**A pair of feet on a desk

Description generated with high confidence**

**Credit Card Transaction Fraud**

**May 2018 - Supervised Fraud Algorithm Report**

- Modan Wang

- Zhengli Zhang

- Raman Kumar

**Executive Summary**

As a financial institution, credit card transactions contain the core of your business and fraudulent transactions could causes both financial and reputational damage to your business. Based on our research the number of fraudulent transactions will grow in the next decades by factor of 2, and they will be more sophisticated. These trends call for a systematic method to identify and flag these fraudulent transactions. We started our process with in depth data quality report, in which, we draw our insights to build the most impactful expert variables. Furthermore, we used K-S test to reduce dimensionality and simplify our model using the most significant variables. Our team was able to build a model using the selected variables to identify these doubtful transactions in timely manner. Our research was based on the supervised data provided which included few missing values which we addressed before creating the models. In our pursuit of the best model, we experimented with Decision Tree, Random Forest, Logistic Regression, Gradient Boosting Tree, and SVM. We concluded that “Random Forest” was the best model which provided 74.55% fraud detection in OOT (out of Time) period.

Based on our data, we would be able to save the company $220K in just OOT (Out of time) window which equates to $1.320 million dollars of saving. We recommend rejecting transaction below 2% threshold to avoid loss.

**Data Description**

File Description:

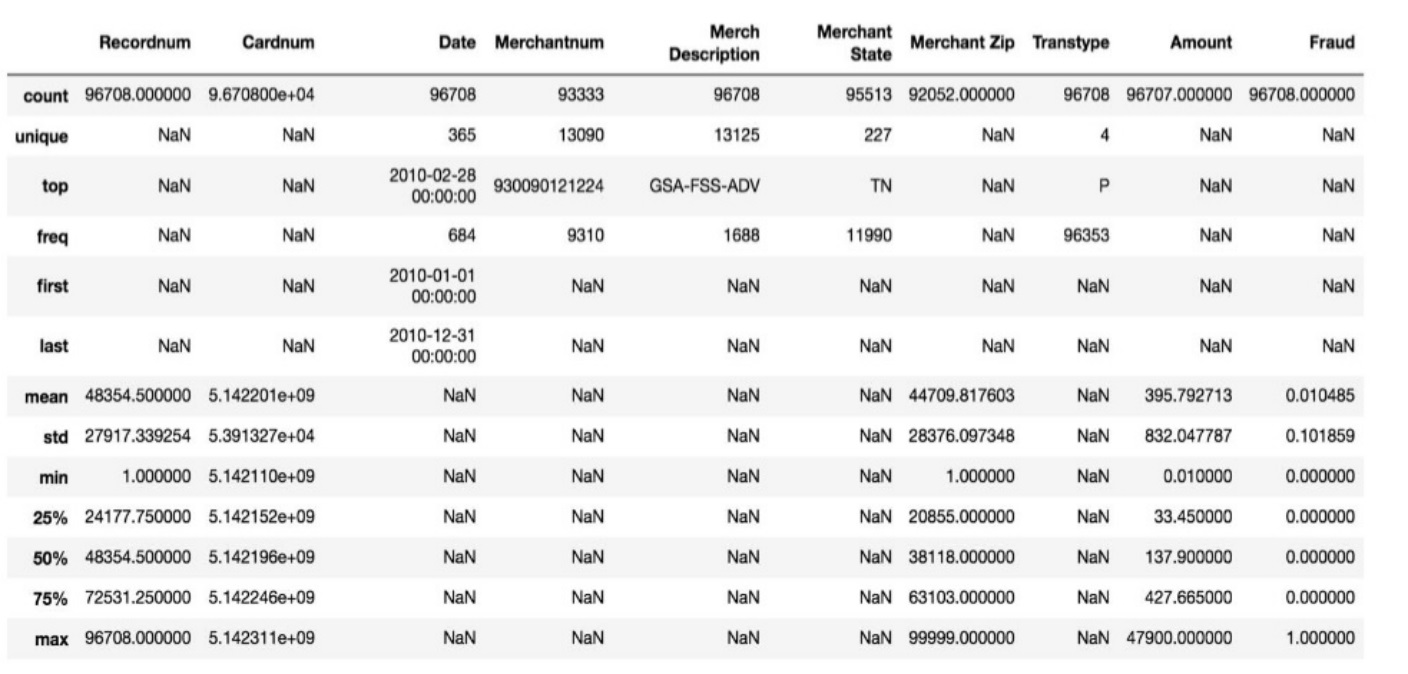
The dataset is called ‘card transactions’ which contains 96,708 records and 10 variables. It includes information about the record number, card number, the date the card transactions made, merchant number, merchant description, merchant state, merchant zip code, transaction type, transaction amount and fraud condition.

File Name: card transactions.xlsx

Data Source: This dataset is the simulated data based on real card transactions records.

Number of Records**:** 96,708 records

Number of Fields: 10 variables in total:

****

|  |  |
| --- | --- |
| **Field Name** | **Data Type** |
| Recordnum | Numerical variable |
| Cardnum | Numerical variable |
| Date | Date variable |
| Merchantnum | Numerical variable |
| Merch Description | Text variable |
| Merchant State | Categorical variable |
| Merchant Zip | Numerical variable |
| Transtype | Categorical variable |
| Amount | Numerical variable |
| Fraud | Categorical variable |

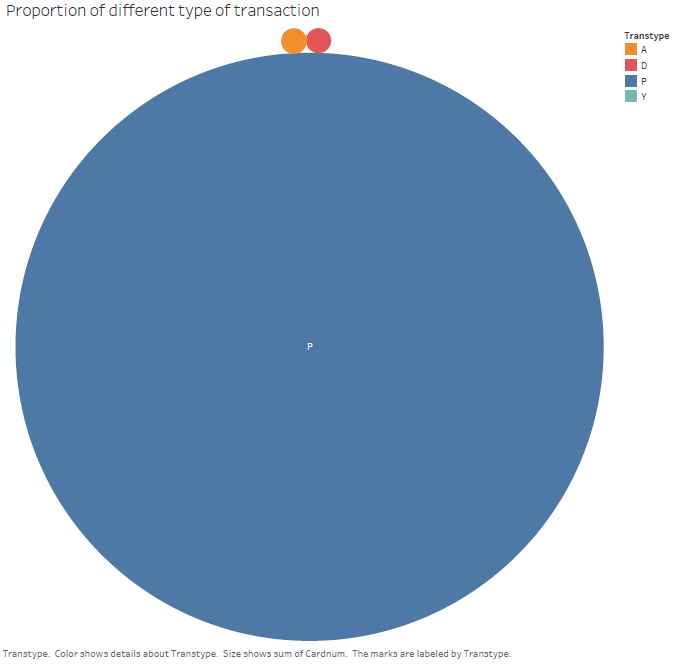
**Data Cleaning**

After exploring our dataset, we focused on addressing the missing values and out of character elements. There are three variables with missing values: Merchant Number, Merchant Zip Codes, and Merchant State. The below table summarizes our treatment of these missing values:

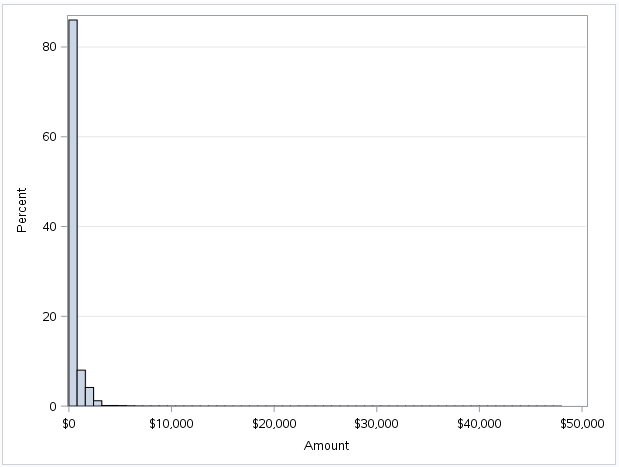
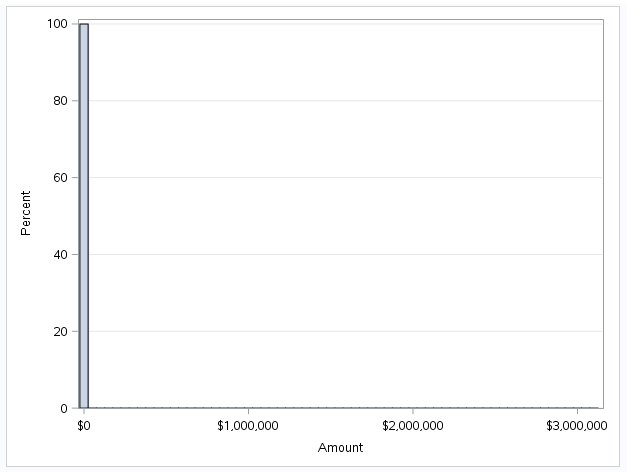
|  |  |
| --- | --- |
| Name of the variable | Treatment of the missing variables |
| Merchant Number | In order to create unique number, we assigned number from 1 to 3375. Every identified missing merchantnum got a unique number in this range. |
| Merchant Zip | Replace with “Blank” |
| Merchant State | Replace with “Blank” |

\* It is important to note that we cannot replace the missing “Merchantnum” with “Blank” because we use this variable to link and group and it would create a problem going forward. Therefore, we need to make this a unique number.

- After discussion with industry experts, we decided to work only with P type transaction and remove any other type (A, Y, and D). Here is the proportion of transaction P to other types of transactions.



Furthermore, we noticed an out of character “Amount” done by “INTERMEXICO” Merchant. We removed the transaction which had really high transaction amount. There is a mistake in one of the entry’s in which amount is noted as “$3,102,045.53” This is a mistake in entering the dollar amount instead of foreign value. As seen here data is disproportionately distributed because of that value. After removing the outlier, the histogram changes to:



**Expert variables**

We built 80 variables using RSQLite and did linkage analysis mainly focus on 3 variables in the original data: Cardnum, Merchantnum and MerchantDescription. We focused on different combinations of other variables like Merchant Zip, Fraud, Amount with these core variables and looked at how many times the same of different variables have shown up within past 1,3,7,14,30 days. For the records in the first few days, we replaced it with the average amount given the same core variables. All of our records were constructed to look only into the past data. Then we used K-S test to select further good variables.

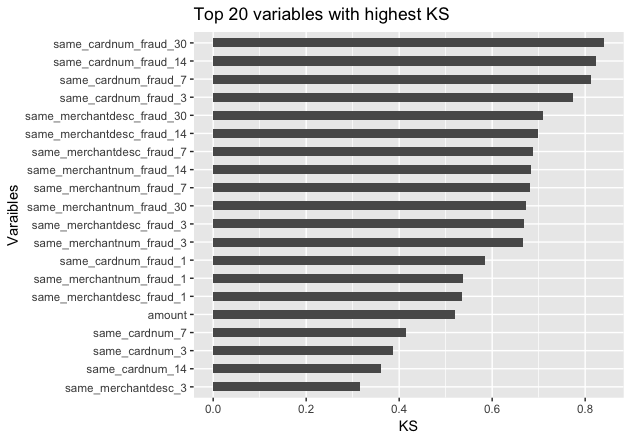
**Feature Selection**:

In order to increase the learning accuracy and reduce complexity, we proceed with feature selection process to select the most impactful variables and eliminate the rest. Most widely used and generally accepted method is the univariate K-S Test. The K-S test enables us to observe the distribution of each variable between goods and bads. K-S test allows us to test for differences in the shape of two sample distribution. The null hypothesis is that two samples drawn from populations should have same cumulative distribution functions. If null hypothesis is true, the defined difference should be zero and two sample follow the same distribution. The alternative hypothesis indicates that the data do not follow the same distribution. We used K-S test in R to realize this calculation and get the KS statistics for each variable. We finally decided to select top 20 variables with highest KS statistics (shown in the graph below).

In regard to feature selection, we decided to choose 20 variables for two reasons:

1) The effectiveness of the variables after the 20Th variables were much lower in comparison to the other variables

2) We decided to use less variables to avoid unnecessary complexity since we are dealing with small dataset.

****

Then we took the remaining variables and passed through a forward stepwise linear regression algorithm which helped us further reduce the candidate variables. The stepwise regression allows us to identify and remove the highly correlated variables. Therefore, stepwise will choose the uncorrelated variables which ensures the dimensionality reduction happens with minimum loss of important data. We select number of features based on adjusted Cp, which is minimum when we chose 14 variables in our case. The final 14 variables used for modeling and their descriptions are listed here:

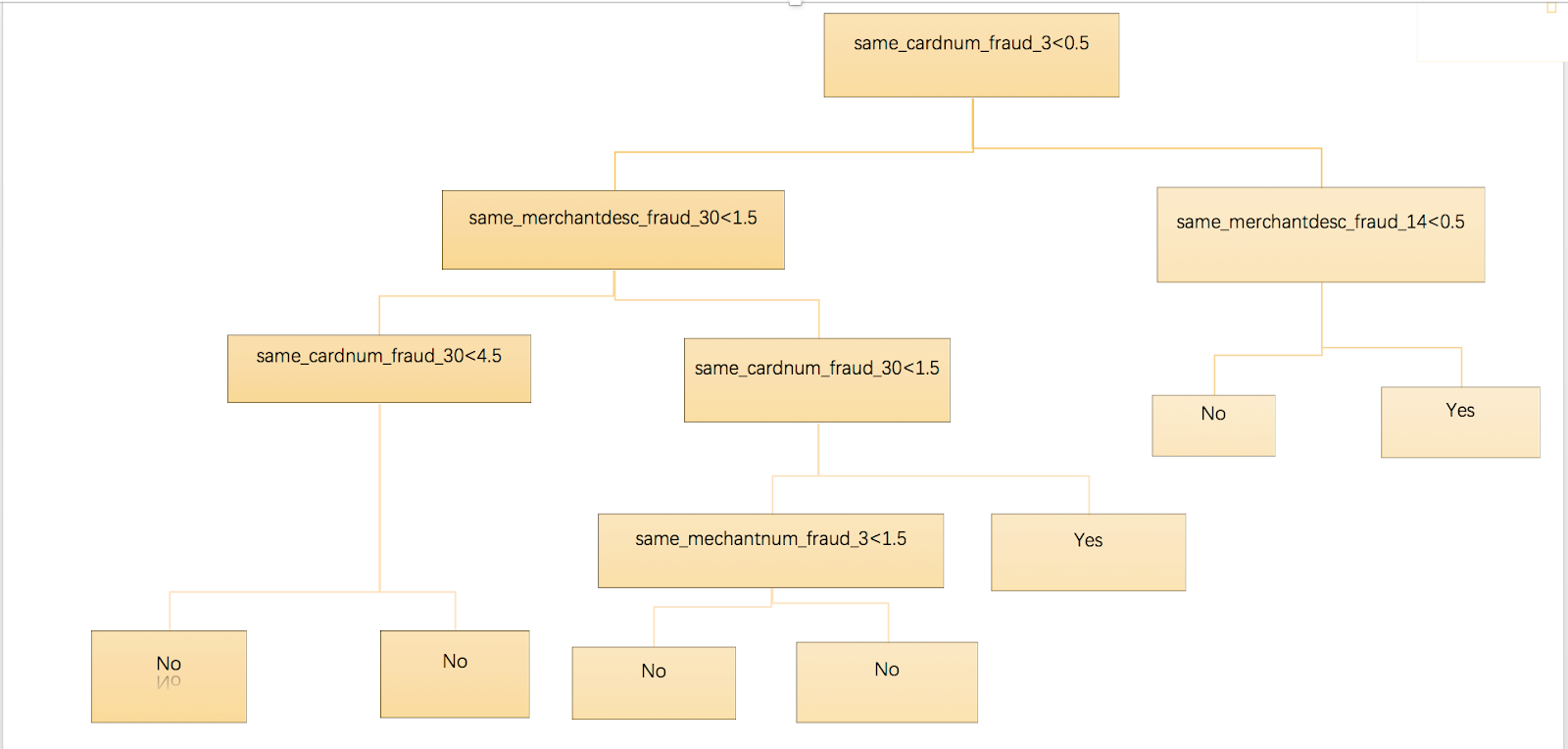
|  |  |
| --- | --- |
| Variables | Description |
| same\_cardnum\_fraud\_30 | In the past 30 days, the count of fraud found for the same card number |
| same\_cardnum\_fraud\_14 | In the past 14 days, the count of fraud found for the same card number |
| same\_cardnum\_fraud\_7 | In the past 7 days, the count of fraud found for the same card number |
| same\_cardnum\_fraud\_3 | In the past 3 days, the count of fraud found for the same card number |
| same\_merchantnum\_fraud\_3 | In the past 3 days, the count of fraud found for the merchant number |
| same\_merchantdesc\_fraud\_30 | In the past 30 days, the count of fraud found for the merchant description |
| same\_merchantdesc\_fraud\_14 | In the past 14 days, the count of fraud found for the merchant description |
| same\_merchantdesc\_fraud\_7 | In the past 7 days, the count of fraud found for the merchant description |
| same\_merchantdesc\_fraud\_3 | In the past 3 days, the count of fraud found for the merchant description |
| same\_cardnum\_fraud\_1 | In the past 1 days, the count of fraud found for the same card number |
| same\_merchantdesc\_fraud\_1 | In the past 1 days, the count of fraud found for the merchant description |
| same\_cardnum\_7 | In the past 7 days, the count of same card number |
| same\_cardnum\_3 | In the past 3 days, the count of same card number |
| same\_merchantdesc\_3 | In the past 3 days, the count of same merchant description |

**Models**

In our analysis, we utilized 5 different models to be able to predict fraudulent transaction in the OOT segment of the data. These models are Decision Trees, Random Forest, Logistic Regression, Gradient Boosting Tree, and SVM. The first 10 months were used to train and test our models, and the last 2 months data was used to test the model in an environment which it has never seen before.

**1) Decision Trees:**

Decision trees divides the dataset into smaller and smaller data sets according to features introduced in the analysis until small enough set is achieved which describes only one feature. We trained the decision tree on the training dataset. The decision tree in our model used 5 variables: In this case, we used same\_cardnum\_fraud\_3, same\_merchantdesc\_fraud\_30, same\_cardnum\_fraud\_30 same\_merchantnum\_fraud\_3, same\_merchantdesc\_fraud\_14. The trained tree has 7 terminal nodes. Below is an illustration of the decision tree:



**FDR:** Combined the predicted probability to be fraud with label of ‘Yes’ and reorder from high to low according probability:

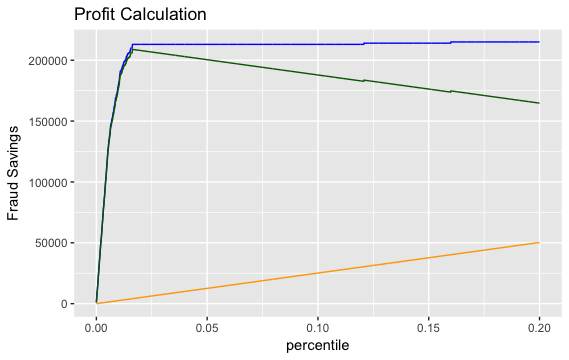
Then calculated FDR of top 2% records on training data is 0.9314159; on test data is 0.9285714; and on OOT data is 0.6449704.

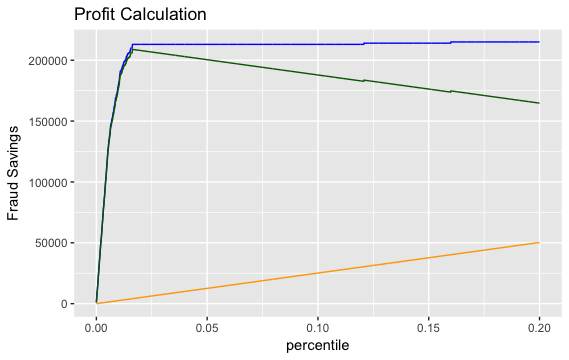
**Confusion Matrix and Statistics**:

|  |  |  |
| --- | --- | --- |
| Train | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 58,166 | 169 |
| Actual:  YES | 20 | 283 |

|  |  |  |
| --- | --- | --- |
| Test | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 24,901 | 96 |
| Actual:  YES | 6 | 128 |

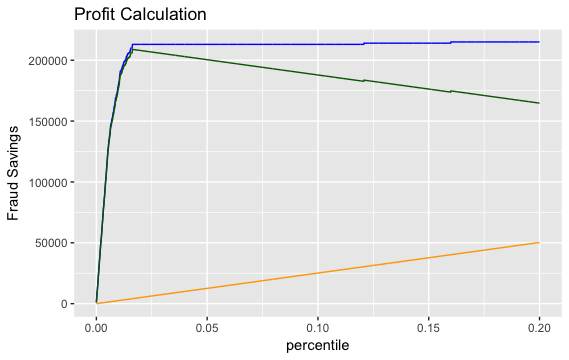
|  |  |  |
| --- | --- | --- |
| OOT | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 12,232 | 162 |
| Actual:  YES | 15 | 176 |

**ROI Curve:**



Train Datasets

Test Datasets



OOT Datasets

**2) Random Forest:**

The decision trees suffer from high variance, so we try the other advanced tree model: random forest. We built the random forest tree in R on the training data and use the trained model to predict the label for each record in the testing data.

**FDR**: Combined the predicted probability to be fraud with label of ‘Yes’ and reorder from high to low according probability:

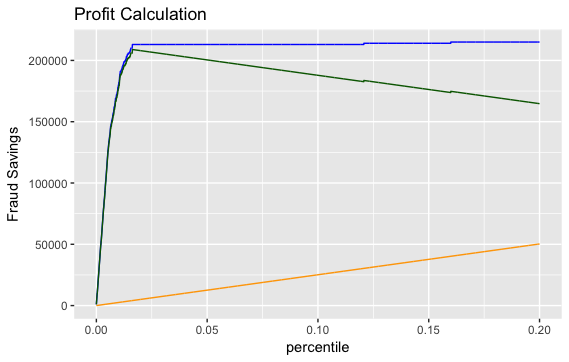
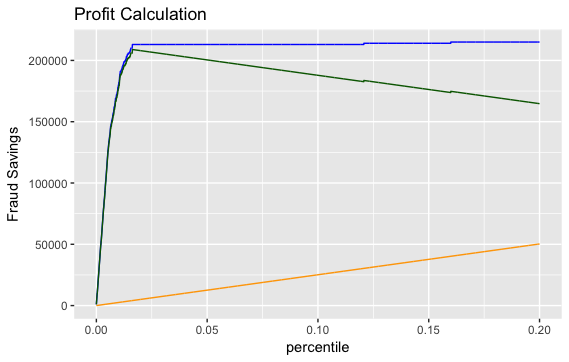
Then calculated FDR of top 2% records on training data is 0.9535398; on test data is 0.9642857; and on OOT data is 0.7455621.

**Confusion Matrix and Statistics**:

|  |  |  |
| --- | --- | --- |
| Train | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 58,186 | 37 |
| Actual:  YES | 0 | 415 |

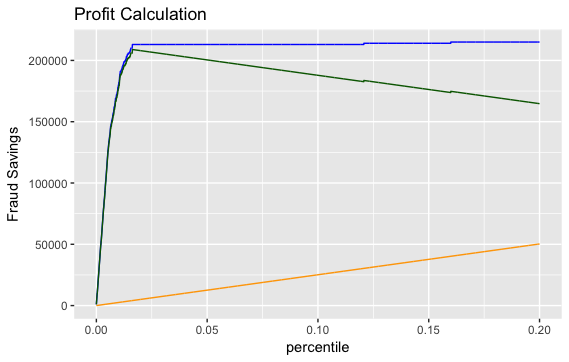
|  |  |  |
| --- | --- | --- |
| Test | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 24,895 | 41 |
| Actual:  YES | 12 | 183 |

|  |  |  |
| --- | --- | --- |
| OOT | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 12,227 | 49 |
| Actual:  YES | 20 | 289 |

**ROI Curve:**

Test Datasets

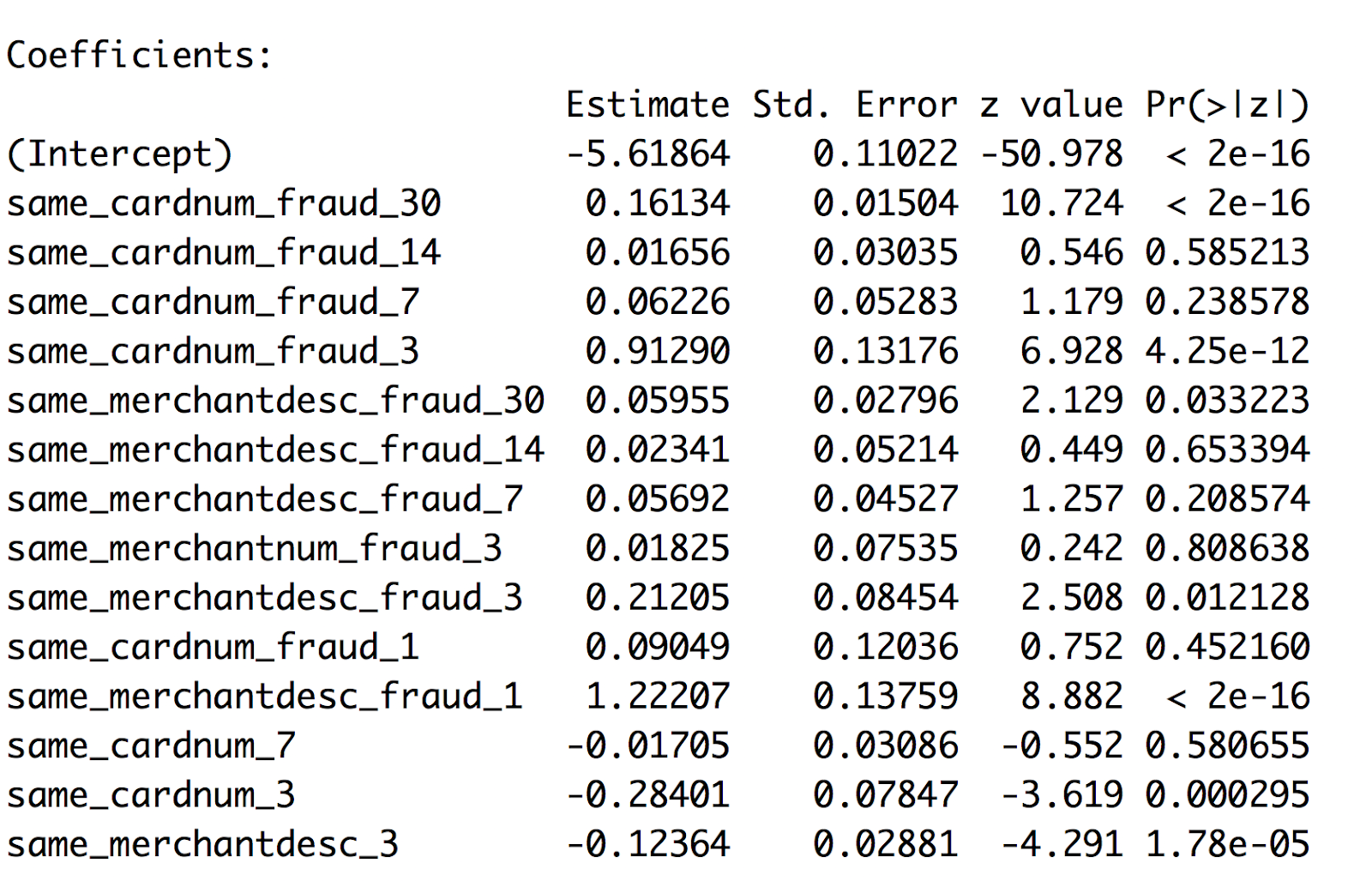
Train Datasets



OOT Datasets

**3) Logistic Regression**:

In our search for best model, we applied the logistic regression model using our training dataset to predict the probability of fraud. We use the predict () function and set the type option be to "response" to tell R to output probabilities of the form P (Y = 1|X). Then we set the threshold is 0.5. It means that if the probability of one record to have the label of “download” is larger than 0.5, then the record would be predicted as “Fraud”, otherwise “Not Fraud”. The summary of the trained model is shown as below:

****

In the table above, we can find that the significant variables are same\_cardnum\_fraud\_30, same\_cardnum\_fraud\_3, same\_merchantdesc\_fraud\_30, same\_merchantdesc\_fraud\_3, same\_merchantdesc\_fraud\_1, same\_cardnum\_3, and same\_merchantdesc\_3.

**FDR:** Combined the predicted probability to be fraud with label of ‘Yes’ and reorder from high to low according probability:

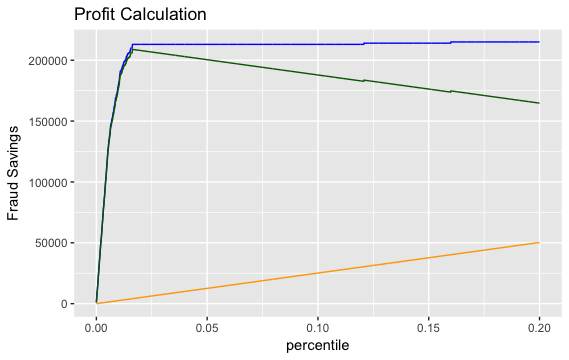
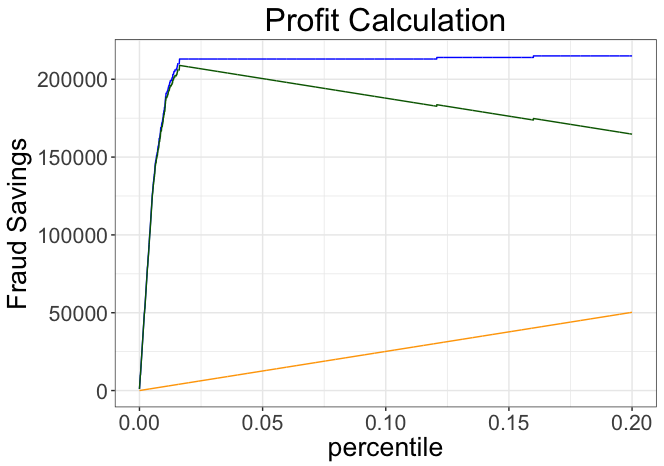
Then calculated FDR of top 2% records on training data is 0.95089; on test data is 0.9446; and on OOT data is 0.6686391.

|  |  |  |
| --- | --- | --- |
| Test | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 24,901 | 96 |
| Actual:  YES | 6 | 128 |

**Confusion Matrix and Statistics**:

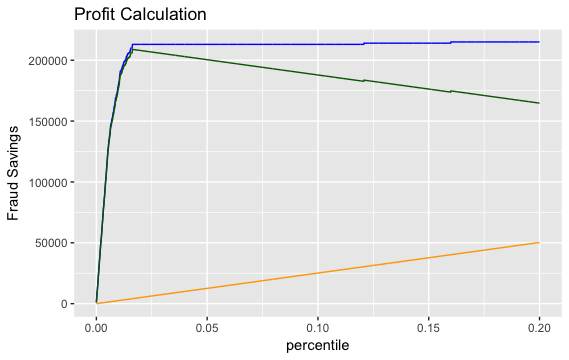
|  |  |  |
| --- | --- | --- |
| Train | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 58,166 | 169 |
| Actual:  YES | 20 | 283 |

|  |  |  |
| --- | --- | --- |
| OOT | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 12,232 | 162 |
| Actual:  YES | 15 | 176 |

**ROI Curve:**

Test Datasets

Train Datasets



OOT Datasets

**4) Gradient Boosting Tree:**

We further used boosting to make the predictors consecutively and not independently. In boosting the succeeding predictors learns from the mistakes of the prior predictors. The following steps were taken:

- We used the cross validation (a machine learning method) to choose most suitable parameters for model, the Tuning parameter 'n.trees' was held constant at a value of 0.001.

- Tuning parameter 'n.minobsinnode' was held constant at a value of 20.

- Accuracy was used to select the optimal model using the largest value.

- The final values used for the model were n.trees = 200, interaction.depth = 1, shrinkage = 0.001 and n.minobsinnode = 20.

- Using these parameters, we made a prediction model, and apply this model on train data and get the 'strange results’

|  |  |  |
| --- | --- | --- |
| Train | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 58,186 | 452 |
| Actual:  YES | 0 | 0 |

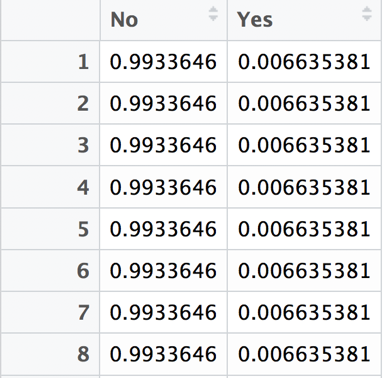
- In order to solve the strange thing, we used down sampeling to choose records again. After down sampeling, we apply the model again and get the final table (the training, the test, the oot), which are reasonable.

|  |  |  |
| --- | --- | --- |
| Test | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 24,152 | 10 |
| Actual:  YES | 755 | 214 |

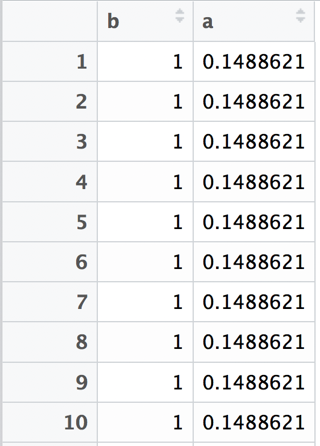
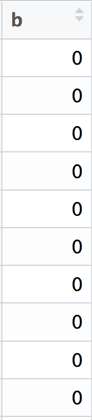
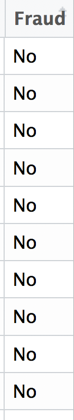
**Confusion Matrix and Statistics**:

|  |  |  |
| --- | --- | --- |
| Train | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 367 | 22 |
| Actual:  YES | 20 | 430 |

|  |  |  |
| --- | --- | --- |
| OOT | Predicted:  NO | Predicted:  YES |
| Actual:  NO | 11,438 | 10 |
| Actual:  YES | 809 | 328 |

we run the ‘pred\_train\_nroc = predict(train\_nroc,train\_apply,type="prob")’ and got the following table:

Change the column ‘fraud’ in train\_apply dataset into 0/1 variables (if the fraud is yes, then 1; if the fraud is no, then 0) and make it as.numerical:

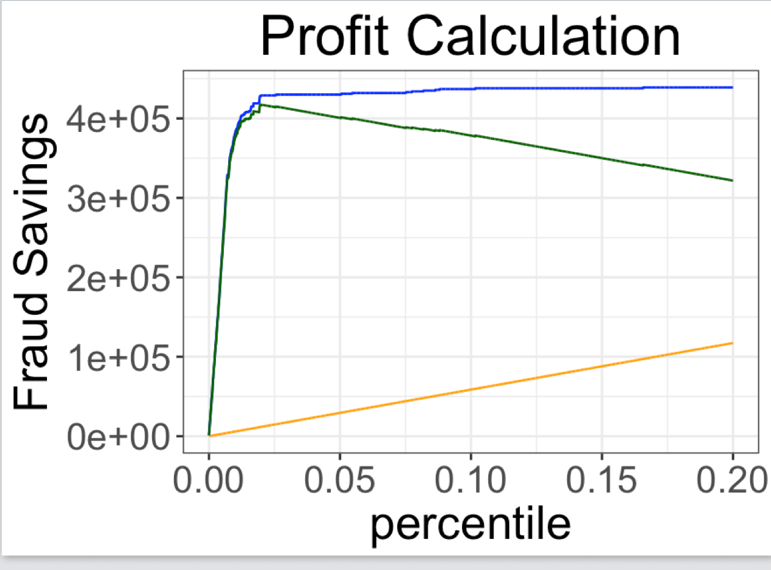
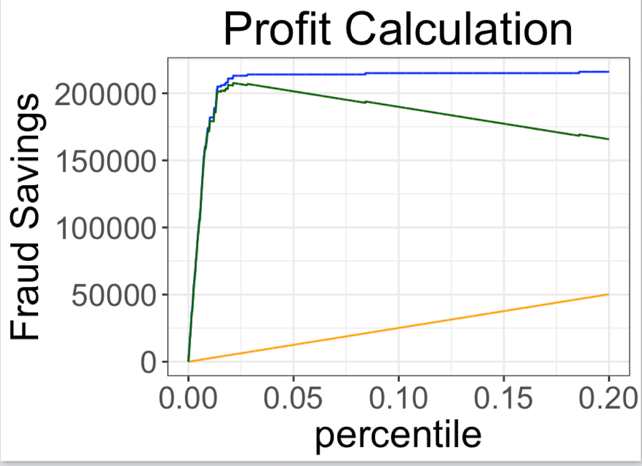


Combined column b with column ‘Yes’ and reorder from high to low according probability.

https://lh5.googleusercontent.com/-XazbsmcfId6UmgM7WZ07cXuH_Hhkt3rfKUNT_nXn_leEl0WiSoSHejlJGWtQHGjFNMwqrJ4L5MqwptVXV_D5Q0YimeuUtAeldlMBHadgj3iwBMiYlTZlKMAwbdHhzNSNGT6FtJ4120

**FDR:** Next, based on training data, we calculated the top 2% records and to see how many fraud are detected in top 2% recordings. We have calculated that the total fraud for all are 452. So the FDR for top 2% is 0.9380531.

For test data and OOT data, we repeat the above steps and calculated that in test data, the FDR for top 2% is 0.9464286. In OOT data, the FDR for top 2% is 0.7218935.

**ROIC Curve:**

Test Datasets

Train Datasets

****

OOT Datasets

**5) SVM:**

We finally used the SVM model as follows:

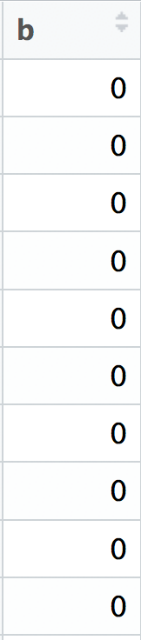
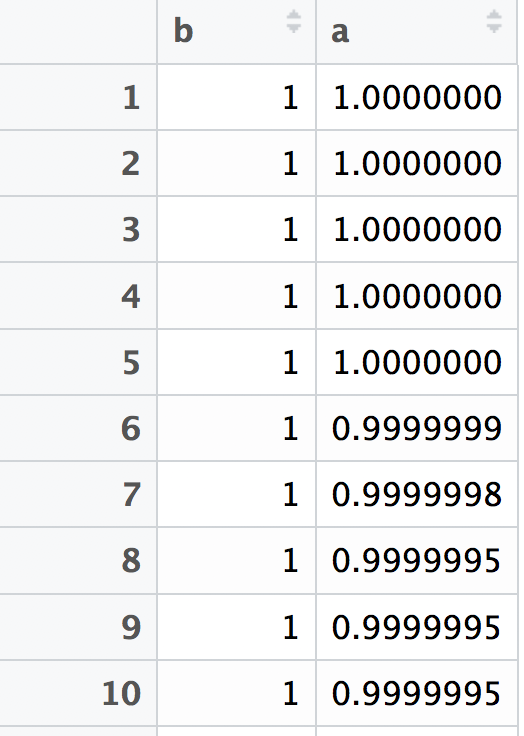
- We used SVM to make prediction model: Call: svm(formula = Fraud ~ ., data = train\_apply, probability = TRUE)

- Parameters: SVM-Type:  C-classification, SVM-Kernel:  radial

- cost:  1 and gamma:  0.07142857

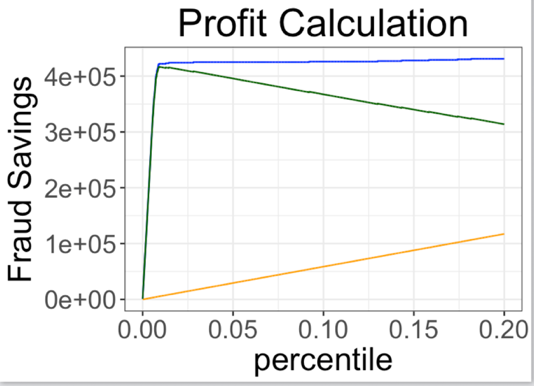
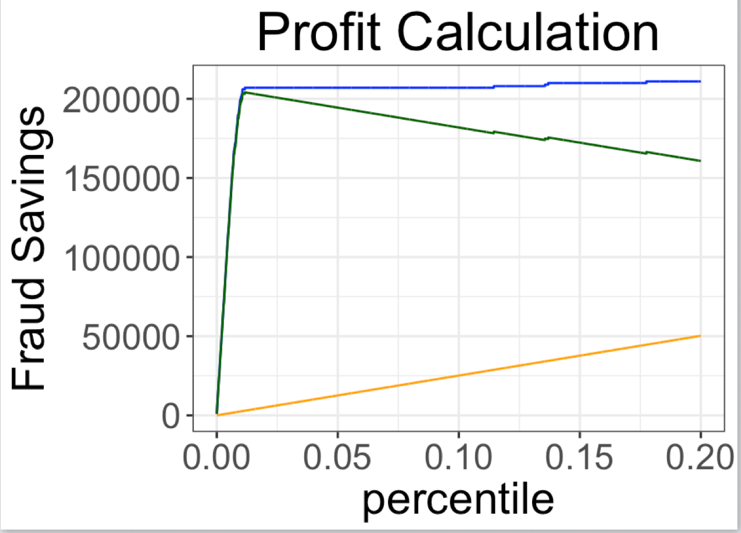
- Number of Support Vectors:  748 (353 395 )

**Next, calculated FDR**

Change the column ‘fraud’ in train\_apply dataset into 0/1 variables (if the fraud is yes, then 1; if the fraud is no, then 0) and make it as.numerical:

Combined column b with column ‘Yes’ and reorder from high to low according probability.

**FDR:** Then calculated FDR of top 2% records on training data is 0.9380531; on test data is 0.9241071; and on OOT data is 0.7218935.

**ROIC Curve**

Train Datasets

Test Datasets



OOT Datasets

**Results**

The models’ performance can be summarized in below table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Comparison of FDR** | | | | | |
| **Dataset** | **Gradient Boosting Tree** | **SVM** | **Decision Tree** | **Logistic Regression** | **Random**  **Forest** |
| **Train** | 93.81% | 92.72% | 93.14159% | 94.46% | 95.35398% |
| **Test** | 94.64% | 92.41% | 92.85714% | 95.089% | 96.42857% |
| **OOT** | 72.19% | 69.18% | 64.49704% | 66.86391% | 74.55621% |

Based on our analysis, our best model performance belongs to Random Forest Model which was able to identify 74.56% of fraudulent transactions in Out of Time period.

Even though our model is not able to identify all of the fraudulent transaction, we are still able to offer saving of about $220K for top 2% of the transactions.