DSO 562 Project 3

Finding Anomalies in Transaction Data

Team 15

Badawy, Yasmine

Chuang, Hungli

Gupta, Varun

Hu, Yiting

Lin, Qiongqiong

Rawat, Akash

Srivastava, Astha

Yu, Shui

Table of Contents

Executive Summary

Every year, millions of Americans fall victim to credit card fraud that costs the national economy billions of dollars. According to a report released by the Federal Trade Commission, Credit card fraud accounted for **32.7%** of total identity theft complaints from 2014 to 2016. Among all, California is one of the states that reported the highest rates of theft complaints per 100,000 people in 2016. We can see that credit card theft is a significant problem in the U.S. Therefore, the demand of using modern techniques to identify fraud transactions as soon and accurate as possible is increasing.

Our data scientist team is on this mission as well. The purpose of this project is to build a supervised machine learning model that detects anomalies to find out potential fraud credit card transaction records. We are utilizing a government agency dataset containing information on credit card transactions in 2010. It contains 96,753 records and ten fields on card transactions.

Our work is composed of 5 parts, Data Cleaning, Candidate Variables Creation, Feature Selection, Model Building, and Results. One can find details of each part in its corresponding section in the report. A high-level overview is as below:

- Data Cleaning:
  - The data was cleaned by filling in missing values. We then used our best judgment to fill in the fields of Merchnum, MerchState, and MerchZip. Further, we detected and removed outliers that would potentially bias our model.
- Candidate Variable Creation:
  - Using the original variables, we created Amount, Frequency, Days Since, Velocity, and Benford's Law Variables. There were 383 variables created by the end of this process.
- Feature Selection:
  - To reduce the dimensionality, we conducted the feature selection step to keep only the necessary variables. We first calculated univariate KS and FDR at 3%, and further utilized wrapper methods, including stepwise logistic regression and cross-validation, to narrow the number of variables to 25.
- Model Building:
  - Using the final 25 variables, we build four supervised models: Logistic Regression, Random Forest, Gradient Boosted Trees, and Neural Network
- Results:
  - Based on our analysis, we conclude that gradient boosted trees is the best model out of four. It has the highest average FDR of 58% on OOT (out-of-time) data.
  - We created a threshold for the top **8** percent of the transaction to be potential fraud to optimize the balance between detecting real fraudulent transactions and reducing mistreating a legal transaction as a fraud transaction.

Description of Data

Overall Description

The dataset contains the information of credit card transaction of a government agency across 2010. It also contains a label for fraud identification, which enables us to train supervised learning algorithms to identify fraud records. There are altogether 10 fields and 96,753 records in the dataset. There are nine categorical variables with the 'Recnum' variable uniquely defining each row. Following is a summary table of the categorical variables:

Table 1.1 Summary of categorical variables

Variable	Number of records with value	% populated	# unique values	# missing values	Most common value (MCV)	Frequency of MCV
Cardnum	96,753	100%	1,645	0	5142148452	1,192
Date	96,753	100%	365	0	2/28/2010	684
Merchnum	96,753	96.5%	13,091	3,375	930090121224	9,310
Merch description	96,753	100%	13,126	0	GSA-FSS-ADV	1,688
Merch state	96,753	98.8%	227	1,195	TN	12,035
Merch zip	96,753	95.3%	4,567	4,656	38118	11,868
Transtype	96,753	100%	4	0	P	96,398
Fraud	96,753	100%	2	0	0	95,694

	Number of records with value	% Populated	# unique values	Min	Max	Mean	Std	# records with zero value
Amount	96,753	100%	34,909	0.01	3102046.00	427.89	10006.14	0

Table 1.2 Summary of numerical variables

Description of Variables

Cardnum (Categorical, 10-digit code)

This categorical variable defines the credit card number of each transaction. There are 1,645 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

Cardnum	Count
5142148452	1192
5142184598	921
5142189108	663
5142297710	583
5142223373	579
5142187452	526
5142299634	515
5142189945	512
5142149691	497
5142190147	488

Table 1.3 Top values of 'Cardnum'

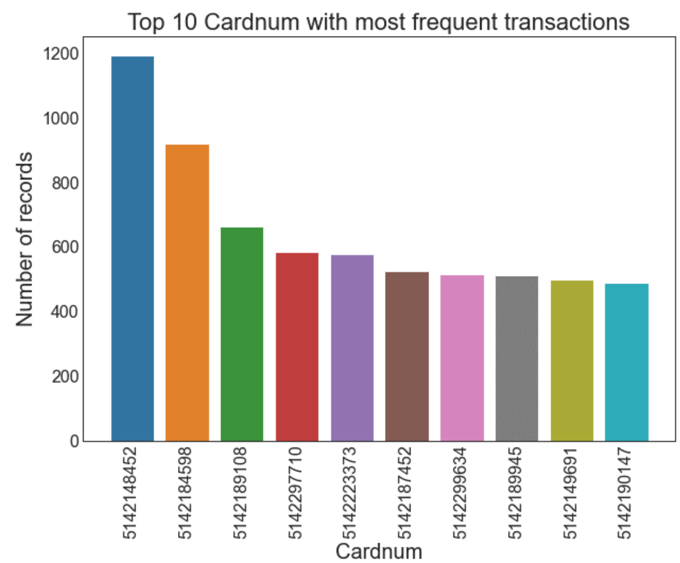


Fig 1.1 Categorical distribution of 'Cardnum

Date (Categorical, datetime)

This variable defines date for each transaction in the data. There are 365 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

Date	Count
2/28/2010	684
8/10/2010	610
3/15/2010	594
9/13/2010	564
9/7/2010	536
8/9/2010	536
9/14/2010	533
9/21/2010	522
8/1/2010	521
8/31/2010	518

Table 1.4 Top values of 'Date'

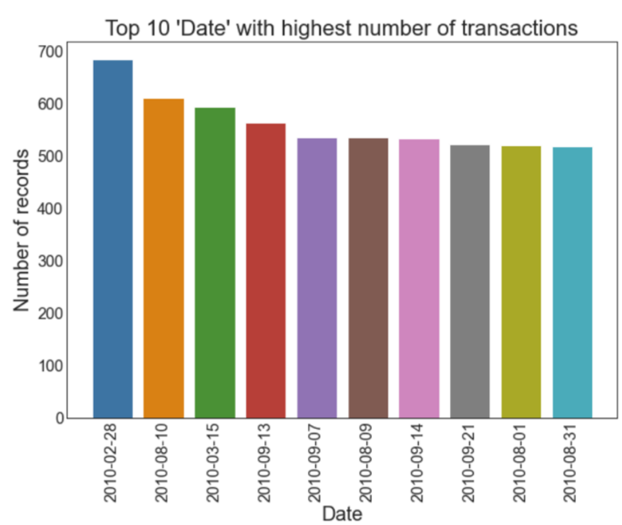


Fig 1.2 Categorical distribution of 'Date'

Merchnum (Categorical, 13-digit code)

This categorical variable defines the merchant number of each transaction/record. There are 13,091 unique values for this field with 3,375 (3.5%) missing/null values. Following is a distribution of the top 10 categories:

Merchnum	count
930090121224	9310
5509006296254	2131
9900020006406	1714
602608969534	1092
4353000719908	1020
410000971343	982
9918000409955	956
5725000466504	872
9108234610000	817
602608969138	783

Table 1.5 Top values of 'Merchnum'

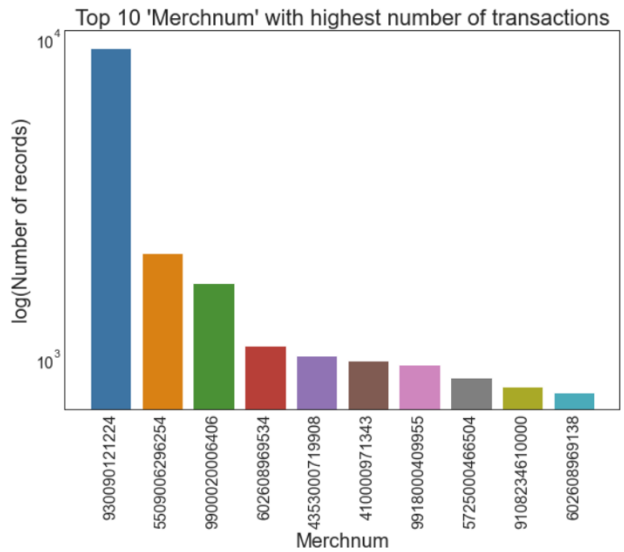


Fig 1.3 Categorical distribution of 'Merchnum'

Merch state (Categorical, string)

This categorical variable defines the state of the merchant, indicating location of the merchant. There are 227 unique values for this field with 1195 (1.2%) missing/null values. Following is a distribution of the top 10 categories:

Merch state	Count
TN	12035
VA	7872
CA	6817
IL	6508
MD	5398
GA	5025
PA	4899
NJ	3912
TX	3790
NC	3322

Table 1.7 Top values of 'Merch state'

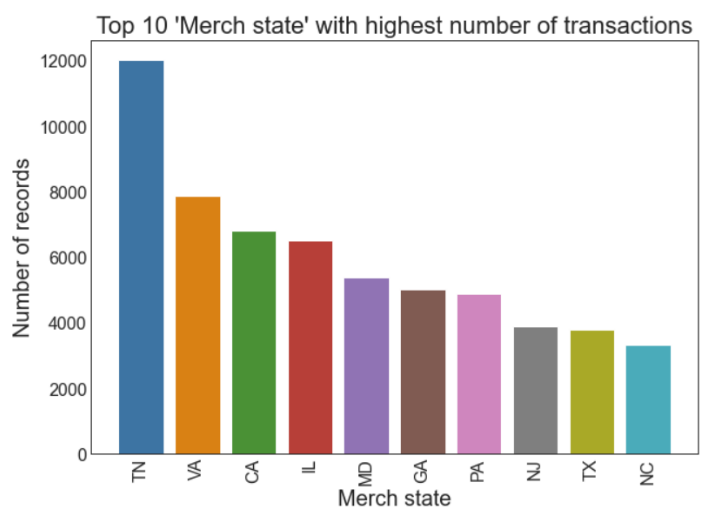


Fig 1.5 Categorical distribution of 'Merch state'

Merch zip (Categorical, 5-digit code)

This categorical variable defines the zip code of the merchant. There are 4,567 unique values for this field with 4,656 (4.8%) missing/null values. Following is a distribution of the top 10 categories:

Merch zip	Count
38118	11868
63103	1650
8701	1267
22202	1250
60061	1221
98101	1197
17201	1180
30091	1092
60143	942
60069	826

Table 1.8 Top values of 'Merch zip'

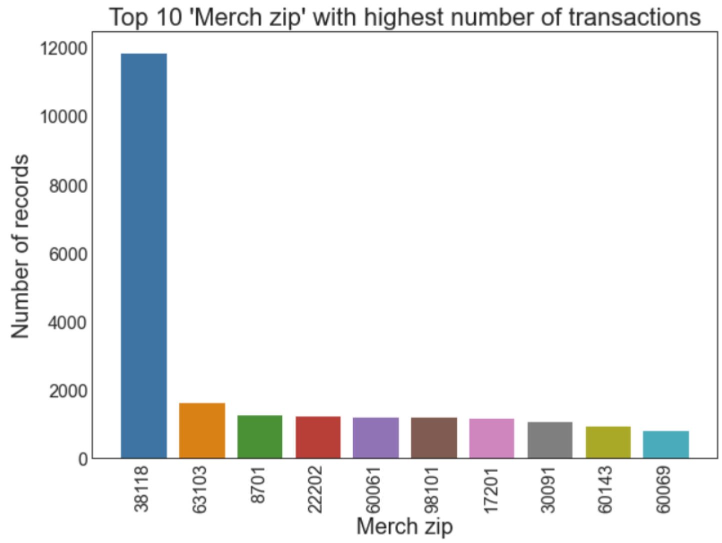


Fig 1.6 Categorical distribution of 'Merch zip'

Transtype (Categorical, single letter)

This categorical variable defines the 5-digit zip code of the applicant for each record/row. There are 4 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

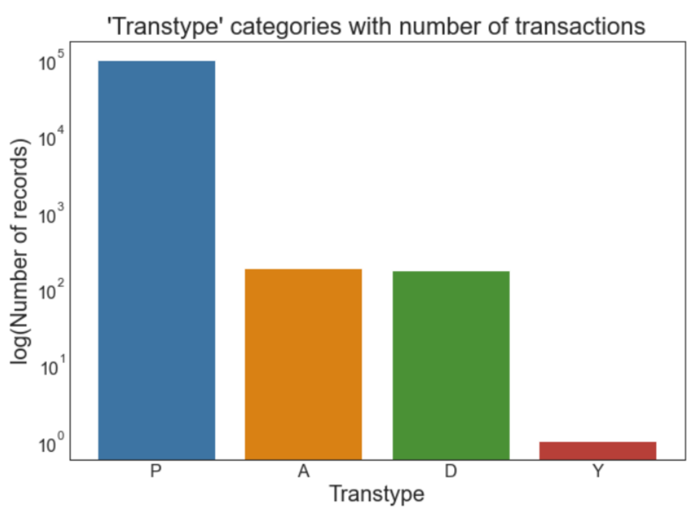


Fig 1.7 Categorical distribution of 'Transtype'

Amount (Categorical, float)

This numerical variable defines the amount of each transaction. There are no missing/null values. Following graph shows the distribution of the 'Amount' variable:

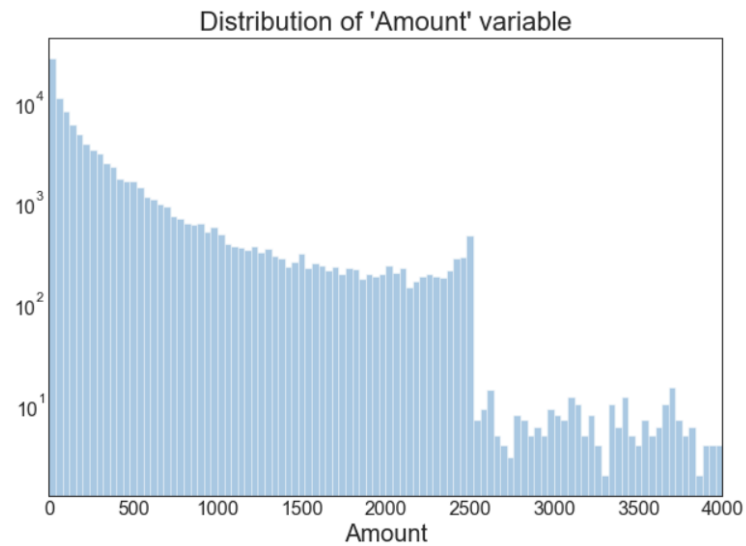


Fig 1.8 Distribution of 'amount'

Fraud (Categorical, 1-digit-code)

This categorical variable indicates if the transaction is fraud or not. 1 indicates that it is a fraudulent transaction while 0 indicates that it is not fraudulent. There are 2 unique values for this field with no missing/null values. Following is the distribution of the two categories:

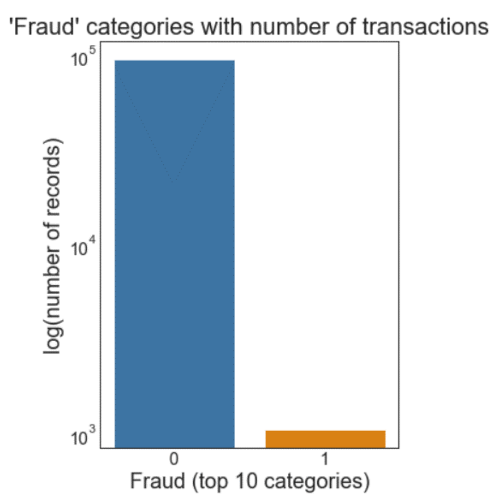


Fig 1.9 Categorical distribution of 'Fraud'

Data Cleaning

Filling Missing Value

- Filling Merchnum

  We first aggregate merchant number by merchant description and fill in the most common merchant number for each merchant description. Then we aggregate it by merchant zip and fill in the mode of merchant number for each merchant zip code. Lastly, we then aggregate it by merchant state and fill in the most common merchant number for each merchant state. After doing these, we have filled in all the missing merchant number.

- Filling Merch state

  First, we aggregate merchant state by merchant zip code and fill in the most common merchant state for each zip code. Second, we then aggregate by merchant number and fill in the most common merchant state for each merchant number. Lastly, we aggregate it by merchant description and did the same process as previous two steps. After doing these, we have filled in most of the missing values. However, there are still a few records contained missing merchant state. Consequently, we decided to fill "TN", which is the most common value of merchant state across the whole dataset.

- Filling Merch zip

  Similar to the approach mentioned above, we first group merchant zip by merchant number and merchant state at the same time and fill in the most common merchant zip for each merchant number and merchant state. We then group it by merchant description and merchant state, respectively, and fill in the mode of merchant zip for each merchant description and merchant state. After doing these, we still notice that there are some records with missing merchant zip code within the dataset. We decide to fill in 38118, which is the most common merchant zip code across the whole dataset.

Remove Outliers

- Remove the records with transaction types which are A, D and Y because the transactions with those transaction types are too low in quantity.

- Remove record number 52715 because this record has amount value more than 3,000,000, which is an outlier.

Candidate Variables

Create Variables Across Time

For each entity and combination group, we created the amount, frequency, days since and velocity variables to make the model more robust and invariant to seasonality.

- Amount Variables - 250

  We created variables based on the transaction amount. First, we calculated the mean, maximum, median, sum and count of amount for each card number or merchant number separately over five-time windows ([1, 3, 7, 14, 30] days) using rolling function. We created 50 amount variables in step 1. Further, we aggregated by card number with merchant number, merchant zip code and merchant state respectively to calculate the mean, maximum, median, sum and count of amount for each combination group using the same time window frame in step 1. This resulted in 75 variables. Further, we also created variables by using actual amount of each entity or combination group divided by the outcome we calculated for the entity or group resulting in 125 variables. Hence, we ended up with 250 amount variables.

- Frequency Variables - 30

  We then focused on each card number and merchant number and calculated how many times they occur over six-time windows [0, 1, 3, 7, 14, 30] days. Then we created three new combination by grouping card number with merchant number, merchant zip code and merchant state and calculated their frequency over the six-time windows mentioned previously. This resulted in 30 frequency variables.

- Days since variables - 5

  To create days-since variable, we calculated the number of days since we last saw a specific combination group or entity. We created 5 'Days since' variables for 2 entities and 3 combination groups. For example, group 'cardnum, merchnum' indicates how many days since a transaction has been processed with a unique combination of card number and merchant number. If the day of last saw is null, we will replace it with 365 days.

- Velocity variables - 96

  As for velocity variables, we first created a timeframe called lags = [7, 14, 30], and calculated the velocity that sum of amount or number of transactions for each card number or merchant number in past one or three days over the sum or count of amount for the card number or merchant number in past lags day, which represented the percentage of a card's short term amount in its long term amount. We created 96 velocity variables. For example, 'sum_Cardnum_1d/sum_Merchnum_7d' means the sum of amount of one card number in one day over the sum of amount of one merchant number in last 7 days

- Benford Law's Variables - 2

  We then employed Benford's Law, which states that the distribution of first digit of many measurements is not uniform. For instance, the first digit "1" appears about 30% of the time! If a person is making up transactions, often they are not aware of Benford's Law and the transaction amounts could be uniformly distributed random numbers. Hence, we can look at the amount distributions for each cardholder and merchant to see if the amount distributions substantially violate Benford's Law and flag them as unusual.

  Steps:

1) Variable Creation: Then, we created a new column which indicates if the amount for each record starts with 1 or 2. If the amount starts with 1 or 2, we flagged it as 0, else as 1

2) Then, we grouped the data by 'Merchnum' and 'Cardnum' separately to form two data sets. Also, counted the number of transactions starting with 1 or 2 as nlow and for the other digits (3-9) as nhigh using the column defined above for each of the groups in these two data sets.

   In short, we calculated the distribution of the first digit of the purchase amount for each group in the two groupings.

3) The expected ratio for Benford's Law is 1.96. For each Merchantnum and Cardnum group respectively, we then calculated the value of R, which is (1.096/ (nhigh/nlow)). During the above process we also ensured that values where nhigh and nlow = 0 are properly taken care off by replacing them with 1

4) After this we calculated U, the measure of unusualness for each merchant/card as the maximum of R and 1/R.

   To ensure, measure of unusualness doesn't get skewed when no. of transactions on a merchant/ card are not high enough, we chose the following smoothing parameters in the calculation of 'U*', which is a better measure of unusualness:

$$U^* = 1 + \left(\frac{U - 1}{1 + \exp^{-t}} \right)$$

$$t = (n - n_{mid})/c$$

   Based on expert opinion, we used nmid as 15 and c as 3

Finally, based on the above steps we ended up with 383 expert variables.

Feature Selection Process
Univariate Filter using KS and FDR

For each of our candidate variables, we calculate Kolmogorov–Smirnov (KS) score and Fraud Detection Rate (FDR) individually. Both the KS score and the FDR rate will help us determine how well candidate variables individually predict fraud, allowing us to rank order the variables in terms of usefulness for our models.

The KS score is a filter method that helps determine how well a candidate variable separates the goods from the bads, or in this case, frauds and not frauds. KS distance refers to the maximum of the difference of the cumulative fraud records and non-fraud records. The value of KS distance is always between 0 and 100. The higher the KS distance is, the more likely a variable is a good indicator of detecting fraud. For each variable, we will use the formula below to calculate a KS score and rank order the variables by the score.

$$KS = \max_{x} \sum_{x_{min}}^{x} [P_{\text{goods}} - P_{\text{bads}}]$$

FDR refers to the percentage of fraud records that each variable capture after sorted in descending or ascending order, which depends on which value is higher. The FDR for each variable is determined at a 3% level. It's the value representing the percentage of all frauds caught at a particular examination cutoff. For each variable, we will determine the percentage of frauds that are captured by the top 3% of the variable and rank order as such.

First, we divide the whole dataset into training, test and out of time sets. We set the records between '2010-01-15' and '2010-10-31' for application date to be the training and test to based on the records from '2010-10-31'.

Then, we calculate the Kolmogorov-Smirnov (KS) and Fraud Detection Rate (FDR) of each variable and rank them by KS and FDR respectively. After that, to select top ranked variables, we calculated the average rank of each variable and selected the top 80 variables by the average rank. The full table of the KS and FDR rank can be found in the appendix.

Recursive Feature Elimination and Cross-validated selection

The wrapper methods we chose were the recursive feature elimination and cross-validated selection. Recursive Feature Elimination (RFE) is a feature selection method that fits a model and removes the weakest feature until the specified number of features is reached. Features are ranked by the model's coefficients or feature importance attribute, followed by recursive elimination of a small number of features per loop. Cross validation is combined to select the best parameters for the RFE.
This method was implemented by using the RFECV function in the Scikit-learn package in Python. For the parameters, we used logistic regression as the estimator, with the settings "step" set to 1 and we set the

'Cross Validation' count as 3 which essentially splits the data into 3 parts and chooses 1 part as test set and the other two as the training sets.Based on this, we finally got a list of 25 variables on which we built our below models. The 25 variables are showed in Table 4.2.

	field	ks	FDR	rank_ks	rank_FDR	average_rank
0	Fraud	1.000000	1.000000	390.0	390.0	390.0
1	sum_Cardnum_Merch zip_7d	0.686335	0.640553	389.0	389.0	389.0
2	sum_Cardnum_Merchnum_7d	0.682346	0.639401	388.0	388.0	388.0
3	sum_Cardnum_Merchnum_14d	0.677904	0.632488	387.0	386.0	386.5
4	sum_Cardnum_Merch zip_14d	0.675991	0.635945	386.0	387.0	386.5
5	sum_Cardnum_Merch zip_3d	0.672995	0.623272	385.0	385.0	385.0
6	sum_Cardnum_Merch state_3d	0.672717	0.614055	384.0	383.0	383.5
7	sum_Cardnum_Merch state_7d	0.672126	0.607143	383.0	382.0	382.5
8	sum_Cardnum_Merchnum_3d	0.668004	0.616359	382.0	384.0	383.0
9	sum_Cardnum_Merch state_14d	0.667639	0.540323	381.0	373.0	377.0

Table 4.1 Top 10 variables ranked by KS and FDR

1	mean_Cardnum_Merchnum_7d
2	median_Cardnum_Merchnum_7d
3	sum_Cardnum_Merch state_7d
4	median_Cardnum_Merch state_1d
5	mean_Cardnum_Merch state_1d
6	mean_Cardnum_Merch zip_7d
7	median_Cardnum_Merch zip_7d
8	max_Merchnum_3d
9	sum_Merchnum_3d
10	max_Cardnum_Merchnum_30d
11	max_Cardnum_Merchnum_14d
12	mean_Cardnum_Merchnum_30d
13	mean_Cardnum_Merch zip_14d
14	sum_Cardnum_Merchnum_30d
15	sum_Cardnum_Merchnum_14d
16	sum_Cardnum_Merch zip_14d
17	median_Merchnum_1d
18	mean_Merchnum_1d
19	mean_Cardnum_1d
20	max_Cardnum_1d
21	median_Cardnum_1d
22	sum_Cardnum_1d
23	sum_Cardnum_Merchnum_7d
24	max_Cardnum_Merchnum_3d
25	max_Cardnum_Merch zip_3d

Table 4.2 25 variables selected by RFECV

Models

Logistic Regression

The goal of multiple logistic regression is to predict the likelihood of the target variable (Y) using multiple variables (X). Using the concept of least squares method, the model optimizes the coefficients for each of the predictor variables.

We ran a logistic regression using different combinations of our identified 25 wrapper variables. The model's fraud detection rate at 3% threshold was 59% for training, 59% for testing and 31% for the holdout sample. This model would serve as our baseline for to improve upon with more advanced algorithms.

Random Forest

In random forests, when building these decision trees, each time a split in a tree is considered, a random sample of predictors is chosen as split candidates from the full set of predictors. The number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

In other words, in building a random forest, at each split in the tree, the algorithm is not allowed to consider most of the available predictors. Random forests considers a subset of predictors and this helps to reduce the effect of highly correlated predictors. On a long run, this will help to reduce variance when we take average of predicted values.

We used the RandomForestClassifier package from the library sklearn to make the Random Forest model on our reduced set of variables. We varied the number of estimators i.e. no. of trees and then we trained our model on training data. Further, we predicted the probability of Fraud over training, test and OOT (validation data).

Our model's top performance occurred with the number of trees 100 and maximum depth 10. The model's fraud detection rate at 3% threshold was 65% for training, 59% for testing and 58% for the holdout sample. We have added our top performing models in the appendix.

Boosted Trees

Boosted trees is another approach for improving the predictions resulting from a decision tree. Boosting can be applied to many statistical models for regression and classification. In boosting, trees are grown sequentially, with each tree grown using information from previously grown trees. Each tree is fit on a modified version of the original data set, with each boost learning slowly. This approach is different than fitting a single large decision tree to the data, which results in fitting the data hard and potentially overfitting.

Given the current model, we fit the decision tree to the residuals from the model. That is, we fit a tree using the current residuals, rather than the outcome Y, as the response. We then added this new decision tree into the fitted function in order to update the residuals. By fitting small trees to the residuals, we slowly improved. In general, statistical learning approaches that learn slowly tend to perform well. In boosting, the construction of each tree depends strongly on the trees that have already been grown. In summary, the boosted trees approach combines many simple models in a linear fashion, creating a series of weak learners. The linear combinations of all the simple models create a strong learner.

Our model's top performance occurred with the number of trees 100, max depth 5 and learning rate 0.02. The model's fraud detection rate at 3% threshold was 76% for training, 78% for testing and 58% for the holdout sample. Our Boosted Trees model was our top performing model, boasting an OOT accuracy of 58%. We have added our top performing models in the appendix.

Neural Network

Neural Net is a type of machine learning designed to recognize patterns. The neural net was inspired by the biological neural networks that constitutes animal brains. The typical neural net consists of an input layer, some number of hidden layers and an output layer. A neural net with more than one hidden layer is a deep learning neural net. Deep learning is a neural net architecture. With deep learning, the computer trains itself to process and learn from data instead of teaching computers to process and learn from data (which is how machine learning works).

Each node in the hidden layer receives weighted signals from all the nodes in the incoming layer and does a transformation on this linear combination of signals. The transform/activation function can be one of a number of functions, for example a logistic function (sigmoid). To obtain a more robust understanding of the model's performance, we trained the network six times, tuning a combination of various parameters into it for each run.

Our model's top performance occurred with one hidden layers of size 15, learning rate of 0.0001 and 20 iterations. The model's fraud detection rate at 3% threshold was 39% for training, 48% for testing and 29% for the holdout sample. We have added our top performing models in the appendix.

Result

Our best performing algorithm is Boosted Trees model with 100 trees, 5 depth and 0.02 learning rate. We have generated cumulative Good, Bads, % Good, % Bad (FDR), KS and FPR for all three populations (training, testing, and Validation (OOT), and the fraud savings plot. We have listed the top 20 batches for each set of data. The complete list can be found in the appendix.

1) Training Data

Training	# Records	# Goods	# Bads	Fraud Rate
	60,474	59,818	656	1.08%

	Bin Statistics					Cumulative Statistics						
Population Bin %	Records	Goods	Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads	KS	FPR
0	605	162	443	26.78%	73.22%	605	162	443	0.27%	67.53%	67.26%	0.37
1	605	569	36	94.05%	5.95%	1210	731	479	1.22%	73.02%	71.80%	1.53
2	605	566	39	93.55%	6.45%	1815	1297	518	2.17%	78.96%	76.80%	2.50
3	605	589	16	97.36%	2.64%	2420	1886	534	3.15%	81.40%	78.25%	3.53
4	605	598	7	98.84%	1.16%	3025	2484	541	4.15%	82.47%	78.32%	4.59
5	605	601	4	99.34%	0.66%	3630	3085	545	5.16%	83.08%	77.92%	5.66
6	605	604	1	99.83%	0.17%	4235	3689	546	6.17%	83.23%	77.06%	6.76
7	605	596	9	98.51%	1.49%	4840	4285	555	7.16%	84.60%	77.44%	7.72
8	605	595	10	98.35%	1.65%	5445	4880	565	8.16%	86.13%	77.97%	8.64
9	605	603	2	99.67%	0.33%	6050	5483	567	9.17%	86.43%	77.27%	9.67
10	605	598	7	98.84%	1.16%	6655	6081	574	10.17%	87.50%	77.33%	10.59
11	605	602	3	99.50%	0.50%	7260	6683	577	11.17%	87.96%	76.79%	11.58
12	605	601	4	99.34%	0.66%	7865	7284	581	12.18%	88.57%	76.39%	12.54
13	605	605	0	100.00%	0.00%	8470	7889	581	13.19%	88.57%	75.38%	13.58
14	605	601	4	99.34%	0.66%	9075	8490	585	14.19%	89.18%	74.98%	14.51
15	605	597	8	98.68%	1.32%	9680	9087	593	15.19%	90.40%	75.21%	15.32
16	605	603	2	99.67%	0.33%	10285	9690	595	16.20%	90.70%	74.50%	16.29
17	605	603	2	99.67%	0.33%	10890	10293	597	17.21%	91.01%	73.80%	17.24
18	605	602	3	99.50%	0.50%	11495	10895	600	18.21%	91.46%	73.25%	18.16
19	605	602	3	99.50%	0.50%	12100	11497	603	19.22%	91.92%	72.70%	19.07
20	605	603	2	99.67%	0.33%	12705	12100	605	20.23%	92.23%	72.00%	20.00

2) Test Data

Test	# Records	# Goods	# Bads	Fraud Rate
	20,158	19,946	212	1.05%

	Bin Statistics					Cumulative Statistics						
Population Bin %	Records	Goods	Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads	KS	FPR
0	202	60	142	29.70%	70.30%	202	60	142	0.30%	66.98%	66.68%	0.42
1	202	182	20	90.10%	9.90%	404	242	162	1.21%	76.42%	75.20%	1.49
2	202	196	6	97.03%	2.97%	606	438	168	2.20%	79.25%	77.05%	2.61
3	202	198	4	98.02%	1.98%	808	636	172	3.19%	81.13%	77.94%	3.70
4	202	201	1	99.50%	0.50%	1010	837	173	4.20%	81.60%	77.41%	4.84
5	202	200	2	99.01%	0.99%	1212	1037	175	5.20%	82.55%	77.35%	5.93
6	202	202	0	100.00%	0.00%	1414	1239	175	6.21%	82.55%	76.34%	7.08
7	202	200	2	99.01%	0.99%	1616	1439	177	7.21%	83.49%	76.28%	8.13
8	202	202	0	100.00%	0.00%	1818	1641	177	8.23%	83.49%	75.26%	9.27
9	202	199	3	98.51%	1.49%	2020	1840	180	9.22%	84.91%	75.68%	10.22
10	202	200	2	99.01%	0.99%	2222	2040	182	10.23%	85.85%	75.62%	11.21
11	202	201	1	99.50%	0.50%	2424	2241	183	11.24%	86.32%	75.09%	12.25
12	202	202	0	100.00%	0.00%	2626	2443	183	12.25%	86.32%	74.07%	13.35
13	202	201	1	99.50%	0.50%	2828	2644	184	13.26%	86.79%	73.54%	14.37
14	202	201	1	99.50%	0.50%	3030	2845	185	14.26%	87.26%	73.00%	15.38
15	202	201	1	99.50%	0.50%	3232	3046	186	15.27%	87.74%	72.46%	16.38
16	202	202	0	100.00%	0.00%	3434	3248	186	16.28%	87.74%	71.45%	17.46
17	202	202	0	100.00%	0.00%	3636	3450	186	17.30%	87.74%	70.44%	18.55
18	202	201	1	99.50%	0.50%	3838	3651	187	18.30%	88.21%	69.90%	19.52
19	202	200	2	99.01%	0.99%	4040	3851	189	19.31%	89.15%	69.84%	20.38
20	202	201	1	99.50%	0.50%	4242	4052	190	20.31%	89.62%	69.31%	21.33

3) OOT Data

OOT	# Records	# Goods	# Bads	Fraud Rate
	12,427	12,248	179	1.44%

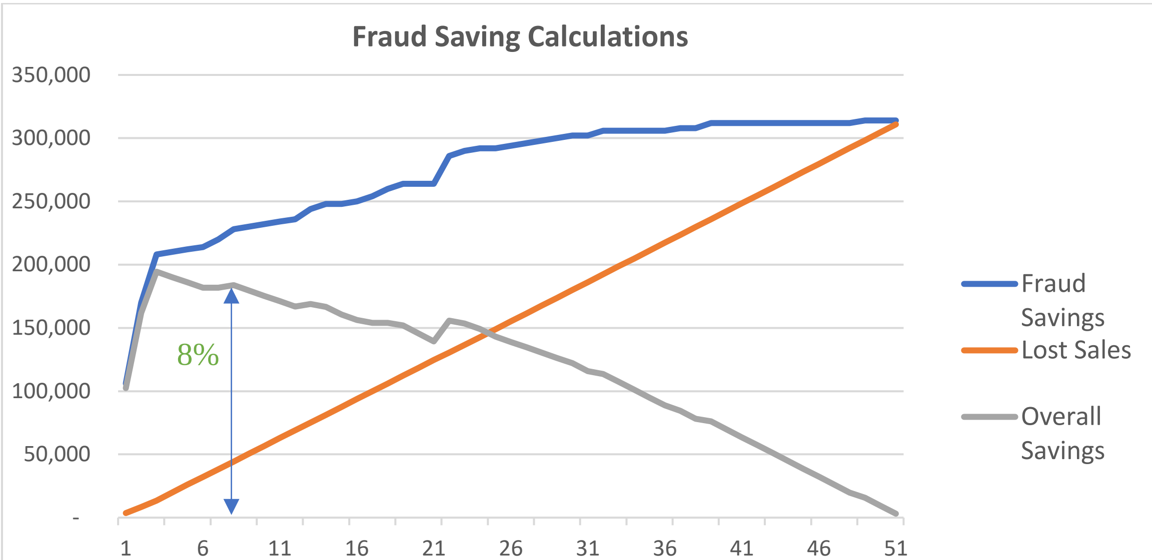
	Bin Statistics					Cumulative Statistics						
Population Bin %	**Records**	**Goods**	**Bads**	**% Goods**	**% Bads**	**Total # Records**	**Cumulative Goods**	**Cumulative Bads**	**% Goods**	**% Bads**	**KS**	**FPR**
0	125	72	53	57.60%	42.40%	125	72	53	0.59%	29.61%	29.02%	1.36
1	125	93	32	74.40%	25.60%	250	165	85	1.35%	47.49%	46.14%	1.94
2	125	106	19	84.80%	15.20%	375	271	104	2.21%	58.10%	55.89%	2.61
3	125	124	1	99.20%	0.80%	500	395	105	3.23%	58.66%	55.43%	3.76
4	125	124	1	99.20%	0.80%	625	519	106	4.24%	59.22%	54.98%	4.90
5	125	124	1	99.20%	0.80%	750	643	107	5.25%	59.78%	54.53%	6.01
6	125	122	3	97.60%	2.40%	875	765	110	6.25%	61.45%	55.21%	6.95
7	125	121	4	96.80%	3.20%	1,000	886	114	7.23%	63.69%	56.45%	7.77
8	125	124	1	99.20%	0.80%	1,125	1,010	115	8.25%	64.25%	56.00%	8.78
9	125	124	1	99.20%	0.80%	1,250	1,134	116	9.26%	64.80%	55.55%	9.78
10	125	124	1	99.20%	0.80%	1,375	1,258	117	10.27%	65.36%	55.09%	10.75
11	125	124	1	99.20%	0.80%	1,500	1,382	118	11.28%	65.92%	54.64%	11.71
12	125	121	4	96.80%	3.20%	1,625	1,503	122	12.27%	68.16%	55.89%	12.32
13	125	123	2	98.40%	1.60%	1,750	1,626	124	13.28%	69.27%	56.00%	13.11
14	125	125	-	100.00%	0.00%	1,875	1,751	124	14.30%	69.27%	54.98%	14.12
15	125	124	1	99.20%	0.80%	2,000	1,875	125	15.31%	69.83%	54.52%	15.00
16	125	123	2	98.40%	1.60%	2,125	1,998	127	16.31%	70.95%	54.64%	15.73
17	125	122	3	97.60%	2.40%	2,250	2,120	130	17.31%	72.63%	55.32%	16.31
18	125	123	2	98.40%	1.60%	2,375	2,243	132	18.31%	73.74%	55.43%	16.99
19	125	125	-	100.00%	0.00%	2,500	2,368	132	19.33%	73.74%	54.41%	17.94
20	125	125	-	100.00%	0.00%	2,625	2,493	132	20.35%	73.74%	53.39%	18.89

4) Fraud Saving Table and Plot for the OOT (Validation Data)

After obtaining the best model, we calculated the savings by multiplying number of bads in each bin with \$2,000 and number of goods in each bin with \$50. We give more weightage to capturing bad as it would lead to potential losses for the company.

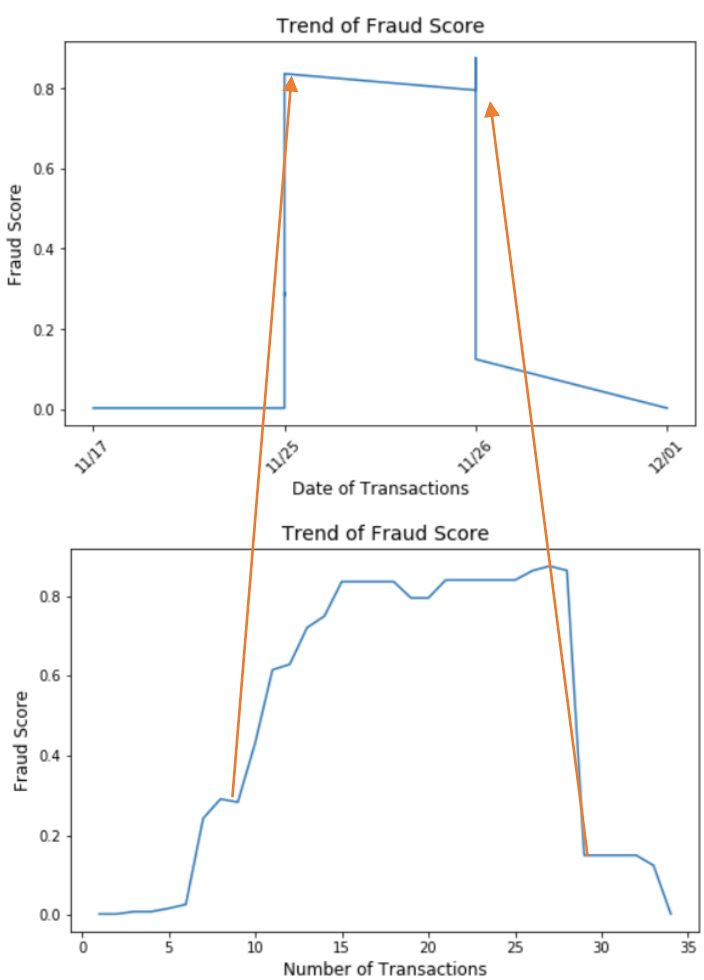
Population Bin %	**Records**	**Goods**	**Bads**	**Fraud Savings**	**Lost Sales**	**Overall Savings**
0	125	72	53	106,000	3,600	102,400
1	125	93	32	170,000	8,250	161,750
2	125	106	19	208,000	13,550	194,450
3	125	124	1	210,000	19,750	190,250
4	125	124	1	212,000	25,950	186,050
5	125	124	1	214,000	32,150	181,850
6	125	122	3	220,000	38,250	181,750
7	125	121	4	228,000	44,300	183,700
8	125	124	1	230,000	50,500	**179,500**
9	125	124	1	232,000	56,700	175,300
10	125	124	1	234,000	62,900	171,100
11	125	124	1	236,000	69,100	166,900
12	125	121	4	244,000	75,150	168,850
13	125	123	2	248,000	81,300	166,700
14	125	125	-	248,000	87,550	160,450
15	125	124	1	250,000	93,750	156,250
16	125	123	2	254,000	99,900	154,100
17	125	122	3	260,000	106,000	154,000
18	125	123	2	264,000	112,150	151,850
19	125	125	-	264,000	118,400	145,600
20	125	125	-	264,000	124,650	139,350

The following plot shows the variation in fraud savings in $ as we move over each bin in the OOT data. Based on our model we would recommend cut-off score of 8% to maximize savings.



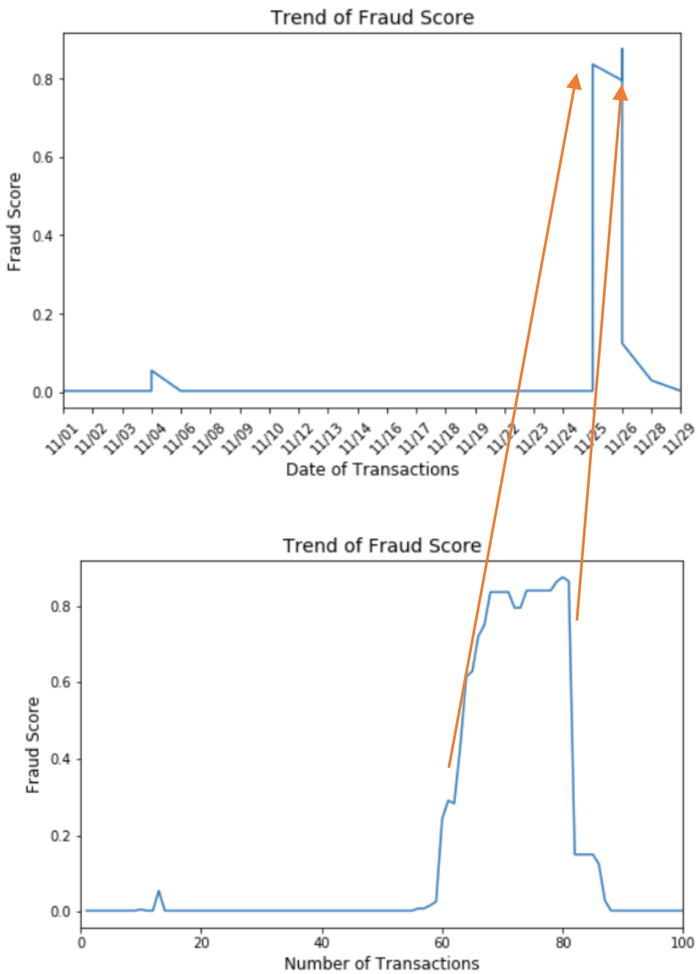
5) Dynamic plots for Entities

We further observed how the fraud score changed with number of applications and time period for credit card number 5142235211.



- This credit card had 32 transactions within 2 days (11/25-11/26)
- We can observe from the fraud score trend of credit card number 5142235211 that there is a sharp increase in fraud score with high number of transactions within a few days

We saw a similar trend for merchant number 4353000719908.



- This merchant had 32 transactions within 2 days (11/25-11/26)
- We can observe from the fraud score trend of merchant number '4353000719908' that there is a sharp increase in fraud score with high number of transactions within a few days

Conclusions

Conclusions

Credit card fraud is one of the most common identity frauds which costs economies billions of dollars. In this report, we have examined the dataset to draw the following conclusion.

Comparing all the above models, we can conclude that Gradient Boosted Trees performed the best. The FDR on training dataset is 76%, 78% on test set and 58% on the validation dataset. We used supervised algorithms including logistic regression, Random Forest, Gradient Boosted Trees and Neural Nets.

Based on our model we would recommend cut-off score of 8% on out of sample set to maximize savings.

Potential Improvements

We trained our models by training, testing and validating with the original dataset, which had only 1.09% of potential fraudulent records. In our perspective, weighting a dataset can improve the model accuracy. Also, as fraud datafiles are imbalanced, we can choose to scramble the goods or unscramble the bads to increase the model accuracy.

Gains in FDR can be achieved with the addition of external datasets related to credit card transactions. For example, more legitimate like credit card scores of the person could make it much easier to identify algorithmically whether or not someone is using falsified information in their application. Similarly, a collection of addresses and the last name of the owner could potentially lead to greater accuracy if utilized correctly.

Adding additional variables or information related to the interactions between variables in the dataset could potentially help increase FDR in the future.

Appendix

Full Data Quality Report

Cardnum (Categorical, 10-digit code)

This categorical variable defines the credit card number of each transaction. There are 1,645 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

Cardnum	Count
5142148452	1192
5142184598	921
5142189108	663
5142297710	583
5142223373	579
5142187452	526
5142299634	515
5142189945	512
5142149691	497
5142190147	488

Table 1.3 Top values of 'Cardnum'

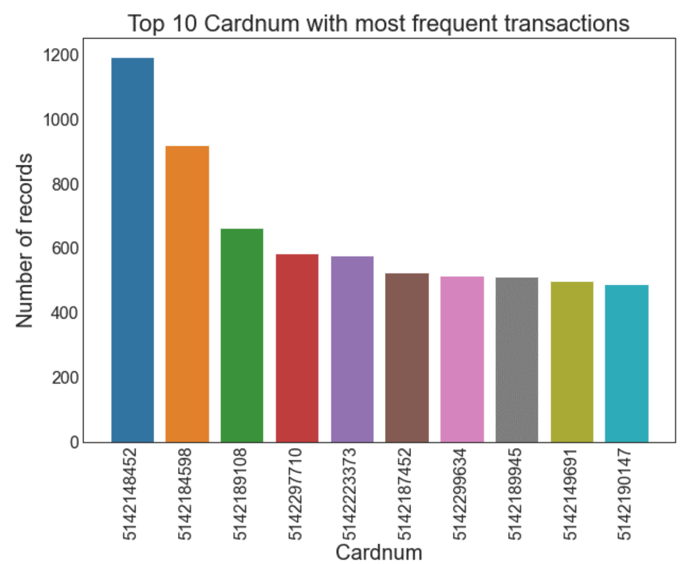


Fig 1.1 Categorical distribution of 'Cardnum

Date (Categorical, datetime)

This variable defines date for each transaction in the data. There are 365 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

Date	Count
2/28/2010	684
8/10/2010	610
3/15/2010	594
9/13/2010	564
9/7/2010	536
8/9/2010	536
9/14/2010	533
9/21/2010	522
8/1/2010	521

8/31/2010	518

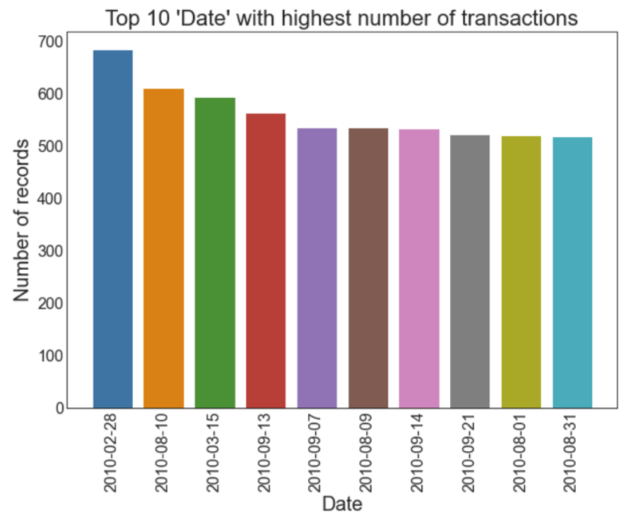


Table 1.4 Top values of 'Date'

Merchnum (Categorical, 13-digit code)

This categorical variable defines the merchant number of each transaction/record. There are 13,091 unique values for this field with 3,375 (3.5%) missing/null values. Following is a distribution of the top 10 categories:

Merchnum	count
930090121224	9310
5509006296254	2131
9900020006406	1714
602608969534	1092
4353000719908	1020
410000971343	982
9918000409955	956
5725000466504	872
9108234610000	817
602608969138	783

Table 1.5 Top values of 'Merchnum'

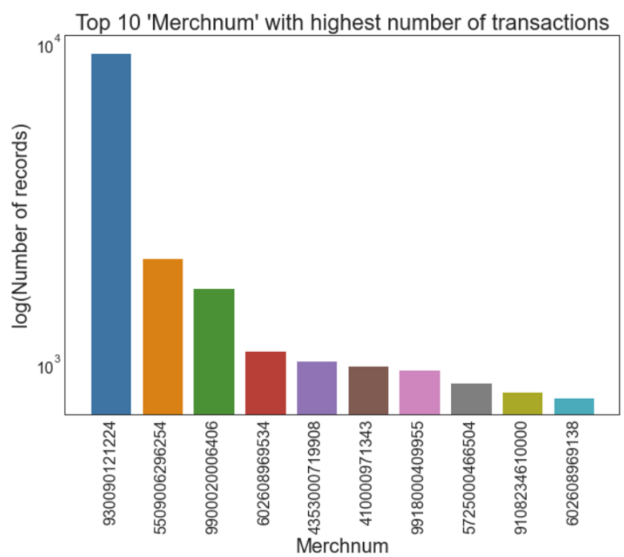
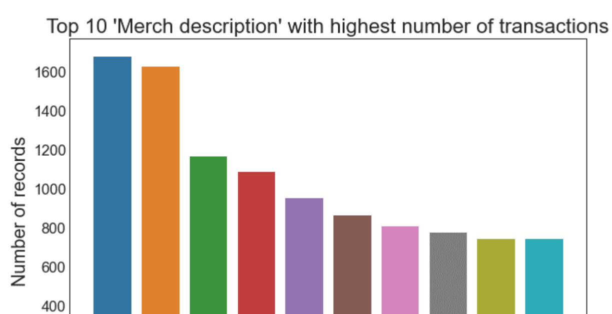


Fig 1.3 Categorical distribution of 'Merchnum'

Merch description (Categorical, string)

This categorical variable defines the merchant description, indicating the location of each merchant. There are 13,126 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

Merch description	Count



GSA-FSS-ADV	1688
SIGMA-ALDRICH	1635
STAPLES #941	1174
FISHER SCI ATL	1093
MWI*MICRO WAREHOUSE	958
CDW*GOVERNMENT INC	872
DELL MARKETING L.P.	816
FISHER SCI CHI	783
AMAZON.COM *SUPERSTOR	750
OFFICE DEPOT #1082	748

Table 1.6 Top values of 'Merch description'

Fig 1.4 Categorical distribution of 'Merch description'

Merch state (Categorical, string)

This categorical variable defines the state of the merchant, indicating location of the merchant. There are 227 unique values for this field with 1195 (1.2%) missing/null values. Following is a distribution of the top 10 categories:

Merch state	Count
TN	12035
VA	7872
CA	6817
IL	6508
MD	5398
GA	5025
PA	4899
NJ	3912
TX	3790
NC	3322

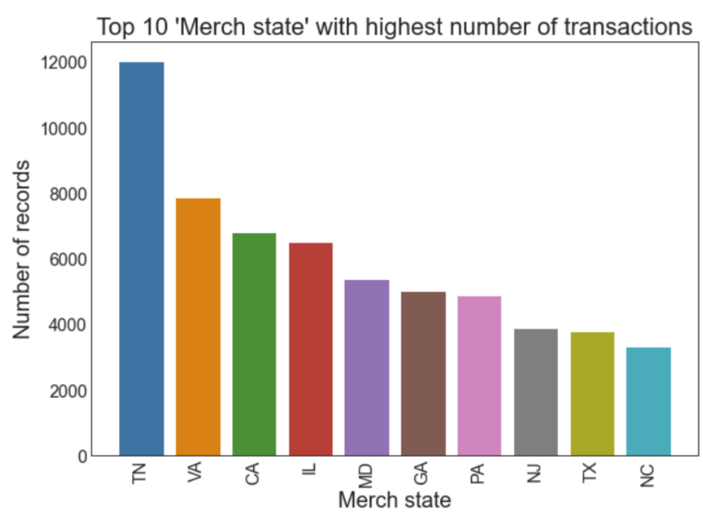


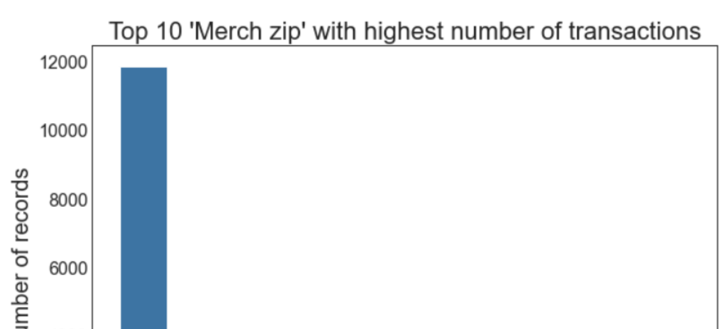
Table 1.7 Top values of 'Merch state'

Fig 1.5 Categorical distribution of 'Merch state'

Merch zip (Categorical, 5-digit code)

This categorical variable defines the zip code of the merchant. There are 4,567 unique values for this field with 4,656 (4.8%) missing/null values. Following is a distribution of the top 10 categories:

Merch zip	Count
38118	11868



63103	1650
8701	1267
22202	1250
60061	1221
98101	1197
17201	1180
30091	1092
60143	942
60069	826

Table 1.8 Top values of ‘Merch zip’

Fig 1.6 Categorical distribution of ‘Merch zip’

Transtype (Categorical, single letter)

This categorical variable defines the 5-digit zip code of the applicant for each record/row. There are 4 unique values for this field with no missing/null values. Following is a distribution of the top 10 categories:

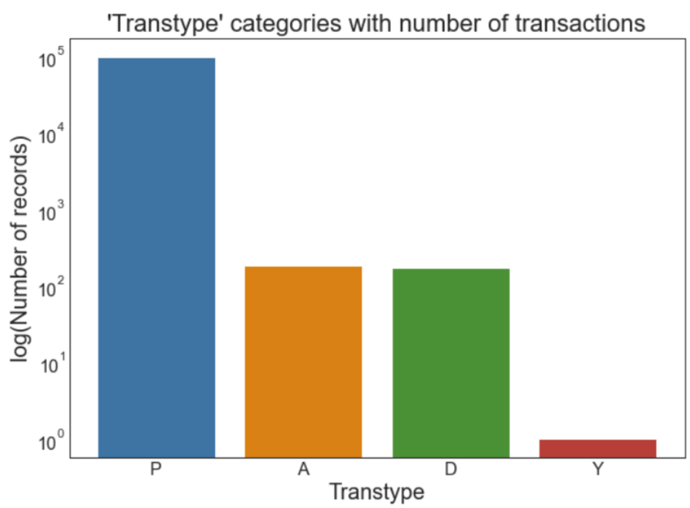


Fig 1.7 Categorical distribution of ‘Transtype’

Amount (Categorical, float)

This numerical variable defines the amount of each transaction. There are no missing/null values. Following graph shows the distribution of the ‘Amount’ variable:

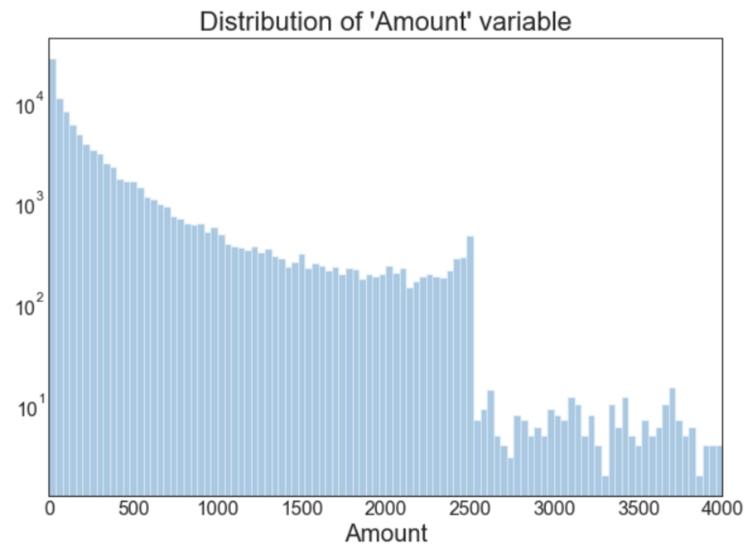


Fig 1.8 Distribution of ‘amount’

Fraud (Categorical, 1-digit-code)

This categorical variable indicates if the transaction is fraud or not. 1 indicates that it is a fraudulent transaction while 0 indicates that it is not fraudulent. There are 2 unique values for this field with no missing/null values. Following is the distribution of the two categories:

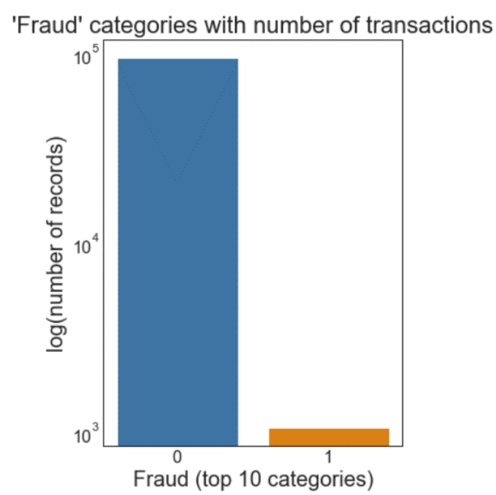


Fig 1.9 Categorical distribution of ‘Fraud’

Expert Variables Ranked by KS and FDR

	field	ks	FDR	rank_ ks	rank_ FDR	average_ rank
1	Fraud	1	1	390	390	390

2	sum_Cardnum_Merch zip_7d	0.686335	0.640553	389	389	389
3	sum_Cardnum_Merchnum_7d	0.682346	0.639401	388	388	388
4	sum_Cardnum_Merchnum_14d	0.677904	0.632488	387	386	386.5
5	sum_Cardnum_Merch zip_14d	0.675991	0.635945	386	387	386.5
6	sum_Cardnum_Merch zip_3d	0.672995	0.623272	385	385	385
7	sum_Cardnum_Merch state_3d	0.672717	0.614055	384	383	383.5
8	sum_Cardnum_Merch state_7d	0.672126	0.607143	383	382	382.5
9	sum_Cardnum_Merchnum_3d	0.668004	0.616359	382	384	383
10	sum_Cardnum_Merch state_14d	0.667639	0.540323	381	373	377
11	sum_Cardnum_Merch zip_30d	0.659735	0.554147	380	376	378
12	max_Cardnum_Merch zip_14d	0.65841	0.476959	379	365.5	372.25
13	sum_Cardnum_Merchnum_30d	0.658091	0.563364	378	381	379.5
14	max_Cardnum_Merch zip_7d	0.656611	0.461982	377	359	368
15	max_Cardnum_Merchnum_14d	0.656025	0.474654	376	363.5	369.75
16	max_Cardnum_Merchnum_30d	0.651938	0.473502	375	362	368.5
17	max_Cardnum_Merch zip_30d	0.651765	0.483871	374	368.5	371.25
18	max_Cardnum_Merch state_7d	0.651755	0.483871	373	368.5	370.75
19	max_Cardnum_Merchnum_7d	0.651718	0.460829	372	358	365
20	max_Cardnum_Merch state_3d	0.643867	0.457373	371	357	364
21	max_Cardnum_Merch zip_3d	0.639804	0.467742	370	360.5	365.25
22	max_Cardnum_Merchnum_3d	0.635849	0.467742	369	360.5	364.75
23	sum_Cardnum_Merch state_30d	0.634864	0.451613	368	356	362
24	max_Cardnum_Merch state_14d	0.633711	0.486175	367	370	368.5
25	sum_Cardnum_Merch zip_1d	0.6136	0.557604	366	378.5	372.25

26	sum_Cardnum_Merchnum_1d	0.611473	0.559908	365	380	372.5
27	sum_Cardnum_Merch state_1d	0.611185	0.557604	364	378.5	371.25
28	sum_Cardnum_7d	0.606535	0.525346	363	372	367.5
29	max_Cardnum_Merch zip_1d	0.605277	0.415899	362	347.5	354.75
30	max_Cardnum_Merch state_1d	0.604568	0.417051	361	349	355
31	max_Cardnum_Merchnum_1d	0.600902	0.415899	360	347.5	353.75
32	max_Merchnum_1d	0.600647	0.43318	359	354	356.5
33	max_Cardnum_Merch state_30d	0.599058	0.476959	358	365.5	361.75
34	mean_Cardnum_Merch zip_30d	0.598974	0.293779	357	300.5	328.75
35	mean_Cardnum_Merch zip_14d	0.596294	0.292627	356	297.5	326.75
36	mean_Cardnum_Merch state_7d	0.595144	0.308756	355	316.5	335.75
37	sum_Merchnum_3d	0.595092	0.419355	354	350	352
38	sum_Cardnum_3d	0.594878	0.556452	353	377	365
39	mean_Cardnum_Merch state_3d	0.592397	0.3053	352	314	333
40	mean_Cardnum_Merchnum_30d	0.592073	0.291475	351	295.5	323.25
41	mean_Cardnum_Merchnum_14d	0.591374	0.292627	350	297.5	323.75
42	mean_Cardnum_Merchnum_7d	0.590406	0.293779	349	300.5	324.75
43	mean_Cardnum_Merch zip_7d	0.589149	0.296083	348	304	326
44	mean_Merchnum_1d	0.585205	0.301843	347	312	329.5
45	max_Cardnum_1d	0.585192	0.425115	346	352	349
46	mean_Cardnum_Merch zip_3d	0.583752	0.301843	345	312	328.5
47	mean_Cardnum_Merchnum_3d	0.583001	0.300691	344	309.5	326.75
48	mean_Cardnum_Merch state_14d	0.577755	0.309908	343	318	330.5

49	mean_Merchnum_3d	0.575506	0.298387	342	306	324
50	mean_Cardnum_Merch state_1d	0.575412	0.321429	341	330	335.5
51	mean_Cardnum_3d	0.574355	0.362903	340	337	338.5
52	mean_Cardnum_Merchnum_1d	0.573662	0.319124	339	327	333
53	mean_Cardnum_Merch zip_1d	0.573385	0.319124	338	327	332.5
54	max_Cardnum_3d	0.573021	0.440092	337	355	346
55	median_Cardnum_Merchnum_30d	0.572801	0.27765	336	283.5	309.75
56	sum_Cardnum_1d	0.57088	0.551843	335	375	355
57	sum_Merchnum_1d	0.570489	0.544931	334	374	354
58	median_Cardnum_Merch state_3d	0.570457	0.294931	333	303	318
59	mean_Cardnum_1d	0.570021	0.328341	332	335	333.5
60	median_Cardnum_Merchnum_3d	0.569145	0.291475	331	295.5	313.25
61	median_Cardnum_Merch zip_3d	0.568927	0.293779	330	300.5	315.25
62	mean_Cardnum_Merch state_30d	0.5669	0.326037	329	333	331
63	median_Cardnum_Merch state_1d	0.565641	0.300691	328	309.5	318.75
64	median_Cardnum_Merchnum_1d	0.5636	0.299539	327	307.5	317.25
65	median_Cardnum_Merch zip_1d	0.562536	0.299539	326	307.5	316.75
66	median_Cardnum_Merch zip_30d	0.561298	0.276498	325	282	303.5
67	max_Merchnum_3d	0.559018	0.421659	324	351	337.5
68	median_Cardnum_Merchnum_14d	0.55894	0.278802	323	286	304.5
69	sum_Merchnum_7d	0.558549	0.387097	322	341	331.5
70	sum_Cardnum_14d	0.557788	0.474654	321	363.5	342.25
71	median_Cardnum_1d	0.557773	0.308756	320	316.5	318.25

72	median_Cardnum_Merchnum_7d	0.556298	0.279954	319	288.5	303.75
73	max_Merchnum_7d	0.554619	0.37788	318	338	328
74	median_Cardnum_Merch zip_7d	0.553747	0.282258	317	291.5	304.25
75	median_Cardnum_Merch zip_14d	0.553673	0.278802	316	286	301
76	median_Cardnum_3d	0.553661	0.323733	315	331.5	323.25
77	median_Cardnum_Merch state_7d	0.553313	0.281106	314	290	302
78	median_Merchnum_1d	0.551772	0.284562	313	293	303
79	max_Cardnum_7d	0.551698	0.480415	312	367	339.5
80	median_Cardnum_Merch state_30d	0.550062	0.297235	311	305	308

Random Forest Result

	Parameter		Average FDR		
Rank by OOT FDR	**Number of trees**	**Max depth**	**Train**	**Test**	**OOT**
1	100	10	0.65	0.59	0.58
2	500	10	0.64	0.59	0.58
3	50	10	0.62	0.57	0.57
4	100	15	0.8	0.62	0.57
5	200	10	0.62	0.57	0.56
6	50	15	0.79	0.64	0.55
7	500	15	0.79	0.64	0.54
8	100	20	0.91	0.64	0.53
9	200	15	0.8	0.63	0.53
10	500	20	0.91	0.64	0.53

Boosted Trees Result

	Parameters	Average FDR

Rank by OOT FDR	Number of trees	Max depth	Learning rate	Train	Test	OOT
1	100	5	0.02	0.76	0.78	0.58
2	50	5	0.05	0.76	0.78	0.57
3	100	5	0.01	0.74	0.77	0.56
4	50	5	0.02	0.75	0.78	0.55
5	100	7	0.02	0.83	0.8	0.54
6	50	5	0.01	0.75	0.77	0.54
7	50	7	0.05	0.85	0.81	0.54
8	100	10	0.02	0.91	0.83	0.53
9	50	7	0.02	0.8	0.8	0.53
10	100	7	0.01	0.81	0.8	0.52

Neural Net Result

	Parameters			Average FDR		
Rank by OOT FDR	Hidden size	Alpha	Epochs	Train	Test	OOT
1	15	0.0001	20	0.39	0.48	0.29
2	15	0.0005	20	0.39	0.48	0.29
3	15	0.001	20	0.39	0.48	0.29
4	15	0.005	20	0.39	0.48	0.29
5	15	0.01	20	0.39	0.48	0.29
6	15	0.02	20	0.39	0.48	0.29
7	15	0.05	20	0.39	0.48	0.29
8	15	0.1	20	0.39	0.48	0.29
9	15	0.5	20	0.39	0.48	0.29
10	15	0.0001	10	0.4	0.47	0.24