Homework 12. Name: I-Chieh (Hazel) Huang

I. Executive Summary

In fraud, one of the most common types is transaction fraud, which is this project mainly about. First, after data cleaning, I will create variables to increase accuracy in machine learning models of fraud detection. Later on, I will try out different model sets to do feature selection on these variables top get the best performance. Then, use these selected features to fit in 8 machine learning models with different parameters. Among these models, get the best OOT value without overfitting or underfitting the model with the FDR @ 3%. With this model, I will decide the best financial cutoff line to get the best estimated savings.

II. Data Description

This dataset is a credit card transaction data. It contains information of transactions for credit cards from 2010-01-01 to 2010-12-31. There are 10 fields and 96,753 transaction data.

Numerical Table:

Field Name	%Populated	Min	Max	Mean	Stdev	% Zero
Date	100	NA	NA	NA	NA	0
Amount	100	0.01	3102045.53	427.89	10006.14	0

Categorical Table:

Field Name	% Populated	# Unique Values	Most Common Value
Recnum	100	96,753	NA
Cardnum	100	1645	5142148452
Merchnum	96.51	13,092	930090121224
Merch description	100	13,126	GSA-FSS-ADV
Merch state	98.76	228	TN
Merch zip	95.19	4568	38118
Transtype	100	4	P
Fraud	100	2	0

III. Data Cleaning

First of all, I deleted a single amount outlier of 3102045.53, which is not recorded in USD. Then, since I built model for purchase transaction, we only kept "purchase" data. After these, I could see that there were missing values in Merchnum, Merch state and Merch zip. For Merchnum, I filled in null values with corresponding merch description, and assigned the rest of them as "unknown". For Merch state, I filled in null values with corresponding merch description and merch zip code, assigned non-US states as "foreign", and the rest as "unknown". Finally, for Merch zip, I filled in null values with corresponding merch description and merchnum, and the rest as "unknown".

IV. Variable Creation

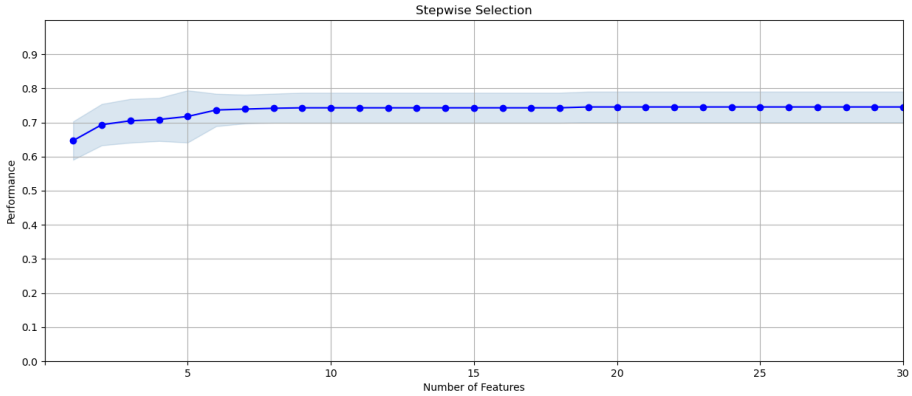
To identify fraud transaction more accurately, I added different variables into the data set. Some calculate the frequency and some measure the velocity changes in different days. These contains different aspect to see whether the transaction is unusual or not. First, I created day since variables to see the period of time for each entity first appeared. Next, I created frequency & amount variables, velocity change variables, velocity days since variables, cardnum frequency variables, acceleration variables to see the transaction fluctuation of each entity. Then, I created unique value variables to see the total number of unique values in each entity. I looked into the amount variables to see if there are changes for their average, maximum and median values. Lastly, I created unique count variables to see the counts of entities in each period of time. I created 2634 variables in total. These variables will all help detect anomalous transaction.

Description of the variables	# Variables created
Day Since Variables: How many days since the last transaction for that entity.	15
Frequency & Amount Variables: The number of transactions with the same entity over 0, 1, 3, 7, 14, 30	810
Velocity Change Variables: This set of variables is the number of each entity in the past 0 or 1 day out of the total number in 7, 14, 30 days	90

Velocity Days Since Variables: This set of variables is the number of each entity of velocity change variables out of day since variables	90
Unique Value Variables: Number of unique values for each entity	210
Cardnum Frequency Variables: Card number counts in the past 0 or 1 day out of the total number in 7, 14, 30 days	8
Amount Measurements Variables: The average, maximum and median amount for different entities in 0, 1, 3, 7, 14, 30 days	270
Unique Count Variables: The unique counts for each entity in 1, 3, 7, 14, 30 days	1050
Amount bins variable	1
Acceleration Variables: This set of variables is the number of each entity of velocity change variables out of power of 7, 14, 30 days	90

V. Feature selection

In feature selection, I used LGBM stepwise selection with 400 filters and 30 wrappers, and the performance of this set is around 0.75 at about 8 features. I kept 20 features for building models later.



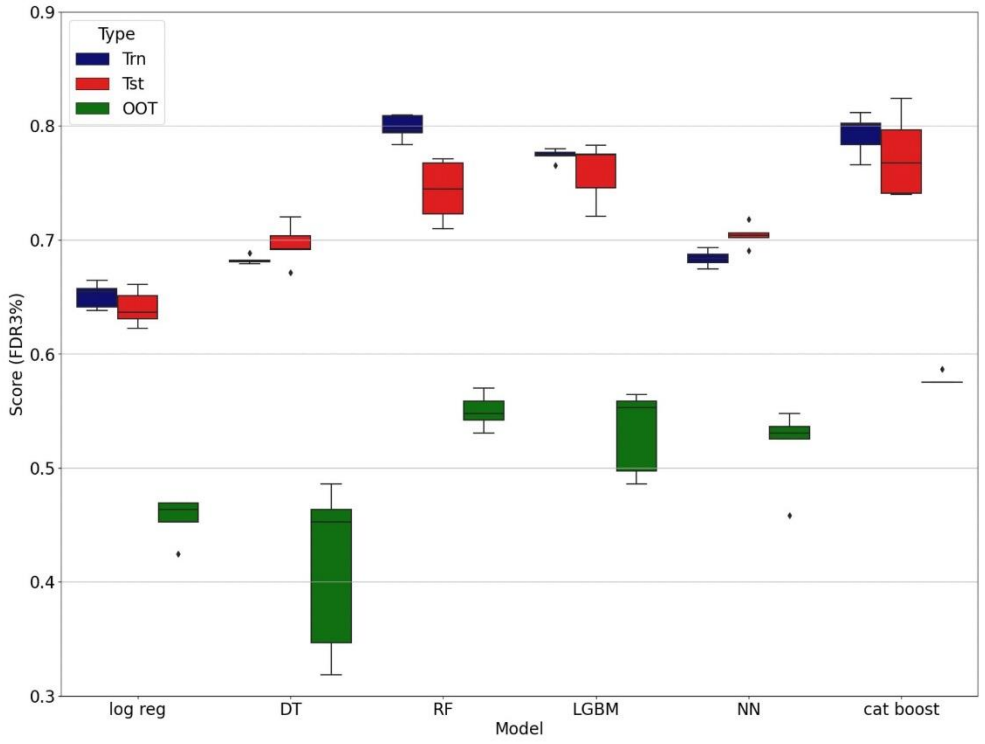
#	Variable	Filter Score
1	Cardnum_Merch description_total_3	0.647761958
2	Merch state_Merch zip_med_3	0.391859628
3	Merch description_Merch state_max_0	0.591700987
4	Merch description_max_1	0.583153831
5	Merch description_Merch state_avg_1	0.557381231
6	Merch zip_avg_7	0.463849386
7	Merch description_Merch zip_med_14	0.399856662
8	Merch state_actual/max_30	0.511324618
9	Merch zip_max_0	0.560355981
10	Merch description_max_0	0.591250867
11	Cardnum_Merch description_total_0	0.589923167
12	Merch description_Merch zip_max_0	0.58431182
13	Merchnum_max_0	0.57861836
14	Merch description_Merch zip_total_0	0.560515957
15	Merch description_avg_1	0.557201183
16	Merchnum_Merch description_total_30	0.508639997
17	Merch description_Merch state_total_30	0.508040857
18	Merch description_Merch state_max_1	0.583886884
19	Merch description_Merch zip_avg_1	0.553936664
20	Merchnum_Merch zip_max_0	0.578405217

VI. Preliminary model explores

I tried out 8 machine learning models with different parameters to get the best FDR @ 3% for OOT without overfitting or underfitting the model. In the end, I chose random forest as my final model since it performs well and get a proper OOT. It caught about 55% of fraud by only rejecting 3% of the transactions.

Model Comparison:

Model	Parameters									Average FDR at 3%		
Logistic Regression	Iteration	n_variables	penalty	C	solver	l1_ratio	max_iter			Train	Test	OOT
	1	10	l2	1	lbfgs	0.5	20			53.38%	54.56%	39.68%
	2	10	l1	0.5	lbfgs	0.5	30			54.29%	51.72%	41.12%
	3	10	l1	0.3	saga	0.8	10			56.30%	52.90%	40.57%
	4	10	elasticnet	0.7	saga	1	20			53.96%	55.54%	42.39%
	5	10	l2	0.5	liblinear	0.1	30			54.75%	54.55%	39.84%
Decision Tree	Iteration	n_variables	criterion	max_depth	min_samples_split	min_samples_leaf	max_features			Train	Test	OOT
	1	10	gini	5	50	30	3			66.33%	64.39%	42.67%
	2	10	gini	10	40	25	8			69.34%	60.50%	43.86%
	3	10	gini	15	30	20	5			69.76%	52.45%	44.29%
	4	10	gini	20	20	10	8			69.43%	52.34%	48.73%
	5	10	gini	10	20	20	10			68.29%	51.45%	47.29%
Random Forest	Iteration	n_variables	criterion	n_estimators	max_depth	min_samples_split	min_samples_leaf	max_features		Train	Test	OOT
	1	10	entropy	5	5	50	30	3		77.50%	72.18%	55.28%
	2	10	entropy	20	10	40	20	5		89.82%	79.40%	54.94%
	3	10	entropy	50	15	30	15	8		79.24%	74.58%	56.37%
	4	10	gini	80	20	20	10	8		77.38%	75.38%	56.30%
	5	10	gini	100	30	10	5	10		81.86%	76.94%	55.85%
LightGBM	Iteration	n_variables	boosting_type	max_depth	num_leaves	n_estimators	colsample_bytree	subsample	learning_rate	Train	Test	OOT
	1	10	gbdt	3	100	20	0.2	0.2	0.05	75.72%	73.72%	56.48%
	2	10	gbdt	10	200	50	0.5	0.5	0.05	80.47%	73.97%	51.15%
	3	10	GOSS	10	100	50	0.5	0.5	0.01	83.75%	73.05%	54.70%
	4	10	GOSS	20	300	70	0.8	0.5	0.01	81.73%	71.48%	56.83%
	5	10	GOSS	40	500	100	0.8	0.8	0.01	80.38%	75.68%	53.94%
Neural Network	Iteration	n_variables	hidden_layer_sizes	activation	alpha	learning_rate	solver	learning_rate_init		Train	Test	OOT
	1	10	5	logistic	0.1	constant	adam	0.01		58.43%	60.16%	56.93%
	2	10	10	logistic	0.05	adaptive	lbfgs	0.005		75.38%	71.35%	55.39%
	3	10	(10,10,10)	relu	0.01	adaptive	lbfgs	0.001		77.99%	71.19%	54.96%
	4	10	(10,10,10)	relu	0.001	constant	adam	0.0005		72.75%	73.12%	55.48%
	5	10	(20,20,20)	relu	0.0001	constant	adam	0.0001		68.41%	66.49%	55.73%
GBC	Iteration	n_variables	learning_rate	subsample	max_depth	n_estimators	min_samples_split	min_samples_leaf		Train	Test	OOT
	1	10	0.1	1	2	5	2	1		63.89%	63.36%	55.73%
	2	10	0.3	0.2	5	15	8	3		52.69%	47.61%	55.93%
	3	10	0.05	0.8	20	20	3	1		80.35%	81.50%	54.76%
	4	10	0.01	0.5	10	15	3	1		84.35%	79.33%	56.29%
	5	10	0.01	0.8	10	10	5	2		85.27%	80.24%	53.52%
Catboost	Iteration	n_variables	verbose	max_depth	iterations	l2_leaf_reg	learning_rate	bootstrap_type		Train	Test	OOT
	1	10	0	2	5	1	0.01	Bayesian		65.38%	46.33%	53.23%
	2	10	2	5	20	3	0.1	Bayesian		71.15%	66.34%	54.24%
	3	10	5	8	20	5	0.5	Bayesian		77.33%	68.34%	53.70%
	4	10	8	10	30	10	0.05	Bernoulli		72.24%	63.84%	51.86%
	5	10	10	5	20	15	0.1	Bernoulli		70.78%	65.85%	53.56%
XGBoost	Iteration	n_variables	booster	max_depth	n_estimators	min_child_weight	colsample_bytree	subsample	eta	Train	Test	OOT
	1	10	gbtree	2	5	1	1	1	0.2	67.64%	66.16%	56.38%
	2	10	gbtree	5	20	10	0.8	0.8	0.2	81.53%	80.23%	55.00%
	3	10	gbtree	20	50	20	0.5	0.7	0.4	87.39%	81.59%	55.76%
	4	10	dart	10	30	10	0.7	0.7	0.3	88.34%	83.38%	54.99%
	5	10	dart	20	80	20	0.8	0.8	0.3	88.63%	84.59%	54.80%



VII. Final model performance

In the following 3 tables, they displayed the testing, train and OOT data performance of the final model. I chose random forest as my final model.

model = RandomForestClassifier(criterion='entropy',n_estimators=20,max_depth=10,min_samples_split=40,min_samples_leaf=20,max_features=5)

Test:

bin	#recs	#g	#b	%g	%b	tot	cg	cb	%cg	FDR	KS	FPR
0	0	0	0	0	0	0	0	0	0	0	0	0
1	253	117	136	46.24506	53.75494	253	117	136	0.467215	54.83871	54.37149	0.860294
2	253	219	34	86.56126	13.43874	506	336	170	1.341746	68.54839	67.20664	1.976471
3	253	242	11	95.65217	4.347826	759	578	181	2.308122	72.98387	70.67575	3.19337
4	253	241	12	95.25692	4.743083	1012	819	193	3.270506	77.82258	74.55208	4.243523
5	252	248	4	98.4127	1.587302	1264	1067	197	4.260842	79.43548	75.17464	5.416244
6	253	250	3	98.81423	1.185771	1517	1317	200	5.259165	80.64516	75.386	6.585
7	253	248	5	98.02372	1.976285	1770	1565	205	6.249501	82.66129	76.41179	7.634146
8	253	251	2	99.20949	0.790514	2023	1816	207	7.251817	83.46774	76.21592	8.772947
9	253	251	2	99.20949	0.790514	2276	2067	209	8.254133	84.27419	76.02006	9.889952
10	253	251	2	99.20949	0.790514	2529	2318	211	9.256449	85.08065	75.8242	10.98578
11	253	253	0	100	0	2782	2571	211	10.26675	85.08065	74.81389	12.18483
12	253	253	0	100	0	3035	2824	211	11.27705	85.08065	73.80359	13.38389
13	253	253	0	100	0	3288	3077	211	12.28736	85.08065	72.79329	14.58294
14	253	253	0	100	0	3541	3330	211	13.29766	85.08065	71.78299	15.78199
15	253	252	1	99.60474	0.395257	3794	3582	212	14.30397	85.48387	71.1799	16.89623
16	252	249	3	98.80952	1.190476	4046	3831	215	15.2983	86.69355	71.39525	17.8186
17	253	249	4	98.41897	1.581028	4299	4080	219	16.29263	88.30645	72.01382	18.63014
18	253	253	0	100	0	4552	4333	219	17.30293	88.30645	71.00352	19.78539
19	253	251	2	99.20949	0.790514	4805	4584	221	18.30525	89.1129	70.80766	20.74208
20	253	251	2	99.20949	0.790514	5058	4835	223	19.30756	89.91935	70.61179	21.68161
21	253	253	0	100	0	5311	5088	223	20.31787	89.91935	69.60149	22.81614
22	253	252	1	99.60474	0.395257	5564	5340	224	21.32418	90.32258	68.99841	23.83929
23	253	252	1	99.60474	0.395257	5817	5592	225	22.33048	90.72581	68.39532	24.85333
24	253	253	0	100	0	6070	5845	225	23.34079	90.72581	67.38502	25.97778
25	252	251	1	99.60317	0.396825	6322	6096	226	24.3431	91.12903	66.78593	26.97345

Train:

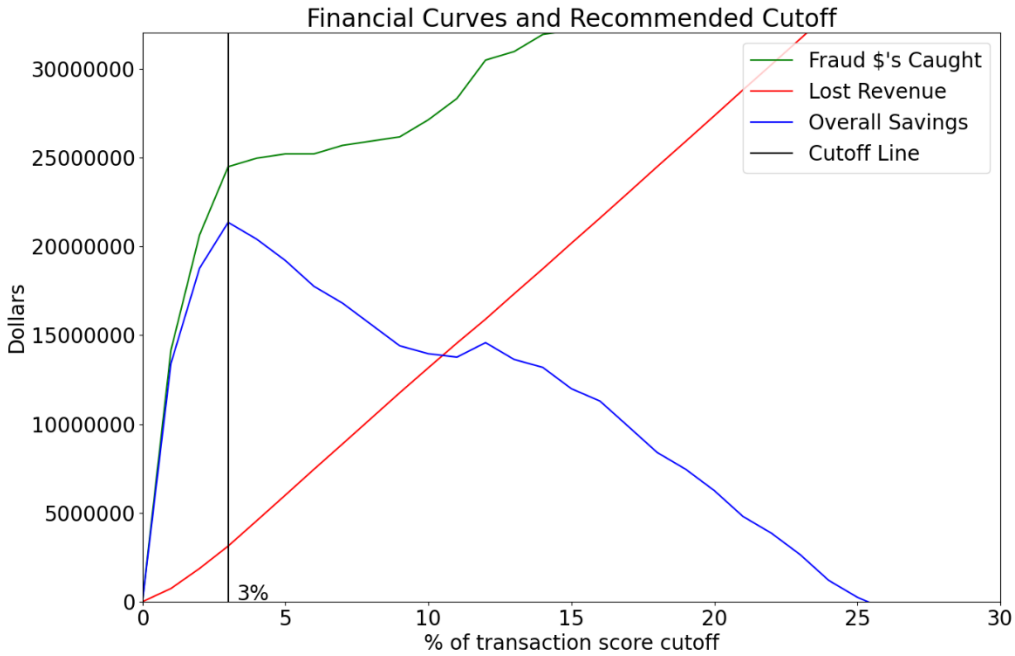
bin	#recs	#g	#b	%g	%b	tot	cg	cb	%cg	FDR	KS	FPR
0	0	0	0	0	0	0	0	0	0	0	0	0
1	590	212	378	35.9322	64.0678	590	212	378	0.363151	59.81013	59.44698	0.560847
2	590	489	101	82.88136	17.11864	1180	701	479	1.200795	75.79114	74.59034	1.463466
3	590	549	41	93.05085	6.949153	1770	1250	520	2.141218	82.27848	80.13726	2.403846
4	590	566	24	95.9322	4.067797	2360	1816	544	3.110761	86.07595	82.96519	3.338235
5	590	577	13	97.79661	2.20339	2950	2393	557	4.099147	88.13291	84.03376	4.29623
6	591	583	8	98.64636	1.353638	3541	2976	565	5.097811	89.39873	84.30092	5.267257
7	590	578	12	97.9661	2.033898	4131	3554	577	6.08791	91.29747	85.20956	6.159445
8	590	583	7	98.81356	1.186441	4721	4137	584	7.086574	92.40506	85.31849	7.083904
9	590	587	3	99.49153	0.508475	5311	4724	587	8.092089	92.87975	84.78766	8.0477
10	590	585	5	99.15254	0.847458	5901	5309	592	9.094179	93.67089	84.57671	8.967905
11	590	587	3	99.49153	0.508475	6491	5896	595	10.0997	94.14557	84.04587	9.909244
12	590	588	2	99.66102	0.338983	7081	6484	597	11.10692	94.46203	83.3551	10.86097
13	590	586	4	99.32203	0.677966	7671	7070	601	12.11073	95.09494	82.98421	11.76373
14	590	589	1	99.83051	0.169492	8261	7659	602	13.11967	95.25316	82.1335	12.72259
15	591	588	3	99.49239	0.507614	8852	8247	605	14.1269	95.72785	81.60095	13.6314
16	590	588	2	99.66102	0.338983	9442	8835	607	15.13413	96.0443	80.91018	14.55519
17	590	586	4	99.32203	0.677966	10032	9421	611	16.13793	96.67722	80.53929	15.41899
18	590	590	0	100	0	10622	10011	611	17.14858	96.67722	79.52863	16.38462
19	590	590	0	100	0	11212	10601	611	18.15924	96.67722	78.51798	17.35025
20	590	590	0	100	0	11802	11191	611	19.16989	96.67722	77.50732	18.31588
21	590	589	1	99.83051	0.169492	12392	11780	612	20.17883	96.83544	76.65661	19.24837
22	590	589	1	99.83051	0.169492	12982	12369	613	21.18778	96.99367	75.80589	20.17781
23	590	589	1	99.83051	0.169492	13572	12958	614	22.19672	97.1519	74.95518	21.10423
24	590	587	3	99.49153	0.508475	14162	13545	617	23.20223	97.62658	74.42435	21.953
25	590	589	1	99.83051	0.169492	14752	14134	618	24.21118	97.78481	73.57363	22.87055

OOT:

bin	#recs	#g	#b	%g	%b	tot	cg	cb	%cg	FDR	KS	FPR
0	0	0	0	0	0	0	0	0	0	0	0	0
1	121	62	59	51.23967	48.76033	121	62	59	0.520222	32.96089	32.44067	1.050847
2	121	94	27	77.68595	22.31405	242	156	86	1.308944	48.04469	46.73575	1.813953
3	121	105	16	86.77686	13.22314	363	261	102	2.189965	56.98324	54.79328	2.558824
4	121	119	2	98.34711	1.652893	484	380	104	3.188454	58.10056	54.9121	3.653846
5	121	120	1	99.17355	0.826446	605	500	105	4.195335	58.65922	54.46388	4.761905
6	121	121	0	100	0	726	621	105	5.210606	58.65922	53.44861	5.914286
7	121	119	2	98.34711	1.652893	847	740	107	6.209095	59.77654	53.56744	6.915888
8	121	120	1	99.17355	0.826446	968	860	108	7.215976	60.3352	53.11922	7.962963
9	121	120	1	99.17355	0.826446	1089	980	109	8.222856	60.89385	52.671	8.990826
10	121	117	4	96.69421	3.305785	1210	1097	113	9.204565	63.12849	53.92393	9.707965
11	121	116	5	95.86777	4.132231	1331	1213	118	10.17788	65.92179	55.74391	10.27966
12	121	112	9	92.56198	7.438017	1452	1325	127	11.11764	70.94972	59.83208	10.43307
13	121	119	2	98.34711	1.652893	1573	1444	129	12.11613	72.06704	59.95091	11.1938
14	121	117	4	96.69421	3.305785	1694	1561	133	13.09784	74.30168	61.20384	11.73684
15	121	120	1	99.17355	0.826446	1815	1681	134	14.10472	74.86034	60.75562	12.54478
16	121	118	3	97.52066	2.479339	1936	1799	137	15.09481	76.53631	61.4415	13.13139
17	120	120	0	100	0	2056	1919	137	16.10169	76.53631	60.43462	14.0073
18	121	121	0	100	0	2177	2040	137	17.11697	76.53631	59.41935	14.89051
19	121	119	2	98.34711	1.652893	2298	2159	139	18.11546	77.65363	59.53818	15.53237
20	121	120	1	99.17355	0.826446	2419	2279	140	19.12234	78.21229	59.08995	16.27857
21	121	121	0	100	0	2540	2400	140	20.13761	78.21229	58.07468	17.14286
22	121	119	2	98.34711	1.652893	2661	2519	142	21.1361	79.32961	58.19351	17.73944
23	121	120	1	99.17355	0.826446	2782	2639	143	22.14298	79.88827	57.74529	18.45455
24	121	121	0	100	0	2903	2760	143	23.15825	79.88827	56.73002	19.3007
25	121	119	2	98.34711	1.652893	3024	2879	145	24.15674	81.00559	56.84885	19.85517

VIII. Financial curves and recommended cutoff

I recommend a score cutoff at 3%, and anticipated an overall savings at about $21.5 million/year.



IX. Summary

In this credit card transaction fraud detection analysis, there are three main parts of this project, data exploratory, feature selection, model selection.

First of all, in data exploratory, I first did a data quality report to record the overall view of this dataset and show what the original dataset in each column looked like. Then I did data cleansing to eliminate one outlier that used different currency, which caused the value to be extremely larger than other transactions. For this project, I focus on purchase transactions only to detect fraud in those transactions. Therefore, I eliminated those that are not in purchase category. Finally, I filled in the null values using the mode of different corresponding fields and marked those without any corresponding values as unknown to take all the data into prediction.

Secondly, in feature selection process, I created comprehensive variables to have higher possibility in getting more accuracy in predicting fraud transactions. I created variables such as those put fluctuations of transactions counts into consideration and those calculated the measures of transactions in each entity. These will help detect fraud in more detailed way. Then, in feature selection, I used different numbers of selection and different selection methods such as LGBM or random forest with forward and backward selection to try out which set of method performed the best. In the end, I chose LGBM stepwise selection with 400 filters and 30 wrappers, which had the best performance around 0.75. I kept 20 features left for later modeling.

Last but not least, in model selection, I tried out 8 machine learning models with different parameters to get the optimized model, and chose random forest as final model. It caught about 55% of fraud by only rejecting 3% of the transactions. For this model, I recommend the score cutoff point at 3%, which anticipated an overall savings at about $21.5 million/year.

In conclusion, I use random forest as my final model for this transaction fraud detection project. It has a 55% FDR at 3% and an estimated overall savings of $21.5 million per year.

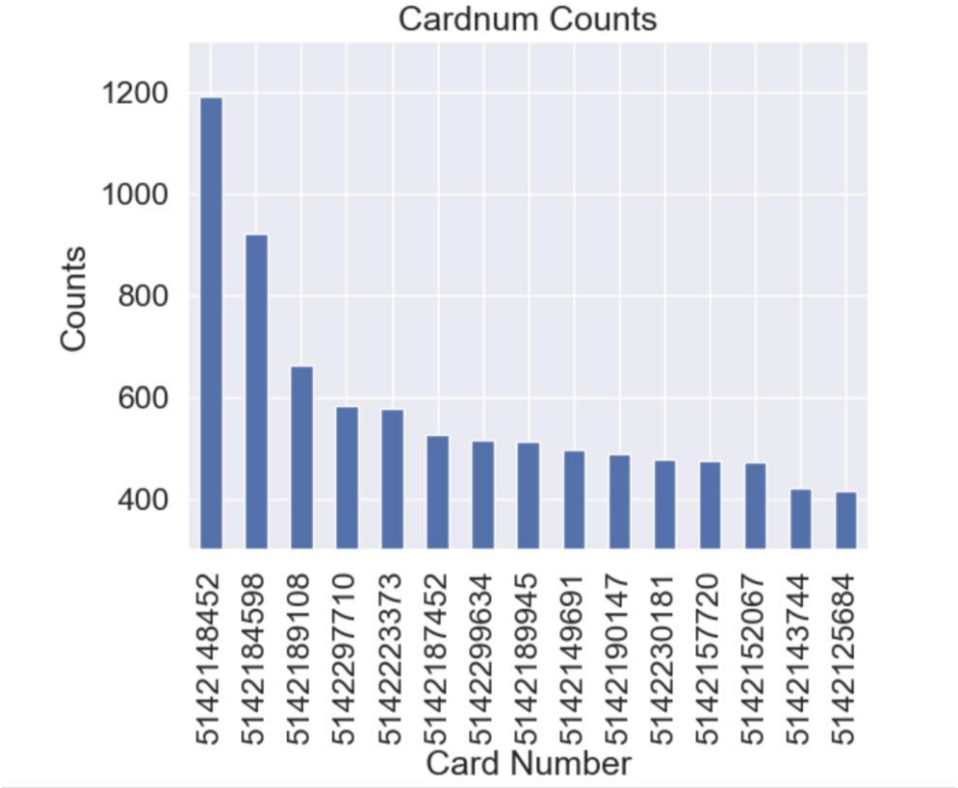
X. Appendix

1. Field Name: Recnum

   Description: The serial number of the record number, starts from 1 to 96,753.

2. Field Name: Cardnum

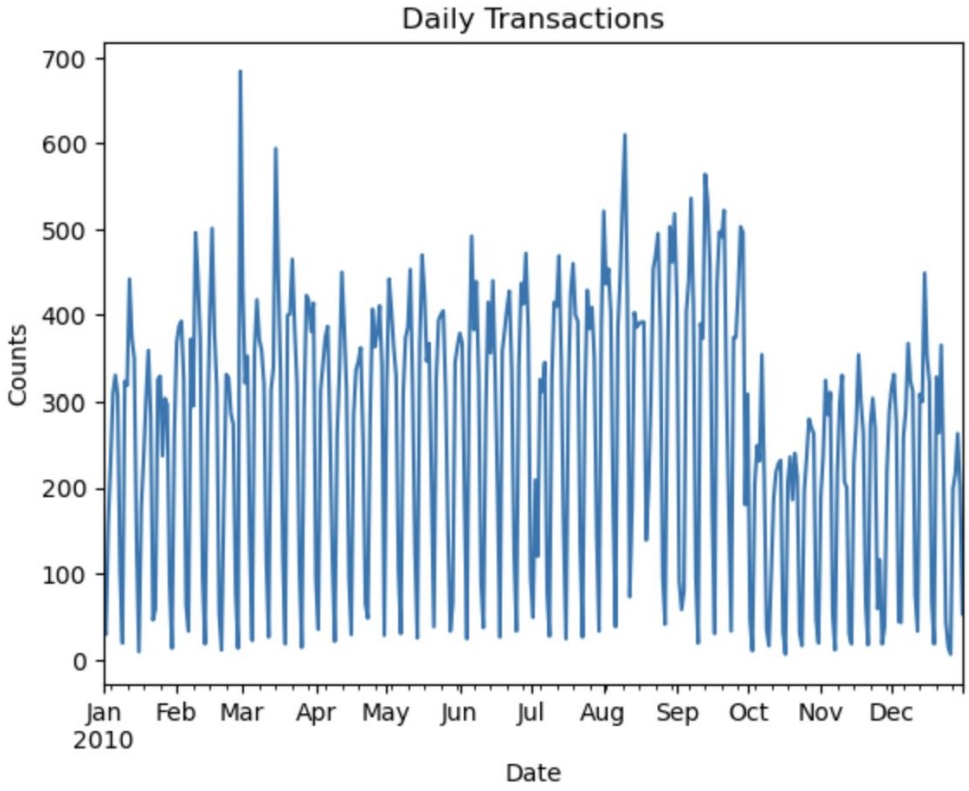
   Description: Credit card numbers. The bar chart below displays top 15 credit card numbers that had the most transaction records. The most common number is 5142148452, which has 1192 counts in total.

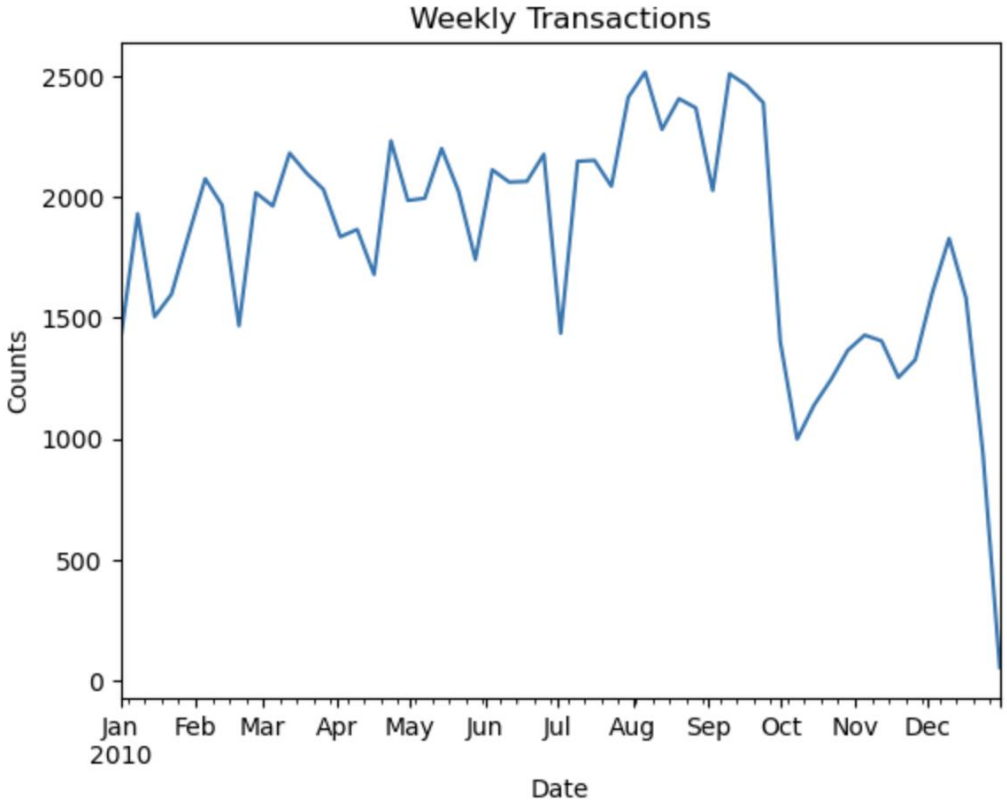


3. Field Name: Date

Description: Transaction date. Transactions from 2010-01-01 to 2010-12-31.

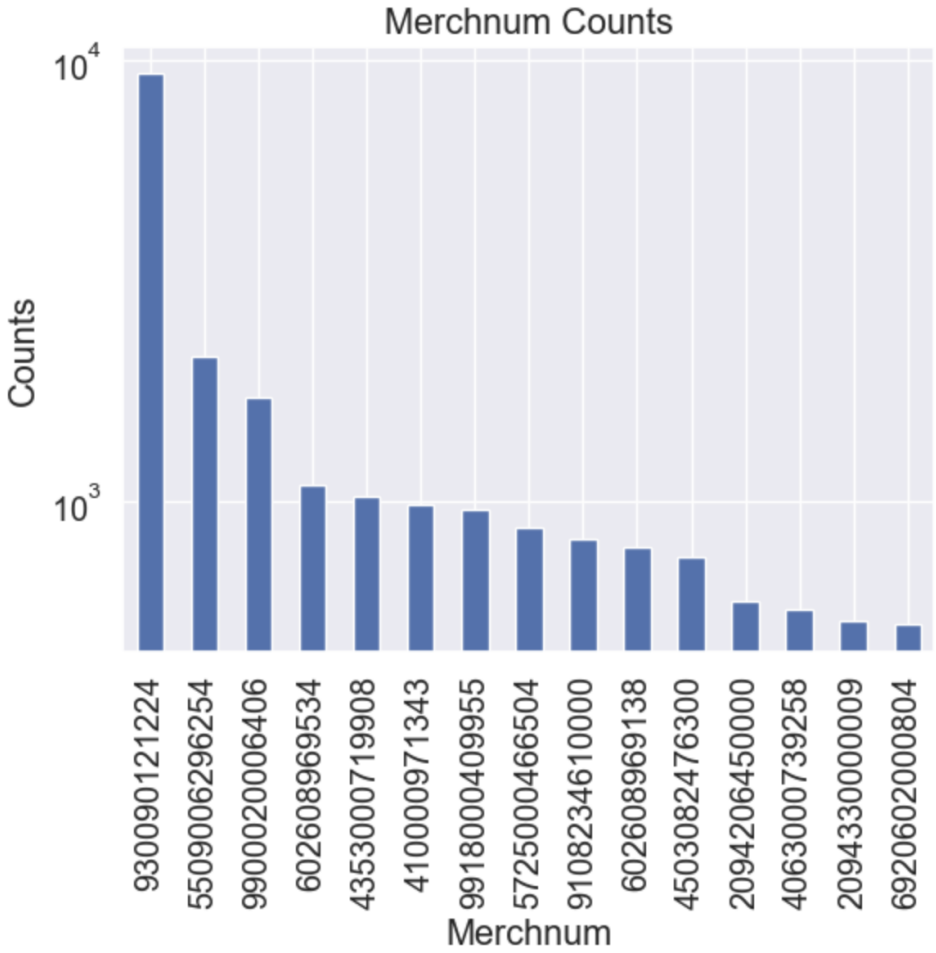
The charts below display the counts of daily and weekly applications.





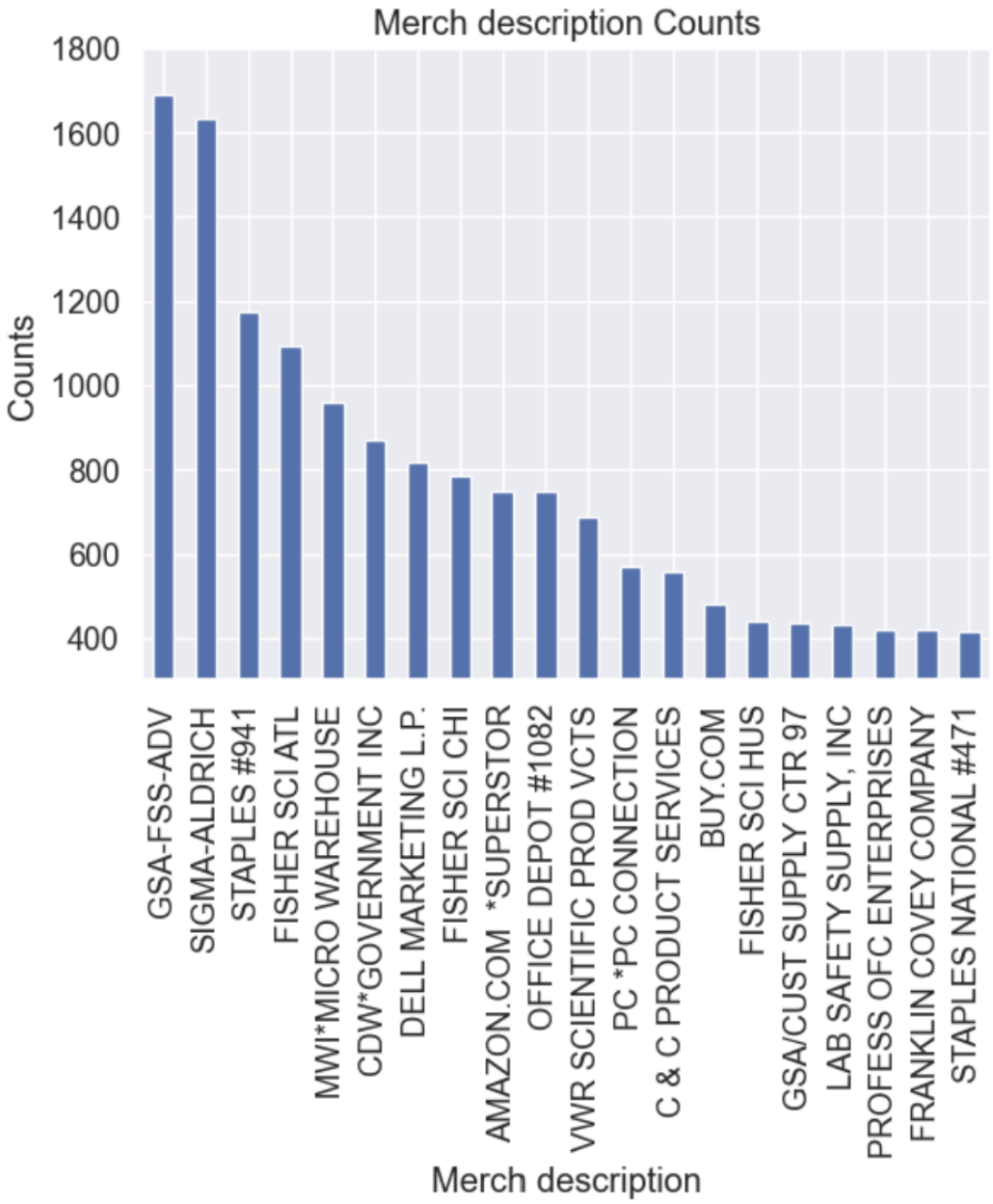
4. Field Name: Merchnum

Description: The corresponding number for merchants. The bar chart below displays top 15 merchant numbers. The most common merchant number is 930090121224, which has 9310 counts in total.



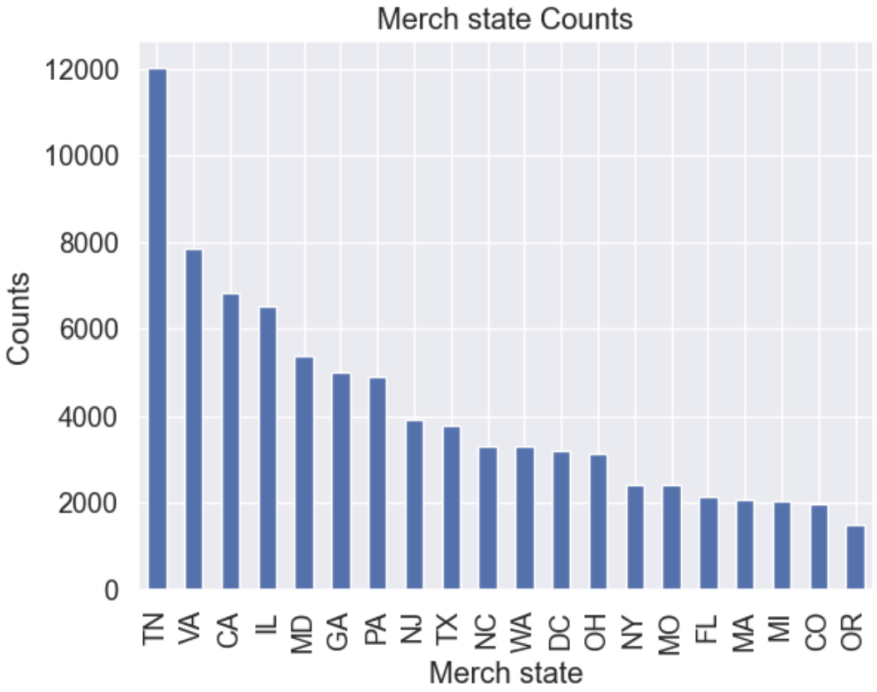
5. Field Name: Merch description

Description: Merchant's code name. The bar chart below displays top 20 merchant's names. The most common Merchant's name is GSA-FSS-ADV, which has 1688 counts in total.



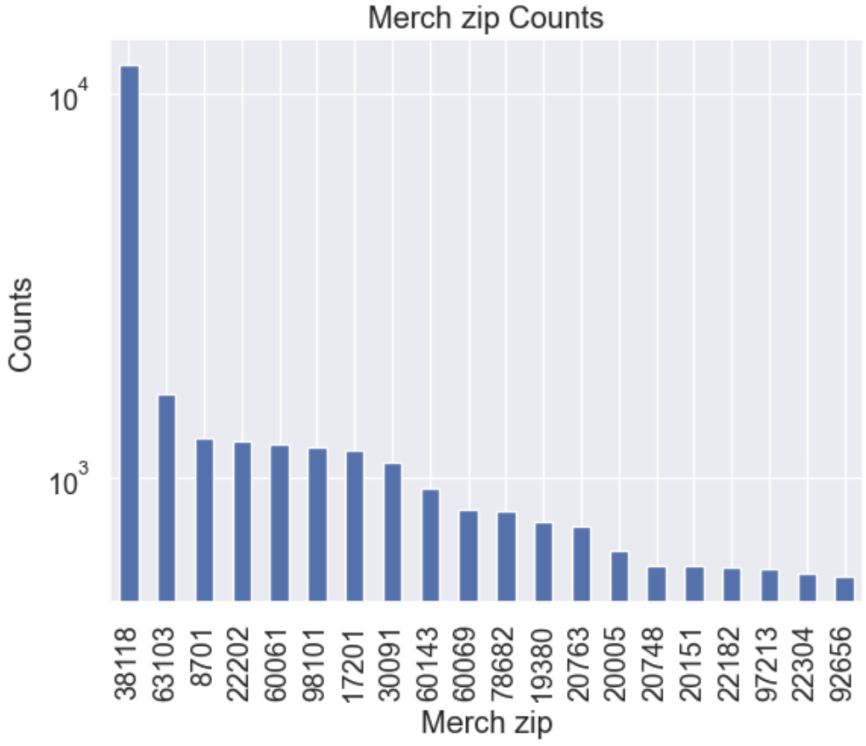
6. Field Name: Merch state

Description: The state where merchant located. The bar chart below displays top 20 states. The most common state is TN, which has 12035 counts in total.



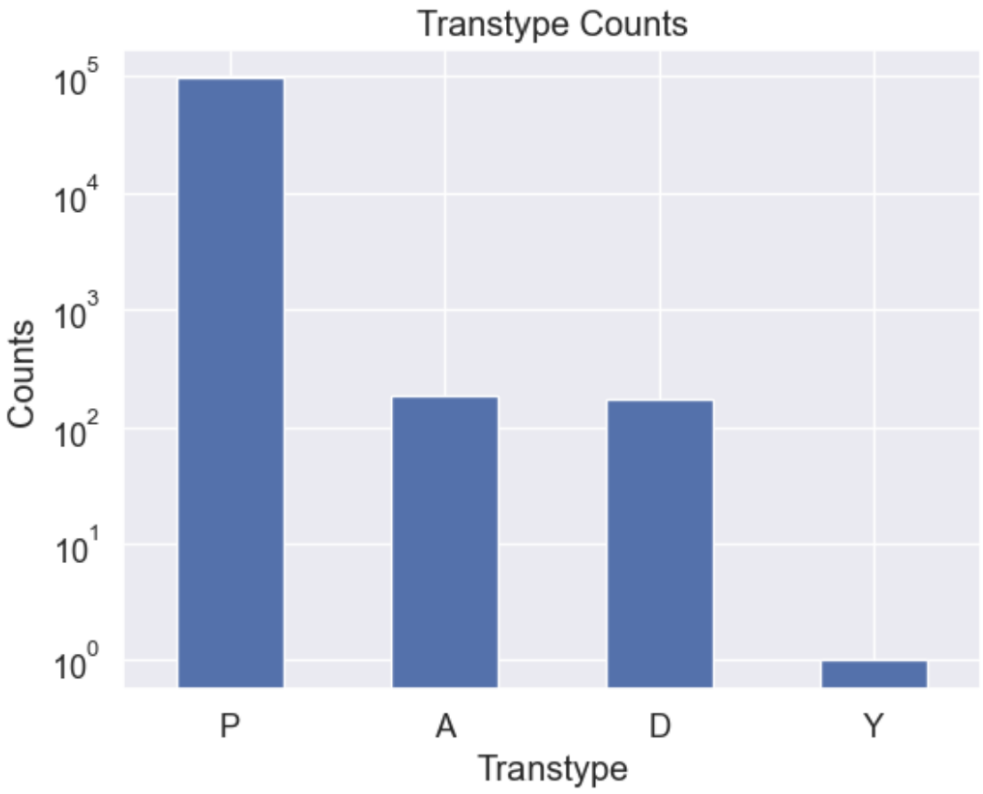
7. Field Name: Merch zip

Description: The zip code where merchant located. The bar chart below displays top 20 zip codes. The most common zip code is 38118, which has 11868 counts in total.



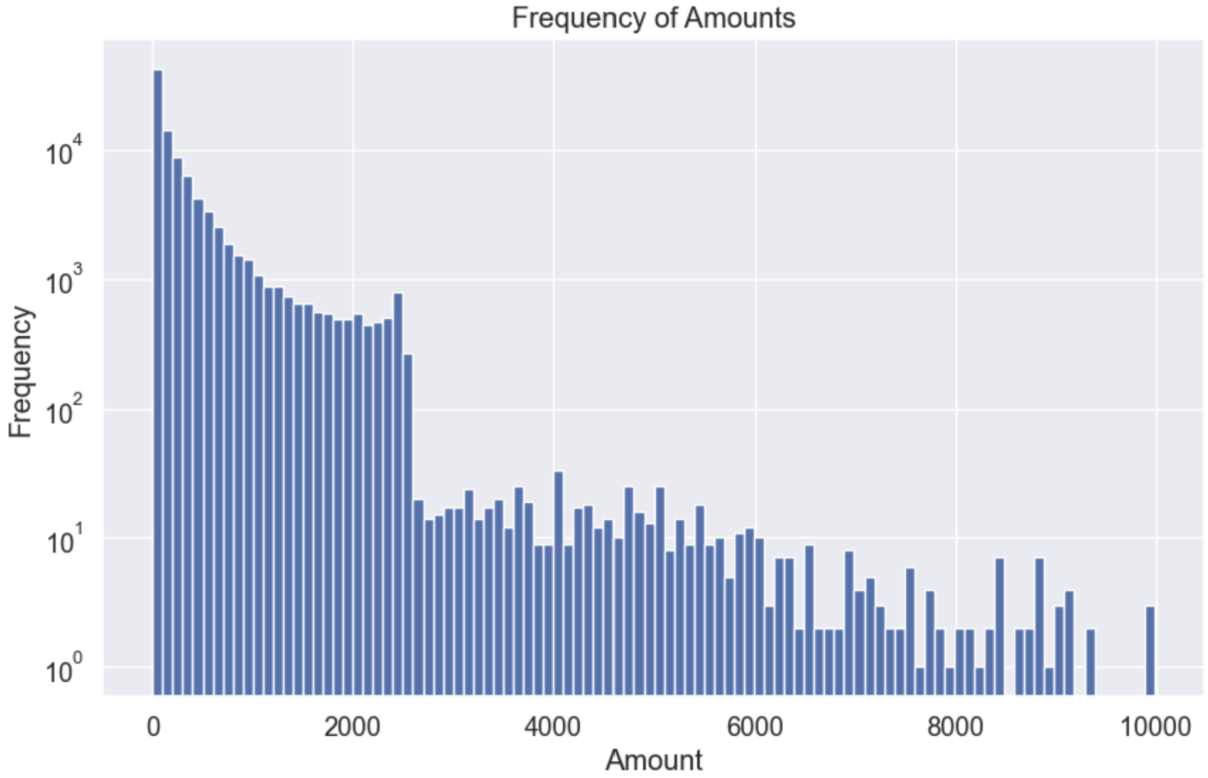
8. Field Name: Transtype

Description: Transaction type. The bar chart below displays 4 transaction types. The most common type is P (Purchase), which has 96398 counts in total.



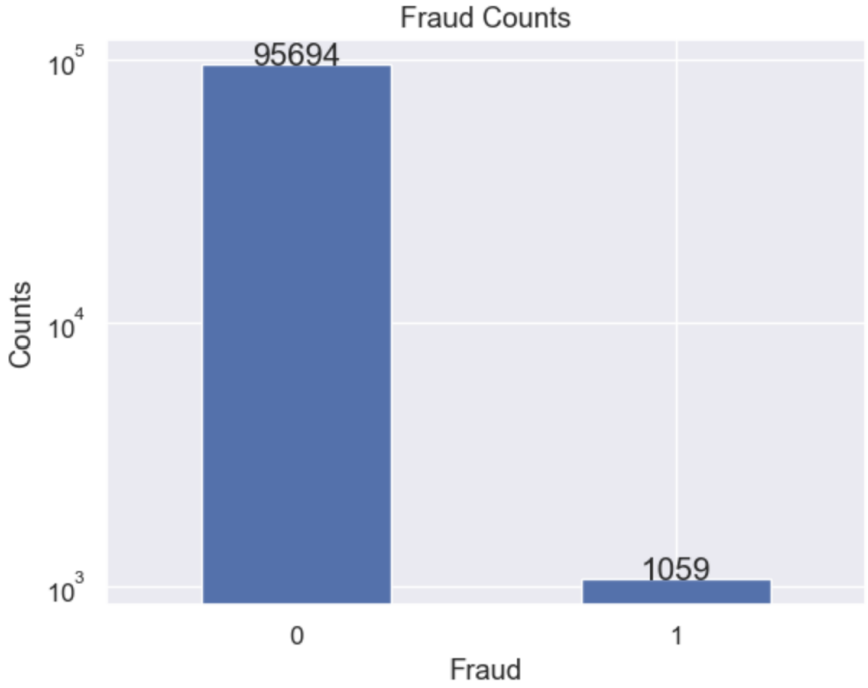
9. Field Name: Amount

Description: The transaction amount for the record. The histogram chart below displays the frequency of amounts under $10000 with 100 bins.



10. Field Name: Fraud

Description: Fraud = 0 (Normal transaction); Fraud = 1 (Fraud transaction). The counts of normal transaction are 95694, and the counts of fraud application are 1059.



Green line is normal transaction proportion of each day. Red line is fraud transaction proportion of each day.