# Credit Card Transaction Detection

Image from Stockvault via link

Group 5

Fangyu Lo, MSBA

Qing Gong, MSBA

Biyun Jing, MSBA

Chenyue Wang, MSBA

Ryan DiBenedetto, MSBA

June 10th, 2021

Project Advisor:
Professor Stephen Coggeshall

# Table of Contents

# Executive Summary

Credit card transaction fraud is a common type of financial fraud during the account usage process. It can happen through a variety of ways: theft of physical credit cards or numbers and PINs intercepted credit card information by dishonest waiters, information leak when offered fake travel packages, and hacked websites with critical account information. Credit card issuers are aware of these frauds and keep developing better methods to supervise credit card transactions.

This project will be focusing on building supervised learning models on the card transaction data to predict fraud in the credit card transaction process. The overall goal of the project is to utilize different machine learning algorithms to predict and identify fraudulent transactions effectively and efficiently.

This report includes the complete development process of fraud detection machine learning algorithms. The credit card transaction dataset was cleaned by removing exclusions and filling in missing values. Moreover, new features were created based on existing data entities to better capture suspicious card transactions and train machine learning models. Next, we ran a filter to reduce the number of candidates features to 80 and then applied a wrapper to get the number down to 30. After feature selection, we got a list of ranked-ordered features by multivariate importance. Later, we trained and tested different machine learning models to discover the best-performing model. We then showed the results of the final algorithm and hyperparameter selection. Last but not least, conclusions were made to summarize the model and make recommendations for the future fraud identification process.

Our team decided to apply different machine learning algorithms to fit the data, including Logistic Regression, Random Forest, Boosted Tree, and Neural Network. We achieved an FDR@3% of 0.674, which means our fraud algorithm is capable of removing up to 2/3 of fraudulent transactions, saving an estimated $473,450.

# 1. Data Description

## 1.1 Dataset Description

The "card transactions.csv" is a dataset that contains actual credit card purchases from a US government organization. Each entry of the data has a label indicating whether the transaction is a fraud. The purpose of this dataset is to detect potential credit card transaction fraud. For more information on the data, please see Appendix I for reference.

- Name: Credit Card Transaction Data
- Purpose: Actual credit card purchases from a US government organization to look for credit card transaction fraud
- Source: US government organization
- Time period: January 1st, 2010 to December 31st, 2010
- Number of fields: 10
- Number of records: 96,753

## 1.2 Data Summary

The variables in the dataset contain both categorical variables and numeric variables. We summarized the categorical data in Table 1.2.1 and the numeric variables in Table 1.2.2.

Table 1.2.1 Categorical Variable

| Item | Column name | # of records | % populated | Unique values | Most common field value |
|---|---|---|---|---|---|
| 1 | Recnum | 96,753 | 100.00% | 96,753 | N/A |
| 2 | Cardnum | 96,753 | 100.00% | 1,645 | 5142148452 |
| 3 | Date | 96,753 | 100.00% | 365 | 2/28/10 |
| 4 | Merchnum | 93,378 | 96.51% | 13,091 | 930090121224 |
| 5 | Merch description | 96,753 | 100.00% | 13,126 | GSA-FSS-ADV |
| 6 | Merch state | 95,558 | 98.76% | 227 | TN |
| 7 | Merch zip | 92,097 | 95.19% | 4,567 | 38118 |
| 8 | Transtype | 96,753 | 100.00% | 4 | P |
| 9 | Fraud | 96,753 | 100.00% | 2 | 0 |

Table 1.2.2 Numeric Variable

| Item | Column name | # of records | % populated | Unique values | Mean | Standard deviation | Minimum value | Maximum value | # of zeros |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Amount | 96,753 | 100.00% | 34,909 | 427.89 | 10006.14 | 0.01 | 3,102,045.53 | 0 |

The transactions over time in the data are presented in Figure 1.2.1 and Figure 1.2.2. The volume of the transactions increases in time until the end of September. It drops significantly at the end of September but follows by another increase. The cause is that this is government transaction

data with government-issued credit cards, and the end of a fiscal year for the US government is in September. To be more specific, as the end of a fiscal year approaching, people in the government department usually charge more and more to burn up their budget. When a new fiscal year starts, the budget got reset, and they become relatively conservative. For more information on the data summary, please see Appendix I for reference.

Figure 1.2.1 Daily volume of the transactions

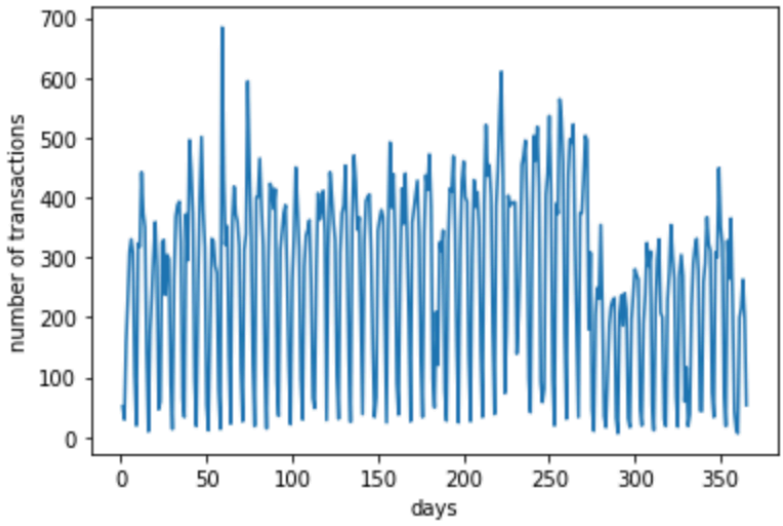
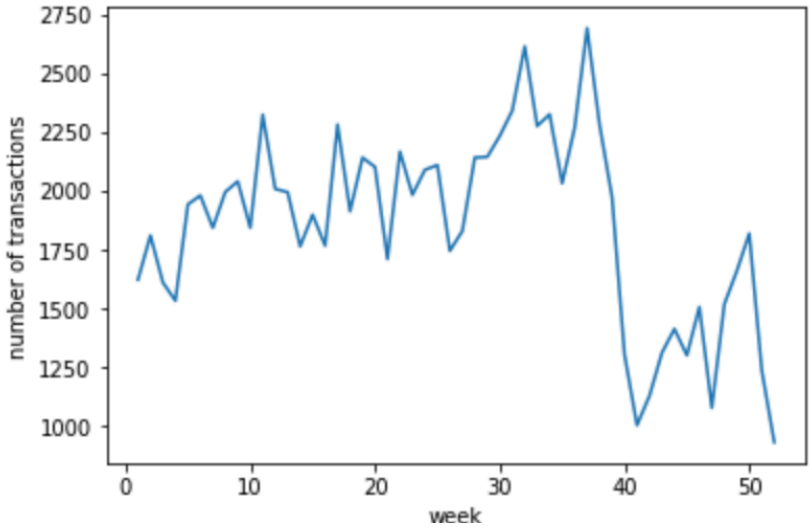



Figure 1.2.2 Weekly volume of the transactions

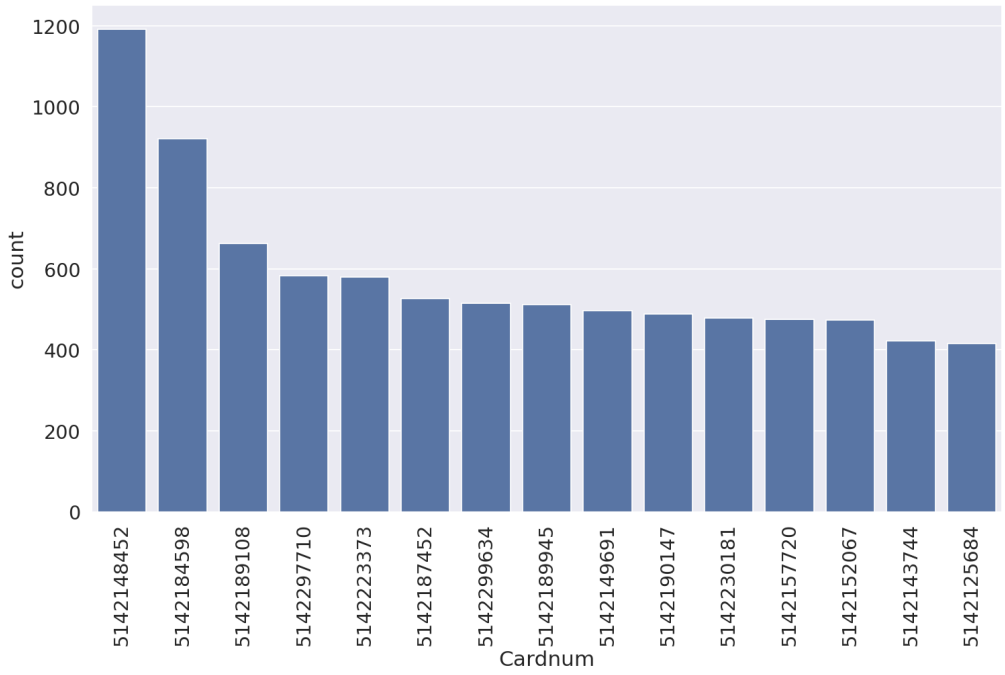



## 1.3 Histograms of Important Fields

This dataset contains a total of 10 fields. Before introducing our methodology and analysis, we will first present the histogram of the import fields for examining their distribution. We will also describe some idiosyncrasies in the fields.

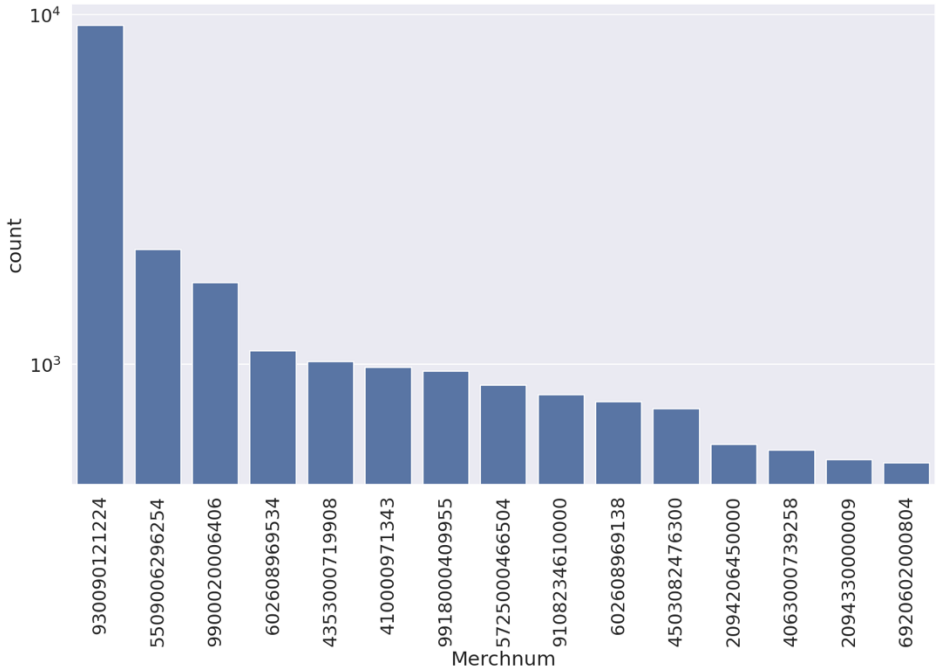
Field 1.

- Name: Cardnum
- Description: unique identifier of the card that made each purchase
- Count plot: (top 15 categories)



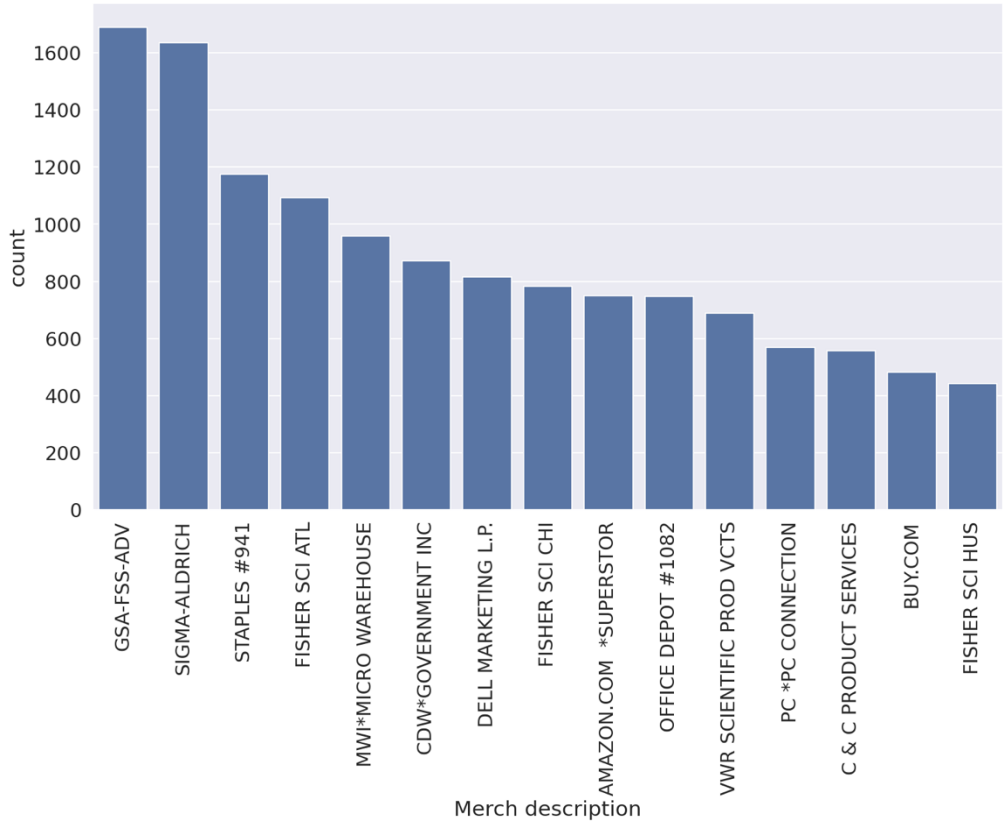

Field 2.

- Name: Merchnum
- Description: specific merchandise type number
- Count plot: (top 15 categories)



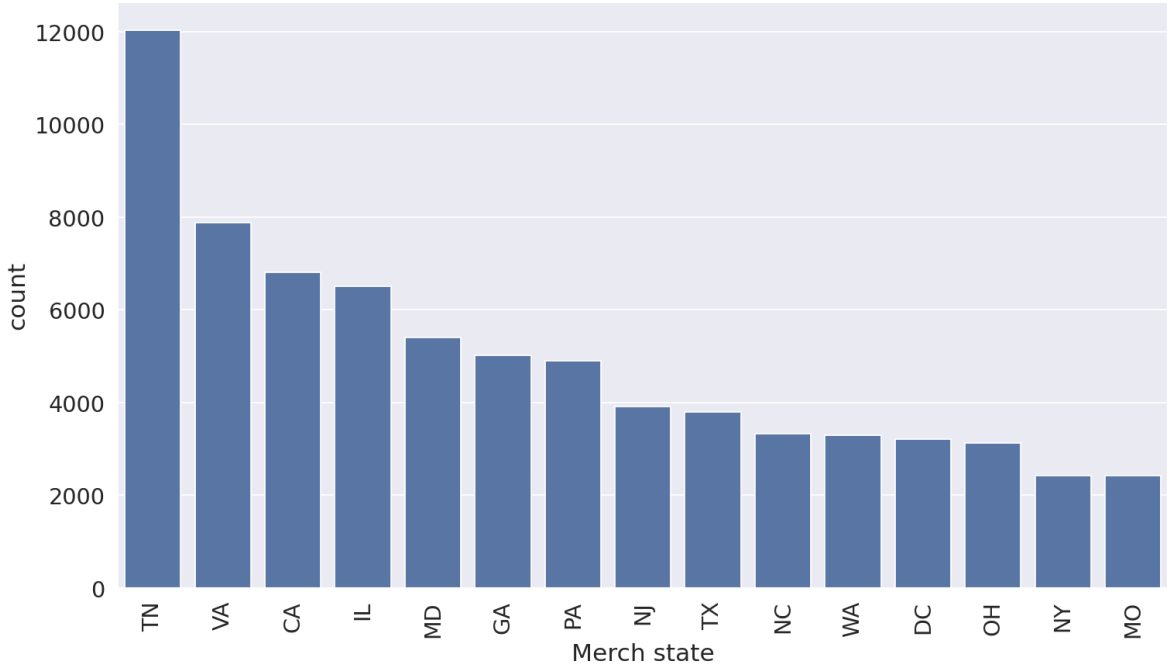

Field 3.

- Name: Merch description
- Description: description of the specific merchandise type. There is FedEx related merchandise, and we notice they violate Benford's Law, the non-intuitive fact that the first digit of many measurements is not uniformly distributed.
- Count plot: (top 15 categories)



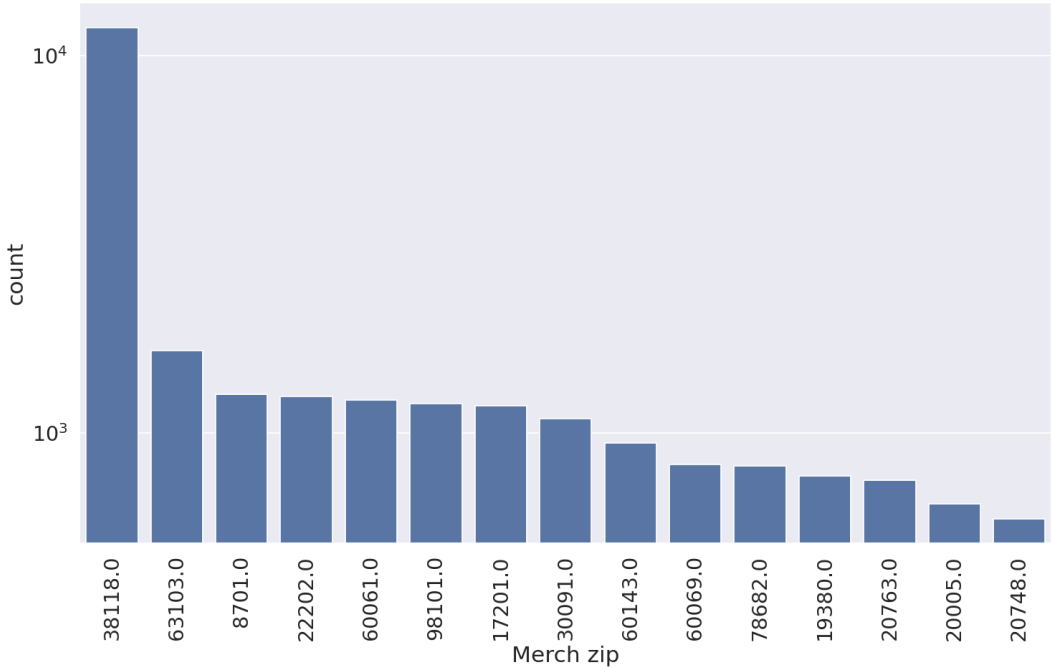

Field 4.

- Name: Merch state
- Description: the state of the merchandise
- Count plot: (top 15 categories)



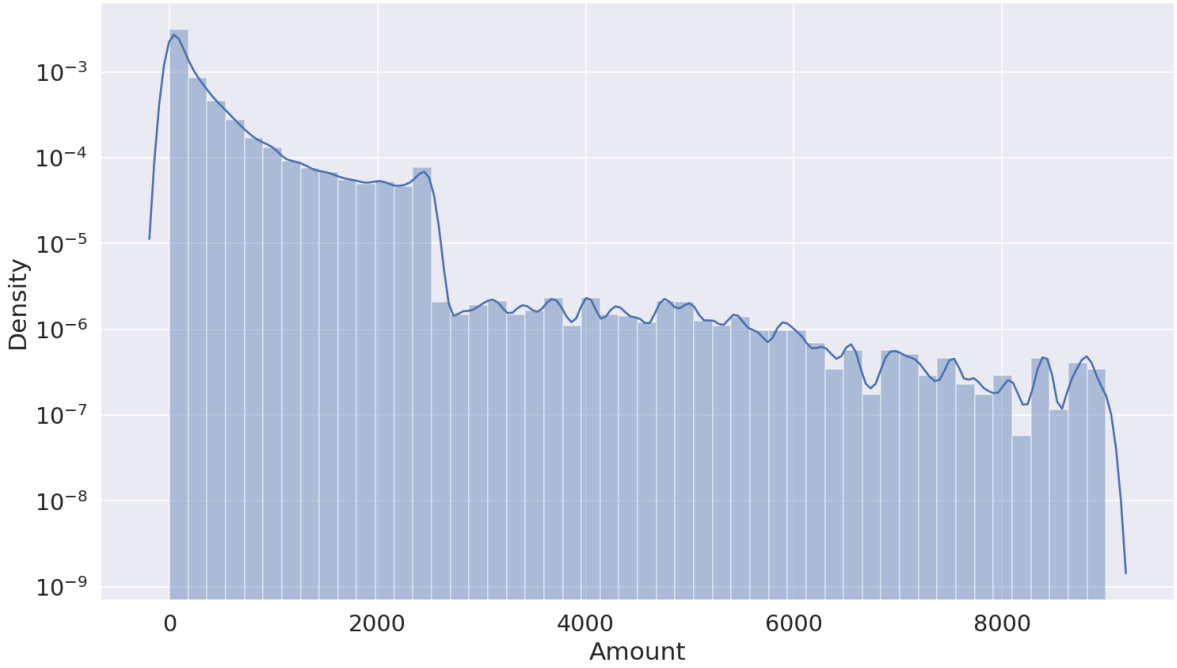

Field 5.

- Name: Merch zip
- Description: the zip code of the location of the merchandise
- Count plot: (top 15 categories)



Field 6.

- Name: Amount
- Description: the amount of the transaction
- Histogram: (data in the histogram excludes outliers > 9000 and is 99.91% populated)




Field 7.

- Name: Fraud
- Description: indication of a fraud. The fraud labels are unbalanced from the plot.
- Count plot:


$10^5$
$10^4$
$10^3$
count
0
1
Fraud

# 2. Data Cleaning

## 2.1 Data Exclusions

The number of records in the dataset is 96,753. However, this data includes some records that we are not interested in as part of this transaction fraud detection or that are a significant outlier.

To be more specific, among the four transaction types in the data, we are only interested in the "P" type transactions, representing a purchasing transaction. Thus, we first removed all the records that were not in the transaction type of "P".

Moreover, we found that in the transaction amount field, there was a transaction that had a significantly large amount ($3,102,045.53). We considered this record as an outlier. In order not to affect the model performance, we removed this single record for further analysis.

## 2.2 Data Imputations

In this data set, the merchandise state, merchandise number, and merchandise ZIP code are containing missing values. We chose to fill these 3 fields because they are important for us to create variables and build algorithms for fraud detection.

The first is the merchandise state (Merch state). If a record had a merchandise ZIP code but was missing a merchandise state, we used the state for that ZIP code if known. In addition, if the ZIP code is in the range 00600 – 00799 or 00900 – 00999, we filled in the state as PR, representing Puerto Rico. Then, if a record was missing a ZIP code with a known state or PR, we filled in with the mode of state of the Merchnum or Merch description it belonged to. Finally, if none of the above methodologies worked, we imputed the state of the record with" Unk", which means unknown state.

Secondly, we imputed the records with missing merchandise numbers (Merchnum). If the merchandise number was 0, we replaced then with Nan, representing the missing merchandise number because 0 is not a normal merchandise number. We then filled in the missing merchandise number of a record with the mode of merchandise number based on the merchandise description (Merch description) that the record belonged to. Finally, if the methodology didn't work (e.g., having multiple modes), we imputed the merchandise number of the record with" Unk", which means unknown merchandise number.

Last but not least, we imputed the merchandise ZIP code (Merch zip). If a record was missing a merchandise ZIP code, we filled in it with the mode of the merchandise ZIP code of the merchandise number it had. If the methodology didn't work, we imputed the merchandise ZIP code of the record with" Unk", which means an unknown ZIP code.

# 3. Variable Creation

To better capture unusual amount or frequency credit card transaction activities, 6 categories and a total of 495 candidate variables were created based on existing fields. Table 3.0.1 describes the category and the number of variables in each category.

Table 3.0.1: Summary 5 New Consolidated Variables

| Category | Description | # of Variables |
|---|---|---|
| Consolidated Variables | Combination of existing fields (card, merchnum, zip and state) | 5 |
| Amount Variables | Basic statistic (mean, max, medium and sum) of field values | 384 |
| Frequency Variables | Number of transactions with the same transaction character used over a certain time frame | 48 |
| Day-since Variables | Number of days since the same transaction character been used last time | 8 |
| Velocity Change Variables | Ratio of shorter time frequency over a longer time frequency | 48 |
| Target Encoded Variables | Probability of transaction fraud happen for each date/state | 2 |

## 3.1 Consolidated Variables

Since we only have a limited amount of fields that are available (Cardnum, Merchnum, Merch Zip and State), there are still other transaction activities that cannot be captured by existing fields. We first made combinations of existing fields to create our consolidated variables, Table 3.1.1 below contains formula and description of each.

Table 3.1.1: Summary 5 New Consolidated Variables

| Variable Name | Formula | Description |
|---|---|---|
| card_merch | Merchnum + Cardnum | card at this merchant |
| card_zip | Merch Zip + Cardnum | card in this zip code |
| card_state | Merch State + Cardnum | card in this state |
| merchnum_zip | Cardnum + Merch Zip | merchant in this zip |
| merchnum_state | Merchnum + Merch State | merchant in this state |

With the 5 new variables that we just created, we have a total of 8 character variables. Each character variable is a unique feature belonging to a transaction record.

Cardnum Merch zip card_zip merchnum_zip

Merchnum card_merch card_state merchnum_state

## 3.2 Amount Variables

We then created amount variables to gain insight from statistical measurements of transaction amounts. For example, suppose for a credit card, the amount (average amount, maximum amount, median amount and total amount) of the transaction was significantly higher than others' over the same time frame. In this case, this transaction is more likely to be a fraud activity.

Therefore, the greater the amount variables are, the more likely the transaction is unusual. Figure 3.2.1 below shows how 384 amount variables are created.

Figure 3.2.1: Creation of 384 *amount* Variables

| Variable Name |
|---|
| average |
| maximum |
| median |
| total |
| actual / average |
| actual / maximum |
| actual / median |
| actual / total |

amount by/at this

| Character Name | Description |
|---|---|
| cardnum | card number |
| merchnum | merchant account number |
| merch_zip | merchant zip code |
| card_merch | card at this merchan |
| card_zip | card in this zip code |
| card_state | card in this state |
| merchnum_zip | merchant in this zip |
| merchnum_state | merchant in this state |

over the past

| # of Days |
|---|
| 0 |
| 1 |
| 3 |
| 7 |
| 14 |
| 30 |

- 8 x 8 x 6 = 384 *amount* variables

## 3.3 Frequency Variables

Frequency of a character being used within a certain period of time is another important measurement for transaction activities. A transaction is considered suspicious if one of the transaction characters occurred a lot more than others. A bigger frequency value means frequent transactions of the same character, hence a higher chance of unusual activity.

We calculated frequency for each character over the past 0, 1, 3, 7, 14, 30 days. Figure 3.3.1 shows how the 48 frequency variables are built.

Figure 3.3.1: Creation of 48 *frequency* Variables

# of records with the same

| Character Name | Description |
|---|---|
| cardnum | card number |
| merchnum | merchant account number |
| merch_zip | merchant zip code |
| card_merch | card at this merchan |

| | |
|---|---|
| card_zip | card in this zip code |
| card_state | card in this state |
| merchnum_zip | merchant in this zip |
| merchnum_state | merchant in this state |

over the past 0, 1, 3, 7, 14, 30 days

- 6 x 8 = 48 *frequency* variables

## 3.4 Days-since Variables

Day since last transaction, means the number of days since the last occurrence of the same character. The smaller the *days-since* value, the closer the previous occurrence of the same character, the higher likelihood of unusual transaction activity for this record. If a character does not have a previous value, which means that this character is the first time seen, 365 will be used for this variable.

Figure 3.4 below shows how 8 days-since variables are created.

Figure 3.4.1: Creation of 8 *days-since* Variables

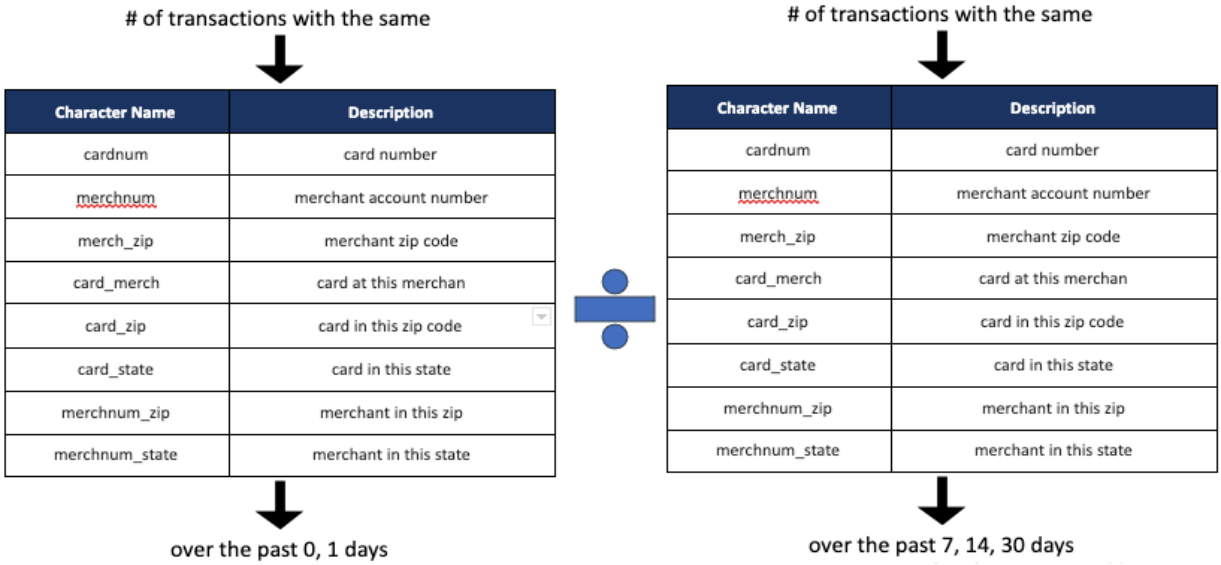
# of days since the last occurrence of

| Character Name | Description |
|---|---|
| cardnum | card number |
| merchnum | merchant account number |
| merch_zip | merchant zip code |
| card_merch | card at this merchant |
| card_zip | card in this zip code |
| card_state | card in this state |
| merchnum_zip | merchant in this zip |
| merchnum_state | merchant in this state |

## 3.5 Velocity Change Variables

Based on the frequency variables we already created, we further made velocity change variables. Velocity change variables measure the number of occurrences of one character in a short period relative to a longer time frame. A transaction may be fraud if in a short period of time, it occurred too many times compared to a longer time frame. The detailed calculation process is shown in figure 3.5.1 below, and a total of 48 velocity change variables were created.

Figure 3.5.1: Creation of 48 *velocity change* Variables




- 2 x 3 x 8 = 48 *velocity change* variables

## 3.6 Target Encoded Variables

Categorical fields *Date* and *State* help us better understand what are the chances of transaction fraud by date or by state. We applied target encoding here to fit these categorical variables into machine learning models.

In target encoding, we calculated the average probability of fraud happening by grouping data by date. Similarly, we grouped data by state and calculated the likelihood of fraud happening in that state. Table 3.6.1 below shows the variable name and description.

Table 3.6.1: Table of Target Encoded Variables

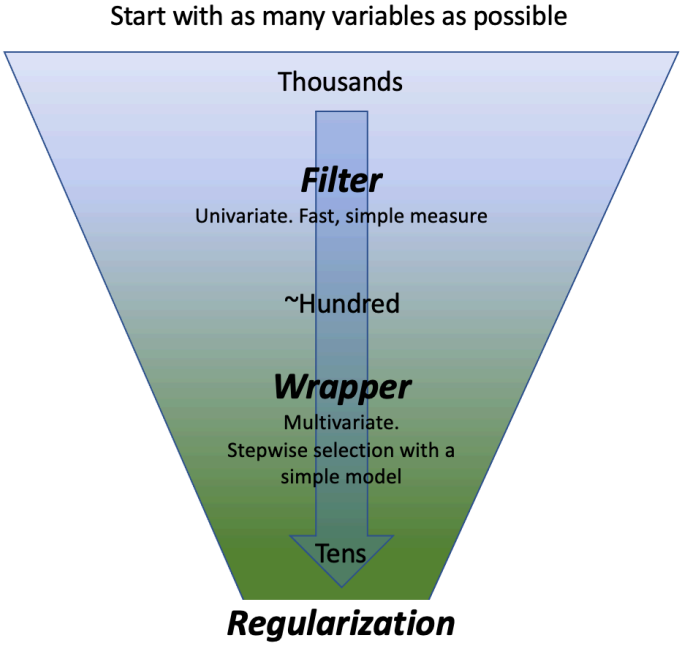
| Variable Name | Description |
|---|---|
| day_fraud_prob | probability of fraud happened on this day |
| state_fraud_prob | probability of fraud happened in this state |

# 4. Feature Selection

After creating the candidate variables, we used the feature selection to get the number of variables down. The purpose of this step is to select the most important independent variables to the dependent variable y and to help the fraud models perform efficiently. Before starting the selection process, we first removed the out-of-time records (last 4 months) and the first 2 weeks of records as well as added fraud label and a random set of numbers in the variable set for a purpose of comparison.

The feature selection process mainly contains two steps, filter, and wrapper. Figure 4.0.1 is a high-level summary of the feature selection process. We first used a filter to apply univariate testing that ignores correlations and scales the variables linearly with the number of variables. The filter is independent of any modeling method and is fast and simple. We then selected the most important 80 variables to put into the wrapper that builds many models with moderate speed and complexity to remove correlations. After the wrapper, we got down the number of variables to 30, and those 30 variables are the only variables that we used for modeling.

Figure 4.0.1 high-level summary of feature selection




## 4.1 Filter

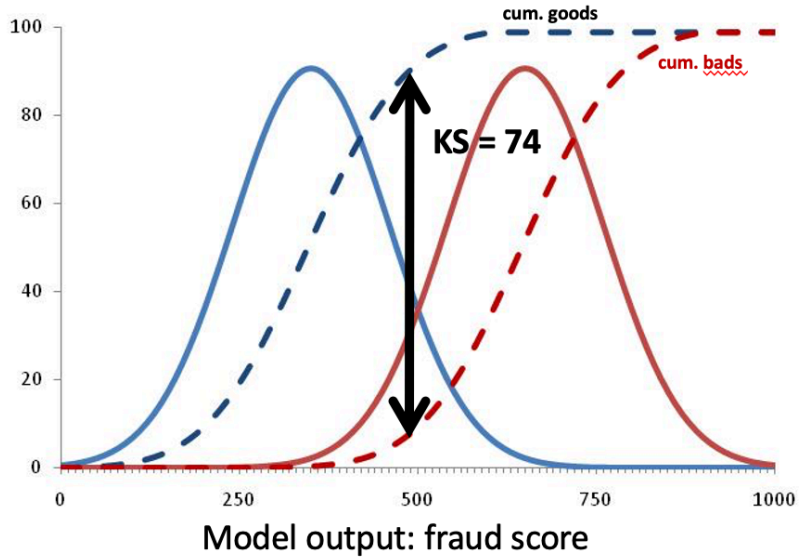
We used two measures in the filter: univariate KS and FDR. The univariate KS is a filter feature selection method for binary classification problems and a global measure for variables that are continuous or have a metric/ordering. KS is short for Kolmogorov-Smirnov and is a statistical measure of how well two distributions are separated. For each variable, it makes separate distributions for the two populations (good, bad) as figure 4.1.1 showed. The more different the

good and bad curves, the better the variable for separating, and thus the more important the variable is. It then independently measures the distance between two distributions as KS score (the maximum of the difference of the cumulative). The calculation function is presented below:

$$KS = \max_x \left| \int_{x_{min}}^{x} P_{\text{good}}\, dx - \int_{x_{mi}}^{x} P_{\text{bad}}\, dx \right|$$

$$= \max_x \left| \int_{x_{min}}^{x} [P_{\text{good}} - P_{\text{bad}}]\, dx \right|$$

Figure 4.1.1 KS (Kolmogorov-Smirnov) measurement

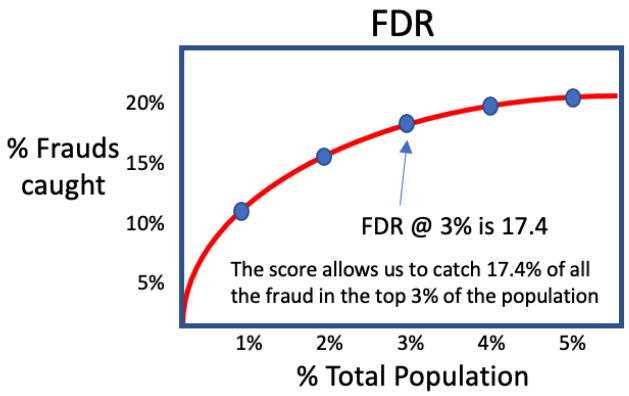



The second measure we used in the filter is FDR (Fraud Detection Rate), a main measure of goodness for fraud. FDR is the percentage of all the frauds caught at a particular examination cutoff location. For example, FDR 50% at 5% means the model catches 50% of all the frauds in 5% of the population. In the project, we first sorted the values of a column (independent variable) and got the head 3% and the tail 3% of the data to examine subpopulations. Then, we divided the number of frauds into the two subpopulations by the total number of frauds in all the data. Finally, we chose the maximum between the two subpopulations as the FDR score. Figure 4.1.2 is a fraud detection rate plot that explains the % fraud detected by this method.

We then ranked the KS and FDR scores for all records in descending order. Since the two measures are both typical business measures for the binary classification model, we calculated the average ranking of the KS and FDR scores and selected the variables that ranked the highest (except Fraud label) by this average ranking. The number of variables we selected in this filter step is 80, which

is a reasonable number of variables that work well in the wrapper step with a reasonable amount of computing time.

Figure 4.1.2 fraud detection rate plot




## 4.2 Wrapper

We then moved the selected top 80 variables into a wrapper. There are 3 types of wrapper process, forward selection, backward selection, and general stepwise selection. Our goal of this step is to get a short but conservative list of variables in the order of multivariate importance. In other words, the correlations among independent variables must be considered in the wrapper. With the ranked order by multivariate importance, we can remove or increase variables in the given order to test nonlinear algorithms on various numbers of input and to see how performance changes. Both forward and backward selection work well in achieving the above goal and expectation, and we chose to use forward selection in our project.

The forward selection started with no variables selected. Since we had 80 variables to input the wrapper, it first built 80 separate 1D models and tried adding in each variable that we didn't already have. It kept the best variable by the overall model performance as a measure of strength. It then continued the process as Figure 4.2.1 showed. In the wrapper, we used the Sequential Feature Selector in the Python mlxtend package with a forward selection and a random forest classifier, a simple non-linear tree-based model. We used FDR as the measure of goodness and used 2-fold cross-validation in the selection process. In each step of selecting variables, the forward selection removed the correlation and used a greedy search to look for the next best position given the current location.

After building all possible models, it had selected all the 80 variables in an order of multivariate importance. We chose the best 30 variables to move on to our fraud algorithm. Table 4.2.2 is a list of final variables in ranked order of importance.

Figure 4.2.1 forward selection process

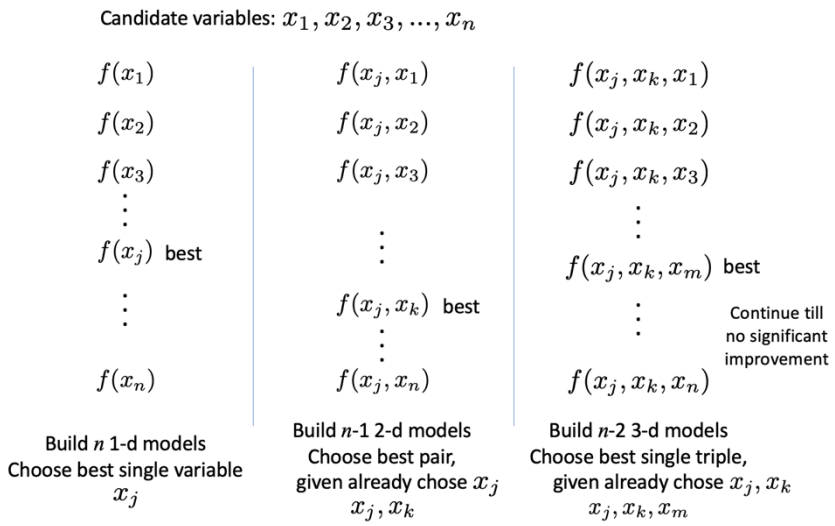



Figure 4.2.2 ranked importance of final variables

| Item | Variable |
|---|---|
| 1 | card_zip_total_7 |
| 2 | card_zip_total_3 |
| 3 | card_state_total_7 |
| 4 | card_merch_total_14 |
| 5 | card_merch_total_3 |
| 6 | card_state_total_1 |
| 7 | card_merch_total_30 |
| 8 | card_zip_total_1 |
| 9 | card_zip_max_14 |
| 10 | card_zip_max_30 |
| 11 | card_state_max_7 |
| 12 | card_zip_max_7 |
| 13 | card_state_max_14 |
| 14 | cardnum_total_7 |
| 15 | card_merch_max_14 |
| 16 | card_zip_max_3 |
| 17 | card_merch_max_7 |
| 18 | card_zip_max_1 |
| 19 | card_merch_max_1 |
| 20 | card_state_max_1 |
| 21 | cardnum_total_0 |
| 22 | merchnum_state_total_1 |
| 23 | merchnum_state_total_0 |
| 24 | merchnum_zip_total_0 |
| 25 | merchnum_zip_max_0 |
| 26 | card_merch_max_0 |
| 27 | cardnum_max_0 |
| 28 | merchnum_zip_max_3 |
| 29 | card_state_avg_14 |
| 30 | merchnum_total_0 |

# 5. Fraud Model Algorithms

The best 30 variables in order of ranked importance are now ready for our model training and parameter tuning purposes. We compared different algorithms, including Logistic Regression, Random Forests, XGboost and Neural Network, and all the model results are in Figure 6.0.8 in the 6. Results section.

For each of these models, we tried different combinations of hyperparameters and train/test split , to acquire the best performance and robustness. 10-fold cross-validation was implemented to prevent overfitting. The reason why we chose to run each model 10 times is that our data set is small with a small number of frauds in it. Each model run even with the same hyperparameters can give very different results, because the algorithms start with a random set of initial internal fitting parameters. The cross validation was implemented in following steps:

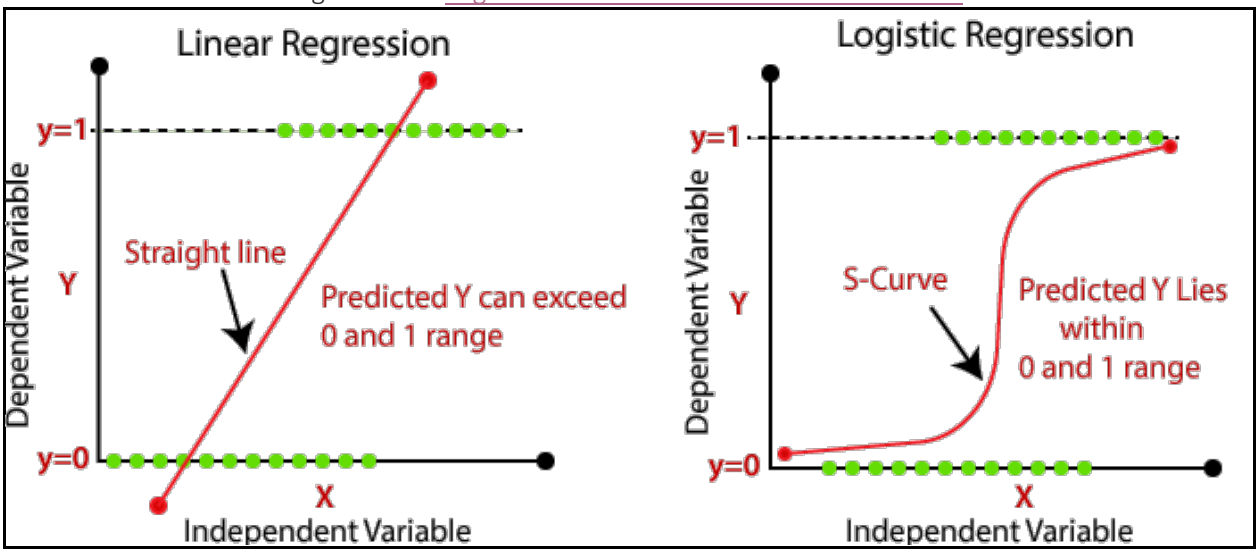
1. Randomly shuffle the train set and split it into 10 groups.
2. For each group, we take it as the validate group. Then, the model is trained based on the remaining records.
3. Evaluate the scores in each of the 10 rounds, based on the FDR at 3% of train set, validation set and OOT data, making sure there is no significant variance.
4. Calculate the average evaluation score.

We also used a voting classifier which trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output.

## 5.1 Logistic Regression

Logistic Regression is a commonly used binary classification method to estimate the probability that a given entity belongs to a particular class. If the estimated probability is greater than the cutoff value(often 50%), then the classifier predicts the instance to belong to the positive class, labeled 1. If the estimated probability is less than the cutoff value, then the classifier predicts the instance to belong to the negative class, labeled 0. Thus the output belongs to either 1 or 0, making it a binary classifier. A graph that indicates the logics of the Logistic Regression is shown below.

Figure 5.1.1: Logistic Model and Linear Model Illustration




Similar to a linear regression model, a Logistic Regression model takes input features and outputs a weighted sum of them and a bias term. However, instead of directly outputting results, the Logistic Regression uses a sigmoid function to output a number between 0 and 1.

Due to the simplicity of logistic regression, we decided to use it as the baseline model, taking our best candidate variables as the independent variables and output the fraud label as the dependent variable. In our case, the instance is either fraudulent(1) or non-fraudulent(0).

The choice of hyperparameters we tuned in model training is shown in the following table.

- Package: sklearn.linear_model.LogisticRegression

Table 5.1.1: Essential Parameters in Logistic Regression

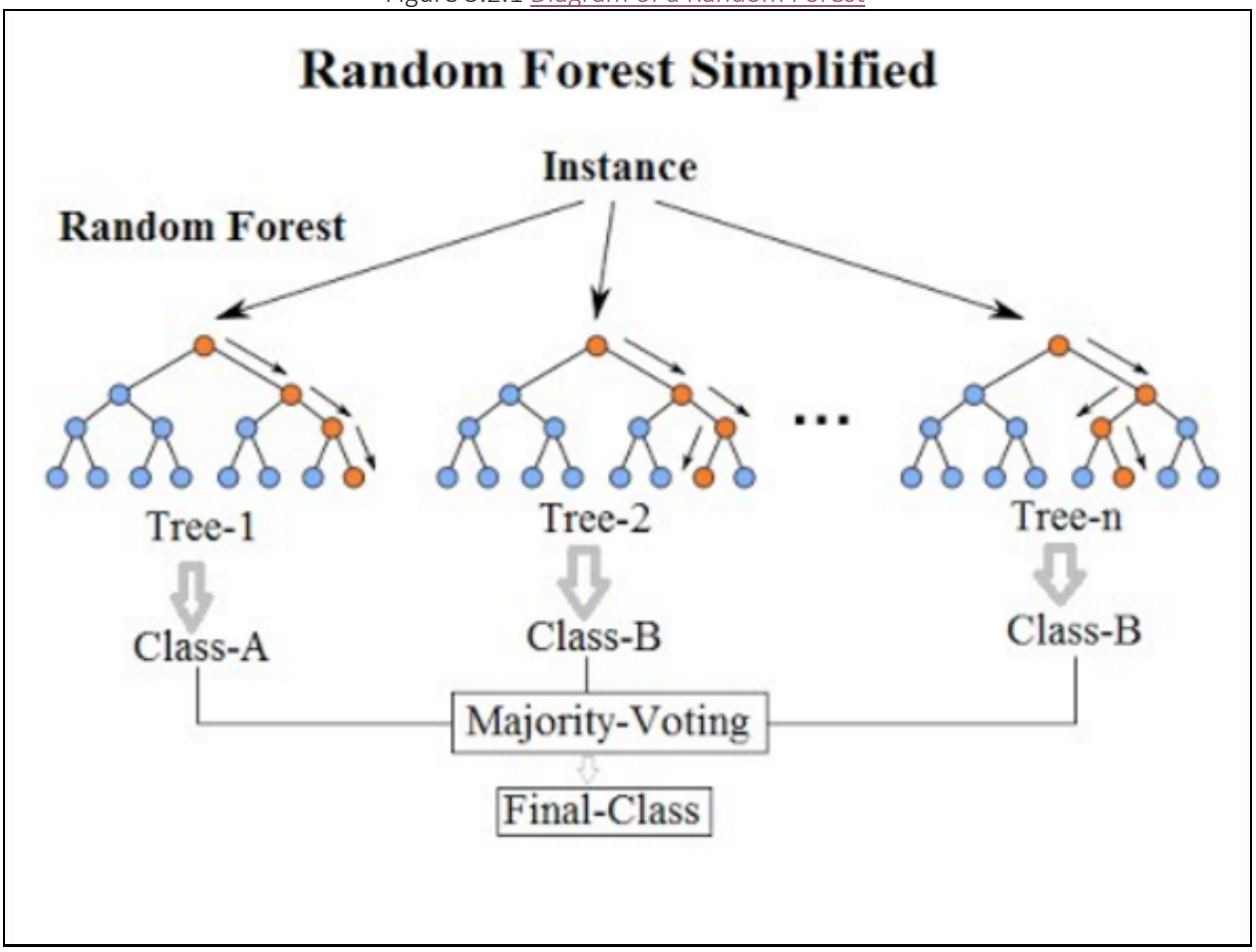
| Name | Description |
|---|---|
| penalty | Used to specify the norm used in the penalization. |
| C | Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization. |
| solver | Algorithm to use in the optimization problem. |
| max_iter | Maximum number of iterations taken for the solvers to converge. |
| l1_ratio | The Elastic-Net mixing parameter, with 0 <= l1_ratio <= 1. Only used when penalty='elasticnet'. |

## 5.2 Random Forest

Random forest is an ensemble of decision trees. Random Forest is constructed by a multitude of decision trees and outputs the results selected by most trees in classification. Compared to decision trees, Random Forest is better in beating overfitting, but it is generally less accurate than Gradient boosting trees.

Random Forest is generally trained via the bagging algorithm, and it has all the hyperparameters of decision trees and all the hyperparameters of bagging classifiers. It introduces extra randomness because instead of searching for the best feature when splitting a node, it searches for the best feature among a random subset of features.

Figure 5.2.1 Diagram of a Random Forest




We used the following hyperparameters:

- Package: sklearn.ensemble.RandomForestClassifier

Table 5.2.1: Essential Parameters in Random Forest

| Name | Description |
|---|---|
| n_estimators | The number of trees in the forest. |
| max_depth | The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. |
| max_features | The number of features to consider when looking for the best split |
| min_samples_leaf | The minimum number of samples required to be at a leaf node. |
| min_samples_split | The minimum number of samples required to split an internal node |
| criterion | The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain. Note: this parameter is tree-specific. |

## 5.3 Boosted Tree

The gradient boosting uses an ensemble of weak learners such as decision trees to predict the outcome variable. Each subsequent tree in the series tries to capture the residual errors of the previous tree.

We used XGBoost, which is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework.

We tuned the below parameters (see Table 5.3.1):

- Package: xgboost

Table 5.3.1: Essential Parameters in Boosted Tree

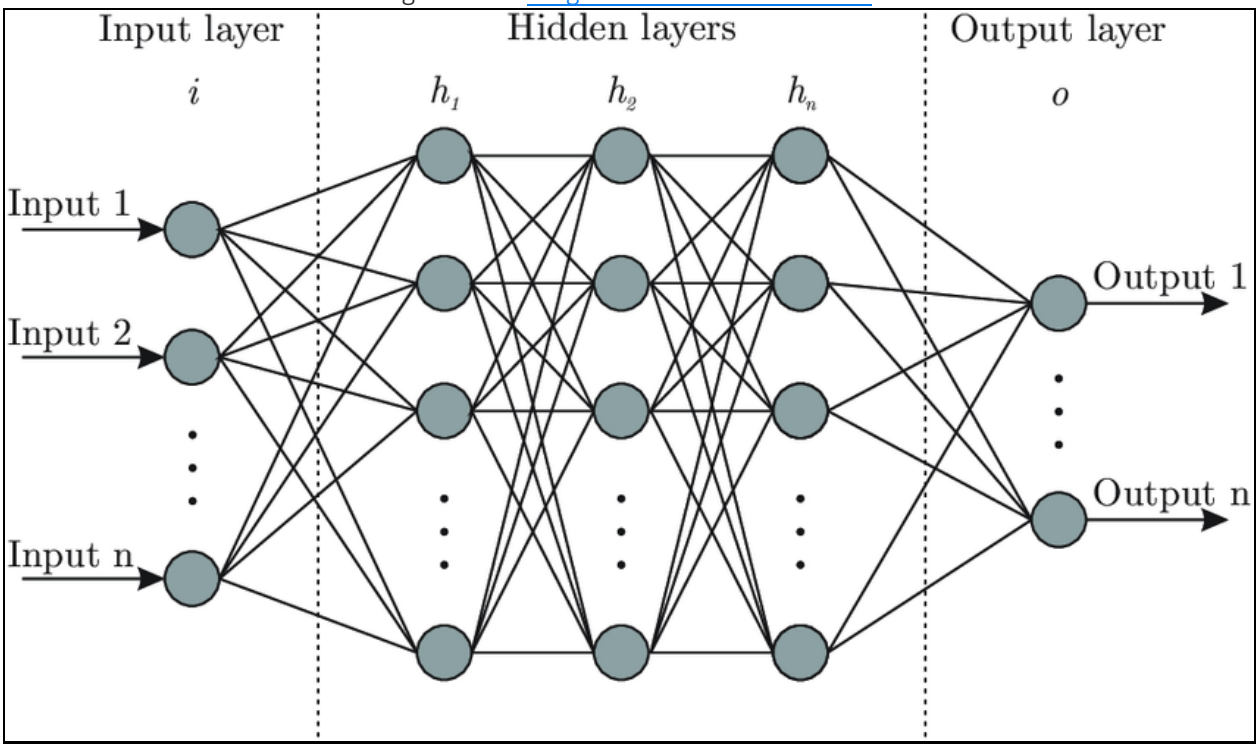
| Name | Description |
|---|---|
| n_estimators | Number of gradient boosted trees. Equivalent to number of boosting rounds. |
| max_depth | Maximum tree depth for base learners. |
| learning_rate | Boosting learning rate |
| eval_metric | Evaluation metrics for validation data, a default metric will be assigned according to objective |

## 5.4 Neural Network

An Artificial neural network is a loose model of neurons in a human brain. It is based on a collection of nodes and each node in the network can transmit a signal to other nodes. A Neural Network has an input layer, hidden layers and an output layer. Each neuron in the structure gets a signal from the previous layer's nodes, applies a transfer function to it, and then sends a new signal to the following layer's nodes. In general, the signal received from the previous layer is a linear combination of the previous layer nodes' outputs.

For a binary classification problem in our case, we just need a single output node using the logistic activation function. It outputs a number between 0 and 1, which could be interpreted as the estimated probability of the positive class, the frauds , and the estimated probability of the negative class(non-frauds) is 1 minus this number.

Figure 5.4.1 Diagram of a Neural Network




The parameters used for Neural Network are as belows (see table 5.4.1).

- Package: sklearn.neural_network.MLPClassifier

Table 5.4.1: Essential Parameters in Neural Network

| Name | Description |
|---|---|
| hidden_layer_sizes | The ith element represents the number of neurons in the ith hidden layer. |
| activation | Activation function for the hidden layer. |
| max_iter | Maximum number of iterations. |
| alpha | L2 penalty (regularization term) parameter. |
| solver | The solver for weight optimization. |
| verbose | Whether to print progress messages to stdout. |
| learning_rate | Learning rate schedule for weight updates. |

# 6. Results

The logistic regression model achieved the lowest scores of the five classifiers used on all datasets. Using the best parameters of the top 20 variables, L1 penalty, C of 0.5, SAGA solver, the logistic regression model's scores were 0.676 on the training data, 0.756 on the test data, and 0.511 on the oot data.

Next, the best neural network parameters were the top 20 variables, 2 layers, verbosity of 3, 20 nodes, RELU activation function, and Adam solver. The neural network scored 0.808 on the training data, 0.862 on the test data, and 0.615 on the oot data.

XG boosted trees placed third using the top 20 variables, 500 estimators, maxdepth of 2, and a learning rate of 0.1. This model scored 0.933 on the training data, 0.916 on the test data, and 0.64 on the oot data.

As the highest performing single model, the random forest's best parameters were the top 20 variables, 500 estimators, no warm start, a verbosity of 4, True for the oob score parameter, and min sample leaves of 3. The random forest scored 1 on the training data, 0.916 on the test data, and 0.657 on the oot data.

Various ensemble models were tried in the form of voting classifiers: a combination of all four prior models, a combination of all three non-linear models, and several combinations of two of the nonlinear models. The voting classifier model consisting of the neural network and random forest was selected as the final model and achieved the highest performance of all models tested with scores of 1 on training. 0.899 on validation, and 0.674 on the oot data (see fig ).

Performance statistics such as false positive rate, KS, and fraud detection rate for the top performing model are listed in Figure 6.0.1, Figure 6.0.2, and Figure 6.0.3 for the top 20th percentiles.

Figure 6.0.1: Population bins with performance statistics for the training data set used to train the final voting classifier model. Top 20 bins shown.

| | # Records | # Goods | # Bads | Fraud Rate | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 45995 | 45527 | 468 | 0.0102 | | | | | | | |
| Training | Bin Statistics | | | | | Cumulative Statistics | | | | | |
| Population Bin % | # Records | # Goods | # Bads | % Goods | % Bads | Total # Records | Cumulative Good | Cumulative Bad | % Good | % Bad (FDR) | KS | FPR |
| 1 | 460 | 68 | 392 | 14.78 | 85.22 | 460 | 68 | 392 | 0.15 | 83.76 | 83.61 | 0.17 |
| 2 | 460 | 384 | 76 | 83.48 | 16.52 | 920 | 452 | 468 | 0.99 | 100.00 | 99.01 | 0.97 |
| 3 | 460 | 460 | 0 | 100.00 | 0.00 | 1380 | 912 | 468 | 2.00 | 100.00 | 98.00 | 1.95 |
| 4 | 460 | 460 | 0 | 100.00 | 0.00 | 1840 | 1372 | 468 | 3.01 | 100.00 | 96.99 | 2.93 |
| 5 | 460 | 460 | 0 | 100.00 | 0.00 | 2300 | 1832 | 468 | 4.02 | 100.00 | 95.98 | 3.91 |
| 6 | 460 | 460 | 0 | 100 | 0 | 2760 | 2292 | 468 | 5.03 | 100.00 | 94.97 | 4.90 |
| 7 | 460 | 460 | 0 | 100 | 0 | 3220 | 2752 | 468 | 6.04 | 100.00 | 93.96 | 5.88 |
| 8 | 460 | 460 | 0 | 100 | 0 | 3680 | 3212 | 468 | 7.06 | 100.00 | 92.94 | 6.86 |
| 9 | 460 | 460 | 0 | 100 | 0 | 4140 | 3672 | 468 | 8.07 | 100.00 | 91.93 | 7.85 |
| 10 | 460 | 460 | 0 | 100 | 0 | 4600 | 4132 | 468 | 9.08 | 100.00 | 90.92 | 8.83 |
| 11 | 460 | 460 | 0 | 100 | 0 | 5060 | 4592 | 468 | 10.09 | 100.00 | 89.91 | 9.81 |
| 12 | 460 | 460 | 0 | 100 | 0 | 5520 | 5052 | 468 | 11.10 | 100.00 | 88.90 | 10.79 |
| 13 | 460 | 460 | 0 | 100 | 0 | 5980 | 5512 | 468 | 12.11 | 100.00 | 87.89 | 11.78 |
| 14 | 460 | 460 | 0 | 100 | 0 | 6440 | 5972 | 468 | 13.12 | 100.00 | 86.88 | 12.76 |
| 15 | 460 | 460 | 0 | 100 | 0 | 6900 | 6432 | 468 | 14.13 | 100.00 | 85.87 | 13.74 |
| 16 | 460 | 460 | 0 | 100 | 0 | 7360 | 6892 | 468 | 15.14 | 100.00 | 84.86 | 14.73 |
| 17 | 459 | 459 | 0 | 100 | 0 | 7819 | 7351 | 468 | 16.15 | 100.00 | 83.85 | 15.71 |
| 18 | 460 | 460 | 0 | 100 | 0 | 8279 | 7811 | 468 | 17.16 | 100.00 | 82.84 | 16.69 |
| 19 | 460 | 460 | 0 | 100 | 0 | 8739 | 8271 | 468 | 18.17 | 100.00 | 81.83 | 17.67 |
| 20 | 460 | 460 | 0 | 100 | 0 | 9199 | 8731 | 468 | 19.18 | 100.00 | 80.82 | 18.66 |

Figure 6.0.2: Population bins with performance statistics for the test data set used to test the final voting classifier model. Top 20 bins shown.

| | # Records | # Goods | # Bads | Fraud Rate | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 19713 | 19490 | 223 | 0.0113 | | | | | | | |
| Testing | Bin Statistics | | | | | Cumulative Statistics | | | | | |
| Population Bin % | # Records | # Goods | # Bads | % Goods | % Bads | Total # Records | Cumulative Good | Cumulative Bad | % Good | % Bad (FDR) | KS | FPR |
| 1 | 198 | 46 | 152 | 23.23 | 76.77 | 198 | 46 | 152 | 0.24 | 68.16 | 67.93 | 0.30 |
| 2 | 197 | 165 | 32 | 83.76 | 16.24 | 395 | 211 | 184 | 1.08 | 82.51 | 81.43 | 1.15 |
| 3 | 197 | 190 | 7 | 96.45 | 3.55 | 592 | 401 | 191 | 2.06 | 85.65 | 83.59 | 2.10 |
| 4 | 197 | 193 | 4 | 97.97 | 2.03 | 789 | 594 | 195 | 3.05 | 87.44 | 84.40 | 3.05 |
| 5 | 197 | 197 | 0 | 100.00 | 0.00 | 986 | 791 | 195 | 4.06 | 87.44 | 83.39 | 4.06 |
| 6 | 197 | 196 | 1 | 99.49 | 0.51 | 1183 | 987 | 196 | 5.06 | 87.89 | 82.83 | 5.04 |
| 7 | 197 | 195 | 2 | 98.98 | 1.02 | 1380 | 1182 | 198 | 6.06 | 88.79 | 82.72 | 5.97 |
| 8 | 197 | 196 | 1 | 99.49 | 0.51 | 1577 | 1378 | 199 | 7.07 | 89.24 | 82.17 | 6.92 |
| 9 | 198 | 197 | 1 | 99.49 | 0.51 | 1775 | 1575 | 200 | 8.08 | 89.69 | 81.61 | 7.88 |
| 10 | 197 | 194 | 3 | 98.48 | 1.52 | 1972 | 1769 | 203 | 9.08 | 91.03 | 81.95 | 8.71 |
| 11 | 197 | 197 | 0 | 100.00 | 0.00 | 2169 | 1966 | 203 | 10.09 | 91.03 | 80.94 | 9.68 |
| 12 | 197 | 194 | 3 | 98.48 | 1.52 | 2366 | 2160 | 206 | 11.08 | 92.38 | 81.29 | 10.49 |
| 13 | 197 | 196 | 1 | 99.49 | 0.51 | 2563 | 2356 | 207 | 12.09 | 92.83 | 80.74 | 11.38 |
| 14 | 197 | 196 | 1 | 99.49 | 0.51 | 2760 | 2552 | 208 | 13.09 | 93.27 | 80.18 | 12.27 |
| 15 | 197 | 197 | 0 | 100.00 | 0.00 | 2957 | 2749 | 208 | 14.10 | 93.27 | 79.17 | 13.22 |
| 16 | 197 | 196 | 1 | 99.49 | 0.51 | 3154 | 2945 | 209 | 15.11 | 93.72 | 78.61 | 14.09 |
| 17 | 198 | 197 | 1 | 99.49 | 0.51 | 3352 | 3142 | 210 | 16.12 | 94.17 | 78.05 | 14.96 |
| 18 | 197 | 194 | 3 | 98.48 | 1.52 | 3549 | 3336 | 213 | 17.12 | 95.52 | 78.40 | 15.66 |
| 19 | 197 | 196 | 1 | 99.49 | 0.51 | 3746 | 3532 | 214 | 18.12 | 95.96 | 77.84 | 16.50 |
| 20 | 197 | 197 | 0 | 100.00 | 0.00 | 3943 | 3729 | 214 | 19.13 | 95.96 | 76.83 | 17.43 |

Figure 6.0.3: Population bins with performance statistics for the OOT variables using the final voting classifier model. Top 20 bins shown.

| | # Records | # Goods | # Bads | Fraud Rate | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 27351 | 26995 | 356 | 0.0130 | | | | | | | |
| OOT | Bin Statistics | | | | | Cumulative Statistics | | | | | |
| Population Bin % | # Records | # Goods | # Bads | % Goods | % Bads | Total # Records | Cumulative Good | Cumulative Bad | % Good | % Bad (FDR) | KS | FPR |
| 1 | 274 | 97 | 177 | 35.40 | 64.60 | 274 | 97 | 177 | 0.36 | 49.72 | 49.36 | 0.55 |
| 2 | 273 | 221 | 52 | 80.95 | 19.05 | 547 | 318 | 229 | 1.18 | 64.33 | 63.15 | 1.39 |
| 3 | 274 | 258 | 16 | 94.16 | 5.84 | 821 | 576 | 245 | 2.13 | 68.82 | 66.69 | 2.35 |
| 4 | 273 | 264 | 9 | 96.70 | 3.30 | 1094 | 840 | 254 | 3.11 | 71.35 | 68.24 | 3.31 |
| 5 | 274 | 266 | 8 | 97.08 | 2.92 | 1368 | 1106 | 262 | 4.10 | 73.60 | 69.50 | 4.22 |
| 6 | 273 | 267 | 6 | 97.80 | 2.20 | 1641 | 1373 | 268 | 5.09 | 75.28 | 70.19 | 5.12 |
| 7 | 274 | 270 | 4 | 98.54 | 1.46 | 1915 | 1643 | 272 | 6.09 | 76.40 | 70.32 | 6.04 |
| 8 | 273 | 261 | 12 | 95.60 | 4.40 | 2188 | 1904 | 284 | 7.05 | 79.78 | 72.72 | 6.70 |
| 9 | 274 | 267 | 7 | 97.45 | 2.55 | 2462 | 2171 | 291 | 8.04 | 81.74 | 73.70 | 7.46 |
| 10 | 273 | 271 | 2 | 99.27 | 0.73 | 2735 | 2442 | 293 | 9.05 | 82.30 | 73.26 | 8.33 |
| 11 | 274 | 270 | 4 | 98.54 | 1.46 | 3009 | 2712 | 297 | 10.05 | 83.43 | 73.38 | 9.13 |
| 12 | 273 | 270 | 3 | 98.90 | 1.10 | 3282 | 2982 | 300 | 11.05 | 84.27 | 73.22 | 9.94 |
| 13 | 274 | 273 | 1 | 99.64 | 0.36 | 3556 | 3255 | 301 | 12.06 | 84.55 | 72.49 | 10.81 |
| 14 | 273 | 272 | 1 | 99.63 | 0.37 | 3829 | 3527 | 302 | 13.07 | 84.83 | 71.77 | 11.68 |
| 15 | 274 | 273 | 1 | 99.64 | 0.36 | 4103 | 3800 | 303 | 14.08 | 85.11 | 71.04 | 12.54 |
| 16 | 273 | 270 | 3 | 98.90 | 1.10 | 4376 | 4070 | 306 | 15.08 | 85.96 | 70.88 | 13.30 |
| 17 | 274 | 272 | 2 | 99.27 | 0.73 | 4650 | 4342 | 308 | 16.08 | 86.52 | 70.43 | 14.10 |
| 18 | 273 | 270 | 3 | 98.90 | 1.10 | 4923 | 4612 | 311 | 17.08 | 87.36 | 70.27 | 14.83 |
| 19 | 274 | 270 | 4 | 98.54 | 1.46 | 5197 | 4882 | 315 | 18.08 | 88.48 | 70.40 | 15.50 |
| 20 | 273 | 270 | 3 | 98.90 | 1.10 | 5470 | 5152 | 318 | 19.09 | 89.33 | 70.24 | 16.20 |

Predictions were made for each record in the oot and were binned in percentiles based on the fraud probability. Based on the model's predictions, it was found that the 9th percentile bin was the best cutoff to maximize profits and fraud records caught (saving $2000 each time) while minimizing false positives (costing $50 everytime). Selecting the records in the 9th percentile and above results in fraud savings of $473,450. You can see Figure 6.0.4 to Figure 6.0.8 below for reference.

Figure 6.0.4 shows the transactions per week with card number 5142199009. The top graph highlights that usage experiences a small spike before week 40 and a large spike around week 50. During the periods of week 34 onward, average fraud score (red trend line) increases with higher card usage. This indicates that unusually high numbers of transactions in a small period are associated with fraudulent activity.

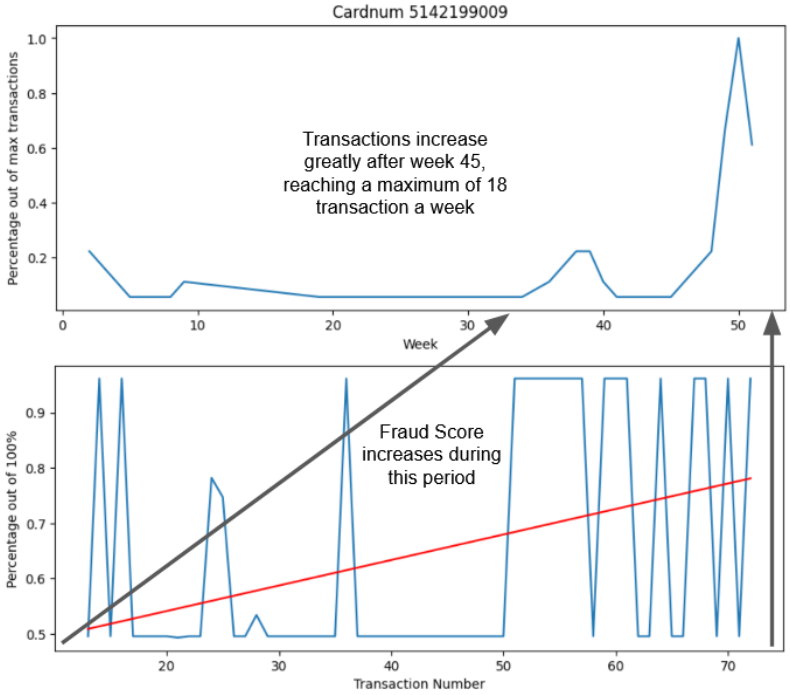



Figure 6.0.5 shows the transactions per week with card number 5142199009. The top graph highlights that usage experiences a small spike before week 40 and a large spike around week 50. During the periods of week 34 onward, average fraud score (red trend line) increases with higher card usage. This indicates that unusually high numbers of transactions in a small period are associated with fraudulent activity.

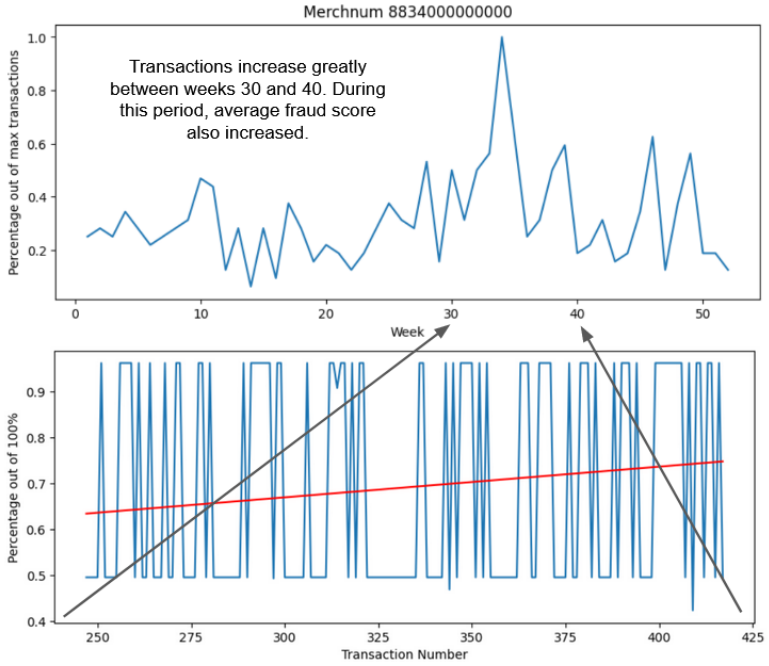


Figure 6.0.6 shows the transactions per week with merchant number 8834000000000. The top graph highlights that the merchant experiences a large spike between weeks 30 and 40. During the periods of week 30-40, average fraud score (red trend line) increases with the large increase in transactions. Like card transactions, higher than usual numbers of transactions with merchants in a short period are indicative of fraudulent activity.

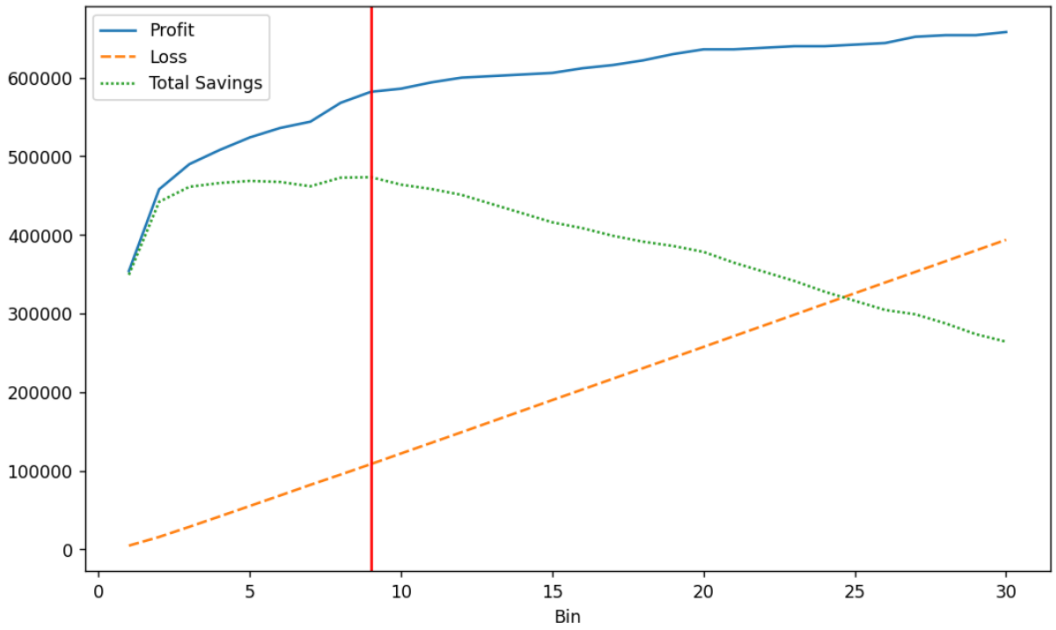



Figure 6.0.7: Savings from the fraud model are highest with a cutoff of 9% on the OOT records. $2000 is saved with every fraudulent transaction discovered. $50 is lost with every good transaction declared fraudulent.

| Model | | | Parameters | | | Average FDR at 3% | | |
|---|---|---|---|---|---|---|---|---|
| Logistic Regression | Variables | Penalty | C | Solver | | Train | Validate | OOT |
| 1 | 10 | L2 | 1 | LBFGS | | 0.676 | 0.753 | 0.499 |
| 2 | 15 | L2 | 1 | LBFGS | | 0.663 | 0.764 | 0.492 |
| 3 | 20 | L2 | 1 | LBFGS | | 0.669 | 0.784 | 0.507 |
| 4 | 25 | L2 | 1 | LBFGS | | 0.673 | 0.783 | 0.493 |
| 5 | 30 | L2 | 1 | LBFGS | | 0.684 | 0.785 | 0.492 |
| 6 | 20 | L1 | 2 | Liblinear | | 0.668 | 0.785 | 0.501 |
| 7 | 20 | L1 | 0.5 | SAGA | | 0.676 | 0.756 | 0.511 |
| 8 | 20 | L2 | 0.1 | SAGA | | 0.689 | 0.744 | 0.507 |
| 9 | 20 | L1 | 0.5 | Liblinear | | 0.674 | 0.760 | 0.497 |
| 10 | 20 | ElasticNet | 0.1 | SAGA | | 0.674 | 0.765 | 0.509 |
| Neural Network | Variables | Layers | Verbose | Nodes | Activation | Solver | Train | Validate | OOT |
| 1 | 10 | 1 | 0 | 10 | Relu | Adam | 0.706 | 0.843 | 0.564 |
| 2 | 15 | 1 | 0 | 10 | Relu | Adam | 0.721 | 0.812 | 0.563 |
| 3 | 20 | 1 | 0 | 10 | Relu | Adam | 0.740 | 0.835 | 0.596 |
| 4 | 25 | 1 | 0 | 10 | Relu | Adam | 0.775 | 0.838 | 0.602 |
| 5 | 30 | 1 | 0 | 10 | Relu | Adam | 0.777 | 0.850 | 0.586 |
| 6 | 20 | 1 | 0 | 20 | Tanh | SGD | 0.663 | 0.743 | 0.551 |
| 7 | 20 | 2 | 2 | 15 | Relu | Adam | 0.798 | 0.847 | 0.610 |
| 8 | 20 | 2 | 3 | 20 | Relu | Adam | 0.808 | 0.862 | 0.615 |
| 9 | 20 | 2 | 4 | 30 | Relu | Adam | 0.815 | 0.846 | 0.605 |
| 10 | 20 | 3 | 3 | 15 | Logistic | LBFGS | 0.861 | 0.848 | 0.585 |

Figure 6.0.8: Model selection processing with scoring for various hyperparameters and variable counts. The best performing model was a voting classifier with a neural network and random forest.

| Boosted Trees | Variables | Estimators | Max Depth | Learning Rate | Train | Validate | OOT |
|---|---|---|---|---|---|---|---|
| 1 | 10 | 500 | 2 | 0.1 | 0.918 | 0.895 | 0.632 |
| 2 | 15 | 500 | 2 | 0.1 | 0.919 | 0.899 | 0.635 |
| 3 | 20 | 500 | 2 | 0.1 | 0.933 | 0.916 | 0.640 |
| 4 | 25 | 500 | 2 | 0.1 | 0.934 | 0.925 | 0.633 |
| 5 | 30 | 500 | 2 | 0.1 | 0.941 | 0.927 | 0.638 |
| 6 | 20 | 600 | 3 | 0.01 | 0.837 | 0.903 | 0.635 |
| 7 | 20 | 750 | 4 | 0.5 | 1.000 | 0.932 | 0.584 |
| 8 | 20 | 800 | 3 | 0.1 | 0.999 | 0.935 | 0.623 |
| 9 | 20 | 400 | 2 | 0.1 | 0.918 | 0.916 | 0.639 |
| 10 | 20 | 100 | 5 | 0.01 | 0.760 | 0.774 | 0.615 |

| Random Forest | Variables | Estimators | Warm Start | Verbose | OOB Score | Min Samples Leaves | Train | Validate | OOT |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 500 | FALSE | 0 | FALSE | 2 | 1 | 0.929 | 0.634 |
| 2 | 15 | 500 | FALSE | 0 | FALSE | 2 | 1 | 0.912 | 0.630 |
| 3 | 20 | 500 | FALSE | 0 | FALSE | 2 | 1 | 0.920 | 0.654 |
| 4 | 25 | 500 | FALSE | 0 | FALSE | 2 | 1 | 0.925 | 0.648 |
| 5 | 30 | 500 | FALSE | 0 | FALSE | 2 | 1 | 0.925 | 0.638 |
| 6 | 20 | 600 | TRUE | 1 | FALSE | 3 | 1 | 0.912 | 0.653 |
| 7 | 20 | 750 | TRUE | 2 | TRUE | 4 | 1 | 0.914 | 0.654 |
| 8 | 20 | 800 | FALSE | 3 | TRUE | 3 | 1 | 0.912 | 0.657 |
| 9 | 20 | 500 | FALSE | 4 | TRUE | 3 | 1 | 0.916 | 0.657 |
| 10 | 20 | 100 | FALSE | 3 | FALSE | 2 | 1 | 0.925 | 0.646 |

| Voting Classifier | Variables | Models (Best from Above) | Train | Validate | OOT |
|---|---|---|---|---|---|
| 1 | 20 | Logistic Regression, Neural Network, Boosted Trees, Random Forest | 1 | 0.913 | 0.653 |
| 2 | 20 | Neural Network, Boosted Trees, Random Forest | 1 | 0.921 | 0.664 |
| 3 | 20 | Boosted Trees, Random Forest | 1 | 0.926 | 0.654 |
| 4 | 20 | Boosted Trees, Neural Network | 0.912 | 0.903 | 0.647 |
| 5 | 20 | Neural Network, Random Forest | 1 | 0.899 | 0.674 |

# 7. Summary and Conclusions

To sum up, we used the credit card transaction data with labels of fraudulent and not fraudulent to create algorithms to predict fraudulent records. This transaction data contained details about the transactions such as date of occurrence, card details, merchant details, and the amount. These fields were cleaned by removing errors, eliminating uninteresting records, and filling missing values.

5 Consolidated variables, 384 amount variables, 48 frequency variables, 8 day-since variables, 48 velocity change variables, and 2 target encoded variables were created for a total of 495 new variables to aid in predicting fraudulent transactions. In time variables and out of time variables were created by separating the dataset by transactions before and after September 1st. In time was before September 1st and out of time was after September first. Next, using the in-time data, these new variables were run through a feature selection process, beginning with a univariate filter to provide a quick usefulness ranking. The top 80 ranked variables were then passed through a stepwise selection wrapper that isolated the top 30 most useful variables.

Varying amounts of these variables were then used with hyperparameter testing and 4 different algorithms to produce the highest performing fraud detection model on the out of time data. This model was then extensively analyzed and used to produce predictions for the out of time data. These predictions were binned by fraud probability percentiles. It is imperative that the fraud models not misclassify a harmfully large amount of clean transactions as fraudulent to prevent lost business opportunities. Annoyed clients may choose to discontinue contracts due to transactions being canceled, which is harmful for both clients and the host business. However, it is also important to block as many fraudulent transactions as possible to reduce losses. Clients will also leave if fraudulent transactions remain unchecked. This is why it is so important to use fraud detection algorithms to strike a balance between these two extremes and to choose the right percentile as a cutoff value. Using the highest savings percentile of 9, $473,450 can be saved through the use of this fraud detection model.

Given more time, we would create more variables and further optimize our models to achieve higher OOT scores. We would also consult industry experts and produce more sophisticated code to streamline usage.

# 8. References


- Figure 5.1.1
  https://www.javatpoint.com/linear-regression-vs-logistic-regression-in-machine-learning
- Figure 5.2.1
  By Venkata Jagannath - https://community.tibco.com/wiki/random-forest-template-tibco-spotfirer-wiki-page
- Figure 5.4.1
  https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051

# Appendix I. Data Quality Report

## Appendix I-1. Data Set Description

- Name: Credit Card Transaction Data
- Purpose: Actual credit card purchases from a US government organization to look for credit card transaction fraud
- Source: US government organization
- Time period: January 1st, 2010 to December 31st, 2010
- Number of fields: 10
- Number of records: 96,753

## Appendix I-2. Data Set Summary

- Numeric Fields

| Item | Column name | # of records | % populated | Unique values | Mean | Standard deviation | Minimum value | Maximum value | # of zeros |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Amount | 96,753 | 100.00% | 34,909 | 427.89 | 10006.14 | 0.01 | 3,102,045.53 | 0 |

- Categorical Fields

| Item | Column name | # of records | % populated | Unique values | Most common field value |
|---|---|---|---|---|---|
| 1 | Recnum | 96,753 | 100.00% | 96,753 | N/A |
| 2 | Cardnum | 96,753 | 100.00% | 1,645 | 5142148452 |
| 3 | Date | 96,753 | 100.00% | 365 | 2/28/10 |
| 4 | Merchnum | 93,378 | 96.51% | 13,091 | 930090121224 |
| 5 | Merch description | 96,753 | 100.00% | 13,126 | GSA-FSS-ADV |
| 6 | Merch state | 95,558 | 98.76% | 227 | TN |
| 7 | Merch zip | 92,097 | 95.19% | 4,567 | 38118 |
| 8 | Transtype | 96,753 | 100.00% | 4 | P |
| 9 | Fraud | 96,753 | 100.00% | 2 | 0 |

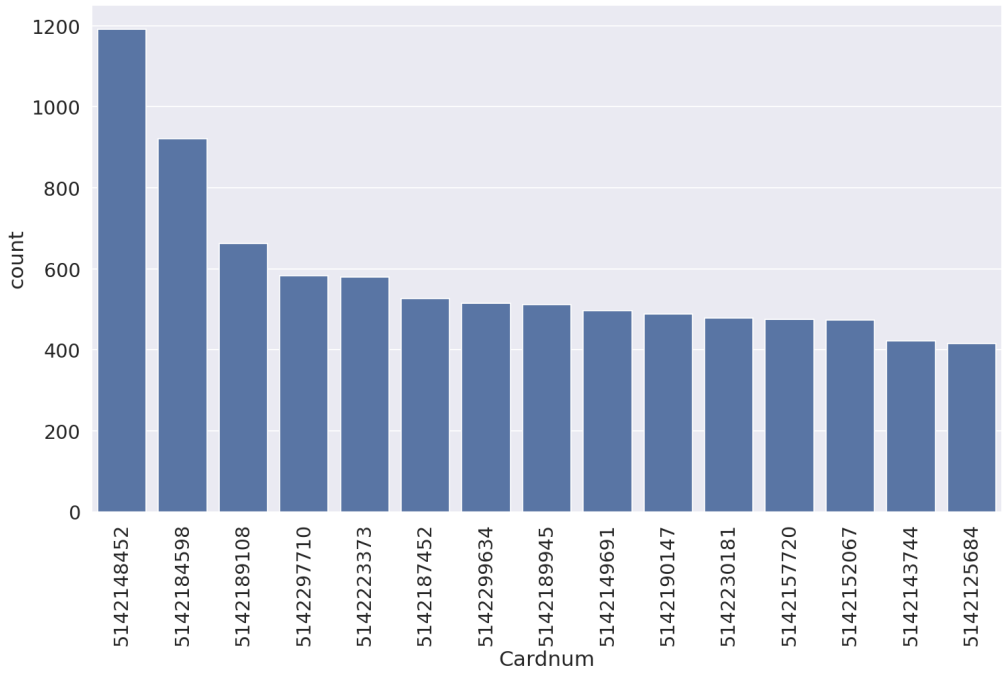
## Appendix I-3. Data Field Exploration

Field 8.

- Name: Recnum
- Description: unique identifier of each entry in the data

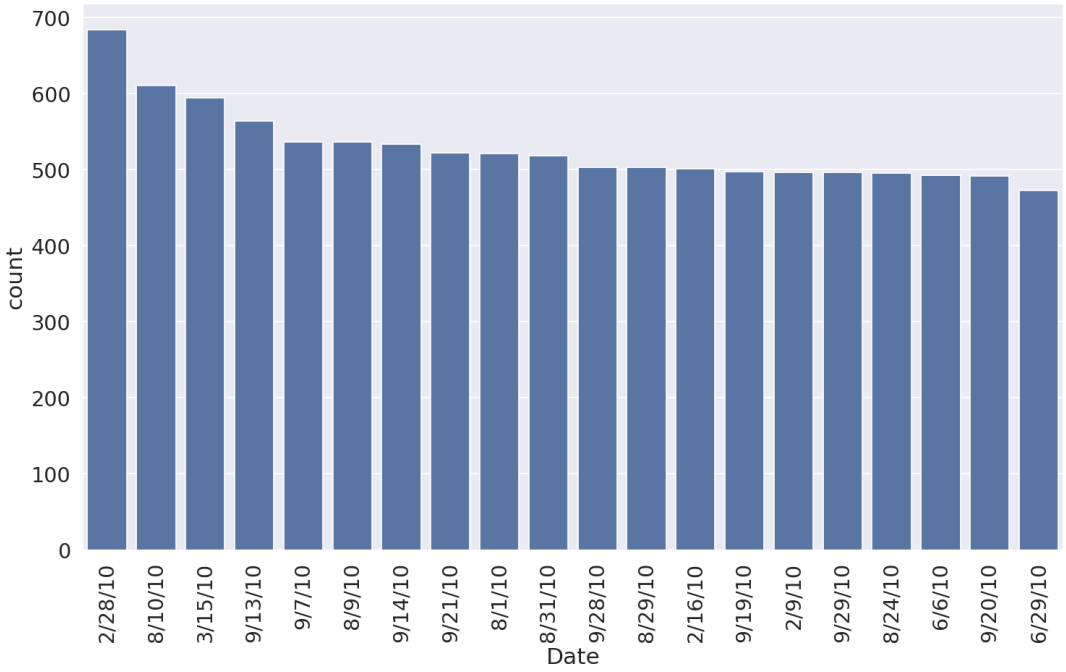
Field 9.

- Name: Cardnum
- Description: unique identifier of the card that made each purchase
- Count plot: (top 15 categories)



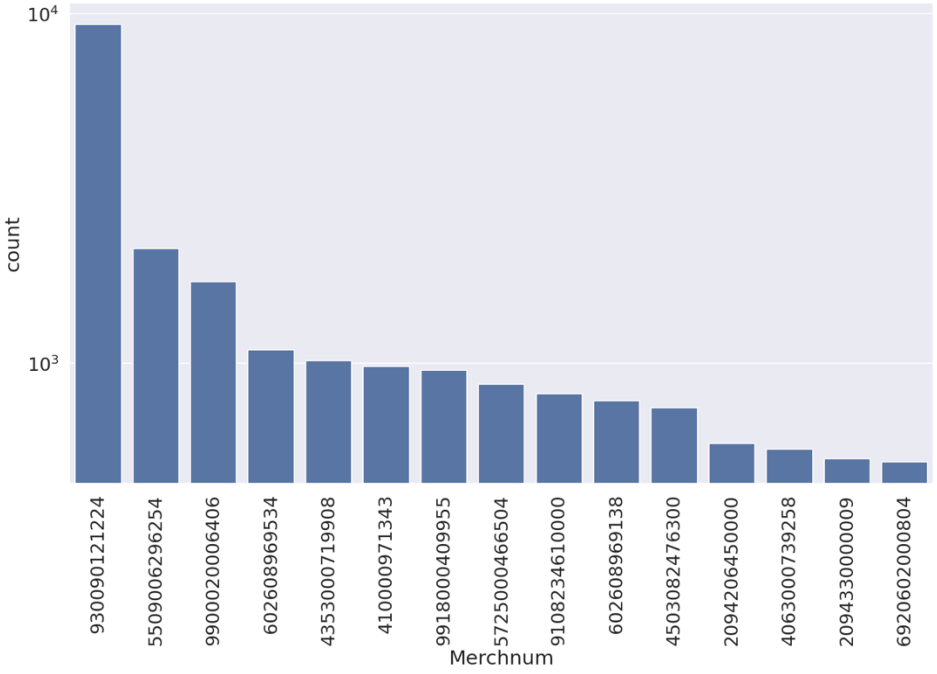

Field 10.

- Name: Date
- Description: the date when each transaction made
- Count plot: (top 20 categories)




Field 11.

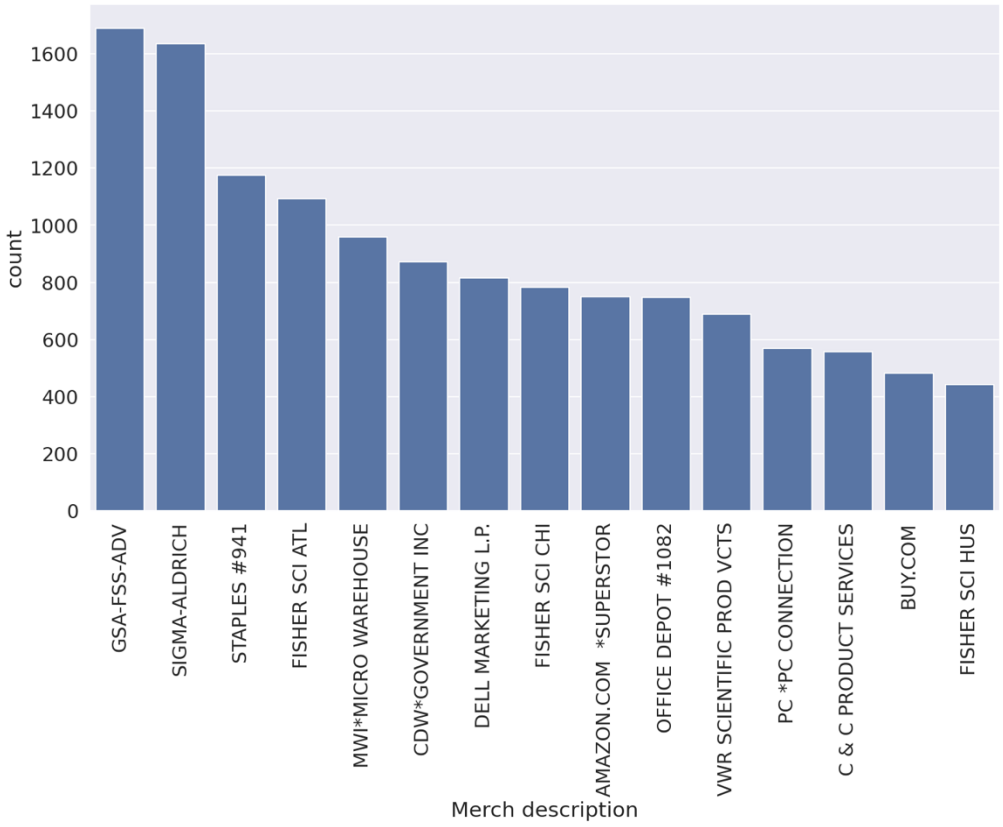
- Name: Merchnum
- Description: specific merchandise type number
- Count plot: (top 15 categories)




Field 12.

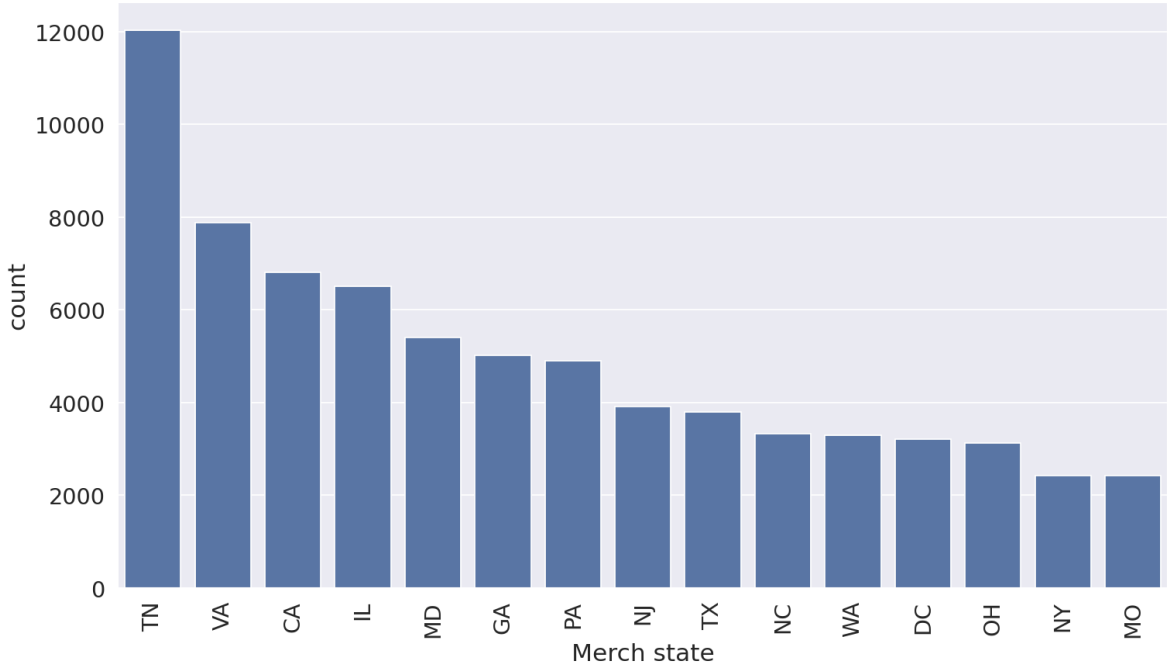
- Name: Merch description

- Description: description of the specific merchandise type
- Count plot: (top 15 categories)



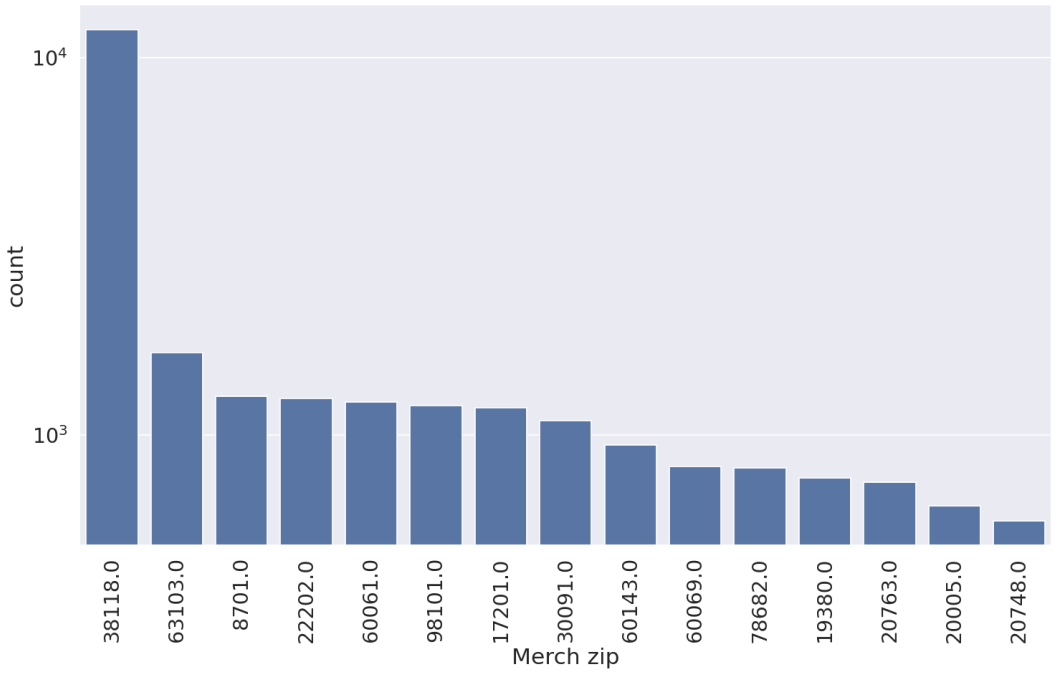

Field 13.

- Name: Merch state
- Description: the state of the merchandise
- Count plot: (top 15 categories)




Field 14.

- Name: Merch zip
- Description: the zip code of the location of the merchandise
- Count plot: (top 15 categories)

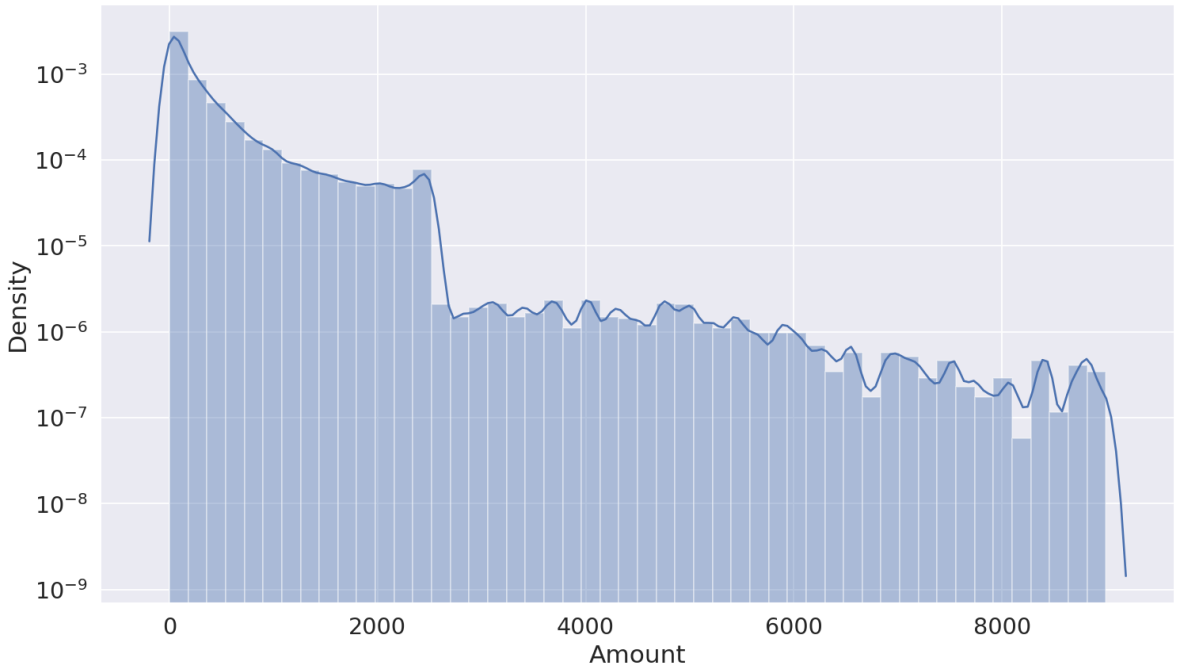


Field 15.

- Name: Transtype
- Description: the type of the transaction
- Count plot:

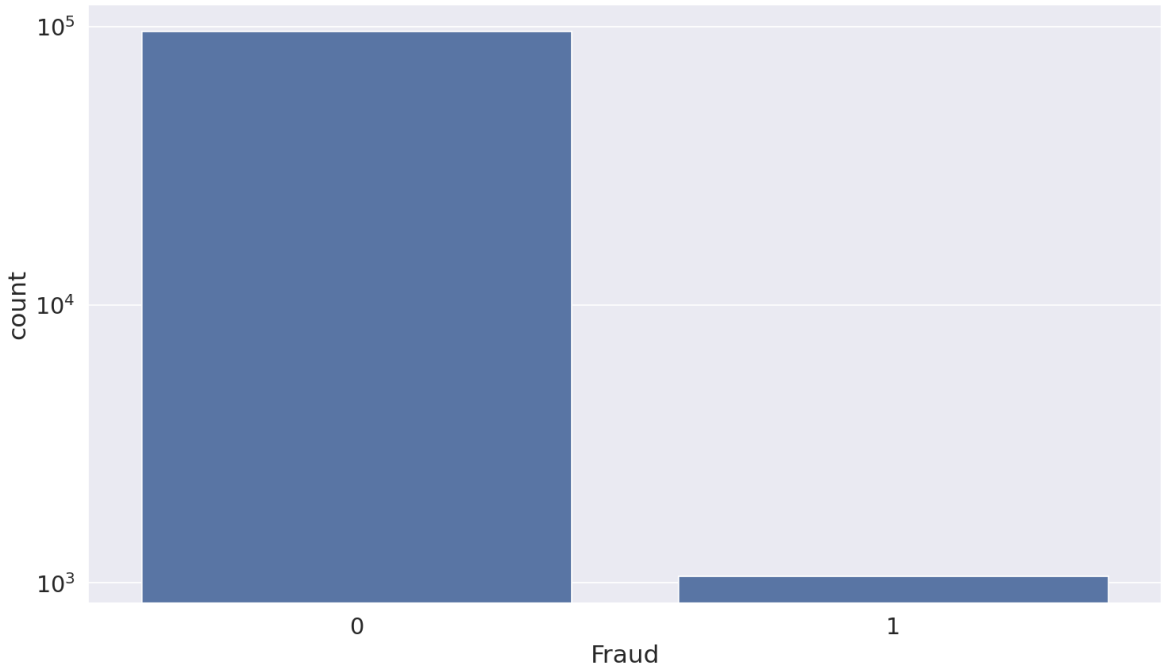

Field 16.

- Name: Amount
- Description: the amount of the transaction
- Histogram: (data in the histogram excludes outliers > 9000 and is 99.91% populated)




Field 17.

- Name: Fraud
- Description: indication of a fraud
- Count plot:



# Appendix II. Table of All Candidate Variables

| Variable Name | Variable Name | Variable Name | Variable Name | Variable Name |
|---|---|---|---|---|
| card_merch | card_zip | card_state | merchnum_zip | merchnum_state |
| Cardnum_day_since | Cardnum_count_0 | Cardnum_avg_0 | Cardnum_max_0 | Cardnum_med_0 |
| Cardnum_total_0 | Cardnum_actual/avg_0 | Cardnum_actual/max_0 | Cardnum_actual/med_0 | Cardnum_actual/toal_0 |
| Cardnum_count_1 | Cardnum_avg_1 | Cardnum_max_1 | Cardnum_med_1 | Cardnum_total_1 |
| Cardnum_actual/avg_1 | Cardnum_actual/max_1 | Cardnum_actual/med_1 | Cardnum_actual/toal_1 | Cardnum_count_3 |
| Cardnum_avg_3 | Cardnum_max_3 | Cardnum_med_3 | Cardnum_total_3 | Cardnum_actual/avg_3 |
| Cardnum_actual/max_3 | Cardnum_actual/med_3 | Cardnum_actual/toal_3 | Cardnum_count_7 | Cardnum_avg_7 |
| Cardnum_max_7 | Cardnum_med_7 | Cardnum_total_7 | Cardnum_actual/avg_7 | Cardnum_actual/max_7 |
| Cardnum_actual/med_7 | Cardnum_actual/toal_7 | Cardnum_count_14 | Cardnum_avg_14 | Cardnum_max_14 |
| Cardnum_med_14 | Cardnum_total_14 | Cardnum_actual/avg_14 | Cardnum_actual/max_14 | Cardnum_actual/med_14 |
| Cardnum_actual/toal_14 | Cardnum_count_30 | Cardnum_avg_30 | Cardnum_max_30 | Cardnum_med_30 |
| Cardnum_total_30 | Cardnum_actual/avg_30 | Cardnum_actual/max_30 | Cardnum_actual/med_30 | Cardnum_actual/toal_30 |
| merchnum_day_since | merchnum_count_0 | merchnum_avg_0 | merchnum_max_0 | merchnum_med_0 |
| merchnum_total_0 | merchnum_actual/avg_0 | merchnum_actual/max_0 | merchnum_actual/med_0 | merchnum_actual/toal_0 |
| merchnum_count_1 | merchnum_avg_1 | merchnum_max_1 | merchnum_med_1 | merchnum_total_1 |
| merchnum_actual/avg_1 | merchnum_actual/max_1 | merchnum_actual/med_1 | merchnum_actual/toal_1 | merchnum_count_3 |
| merchnum_avg_3 | merchnum_max_3 | merchnum_med_3 | merchnum_total_3 | merchnum_actual/avg_3 |
| merchnum_actual/max_3 | merchnum_actual/med_3 | merchnum_actual/toal_3 | merchnum_count_7 | merchnum_avg_7 |
| merchnum_max_7 | merchnum_med_7 | merchnum_total_7 | merchnum_actual/avg_7 | merchnum_actual/max_7 |

| Variable Name | Variable Name | Variable Name | Variable Name | Variable Name |
|---|---|---|---|---|
| merchnum_actual/med_7 | merchnum_actual/toal_7 | merchnum_count_14 | merchnum_avg_14 | merchnum_max_14 |
| merchnum_med_14 | merchnum_total_14 | merchnum_actual/avg_14 | merchnum_actual/max_14 | merchnum_actual/med_14 |
| merchnum_actual/toal_14 | merchnum_count_30 | merchnum_avg_30 | merchnum_max_30 | merchnum_med_30 |
| merchnum_total_30 | merchnum_actual/avg_30 | merchnum_actual/max_30 | merchnum_actual/med_30 | merchnum_actual/toal_30 |
| merch_zip_day_since | merch_zip_count_0 | merch_zip_avg_0 | merch_zip_max_0 | merch_zip_med_0 |
| merch_zip_total_0 | merch_zip_actual/avg_0 | merch_zip_actual/max_0 | merch_zip_actual/med_0 | merch_zip_actual/toal_0 |
| merch_zip_count_1 | merch_zip_avg_1 | merch_zip_max_1 | merch_zip_med_1 | merch_zip_total_1 |
| merch_zip_actual/avg_1 | merch_zip_actual/max_1 | merch_zip_actual/med_1 | merch_zip_actual/toal_1 | merch_zip_count_3 |
| merch_zip_avg_3 | merch_zip_max_3 | merch_zip_med_3 | merch_zip_total_3 | merch_zip_actual/avg_3 |
| merch_zip_actual/max_3 | merch_zip_actual/med_3 | merch_zip_actual/toal_3 | merch_zip_count_7 | merch_zip_avg_7 |
| merch_zip_max_7 | merch_zip_med_7 | merch_zip_total_7 | merch_zip_actual/avg_7 | merch_zip_actual/max_7 |
| merch_zip_actual/med_7 | merch_zip_actual/toal_7 | merch_zip_count_14 | merch_zip_avg_14 | merch_zip_max_14 |
| merch_zip_med_14 | merch_zip_total_14 | merch_zip_actual/avg_14 | merch_zip_actual/max_14 | merch_zip_actual/med_14 |
| merch_zip_actual/toal_14 | merch_zip_count_30 | merch_zip_avg_30 | merch_zip_max_30 | merch_zip_med_30 |
| merch_zip_total_30 | merch_zip_actual/avg_30 | merch_zip_actual/max_30 | merch_zip_actual/med_30 | merch_zip_actual/toal_30 |
| card_merch_day_since | card_merch_count_0 | card_merch_avg_0 | card_merch_max_0 | card_merch_med_0 |
| card_merch_total_0 | card_merch_actual/avg_0 | card_merch_actual/max_0 | card_merch_actual/med_0 | card_merch_actual/toal_0 |
| card_merch_count_1 | card_merch_avg_1 | card_merch_max_1 | card_merch_med_1 | card_merch_total_1 |
| card_merch_actual/avg_1 | card_merch_actual/max_1 | card_merch_actual/med_1 | card_merch_actual/toal_1 | card_merch_count_3 |
| card_merch_avg_3 | card_merch_max_3 | card_merch_med_3 | card_merch_total_3 | card_merch_actual/avg_3 |

| Variable Name | Variable Name | Variable Name | Variable Name | Variable Name |
|---|---|---|---|---|
| card_merch_actual/max_3 | card_merch_actual/med_3 | card_merch_actual/total_3 | card_merch_count_7 | card_merch_avg_7 |
| card_merch_max_7 | card_merch_med_7 | card_merch_total_7 | card_merch_actual/avg_7 | card_merch_actual/max_7 |
| card_merch_actual/med_7 | card_merch_actual/total_7 | card_merch_count_14 | card_merch_avg_14 | card_merch_max_14 |
| card_merch_med_14 | card_merch_total_14 | card_merch_actual/avg_14 | card_merch_actual/max_14 | card_merch_actual/med_14 |
| card_merch_actual/toal_14 | card_merch_count_30 | card_merch_avg_30 | card_merch_max_30 | card_merch_med_30 |
| card_merch_total_30 | card_merch_actual/avg_30 | card_merch_actual/max_30 | card_merch_actual/med_30 | card_merch_actual/total_30 |
| card_zip_day_since | card_zip_count_0 | card_zip_avg_0 | card_zip_max_0 | card_zip_med_0 |
| card_zip_total_0 | card_zip_actual/avg_0 | card_zip_actual/max_0 | card_zip_actual/med_0 | card_zip_actual/toal_0 |
| card_zip_count_1 | card_zip_avg_1 | card_zip_max_1 | card_zip_med_1 | card_zip_total_1 |
| card_zip_actual/avg_1 | card_zip_actual/max_1 | card_zip_actual/med_1 | card_zip_actual/toal_1 | card_zip_count_3 |
| card_zip_avg_3 | card_zip_max_3 | card_zip_med_3 | card_zip_total_3 | card_zip_actual/avg_3 |
| card_zip_actual/max_3 | card_zip_actual/med_3 | card_zip_actual/toal_3 | card_zip_count_7 | card_zip_avg_7 |
| card_zip_max_7 | card_zip_med_7 | card_zip_total_7 | card_zip_actual/avg_7 | card_zip_actual/max_7 |
| card_zip_actual/med_7 | card_zip_actual/toal_7 | card_zip_count_14 | card_zip_avg_14 | card_zip_max_14 |
| card_zip_med_14 | card_zip_total_14 | card_zip_actual/avg_14 | card_zip_actual/max_14 | card_zip_actual/med_14 |
| card_zip_actual/toal_14 | card_zip_count_30 | card_zip_avg_30 | card_zip_max_30 | card_zip_med_30 |
| card_zip_total_30 | card_zip_actual/avg_30 | card_zip_actual/max_30 | card_zip_actual/med_30 | card_zip_actual/toal_30 |
| card_state_day_since | card_state_count_0 | card_state_avg_0 | card_state_max_0 | card_state_med_0 |
| card_state_total_0 | card_state_actual/avg_0 | card_state_actual/max_0 | card_state_actual/med_0 | card_state_actual/toal_0 |
| card_state_count_1 | card_state_avg_1 | card_state_max_1 | card_state_med_1 | card_state_total_1 |

| Variable Name | Variable Name | Variable Name | Variable Name | Variable Name |
|---|---|---|---|---|
| card_state_actual/avg_1 | card_state_actual/max_1 | card_state_actual/med_1 | card_state_actual/toal_1 | card_state_count_3 |
| card_state_avg_3 | card_state_max_3 | card_state_med_3 | card_state_total_3 | card_state_actual/avg_3 |
| card_state_actual/max_3 | card_state_actual/med_3 | card_state_actual/toal_3 | card_state_count_7 | card_state_avg_7 |
| card_state_max_7 | card_state_med_7 | card_state_total_7 | card_state_actual/avg_7 | card_state_actual/max_7 |
| card_state_actual/med_7 | card_state_actual/toal_7 | card_state_count_14 | card_state_avg_14 | card_state_max_14 |
| card_state_med_14 | card_state_total_14 | card_state_actual/avg_14 | card_state_actual/max_14 | card_state_actual/med_14 |
| card_state_actual/toal_14 | card_state_count_30 | card_state_avg_30 | card_state_max_30 | card_state_med_30 |
| card_state_total_30 | card_state_actual/avg_30 | card_state_actual/max_30 | card_state_actual/med_30 | card_state_actual/toal_30 |
| merchnum_zip_day_since | merchnum_zip_count_0 | merchnum_zip_avg_0 | merchnum_zip_max_0 | merchnum_zip_med_0 |
| merchnum_zip_total_0 | merchnum_zip_actual/avg_0 | merchnum_zip_actual/max_0 | merchnum_zip_actual/med_0 | merchnum_zip_actual/toal_0 |
| merchnum_zip_count_1 | merchnum_zip_avg_1 | merchnum_zip_max_1 | merchnum_zip_med_1 | merchnum_zip_total_1 |
| merchnum_zip_actual/avg_1 | merchnum_zip_actual/max_1 | merchnum_zip_actual/med_1 | merchnum_zip_actual/toal_1 | merchnum_zip_count_3 |
| merchnum_zip_avg_3 | merchnum_zip_max_3 | merchnum_zip_med_3 | merchnum_zip_total_3 | merchnum_zip_actual/avg_3 |
| merchnum_zip_actual/max_3 | merchnum_zip_actual/med_3 | merchnum_zip_actual/toal_3 | merchnum_zip_count_7 | merchnum_zip_avg_7 |
| merchnum_zip_max_7 | merchnum_zip_med_7 | merchnum_zip_total_7 | merchnum_zip_actual/avg_7 | merchnum_zip_actual/max_7 |
| merchnum_zip_actual/med_7 | merchnum_zip_actual/toal_7 | merchnum_zip_count_14 | merchnum_zip_avg_14 | merchnum_zip_max_14 |
| merchnum_zip_med_14 | merchnum_zip_total_14 | merchnum_zip_actual/avg_14 | merchnum_zip_actual/max_14 | merchnum_zip_actual/med_14 |
| merchnum_zip_actual/toal_14 | merchnum_zip_count_30 | merchnum_zip_avg_30 | merchnum_zip_max_30 | merchnum_zip_med_30 |
| merchnum_zip_total_30 | merchnum_zip_actual/avg_30 | merchnum_zip_actual/max_30 | merchnum_zip_actual/med_30 | merchnum_zip_actual/toal_30 |
| merchnum_state_day_since | merchnum_state_count_0 | merchnum_state_avg_0 | merchnum_state_max_0 | merchnum_state_med_0 |

| Variable Name | Variable Name | Variable Name | Variable Name | Variable Name |
|---|---|---|---|---|
| merchnum_state_total_0 | merchnum_state_actual/avg_0 | merchnum_state_actual/max_0 | merchnum_state_actual/med_0 | merchnum_state_actual/toal_0 |
| merchnum_state_count_1 | merchnum_state_avg_1 | merchnum_state_max_1 | merchnum_state_med_1 | merchnum_state_total_1 |
| merchnum_state_actual/avg_1 | merchnum_state_actual/max_1 | merchnum_state_actual/med_1 | merchnum_state_actual/toal_1 | merchnum_state_count_3 |
| merchnum_state_avg_3 | merchnum_state_max_3 | merchnum_state_med_3 | merchnum_state_total_3 | merchnum_state_actual/avg_3 |
| merchnum_state_actual/max_3 | merchnum_state_actual/med_3 | merchnum_state_actual/toal_3 | merchnum_state_count_7 | merchnum_state_avg_7 |
| merchnum_state_max_7 | merchnum_state_med_7 | merchnum_state_total_7 | merchnum_state_actual/avg_7 | merchnum_state_actual/max_7 |
| merchnum_state_actual/med_7 | merchnum_state_actual/toal_7 | merchnum_state_count_14 | merchnum_state_avg_14 | merchnum_state_max_14 |
| merchnum_state_med_14 | merchnum_state_total_14 | merchnum_state_actual/avg_14 | merchnum_state_actual/max_14 | merchnum_state_actual/med_14 |
| merchnum_state_actual/toal_14 | merchnum_state_count_30 | merchnum_state_avg_30 | merchnum_state_max_30 | merchnum_state_med_30 |
| merchnum_state_total_30 | merchnum_state_actual/avg_30 | merchnum_state_actual/max_30 | merchnum_state_actual/med_30 | merchnum_state_actual/toal_30 |
| Cardnum_count_0_by_7 | Cardnum_count_0_by_14 | Cardnum_count_0_by_30 | Cardnum_count_1_by_7 | Cardnum_count_1_by_14 |
| Cardnum_count_1_by_30 | merchnum_count_0_by_7 | merchnum_count_0_by_14 | merchnum_count_0_by_30 | merchnum_count_1_by_7 |
| merchnum_count_1_by_14 | merchnum_count_1_by_30 | merch_zip_count_0_by_7 | merch_zip_count_0_by_14 | merch_zip_count_0_by_30 |
| merch_zip_count_1_by_7 | merch_zip_count_1_by_14 | merch_zip_count_1_by_30 | card_merch_count_0_by_7 | card_merch_count_0_by_14 |
| card_merch_count_0_by_30 | card_merch_count_1_by_7 | card_merch_count_1_by_14 | card_merch_count_1_by_30 | card_zip_count_0_by_7 |
| card_zip_count_0_by_14 | card_zip_count_0_by_30 | card_zip_count_1_by_7 | card_zip_count_1_by_14 | card_zip_count_1_by_30 |
| card_state_count_0_by_7 | card_state_count_0_by_14 | card_state_count_0_by_30 | card_state_count_1_by_7 | card_state_count_1_by_14 |
| card_state_count_1_by_30 | merchnum_zip_count_0_by_7 | merchnum_zip_count_0_by_14 | merchnum_zip_count_0_by_30 | merchnum_zip_count_1_by_7 |
| merchnum_zip_count_1_by_14 | merchnum_zip_count_1_by_30 | merchnum_state_count_0_by_7 | merchnum_state_count_0_by_14 | merchnum_state_count_0_by_30 |
| merchnum_state_count_1_by_7 | merchnum_state_count_1_by_14 | merchnum_state_count_1_by_30 | day_fraud_prob | state_fraud_prob |

# Appendix III. Variable Statistics

| Variable Name | Max | Min | Mean | Standard Deviation |
|---|---|---|---|---|
| card_zip_total_7 | 306633.41 | 0.01 | 710.2010873 | 4111.683788 |
| card_zip_total_3 | 306633.41 | 0.01 | 641.8643164 | 4066.681882 |
| card_state_total_7 | 306633.41 | 0.01 | 901.8398092 | 4245.027357 |
| card_merch_total_14 | 306633.41 | 0.01 | 798.2842114 | 4184.495688 |
| card_merch_total_3 | 306633.41 | 0.01 | 639.9634921 | 4065.295416 |
| card_zip_total_1 | 306633.41 | 0.01 | 605.9346134 | 4022.737371 |
| card_zip_max_30 | 47900 | 0.01 | 533.9894551 | 1086.531219 |
| card_state_max_7 | 47900 | 0.01 | 540.1076961 | 1121.831453 |
| card_state_max_14 | 47900 | 0.01 | 604.5318782 | 1190.44609 |
| cardnum_total_7 | 312616.06 | 0.14 | 2384.036098 | 7158.500841 |
| card_zip_max_3 | 47900 | 0.01 | 446.3101365 | 1018.056985 |
| card_merch_max_7 | 47900 | 0.01 | 465.7542958 | 1028.76437 |
| card_zip_max_1 | 47900 | 0.01 | 435.4758389 | 1011.950734 |
| card_state_max_1 | 47900 | 0.01 | 457.4100003 | 1030.02531 |
| cardnum_total_0 | 218301.83 | 0.01 | 741.6455649 | 3431.446131 |
| merchnum_state_total_1 | 306633.41 | 0.01 | 1192.065963 | 4437.529402 |
| merchnum_zip_total_0 | 217467.18 | 0.01 | 807.0714059 | 2881.862756 |
| card_merch_max_0 | 47900 | 0.01 | 423.0262353 | 936.0123468 |
| cardnum_max_0 | 47900 | 0.01 | 498.2058087 | 1030.95736 |
| merchnum_zip_max_3 | 47900 | 0.01 | 706.487517 | 1319.096124 |