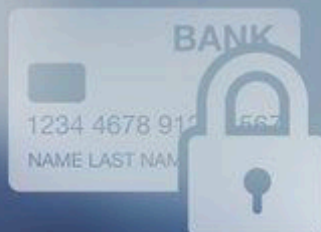
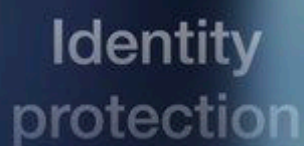


 Credit card
protection**FRAUD
PREVENTION**

Identity Fraud Identification

PREPARED BY USC CONSULTING GROUP

Identity
protection

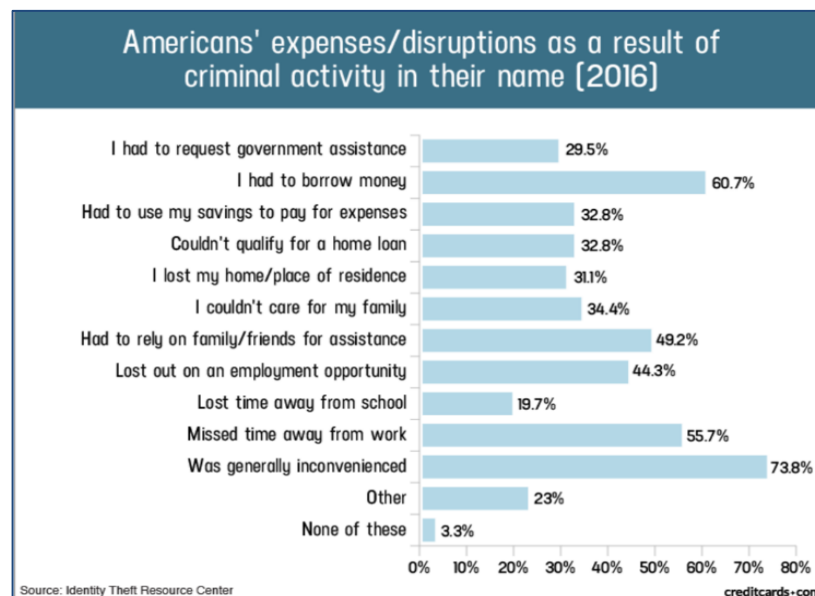
TEAM 6

Shreerang Javadekar | Yuanyu Mu | James Lee
Liana Liang | Vandik Zaveri | Yadi Gong
April 2019

Project Motivation

Identity theft or identity fraud is a type of fraud where someone uses another person's personal information without authorization or creates a synthetic identity to commit a crime or to deceive or defraud that other person or a third person. Typically, such instances result in undue extraction of credit out of the system at the expense of other entities. These other entities may be the government, companies or the common folk.

According to Statista, by September 2017, 22% of the total internet users in the United States had been victims of identity theft. 158 Million Social Security Numbers were breached in 2017 alone, making 49% of the total population of United States vulnerable to identity fraud.



These statistics explain the gravity of identity fraud in the United States. Overall, identity fraud has costed consumers over \$16 billion dollars in the past few years.

This report mainly focuses on building a data-driven algorithm to detect real-time identity fraud with an aim to provide actionable insights to the business.

Table of Contents

Executive Summary	1
Description of Data	3
Data Preparation.....	10
Variable Creation	13
Feature Selection.....	17
Algorithm - Model Building	23
Algorithm – Model Comparison.....	28
Results.....	32
Conclusion	37
Appendix 1: Data Quality Report	38
Appendix 2: Glossary of Variables.....	55

Executive Summary

This report has been prepared by the USC Consulting Group for Dr. Stephen Coggeshall, owner of the “Application Fraud” dataset. The report details an end-to-end algorithm that uses supervised machine learning methods with the purpose of identifying fraudulent transactions in the dataset.

Synthetic Identity Fraud is the process of fabricating identities of innocent individuals and using their identities to apply for credit cards and other monetary services. Typically, fabricating identities include either buying somebody’s Social Security Number (SSN) online or entering a random number as the SSN and hoping it would be undetected. Surprisingly, it does go undetected and this type of frauds is growing fast since the past few years. They can cost millions of dollars to the business, and hence we build a product that does an exceptional job in detecting, explaining and preventing these frauds.

A thorough process of data cleaning, identifying and creating relevant variables, dimensionality reduction has been carried out before building four supervised learning models – logistic regression, neural networks, boosted trees and random forests – to calculate the likelihood of each transaction being fraudulent. The models were trained over a subset of the data and then validated over a testing set as well as an out-of-time set to see how the models work over data they have never seen before. A detailed explanation of the same follows as one proceeds through the report. A Data Quality Report, which was aligned beforehand with Dr. Coggeshall, has been supplemented in the appendix for reference.

Based on our findings, we note that fraudsters tend to have one or more these following characteristics:

1. They tend to use some bit of their personal information while filling in the application. It could be either of phone number, address, email id, etc.
2. A high number of applications start coming in within a short period of time.
3. One SSN can have applications coming in from anywhere and everywhere.

A summary of the model performance has been documented in the Results section. Every model scores the entries on its likelihood of being fraudulent. The entries are then ordered in a decreasing

level of likelihood. The model's performance has been evaluated on its actual "Fraud Detection Rate" in the top 3% of this ordered dataset.

Based on the performance of these models, the final model of Random Forests has been selected on the basis of its highest fraud-detection rate. An approximation of how much savings this model would offer based on its precision has been calculated in the Results section for further business understanding.

Description of Data

1. Brief on the Dataset

The dataset on hand consists of 1,000,000 applications made using 835,819 unique Social Security Numbers. The uniqueness of these 835,819 Social Security Numbers, however, is questionable.

We have a total of 10 unique fields in the dataset (including record number) that describe personal identification details of every single application – details including name, Social Security Number, Address, Zipcode and Phone Number. Details on the date when the application was made are also available. A label is also available that indicates whether the application is fraudulent or not.

Every field is 100% filled. There are no blank entries in the dataset. Further details have been mentioned in the Summary Table.

2. Summary Tables

Below is a summary table for all the fields in the dataset. All the fields have been treated as categorical.

Total number of records: 1,000,000

Total number of fields: 10.

Unique Identifiers for Every Record: Record Number (Range: 1-1,000,000).

Table 1: Summary of numerical variables

Field/Metric	# Records with a Value	% Populated	# Unique Categories	Most Common Value
Record	1,000,000	100%	1,000,000	Every value appears just once.
Date (date)	1,000,000	100%	365	"2016-08-16"
Social Security Number (ssn)	1,000,000	100%	835,819	"999-99-9999"
First Name (firstname)	1,000,000	100%	78,136	"EAMSTRMT"
Last Name (lastname)	1,000,000	100%	177,001	"ERJSAXA"
Address (address)	1,000,000	100%	828,774	"123 MAIN ST"
Zipcode (zip5)	1,000,000	100%	26,370	"68138"
Date of Birth (dob)	1,000,000	100%	42,673	"1907-06-26"
Phone Number (homephone)	1,000,000	100%	28,244	"999-999-9999"
Fraud Label (fraud_label)	1,000,000	100%	2	"0"

3. Exploratory Analysis for Important Fields:

Below is an exploratory analysis for some of the important fields in the dataset. Adequate visualizations have been added to supplement the description. A comprehensive Data Quality Report previously aligned with Dr. Coggeshall has been supplemented in the Appendix for reference.

3.1 Record (record)

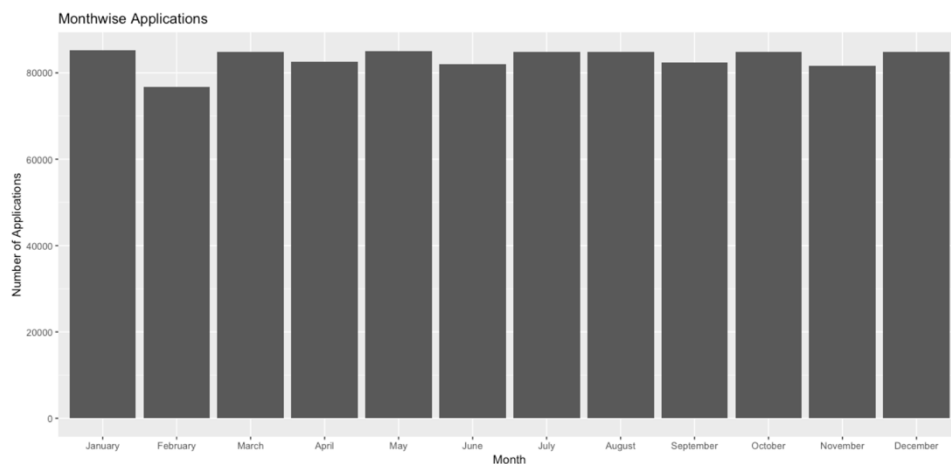
“record” is a unique identifier for every entry. The record indexing is numerical. It starts at 1 and ends at 1000000 with step increases of 1

3.2 Date (date)

“date” is an indicator for the date when the application was filed.

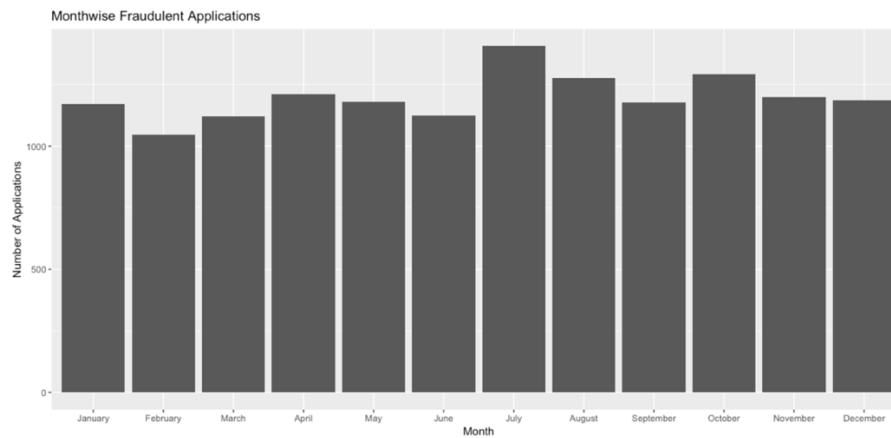
100% of this field is filled. The dates in this field are the 366 days of the leap year 2016. However, there are no applications for the date of 02/29/2016.

A monthwise distribution of the number of applications (both fraudulent and non-fraudulent) is as below:

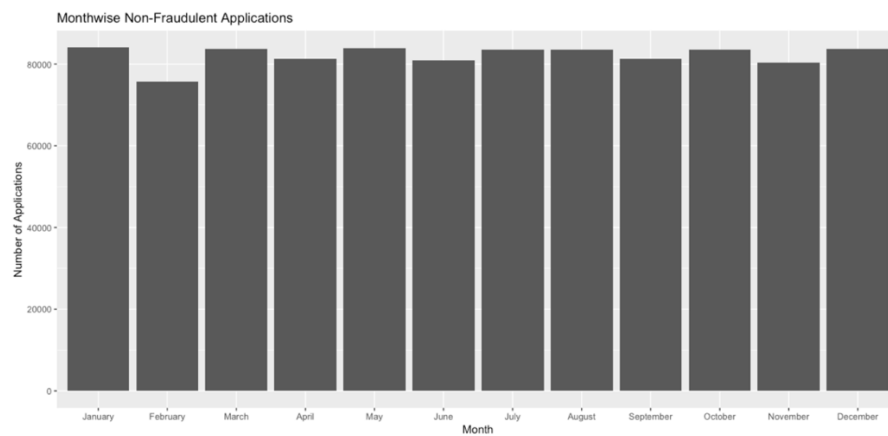


The distribution for number of applications seems to be uniform across all months with just a small dip in the month of February.

For fraudulent applications the distribution is as below:

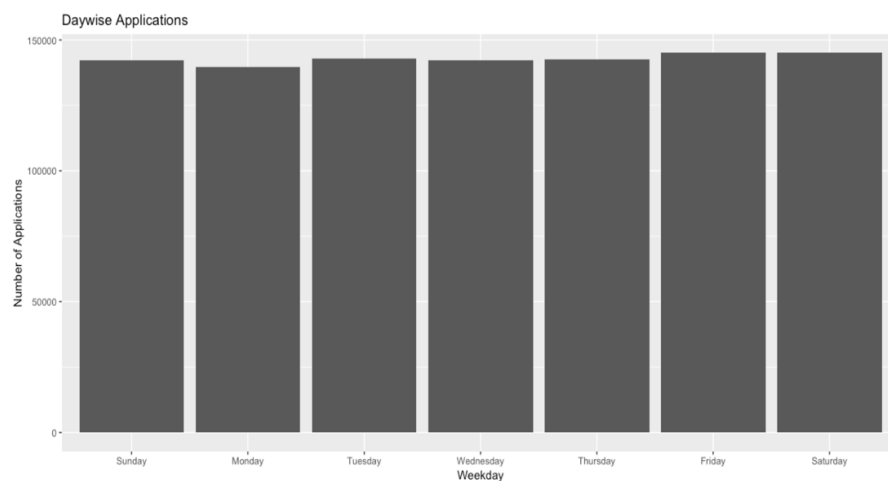


The month of July seems to have witnessed the highest number of fraudulent applications.

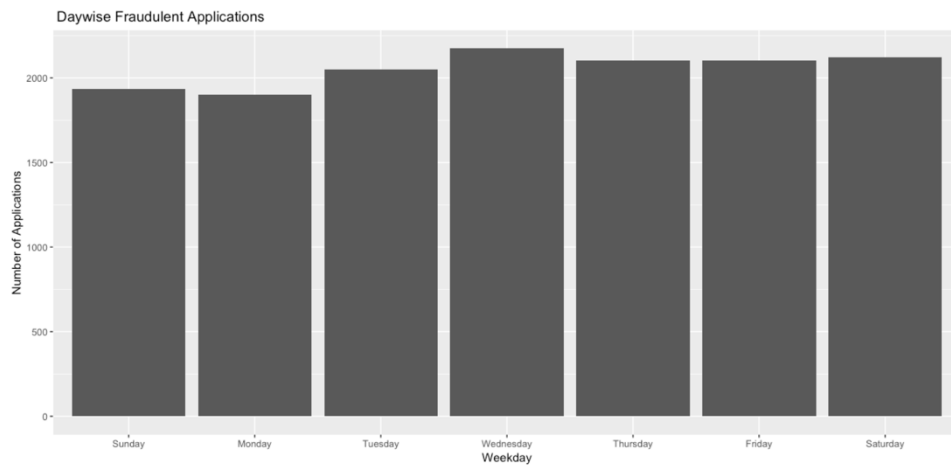


There is no significant trend in the non-fraudulent applications across months.

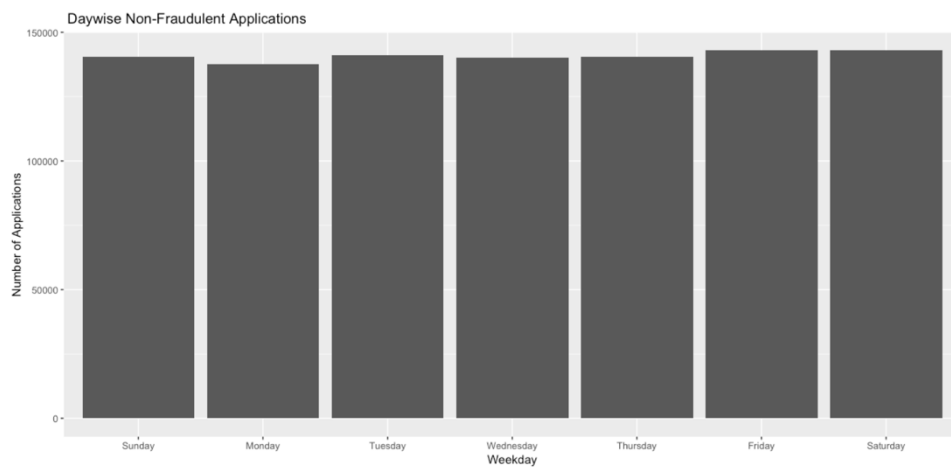
A day wise distribution of all the transactions is as follows:



For fraudulent applications, the daywise distribution is as below:

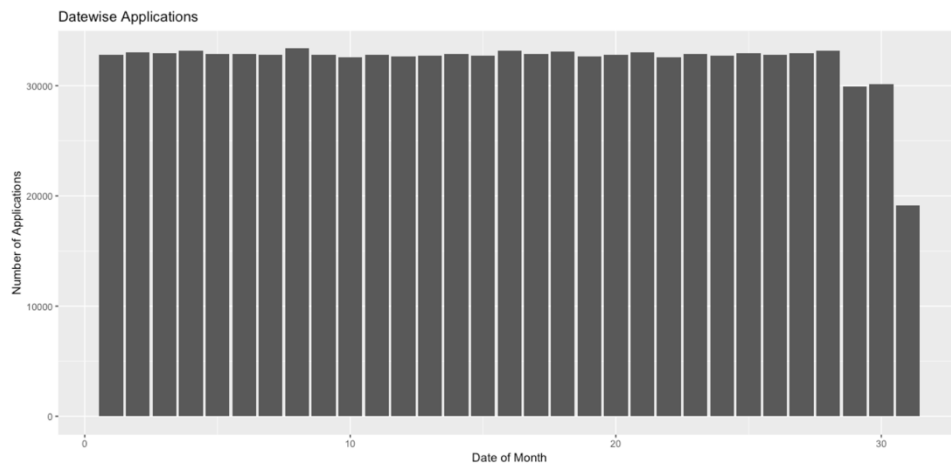


For non-fraudulent applications, the daywise distribution is as below:



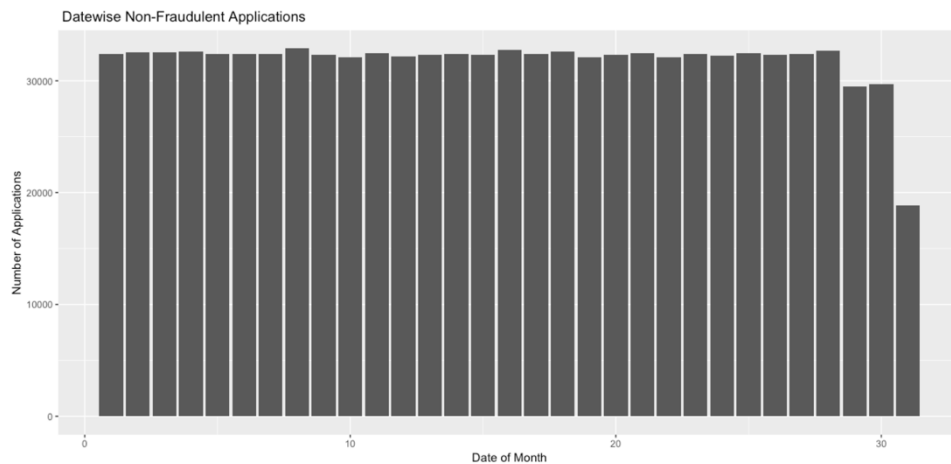
There is no significant trend in the daywise distribution across both fraudulent and non-fraudulent applications.

A datewise distribution of the number of applications (both fraudulent and non-fraudulent) is as below:

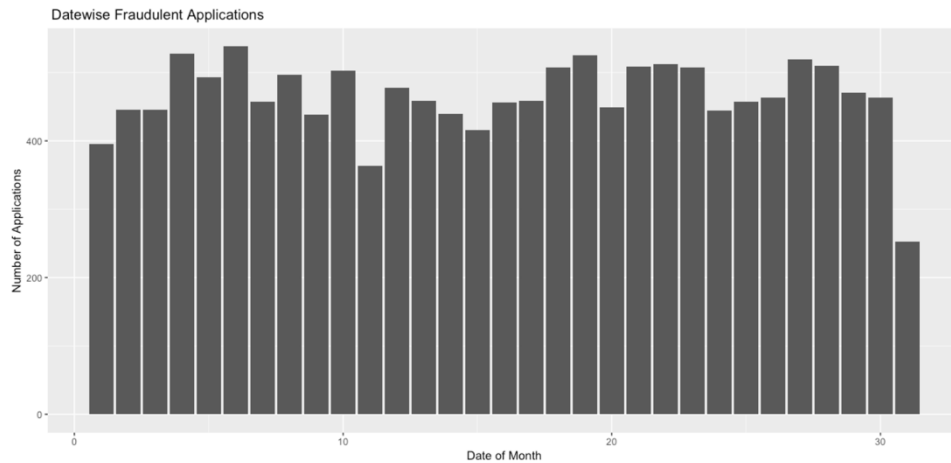


There seems to be a dip in the number of transactions nearing the month. For 31st, the number is very low because of the few months that have the 31st date.

For non-fraudulent applications, the distribution is as below:



For fraudulent applications, the distribution is as below:

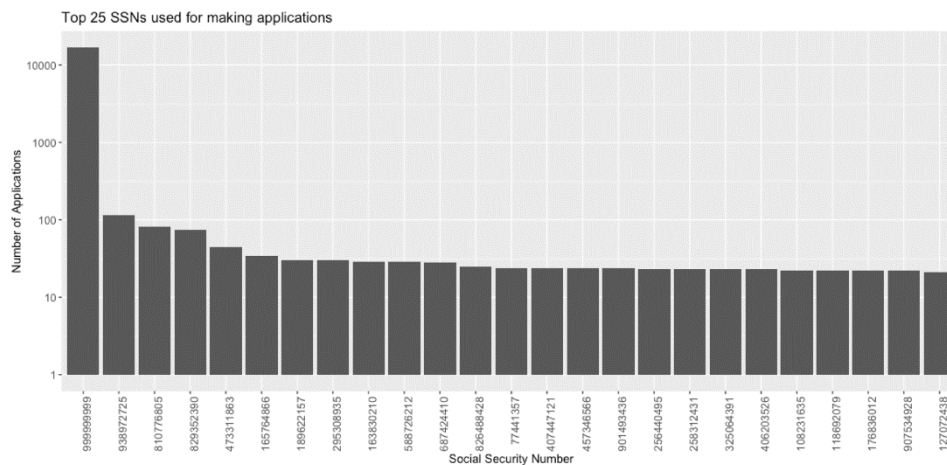


3.3 Social Security Number (ssn)

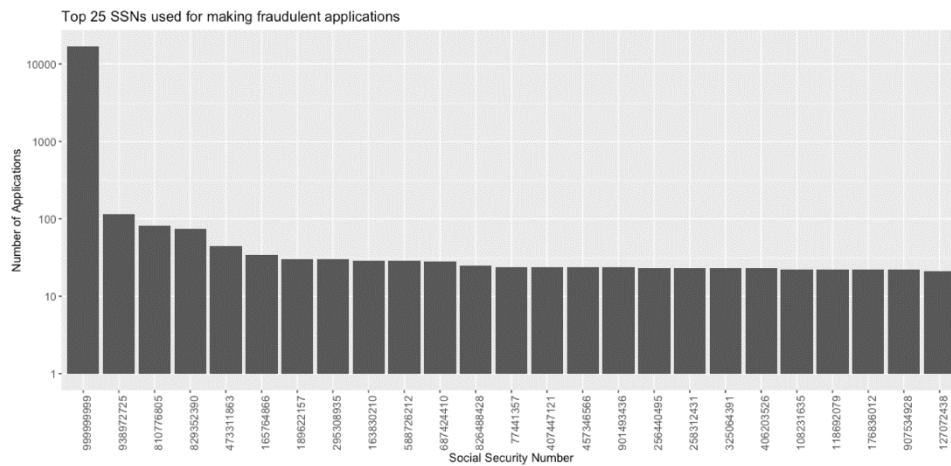
“ssn” is an indicator of the Social Security Number used for making the application. Amongst all the identity levels available for every individual, given that this field is unique to every individual, following is how the distribution for this field looks like:

There are 835,819 unique Social Security numbers in the dataset. Their uniqueness and authenticity, however, is questionable. 100% of the dataset is filled.

A distribution of the top 25 Social Security numbers used for making the applications is as follows. The Y-axis is in log scale.



The top 25 Social Security Numbers used for making fraudulent applications is as follows:



It is clear that the “999-99-9999” is a proxy number being used for making applications to mask one’s identity.

3.4 Fraud Indicator (fraud_label):

“fraud_label” is an indicator to indicate whether the application is fraudulent or not. It is a binary classifier with two categories.

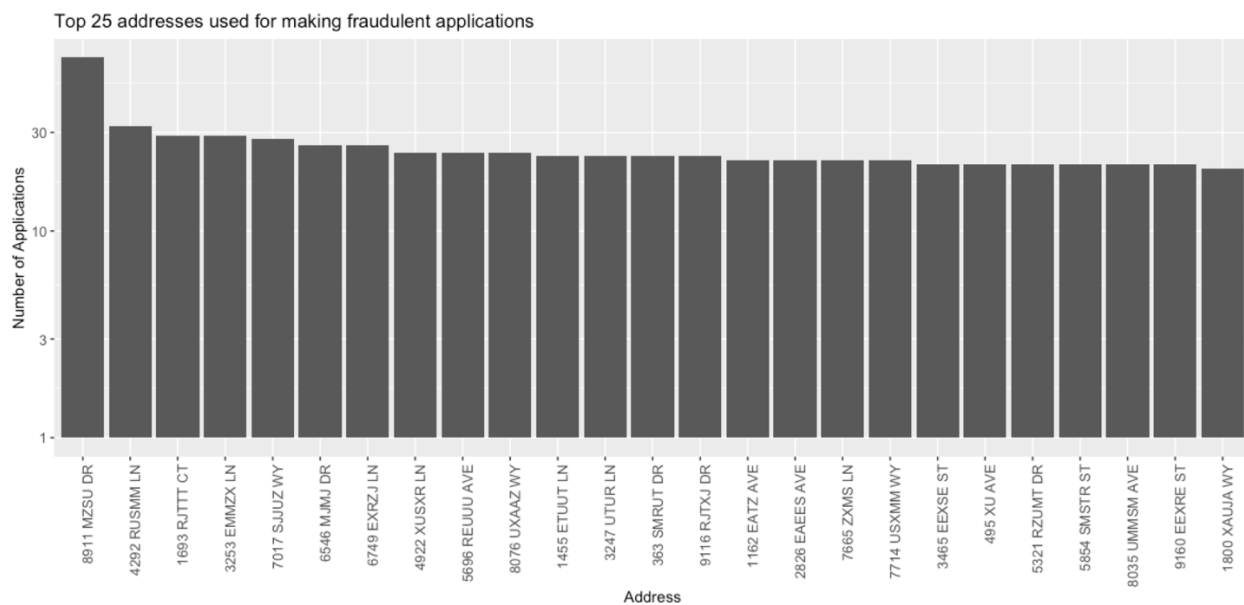
100% of this column is filled.

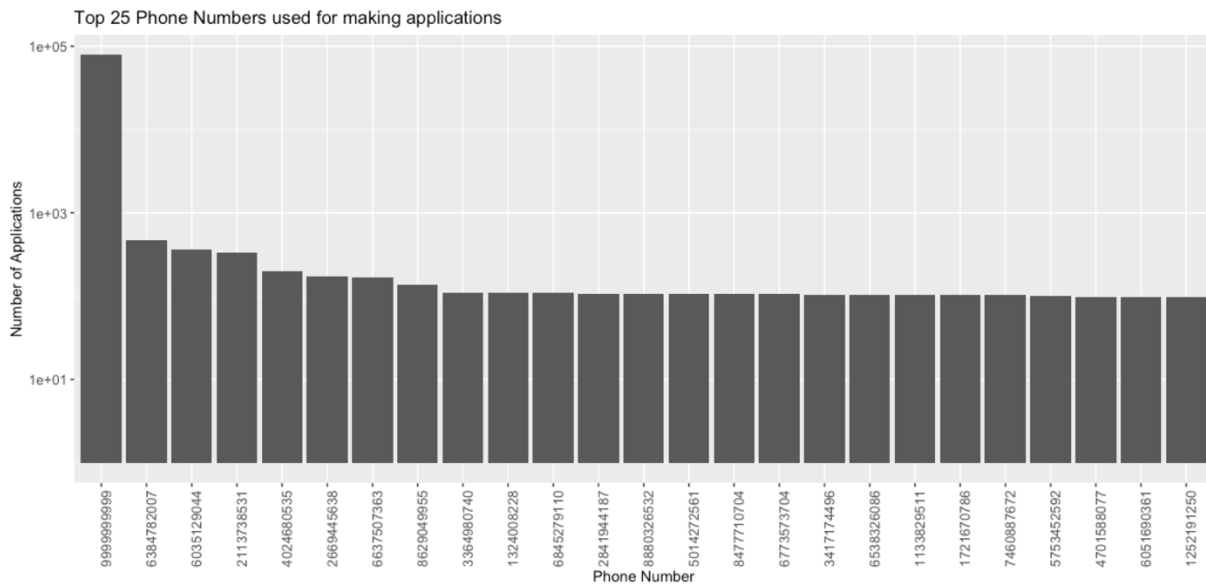
A summary of this field is as below:

Fraud Category	Category Description	Count
0	Non-fraudulent application	985,607
1	Fraudulent application	14,393

Data Preparation

The data preparation was carried out by following a systematic process of filling missing values with adherence to the business objective. Across different fields, various frivolous values were observed which were being used as proxies in the application. Such a claim can be made given the disproportionately high usage of such frivolous values as compared to other entities over the field. The disproportionately high usage of these specific values could be attributed to the possibilities of these values being the default entries while filling the application or purposefully entering frivolous entries to commit fraud. The business, however, with its industry knowledge could estimate better. To illustrate their usage, following are the distributions of the top 25 entities in the Address and Phone Number fields.





It is clear that the “123 MAIN STREET” entity in the address field and the “999-999-9999” entity in the phone number field have a very high occurrence compared to other entities in the field.

These values needed to be replaced in such a way that they prevent any loss of information, while ensuring that the values entered do not link with other records in the dataset. This is important because the variables built here onwards link records based over entities observed across a field. Various statistical summaries such as frequency of observing a certain value over a given field may get biased if these frivolous values are replaced with entries that match other entities in the field. A unique approach was taken by filling every element with the corresponding record number given that this value was unique to every single record in the dataset. This ensured no undue linkages with records of other entities and made sure that the model did not fire alarms over our replacements.

The kind of frivolous values replaced over every field, and their replacements are summarized as below:

Field	Frivolous Value Observed	Reasoning behind Frivolous Value	Replacement Strategy
Social Security Number	“999-99-9999”	Disproportionately High Usage	Replace with Record Number

Address	"123 MAIN STREET"	Disproportionately High Usage	Replace with Record Number
Date of Birth	"1907-06-26"	Disproportionately High Usage – Unlikely to have so many people with age > 100 years	Replace with Record Number
Phone Number	"999-999-9999"	Disproportionately High Usage	Replace with Record Number

Also, we created two new fields for the further variable creation:

1. nameDOB : the combination of the first name, last name and date of birth.
2. fulladdress : the combination of address and zip_5

Variable Creation

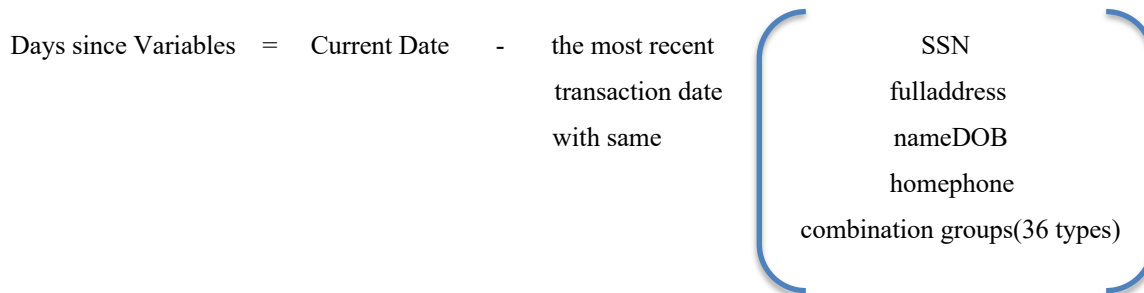
The variable creation process was based on typical behavioral patterns one would expect from a fraudster such as high frequency usage of compromised personal details. The candidate expert variables were created using the given original fields. All the 250 variables created can be broken down into the following four categories:

1. Days Since Variables (40)
2. Velocity Variables (197)
3. Risk Table Variables and Others (6+6)
4. One Random Variable for KS (1)

Following is a detailed explanation of the variable creation process:

Days Since Variables:

The 40 variables in this category were created in the following fashion:



Combination Groups for Day Since Variables: every two variable combination from the following 9 variables, including ssn, first name, last name, address, zip_5, DOB, homephone, nameDOB, and full address, which are $9 \times 8 / 2 = 36$ combinations, such as (ssn,fulladdress), (ssn,namedob), (ssn,phone), (fulladdress,namedob), (fulladdress,phone), (namedob,phone), (firstname,ssn), (lastname,ssn)...

In the Days Since Variable section, we get the current date of the transaction and subtract the date of the most recent transaction with the same aggregate variable or the same combination group and here we will use each of the fields or combination groups to create 40 more variables for each transaction. In the days since section we had some transactions without the most recent transaction

date, which means that there were no other transactions in the previous one year. For this problem we assumed that the date that we last saw that entity or combination group was 365 days ago, so that we can keep the valuable information within the data. This gives us a total of 40 variables.

Velocity Variables:

The 197 variables for this section are created as follows:



For the velocity variables, they depict how many times have we seen that entity or combination group over the past 0,1,3,7,14 days and this would show how frequently an applicant submit credit card applications.

Combination Groups for velocity variables: every two variable combination from the following 9 variables, including ssn, first name, last name, address, zip_5, DOB, homephone, nameDOB, and full address, which are $9*8/2=36$ combinations, such as (ssn,fulladdress), (ssn,namedob), (ssn,phone), (fulladdress,namedob), (fulladdress,phone), (namedob,phone), (firstname,ssn), (lastname,ssn)... Therefore, 36 combinations for 5 time periods are 180 variables intotal.

For the variables created by entities, we were supposed to be able to generate $4*5=20$ variables. However, because of the computational limit, the count of homephone numbers that appeared over the past 3,7 and 14 days were not created. Eventually, we only got 17 velocity variables for entities.

Risk Table :

Since we have the label variables, we want to fully utilize the information brought by the categorical variables, so we used the risk table to create new variables-that is to say-for each possible category for categorical variables we assigned a value to that group. Compared with

dummy encoding, risk table does not expand dimensionality and each categorical field becomes one numeric variable.

Category for categorical variables we used are the following:

1. The first 3 digits of zip code
2. The first 3 digits of the homephone number
3. The year when applicants submitted the applications
4. The month when applicants submitted the applications
5. The day when applicants submitted the applications
6. The day of week when applicants submitted the applications

We created the risk table variables based on the training and testing dataset, but not OOT. We firstly grouped the records by the categories listed above and calculated the number of goods and the number of Bad's for each group and all records. The number of Bad's for each group is the overall fraud rate. Then, we used the smoothing formula to calculate the assigned values and assign the same value to the same group from the same category.

Smoothing Formula: $\langle y \rangle = \frac{n_b + 20r}{(n_b + n_g) + 20}$

Overall fraud rate
 $r = \frac{n_b}{n_g + n_b} \Bigg|_{\text{All records}}$

For the risk table value in the OOT dataset, we used the value calculated from the training and testing datasets to fill in the records which is in the corresponding category. As for the records whose categories have never shown up in the training and testing dataset (only 1 record in our dataset), we just used its own label and the smoothing formula to calculate the value.

Others (len_ssn, mean_fraud, num_phones, num_zip, num_fraud):

len_ssn: We believed that the records whose ssn length is abnormal have higher possibility to be fraud, because they probably just filled in their ssn randomly.

mean_fraud: the average number of applications recognized as fraud for every ssn.

num_fraud: the number of frauds committed by every ssn.

Also, we believe that the more phone numbers and addressed the applicants used for the applications, the more possible that record could be fraud.

num_phones: the number of phone numbers owned by every ssn.

num_zip: the number of zipcodes for every ssn.

Random Variable:

In addition to these 249 variables, we created another variable called “Random Variable” which assigns a random value between 0 and 1 to every single record. The motivation behind this variable is to see if the frauds can be strongly predicted simply by a random guess.

This explains how all of the 250 expert variables were created.

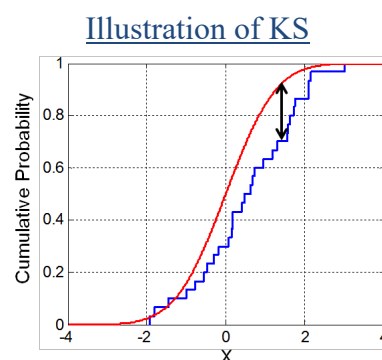
Feature Selection

A total of 250 expert variables were created through the feature creation process. However, including all 250 variables into our model is not recommended as in higher dimensions, the results would be nonintuitive and it is also very likely that all the data points become outliers in one or more dimensions. This phenomenon, in common data-science terms, is called the curse of dimensionality. An increase for every dimension would demand an exponential times more number of data points as otherwise we may end up seeing fitting the model to noise rather than observing the true non-linearities. To get around this problem, we took two crucial steps that help select the key variables of interest and filter out variables that did not throw much light on which transactions were fraudulent and which weren't. Below is a detailed explanation of the feature selection process:

1. Filtering out uninformative variables based on statistical filters:

For the first step of feature selection, we filtered out unwanted variables using the Kolmogorov-Smirnov (KS) filter and the Fraud Detection Rate (FDR) at 3% filter. A description of the two methods is as below:

- a. Kolmogorov Smirnov (KS) filter: The KS filter measures the maximum separation between two given distributions. The distributions of interest here are the distribution of all fraudulent transactions and the non-fraudulent transactions over its values on a given variable. In this setting, KS acts as a robust measure of how separated these fraudulent and non-fraudulent transactions are.



The red and blue lines are two cumulative probability distributions and the black vertical line is the KS distance (maximum separation) in between the two curves.

- b. Fraud Detection Rate at 3% filter: To calculate the Fraud Detection Rate at 3%, we arrange all our records in a decreasing or increasing order by the magnitude of their values for a

given variable. We then calculate the percentage of fraudulent records in the top 3% records to check if the variable is significantly able to identify records of one kind. Suppose for a particular variable, there are 300 records in the top 3% of which 200 are fraudulent and 100 aren't. The FDR for this variable would therefore be $200/300 = 66\%$.

Now after calculating these two metrics for every single variable, we rank order all the variables based on both, their KS and FDR values and assign each variable two ranks respectively in an increasing order of magnitude. Each variable now has one KS Rank and one FDR Rank. We take an average of these two ranks as a final score metric and then again rank order all the variables by their final scores in a decreasing fashion.

Below is the output of top 10 variables:

	field	ks	FDR	rank_ks	rank_FDR	average_rank
0	fraud_label	1.000000	1.000000	259.0	257.5	258.25
1	freq_zip_1	0.999871	1.000000	258.0	257.5	257.75
2	mean_fraud	0.997445	1.000000	257.0	257.5	257.25
3	num_fraud	0.997441	1.000000	256.0	257.5	256.75
4	Days_since_per_zip5_strfulladdress	0.329110	0.354377	254.5	254.5	254.50
5	Days_since_per_addressfulladdress	0.329110	0.354377	254.5	254.5	254.50
6	Days_since_per_fulladdress	0.328687	0.353710	253.0	253.0	253.00
7	Days_since_per_addresszip5_str	0.328349	0.353377	252.0	252.0	252.00
8	dataaddressfulladdress14agg	0.317601	0.340468	249.5	249.5	249.50
9	datazip5_strfulladdress14agg	0.317601	0.340468	249.5	249.5	249.50

After this process, we filter in only the top 128 variables where the magnitude of the final score is the highest and filter out the rest as these variables are not providing enough information in terms of how well the records are separated over a given expert variable.

128 variables are a safe bet in terms of the number. A smaller number cannot be selected as this filtering process doesn't take into consideration collinearity between variables.

2. Backward Selection of Variables:

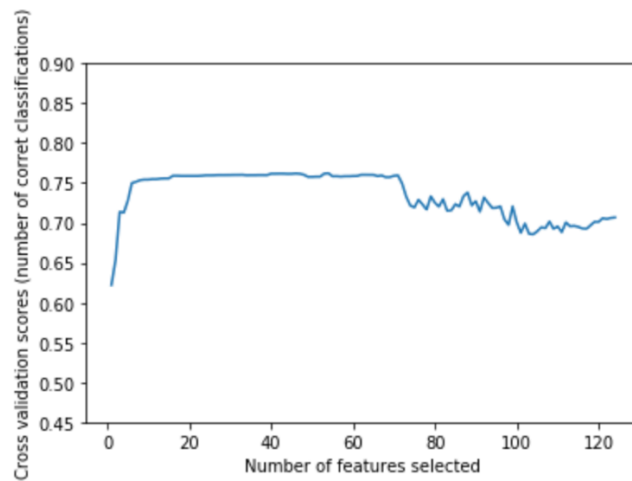
To eliminate dependence and collinearity among variables, a Recursive Feature Elimination (RFE) with Cross Validation process was applied to the remaining 128 variables. RFE is a feature selection method that fits a model to a number of features and then eliminates the weakest feature(s) until a specified number of features is reached. The process eliminates dependencies and collinearities between variables by keeping only one of the two variables that might be highly correlated. A description of the process is as follows:

- a. The estimator is trained on the initial set of features and the importance of each feature is obtained either through a coefficient attribute or a feature importance attribute like p-value.
- b. The least important features are pruned based on their importance.
- c. The process is repeated recursively on the pruned set until the desired number of features is reached.

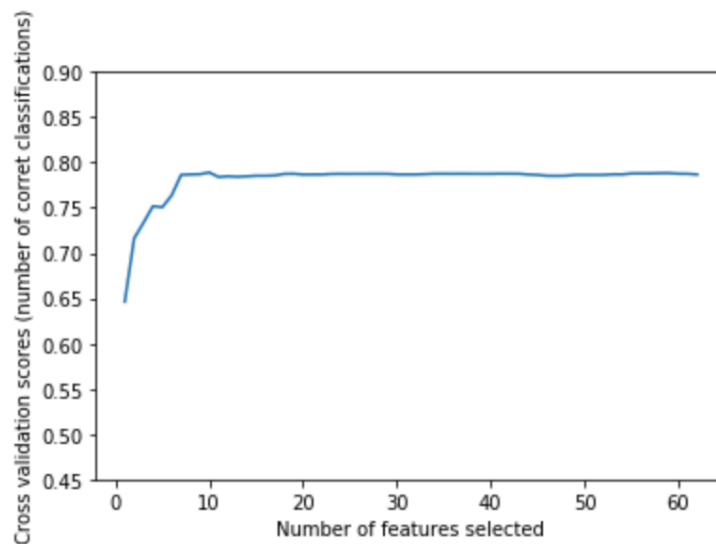
RFE with cross-validation has a function of automatically tuning the number of features to be selected with cross validation to help determine what the ideal number of features would be. Here, we applied RFE with 3-fold cross validation using a logistic regression function as typically for this process, it is recommended to use as simple a model as possible.

We applied two iterations of cross validation for the following reasons:

1. On the first iteration, we found out that the optimal number of variables is 54 out of 128. However, this still contributes to a high dimensionality, therefore we made the cut-off at half (64 variables) and selected the best 21 variables that are strong indicators for fraud. A graph, indicating the CV scores based on the number of features, is shown below:



2. After identifying the best 64 variables, the process was repeated to identify 10 as the optimal number of variables that are the strongest indicators for identifying fraud. To avoid losing information and to be safe, we decided to take top 21 variables as predictors. The relevant graph is as below:



The final 21 variables are:

1. Days_since_per_lastnamedob_str

Number of days since the combination of the last name and that date of birth last appeared

2. Days_since_per_lastnamenameDOB

Number of days since the combination of the full name and date of birth last appeared

3. Days_since_per_nameDOB:

Number of days since that full name and date of birth last appeared

4. datafirstnamelastname14agg:

Number of times the combination of a first name and last name appears over that past 14 days

5. datassnfirstname14agg:

Number of times the combination of an SSN and first name appears over that past 14 days

6. datassnlastname14agg:

Number of times the combination of an SSN and last name appears over that past 14 days

7. datassnnameDOB14agg:

Number of times the combination of an SSN, full name and date of birth appears over that past 14 days

8. datassnzip5_str14agg:

Number of times the combination of that SSN and full name and that zip code appears over that past 14 days

9. datassndob_str14agg:

Number of times the combination of an SSN and date of birth appears over the past 14 days

10. Days_since_per_dob_strnameDOB:

Number of days since the combination of that date of birth and full name last appeared

11. datazip5_strfulladdress7agg:

Number of times the combination of a zip code and full address appears over the past 7 days.

12. dataaddresszip5_str14agg:

Number of times the combination of an address and zip code appears over the past 14 days

13. datafulladdress14agg:

Number of times the full address appears over the past 14 days

14. datafulladdress7agg:

Number of times the full address appears over the past 7 days

15. datazip5_strfulladdress14agg:

Number of times the combination of a zip code and full address appears over the past 14 days

16. Days_since_per_addresszip5_str:

Number of days since the combination of an address and zip code last appeared

17. datanameDOB14agg:

Number of times the combination of a name and date of birth appears over the last 14 days

18. dataaddresszip5_str3agg:

Number of times the combination of an address and zip code appears over the past 3 days

19. datahomephonefulladdress14agg:

Number of times the combination of a home phone and full address appears over the past 14 days

20. Days_since_per_addressfulladdress:

Number of days since a full address last appeared

21. datafirstnamennameDOB14agg:

Number of times a combination of a first name, full name and date of birth appeared over the past 14 days

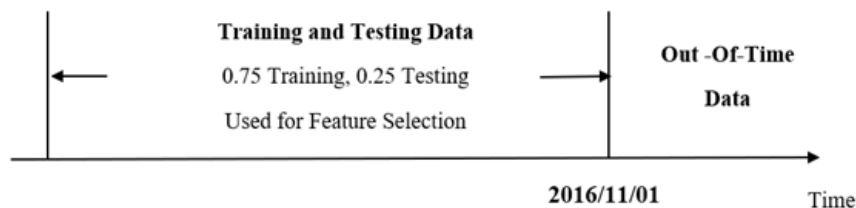
Algorithm - Model Building

Before building the models, to verify how they perform on the data, it was imperative to divide the data into training and testing for validation. This is done by the following process:

First, an Out-of-Time partition was made in the dataset for all the records after November 1st, 2016. This out-of-time dataset is used to predict how the model will perform on data it has never seen before, i.e., to evaluate how the model will perform on future data as it will come in with time to help prior identify which transactions are fraud.

Within the remaining data, i.e., data before 2016/11/01, a training/ testing split of 0.75 was carried out, meaning 75% percent of the data is randomly chosen as training data, the remaining 25% becomes the testing data.

A pictorial representation of the process is as follows:



Using the 21 final variables identified, the team built 4 supervised machine learning models to predict the probability of a record being a fraud. A simple Logistic Regression model was used as the baseline, on which three non-linear models – Random Forest, Boosted Trees and Neural Network – were built to evaluate the improvement of model performance in terms of non-linearity.

The thought process and parameters behind using the above four models is as follows:

1. Logistic Regression:

Logistic regression is one of the apt regression models to use when the dependent variable is binary. In the identity fraud case, the dependent variable is whether the application is fraudulent, which makes logistics regression a reasonable model to use.

We applied the simple Logistic Regression model without penalty factor in R using the `glm()` function.

2. Random Forest:

Random Forest is a powerful off-self algorithm, which ensembles a set of decision tree models built with a subset of all variables. Within each decision tree, the node represents a predictor, and the leaf represents the number of samples in each class. When building trees in Random Forest, each time we consider a split in the tree, a random sample of predictors is chosen as split candidates from the full set of available predictors. Typically, the number of predictors considered at each split is approximately equal to the square root of the total number of predictors. Given the flexibility available with this model, it was considered strongly as a contender model. Important parameters for Random Forest are the number of trees, maximum depth of each tree, and minimum number of samples allowed in each leaf.

The Ranger package in R was used to implement the Random Forest model. The specified parameters in the model were:

- Importance: Importance measures variable importance mode. There are three modes: “none”, ‘impurity’ and ‘impurity_corrected’, ‘permutation’. We chose the ‘impurity’ measure, which is the Gini index for classification.
- Minimal Node Size: It represents the minimal node size for each decision tree. In other words, the minimum number of samples allowed in each leaf. The default setting in R is to use 1 for classification, 5 for regression, 3 for survival and 10 for probability. We chose the default setting of 10.
- Maximum Depth: It is the Maximum interaction depth of each tree. A value of NULL or 0 (the default) corresponds to unlimited depth, 1 to tree stumps (1 split per tree). The default setting of 0 was chosen to not limit the depth of the tree.
- Number of Trees: This measures the total number of trees in the forest. This parameter was set to 1000. For random forests, the number of trees is an important parameter but might not be a critical parameter as using a large value of trees might not lead to overfitting due to the randomization of variable selection.

3. Boosted Trees:

Boosted Tree is a common non-linear model which combines weak models to build a powerful predictor. The boosted tree algorithm is based on the general idea of computing a sequence of simple decision trees where each successive decision tree is built on the prediction residual of the previous tree.

A boosted tree is built by training a series of weak learners to finally build a strong learner. Each weak learner is basically a decision tree that is built to predict the residual error of the previous tree. These residual errors are used to allot weights relative to their accuracies.

The trees are then combined by an aggregating function (weighted average/sum) and the final aggregation will give the final splits as well as the average values for each of the node/leaf.

The key parameters in a boosted decision tree are:

- Loss Function: The loss function will be based on the sum of all the loss functions (Gini, Information Gain) for each tree. The objective is to minimize this loss function.
- Number of Trees: The number of trees will determine the number of iterations carried out.
- Max Depth: The depth will determine how many nodes between the leaf and the root node and will impact the accuracy of the model.

The gbm package in R was used to perform a Gradient Boosted Tree model. Gradient Boosting increases the weights on misclassified records so the next iteration can pay more attention to them. Important parameters include:

- Distribution: Distribution specifies the loss function. Currently available options are "gaussian" (squared error), "laplace" (absolute loss), "tdist" (t-distribution loss), "bernoulli" (logistic regression for 0-1 outcomes), "huberized" (huberized hinge loss for 0-1 outcomes). As gmb package indicates, for most classification problems either bernoulli or adaboost will be appropriate, the former being recommended. Therefore, "bernoulli" was chosen to implement the model.
- Number of Trees: It specifies the total number of trees to fit. This is equivalent to the number of iterations and the number of basic functions in the additive expansion. This parameter was set to 1000.
- Interaction Depth: This is an integer that indicates the maximum depth of each tree (i.e., the highest level of variable interactions allowed). A value of 1 implies an additive model, a value of 2 implies a model with up to 2-way interactions, etc. Default setting is 1. We chose the default setting to build trees that only have one node.

4. Neural Network:

Neural Networks are computing algorithms that are emulated from biological neural nets with a neuron as its fundamental unit. They are used for solving machine learning problems dealing with calculating outputs based on patterns in the bulk of the data. It does so by learning on the data inputted to a net. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

The typical architecture of a neural network consists of the following key elements:

- Input Layer: The input layer is the net's layer that receives the data. In other words, the input layer contains all x variables. The input data is usually all the vector elements of all the independent variables for every given field.
- Hidden Layers: These are the layers between the input and output layers. There may or may not be more than one layers based on the input given by the solver. At each of the layer, nodes receive weighted signals from the previous layer and transform them to an output through the assigned activation function.
- Output Layer: It is last layer that outputs the dependent variables based on the transformations at each layer.

Important parameters for a neural network are:

- Number of Hidden Layers: Every layer will assign weights and apply a transformation function based on its learnings from the previous layer and therefore the number of hidden layers is an important input parameter.
- Number of Nodes per Layer: This will decide how accurately we are able to produce the desired output. However, one must be careful of overfitting. Usually, this parameter is also usually optimized based on different iterations. The input layer will have as many nodes as features. The output layer will have as many nodes as the number of dependent variables.
- Activation function: Activation function of a node transforms the received signal using the function and passes on the output to the next layer. This function helps to introduce non-linearity as the model learns from the given set of inputs to produce outputs with minimum possible errors. Different types of activation functions are sigmoid, tanh, Relu and SoftMax.

A Neural Network with a single hidden layer using the nnet package in R. The important parameters include:

- Size: Size represents the number of nodes in the single hidden layer. It can be zero if there are skip-layer units. The number of nodes per layer was specified to be 10.
- Softmax: Softmax is suitable for log-linear models and maximum conditional likelihood fitting. Therefore, we specified softmax as the activation function.
- Number of layers: The nnet function in R builds only single layer Neural Nets.

Algorithm – Model Comparison

For each model, the performance is evaluated by comparing the Fraud Detection Rate (FDR) at 3%, i.e., the percentage actual fraudulent transactions identified in the top 3% of the population based on the decreasing score of each transaction being fraudulent according to the model.

The Fraud Detection Rate at 3% was calculated for all the models for training, testing and out-of-time data. Below is a summary of the model performance:

	FDR @3%		
	Training	Testing	Out of Time
Logistic Regression	48.99%	48.99%	46.61%
Random Forest	53.07%	52.96%	50.00%
Boosted Tree	51.48%	52.75%	49.08%
Neural Net	50.49%	51.52%	47.86%

From the above table, Random Forest consistently outperformed Logistic Regression, Boosted Tree and Neural Net. In the Training set, Random Forest outperformed Logistic Regression by 4.08%, Boosted Tree by 1.59% and Neural Net by 2.58%. In the testing set, Random Forest outperformed Logistic Regression by 3.97%, Boosted Tree by 0.21% and Neural Net by 1.44%. In the Out of Time (OTT) data, Random Forest has a FDR of 50%, which outperformed Logistic Regression (46.61%), Boosted Tree (49.08%) and Neural Net (47.86%).

Based on the above comparison, Random Forest is selected as the final model. Below charts show a detailed breakdown of the model performance in training, testing and Out-of-Time data. The steps to obtain the breakdown is:

- All the records were sorted in a decreasing order of their likelihoods of being fraudulent.
- The entire ordered population is then divided into 100 bins of equal size. We look at only the top 20 bins of interest. Some bins may contain 1 more record than the other, since the number of records in each bin must be an integer.
- Calculate the number of “Good’s” (non-fraudulent) and the number of “Bad’s” (fraudulent) in each of the 20 bins.
- Calculate the percentage of “Good’s” (non-fraudulent) and the percentage of “Bad’s” (fraud) in each of the 20 bins.
- After calculating this information for each bin, calculate the cumulative number and percentage of “Good’s” and “Bad’s” out of all the “Good’s” and “Bad’s” in the dataset from bins 1 to 20.

- f. KS is defined as the maximum separation of two distributions. Based on this definition, we calculate the KS by subtracting the percentage of “Good’s” from the “Bad’s”.
- g. We calculate the False Positive Rate (FPR) by dividing the Cumulative Good by Cumulative Bad.

Below is the summary table for training data:

Training	# Records		# Goods		# Bads		Fraud Rate					
	625131		616111		9020		0.0144					
	Bin Statistics						Cumulative Statistics					
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	FPR
1	6251	1949	4302	31.17901	68.8	6251	1949	4302	0.3	47.7	47.4	0.5
2	6251	5956	295	95.28	4.72	12502	7905	4597	1.3	51.0	49.7	1.7
3	6251	6061	190	96.96049	3.0395137	18753	13966	4787	2.3	53.1	50.8	2.9
4	6251	6148	103	98.35226	1.6477364	25004	20114	4890	3.3	54.2	50.9	4.1
5	6251	6161	90	98.56023	1.4397696	31255	26275	4980	4.3	55.2	50.9	5.3
6	6251	6182	69	98.89618	1.1038234	37506	32457	5049	5.3	56.0	50.7	6.4
7	6251	6201	50	99.20013	0.799872	43757	38658	5099	6.3	56.5	50.3	7.6
8	6251	6212	39	99.3761	0.6239002	50008	44870	5138	7.3	57.0	49.7	8.7
9	6251	6213	38	99.3921	0.6079027	56259	51083	5176	8.3	57.4	49.1	9.9
10	6251	6208	43	99.31211	0.6878899	62510	57291	5219	9.3	57.9	48.6	11.0
11	6251	6217	34	99.45609	0.543913	68761	63508	5253	10.3	58.2	47.9	12.1
12	6251	6205	46	99.26412	0.7358823	75012	69713	5299	11.3	58.7	47.4	13.2
13	6251	6197	54	99.13614	0.8638618	81263	75910	5353	12.3	59.3	47.0	14.2
14	6251	6201	50	99.20013	0.799872	87514	82111	5403	13.3	59.9	46.6	15.2
15	6251	6206	45	99.28012	0.7198848	93765	88317	5448	14.3	60.4	46.1	16.2
16	6251	6217	34	99.45609	0.543913	100016	94534	5482	15.3	60.8	45.4	17.2
17	6251	6209	42	99.32811	0.6718925	106267	100743	5524	16.4	61.2	44.9	18.2
18	6251	6211	40	99.3601	0.6398976	112518	106954	5564	17.4	61.7	44.3	19.2
19	6251	6217	34	99.45609	0.543913	118769	113171	5598	18.4	62.1	43.7	20.2
20	6251	6221	30	99.52008	0.4799232	125020	119392	5628	19.4	62.4	43.0	21.2

In the training set, the model captures all fraud record in the first two bins. The number of “Bad’s” in the first bin is 4302 and in the following 19 bins, over 700 “Bad’s” are captured, adding up to 5628. This indicates that the model has strong predictive power.

Below are the results for testing data:

Testing	# Records		# Goods		# Bads		Fraud Rate					
	208376		205389		2987		0.0143					
Population Bin %	Bin Statistics					Cumulative Statistics						
	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Good	Cumulative Bad	% Good	% Bad (FDR)	KS	FPR
1	2084	634	1450	30.42	69.58	2084	634	1450	0.3	48.5	48.2	0.4
2	2084	2011	73	96.50	3.50	4168	2645	1523	1.3	51.0	49.7	1.7
3	2084	2025	59	97.17	2.83	6252	4670	1582	2.3	53.0	50.7	3.0
4	2084	2057	27	98.70	1.30	8336	6727	1609	3.3	53.9	50.6	4.2
5	2084	2068	16	99.23	0.77	10420	8795	1625	4.3	54.4	50.1	5.4
6	2084	2068	16	99.23	0.77	12504	10863	1641	5.3	54.9	49.6	6.6
7	2084	2075	9	99.57	0.43	14588	12938	1650	6.3	55.2	48.9	7.8
8	2084	2070	14	99.33	0.67	16672	15008	1664	7.3	55.7	48.4	9.0
9	2084	2071	13	99.38	0.62	18756	17079	1677	8.3	56.1	47.8	10.2
10	2084	2072	12	99.42	0.58	20840	19151	1689	9.3	56.5	47.2	11.3
11	2084	2070	14	99.33	0.67	22924	21221	1703	10.3	57.0	46.7	12.5
12	2084	2068	16	99.23	0.77	25008	23289	1719	11.3	57.5	46.2	13.5
13	2084	2069	15	99.28	0.72	27092	25358	1734	12.3	58.1	45.7	14.6
14	2084	2065	19	99.09	0.91	29176	27423	1753	13.4	58.7	45.3	15.6
15	2084	2069	15	99.28	0.72	31260	29492	1768	14.4	59.2	44.8	16.7
16	2084	2067	17	99.18	0.82	33344	31559	1785	15.4	59.8	44.4	17.7
17	2084	2073	11	99.47	0.53	35428	33632	1796	16.4	60.1	43.8	18.7
18	2084	2073	11	99.47	0.53	37512	35705	1807	17.4	60.5	43.1	19.8
19	2084	2079	5	99.76	0.24	39596	37784	1812	18.4	60.7	42.3	20.9
20	2084	2074	10	99.52	0.48	41680	39858	1822	19.4	61.0	41.6	21.9

The testing data does not perform as good as the training data. However, in the first bin, the model still captures 69.58% of the fraudulent records. The cumulative FDR is 48.5%, showing the power of this model.

Below are the results in Out-of-Time (OTT) data:

OOT	# Records		# Goods		# Bads		Fraud Rate					
	166493		164107		2386		0.0143					
	Bin Statistics					Cumulative Statistics						
Population	#					Total #	Cumulative	Cumulative		% Bad		
Bin %	Records	# Goods	# Bads	% Goods	% Bads	Records	Good	Bad	% Good	(FDR)	KS	FPR
1	1665	574	1091	34.47	65.53	1665	574	1091	0.3	45.7	45.4	0.5
2	1665	1608	57	96.58	3.42	3330	2182	1148	1.3	48.1	46.8	1.9
3	1665	1620	45	97.30	2.70	4995	3802	1193	2.3	50.0	47.7	3.2
4	1665	1634	31	98.14	1.86	6660	5436	1224	3.3	51.3	48.0	4.4
5	1665	1637	28	98.32	1.68	8325	7073	1252	4.3	52.5	48.2	5.6
6	1665	1644	21	98.74	1.26	9990	8717	1273	5.3	53.4	48.0	6.8
7	1665	1648	17	98.98	1.02	11655	10365	1290	6.3	54.1	47.7	8.0
8	1665	1655	10	99.40	0.60	13320	12020	1300	7.3	54.5	47.2	9.2
9	1665	1659	6	99.64	0.36	14985	13679	1306	8.3	54.7	46.4	10.5
10	1665	1660	5	99.70	0.30	16650	15339	1311	9.3	54.9	45.6	11.7
11	1665	1646	19	98.86	1.14	18315	16985	1330	10.3	55.7	45.4	12.8
12	1665	1649	16	99.04	0.96	19980	18634	1346	11.4	56.4	45.1	13.8
13	1665	1656	9	99.46	0.54	21645	20290	1355	12.4	56.8	44.4	15.0
14	1665	1653	12	99.28	0.72	23310	21943	1367	13.4	57.3	43.9	16.1
15	1665	1653	12	99.28	0.72	24975	23596	1379	14.4	57.8	43.4	17.1
16	1665	1655	10	99.40	0.60	26640	25251	1389	15.4	58.2	42.8	18.2
17	1665	1658	7	99.58	0.42	28305	26909	1396	16.4	58.5	42.1	19.3
18	1665	1645	20	98.80	1.20	29970	28554	1416	17.4	59.3	41.9	20.2
19	1665	1647	18	98.92	1.08	31635	30201	1434	18.4	60.1	41.7	21.1
20	1665	1653	12	99.28	0.72	33300	31854	1446	19.4	60.6	41.2	22.0

The model does not perform as well on the OOT data as it does on the training or testing data. In the first bin, the model only captures 65.53% of all fraud records in the top 1% bin. However, the cumulative FDR is 45.7%, which suggests the model still has a fairly good prediction power.

Results

To evaluate Random Forest's effectiveness towards the business' bottom-line, the team estimated the fraud savings, lost sales and overall savings achieved after implementing the Random Forests model. Based on the overall savings, the team has made a recommendation for a score cutoff that helps maximize overall savings. Certain assumptions were made for the same and are listed below:

1. \$6,000 loss for every fraud that isn't caught.
2. \$50 loss for every false positive fraud (An application that isn't fraud but is flagged fraudulent by our model).

Fraud Savings Calculation:

From our predictions in the OOT dataset using our final model, we divide the results into 100 bins in the decreasing order of their likelihood of being fraudulent. This way, the topmost bins have the highest likelihood of fraud. For each bin, we then calculated the number of actual frauds and false positives assuming the model flags all of the applications as fraudulent. Now, by comparing the predictions with the actual results, we can calculate the savings, lost sales and overall savings as below:

Fraud Savings = Total actual frauds * 6000

Lost Sales = False Positives * 50

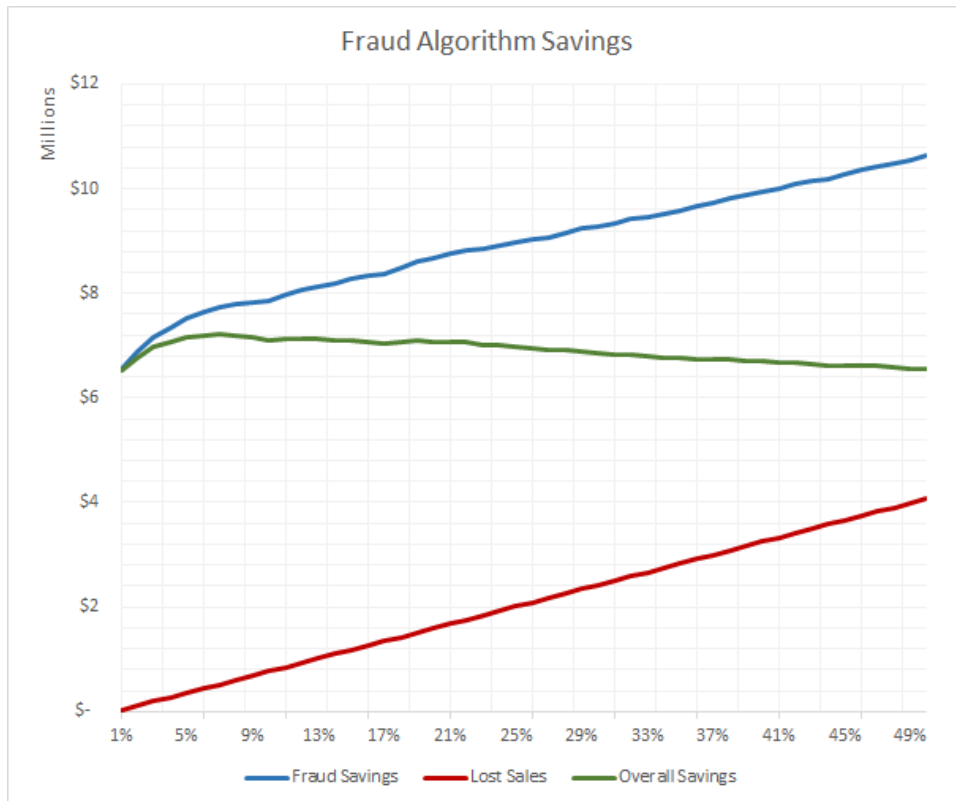
Overall Savings = Fraud Savings – Lost Sales

The below table shows the savings breakdown for the top 50%:

Population Bin %	# Records	# Goods	# Bads	Cumulative Good	Cumulative Bad	Fraud Savings	Lost Sales	Overall Savings
1%	1665	574	1091	574	1091	\$ 6,546,000	\$ 28,700	\$ 6,517,300
2%	1665	1608	57	2182	1148	\$ 6,888,000	\$ 109,100	\$ 6,778,900
3%	1665	1620	45	3802	1193	\$ 7,158,000	\$ 190,100	\$ 6,967,900
4%	1665	1634	31	5436	1224	\$ 7,344,000	\$ 271,800	\$ 7,072,200
5%	1665	1637	28	7073	1252	\$ 7,512,000	\$ 353,650	\$ 7,158,350
6%	1665	1644	21	8717	1273	\$ 7,638,000	\$ 435,850	\$ 7,202,150
7%	1665	1648	17	10365	1290	\$ 7,740,000	\$ 518,250	\$ 7,221,750
8%	1665	1655	10	12020	1300	\$ 7,800,000	\$ 601,000	\$ 7,199,000
9%	1665	1659	6	13679	1306	\$ 7,836,000	\$ 683,950	\$ 7,152,050
10%	1665	1660	5	15339	1311	\$ 7,866,000	\$ 766,950	\$ 7,099,050
11%	1665	1646	19	16985	1330	\$ 7,980,000	\$ 849,250	\$ 7,130,750
12%	1665	1649	16	18634	1346	\$ 8,076,000	\$ 931,700	\$ 7,144,300
13%	1665	1656	9	20290	1355	\$ 8,130,000	\$ 1,014,500	\$ 7,115,500
14%	1665	1653	12	21943	1367	\$ 8,202,000	\$ 1,097,150	\$ 7,104,850
15%	1665	1653	12	23596	1379	\$ 8,274,000	\$ 1,179,800	\$ 7,094,200
16%	1665	1655	10	25251	1389	\$ 8,334,000	\$ 1,262,550	\$ 7,071,450
17%	1665	1658	7	26909	1396	\$ 8,376,000	\$ 1,345,450	\$ 7,030,550
18%	1665	1645	20	28554	1416	\$ 8,496,000	\$ 1,427,700	\$ 7,068,300
19%	1665	1647	18	30201	1434	\$ 8,604,000	\$ 1,510,050	\$ 7,093,950
20%	1665	1653	12	31854	1446	\$ 8,676,000	\$ 1,592,700	\$ 7,083,300
21%	1665	1653	12	33507	1458	\$ 8,748,000	\$ 1,675,350	\$ 7,072,650
22%	1665	1654	11	35161	1469	\$ 8,814,000	\$ 1,758,050	\$ 7,055,950
23%	1665	1659	6	36820	1475	\$ 8,850,000	\$ 1,841,000	\$ 7,009,000
24%	1665	1653	12	38473	1487	\$ 8,922,000	\$ 1,923,650	\$ 6,998,350
25%	1665	1657	8	40130	1495	\$ 8,970,000	\$ 2,006,500	\$ 6,963,500
26%	1665	1653	12	41783	1507	\$ 9,042,000	\$ 2,089,150	\$ 6,952,850
27%	1665	1659	6	43442	1513	\$ 9,078,000	\$ 2,172,100	\$ 6,905,900
28%	1665	1651	14	45093	1527	\$ 9,162,000	\$ 2,254,650	\$ 6,907,350
29%	1665	1652	13	46745	1540	\$ 9,240,000	\$ 2,337,250	\$ 6,902,750
30%	1665	1658	7	48403	1547	\$ 9,282,000	\$ 2,420,150	\$ 6,861,850
31%	1665	1655	10	50058	1557	\$ 9,342,000	\$ 2,502,900	\$ 6,839,100
32%	1665	1653	12	51711	1569	\$ 9,414,000	\$ 2,585,550	\$ 6,828,450
33%	1665	1657	8	53368	1577	\$ 9,462,000	\$ 2,668,400	\$ 6,793,600
34%	1665	1655	10	55023	1587	\$ 9,522,000	\$ 2,751,150	\$ 6,770,850
35%	1665	1654	11	56677	1598	\$ 9,588,000	\$ 2,833,850	\$ 6,754,150
36%	1665	1653	12	58330	1610	\$ 9,660,000	\$ 2,916,500	\$ 6,743,500
37%	1665	1652	13	59982	1623	\$ 9,738,000	\$ 2,999,100	\$ 6,738,900
38%	1665	1651	14	61633	1637	\$ 9,822,000	\$ 3,081,650	\$ 6,740,350
39%	1665	1655	10	63288	1647	\$ 9,882,000	\$ 3,164,400	\$ 6,717,600
40%	1665	1653	12	64941	1659	\$ 9,954,000	\$ 3,247,050	\$ 6,706,950
41%	1665	1656	9	66597	1668	\$ 10,008,000	\$ 3,329,850	\$ 6,678,150
42%	1665	1652	13	68249	1681	\$ 10,086,000	\$ 3,412,450	\$ 6,673,550
43%	1665	1655	10	69904	1691	\$ 10,146,000	\$ 3,495,200	\$ 6,650,800
44%	1665	1658	7	71562	1698	\$ 10,188,000	\$ 3,578,100	\$ 6,609,900
45%	1665	1650	15	73212	1713	\$ 10,278,000	\$ 3,660,600	\$ 6,617,400
46%	1665	1652	13	74864	1726	\$ 10,356,000	\$ 3,743,200	\$ 6,612,800
47%	1665	1652	13	76516	1739	\$ 10,434,000	\$ 3,825,800	\$ 6,608,200
48%	1665	1656	9	78172	1748	\$ 10,488,000	\$ 3,908,600	\$ 6,579,400
49%	1665	1655	10	79827	1758	\$ 10,548,000	\$ 3,991,350	\$ 6,556,650
50%	1665	1650	15	81477	1773	\$ 10,638,000	\$ 4,073,850	\$ 6,564,150

Score Cutoff:

To get a better understanding of how the overall savings change as we look at an increasing proportion of the population, we have plotted the cumulative three values mentioned above as the population percentage changes. The blue line denotes Fraud Savings, the red line denotes Lost Sales and the green line denotes Overall Savings.

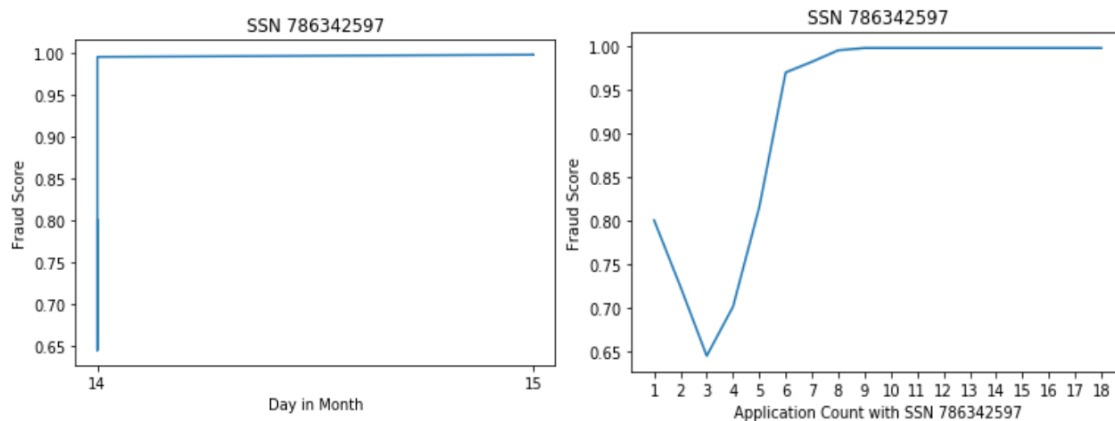


We can see that the overall savings peak at 7% at \$ 7,221,750 catching 1290 out of the 2386 fraudulent transactions. The overall savings keep decreasing after 7%.

To sum up, to maximize savings, we recommend a cutoff of 7% that also gives a good balance between true and false positives.

Fraud Score Charts for SSN and NameDOB

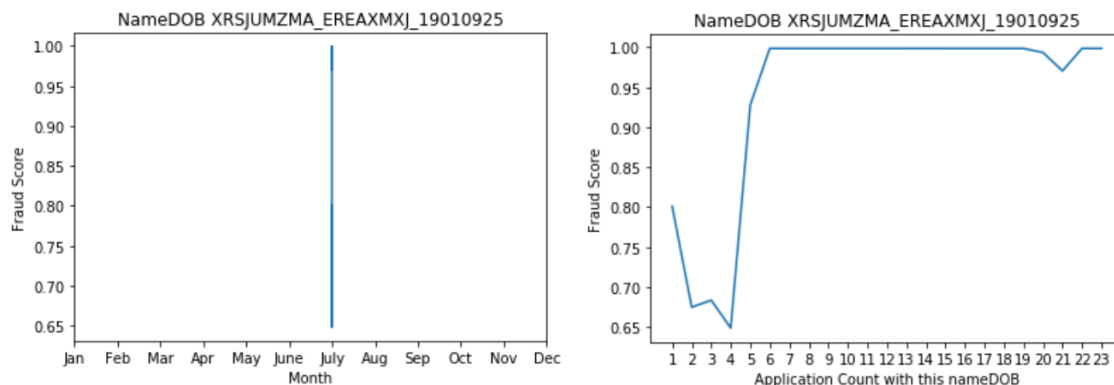
Choose the applications associated with SSN number of 786342597 and draw fraud score charts as follows:



The left plot shows 18 applications associated with SSN of 786342597 occur in 1 day.

The plot on the right indicates, as the number of applications increases, the score first drops down and rises quickly while 3 applications happen on the that day. It is possible that our model assumes less than 3 applications in a day is likely and is not a fraud signal, however, when there are more than 3 applications for same SSN on same day, then that applicant is possibly manipulating identity information and making a fraud.

Same with nameDOB :



The left plot shows 23 applications from XRSJUMZMA EREAXMXJ who born in 1905 which is very likely a faked identity. These applications are made in July and the fraud score predicted by our model reached to 100%.

The plot on the right indicates, as the number of applications increases, the score first drops down and rises quickly while 4 applications happened in that month. It is possible that our model assumes

less than 4 applications in a month is possible and is not a fraud signal, however, when there are more than 4 applications for same name and DOB in a month, then that applicant is possibly manipulating identity information and making a fraud.

Conclusion

Following a rigorous process of data exploration, cleaning and feature creation to create 250 variables, we reduced the dimensionality by feature selection using the Kolmogorov-Smirnov and Fraud Detection Rate filters and then used the Recursive Feature Elimination-Cross Validation method to eliminate collinearity. This process left us with the strongest 21 independent variables of interest. We then built four different models – logistic regression, neural network, boosted trees and random forest – to identify fraudulent transactions on a training subset and compared their fraud detection rates across training, testing and out of time data.

Based on these FDR's, we recommend the Random Forest model that gives an FDR of 50.00% on the out-of-time data. We recommend labelling the top 7% of the records as fraudulent for the future out-of-time data that gets scored by the model once it is implemented to maximize savings.

The current study was completed based on our understanding of the dataset and the credit card industry landscape in the United States. Further investigation should be conducted with more industry expertise to understand why each of the application is tagged fraudulent.

Appendix 1: Data Quality Report

Identity Fraud Identification

Data Quality Report

For: Dr. Stephen Coggeshall

1. Preface

1.1 Introduction

Following is a data quality report for Dr. Stephen Coggeshall, Professor at the University of Southern California, and owner of the dataset of “Applications Dataa” for Fraud Identification.

The intent of this report is to align with the owner of the dataset the primary exploratory analysis done by Mr. Shreerang Javadekar to proceed further with building a supervised learning model to help identify fraudulent transactions.

1.2 Brief on the Dataset

The dataset on hand consists of 1,000,000 applications made using 835,819 unique Social Security Numbers. The uniqueness of these 835,819 Social Security Numbers, however, is questionable.

We have a total of 10 unique fields in the dataset (including record number) that describe personal identification details of every single application – details including name, Social Security Number, Address, Zipcode and Phone Number. Details on the date when the application was made are also available. A label is also available that indicates whether the application is fraudulent or not.

Every field is 100% filled. There are no blank entries in the dataset. Further details have been mentioned in the Summary Table.

2. Summary Tables

Below is a summary table for all the fields in the dataset. All the fields have been treated as categorical.

Total number of records: 1,000,000

Total number of fields: 10.

Unique Identifiers for Every Recods: Record Number (Range: 1-1,000,000).

Field/Metric	# Records with a Value	% Populated	# Unique Categories	Most Common Value
Record	1,000,000	100%	1,000,000	Every value appears just once.
Date (date)	1,000,000	100%	365	“2016-08-16”
Social Security Number (ssn)	1,000,000	100%	835,819	“999-99-9999”
First Name (firstname)	1,000,000	100%	78,136	“EAMSTRMT”
Last Name (lastname)	1,000,000	100%	177,001	“ERJSAXA”
Address (address)	1,000,000	100%	828,774	“123 MAIN ST”
Zipcode (zip5)	1,000,000	100%	26,370	“68138”
Date of Birth (dob)	1,000,000	100%	42,673	“1907-06-26”
Phone Number (homephone)	1,000,000	100%	28,244	“999-999-9999”
Fraud Label (fraud label)	1,000,000	100%	2	“0”

3. Exploratory Analysis

Below is an exploratory analysis for every single field in the dataset. Adequate visualizations have been added to supplement the description.

3.1 Record (record)

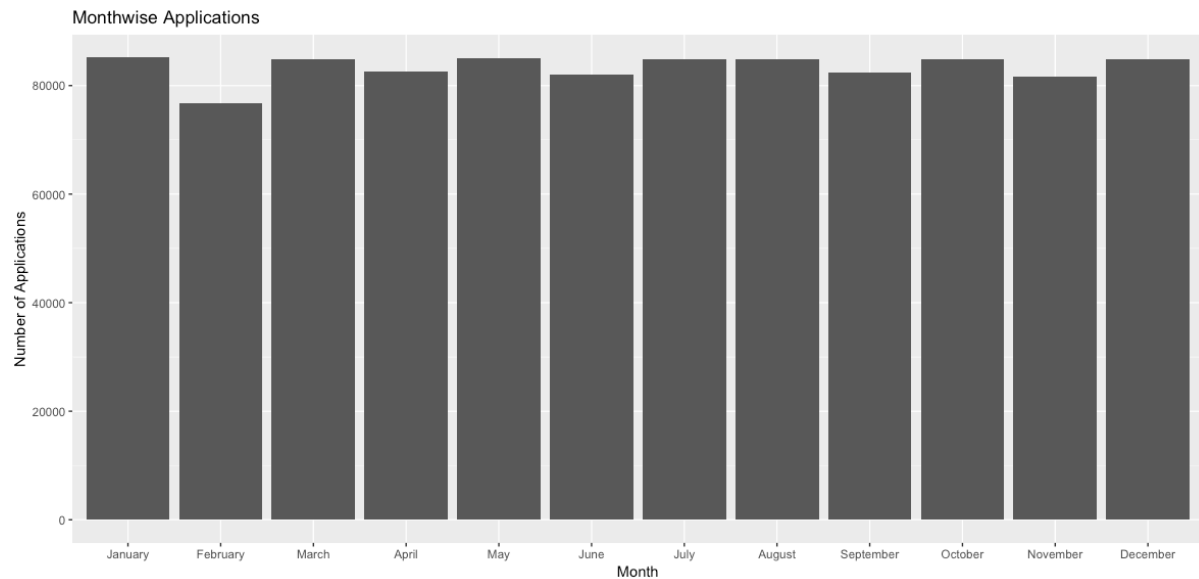
“record” is a unique identifier for every entry. The record indexing is numerical. It starts at 1 and ends at 1000000 with step increases of 1.

3.2 Date (date)

“date” is an indicator for the date when the application was filed.

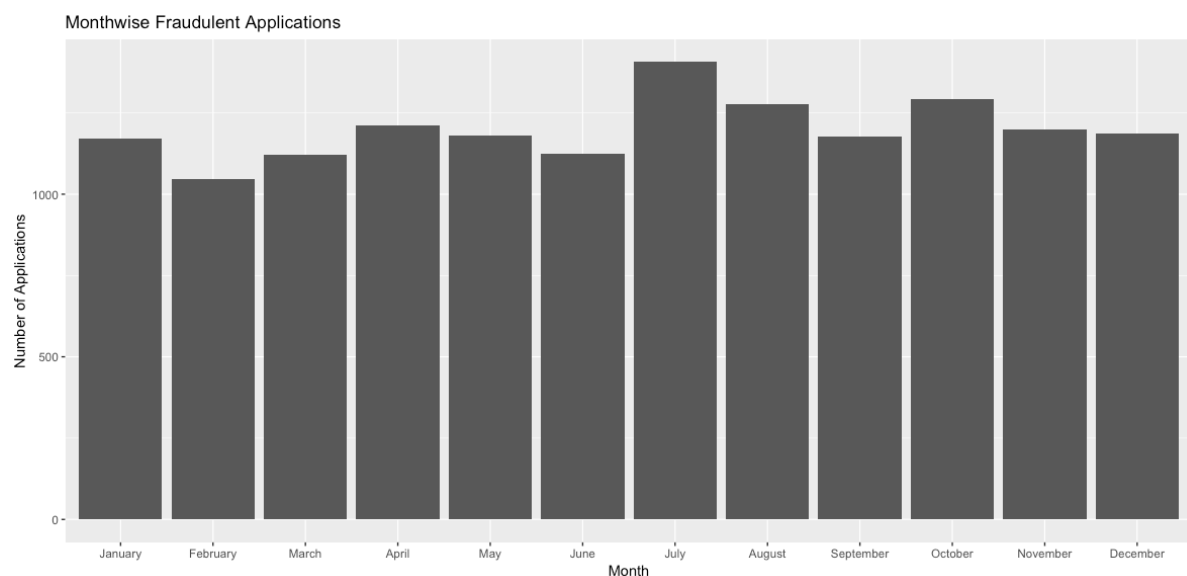
100% of this field is filled. The dates in this field are the 366 days of the leap year 2016. However, there are no applications for the date of 02/29/2016.

A monthwise distribution of the number of applications (both fraudulent and non-fraudulent) is as below:

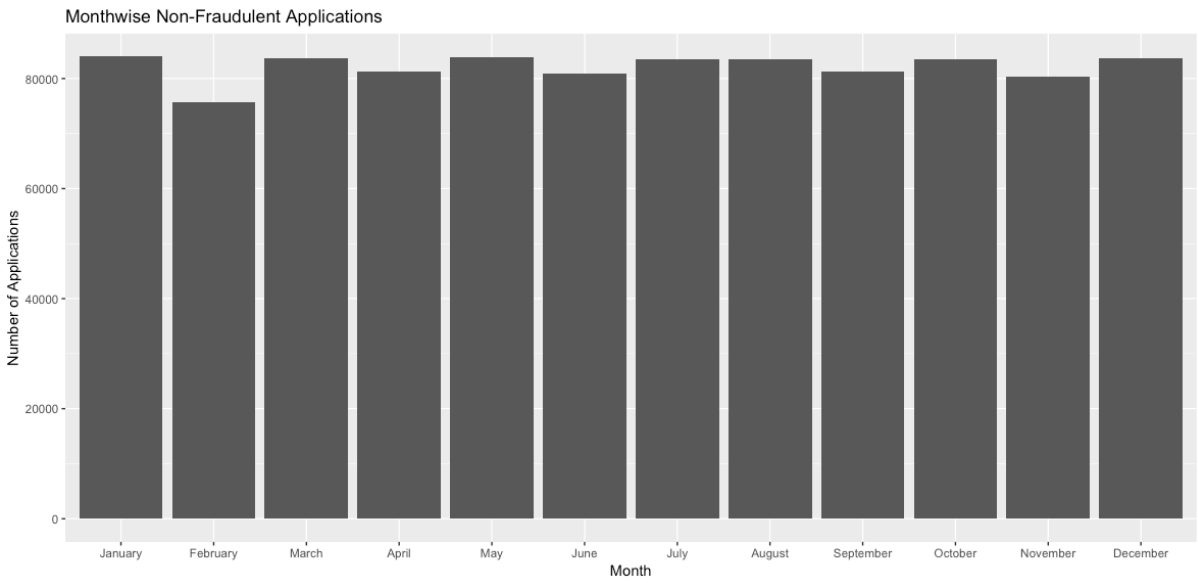


The distribution for number of applications seems to be uniform across all months with just a small dip in the month of February.

For fraudulent applications the distribution is as below:

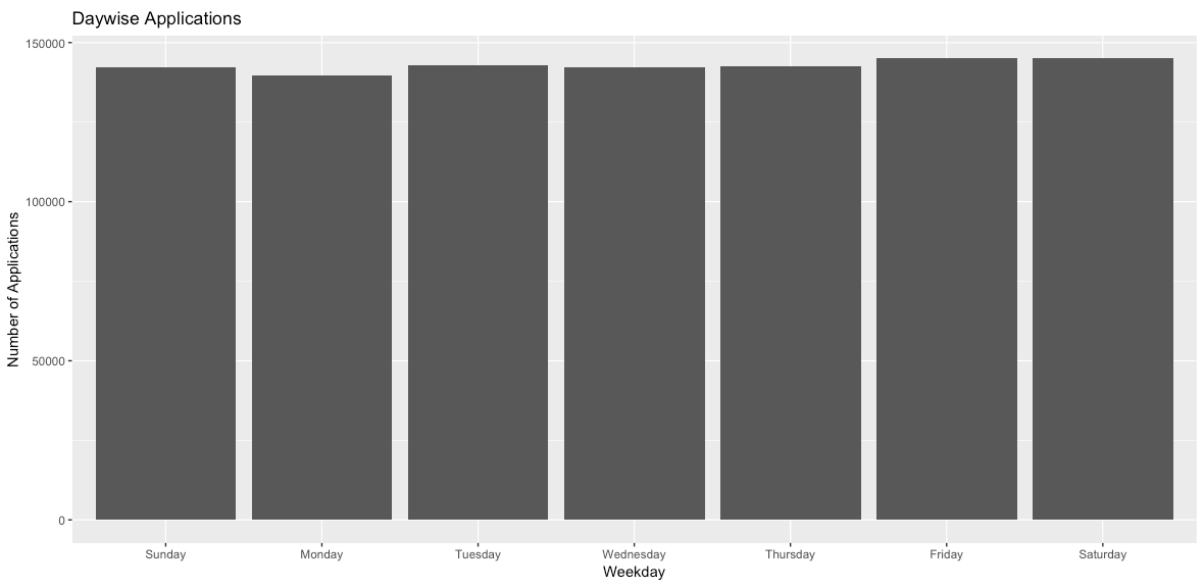


The month of July seems to have witnessed the highest number of fraudulent applications.

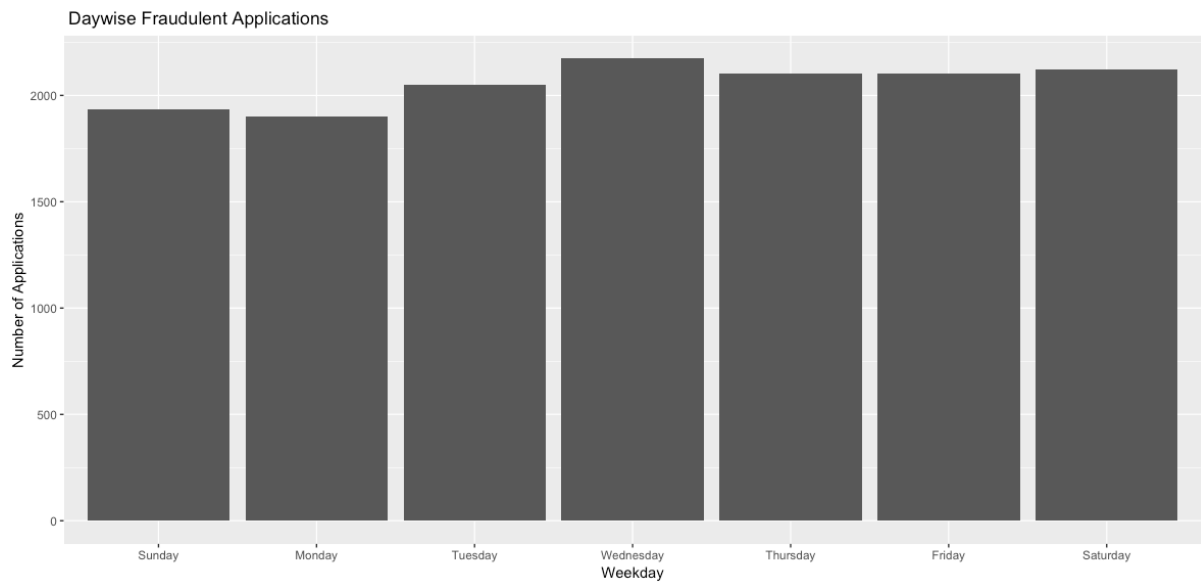


There is no significant trend in the non-fraudulent applications across months.

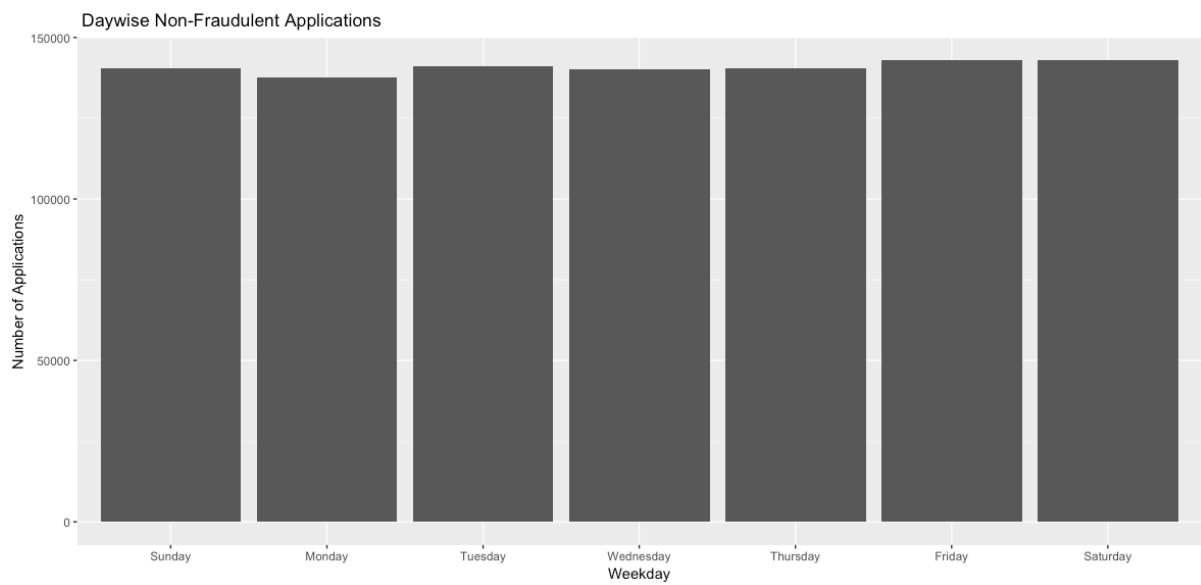
A daywise distribution of all the transactions is as follows:



For fraudulent applications, the daywise distribution is as below:

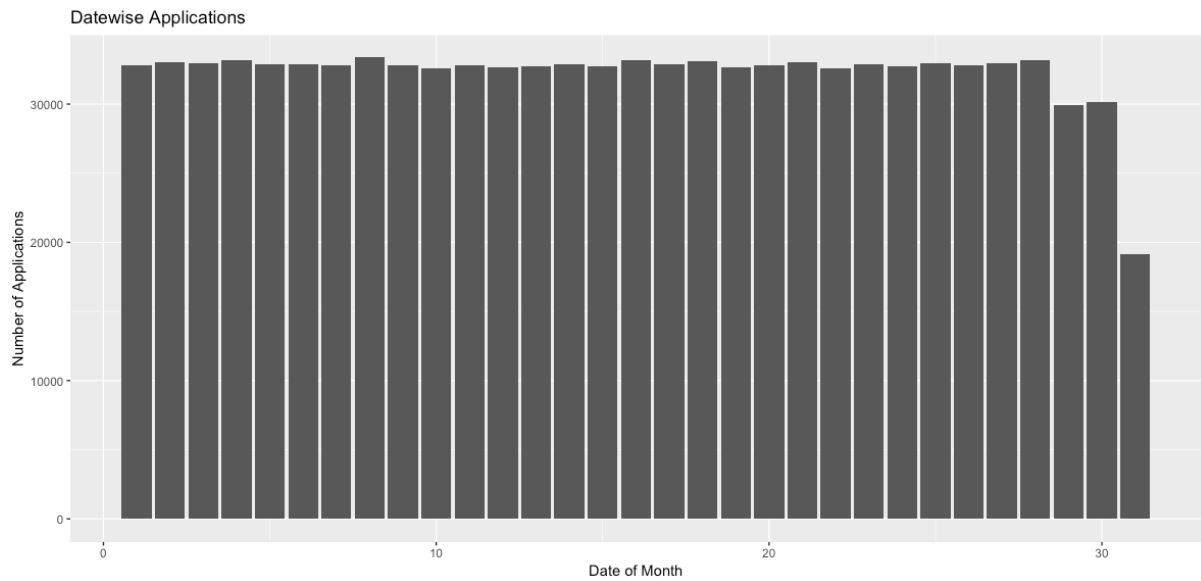


For non-fraudulent applications, the daywise distribution is as below:



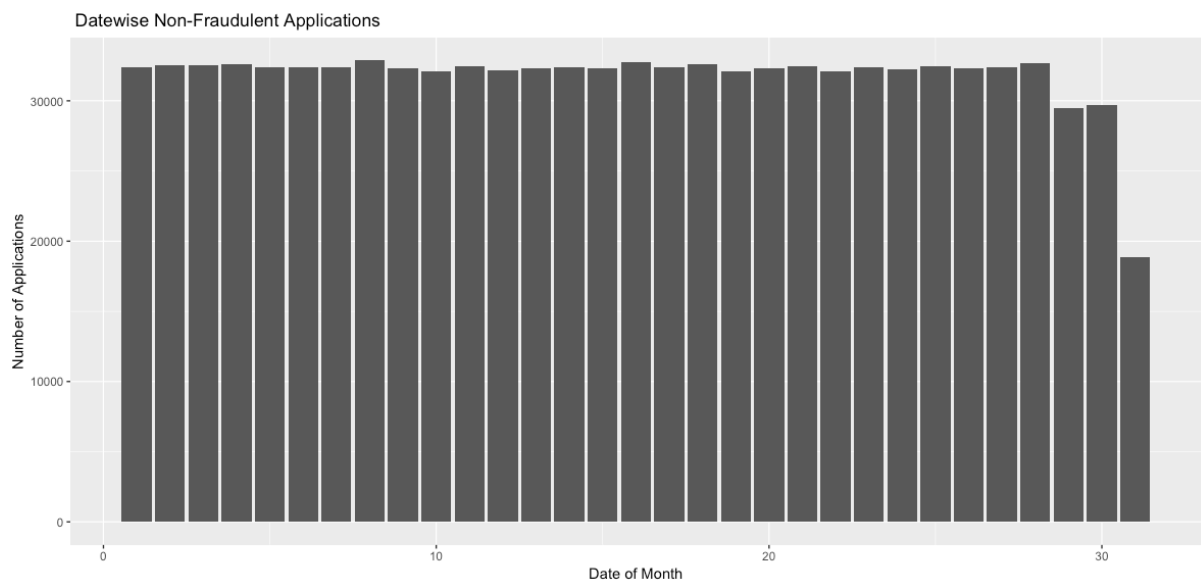
There is no significant trend in the daywise distribution across both fraudulent and non-fraudulent applications.

A datewise distribution of the number of applications (both fraudulent and non-fraudulent) is as below:

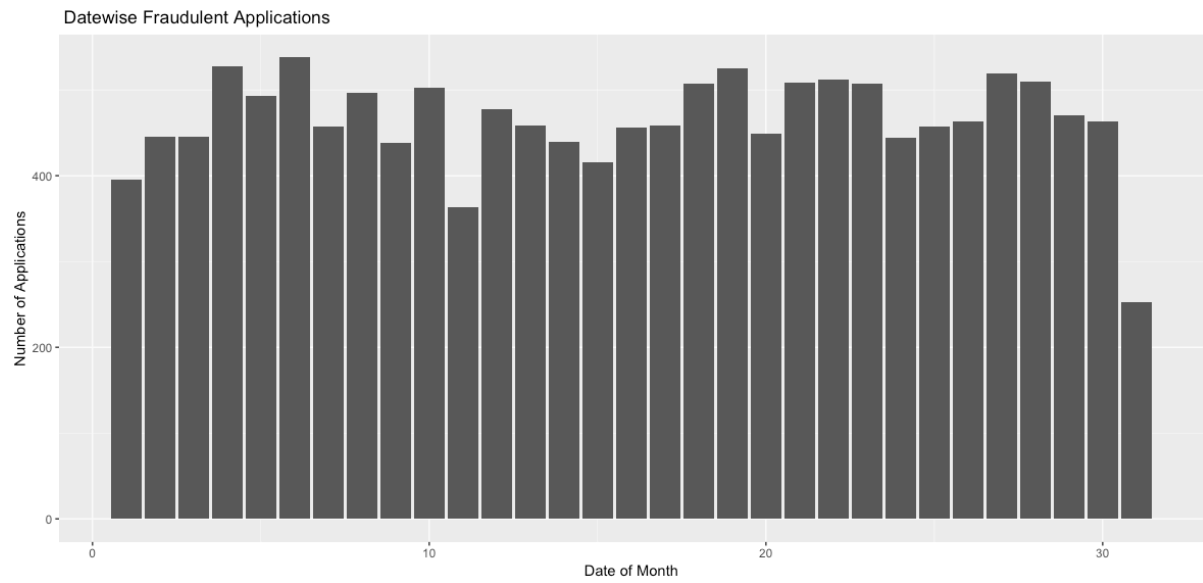


There seems to be a dip in the number of transactions nearing the month. For 31st, the number is very low because of the few months that have the 31st date.

For non-fraudulent applications, the distribution is as below:



For fraudulent applications, the distribution is as below:

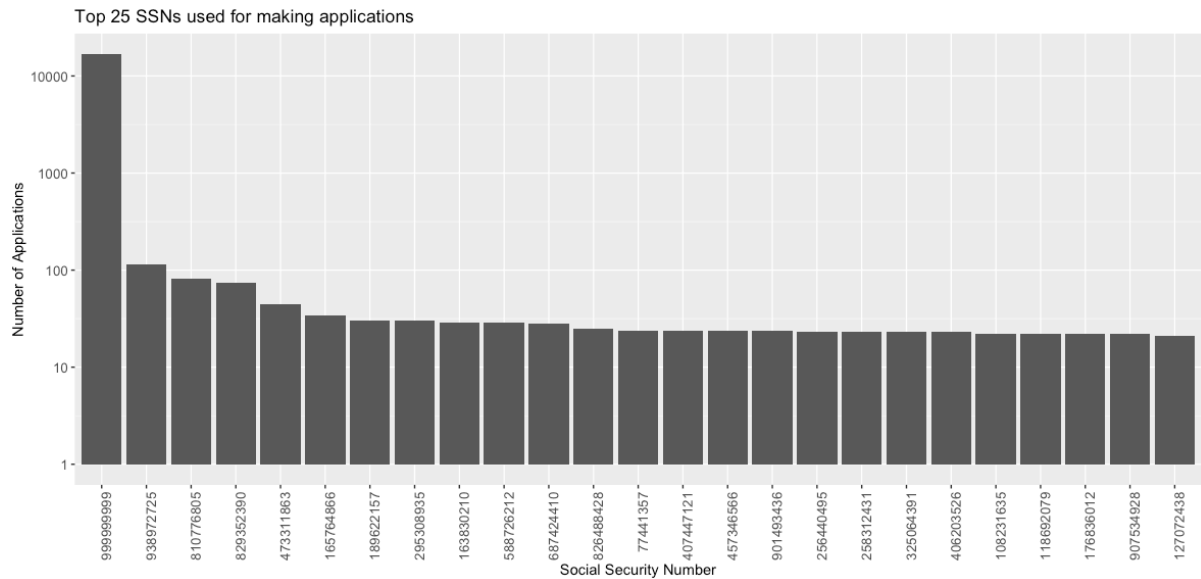


3.3 Social Security Number (ssn)

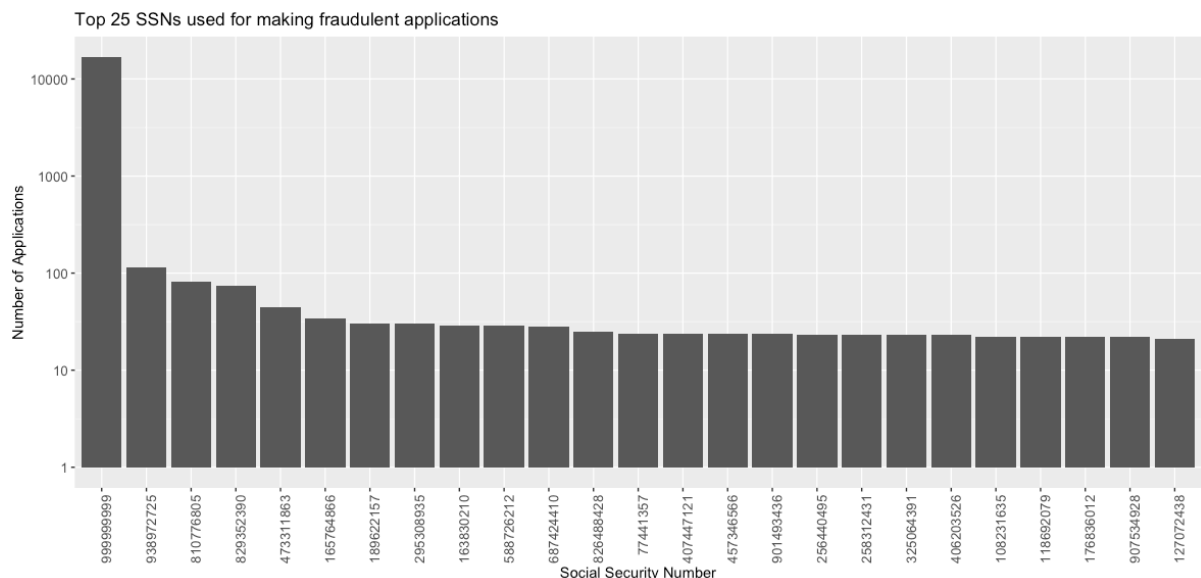
“ssn” is an indicator of the Social Security Number used for making the application.

There are 835,819 unique Social Security numbers in the dataset. Their uniqueness and authenticity, however, is questionable. 100% of the dataset is filled.

A distribution of the top 25 Social Security numbers used for making the applications is as follows. The Y-axis is in log scale.



The top 25 Social Security Numbers used for making fraudulent applications is as follows:



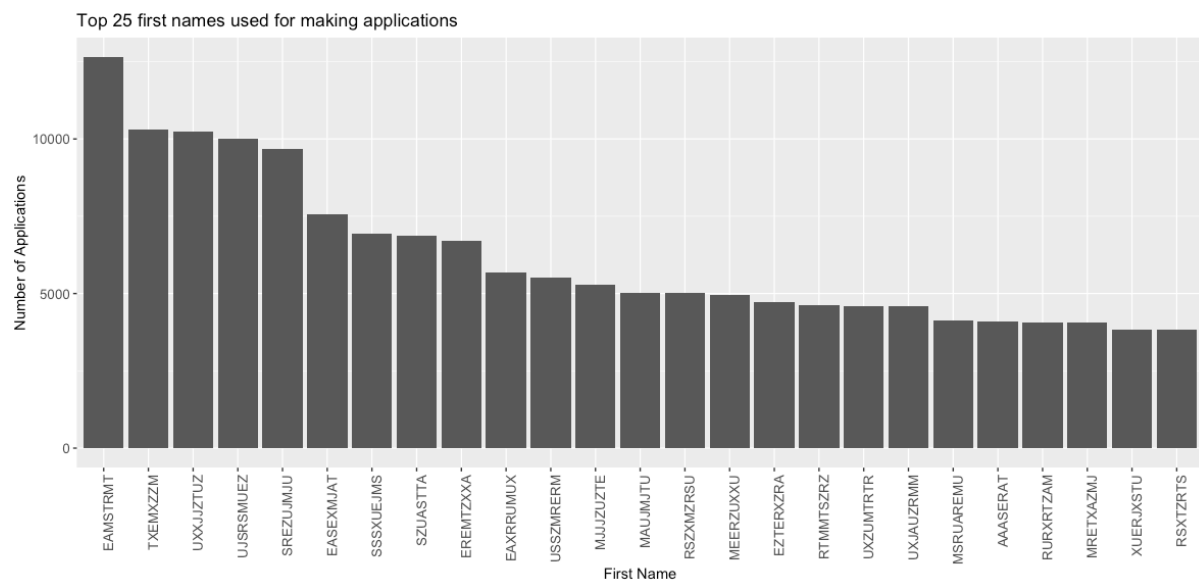
It is clear that the “999-99-9999” is a proxy number being used for making applications to mask one’s identity.

3.4 First Name (firstname)

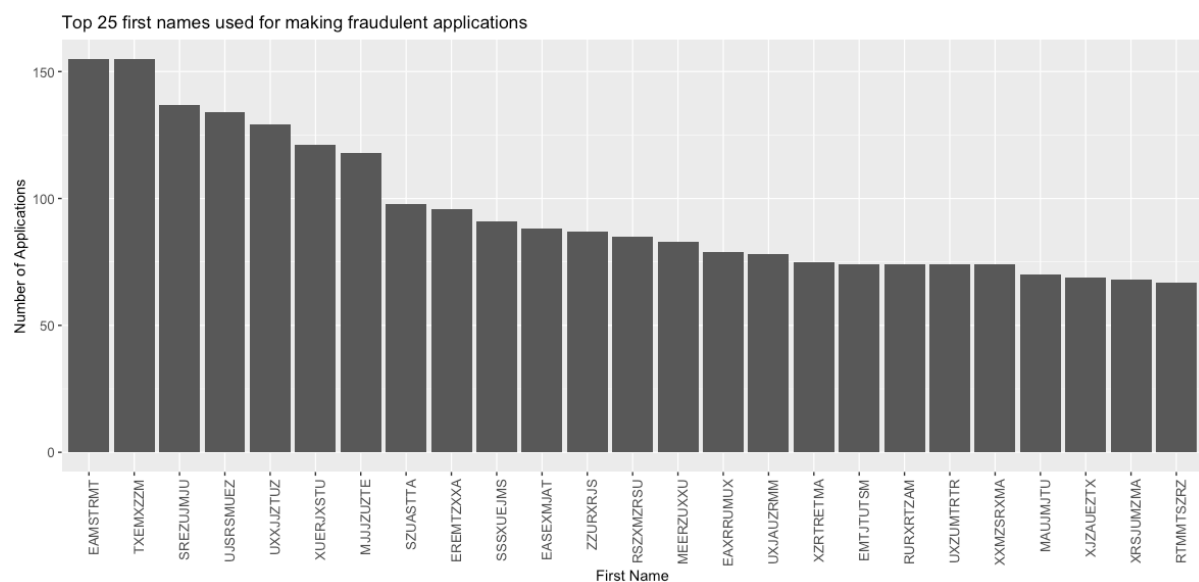
“firstname” is an indicator for the first name of the applicant.

There are 78,136 unique first names in the dataset. 100% of the fields are filled in this column.

A distribution of the top 25 first names with the highest number of applications is as follows:



A distribution of the top 25 first names with the highest number of fraudulent applications is as follows:

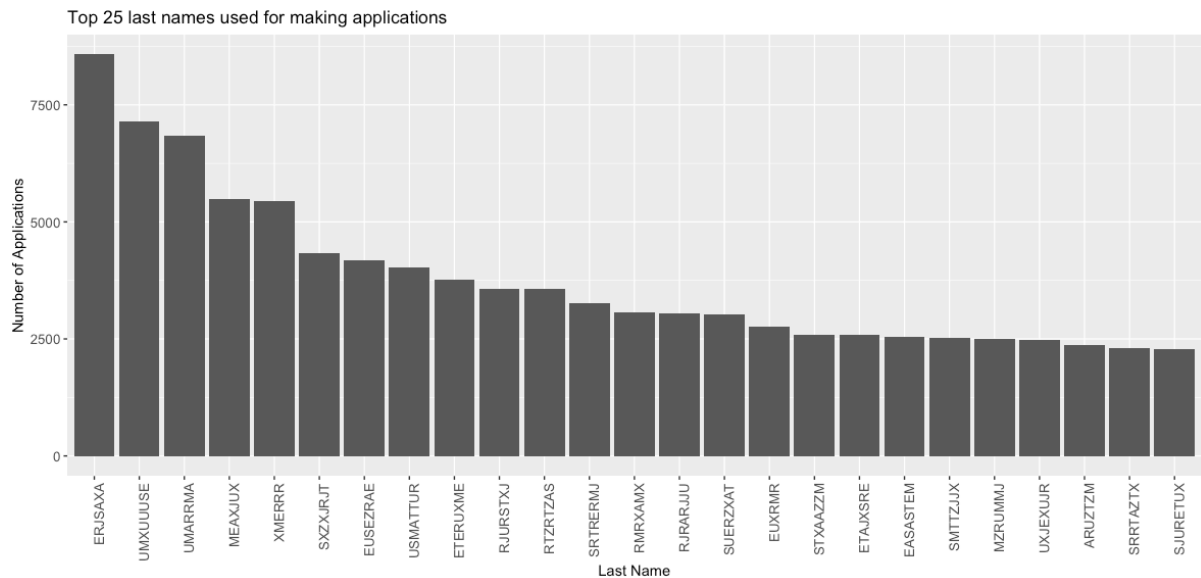


3.5 Last Name (lastname)

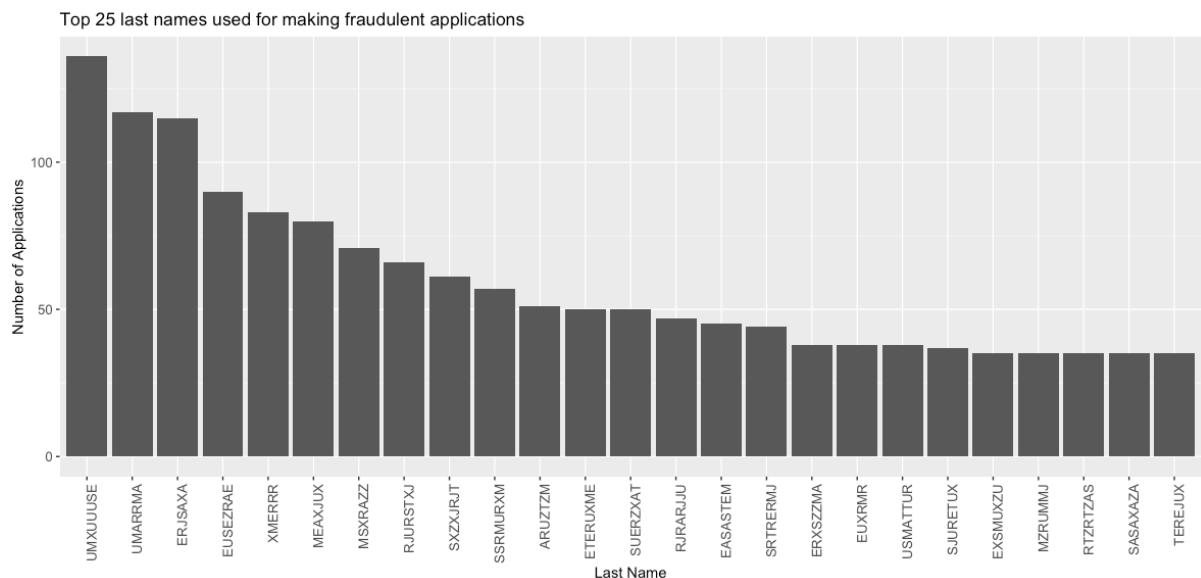
“lastname” is an indicator of the last name used while making the application.

There are 177,001 unique last names in the dataset. 100% of the dataset is filled

A distribution of the top 25 last names used for making the applications is as follows:



For fraudulent applications, the distribution is as below:

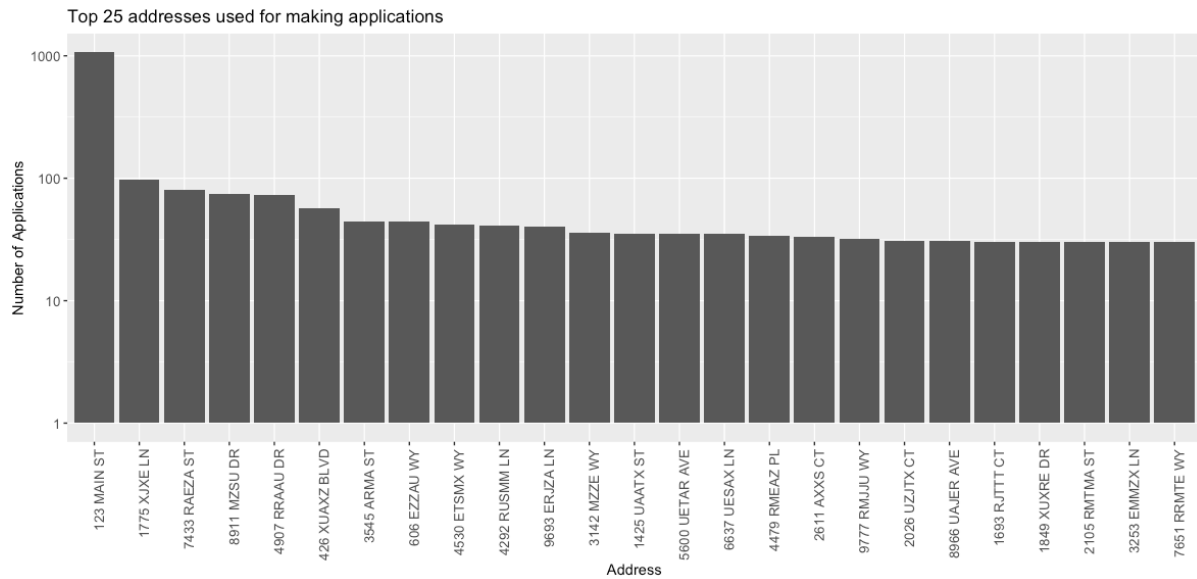


3.6 Address (address)

“address” is an indicator of the Address of the applicant filing the application.

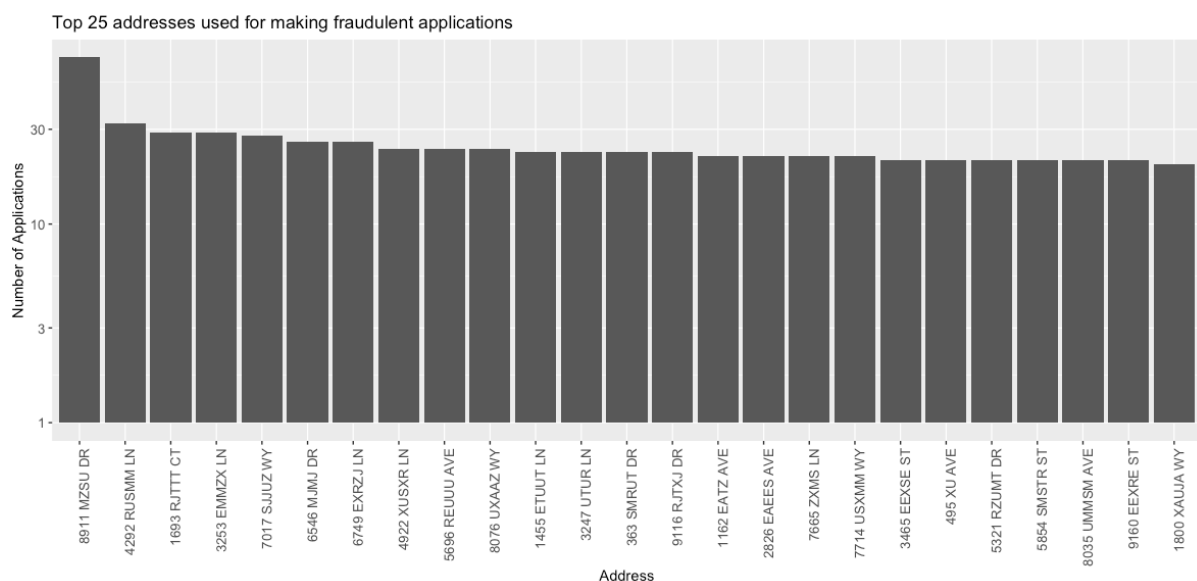
There are 828,774 unique addresses in the dataset. 100% of the fields are filled in this column.

A distribution of the top 25 addresses with the highest number of applications is as follows:



“123 MAIN ST” seems to be a proxy address being used while filing applications.

A distribution of the top 25 addresses with the highest number of fraudulent applications is as follows:



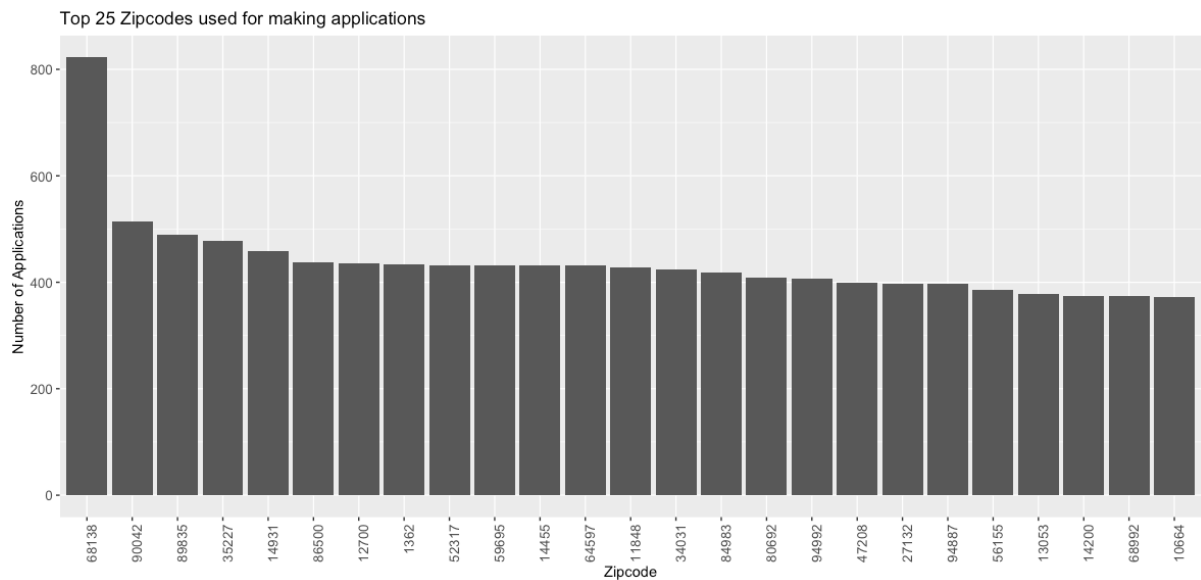
The proxy address surprisingly doesn't appear in the fraudulent applications.

3.7 Zipcode (zip5):

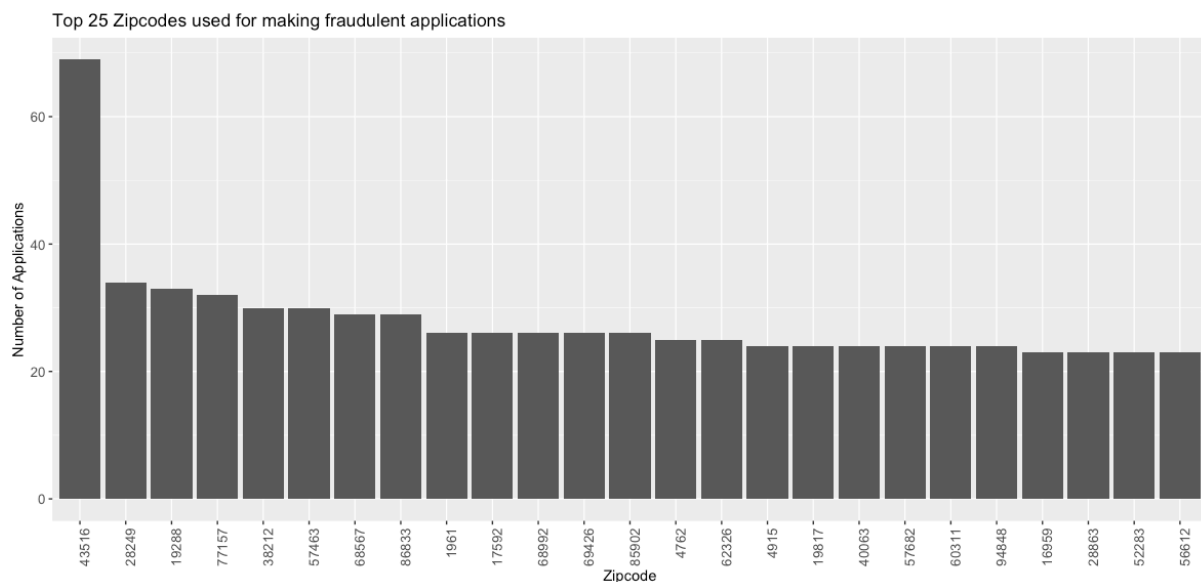
“zip5” is an indicator of the Zipcode of the applicant filing the application.

There are 26,370 unique zipcodes in the dataset. 100% of the fields are filled in this column.

A distribution of the top 25 zipcodes with the highest number of applications is as follows:



A distribution of the top 25 zipcodes with the highest number of fraudulent applications is as follows:

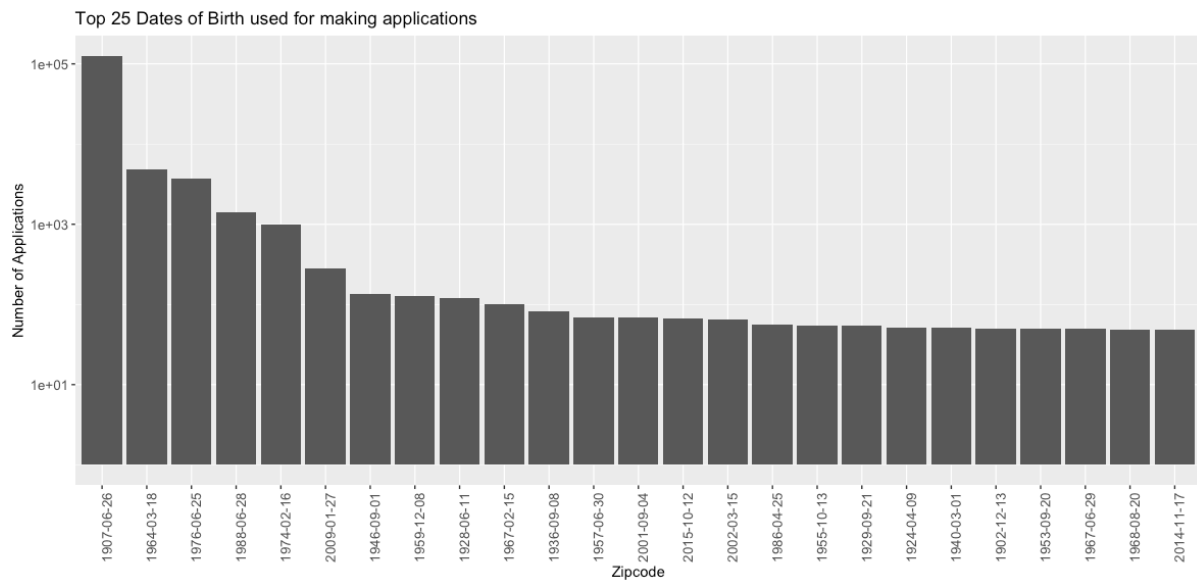


3.8 Date of Birth (dob):

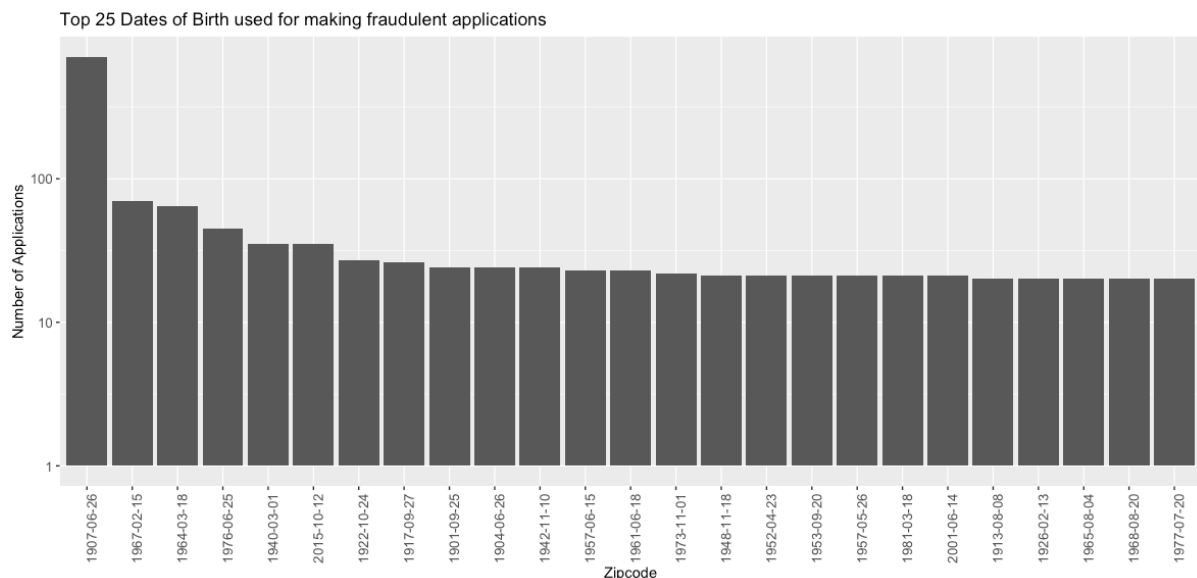
“dob” is an indicator of the Date of Birth of the applicant filing the application.

There are 42,673 unique dates of birth in the dataset. 100% of the dataset is filled.

A distribution of the top 25 birthdates used for filing applications is as follows:



A distribution of the top 25 birthdates used for filing fraudulent applications is as follows:



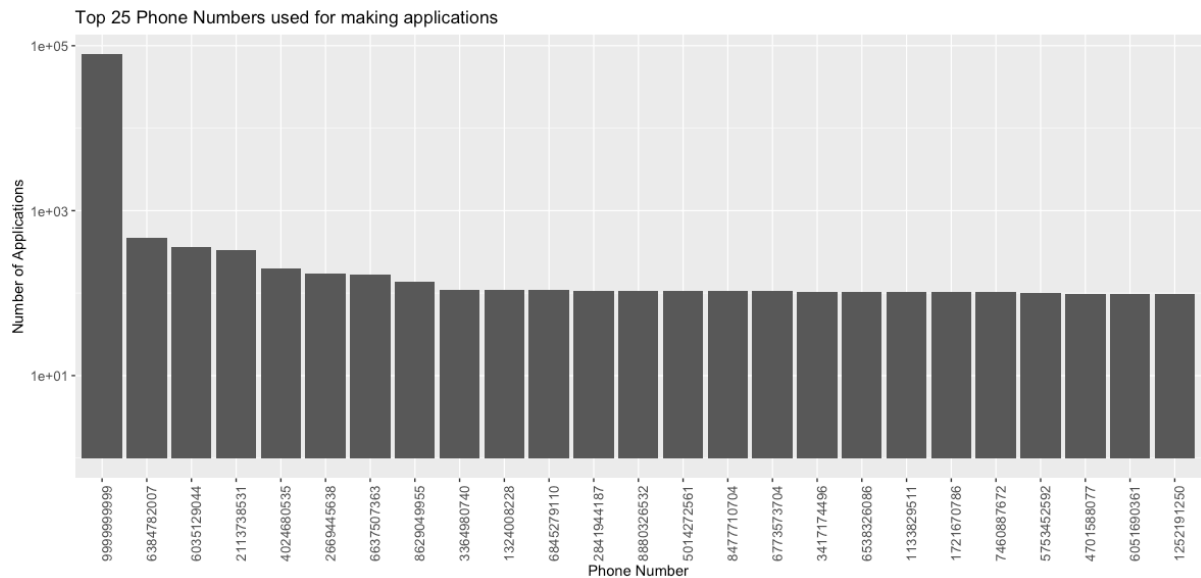
“1907-06-26” seems to be a proxy date used for birthdate. It is unlikely for people with a greater than 100 years of age to exist to make any applications.

3.9 Phone Number (homephone):

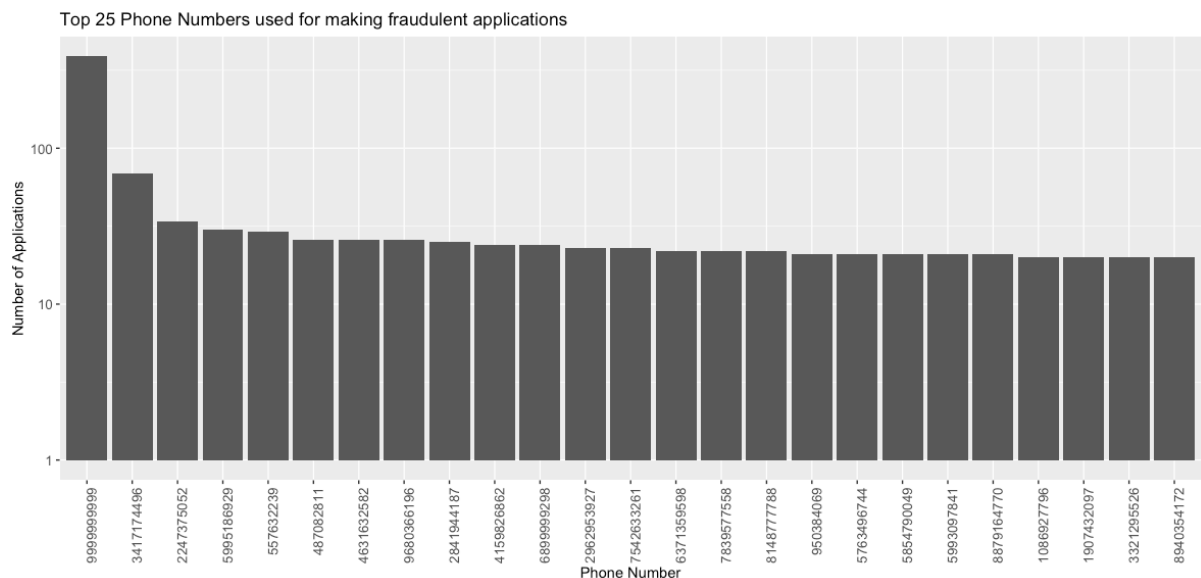
“homephone” is an indicator of the Phone Number of the applicant filing the application.

There are 28,244 unique phone numbers in the dataset. 100% of the dataset is filled.

A distribution of the top 25 phone numbers used for filing applications is as follows:



A distribution of the top 25 phone numbers used for filing fraudulent applications is as follows:



“999-999-9999” is clearly a proxy number being used for phone number while filing applications.

3.10 Fraud Indicator (fraud_label):

“fraud_label” is an indicator to indicate whether the application is fraudulent or not. It is a binary classifier with two categories.

100% of this column is filled.

A summary of this field is as below:

Fraud Category	Category Description	Count
0	Non-fraudulent application	985,607
1	Fraudulent application	14,393

Appendix 2: Glossary of Variables

day since variables	velocity for entities	velocity for combination	risk table and other
Days since per ssn	'datassn0agg',	'datassnfirstname0agg',	'smoothed zip3',
Days since per fulladdress	'datassn1agg',	'datassnfirstname1agg',	smoothed homephone3
Days since per nameDOB	'datassn3agg',	'datassnfirstname3agg',	'smoothed byear',
Days since per homephone	'datassn7agg',	'datassnfirstname7agg',	'smoothed bmonth',
'Days since per ssnfirstname',	'datassn14agg',	'datassnfirstname14agg',	'smoothed bday',
'Days since per ssnlastname',	'datafulladdress0agg',	'datassnlastname0agg',	'smoothed dayofweek
'Days since per ssnaddress',	'datafulladdress1agg',	'datassnlastname1agg',	'no_of_apps',
'Days since per ssnzip5 str',	'datafulladdress3agg',	'datassnlastname3agg',	'len_ssn',
'Days since per ssndob str',	'datafulladdress7agg',	'datassnlastname7agg',	'mean_fraud',
'Days since per ssnhomophone',	'datafulladdress14agg',	'datassnlastname14agg',	'num_phones',
'Days since per ssnnameDOB',	'datanameDOB0agg',	'datassnaddress0agg',	'num_zip',
'Days since per ssnfulladdress',	'datanameDOB1agg',	'datassnaddress1agg',	'num_fraud'
'Days since per firstnamelastname',	'datanameDOB3agg',	'datassnaddress3agg',	
'Days since per firstnameaddress',	'datanameDOB7agg',	'datassnaddress7agg',	
'Days since per firstnamezip5 str',	'datanameDOB14agg',	'datassnaddress14agg',	
'Days since per firstnamedob str',	'datahomephone0agg',	'datassnzip5_str0agg',	
'Days since per firstnamehomephone',	'datahomephone1agg',	'datassnzip5_str1agg',	
'Days since per firstnamenameDOB',		'datassnzip5_str3agg',	
'Days since per firstnamefulladdress',		'datassnzip5_str7agg',	
'Days since per lastnameaddress',		'datassnzip5_str14agg',	
'Days since per lastnamezip5 str',		'datassndob_str0agg',	
'Days since per lastnamedob str',		'datassndob_str1agg',	
'Days since per lastnamehomephone',		'datassndob_str3agg',	
'Days since per lastnamenameDOB',		'datassndob_str7agg',	
'Days since per lastnamefulladdress',		'datassndob_str14agg',	
'Days since per addresszip5 str',		'datassnhomophone0agg',	
'Days since per addressdob str',		'datassnhomophone1agg',	
'Days since per addresshomephone',		'datassnhomophone3agg',	
'Days since per addressnameDOB',		'datassnhomophone7agg',	
'Days since per addressfulladdress',		'datassnhomophone14agg',	
'Days since per zip5 strdob str',		'datassnnameDOB0agg',	
'Days since per zip5 strhomephone',		'datassnnameDOB1agg',	
'Days since per zip5 strnameDOB',		'datassnnameDOB3agg',	
'Days since per zip5 strfulladdress',		'datassnnameDOB7agg',	
'Days since per dob strhomephone',		'datassnnameDOB14agg',	
'Days since per dob strnameDOB',		'datassnfulladdress0agg',	
'Days since per dob strfulladdress',		'datassnfulladdress1agg',	
'Days since per homephonenumberDOB',		'datassnfulladdress3agg',	
'Days since per homephonefulladdress',		'datassnfulladdress7agg',	
'Days since per nameDOBfulladdress',		'datassnfulladdress14agg',	
		'datafirstnamelastname0agg',	
		'datafirstnamelastname1agg',	
		'datafirstnamelastname3agg',	
		'datafirstnamelastname7agg',	
		'datafirstnamelastname14agg',	
		'datafirstnameaddress0agg',	
		'datafirstnameaddress1agg',	

		'datafirstnameaddress3agg',	
		'datafirstnameaddress7agg',	
		'datafirstnameaddress14agg',	
		'datafirstnamezip5_str0agg',	
		'datafirstnamezip5_str1agg',	
		'datafirstnamezip5_str3agg',	
		'datafirstnamezip5_str7agg',	
		'datafirstnamezip5_str14agg',	
		'datafirstnamedob_str0agg',	
		'datafirstnamedob_str1agg',	
		'datafirstnamedob_str3agg',	
		'datafirstnamedob_str7agg',	
		'datafirstnamedob_str14agg',	
		'datafirstnamehomephone0agg',	
		'datafirstnamehomephone1agg',	
		'datafirstnamehomephone3agg',	
		'datafirstnamehomephone7agg',	
		'datafirstnamehomephone14agg',	
		'datafirstnamennameDOB0agg',	
		'datafirstnamennameDOB1agg',	
		'datafirstnamennameDOB3agg',	
		'datafirstnamennameDOB7agg',	
		'datafirstnamennameDOB14agg',	
		'datafirstnamefulladdress0agg',	
		'datafirstnamefulladdress1agg',	
		'datafirstnamefulladdress3agg',	
		'datafirstnamefulladdress7agg',	
		'datafirstnamefulladdress14agg',	
		'datalastnameaddress0agg',	
		'datalastnameaddress1agg',	
		'datalastnameaddress3agg',	
		'datalastnameaddress7agg',	
		'datalastnameaddress14agg',	
		'datalastnamezip5_str0agg',	
		'datalastnamezip5_str1agg',	
		'datalastnamezip5_str3agg',	
		'datalastnamezip5_str7agg',	
		'datalastnamezip5_str14agg',	
		'datalastnamedob_str0agg',	
		'datalastnamedob_str1agg',	
		'datalastnamedob_str3agg',	
		'datalastnamedob_str7agg',	
		'datalastnamedob_str14agg',	
		'datalastnamehomephone0agg',	
		'datalastnamehomephone1agg',	
		'datalastnamehomephone3agg',	
		'datalastnamehomephone7agg',	
		'datalastnamehomephone14agg',	
		'datalastnamennameDOB0agg',	
		'datalastnamennameDOB1agg',	
		'datalastnamennameDOB3agg',	
		'datalastnamennameDOB7agg',	

		'datalastnameDOB14agg',	
		'datalastnamefulladdress0agg',	
		'datalastnamefulladdress1agg',	
		'datalastnamefulladdress3agg',	
		'datalastnamefulladdress7agg',	
		'datalastnamefulladdress14agg',	
		'dataaddresszip5_str0agg',	
		'dataaddresszip5_str1agg',	
		'dataaddresszip5_str3agg',	
		'dataaddresszip5_str7agg',	
		'dataaddresszip5_str14agg',	
		'dataaddressdob_str0agg',	
		'dataaddressdob_str1agg',	
		'dataaddressdob_str3agg',	
		'dataaddressdob_str7agg',	
		'dataaddressdob_str14agg',	
		'dataaddresshomephone0agg',	
		'dataaddresshomephone1agg',	
		'dataaddresshomephone3agg',	
		'dataaddresshomephone7agg',	
		'dataaddresshomephone14agg',	
		'dataaddressnameDOB0agg',	
		'dataaddressnameDOB1agg',	
		'dataaddressnameDOB3agg',	
		'dataaddressnameDOB7agg',	
		'dataaddressnameDOB14agg',	
		'dataaddressfulladdress0agg',	
		'dataaddressfulladdress1agg',	
		'dataaddressfulladdress3agg',	
		'dataaddressfulladdress7agg',	
		'dataaddressfulladdress14agg',	
		'datazip5_strdob_str0agg',	
		'datazip5_strdob_str1agg',	
		'datazip5_strdob_str3agg',	
		'datazip5_strdob_str7agg',	
		'datazip5_strdob_str14agg',	
		'datazip5_strhomephone0agg',	
		'datazip5_strhomephone1agg',	
		'datazip5_strhomephone3agg',	
		'datazip5_strhomephone7agg',	
		'datazip5_strhomephone14agg',	
		'datazip5_strnameDOB0agg',	
		'datazip5_strnameDOB1agg',	
		'datazip5_strnameDOB3agg',	
		'datazip5_strnameDOB7agg',	
		'datazip5_strnameDOB14agg',	
		'datazip5_strfulladdress0agg',	
		'datazip5_strfulladdress1agg',	
		'datazip5_strfulladdress3agg',	
		'datazip5_strfulladdress7agg',	
		'datazip5_strfulladdress14agg',	
		'datadob_strhomephone0agg',	

		'datadob_strhomephone1agg',	
		'datadob_strhomephone3agg',	
		'datadob_strhomephone7agg',	
		'datadob_strnameDOB0agg',	
		'datadob_strnameDOB1agg',	
		'datadob_strnameDOB3agg',	
		'datadob_strnameDOB7agg',	
		'datadob_strnameDOB14agg',	
		'datadob_strfulladdress0agg',	
		'datadob_strfulladdress1agg',	
		'datadob_strfulladdress3agg',	
		'datadob_strfulladdress7agg',	
		'datadob_strfulladdress14agg',	
		'datahomephonenameDOB0agg',	
		'datahomephonenameDOB1agg',	
		'datahomephonenameDOB3agg',	
		'datahomephonenameDOB7agg',	
		'datahomephonenameDOB14agg',	
		'datahomephonefulladdress0agg',	
		'datahomephonefulladdress1agg',	
		'datahomephonefulladdress3agg',	
		'datahomephonefulladdress7agg',	
		'datahomephonefulladdress14agg',	
		'datanameDOBfulladdress0agg',	
		'datanameDOBfulladdress1agg',	
		'datanameDOBfulladdress3agg',	
		'datanameDOBfulladdress7agg',	
		'datanameDOBfulladdress14agg',	