USC Marshall School of Business

Supervised Fraud Model on Applications Data

Group 4 - Di Zhang, Yifeng Wang,

Juxing Wang, Yu Qin, Tianxing Wang

May 2nd, 2019

Table of Content

Executive Summary

This report provides an analysis and evaluation of application data for detecting fraud using supervised machine learning methods. The original data set contains 1,000,000 records of application and has 10 fields describing the details of the application, like application date, ssn, address, name, zip code and date of birth of the applicant.

The general process of analysis follows data quality checking, data cleaning, expert variables creation, important variables selection, applying fraud algorithms, calculating fraud scores and evaluating results. We divided the dataset into training, testing (January to October)and out-of-time(November to December) to evaluate the fraud score for each of these brackets.

The tools used include R, Python and Excel, and some of the algorithms used for analysis are Decision Trees, Random Forest, Boosted Trees, Neural Networks and Logistic Regression. These 5 models were built and tested by a combination of data analytics tools and their respective performances were recorded. Among all the models built, with 100 trees, max depth of 4, min-sample-leaves of 4 predict the fraud in out-of-time data best. The Average FDR for out-of-time data at 3% is 49.3% average over 10 times.

Using this best model, we got Fraud Detection Rates (FDR) of 52.03%, 53.30% and 49.30% for training, testing and out-of-time brackets. And when looking at the importance of expert variables in the algorithm, we found that address, zip and date of birth would be great predictors of fraud.

1. Description of Data

1.1 Description of dataset

This dataset contains 1000000 records of applications happened in 2016, and 14393 of them are fraud applications as their fraud_lable are marked as 1. There are 10 fields in the dataset: record, date, ssn, firstname, lastname, address, zip5, dob, homephone, fraud_lable.

Record is the unique record number of each application happened, also indicating the time sequence of those transactions. Also, date is from 2016-01-01 to 2016-12-31, showing the actual data when the application happened. And fraud_label illustrate whether an application is fraud or not. Other fields are all describing the applicants, their ssn, name, address, zip code, date of birth and home phone, which can help us treat applicant as an entity.

A Data Quality Report in the appendix was attached for further details.

1.2 Summary of variables

The Table1 below shows the summary of all categorical variables:

Field	#values	%pupolated	#unique values	Most common
record	1000000	100	1000000	/
date	1000000	100	365	2016-08-16
ssn	1000000	100	835819	999999999
firstname	1000000	100	78136	EMSTRMT
lastname	1000000	100	177001	ERJSAXA
address	1000000	100	828774	123 MAIN ST
zip5	1000000	100	863348	68138
dob	1000000	100	42673	1907-06-26
homephone	1000000	100	28244	9999999999
Fraud_lable	1000000	100	2	0

The most common fillings in some fields are obviously made by hand, so that we need to do some data cleaning to get rid of those frivolous field fillings.

1.3 Important variables:

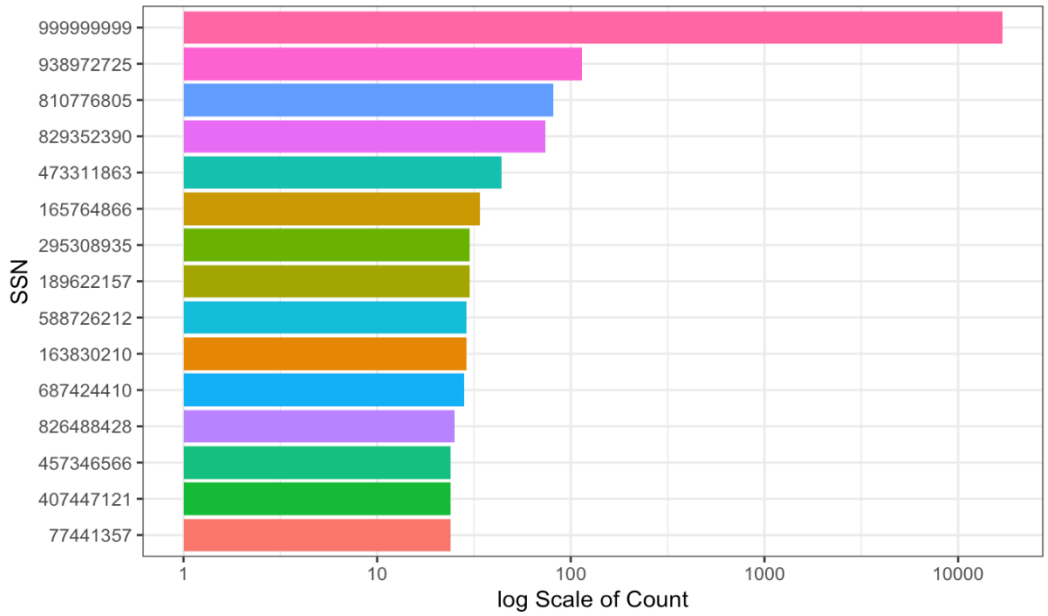
- **Field: ssn**

SSN(Social Sercuity Number) is a specific number for everyone, so ssn is categorical variable.

Number of values: 1000000

Number of unique values: 835819

%populated: 100



From the chart we could know that SSN 999999999 has the highest amount, but they are obviously made up by hand, so some methods are needed to be done to change those 999999999 ssn.

- **Field: date**

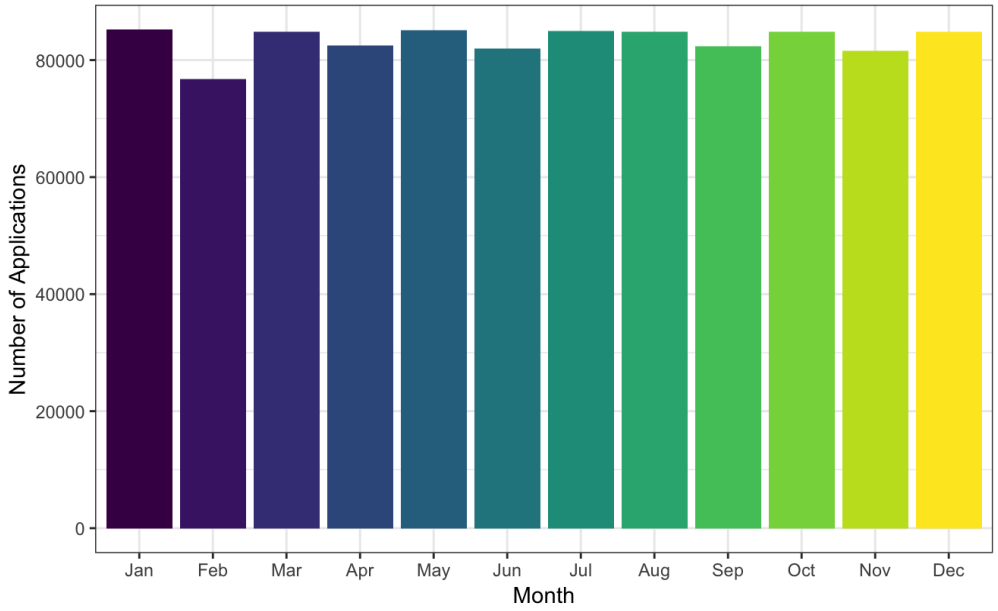
Date is the date then the application happened, so Date is categorical variable.

Number of values: 1000000

Number of unique values: 365

%populated: 100

Distribution of application of month:



From the chart above, we could not see any seasonality of month of the applications.

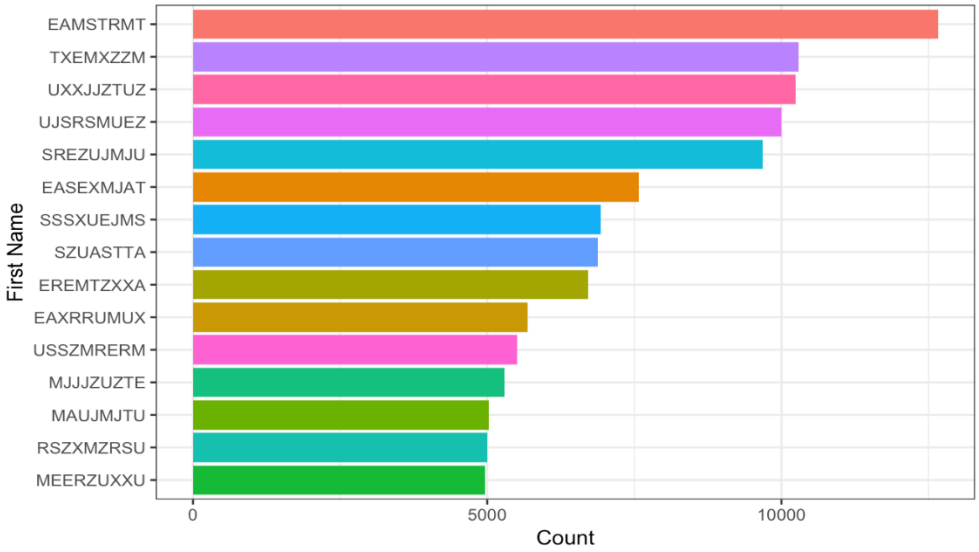
- **Field: firstname**

Firstname is the first name of applicant, so it should be categorical.

Number of values: 1000000

Number of unique values: 78136

%populated: 100



From the chart we could know that the first name appears most in all applications is EAMSTRMT.

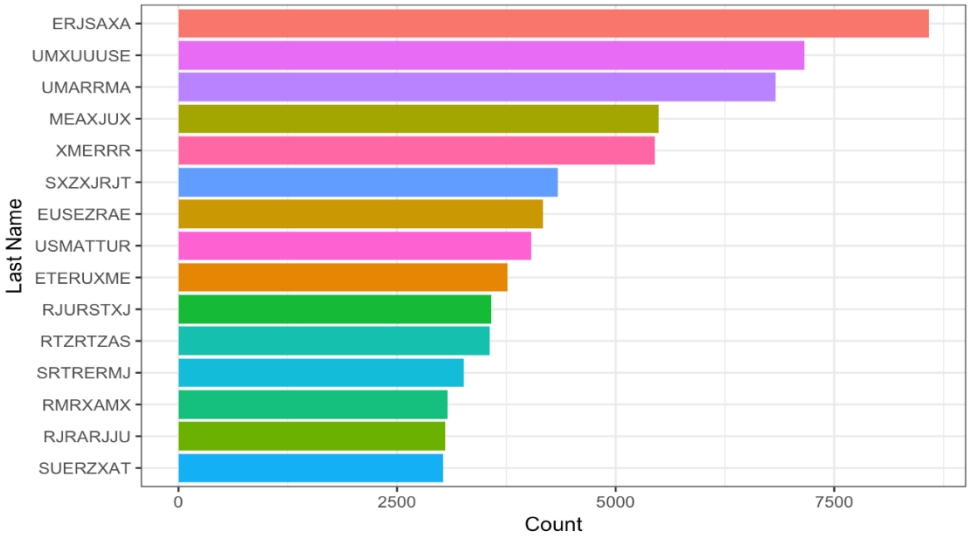
- **Field: lastname**

Lastname is the last name of applicant, so it should be categorical.

Number of values: 1000000

Number of unique values: 177001

%populated: 100



From the chart we could know that the last name appears most in all applications is ERJSAXA.

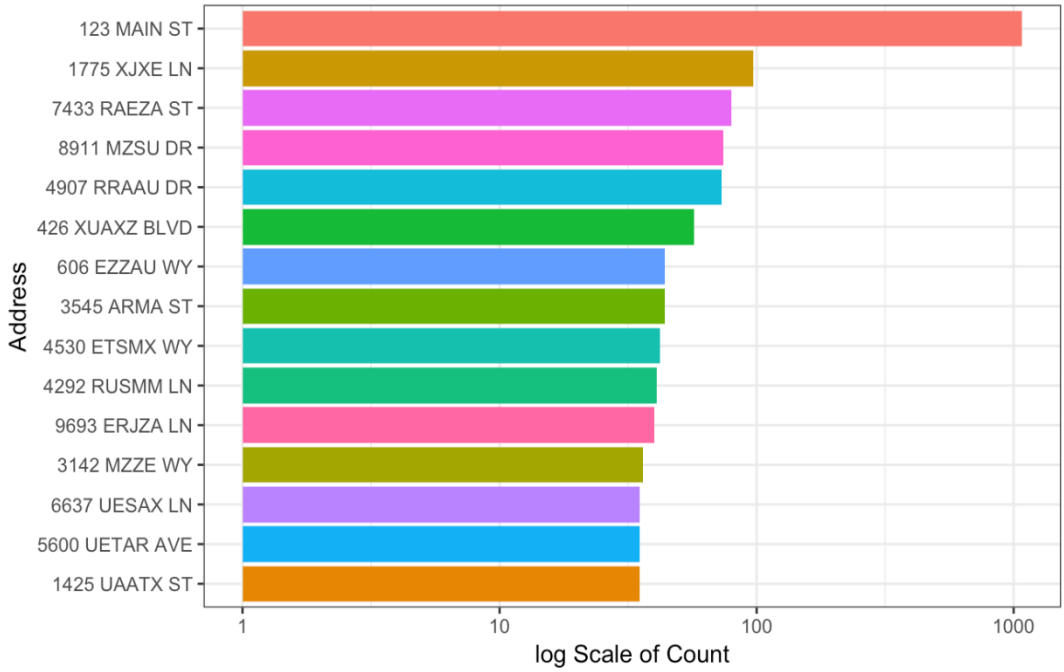
- **Field: address**

Address is the applicant's address, so it is categorical.

Number of values: 1000000

Number of unique values: 828774

%populated: 100



From the chart above, we could see that the address "123 MAIN ST" has the highest amount, but it is obviously made up by hand, so we need to clean the address.

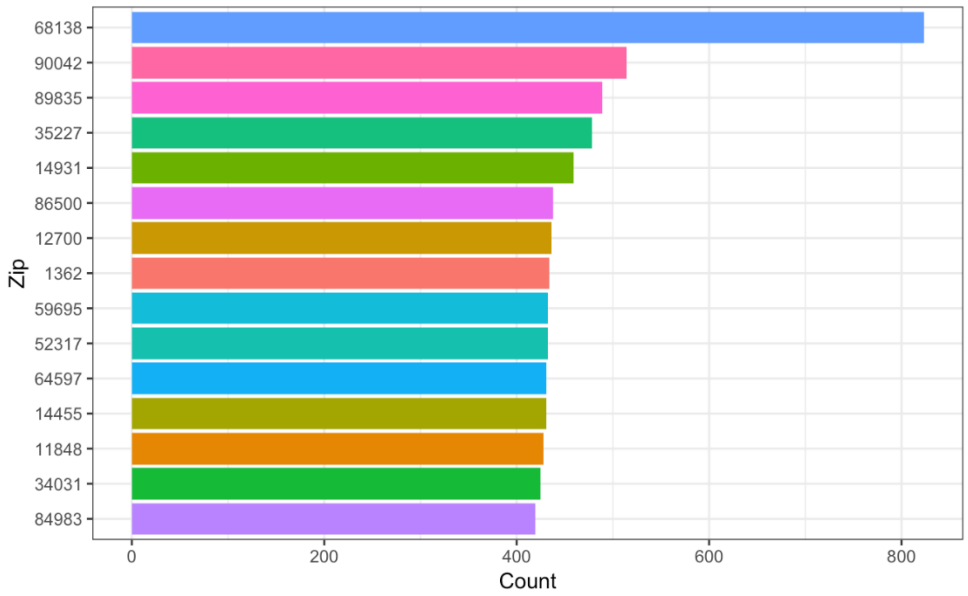
- **Field: zip5**

Zip5 is the zip code of address of applications, so zip5 is a categorical variable.

Number of values: 1000000

Number of unique values: 26370

%populated: 100



From the chart we could know that place with zip code 68138 has highest number of applications.

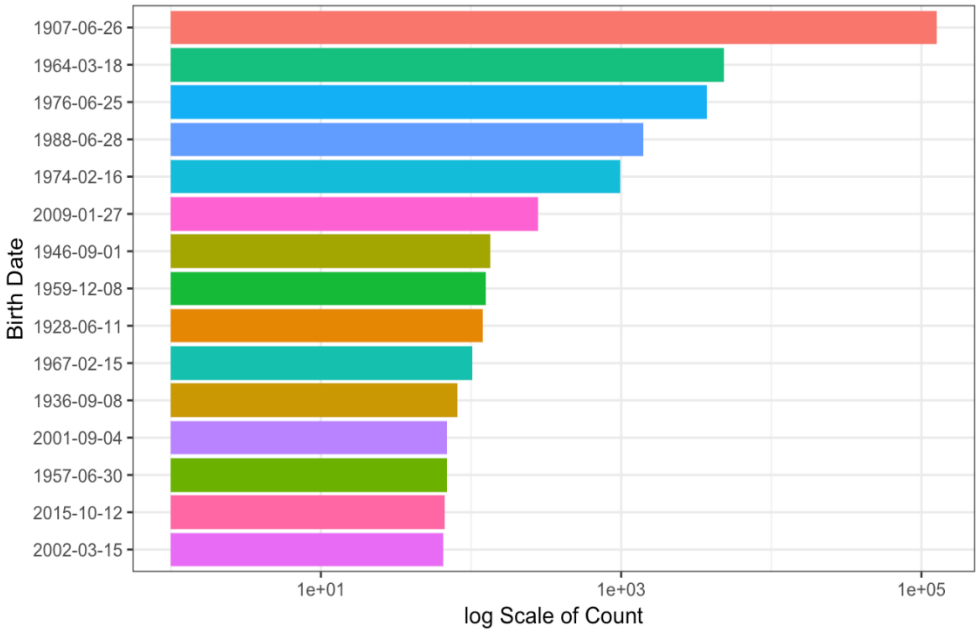
- **Field: dob**

Dob is the birthdate of the applicant, so it is categorical variable.

Number of values: 1000000

Number of unique values: 42673

%populated: 100



From the chart above, we could see that the bithdate “1907-06-26” takes the most proportion, but this date is too far ago and obviously made up by hand.

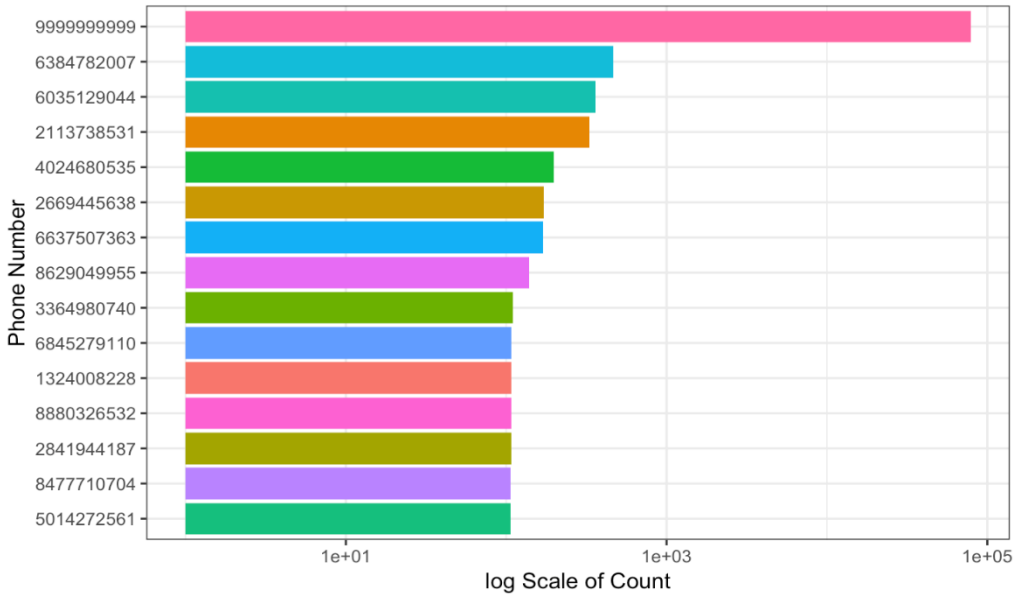
- **Field: homephone**

Homephone is the phone number of the application, so it is a categorical variable.

Number of values: 1000000

Number of unique values: 28244

%populated: 100



From the chart above, we could see that the phone number “9999999999” takes the most proportion, but the this phone number is obviously made up by hand.

2. Data Cleaning

From data quality report, we have noticed there are several abnormal values occurred in some fields, which are SSN, Address, Homephone, and Date of Birth (DOB).

The reason we determine them as abnormal is that these fields all have one type of fillings that is about two order of magnitude larger than other fillings, which is frivolous value for that field. Here is an example, for social security number (SSN), the unusual type is 999999999 and obviously has no meaning. If we want to build variable that links those records, all the variables would have very strong linkage with this value, so the linking by SSN will be strongly skewed.

We think these records are innocuous, and it may be caused by someone who randomly typed them in. So, we probably would get spurious linkage or spurious high fraud score that is not necessarily related to fraud. The way we take care these is to neutralize them by replacing them by the record number of those records. In this way, each record will have a unique value to substitute the frivolous value, and these records will not have any linkages with other records. Table 1 list all the frivolous value and its corresponding variables:

Table 2-1 Frivolous value corresponds to the variable

Variable	**Frivolous Value**
SSN	999999999
Address	123 MAIN ST
Homephone	9999999999
DOB	19070626

3. Variable Creation

We created 314 expert variables for the training set. For the purposes of variable creation, we did not use *Fraud* variable since, using class to predict class isn't an ideal case. Having knowledge about class may give us perfect models but that model might not make business sense.

We then built two types of variables using Python.

Frequency variables are intended to capture unusual frequency of entity occurrence including ssn, dob, name, etc and their combinations.

Velocity variables are intended to capture the change of frequency variables in more recent timeline compared to longer past timeline.

We counted #occurrence for different entities(and combination of entities) across 7 different time windows, namely, 0,1,3,7,30,90 and 180 days. The rationale is to capture more (and different types of) fraudulent entities that might be detected in those time windows.

Here we list the explanation of some representative variables, and all the final variables are shown in the following section.

Variable Name	Description/ Formular
address-zip5-dob_0_count_address-zip5-dob_30_count	#occurrence of the combination of this address, zip5 and date of birth over the past 0 days div sided by #occurrence of the combination of this address, zip5 and date of birth over the past 30 days
address-zip5-dob_1_count_address-zip5-dob_30_count	#occurrence of the combination of this address, zip5 and date of birth over the past 0 days div sided by #occurrence of the combination of this address, zip5 and date of birth over the past 30 days
address-zip5-dob_0_count_address-zip5-dob_90_count	#occurrence of the combination of this address, zip5 and date of birth over the past 0 days div sided by #occurrence of the combination of this address, zip5 and date of birth over the past 90 days
address-zip5-dob_1_count_address-zip5-dob_90_count	#occurrence of the combination of this address, zip5 and date of birth over the past 1 days div sided by #occurrence of the combination of this address, zip5 and date of birth over the past 90 days
address-zip5-dob_0_count_address-zip5-dob_180_count	#occurrence of the combination of this address, zip5 and date of birth over the past 0 days div sided by #occurrence of the combination of this address, zip5 and date of birth over the past 180 days
address-zip5-dob_1_count_address-zip5-dob_180_count	#occurrence of the combination of this address, zip5 and date of birth over the past 1 days div sided by #occurrence of the combination of this address, zip5 and date of birth over the past 180 days
address_count0_date	#occurrence of this address over the past 0 day
address_count1_date	#occurrence of this address over the past 1 day
address_count3_date	#occurrence of this address over the past 3 day

address_count7_date	#occurrence of this address over the past 7 day
address_count30_date	#occurrence of this address over the past 30 day
address_count90_date	#occurrence of this address over the past 90 day
address_count180_date	#occurrence of this address over the past 180 day
address-zip5-dob_count0_date	#occurrence of the combination of this address, zip5 and date of birth over the past 0 days
address-zip5-dob_count1_date	#occurrence of the combination of this address, zip5 and date of birth over the past 1 days
address-zip5-dob_count3_date	#occurrence of the combination of this address, zip5 and date of birth over the past 3 days
address-zip5-dob_count7_date	#occurrence of the combination of this address, zip5 and date of birth over the past 7 days
address-zip5-dob_count30_date	#occurrence of the combination of this address, zip5 and date of birth over the past 30 days
address-zip5-dob_count90_date	#occurrence of the combination of this address, zip5 and date of birth over the past 90 days
address-zip5-dob_count180_date	#occurrence of the combination of this address, zip5 and date of birth over the past 180 days

4. Feature Selection

Feature selection is the process of selecting a subset of relevant variables or predictors in order to diminish dimensionality for model construction. In this process, we used two categories methods, filter and wrapper, to prepare for the modeling.

4.1 KS and FDR

This method does not involve with modeling method. For each variable, we classified distribution between good and bad. By doing these, Kolmogorov-Smirnov (KS) and Fraud Detection Rate (FDR) methods were applied here.

(1) Kolmogorov-Smirnov (KS)

This method creates a robust measure of how well fraud and non-fraud distributions are separated. This test serves as a goodness of fit test, and it measures the differences between the cumulative distribution of fraud and non-fraud for each variable. The procedure we applied is described below.

Kolmogorov-Smirnov (KS) Procedure
1. Prepare the dataset: the dataset was extracted up until end of October for this process a) January – October Dataset: training and testing for modeling (training randomly choose 80%, testing is the remained 20%) b) November – December Dataset: out of time data
2. Separate "January-October" dataset based on the fraud score (0/1) for each variable
3. Calculate the KS score for each variable based on the two-fraud score (0/1) datasets
4. Re-ranked the variable based on theirs KS score

4.2 Fraud Detection Rate (FDR)

Fraud detection rate is percent of total frauds caught at a cutoff point. In this project, we used 0.03 as our cutoff point, so FDR would detect the percent of total fraud at top 3% of each variable. The procedure we applied is described below.

Fraud Detection Rate (FDR) Procedure
1. Prepare the dataset: the dataset was extracted up until end of October for this process a) January – October Dataset: training and testing for modeling b) November – December Dataset: out of time data
2. Sort each variable in descending order and in ascending order
3. Calculate the fraud percentage that being caught at top 3% of each variable
4. Set the larger FDR score as final score by comparing both descending and ascending columns arrange by each variable
5. Re-ranked the variable based on the final FDR score

4.3 Sum of KS and FDR

After KS and FDR was completed, there were two rankings for each variable, FDR ranking and KS ranking. By calculating the sum of them, we got the final rankings for all the variables. The figure below illustrates the final results in ascending order of sum rank before wrapper. We choose the top 180 variables as the filtered variables.

Variable	FDR	KS	FDR_Rank	KS_Rank	FINAL_Rank
FraudLabel	1	1	1	1	2
address_count30_date	0.35	0.33	2	2	4
address-zip5_count30_date	0.35	0.33	3	3	6
address-zip5_count90_date	0.35	0.32	5	4	9
address_count90_date	0.35	0.32	4	5	9
address_count180_date	0.34	0.32	7	6	13
address-zip5_count180_date	0.34	0.32	6	7	13
address-zip5_count7_date	0.32	0.3	8	8	16
address_count7_date	0.32	0.3	9	9	18
address-zip5_count3_date	0.3	0.28	11	12	23
address_count1_date	0.28	0.26	12	15	27
address-zip5_0_count_address-zip5_7_count_Ave	0.28	0.25	14	17	31

From the chart above, we could see that Fraud has a perfect performance, but Random label is near zero, having the worst prediction.

4.4 Wrapper

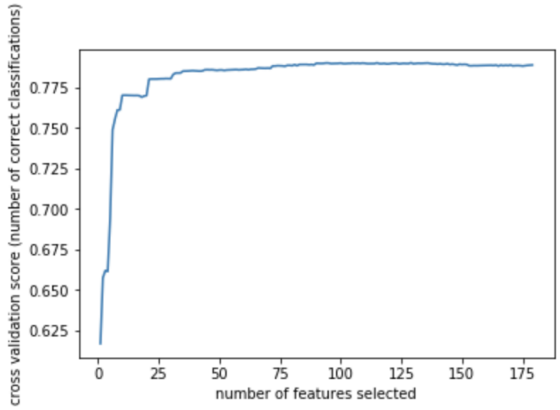
Recursive feature elimination and cross-validated selection with logistics regression is used to get the feature rankings, then selecting best number of features.

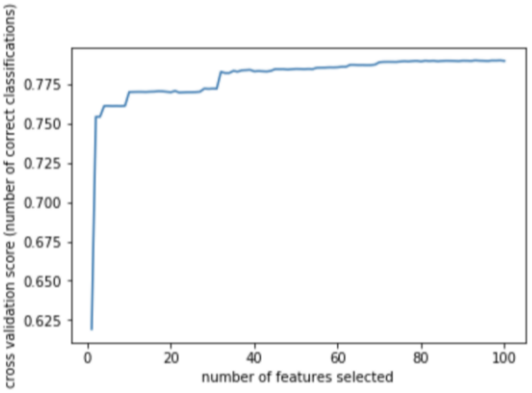
The recursive feature elimination function implements backwards selection of features based on the importance ranking of features. It trains all features in the Intime dataset, get the importance from the logistics regression function as set, then removing the weakest feature. It conducts this step recursively until specific number of features is reached.

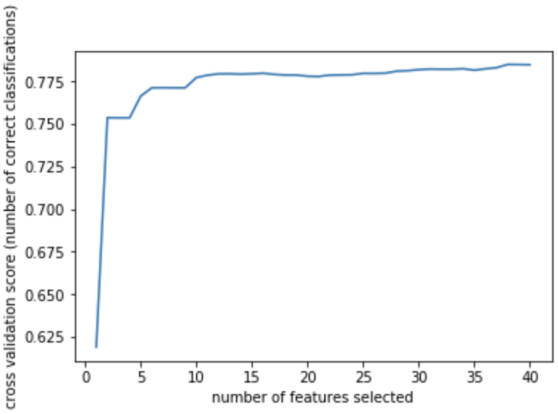
But we do not know the number that we want to choose, the cross-validation provides a score for us to select the most proper subset containing the features we want. The RFECV visualizer provides us a plot with number of features of subset and corresponding cross-validation test score.

Also, since our Fraud label is binomial (0 or 1), it's better to use logistics regression in the RFECV function to get the importance of each feature.

The following 3 plots shows the 3 backward stepwise selection from 180 to 100 to 50 to 25







Finally, the first top 50 features from last wrapper ranking results are gotten and RFE(,n) function is conducted in python, the n is set to 25 to get the most important 25 features from the original 314 features. Then we get the list of final 25 features as follows:

'address-zip5-homephone_count30_date'	'address-zip5_0_count_address-zip5_30_count_Ave'	'address-zip5_0_count_address-zip5_7_count_Ave'
'address-zip5_count0_date', 'address-zip5_count180_date'	'address-zip5_count1_date'	'address-zip5_count30_date'
'address-zip5_count3_date'	'address-zip5_count7_date'	'address-zip5_count90_date'
'address_0_count_address_180_count_Ave'	'address_0_count_address_30_count_Ave'	address_0_count_address_7_count_Ave
'address_count0_date'	'address_count180_date	'address_count1_date
'address_count30_date	'address_count3_date'	'address_count7_date'
'address_count90_date'	'name-dob_count180_date'	'name-dob_count30_date'
'name-dob_count90_date'	'ssn-dob_count180_date'	'ssn-dob_count30_date'

5. Algorithms

We have tried five supervised models combining the feature selection algorithm.

5.1 Logistic Regression

Use sigmoid function to estimate the probability, and predict probability of each record. Explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

	Training	**Test**	**OOT**
1	0.5160	0.5102	0.4966
2	0.5071	0.5099	0.4891
3	0.5061	0.5130	0.4887
4	0.5096	0.5049	0.4895
5	0.5168	0.5111	0.4962
6	0.5140	0.5165	0.4962
7	0.5089	0.5330	0.4958
8	0.5109	0.4947	0.4887
9	0.5149	0.5131	0.4958
10	0.5086	0.5082	0.4895
Average	0.5113	0.5115	0.4926

5.2 Neural Network

A Neural Network is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. Our Neural Network has 20 input layer nodes and 20 hidden layer nodes.

	Training	**Test**	**OOT**
1	0.4836	0.3545	0.3817
2	0.5658	0.4694	0.3929
3	0.4666	0.3947	0.3538
4	0.5473	0.4465	0.3761

5	0.5459	0.4522	0.3761
6	0.5190	0.4062	0.3538
7	0.4893	0.4120	0.3761
8	0.5233	0.4407	0.3761
9	0.4289	0.4350	0.3873
10	0.4111	0.4924	0.4040
average	0.4831	0.4733	0.4642

5.3 Random Forest Classifier

A random forest is an ensemble combining a number of decision tree classifiers on various subsets of samples in the dataset and using averaging methods to improve the predictive accuracy and decrease over-fitting.

One of the main parameters in the Random Forest classifier is n_estimators, which represents number of trees in the forest, the larger the better. But it will take longer time to compute and sometimes the increasing number of trees has no significant influence of results, so we hope to take a trade-off in choosing the number to have better efficacy. And in this project, we tried 50, 100, 200 and 300 and 200 works best from those 4 choices.

In addition, two criterions have been tried in this classifier, “gini” for the Gini impurity and “entropy” for the information gain. From the results we could obviously see that “entropy” works much better in this dataset.

Also, for the min_samples_leaf which means minimum number of samples required to be at a leaf node, we choose 10. And we defined those trees have a max depth of 15. The results of ten times running shows as follows, and we could notice that the average reaches 51.65%, 52.7% and 49.09% in training, testing and out of time dataset. Those results demonstrate the algorithm is not over fitting and could predict frauds very well.

	Training	**Test**	**OOT**
1	0.5170	0.5284	0.4916
2	0.5154	0.5237	0.4895
3	0.5171	0.5289	0.4916
4	0.5171	0.5271	0.4907
5	0.5151	0.5237	0.4895

6	0.5170	0.5284	0.4916
7	0.5153	0.5237	0.4890
8	0.5170	0.5284	0.4920
9	0.5171	0.5289	0.4920
10	0.5170	0.5284	0.4916
Average	0.5165	0.5270	0.4909

5.4 Decision Tree Classifier

Decision Tree partitions the feature spaces into multiple high-dimensional boxes, and give predictions according to the majority vote in each box. A single tree tend to have bias. We tested different parameters manually to try to arrive at higher FDR. Even so, the results are less than satisfactory. Among the adjustments, this is a relatively good model: criterion is gini impurity, maximum depth is 5, minimum samples leaf is also 5. The FDR table for the Tree model is as follows:

	Training	Testing	OOT
0	0.497414	0.444186	0.345251
1	0.483604	0.37801	0.334078
2	0.493853	0.389055	0.311732
3	0.519708	0.40765	0.317318
4	0.489209	0.413873	0.328492
5	0.467525	0.407101	0.306145
6	0.474889	0.430769	0.306145
7	0.541534	0.471186	0.328492
8	0.462391	0.447253	0.356425
9	0.475556	0.407254	0.345251
10	0.490568	0.419634	0.327933
Average	0.455368	0.446434	0.424233

5.5 Boosting

Boosting is the process of building a large, additive decision tree by fitting a sequence of smaller decision trees, called layers. The tree at each layer consists of a small number of splits. The tree is fit based on the

residuals of the previous layers, which allows each layer to correct the fit for bad fitting data from the previous layers. The final prediction for an observation is the sum of the predictions for that observation over all of the layers.

	Training	**Test**	**OOT**
1	50.5	51.9	48.2
2	51.5	52.7	49.1
3	52.0	53.3	49.3
4	52.4	53.4	49.7
5	52.8	53.9	50.3
6	53.4	54.4	51.0
7	53.9	54.9	51.3
8	54.3	55.3	51.8
9	54.8	56.0	52.3
10	55.3	56.6	52.8
average	52.03	53.30	49.30

6. Results and Observations

Among the 5 supervised learning models we built, Boosted Trees gave us the best results.

	Training	Test	Out of time
Log Reg	51.13	51.15	49.26
NN	48.31	47.33	46.42
RF	51.70	52.70	49.10
DT	45.53	44.64	42.42
Boosting	52.03	53.30	49.30

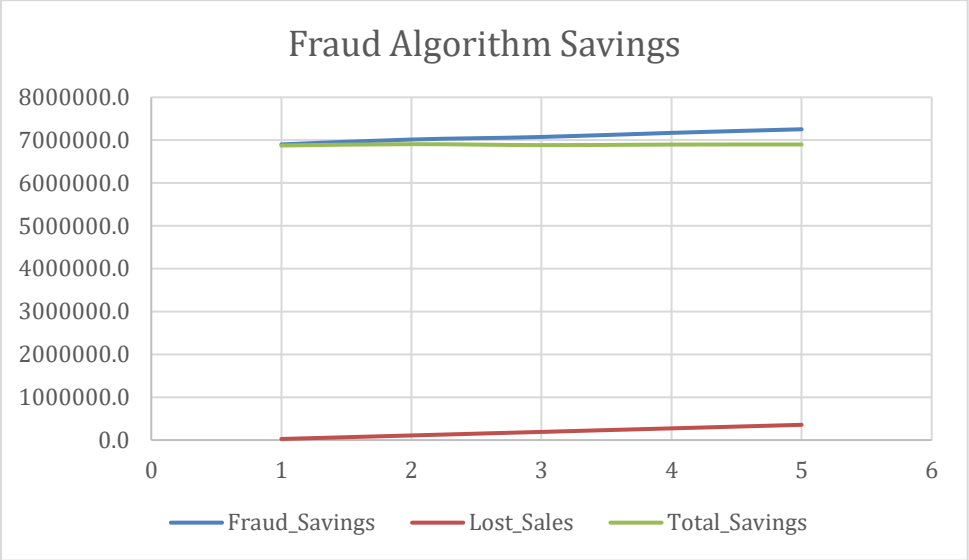
For our final choice, we used 100 trees, 4 max depth, 4 min-sample-leaves boosted trees model. There was a certain level of overfitting, nonetheless, the results were decent. The 3 tables shown below summarizes the FDR at all levels from 1% to 20% of Train, test and OOT for boosted trees.

Training	#Records	#Goods	#Bads	Fraud Rate						
	640433	631186	9247	0.0144						
	Bin Statistics					Cumulative Statistics						
Population Bin %	#Records	#Good	#Bad	%Good	%Bad	Total #Records	Cumulative Good	Cumulative Bad	%Good	%Bad(FDR)	KS	FPR
1	6404	1738	4666	27.1	72.9	6404	1738	4666	0.3	50.5	50.2	0.4
2	6404	6307	97	98.5	1.5	12808	8045	4763	1.3	51.5	50.2	1.7
3	6404	6356	48	99.3	0.7	19212	14401	4811	2.3	52.0	49.7	3.0
4	6405	6369	36	99.4	0.6	25617	20770	4847	3.3	52.4	49.1	4.3
5	6404	6365	39	99.4	0.6	32021	27135	4886	4.3	52.8	48.5	5.6
6	6404	6351	53	99.2	0.8	38425	33486	4939	5.3	53.4	48.1	6.8
7	6405	6361	44	99.3	0.7	44830	39847	4983	6.3	53.9	47.6	8.0
8	6404	6364	40	99.4	0.6	51234	46211	5023	7.3	54.3	47.0	9.2
9	6404	6360	44	99.3	0.7	57638	52571	5067	8.3	54.8	46.5	10.4
10	6405	6361	44	99.3	0.7	64043	58932	5111	9.3	55.3	45.9	11.5
11	6404	6359	45	99.3	0.7	70447	65291	5156	10.3	55.8	45.4	12.7
12	6404	6356	48	99.3	0.7	76851	71647	5204	11.4	56.3	44.9	13.8
13	6405	6356	49	99.2	0.8	83256	78003	5253	12.4	56.8	44.4	14.8
14	6404	6351	53	99.2	0.8	89660	84354	5306	13.4	57.4	44.0	15.9
15	6404	6361	43	99.3	0.7	96064	90715	5349	14.4	57.8	43.5	17.0
16	6405	6350	55	99.1	0.9	102469	97065	5404	15.4	58.4	43.1	18.0
17	6404	6364	40	99.4	0.6	108873	103429	5444	16.4	58.9	42.5	19.0
18	6404	6348	56	99.1	0.9	115277	109777	5500	17.4	59.5	42.1	20.0
19	6405	6356	49	99.2	0.8	121682	116133	5549	18.4	60.0	41.6	20.9
20	6404	6361	43	99.3	0.7	128086	122494	5592	19.4	60.5	41.1	21.9

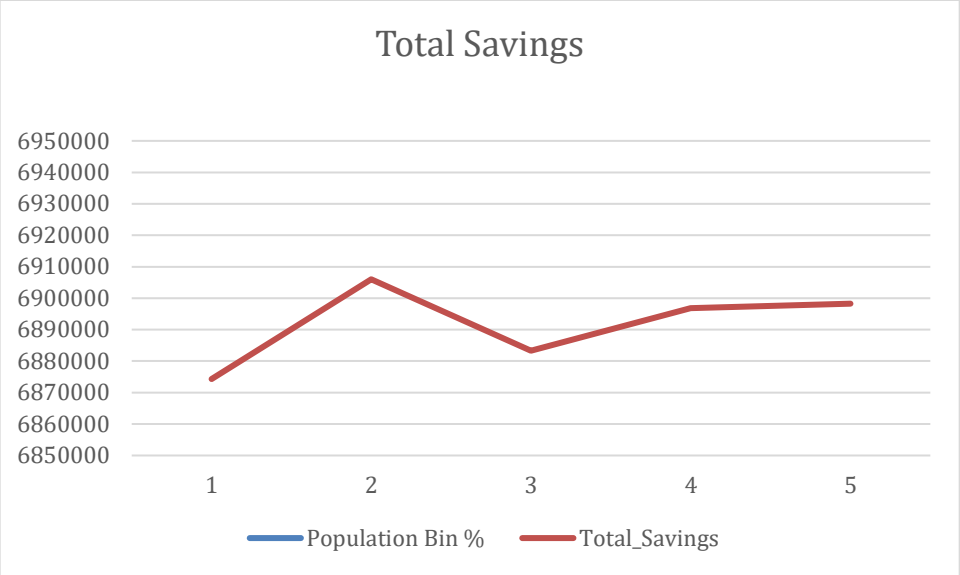
Testing	#Records	#Goods	#Bads	Fraud Rate						
	160109	157814	2295	0.0143						
	Bin Statistics					Cumulative Statistics						
Population Bin %	#Records	#Good	#Bad	%Good	%Bad	Total #Records	Cumulative Good	Cumulative Bad	%Good	%Bad(FDR)	KS	FPR
1	1601	411	1190	25.7	74.3	1601	411	1190	0.3	51.9	51.6	0.3
2	1601	1581	20	98.8	1.2	3202	1992	1210	1.3	52.7	51.5	1.6
3	1601	1587	14	99.1	0.9	4803	3579	1224	2.3	53.3	51.1	2.9
4	1601	1599	2	99.9	0.1	6404	5178	1226	3.3	53.4	50.1	4.2
5	1601	1591	10	99.4	0.6	8005	6769	1236	4.3	53.9	49.6	5.5
6	1601	1589	12	99.3	0.7	9606	8358	1248	5.3	54.4	49.1	6.7
7	1601	1590	11	99.3	0.7	11207	9948	1259	6.3	54.9	48.6	7.9
8	1601	1590	11	99.3	0.7	12808	11538	1270	7.3	55.3	48.0	9.1
9	1601	1586	15	99.1	0.9	14409	13124	1285	8.3	56.0	47.7	10.2
10	1601	1588	13	99.2	0.8	16010	14712	1298	9.3	56.6	47.2	11.3
11	1601	1591	10	99.4	0.6	17611	16303	1308	10.3	57.0	46.7	12.5
12	1602	1591	11	99.3	0.7	19213	17894	1319	11.3	57.5	46.1	13.6
13	1601	1589	12	99.3	0.7	20814	19483	1331	12.3	58.0	45.7	14.6
14	1601	1591	10	99.4	0.6	22415	21074	1341	13.4	58.4	45.1	15.7
15	1601	1590	11	99.3	0.7	24016	22664	1352	14.4	58.9	44.5	16.8
16	1601	1591	10	99.4	0.6	25617	24255	1362	15.4	59.3	44.0	17.8
17	1601	1591	10	99.4	0.6	27218	25846	1372	16.4	59.8	43.4	18.8
18	1601	1592	9	99.4	0.6	28819	27438	1381	17.4	60.2	42.8	19.9
19	1601	1594	7	99.6	0.4	30420	29032	1388	18.4	60.5	42.1	20.9
20	1601	1589	12	99.3	0.7	32021	30621	1400	19.4	61.0	41.6	21.9

Out Of Time	#Records		#Goods		#Bads		Fraud Rate					
	166493		164107		2386		0.0143					
	Bin Statistics					Cumulative Statistics						
Population Bin %	#Records	#Good	#Bad	%Good	%Bad	Total #Records	Cumulative Good	Cumulative Bad	%Good	%Bad(FDR)	KS	FPR
1	1664	514	1150	30.9	69.1	1664	514	1150	0.3	48.2	47.9	0.4
2	1665	1644	21	98.7	1.3	3329	2158	1171	1.3	49.1	47.8	1.8
3	1665	1660	5	99.7	0.3	4994	3818	1176	2.3	49.3	47.0	3.2
4	1665	1654	11	99.3	0.7	6659	5472	1187	3.3	49.7	46.4	4.6
5	1665	1651	14	99.2	0.8	8324	7123	1201	4.3	50.3	46.0	5.9
6	1665	1648	17	99.0	1.0	9989	8771	1218	5.3	51.0	45.7	7.2
7	1665	1660	5	99.7	0.3	11654	10431	1223	6.4	51.3	44.9	8.5
8	1665	1652	13	99.2	0.8	13319	12083	1236	7.4	51.8	44.4	9.8
9	1665	1653	12	99.3	0.7	14984	13736	1248	8.4	52.3	43.9	11.0
10	1665	1653	12	99.3	0.7	16649	15389	1260	9.4	52.8	43.4	12.2
11	1665	1656	9	99.5	0.5	18314	17045	1269	10.4	53.2	42.8	13.4
12	1665	1648	17	99.0	1.0	19979	18693	1286	11.4	53.9	42.5	14.5
13	1665	1657	8	99.5	0.5	21644	20350	1294	12.4	54.2	41.8	15.7
14	1665	1653	12	99.3	0.7	23309	22003	1306	13.4	54.7	41.3	16.8
15	1664	1658	6	99.6	0.4	24973	23661	1312	14.4	55.0	40.6	18.0
16	1665	1657	8	99.5	0.5	26638	25318	1320	15.4	55.3	39.9	19.2
17	1665	1655	10	99.4	0.6	28303	26973	1330	16.4	55.7	39.3	20.3
18	1665	1648	17	99.0	1.0	29968	28621	1347	17.4	56.5	39.0	21.2
19	1665	1640	25	98.5	1.5	31633	30261	1372	18.4	57.5	39.1	22.1
20	1665	1649	16	99.0	1.0	33298	31910	1388	19.4	58.2	38.7	23.0

We assume $6000 gain for each fraud caught and $50 loss for each false positive, below is the savings chart:



Let's look at the total savings alone:



The optimal total savings is reached by targeting the top 2% suspicious applications.

7. Conclusion

For this project, analysis and evaluation of application data were conducted for detecting fraud using supervised machine learning methods.

First a detailed data quality report is made to find the type and value distribution of each field, and some distribution plots are made to understand the data better. Then, data cleaning is conducted with R, to filter and get rid of exclusions, outliers and frivolous field data.

After the cleaning, 314 expert variables are built focusing on the different combinations of applicant's information based the different time windows. And one risk table variable is built based on ssn of applicants.

The dataset is divided into training, testing (January to October) and out-of-time (November and December) to fit feature selection and model building. KS score and FDR score are calculated first to get the first 180 ranking features, then backward wrapper is conducted twice to condense the subset of variables, from 180 to 100, from 100 to 50 variables, and finally from 50 to 25 variables for future modeling.

The algorithms used for modeling are Decision Trees, Random Forest, Boosted Trees, Neural Networks and Logistic Regression. We run them with our training(randomly select 80% from the Intime data), testing (remaining 20% from the Intime data) and out-of-time data. Among all the models built, Boosted Tree with 100 trees, max depth of 4, min-sample-leaves of 4 predict the fraud in out-of-time data best. The Average FDR for out-of-time data at 3% is 49.3% average over 10 times.

Through our analysis we could further identify that there were several fields that were strong predictors of fraud, like address, zip and date of birth. Detailed examination of the importance of expert variables in algorithm illustrates that application number counted based on address and count in 30-day time

window and application number counted based on address in 30-day window really contain large amount of information for fraud detection

If we could do more analysis on this project, we hope to build better wrapper to select the feature more quickly, for that it runs too long time and there are too many steps for wrapper to get relatively important variables. Besides, we have tried Decision Tree in this project, but FDR for out-of-time data is only 42.42%, so if given more time we hope to gain more appropriate parameters for the algorithm from testing the parameters. Also, we would like to try better weighting when creating expert variables to show our understanding of fraud pattern.

Appendix: Data Quality Report

Section1: Basic information of dataset

This dataset contains 1000000 records of applications happened in 2016, and 14393 of them are fraud applications as their fraud_lable are marked as 1. There are 10 fields in the dataset: record, date, ssn, firstname, lastname, address, zip5, dob, homephone, fraud_lable.

Section2: Introduction of dataset

1.dimension of dataset

```
## [1] 1000000      10
```

The dataset has 1000000 rows and 10 columns.

2.names of fields

```
##   "record"      "date"        "ssn"         "firstname"    "lastname"

##   "address"     "zip5"        "dob"         "homephone"    "fraud_label"
```

3.brief introduction of each field

```
## 'data.frame':    1000000 obs. of  10 variables:

##  $ record     : int  1 2 3 4 5 6 7 8 9 10 ...

##  $ date       : int  20160101 20160101 20160101 20160101 20160101 20160101 20160101 20160101 2016
0101 20160101 ...

##  $ ssn        : int  379070012 387482503 200332444 747451317 24065868 922264214 415812149 3737520
50 769970791 732119085 ...

##  $ firstname  : Factor w/ 78136 levels "AAAEEETR","AAAEJAEM",..: 69302 14459 47838 38352 42762
64810 75739 4423 33436 6762 ...

##  $ lastname   : Factor w/ 177001 levels "AAAAARA","AAAAXZR",..: 99191 73286 33683 28863 109224
 20833 112223 1478 70261 36871 ...

##  $ address    : Factor w/ 828774 levels "0 EAJA DR","0 EAJMZ ST",..: 539937 578643 421781 35705 164
584 305690 347827 728287 669434 752457 ...

##  $ zip5       : int  2765 57169 56721 35286 3173 8391 41640 60567 37934 93751 ...

##  $ dob        : int  19070626 19340615 19070626 19440430 19980315 19480613 19640318 19190528 1990
0314 19750127 ...

##  $ homephone  : num  1.80e+09 4.16e+09 2.17e+08 1.32e+08 6.10e+09 ...

##  $ fraud_label: int  0 1 0 0 0 0 0 0 0 0 ...
```

4.summary of categorical variables

Field	#values	%pupolated	#unique values
record	1000000	100	1000000
date	1000000	100	365
ssn	1000000	100	835819
firstname	1000000	100	78136
lastname	1000000	100	177001
address	1000000	100	828774
zip5	1000000	100	863348
dob	1000000	100	42673
homephone	1000000	100	28244
Fraud_lable	1000000	100	2

Section3: Introduction of each field

Field1: record

Record is the key of each record, so it is categorical variable.

Type: Categorical

int [1:1000000] 1 2 3 4 5 6 7 8 9 10 ...

Number of values: 1000000

Number of unique values: 1000000

Number of unique values: 365

%populated: 100

Field2: date

Type: Categorical

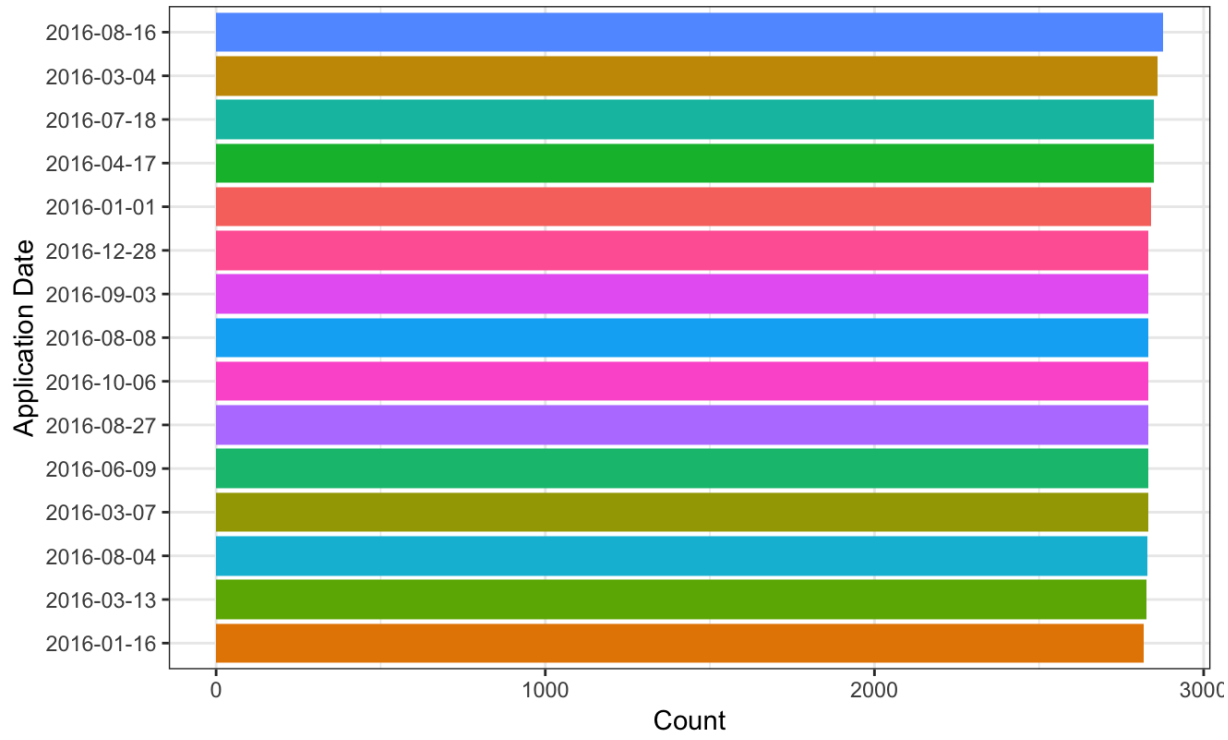
int [1:1000000] 20160101 20160101 20160101 20160101 20160101 20160101 20160101 20160101 20160101 20160101 ...

Date is the date then the application happened, so Date is categorical variable.

Number of values: 1000000

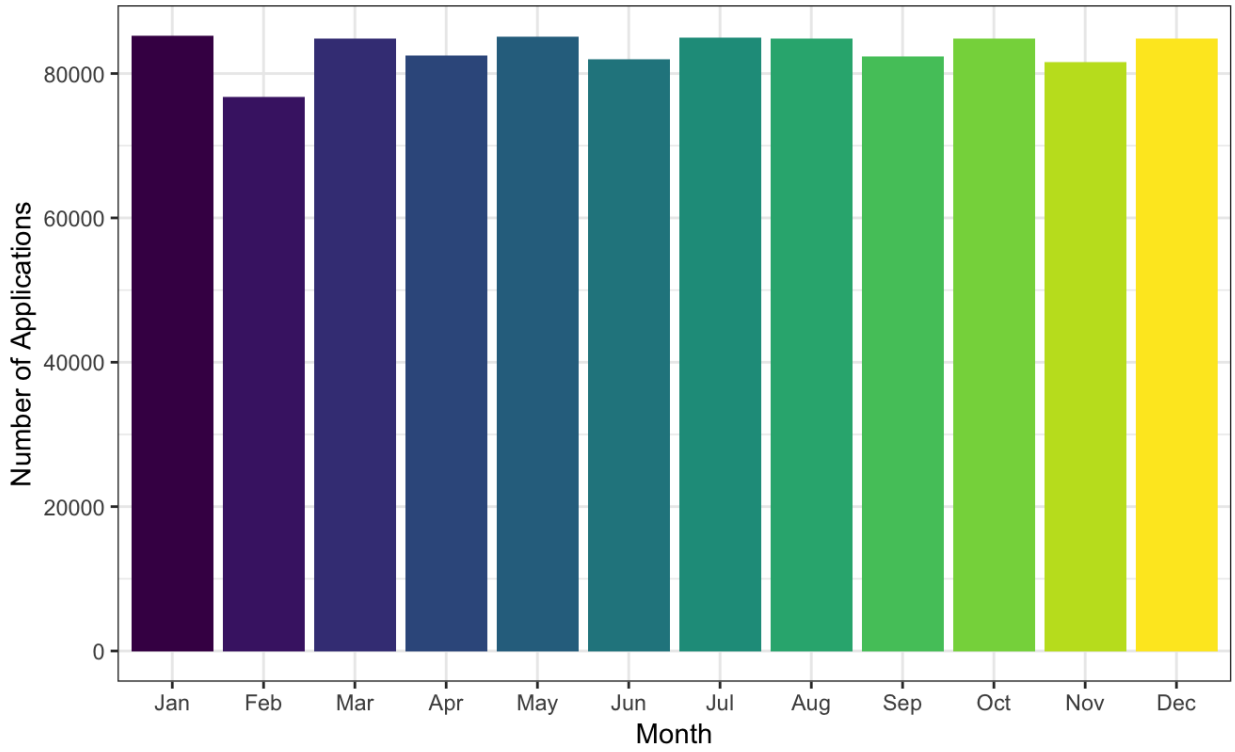
Number of unique values: 365

%populated: 100



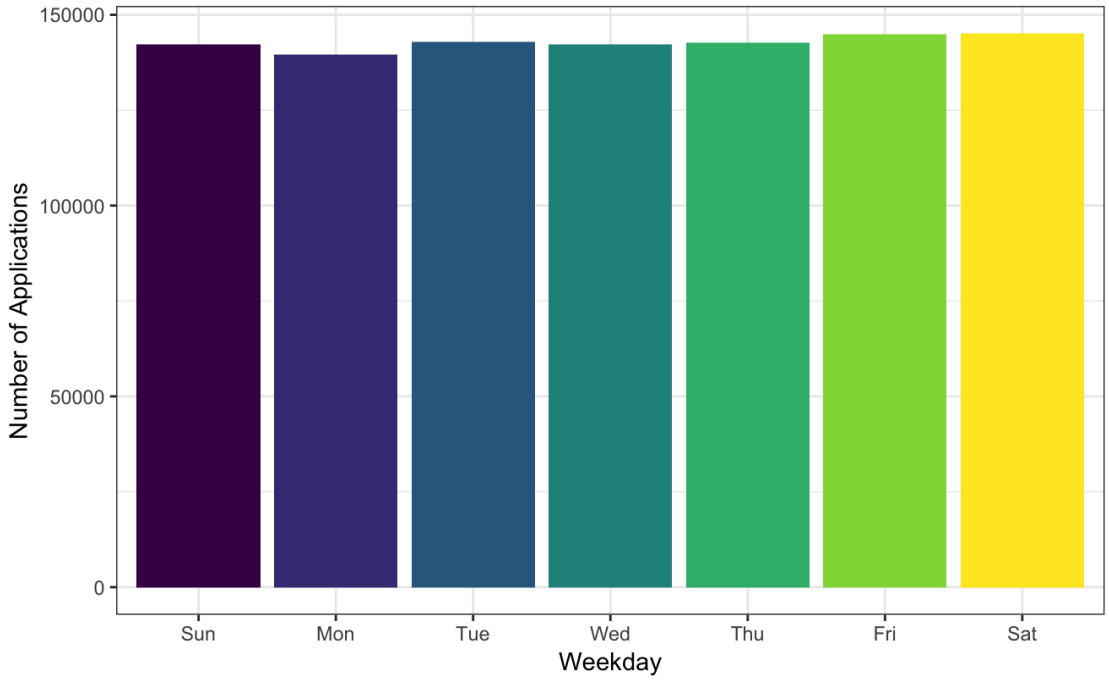
The chart above shows the 15 dates with the highest amount of applications.

Distribution of application of month:



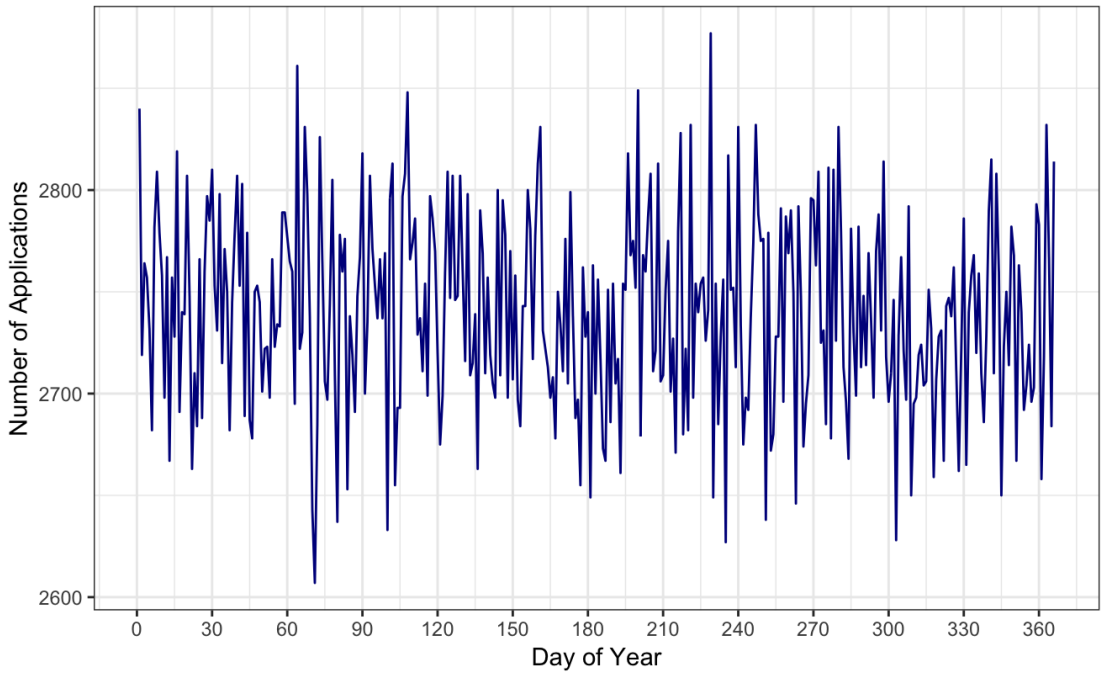
From the chart above, we could not see any seasonilty of month of the applications.

Distribution of application of weekday:



From the chart above, we could not see great difference in number of applications among different weekdays.

Distribution of application of day of year:



From the chart above, we could know that the application also does not have weekly periodicity.

Field3: ssn

Type: Categorical

int [1:1000000] 379070012 387482503 200332444 747451317 24065868 922264214 415812149 373752050 769970791 732119085 ...

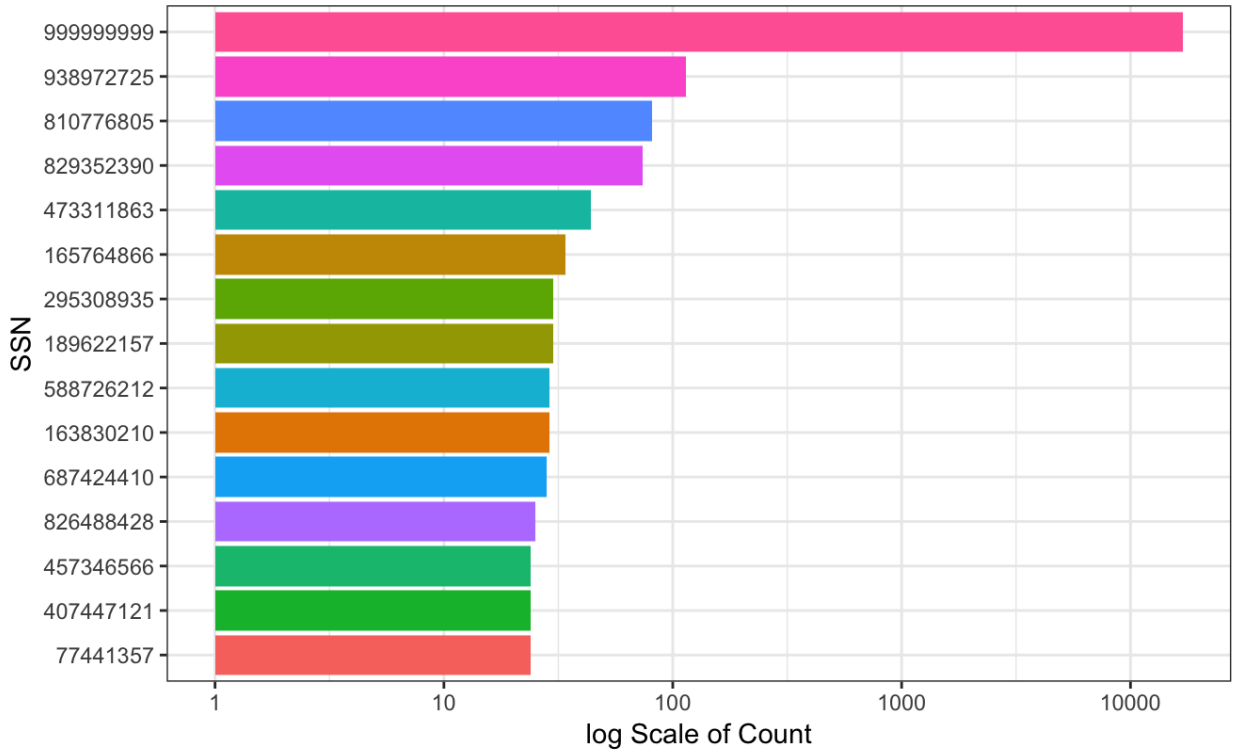
SSN(Social Sercuity Number) is a specific number for everyone, so ssn is categorical variable.

Number of values: 1000000

Number of unique values: 835819

%populated: 100

Number of NAs: 0



From the chart we could know that SSN 999999999 has the highest amount, but they are obviously made up by hand, so some methods are needed to be done to change those 999999999 ssn.

Field4: firstname

Type: Categorical

Factor w/ 78136 levels "AAAEEETR","AAAEJAEM",..: 69302 14459 47838 38352 42762 64810 75739 4423 33436 6762 ...

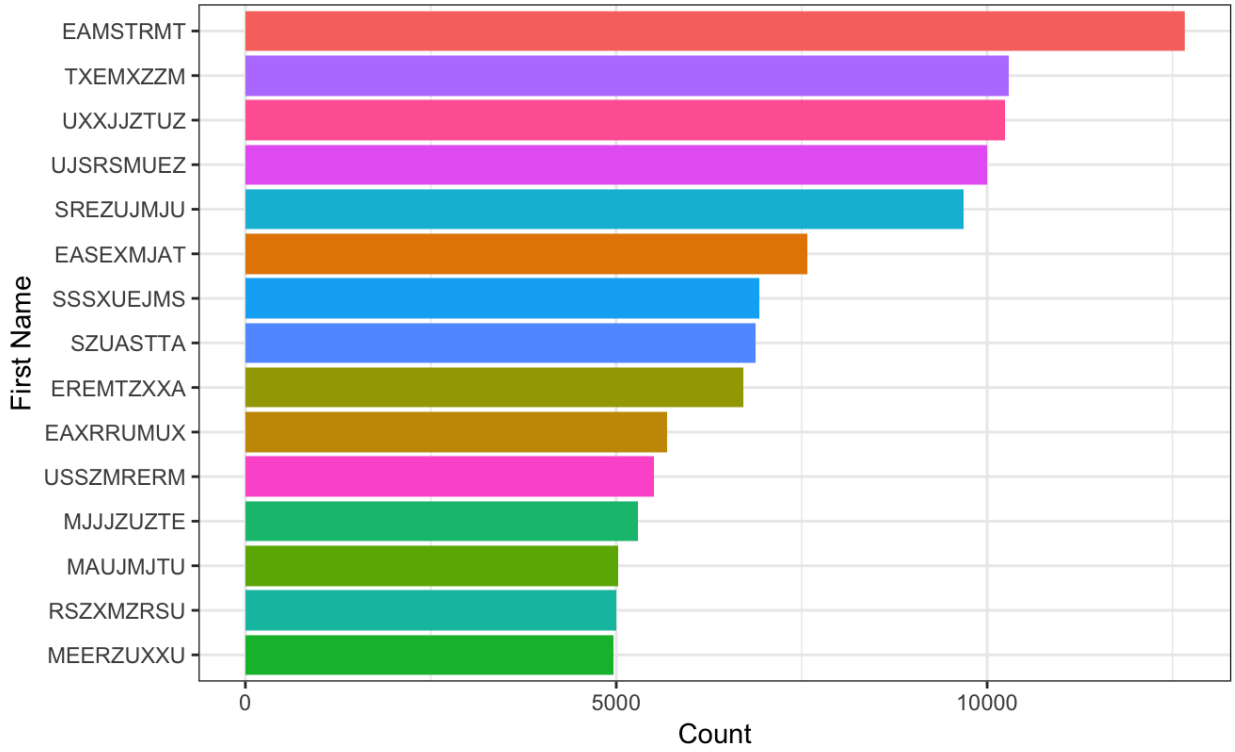
Firstname is the first name of applicant, so it should be categorical.

Number of values: 1000000

Number of unique values: 78136

%populated: 100

Number of NAs: 0



From the chart we could know that the first name appears most in all applications is EAMSTRMT.

Field5: lastname

Type: Categorical

Factor w/ 177001 levels "AAAAARA","AAAAXZR",..: 99191 73286 33683 28863 109224 20833 112223 1478 70261 36871 ...

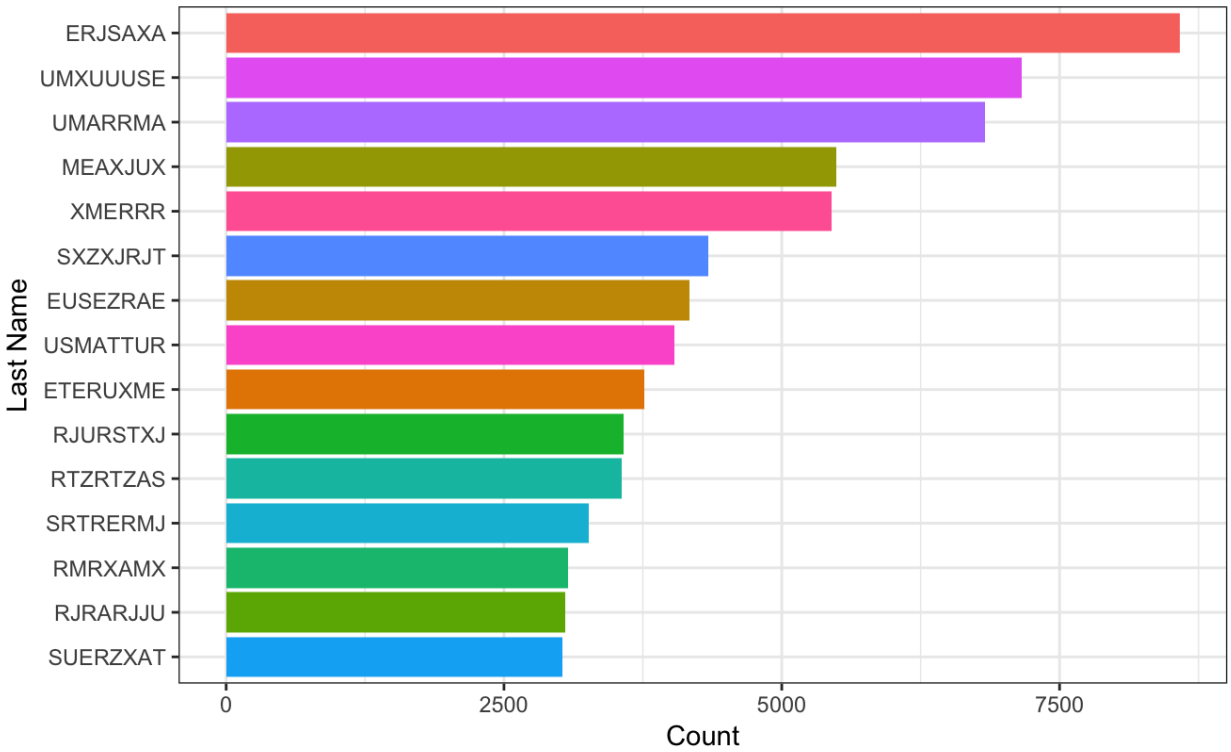
Firstname is the last name of applicant, so it should be categorical.

Number of values: 1000000

Number of unique values: 177001

%populated: 100

Number of NAs: 0



From the chart we could know that the last name appears most in all applications is ERJSAXA.

Field6: address

Type: Categorical

Factor w/ 828774 levels "0 EAJA DR","0 EAJMZ ST",..: 539937 578643 421781 35705 164584 305690 347827 728287 669434 752457 ...

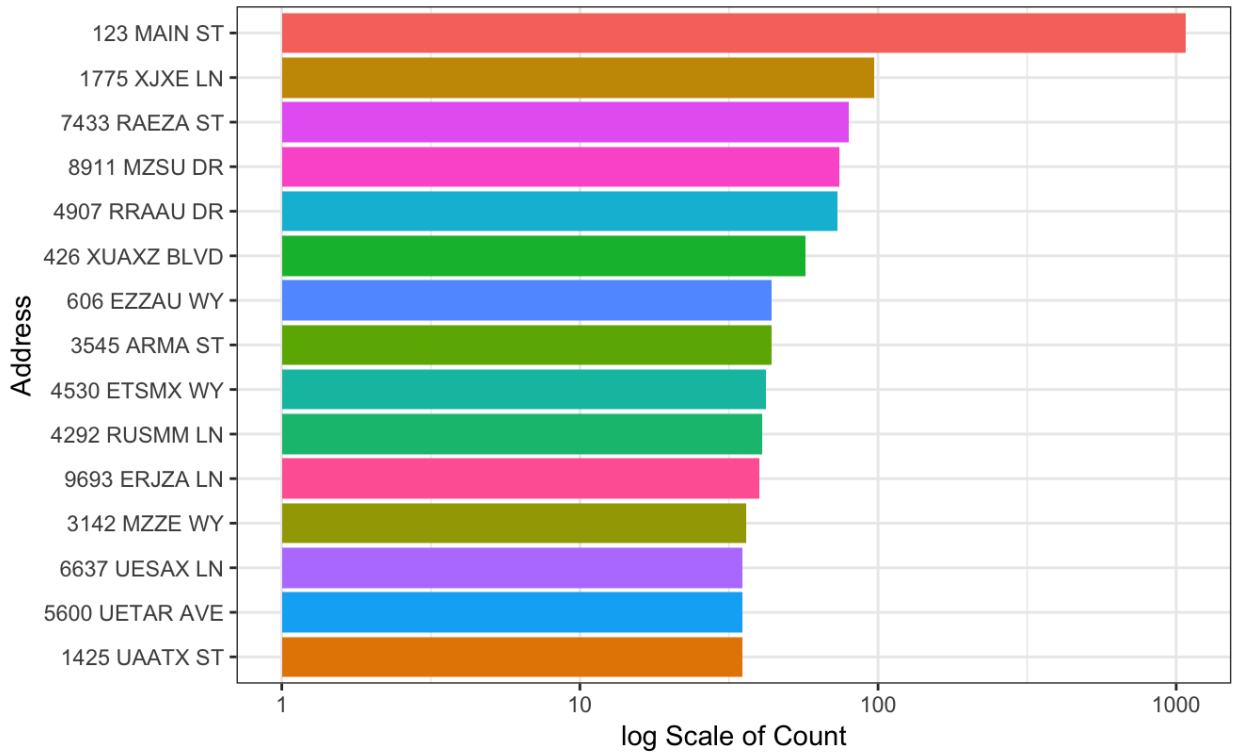
Address is the applicant's address, so it is categorical.

Number of values: 1000000

Number of unique values: 828774

%populated: 100

Number of NAs: 0



From the chart above, we could see that the address “123 MAIN ST” has the highest amount, but it is obviously made up by hand, so we need to clean the address.

Field7: zip5

Type: Categorical

int [1:1000000] 2765 57169 56721 35286 3173 8391 41640 60567 37934 93751 ...

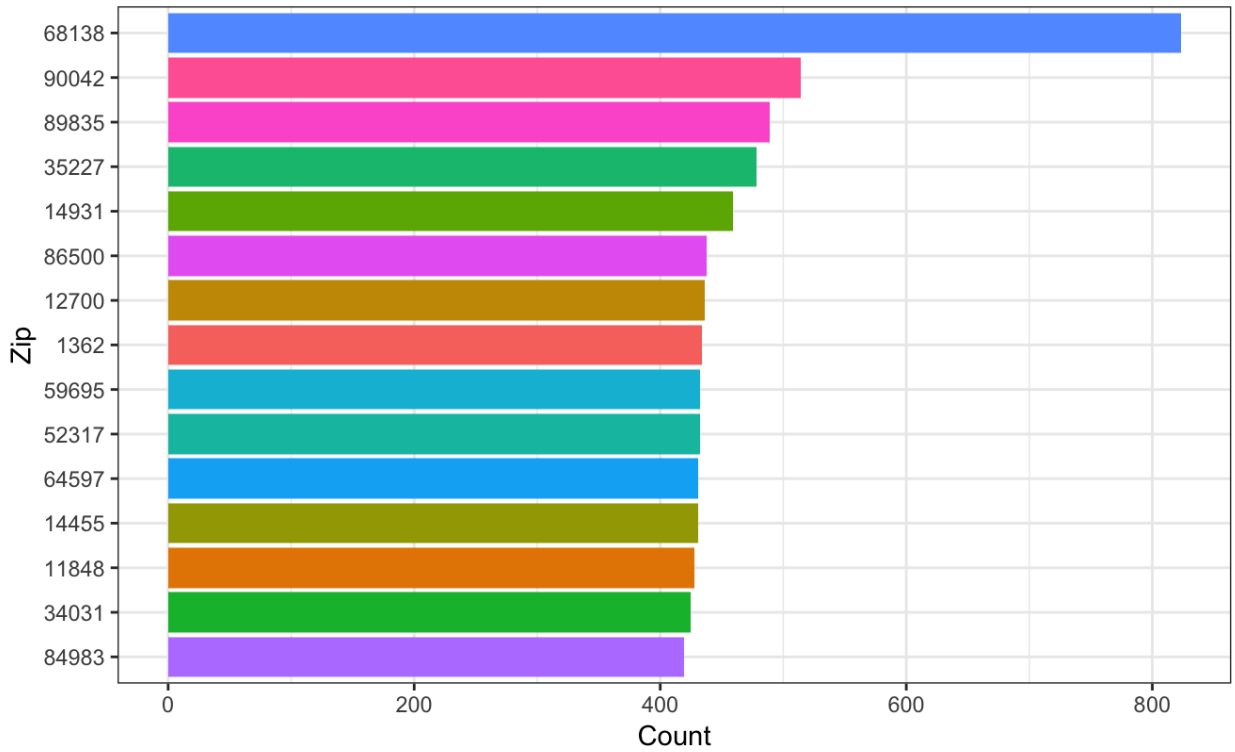
Merch zip5 is the zip code of address of applications, so zip5 is a categorical variable.

Number of values: 1000000

Number of unique values: 26370

%populated: 100

Number of NAs: 0



From the chart we could know that palce of zip code 68138 has highest number of applications.

Field8: dob

Type: Categorical

int [1:1000000] 19070626 19340615 19070626 19440430 19980315 19480613 19640318 19190528 19900314 19750127 ...

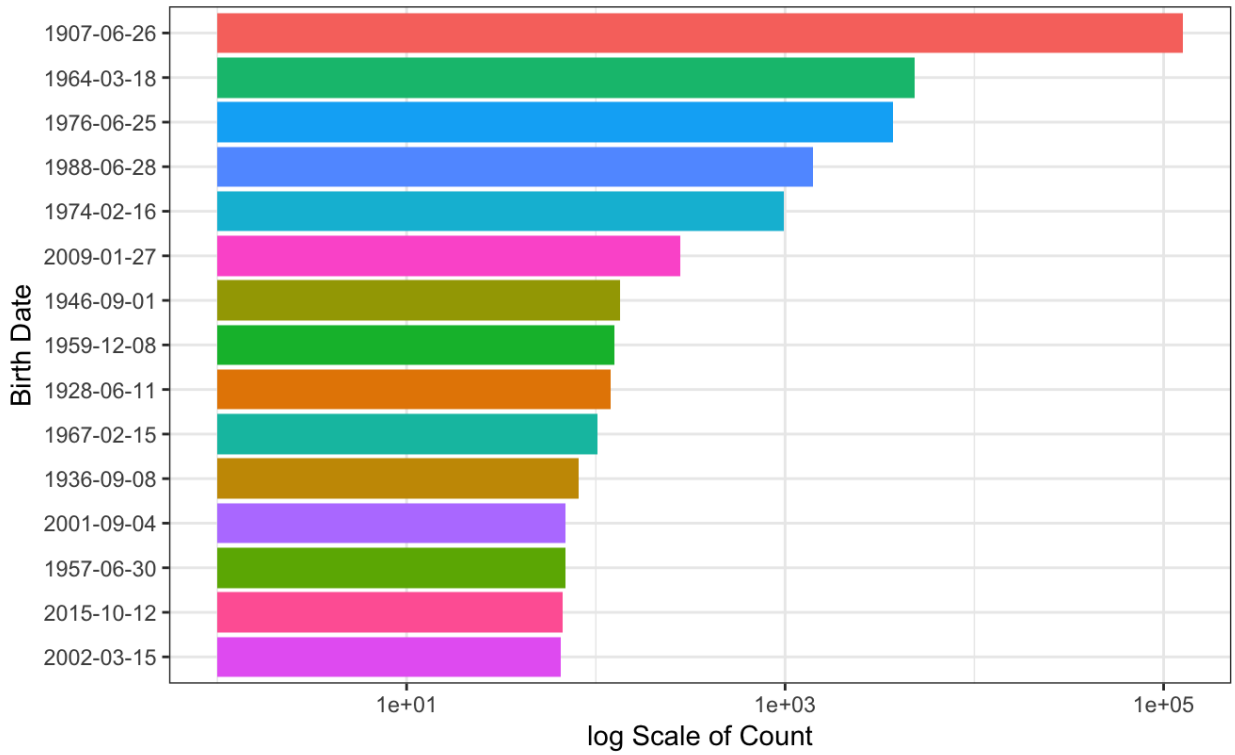
Dob is the birthdate of the applicant, so it is categorical variable.

Number of values: 1000000

Number of unique values: 42673

%populated: 100

Number of NAs: 0



From the chart above, we could see that the bithdate "1907-06-26" takes the most proportion, but this date is too far ago and obviously made up by hand.

Field9: homephone

Type: Categorical

num [1:1000000] 1.80e+09 4.16e+09 2.17e+08 1.32e+08 6.10e+09 ...

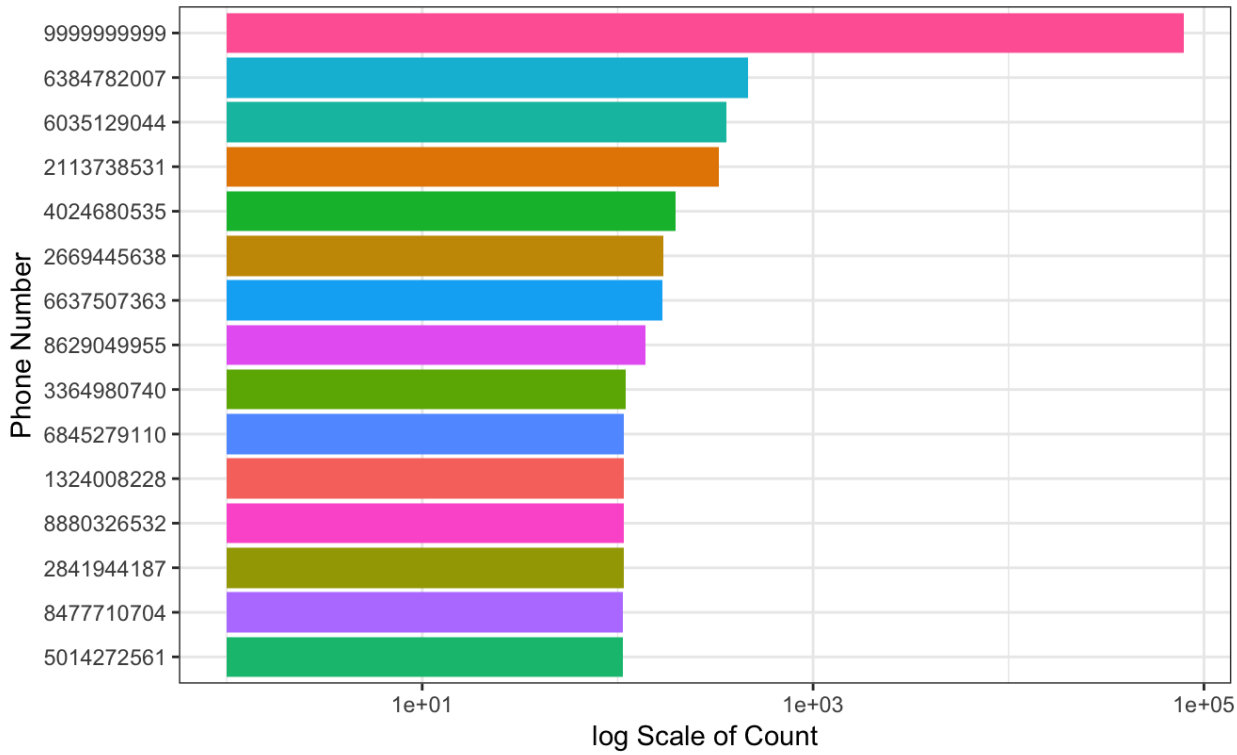
Homephone is the phone number of the application, so it is a categorical variable.

Number of values: 1000000

Number of unique values: 28244

%populated: 100

Number of NAs: 0



From the chart above, we could see that the phone number "9999999999" takes the most proportion, but the this phone number is obviously made up by hand.

Field10: fraud_label

Type: Categorical

int [1:1000000] 0 1 0 0 0 0 0 0 0 0 ...

Fraud is the mark of wheter a record is fraud, and 1 represent that the record is fraud, whereas 0 represents the record is not fraud.

Number of values: 1000000

Number of unique values: 2

%populated: 100

Number of NAs: 0

The number of 0 and 1 is as follows:

x   freq
1 0 985607
2 1 14393

We could see that there are 14393 fraud records.