Supervised Credit Card Transactions Fraud Algorithm

Heng Chi

Table of Contents

Executive Summary

According to the Federal Bureau of Investigation, "Credit card fraud is the unauthorized use of a credit or debit card, or similar payment tool, to fraudulently obtain money or property". In the United States, for the year 2020, with over 270,000 reports, credit card fraud was the most common type of identity theft and more than doubled from 2017 to 2019. Almost 165 million records containing personal data were exposed through data breaches in 2019. These records had led to about $28.65 billion worth of losses worldwide in 2019, $11 billion in the United States, and the number keeps increasing. (CNBC)

The trend of increasing losses of credit card fraud must be interrupted before it hurts the root of our whole financial system. One effective way to do this is to use the technology of machine learning to detect fraudulent transactions and hold their processing, which is already applied by almost every financial services provider. In this project, we are going to study this detection technique and try to learn and build up some predictive models to help understand the fraud detection process.

The data set we used for this project contains 96,753 records and 10 variables. 1,059 records have the fraud column flagged positive, which is about 1.1% of the total. We first went through it by exploratory data analysis to show a general understanding of the raw data set. Then, in the data preparation stage, we cleaned the data set, by excluding many useless samples such as the ones with transaction type not equal to "P". We filled in missing values for some variables either by its other related variables or by the method of mode (most common contents). We created a lot of variables based on the data. Including amount variables, frequency variables, days-since candidate variables, velocity change candidate variables, and Benford's law variables.

After the data is prepared, out of more than 500 variables we had, we performed feature selection to find out the 30 best variables for modeling. We ran several classification models with different algorithms (Logistic regression, Gradient Boosting, Neural Network, and Random Forest). Finally, we proceed with Random Forests due to its best performance shown by the fraud detection rate, with 85% FDR at 3% for the test data and 67% FDR at 3% for the oot data.

Data Set Description

Our data set ("card transactions.csv") is collected and provided by the U.S. government organization. It is a set of actual historical data about credit card purchases' transaction information from January 1st, 2021 to December 31st, 2021. Inside it, there are 96,753 lines (header excluded) of different card transaction records. Each record contains 10 relevant information fields about the particular card transaction including a flag that shows whether or not the transaction turned out to be a fraudulent transaction.

Summary Table

Numeric Fields:

Field	count	mean	std	min	25%	50%	75%	max	Unique Value	# Zero	% populated
Amount	96753	427.89	10006.14	0.01	33.48	137.98	428.20	3102045.53	34909	0	100.00

There is only one variable in the numeric fields: **Amount**. This variable indicates the amount of money in a particular transaction. The distribution of the Amount is very wide with a standard deviation of 10,006.13. One thing worth notice was the maximum number of the variable is very strange, likely to be an outlier. We will deal with this in the later data cleaning part.

Categorical Fields:

Field	count	% populated	Unique Values	Most Common Field Values
Recnum	96753	100.00	96753	1
Cardnum	96753	100.00	1645	5142148452
Date	96753	100.00	365	2/28/10
Merchnum	93378	96.51	13091	930090121224
Merch description	96753	100.00	13126	GSA-FSS-ADV
Merch state	95558	98.76	227	TN
Merch zip	92097	95.19	4567	38118.0
Transtype	96753	100.00	4	P
Fraud	96753	100.00	2	0

The rest of the variables are categorical data. The ones that are noticeable are Merchnum, Merch state, Merch zip. **Merchnum** is the variable that identifies the Merchants that had the transaction. It has a 93,378 population, which means it is missing 3,375 samples. Similar to Merchnum, **Merch state** and **Merch zip** (indicating the Merchant's geographical information) missed 1,195 and 4,656 samples. This issue will be discussed later in the part of data filling.

The most important field in our data set is **Fraud**, it is the flag and standard that we use to train, validate, and test our models. It is 100% populated, with 2 unique values: {0,1}. 0 means the transaction was not a fraud, 1 means the transaction was a fraud. Specifically, 95,694 samples were not fraud and 1,059 samples were frauds.

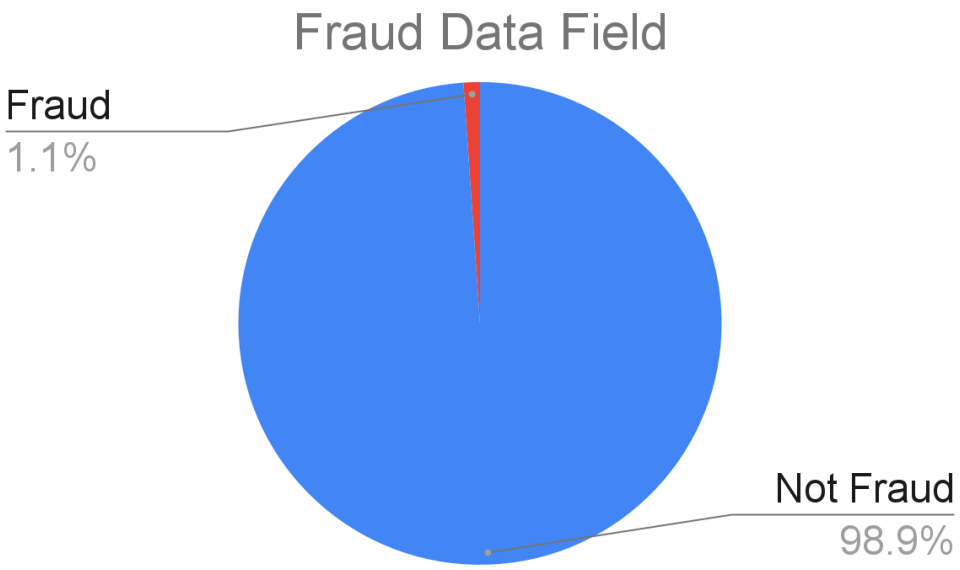


Figure 1. Fraud Data Field

Data Preparation

In the real world, data is never perfect. To maximize the performance of our analysis, it is necessary to clean and manipulate the data to the preferred status. Creating variables based on existing data is another good idea for preparation.

Data Cleaning

The goal of data cleaning is to identify any exclusion and bad records. A bad record could be filled with a reasonable value or simply deleted. In our cases, we did both exclusion and data filling.

Exclusion

Among all 96,753 rows, there exist 4 types of transaction ("Transtype"): {P, A, D, Y}. The types A, D, and Y have a total of only 355 records, which was too small to help our study. We thought focusing only on purchasing transactions would be a better idea to exclude the potential noise.

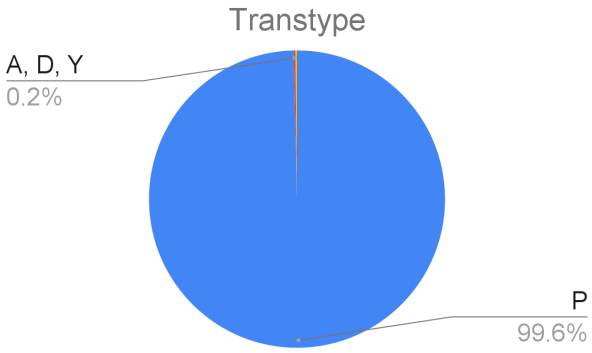


Figure 2. Transtype

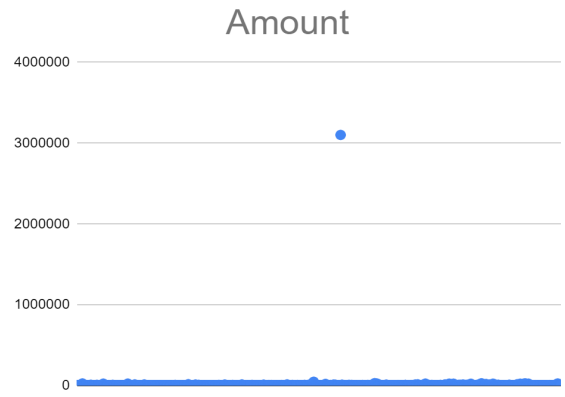


Figure 3. Amount

Also, while going through the data set, we found an outlier from the numeric variable "Amount". As shown in the scatter plot on the left, it is obvious that one dot is an outlier. That dot is Recnum "52715" and Cardnum "5142189135". It has the number 3,102,045.53 as the value of "Amount". It is indeed possible that some super-rich swiped his/her credit card for 3 million dollars. But we thought such data was after all not very helpful for future modeling and research. Hence that row was deleted.

To conclude the exclusion part, we have excluded a total of 356 rows.

Data Filling

We found that there were 3 variables that contained missing values: “Merchnum”, “Merch zip”, and “Merch state”. All of them are quite important for our study and lacking too many rows so that we could not simply delete the rows with a missing value. Hence, tried different methods to fill in the missing value.

1. “Merchnum”:

   This variable is a categorical variable indicating the merchant number of the transaction. We found 3,198 rows with Merchnum as a missing value. Also, we found 53 rows with Marchunm as 0. 0 value in Merchnum does not provide anything more than a missing value, hence we converted that 0 Merchnum into NaN (missing value), which gives us a total of 3,251 missing values.

   We noticed that the variable Merchnum is connected with the variable Merch description. Hence, in order to fill the missing Merchnums, we applied the mode method (the contents which appear most often) of the Merch description. That is if a row with missing Merchnum but has Merch description filled. We will fill the blank with the most common Merchnum of its Merch description.

   Furthermore, there were a few rows that were not applicable for the mode method. Such as the rows with Merch description as “RETAIL CREDIT ADJUSTMENT " or “RETAIL DEBIT ADJUSTMENT”, forcing these rows to get a Merchnum is not smart. What we did was assign “unknown” for these special cases. Until this step, we were able to make sure all the variables have Merchnum filled with a valid value.

2. “Merch State”:

   This variable is a categorical variable indicating the state where the merchant in the transaction. We found 1,021 rows with Merch State as a missing value.

   It is obvious that the variable Merch zip can provide the correct answer to any blank Merch State. Hence, the very first step we took was going over the 1,021 rows, finding the rows with known zip codes, and using the state of that zip to fill the blank. One thing worth mentioning is that, instead of unknown, the zip code in the range of 00600 - 00799 and 00900 - 00999 was set as “PR” (Puerto Rico).

After the zip code treatment, like what we did in Merchnum, we applied the mode method to the remaining blank of the Merch description. Again, the rows with Merch description as “RETAIL CREDIT ADJUSTMENT" or “RETAIL DEBIT ADJUSTMENT” were assigned “Unk” for unknown. Until this step, we were able to make sure all the variables have Merch State filled with a valid value.

3. “Merch zip”:

   This variable is a categorical variable indicating the zip code where the merchant has the transaction. We found 4,301 rows with Merch zip as a missing value.

   Taking the experience in Merchnum and Merch state, we searched over the other merchant’s information including state, number, and description to find the fitting value to fill. Then, for the rows that could not find a fit, we applied the mode method to them of Merch description. Finally, the rows with special descriptions about adjustment were assigned “Unk” for unknown. And that concluded our data filling.

Making Variables

To help our predictive analysis, we decided to use more variables that are made out of the existing variables. Making more variables can help improve the machine learning process, by helping the model to identify extra important information from the data, it makes the model slower but more accurate. Recall that we had these entities available: {cardnum, Merchnum, Merch description, Merch state, Merch zip, Amount} From them, we will first create some initial variables with some simple merging, then we will create a lot of new variables including Amount Variables, Frequency Variables, Days-Since Candidate Variables, Velocity Change Candidate Variables, and Benford’s Law Variables.

Creating Initial Variables

The first step of making variables is to identify the entities we wanted to expand on and create some variables that represent the merging status of the combinations of different entities. Since the learning process is done by computer, we thought making as many entities as possible would be useful. The initial variables are:

- *df[‘Merchnum’]*
- *df[‘Cardnum’]*

- *df['card_merch'] = df['Cardnum'] + df['Merchnum']*
- *df['card_zip'] = df['Cardnum'] + df['Merch zip']*
- *df['card_state'] = df['Cardnum'] + df['Merch state']*
- *df['merch_zip'] = df['Merchnum'] + df['Merch zip']*
- *df['merch_state'] = df['Merchnum'] + df['Merch state']*

For each of these entities, we created some different types of variables: amount variables, frequency variables, day since variables, and velocity variables.

Amount Variables

These are numeric variables that record the statistics of the monetary amount of each entity's transactions in the data. Including the average, maximum, median, total, actual/average, actual/maximum, actual/median, and actual/total amount of the transactions over the past 0, 1, 3, 7, 14, and 30 days. (0 represents the current status)

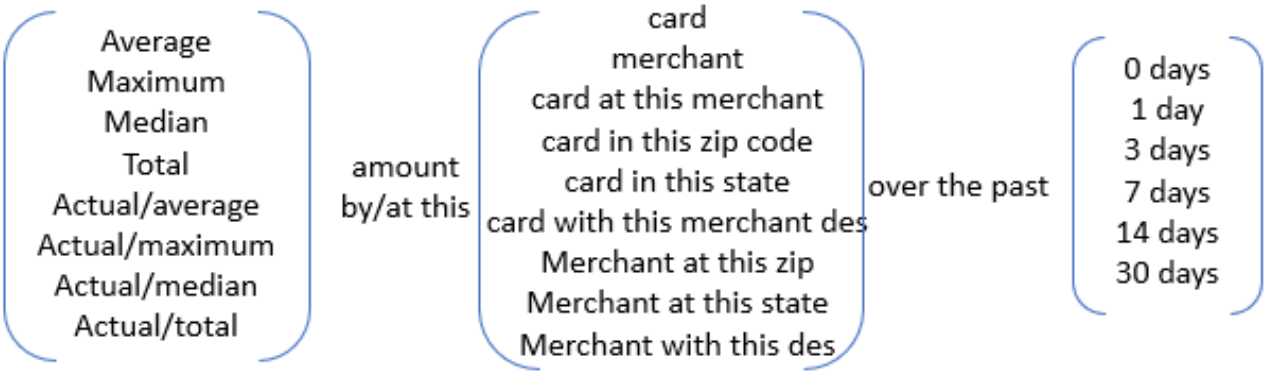


Figure 4. Amount Variables

The formula above showed the logic behind the amount variables' creation. Going through all the combinations here is unnecessary, please check Appendix B to see the whole list. Instead, we can present some examples here to provide an intuitive understanding. All the examples will take the top 1 choice of the formulas.

Let's say we are creating a variable that follows the first row of the formula above. That is focusing on the Average amount of a specific card over the past 0 days (same day). And let's say the card has Cardnum as 5142190439 (the first row of the data set). Then, we will find that the card had shown up 264 times in the data set. If we want to learn about one transaction of that card on 2010-12-20, we would find out that there were two transactions processed that day. They were Recnum 94713 and 94848. And the amounts for the two were 69.99 and 89.99. According to the

formula, the average Amount for that card on 2010-12-20 for the past 0 days would be (69.99 + 89.99) / 2 = 79.99. And this number would belong to the column of "Cardnum_avg_0".

Frequency Variables

These are numeric variables that record the frequency/number of each entity's transactions over the past 0, 1, 3, 7, 14, and 30 days.

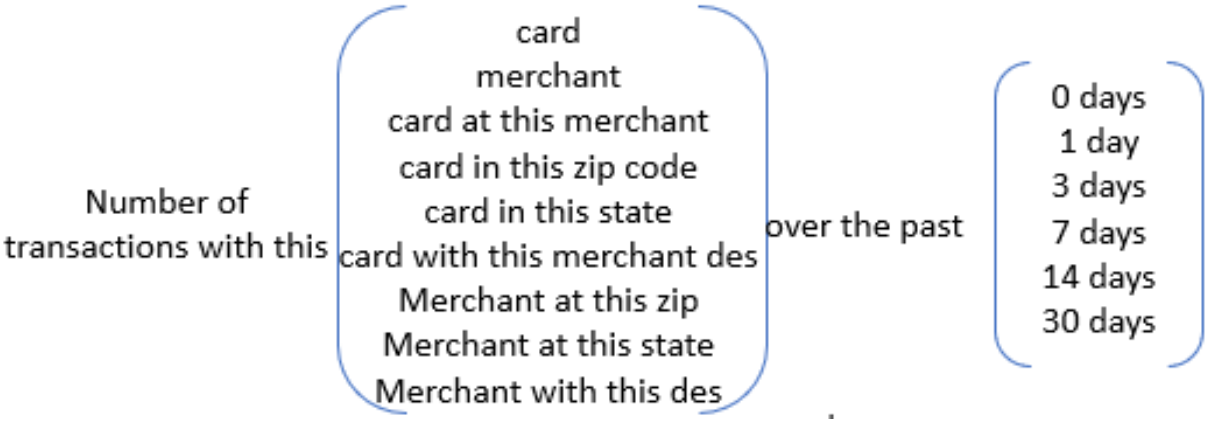


Figure 5. Frequency Variables

The above formula shows how we get the value for the variable we created. Let us again look at the example we had with Recnum 94713 and 94848. We already knew that these two transactions are the only transactions that happened on 2010-12-20 from Cardnum 5142190439. Hence, the value we would input under Cardnum_count_0 for the Recnum 94713 and 94848 would be 2.

Day-since Variables

These are numeric variables that count the number of days since the last time an entity data was seen. We find it by minus the current date by the most recent transaction date. If it is the first time seen of the entity, put 365 as the value.

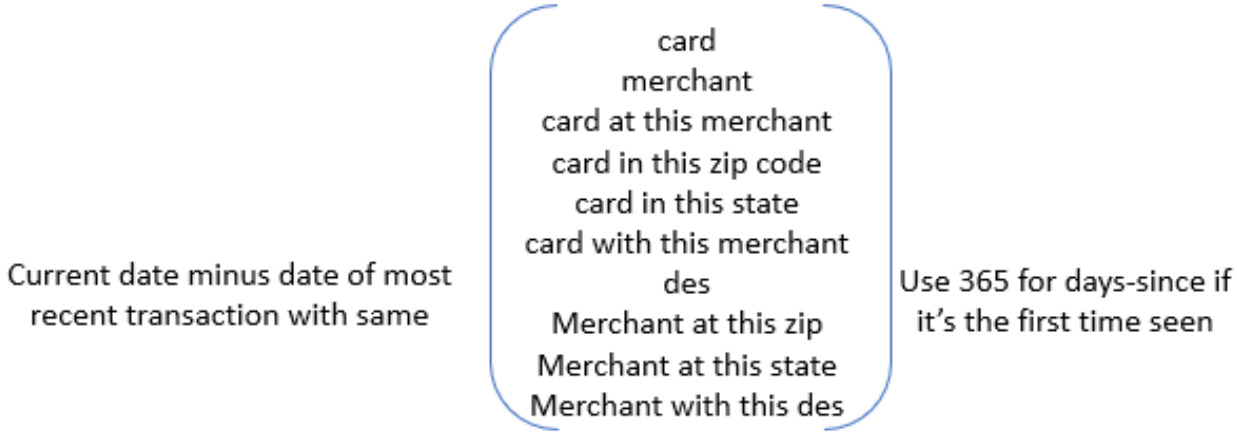


Figure 6. Day-since Variables

The above formula shows how we get the value for the variable we created. Let us again look at the example we had with Recnum 94713 and 94848. For their card, the most recent transaction made before 2010-12-20 was 2010-12-18, which was 2 days before. Hence we got Cardnum_day_since = 2 for Recnum 94713 and 94848.

Velocity Change Variables

These are numeric variables that record the speed at which an entity was seen for a transaction record over a certain period related to what was considered normal. For the numerator, we applied over the past 0-day and 1-day. For the denominator, we applied over the past 7-days, 14-days, and 30-days.

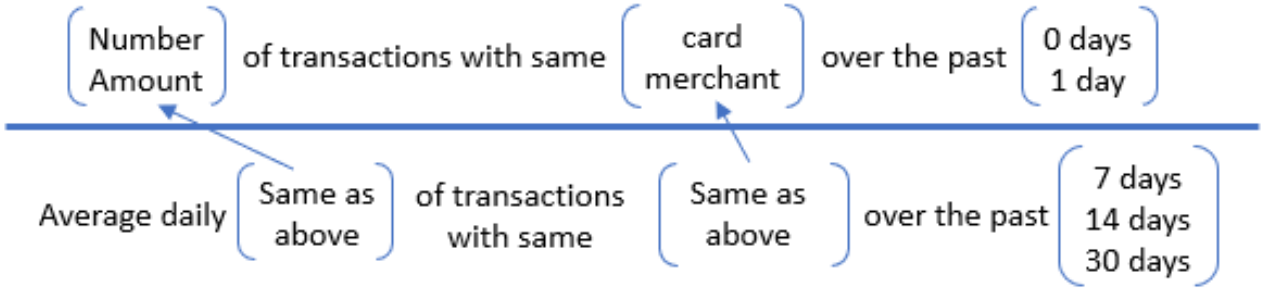


Figure 7. Velocity Change Variables

The above formula shows how we get the value for the variable we created. Let us go through one of our previous examples: Recnum 94713. We have already known the number of transactions with the same card over the past 0 days: 2, which completed the denominator of the formula. For the nominator part, we needed to find the average daily number of transactions with the

same number over the past 7 days. There were a total of 11 transactions with the same card over the past days, 9 transactions on 2010-12-18, 1 on 2010-12-17, and 1 on 2010-12-15. 11/7=1.57 would fill the nominator. As a conclusion, the value of variable "Cardnum_cont_0_by_7" for Recnum 94713 would be 2/1.57 = 0.0351.

Benford's Law Variables

This kind of variable came from Benford's law, which is the principle that about 30% of the time, the numeral 1 will be the leading digit is a genuine data set of numbers. About 18% of the time, numeral 2 will be the leading digit in that data set. And each subsequent numeral, 3 through 9, will be the leading digit with decreasing frequency.

A card transaction data is an example of naturally occurred data, also its sample size tends to be large enough to make sure Benford's law happens. If a transaction data is made up by a human/machine (fraud), it usually tends to distribute uniformly or in a chaotic pattern. Hence, it would be a good idea to pay more attention to card transaction data that does not follow Benford's law.

First, we used the two entities "Cardnum" and "Merchnum". For each of them, we grouped transactions separately. That is, for each of the entities, we divided it into two bins: a high bin contains the data with the entity's first digit as a number from 3 through 9; a low bin contains the data with the entity's first digit as a number like 1 or 2. The ratio turned out to be 52.3/47.7=1.096 of digits (3 through 9) to (1 and 2).

Second, we quantified how different the first digit distribution is from Benford's law distribution. For each group we made, we counted the number of first digits beginning with either 1 or 2 as n_low. Then the n_high (for the high bin) is just n – n_low. Plus, if n_low or n_high turns out to be zero, we set it to 1 so we will not divide by zero.

Third, if the data follow Benford's law, then R = (1.096 * n_low) / n_high should be about 1. Hence, we got a measure of unusualness U, which is the bigger number between R and 1/R. This number would show us how far away the R was from 1.

Fourth, to be careful about statistics, we smoothed U (the measure of unusualness) for a better measure. We name the new measure as U*, with the formula: (we set c=3 and n_mid=15 as a reasonable choice for the smoothing parameters)

$$U^{*} = 1 + \left(\frac{U-1}{1+exp^{-t}}\right) \qquad t = (n - n_{mid})/c$$

Feature Selection Process

As we have a huge amount of variables, with consideration of computing power in later processes like model building and fine-tuning, we would like to eliminate our variables to the top 30 variables that are most related to our prediction goal, the fraud transactions. We call this process feature selection. We applied filters to filter out variables that are with a small separation between the fraud data and non-fraud data, wrappers to find the variables that make the most significant improvement to our prediction models, and embedded regulation to optimize the model complexity.

Filters

- In the filter process, we score the relatedness of each variable column with the fraud column independently and keep the top 80 variables. In this process, many different methods of ranking can be applied. For a linear correlation, the Pearson correlation can be applied. For non-linear correlations, fisher score can be used to measure how much the two means from fraud and non-fraud data are separated, the Kolmogorov-Smirnov(KS) can be used to measure the distribution and scale independence of the fraud and non-fraud data in each variable. Here we want to test the non-linear relationship between these two subsets of data so we choose KS to test the independence as well as the Fraud DetectionRate(FDR) score. The reason we choose independence over the fisher score is that the KS score distribution does not depend on underlying cumulative distribution functions that are tested while it has a reasonably fast runtime. The reason we choose FDR is that it shows what percentage of all the frauds are caught at a particular examination cutoff location and this is a well-acknowledged method in fraud detection. For example, an FDR 50% at 5% shows that the model catches 50% of all the frauds in 5% of the population.
- Filter steps for KS:
    - Check that variables are continuous or have a metric/ordering
    - Make separate distributions for non-fraud(good) and fraud(bad) populations:
        - Formula to calculate the KS score:
            - $KS = max\, x \int_{xmin}^{x} [P_{good} - P_{bad}]dx$
            - Calculate the universal KS value by finding the maximum distance between the cumulative non-fraud data and cumulative fraud data. Shown in the image below.
        - Example visualization below

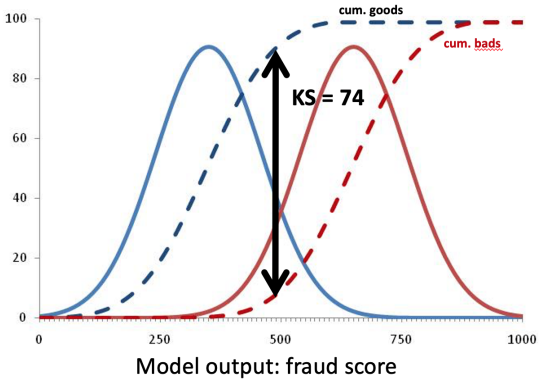


Figure 8. Distribution Plot for KS

- In the plot above, the amount of separation between the distributions is shown. The amount of separation shows the importance of the variables. The more different the curves, the better the variable is separating, the easier we can classify the fraud and non-fraud data, so the more important that variable is to our model. In the KS test, the variables with a high KS score will be kept and those with a low score will be eliminated.

- Filter steps for FDR:
  - Score all records in descending order.

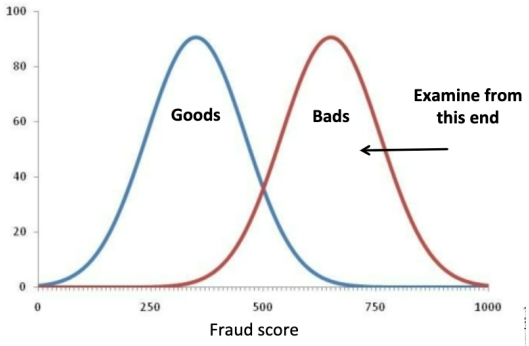


Figure 9. Distribution Plot for FDR

  - Examine the subpopulation of the top 3% of the data from the fraud end, as 3% is the most used measurement in fraud detection. We calculated the number of frauds caught, which is the sum of the fraud score, and then calculated the percentage of fraud caught.
- Take the mean between KS and FDR for the final ranking.

Wrapper

- In order to test how important each variable is in terms of improving our prediction model, we use a wrapper to "wrap" a model around the feature selection process. We can choose the model from many different models including linear and non-linear models. In our project, with consideration of time complexity and memory complexity, we chose a very simple non-linear model, random forest, in our wrapper.
- Here are a few examples of how we can apply a model:
    - Forward selection: In forward selection, we build n(number of variables) separate 1 dimension to n dimension models with greedy algorithms. We begin with one variable. We build all possible models for each variable separately and keep the one that yields the best performance as the first variable. Then we test the combination of this top 1 variable with each of the other variables to find the new variable that gives the best model performance improvement. We repeat this process until all variables are added or no model improvement can be added. With the final list, we sort the variables in terms of how much improvement they can add to the models.
    - Backward selection: In back selection, we build n(number of variables) separate n dimension to 1 dimension models with greedy algorithms. We begin with all variables to have a baseline performance. Then we remove each variable separately to see which one gives the mode decreasing in model performance. With that decrease, that variable is important in our models. We repeat this dropping variable process until no model performance decreases are available or all variables are dropped. With the final list, we sort the variables in terms of how much decrease they can make to the models.

In our project, we selected the forward selection with a random forest method. We choose to use the top 80 variables with the highest mean between KS score and 3% FDR score. We used the feature_selection.SequentialFeatureSelector. With the greedy forward selection method, we can find good subsets of all our variables in terms of linear model performance, while removing correlations. The reason the correlations can be removed is that, for related variables A and B, if we already added A into our model, B will make only a little improvement to our model as all its contributions to the model are pre-used in A. With this method, the outcome for each run of KS score might be different since the different order we add variables to the model might cause different correlation removal order results. Another limitation is that this greedy search is only looking for the next best position given the current location, so since it is not a global search, the global optimum is not guaranteed. Below is the list of our top 30 variables from the wrapper that will be used in the future calculation:

Top 30 variables			
1	card_merch_total_7	16	Merch_Merch_description_total_1
2	card_Merch_description_total_7	17	cardnum_total_7
3	card_Merch_description_total_3	18	Merch_Merch_description_total_0
4	card_zip_total_14	19	card_state_max_1
5	card_zip_total_1	20	merch_state_total_0
6	card_merch_total_1	21	Merch_Merch_description_total_3
7	card_state_total_14	22	cardnum_total_1
8	card_zip_total_30	23	Merch_zip_max_0
9	card_Merch_description_total_0	24	Merch_Merch_description_max_1
10	card_zip_max_30	25	cardnum_total_0
11	card_zip_total_0	26	merch_zip_total_3
12	card_zip_max_7	27	Merch_Merch_description_max_3
13	card_state_max_14	28	Merchnum_total_1
14	card_merch_max_14	29	merch_zip_max_3
15	card_state_max_3	30	cardnum_max_3

Figure 10. Top 30 variables

Model Algorithm

We split the dataset into three parts before training the models: training data, testing data, and out-of-time data. In our project, we took the last four months, between September 1st. 2010 and December 31st, 2010 as our out-of-time data. For the transactions between January 1st, 2010 and August 31st, 2010, we randomly selected 70% of them to be our training data, and the remaining 30% was used as testing data. Then we used the training set to fit the model, trying different combinations of hyperparameters and tuned the model based on the performance of the testing set. Then we chose our model according to the performance of out-of-time data.

For our fraud analysis, we tried four different machine learning algorithms to build supervised models. In each model, we predicted the fraud label, ranked the records by the probability in descending order and calculated the FDR at 3% for training, testing and OOT data separately.

Logistic regression

Logistic regression is one of the simplest and commonly used machine learning algorithms for binary classification problems. It follows a sigmoid function. The sigmoid function gives an 'S' shaped curve that can take any real-valued number and map it into a value between 0 and 1.

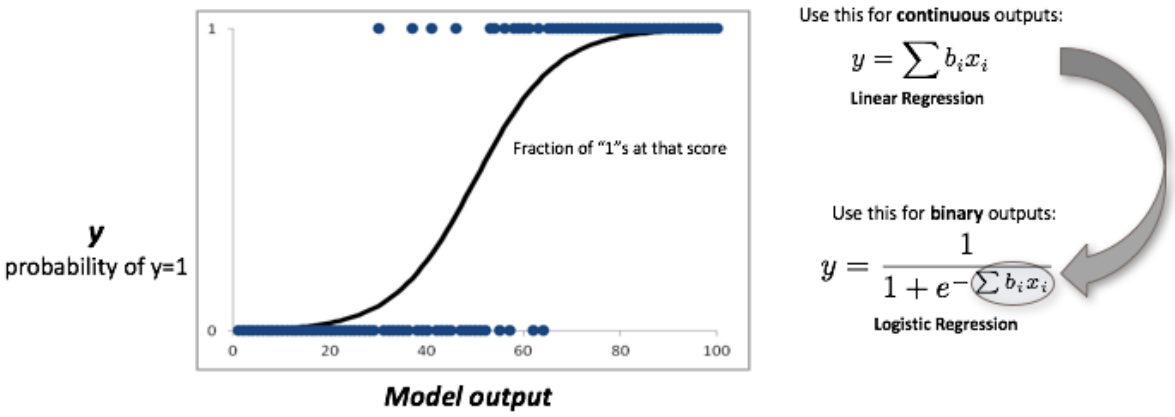


Figure 11. Logistic Regression

When tuning the parameters, the information below provides some general guidelines when attempting to find the best performance of the logistic regression model.

- Penalty: used to specify the norm used in the penalization
- C: inverse of regularization strength, which must be a positive float
- Solver: algorithm to use in the optimization problem.I In particular, the 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.
- L1_ratio: The Elastic-Net mixing parameter, with 0 <= l1_ratio <= 1. Only used if penalty='elasticnet'.

We ran a logistic regression model as a baseline model ten times and took the average FDR at 3%. We used 30 variables and tried different combinations of parameters to locate the model of best performance. Below is the table for our logistic regression model results. In our setting, a logistic regression model with l2 regularization and C being 1 had the highest FDR.

Logistic Regression	Number of Variables	Penalty/Regularization	C	Solver	max_iter	Train FDR	Test FDR	OOT FDR
1	30	l2	1	lbfgs	1000	0.682	0.683	0.51
2	30	l2	0.1	lfgs	1000	0.7	0.653	0.497
3	30	l1	0.1	liblinear	1000	0.68	0.693	0.496

Figure 12. Logistic Regression Results

Neural Network

A neural network consists of multiple layers of neurons: an input layer that is formed by the independent variables, one or more hidden layers, and an output layer that is formed by the dependent variable. Each node in the input layer consists of the information of all selected variables for each record. The nodes in the output layer are either 0 or 1, 0 represents non-fraudulent transactions and 1 represents fraud. For the hidden layer, we choose 1 hidden layer with 5 nodes at the beginning.

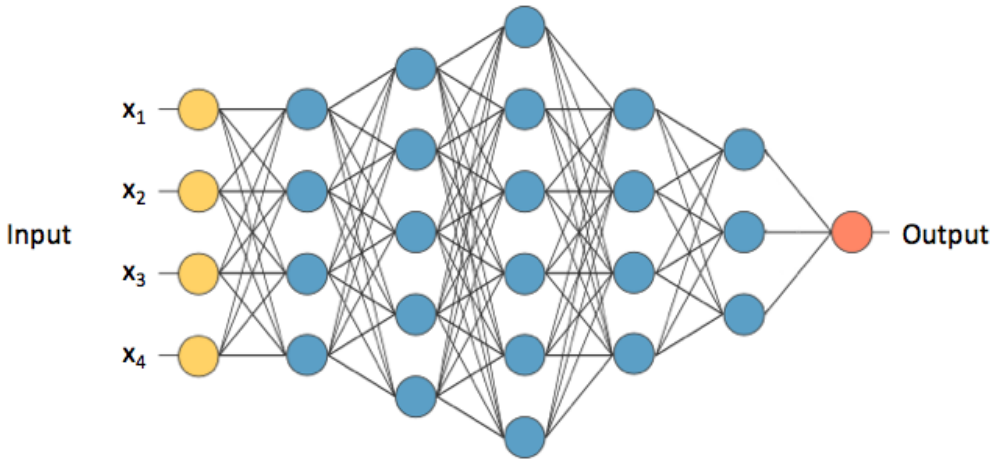


Figure 13. Neural Network

A neural network has a significant number of hyperparameters. To obtain the best results, we tried a grid of parameters to locate the best performance of the model. The parameters we tuned were:

- Number of neurons in hidden layers: we slightly increase the number of hidden nodes from 3 to 20
- Number of hidden layers in the network: up to 2 layers
- Activation function: it serves to scale the input of the neuron and to provide a smooth transition as input values change. In our model, we used the activation function, 'relu' and 'logistic'.
- Learning rate: parameters that affect the convergence time of the optimization algorithm. We tried constant and adaptive learning rates for tuning.
- Optimization algorithm: used to train the network and minimize the loss function. We set it as default.

For the training process, initially, we chose a simple network architecture to detect fraud, one hidden layer containing 3 nodes, and slightly increased the number of nodes from 3 to 20. From the result, we could see that a network with one hidden layer with 5 nodes produced better FDR results compared with more nodes. Then we tried to increase the size of the hidden layer size to 2.

For example, we trained some neural networks with two hidden layers, where both hidden layers contained nodes from 5,5 to 10,10 to 15,15. Unfortunately, these more complicated neural networks didn't perform better than simple networks in terms of the performance of OOT FDR results.

We also tried to tune the algorithm's learning rate, which is also an important step in neural network training, because it affects the algorithms' ability to converge to the loss function's global optimum which in turn significantly affects the mode's results. In our model, we tried constant and adaptive learning rate in the neural networks model.

Neural Network	Number of variables	Layer	Nodes	Activation	Solver	learning_rate	max_iter	Train FDR	Test FDR	OOT FDR
1	30	1	3	logistic	sgd	constant	1000	0.724	0.725	0.518
2	30	1	5	logistic	sgd	adaptive	1000	0.78	0.751	0.503
3	30	1	20	logistic	sgd	adaptive	1000	0.855	0.806	0.496
4	30	1	10	relu	adam	adaptive	1000	0.68	0.654	0.429
5	30	2	5,5	logistic	adam	adaptive	1000	0.677	0.661	0.532
6	30	2	10,10	relu	sgd	adaptive	1000	0.874	0.817	0.453
7	30	2	15,15	logistic	adam	constant	1200	0.86	0.809	0.483
8	30	2	20,20	logistic	sgd	adaptive	1200	0.863	0.825	0.484
9	30	2	3,3	logistic	sgd	constant	1000	0.757	0.745	0.531

Figure 14. Neural Network Results

Finally, below are the results of our training process for the neural network. We could see that the higher the model complexity the lower is the model performance in terms of FDR results on the OOT dataset. In our setting, a simple neural network with two layers and 5 nodes for each resulted in the highest FDR.

Gradient Boosting

Gradient boosting is a supervised learning algorithm that can ensemble multiple weak learners to produce a robust model for the regression and classification problem. And in the case of gradient boosting, the weak learners are decision trees. In order to improve the model, gradient boosting combines weak decision trees in the way that each new learner fits the residuals from the previous step. The final model ensembles the results from every step to achieve a strong learner. In the meanwhile, a loss function is detecting the residuals. For example, the logarithmic loss can be used for the classification problem.

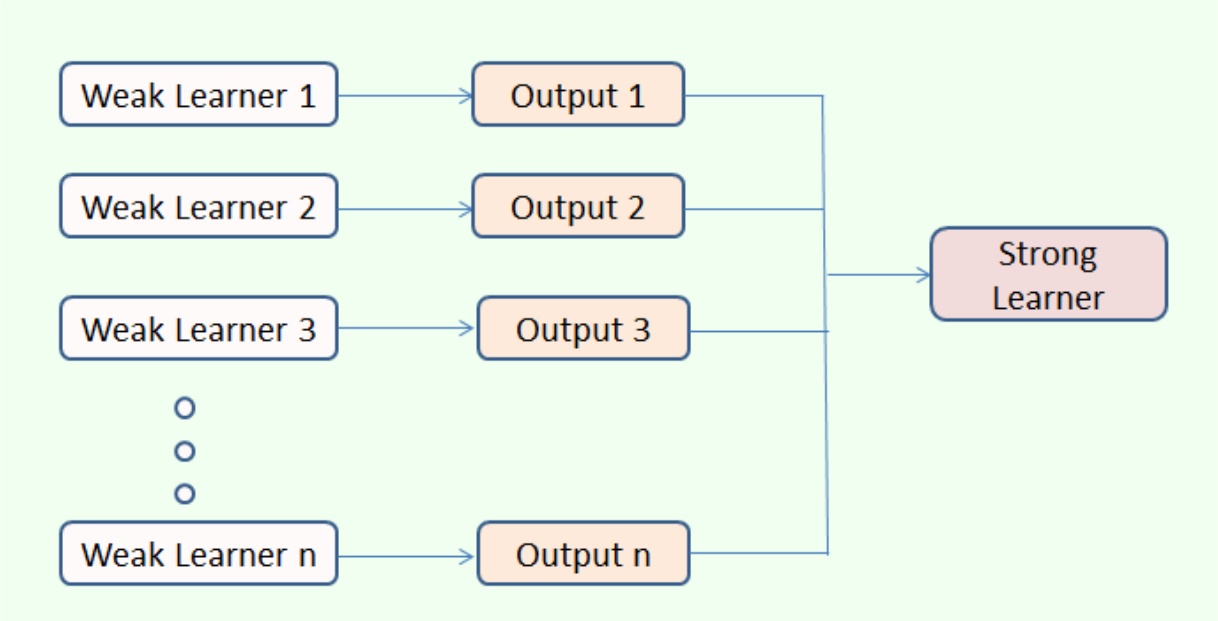


Figure 15. Gradient Boosting

In this project, we were trying to predict whether the transaction is a fraud or not given some variables we created before. The general steps that underlie the gradient boosting algorithm can be described as follows.

- Step I: Get initial prediction

The gradient boosting model first started with a leaf which represents an initial prediction for every transaction. This leaf would be used as a baseline to approach the correct solution in the proceeding steps. For classification, this is equal to the log(odds) of the dependent variable. Then, it was converted to a probability using the logistic function, which got us the initial prediction.

- Step 2: Calculate residuals

After that, the algorithm calculated the residual with the formula residual = actual value - predicted value.

- Step 3: Predict residuals

In the next step, a decision tree is built to predict the residual using the variables we created.

IV. Get a new prediction

The predicted residual will be multiplied by a learning rate and then added to the previous prediction to obtain whether the transaction is a fraud or not. Then

- Step 4: Obtain new residuals

After obtaining the new prediction for all the transactions, we will calculate the new residuals by subtracting these new predicted values from the actual values. After we have the new residuals, we will use these leaves for building the next decision tree as described in step III.

- Step 5: Repeat

Repeat steps 3 to 5 until the residuals converge to a value close to 0 or the number of iterations matches the value given as a hyperparameter while running the algorithm.

While we constructed a gradient boosting model, the values of the following were modified to improve model performance.

- The number of trees (n_estimators): Gradient boosting is fairly robust to overfitting, so a large number usually leads to better performance.
- learning rate (learning_rate): Scales the contribution of each tree as discussed before. There is a trade-off between the learning rate and the number of trees.
- maximum depth (max_depth): Maximum depth of each estimator. It limits the number of nodes of the decision trees
- max_features: The number of features to consider when looking for the best split.

We ran 10 iterations with different hyperparameter combinations because it can help prevent the inherent variances that occur with random initializations.

The results are shown in the figure below.

Gradient Boosting	Number of variables	n_estimators	max_depth	learning_rate	max_features	Train FDR	Test FDR	OOT FDR
1	30	300	3	0.1	5	0.89	0.834	0.638
2	30	100	3	0.1	10	0.955	0.851	0.625
3	30	500	4	0.1	5	0.998	0.861	0.593
4	30	600	4	0.1	10	0.901	0.794	0.54
5	30	700	4	0.01	10	0.911	0.834	0.644
6	30	800	5	0.01	10	0.957	0.861	0.637
7	30	900	5	0.01	15	0.956	0.879	0.63

Figure 16. Gradient Boosting Results

We could see that in our setting, a gradient boosting model with 700 trees and a 0.01 learning rate resulted in the highest FDR.

Random Forest

Random Forest algorithm is a supervised classification algorithm that creates a forest of decision trees and makes it randomly. A direct relationship between the number of trees in the forest and the results it can get is that the larger the number of trees, the more accurate the result.

Like gradient boosting, random forest uses the method called an ensemble. Ensemble methods involve using many learners to enhance the performance of any single one of them individually. These methods can be described as techniques that use a group of weak learners (those who on average achieve only slightly better results than a random model) together, in order to create a stronger, aggregated one. Random forest is an ensemble of many individual decision trees.

But the main differences between gradient boosting and random forest are that random forest builds each tree independently while gradient boosting builds one tree at a time. This additive model (ensemble) works in a forward stage-wise manner, introducing a weak learner to improve the shortcomings of existing weak learners. Also, random forest combines results at the end of the process (by averaging or "majority rules") while gradient boosting combines results along the way.

Random forest is an ensemble of many individual decision trees. One main drawback of decision trees is that they are very likely to overfit. They do well on training data but are not so flexible for making predictions on unseen samples. Even though there are workarounds for this, for example, we could apply it to prune the trees, the overfitting problem reduces random forests' predictive power. Generally, random forest models have medium bias and high variance, but they are simple and easy to interpret.

The Random Forest models combine the simplicity of decision trees with the flexibility and power of an ensemble model. In a forest of trees, we forget about the high variance of a specific tree and are less concerned about each individual element, so we can grow nicer, larger trees that have more predictive power than a pruned one.

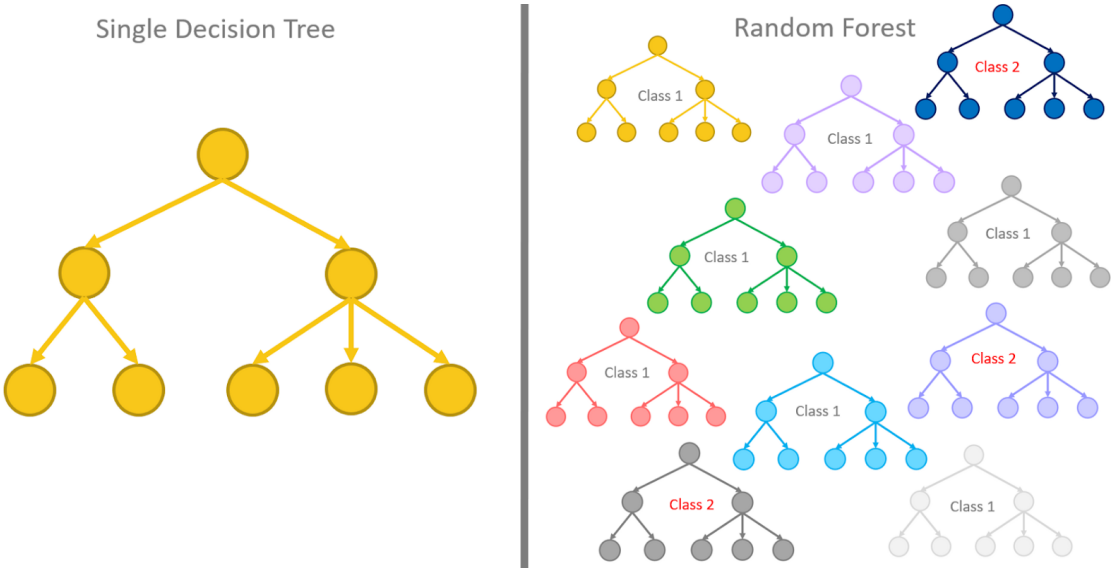


Figure 17. Random Forest

Although Random Forest models don't offer as much interpretability as a single tree, their performance is a lot better, and we don't have to worry so much about perfectly tuning the parameters of the forest as we do with individual trees.

While we constructed a random forest model, the values of the following were modified to improve model performance.

- n_estimators: The number of trees in the forest.
- max_depth: The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.
- min_samples_split: The minimum number of samples required to split an internal node
- min_samples_leaf: The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

We ran 10 iterations with different hyperparameter combinations. The results are shown in the figure below.

Random Forest	Number of variables	n_estimators	max_depth	min_samples_split	min_samples_leaf	Train FDR	Test FDR	OOT FDR
1	30	30	None	2	1	1	0.871	0.631
2	30	60	10	2	3	0.89	0.823	0.646
3	30	60	10	2	5	0.882	0.829	0.662
4	30	90	10	2	3	0.893	0.829	0.654
5	30	90	8	4	8	0.846	0.811	0.65
6	30	120	10	2	3	0.892	0.822	0.654
7	30	120	10	5	5	0.89	0.83	0.659
8	30	120	20	5	5	1	0.857	0.667
9	30	150	25	5	5	1	0.86	0.669
10	30	150	30	8	8	1	0.849	0.67

Figure 18. Random Forest Results

We could see that in our setting, a random forest model with 150 trees and 25 max_depth resulted in the highest FDR.

Result

The analysis from training models with Logistic Regression, Boosting Trees, Random Forests, and Neural Network from the previous session concluded that Random Forests has the best performance. Random Forests resulted in the highest fraud detection rate for both the testing and out-of-time validation dataset, and also the lowest variance overall during hyperparameter tuning. The Random Forest model we proceed with has 150 estimators, max_depth of 30, min_samples_split of 8, min_samples_lead of 8, and all of our top 30 variables, which reached 85% FDR at 3% for the test data and 67% FDR at 3% for the oot data.

The following 3 tables are the results from training, testing, and oot dataset, which includes bin and cumulative statistics from using our Random Forest model to detect fraud from 1% to 20% population. The number of good, bad, summed records and corresponding percentages are presented.

Training	# Records	# Goods	# Bads	Fraud Rate
	48332	47857	475	0.0105

	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	483	68	415	14.07867	85.92133	483	68	415	0.142111	86.09959	85.95747	0.163855
2	483	422	61	87.3706	12.6294	966	490	476	1.024033	98.75519	97.73115	1.029412
3	483	477	6	98.75776	1.242236	1449	967	482	2.020899	100	97.9791	2.006224
4	483	483	0	100	0	1932	1450	482	3.030303	100	96.9697	3.008299
5	483	483	0	100	0	2415	1933	482	4.039707	100	95.96029	4.010373
6	483	483	0	100	0	2898	2416	482	5.049112	100	94.95089	5.012448
7	483	483	0	100	0	3381	2899	482	6.058516	100	93.94148	6.014523
8	483	483	0	100	0	3864	3382	482	7.067921	100	92.93208	7.016598
9	483	483	0	100	0	4347	3865	482	8.077325	100	91.92268	8.018672
10	483	483	0	100	0	4830	4348	482	9.086729	100	90.91327	9.020747
11	483	483	0	100	0	5313	4831	482	10.09613	100	89.90387	10.02282
12	483	483	0	100	0	5796	5314	482	11.10554	100	88.89446	11.0249
13	483	483	0	100	0	6279	5797	482	12.11494	100	87.88506	12.02697
14	483	483	0	100	0	6762	6280	482	13.12435	100	86.87565	13.02905
15	483	483	0	100	0	7245	6763	482	14.13375	100	85.86625	14.03112
16	483	483	0	100	0	7728	7246	482	15.14316	100	84.85684	15.0332
17	483	483	0	100	0	8211	7729	482	16.15256	100	83.84744	16.03527
18	483	483	0	100	0	8694	8212	482	17.16196	100	82.83804	17.03734
19	483	483	0	100	0	9177	8695	482	18.17137	100	81.82863	18.03942
20	483	483	0	100	0	9660	9178	482	19.18077	100	80.81923	19.04149

Figure 19. Best Model Training Data Set Result

Test	# Records	# Goods	# Bads	Fraud Rate
	20715	20487	228	0.0095

	Bin Statistics					Cumulative Statistics					
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	207	50	157	24.15459	75.84541	207	50	157	0.243974	71.04072	70.79675	0.318471
2	207	186	21	89.85507	10.14493	414	236	178	1.151557	80.54299	79.39143	1.325843
3	207	199	8	96.13527	3.864734	621	435	186	2.122572	84.1629	82.04032	2.33871
4	207	202	5	97.58454	2.415459	828	637	191	3.108227	86.42534	83.31711	3.335079
5	207	199	8	96.13527	3.864734	1035	836	199	4.079243	90.04525	85.96601	4.201005
6	207	202	5	97.58454	2.415459	1242	1038	204	5.064897	92.30769	87.2428	5.088235
7	207	204	3	98.55072	1.449275	1449	1242	207	6.06031	93.66516	87.60485	6
8	207	206	1	99.51691	0.483092	1656	1448	208	7.065483	94.11765	87.05216	6.961538
9	207	207	0	100	0	1863	1655	208	8.075534	94.11765	86.04211	7.956731
10	207	207	0	100	0	2070	1862	208	9.085586	94.11765	85.03206	8.951923
11	207	207	0	100	0	2277	2069	208	10.09564	94.11765	84.02201	9.947115
12	207	207	0	100	0	2484	2276	208	11.10569	94.11765	83.01196	10.94231
13	207	206	1	99.51691	0.483092	2691	2482	209	12.11086	94.57014	82.45927	11.8756
14	207	207	0	100	0	2898	2689	209	13.12091	94.57014	81.44922	12.86603
15	207	206	1	99.51691	0.483092	3105	2895	210	14.12609	95.02262	80.89654	13.78571
16	207	207	0	100	0	3312	3102	210	15.13614	95.02262	79.88649	14.77143
17	207	207	0	100	0	3519	3309	210	16.14619	95.02262	78.87644	15.75714
18	207	207	0	100	0	3726	3516	210	17.15624	95.02262	77.86638	16.74286
19	207	207	0	100	0	3933	3723	210	18.16629	95.02262	76.85633	17.72857
20	207	207	0	100	0	4140	3930	210	19.17634	95.02262	75.84628	18.71429

Figure 20. Best Model Test Data Set Result

OOT	# Records	# Goods	# Bads	Fraud Rate
	27351	26995	356	0.013

	Bin Statistics					Cumulative Statistics					
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	274	114	160	41.60584	58.39416	274	114	160	0.4223	44.94382	44.52152	0.7125
2	274	226	48	82.48175	17.51825	548	340	208	1.259492	58.42697	57.16747	1.634615
3	274	259	15	94.52555	5.474453	822	599	223	2.218929	62.64045	60.42152	2.686099
4	274	264	10	96.35036	3.649635	1096	863	233	3.196888	65.44944	62.25255	3.703863
5	274	261	13	95.25547	4.744526	1370	1124	246	4.163734	69.10112	64.93739	4.569106
6	274	266	8	97.08029	2.919708	1644	1390	254	5.149102	71.34831	66.19921	5.472441
7	274	269	5	98.17518	1.824818	1918	1659	259	6.145583	72.75281	66.60723	6.405405
8	274	268	6	97.81022	2.189781	2192	1927	265	7.138359	74.4382	67.29984	7.271698
9	274	268	6	97.81022	2.189781	2466	2195	271	8.131135	76.1236	67.99246	8.099631
10	274	269	5	98.17518	1.824818	2740	2464	276	9.127616	77.52809	68.40047	8.927536
11	274	271	3	98.90511	1.094891	3014	2735	279	10.13151	78.37079	68.23928	9.802867
12	274	270	4	98.54015	1.459854	3288	3005	283	11.13169	79.49438	68.36269	10.61837
13	274	268	6	97.81022	2.189781	3562	3273	289	12.12447	81.17978	69.05531	11.32526
14	274	272	2	99.27007	0.729927	3836	3545	291	13.13206	81.74157	68.60951	12.18213
15	274	270	4	98.54015	1.459854	4110	3815	295	14.13225	82.86517	68.73292	12.9322
16	274	273	1	99.63504	0.364964	4384	4088	296	15.14355	83.14607	68.00252	13.81081
17	274	271	3	98.90511	1.094891	4658	4359	299	16.14743	83.98876	67.84133	14.5786
18	274	272	2	99.27007	0.729927	4932	4631	301	17.15503	84.55056	67.39553	15.38538
19	274	273	1	99.63504	0.364964	5206	4904	302	18.16633	84.83146	66.66513	16.23841
20	274	273	1	99.63504	0.364964	5480	5177	303	19.17763	85.11236	65.93473	17.08581

Figure 21. Best Model OOT Data Set Result

Fraud Savings Plot

To calculate our model's business value, we utilized the oot data to find out the most profitable cutoff point. The graph below includes Fraud Savings, Lost Sales, and Overall Savings plots.

- Fraud Savings: $2000 gain for each fraud caught
  2000 * number of true positive

- Lost Sales: $50 lost for each wrongly caught transaction
  50 * number of false positive

- Overall savings = Fraud Savings - Lost Sales

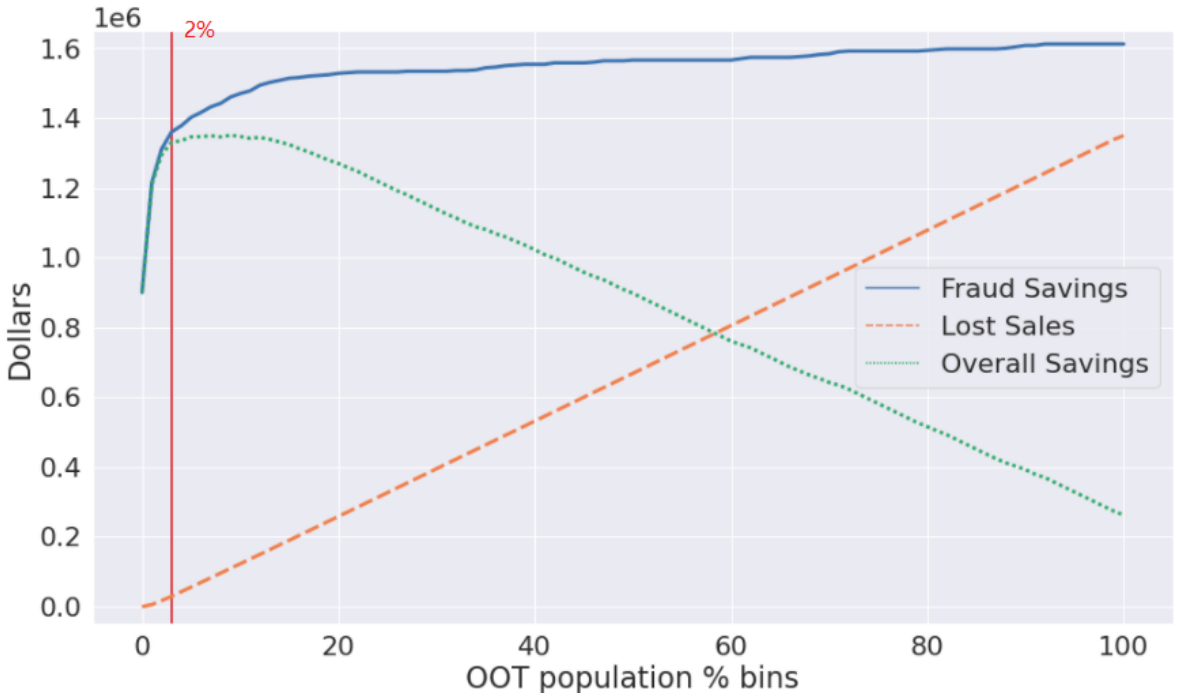


Figure 22. Money Saving of Model

In order to disturb as few innocent customers as possible, we set the cut-off as 2%.

The overall savings of applying our best Random Forest model with oot data suggest that a cut-off point would achieve the maximum overall savings of $1339250 dollars.

Time Plot

Two charts below illustrate why our model cannot detect fraud immediately. When abnormal transactions happen a lot within a short period of time, the average fraud score will increase dramatically, and thus the fraud will be caught.

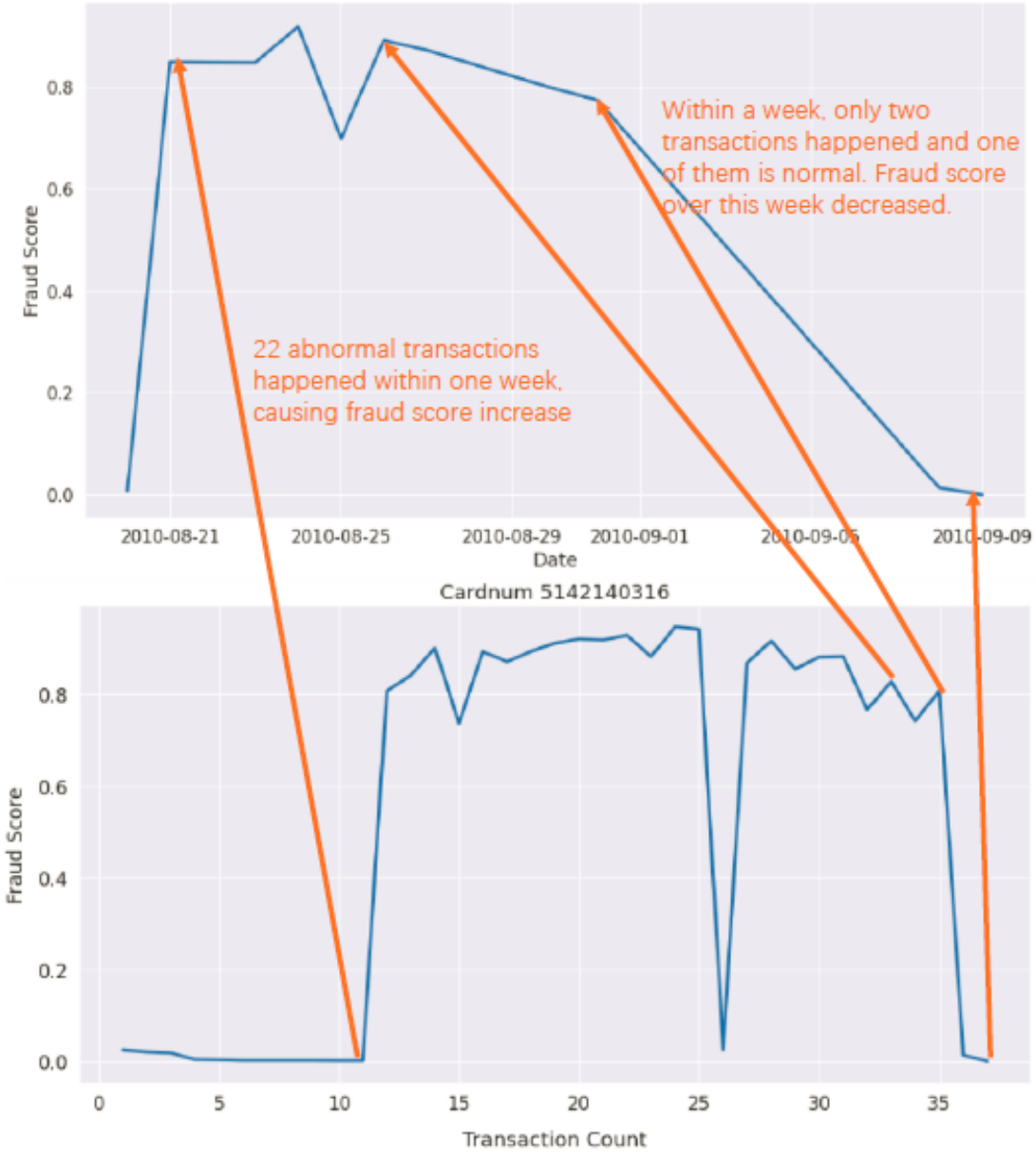


Figure 23. Fraud score change in terms of time and transaction amount (Cardholder 5142140316)

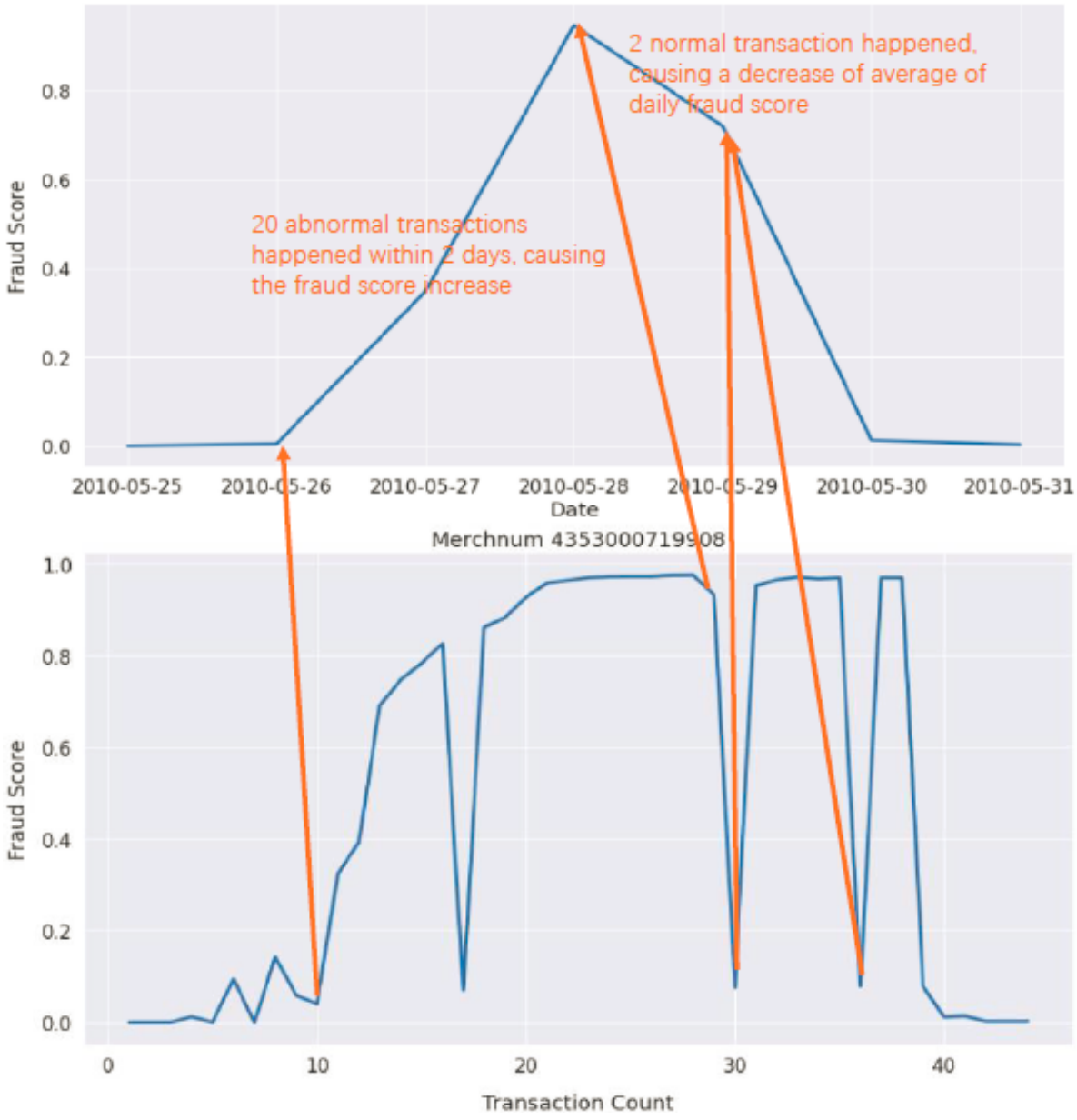


Figure 24. Fraud score change in terms of time and transaction amount (Merchant 4353000719908)

Conclusion

Achieved

We performed a comprehensive analysis, built and selected the optimal supervised fraud detecting model with the Card Transaction Dataset.

Started with building DQR report, we got a general understanding of the data. we then moved to data cleaning to identify exclusion and bad records that would possibly hinder the prediction accuracy. Data were cleaned and missing values in fields “Merchnum”, “Merch zip”, and “Merch state” were filled with different tactics accordingly. Next, we made over 500 candidate variables incorporating time, amount, card number, merchant number, description, and location, etc. The candidate variables are sent to feature selection using the filter and wrapper method to get the best 30 variables. We split the data so that we trained and test the models with the first 8 months, with 70/30 as train/test ratio, and set the last 4 months of data as out-of-time data.

We build 4 supervised fraud detecting models including Logistic Regression, Boosting Trees, Random Forests, and Neural Network, and proceed with Random Forests due to its best performance shown by the fraud detection rate, with 85% FDR at 3% for the test data and 67% FDR at 3% for the oot data.

Looking forward

Given more time and resources, we could do more data manipulation such as reduce seasonality, do downsampling and upsampling to reach a more balanced dataset, which might both arm the prediction. Further, we could modify models to minimize the disparate impact or investigate the reasoning behind the best-performed model, Random Forest, and further improve the fraud detection rate. Finally, we can explore more models and consult fraud experts for industry insights.

Appendix A: Data Quality Report (DQR)

Description

Dataset Name: Card Transaction Data
Dataset Description: Actual credit card purchases' information
Dataset Purpose: The data is about credit card transaction information, which indicates whether this transaction is a fraud or not.
Data Source: US government organization
Time Period: January 1st, 2010 to December 31st, 2010
Number of Fields: #10
Number of Records: 96,753

Summary Table

Numeric Fields:

	count	mean	std	min	25%	50%	75%	max	Unique Value	# Zero	% populated
Amount	96753.00	427.89	10006.14	0.01	33.48	137.98	428.20	3102045.53	34909	0	100.00

Categorical Fields:

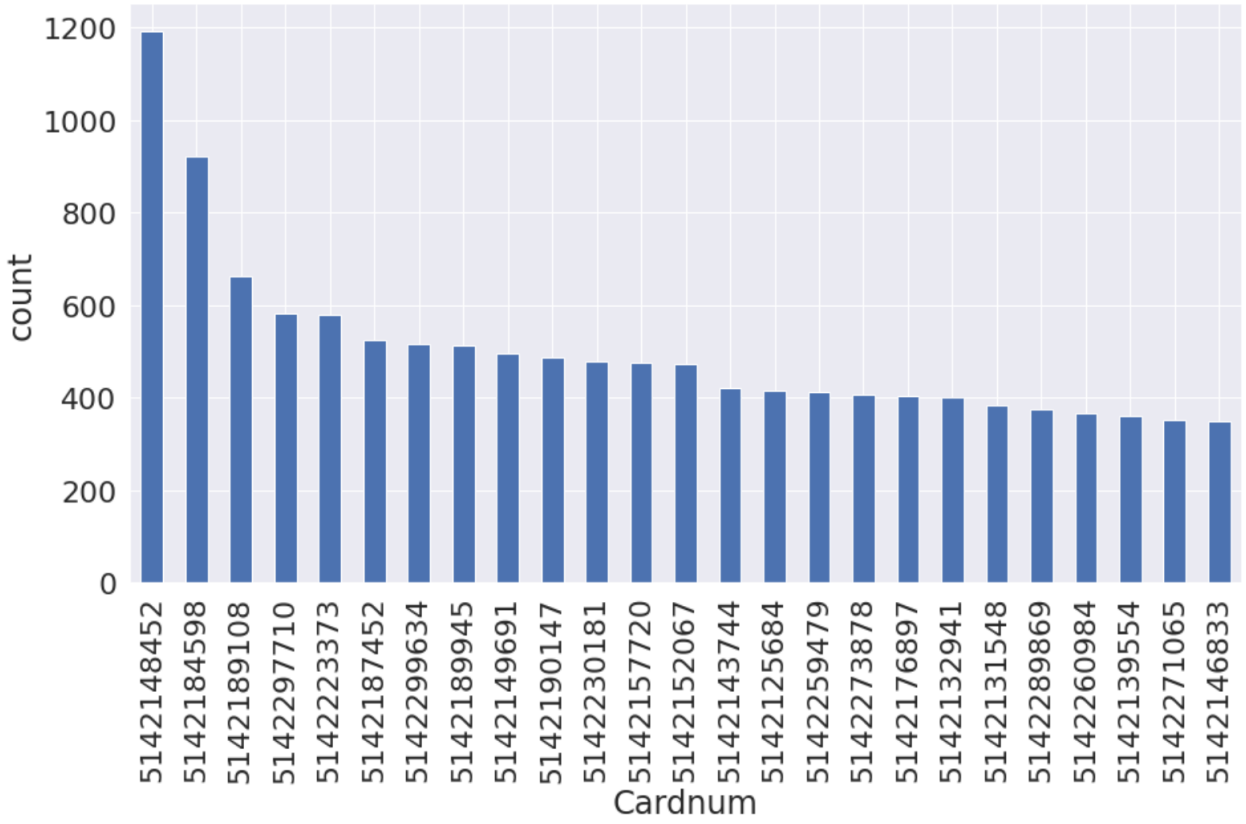
	count	% populated	Unique Values	Most Common Field Values
Recnum	96753	100.00	96753	1
Cardnum	96753	100.00	1645	5142148452
Date	96753	100.00	365	2/28/10
Merchnum	93378	96.51	13091	930090121224
Merch description	96753	100.00	13126	GSA-FSS-ADV
Merch state	95558	98.76	227	TN
Merch zip	92097	95.19	4567	38118.0
Transtype	96753	100.00	4	P
Fraud	96753	100.00	2	0

Data Field Exploration:

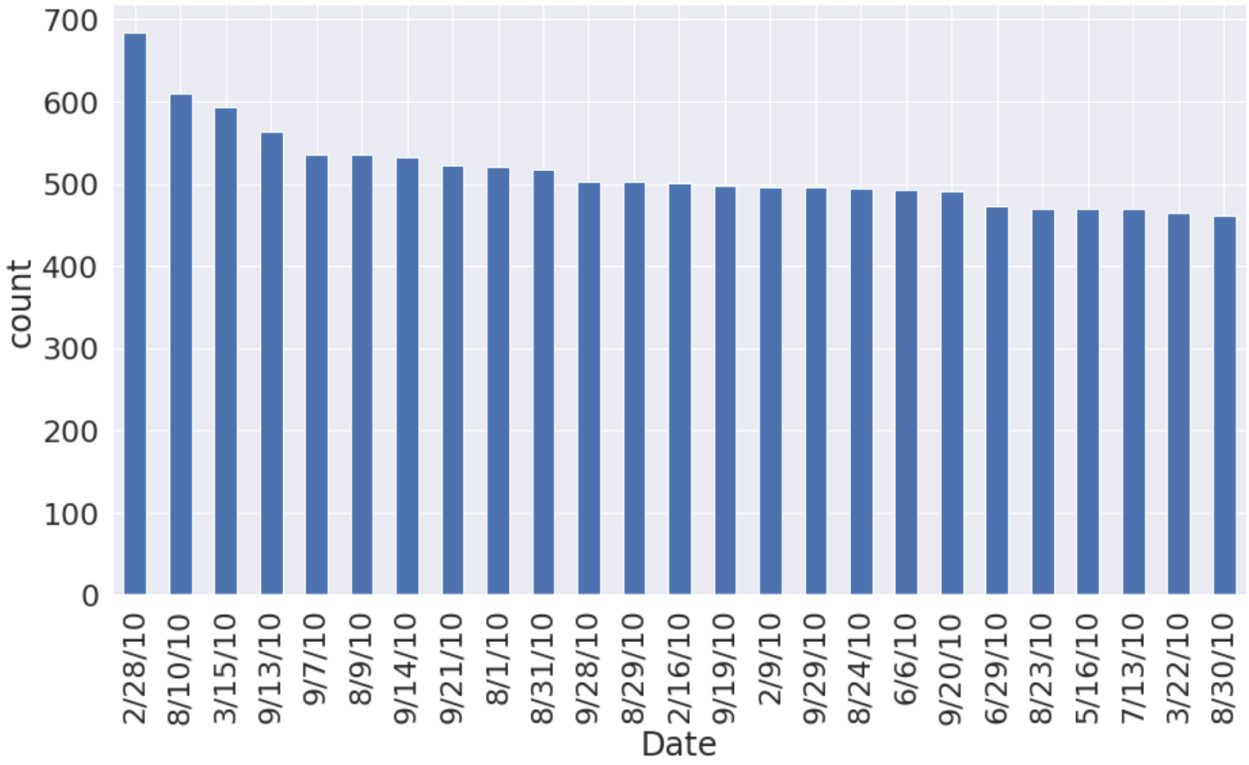
Field 1:
Name: Recnum
Description: Time order of the data.

Field 2:

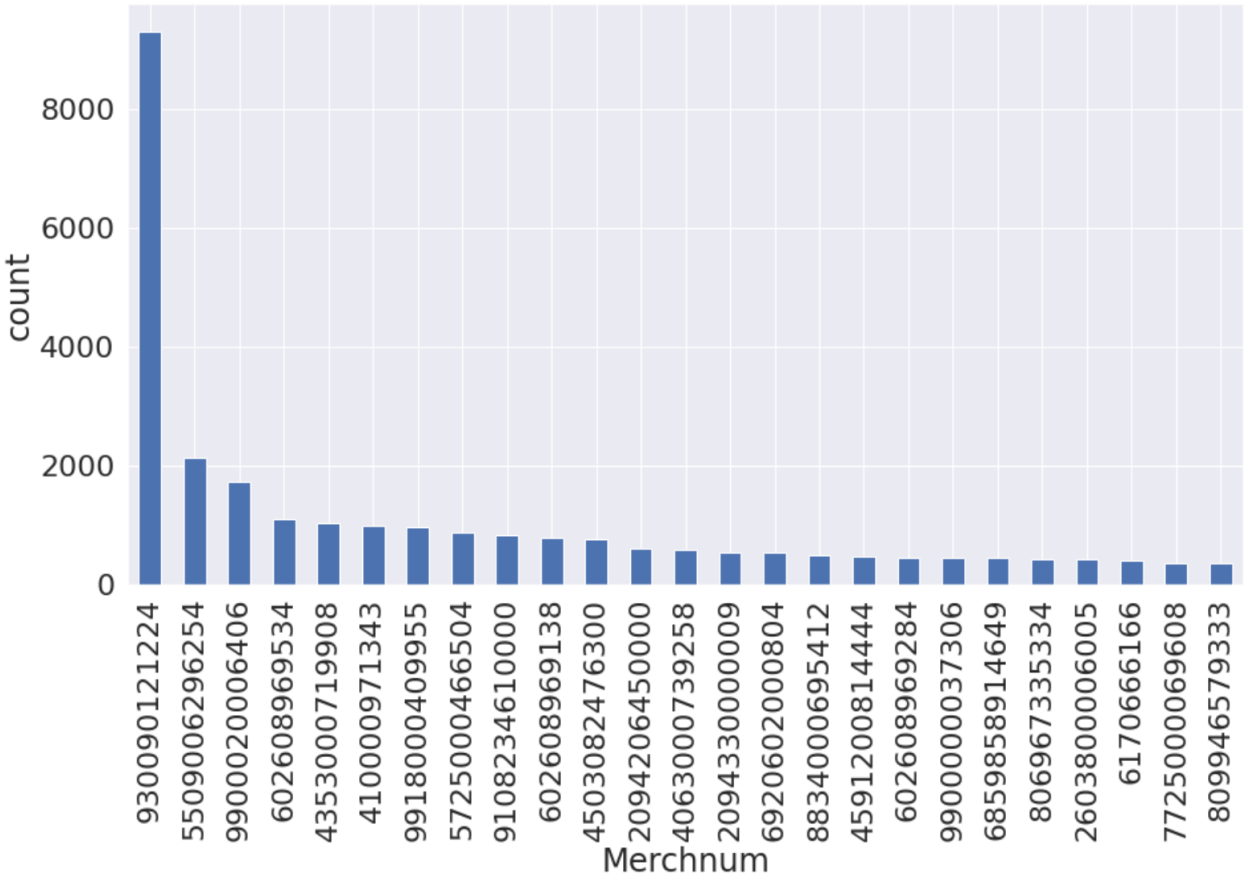
Name: Cardnum
Description: Card number of the transaction.



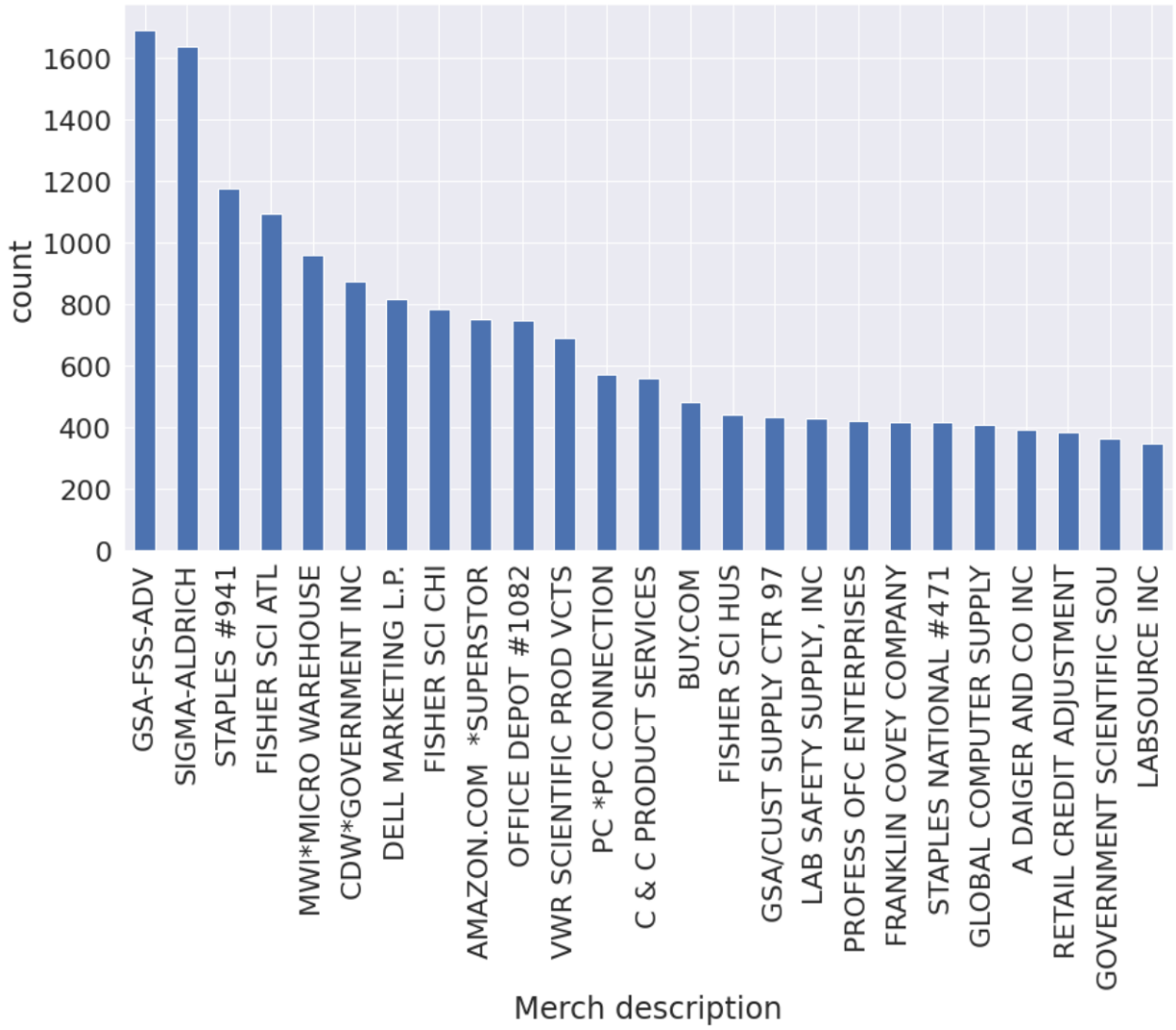
Field 3:
Name: Date
Description: Date of the transaction.



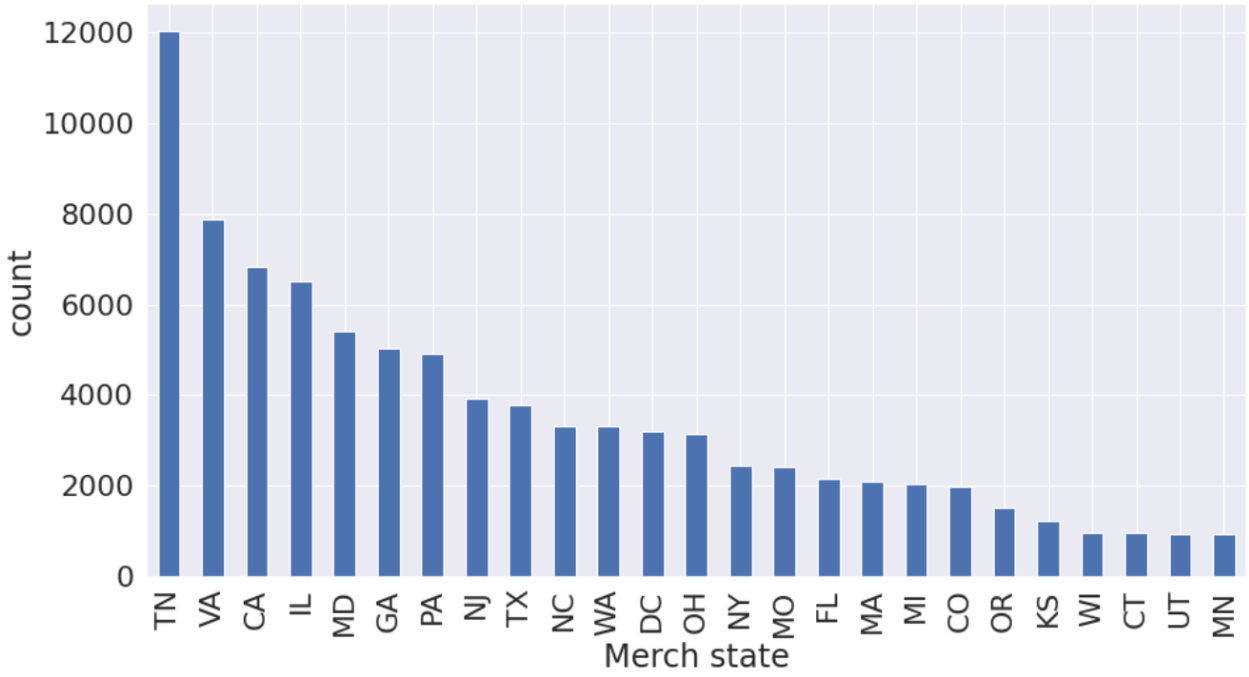
Field 4:
Name: Merchnum
Description: Merchant number of the transaction.



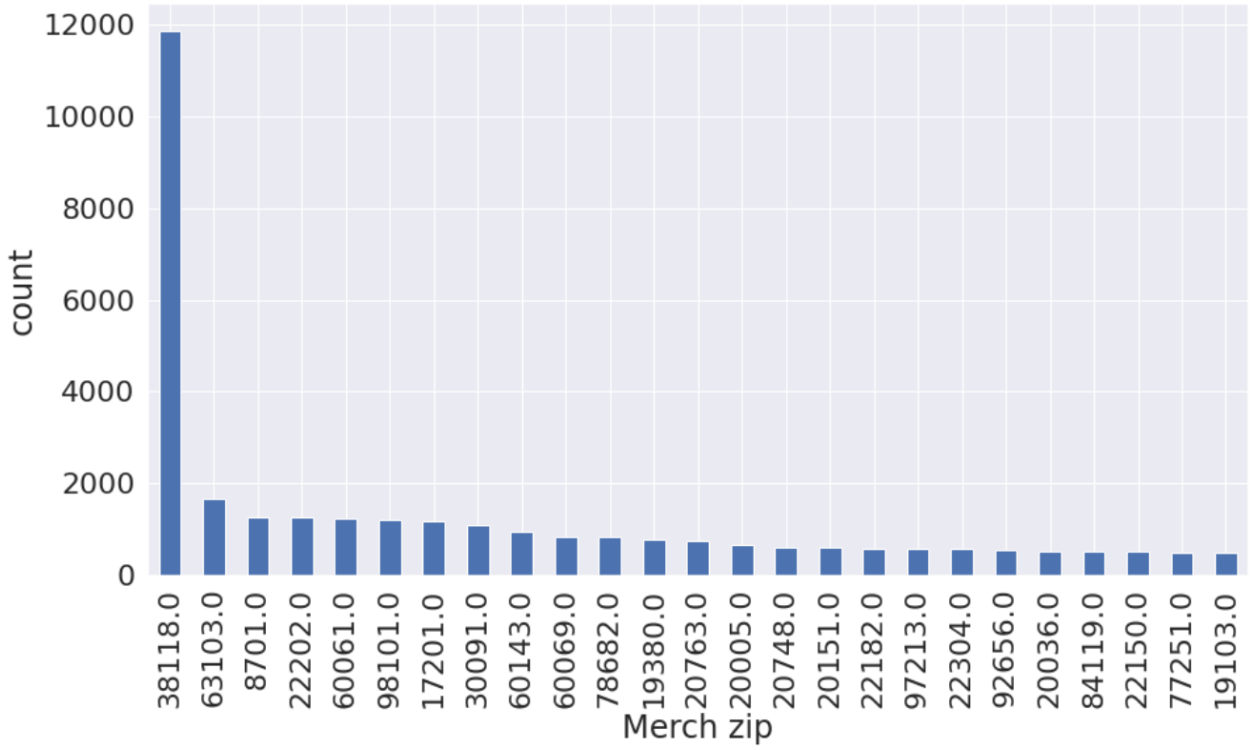
Field 5:
Name: Merch description
Description: Merchant Company Name.



Field 6:
Name: Merch State
Description: State where the Merchant in.

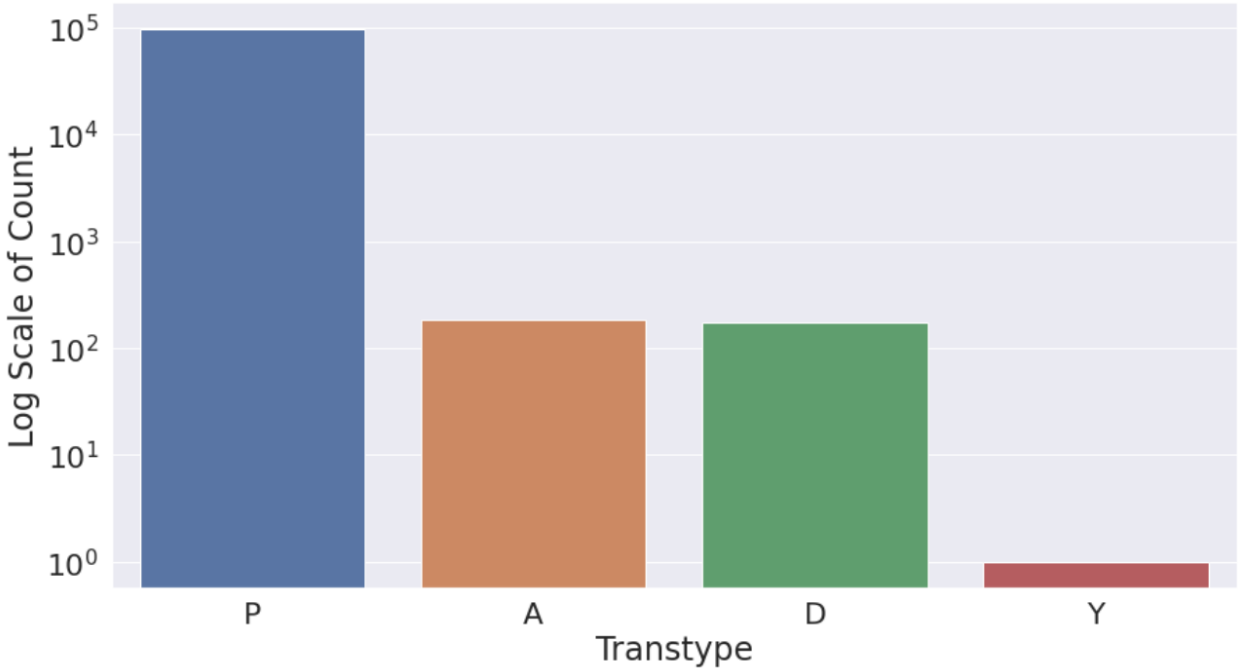


Field 7:
Name: Merch zip
Description: The zip code of Merchant.

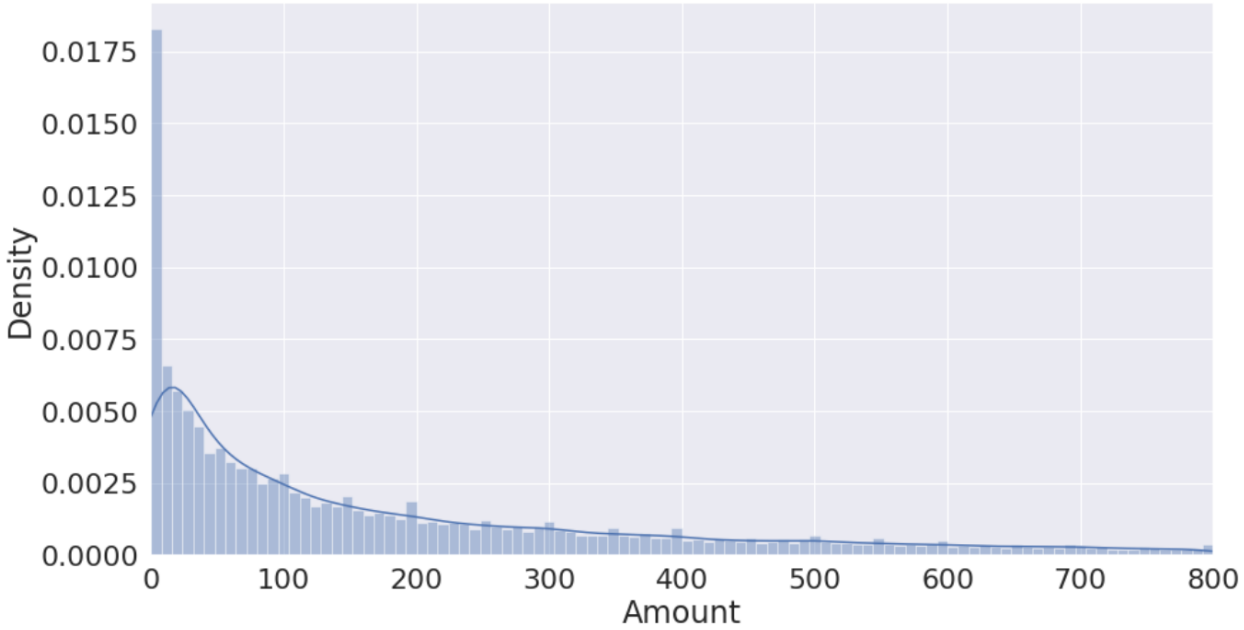


Field 8:

Name: Transtype
Description: Transaction type of each transaction.



Field 9:
Name: Amount
Description: The amount of money of each transaction, exclude outliers > 1000, data in the histogram is 89.10% populated.



Field 10:
Name: Fraud
Description: Whether this transaction is fraud or not (‘1’ = Fraud, ‘0’ = Not fraud)

Appendix B: Created Variables

Candidate Variables			
1	Amount	286	Cardnum_count_1_by_14
2	card_merch	287	Cardnum_count_1_by_30
3	card_merch_actual/avg_0	288	Cardnum_count_1_by_7
4	card_merch_actual/avg_1	289	Cardnum_count_14
5	card_merch_actual/avg_14	290	Cardnum_count_3
6	card_merch_actual/avg_3	291	Cardnum_count_30
7	card_merch_actual/avg_30	292	Cardnum_count_7
8	card_merch_actual/avg_7	293	Cardnum_day_since
9	card_merch_actual/max_0	294	Cardnum_max_0
10	card_merch_actual/max_1	295	Cardnum_max_1
11	card_merch_actual/max_14	296	Cardnum_max_14
12	card_merch_actual/max_3	297	Cardnum_max_3
13	card_merch_actual/max_30	298	Cardnum_max_30
14	card_merch_actual/max_7	299	Cardnum_max_7
15	card_merch_actual/med_0	300	Cardnum_med_0
16	card_merch_actual/med_1	301	Cardnum_med_1
17	card_merch_actual/med_14	302	Cardnum_med_14
18	card_merch_actual/med_3	303	Cardnum_med_3
19	card_merch_actual/med_30	304	Cardnum_med_30
20	card_merch_actual/med_7	305	Cardnum_med_7
21	card_merch_actual/toal_0	306	Cardnum_total_0

22	card_merch_actual/toal_1	307	Cardnum_total_1
23	card_merch_actual/toal_14	308	Cardnum_total_14
24	card_merch_actual/toal_3	309	Cardnum_total_3
25	card_merch_actual/toal_30	310	Cardnum_total_30
26	card_merch_actual/toal_7	311	Cardnum_total_7
27	card_merch_avg_0	312	Date
28	card_merch_avg_1	313	Fraud
29	card_merch_avg_14	314	fraud_prop_state
30	card_merch_avg_3	315	fraud_prop_weekdays
31	card_merch_avg_30	316	Merch description
32	card_merch_avg_7	317	Merch state
33	card_merch_count_0	318	Merch zip
34	card_merch_count_0_by_14	319	Merch_Merch_description
35	card_merch_count_0_by_30	320	Merch_Merch_description_actual/avg_0
36	card_merch_count_0_by_7	321	Merch_Merch_description_actual/avg_1
37	card_merch_count_1	322	Merch_Merch_description_actual/avg_14
38	card_merch_count_1_by_14	323	Merch_Merch_description_actual/avg_3
39	card_merch_count_1_by_30	324	Merch_Merch_description_actual/avg_30
40	card_merch_count_1_by_7	325	Merch_Merch_description_actual/avg_7
41	card_merch_count_14	326	Merch_Merch_description_actual/max_0
42	card_merch_count_3	327	Merch_Merch_description_actual/ma

			x_1
43	card_merch_count_30	328	Merch_Merch_description_actual/max_14
44	card_merch_count_7	329	Merch_Merch_description_actual/max_3
45	card_merch_day_since	330	Merch_Merch_description_actual/max_30
46	Card_Merch_description	331	Merch_Merch_description_actual/max_7
47	Card_Merch_description_actual/avg_0	332	Merch_Merch_description_actual/med_0
48	Card_Merch_description_actual/avg_1	333	Merch_Merch_description_actual/med_1
49	Card_Merch_description_actual/avg_14	334	Merch_Merch_description_actual/med_14
50	Card_Merch_description_actual/avg_3	335	Merch_Merch_description_actual/med_3
51	Card_Merch_description_actual/avg_30	336	Merch_Merch_description_actual/med_30
52	Card_Merch_description_actual/avg_7	337	Merch_Merch_description_actual/med_7
53	Card_Merch_description_actual/max_0	338	Merch_Merch_description_actual/toal_0
54	Card_Merch_description_actual/max_1	339	Merch_Merch_description_actual/toal_1
55	Card_Merch_description_actual/max_14	340	Merch_Merch_description_actual/toal_14
56	Card_Merch_description_actual/max_3	341	Merch_Merch_description_actual/toal_3
57	Card_Merch_description_actual/max_30	342	Merch_Merch_description_actual/toal_30

58	Card_Merch_description_actual/max_7	343	Merch_Merch_description_actual/toal_7
59	Card_Merch_description_actual/med_0	344	Merch_Merch_description_avg_0
60	Card_Merch_description_actual/med_1	345	Merch_Merch_description_avg_1
61	Card_Merch_description_actual/med_14	346	Merch_Merch_description_avg_14
62	Card_Merch_description_actual/med_3	347	Merch_Merch_description_avg_3
63	Card_Merch_description_actual/med_30	348	Merch_Merch_description_avg_30
64	Card_Merch_description_actual/med_7	349	Merch_Merch_description_avg_7
65	Card_Merch_description_actual/toal_0	350	Merch_Merch_description_count_0
66	Card_Merch_description_actual/toal_1	351	Merch_Merch_description_count_0_by_14
67	Card_Merch_description_actual/toal_14	352	Merch_Merch_description_count_0_by_30
68	Card_Merch_description_actual/toal_3	353	Merch_Merch_description_count_0_by_7
69	Card_Merch_description_actual/toal_30	354	Merch_Merch_description_count_1
70	Card_Merch_description_actual/toal_7	355	Merch_Merch_description_count_1_by_14
71	Card_Merch_description_avg_0	356	Merch_Merch_description_count_1_by_30
72	Card_Merch_description_avg_1	357	Merch_Merch_description_count_1_by_7
73	Card_Merch_description_avg_14	358	Merch_Merch_description_count_14

74	Card_Merch_description_avg_3	359	Merch_Merch_description_count_3
75	Card_Merch_description_avg_30	360	Merch_Merch_description_count_30
76	Card_Merch_description_avg_7	361	Merch_Merch_description_count_7
77	Card_Merch_description_count_0	362	Merch_Merch_description_day_since
78	Card_Merch_description_count_0_by_14	363	Merch_Merch_description_max_0
79	Card_Merch_description_count_0_by_30	364	Merch_Merch_description_max_1
80	Card_Merch_description_count_0_by_7	365	Merch_Merch_description_max_14
81	Card_Merch_description_count_1	366	Merch_Merch_description_max_3
82	Card_Merch_description_count_1_by_14	367	Merch_Merch_description_max_30
83	Card_Merch_description_count_1_by_30	368	Merch_Merch_description_max_7
84	Card_Merch_description_count_1_by_7	369	Merch_Merch_description_med_0
85	Card_Merch_description_count_14	370	Merch_Merch_description_med_1
86	Card_Merch_description_count_3	371	Merch_Merch_description_med_14
87	Card_Merch_description_count_30	372	Merch_Merch_description_med_3
88	Card_Merch_description_count_7	373	Merch_Merch_description_med_30
89	Card_Merch_description_day_since	374	Merch_Merch_description_med_7
90	Card_Merch_description_max_0	375	Merch_Merch_description_total_0
91	Card_Merch_description_max_1	376	Merch_Merch_description_total_1
92	Card_Merch_description_max_14	377	Merch_Merch_description_total_14
93	Card_Merch_description_max_3	378	Merch_Merch_description_total_3
94	Card_Merch_description_max_30	379	Merch_Merch_description_total_30

95	Card_Merch_description_max_7	380	Merch_Merch_description_total_7
96	Card_Merch_description_med_0	381	merch_state
97	Card_Merch_description_med_1	382	merch_state_actual/avg_0
98	Card_Merch_description_med_14	383	merch_state_actual/avg_1
99	Card_Merch_description_med_3	384	merch_state_actual/avg_14
100	Card_Merch_description_med_30	385	merch_state_actual/avg_3
101	Card_Merch_description_med_7	386	merch_state_actual/avg_30
102	Card_Merch_description_total_0	387	merch_state_actual/avg_7
103	Card_Merch_description_total_1	388	merch_state_actual/max_0
104	Card_Merch_description_total_14	389	merch_state_actual/max_1
105	Card_Merch_description_total_3	390	merch_state_actual/max_14
106	Card_Merch_description_total_30	391	merch_state_actual/max_3
107	Card_Merch_description_total_7	392	merch_state_actual/max_30
108	card_merch_max_0	393	merch_state_actual/max_7
109	card_merch_max_1	394	merch_state_actual/med_0
110	card_merch_max_14	395	merch_state_actual/med_1
111	card_merch_max_3	396	merch_state_actual/med_14
112	card_merch_max_30	397	merch_state_actual/med_3
113	card_merch_max_7	398	merch_state_actual/med_30
114	card_merch_med_0	399	merch_state_actual/med_7
115	card_merch_med_1	400	merch_state_actual/toal_0
116	card_merch_med_14	401	merch_state_actual/toal_1
117	card_merch_med_3	402	merch_state_actual/toal_14
118	card_merch_med_30	403	merch_state_actual/toal_3
119	card_merch_med_7	404	merch_state_actual/toal_30

120	card_merch_total_0	405	merch_state_actual/toal_7
121	card_merch_total_1	406	merch_state_avg_0
122	card_merch_total_14	407	merch_state_avg_1
123	card_merch_total_3	408	merch_state_avg_14
124	card_merch_total_30	409	merch_state_avg_3
125	card_merch_total_7	410	merch_state_avg_30
126	card_state	411	merch_state_avg_7
127	card_state_actual/avg_0	412	merch_state_count_0
128	card_state_actual/avg_1	413	merch_state_count_0_by_14
129	card_state_actual/avg_14	414	merch_state_count_0_by_30
130	card_state_actual/avg_3	415	merch_state_count_0_by_7
131	card_state_actual/avg_30	416	merch_state_count_1
132	card_state_actual/avg_7	417	merch_state_count_1_by_14
133	card_state_actual/max_0	418	merch_state_count_1_by_30
134	card_state_actual/max_1	419	merch_state_count_1_by_7
135	card_state_actual/max_14	420	merch_state_count_14
136	card_state_actual/max_3	421	merch_state_count_3
137	card_state_actual/max_30	422	merch_state_count_30
138	card_state_actual/max_7	423	merch_state_count_7
139	card_state_actual/med_0	424	merch_state_day_since
140	card_state_actual/med_1	425	merch_state_max_0
141	card_state_actual/med_14	426	merch_state_max_1
142	card_state_actual/med_3	427	merch_state_max_14
143	card_state_actual/med_30	428	merch_state_max_3
144	card_state_actual/med_7	429	merch_state_max_30

145	card_state_actual/toal_0	430	merch_state_max_7
146	card_state_actual/toal_1	431	merch_state_med_0
147	card_state_actual/toal_14	432	merch_state_med_1
148	card_state_actual/toal_3	433	merch_state_med_14
149	card_state_actual/toal_30	434	merch_state_med_3
150	card_state_actual/toal_7	435	merch_state_med_30
151	card_state_avg_0	436	merch_state_med_7
152	card_state_avg_1	437	merch_state_total_0
153	card_state_avg_14	438	merch_state_total_1
154	card_state_avg_3	439	merch_state_total_14
155	card_state_avg_30	440	merch_state_total_3
156	card_state_avg_7	441	merch_state_total_30
157	card_state_count_0	442	merch_state_total_7
158	card_state_count_0_by_14	443	merch_zip
159	card_state_count_0_by_30	444	merch_zip_actual/avg_0
160	card_state_count_0_by_7	445	merch_zip_actual/avg_1
161	card_state_count_1	446	merch_zip_actual/avg_14
162	card_state_count_1_by_14	447	merch_zip_actual/avg_3
163	card_state_count_1_by_30	448	merch_zip_actual/avg_30
164	card_state_count_1_by_7	449	merch_zip_actual/avg_7
165	card_state_count_14	450	merch_zip_actual/max_0
166	card_state_count_3	451	merch_zip_actual/max_1
167	card_state_count_30	452	merch_zip_actual/max_14
168	card_state_count_7	453	merch_zip_actual/max_3
169	card_state_day_since	454	merch_zip_actual/max_30

170	card_state_max_0	455	merch_zip_actual/max_7
171	card_state_max_1	456	merch_zip_actual/med_0
172	card_state_max_14	457	merch_zip_actual/med_1
173	card_state_max_3	458	merch_zip_actual/med_14
174	card_state_max_30	459	merch_zip_actual/med_3
175	card_state_max_7	460	merch_zip_actual/med_30
176	card_state_med_0	461	merch_zip_actual/med_7
177	card_state_med_1	462	merch_zip_actual/toal_0
178	card_state_med_14	463	merch_zip_actual/toal_1
179	card_state_med_3	464	merch_zip_actual/toal_14
180	card_state_med_30	465	merch_zip_actual/toal_3
181	card_state_med_7	466	merch_zip_actual/toal_30
182	card_state_total_0	467	merch_zip_actual/toal_7
183	card_state_total_1	468	merch_zip_avg_0
184	card_state_total_14	469	merch_zip_avg_1
185	card_state_total_3	470	merch_zip_avg_14
186	card_state_total_30	471	merch_zip_avg_3
187	card_state_total_7	472	merch_zip_avg_30
188	card_zip	473	merch_zip_avg_7
189	card_zip_actual/avg_0	474	merch_zip_count_0
190	card_zip_actual/avg_1	475	merch_zip_count_0_by_14
191	card_zip_actual/avg_14	476	merch_zip_count_0_by_30
192	card_zip_actual/avg_3	477	merch_zip_count_0_by_7
193	card_zip_actual/avg_30	478	merch_zip_count_1
194	card_zip_actual/avg_7	479	merch_zip_count_1_by_14

195	card_zip_actual/max_0	480	merch_zip_count_1_by_30
196	card_zip_actual/max_1	481	merch_zip_count_1_by_7
197	card_zip_actual/max_14	482	merch_zip_count_14
198	card_zip_actual/max_3	483	merch_zip_count_3
199	card_zip_actual/max_30	484	merch_zip_count_30
200	card_zip_actual/max_7	485	merch_zip_count_7
201	card_zip_actual/med_0	486	merch_zip_day_since
202	card_zip_actual/med_1	487	merch_zip_max_0
203	card_zip_actual/med_14	488	merch_zip_max_1
204	card_zip_actual/med_3	489	merch_zip_max_14
205	card_zip_actual/med_30	490	merch_zip_max_3
206	card_zip_actual/med_7	491	merch_zip_max_30
207	card_zip_actual/toal_0	492	merch_zip_max_7
208	card_zip_actual/toal_1	493	merch_zip_med_0
209	card_zip_actual/toal_14	494	merch_zip_med_1
210	card_zip_actual/toal_3	495	merch_zip_med_14
211	card_zip_actual/toal_30	496	merch_zip_med_3
212	card_zip_actual/toal_7	497	merch_zip_med_30
213	card_zip_avg_0	498	merch_zip_med_7
214	card_zip_avg_1	499	merch_zip_total_0
215	card_zip_avg_14	500	merch_zip_total_1
216	card_zip_avg_3	501	merch_zip_total_14
217	card_zip_avg_30	502	merch_zip_total_3
218	card_zip_avg_7	503	merch_zip_total_30
219	card_zip_count_0	504	merch_zip_total_7

220	card_zip_count_0_by_14	505	Merchnum
221	card_zip_count_0_by_30	506	Merchnum_actual/avg_0
222	card_zip_count_0_by_7	507	Merchnum_actual/avg_1
223	card_zip_count_1	508	Merchnum_actual/avg_14
224	card_zip_count_1_by_14	509	Merchnum_actual/avg_3
225	card_zip_count_1_by_30	510	Merchnum_actual/avg_30
226	card_zip_count_1_by_7	511	Merchnum_actual/avg_7
227	card_zip_count_14	512	Merchnum_actual/max_0
228	card_zip_count_3	513	Merchnum_actual/max_1
229	card_zip_count_30	514	Merchnum_actual/max_14
230	card_zip_count_7	515	Merchnum_actual/max_3
231	card_zip_day_since	516	Merchnum_actual/max_30
232	card_zip_max_0	517	Merchnum_actual/max_7
233	card_zip_max_1	518	Merchnum_actual/med_0
234	card_zip_max_14	519	Merchnum_actual/med_1
235	card_zip_max_3	520	Merchnum_actual/med_14
236	card_zip_max_30	521	Merchnum_actual/med_3
237	card_zip_max_7	522	Merchnum_actual/med_30
238	card_zip_med_0	523	Merchnum_actual/med_7
239	card_zip_med_1	524	Merchnum_actual/toal_0
240	card_zip_med_14	525	Merchnum_actual/toal_1
241	card_zip_med_3	526	Merchnum_actual/toal_14
242	card_zip_med_30	527	Merchnum_actual/toal_3
243	card_zip_med_7	528	Merchnum_actual/toal_30
244	card_zip_total_0	529	Merchnum_actual/toal_7

245	card_zip_total_1	530	Merchnum_avg_0
246	card_zip_total_14	531	Merchnum_avg_1
247	card_zip_total_3	532	Merchnum_avg_14
248	card_zip_total_30	533	Merchnum_avg_3
249	card_zip_total_7	534	Merchnum_avg_30
250	Cardnum	535	Merchnum_avg_7
251	Cardnum_actual/avg_0	536	Merchnum_count_0
252	Cardnum_actual/avg_1	537	Merchnum_count_0_by_14
253	Cardnum_actual/avg_14	538	Merchnum_count_0_by_30
254	Cardnum_actual/avg_3	539	Merchnum_count_0_by_7
255	Cardnum_actual/avg_30	540	Merchnum_count_1
256	Cardnum_actual/avg_7	541	Merchnum_count_1_by_14
257	Cardnum_actual/max_0	542	Merchnum_count_1_by_30
258	Cardnum_actual/max_1	543	Merchnum_count_1_by_7
259	Cardnum_actual/max_14	544	Merchnum_count_14
260	Cardnum_actual/max_3	545	Merchnum_count_3
261	Cardnum_actual/max_30	546	Merchnum_count_30
262	Cardnum_actual/max_7	547	Merchnum_count_7
263	Cardnum_actual/med_0	548	Merchnum_day_since
264	Cardnum_actual/med_1	549	Merchnum_max_0
265	Cardnum_actual/med_14	550	Merchnum_max_1
266	Cardnum_actual/med_3	551	Merchnum_max_14
267	Cardnum_actual/med_30	552	Merchnum_max_3
268	Cardnum_actual/med_7	553	Merchnum_max_30
269	Cardnum_actual/toal_0	554	Merchnum_max_7

270	Cardnum_actual/toal_1	555	Merchnum_med_0
271	Cardnum_actual/toal_14	556	Merchnum_med_1
272	Cardnum_actual/toal_3	557	Merchnum_med_14
273	Cardnum_actual/toal_30	558	Merchnum_med_3
274	Cardnum_actual/toal_7	559	Merchnum_med_30
275	Cardnum_avg_0	560	Merchnum_med_7
276	Cardnum_avg_1	561	Merchnum_total_0
277	Cardnum_avg_14	562	Merchnum_total_1
278	Cardnum_avg_3	563	Merchnum_total_14
279	Cardnum_avg_30	564	Merchnum_total_3
280	Cardnum_avg_7	565	Merchnum_total_30
281	Cardnum_count_0	566	Merchnum_total_7
282	Cardnum_count_0_by_14	567	RANDOM
283	Cardnum_count_0_by_30	568	state_risk
284	Cardnum_count_0_by_7	569	Transtype
285	Cardnum_count_1	570	weekdays
		571	weekdays_risk