.54	0.09	$177,600	$82,540	$90,310
0.1	950	950	0	1.00	0.00	9,204	296	9.7	99.3	89.6	31.1	96.88	0.1	$177,600	$92,040	$80,810

As shown in the table 7, XGBoost is our best model that has captured 97.7% of fraud records at the top 1% population and the expected return on investment is $163,360.

5.2.1 Fraud Cases Review

Table 8 Fraud Cases Review

record #	cardnum	date	merchnum	merch description	merch state	merch zip	amount	XGB oost
89249	5142199009	12/3/10	4353000719908	ACI*AMAZON.COM INC	WA	98101	306.66	
89250	5142199009	12/3/10	4353000719908	ACI*AMAZON.COM INC	WA	98101	195.42	
29125	5142151962	4/23/10	6929	HILLCREST MOTEL & REST	VA	22485	956.62	√
29126	5142151962	4/23/10	6929	HILLCREST MOTEL & REST	VA	22485	463.54	√
23825	5142205500	4/2/10	5000006000095	IBM INTERNET 01000025	NY	NA	985	
31813	5142205500	5/2/10	5000006000095	IBM INTERNET 01000025	NY	NA	985	√

Our XGBoost has captured three of these six records shown above which are tricky to be detected. Compared with other records, all the detected records have unusual values comparing to the other records in the top important variables in XGBoost. For example, the value of the variables, such as past1day_amount.x of record 29125, the median_card_14damount of record 29125 and 29126, is way beyond the average amount in the top important variable plot from XGBoost. However, to protect overfitting of the model, we have applied some important parameters like, early_stopping, cost and gamma. As a result, these parameters increase the bias in the model, thus not all the six records are detected.

The following are some subsets of our final detection result. Frequent transactions with relatively large amount associated with identical card number or merchant number are more likely to be detected. Record 39343 turns out to be a false positive, mainly because the transaction amount associated with this particular card and merchant is far from the average amount in the near term. Record 88955, 68533, and 83129 share the same card number, merchant and transaction amount, and their transaction dates are exactly one month from each other. This particular pattern is also flagged as fraud.

recor	cardnum	date	merchnum	merch description	merch sta	merch	transty	amount	fra	score	cum all
39308	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1875.86	1	99.909651	0.000315766
39309	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1859.97	1	99.909651	0.000326292
39310	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1849.08	1	99.901140	0.000452598
39311	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1765.83	1	99.889427	0.000484175
39307	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1958.74	1	99.844474	0.000547328
39315	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1647.97	1	99.799758	0.000568379
39336	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1985.94	1	99.799758	0.000578905
39337	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1968.05	1	99.799758	0.00058943
39306	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1968.54	1	99.688679	0.000810467
39338	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1875.94	1	99.579537	0.000894671
39314	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1675.79	1	99.386036	0.00106308
39163	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1697.09	1	99.354666	0.001073605
39305	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1986.95	1	99.340302	0.001084131
39164	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1648.98	1	99.169004	0.001199912
39312	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1764.97	1	98.406994	0.001357795
39162	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1698.47	1	97.325122	0.001515678
39313	5142116864	5/28/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1695.07	1	96.622145	0.001599882
39161	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1748.07	1	96.130490	0.00165251
39339	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1847.96	1	93.458772	0.001736714
39340	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1807.95	1	93.458772	0.00174724
39341	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1785.85	1	86.072046	0.002041955
39342	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1674.97	1	74.568778	0.002199838
39157	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1904.83	1	59.444994	0.002305093
39160	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1748.86	1	51.975393	0.002347195
39159	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1859.48	1	11.579352	0.002968202
39343	5142116864	5/29/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	9.29	0	11.359159	0.002989253
39158	5142116864	5/27/2010	4.353E+12	AMAZON.COM *SUPERSTOR	WA	98101	P	1895.07	1	4.572704	0.003368173

recor	cardnum	date	merchnum	merch description	merch sta	merch	transty	amount	fra	score	cum all
88955	5142205500	12/2/2010	5.00001E+12	IBM INTERNET 01000025	NY	NA	P	985	1	29.833648	0.002610334
66465	5142205500	8/28/2010	6.88101E+12	CASTLE NAVIGATION INC.	CA	92887	P	1889.9	0	12.777863	0.0029261
26469	5142205500	4/12/2010	9.60801E+12	KENNEDY OFFICE SUPPLY CO.	NC	27604	P	87.37	0	6.628592	0.003168188
72143	5142205500	9/14/2010	6.829E+11	THE FINAL CUT	NJ	7712	P	1983.11	0	4.879718	0.003294494
68533	5142205500	9/2/2010	5.00001E+12	IBM INTERNET 01000025	NY	NA	P	985	1	4.634847	0.003347122
83129	5142205500	11/2/2010	5.00001E+12	IBM INTERNET 01000025	NY	NA	P	985	1	4.562730	0.003378698
61406	5142205500	8/11/2010	6.829E+11	THE FINAL CUT	NJ	7712	P	70.75	0	2.911746	0.003652362
25755	5142205500	4/10/2010	NA	MICRON GOVERNMENT SYS INC	ID	83542	P	2499	0	1.131749	0.004641763

6. Future Work

Currently, our models and test performance are exercised based solely on the limited information from this card payment data set. Over the 95,007 records, there are only 298 transactions are labeled as fraud, which is relatively too small. Also, the fraud score distribution from XGboost indicates a bimodal distribution due to the intrinsic principle of manually generating fraud. As a result, some of our models have suffered from potential overfitting problems. We suggest that we could collect more labeled fraud transactions to our model so that we could avoid the problem of overfitting in the future.

In addition, model performance could be improved by creating more decent and useful variables, such as geographic variables. Specifically, there are lots of mismatched information between merchant zipcode and merchant state. For example, for record #19407, zipcode 12983 corresponds to Saranac Lake, NY in reality, but the data shows the merchant is in KS. Since the geographic variable is a beneficial variable to detect fraud, we could built a more decent and efficient model if we have more accurate zipcode data and more valuable variables like distance between transaction coordinate and past average coordinate.

References

Random Forest:

https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

SVM:

https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Classification/SVM

https://rpubs.com/ryankelly/svm

Neural Network:

http://www.ijsce.org/attachments/File/NCAI2011/IJSCE_NCAI2011_025.pdf

XGBoost:

http://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/

Appendix

Data Quality Report

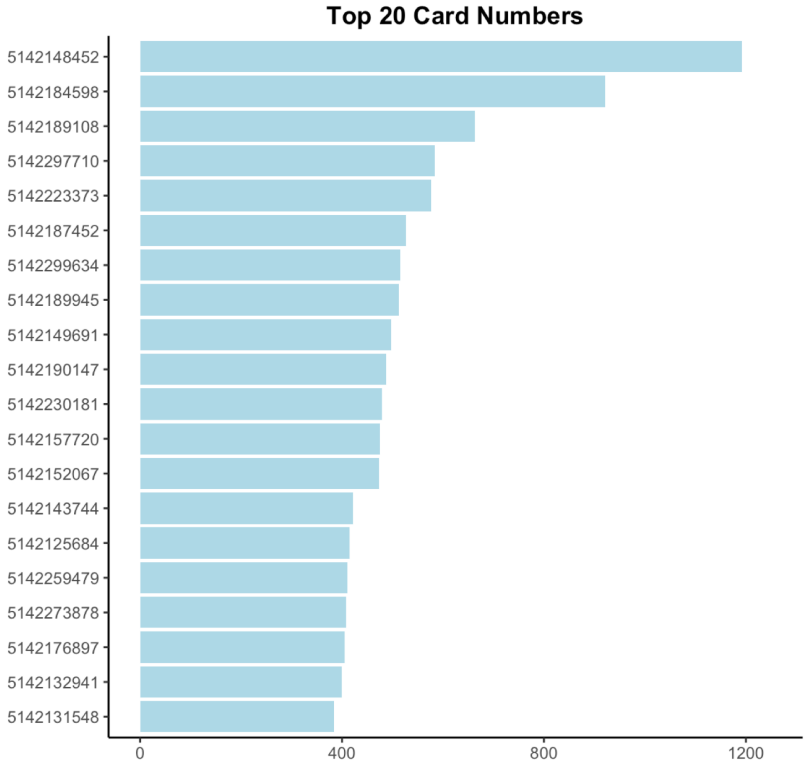
File Description

This file called Card payments.xlsx provides transaction information of card payment from 01/01/2010 to 12/31/2010. It consists of 95,007 records and 10 variables in total.

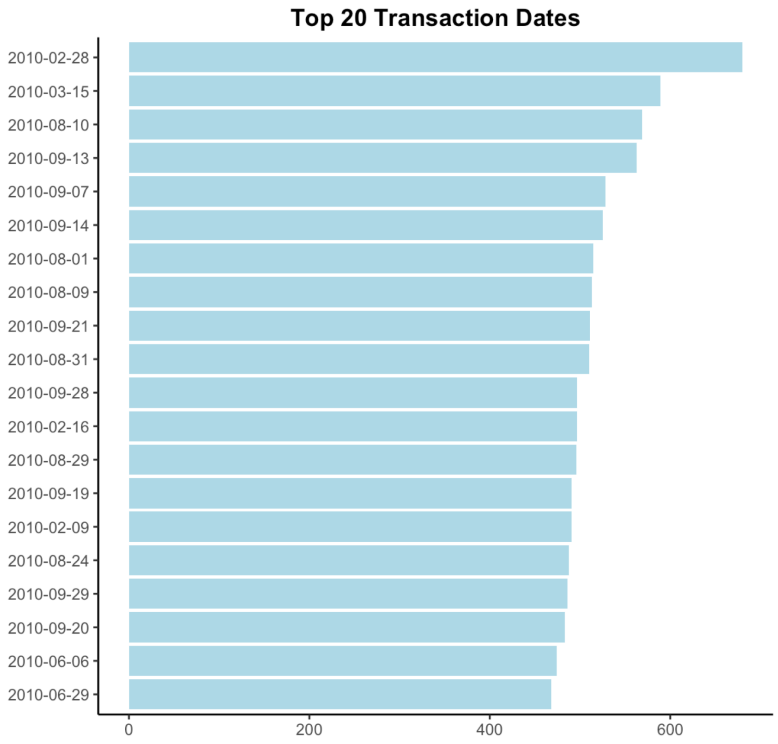
List of Info for Each Field

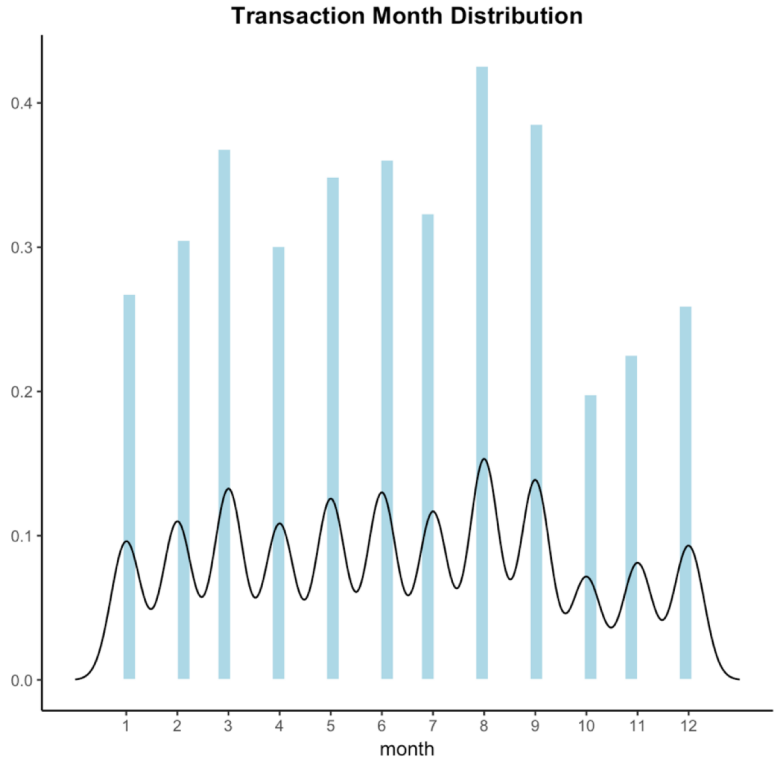
1. **recordnum** (categorical): This field shows the serial number of each record, and 100% populated.

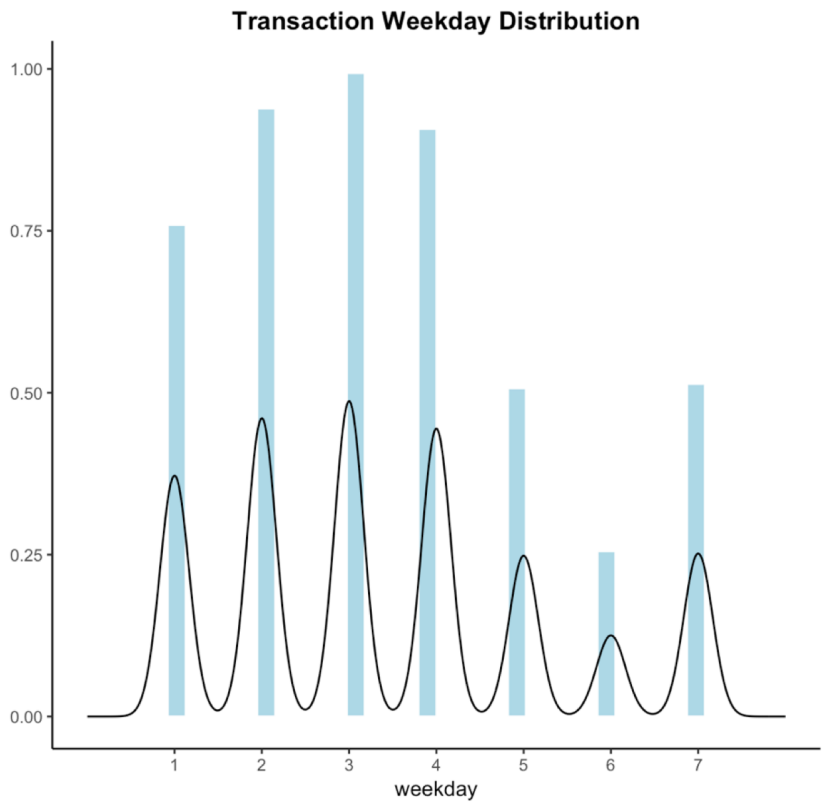
2. **cardnum** (categorical): This filed gives the card number for each payment, and 100% populated with 1,634 unique values. The card number **5142148452** appears the most time.



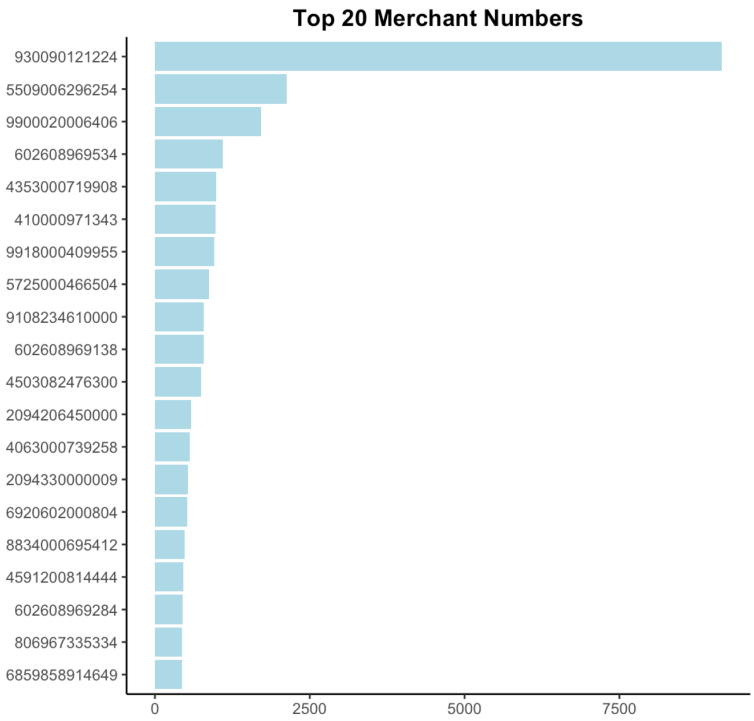
3. **date** (categorical): This field shows the date for each payment, and 100% populated with 365 unique values. The date **2010-02-28** appears the most time.


Top 20 Transaction Dates
2010-02-28
2010-03-15
2010-08-10
2010-09-13
2010-09-07
2010-09-14
2010-08-01
2010-08-09
2010-09-21
2010-08-31
2010-09-28
2010-02-16
2010-08-29
2010-09-19
2010-02-09
2010-08-24
2010-09-29
2010-09-20
2010-06-06
2010-06-29
0
200
400
600

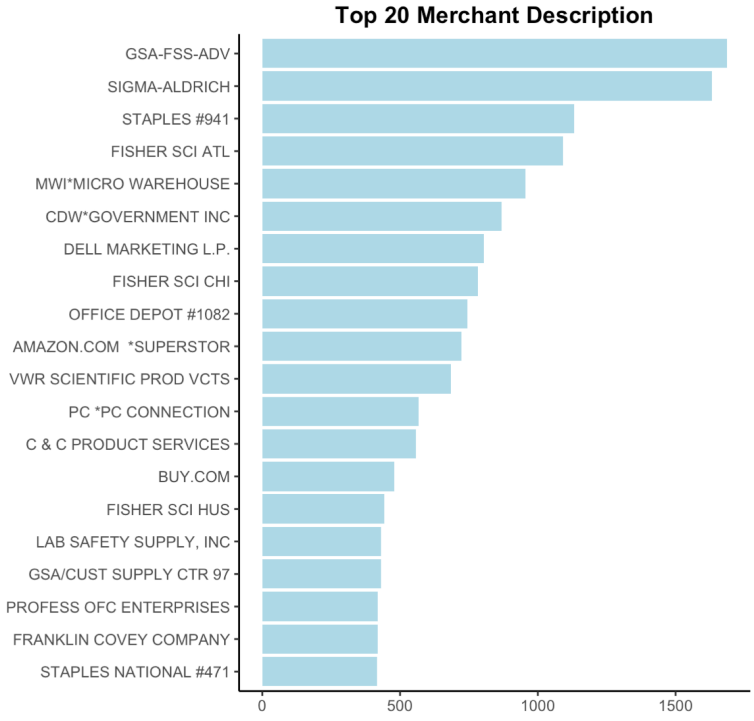

Transaction Month Distribution
0.4
0.3
0.2
0.1
0.0
1
2
3
4
5
6
7
8
9
10
11
12
month



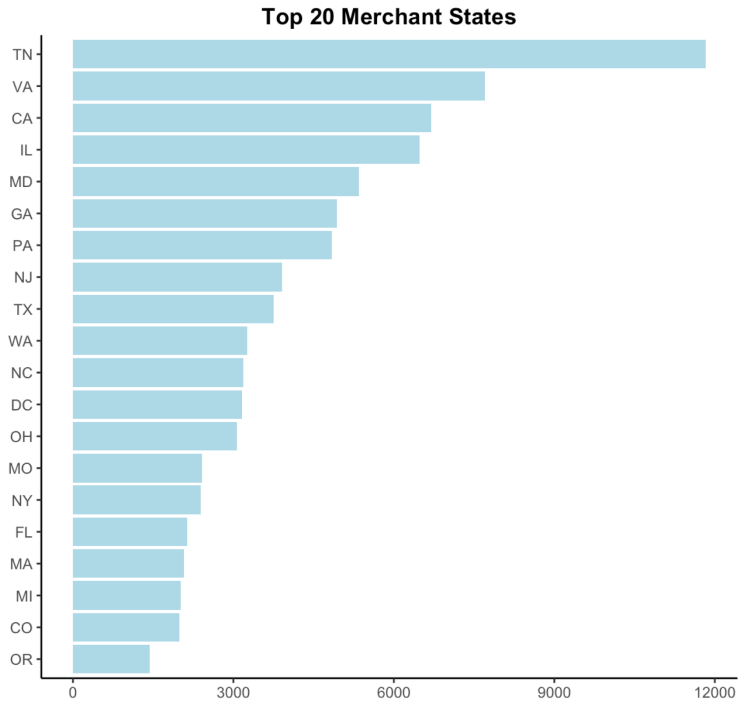
4. **merchnum** (categorical): This field provides merchant number for each payment, and 96.66% populated (3,174 missing values) with 13,088 unique values. The merchant number **930090121224** appears the most time.



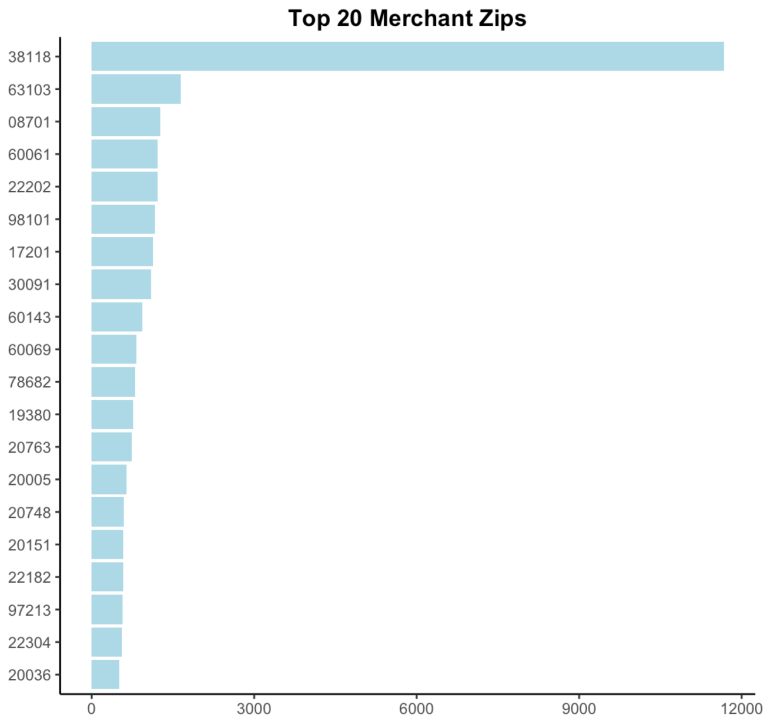
5. **merch description** (categorical): This field gives merchant description for each record, and 100% populated with 12,964 unique values. The description **GSA-FSS-ADV** appears the most time.



6. **merch state** (categorical): This field indicates the state where each merchant was located. It is 98.93% populated (1,016 missing values) and has 60 unique values. The state **TN** appears the most time.



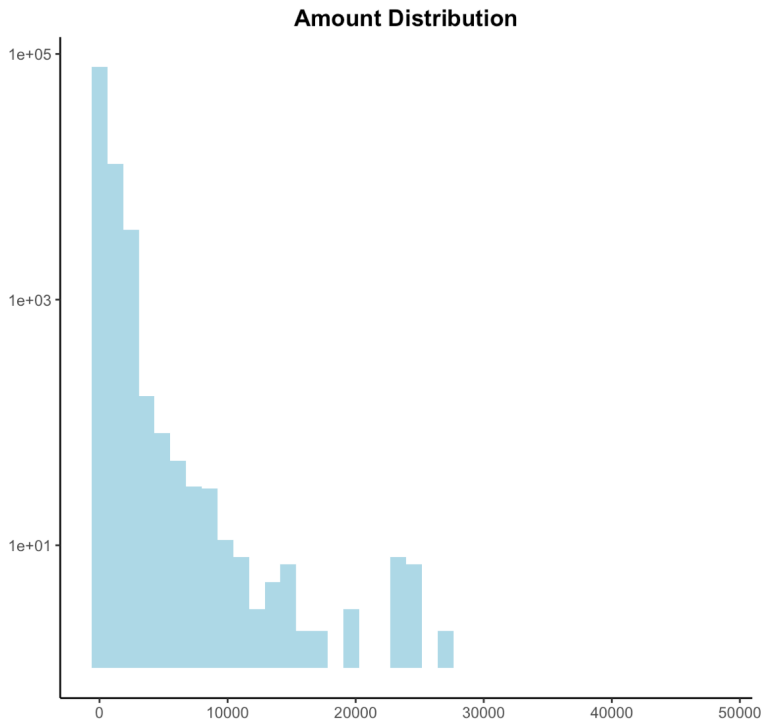
7. **merch zip** (categorical): This field provides merchant zip code for each payment. It is 95.49% populated (4,283 missing values) with 4,583 unique values. The zip code **38118** appears the most time.



8. **transtype** (categorical): This field shows the transaction type for each card payment, and 100% populated with only one unique value **P**. P refers to Prenotification payment that can be used to verify bank account validity.

9. **amount** (numeric): The field provides the transaction amount for each card payment, and 100% populated with 34,075 unique values.

Min	Max	Mean	Median	Mode	Standard Deviation
0.01	47900.00	381.00	136.50	3.62	758.83



10. **fraud** (categorical): This field indicates whether each record is a fraud. It is 3.1% populated, and 298 records are considered as fraud (shown as 1).