DSO 562: Fraud Analytics

Supervised Fraud Model On Applications Data

Team 4

Jasleen Kaur Ahuja

Aishwarya Joshi

Dhwani Kapadia

Mahesh Pandit

Shubham Rishishwar

Shashank Ravi Shankar

Shringar Sharan

TABLE OF CONTENTS

1.0 EXECUTIVE SUMMARY

1.1 BUSINESS PROBLEM

According to Statista, by September 2017, 22% of internet users in the United States had been victims of identity theft. This causes the businesses loss of millions of dollars, and also puts customers at risk.

1.2 OBJECTIVE

The goal of this project is to identify fraud in applications data. The data provided to us is for applications received for the year 2016.

1.3 PROJECT OVERVIEW

The following summarizes the steps entailed to analyze the data and build the fraud detection model:

1. **Data Cleaning:** The data has 1 million records and 10 fields (including record number) between Jan-Dec 2016. While there were no missing values, there were several frivolous values for SSN, Address and HomePhone which had to be replaced
2. **Building Expert Variables:** We created around 300 variables by combining different entities together and using velocity/number of occurrences
3. **Feature Selection:** We employed the following method to select the best features for our models:
    - We used KS and FDR as filter to reduce the number of variables we used in the fraud detection algorithms.
    - We also employed a wrapper method such as recursive feature elimination (RFE) using logistic regression to reduce the number of features and retain only those which are important.
    - We then shortlisted variables ranked 1 from the wrapper method and ran Gradient Boosting Classifier to shortlist the top 25 features.
4. **Supervised Learning Algorithms for Fraud Detection:** We then built several fraud detection algorithms such as Logistic Regression, Random Forest, Gradient Boosting Trees, Decision Trees and Neural Net and to classify records as fraudulent and non-fraudulent. Used the 3% FDR to determine the efficacy of each technique. We then chose the best model out of all models that we built and displayed its results.

In conclusion, we identified the top best fraud detection algorithm for predicting the fraudulent records in the Applications dataset using the above process. We also provide some recommendations for future work and how we can improve.

2.0 DATA DESCRIPTION

The data has been generated using an algorithm to resemble application data for a certain product. Following are some of the characteristics of this dataset:

- Dataset Name: Applications Data
- Number of Records/Rows: 1000000
- Number of Fields/Columns: 10
- There are no missing values in the dataset.
- All the fields in the dataset are categorical.
- Each record represents a customer application in the dataset and the respective fields have various personal information of the customers such as Name ('firstname', 'lastname'), Social Security Number ('ssn'), Date of Birth ('dob'), Address ('address') etc. The fraud label field indicates whether the record represents a fraudulent application or not. Here, 1 corresponds to a fraudulent activity.

2.1 SUMMARY OF CATEGORICAL FIELDS

This data contains 10 categorical fields. Amongst these fields, 2 are date fields which were converted to datetime later. The first field is "record", which is a unique identifier. The table below shows the characteristics of the date fields.

Field Name	# Records Populated	% Populated	# Unique Values	# Records with Value "Zero"	Minimum	Maximum	Most Common Value
date	1000000	100	365	0	2016-01-01	2016-12-31	2016-08-16
dob	1000000	100	42673	0	1900-01-01	2016-10-31	1907-06-26

The following table shows the characteristics of the rest of the categorical fields.

Field Name	# Records Populated	% Populated	# Unique Values	# Records with value "Zero"	Most Common Value
record	1000000	100	1000000	0	N/A
ssn	1000000	100	835819	0	999999999
firstname	1000000	100	78136	0	EAMSTRMT
lastname	1000000	100	177001	0	ERJSAXA
address	1000000	100	828774	0	123 MAIN ST
zip5	1000000	100	26370	0	68138
homephone	1000000	100	28244	0	9999999999
fraud_label	1000000	100	2	985607	0

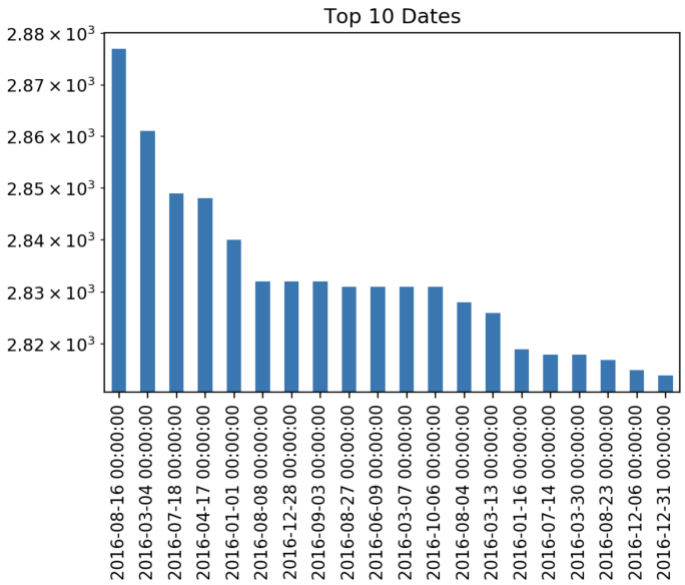
2.2 DESCRIPTION OF PRIMARY FIELDS

Now, we will explore in detail, the characteristics of a few important fields by plotting various graphs based on the category of the field.

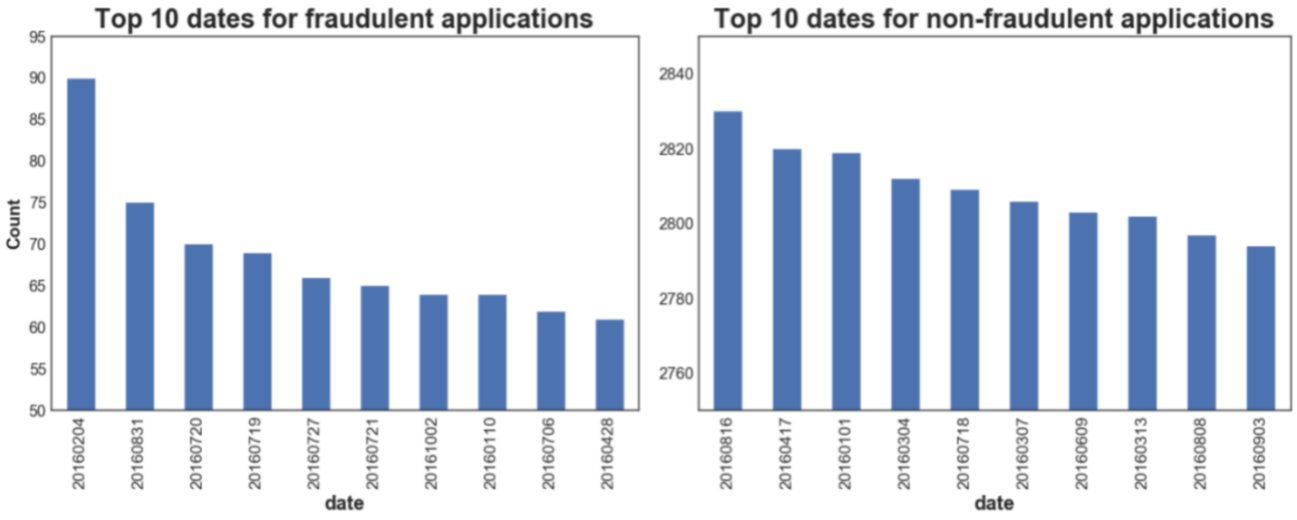
2.2.1 date (dtype: int64)

Description: date is a categorical variable representing the date of each transaction. This field has 365 unique values and no missing value. The distribution is shown below, top 10 categories are listed below:

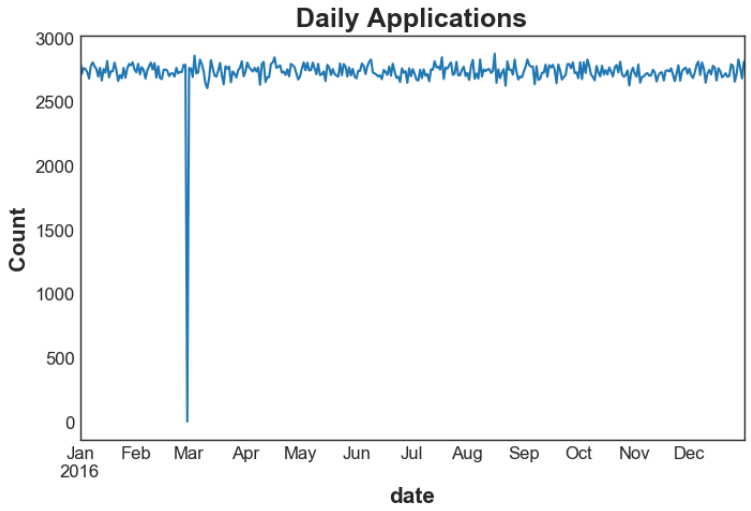
Date	Count
20160816	2877
20160304	2861
20160718	2849
20160417	2848
20160101	2840
20161228	2832
20160903	2832
20160808	2832
20160827	2831
20160609	2831
20160307	2831
20161006	2831
20160804	2828
20160313	2826
20160116	2819



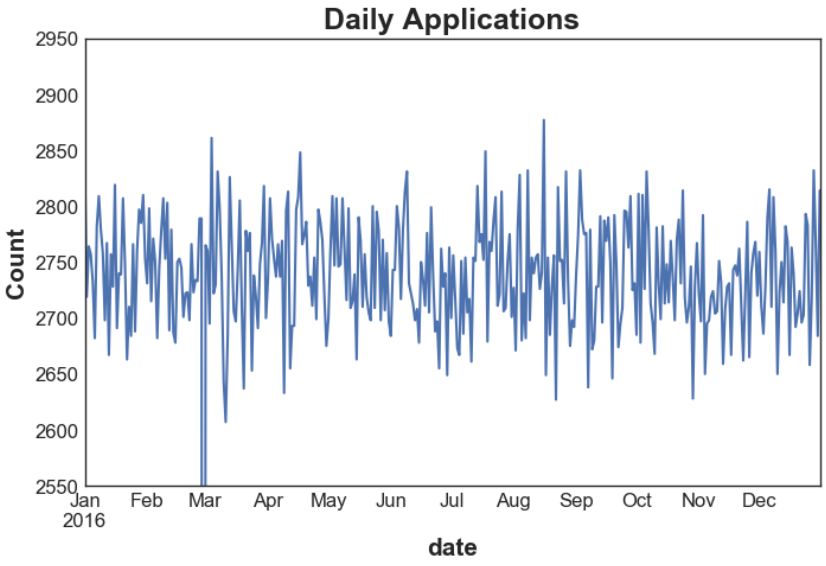
The graphs below show the top 10 dates for fraudulent (fraud_label = 1) as well as non-fraudulent (fraud_label = 0) applications according to the number of applications.



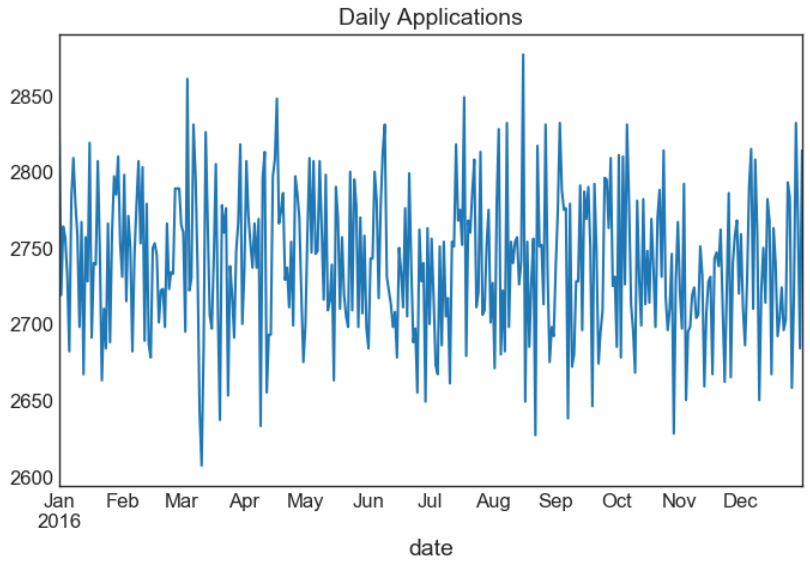
The graph below also shows the number of daily applications. From the graph, the dip in daily application occurs on February 29, 2016 as there are no applications corresponding to this date in the dataset.



We can see from the graph above that the number of applications varies only within a moderate range, except for value corresponding to February 29. Hence, to get a better picture of the distribution of the 'date' field, we set the range of y-axis from 2550 to 2950 and plot the graph as shown below. Here we know the dip is due to no value for February 29. Except that, no other values are out of range.



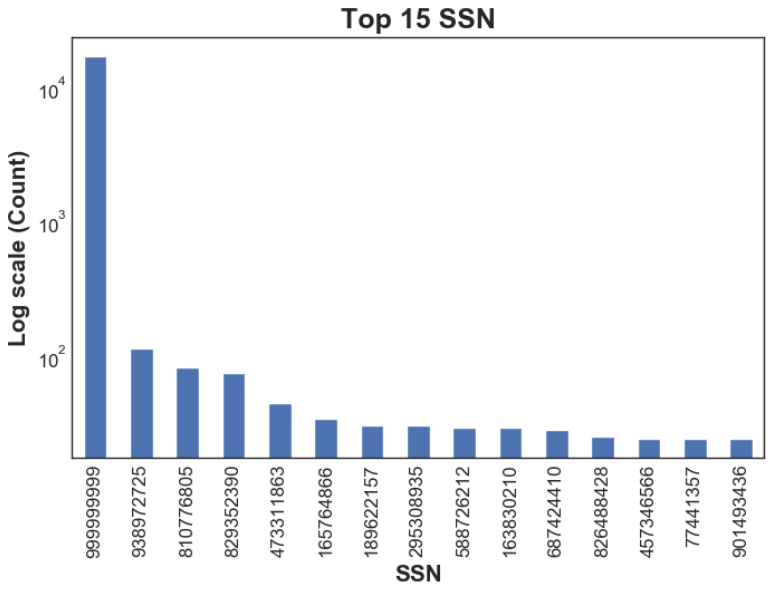
We can also impute an innocuous value such as the value for the previous day. The graph below shows the daily applications after imputation of the count of Feb 29 with the count of daily applications of Feb 28.



2.2.2 ssn (dtype: int64)

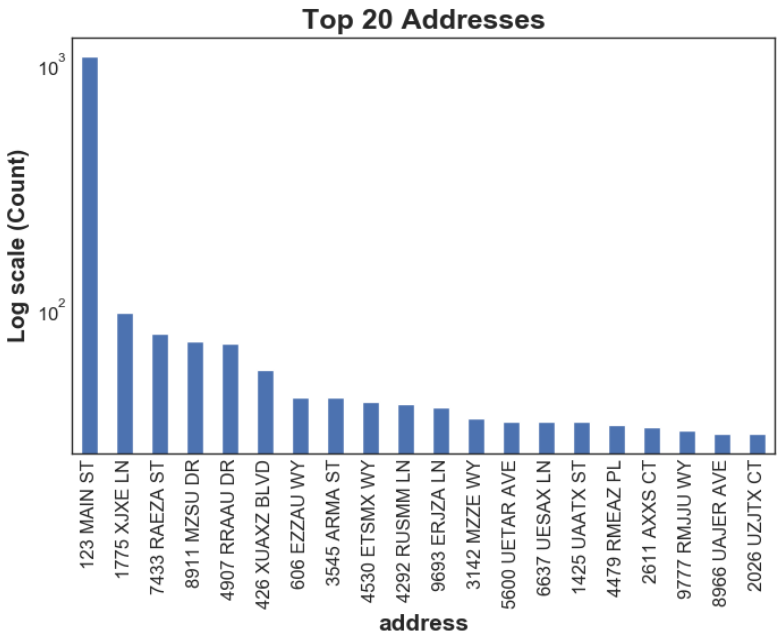
Description: This field contains the Social Security Number of the applicants and has only 835819 unique values. We can see from the table and bar graph that some SSNs are repeated many times. We take special note of the SSN 999999999, which is repeated 16935 times in the data.

SSN	Count
999999999	16935
938972725	114
810776805	81
829352390	74
473311863	44
165764866	34
189622157	30
295308935	30
588726212	29
163830210	29
687424410	28
826488428	25
457346566	24
77441357	24
901493436	24



2.2.3 address (dtype: object)

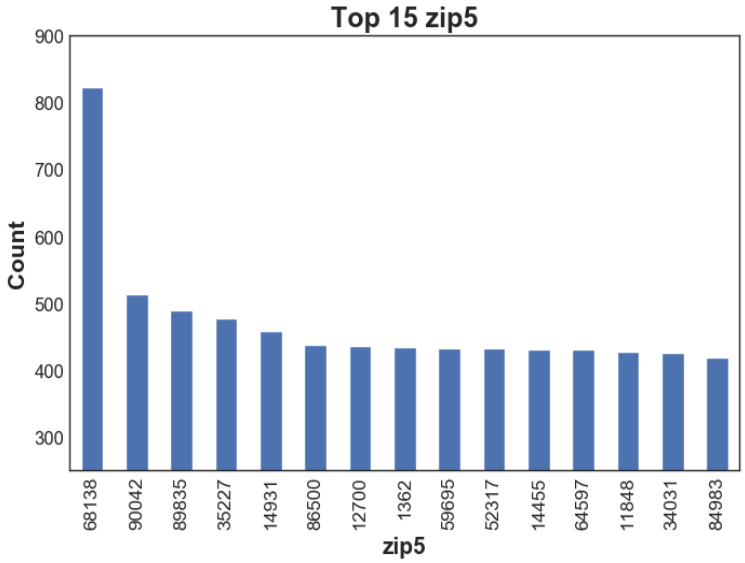
Description: This field contains the street addresses of the applicants and is categorical. The field has 828774 unique values. Moreover, there are no missing values in the field. The bar graph shows the top 20 values of 'address' by count (log scale).



address	Count
123 MAIN ST	1079
1775 XJXE LN	97
7433 RAEZA ST	80
8911 MZSU DR	74
4907 RRAAU DR	73
426 XUAXZ BLVD	57
606 EZZAU WY	44
3545 ARMA ST	44
4530 ETSMX WY	42
4292 RUSMM LN	41
9693 ERJZA LN	40
3142 MZZE WY	36
5600 UETAR AVE	35
6637 UESAX LN	35
1425 UAATX ST	35
4479 RMEAZ PL	34
2611 AXXS CT	33
9777 RMJJU WY	32
2026 UZJTX CT	31
8966 UAJER AVE	31

2.2.4 zip5 (dtype: int64)

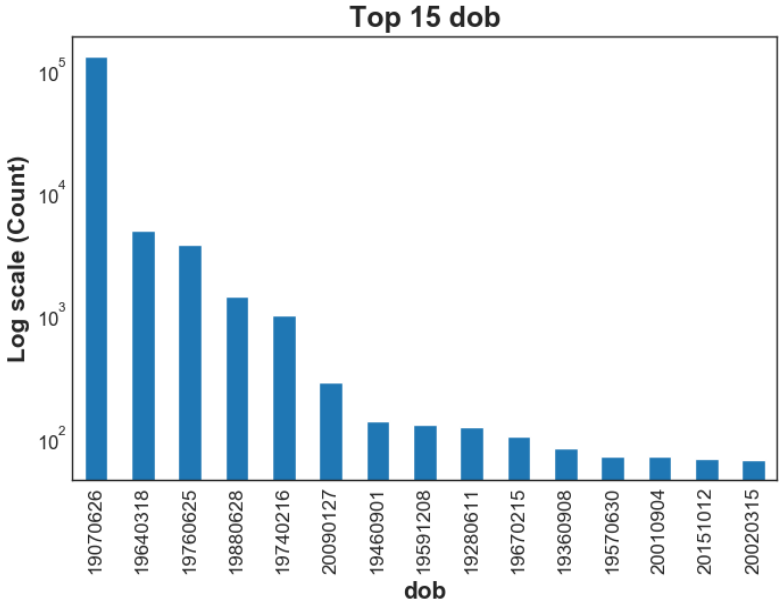
Description: This is a categorical field contains the 5-digit zip codes of the addresses of the applicants and denotes where the application comes from. This field is quite useful for fraud detection and give insights about the location of most fraudulent activities. It has 26370 unique values in the dataset. The bar graph shows the top 15 values of 'zip5' by count.



Zip5	Count
68138	823
90042	514
89835	489
35227	478
14931	459
86500	438
12700	436
1362	434
59695	432
52317	432
14455	431
64597	431
11848	428
34031	425
84983	419

2.2.5 dob

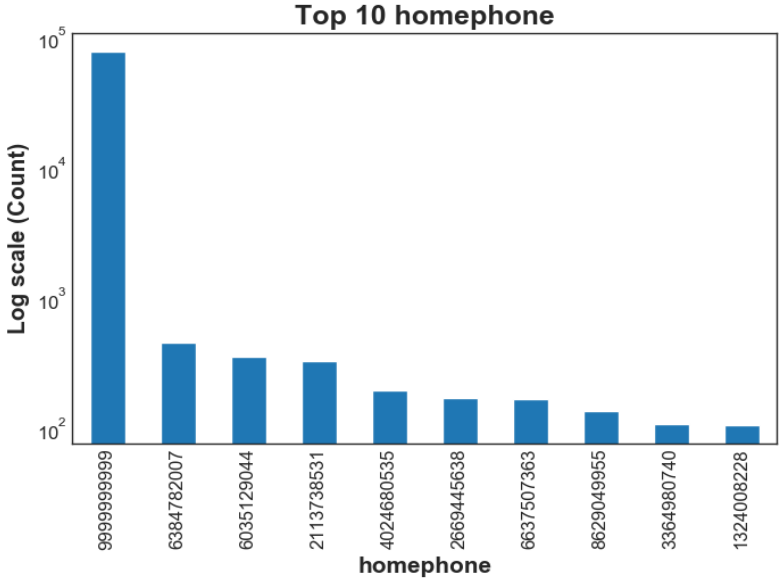
Description: This field contains the Date of Birth of respective applicants and is a categorical field. It has 42673 unique values and no missing values in the dataset. We note that the value 19070626 appears 126568 times and 19640318 appears 4818 in the dataset. The bar graph shows the top 15 values of 'dob' by count (log scale).



dob	Count
19070626	126568
19640318	4818
19760625	3723
19880628	1404
19740216	980
20090127	280
19460901	135
19591208	126
19280611	120
19670215	102
19360908	81
19570630	69
20010904	69
20151012	67
20020315	65

2.2.6 homephone (dtype: int64)

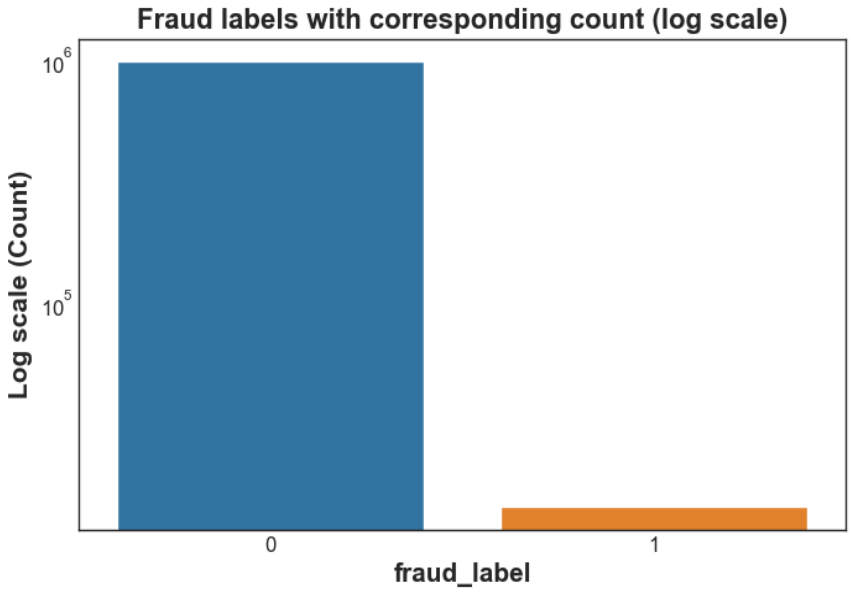
Description: This field contains the phone numbers of the applicants and is a categorical field. It has 28244 unique values and no missing values. We note that the number 9999999999 appears 78512 in the dataset. The bar graph shows the top 10 values of 'homephone' by count (log scale).



homephone	Count
9999999999	78512
6384782007	466
6035129044	360
2113738531	331
4024680535	198
2669445638	172
6637507363	169
8629049955	139
3364980740	110
1324008228	108

2.2.7 fraud_label (dtype: int64)

Description: This field contains the label which indicates whether the particular record is identified as fraud or not. It has two unique values, 0 which indicates the particular record is not fraud whereas 1 indicates that the record is fraudulent. There are 985607 zeros and 14393 ones. This shows that around 1.44% of the dataset consists of fraudulent applications. Below is the log distribution of these two categories.



fraud_label	Count
0	985607
1	14393

2.3 DATA CLEANING

The fields ssn, homephone, address, and dob (Date of Birth) had some abnormal values, or frivolous values. These values create unnecessary noise in the data and can mislead the model and lead to inaccurate results. They must be replaced with a random number. In our case, we have replaced these frivolous values by the corresponding record number*(-1). Following are the details of the values to be replaced:

Field	Frivolous value replaced
ssn	9999999
homephone	9999999999
address	123 MAIN ST
dob	19070626

Further, we converted the datatype of the fields ‘dob’ and ‘date’ from integer to datetime.

3.0 CANDIDATE VARIABLES

Before separating the dataset into train-test and OOT, we created 286 new fields to find true unique identifiers for these applications, by combining existing fields in the dataset. For example,

- The combination of ‘firstname’, ‘lastname’, and ‘dob’ can be a really good unique identifier of a person as compared to using these entities individually.
- Also, there can be several similar addresses, but they can be located at completely different locations, so it’s important to attach the Zip code with an address value to make it a unique address identifier.

These new variables are as follows:

Created Field	Combination Fields
name	firstname, lastname
full_address	address, zip5
name-dob	firstname, lastname, dob
name-full_address	firstname, lastname, address, zip
name-homephone	firstname, lastname, homephone
dob-full_address	dob, address, zip5
dob-homephone	dob, homephone
full_address-homephone	address, zip5, homephone
name-dob-full_address	firstname, lastname, dob, address, zip5
name-dob-homephone	firstname, lastname, dob, homephone
name-full_address-homephone	firstname, lastname, address, zip5, homephone
dob-full_address-homephone	dob, address, zip5, homephone
name-dob-full_address-homephone	firstname, lastname, dob, address, zip5, homephone
ssn-firstname	ssn, firstname
ssn-lastname	ssn, lastname
ssn-address	ssn, address
ssn-zip5	ssn, zip5
ssn-dob	ssn, dob
ssn-homephone	ssn, homephone
ssn-name	ssn, firstname, lastname
ssn-full_address	ssn, address, zip5
ssn-name-dob	ssn, firstname, lastname, dob

Expert variables for each group:

1. **Velocity Variables**
   - Number of records seen over the past n days, where n = 0, 1, 3, 7, 14, 30. These variables tell us how many times an entity or a combination group is seen over past n days.
   - For each entity and combination variable, we created 6 variables (one for each time stamp).
   - For example, 'ssn_lag0_count' means number of applications filed with an SSN on any current date.
   - Hence, since we had 26 fields, we finally created 156 Velocity variables

2. **Days Since Variables**
   - For each entity or combination, this field denotes the number of days since we last saw that element.
   - For example, 'ssn_days_since' means number of days since we last saw that particular SSN.
   - If we see multiple SSNs on the same day, the count for the second occurrence becomes zero, as we had already seen that SSN earlier in the day (the record number denotes the order in which the applications came on a particular day).
   - Since there was only 1 'days_since' variable for each existing field, we created 26 'days since' fields overall.

3. **Relative Velocity Variables**
   - Relative number of records seen over the past i days compared to the past j days, where i = 1 and j = 3, 7, 14, 30. These variables tell us how the velocity variables fluctuate over time and if any day has an unusually high number of applications (compared to the average of the past j days)
   - data[numerator] / (data[denominator]/j)
     where,
     numerator = velocity variables for n=1
     denominator = velocity variables for n = 3, 7, 14, 30
     j = 3, 7, 14, 30
   - For each entity and combination variable, we created 4 variables (one for each numerator-denominator combination).
   - For example, 'ssn_lag1_lag3_avg' means average number of applications filed with an SSN on any current date as compared to the daily average of the past 3 days.
   - Hence, since we had 26 fields, we finally created 104 Velocity variables

4.0 FEATURE SELECTION

4.1 Filtering using KS score and Fraud Detection Rate (FDR)

Before using KS and FDR, we z-scaled our candidate variables. For each of our candidate variable, we calculated Kolmogorov–Smirnov (KS) score and fraud detection rate individually. Both the KS score and the FDR rate helped us determine how well candidate variables individually predict fraud, allowing us to rank order the variables in terms of usefulness for our models.

We separated the data after Oct 31, 2016 as the Out of Time Data, whereas the records up to Oct 31, 2016 were taken for feature selection and for further model building.

The KS score is a filter method that helps determine how well a candidate variable separates the 'goods' from the 'bads'. In this case, the frauds and the not frauds. For each variable, we used the formula below to calculate a KS score and rank order the variables by the score.

$$KS = \max_x \int_{x_{min}}^{x} [P_{\text{goods}} - P_{\text{bads}}]\, dx$$

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

The FDR for each variable was determined at a 3% level. It is the value representing the percentage of all frauds caught at an examination cutoff. For each variable, we determined what percentage of frauds are captured by the top 3% of the variable and ranked order as such.

After getting KS and FDR table for all variables, we followed the below steps to select around 100 of the top variables:

1. Sorted records by KS in the descending order
2. Replaced KS with the sorted rank order (Highest score getting rank 1 and so on)
3. Sorted records by FDR in the descending order
4. Replaced FDR with the sorted rank order (Highest score getting rank 1 and so on)
5. KS and FDR were on the same scale after step 4. Next, we took their average to get a cumulative score, which is our final importance rank of variables

4.2 Wrapper Method

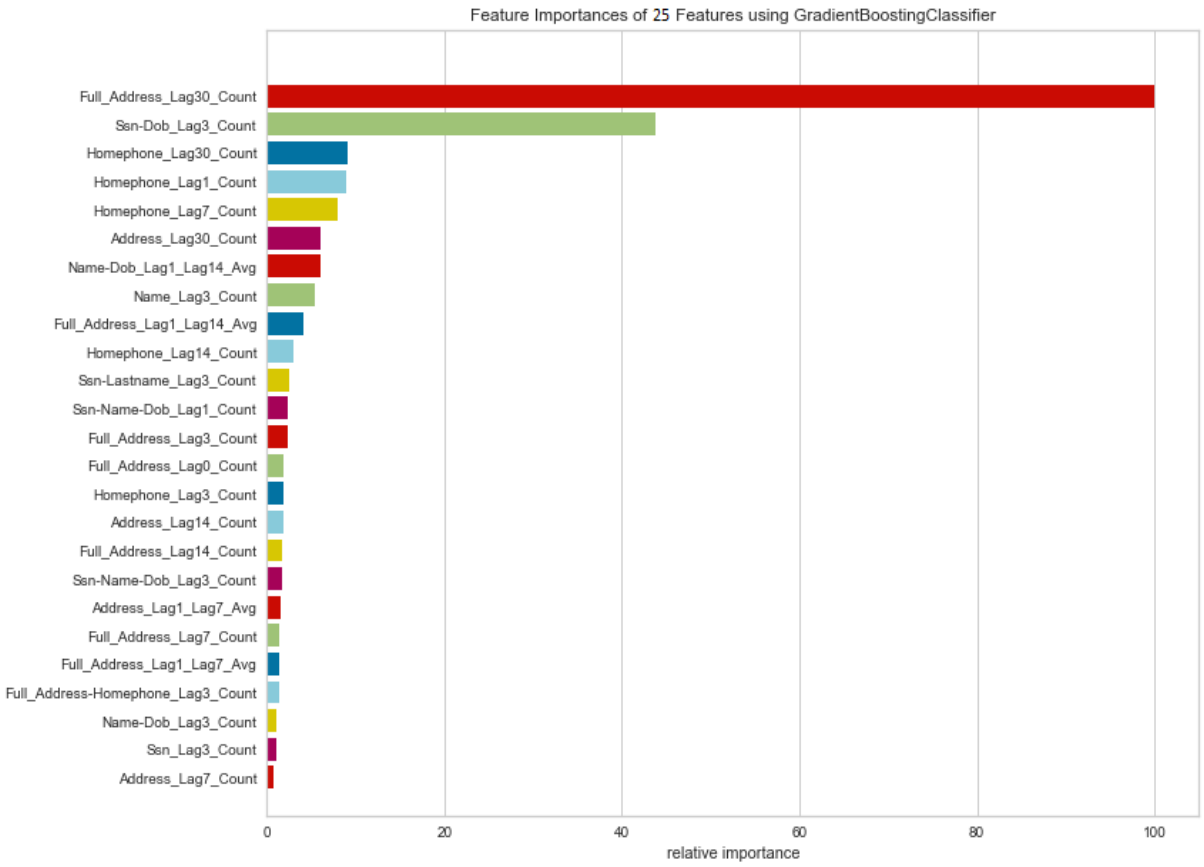
A wrapper method was implemented to determine the optimum candidate variables that should be used to fit a model. It uses a logic similar to a stepwise approach to determine which variables are valuable for our machine learning model for prediction.

To implement this, we used Recursive Feature Elimination with logistic regression and used the ranking of the features to shortlist them. The highest ranking corresponds to the best variables. Around 55 of these variables had the same rank 1.

4.3 Gradient Boosting Classifier

To shortlist the variables further, we used Gradient Boosting Classifier. We finally shortlist the top 25 candidate variables from the 55 because the rest have very small feature importance as shown below.

The horizontal bar plot below shows the relative feature importance of the top 25 features when we used the Gradient Boosting Classifier for feature selection.



5.0 ALGORITHMS

We attempted to try five models: Logistic Regression, Random Forest, Decision Trees, Neural Net and Support Vector Machine (SVM) while the SVM model was too computationally costly so we run output for other four models.

5.1 LOGISTIC REGRESSION

Logistic regression is used to predict the probability of a dependent binary variable belonging to one of the two classes using the formula given below:

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_p x_p)}}$$

Where b_0 is a constant,

x_1 , x_2, … x_n are independent variables

b_1, b_2, … b_n are the coefficients associated with each of the independent variables.

Logistic regression was run using all 25 independent variables that were identified using the wrapper, and also using certain combinations of these variables. Although we used the wrapper to identify the top 25 variables, we also needed to use a different tool to identify smaller combinations of variables that would perform best. Recursive feature elimination (RFE) was used to find the most effective combination of independent variables to predict the fraud label. RFE recursively eliminates the least important features until the specified number of features that are most important for predicting the dependent variable remains.

Logistic models of sizes ranging from 10 to 25 were created and tested, and the most effective model had 18 independent variables. This model had a fraud detection rate of 50.71% at the top 3% of the holdout sample.

5.2 RANDOM FOREST

Random Forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. Decisions trees are very sensitive to the data they are trained on — small changes to the training set can result in significantly different tree structures. Random forest takes advantage of this by allowing each individual tree to randomly

sample from the dataset with replacement, resulting in different trees. The number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

While building a random forest, at each split in the tree, the algorithm is not allowed to consider most of the available predictors. A main disadvantage with bagging technique is that if predictors are highly correlated, it will lead to split due to strongest predictor at top and hence all bagged trees will have the tendency to look similar. Hence predictions of response variable from bagging will be close to each other and average of these correlated quantities will not lead to enough reduction in variance.

We used the RandomForestClassifier package from the library sklearn to make the Random Forest model on our reduced set of variables. Then we predicted the probability of Fraud over training, test and OOT (validation data). Our model has 58.7% train FDR, 61% test FDR and 52.6% out of time set FDR.

5.3 GRADIENT BOOSTING TREES

Gradient Boosting trains weaker models additively, each model is fitted on a modified version of the original dataset. Initially a tree is trained with equal weight for each observation, then this tree is evaluated and for the second tree the weight for the records that the first model misclassified is increased. This iterative process improves over the previous trees, the subsequent trees help classify records that are not clearly classified by the initial trees. A gradient boosting tree uses a loss function for its training process. The loss function is a measure of the difference between the model's prediction and true values and represents the model's error in fraud prediction. The iterative training process would take the model to the global optima.

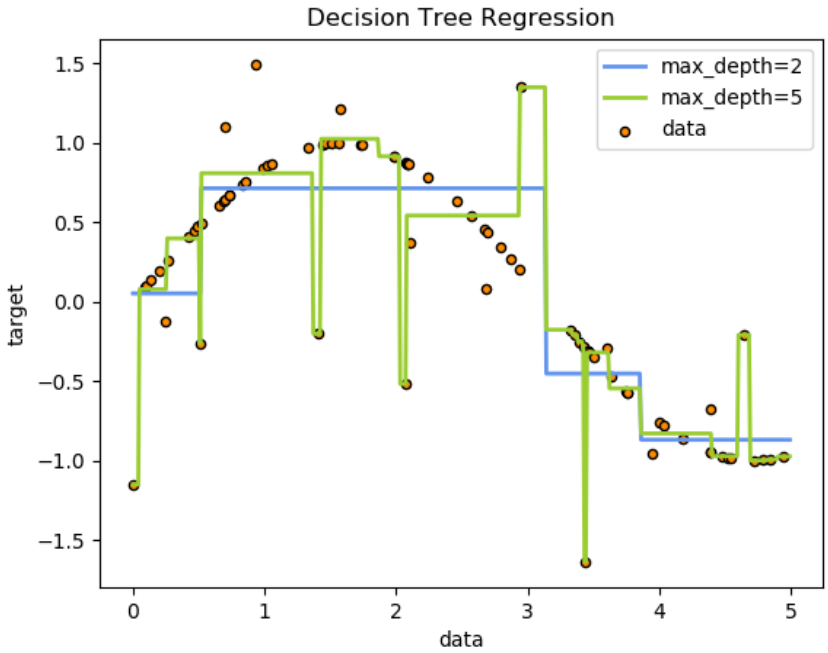
We used the 'Catboost' library for our gradient boosting tree and achieved an out of time test FDR of 53.81% with 25 variables. The model had a depth of 6, had a 1000 tress, L2 regularization of 100, subsample of 0.66 and a learning rate of 0.03.

5.4 DECISION TREES

Decision Trees are a non-parametric supervised learning algorithm used for classification and regression problems. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the features in the dataset. In this project, we have used Decision Tree for classifying fraudulent records in the application data.

A decision tree is a flowchart like tree structure. Each internal node represents a conditional statement on an attribute. Each branch represents the outcome of the conditional statement and, each leaf node represents a class label for a classification problem. The paths from the root node to the leaf node represent classification rules.

In the example below (Source: Scikit Learn Website), decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.



In order to implement decision trees in Python we imported *tree* from Scikit Learn library and achieved an out of time test FDR of 50.21% with 17 variables.

5.5 NEURAL NET

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. 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.

We used different values of nodes and epochs to run different models. Our best model is with 10 nodes and 20 epochs which give 55.26% train FDR, 55.46% test FDR, whereas 53.01% out of time set FDR.

5.6 FDR (TRAIN, TEST and OOT) OF DIFFERENT MODELS

FDR - LOGISTIC REGRESSION				
Model	Train	Test	OOT	# Variables
1	50.21%	49.61%	49.83%	25
2	50.55%	50.51%	50.13%	20
3	50.62%	50.56%	50.71%	19
4	**50.62%**	**50.56%**	**50.71%**	**18**
5	50.43%	50.62%	50.67%	17

FDR - RANDOM FOREST					
Model	# Trees	Train	Test	OOT	# variables
1	100	58.80%	61.10%	52.50%	25
2	**200**	**58.70%**	**61.00%**	**52.60%**	**25**
3	300	58.70%	61.00%	52.50%	25
4	400	58.70%	61.00%	52.50%	25
5	500	58.70%	61.00%	52.50%	25

FDR - GRADIENT BOOSTING							
Model	# Trees	Depth	Learning Rate	TRAIN	TEST	OOT	Variables
1	1000	6	0.03	55.18%	54.92%	52.44%	20
2	**1000**	**6**	**0.03**	**56.18%**	**55.76%**	**53.81%**	**25**
3	500	4	0.3	54.98%	55.06%	52.15%	20
4	500	4	0.05	55.03%	55.09%	52.44%	20

FDR - DECISION TREES				
Model	Train	Test	OOT	# Variables
1	58.99%	48.82%	47.69%	25
2	56.15%	52.00%	49.62%	19
3	56.11%	51.88%	49.87%	18
4	**55.67%**	**52.39%**	**50.21%**	**17**

FDR - NEURAL NET							
Model	Layer	Nodes	Epoch	TRAIN	TEST	OOT	# Variables
1	**1**	**10**	**20**	**55.26%**	**55.46%**	**53.01%**	**25**
2	1	30	50	55.98%	55.76%	52.44%	25
3	1	40	50	55.84%	55.46%	52.15%	25
4	1	10	20	55.67%	55.60%	52.72%	25
5	1	30	50	55.99%	55.88%	52.15%	25
6	1	10	50	55.31%	55.17%	52.44%	25

6.0 RESULTS

Our best performing algorithm is Gradient Boosting Trees model and we have generated cumulative Good, Bads, % Good, % Bad (FDR), KS and FPR for all three populations (training, testing, and Validation (OOT), and the fraud savings plot.

6.1 TRAINING DATA

- **Total no. of records in training set: 556,497**
- **Total no. of original Bads: 7,930**

Training	# Records	# Goods	# Bads	Fraud Rate								
	556497	548567	7930	0.014455846								
Bin Statistics						Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads	KS	FPR
1	5564	1267	4297	22.77138749	77.22861	5564	1267	4297	0.230965	55.9573	55.72633	0.294857
2	5564	5362	202	96.36951833	3.630482	11128	6629	4499	1.208421	56.0309	54.82248	1.473439
3	5564	5503	61	98.90366643	1.096334	16692	12132	4560	2.21158	56.18372	53.97214	2.660526
4	5564	5521	43	99.22717469	0.772825	22256	17653	4603	3.218021	58.0454	54.82738	3.835108
5	5564	5504	60	98.92163911	1.078361	27820	23157	4663	4.221362	58.80202	54.58066	4.966116
6	5564	5498	66	98.81380302	1.186197	33384	28655	4729	5.22361	59.6343	54.41069	6.059421
7	5564	5510	54	99.0294752	0.970525	38948	34165	4783	6.228045	60.31526	54.08721	7.143006
8	5564	5531	33	99.40690151	0.593098	44512	39696	4816	7.236308	60.7314	53.49509	8.242525
9	5564	5522	42	99.24514738	0.754853	50076	45218	4858	8.242931	61.26103	53.0181	9.307946
10	5564	5504	60	98.92163911	1.078361	55640	50722	4918	9.246273	62.01765	52.77138	10.31354
11	5564	5531	33	99.40690151	0.593098	61204	56253	4951	10.25454	62.4338	52.17926	11.36195
12	5564	5513	51	99.08339324	0.916607	66768	61766	5002	11.25952	63.07692	51.81741	12.34826
13	5564	5521	43	99.22717469	0.772825	72332	67287	5045	12.26596	63.61917	51.35321	13.33736
14	5564	5525	39	99.29906542	0.700935	77896	72812	5084	13.27313	64.11097	50.83784	14.32179
15	5564	5517	47	99.15528397	0.844716	83460	78329	5131	14.27884	64.70366	50.42482	15.26584
16	5564	5522	42	99.24514738	0.754853	89024	83851	5173	15.28546	65.23329	49.94783	16.20936
17	5564	5528	36	99.35298347	0.647017	94588	89379	5209	16.29318	65.68726	49.39409	17.15857
18	5564	5523	41	99.26312006	0.73688	100152	94902	5250	17.29998	66.20429	48.9043	18.07657
19	5564	5520	44	99.20920201	0.790798	105716	100422	5294	18.30624	66.75914	48.4529	18.96902
20	5564	5521	43	99.22717469	0.772825	111280	105943	5337	19.31268	67.30139	47.98871	19.85067

6.2 TEST DATA

- **Total no. of records in test set: 238,499**
- **Total no. of original Bads: 3,556**

Testing	# Records	# Goods	# Bads	Fraud Rate								
	238499	234943	3556	0.015135586								
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads	KS	FPR
1	2384	467	1917	19.58892617	80.41107	2384	467	1917	0.198772	53.90889	53.71011	0.24361
2	2384	2278	106	95.55369128	4.446309	4768	2745	2023	1.168368	54.9856	53.81723	1.356896
3	2384	2358	26	98.90939597	1.090604	7152	5103	2049	2.172016	55.7614	53.58938	2.490483
4	2384	2358	26	98.90939597	1.090604	9536	7461	2075	3.175664	57.4642	54.28854	3.595663
5	2384	2366	18	99.24496644	0.755034	11920	9827	2093	4.182717	58.85827	54.67555	4.695174
6	2384	2367	17	99.28691275	0.713087	14304	12194	2110	5.190195	59.33633	54.14614	5.779147
7	2384	2355	29	98.78355705	1.216443	16688	14549	2139	6.192566	60.15186	53.95929	6.801777
8	2384	2369	15	99.37080537	0.629195	19072	16918	2154	7.200896	60.57368	53.37278	7.854225
9	2384	2363	21	99.11912752	0.880872	21456	19281	2175	8.206671	61.16423	52.95756	8.864828
10	2384	2362	22	99.07718121	0.922819	23840	21643	2197	9.212022	61.7829	52.57088	9.851161
11	2384	2364	20	99.16107383	0.838926	26224	24007	2217	10.21822	62.34533	52.12711	10.8286
12	2384	2370	14	99.41275168	0.587248	28608	26377	2231	11.22698	62.73903	51.51205	11.82295
13	2384	2366	18	99.24496644	0.755034	30992	28743	2249	12.23403	63.24522	51.01119	12.78035
14	2384	2363	21	99.11912752	0.880872	33376	31106	2270	13.23981	63.83577	50.59596	13.70308
15	2384	2373	11	99.5385906	0.461409	35760	33479	2281	14.24984	64.14511	49.89527	14.67733
16	2384	2363	21	99.11912752	0.880872	38144	35842	2302	15.25562	64.73566	49.48004	15.56994
17	2384	2372	12	99.4966443	0.503356	40528	38214	2314	16.26522	65.07312	48.80789	16.51426
18	2384	2369	15	99.37080537	0.629195	42912	40583	2329	17.27355	65.49494	48.22139	17.42508
19	2384	2361	23	99.0352349	0.964765	45296	42944	2352	18.27848	66.14173	47.86326	18.2585
20	2384	2368	16	99.32885906	0.671141	47680	45312	2368	19.28638	66.59168	47.3053	19.13514

6.3 VALIDATION (OUT OF TIME) DATA

- **Total no. of records in test set: 166,493**
- **Total no. of original Bads: 2,386**

OOT	# Records	# Goods	# Bads	Fraud Rate								
	166493	164107	2386	0.014330933								
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads	KS	FPR
1	1664	426	1238	25.60096154	74.39904	1664	426	1238	0.259587	51.88600	51.62641	0.344103
2	1664	1604	60	96.39423077	3.605769	3328	2030	1298	1.236998	52.97650	51.7395	1.563945
3	1664	1647	17	98.97836538	1.021635	4992	3677	1315	2.240611	53.81250	51.57189	2.796198
4	1664	1649	15	99.09855769	0.901442	6656	5326	1330	3.245444	55.47810	52.23266	4.004511
5	1664	1649	15	99.09855769	0.901442	8320	6975	1345	4.250276	56.37049	52.12022	5.185874
6	1664	1650	14	99.15865385	0.841346	9984	8625	1359	5.255717	56.95725	51.70153	6.346578
7	1664	1651	13	99.21875	0.78125	11648	10276	1372	6.261768	57.50210	51.24033	7.489796
8	1664	1654	10	99.39903846	0.600962	13312	11930	1382	7.269647	57.92121	50.65156	8.632417
9	1664	1650	14	99.15865385	0.841346	14976	13580	1396	8.275089	58.50796	50.23287	9.727794
10	1664	1651	13	99.21875	0.78125	16640	15231	1409	9.28114	59.05281	49.77167	10.80979
11	1664	1658	6	99.63942308	0.360577	18304	16889	1415	10.29146	59.30427	49.01282	11.93569
12	1664	1652	12	99.27884615	0.721154	19968	18541	1427	11.29812	59.80721	48.50909	12.99299
13	1664	1653	11	99.33894231	0.661058	21632	20194	1438	12.30539	60.26823	47.96285	14.04312
14	1664	1650	14	99.15865385	0.841346	23296	21844	1452	13.31083	60.85499	47.54416	15.04408
15	1664	1655	9	99.45913462	0.540865	24960	23499	1461	14.31932	61.23219	46.91287	16.08419
16	1664	1656	8	99.51923077	0.480769	26624	25155	1469	15.32841	61.56748	46.23906	17.12389
17	1664	1651	13	99.21875	0.78125	28288	26806	1482	16.33446	62.11232	45.77786	18.08772
18	1664	1655	9	99.45913462	0.540865	29952	28461	1491	17.34295	62.48952	45.14657	19.08853
19	1664	1653	11	99.33894231	0.661058	31616	30114	1502	18.35022	62.95054	44.60032	20.04927
20	1664	1650	14	99.15865385	0.841346	33280	31764	1516	19.35566	63.5373	44.18164	20.95251

7.0 CONCLUSION

7.1 SUMMARY

Application fraud is one of the most common identity frauds. For this project, we were required to examine the applications data and develop supervised machine learning models to predict fraud. We started off by building a baseline linear model (Logistic Regression) and then moved onto several non-linear models, namely, Neural nets, Decision trees, Boosted trees and Random Forest.

After comparing all the models, we can conclude that Gradient Boosting Trees performed the best.

The following are the FDR (Fraud Detection Rate) at 3% for the training, testing and out of time datasets:

Train	56.18%
Test	55.76%
OOT	53.81%

7.2 RECOMMENDATIONS

We hereby provide some recommendations to make the process of fraud detection more robust in future.

1. Since fraud problems are highly imbalanced, for future implementation of our algorithms, we would create balanced classes to reduce bias. Usually good to bad ratio should be 1:1 to 10:1

   Two ways we can do this are:

   - Subsampling the Goods: This involves sampling only some of the observations from the majority class (goods) so that the classes (good and bad) are not as disproportionate.
   - Up-sampling the Bads: This involves generating synthetic observations for the minority class (bads). We can do this using two methods- ROSE and SMOTE.

2. It would also be beneficial to have external datasets like records from phone carriers that would have the correct names associated with the phone numbers in our data. We can then identify who provided falsified information in their applications.

8.0 APPENDIX

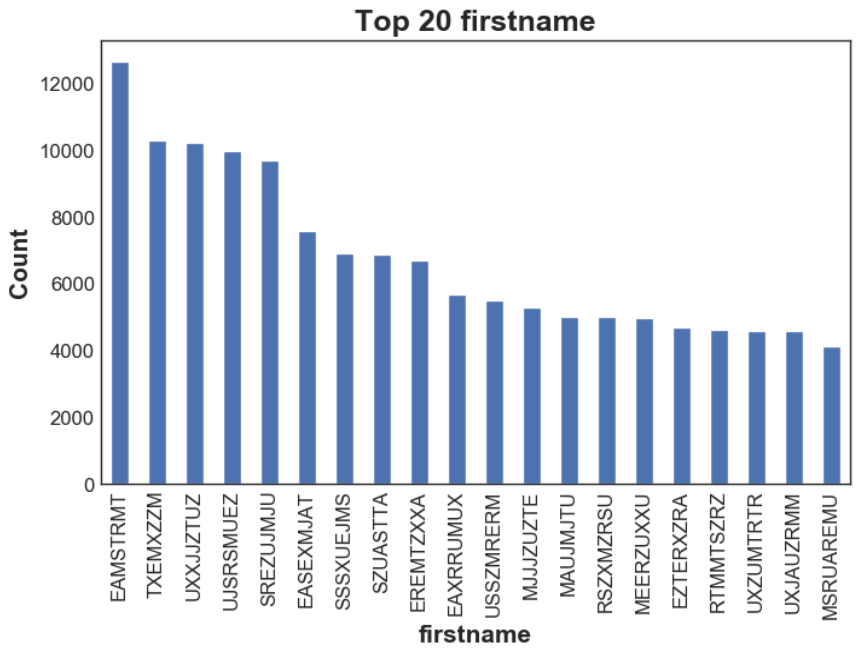
The Appendix includes the Data Quality Report related to the description of the remaining fields of the dataset.

8.1 record

Description: This field represents a unique identifier for each record or row in the dataset. Values in this field have a unique number for each of the 1000000 rows in the dataset.

8.2 firstname

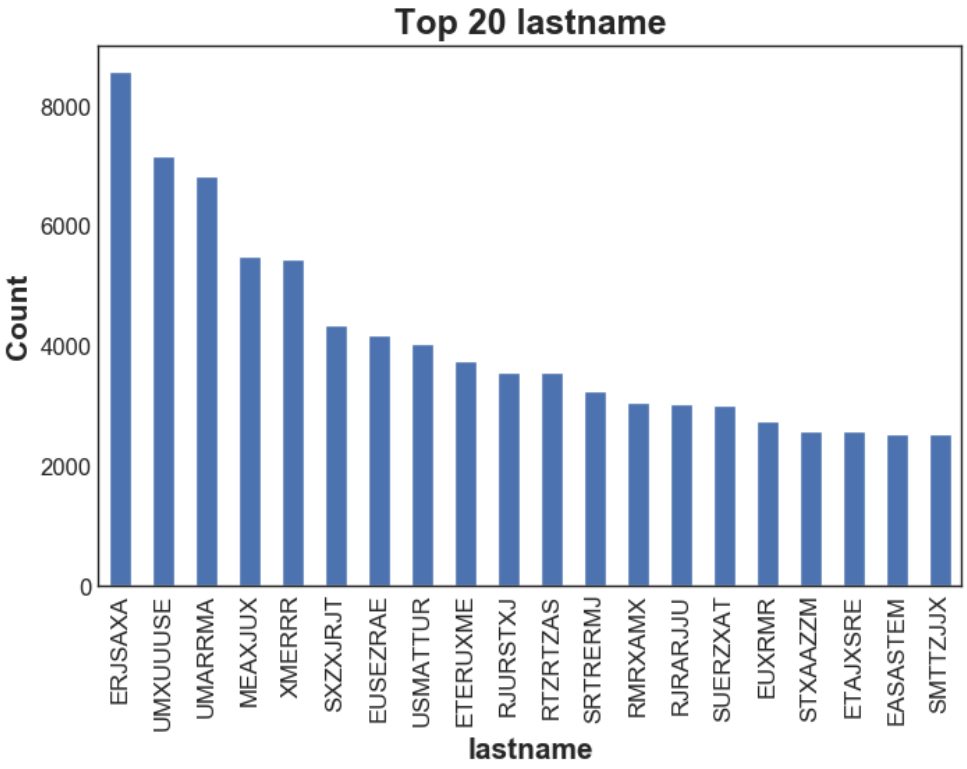
Description: This is a categorical field and contains the first names of the applicants. The field had 78136 unique values. The bar graph and the table show the count of the top 20 values of 'firstname' by count.



firstname	Count
EAMSTRMT	12658
TXEMXZZM	10297
UXXJJZTUZ	10235
UJSRSMUEZ	9994
SREZUJMJU	9688
EASEXMJAT	7576
SSSXUEJMS	6923
SZUASTTA	6878
EREMTZXXA	6717
EAXRRUMUX	5686
USSZMRERM	5511
MJJJZUZTE	5294
MAUJMJTU	5022
RSZXMZRSU	5006
MEERZUXXU	4966
EZTERXZRA	4716
RTMMTSZRZ	4618
UXZUMTRTR	4600
UXJAUZRMM	4580
MSRUAREMU	4145

8.3 lastname

Description: This is a categorical field and contains the last names of the applicants. The field has 177001 unique values. The bar graph shows the top 20 values of 'lastname' by count.



lastname	Count
ERJSAXA	8580
UMXUUUSE	7156
UMARRMA	6832
MEAXJUX	5492
XMERRR	5451
SXZXJRJT	4340
EUSEZRAE	4173
USMATTUR	4036
ETERUXME	3762
RJURSTXJ	3575
RTZRTZAS	3559
SRTRERMJ	3259
RMRXAMX	3074
RJRARJJU	3048
SUERZXAT	3026
EUXRMR	2757
STXAAZZM	2594
ETAJXSRE	2584
EASASTEM	2535
SMTTZJJX	2527