USC Marshall
School of Business

Credit Card Transaction Data: Fraud Analytics Project 2 Report

March 2019

6:30-9:30pm Session Team 1:
Justice League Consulting Group

Team Members:
Zongyang Jiao, Chengyin Liu, Jiayi Ma,
Xinyue Niu, Xueyan Gu, Jie Zhao

USC Marshall
School of Business

Table of Contents

Executive Summary 3
Description of Data 4
Data Cleaning 9
Candidate Variables 11
Feature Selection Process 13
Model Algorithms 19
Results 23
Conclusions 26
Appendix 27

Executive Summary

As a team of six, the Justice League Consulting Group has built a supervised model to identify fraudulent events in the credit card transaction data on companies in Tenness during the year of 2010.

Began by describing the data, we also illustrated a few important distribution graphs. Then, we explained the following processes in detail: data cleaning, candidate variables, and feature selection. Next, we described the five algorithms tried in this project: Logistic Regression, Naïve Bayes, Random Forest, Boosted Trees, and Neural Nets. In the end, we presented the results and gave our conclusions, where we demonstrated our overall findings and insights.

We first took the time to understand the data and the business problem, which was determining which data records are fraudulent in the credit card transaction dataset. The approach was to find anomalies within the dataset by building supervised fraud models. After cleaning the data and filling in missing fields using values that would not cause unwanted dramatic changes in the records, we created over 300 expert variables. To ensure the proper treatment of time, we then separated the data into training, testing, and out-of-time (OOT) validation data sets. Next, we performed feature selection on the training and testing dataset using filter, wrapper, and embedded methods. During the univariate filter step, we removed about 2/3 of the variables, leaving 123 variables. Then, we reduced the number of variables to 20 using the wrapper method, with a stepwise logistic regression. On the final dataset, we used regularization while exploring a handful of nonlinear models.

In the end, the model results indicated that the boosted tree algorithm worked the best in terms of correctly identifying fraudulent credit card transactions. Using top 3% of the population with the highest predictions, the model achieved a 100% fraud detection rate on the training set, 88% on testing, and 37% on out-of-time dataset, respectively. As for the saving plots, the overall saving reached the highest point of $140,550 when targeting the top 14% of population with highest predictions. Therefore, we recommend that the client set a cutoff point at 14%.

Description of Data

This dataset contains credit card purchase records of companies in Tennessee in the year of 2010. There are 10 fields with information on the transactions and 96,753 rows/records in the dataset. Each record has a field indicating the status of fraudulent: fraud = 1, not fraud = 0.

Field Statistics Summary:

No.	Field Name	Field Type	# Records	%Populated	# Unique Values	# Records with Value NaN/" "	Other
1	Recnum	Ordinal	96,753	100.00%	96,753	0	From 1 to 96,753
2	Cardnum	Categorical	96,753	100.00%	1,645	0	5142148452: 1,192 5142184598: 921 5142189108: 663 …
3	Date	Time	96,753	100.00%	365	0	2010-02-28: 684 2010-08-10: 610 2010-03-15: 594 … [2010-01-01 to 2010-12-31]
4	Merchnum	Categorical	93,378	96.39%	13,091	3,375	930090121224: 9,310 5509006296254: 2,131 9900020006406: 1,714 …
5	Merch description	Categorical	96,753	100.00%	13,126	0	GSA-FSS-ADV: 1.688 SIGMA-ALDRICH: 1,635 STAPLES #941 : 1,174 …
6	Merch state	Categorical	96,753	98.76%	227	1,195	TN: 12,035 VA: 7,872 CA: 6,817 …
7	Merch zip	Categorical	96,753	95.19%	4,567	4,656	38118: 11,868 63103: 1,650 8701: 1,267 …
8	Transtype	Categorical	96,753	100.00%	4	0	P: 96,398 A: 181 D: 173 Y: 1

9	Amount	Numeric	96,753	100.00%	-	0	Unit: US Dollar Min: 0.01 Max: 3,102,045.53 Mean: 427.89 Std: 10,006.14
10	Fraud	Categorical	96,753	100.00%	2	0	0: 95,694 1: 1,059

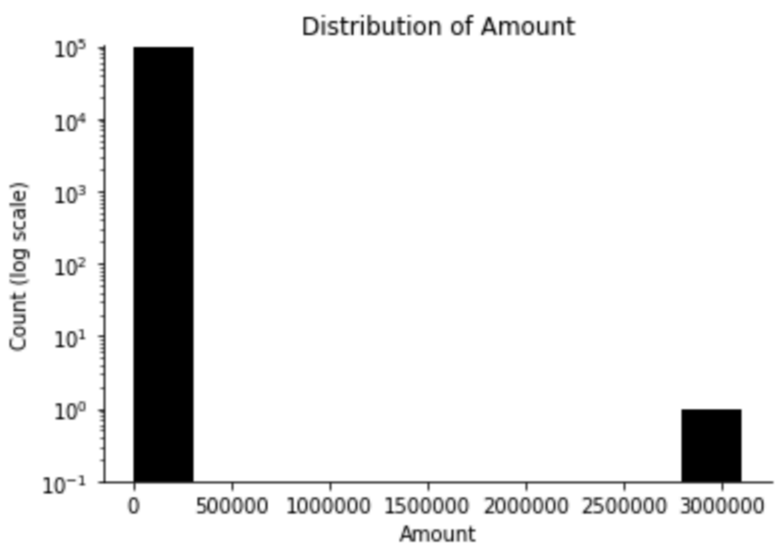
Important Distributions (amount, fraud):

a. Amount

Amount stands for the amount a customer spent in such record.

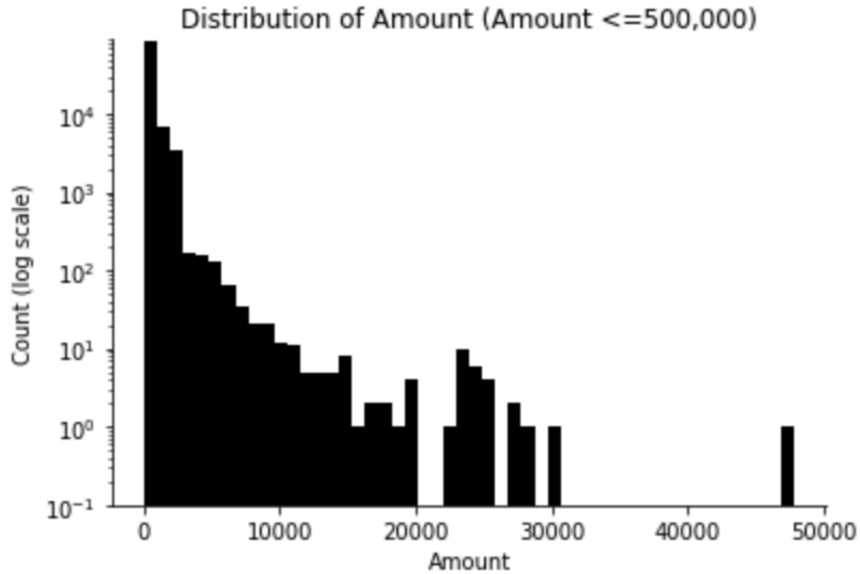
- **All Records**

Unit	Max	Min	Mean	Std
US Dollar	3,102,045.53	0.01	427.89	10,006.14



As we can see from above, there are outliers in the distribution of the variable amount. Therefore, we plotted the data again without the outliers. (Amount <= 500,000)

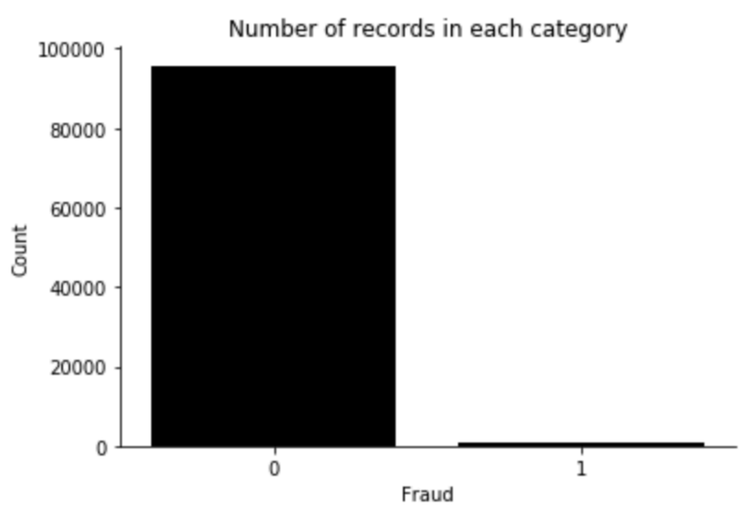
This time, the distribution makes more sense as below:



b. Fraud

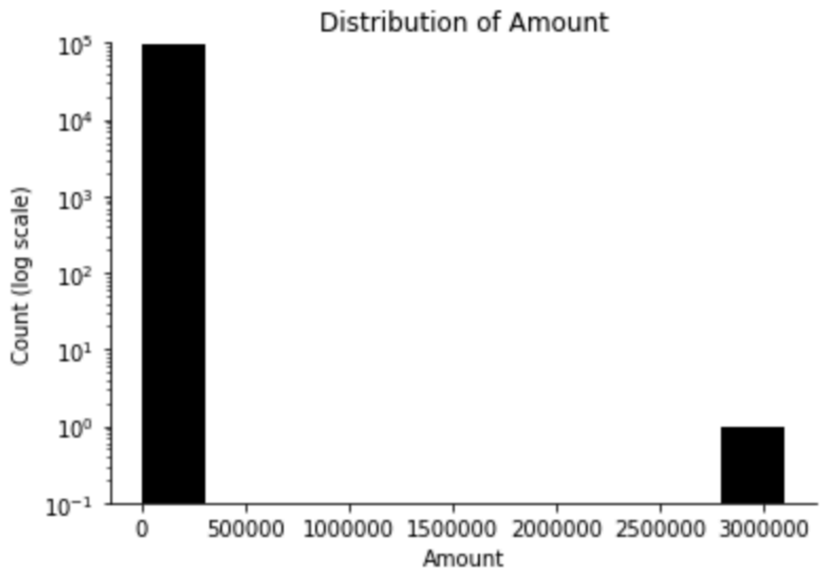
Fraud is the label for each record, indicating such record is fraudulent or not.

Merch state	Count
0	95,694
1	1,059

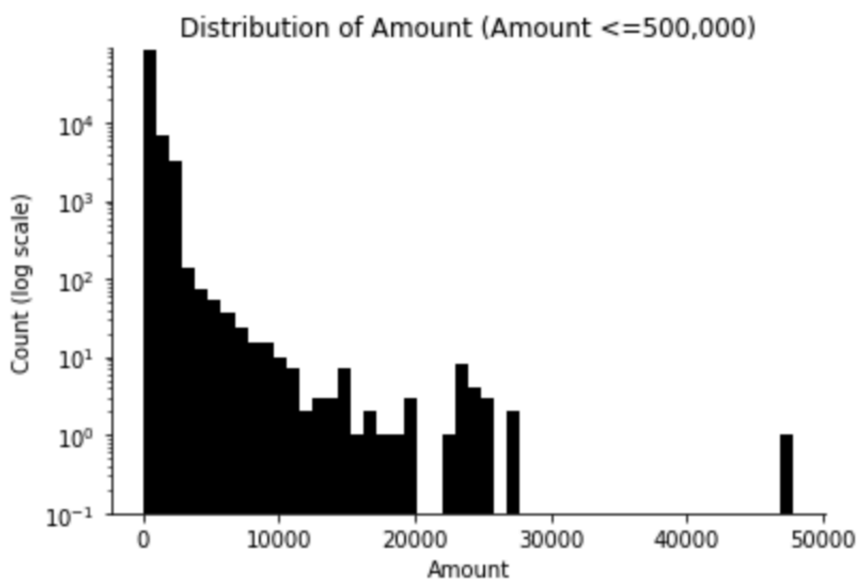


Non-Fraud Records

Unit	Max	Min	Mean	Std
US Dollar	3,102,045.53	0.01	409.34	10,054.62

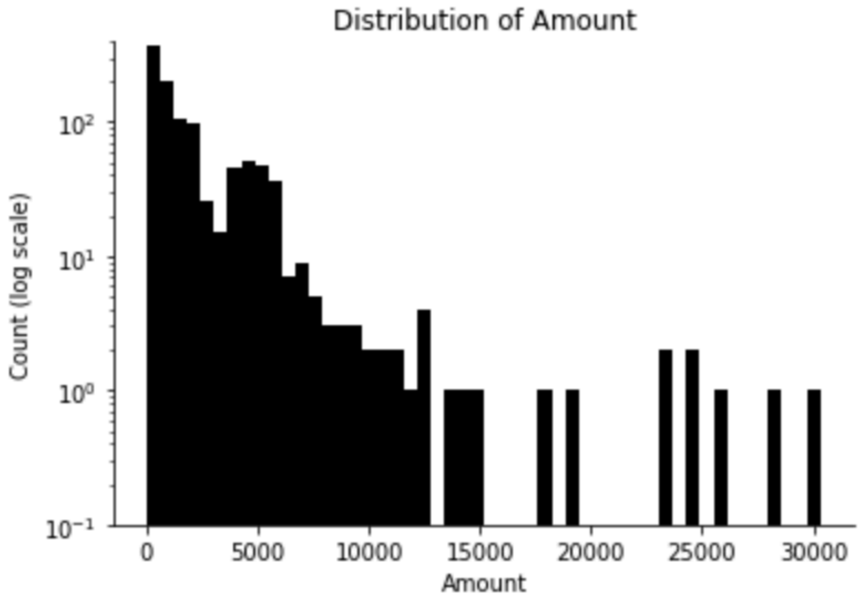


Non-Fraud records without the outliers. (Amount <= 500,000)



Fraud Records

Unit	Max	Min	Mean	Std
US Dollar	30,372.46	0.22	2,103.35	3,068.53



Data Cleaning

To achieve better results, we used R to clean and manipulate the data before proceeding to next steps in Python. Before filling missing values, we made three adjustments to the dataset:

- Only included the values with [Transtype] P, or present
- Excluded the outlier record with the maximum transaction amount dramatically higher than the rest of records
- Changed all invalid state abbreviations to 'others'

The fields that required cleaning and filling missing values include [Merch state], [Merch zip] and [Merchnum]. The general idea of filling missing fields is to use summary statistics of similar groups that the records identify with, while minimizing the likelihood of causing any unwanted abnormal results. The following table summarizes the process of filling missing value:

	1st group	***# NA***	***2nd group***	***# NA***	***3rd group***	***# NA***	***Fill***
Merch state	Merchnum	93	Merch description	90	Merch zip	48	'TN'
Merch zip	Merchnum	1089	Merch description	1048	Merch state	0	
Merchnum	Merch description	2038	Merch zip	161	Merch state	0	

Merch state

Description: there were 1020 NA values in Merch state field.

Method:

- We filled the null state values based on the mode of other records with the same [Merchnum], [Merch description] and [Merch zip].
- After three rounds of filling, some records were still NA. So, we filled these remaining NAs using 'TN' since it is the most common value in the field.

Merch zip

Description: there are 4300 NA values in Merch zip field.

Method:

- We filled the null zip values based on the mode of other records with the same [Merchnum], [Merch description] and [Merch state].
- There was no missing value after the filling process.

Merchnum

Description: there were 3198 NA values in Merchnum field.

Method:

- We filled the null zip values using the mode of other records with the same [Merch description], [Merch zip] and [Merch state].
- There was no missing value after the filling process.

Candidate Variables

After cleaning the data, we then created variables for further analysis. There are four types of variables, and the steps of variable creation are listed as the following:

a. Amount variables

First, we grouped all the records respectively by Cardnum, Merchnum, [Cardnum & Merchnum], [Cardnum & Merch zip], and [Card & Merch state].

For each group g, we calculated the average, maximum, median, and the total amount over the past 0, 1, 3, 7, 14, and 30 days respectively.

Based on the results above, for each group g, we then calculated the [actual/average], [actual/maximum], [actual/median], [actual/total] amount over the past 0, 1, 3, 7, 14, and 30 days respectively.

In this step, we created 5 x 8 x 6 = 240 amount variables.

b. Frequency variables

Same as above, we grouped all the records respectively by Cardnum, Merchnum, [Cardnum & Merchnum], [Cardnum & Merch zip], [Card & Merch state].

For each group g, we counted the number of transactions over the past 0, 1, 3, 7, 14, and 30 days respectively.

In this step, we created 5 x 6 = 30 frequency variables.

c. Days since variables

Same as above, we again grouped all the records respectively by Cardnum, Merchnum, [Cardnum & Merchnum], [Cardnum & Merch zip], [Card & Merch state].

For each group g, we also calculated the current date minus date of most recent transaction within that group.

In this step, we created 5 days since variables.

d. Velocity change variables

To create the velocity change variables, we first grouped all the records respectively by Cardnum, and Merchnum.

For each group g, we then calculated the [number] and [amount] of transactions over the past 0 and 1 day respectively.

For each group g, we also calculated the average daily [number] and [amount] of transactions over the past 7,14, and 30 days respectively.

Then, we calculated the velocity change variables using the formula below:

$$\frac{\begin{matrix}\{number|amount\}\ of\ transactions\ with\ same\ group\{cardnum|merchnum\}\\ over\ the\ past\ \{0|1\}\ day\end{matrix}}{\begin{matrix}average\ daily\ \{number|amount\}\ of\ transactions\ with\ same\ group\ \{cardnum|merchnum\}\\ over\ the\ past\ \{7|14|30\}\ days\end{matrix}}$$

In this step, we created 2 x 2 x 2 x 2 x 2 x 3 = 96 velocity change variables.

e. Brief summary

Overall, we created 240 + 30 + 5 + 96 = 371 variables in total.

We have attached the full list of 371 variables in the Appendix.

Feature Selection Process

Before moving on to the feature selection process, we separated the data into training, testing, and out-of-time (OOT) validation data sets to ensure the proper treatment of time. We excluded the last two months of data as OOT data, and we randomly divided the rest into 70% training and 30% testing.

Next, with the newly created 371 variables, we performed feature selection on the training and testing dataset using filter, wrapper, and embedded methods. During the univariate filter step, we removed about 2/3 of the variables, leaving 123 variables. Then, we reduced the number of variables to 20 using the wrapper method, with a stepwise logistic regression. On the final dataset, we used regularization while exploring a handful of nonlinear models. More details are given below.

Filter

After we built 371 variables, we calculated the KS and Fraud Detection Rate at 3% for each variable. Then we ranked them in descending order and selected the top 123 for further process. Variables and their rank of KS, FDR, as well as average rank are shown in the table below.

	Variables	KS_RANK	FDR_RANK	AVG_RANK
1	Fraud	1	1	1
2	Amount_sum_card_merchant_7	2	2	2
3	Amount_sum_card_merchant_14	3	3	3
4	Amount_sum_card_state_7	6	4	5
5	Amount_sum_card_zip_7	7	5	6
6	Amount_sum_card_state_3	8	7.5	7.75
7	Amount_sum_card_merchant_3	10	6	8
8	Amount_sum_card_zip_3	9	7.5	8.25
9	Amount_sum_card_state_14	4	15.5	9.75
10	Amount_sum_card_zip_14	5	15.5	10.25
11	Amount_sum_card_merchant_30	11	11	11
12	Amount_max_card_state_7	14	20.5	17.25
13	Amount_max_card_zip_7	15	20.5	17.75
14	Amount_sum_card_zip_1	26.5	9.5	18
15	Amount_sum_card_state_1	26.5	9.5	18
16	Amount_sum_card_merchant_1	25	12	18.5

17	Amount_max_card_merchant_14	12	28	20
18	Amount_max_card_merchant_7	13	31	22
19	Amount_max_card_state_14	22	22.5	22.25
20	Amount_max_card_merchant_30	16	29	22.5
21	Amount_sum_card_7	28	17	22.5
22	Amount_max_card_zip_14	23	22.5	22.75
23	Amount_max_card_state_3	17	32.5	24.75
24	Amount_max_card_zip_3	18	32.5	25.25
25	Amount_max_card_merchant_3	21	30	25.5
26	Amount_sum_card_state_30	19	34.5	26.75
27	Amount_sum_card_zip_30	20	34.5	27.25
28	Amount_sum_card_3	42	13	27.5
29	Amount_max_card_state_30	31	26.5	28.75
30	Amount_max_card_zip_30	32	26.5	29.25
31	Amount_sum_merchant_1	49	18.5	33.75
32	Amount_sum_merchant_3	24	45	34.5
33	Amount_max_card_zip_1	29.5	40.5	35
34	Amount_max_card_state_1	29.5	40.5	35
35	Amount_max_merchant_1	33	39	36
36	Amount_sum_card_1	61	14	37.5
37	Amount_max_card_merchant_1	34	43	38.5
38	Amount_max_merchant_3	43	37	40
39	Amount_max_card_1	46	38	42
40	Amount_max_card_3	55	36	45.5
41	Amount_sum_card_14	70	24.5	47.25
42	Amount_avg_card_3	50	55	52.5
43	Amount_avg_card_1	56	59	57.5
44	Amount_avg_card_zip_1	53.5	65.5	59.5
45	Amount_avg_card_state_1	53.5	65.5	59.5
46	Amount_avg_card_merchant_1	51	68	59.5
47	Amount_avg_card_zip_30	62	61	61.5
48	Amount_max_card_7	99	24.5	61.75
49	NC0_ACC7	77	50	63.5
50	NM0_ACC7	77	50	63.5
51	Amount_avg_card_7	77	50	63.5
52	Amount_avg_card_state_30	63	64	63.5
53	Amount_max_card_14	110	18.5	64.25
54	Amount_avg_card_zip_7	37	94	65.5

55	Amount_median_card_3	71	61	66
56	Amount_avg_card_state_7	38	94	66
57	Amount_avg_card_state_14	47	91	69
58	Amount_avg_card_state_3	40	98.5	69.25
59	Amount_avg_card_zip_14	48	91	69.5
60	Amount_avg_card_zip_3	41	98.5	69.75
61	Amount_avg_merchant_1	45	96	70.5
62	Amount_avg_card_merchant_30	35	110.5	72.75
63	Amount_avg_card_merchant_3	44	102	73
64	Amount_avg_card_merchant_14	36	112.5	74.25
65	Amount_avg_card_merchant_7	39	112.5	75.75
66	Amount_avg_merchant_3	59	94	76.5
67	NC0_ACC14	106.5	47	76.75
68	NM0_ACC14	106.5	47	76.75
69	Amount_avg_card_14	108	47	77.5
70	Amount_max_card_30	118	44	81
71	Amount_sum_card_30	121	42	81.5
72	Amount_median_card_state_3	58	105.5	81.75
73	NC0_ACC30	112	53	82.5
74	Amount_avg_card_30	112	53	82.5
75	NM0_ACC30	112	53	82.5
76	Amount_median_card_zip_3	60	105.5	82.75
77	Amount_sum_merchant_7	109	57	83
78	Amount_median_card_1	69	97	83
79	Amount_median_card_zip_1	65.5	100.5	83
80	Amount_median_card_state_1	65.5	100.5	83
81	Amount_median_card_merchant_1	64	103	83.5
82	Amount_sum_card_zip_0	88.5	79	83.75
83	Amount_median_merchant_0	88.5	79	83.75
84	Amount_max_merchant_0	88.5	79	83.75
85	Amount_max_card_zip_0	88.5	79	83.75
86	Amount_median_card_merchant_0	88.5	79	83.75
87	Amount_sum_card_merchant_0	88.5	79	83.75
88	Amount_median_card_state_0	88.5	79	83.75
89	Amount_median_card_zip_0	88.5	79	83.75
90	Amount_sum_card_state_0	88.5	79	83.75
91	Amount_sum_card_0	88.5	79	83.75
92	Amount_median_card_0	88.5	79	83.75

93	Amount_max_card_merchant_0	88.5	79	83.75
94	Amount_max_card_state_0	88.5	79	83.75
95	Amount_avg_card_0	88.5	79	83.75
96	Amount_avg_card_merchant_0	88.5	79	83.75
97	Amount_avg_card_state_0	88.5	79	83.75
98	Amount_avg_card_zip_0	88.5	79	83.75
99	Amount_sum_merchant_0	88.5	79	83.75
100	Amount_max_card_0	88.5	79	83.75
101	Amount_avg_merchant_0	88.5	79	83.75
102	Amount_median_card_merchant_3	57	114	85.5
103	Amount_max_merchant_7	114	58	86
104	Amount_median_card_merchant_30	52	126.5	89.25
105	Amount_median_card_zip_30	74	105.5	89.75
106	Amount_median_card_state_30	75	105.5	90.25
107	AC1_ANC30	127	56	91.5
108	Amount_median_card_14	122	61	91.5
109	Amount_median_card_merchant_14	67	123.5	95.25
110	Amount_median_card_merchant_7	68	123.5	95.75
111	Amount_median_card_zip_7	72	120	96
112	Amount_median_card_state_7	73	120	96.5
113	AC1_ACC30	134	63	98.5
114	Amount_median_card_7	120	79	99.5
115	AC1_ANC14	133	67	100
116	Amount_median_card_zip_14	104	108.5	106.25
117	Amount_median_card_state_14	105	108.5	106.75
118	Amount_median_merchant_1	100	115	107.5
119	NC0_AAM7	102	120	111
120	NM0_AAM7	102	120	111
121	Amount_avg_merchant_7	102	120	111
122	AC1_ANC7	135	91	113
123	Amount_median_merchant_3	119	116	117.5

Wrapper

From the table above, we chose 123 variables to perform our next step of feature selection (except for the Fraud variable, which is the target variable for our models. We included it to examine whether the KS and FDR at 3% are calculated in the right way because it should have the highest rank among all the variables). In this step, we used the backward stepwise selection to screen out 20 variables as the input of our models, which are listed as the table below.

Variables	Rank
AC1_ACC30	1
Amount_avg_card_merchant_7	2
Amount_median_card_merchant_7	3
Amount_avg_card_14	4
Amount_avg_card_30	5
Amount_median_card_30	6
Amount_max_card_state_14	7
Amount_avg_card_state_14	8
Amount_max_card_zip_30	9
Amount_median_card_zip_30	10
Amount_avg_card_1	11
Amount_median_card_merchant_1	12
Amount_max_merchant_7	13
Amount_max_card_merchant_1	14
Amount_avg_merchant_7	15
Amount_median_card_merchant_30	16
Amount_median_card_merchant_14	17
Amount_sum_card_merchant_7	18
Amount_avg_merchant_1	19
Amount_sum_card_3	20

Embedded

The last step in the feature selection is the embedded method, which includes 1) decision trees and 2) regularization.

Since using all features to build a decision tree model is very likely to overfit the testing data, we usually select a certain group of variables with the highest explanatory power. In this project, though we did not directly use decision tree, we did use random forest, which took effect in the final model performance.

On the other hand, regularization is adding a penalty term to the loss function while building a very complex model or a model with too many parameters. In the case of linear regression, this is interpreted as Ridge or Lasso regression. In the case of tree algorithms, the penalty term might be a parameter multiplied by the number of nodes in the tree to reduce the complexity of the model. We used regularization in some of the models we tried.

Model Algorithms

After we reduced variables to the final 20, we tried a handful of nonlinear models to look for the best model with the optimal performance. The models are Logistic Regression, Naïve Bayes, Random Forest, Boosted Trees, and Neural Nets. In this next section, we described each model in more details.

Model 1: Logistic Regression

Logistic Regression can model a binary dependent variable. When doing Logistic Regression, we want to know that given X = [x1, x2, …, xn]T, what is the probability of Y=1 happens. This method can be used in situations like predicting the probability of a sunny or cloudy weather tomorrow, given today's weather and today's temperature. Here, Xs are the weather and the temperature today, Y is the weather tomorrow, and 1 means sunny while 0 means not sunny (or cloudy). And the way of calculating such probability is by using the formula below.

$$P(Y = 1|X = x) = \frac{e^{x'\beta}}{1 + e^{x'\beta}}$$

In the formula above, x is a vector with n rows and one column, β is the coefficient vector with n rows and one column, x prim means the transposition of vector x. To estimate β in the Logistic Regression, we use Maximum Likelihood Estimation. After estimating β, when we have x and want to know the likelihood of Y=1 happening, we can plunge x in the formula above and get the probability of Y=1.

Parameters:

- **C**: Inverse of regularization strength; must be a positive float. Smaller values specify stronger regularization.
- **random_state**: The seed of the pseudo random number generator to use when shuffling the data.

We selected the best logistic regression model with C equal to 0.5, adding a regularization effect to the model.

Model 2: Naïve Bayes

Naïve Bayes is a classifier of machine learning based on Bayes' theorem assuming naive independence between all variables. Unlike other machine learning models that try to predict Y given X, Naïve Bayes predicts, given Y, how likely the records display features of X. After building the model, we would be able to use Bayes theorem to calculate, given new X, the probability of Y being any class and choose the most likely predicted result. The following steps show how the model works.

How it works:

1. Bayes' theorem: the probability of class variable y given all dependent feature variables x_1 through x_n. (y can be multiple classes)

$$P(y|x_1, \ldots, x_n) = \frac{\mathrm{P}(\mathrm{y})\mathrm{P}(x_1, \ldots, x_n|y)}{\mathrm{P}(x_1, \ldots, x_n)}$$

2. Naïve assumption: every variable is independent from each other. Therefore, the probability becomes this:

$$P\left(x_i \middle| y, x_1, \ldots, x_{i,} x_{i+1,\ldots,} x_n\right) = \mathrm{P}(x_i|y)$$

3. Put such assumption into Bayes' theorem, for all i, the Bayes' theorem becomes:

$$P(y|x_1, \ldots, x_n) = \frac{\mathrm{P}(\mathrm{y}) \prod_{i=1}^{n} \mathrm{P}(x_i|y)}{\mathrm{P}(x_1, \ldots, x_n)}$$

4. Since $\mathrm{P}(x_1, \ldots, x_n)$ is constant, we should mainly focus the numerator part $\mathrm{P}(\mathrm{y}) \prod_{i=1}^{n} \mathrm{P}(x_i|y)$.
   Then we have such classification rule referred as Maximum A Posteriori Estimation (MAP):

$$P(y|x_1, \ldots, x_n) \propto \mathrm{P}(\mathrm{y}) \prod_{i=1}^{n} \mathrm{P}(x_i|y)$$

$$\Downarrow$$

$$\hat{y} \propto \arg\max_{y} \mathrm{P}(\mathrm{y}) \prod_{i=1}^{n} \mathrm{P}(x_i|y)$$

In the formula above, xs are all variables except class variable y (here we assume y is binary variable). To estimate $\mathrm{P}(\mathrm{y} = 1)$ and $\mathrm{P}(x_i|y = 1)$, we use Maximum A Posteriori Estimation. After estimating them, when we have xs and want to know the likelihood of y = 1 class happening, we can plunge x in the formula above and get the probability of Y=1.

Model 3: Random Forest

Random Forest is a bagging technique for both classification and regression based on a decision tree. It solves Decision Tree's problem of finding the right tree depth, as it reduces the variance by averaging multiple deep decision trees trained on different parts of the same training set. This comes at the expense of a small increase in the bias and some loss of interpretability, but Random Forest greatly boosts the performance in the final model generally.

Parameters:

- **n_estimators**: The number of trees in the forest. The more estimators usually mean a better performance. 500 or 1000 is usually sufficient.
- **max_features**: The number of features to consider when looking for the best split.
- **max_depth**: The maximum depth of the tree. Reduction of the maximum depth helps fight with overfitting. If nothing is given to this parameter, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

For the random forest model, we selected 500 estimators and had a maximum depth of 20. We took square root of the number of features as the maximum number of features.

Model 4: Boosted Trees

Boosted Decision Tree is a machine learning algorithm that produces a prediction model in the form of an ensemble of weak classifiers which are decision trees in this case. Given dataset (X(1) ,y(1)), … ,(X(n), y(n)):

- Initially assign every point equal weight;
- Repeat for t = 1, 2, …:
    - Feed weighted dataset to the decision tree and get a weak classifier dt
    - Reweight the data to put more emphasis on points that dt gets wrong
- Combine all the dt linearly

Parameters:

- **learning rate**: shrinks the contribution of each tree
- **number of estimators**: the number of boosting stages to perform
- **criterion**: the function to measure quality of a split

For the boosted tree, we used a learning rate of 0.1 and the number of estimators of 1500.

Model 5: Neural Nets

Neural Net is a mathematical function mapping inputs to an output with a set of adjustable parameters. A typical neural net consists of an input layer, number of hidden layers and an output layer. An input layer has all the independent variables. An output layer refers to the dependent variable. The hidden layer contains a set of nodes. Each node in each hidden layer contains a linear combination of all the nodes in the previous layer and does a transform on this linear combination. The transform function can be a logistic function, a step function, a linear function, etc.

Parameters:

- **number of inputs**: independent variables in the dataset
- **number of hidden layers**: it depends on different situations
- **number of nodes in each hidden layer**: it depends on different situations
- **transform function**: a logistic function, a step function, a linear function, etc.
- **learning rate**: a hyper-parameter that controls how much we are adjusting the weights of our model with respect the loss gradient.

In total, we tried five neural networks and it turned out that the model with three hidden layers and 128 neurons within each had the best performance. The first two hidden layers used 'tanh' as activation function and the third used 'relu'.

High-Level Performance for Each Model:

	FDR @ 3%		
	Training set	**Testing set**	**Out of time**
Logistic Regression	0.36	0.35	0.20
Naive Bayes	0.53	0.51	0.26
Random Forest	0.98	0.83	0.40
Boosted Tree	1.0	0.88	0.37
Neural Network	0.57	0.54	0.43

Results

Final Model Selection:

According to the above table demonstrating a high-level overview of all five models' FDRs at 3% in training, testing and out-of-time dataset, both random forest and boosted tree performed well on the training set; boosted tree did better in the testing set while random forest did better in out-of-time set. In the end, we chose the boosted tree as our final model. Using top 3% of the population with highest predictions, the boosted tree model achieved a 100% fraud detection rate on the training set, 88% on testing, and 37% on out-of-time dataset, respectively.

We also generated three tables that showcase the final model performance in training, testing and out of time datasets separately. In each of them, we collected their bin statistics and cumulative statistics according to the population bin.

Training:

	# Records			# Goods			# Bads			Fraud Rate		
	58779			58170			609			1.04%		
Training	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	FPR
1	588	120	468	20	80	588	120	468	0.2	79.3	0.79	0.26
2	588	506	82	86	14	1176	626	550	1.1	93.2	0.92	1.14
3	588	561	27	95	5	1764	1187	577	2	97.8	0.96	2.06
4	588	586	2	100	0	2352	1773	579	3	98.1	0.95	3.06
5	587	584	3	99	1	2939	2357	582	4.1	98.6	0.95	4.05
6	588	587	1	100	0	3527	2944	583	5.1	98.8	0.94	5.05
7	588	586	2	100	0	4115	3530	585	6.1	99.2	0.93	6.03
8	588	586	2	100	0	4703	4116	587	7.1	99.5	0.92	7.01
9	588	588	0	100	0	5291	4704	587	8.1	99.5	0.91	8.01
10	587	586	1	100	0	5878	5290	588	9.1	99.7	0.91	9
11	588	587	1	100	0	6466	5877	589	10.1	99.8	0.9	9.98
12	588	588	0	100	0	7054	6465	589	11.1	99.8	0.89	10.98
13	588	587	1	100	0	7642	7052	590	12.1	100	0.88	11.95
14	587	587	0	100	0	8229	7639	590	13.1	100	0.87	12.95
15	588	588	0	100	0	8817	8227	590	14.1	100	0.86	13.94
16	588	588	0	100	0	9405	8815	590	15.1	100	0.85	14.94
17	588	588	0	100	0	9993	9403	590	16.2	100	0.84	15.94
18	588	588	0	100	0	10581	9991	590	17.2	100	0.83	16.93
19	587	587	0	100	0	11168	10578	590	18.2	100	0.82	17.93
20	588	588	0	100	0	11756	11166	590	19.2	100	0.81	18.93

Testing:

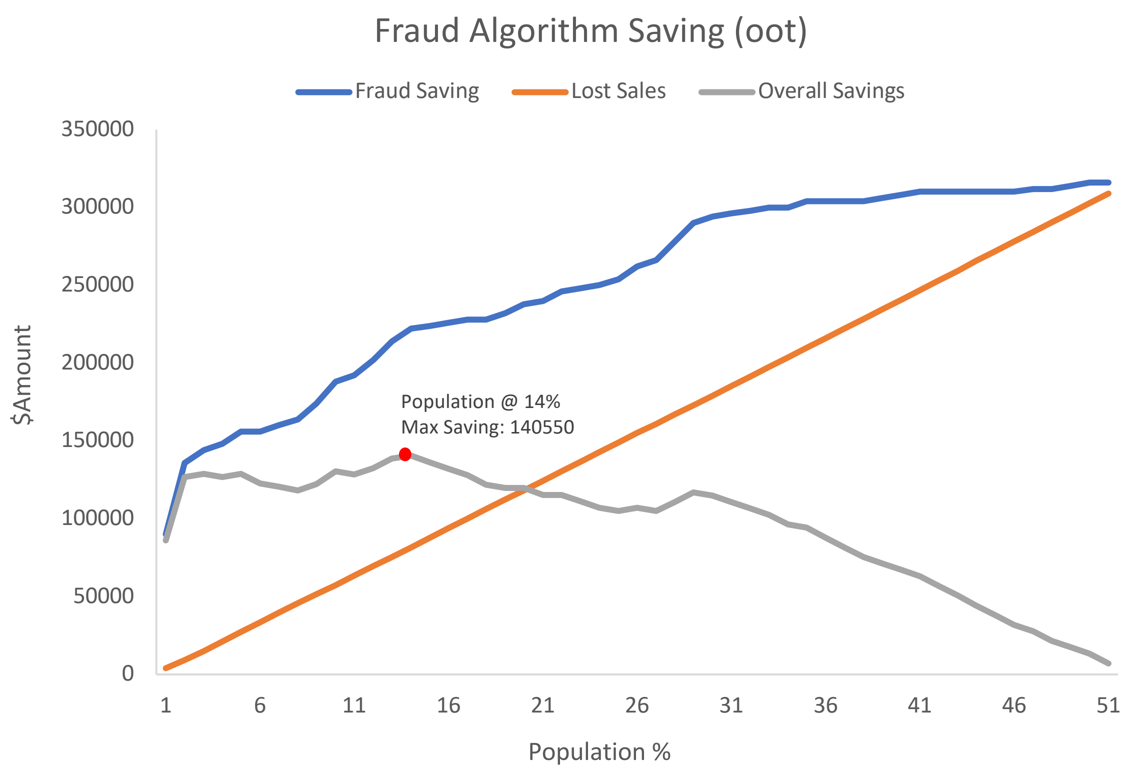
Testing	# Records			# Goods			# Bads			Fraud Rate		
	25191			24920			271			1.08%		
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	FPR
1	252	67	185	27	73	252	67	185	0.3	63.8	0.64	0.36
2	252	215	37	85	15	504	282	222	1.1	76.6	0.75	1.27
3	252	232	20	92	8	756	514	242	2.1	83.4	0.81	2.12
4	252	246	6	98	2	1008	760	248	3.1	85.5	0.82	3.06
5	252	244	8	97	3	1260	1004	256	4	88.3	0.84	3.92
6	252	244	8	97	3	1512	1248	264	5	91	0.86	4.73
7	252	247	5	98	2	1764	1495	269	6	92.8	0.87	5.56
8	252	251	1	100	0	2016	1746	270	7	93.1	0.86	6.47
9	252	250	2	99	1	2268	1996	272	8	93.8	0.86	7.34
10	252	251	1	100	0	2520	2247	273	9	94.1	0.85	8.23
11	251	248	3	99	1	2771	2495	276	10	95.2	0.85	9.04
12	252	251	1	100	0	3023	2746	277	11	95.5	0.84	9.91
13	252	252	0	100	0	3275	2998	277	12	95.5	0.83	10.82
14	252	252	0	100	0	3527	3250	277	13.1	95.5	0.82	11.73
15	252	252	0	100	0	3779	3502	277	14.1	95.5	0.81	12.64
16	252	252	0	100	0	4031	3754	277	15.1	95.5	0.8	13.55
17	252	252	0	100	0	4283	4006	277	16.1	95.5	0.79	14.46
18	252	251	1	100	0	4535	4257	278	17.1	95.9	0.79	15.31
19	252	251	1	100	0	4787	4508	279	18.1	96.2	0.78	16.16
20	252	251	1	100	0	5039	4759	280	19.1	96.6	0.77	17

Out of Time:

Out of Time	# Records			# Goods			# Bads			Fraud Rate		
	12426			12247			179			1.44%		
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	FPR
1	125	80	45	64	36	125	80	45	0.7	25.1	0.24	1.78
2	124	101	23	81	19	249	181	68	1.5	38	0.37	2.66
3	124	120	4	97	3	373	301	72	2.5	40.2	0.38	4.18
4	125	123	2	98	2	498	424	74	3.5	41.3	0.38	5.73
5	124	120	4	97	3	622	544	78	4.4	43.6	0.39	6.97
6	124	124	0	100	0	746	668	78	5.5	43.6	0.38	8.56
7	124	122	2	98	2	870	790	80	6.5	44.7	0.38	9.88
8	125	123	2	98	2	995	913	82	7.5	45.8	0.38	11.13
9	124	119	5	96	4	1119	1032	87	8.4	48.6	0.4	11.86
10	124	117	7	94	6	1243	1149	94	9.4	52.5	0.43	12.22
11	124	122	2	98	2	1367	1271	96	10.4	53.6	0.43	13.24
12	125	120	5	96	4	1492	1391	101	11.4	56.4	0.45	13.77
13	124	118	6	95	5	1616	1509	107	12.3	59.8	0.47	14.1
14	124	120	4	97	3	1740	1629	111	13.3	62	0.49	14.68
15	124	123	1	99	1	1864	1752	112	14.3	62.6	0.48	15.64
16	125	124	1	99	1	1989	1876	113	15.3	63.1	0.48	16.6
17	124	123	1	99	1	2113	1999	114	16.3	63.7	0.47	17.54
18	124	124	0	100	0	2237	2123	114	17.3	63.7	0.46	18.62
19	124	122	2	98	2	2361	2245	116	18.3	64.8	0.46	19.35
20	125	122	3	98	2	2486	2367	119	19.3	66.5	0.47	19.89

Fraud Savings Calculation:

We created the graph below to show our fraud algorithm savings. We assumed that we could gain $2000 for every true fraud we caught (blue curve) and lose $50 for every inaccurately identified fraud (red curve). Then, the overall savings (grey curve) is equal to fraud savings minus lost sales. Since we would like to save as much money as possible, as demonstrated below, the overall saving reached the highest point of $140,550 when targeting the top 14% of population with highest predictions. Therefore, we recommend that the client set a cutoff point at 14%.



Conclusions

Overall, our goal was to build a supervised model to identify fraudulent events in the credit card transaction data during the year of 2010. First, we summarized the data and also cleaned the data by excluding outliers and filling in missing fields. Then, we created more than 300 variables for model-building and divided the entire dataset into training, testing and out-of-time data. Next, we performed consecutive feature selection methods, including filter (KS, FDR), wrapper and embedded, on the training and test dataset to reduce the correlation among variables. For the final dataset, we narrowed it down to 20 variables.

In terms of the model algorithms, we decided to perform five models, including Logistic Regression, Naïve Bayes, Random Forest, Boosted Trees, and Neural Nets. For each model, we briefly summarized the basic architecture, consecutive steps and important parameters. Also, after building the models, we calculated FDRs at 3% population in training, testing and out-of-time datasets separately. We found that both random forest and boosted tree performed well on the training set, while boosted tree had better performance in testing set and random forest did better in out-of-time set.

According to the results, we decided to choose boosted tree as our final model algorithm. Then, we generated three tables to show the boosted tree model performance in training, testing and out-of-time datasets respectively. For each table, we returned the bin statistics and cumulative statistics based on population bin. Besides, we made the fraud savings plot to show the tendency of our fraud algorithm savings. From the graph, we found that at 14% population, the overall saving reaches the highest point $140,550. Therefore, we recommend that the client should set a score cutoff at 14%.

We also raised some further considerations that could be addressed in the project if we had more time. First, when conducting feature selection process, we can try more methods to select the variables in a more convincing way. Besides, for the model algorithm, we can find more relevant data and train the existing models with those data in order to improve our model efficiency and accuracy. In addition, except for existing models, we can perform more model machine learning algorithms to compare and select the best performing model. Finally, we can consult industry experts to gain more knowledge about credit card policies and how to detect fraud transactions, better identifying fraudulent events.

Appendix 1

All the variables that we created are listed below:

Amount variables			
No.	Variable	No.	Variable
1	amount_avg_card_0	121	actual_avg_card_merchant_0
2	amount_avg_card_1	122	actual_avg_card_merchant_1
3	amount_avg_card_3	123	actual_avg_card_merchant_3
4	amount_avg_card_7	124	actual_avg_card_merchant_7
5	amount_avg_card_14	125	actual_avg_card_merchant_14
6	amount_avg_card_30	126	actual_avg_card_merchant_30
7	amount_max_card_0	127	actual_max_card_merchant_0
8	amount_max_card_1	128	actual_max_card_merchant_1
9	amount_max_card_3	129	actual_max_card_merchant_3
10	amount_max_card_7	130	actual_max_card_merchant_7
11	amount_max_card_14	131	actual_max_card_merchant_14
12	amount_max_card_30	132	actual_max_card_merchant_30
13	amount_median_card_0	133	actual_med_card_merchant_0
14	amount_median_card_1	134	actual_med_card_merchant_1
15	amount_median_card_3	135	actual_med_card_merchant_3
16	amount_median_card_7	136	actual_med_card_merchant_7
17	amount_median_card_14	137	actual_med_card_merchant_14
18	amount_median_card_30	138	actual_med_card_merchant_30
19	amount_sum_card_0	139	actual_sum_card_merchant_0
20	amount_sum_card_1	140	actual_sum_card_merchant_1
21	amount_sum_card_3	141	actual_sum_card_merchant_3
22	amount_sum_card_7	142	actual_sum_card_merchant_7
23	amount_sum_card_14	143	actual_sum_card_merchant_14
24	amount_sum_card_30	144	actual_sum_card_merchant_30
25	actual_avg_card_0	145	amount_avg_card_zip_0
26	actual_avg_card_1	146	amount_avg_card_zip_1
27	actual_avg_card_3	147	amount_avg_card_zip_3
28	actual_avg_card_7	148	amount_avg_card_zip_7
29	actual_avg_card_14	149	amount_avg_card_zip_14
30	actual_avg_card_30	150	amount_avg_card_zip_30
31	actual_max_card_0	151	amount_max_card_zip_0

32	actual_max_card_1	152	amount_max_card_zip_1
33	actual_max_card_3	153	amount_max_card_zip_3
34	actual_max_card_7	154	amount_max_card_zip_7
35	actual_max_card_14	155	amount_max_card_zip_14
36	actual_max_card_30	156	amount_max_card_zip_30
37	actual_med_card_0	157	amount_median_card_zip_0
38	actual_med_card_1	158	amount_median_card_zip_1
39	actual_med_card_3	159	amount_median_card_zip_3
40	actual_med_card_7	160	amount_median_card_zip_7
41	actual_med_card_14	161	amount_median_card_zip_14
42	actual_med_card_30	162	amount_median_card_zip_30
43	actual_sum_card_0	163	amount_sum_card_zip_0
44	actual_sum_card_1	164	amount_sum_card_zip_1
45	actual_sum_card_3	165	amount_sum_card_zip_3
46	actual_sum_card_7	166	amount_sum_card_zip_7
47	actual_sum_card_14	167	amount_sum_card_zip_14
48	actual_sum_card_30	168	amount_sum_card_zip_30
49	amount_avg_merchant_0	169	actual_avg_card_zip_0
50	amount_avg_merchant_1	170	actual_avg_card_zip_1
51	amount_avg_merchant_3	171	actual_avg_card_zip_3
52	amount_avg_merchant_7	172	actual_avg_card_zip_7
53	amount_avg_merchant_14	173	actual_avg_card_zip_14
54	amount_avg_merchant_30	174	actual_avg_card_zip_30
55	amount_max_merchant_0	175	actual_max_card_zip_0
56	amount_max_merchant_1	176	actual_max_card_zip_1
57	amount_max_merchant_3	177	actual_max_card_zip_3
58	amount_max_merchant_7	178	actual_max_card_zip_7
59	amount_max_merchant_14	179	actual_max_card_zip_14
60	amount_max_merchant_30	180	actual_max_card_zip_30
61	amount_median_merchant_0	181	actual_med_card_zip_0
62	amount_median_merchant_1	182	actual_med_card_zip_1
63	amount_median_merchant_3	183	actual_med_card_zip_3
64	amount_median_merchant_7	184	actual_med_card_zip_7
65	amount_median_merchant_14	185	actual_med_card_zip_14
66	amount_median_merchant_30	186	actual_med_card_zip_30
67	amount_sum_merchant_0	187	actual_sum_card_zip_0
68	amount_sum_merchant_1	188	actual_sum_card_zip_1

69	amount_sum_merchant_3	189	actual_sum_card_zip_3
70	amount_sum_merchant_7	190	actual_sum_card_zip_7
71	amount_sum_merchant_14	191	actual_sum_card_zip_14
72	amount_sum_merchant_30	192	actual_sum_card_zip_30
73	actual_avg_merchant_0	193	amount_avg_card_state_0
74	actual_avg_merchant_1	194	amount_avg_card_state_1
75	actual_avg_merchant_3	195	amount_avg_card_state_3
76	actual_avg_merchant_7	196	amount_avg_card_state_7
77	actual_avg_merchant_14	197	amount_avg_card_state_14
78	actual_avg_merchant_30	198	amount_avg_card_state_30
79	actual_max_merchant_0	199	amount_max_card_state_0
80	actual_max_merchant_1	200	amount_max_card_state_1
81	actual_max_merchant_3	201	amount_max_card_state_3
82	actual_max_merchant_7	202	amount_max_card_state_7
83	actual_max_merchant_14	203	amount_max_card_state_14
84	actual_max_merchant_30	204	amount_max_card_state_30
85	actual_med_merchant_0	205	amount_median_card_state_0
86	actual_med_merchant_1	206	amount_median_card_state_1
87	actual_med_merchant_3	207	amount_median_card_state_3
88	actual_med_merchant_7	208	amount_median_card_state_7
89	actual_med_merchant_14	209	amount_median_card_state_14
90	actual_med_merchant_30	210	amount_median_card_state_30
91	actual_sum_merchant_0	211	amount_sum_card_state_0
92	actual_sum_merchant_1	212	amount_sum_card_state_1
93	actual_sum_merchant_3	213	amount_sum_card_state_3
94	actual_sum_merchant_7	214	amount_sum_card_state_7
95	actual_sum_merchant_14	215	amount_sum_card_state_14
96	actual_sum_merchant_30	216	amount_sum_card_state_30
97	amount_avg_card_merchant_0	217	actual_avg_card_state_0
98	amount_avg_card_merchant_1	218	actual_avg_card_state_1
99	amount_avg_card_merchant_3	219	actual_avg_card_state_3
100	amount_avg_card_merchant_7	220	actual_avg_card_state_7
101	amount_avg_card_merchant_14	221	actual_avg_card_state_14
102	amount_avg_card_merchant_30	222	actual_avg_card_state_30
103	amount_max_card_merchant_0	223	actual_max_card_state_0
104	amount_max_card_merchant_1	224	actual_max_card_state_1
105	amount_max_card_merchant_3	225	actual_max_card_state_3

106	amount_max_card_merchant_7	226	actual_max_card_state_7
107	amount_max_card_merchant_14	227	actual_max_card_state_14
108	amount_max_card_merchant_30	228	actual_max_card_state_30
109	amount_median_card_merchant_0	229	actual_med_card_state_0
110	amount_median_card_merchant_1	230	actual_med_card_state_1
111	amount_median_card_merchant_3	231	actual_med_card_state_3
112	amount_median_card_merchant_7	232	actual_med_card_state_7
113	amount_median_card_merchant_14	233	actual_med_card_state_14
114	amount_median_card_merchant_30	234	actual_med_card_state_30
115	amount_sum_card_merchant_0	235	actual_sum_card_state_0
116	amount_sum_card_merchant_1	236	actual_sum_card_state_1
117	amount_sum_card_merchant_3	237	actual_sum_card_state_3
118	amount_sum_card_merchant_7	238	actual_sum_card_state_7
119	amount_sum_card_merchant_14	239	actual_sum_card_state_14
120	amount_sum_card_merchant_30	240	actual_sum_card_state_30
Frequency variables			
241	count_card_0	256	count_card_merchant_7
242	count_card_1	257	count_card_merchant_14
243	count_card_3	258	count_card_merchant_30
244	count_card_7	259	count_card_zip_0
245	count_card_14	260	count_card_zip_1
246	count_card_30	261	count_card_zip_3
247	count_merchant_0	262	count_card_zip_7
248	count_merchant_1	263	count_card_zip_14
249	count_merchant_3	264	count_card_zip_30
250	count_merchant_7	265	count_card_state_0
251	count_merchant_14	266	count_card_state_1
252	count_merchant_30	267	count_card_state_3
253	count_card_merchant_0	268	count_card_state_7
254	count_card_merchant_1	269	count_card_state_14
255	count_card_merchant_3	270	count_card_state_30
Days since variables			
271	days_since_card	274	days_since_card_zip
272	days_since_merchant	275	days_since_card_state
273	days_since_card_merchant		
Velocity change variables			
276	number_card_0/number_card_7	324	number_merchant_0/number_card_7

277	number_card_0/number_card_14	325	number_merchant_0/number_card_14
278	number_card_0/number_card_30	326	number_merchant_0/number_card_30
279	number_card_1/number_card_7	327	number_merchant_1/number_card_7
280	number_card_1/number_card_14	328	number_merchant_1/number_card_14
281	number_card_1/number_card_30	329	number_merchant_1/number_card_30
282	amount_card_0/number_card_7	330	amount_merchant_0/number_card_7
283	amount_card_0/number_card_14	331	amount_merchant_0/number_card_14
284	amount_card_0/number_card_30	332	amount_merchant_0/number_card_30
285	amount_card_1/number_card_7	333	amount_merchant_1/number_card_7
286	amount_card_1/number_card_14	334	amount_merchant_1/number_card_14
287	amount_card_1/number_card_30	335	amount_merchant_1/number_card_30
288	number_card_0/amount_card_7	336	number_merchant_0/amount_card_7
289	number_card_0/amount_card_14	337	number_merchant_0/amount_card_14
290	number_card_0/amount_card_30	338	number_merchant_0/amount_card_30
291	number_card_1/amount_card_7	339	number_merchant_1/amount_card_7
292	number_card_1/amount_card_14	340	number_merchant_1/amount_card_14
293	number_card_1/amount_card_30	341	number_merchant_1/amount_card_30
294	amount_card_0/amount_card_7	342	amount_merchant_0/amount_card_7
295	amount_card_0/amount_card_14	343	amount_merchant_0/amount_card_14
296	amount_card_0/amount_card_30	344	amount_merchant_0/amount_card_30
297	amount_card_1/amount_card_7	345	amount_merchant_1/amount_card_7
298	amount_card_1/amount_card_14	346	amount_merchant_1/amount_card_14
299	amount_card_1/amount_card_30	347	amount_merchant_1/amount_card_30
300	number_card_0/number_merchant_7	348	number_merchant_0/number_merchant_7
301	number_card_0/number_merchant_14	349	number_merchant_0/number_merchant_14
302	number_card_0/number_merchant_30	350	number_merchant_0/number_merchant_30
303	number_card_1/number_merchant_7	351	number_merchant_1/number_merchant_7
304	number_card_1/number_merchant_14	352	number_merchant_1/number_merchant_14
305	number_card_1/number_merchant_30	353	number_merchant_1/number_merchant_30
306	amount_card_0/number_merchant_7	354	amount_merchant_0/number_merchant_7

307	amount_card_0/number_merchant_14	355	amount_merchant_0/number_merchant_14
308	amount_card_0/number_merchant_30	356	amount_merchant_0/number_merchant_30
309	amount_card_1/number_merchant_7	357	amount_merchant_1/number_merchant_7
310	amount_card_1/number_merchant_14	358	amount_merchant_1/number_merchant_14
311	amount_card_1/number_merchant_30	359	amount_merchant_1/number_merchant_30
312	number_card_0/amount_merchant_7	360	number_merchant_0/amount_merchant_7
313	number_card_0/amount_merchant_14	361	number_merchant_0/amount_merchant_14
314	number_card_0/amount_merchant_30	362	number_merchant_0/amount_merchant_30
315	number_card_1/amount_merchant_7	363	number_merchant_1/amount_merchant_7
316	number_card_1/amount_merchant_14	364	number_merchant_1/amount_merchant_14
317	number_card_1/amount_merchant_30	365	number_merchant_1/amount_merchant_30
318	amount_card_0/amount_merchant_7	366	amount_merchant_0/amount_merchant_7
319	amount_card_0/amount_merchant_14	367	amount_merchant_0/amount_merchant_14
320	amount_card_0/amount_merchant_30	368	amount_merchant_0/amount_merchant_30
321	amount_card_1/amount_merchant_7	369	amount_merchant_1/amount_merchant_7
322	amount_card_1/amount_merchant_14	370	amount_merchant_1/amount_merchant_14
323	amount_card_1/amount_merchant_30	371	amount_merchant_1/amount_merchant_30

USCMarshall
School of Business

Appendix 2

Data Quality Report:
Credit Card Transaction Data

March 2019

6:30-9:30pm Session Team 1:
Justice League Consulting Group

Team Members:
Zongyang Jiao, Chengyin Liu, Jiayi Ma,
Xinyue Niu, Xueyan Gu, Jie Zhao

1.0 INTRODUCTION

The name of the Dataset is Card Transactions Data. There are 10 variables in this dataset, each record has 1 field indicating such record is Fraud (1) or not (0).

Covered Time Period: 2010-01-01 to 2010-12-31
Number of Fields: 10

2.0 FILEDS SUMMARY

2.0.1 All Records

Table 2.1

No.	Field Name	Field Type	# Records	%Populated	# Unique Values	# Records with Value NaN/" "	Other
1	Recnum	Ordinal	96,753	100.00%	96,753	0	From 1 to 96,753
2	Cardnum	Categorical	96,753	100.00%	1,645	0	5142148452: 1,192 5142184598: 921 5142189108: 663 …
3	Date	Time	96,753	100.00%	365	0	2010-02-28: 684 2010-08-10: 610 2010-03-15: 594 … [2010-01-01 to 2010-12-31]
4	Merchnum	Categorical	93,378	96.39%	13,091	3,375	930090121224: 9,310 5509006296254: 2,131 9900020006406: 1,714 …
5	Merch description	Categorical	96,753	100.00%	13,126	0	GSA-FSS-ADV: 1.688 SIGMA-ALDRICH: 1,635 STAPLES #941 : 1,174 …
6	Merch state	Categorical	96,753	98.76%	227	1,195	TN: 12,035 VA: 7,872 CA: 6,817 …
7	Merch zip	Categorical	96,753	95.19%	4,567	4,656	38118: 11,868 63103: 1,650 8701: 1,267 …
8	Transtype	Categorical	96,753	100.00%	4	0	P: 96,398 A: 181 D: 173 Y: 1

9	Amount	Numeric	96,753	100.00%	-	0	Unit: US Dollar Min: 0.01 Max: 3,102,045.53 Mean: 427.89 Std: 10,006.14
10	Fraud	Categorical	96,753	100.00%	2	0	0: 95,694 1: 1,059

2.0.2 Not Fraud Records

Table 2.2

No.	Field Name	Field Type	# Records	%Populated	# Unique Values	# Records with Value NaN/" "	Other
1	Recnum	Ordinal	95,694	100.00%	95,694	0	-
2	Cardnum	Categorical	95,694	100.00%	1,640	0	5142148452: 1,192 5142184598: 921 5142189108: 663 …
3	Date	Time	95,694	100.00%	365	0	2010-02-28: 684 2010-03-15: 594 2010-08-10: 577 … [2010-01-01 to 2010-12-31]
4	Merchnum	Categorical	95,694	96.49%	13,087	3,362	930090121224: 9,260 5509006296254: 2,131 9900020006406: 1,689 …
5	Merch description	Categorical	95,694	100.00%	13,122	0	GSA-FSS-ADV: 1,688 SIGMA-ALDRICH: 1,635 STAPLES #941 : 1,174 …
6	Merch state	Categorical	95,694	98.75%	227	1,192	TN: 11,937 VA: 7,776 CA: 6,726 …
7	Merch zip	Categorical	95,694	95.16%	4,567	4,630	38118: 11,772 63103: 1,647 8701: 1,263 …
8	Transtype	Categorical	95,694	100.00%	4	0	P: 95,339 A: 181 D: 173 Y: 1
9	Amount	Numeric	95,694	100.00%	-	0	Unit: US Dollar Min: 0.01 Max: 3,102,045.53 Mean: 409.34 Std: 10,054.62

10	Fraud	Categorical	95,694	100.00%	1	0	0: 95,694

2.0.3 Fraud Records

Table 2.3

No.	Field Name	Field Type	# Records	%Populated	# Unique Values	# Records with Value NaN/" "	Other
1	Recnum	Ordinal	1,059	100.00%	1,059	0	-
2	Cardnum	Categorical	1,059	100.00%	111	0	5142140316: 46 5142847398: 45 5142199009: 45 …
3	Date	Time	1,059	100.00%	241	0	2010-08-10: 33 2010-05-17: 30 2010-08-04: 28 … [2010-01 to 2010-12]
4	Merchnum	Categorical	1,059	98.77%	257	13	4353000719908: 107 930090121224: 50 8834000695423: 46 …
5	Merch description	Categorical	1,059	100.00%	276	0	AMAZON.COM *SUPERSTOR: 54 ACI*AMAZON.COM INC: 48 STEVES COMPUTER REPAIR : 46 …
6	Merch state	Categorical	1,059	99.72%	35	3	WA: 139 TN: 98 VA: 96 …
7	Merch zip	Categorical	1,059	97.54%	194	26	98101: 107 38118: 96 22202: 53 …
8	Transtype	Categorical	1,059	100.00%	1	0	P: 1,059
9	Amount	Numeric	1,059	100.00%	-	0	Unit: US Dollar Min: 0.22 Max: 30,372.46 Mean: 2,103.35 Std: 3,068.53
10	Fraud	Categorical	1,059	100.00%	1	0	1: 1,059

3.0 DATA QUALITY ASSESSMENT

3.01 Recnum

FILE KEY, to uniquely identify each record, ranging from 1 to 96,753

3.02 Cardnum

Cardnum is the card number of each record.

- **All Records**

Table 3.1.1

Cardnum	Count
5142148452	1,192
5142184598	921
5142189108	663
5142297710	583
5142223373	579
5142187452	526
5142299634	515
…	…

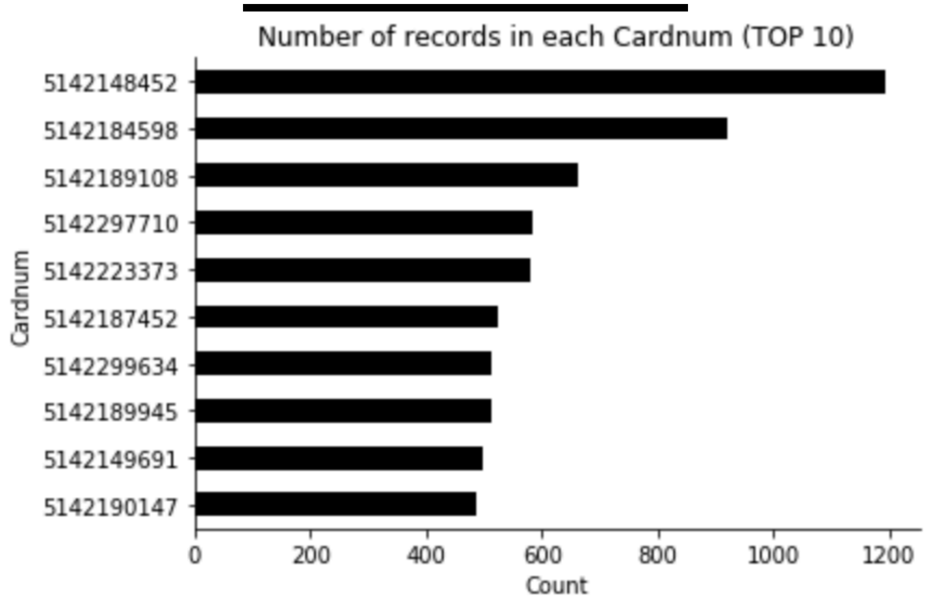


Figure 3.1.1

- **Not Fraud Records**

Table 3.1.2

Cardnum	Count
5142148452	1,192
5142184598	921
5142189108	663
5142297710	583
5142223373	575
5142187452	526
5142299634	514
…	…

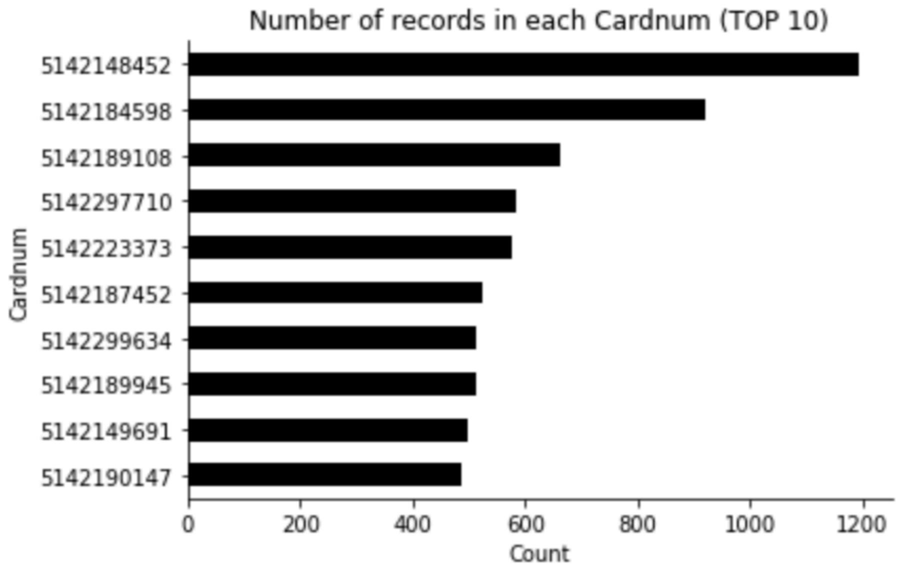


Figure 3.1.2

- **Fraud Records**

Table 3.1.3

Cardnum	Count
5142140316	46
5142847398	45
5142199009	45
5142160778	41
5142189341	41
5142181728	39
5142212038	39
…	…

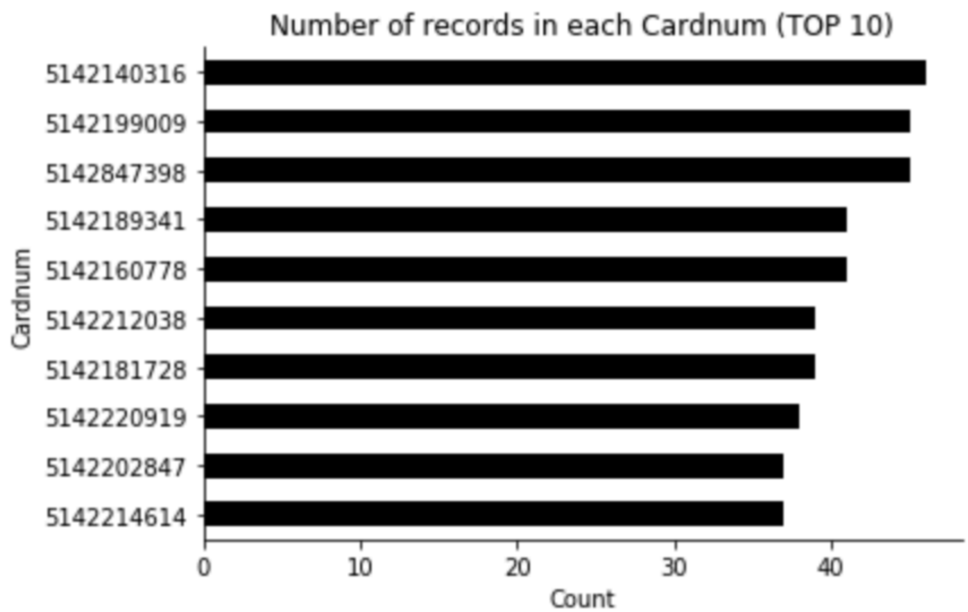


Figure 3.1.3

3.03 Date

Date is the date that record generated.

- **All Records**

Table 3.2.1

Date	Count
2010-02-28	684
2010-08-10	610
2010-03-15	594
2010-09-13	564
2010-08-09	536
…	…

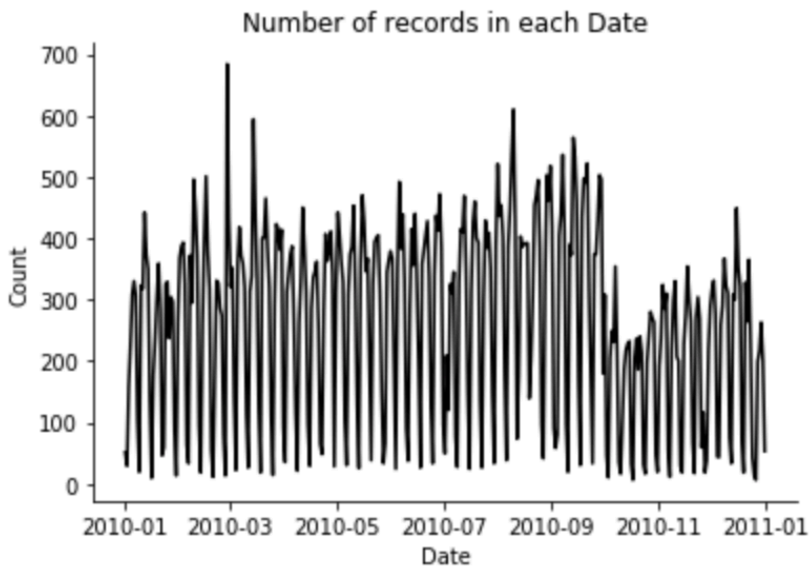


Figure 3.2.1

- **Not Fraud Records**

Table 3.2.2

Date	Count
2010-02-28	684
2010-03-15	594
2010-08-10	577
2010-09-13	564
2010-09-07	535
…	…

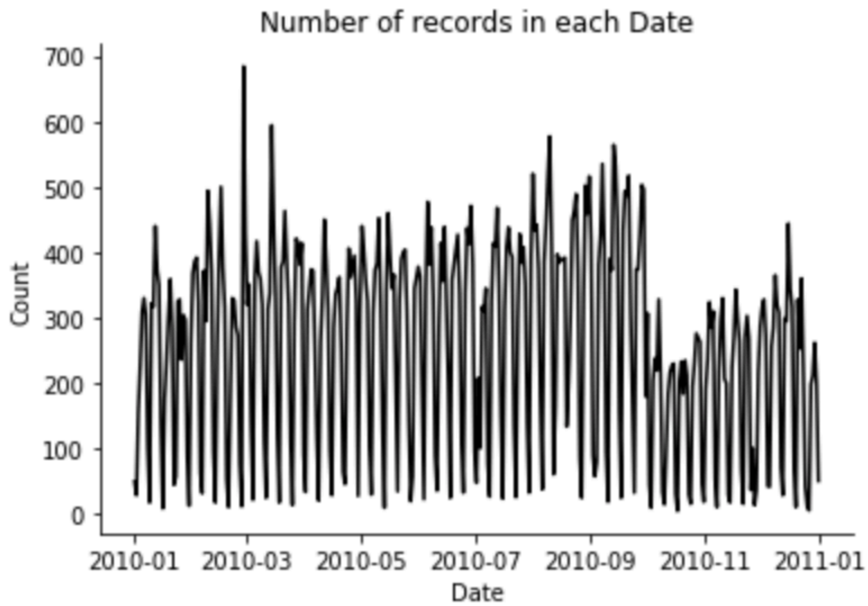


Figure 3.2.2

- **Fraud Records**

Table 3.2.3

Date	Count
2010-08-10	33
2010-05-17	30
2010-08-04	28
2010-10-07	26
2010-09-05	24
…	…

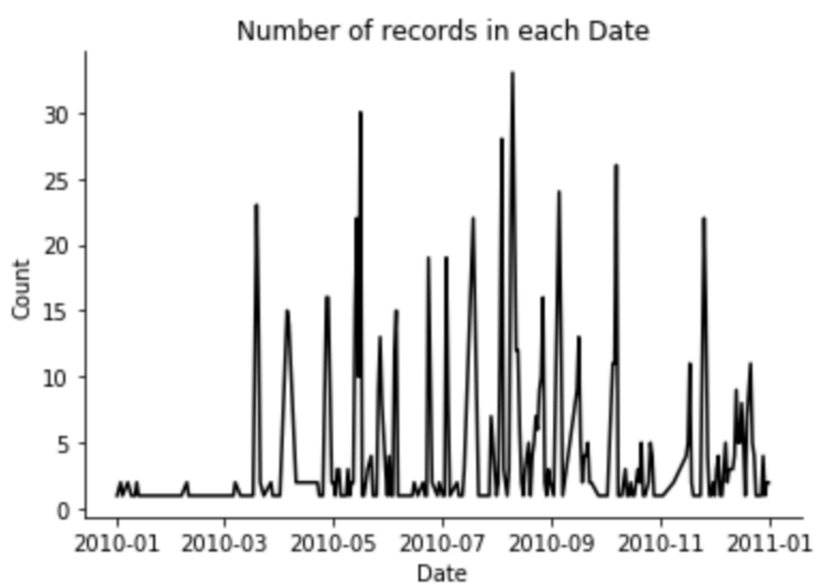


Figure 3.2.3

3.04 Merchnum

Merchnum is the number of a certain merchant.

- **All Records**

Table 3.3.1

Merchnum	Count
930090121224	9,310
5509006296254	2,131
9900020006406	1,714
602608969534	1,092
4353000719908	1,020
…	…

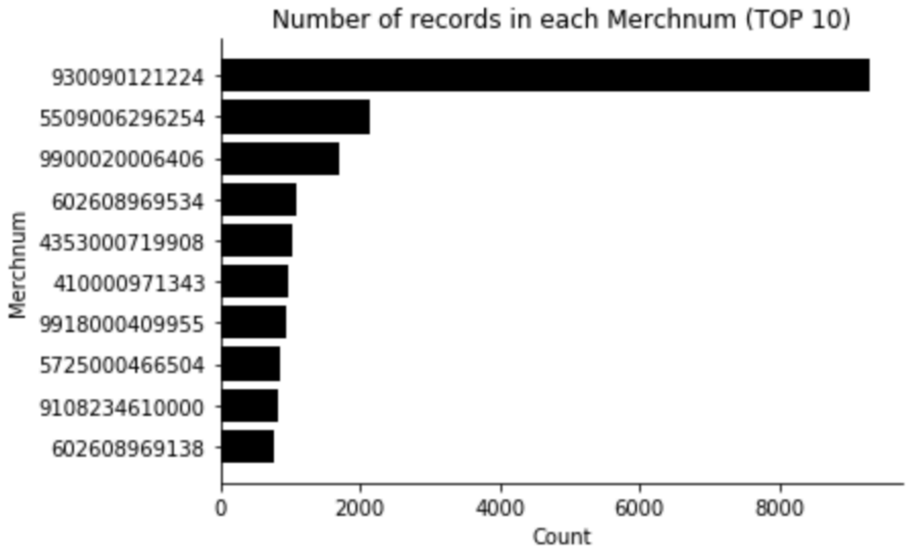


Figure 3.3.1

- **Not Fraud Records**

Table 3.3.2

Merchnum	Count
930090121224	9,260
5509006296254	2,131
9900020006406	1,689
602608969534	1,091
410000971343	981
…	…

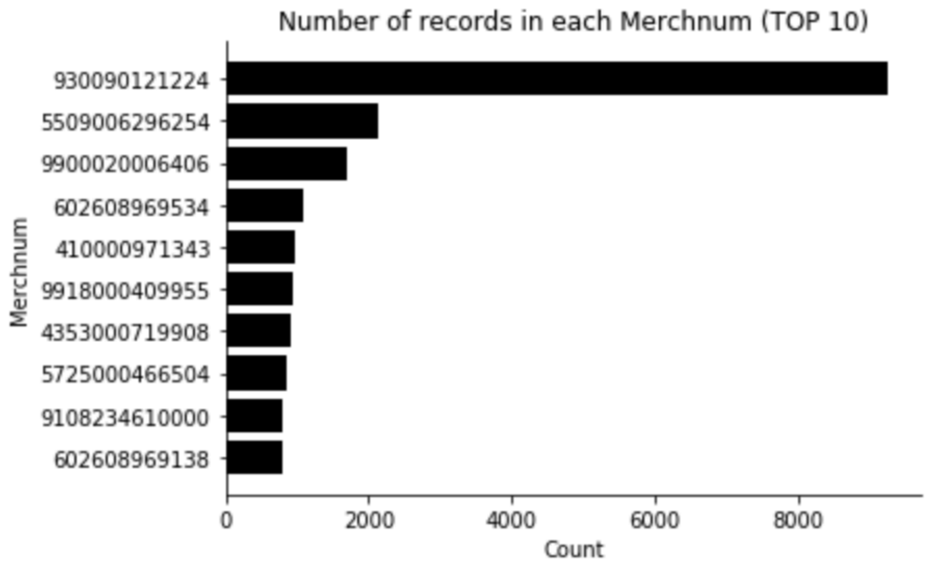


Figure 3.3.2

- **Fraud Records**

Table 3.3.3

Merchnum	Count
4353000719908	107
930090121224	50
8834000695423	46
4503738417400	45
4620009957157	39
…	…

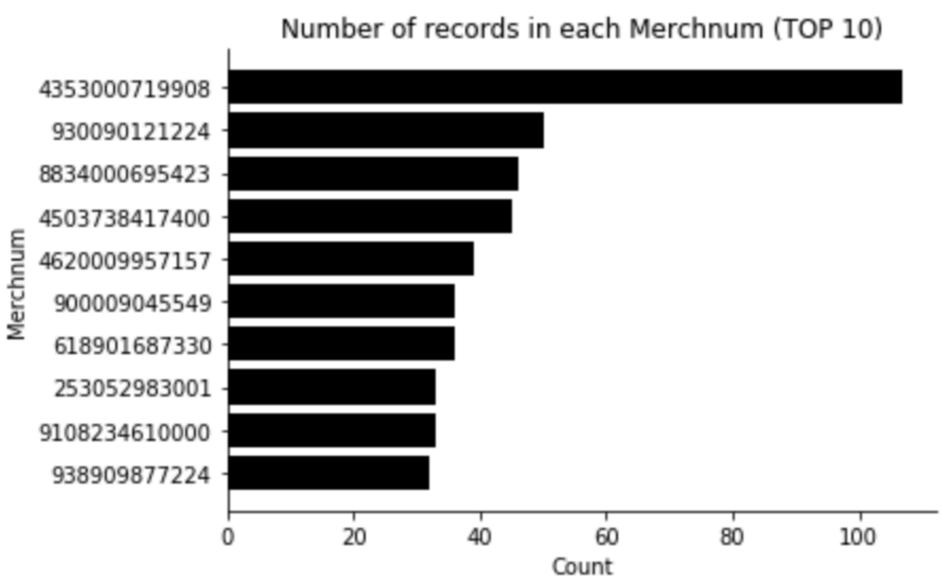


Figure 3.3.3

3.05 Merch description

Merch description is the description of a certain merchant.

- **All Records**

Table 3.4.1

Merch description	Count
GSA-FSS-ADV	1,688
SIGMA-ALDRICH	1,635
STAPLES #941	1,174
FISHER SCI ATL	1,093
MWI*MICRO WAREHOUSE	958
…	…

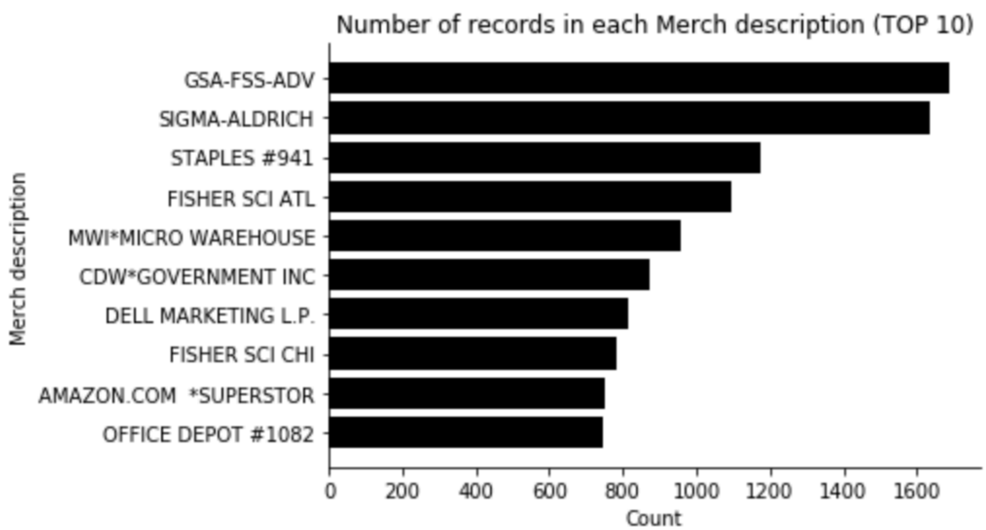


Figure 3.4.1

- **Not Fraud Records**

Table 3.4.2

Merch description	Count
GSA-FSS-ADV	1,663
SIGMA-ALDRICH	1,632
STAPLES #941	1,131
FISHER SCI ATL	1,092
MWI*MICRO WAREHOUSE	955
…	…

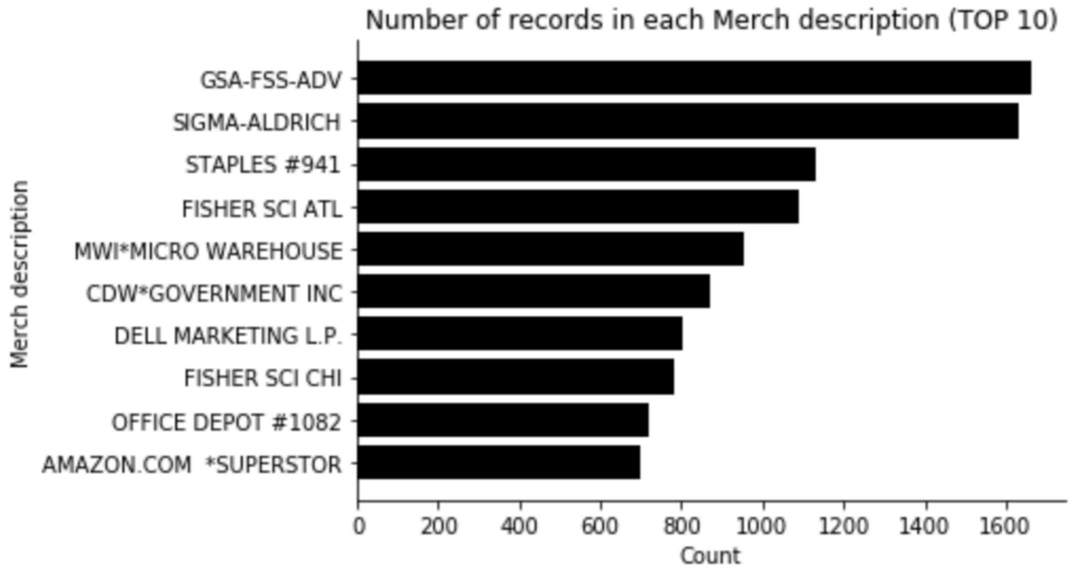


Figure 3.4.2

- **Fraud Records**

Table 3.4.3

Merch description	Count
AMAZON.COM *SUPERSTOR	54
ACI*AMAZON.COM INC	48
STEVES COMPUTER REPAIR	46
DIRKS PLUMBING/HEATING REPAIRS	45
STAPLES #941	43
…	…

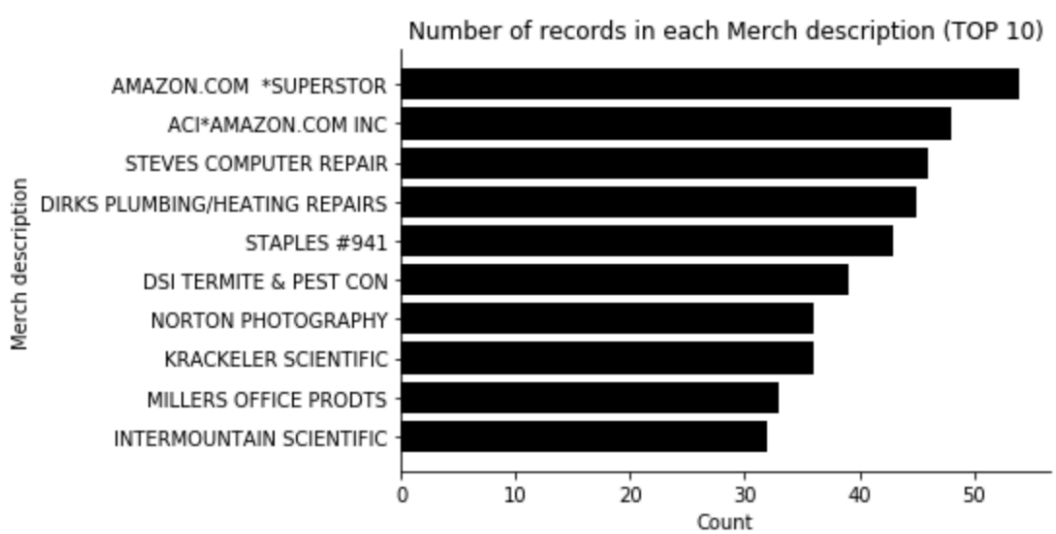


Figure 3.4.3

3.06 Merch state

Merch state is the state the merchant in such record at.

- **All Records**

Table 3.5.1

Merch state	Count
TN	12,035
VA	7,872
CA	6,817
IL	6,508
MD	5,398
…	…

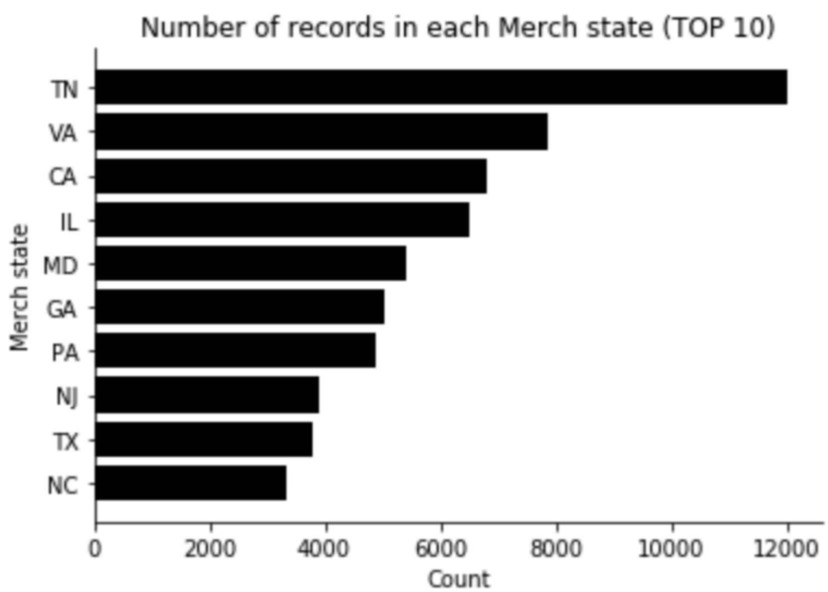


Figure 3.5.1

- **Not Fraud Records**

Table 3.5.2

Merch state	Count
TN	11,937
VA	7,776
CA	6,726
IL	6,466
MD	5,316
…	…

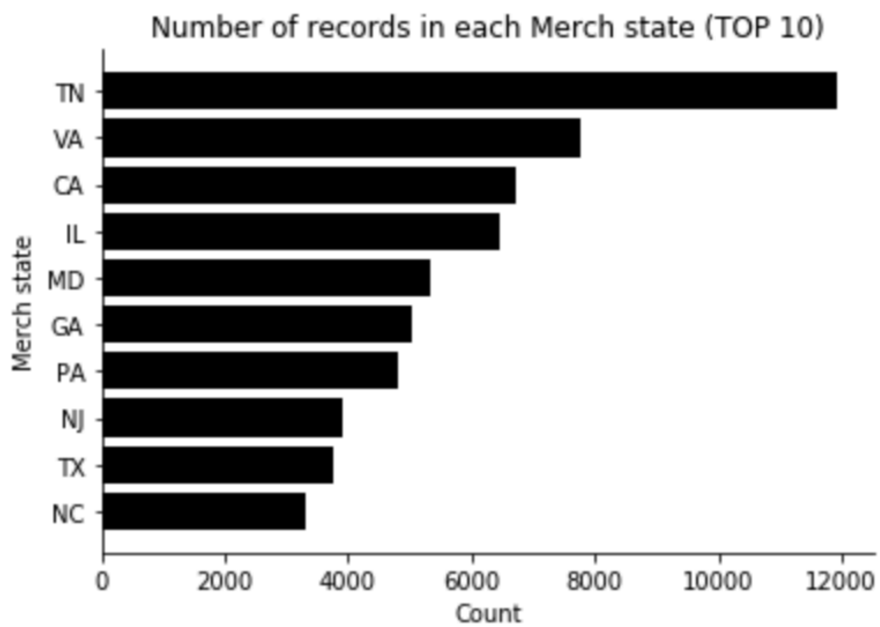


Figure 3.5.2

- **Fraud Records**

Table 3.5.3

Merch state	Count
WA	139
TN	98
VA	96
CA	91
PA	85
…	…

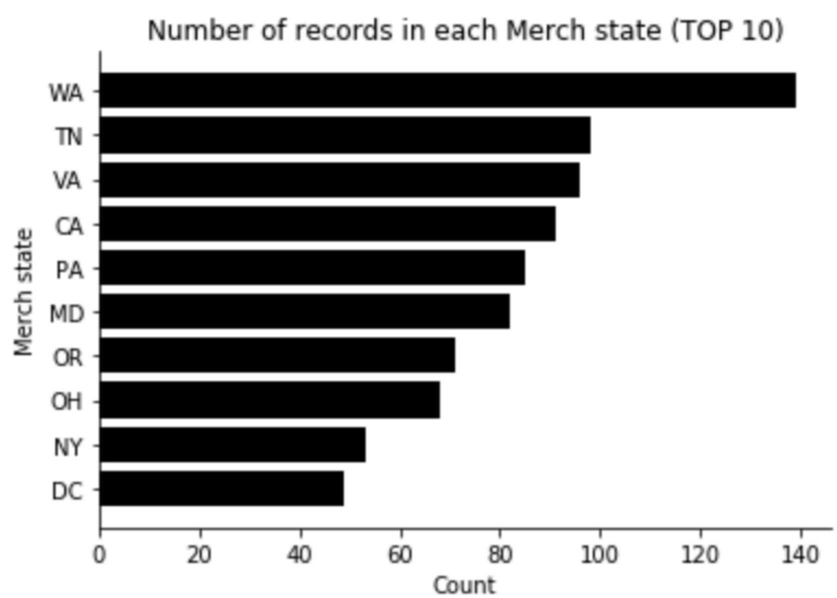


Figure 3.5.3

3.07 Merch zip

Merch zip is the zip code that a certain merchant located.

- **All Records**

Table 3.6.1

Merch zip	Count
38118	11,868
63103	1,650
8701	1,267
22202	1,250
60061	1,221
…	…

- **Not Fraud Records**

Table 3.6.2

Merch zip	Count
38118	11,772
63103	1,647
8701	1,263
60061	1,217
22202	1,197
…	…

- **Fraud Records**

Table 3.6.3

Merch zip	Count
98101	107
38118	96
22202	53
92656	49
17201	44
…	…

3.08 Transtype

Transtype stands for the transaction type.

- **All Records**

Table 3.7.1

Transtype	Count
P	96,398
A	181
D	173
Y	1

- **Not Fraud Records**

Table 3.7.2

Transtype	Count
P	95,339
A	181
D	173
Y	1

- **Fraud Records**

Table 3.7.3

Transtype	Count
P	1,059

3.09 Amount

Amount stands for the amount a customer spent in such record.

- **All Records**

Table 3.8.1

Unit	Max	Min	Mean	Std
US Dollar	3,102,045.53	0.01	427.89	10,006.14

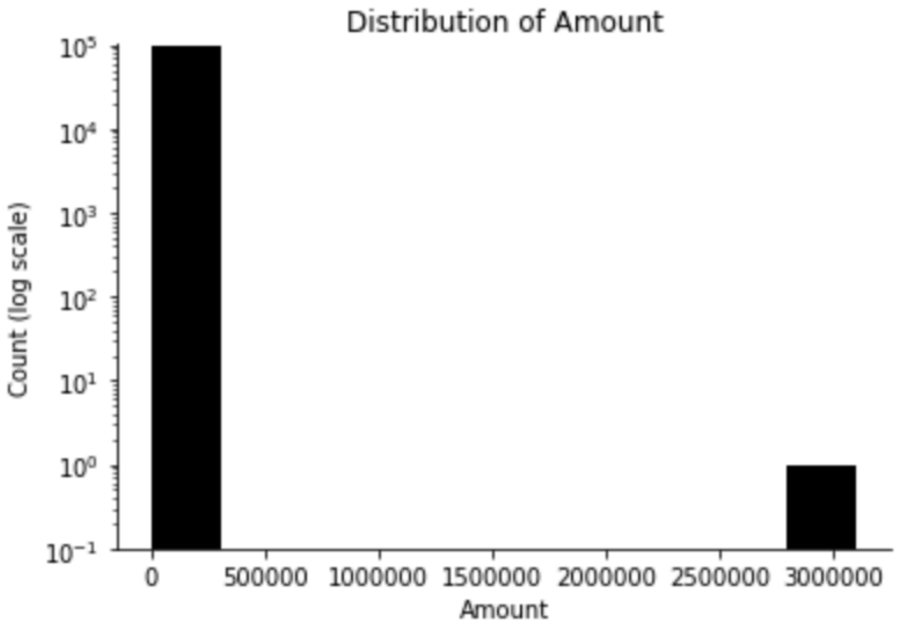


Figure 3.8.1

As we can see that there are outliers in the graph above, which make us cannot see the details on the left, so we try to plot the data without the outliers. (Amount <= 500,000)

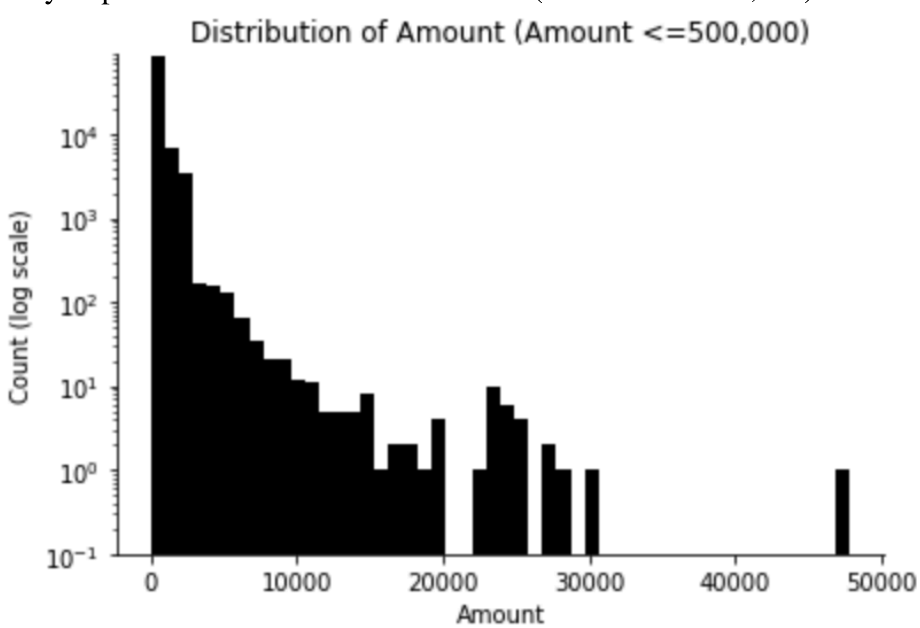


Figure 3.8.2

- **Not Fraud Records**

Table 3.8.2

Unit	Max	Min	Mean	Std
US Dollar	3,102,045.53	0.01	409.34	10,054.62

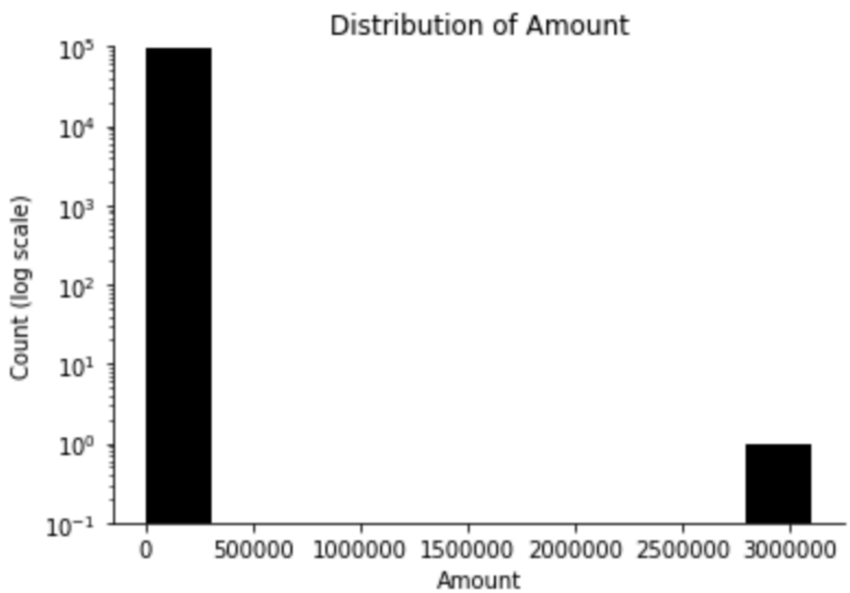


Figure 3.8.3

without the outliers. (Amount <= 500,000)

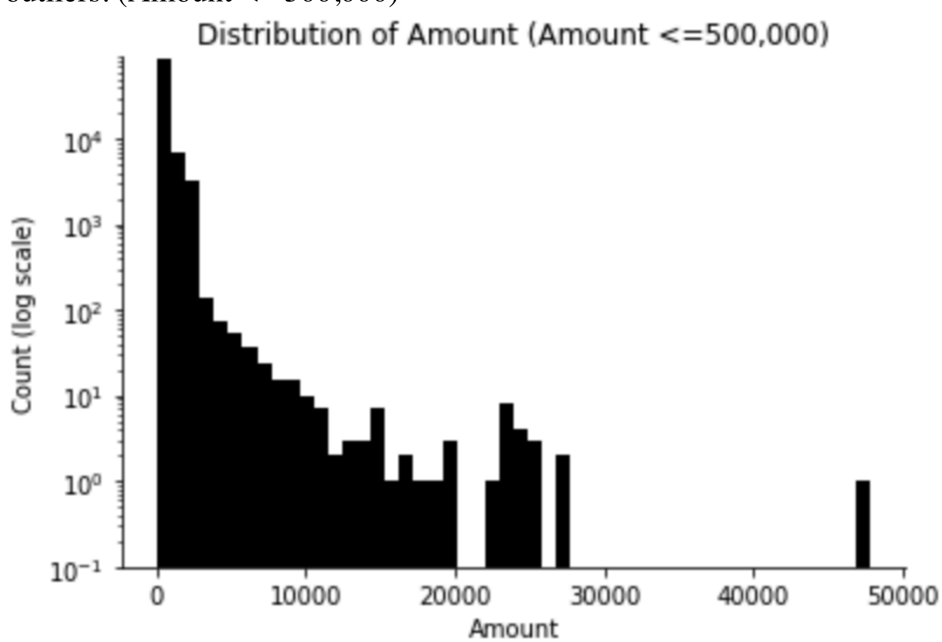


Figure 3.8.4

- **Fraud Records**

Table 3.8.3

Unit	Max	Min	Mean	Std
US Dollar	30,372.46	0.22	2,103.35	3,068.53

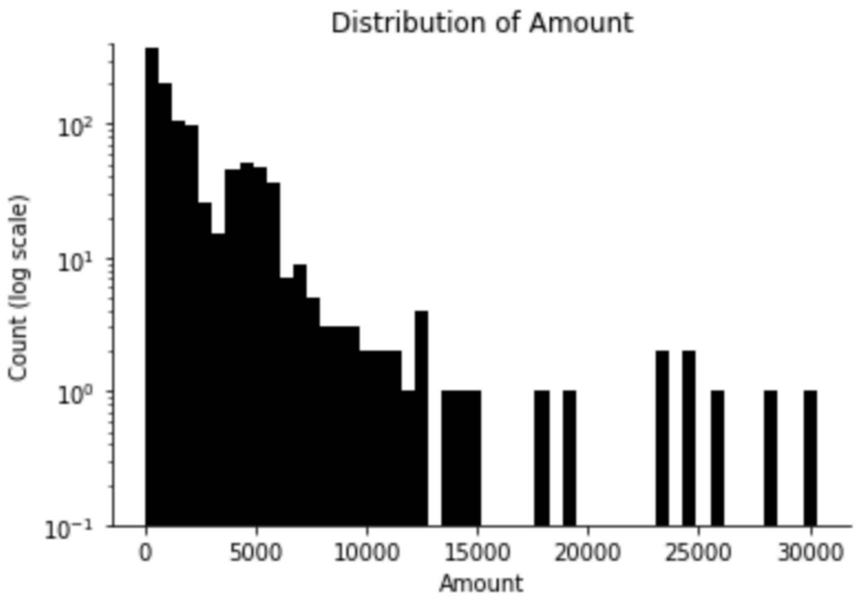


Figure 3.8.5

3.10 Fraud

Fraud is the label for each record, indicating such record is Fraud or not.

Table 3.9

Merch state	Count
0	95,694
1	1,059