Shuhan Wang

Fraud Analytics

Project 2 Informal Model Report

I. Executive Summary

This project built a **fraud detection classification model** based on the **credit card transaction data**. The steps include a data quality report that provide statistical summary of each variable and data cleaning process, in which we fill in missing values and removing outlier. The next steps are variable creation and feature selections to determine the final amount of chosen variables, and preliminary model explorations that explores different types of models and tune different parameters. With FDR@3%, I obtained out-of-time rate at **70.84%**. Lastly, I examined the model performance, suggested a score cut-off of **3.5%** based on the financial curve, and concludes with the maximum savings of **$26,328,000** from this build fraud detection model.

II. Data Description

The card transactions data contains data about each credit card transactions. Specifically, it has transaction information and Personal Identifiable Information (PII) about the credit card transactions. The credit card transaction data covers **year 2010**. The dataset has **10 fields** of transaction information and **96,753 records**.

2. Summary Tables

(1) Numerical Table

Field Name	% Populated	Min	Max	Mean	Stdev	% Zero
Date	100.00	2010-01-01	2010-12-31	N/A	N/A	0.00
Amount	100.00	0.01	3,102,046.0	427.89	10,006.14	0.00

(2) Categorical Table

Field Name	% Populated	# Blank	# Zeros	# Unique Values	Most Common Value
Recnum	100.00	0	0	96,753	N/A
Cardnum	100.00	0	0	1,645	5142148452
Merchnum	96.51	3375	231	13,091	930090121224
Merch Description	100.00	0	0	13,126	GSA-FSS-ADV

Merch state	98.76	1195	0	227	TN
Merch zip	95.19	4656	0	4,567	38118
Transtype	100.00	0	0	4	P
Fraud	100.00	0	0	2	0

III. Data Cleaning

In the data cleaning stage, our data imputation method of the credit card transaction dataset includes detecting outlier value of 3102045.53 in amount field, which I deleted to ensure that it does not impact the model building. For data exclusion, I kept only the rows with "P" (purchase) in the Transaction type (Transtype) field and leaving out the rest. For the missing value in the field **Merchnum** (Total missing 3374), we replace the missing value with the mode of merchant description.

For the null values in **Merch state** (Total missing 1194), we map them to the corresponding zipcodes, fill in the missing values with the mode of Merchnum and Merch description and assign unknown for the rest. We indicate the values as 'PR' is the zipcode falls in the range 00600-00799, 00900-00999 and values as 'foreign' if the states are not in the US.

Lastly, for the missing data values in **Merch zip**, (Total missing 4616), we fill in the missing value with the mode of Merchnum.

IV. Variable Creation

In the Variable Creation stage after data cleaning, I created 12 variables. Creating variables based on the original variables given in the dataset helps imitating fraudulent situations in the real life. The first variable is the **dow_risk (Day of Week Risk),** which gives the average fraud rate on the credit card transactions on each day of the week. I applied Benford's Law and U_smoothed score on Cardnum and Merchnum to see how likely the records will be synthetic. I created variables like **Day since** and **Frequency**, which tells the number of days since the last time a business saw the same value of entities and the number of times the same values appear in the past 0,1,3,7,14,30 days. These variables are meaningful because in real life, fraudulent transactions are made frequently. I also created variables like **Velocity Day Since** and **Velocity change**, which detect the ratio of changing value and the frequency of appearances. This helps detecting change of fraudulent behaviors. Below is the table providing the description and number of each variable I created.

Description of Variables	**# Variable Created**

Day of Week Risk (dow_risk): Average fraud rate on the credit card transactions on each day of the week	1
Day since Variables: Records the number of days since the last time a business saw the same value of entities of a credit card transaction, such as Cardnum, Merchnum, card_merch, card_zip, card_state, merch_zip, merch_state	7
Frequency Variables: Records the number of times that the same value of each entity occurs in the past 0,1,3,7,14,30 days	42
Amount Variables: Average, Maximum, Median, Total, Actual/average, Actual/maximum, Actual/median, Actual/total number of occurrences of each entity in the past 0,1,3,7,14,30 days	336
Cross entity uniqueness variables: Combination of 2 entities selected from a set of 7 distinct entities (Cardnum, Merchnum, card_merch, card_zip, card_state, merch_zip, merch_state) and count the unique values of each combination with the 7 entities.	42
Velocity change Variables: Velocity change is the ratio that calculates # of the transactions with a specific entity value seen in the recent past (0 or 1 day) divided by the average daily frequency seen in the past 7,14, and 30 days.	42
Velocity days since Variables: The ratio that calculates the velocity change over the past 0,1 days divided by the velocity change in the past 7,14,30 days, and the day since variables for the given entity	42
Variability Variable: The variability (Maximum, medium and minimum) of each transaction amount of each entity in the past 1,3,7,14,30 days	126
Acceleration: The ratio that calculates the squared velocity of each entity in the past 0,1 days over the ratio of count of occurrences of each entity and its power in the past 7,14,30 days	42
Minimum Indicator (New Variable): This variable includes the minimum count of each entity that was seen in the last (1,3,7,30) days	28
Maximum Indicator (New Variable): This variable includes the maximum count of each entity that was seen in the last (1,3,7,30) days	28
Mean Indicator (New Variable): This variable includes the mean values of each entity that was seen in the last (1,3,7,30) days	28

V. Feature Selection

Performing feature selection to narrow down the number of final variables to choose from the dataset help prevent overfitting issue or curse of dimensionality. Curse of dimensionality that arises from too many variables is problematic because it will become difficult to fit non-linear model. Performing feature selection also helps us to get a sorted list of variables that we should add or remove, allows faster nonlinear model runs, as well as testing many potential variables.

For this binary classification problem, our original dataset has **96,753** variables, which makes the dimensionality of model building very high. I tested forward selection and backward selection as the wrapper method for both LGBM and Random Forest and tested different num_filter and num_wrapper. Finally, I chose to use LGBM with forward selection method for this feature selection, which has num_filter = 400 and gives 30 top variables(num_wrapper). This model reached 0.78 performance score, which is the highest among all the models I have tested.

I was able to get the Filter Score (Univariate KS's) for each independent variables. The selected 400 variables sorted by the highest KS's will then be applied with forward selection wrapper method and the rest will be discarded. By starting with 1-d models and adding more variables into the model to test the performance and using detection rate as the measure of model performance, the result indicates that performance saturates at 0.78 and the optimal variables to be chosen is around 30.

List of Final Variables & Univariate KS's

Wrapper Order	Variable	Filter Score (Univariate KS's)
1	Card_Merchdesc_total_14	0.6656
2	zip3_med_3	0.4123
3	card_zip3_total_7	0.6199
4	Cardnum_total_amount_1_by_30	0.5704
5	Merchnum_desc_total_1	0.5154
6	Cardnum_max_60	0.6818
7	state_risk	0.4378
8	card_merch_variability_max_1	0.6819
9	Merchnum_desc_variability_med_14	0.6762
10	Cardnum_total_amount_1_by_14	0.6719
11	zip3_max_1	0.6663

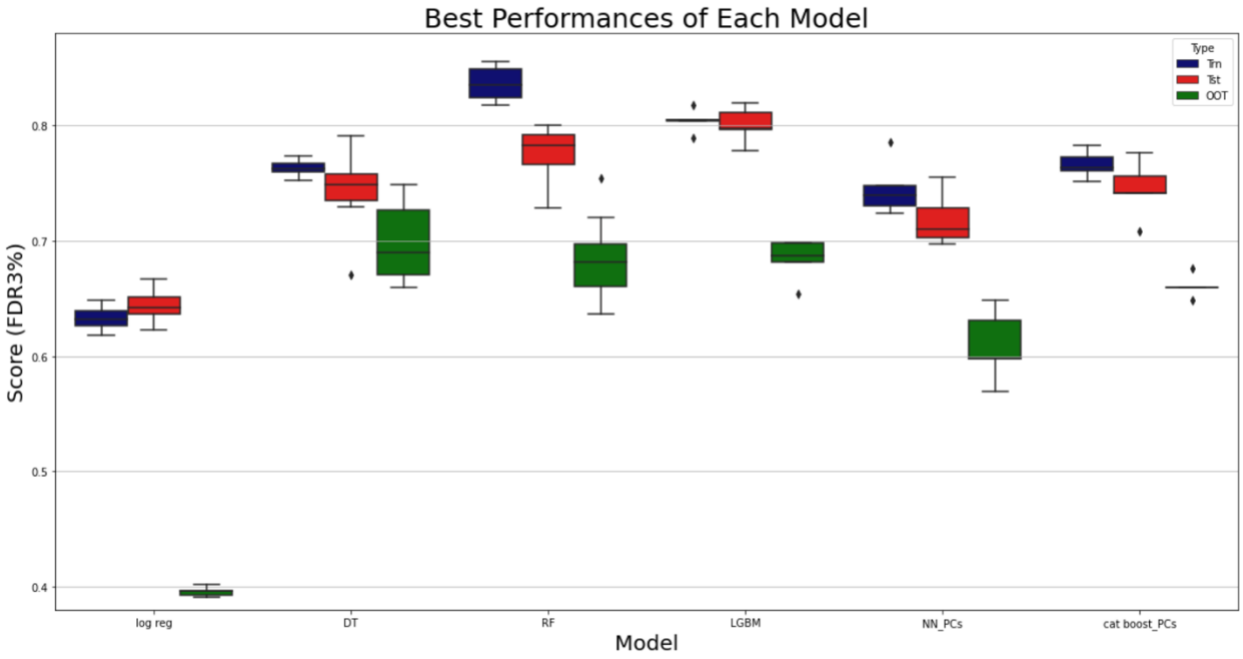
12	Merchnum_desc_med_30	0.6634
13	merch_zip_total_7	0.6141
14	Merchnum_variability_med_7	0.6091
15	Merchnum_total_14	0.5751
16	card_merch_total_14	0.5450
17	card_zip_total_14	0.5424
18	Card_Merchnum_desc_total_14	0.5067
19	card_zip3_total_30	0.4831
20	card_merch_total_30	0.6463
21	Card_Merchdesc_total_30	0.6452
22	Card_Merchnum_desc_total_30	0.6430
23	card_zip_total_30	0.6073
24	card_zip_total_60	0.6051
25	Card_Merchnum_desc_total_60	0.5451
26	Card_Merchdesc_total_60	0.5337
27	card_merch_total_60	0.5222
28	Merchnum_total_1	0.6433
29	merch_zip_total_1	0.5365
30	Merchnum_variability_med_7	0.5876

VI. Preliminary Model Exploration

Model Name	Parameter							Average FDR at 3%		
Logistic Regression	**Number of Variables**	**max_iter**	**penalty**	**C**	**solver**	**l1_ratio**	**fit_intercept**	**Train**	**Test**	**OOT**
1	5	150	eslastinet	1	saga	0.7	TRUE	0.5997	0.5879	0.4026
2	15	100	N/A	0.5	saga	N/A	TRUE	0.6372	0.6264	0.4309
3	10	200	N/A	1	lbfgs	0.7	TRUE	0.6285	0.6264	0.4469
4	20	200	N/A	0.7	lbfgs	N/A	FALSE	0.5694	0.5335	0.4006
5	20	100	l1	1	saga	N/A	TRUE	0.6057	0.6077	0.4338
Decision Tree	**Number of Variables**	**criterion**	**splitter**	**max_depth**	**min_samples_split**	**min_samples_leaf**		**Train**	**Test**	**OOT**
1	5	gini	best	5	50	30		0.6878	0.6731	0.4797
2	10	entropy	random	10	40	70		0.5878	0.5923	0.5863
3	10	gini	best	10	30	50		0.6835	0.6904	0.4684
4	10	gini	best	20	200	150		0.6986	0.6428	0.5053
5	15	entropy	random	20	20	200		0.7054	0.7952	0.4882
6	20	entropy	random	25	15	100		0.6846	0.6782	0.6021
7	20	gini	best	10	100	150		0.7619	0.7564	0.7084
Random Forest	**Number of Variables**	**criterion**	**n_estimators**	**min_samples_split**	**min_samples_leaf**	**max_features**	**max_depth**	**Train**	**Test**	**OOT**
1	10	gini	5	50	30	2	10	0.8623	0.7689	0.5581
2	10	gini	10	40	25	4	15	0.7844	0.7872	0.5683
3	10	gini	30	30	16	5	15	0.8112	0.7826	0.4927
4	15	gini	35	20	13	7	10	0.8145	0.7903	0.5471
5	15	entropy	70	10	10	10	30	0.7798	0.7532	0.532
6	15	entropy	80	5	4	10	25	0.7386	0.7012	0.5575
LightGBM	**Number of Variables**	**n_estimators**	**learning_rate**	**reg_alpha**	**reg_lambda**	**num_leaves**	**max_depth**	**Train**	**Test**	**OOT**
1	20	10	0.1	0	0	100	5	0.792	0.7588	0.5537
2	10	20	0.1	0	0	200	10	0.8067	0.7625	0.5648
3	10	35	0.5	0	0.5	300	15	0.7659	0.7692	0.5423
4	15	60	0.01	0.5	0	400	30	0.7685	0.7743	0.5476
5	10	80	0.01	0	0	500	40	0.9041	0.5641	0.6715
Neural Network	**Number of Variables**	**hidden_layer_size**	**activation**	**alpha**	**learning_rate**	**max_iteration**	**learning_rate_init**	**Train**	**Test**	**OOT**
1	5	(10,10)	relu	0.1	constant	100	0.1	0.6527	0.4671	0.4836
2	20	(5,)	logistic	0.01	constant	300	0.1	0.6573	0.5693	0.4978
3	10	(10,10)	relu	0.01	adaptive	400	0.001	0.7627	0.7539	0.5497
4	15	(10,10)	logistic	0.0001	constant	50	0.005	0.713	0.7086	0.5289
5	5	(20,20,20)	relu	0.001	adaptive	100	0.001	0.7244	0.6952	0.4862
CatBoost	**Number of Variables**	**l2_leaf_reg**	**max_depth**	**random_state**	**learning_rate**	**iteration**	**bootstap_type**	**Train**	**Test**	**OOT**
1	10	1	2	30	0.1	100	Bayesian	0.8109	0.8298	0.5234
2	10	20	10	2	0.05	300	Bernoulli	0.8196	0.7226	0.5813
3	10	5	5	15	0.1	200	Bayesian	0.7342	0.753	0.5565
4	15	7	8	10	0.05	200	Bayesian	0.7823	0.7532	0.5798
5	15	10	10	5	0.005	150	Bayesian	0.712	0.7132	0.5018

For preliminary model exploration, I applied the method of cross validation, which splits the data into separate groups. Cross validation helps training part of the data and then test on the remaining data when I build 1 linear model and 5 non-linear models. In this situation, the proportion of training is 70% and testing is 30%. 5 different non-linear models (Decision Tree, Random Forest, LightGBM, Neural Network, and CatBoost) was built to better look at the performances between these models and their statistical fluctuation. All models were evaluated 5 times so that I was able to tune the hyperparameters to obtain the one with the strongest performance.

I started with the logistic regression as the linear model because it will provide a baseline and robust result that is hard to mess up. Logistic Regression showed the least training and testing data percentage. For each of the 5 non-linear models, I started minimum complexity and then gradually increase to compare training performance and testing performance, which enabled me to tune hyperparameters to get both overfitting and underfitting for each model. Overfitting is when my training data is much better than the testing data and underfitting is when my testing data is better than my training data.



The model with the best performance is **Decision Tree**, because the train (0.7619) and test scores(0.7564) are both very high. The test score is just a little bit higher than the train score and the out-of-time score is stable.

VII. Final Model Performance

For the final model, I chose Decision Tree. The parameters of the model is **DecisionTreeClassifier(max_depth=10, min_samples_split=100, min_samples_leaf=150, criterion = 'gini', splitter = 'best')**. The following three tables are the statistics of the training, testing, and out-of-time dataset. In the training table, I have obtained fraud rate of **1.11%** with 750 of the records are bad and 66727 records are good. For the testing table, 28611 records are good and 309 records are bad, which gives fraud rate of **1.07%**. Finally, in the oot table, I have 12248 good records and 179 bad records, which gives me a fraud rate of **1.44%**.

Trn	# Records	#Goods	#Bads	Fraud Rate
	67477	66727	750	0.0111

	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	#Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads(FDR)	KS	FPR
1	0	0	0	0	0	0	0	0	0	0	0	0
2	675	298	377	44.1481481	55.8518519	675	298	377	0.44659583	50.26666667	49.8200708	0.79045093
3	675	558	117	82.6666667	17.3333333	1350	856	494	1.28283903	65.86666667	64.5838276	1.73279352
4	674	586	88	86.9436202	13.0563798	2024	1442	582	2.16104425	77.6	75.4389557	2.47766323
5	675	633	42	93.7777778	6.22222222	2699	2075	624	3.10968573	83.2	80.0903143	3.32532051
6	675	653	22	96.7407407	3.25925926	3374	2728	646	4.08830009	86.13333333	82.0450332	4.22291022
7	675	661	14	97.9259259	2.07407407	4049	3389	660	5.07890359	88	82.9210964	5.13484848
8	674	666	8	98.8130564	1.18694362	4723	4055	668	6.07700031	89.06666667	82.9896664	6.07035928
9	675	670	5	99.2592593	0.74074074	5398	4725	673	7.08109161	89.73333333	82.6522417	7.02080238
10	675	666	9	98.6666667	1.33333333	6073	5391	682	8.07918833	90.93333333	82.854145	7.90469208
11	675	673	2	99.7037037	0.2962963	6748	6064	684	9.08777556	91.2	82.1122244	8.86549708
12	674	669	5	99.2581602	0.74183976	7422	6733	689	10.0903682	91.86666667	81.7762984	9.77213353
13	675	671	4	99.4074074	0.59259259	8097	7404	693	11.0959582	92.4	81.3040418	10.6839827
14	675	673	2	99.7037037	0.2962963	8772	8077	695	12.1045454	92.66666667	80.5621213	11.6215827
15	675	671	4	99.4074074	0.59259259	9447	8748	699	13.1101353	93.2	80.0898647	12.5150215
16	675	675	0	100	0	10122	9423	699	14.1217198	93.2	79.0782802	13.4806867
17	674	673	1	99.851632	0.14836795	10796	10096	700	15.1303071	93.33333333	78.2030263	14.4228571
18	675	673	2	99.7037037	0.2962963	11471	10769	702	16.1388943	93.6	77.4611057	15.3404558
19	675	674	1	99.8518519	0.14814815	12146	11443	703	17.1489802	93.73333333	76.5843532	16.2773826
20	675	671	4	99.4074074	0.59259259	12821	12114	707	18.1545701	94.26666667	76.1120966	17.1343706

TST	# Records	#Goods	#Bads	Fraud Rate
	28920	28611	309	0.0107

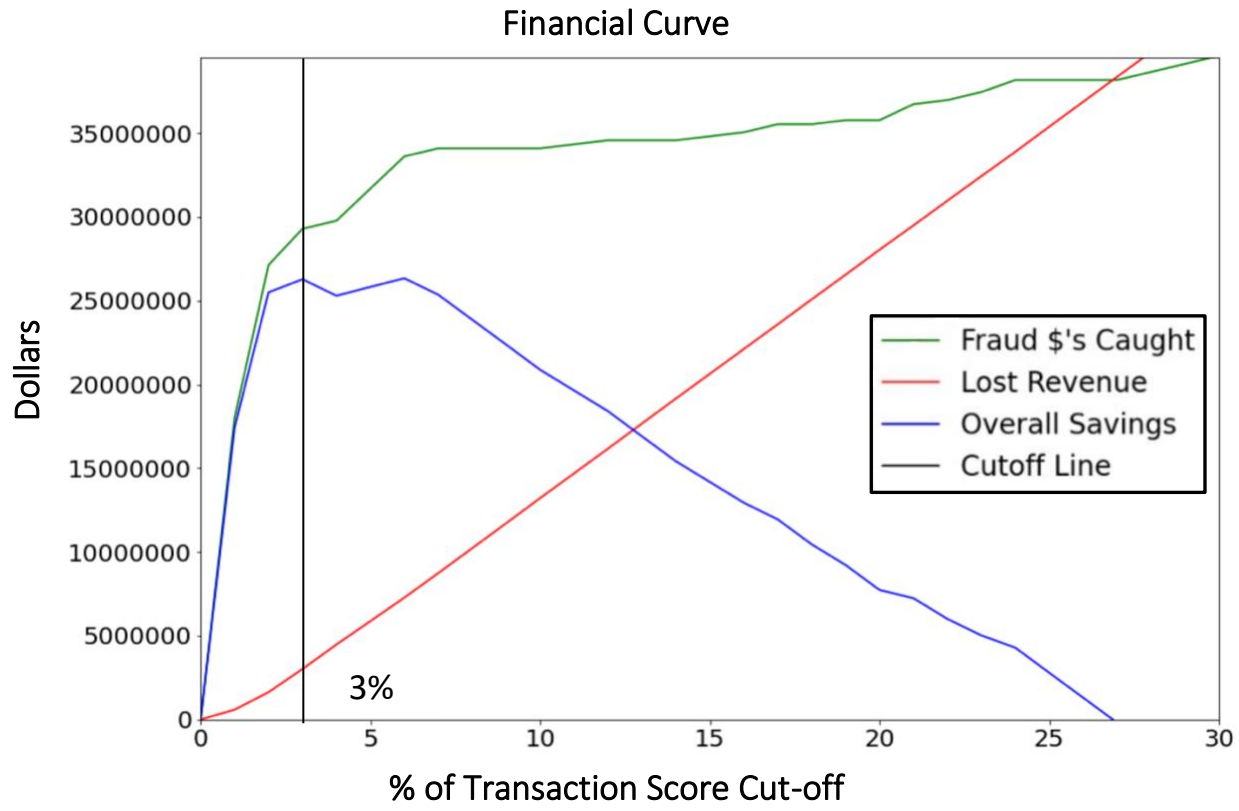
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	#Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads(FDR)	KS	FPR
1	0	0	0	0	0	0	0	0	0	0	0	0
2	289	128	161	44.2906574	55.7093426	289	128	161	0.44738038	52.10355987	51.6561795	0.79503106
3	289	255	34	88.2352941	11.7647059	578	383	195	1.33864598	63.10679612	61.7681501	1.96410256
4	290	256	34	88.2758621	11.7241379	868	639	229	2.23340673	74.11003236	71.8766256	2.79039301
5	289	266	23	92.0415225	7.95847751	1157	905	252	3.16311908	81.55339806	78.390279	3.59126984
6	289	285	4	98.615917	1.38408304	1446	1190	256	4.15923945	82.84789644	78.688657	4.6484375
7	289	285	4	98.615917	1.38408304	1735	1475	260	5.15535983	84.14239482	78.987035	5.67307692
8	289	288	1	99.6539792	0.34602076	2024	1763	261	6.16196568	84.46601942	78.3040537	6.75478927
9	290	289	1	99.6551724	0.34482759	2314	2052	262	7.17206669	84.78964401	77.6175773	7.83206107
10	289	287	2	99.3079585	0.69204152	2603	2339	264	8.17517738	85.4368932	77.2617158	8.85984848
11	289	287	2	99.3079585	0.69204152	2892	2626	266	9.17828807	86.08414239	76.9058543	9.87218045
12	289	287	2	99.3079585	0.69204152	3181	2913	268	10.1813988	86.73139159	76.5499928	10.869403
13	289	285	4	98.615917	1.38408304	3470	3198	272	11.1775191	88.02588997	76.8483708	11.7573529
14	290	290	0	100	0	3760	3488	272	12.1911153	88.02588997	75.8347747	12.8235294
15	289	288	1	99.6539792	0.34602076	4049	3776	273	13.1977212	88.34951456	75.1517934	13.8315018
16	289	288	1	99.6539792	0.34602076	4338	4064	274	14.204327	88.67313916	74.4688122	14.8321168
17	289	289	0	100	0	4627	4353	274	15.214428	88.67313916	73.4587111	15.8868613
18	289	287	2	99.3079585	0.69204152	4916	4640	276	16.2175387	89.32038835	73.1028496	16.8115942
19	290	288	2	99.3103448	0.68965517	5206	4928	278	17.2241446	89.96763754	72.743493	17.7266187
20	289	287	2	99.3079585	0.69204152	5495	5215	280	18.2272553	90.61488673	72.3876315	18.625

OOT	# Records	#Goods	#Bads	Fraud Rate
	12427	12248	179	0.0144

	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	#Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads(FDR)	KS	FPR
1	0	0	0	0	0	0	0	0	0	0	0	0
2	124	49	75	39.516129	60.483871	124	49	75	0.40006532	41.89944134	41.499376	0.65333333
3	125	87	38	69.6	30.4	249	136	113	1.11038537	63.12849162	62.0181063	1.20353982
4	124	115	9	92.7419355	7.25806452	373	251	122	2.04931417	68.15642458	66.1071104	2.05737705
5	124	122	2	98.3870968	1.61290323	497	373	124	3.04539517	69.27374302	66.2283479	3.00806452
6	124	116	8	93.5483871	6.4516129	621	489	132	3.99248857	73.74301676	69.7505282	3.70454545
7	125	117	8	93.6	6.4	746	606	140	4.94774657	78.2122905	73.2645439	4.32857143
8	124	122	2	98.3870968	1.61290323	870	728	142	5.94382756	79.32960894	73.3857814	5.12676056
9	124	124	0	100	0	994	852	142	6.95623775	79.32960894	72.3733712	6
10	124	124	0	100	0	1118	976	142	7.96864794	79.32960894	71.360961	6.87323944
11	125	125	0	100	0	1243	1101	142	8.98922273	79.32960894	70.3403862	7.75352113
12	124	123	1	99.1935484	0.80645161	1367	1224	143	9.99346832	79.88826816	69.8947998	8.55944056
13	124	123	1	99.1935484	0.80645161	1491	1347	144	10.9977139	80.44692737	69.4492135	9.35416667
14	125	125	0	100	0	1616	1472	144	12.0182887	80.44692737	68.4286387	10.2222222
15	124	124	0	100	0	1740	1596	144	13.0306989	80.44692737	67.4162285	11.0833333
16	124	123	1	99.1935484	0.80645161	1864	1719	145	14.0349445	81.00558659	66.9706421	11.8551724
17	124	123	1	99.1935484	0.80645161	1988	1842	146	15.0391901	81.56424581	66.5250557	12.6164384
18	125	123	2	98.4	1.6	2113	1965	148	16.0434357	82.68156425	66.6381286	13.277027
19	124	124	0	100	0	2237	2089	148	17.0558459	82.68156425	65.6257184	14.1148649
20	124	123	1	99.1935484	0.80645161	2361	2212	149	18.0600914	83.24022346	65.180132	14.8456376

VIII. Financial Curve and Recommended Cutoff

For the financial curve, we assumed **$400** gain for every fraud detected, shown in the green curve, and **$20** loss for every false positive, shown in the red curve. The blue curve indicates the overall savings. I recommended a **cut-off score** at approximately **3.5**, which gives us **max possible savings** of **$26,328,000**.



IX. Summary

In summary, to build the fraud model for the credit card transactions, it took stages beginning with providing statistical summary of each of 10 variables with **10,753** rows in the given dataset. Presenting a statistical summary for the categorical and numerical variables and visualizing them in form of bar chart and histograms is an essential step before starting the model building because it helps me to understand the distribution of each variable in the dataset, which can provide insights into the nature of the data. Data cleaning helps to identify data quality issues. Data quality issues include missing values, outliers, data entry errors, and fixing them make sure that the following stages are accurate.

Following data cleaning is the variable creation, which started the model building with creating as many variables as possible from the original dataset and experiment and explore. This step helps improving model accuracy and account for non-linear relationships between the columns.

The use of feature selection is crucial when dealing with high-dimensional datasets to reduce the number of variables and prevent overfitting. Forward selection was used as the wrapper method for LGBM, and the top 400 variables were selected based on their Filter Score. By gradually adding more variables, the optimal number of variables was found to be around 30, achieving the highest performance score of 0.78.

The last few steps are preliminary model exploration, in which I explored 6 different models and compared the traning, testing, and oot performance after splitting the dataset into training and testing. Based on the overall training, testing, and oot score, the final model that I chose is the **Decision Tree,** which has best training accuracy and highest oot of 0.7084. The associated parameters that I choose are

max_depth=10, min_samples_split=100, min_samples_leaf=150, criterion = 'gini', splitter = 'best'

The final oot, training, and testing data result table are shown in **Final Model Performance** section. The fraud rate for out-of- time is 1.444%, for training data is 1.11%, for testing data is 1.07%.

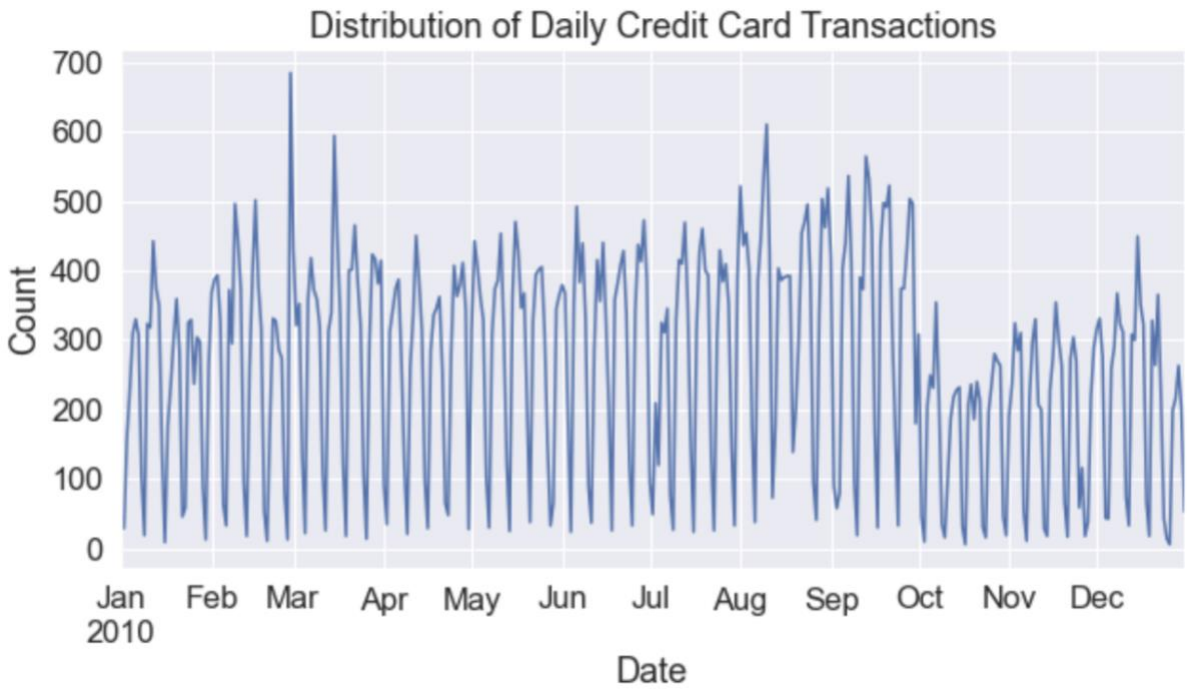
X. Appendix

(1) Field Name: **Recnum**

Description: Recnum field is categorical and tracks the number of rows/records of applications data in the dataset, using ordinal unique positive number from **1 to 96,753**.

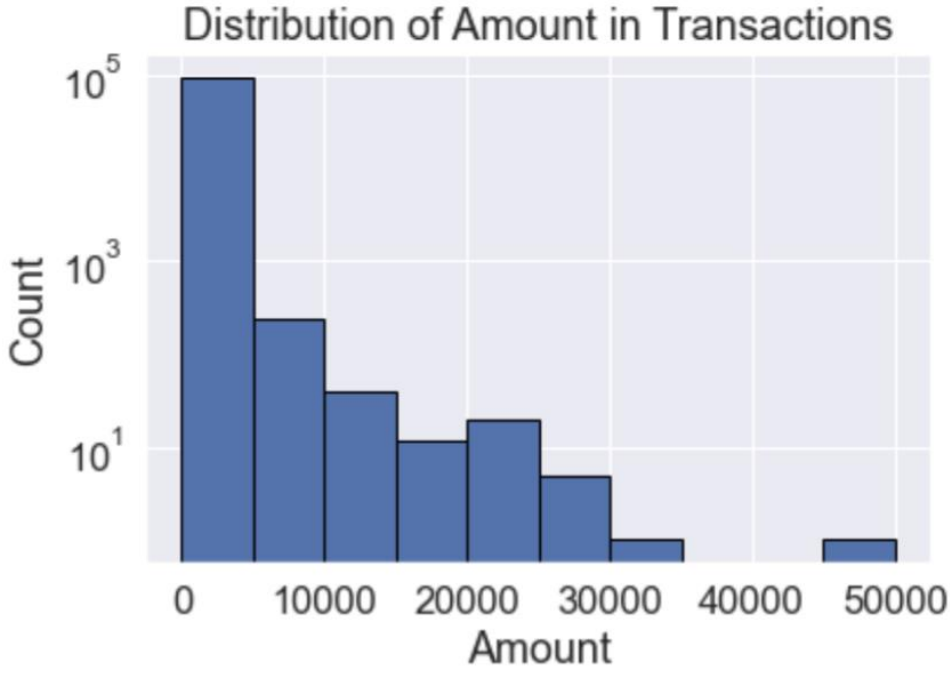
(2) Field Name: **Date**

Description: Date field, which is numerical and indicate the date on which the credit card transaction was made. The plot shows how many credit card transactions was made each day from 2010-01-01 to 2010-12-31. On **2010-2-28**, the most transactions was made, which is 648.



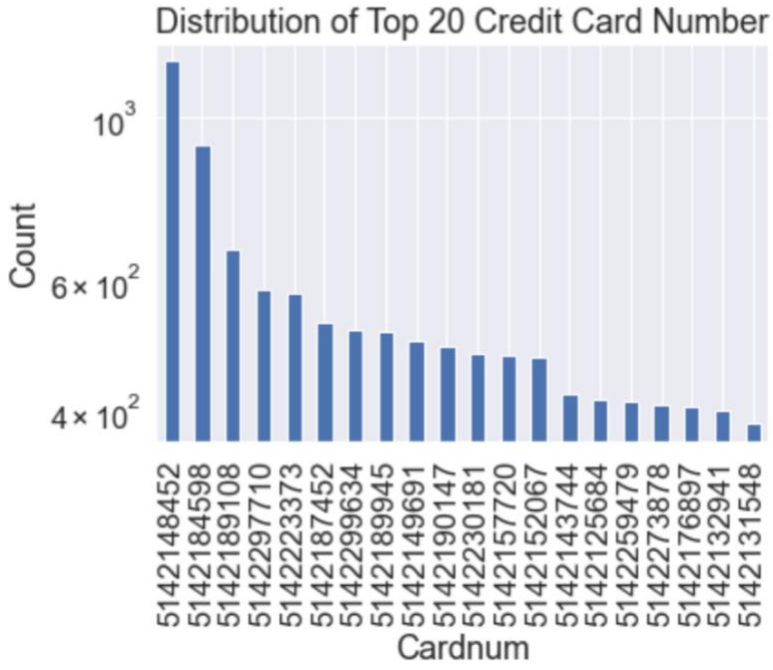
(3) Field Name: **Amount**

Description: Amount of money being transferred in each credit card transaction field, which is numerical field. Below displays histogram with values of amounts with range from 0 to 50,000 dollars and their frequency of appearing in the transactions. The most common value of Amounts is **$3.62** which has total count of **4,283**. However, after I grouped the amount into 10 bins (0-5,000, 5,000-10,000 etc), the most common amount in transactions is bin 1 (0-5,000) which has total count of about **100,000**.



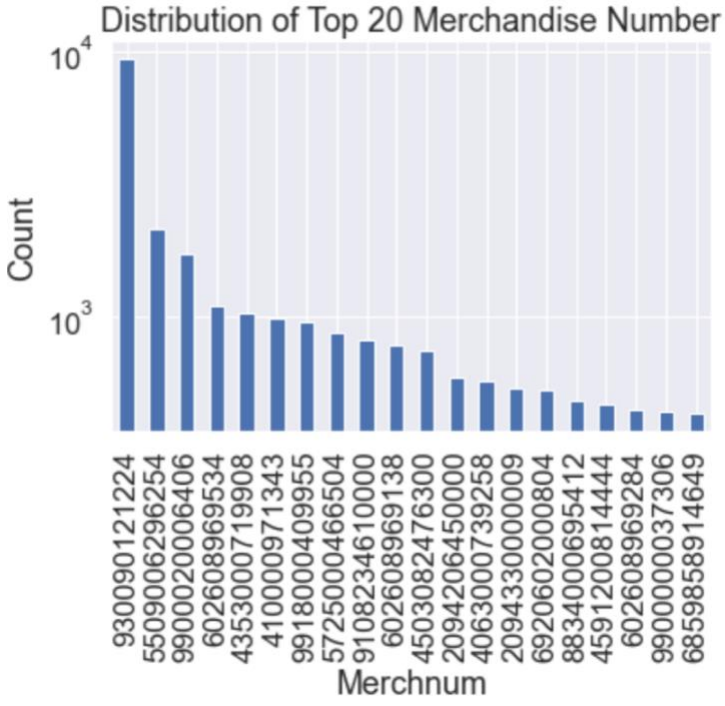
(4) Field Name: **Cardnum**

Description: Fixed 10-digits credit card number field. It is a categorical field. Below displays a bar chart of the top 20 values. The most common credit card number among all credit card transactions is **5142148452** with a total count of **1,192**.



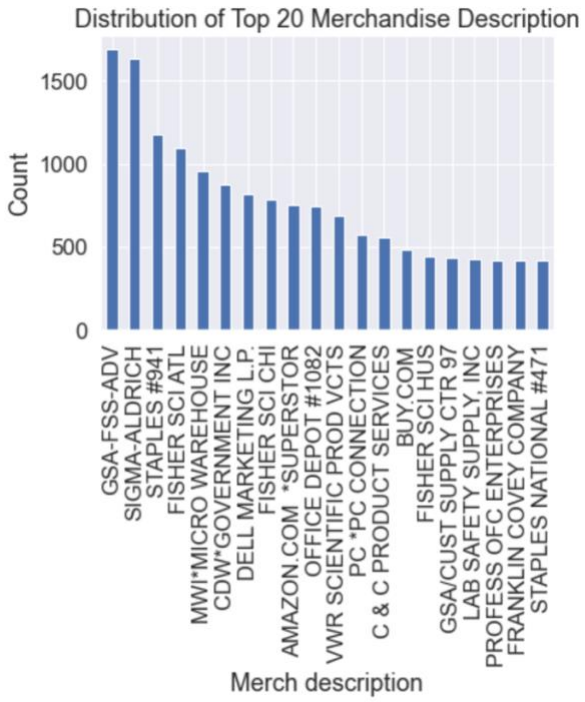
(5) Field Name: **Merchnum**

Description: Merchandise number field. It is a categorical field. Below displays a bar chart of the top 15 values. The most common merchandise number among all credit card transactions is **930090121224** with a total count of **9310.**



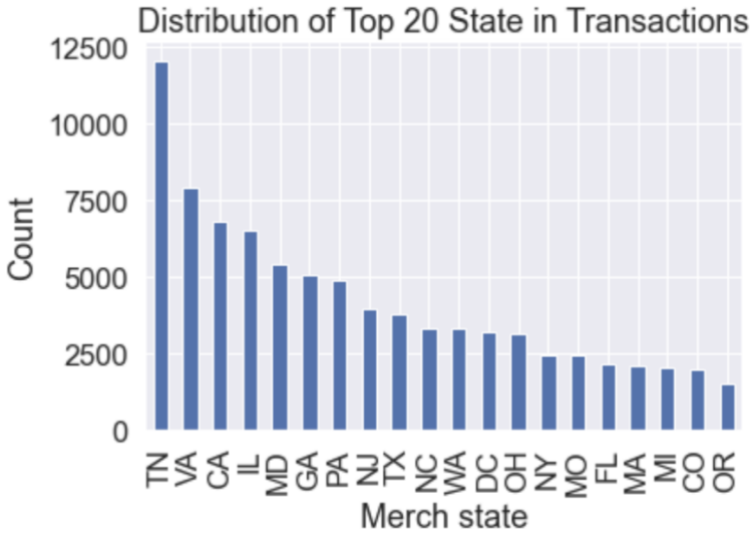
(6) Field Name: **Merch description**

Description: Merchandise description field of credit card transactions data, which tells the name of the merchandise being purchased in each transaction. It is a categorical field which display bar chart of the top 20 values. The most common merchandise name among all transactions is **GSA-FSS-ADV** with a total count of **1,688**.



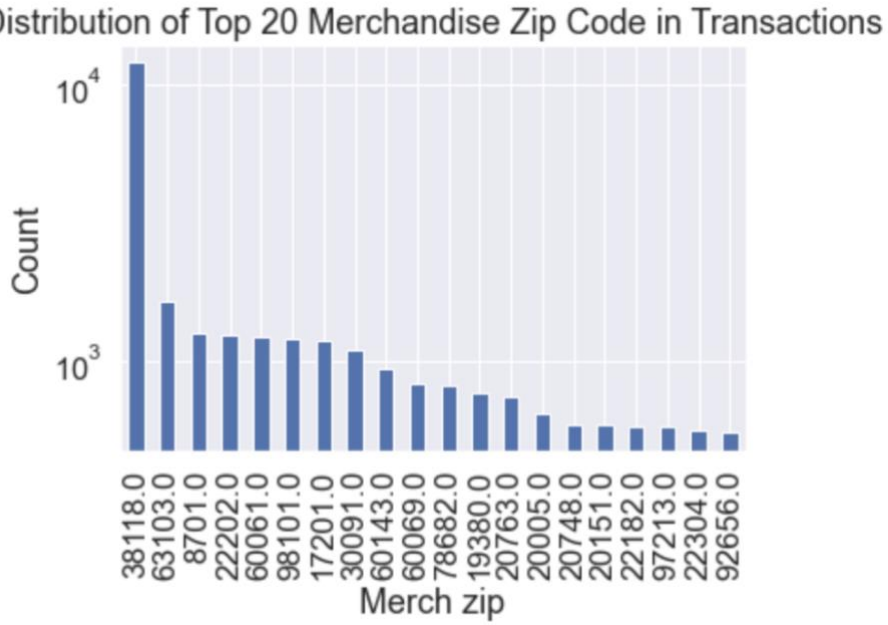
(7) Field Name: **Merch state**

Description: Merchandise state field of credit card transaction data, which is in the format of state abbreviation and tells the state location in which the transaction was made. It is a categorical field which displays bar chart of the top 20 values. The most common merchandise state among all transactions is **TN(Tennessee)** with a total transaction count of **12035.**



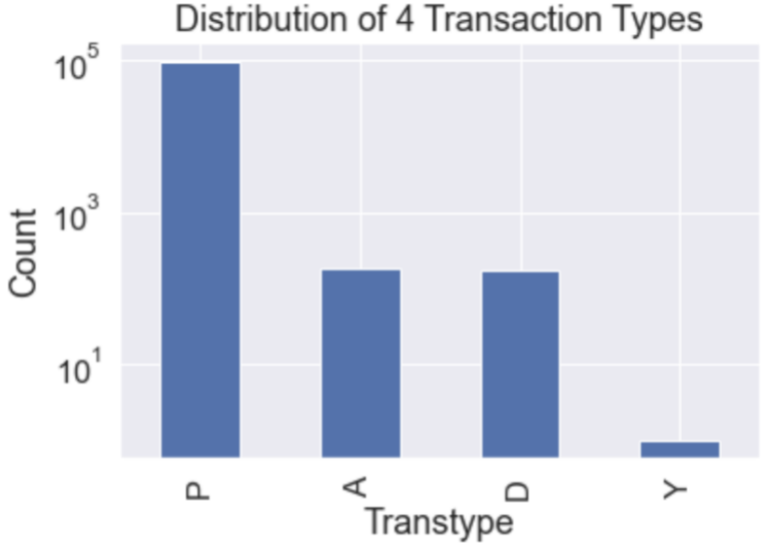
(8) Field Name: **Merch zip**

Description: Merchandise zip code field of each credit card transaction, which is categorical and is in the format of 5 digits. The most common merchandise zip code is **38118** which has total transaction count of **11,868**.



(9) Field Name: **Transtype**

Description: Transaction type field of applications data, which have 4 categories: P, A, D, Y. It is a categorical field. The bar graph below shows the frequency of each transaction type in credit card transactions. The most common transaction type among all transactions is **P** with a total count of **96,398.**



(9) Field Name: **Fraud**

Description: Fraud value field of credit card transaction, which takes binary values of 1 if a transaction is labeled as fraud and 0 if a transaction is labeled as not fraud. It is a categorical field

which displays bar chart of the only two values (0 and 1). The most common fraud label among all transactions is **0** with a total count of **95,694**.