Credit Card Transaction Fraud Identification

Team member: Rui Liu, Shangfu Chen, Xuanzhong Chen,

Yingluo Li Yuquan Zheng, Zezheng Hao

Advisor: Professor Stephen Coggeshall

6/10/21

MGTA 463

Table of Contents

Executive Summary

Credit card fraud is one of the most common frauds worldwide, it was usually committed using a payment card, such as a credit card or debit card, to purchase online goods or services or directly using other one's information to apply for the credit card as application fraud. Since the compromise can occur in a number of ways and can usually occur without the knowledge of the cardholder due to the small amount for each transaction, it has been a major problem for all banks and credit bureaus.

This project will be focusing on identifying suspicious and unusual card transactions in the data by building a data-driven, supervised fraud model that is able to detect and report the abnormal transactions to the bank or credit bureaus. The overall goal of the project is to use proper methodologies to build a supervised fraud model to detect the potential credit fraud.

The report details the creation and completion process of building fraud scores and model which detects potential fraud transactions that is abnormal and unusual. Our team completed the following objectives to accomplish the project goal:

- Data cleaning – remove exclusions and fill in missing values for the fields that have missing values.
- Variable creation – build many candidates including two target encoded variables: likelihood of fraud for specific day of the week and for the specific state, and z-scale them.
- Feature selection – filter out the top 80 variables by their calculated KS and FDR, then do a wrapper to get down to the top 30 variables by multivariate importance. Add in two test variables: fraud label and a random number. Remove the out-of-time records for last 4 months and the first 2 weeks of records.
- Modeling – Use the top 30 variables to build the final fraud model with logistic regression, Boost Tree, Random Forest and Neural Network. Test their performances with different choices of hyperparameters and select the model that provides the best performance.

1 Data Description

1.1 File Description

The data is Card Transaction Data, which includes actual credit card purchases' information provided by U.S. government organization during the year 2010. The purpose of this data is to indicate whether the transaction is a fraud or not. There are 10 number of fields in the data: Amount, Recnum, Cardnum, Merchnum, Merch Description, Merch State, Merch Zip, Transtype, Fraud and Data. There are 96,753 number of records in this data, and 1,059 records are indicated as a fraud.

Dataset Name	Card Transaction Data
Dataset Purpose	The data is about credit card transaction information, which indicates whether this transaction is a fraud or not.
Data Source	Came from U.S. government organization
Time Period	From Jan. 1st, 2010 to Dec. 31st, 2010
Number of Fields	10 Fields in total – 1 numeric, 8 categorical, 1date
Number of Records	96,753

Table 1.1: File Description

1.2 Summary Statistics Table

All fields in the data can be treated as categorical, numeric and date: 1 field is numeric, 8 fields are categorical and 1 field is date. Among all fields, 7 fields are fully populated and other 3 fields are not fully populated. Key statistics of these fields are summarized as follows.

1. Numeric Fields:

Variable	Field Type	Count	Mean	Std	Min	Max	Unique Value	# Zero	% Populated
Amount	Numeric	96753	427.89	10,006.14	0.01	3,102,045.53	34909	0	100

Table 1.2.1 Summary Statistics of Numeric Fields

2. Categorical Fields:

Variable	Field Type	Count	% Populated	Unique values	Most Common Field Values
Recnum	Categorical	96753	100	96,753	N/A
Cardnum	Categorical	96753	100	1645	5142148452
Merchnum	Categorical	93378	96.5	13,091	930,090,121,224
Merch Description	Categorical	96753	100	13,126	GSA-FSS-ADV
Merch State	Categorical	95558	98.8	227	TN
Merch Zip	Categorical	92097	95.2	4,567	38118
Transtype	Categorical	96753	100	4	P
Fraud	Categorical	96753	100	2	0

Table 1.2.2 Summary Statistics of Categorical Fields

3. Date Fields:

Variable	Field Type	count	Unique Values	Most Common Field Values	Min	Max	Records that have a value	% populated
Date	Date	96753	365	2010/2/28	2010/1/1	2010/12/31	96753	100

Table 1.2.3 Summary Statistics of Categorical Fields

1.3 Field Examples

1.3.1 Field "Amount"

Description	The amount of money of each transaction, exclude outliers > 1000, data in histogram is 89.10% populated
Type	Numeric
Mean	427.89
Maximum	3,102,045.53
Minimum	0.01
Std	10,006.14

Table 1.3.1: Amount

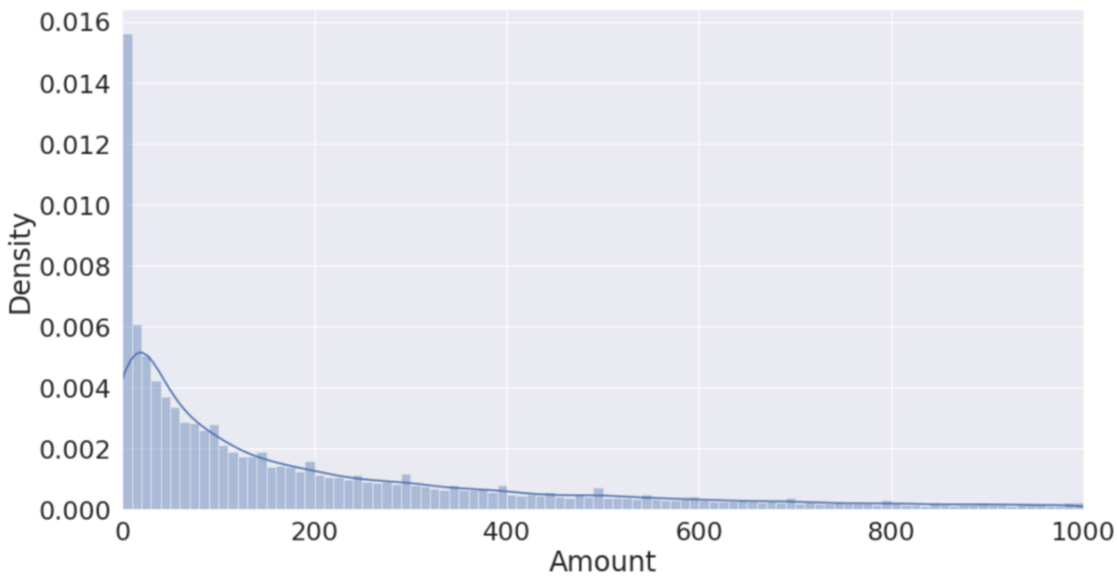


Figure 1.3.1: Frequency Distribution of Amount Field

1.3.2 Field "Transtype"

Description	Transaction type of each transaction
Type	Categorical
Most Common Field Value	'P' occurred the most for 96,396 times

Table 1.3.2: Transtype

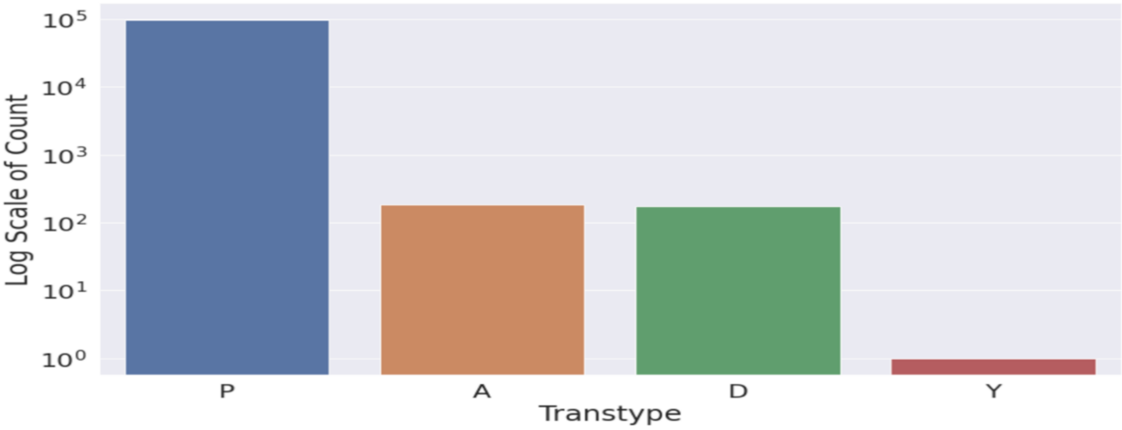


Figure 1.3.2: Frequency Distribution of Transtype Field

1.3.3 Field "Date"

Description	The date of the transaction. Month, day and year only (no time of day). Data in histogram is 100% populated.
Type	Date
Unique Values	365
Maximum	2010/12/31
Minimum	2010/1/1
Most Common Field Value	'2010/2/28' occurred the most for 684 times

Table 1.3.3: Date

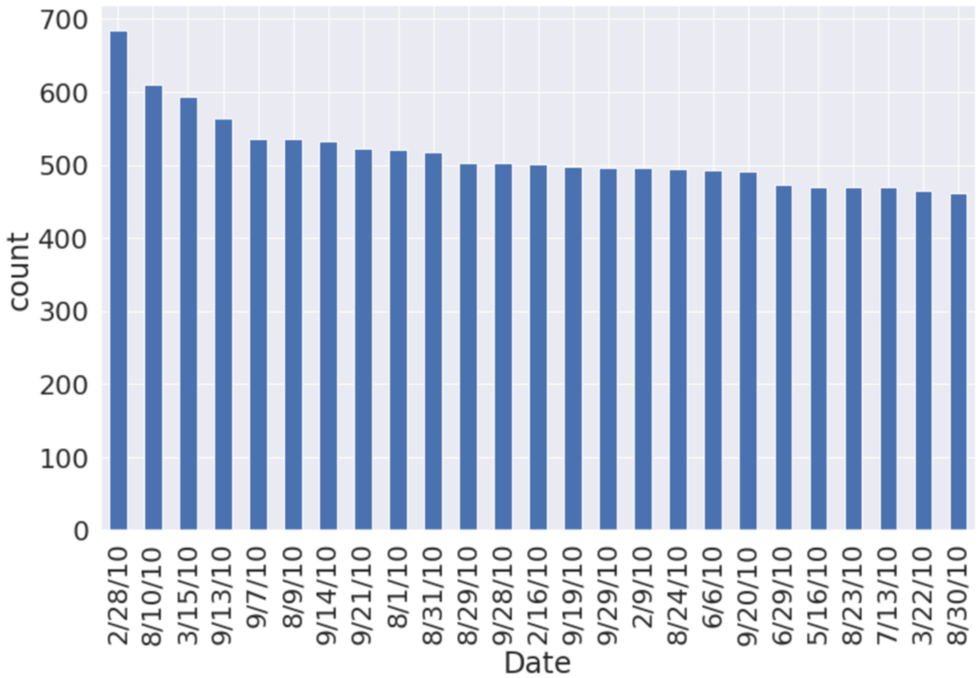


Figure 1.3.3: Frequency Distribution of Date Field

2 Data Cleaning

First of all, we checked the only numeric field “Amount” in the data, and we draw a boxplot of the field which is shown below (figure 2.1.1). We found a red outlier in the boxplot whose value is larger than 3 million. Even we checked that the outlier is not recorded as a fraud, we still decide to exclude it for further analysis because it might influence the accuracy of our final result.

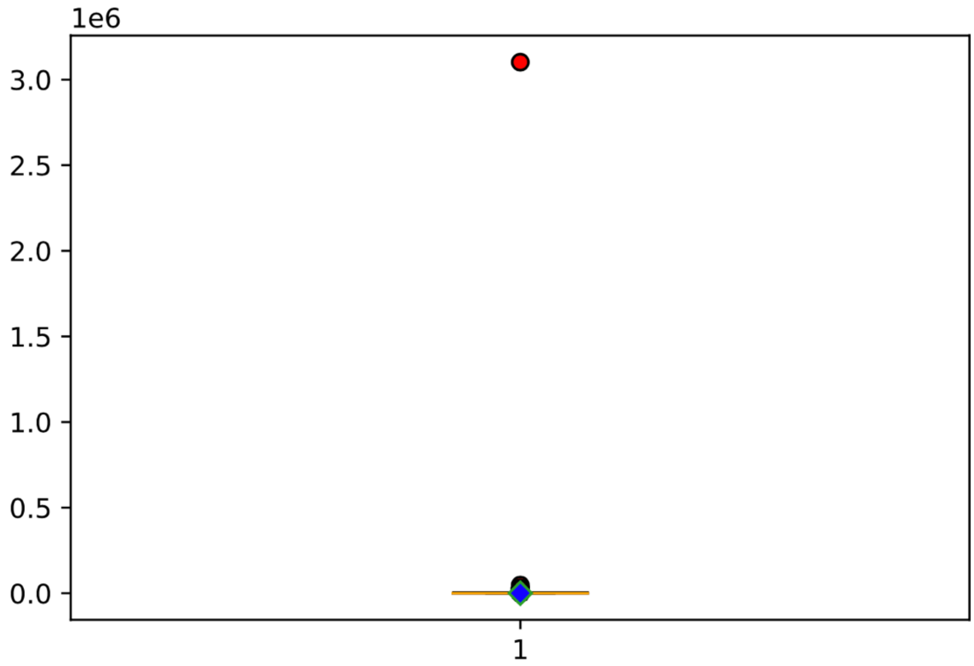


Figure 2.1: Boxplot of Field “Amount”

After excluding the outlier, we selected the records with “P” in “Transtype” Field because we would focus on these records. Besides, we check the missing values in all field and the result is given below. There were 3,198 missing values in the “Merchnum” field, 1,020 missing values in the “Merch state” field and 4,300 missin gvalues in the “Merch zip” field. For the missing values in “Merchnum” field, we replaced with “NaN”, if the missing value is 0, and we filled in with mode of “Merch Description” Field, and we also filled in with “Unk” for unknown. For the missing values in “Merch zip” field, we filled in with the mode of “Merchnum” Field. For the missing values in “Merch state”, we matched the state for the zip in “Merch zip” field if the record had a “Merch zip”. Moreover, if the “Merch zip” was in the range of “00600 - 00799” and “00900 - 00999”, we filled in “Merch state” with “PR” for Puerto Rico. Besides, we used the mode of “Merchnum” or “Merch Description” for other records with missing values, and we finally filled in with “Unk” for unknown.

Name of Field	Number of Missing Value
Recnum	0
Cardnum	0
Date	0
Merchnum	3198
Merch Description	0
Merch state	1020
Merch zip	4300
Transtype	0
Amount	0
Fraud	0

Table 2.1: Missing values of each field

3 Variables Creation

Before building model, it is critical to create candidate variables. We have examined the clean data set and decide to create a large set of variables for supervised models based on the existing PII fields, such as Cardnum, Merchnum, Merch zip and Merch state and derived PII fields showed in table 3.2. At this step, we will use a variety of method to create many candidate variables as possible and then we will use several feature selection methods to select final variables for models. We will create candidate variables in four aspects, amount, frequency, day-since and velocity.

PII fields	Description
Cardnum	Card number of each transaction
Merchnum	Merchant number of each transaction
Merch zip	The zip code of merchant
Merch state	State where merchant is in

Table 3.1

PII combination fields	Algorithm
Card_merch	Cardnum + Merchnum
Card_zip	Cardnum + Merch zip
Card_state	Cardnum + Merch state
Merch_zip	Merchnum + Merch zip
Merch_state	Merchnum + Merch state
Card_merch_zip	Cardnum + Merchnum + Merch zip
Card_merch_state	Cardnum + Merchnum + Merch state

Table 3.2

3.1 Amount Variables

First, we create amount candidate variables, which measure amount spent of each transaction over the past period of the same PII fields and PII combination fields. Amount variables are created by average, maximum, median, total, actual/average, actual/max, actual/total, actual/median amount by/at this card, merchant, card at this merchant, card in this zip code over the past 0 days, 1 day, 3 days, 7 days, 14 days, 30 days. Amount variables are important for detecting fraud because anomaly will be detected if there is an unusual amount spent. For example, if one person usually spends around 100 dollars and one day there is a transaction with amount of 1,000 dollars. It would indicate that this transaction may be fraudulent and transactions will be intervened.

$$\begin{pmatrix} Average \\ Maximum \\ Median \\ Total \\ \frac{Actual}{average} \\ Actual/max \\ \frac{Actual}{total} \\ \frac{Actual}{median} \end{pmatrix} \text{amount by/at this} \begin{pmatrix} Card \\ Merchant \\ Card\ at\ this\ merchant \\ Card\ in\ this\ zip\ code \\ Card\ in\ this\ state \\ card\ at\ this\ merchant\ in\ this\ zip\ code \\ card\ at\ this\ merchant\ in\ this\ merchant \end{pmatrix} \text{over the past} \begin{pmatrix} 0\ days \\ 1\ day \\ 3\ days \\ 7\ days \\ 14\ days \\ 30\ days \end{pmatrix}$$

Figure 3.1

3.2 Frequency Variables

Beside amount variables, we also create frequency candidate variables, which shows the number of transactions over the past period of the same PII fields or PII combination fields. Frequency variables are generated by number of transactions with this card, merchant, card at this merchant, card in this zip code, card in this state, card at this merchant in this zip code, card at this merchant in this merchant over the past 0 days, 1 days, 3 days, 7 days, 14 days, 30 days. Frequency variables are crucial for finding anomaly in transactions. Fraud will be detected if there is a huge difference in number of transactions during a period. For instance, if one person typically makes 2 transactions daily, but one day he/she makes more than 100 transactions, which would be a signal of fraud. These frequency variables are good predictors to find potential frauds among transactions.

$$\text{Number of transactions with this} \begin{pmatrix} Card \\ Merchant \\ Card\ at\ this\ merchant \\ Card\ in\ this\ zip\ code \\ Card\ in\ this\ state \\ Card\ at\ this\ merchant\ in\ this\ zip\ code \\ Card\ at\ this\ merchant\ in\ this\ merchant \end{pmatrix} \text{over the past} \begin{pmatrix} 0\ days \\ 1\ day \\ 3\ days \\ 7\ days \\ 14\ days \\ 30\ days \end{pmatrix}$$

Figure 3.2

3.3 Days-Since Variables

After creating amount variables and frequency variables, we also create day-since variables, which shows the number of days since the last transactions of the same PII fields or PII combination fields. Day-since variables are generated by current date minus date of most recent transaction with same card, merchant, card at this merchant, card in this zip code, card in this state. For records that it is the first time seen, we use 365 for its value instead of 0. Day-since variables can identify frauds in transactions if there is a huge difference in interval between current transaction date and last-time transaction. One good example to illustrate is that if records show that one person makes last transaction 10 months ago, transactions occur during that 10 month or ongoing transaction might be fraud. Transactions will be prevented if there is a signal of suspicious activities.

$$\text{Current date minus date of most recent transaction with same} \begin{pmatrix} \textit{Card} \\ \textit{Merchant} \\ \textit{Card at this merchant} \\ \textit{Card in this zip code} \\ \textit{Card in this state} \\ \textit{Card at this merchant in this zip code} \\ \textit{Card at this merchant in this merchant} \end{pmatrix}$$

Figure 3.3

3.4 Velocity Variables

After using method of creating variables above, we create velocity variables, which is similar to the frequency variables since it also measures how a person do a large number of transactions over a day compared to his average transactions over a period of time. Velocity variables are created by number or amount of transactions with same card or merchant over the past 0 days, 1 days that is divided by daily average number/amount of transactions with same card/merchant over the past 7 days, 14 days, 30 days. Velocity variables are important since ratio of transactions within one day to transactions over a period of time can be indicative of fraudulent activity. The higher velocity variable value, the high probability of being a fraud. For example, if a person makes 5 transactions on average over past 30 days and he/she makes 20 transactions on one day, it would cause a high value of ratio and such high value of ratio will indicate recent activities are unusual.

$$\frac{\binom{Number}{Amount} \ of \ transactions \ with \ same \ \binom{Card}{Merchant} \ over \ the \ past \ \binom{0 \ days}{1 \ day}}{Average \ daily \ \binom{Number}{Amount} \ of \ transactions \ with \ same \ \binom{Card}{Merchant} \ over \ the \ past \ \begin{pmatrix} 7 \ days \\ 14 \ days \\ 30 \ days \end{pmatrix}}$$

Figure 3.4

3.5 Target Encoded Variables

At last, two target encoded variables, likelihood of fraud by week of day and by state are created. These two variables measure how the likelihood of someone who commits fraud varies from a different day of a week or a different state. Before creating two target encoded variables, the most recent data time, September 2010, is reserved as an OOT, out of time and is not included when creating two target encoded variables. For probability of fraud by weekday also known as weekday risk, it is first calculated on the dataset without recent one-month transactions and then is applied to the whole dataset. The average probabilities of fraud by weekday are listed in Table 3.3 below. Probability of fraud by state is also first calculated on dataset which does not include recent one-month records and then is applied to the whole dataset. The result, average probabilities of fraud by state is listed in table 3.4. By using method of creating candidate variables, we have 521 variables in total. These 521 candidate variables and statistics of them are listed in Appendix B.

Weekday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Average fraud probability	0.00855	0.006652	0.009811	0.01604	0.040022	0.011144	0.008593

Table 3.3 Average fraud probability by weekday

State	Average fraud probability
AK	0
AL	0.004201681
AR	0
AZ	0
BC	0
CA	0.016839917
CO	0.000670691
CT	0.004273504
DC	0.02125775
DE	0
FL	0.000651042
GA	0.000822594
HI	0
IA	0
ID	0
IL	0.002398604
IN	0
KS	0.001153403
KY	0.00295858
LA	0
MA	0.002054795
MD	0.021974802
ME	0
MI	0.008385744
MN	0.001538462
MO	0.00058651
MS	0
MT	0
NC	0.000844238
ND	0
NE	0
NH	0.004983389
NJ	0.001850481
NM	0.011363636
NV	0
NY	0.027305825

OH	0.018390805
OK	0
ON	0
OR	0.028243601
PA	0.019770774
PQ	0
PR	0
RI	0
SC	0.009090909
SD	0.009708738
TN	0.008748115
TX	0.013816281
UT	0.057391304
Unk	0.001451379
VA	0.013940703
VT	0
WA	0.013673655
WI	0.001424501
WV	0
WY	0

Table 3.4 Average fraud probability by state

4 Feature Selection

After building all the 521 candidate variables, we would avoid using all of them to build models, for which will result in high dimensionality. Nonintuitive things happen in high dimensions as the data becomes sparse very quickly. All the points will become outliers in an extremely high dimension as figure 4.1 shows. Besides, we will need exponentially more data to see true nonlinearities rather than noise. In order to avoid the curse of high dimensionality, we choose to do a feature selection, which can not only reduce the dimensionality but also help discover the variables with most importance and information.

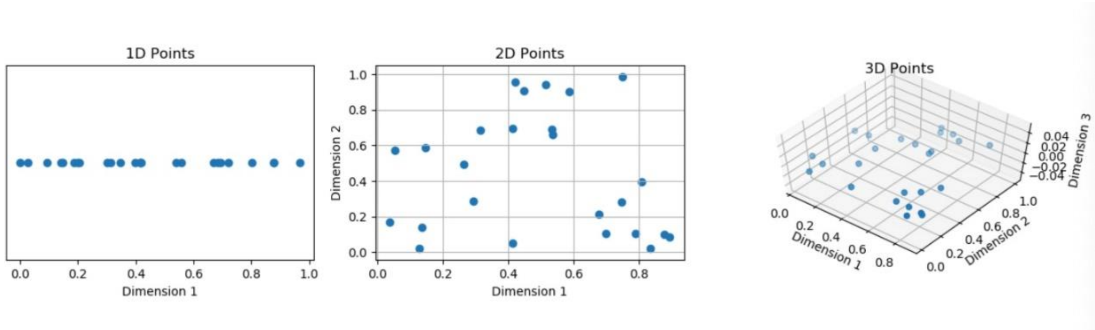


Figure 4.1 Points go sparse and become outliers with a high dimensionality

For the selection methods, there are 3 ways to categorize a feature selection process. The first one is a filter, which is independent of any modeling method. The second one is a wrapper, which usually uses a model 'wrapped' around the feature selection. It's usually a stepwise selection that can either go forward or backward. The third one is an embedded method, which directly uses the model built in the modeling stage to do a feature selection, such as a decision tree.

Before the selection process, we would add two special test variables - a random number and a fraud label. The random number should behave worse than most of the variables while the fraud label itself should behave perfectly in separating frauds and non-frauds. What's more, in this feature selection stage, we would remove records in the last 4 months since they are out-of-time records which will be used to evaluate the models. Also, we would remove records in the first two weeks since some of our variables are related with past records that are more than two weeks ago.

In our project, we utilized the former two methods – filter and wrapper to reduce the number of variables as the figure 4.2 shows. Using a filter, we get 80 variables from 521 candidate variables. Using a wrapper, we get the final 30 variables from 80 variables.

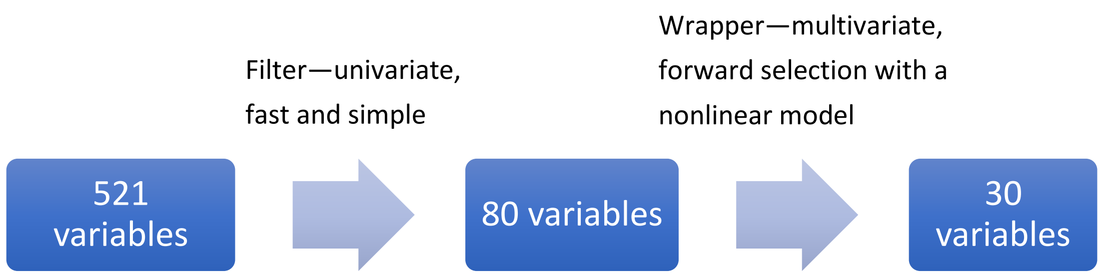


Figure 4.2 Process of a Feature Selection

4.1 Filter

For the filter, we will use two measures: the univariate Kolmogorov-Smirnov (KS) and the univariate fraud detection rate at 3% population. The KS generally measures the distance between two distributions – in this project they are the good records and the bad ones. So, KS is described as following formula:

$$\mathrm{KS} = \max_{x} \int_{xmin}^{x} [P_{goods} - P_{bads}] dx$$

We would build the frauds and non-frauds distribution of each candidate variable and use the KS measure to estimate the effectiveness of that variable in differentiating fraud records and non-fraud records.

Another measure we will use in the filter is the FDR (fraud detection rate) at 3% population. To be specific, FDR at 3% means what % of all the frauds are caught at 3% of the population. For each candidate variable, we will sort the population by the value of the candidate variable. Then we will select the top 3% population and tail 3% population. The ratio of frauds in that population to the total number of frauds in the entire population is then calculated. We will pick the larger ratio (comparing that from the top 3% population and the bottom 3% population) to represent the FDR of the variable. This FDR will also be a good measure to represent the ability of differentiating frauds and non-frauds.

Using these two measures in the filter, we're able to get a table of all the 523 variables with their corresponding KS and FDR. We rank the variables using KS and FDR separately and finally use the average rank to measure the goodness of a variable. The variables with higher average rank are better. The fraud label has the highest rank while the random number stays at the bottom. We sort the variables by average rank and pick the top 80 variables (excluding the fraud label) as the result of filtering process.

4.2 Wrapper

Then we use a wrapper to further reduce the dimensionality from 80 variables to 30 variables. For a typical wrapper, it has a model wrapped around the process. There will be many models built with different variables in the process. The wrapper will change the number of variables based on the result of previous models. The most common wrapper methods are forward selection, backward selection and general stepwise selection. Suppose we have n variables. Forward selection starts with n separate 1-dimension models. Then it will keep the best variable at each step and test which variable is the next best one to add by building all possible models at each step. It keeps adding the best variable until no significant improvement occurs. Backward selection goes with the opposite direction. It starts with a single model using all variables. Then it removes the variable that causes the least model performance decrease at each step and keep doing it until reaching the final model. The general stepwise selection shares the same ideas with the previous two selections. The difference is that it can either add or remove a variable at each step.

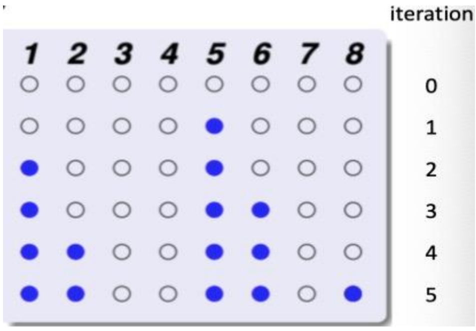


Figure 4.3 Forward selection by adding a variable each time

We choose a simple nonlinear model – a random forest model with forward selection for the wrapper. The modeling method inside a wrapper can be any model. In this project, as we are predicting binary outcomes, nonlinear models are used. Since the selection process tries many combinations of variables, a faster modeling method is preferred like the logistic regression or random forest. Random forest is a fast tree-like model algorithm that will be explained in the model algorithm part. And the forward selection will help us find good subset of important variables, remove correlations and reduce dimensionalities. Inside the forward selection, we use the FDR as the scoring metric. This FDR metric helps us determine which sub-models has stronger performances and then we can determine which variable to add each time.

By fitting the 80 variables within the wrapper model, we get the final 30 variables. They are ranked by importance so that we know which variable is more valuable. The final 30 variables ranked by importance are shown in Table 4.1.

Rank	Variables	Rank	Variables
1	card_merch_state_total_14	**16**	card_state_total_30
2	card_merch_zip_total_7	**17**	card_zip_max_7
3	card_zip_total_3	**18**	card_state_max_30
4	card_merch_total_14	**19**	card_merch_zip_max_7
5	card_state_total_3	**20**	card_merch_zip_max_3
6	card_merch_zip_total_14	**21**	card_state_max_1
7	card_merch_state_total_3	**22**	card_zip_max_1
8	card_state_total_7	**23**	merch_zip_total_1
9	card_zip_total_1	**24**	Merchnum_max_0
10	card_merch_state_total_30	**25**	Merchnum_total_0
11	card_merch_total_1	**26**	Cardnum_total_1
12	card_merch_zip_total_30	**27**	Merchnum_total_3
13	card_merch_zip_total_1	**28**	merch_zip_max_3
14	card_state_max_7	**29**	card_zip_max_0
15	card_zip_max_30	**30**	Cardnum_total_0

Table 4.1 Performance of Models

5 Model Algorithms

Before stepping into any specific model, we must think about the results from a model. To ensure a robust model result, typically, the Cross-Validation will be implemented. A standard K-fold Cross-Validation process is to

- Randomly divide data into K groups.
- Consider each group as test data which would be tested based on a training result by the remaining data.
- keeping doing this K times, so that when all groups are tested meaning that each record in the data set was considered as test data in one of the K times.
- Extract and check K time's results.

This skill is commonly useful when we only have a limited number of records. A key disadvantage of this skill is that when K goes up, each group shrinks. If we want to use a mean result of 20 times fitting. Each time, only 5% of the data will be tested.

Therefore, to make the best use of this method to stable results, in this project, instead of shuffling data into K groups, and fitting K times on the one-time shuffle, we divide data before each training. By reshuffling data each time before fitting, we can control the data size used and avoid testing on a too-small test data set.

As for the metrics of results, we use FDR in top 3%, which is to rank model predicted results from top to bottom and considering all top 3% records as "Fraud". Then calculated the ratio of how many real frauds among these top 3%.

In the following modeling, we "Cross Validate" model results 10 times in each specific model, and each time we re-split the whole data set but keeping train and test data set as a 7:3 ratio. Then, we calculate FDR of train, test and OOT data to evaluate the model result. We adapt Logistic regression as a baseline and implement 4 nonlinear models, Neural Network, Gradient Boosting, Random Forest, and SVM, to get an optimized result after parameters tunning. Details of model performances are listed in the following table.

Model	Parameter					Average FDR(%) at 3%		
Logistic Regression	Total Variables			C	class_weight	TRAIN	TEST	OOT
1	30			1000	balanced	0.692909	0.679418	0.508708
2	30			100	balanced	0.693932	0.676305	0.510112
3	30			10	balanced	0.694614	0.67605	0.513764
4	30			1	balanced	0.690389	0.679658	0.510112
5	30			1000	None	0.670789	0.671726	0.492135
6	30			100	None	0.672442	0.662191	0.490169
7	30			10	None	0.676366	0.656386	0.49382
8	30			1	None	0.672678	0.666992	0.496348
Neural Net	# of Variables Selected	Layer	Node	max_iter	learning rate	TRAIN	TEST	OOT
1	30	1	10	1000	0.0001	0.821107	0.790421	0.568539
2	30	1	30	1000	0.0001	0.857716	0.817777	0.55927
3	30	1	40	1000	0.0001	0.87318	0.812662	0.555337
4	30	1	50	1000	0.0001	0.871377	0.811954	0.562921
5	30	2	(10,10)	1000	0.0001	0.849261	0.79374	0.560393
6	30	2	(30,10)	1000	0.0001	0.890956	0.821943	0.527247
7	30	2	(30,10)	500	0.0001	0.878125	0.820101	0.546348
8	30	2	(30,10)	500	0.001	0.8629	0.824626	0.566011
9	30	2	(30,10)	500	0.01	0.726586	0.722006	0.597753
10	30	2	(30,10)	500	0.1	0.65959	0.648998	0.513202
Gradient Boosting	# of Variables Selected	# of Trees	Max Depth	Learning Rate		TRAIN	TEST	OOT
1	30	100	2	0.1		0.818234	0.772457	0.576124
2	30	100	2	0.01		0.65237	0.638253	0.527247
3	30	200	2	0.1		0.863685	0.801835	0.582865
4	30	500	2	0.1		0.942135	0.867921	0.591854
5	30	1000	1	0.01		0.699643	0.672664	0.545225
Random Forest	# of Variables Selected		# of Trees	Max Depth	Max Features	TRAIN	TEST	OOT

Model	Parameter				Average FDR(%) at 3%		
1	30	100	10	20	0.888674	0.832183	0.633708
2	30	60	15	30	0.989147	0.872133	0.62191
3	30	100	20	20	1	0.874301	0.614326
4	30	100	10	20	0.887774	0.818445	0.623315
5	30	100	10	30	0.879317	0.82147	0.628371
6	30	60	20	30	1	0.869344	0.609831
SVM	# of Variables Selected	C		Kernel	TRAIN	TEST	OOT
1	30	1		poly	0.718059	0.697391	0.49073
2	30	1		sigmoid	0.133322	0.138413	0.124438
3	30	1		rbf	0.764503	0.742637	0.586798
4	30	0.1		poly	0.691555	0.682982	0.474157
5	30	0.1		rbf	0.682563	0.680915	0.625562
6	30	0.01		poly	0.686868	0.67901	0.471067
7	30	0.01		rbf	0.691001	0.666645	0.610674

Table 5.0 Performance of Models

5.1 Logistic Regression

Logistics Regression is the most frequently used classification method in industries. It can always serve as a good baseline before going to the nonlinear method. Unlike Linear Regression, whose result will be located beyond 0 and 1. The binary logistic regression can offer prediction within 0 and 1, therefore, based on the distance the result located, we can easily recognize how possible the record belongs to each class (1 or 0). Here is a graph to show the difference between linear regression and logistic regression.

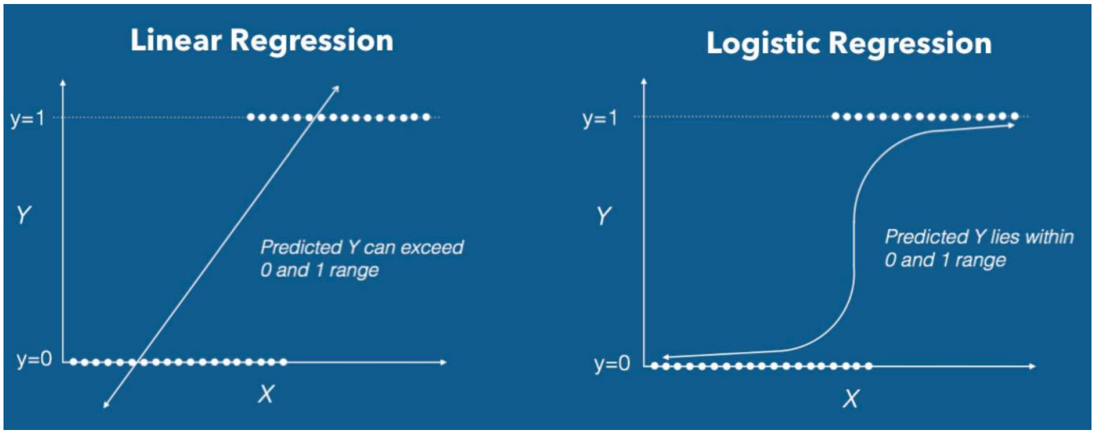


Figure 5.1 Comparison of Linear Regression and Logistic Regression

Because of the simplicity and effectiveness in classification, we firstly build a logistic regression as a reference of our baseline. To get a better result from this model, we also tune the parameters. Here is the list of parameters that we choose to tune.

Parameters	Description
c	float, default=1.0 Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.
class_weight	Weights associated with classes in the form {class_label: weight}. If not given, all classes are supposed to have weight one. The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y)).

Table 5.1.1 Result of Logistic Regression

After a parameter tuning (see Table 6.0), we get the baseline as followed.

Average FDR(%）at 3%		
TRAIN	TEST	OOT
0.694614	0.67605	0.513764

Table 5.1.2 Result of Logistic Regression

5.2 Neural Network

Neural Network is an algorithm that was first raised to simulate the mechanism of information transformation in the neurons of the human brain. The algorithm tries to explore the patterns between the input layer and output layer with hidden layers and nodes. Each hidden layer will have multiple nodes, presenting certain transformations. The data were transferred to each node in a specific layer and make certain calculations, then be passed to the next layer. So that through these layers and nodes, the input layer data can be transformed as close as possible to output layer data. The following graph shows how input is transfer to output.

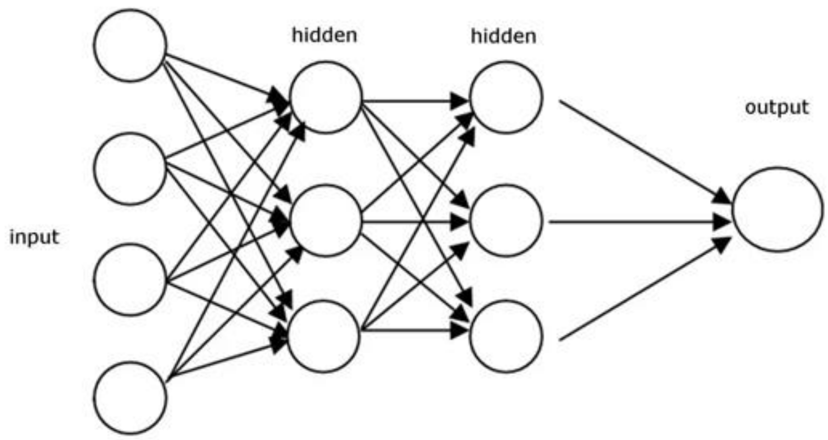


Figure 5.2 Illustration of Neural Network

The certain parameters we choose to tune as list below.

Parameters	Description
Node, Layer	The ith element represents the number of neurons in the ith hidden layer.
max_iter	int, default=200 Maximum number of iterations. The solver iterates until convergence (determined by 'tol') or this number of iterations. For stochastic solvers ('sgd', 'adam'), note that this determines the number of epochs (how many times each data point will be used), not the number of gradient steps.
learning rate	double, default=0.001 The initial learning rate used. It controls the step-size in updating the weights. Only used when solver='sgd' or 'adam'.

Table 5.2.1 Result of Neural Network

After a parameter tuning (see Table 5.0), we get the best result of Neural Network model.

Average FDR(%）at 3%		
TRAIN	TEST	OOT
0.8629	0.824626	0.566011

Table 5.2.1 Result of Neural Network

5.3 Gradient Boosting Decision Tree

Boosting is a method to improve the performance of weak learners by ensemble multiple learners. The Gradient Boosting Decision Tree model starts from a single Decision Tree. After each iteration, the model learns from the residual of iteration by adding another "tree" to capture the residual. In Gradient Boosting, the model use gradient decreasing as the direction to reduce residual. Keeping this process, the residual decreased, and overall performance improved. The result is the prediction ensembled through all these "trees" by sequence.

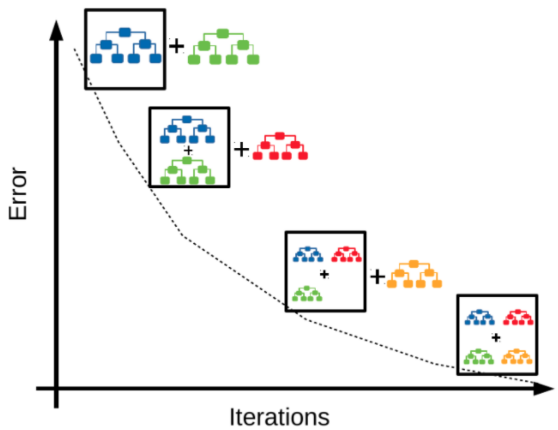


Figure 5.3 Illustration of Neural Network

The certain parameters we choose to tune as list below.

Parameters	Description
n_estimators	int, default=100 The number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance.
max_depth	int, default=3 The maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree. Tune this parameter for best performance; the best value depends on the interaction of the input variables..
learning_rate	float, default=0.1 Learning rate shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators.

Table 5.3.1 Parameters of Gradient Boosting

After a parameter tuning (see Table 5.0), we get the best result of Gradient Boosting Decision Tree.

Average FDR(%) at 3%		
TRAIN	TEST	OOT
0.863685	0.801835	0.582865

Table 5.3.2 Result of Gradient Boosting Decision Tree

5.4 Random Forest

Random Forest is another model that begins with a Decision Tree. However, instead of adding another tree to capture the residual of each iteration, the Random Forest generates a "forest of trees" to fit the data. Each tree will be allocated a random subset of records and the "tree" grows based on these data. Finally, the model lets every tree "vote" for a certain record based on their classifier and then ensemble the result. In Random Forest, the result does not need to proceed following along the exact sequence of "trees" as Gradient Boosting. Each "tree" grows independently. The following graph shows the theory of Random Forest.

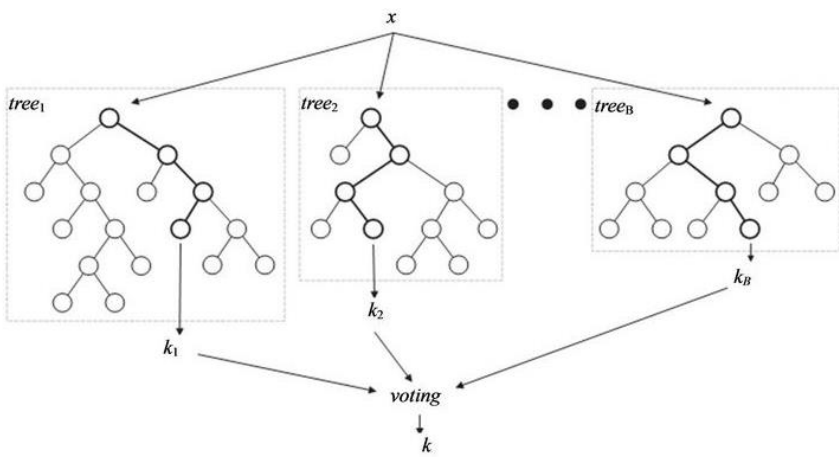


Figure 5.4 Illustration of Random Forest

The certain parameters we choose to tune as list below.

Parameters	Description
n_estimators	int, default=100 The number of trees in the forest.
max_depth	int, default=None 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.
max_features	{"auto", "sqrt", "log2"}, int or float, default="auto" The number of features to consider when looking for the best split:

Table 5.4.1 Parameters of Random Forest

After a parameter tuning (see Table 5.0), we get the best result as followed.

Average FDR(%) at 3%		
TRAIN	TEST	OOT
0.888674	0.832183	0.633708

Table 5.4.2 Result of Random Forest

5.5 Support Vector Machine

Support Vector Machine is an algorithm to find the best “separator” between two classes in supervised machine learning. The term “support vector” refers to the records located closest to the “separator”. The sum of distances of support vectors of two classes to the “separator” is called Margin. Therefore, the best “separator” is the one that can maximize the Margin so that it can best tolerance the noise of the data. Here is a graph showing the algorithm.

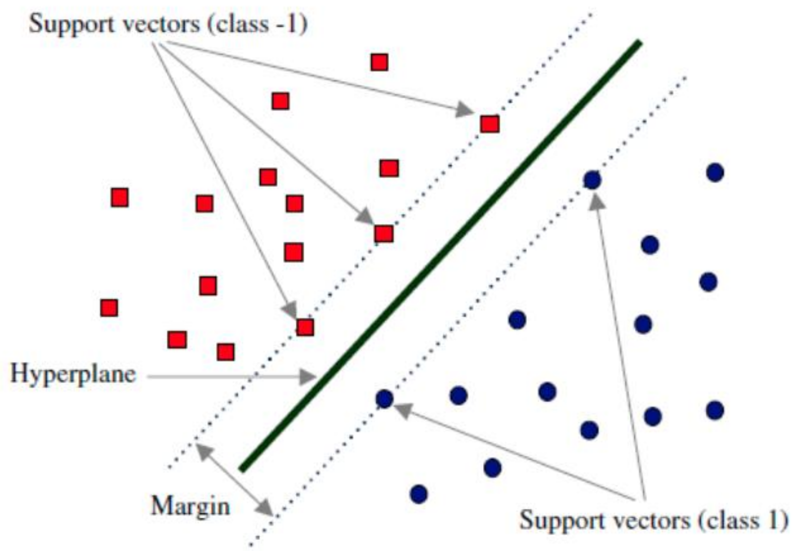


Figure 5.5 Illustration of Support Vector Machine

The certain parameters we choose to tune as list below.

Parameters	Description
c	float, default=1.0 Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.
Kernel	{‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’}, default=’rbf’ Specifies the kernel type to be used in the algorithm. It must be one of ‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’ or a callable. If none is given, ‘rbf’ will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n_samples, n_samples).

Table 5.5.1 Parameters of Support Vector Machine

After a parameter tuning (see Table 5.0), we get the best result as followed.

Average FDR(%） at 3%		
TRAIN	TEST	OOT
0.682563	0.680915	0.625562

Table 5.4.2 Result of Support Vector Machine

6 Results

After comparing effects of different models, we choose Random Forest with parameters {n_estimators = 100, max_depth = 10, max_features = 20}. We chose top 20 bins (20% population) for table following. Each of tables below show important model results and from those we can detect that whether the model is working properly, not overfitting and performing well on the OOT population and also, examine trade offs with various choices of cutoff point.

Training	# Records	# Bads	# Goods	Fraud Rate								
	43367	424	42943	0.0098								
	Bin Statistics						Cumulative Statistics					
Population Bin%	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cum Goods	Cum Bads	FPR	% Good	% Bad(FDR)	KS
0	434	92	342	21.2	78.8	434	92	342	0.3	0.2	80.7	80.5
1	434	416	18	95.9	4.2	868	508	360	1.4	1.2	84.9	83.7
2	433	430	3	99.3	0.7	1301	938	363	2.6	2.2	85.6	83.4
3	434	420	14	96.8	3.2	1735	1358	377	3.6	3.2	88.9	85.8
4	434	434	0	100.0	0.0	2169	1792	377	4.8	4.2	88.9	84.8
5	433	431	2	99.5	0.5	2602	2223	379	5.9	5.2	89.4	84.2
6	434	426	8	98.2	1.8	3036	2649	387	6.8	6.2	91.3	85.1
7	433	430	3	99.3	0.7	3469	3079	390	7.9	7.2	92.0	84.8
8	434	433	1	99.8	0.2	3903	3512	391	9.0	8.2	92.2	84.0
9	434	433	1	99.8	0.2	4337	3945	392	10.1	9.2	92.5	83.3
10	433	433	0	100.0	0.0	4770	4378	392	11.2	10.2	92.5	82.3
11	434	432	2	99.5	0.5	5204	4810	394	12.2	11.2	92.9	81.7
12	433	433	0	100.0	0.0	5637	5243	394	13.3	12.2	92.9	80.7
13	434	431	3	99.3	0.7	6071	5674	397	14.3	13.2	93.6	80.4
14	434	434	0	100.0	0.0	6505	6108	397	15.4	14.2	93.6	79.4
15	433	432	1	99.8	0.2	6938	6540	398	16.4	15.2	93.9	78.6
16	434	434	0	100.0	0.0	7372	6974	398	17.5	16.2	93.9	77.6
17	433	432	1	99.8	0.2	7805	7406	399	18.6	17.3	94.1	76.9
18	434	434	0	100.0	0.0	8239	7840	399	19.7	18.3	94.1	75.8
19	434	433	1	99.8	0.2	8673	8273	400	20.7	19.3	94.3	75.1

Table 6.1 Performance of Final model on training data

Testing	# Records	# Bads	# Goods	Fraud Rate							
	22341	267	22074	0.012							
	Bins Statistics						Cumulative Statistics				
Population Bin%	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cum Goods	Cum Bads	FPR	% Good	% Bad(FDR)	KS
0	224	49	175	21.9	78.1	224	49	175	0.3	0.2	65.5	65.3
1	223	194	29	87.0	13.0	447	243	204	1.2	1.1	76.4	75.3
2	224	215	9	96.0	4.0	671	458	213	2.2	2.1	79.8	77.7
3	223	219	4	98.2	1.8	894	677	217	3.1	3.1	81.3	78.2
4	223	222	1	99.6	0.5	1117	899	218	4.1	4.1	81.7	77.6
5	224	222	2	99.1	0.9	1341	1121	220	5.1	5.1	82.4	77.3
6	223	213	10	95.5	4.5	1564	1334	230	5.8	6.0	86.1	80.1
7	223	216	7	96.9	3.1	1787	1550	237	6.5	7.0	88.8	81.7
8	224	224	0	100.0	0.0	2011	1774	237	7.5	8.0	88.8	80.7
9	223	220	3	98.7	1.4	2234	1994	240	8.3	9.0	89.9	80.9
10	223	222	1	99.6	0.5	2457	2216	241	9.2	10.0	90.3	80.2
11	224	224	0	100.0	0.0	2681	2440	241	10.1	11.1	90.3	79.2
12	223	223	0	100.0	0.0	2904	2663	241	11.1	12.1	90.3	78.2
13	223	222	1	99.6	0.5	3127	2885	242	11.9	13.1	90.6	77.6
14	224	223	1	99.6	0.5	3351	3108	243	12.8	14.1	91.0	76.9
15	223	223	0	100.0	0.0	3574	3331	243	13.7	15.1	91.0	75.9
16	223	223	0	100.0	0.0	3797	3554	243	14.6	16.1	91.0	74.9
17	224	223	1	99.6	0.5	4021	3777	244	15.5	17.1	91.4	74.3
18	223	222	1	99.6	0.5	4244	3999	245	16.3	18.1	91.8	73.6
19	224	223	1	99.6	0.5	4468	4222	246	17.2	19.1	92.1	73.0

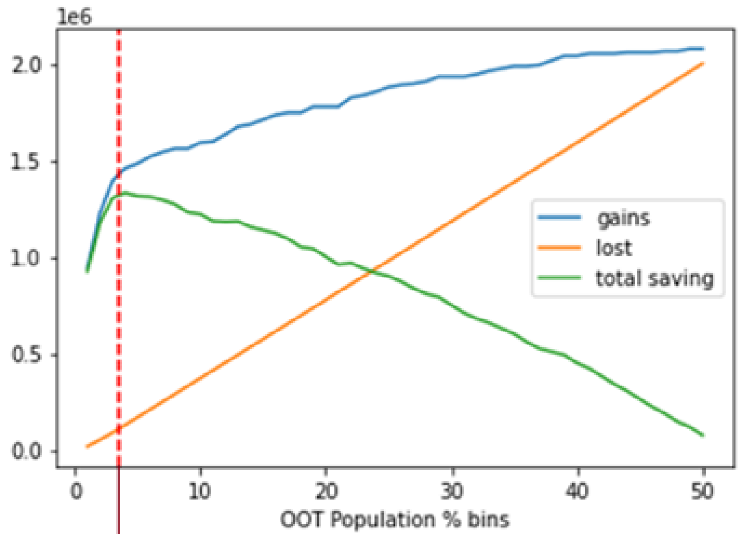
Table 6.2 Performance of Final model on testing data

OOT	# Records	# Bads	# Goods	Fraud Rate								
	27351	356	26995	0.013								
	Bin Statistics						Cumulative Statistics					
Population Bins%	# Records	# Goods	# Bads	% Goods	% Bads	Total #Records	Cum Goods	Cum Bads	FPR	% Good	% Bad(FDR)	KS
0	274	128	146	46.7	53.3	274	128	146	0.9	0.5	41.0	40.5
1	273	211	62	77.3	22.7	547	339	208	1.6	1.3	58.4	57.2
2	274	257	17	93.8	6.2	821	596	225	2.7	2.2	63.2	61.0
3	273	261	12	95.6	4.4	1094	857	237	3.6	3.2	66.6	63.4
4	274	268	6	97.8	2.2	1368	1125	243	4.6	4.2	68.3	64.1
5	273	272	1	99.6	0.4	1641	1397	244	5.7	5.2	68.5	63.4
6	274	270	4	98.5	1.5	1915	1667	248	6.7	6.2	69.7	63.5
7	273	266	7	97.4	2.6	2188	1933	255	7.6	7.2	71.6	64.5
8	274	271	3	98.9	1.1	2462	2204	258	8.5	8.2	72.5	64.3
9	273	270	3	98.9	1.1	2735	2474	261	9.5	9.2	73.3	64.2
10	273	269	4	98.5	1.5	3008	2743	265	10.4	10.2	74.7	64.6
11	274	265	9	96.7	3.3	3282	3008	274	11.0	11.1	77.0	65.8
12	273	273	0	100.0	0.0	3555	3281	274	12.0	12.2	77.0	64.8
13	274	272	2	99.3	0.7	3829	3553	276	12.9	13.2	77.5	64.4
14	273	272	1	99.6	0.4	4102	3825	277	13.8	14.2	77.8	63.6
15	274	269	5	98.2	1.8	4376	4094	282	14.5	15.2	79.2	64.0
16	273	270	3	98.9	1.1	4649	4364	285	15.3	16.2	80.1	63.9
17	274	270	4	98.5	1.5	4923	4634	289	16.0	17.2	81.2	64.0
18	273	272	1	99.6	0.4	5196	4906	290	16.9	18.2	81.5	63.3
19	274	271	3	98.9	1.1	5470	5177	293	17.7	19.2	82.3	63.1

Table 6.3 Performance of Final model on OOT data

From Training, testing and OOT performance table, we can conclude that there is not overfitting happened on the model we chose and its adoption on OOT performed well. To discover a optimal score cutoff point, we adopt 50% population to create 50 bins and make assumptions below to calculate the gains, lost and total savings:

- Assume $2000 gain for every fraud that's caught (blue curve)
- Assume $50 loss for every false positive (a good that's flagged as a bad) (orange)
- Since the OOT is 1/3 of the annual transactions, we multiply numbers by 3 and label the plot as annual savings.



3.5% is a recommended cut off point

Figure 6.1 Gains, Lost and Total Saving under final model performance

From the line plot above, we recommend a score cut off at 3.5%, which is slightly before the highest point.

To explore more useful insights, we chose a certain card number and merchant number separately, creating two plots for each of them. One plot is over time while the other is over transaction count.

For Cardnum 5142299705, we plot its change of average fraud score from September, which is shown below.

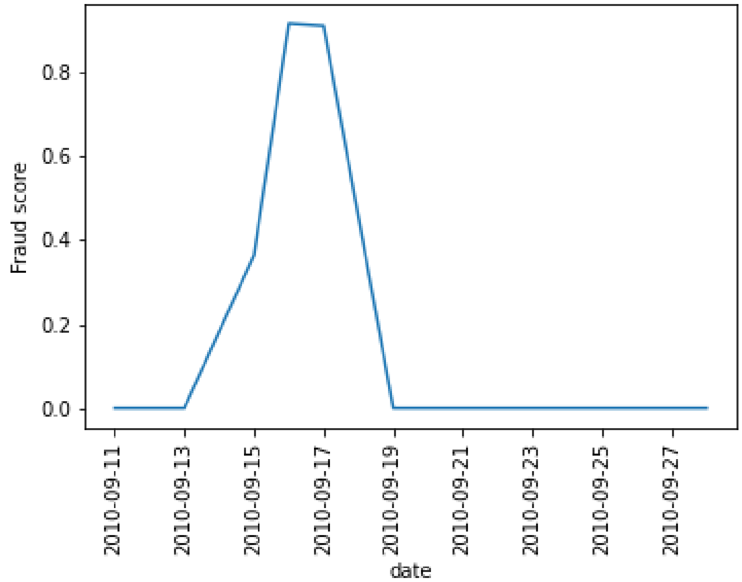


Figure 6.2 Change of average fraud score from September

As we can see, there is a greatly score increase between 13th September and 19th September. To figure out the reason behind that, we plot the changes in the score with the number of transactions during this time. From the plot, we find the sudden growth may come from the sharp increase in the number of transactions.

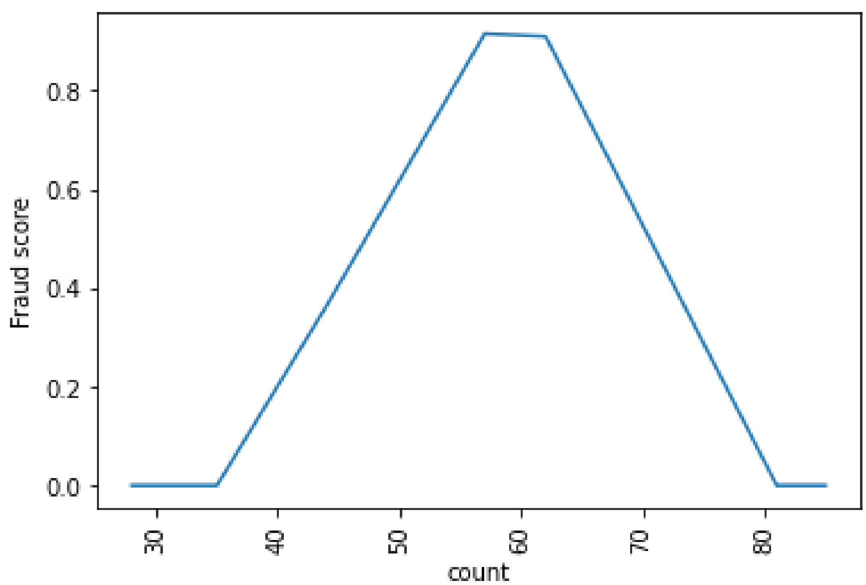


Figure 6.2 the score with the number of transactions during this time

For Merchnum 6005030600003, about change of average fraud score from September to October, and its related changes in the score with the number of transactions. There is a clearly sharp growth from 5th October, which could be explained with a continually transaction increase.

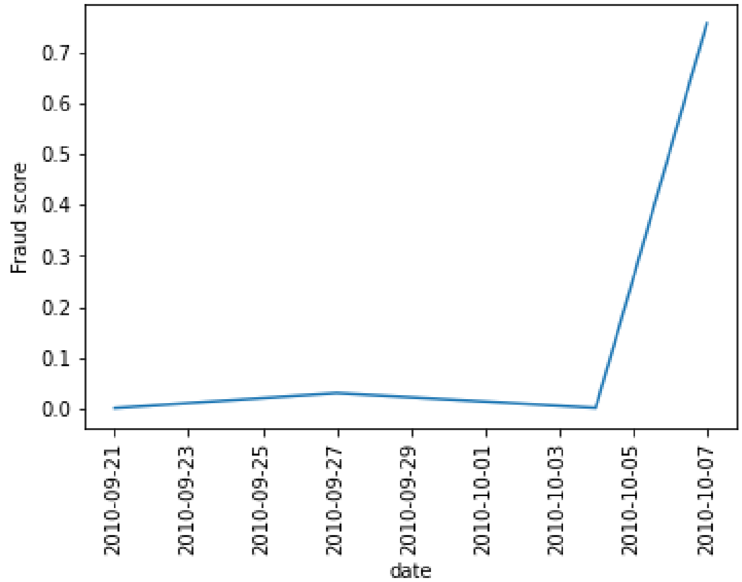


Figure 6.3 Change of average fraud score from September to October

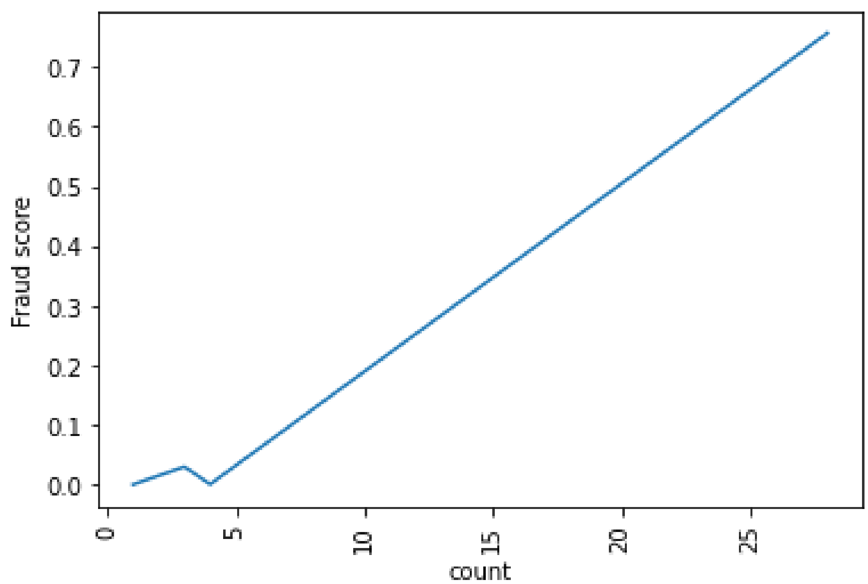


Figure 6.4 the score with the number of transactions during this time

From two cases above, we can generally conclude that our model could perform well for detecting abnormal transaction increase among cards or merchants.

7 Conclusion

In this project, we start with studying the data, evaluating data quality, and filling missing values. Next, we create candidate variables following the logic of common fraud cases in reality. Then using KS and FDR ratio to select the most efficient 80 variables and adapting wrapping to decrease dimension to 30 based on the first 8 months data. The other 4 months are considered as out-of-Time (OOT).

The modeling process uses FDR as a metric of performance. We run each model of different parameters 10 times and extract mean FDR for train, test, and OOT data to compare models. Finally, we implement the model of the best performance to create bins for data by the predicted fraud score and figure out the most profitable threshold of fraud score.

The project also has room for future improvement. Unbalanced is a significant nature of credit card fraud records which is to say that in a large number of records, only very few frauds hidden in it. Therefore, to build a model that can capture these records, a weight factor will be greatly helpful. Another approach to strengthen the accuracy of the model is chopping. This will "chop" the most obvious records and force the model to focus on more ambiguous records.

References

Figure 4.1 Points go sparse and become outliers with a high dimensionality

https://aiaspirant.com/curse-of-dimensionality/

Figure 6.1 Comparison of Linear Regression and Logistic Regression:

http://www.srcmini.com/46323.html

Figure 6.2 Illustration of Neural Network:

https://www.fmi.com.cn/ueditor/php/upload/image/20180729/1532854645968516.jpg

Figure 6.3 Illustration of Neural Network:

https://medium.com/swlh/gradient-boosting-trees-for-classification-a-beginners-guide-596b594a14ea

Figure 6.4 Illustration of Random Forest:

https://www.cnblogs.com/ooon/p/5674527.html

Figure 6.5 Illustration of Support Vector Machine:

https://zhuanlan.zhihu.com/p/49331510

Appendix A: Data Quality Report

Fild description:

The data is Card Transacation Data, which includes actual credit card purchases' information provided by U.S. government organization during the year 2010. The purpose of this data is to indicate whether the transaction is a fraud or not. There are 10 number of fields in the data: Amount, Recnum, Cardnum, Merchnum, Merch Description, Merch State, Merch Zip, Transtype, Fraud and Data. There are 96,753 number of records in this data, and 1,059 records are indicated as a fraud.

Table 8.1: File Description

Dataset Name	Card Transacation Data
Dataset Purpose	The data is about credit card transaction information, which indicates whether this transaction is a fraud or not.
Data Source	Came from U.S. government organization
Time Period	From Jan. 1st, 2010 to Dec. 31st, 2010
Number of Fields	10 Fields in total – 1 numeric, 8 categorical, 1date
Number of Records	96,753

Summary Statistics Table:

All fields in the data can be treated as categorical, numeric and date: 1 field is numeric, 8 fields are cateforical and 1 field is date. Among all fields, 7 fields are fully populated and other 3 fields are not fully populated. Key statistics of these fields are summarized as follows.

Table 8.2 Summary Statistics of Numeric Fields

	Field Type	count	mean	std	min	max	Unique Value	# Zero	% populated
Amount	Numeric	96753	427.89	10,006.14	0.01	3,102,045.53	34909	0	100

Table 8.3 Summary Statistics of Categorical Fields

	Field Type	count	% Populated	Unique values	Most Common Field Values
Recnum	Categorical	96753	100	96,753	N/A
Cardnum	Categorical	96753	100	1645	5142148452
Merchnum	Categorical	93378	96.5	13,091	930,090,121,224
Merch Description	Categorical	96753	100	13,126	GSA-FSS-ADV
Merch State	Categorical	95558	98.8	227	TN
Merch Zip	Categorical	92097	95.2	4,567	38118
Transtype	Categorical	96753	100	4	P
Fraud	Categorical	96753	100	2	0

Table 8.4 Summary Statistics of Categorical Fields

	Field Type	count	Unique Values	Most Common Field Values	Min	Max	Records that have a value	% populated
Date	Date	96753	365	2010/2/28	2010/1/1	2010/12/31	96753	100

Field Description and Distribution:

Field 1: Amount

Table 8.5: Amount

Description	The amount of money of each transaction, exclude outliers > 1000, data in histogram is 89.10% populated
Type	Numeric
Mean	427.89
Maximum	3,102,045.53
Minimum	0.01
Std	10,006.14

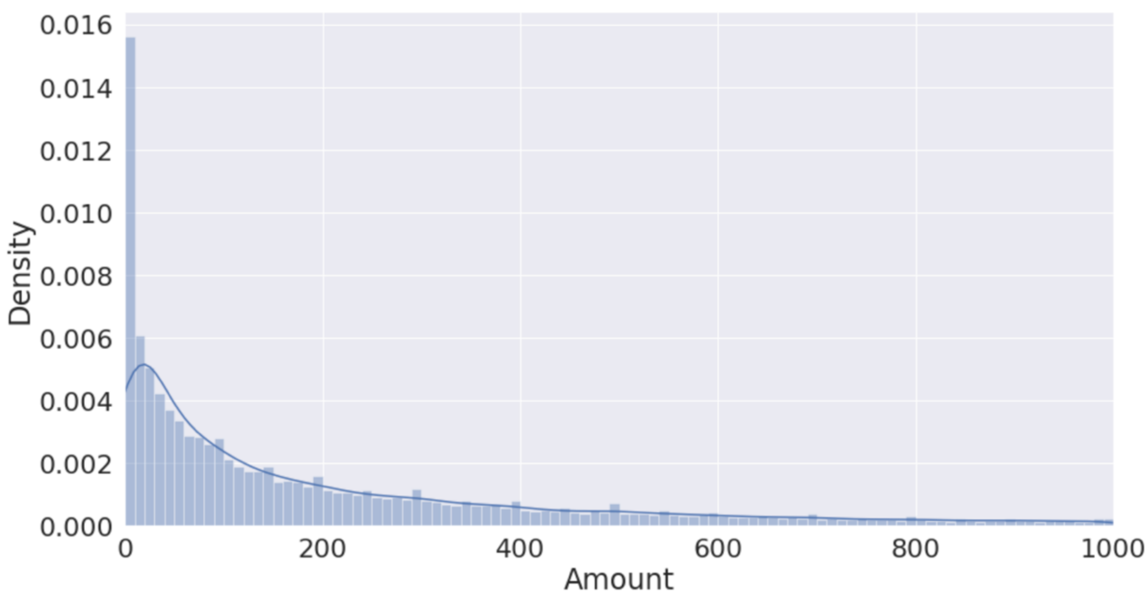


Figure 8.1: Frequency Distribution of Amount Field

Field 2: Transtype

Table 8.6: Transtype

Description	Transaction type of each transaction
Type	Categorical
Most Common Field Vlaue	'P' occurred the most for 96,396 times

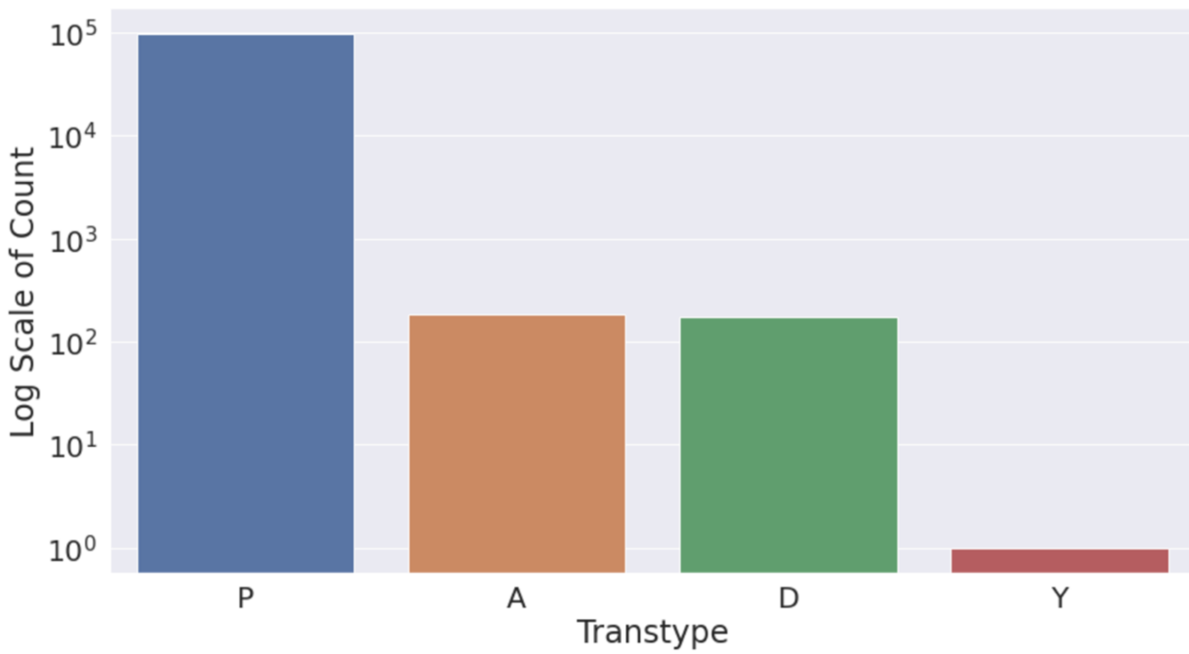


Figure 8.2: Frequency Distribution of Transtype Field

Field 3: Recnum

Table 8.7: Recnum

Description	Time order of the data
Type	Categorical
Most Common Field Vlaue	All the records for this field are unique

Field 4: Cardnum

Table 8.8: Cardnum

Description	Card number of each transaction
Type	Categorical
Most Common Field Vlaue	‘5142148452’ occurred the most for 1,192 times

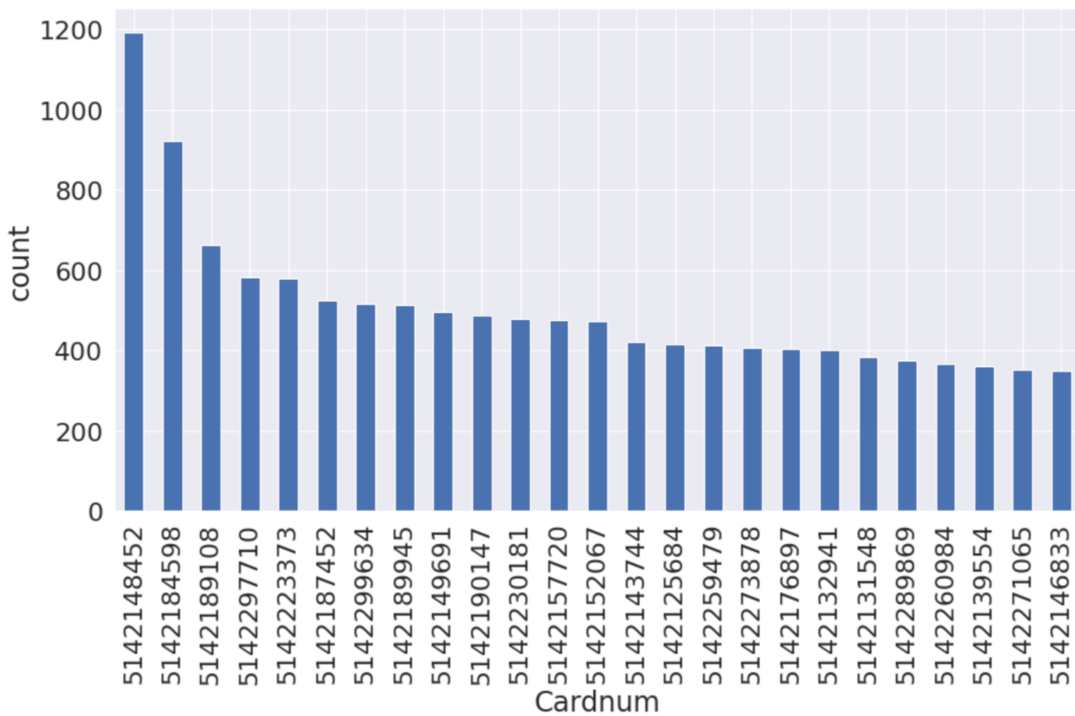


Figure 8.3: Frequency Distribution of Cardnum Field

Field 5: Date

Table 8.9: Date

Description	The date of the transaction. Month, day and year only (no time of day). Data in histogram is 100% populated.
Type	Date
Unique Values	365
Maximum	2010/12/31
Minimum	2010/1/1
Most Common Field Value	‘2010/2/28’ occurred the most for 684 times

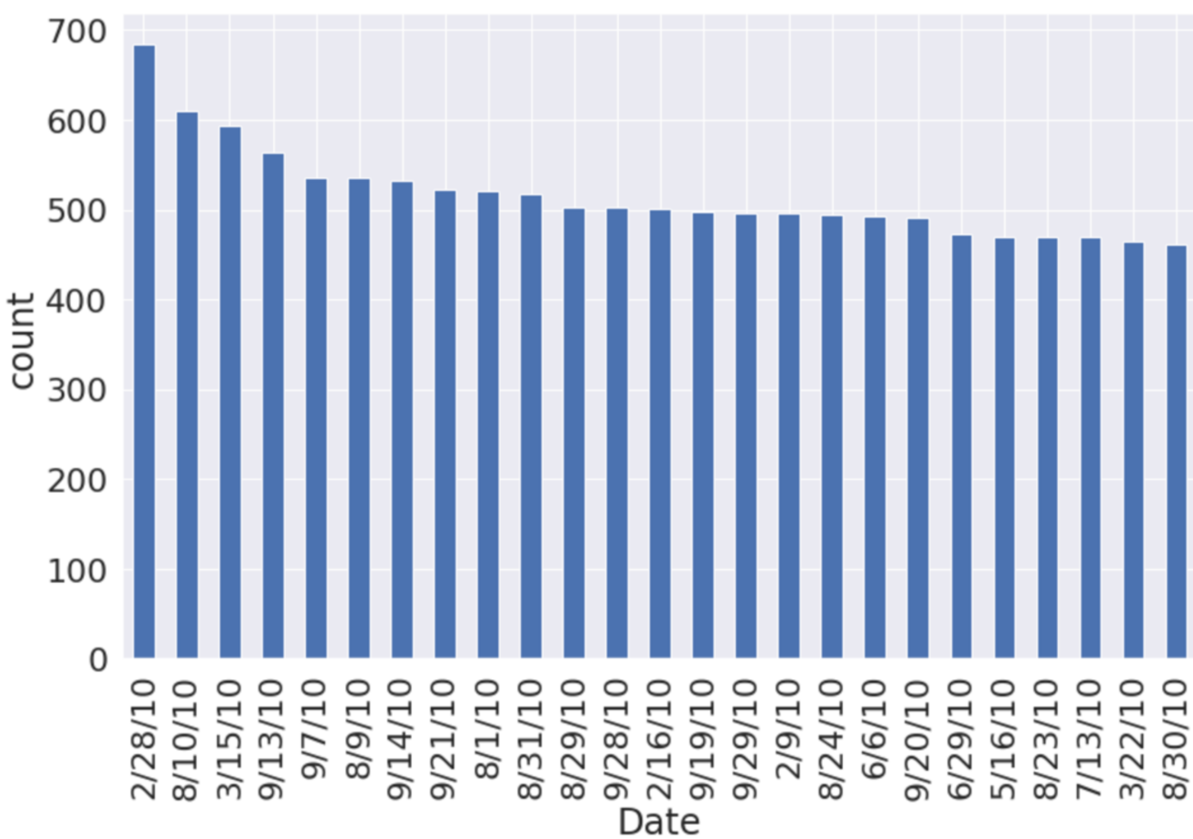


Figure 8.4: Frequency Distribution of Date Field

Field 6: Merchnum

Table 8.10: Merchnum

Description	Merchant number of each transaction
Type	Categorical
Most Common Field Vlaue	‘930090121224’ occurred the most for 9,310 times

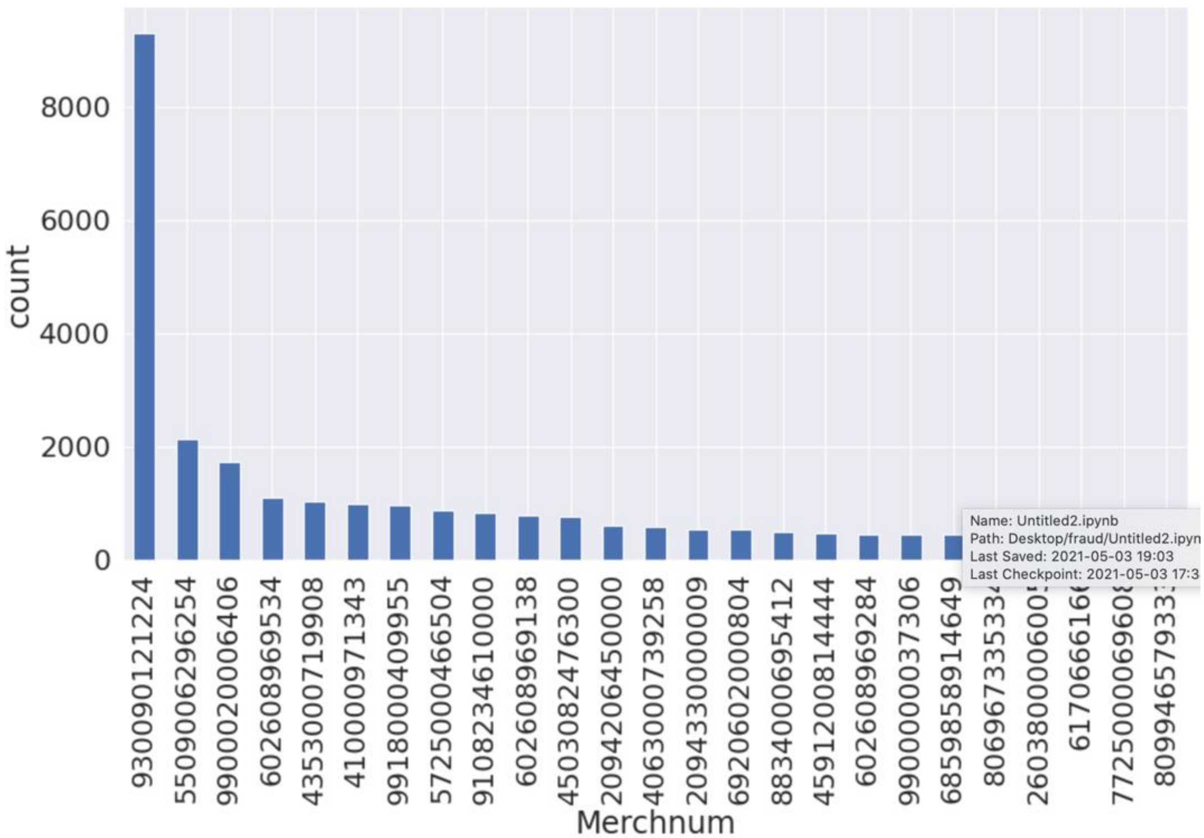


Figure 8.5: Frequency Distribution of Merchnum Field

Field 7: Merch Description

Table 8.11: Merch Description

Description	Merchant Company Name
Type	Categorical
Most Common Field Vlaue	‘GSA-FSS-ADV’ occurred the most for 1,688 times

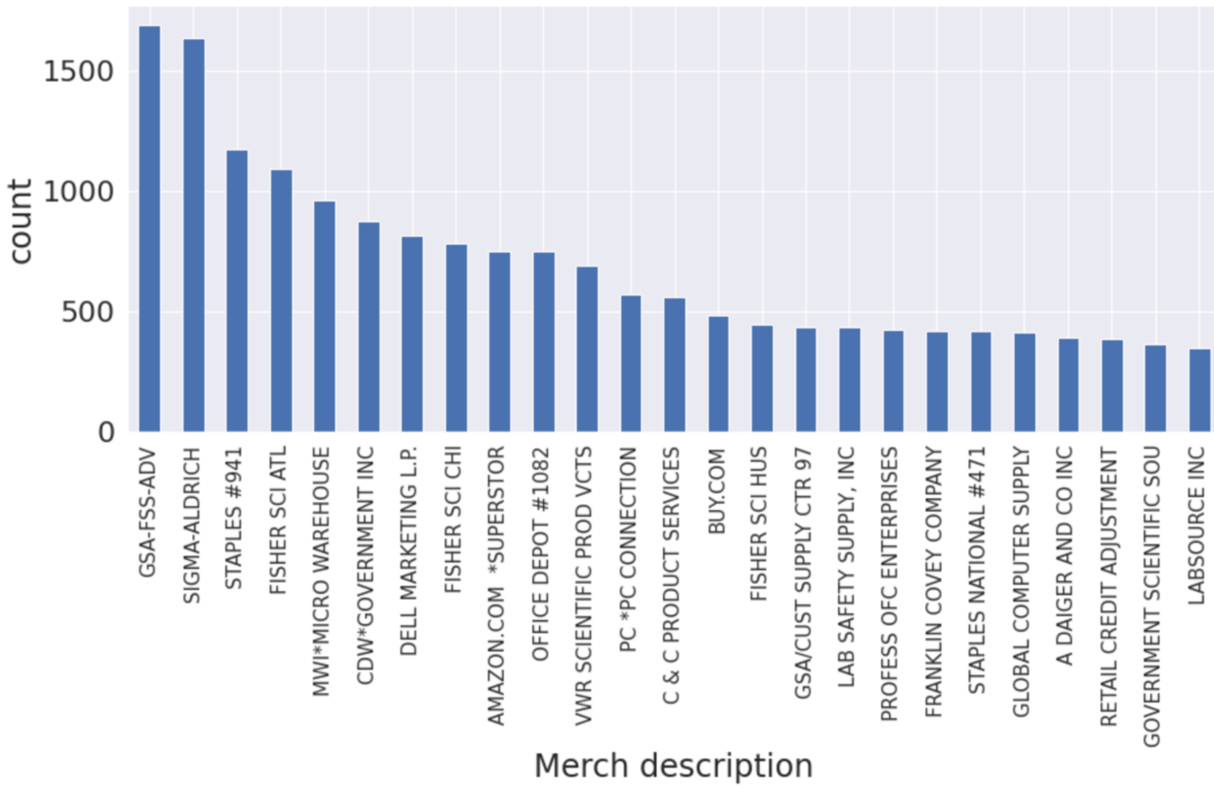


Figure 8.6: Frequency Distribution of Merch Description Field

Field 8: Merch State

Table 8.12: Merch State

Description	State where the Merchant in
Type	Categorical
Most Common Field Vlaue	'TN' occurred the most for 12,035 times

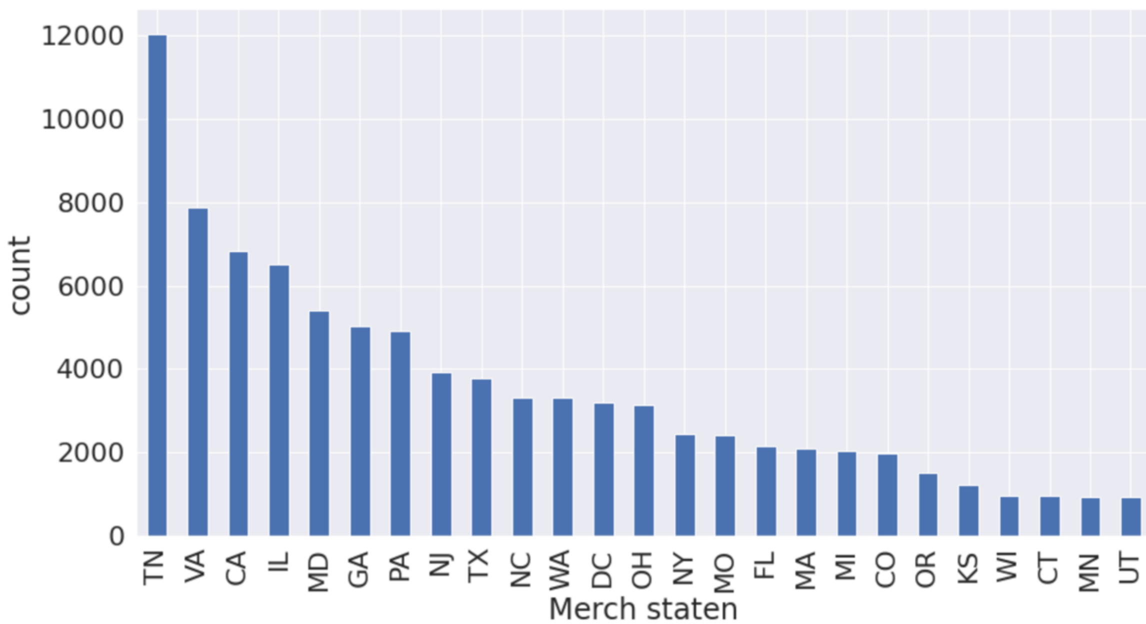


Figure 8.7: Frequency Distribution of Merch State Field

Field 9: Merch Zip

Table 8.13: Merch Zip

Description	The zip code of Merchant
Type	Categorical
Most Common Field Vlaue	'38118' occurred the most for 11,868 times

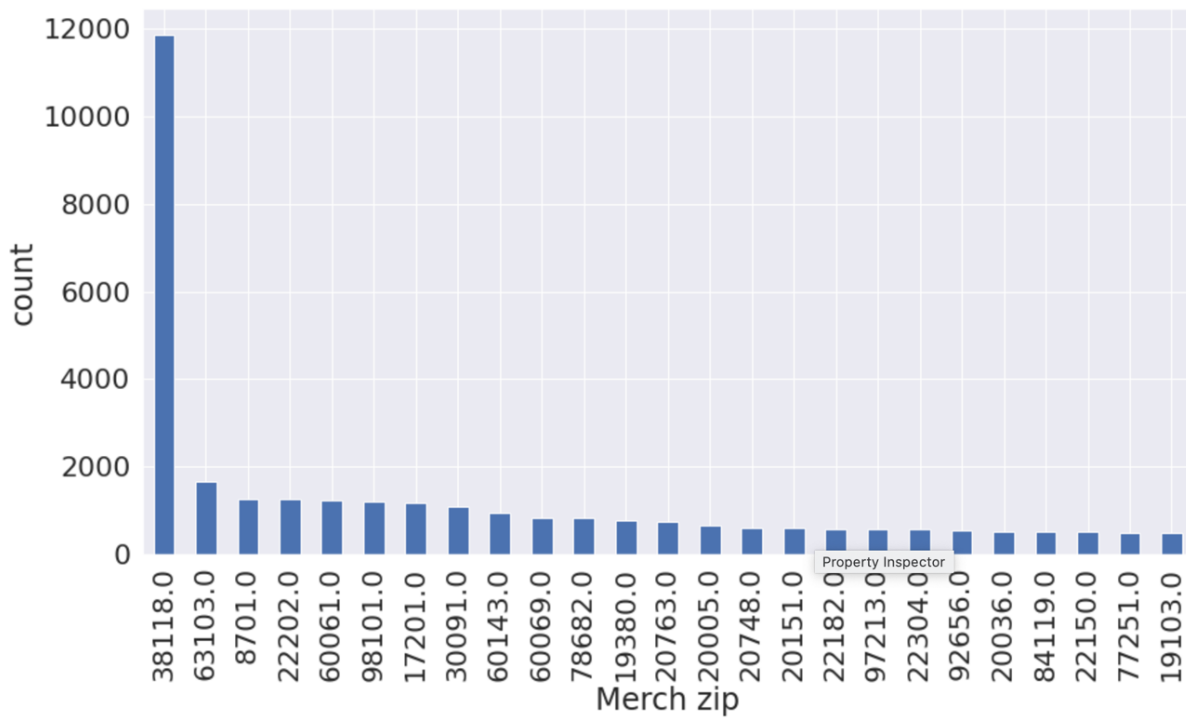


Figure 8.8: Frequency Distribution of Merch Zip Field

Field 10: Fraud

Table 8.14: Fraud

Description	Whether this transaction is fraud or not ('1' = Fraud, '0' = Not fraud)
Type	Categorical
Most Common Field Vlaue	'0' occurred the most for 95,694 times, "1" occurred for 1,059 times

Figure 8.9: Frequency Distribution of Fraud Field

Appendix B: Statistics Variables

	mean	std	min	max
Cardnum_day_since	5.419857464443910	17.18793296218270	0.0	356.0
Merchnum_day_since	29.84113613494200	65.91536684347890	0.0	364.0
card_merch_day_since	81.01504196188680	99.4105093913853	0.0	364.0
card_zip_day_since	70.67455418737100	93.0911013618731	0.0	364.0
card_state_day_since	38.50597010280400	67.38494857144710	0.0	364.0
merch_zip_day_since	30.807566625517400	67.04622199079890	0.0	364.0
merch_state_day_since	29.93547517038910	66.04981263119710	0.0	364.0
card_merch_zip_day_since	81.72435864186640	99.90020545096600	0.0	364.0
card_merch_state_day_since	81.10603027065160	99.47519095404210	0.0	364.0
Cardnum_count_past_0	2.4736558191645000	6.002116198857810	1.0	146.0
Cardnum_avg_0	393.5574898450110	726.8458455852550	0.01	28392.84
Cardnum_max_0	498.2058086869990	1030.9573600748500	0.01	47900.0
Cardnum_med_0	381.35218466342600	718.7290925073750	0.01	28392.84
Cardnum_total_0	741.6455649034840	3431.4461307276600	0.01	218301.83
Cardnum_actual/avg_0	1.0019535804431900	0.4449284694816710	4.20132761952777E-05	23.7912340420984
Cardnum_actual/max_0	0.8752297921212110	0.28284808289443000	1.4160495050907E-05	1.0
Cardnum_actual/med_0	1.4103174995030400	9.89588971483239	8.94674450334385E-05	657.8947368421050
Cardnum_actual/total_0	0.772758278215924	0.3502753761014780	1.40044253984259E-05	1.0
Cardnum_count_past_1	3.367106860172000	7.944994584257990	1.0	177.0

	mean	std	min	max
Cardnum_avg_1	395.45068549285400	675.8256592221490	0.01	28392.84
Cardnum_max_1	610.8730460491430	1212.8642673374200	0.01	47900.0
Cardnum_med_1	364.2218742803180	658.7215608317700	0.01	28392.84
Cardnum_total_1	1110.0455872071100	5669.434126818710	0.01	307468.0600000000
Cardnum_actual/avg_1	0.9981707088954640	0.6401660683833210	5.52600676935829E-05	23.7912340420984
Cardnum_actual/max_1	0.7721749274103550	0.35850507326817200	1.4160495050907E-05	1.0
Cardnum_actual/med_1	1.8329946129450500	13.351207057736100	8.94674450334385E-05	674.7611336032390
Cardnum_actual/total_1	0.6359693559412100	0.393745825424382	1.38150169233957E-05	1.0
Cardnum_count_past_3	4.794267456456110	11.450064019850400	1.0	251.0
Cardnum_avg_3	395.8529294763980	629.3743405182390	0.01	28392.84
Cardnum_max_3	739.7859984231930	1367.5784927204600	0.01	47900.0
Cardnum_med_3	341.3568812307400	592.6485314308110	0.01	28392.84
Cardnum_total_3	1512.9329523740700	6115.505328808330	0.01	310843.0600000000
Cardnum_actual/avg_3	1.0024146314058600	0.8429018101256070	5.52600676935829E-05	38.00259670364970
Cardnum_actual/max_3	0.6735701781649560	0.3973695448783160	1.4160495050907E-05	1.0
Cardnum_actual/med_3	2.340588008341570	17.122958918382500	0.0002261420171867930	1570.855614973260
Cardnum_actual/total_3	0.513770107076747	0.40048741087085400	1.38150169233957E-05	1.0
Cardnum_count_past_7	7.627799620320130	16.612641646325500	1.0	369.0
Cardnum_avg_7	397.18570119775000	560.0757100639390	0.14	25500.0
Cardnum_max_7	960.4289108582070	1603.1291440925400	0.14	47900.0
Cardnum_med_7	307.1587480419520	501.7307719646100	0.14	25500.0

	mean	std	min	max
Cardnum_total_7	2384.036098115070	7158.500841268020	0.14	312616.0600000000
Cardnum_actual/avg_7	0.9944632451323750	1.0772847339152500	5.52600676935829E-05	59.867504671309600
Cardnum_actual/max_7	0.5371763866302020	0.4127837183979270	1.4160495050907E-05	1.0
Cardnum_actual/med_7	2.9856564665587300	27.091155861616800	0.0001639344262295080	5747.538461538460
Cardnum_actual/total_7	0.3575300079059750	0.36726485824258600	1.38150169233957E-05	1.0
Cardnum_count_past_14	11.800014523273500	20.717933181203700	1.0	380.0
Cardnum_avg_14	396.99347383779300	522.9165589330730	0.14	25500.0
Cardnum_max_14	1188.6369737647300	1829.4995721525500	0.14	47900.0
Cardnum_med_14	279.0155984107320	456.1247076631080	0.14	25500.0
Cardnum_total_14	3768.183808106090	9421.917378764420	0.14	313995.0600000000
Cardnum_actual/avg_14	0.9983638051180760	1.2730719118676300	5.52600676935829E-05	71.33216664838750
Cardnum_actual/max_14	0.43731090449052900	0.4001637204108650	1.4160495050907E-05	1.0
Cardnum_actual/med_14	3.47249671253273	29.246115094636500	0.0001393116610825910	6145.636363636360
Cardnum_actual/total_14	0.250992811511247	0.3173053175042670	1.30748949595626E-05	1.0
Cardnum_count_past_30	20.359461394026800	30.906710487132000	1.0	426.0
Cardnum_avg_30	396.58450373681200	479.34251908994800	0.17	25500.0
Cardnum_max_30	1482.1747297114800	2076.8761441000800	0.17	47900.0
Cardnum_med_30	250.9419457555640	402.4264074932110	0.17	25500.0
Cardnum_total_30	6675.654419224700	14591.232856682700	0.17	353997.2900000000
Cardnum_actual/avg_30	1.0055215342355500	1.610792313333180	5.00904410741617E-05	137.9869655724580
Cardnum_actual/max_30	0.3434314928454380	0.369031758926377	9.36886632033091E-06	1.0

	mean	std	min	max
Cardnum_actual/med_30	3.952411863539500	32.61331152996260	0.0001199616122840690	6288.5581395348800
Cardnum_actual/total_30	0.1587537534061380	0.2512612138564970	5.13177105116632E-06	1.0
Merchnum_count_past_0	6.88036972104941	18.990212847322900	1.0	260.0
Merchnum_avg_0	395.5387295944950	753.1442987268120	0.01	27218.0
Merchnum_max_0	513.8221625154380	1035.0241816555500	0.01	47900.0
Merchnum_med_0	379.528930205302	742.3346810425130	0.01	27218.0
Merchnum_total_0	814.2191609697420	2883.1723054211200	0.01	217467.18
Merchnum_actual/avg_0	1.0032667872203500	0.6430889021560150	8.94674450334385E-05	37.9561424763751
Merchnum_actual/max_0	0.8065597315208760	0.3443309221659230	4.47357237121704E-05	1.0
Merchnum_actual/med_0	1.2766889253471300	3.9927572148437400	8.94674450334385E-05	553.8194444444450
Merchnum_actual/total_0	0.7112266091381920	0.3976114984007920	4.47337225167192E-05	1.0
Merchnum_count_past_1	11.748633256221700	31.611340676005200	1.0	327.0
Merchnum_avg_1	397.4570011754880	741.7365870103970	0.01	27218.0
Merchnum_max_1	611.2902195089120	1293.2077881886000	0.01	47900.0
Merchnum_med_1	368.79498734400900	728.9218755340770	0.01	27218.0
Merchnum_total_1	1216.6088834714800	4482.082901617510	0.01	306633.4100000000
Merchnum_actual/avg_1	1.000582149140310	0.7857991258655910	8.94674450334385E-05	43.42602163523040
Merchnum_actual/max_1	0.7362614854907080	0.3852068624542630	2.99371249934843E-05	1.0
Merchnum_actual/med_1	1.3980264439620100	4.482329832231520	8.94674450334385E-05	405.449591280654
Merchnum_actual/total_1	0.6255970731231890	0.4227974640500190	2.82986486230044E-05	1.0
Merchnum_count_past_3	21.352739193128400	55.15716003845700	1.0	466.0

	mean	std	min	max
Merchnum_avg_3	397.30077492523400	719.5436560091260	0.01	27218.0
Merchnum_max_3	706.4573087336930	1417.1553249472700	0.01	47900.0
Merchnum_med_3	359.10515882237400	707.3082219419440	0.01	27218.0
Merchnum_total_3	1754.265649864600	5415.770579688840	0.01	307302.5800000000
Merchnum_actual/avg_3	1.0008746084713600	0.9390602130965250	8.94674450334385E-05	64.09280649108990
Merchnum_actual/max_3	0.6798687747361060	0.4084716451215100	2.99371249934843E-05	1.0
Merchnum_actual/med_3	1.490439687099310	5.016979787456040	8.94674450334385E-05	467.8285714285710
Merchnum_actual/total_3	0.5605356324743340	0.4340056412455310	1.84107143507023E-05	1.0
Merchnum_count_past_7	42.887652105356000	106.07137569047000	1.0	762.0
Merchnum_avg_7	396.48436264605700	682.1470224751130	0.01	27218.0
Merchnum_max_7	883.3783577289190	1751.3683580699000	0.01	47900.0
Merchnum_med_7	344.04851691442700	670.4268962989360	0.01	27218.0
Merchnum_total_7	3005.6567033206100	7081.8891558188400	0.01	313984.5500000000
Merchnum_actual/avg_7	0.9978772312032250	1.12103819816434	0.0001244187312399880	82.46059846404050
Merchnum_actual/max_7	0.6070909488323900	0.4254729302143540	2.99371249934843E-05	1.0
Merchnum_actual/med_7	1.564416698275830	4.710161394120110	0.0001350256548744260	473.9842105263160
Merchnum_actual/total_7	0.4716647537623040	0.4336334417304750	1.59201112061663E-05	1.0
Merchnum_count_past_14	77.42573939023000	191.79115443506800	1.0	1091.0
Merchnum_avg_14	397.77330477197400	655.3880036815300	0.01	27218.0
Merchnum_max_14	1078.9790456135200	2154.0037830579100	0.01	47900.0
Merchnum_med_14	333.75416900941500	637.1137130586920	0.01	27218.0

	mean	std	min	max
Merchnum_total_14	5044.620943079140	10464.815337761000	0.01	319334.6800000000
Merchnum_actual/avg_14	0.9958621272404580	1.285389933824540	0.0001238554430043790	133.5333353248190
Merchnum_actual/max_14	0.5511232195552970	0.4297124036791060	2.99371249934843E-05	1.0
Merchnum_actual/med_14	1.6195612741994500	4.653939150328190	0.0001350256548744260	473.9842105263160
Merchnum_actual/total_14	0.4048856321671140	0.4236444486790530	9.04250357416852E-06	1.0
Merchnum_count_past_30	149.43319812857200	376.16252386096700	1.0	1828.0
Merchnum_avg_30	397.1855189160630	614.1737038107620	0.01	27218.0
Merchnum_max_30	1379.3728862932100	2925.1876617425300	0.01	47900.0
Merchnum_med_30	320.0217699202420	583.3583266833970	0.01	27218.0
Merchnum_total_30	9414.114866126810	18306.61809414610	0.01	320373.0000000000
Merchnum_actual/avg_30	0.9975914584330570	1.4663439999924800	5.05178075271533E-05	172.6358842587750
Merchnum_actual/max_30	0.4916692348205340	0.4265851538168060	1.83482871873911E-05	1.0
Merchnum_actual/med_30	1.6817322694576700	4.519005790107470	6.75173857268247E-05	481.5882352941180
Merchnum_actual/total_30	0.3360479567177290	0.4044789000000210	4.0417775141877E-06	1.0
card_merch_count_past_0	2.097005093519510	5.910720315287180	1.0	145.0
card_merch_avg_0	395.79548822027500	796.9244773481780	0.01	28392.84
card_merch_max_0	421.2988777659050	935.804824774836	0.01	47900.0
card_merch_med_0	393.4523657893930	790.6278397347720	0.01	28392.84
card_merch_total_0	528.9286867848610	2621.904523589270	0.01	217467.18
card_merch_actual/avg_0	0.999829471433516	0.2208700126228540	8.94674450334385E-05	20.24247672656420
card_merch_actual/max_0	0.956721410691394	0.1662173632839750	4.47357237121704E-05	1.0

	mean	std	min	max
card_merch_actual/med_0	1.0266486899410700	0.6765806947906870	8.94674450334385E-05	100.0
card_merch_actual/total_0	0.8845090561404220	0.274822569080925	4.47337225167192E-05	1.0
card_merch_count_past_1	2.416797203232470	7.593732798303420	1.0	177.0
card_merch_avg_1	397.1925216136950	799.4981746532920	0.01	28392.84
card_merch_max_1	432.584154071183	1010.288987531200	0.01	47900.0
card_merch_med_1	394.4320962270640	794.0080605559890	0.01	28392.84
card_merch_total_1	599.5728184487100	4020.3486989250300	0.01	306633.4100000000
card_merch_actual/avg_1	0.9972979760148010	0.2550183729079900	8.94674450334385E-05	20.24247672656420
card_merch_actual/max_1	0.9413394213739110	0.1933376049866120	4.47357237121704E-05	1.0
card_merch_actual/med_1	1.0326693565770200	0.70832958111957	8.94674450334385E-05	71.11111111111110
card_merch_actual/total_1	0.8588057129422570	0.2988076158101400	4.47337225167192E-05	1.0
card_merch_count_past_3	3.0263701152525500	10.97630552745810	1.0	248.0
card_merch_avg_3	398.1088690521520	797.2861335451370	0.01	28392.84
card_merch_max_3	441.41006576967900	1014.5914781804200	0.01	47900.0
card_merch_med_3	394.6648400365180	792.117861960962	0.01	28392.84
card_merch_total_3	631.233482992212	4063.0028297761200	0.01	306633.4100000000
card_merch_actual/avg_3	0.9948002263609010	0.2964869675276880	8.94674450334385E-05	20.24247672656420
card_merch_actual/max_3	0.9210525948220630	0.2228440980055540	4.47357237121704E-05	1.0
card_merch_actual/med_3	1.0535906558640300	1.5011273651083100	8.94674450334385E-05	301.1031518624640
card_merch_actual/total_3	0.8279756860408540	0.3246456525882020	4.47337225167192E-05	1.0
card_merch_count_past_7	4.050447628038220	15.654321419036400	1.0	358.0

	mean	std	min	max
card_merch_avg_7	399.82025280146000	792.5192243040290	0.01	28392.84
card_merch_max_7	459.0136919198730	1022.8167732452100	0.01	47900.0
card_merch_med_7	394.6622679129090	787.8122005339970	0.01	28392.84
card_merch_total_7	690.6114004585240	4104.138244905570	0.01	306633.4100000000
card_merch_actual/avg_7	0.9893295149850480	0.3626810494240160	8.94674450334385E-05	20.24247672656420
card_merch_actual/max_7	0.8875167501354710	0.2630259806776110	4.47357237121704E-05	1.0
card_merch_actual/med_7	1.0799209621055200	2.1987608898877300	8.94674450334385E-05	442.8697962798940
card_merch_actual/total_7	0.7799977288416480	0.3565705658722070	4.47337225167192E-05	1.0
card_merch_count_past_14	5.342469163978130	19.044508318182400	1.0	369.0
card_merch_avg_14	401.8828749171510	791.3055934531400	0.01	28392.84
card_merch_max_14	480.7200715789900	1047.3193505325500	0.01	47900.0
card_merch_med_14	394.68393757068000	787.4954167914590	0.01	28392.84
card_merch_total_14	772.1092779858310	4170.6448348325500	0.01	306633.4100000000
card_merch_actual/avg_14	0.9870732489017860	0.4318085804779140	8.94674450334385E-05	23.11321693279610
card_merch_actual/max_14	0.854350084663659	0.2941683718590440	4.47357237121704E-05	1.0
card_merch_actual/med_14	1.1129694906511800	2.287056816894250	8.94674450334385E-05	400.0
card_merch_actual/total_14	0.7341230381409850	0.379470666677243	4.47337225167192E-05	1.0
card_merch_count_past_30	7.737460709358180	27.430197589877600	1.0	409.0
card_merch_avg_30	404.09159582085200	786.6047208729920	0.01	28392.84
card_merch_max_30	513.7809415230760	1073.368394879080	0.01	47900.0
card_merch_med_30	393.37383684140500	785.1282599541560	0.01	28392.84

	mean	std	min	max
card_merch_total_30	926.9286743363380	4304.302282890020	0.01	306633.4100000000
card_merch_actual/avg_30	0.9834696990989390	0.5141092602181470	0.0001012726597576210	25.02564005165990
card_merch_actual/max_30	0.8098454514615320	0.3279177625071520	4.47357237121704E-05	1.0
card_merch_actual/med_30	1.1515185122466000	2.285311759731140	6.75173857268247E-05	397.8609625668450
card_merch_actual/total_30	0.6743707070605940	0.4002527316115950	3.08737697855255E-05	1.0
card_zip_count_past_0	2.1179912237932700	5.940791140554750	1.0	146.0
card_zip_avg_0	395.6617444020470	794.784724355782	0.01	28392.84
card_zip_max_0	422.72435013537600	936.570247400711	0.01	47900.0
card_zip_med_0	393.1422954552550	788.422831346689	0.01	28392.84
card_zip_total_0	532.0606200400460	2623.539536880330	0.01	217467.18
card_zip_actual/avg_0	0.9998438371937170	0.2315463797880300	8.94674450334385E-05	20.24247672656420
card_zip_actual/max_0	0.9525201977229490	0.1747404583629160	4.47357237121704E-05	1.0
card_zip_actual/med_0	1.032971403803560	1.0812183546545000	8.94674450334385E-05	234.7928176795580
card_zip_actual/total_0	0.8789238894310040	0.2797001400004950	4.47337225167192E-05	1.0
card_zip_count_past_1	2.476031411765930	7.821298032863150	1.0	177.0
card_zip_avg_1	397.2198064235460	797.2063169172440	0.01	28392.84
card_zip_max_1	435.3610908015750	1011.9461648838400	0.01	47900.0
card_zip_med_1	394.193235940956	791.6672960071880	0.01	28392.84
card_zip_total_1	605.9220199798790	4022.902518490990	0.01	306633.4100000000
card_zip_actual/avg_1	0.9968066868094690	0.2768368435829370	8.94674450334385E-05	20.17129251217170
card_zip_actual/max_1	0.9336882377279630	0.2062327451794610	4.47357237121704E-05	1.0

	mean	std	min	max
card_zip_actual/med_1	1.0422413231873200	1.1326813925140600	8.94674450334385E-05	231.5940054495910
card_zip_actual/total_1	0.8491437475968800	0.3062425872874410	4.47337225167192E-05	1.0
card_zip_count_past_3	3.1276492007012700	11.283277902656700	1.0	251.0
card_zip_avg_3	398.0906138642790	794.0034844610900	0.01	28392.84
card_zip_max_3	446.21801778063200	1018.0758588566700	0.01	47900.0
card_zip_med_3	394.0808260630550	788.8846639027270	0.01	28392.84
card_zip_total_3	642.0471446206870	4067.0192215392200	0.01	306633.4100000000
card_zip_actual/avg_3	0.9938943238246760	0.3238239526236380	8.94674450334385E-05	20.17129251217170
card_zip_actual/max_3	0.9094137762475200	0.2393285657895630	4.47357237121704E-05	1.0
card_zip_actual/med_3	1.0698854541417300	1.8530468988901700	8.94674450334385E-05	301.1031518624640
card_zip_actual/total_3	0.8144639460786370	0.3335078022667420	4.47337225167192E-05	1.0
card_zip_count_past_7	4.249333485481910	16.307700633084400	1.0	369.0
card_zip_avg_7	399.99344852692800	787.93911172088	0.01	28392.84
card_zip_max_7	467.5064052823220	1028.9917977613500	0.01	47900.0
card_zip_med_7	393.79334657718400	783.3987815777450	0.01	28392.84
card_zip_total_7	710.6900768696170	4112.523138220700	0.01	306633.4100000000
card_zip_actual/avg_7	0.9874480829172540	0.4032745359609310	8.94674450334385E-05	32.76300707032800
card_zip_actual/max_7	0.8694202989280600	0.2827922880873230	4.47357237121704E-05	1.0
card_zip_actual/med_7	1.1084937362050800	2.780199678814770	8.94674450334385E-05	442.8697962798940
card_zip_actual/total_7	0.760361030973834	0.366190600175941	4.47337225167192E-05	1.0
card_zip_count_past_14	5.695446953743370	19.978257749997800	1.0	380.0

	mean	std	min	max
card_zip_avg_14	402.24136941407000	785.7146640785260	0.01	28392.84
card_zip_max_14	493.65752118842000	1055.2492897976500	0.01	47900.0
card_zip_med_14	393.19614609376800	780.9046450105180	0.01	28392.84
card_zip_total_14	805.6120173864370	4186.944348413340	0.01	306633.4100000000
card_zip_actual/avg_14	0.9849499702415510	0.476512195764123	8.94674450334385E-05	32.76300707032800
card_zip_actual/max_14	0.8306439922249370	0.3150639883248310	4.47357237121704E-05	1.0
card_zip_actual/med_14	1.1516729999738000	2.9254208462344100	8.94674450334385E-05	400.0
card_zip_actual/total_14	0.7080156965743170	0.3887695660702500	4.47337225167192E-05	1.0
card_zip_count_past_30	8.412606201437800	28.949201686401800	1.0	425.0
card_zip_avg_30	404.3553742996130	776.7897779032200	0.01	28392.84
card_zip_max_30	534.0271742896510	1086.2665033405500	0.01	47900.0
card_zip_med_30	390.51671986681	774.9985390556500	0.01	28392.84
card_zip_total_30	988.2753613701760	4344.55590525804	0.01	306633.4100000000
card_zip_actual/avg_30	0.9812813027402250	0.5677185297243220	0.0001012726597576210	32.76300707032800
card_zip_actual/max_30	0.7792611469989220	0.3481615513083910	4.47357237121704E-05	1.0
card_zip_actual/med_30	1.2319785859996600	7.913666638636010	6.75173857268247E-05	2248.7
card_zip_actual/total_30	0.6394777066767660	0.407812861541831	3.08737697855255E-05	1.0
card_state_count_past_0	2.1642685975704600	5.940302040739370	1.0	146.0
card_state_avg_0	395.41728053335800	787.3466628839980	0.01	28392.84
card_state_max_0	432.0083419608490	944.1792525289100	0.01	47900.0
card_state_med_0	392.11816156104700	781.0292374304400	0.01	28392.84

	mean	std	min	max
card_state_total_0	553.2163506125740	2639.2858184478800	0.01	217467.18
card_state_actual/avg_0	1.0005855264124900	0.2604459814129140	8.94674450334385E-05	20.24247672656420
card_state_actual/max_0	0.9415210220504240	0.1944492911087550	4.47357237121704E-05	1.0
card_state_actual/med_0	1.0456555263692200	1.4121503904175000	8.94674450334385E-05	234.7928176795580
card_state_actual/total_0	0.8618810504398870	0.2933466866636990	4.47337225167192E-05	1.0
card_state_count_past_1	2.5851945599966800	7.818864919759400	1.0	177.0
card_state_avg_1	396.9593546995090	784.7767688525020	0.01	28392.84
card_state_max_1	456.9035202340320	1029.8880605139500	0.01	47900.0
card_state_med_1	391.4680616098030	779.0753983663770	0.01	28392.84
card_state_total_1	657.0016459018480	4052.081354242080	0.01	306633.4100000000
card_state_actual/avg_1	0.9977484013140610	0.3293959681279860	8.94674450334385E-05	20.17129251217170
card_state_actual/max_1	0.9096930676837590	0.2395221950500990	4.47357237121704E-05	1.0
card_state_actual/med_1	1.072879341467640	1.6081212004174600	8.94674450334385E-05	231.5940054495910
card_state_actual/total_1	0.8128935575468900	0.3291058671503270	4.47337225167192E-05	1.0
card_state_count_past_3	3.3289521458136700	11.275749461055300	1.0	251.0
card_state_avg_3	397.6039035437520	771.2263418594580	0.01	28392.84
card_state_max_3	484.6804003236640	1055.2598794305100	0.01	47900.0
card_state_med_3	388.1186040021980	764.5810674845470	0.01	28392.84
card_state_total_3	734.9110257580670	4118.215149664430	0.01	306633.4100000000
card_state_actual/avg_3	0.9944875833841150	0.4009252657856260	8.94674450334385E-05	20.17129251217170
card_state_actual/max_3	0.8697801291958170	0.2825778435275610	4.47357237121704E-05	1.0

	mean	std	min	max
card_state_actual/med_3	1.1261896456011000	2.3451984125922600	8.94674450334385E-05	301.1031518624640
card_state_actual/total_3	0.7569100021634650	0.3615000265588710	4.47337225167192E-05	1.0
card_state_count_past_7	4.6548958992499800	16.29109168419270	1.0	369.0
card_state_avg_7	399.7250000816710	755.5824872640320	0.01	28392.84
card_state_max_7	538.4321487183210	1121.4296079225300	0.01	47900.0
card_state_med_7	381.636875577046	745.0824051951480	0.01	28392.84
card_state_total_7	896.4804810315750	4242.361692560150	0.01	306633.4100000000
card_state_actual/avg_7	0.9902881567489600	0.5150831701838620	8.94674450334385E-05	32.76300707032800
card_state_actual/max_7	0.8061351216885590	0.3321752828221120	4.47357237121704E-05	1.0
card_state_actual/med_7	1.224398945770200	3.6644222680719000	8.94674450334385E-05	442.8697962798940
card_state_actual/total_7	0.6708510573723900	0.3934094932305890	4.47337225167192E-05	1.0
card_state_count_past_14	6.429888896957370	19.955879330175400	1.0	380.0
card_state_avg_14	401.5220656037990	735.3527778762170	0.01	28392.84
card_state_max_14	602.031480543999	1189.5485006450800	0.01	47900.0
card_state_med_14	373.464406257457	726.812429068815	0.01	28392.84
card_state_total_14	1140.0852322167900	4500.491497464010	0.01	306633.4100000000
card_state_actual/avg_14	0.9897066148811940	0.6241113654143360	8.94674450334385E-05	32.76300707032800
card_state_actual/max_14	0.7441225424592460	0.365131110726686	4.47357237121704E-05	1.0
card_state_actual/med_14	1.3277479991181600	4.016721997702380	8.94674450334385E-05	416.2493333333330
card_state_actual/total_14	0.590067377096903	0.4088410906684600	4.47337225167192E-05	1.0
card_state_count_past_30	9.836685788976840	28.91499314087260	1.0	425.0

	mean	std	min	max
card_state_avg_30	402.7642068475320	694.8836945434290	0.01	28392.84
card_state_max_30	704.9740406859070	1299.3767952859600	0.01	47900.0
card_state_med_30	359.2138497567390	679.9389545197370	0.01	28392.84
card_state_total_30	1645.0983113582200	5158.971888671390	0.01	306633.4100000000
card_state_actual/avg_30	0.9879140929595150	0.7485945859722530	3.55258717160772E-05	32.76300707032800
card_state_actual/max_30	0.6643691791600630	0.3921734228406390	9.36886632033091E-06	1.0
card_state_actual/med_30	1.473264185997860	4.889567242220630	6.75173857268247E-05	452.8575310207410
card_state_actual/total_30	0.4910476246677010	0.4107971232603310	7.10517434321545E-06	1.0
merch_zip_count_past_0	6.828075562517510	18.992925801748000	1.0	260.0
merch_zip_avg_0	395.4580816466210	759.38802118802	0.01	28392.84
merch_zip_max_0	505.2906640248190	1012.0467225402	0.01	47900.0
merch_zip_med_0	380.91412113448200	749.9951350743460	0.01	28392.84
merch_zip_total_0	788.1632752056700	2853.404879260310	0.01	217467.18
merch_zip_actual/avg_0	1.0030898150028700	0.6345466718879900	8.94674450334385E-05	37.9561424763751
merch_zip_actual/max_0	0.8112758824923570	0.3412667448450450	4.47357237121704E-05	1.0
merch_zip_actual/med_0	1.26361386116652	3.521195957665380	8.94674450334385E-05	405.449591280654
merch_zip_actual/total_0	0.7174108071822860	0.3951473827423800	4.47337225167192E-05	1.0
merch_zip_count_past_1	11.612799153500600	31.618688288900200	1.0	327.0
merch_zip_avg_1	397.3722272112200	748.3649139390290	0.01	28392.84
merch_zip_max_1	591.8562406506460	1197.0842752745000	0.01	47900.0
merch_zip_med_1	371.01310222310400	737.3602506461880	0.01	28392.84

	mean	std	min	max
merch_zip_total_1	1147.479226013260	4392.427954744750	0.01	306633.4100000000
merch_zip_actual/avg_1	1.0001402400826200	0.774765158422394	8.94674450334385E-05	43.42602163523040
merch_zip_actual/max_1	0.7421620638397570	0.3825121904172780	4.47357237121704E-05	1.0
merch_zip_actual/med_1	1.3781147954077800	4.139716146052460	8.94674450334385E-05	405.449591280654
merch_zip_actual/total_1	0.6329994376281200	0.4208893134267540	4.47337225167192E-05	1.0
merch_zip_count_past_3	21.09813583410270	55.176954456702500	1.0	466.0
merch_zip_avg_3	397.2546675919780	727.5335875454030	0.01	28392.84
merch_zip_max_3	678.3699130678530	1274.530042154950	0.01	47900.0
merch_zip_med_3	361.27099956430400	716.0286878358520	0.01	28392.84
merch_zip_total_3	1626.1635699243600	5218.26128962722	0.01	307302.5800000000
merch_zip_actual/avg_3	1.0008225001133100	0.929420876255094	8.94674450334385E-05	64.09280649108990
merch_zip_actual/max_3	0.6863180320183610	0.406483268102619	4.47357237121704E-05	1.0
merch_zip_actual/med_3	1.4783957836750300	5.062506578883630	8.94674450334385E-05	467.8285714285710
merch_zip_actual/total_3	0.5682246885764270	0.4329194522645410	1.84107143507023E-05	1.0
merch_zip_count_past_7	42.329501955455000	106.13513667742700	1.0	762.0
merch_zip_avg_7	396.57382452926200	691.513103739246	0.01	28392.84
merch_zip_max_7	837.6895842193740	1505.0503946802400	0.01	47900.0
merch_zip_med_7	346.4412469267720	680.078335566715	0.01	28392.84
merch_zip_total_7	2721.5618357417300	6495.681484956400	0.01	313984.5500000000
merch_zip_actual/avg_7	0.9980258513679290	1.103173428894290	0.0001244187312399880	82.46059846404050
merch_zip_actual/max_7	0.6139082687678910	0.4244640194523250	4.47357237121704E-05	1.0

	mean	std	min	max
merch_zip_actual/med_7	1.5622689555098100	4.723198286102180	0.0001350256548744260	473.9842105263160
merch_zip_actual/total_7	0.4793831394993300	0.4339265291831050	1.80151230673805E-05	1.0
merch_zip_count_past_14	76.37049908192160	191.90961456942500	1.0	1091.0
merch_zip_avg_14	397.95170456259600	665.1963822234360	0.01	28392.84
merch_zip_max_14	1012.0587101258800	1832.4974562420900	0.01	47900.0
merch_zip_med_14	336.10888139673000	647.2592986392140	0.01	28392.84
merch_zip_total_14	4512.051020363680	9137.405923942810	0.01	319334.6800000000
merch_zip_actual/avg_14	0.9960130540392900	1.2691994090273800	0.0001238554430043790	133.5333353248190
merch_zip_actual/max_14	0.5582690158432440	0.4293929235363670	4.47357237121704E-05	1.0
merch_zip_actual/med_14	1.614947606182690	4.669742696806790	0.0001350256548744260	473.9842105263160
merch_zip_actual/total_14	0.4124893612371360	0.4250084907706850	1.62444880955689E-05	1.0
merch_zip_count_past_30	147.25618017158200	376.369243050999	1.0	1828.0
merch_zip_avg_30	397.5939574753180	626.3698875318240	0.01	28392.84
merch_zip_max_30	1274.0895096321000	2441.2478293840800	0.01	47900.0
merch_zip_med_30	322.6730097409830	595.6319799315550	0.01	28392.84
merch_zip_total_30	8317.261575982920	15001.649883715500	0.01	320373.00000000000
merch_zip_actual/avg_30	0.997357244034946	1.4492242385628700	5.05178075271533E-05	172.6358842587750
merch_zip_actual/max_30	0.4987958085770370	0.4270692835474520	1.83482871873911E-05	1.0
merch_zip_actual/med_30	1.6740533887840300	4.5152773937687600	6.75173857268247E-05	481.5882352941180
merch_zip_actual/total_30	0.3433472455194510	0.4069179833688750	6.13117025646685E-06	1.0
merch_state_count_past_0	6.876489932259300	18.990557834622500	1.0	260.0

	mean	std	min	max
merch_state_avg_0	395.4935219814540	753.267959422117	0.01	27218.0
merch_state_max_0	513.1464754089930	1034.4322246565300	0.01	47900.0
merch_state_med_0	379.5579303297870	742.4100891781930	0.01	27218.0
merch_state_total_0	812.2904954511050	2881.4207135552400	0.01	217467.18
merch_state_actual/avg_0	1.0033646177618800	0.6426009367728600	8.94674450334385E-05	37.9561424763751
merch_state_actual/max_0	0.8069571357367040	0.3440640234759550	4.47357237121704E-05	1.0
merch_state_actual/med_0	1.2771429191089600	4.008545485853940	8.94674450334385E-05	553.8194444444450
merch_state_actual/total_0	0.7117152615400380	0.3974101656654540	4.47337225167192E-05	1.0
merch_state_count_past_1	11.736942021017300	31.611768481847800	1.0	327.0
merch_state_avg_1	397.47013258826900	741.8916610578320	0.01	27218.0
merch_state_max_1	609.8790436424430	1283.5545255081800	0.01	47900.0
merch_state_med_1	368.92482400905000	729.0488210170950	0.01	27218.0
merch_state_total_1	1211.096614936150	4472.693009185230	0.01	306633.4100000000
merch_state_actual/avg_1	1.0005955883826500	0.7851770282235870	8.94674450334385E-05	43.42602163523040
merch_state_actual/max_1	0.7366911668348510	0.3850154006135250	2.99371249934843E-05	1.0
merch_state_actual/med_1	1.3964750484303	4.471090411710900	8.94674450334385E-05	405.449591280654
merch_state_actual/total_1	0.6260997439783440	0.4226517101525950	2.82986486230044E-05	1.0
merch_state_count_past_3	21.3304874633028	55.158224037187500	1.0	466.0
merch_state_avg_3	397.31171006275700	719.6469064735290	0.01	27218.0
merch_state_max_3	704.831959812046	1408.2107390708000	0.01	47900.0
merch_state_med_3	359.22144911149000	707.3985617039790	0.01	27218.0

	mean	std	min	max
merch_state_total_3	1745.2145516976500	5400.807150321140	0.01	307302.5800000000
merch_state_actual/avg_3	1.0008747224421400	0.9384789449644880	8.94674450334385E-05	64.09280649108990
merch_state_actual/max_3	0.6803456670336430	0.4083291512705670	2.99371249934843E-05	1.0
merch_state_actual/med_3	1.4898490824824400	5.025373090702200	8.94674450334385E-05	467.8285714285710
merch_state_actual/total_3	0.561087327681677	0.4339238679088580	1.84107143507023E-05	1.0
merch_state_count_past_7	42.83912362417920	106.0751821234230	1.0	762.0
merch_state_avg_7	396.4815328498180	682.2602386421580	0.01	27218.0
merch_state_max_7	880.4677548055950	1730.595748093120	0.01	47900.0
merch_state_med_7	344.1577902839310	670.5416929197980	0.01	27218.0
merch_state_total_7	2984.57075043827	7033.5148841291600	0.01	313984.5500000000
merch_state_actual/avg_7	0.997927665411093	1.1204572014994100	0.0001244187312399880	82.46059846404050
merch_state_actual/max_7	0.6075419342851770	0.4253920262019450	2.99371249934843E-05	1.0
merch_state_actual/med_7	1.564566953514130	4.720861537602800	0.0001350256548744260	473.9842105263160
merch_state_actual/total_7	0.4721529046443750	0.4336354099728370	1.59201112061663E-05	1.0
merch_state_count_past_14	77.33626565142070	191.79809428887900	1.0	1091.0
merch_state_avg_14	397.83153390033800	655.5978173165970	0.01	27218.0
merch_state_max_14	1075.6524437483200	2136.758815391500	0.01	47900.0
merch_state_med_14	333.8943384130270	637.2974853080400	0.01	27218.0
merch_state_total_14	5006.781304501180	10367.247458970300	0.01	319334.6800000000
merch_state_actual/avg_14	0.9958551670802780	1.284720162077220	0.0001238554430043790	133.5333353248190
merch_state_actual/max_14	0.5515877550085060	0.4296750069469320	2.99371249934843E-05	1.0

	mean	std	min	max
merch_state_actual/med_14	1.6194509975714900	4.660031104312040	0.0001350256548744260	473.9842105263160
merch_state_actual/total_14	0.4053464111841520	0.4237143027745910	9.04250357416852E-06	1.0
merch_state_count_past_30	149.25030861956300	376.1731995810560	1.0	1828.0
merch_state_avg_30	397.29784337945700	614.4434360475900	0.01	27218.0
merch_state_max_30	1375.047321493470	2907.4268181240500	0.01	47900.0
merch_state_med_30	320.2229599987710	583.6327414390770	0.01	27218.0
merch_state_total_30	9337.042756621300	18075.767433582400	0.01	320373.0000000000
merch_state_actual/avg_30	0.9975050168786200	1.465637311526240	5.05178075271533E-05	172.6358842587750
merch_state_actual/max_30	0.4920943882514660	0.4265982290956930	1.83482871873911E-05	1.0
merch_state_actual/med_30	1.6811651521616700	4.51900828661685	6.75173857268247E-05	481.5882352941180
merch_state_actual/total_30	0.3364901376727520	0.4046275146395960	4.07661110889401E-06	1.0
card_merch_zip_count_past_0	2.0961648184072100	5.910668229775540	1.0	145.0
card_merch_zip_avg_0	395.8020883305830	796.9553802411270	0.01	28392.84
card_merch_zip_max_0	421.26458344139300	935.8063314564060	0.01	47900.0
card_merch_zip_med_0	393.4589484112590	790.6559639950370	0.01	28392.84
card_merch_zip_total_0	528.6314148780590	2621.7525348699100	0.01	217467.18
card_merch_zip_actual/avg_0	0.9998535172917880	0.2207363392788380	8.94674450334385E-05	20.24247672656420
card_merch_zip_actual/max_0	0.9567921050776910	0.166075714085396	4.47357237121704E-05	1.0
card_merch_zip_actual/med_0	1.0266670504983200	0.6765396082655630	8.94674450334385E-05	100.0
card_merch_zip_actual/total_0	0.8847744734856920	0.2745956337274420	4.47337225167192E-05	1.0
card_merch_zip_count_past_1	2.4156145938151600	7.5936692021656000	1.0	177.0

	mean	std	min	max
card_merch_zip_avg_1	397.1751237751270	799.5189758294690	0.01	28392.84
card_merch_zip_max_1	432.4782641575 9600	1010.230384471 7500	0.01	47900.0
card_merch_zip_med_1	394.4298590723810	794.0340146644570	0.01	28392.84
card_merch_zip_total_1	598.9970792659560	4020.006638941 4200	0.01	306633.4100000000
card_merch_zip_actual/avg_1	0.9973260153024170	0.2548227410293100	8.94674450334385E-05	20.24247672656420
card_merch_zip_actual/max_1	0.9414486808838170	0.1931517519762300	4.47357237121704E-05	1.0
card_merch_zip_actual/med_1	1.0326887981182100	0.708265025794619	8.94674450334385E-05	71.11111111111110
card_merch_zip_actual/total_1	0.8591496811033860	0.2985680108743970	4.47337225167192E-05	1.0
card_merch_zip_count_past_3	3.024814050229780	10.976338454097100	1.0	248.0
card_merch_zip_avg_3	398.07990870357100	797.314003497768	0.01	28392.84
card_merch_zip_max_3	441.24808780356	1014.5270805145600	0.01	47900.0
card_merch_zip_med_3	394.6479938172380	792.1389064256750	0.01	28392.84
card_merch_zip_total_3	630.4837220038000	4062.598677760760	0.01	306633.4100000000
card_merch_zip_actual/avg_3	0.9948830176118110	0.2961820189506790	8.94674450334385E-05	20.24247672656420
card_merch_zip_actual/max_3	0.9212729713946590	0.2225395713777360	4.47357237121704E-05	1.0
card_merch_zip_actual/med_3	1.0536509243155900	1.501070063018380	8.94674450334385E-05	301.1031518624640
card_merch_zip_actual/total_3	0.8284570594665280	0.3243563108951840	4.47337225167192E-05	1.0
card_merch_zip_count_past_7	4.047823065033140	15.654380352272600	1.0	358.0
card_merch_zip_avg_7	399.80317453021600	792.7996465006560	0.01	28392.84
card_merch_zip_max_7	458.71362469786400	1022.683415637810	0.01	47900.0
card_merch_zip_med_7	394.66165503076600	788.0835911348910	0.01	28392.84

	mean	std	min	max
card_merch_zip_total_7	689.3981304397480	4103.384530312490	0.01	306633.4100000000
card_merch_zip_actual/avg_7	0.989387867711614	0.3622331698992400	8.94674450334385E-05	20.24247672656420
card_merch_zip_actual/max_7	0.8878587012342440	0.2626790735841150	4.47357237121704E-05	1.0
card_merch_zip_actual/med_7	1.0798525725618600	2.198541089355900	8.94674450334385E-05	442.8697962798940
card_merch_zip_actual/total_7	0.7806931915866960	0.3562847389009100	4.47337225167192E-05	1.0
card_merch_zip_count_past_14	5.338454516219380	19.044779181726300	1.0	369.0
card_merch_zip_avg_14	401.76615371617700	789.966194692878	0.01	28392.84
card_merch_zip_max_14	480.00501716858400	1043.470578465870	0.01	47900.0
card_merch_zip_med_14	394.6059527267530	786.120794235615	0.01	28392.84
card_merch_zip_total_14	769.9882665435680	4168.109411105910	0.01	306633.4100000000
card_merch_zip_actual/avg_14	0.9870930251620140	0.4312271382154610	8.94674450334385E-05	23.11321693279610
card_merch_zip_actual/max_14	0.8548625819983170	0.2937882350153200	4.47357237121704E-05	1.0
card_merch_zip_actual/med_14	1.1127981405534	2.286752118534170	8.94674450334385E-05	400.0
card_merch_zip_actual/total_14	0.7350831828486720	0.3792163773101350	4.47337225167192E-05	1.0
card_merch_zip_count_past_30	7.73098747886345	27.43106785422070	1.0	409.0
card_merch_zip_avg_30	403.85292942176100	784.3059375736170	0.01	28392.84
card_merch_zip_max_30	512.4081991140800	1066.0731727872700	0.01	47900.0
card_merch_zip_med_30	393.2419436808290	783.5668451009590	0.01	28392.84
card_merch_zip_total_30	923.2491665715740	4299.24215329742	0.01	306633.4100000000
card_merch_zip_actual/avg_30	0.983425218092392	0.5131029335555280	0.0001012726597576210	25.02564005165990
card_merch_zip_actual/max_30	0.8106905759171090	0.3274421773697500	4.47357237121704E-05	1.0

	mean	std	min	max
card_merch_zip_actual/med_30	1.1510865890851400	2.2842189391485800	6.75173857268247E-05	397.8609625668450
card_merch_zip_actual/total_30	0.67586418743089	0.4000768499331640	3.08737697855255E-05	1.0
card_merch_state_count_past_0	2.096984345985870	5.910722410854620	1.0	145.0
card_merch_state_avg_0	395.79548822027500	796.9244773481780	0.01	28392.84
card_merch_state_max_0	421.2988777659050	935.804824774836	0.01	47900.0
card_merch_state_med_0	393.4523657893930	790.6278397347720	0.01	28392.84
card_merch_state_total_0	528.9235544674650	2621.904713042960	0.01	217467.18
card_merch_state_actual/avg_0	0.999829471433516	0.2208700126228540	8.94674450334385E-05	20.24247672656420
card_merch_state_actual/max_0	0.956721410691394	0.1662173632839750	4.47357237121704E-05	1.0
card_merch_state_actual/med_0	1.0266486899410700	0.6765806947906870	8.94674450334385E-05	100.0
card_merch_state_actual/total_0	0.8845194299072400	0.2748174914528930	4.47337225167192E-05	1.0
card_merch_state_count_past_1	2.416776455698830	7.5937353031641600	1.0	177.0
card_merch_state_avg_1	397.1925216136950	799.4981746532920	0.01	28392.84
card_merch_state_max_1	432.584154071183	1010.288987531200	0.01	47900.0
card_merch_state_med_1	394.4320962270640	794.0080605559890	0.01	28392.84
card_merch_state_total_1	599.5676861313140	4020.3489126630600	0.01	306633.4100000000
card_merch_state_actual/avg_1	0.9972979760148010	0.2550183729079900	8.94674450334385E-05	20.24247672656420
card_merch_state_actual/max_1	0.9413394213739110	0.1933376049866120	4.47357237121704E-05	1.0
card_merch_state_actual/med_1	1.0326693565770200	0.70832958111957	8.94674450334385E-05	71.11111111111110
card_merch_state_actual/total_1	0.858816086709075	0.2988038381365190	4.47337225167192E-05	1.0
card_merch_state_count_past_3	3.0262352562839100	10.976321445016800	1.0	248.0

	mean	std	min	max
card_merch_state_avg_3	398.10771602526000	797.2880888359830	0.01	28392.84
card_merch_state_max_3	441.3971925474840	1014.5933305732900	0.01	47900.0
card_merch_state_med_3	394.6649539923470	792.1192515254100	0.01	28392.84
card_merch_state_total_3	631.2007192132560	4063.0043203338600	0.01	306633.4100000000
card_merch_state_actual/avg_3	0.9948159080901580	0.2964471856270490	8.94674450334385E-05	20.24247672656420
card_merch_state_actual/max_3	0.9210861455615250	0.2227982241173410	4.47357237121704E-05	1.0
card_merch_state_actual/med_3	1.0536006644595300	1.5011216543342000	8.94674450334385E-05	301.1031518624640
card_merch_state_actual/total_3	0.8280389366859950	0.3246113889293160	4.47337225167192E-05	1.0
card_merch_state_count_past_7	4.050240152701850	15.654349256293300	1.0	358.0
card_merch_state_avg_7	399.83499516031600	792.5273172238670	0.01	28392.84
card_merch_state_max_7	459.0051619863690	1022.8174344086100	0.01	47900.0
card_merch_state_med_7	394.67626653319800	787.8207440887350	0.01	28392.84
card_merch_state_total_7	690.5641550048280	4104.138786193850	0.01	306633.4100000000
card_merch_state_actual/avg_7	0.9893022324468690	0.3626326420557430	8.94674450334385E-05	20.24247672656420
card_merch_state_actual/max_7	0.8875337250259500	0.2630150318700560	4.47357237121704E-05	1.0
card_merch_state_actual/med_7	1.0799014951065100	2.198752093660190	8.94674450334385E-05	442.8697962798940
card_merch_state_actual/total_7	0.7800599364202690	0.356556420556419	4.47337225167192E-05	1.0
card_merch_state_count_past_14	5.34207496083903	19.04457096403430	1.0	369.0
card_merch_state_avg_14	401.9016837649370	791.3249723956890	0.01	28392.84
card_merch_state_max_14	480.6955081589670	1047.319425618860	0.01	47900.0
card_merch_state_med_14	394.702634418092	787.5157525616360	0.01	28392.84

	mean	std	min	max
card_merch_state_total_14	771.9782721454000	4170.624603431230	0.01	306633.4100000000
card_merch_state_actual/avg_14	0.9870630958992160	0.4317616591107430	8.94674450334385E-05	23.11321693279610
card_merch_state_actual/max_14	0.8544057145703620	0.2941382421001110	4.47357237121704E-05	1.0
card_merch_state_actual/med_14	1.112994001415640	2.2871056622324500	8.94674450334385E-05	400.0
card_merch_state_actual/total_14	0.7342434129310320	0.3794492228978490	4.47337225167192E-05	1.0
card_merch_state_count_past_30	7.7368279095822500	27.430299499407100	1.0	409.0
card_merch_state_avg_30	404.1101791757680	786.6247640300260	0.01	28392.84
card_merch_state_max_30	513.7273817649920	1073.36567222341	0.01	47900.0
card_merch_state_med_30	393.40563445958800	785.1463372824670	0.01	28392.84
card_merch_state_total_30	926.7034823697830	4304.260438540560	0.01	306633.4100000000
card_merch_state_actual/avg_30	0.9834500261439720	0.5140269019453740	0.0001012726597576210	25.02564005165990
card_merch_state_actual/max_30	0.8099289633196670	0.3278830034038490	4.47357237121704E-05	1.0
card_merch_state_actual/med_30	1.151422361665380	2.2852482606885000	6.75173857268247E-05	397.8609625668450
card_merch_state_actual/total_30	0.6745177276014610	0.4002386534362250	3.08737697855255E-05	1.0
Cardnum_number0/7	3.3214420545979900	2.212497514934530	0.01977401129943500	7.000000000000000
Cardnum_amt0/7	3.206817167241870	2.6142785491904600	0.0002751896843181190	7.000000000000000
Cardnum_number0/14	4.723966732057960	4.015722100015700	0.036939313984168900	14.00000000000000
Cardnum_amt0/14	4.514457177489080	4.7005362570746000	0.0001830485294338770	14.00000000000000
Cardnum_number0/30	6.54528044012433	7.123550194135620	0.07042253521126760	30.00000000000000
Cardnum_amt0/30	6.167426775231710	8.2195527681087	0.0002162432563639490	30.00000000000000
Cardnum_number1/7	3.9910106067471 00	2.1343940852080000	0.01977401129943500	7.000000000000000

	mean	std	min	max
Cardnum_amt1/7	3.9004026228225000	2.5562753753950400	0.0002751896843181190	7.000000000000000
Cardnum_number1/14	5.664782961623390	4.0745123688833300	0.03713527851458890	14.00000000000000
Cardnum_amt1/14	5.493205448123460	4.822530667765430	0.0001830485294338770	14.00000000000000
Cardnum_number1/30	7.839466552963710	7.458252792026640	0.07058823529411770	30.00000000000000
Cardnum_amt1/30	7.508501599209700	8.675311545538260	0.0002162432563639490	30.00000000000000
Merchnum_number0/7	3.999787708022970	2.6938571182310500	0.010447761194029900	7.000000000000000
Merchnum_amt0/7	3.977396809481850	2.8538439056297600	0.0001271188794396000	7.000000000000000
Merchnum_number0/14	6.7077376040393100	5.526370772497220	0.013820335636722600	14.00000000000000
Merchnum_amt0/14	6.6566539370090600	5.811223081532710	0.0001774596428438280	14.00000000000000
Merchnum_number0/30	11.694380387047900	11.636127487986600	0.016787912702853900	30.00000000000000
Merchnum_amt0/30	11.596159422405400	12.1790472593181	0.0002058881757532520	30.00000000000000
Merchnum_number1/7	4.490929396030730	2.422992666500360	0.014583333333333300	7.000000000000000
Merchnum_amt1/7	4.4727837084323300	2.59930733226613	0.0004725897920604920	7.000000000000000
Merchnum_number1/14	7.354281294668330	5.2454385190896200	0.01680672268907560	14.00000000000000
Merchnum_amt1/14	7.31363785627656	5.556322747641250	0.0009451795841209830	14.00000000000000
Merchnum_number1/30	12.584822094984200	11.391981350030600	0.017814726840855100	30.00000000000000
Merchnum_amt1/30	12.517392058406700	11.973660135055000	0.0003031068451629200	30.00000000000000
Proba DOW	0.01112096581828100	0.00729591604269723	0.006651570902915370	0.0400219298245614
Proba state	0.01033312809098190	0.00943836362530129	0.0	0.0573913043478261

Appendix C: Top 30 Variables

	variables
1	card_merch_state_total_14
2	card_merch_zip_total_7
3	card_zip_total_3
4	card_merch_total_14
5	card_state_total_3
6	card_merch_zip_total_14
7	card_merch_state_total_3
8	card_state_total_7
9	card_zip_total_1
10	card_merch_state_total_30
11	card_merch_total_1
12	card_merch_zip_total_30
13	card_merch_zip_total_1
14	card_state_max_7
15	card_zip_max_30
16	card_state_total_30
17	card_zip_max_7
18	card_state_max_30
19	card_merch_zip_max_7
20	card_merch_zip_max_3
21	card_state_max_1
22	card_zip_max_1
23	merch_zip_total_1
24	Merchnum_max_0
25	Merchnum_total_0
26	Cardnum_total_1
27	Merchnum_total_3
28	merch_zip_max_3
29	card_zip_max_0
30	Cardnum_total_0