**DSO 562 Project Three**

**Identity Fraud in the**

**Card Transaction Dataset**

May 1st, 2018

**Team 5:**

Keying Que

Laura Qin

Ruoxian Jia

Tianqi He

Xu Duo

**Table of Content**

[**Part I: Data Description**](#_gthxibmbbitc) **4**

[**Part II: Data Cleaning**](#_hxufbbb5lus1) **5**

[**Part III: Expert Variable Creation**](#_101veycx4k3u) **6**

[**Part IV: Feature Selection**](#_ehe9ntgl5lnr) **8**

[**Part V: Fraud Algorithms**](#_2qupxvmr8n2o) **10**

[Logistic](#_4vle5pp940bo) 10

[LDA and QDA](#_oxj5uh4wq8x) 10

[Boosted Tree](#_jducl1ko04uw) 10

[Random Forest](#_un6isq4g5djn) 11

[SVM](#_jo3ani34ogkk) 11

[Neural network](#_6qfgvxm6z8et) 11

[**Part VI: Results**](#_18p16h9kapuz) **12**

[Best Model Results](#_2ezyqao5xycq) 12

[Business Implication](#_odhsckqcbnjn) 14

[**Part VII: Conclusions (Claire)**](#_85ihxfy7ddvy) **15**

[**Appendix:**](#_8tidqibije7p) **16**



**Executive Summary**

**Challenge**

Nowadays, credit cards have become a major part of everyday payments, and large amounts of transactions are taking place via credit cards. However, due to exploitable loopholes in various settings of transactions, many credit card holders have become victims of credit card fraud. Currently, there’s an estimate of $8.45 billion credit card fraud loss in the US and the loss is expected to increase dramatically. By 2020, the nation’s loss figure would reach over $12.25 billion. In this project, our goal is to develop a fraud detection model that predicts if a credit card transaction is fraudulent or not.

**Approach and Results**

Our raw data consists of 96708 records of credit card transactions in 2010, including information about card number, date, merchant number, merchant description, merchant state, merchant zip, amount, transaction type and whether the transaction is fraudulent.

Initially, we cleaned the dataset by replacing the missing values and removing some records. We then built an auxiliary variable upon the amount variable and 124 expert variables using different time windows (1 day, 3 days, 7 days, 15 days, and 30 days) to see whether any particular field appears more than once.

Then we used Kolmogorov–Smirnov (KS) values to find the most significant variables and utilized backward selection to reduce the dimensions to 30 variables. After separating the data into training, testing, and out of time set, we built seven different models and evaluated their performances by calculating the fraud detection rate in training, testing, and out-of-time data at 2% penetration. We found that the Random Forest model performs the best, and its fraud detection rate at 2% is 74.56%.

**Benefits**

With this model, our solutions is able to catch 97.9% of frauds and save $330,930 within 2 months of out-of-time data. Going forward we could feed all transaction data into our model and produce a risk score for each transaction and flag records that are above the 3.226% fraud score threshold. We also recommend implementing a real-time alert system to allow card holders to confirm risky transactions. Consequently, this allows us to receive feedbacks and newest fraud labels to enrich and continuously improve our model.

# Part I: Data Description

*In this data overview, we give a general summary of the characteristic of the dataset and highlight certain key variable that were important in our analysis. For a description of all 30 variables in the dataset, please refer to the appendix.*

File Name: Card Transactions

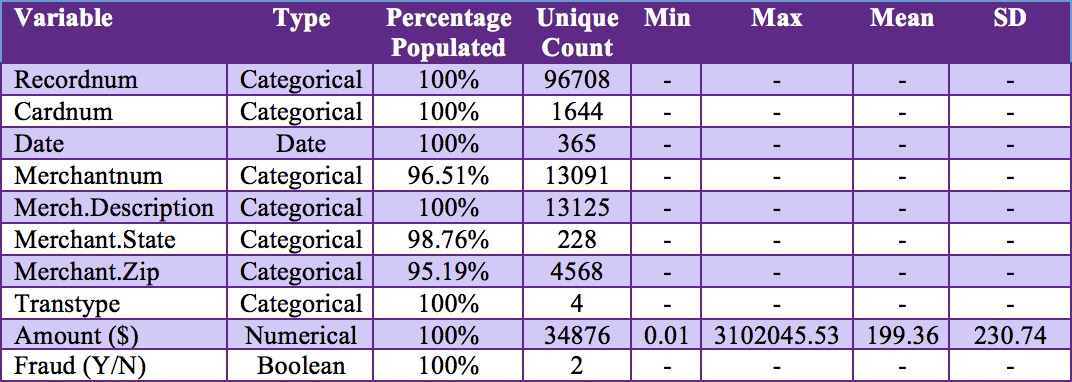
File description: The dataset contains over a hundred thousand credit card translation records, including card information, merchant information, transaction amount, etc.

Number of Records: 96,708 records

Variables: 10 variables overall – 7 categorical variables, 1 numeric variable, 1 date variables, 1 boolean variable

Time of Records: January 1st, 2010 - December 31st, 2010

***Summary statistics for all variables:***

******

Recordnum: Unique identifier for each variable

Cardnum: The credit card number in the card transaction

Data: The date when the transaction occurred

Merchantnum: The merchant account number

Merch.Description: The description of each merchant

Merch.State: The state where the transaction occurred

Merch.Zip: The zip code of the merchant address

Transtype: The transaction type (P, A, D, Y)

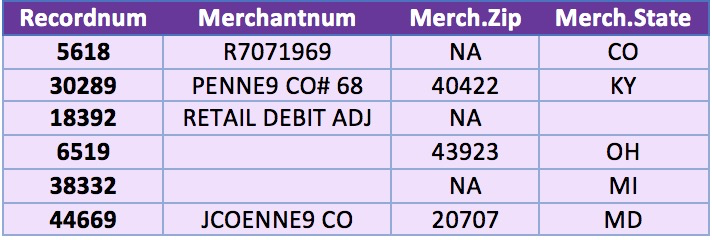
Amount ($): The monetary transaction amount

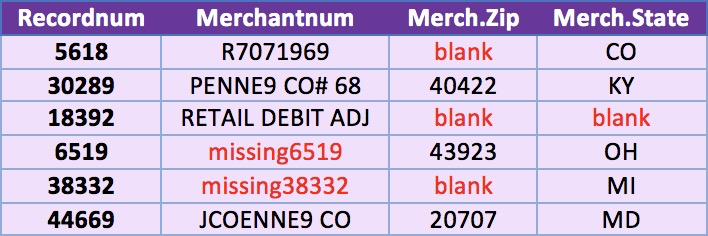
Fraud: Whether the credit card transaction is fraudulent (“1” for Yes, “0” for No)

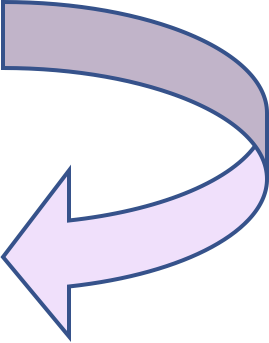
# Part II: Data Cleaning

After inspecting the dataset and producing our initial Data Quality Report, we found that thousands of values for the categorical variables were missing. Some of these values appeared as blanks and others as NAs in our initial data exploration. The problem with having these missing values was that they could create false alarms when running our fraud algorithms later in the analysis. For that reason, we cleaned the data by replacing the missing values with reasonable estimates. Three variables contain missing values that need to be cleaned: “Merchantnum”, “Merch.Zip” and “Merch.State”.

Firstly, we replaced the missing Merchantnum with unique values. Each unique value was made by combining a character string “missing” and the Recordnum associated with that record. In addition, we used character string “blank” to directly replace the missing merchant state and zip code.







Besides missing values, we also need to filter out some records from Transtype and Amount. In the Transtype variable, there are only 0.37% records are type A, D and Y. Those types don’t have any relationship with our objective, so we removed all the records with those type. Moreover, the largest transaction amount is up to $3,102,045.53. The related merchant description of this amount is “INTERMEXICO”. We assumed that the amount is recorded in Peso, instead of USD, When we conduct further analysis, we will exclude this record.

# Part III: Expert Variable Creation

Expert Variable is the variable created from original variables. They are the explicit variables that best relate the raw fields to the output. In this project, we built expert variables to increase the prediction accuracy of our models.

We first translated the amount variable to a categorical variable and saved it as a new variable--AmountLabel. More specifically, we sorted records based on Amount from lowest to highest, and then re-code the amount into six bins (A- F). The first five categories (A-E) contain the same amount of records and already cover 99.9% records. The category E represents the mega transaction, which are also outliers in our data.

Then, we began to build the expert variables based on the original variables. Our expert variables can be grouped into six main types: Recency, Count, Fraudulent Count, Unique Count, Percentage and Weekly trend.

Below is a table that shows the names of our expert variables and how we built them. The detail of each variable is attached in the appendix.

|  |  |  |
| --- | --- | --- |
| **Category** | **Variable Name** | **Description** |
| Recency | Recency\_card  Recency\_merchant  Recency\_amt  Recency\_zip  Recency\_card\_merchant  Recency\_card\_zip | How many days the particular record was last seen |
| Count | Card\_x  Card\_zip\_x  Card\_amt\_x  Card\_state\_x  Card\_merchant\_x  …..  Merchant\_state\_x | How many transactions happened in the last X days that have the same features |
| Fraudulent Count | Card\_fraud\_x  Card\_zip\_fraud\_x  Card\_amt\_fraud\_x  Card\_state\_fraud\_x  Card\_merchant\_fraud\_x  Card\_label\_Fraud\_x  ….  Merchant\_label\_Fraud\_x | How many Fraud transactions happened in the last X days that have the same features |
| Unique Count | Unique\_zip\_x  Unique\_merchant\_x | How many unique Feature for the same Cardnum in the last X days |
| Percentage | Per\_card  Per\_card\_state  Per\_card\_merchant  Per\_day  Per\_month | Record amount / Total average amount for different features |
| Weekly Trend | Trend\_card\_day  Trend\_merchant\_day | Transaction amount for one record compared to the average for records with the same features and day of week |

\* X indicates the time window, X ∈ {1, 3, 7, 15, 30}

# 

# Part IV: Feature Selection

During the feature selection stage, we calculated the Kolmogorov–Smirnov (KS) value for each variable and used backward stepwise method to select 30 variables for our final models. The reason of conducting feature selection is to 1) make sure variables used for model building are the most critical ones to detect fraud; 2) reduce dimensions to avoid collinearity and output a model that is closer to the actual model.

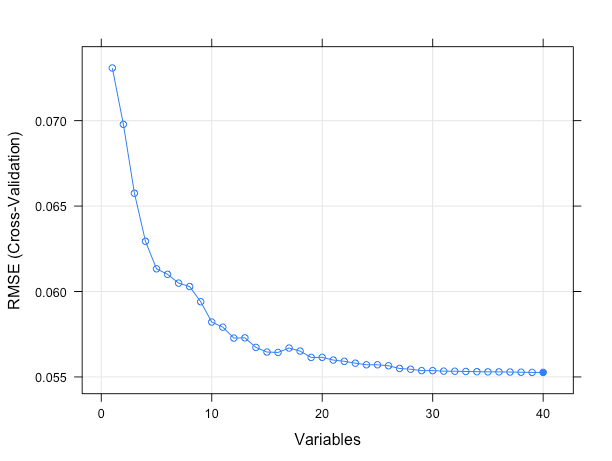
**Univariate Feature Selection: Kolmogorov–Smirnov (KS)**

KS value measures the maximum distances between two distributions. The larger the KS, the more separate the two distributions. In the context of detecting fraud, we used KS to measure the differences between fraud records and non-fraud records for each variable created. Specifically, for each variable, we gathered a list of fields corresponding to fraud records and the other list of fields relevant to non-fraud records. Then we applied*ks.test* function in R to compute KS and level of significance for all variables. We selected the top 40 variables which have the highest KS value and then we used these variables to run backward selection.

**Wrapped Method: Backward Stepwise**

Backward stepwise selection starts with all candidate variables, and remove one variable at a time such that the deterioration of the model is the most insignificant. This process is repeated until no further variables can be eliminated without a statistically significant loss of fit. In this project, we started with 40 variables that are selected by univariate feature selection, and used the *step* function in R to conduct backward stepwise selection on a logistic regression model.

Below is a picture which indicates how the backward selection works.



Below is a list of the 30 variables chosen by backward selection and are ones we used in our final models:

|  |  |  |
| --- | --- | --- |
| Card\_fraud\_1 | Card\_fraud\_3 | Card\_state\_fraud\_1 |
| Card\_state\_fraud\_3 | Card\_state\_fraud\_7 | Card\_state\_fraud\_15 |
| Card\_state\_fraud\_30 | Card\_merchant\_fraud\_3 | Card\_merchant\_fraud\_7 |
| Card\_merchant\_fraud\_15 | Card\_merchant\_fraud\_30 | Card\_label\_fraud\_1 |
| Card\_label\_fraud\_3 | Card\_label\_fraud\_7 | Card\_label\_fraud\_15 |
| Card\_label\_fraud\_30 | Merchant\_label\_Fraud\_1 | Merchant\_label\_Fraud\_3 |
| Merchant\_label\_Fraud\_7 | Merchant\_label\_Fraud\_15 | Merchant\_label\_Fraud\_30 |
| Card\_zip\_fraud\_1 | Card\_zip\_fraud\_3 | Card\_zip\_fraud\_7 |
| Card\_zip\_fraud\_15 | Card\_zip\_fraud\_30 | Merchant\_fraud\_1 |
| Merchant\_fraud\_15 | Per\_day | Per\_month |

# Part V: Fraud Algorithms

## Logistic

Logistic regression uses the sigmoid function to prediction the probability of an observation belonging to either class. Using the*glm* function in R, we fitted a logistic regression model on the training dataset and used the trained model to predict training, testing and out of time records based on their input expert variables. We used the predicted probability to sort the records, and calculated the fraud detection rate at 2% penetration utilizing the function mentioned earlier.

## LDA and QDA

We also used Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) for model building. LDA uses statistic properties of data and Bayes’ theorem to determine the probability of a record belonging to a certain class. It maximize the separation of each class with a linear boundary. QDA is a model that similar with LDA, but it uses quadratic decision boundary. We used default parameters in the MASS package in R for both models and compute fraud detection rate at 2% penetration rate.

## Boosted Tree

Boosting tree is an ensemble method used to improve the prediction accuracy of a single decision tree. Each tree is grown using information from previously grown trees. The package we used in R is called gbm. We set the number of trees to be 500 for a simpler model. To improve precision, we set shrinkage parameter to be 0.01 to give the ensemble model a slower learning rate, which allows us to fit more trees and reduce variance. The interaction depth in our model is 4, which means that at most 4 splits are allowed in each individual tree.

## Random Forest

Random forest is another ensemble model that improves on decision trees by building multiple trees with different subsets of records and variables. We used the *randomForest* package in R. Since the data is imbalanced, we first downsampled the data so that proportion of non-frauds versus frauds is 10:1. When we trained the random forest model, we tried generating 500, 1000 and 5000 decision trees and decided to go with 500 trees since it gave us a reasonably high fraud detection at 2% penetration rate without being too computationally expensive.

## SVM

SVM, short for support vector machine, is an unsupervised classification algorithm that specializes in binary classification. It find a hyperplane in the feature space such that the data points are as well separated as possible, and the margin--which is the minimum distance between the data point and the hyperplane--is maximized. To build a SVM model to analyze card transactions, we used the e1071 package in R. We set the weights of fraud and non-fraud records to be 10 to 1, since we are more interested in catching frauds than identifying non-fraud records. In addition we experimented with different combinations of parameters, and finally set cost to be 2 and kernel to be linear, since it gives the best result on out-of-time data. At 2% penetration, the out-of-time fraud detection rate of SVM model is 73.08%, which is quite good compared to performance of other models.

## Neural network

Neural network takes data through layers of interconnected nodes, and perform a series of linear and nonlinear transformations to produce an output. For our analysits, we used the *keras* package in Python to build the neural network. More specifically, we used *Sequential()* function to create the model and use *model.add()*function to add layers with the activation type *‘relu*’. We tried models of one layer with ten nodes and twenty nodes respectively and found that the one-layer model with 20 nodes generated a better result.

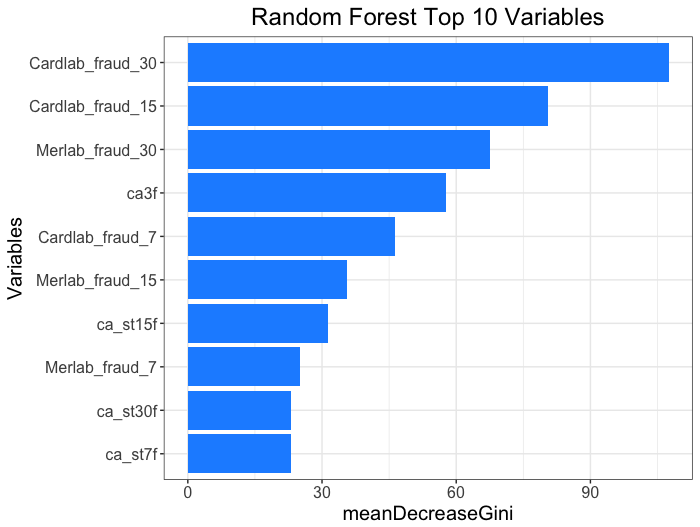
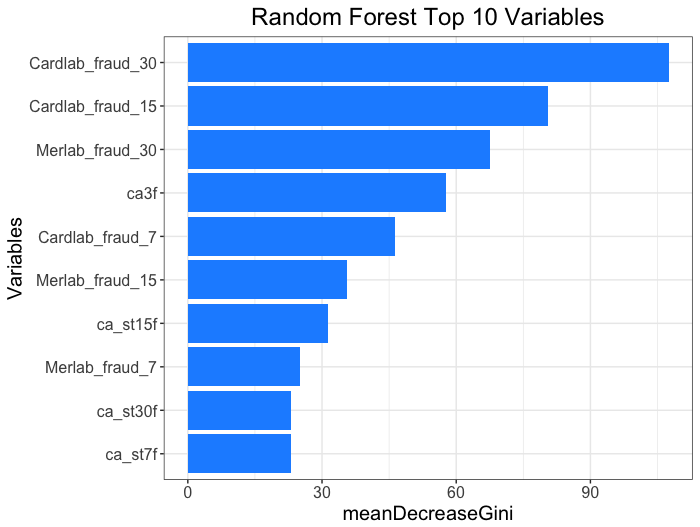
# Part VI: Results

## Best Model Results

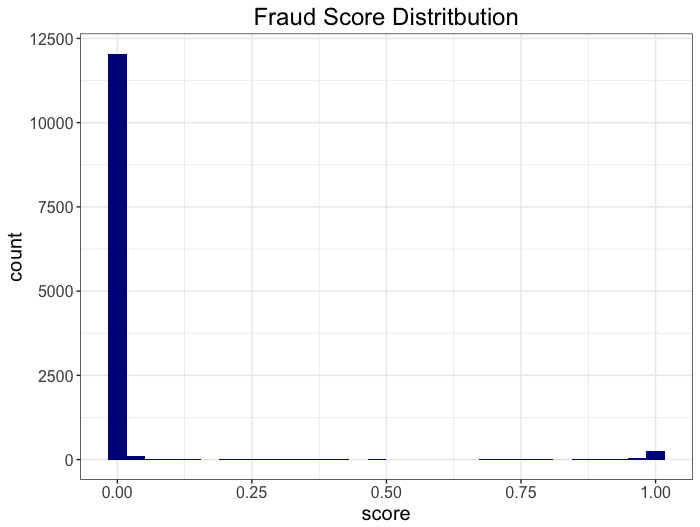
Below is a summary of our model performance on training, testing and out-of-time dataset. All models perform quite well, which could be due to our use of historical fraud labels when creating expert variables. Among all the models, random forest performs the best on training, testing and out-of-time data. The out-of-time fraud detection rate for random forest is 0.7456, which is the highest fraud detection achievable at 2% penetration. Thus, we agreed that random forest is our best model for this project.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | 2% FDR Training | 2% FDR Testing | 2% FDR OOT |
| Logistic | 0.9648 | 0.9552 | 0.7160 |
| LDA | 0.9619 | 0.9433 | 0.6864 |
| QDA | 0.9824 | 0.9850 | 0.6657 |
| SVM | 0.9765 | 0.9701 | 0.7308 |
| Random Forest | 1 | 0.9851 | 0.7456 |
| Boosted Tree | 1 | 0.9761 | 0.7456 |
| Neural Network | 0.9824 | 0.9701 | 0.7189 |

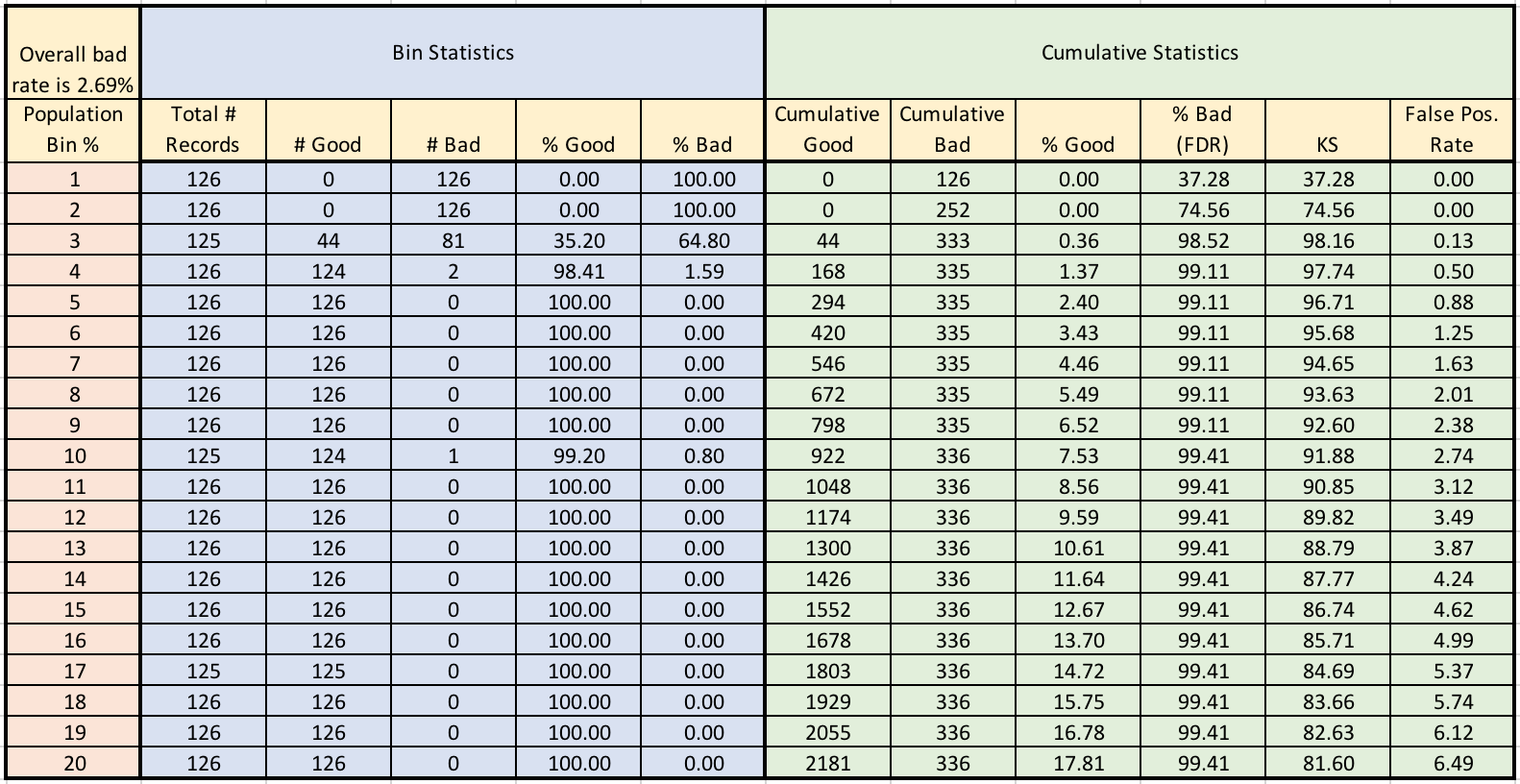
The graph below shows the 10 most important variables in our random forest model, ranked by the average amount of decrease in Gini index by splitting on that variable. As we can see, almost all of the variables in the bar chart have incorporated fraud labels, which tells us that the use of fraud label is one of the biggest reason for us getting such a good result. In addition, 6 out of the 10 best variables have used the binned transaction amount variable named “label”, meaning that binned transaction amount is highly predictive when combined with time window counts.

Below is the distribution of fraud scores of random forest model. As seen in the graph, the vast majority of scores are around 0, with about several hundred scores around 1 and very few scores in between. This suggests that the card transactions are well-separated using this fraud score.



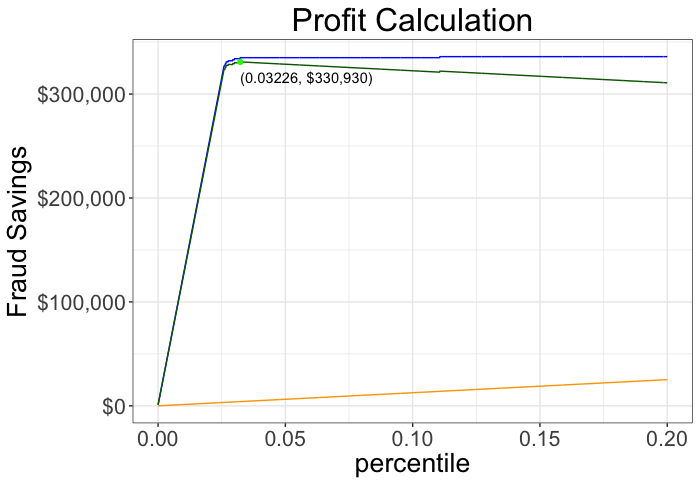
As requested, we made a table that includes bin statistics and cumulative statistics at different levels of penetration in the out-of-time data. The table shows that the fraud detection rates at 1%, 2% and 3% penetration are 37.28%, 74.56% and 98.52% respectively, and after 3% the fraud detection rate does not increase by much. Given that the overall percentage of fraudulent records in out-of-time data is 2.69%, being able to catch vast majority of the frauds at 3% penetration means the model performs very well.



## Business Implication

In our analysis, we assume that each fraud caught saves our client $1,000 in cost, and each rejected transaction incurs a $10 loss in forgone business opportunity. To decide where to set the cutoff for rejecting transactions, we visualized the cumulative fraud savings associated with each percentile as a ROI curve, which is shown below. The blue line represents the saving from catching frauds, and the orange line is the cost of rejecting transactions. The green line is the difference between the two, which corresponds to the net saving from fraud detection at each percentile cutoff. As shown on the graph, the net saving is maximized to $330,930 when we set the cutoff at 3.226%.

Fraud Saving vs. Cutoff



# Part VII: Conclusions

In order to provide banks with a solution to prevent millions and thousands dollars of losses in credit card transaction fraud, we have built a model that is able to predict high risk transactions. Therefore, banks are able to block the transaction in-time to prevent future losses.

We were provided a dataset with 96,708 credit card transaction records. Each records contains card number, transaction data, merchant number and description, merchant location information (state and zip code), transaction type, amount as well as a label indicating whether the record is fraudulent. We carefully examined the data, updated missing values, and removed one record with amount in Peso instead of US currency.

We built total of 124 variables based on the original variables using time windows. We chose five different time windows, namely 1 day, 3 days, 7 days (a week), 15 days (half month), and 30 days (a month). In past number of days given by a time window, we examined recency and count of records with same features, such as card number, merchantant, and zip code, and combination of a few features. Using the same method, we also incorporate number of fraud transactions having the same features happened prior to each transaction. This group of variables later was proven very powerful in predicting fraudulent records. Moreover, we built a variable that captured the standardized differences from average in transaction amounts on each day of the week. In addition, it is worth to mention that because small transaction amounts could be a signal of fraud, we grouped amounts into six categories (A-F) and E being the outlier.

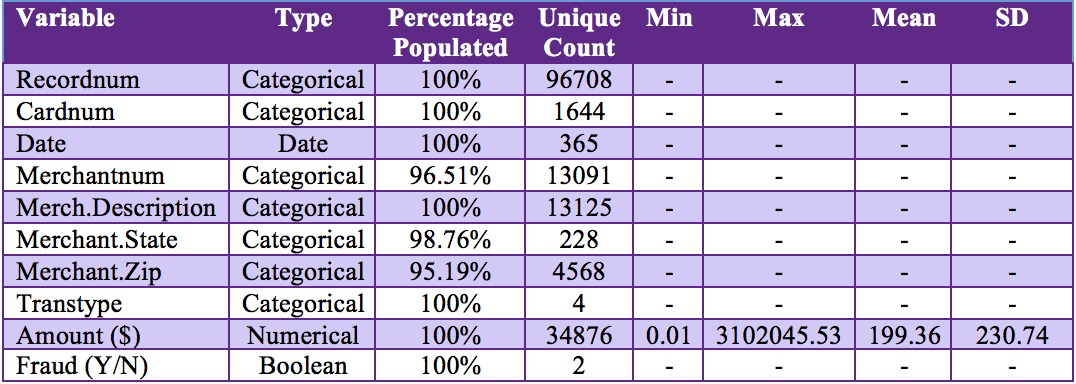
Then KS test and backward selection was conducted, which led us to chose top 30 variables for our models. We used logistic regression, LDA, QDA, SVM, random forest, boosted trees, and neural network to build classification models. We fitted models on our training data and validated them on both testing and out-of-time data. The models were evaluated based on fraud detection rate at top 2% of records with the highest risk scores. Based on this measure, random forest performed the best and achieved perfect rate in training, 98.51% at testing, and 74.56% at out-of-time.

Given the best model, we conducted fraud saving analysis to provide the optimal percentile cutoff which is able to maximize the savings. We identified 3.226% is the best cutoff and it came with $330,930 in saving. We also proposed to implement a real-time fraud alert system to get the newest fraud labels and transactions to enrich the model.

**Appendix**

This dataset contains 96,708 records on credit card transactions from January 1st, 2010 to December 31st, 2010. There are total ten variables including card number, transaction date, merchant number, merchant description, location or state of the merchants, zip code of the merchants, transaction type, transaction amount, and whether a record has been labeled as fraud. Only amount is the only numeric variable, and the rest of the variables are categorical or date.

The table below shows summary statistics of each variable, including the percentage of record populated and the number of unique values.



The section starting next page shows descriptions and visualizations of each variable.