Assignment - 5

Project 1 Report

I. Executive Summary

The problem faced by the company here is the challenge of accurately identifying fraudulent transactions while minimizing the number of legitimate transactions that are flagged as fraudulent. This report analyzes trends in credit card frauds for a U.S. government organization over the span of one year (2010) and builds a model to effectively detect fraudulent transactions. We perform data cleaning, variable creation followed by feature selection to determine our top 20 variables, and finally perform training, testing and validation to tune multiple machine learning models and select our final model. We arrive at a Random Forest model with a Fraud Detection Rate of **56.98%** at a score cutoff of 3% (FDR@3%). Using our model, we estimate a savings of about **$21 million/year** for the company with a low False Positive Rate of 2.65%.

II. Description of the Data

The dataset is **Card Transactions Data**, which contains card transactions of a U.S. government organization for business purposes. The data came from author Mark Nigrini's website and covers the company's card transactions over 1 year (2010). There are **10 fields** and **96,753 records**.

A. Summary Tables

Numeric Fields Table

Field Name	# Records With Values	% Populated	# Zeros	Min	Max	Mean	Most Common	Stdev
Date	96,753	100.00%	0	1/1/2010	12/31/2010	6/25/2010	2/28/2010	98 days 21:38
Amount	96,753	100.00%	0	0.01	3102045.53	427.88	3.62	10006.14

Categorical Fields Table

Field Name	# Records With Values	% Populated	# Zeros	# Unique Values	Most Common
Recnum	96,753	100.00%	0	96,753	1
Cardnum	96,753	100.00%	0	1,645	5142148452
Merchnum	93,378	96.51%	0	13,091	930090121224
Merch description	96,753	100.00%	0	13,126	GSA-FSS-ADV
Merch state	95,558	98.76%	0	227	TN
Merch zip	92,097	95.19%	0	4,567	38118
Transtype	96,753	100.00%	0	4	P
Fraud	96,753	100.00%	95,694	2	0

III. Data Cleaning

A. Exclusions

We are only keeping transactions with transaction type of P (Purchase) and Amount <= 3000000 to remove the outlier transaction that was incorrectly recorded in Mexican pesos and hence very large when compared to USD transactions. We know that it was in fact an actual transaction and not fraudulent.

B. Imputation Logic

We see that there are 3 columns with missing values, ‘Merchnum’, ‘Merch State’, and ‘Merch Zip’. We perform imputations for these 3 columns as follows:

Explanation for merchant number

1. First we replace merchant number 0 with null values as it is highly unlikely that a merchant number would be 0.
2. We see that total null values are now 3251
3. Next we create a data dictionary mapping merchant descriptions to merchant numbers
4. We fill in the missing merchant numbers that have merchant descriptions that using the above dictionary
5. Null values are now 2094
6. Next we assign 'unknown' for transactions that have merchant description as 'Retail Credit Adjustment' and 'Retail Debit Adjustment' as these seem to be adjustment transactions with no merchant records
7. Null values are now 1403
8. Next we count the total number of unique merchant descriptions in the remaining null values, it's 508.
9. Then we create a new merchant number for each unique merchant description and add it to our data by mapping to merchant description, each new merchant number is max(merchnum) + 1
10. Our merchant numbers are all populated now with 0 null values

Explanation for Merchant State

1. Our total null values for Merchant state are 1020
2. Next we create a data dictionary mapping zip codes that exist in the data that have no merchant state assigned to their real world values
3. We create two more data dictionaries, mapping merchant numbers and merchant descriptions to their states.
4. We use the above 3 data dictionaries to impute the values of merchant states.
5. Next we assign 'unknown' for merchant states that have merchant description as 'Retail Credit Adjustment' and 'Retail Debit Adjustment' as these seem to be adjustment transactions with no merchant records
6. The null values are now 346.
7. Next, if we have states outside of U.S. we change their merchant state to 'foreign', this could be useful as foreign transactions could be fraudulent

8. Finally we impute all remaining null values with 'unknown'
9. Our merchant state is now all populated with 0 null values

Explanation for Merchant Zip

1. Our total null values for Merchant Zip are 4300
2. We create a data dictionary mapping merchant numbers to merchant zip codes
3. We create another data dictionary mapping merchant descriptions to merchant zip codes
4. We use the above dictionaries to map missing values of merchant zips using merchant number and descriptions
5. Our null values are now 2658
6. Next we assign 'unknown' for merchant zips that have merchant description as 'Retail Credit Adjustment' and 'Retail Debit Adjustment' as these seem to be adjustment transactions with no merchant records
7. We fill the remaining zip codes as unknown
8. Our zipcode is now completely populated with 0 null values

IV. Variable Creation

A. We look at signals of how fraud usually occurs and use these signals for making variables. Some example signals of fraud can be:

1. Burst of activity at different merchants
2. Larger than normal purchase amounts, at same or different merchants
3. Card used at merchants never used before
4. Card used at a very different geography
5. Used at a high-risk merchant (online, jewelry, electronics...)
6. Increased usage in card-not-present
7. Infrequent recurring charges, same amount or same merchant

B. Below is the summary table of created variables:

Description	#Variables_Created
Day of the week: day of the week of the particular transaction	1
Day of the week target encoded: average fraud percentage of that day	1
Month: month of the particular transaction	1
Month target encoded: average fraud percentage of that month	1
State_risk target encoded: average fraud percentage of that state	1
Days Since: Number of days since a transaction with that entity was last seen	18
Velocity: Number of transactions with the same entity over the last [0,1,3,7,14,30,60] days	1134

Relative Velocity: Number of transactions with the same entities seen in the past [0,1] day divided by the number of transactions with those same entities seen in the last [7,14,30,60] days	288
Velocity Density: Number of transactions with the same entities seen in the past [0,1] day divided by the number of days since a transaction with those same entities in the last [7,14,30,60] days	144
Transaction Amount Variability: Maximum, median, and mean of amount differences between current transaction and transaction seen [0,1,3,7,14,30] days ago while grouping transactions by each entity.	324
Counts by entity: Number of unique entities for a particular field over the last [1,3,7,14,30,60] days	1836
Relative Velocity (square divided): Number of transactions with the same entities seen in the past [0,1] day divided by the number of transactions with those same entities seen in the last [7,14,30,60] days. This result is further divided by the square of [7,14,30,60] days.	144
Amount category: Divide amount column into 5 equal sized bins and assign a label (1,5) to each bin	1

C. Explanation of Target Encoding Variables

Explanation for Target encoded variable "Day of the week"

1. We create a categorical variable dow (day of the week) using the date record for each transaction
2. Next, to convert this variable to a numerical one using target encoding we use training data and remove the last 2 months of transactions (Out of time validation). This is to replicate a real life scenario where we use past data to train our model and then use it to make predictions on current or future transactions and this prevents overfitting.
3. We use a smoothing formula to target encode this variable (this also helps prevent overfitting).

Explanation for Target encoded variable "Month of the transaction"

1. We created a categorical variable month of the transaction using the date record for each transaction
2. Next, to convert this variable to a numerical one using target encoding we used a smoothing formula to target encode this variable (this helps prevent overfitting). Note that using OOT training data here was not suitable as values for month of november and december would not be calculated in that case.

Explanation for Merchant state, Card Number, and Merchant number

1. Next we target encode merchant state using the smoothing formula for target encoding and plot the top 15 fraud merchant states against fraud rate for the population and plot baseline fraud average (using mean of the now numerical merchant state)
2. We calculate the target encoded values for card number using the smoothing formula for target encoding but we don't include it in our data as it overfits (in the plot we can see that the baseline is close to 0 and all values for card numbers lie above it). A possible reason for this could be that we don't have statistically significant samples in each category of card numbers to prevent overfitting and develop good smoothing values.
3. We calculate the target encoded values for merchant numbers using the smoothing formula for target encoding but we don't include it in our data as it overfits. A possible reason for this could be that we don't have statistically significant samples in each category of merchant numbers to prevent overfitting and develop good smoothing values.

V. Feature Selection

After our previous step, we end up with close to 4000 variables but not all of them will be relevant. Hence, we perform feature selection which is a process of selecting a subset of relevant features (variables, attributes) from a larger set of features in a dataset that will be used as input for our machine learning models. Our goal now is to improve the accuracy and efficiency of the model by reducing the number of irrelevant, redundant, or noisy features that may negatively impact the model's performance. This also helps prevent overfitting, leads to faster model training and makes the results more interpretable.

A. We are going to perform feature selection using the following 2 methods:
Filtering - Measures correlation of predictor variables to the response variable and picks the top variables based on a score, using KS statistic (Kolmogorov-Smirnov) for this. We remove the variables with low KS.
Wrapper - Wrapper method is a feature selection technique that involves selecting features based on how well they improve the performance of a specific machine learning model. In this approach, the feature selection process is treated as a search problem, where different subsets of features are evaluated using a machine learning algorithm, and the best subset is selected based on the performance of the model.

B. The Process

1. **Selecting Algorithms for Wrapper Method** - We ran Feature Selection making the filter process select 300 variables and then applying wrapper to get 20 variables for 2 different wrapper algorithms, Random Forest and LightGBM to test which one performs better. We saw that LightGBM provides better performance than Random Forest and hence we used that for our next round of trials with different number of features selected in filter and wrapper. We also prefer **LightGBM** because it is not stochastic in nature and always outputs the same result unlike random forest where we will need multiple runs to see which variables to keep finally. We also see that **Forward Selection** performs better than Backward Selection.

2. **Selecting optimal number of variables that Filter and Wrapper will output** - We then ran Feature selection for different values of number of features selected after filtering and wrapping: (The algorithms used for wrapper are LGBM and forward selection)

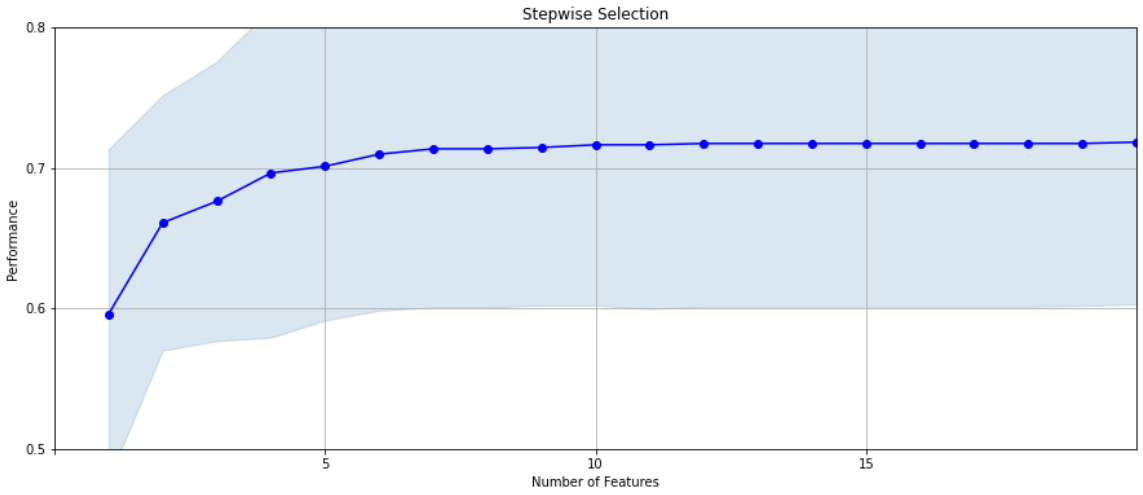
1. **First model** - Number of features - 200 (7% of the total number of variables), number of variables after wrapping - 20
2. **Second model** - Number of features - 300 (10% of the total number of variables), number of variables after wrapping - 30
3. **Third mode**l - Number of features - 400 (14% of the total number of variables), number of variables after wrapping - 20
4. **Fourth model** - Number of features - 300 (10% of the total number of variables), number of variables after wrapping - 20
5. **Fifth model** - Number of features - 300 (10% of the total number of variables), number of variables after wrapping - 50

We see that the model performs better and is more stable in terms of performance when we increase the filter number of variables from 200 to 300, while keeping wrapper variables at 20. Next, on increasing the number of filter variables to 400 we don't see any improvement in performance so we can keep the filter variables at 300. We then increase the number of wrapper variables to 30 and see that the performance remains the same. We also put wrapper variables as 50 to get better insight into where exactly saturation occurs. The saturation point comes somewhere around 10 variables and we would like to be a little conservative in our final selection to account for the variability (that we get from cross-validation) so we will keep around 20 variables. **Hence our final model will have 300 variables after filtering and 20 after wrapping.**

C. List of Final Variables with their respective filter scores

Wrapper Order	Variable	Filter Score
1	Card_merch_total_14	0.630
2	Card_zip3_max_14	0.630
3	Card_merch_avg_14	0.518
4	State_des_max_1	0.524
5	Card_zip_max_60	0.605
6	Card_Merchnum_desc_ avg_30	0.519
7	Card_Merchnum_desc_avg_1	0.511
8	Card_merch_avg_14	0.515
9	Card_zip_avg_14	0.538
10	Card_ Merchdesc_Zip_avg_7	0.518
11	Card_Merchdesc_Zip_avg 30	0.523
12	Card_Merchnum_Zip_total_14	0.627
13	Card_Merchnum Zip avg 14	0.518
14	Card_merch_avg_7	0.524
15	Card_Merchnum_Zip_avg_7	0.523
16	Card_Merchnum_desc_avg_7	0.520
17	Card_Merchdeso_avg_7	0.517
18	Card_Merchnum_Zip_avg_1	0.514
19	Card_Merchdesc_Zlp_max_14	0.603
20	Card_Merchdesc_max_14	0.604

D. Plot of final variables - This is the performance of the model selected by us (300 variables after the filter and 20 after wrapper function).



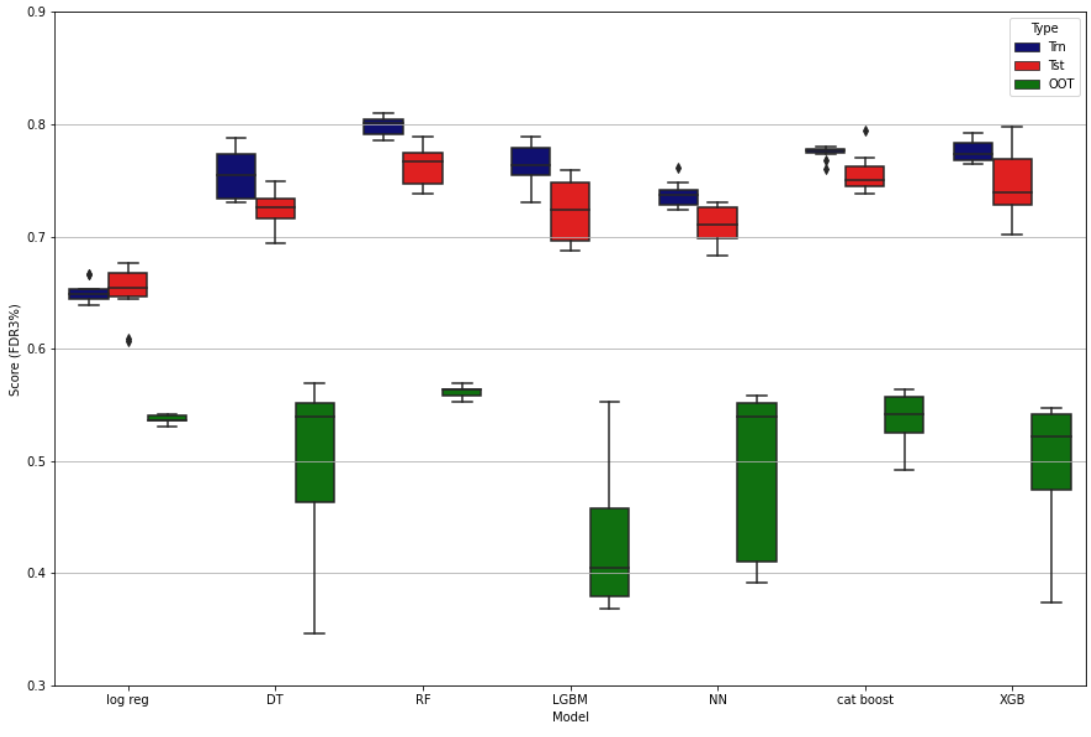
VI. Preliminary Data Exploration

We now take our final top 20 variables, z-scale the independent variables and build a variety of models like Logistic Regression, Decision Tree, Random Forest, XGBoost, Catboost, LGBM, Neural Network. By using multiple algorithms, we can compare their performance on the same dataset and select the one that works best for the problem at hand.Furthermore, using multiple algorithms can help to identify potential issues such as overfitting or underfitting. If two or more algorithms produce similar results, it is a good indication that the model is not overfitting the data, as different algorithms have different assumptions and limitations.

A. Exploration of Hyperparameters for different algorithms

Model		Parameters							Average FDR@3%		
Logistic Regression	Iteration	Penalty	class_weight	Solver	L1_Ratio	max_iter			Train	Test	OOT
	1	l2	None	lbfgs	None	100			61.79	61.57	47.2
	2	None	None	lbfgs	None	1000			62.28	60.99	47.15
	3	l1	None	liblinear	None	1000			62.26	61.5	49.66
	4	l2	Balanced	lbfgs	None	500			64.52	64.26	53.35
	5	elasticnet	None	saga	0.5	500			62.25	61.73	48.49
Decision Tree	Iteration	criterion	max_depth	min_samples_leaf	min_samples_split				Train	Test	OOT
	1	gini	10	20	40				80.31	73.03	43.85
	2	gini	8	60	120				75.68	72.41	53.57
	3	entropy	10	80	160				79.8	74.46	41.22
	4	gini	8	40	90				75.02	70.93	52.34
Random Forest	Iteration	n_estimators	max_depth	min_samples_leaf	min_samples_split				Train	Test	OOT
	1	100	8	60	120				79.68	76.82	55.86
	2	50	8	60	120				80.54	76.62	48.32
	3	20	8	50	90				80.25	76.38	49.27
LightGBM	Iteration	boosting_type	max_depth	n_estimators	num_leaves	colsample_bytree	subsample	learning_rate	Train	Test	OOT
	1	gbdt	8	15	31	1	1	0.01	78.04	73.83	45.69
	2	gbdt	5	50	100	0.8	0.8	0.01	76.96	74.6	39.05
	3	gbdt	5	20	31	0.8	0.8	0.01	74.49	73.99	47.59
	4	gbdt	10	20	31	0.8	0.8	0.01	80.83	75.92	41.22
	5	gbdt	10	20	31	1	1	0.01	79.21	75.96	42.29
	6	dart	5	20	31	0.8	0.8	0.02	75.63	73.56	48.99
XGBoost	Iteration	booster	max_depth	n_estimators	tree_method	min_child_weight	colsample_bytree	subsample	eta	Train	Test	OOT
	1	gbtree	10	20	auto	5	1	1	0.05	76.53	73.38	53.96
	2	gbtree	10	20	auto	5	0.8	0.8	0.05	79.62	75.73	55.3
	3	dart	10	20	auto	5	0.8	0.8	0.05	80.53	76.48	53.79
	4	dart	10	20	auto	5	0.8	0.8	0.08	81.42	77.45	51.06
	5	gbtree	8	15	auto	5	0.8	0.8	0.05	77.78	75.5	54.86
	6	gbtree	8	15	approx	3	0.8	0.8	0.08	78.31	76.38	48.21
CatBoost	Iteration	bootstrap_type	depth	iterations	l2_leaf_reg	random_state	learning_rate		Train	Test	OOT
	1	Bayesian	7	1000	12	10	0.01		89.83	81.41	44.97
	2	Bayesian	5	2000	8	None	0.01		84.52	78.99	46.53
	3	Bayesian	5	1500	12	10	0.01		81.9	77.31	48.21
	4	Bernoulli	5	1500	12	10	0.01		79.11	74.09	53.91
	5	MVS	5	1500	12	10	0.01		81.29	78.61	45.69
Neural Network	Iteration	activation	solver	learning_rate	learning_rate_init	hidden_layer_size	max_iter	alpha	Train	Test	OOT
	1	relu	adam	N/A	0.001	100	200	0.0001	72.83	71.38	43.12
	2	relu	sgd	constant	0.01	70	200	0.001	73.14	70.96	41.45
	3	relu	adam	adaptive	0.001	20	200	0.005	68.24	68.06	55.53
	4	relu	adam	adaptive	0.001	70	200	0.005	70.18	69.35	55.86
	5	relu	adam	constant	0.01	200	500	0.0005	73.54	71.5	37.87
	6	logistic	adam	adaptive	0.01	100	200	0.0005	73.51	73.45	49.38
	7	tanh	adam	adaptive	0.01	150	200	0.0005	75.43	71.58	37.7

B. Boxplot of all models with our best model hyperparameters



We ran all our best models for the top 10 variables after feature selection. Here, logistic regression, Random Forest and Catboost perform well on OOT data with relatively lower variation as compared to the other models. However, LR has much lower training and testing accuracy when compared to RF and Catboost. In this case, our best model performance is with **Random Forest** with the highest training, testing and OOT accuracies with very little variation.

VII. Final Model Performance

The final model is a **RandomForestClassifie**r, a popular supervised machine learning algorithm for classification tasks. The hyperparameters set for the model are:

1. **max_depth=8**: This parameter limits the depth of each decision tree in the forest to eight levels. This helps to prevent overfitting and ensures that the model does not become too complex.
2. **min_samples_split=120**: This parameter specifies the minimum number of samples required to split an internal node. If the number of samples in a node is less than this value, the node will not be split. This parameter helps to control the growth of the decision trees in the forest and prevent overfitting.
3. **min_samples_leaf=60**: This parameter specifies the minimum number of samples required to be at a leaf node. This parameter helps to ensure that the trees in the forest are not too specific to the training data and can generalize well to new data.
4. **n_estimators=100**: This parameter specifies the number of decision trees to be used in the forest. Increasing the number of trees generally leads to better performance, but also increases the computational cost. Here, 100 trees are being used.

Overall, the hyperparameters set for the model are designed to balance the model's complexity and accuracy, prevent overfitting, and ensure good generalization performance on new data.

A. Summary Tables for Training, Testing, and Validation Data

Training Data

Training	# Records	# Goods	# Bads	Fraud Rate								
	58779	58169	610	0.0103779								
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	588	272	316	46.26%	53.74%	588	272	316	0.47%	51.80%	51.34	0.86
2	588	453	135	77.04%	22.96%	1176	725	451	1.25%	73.93%	72.69	1.61
3	587	532	55	90.63%	9.37%	1763	1257	506	2.16%	82.95%	80.79	2.48
4	588	567	21	96.43%	3.57%	2351	1824	527	3.14%	86.39%	83.26	3.46
5	588	579	9	98.47%	1.53%	2939	2403	536	4.13%	87.87%	83.74	4.48
6	588	581	7	98.81%	1.19%	3527	2984	543	5.13%	89.02%	83.89	5.50
7	588	575	13	97.79%	2.21%	4115	3559	556	6.12%	91.15%	85.03	6.40
8	587	581	6	98.98%	1.02%	4702	4140	562	7.12%	92.13%	85.01	7.37
9	588	583	5	99.15%	0.85%	5290	4723	567	8.12%	92.95%	84.83	8.33
10	588	581	7	98.81%	1.19%	5878	5304	574	9.12%	94.10%	84.98	9.24
11	588	585	3	99.49%	0.51%	6466	5889	577	10.12%	94.59%	84.47	10.21
12	587	586	1	99.83%	0.17%	7053	6475	578	11.13%	94.75%	83.62	11.20
13	588	587	1	99.83%	0.17%	7641	7062	579	12.14%	94.92%	82.78	12.20
14	588	584	4	99.32%	0.68%	8229	7646	583	13.14%	95.57%	82.43	13.11
15	588	587	1	99.83%	0.17%	8817	8233	584	14.15%	95.74%	81.58	14.10
16	588	584	4	99.32%	0.68%	9405	8817	588	15.16%	96.39%	81.24	14.99
17	587	587	0	100.00%	0.00%	9992	9404	588	16.17%	96.39%	80.23	15.99
18	588	588	0	100.00%	0.00%	10580	9992	588	17.18%	96.39%	79.22	16.99
19	588	587	1	99.83%	0.17%	11168	10579	589	18.19%	96.56%	78.37	17.96
20	588	586	2	99.66%	0.34%	11756	11165	591	19.19%	96.89%	77.69	18.89

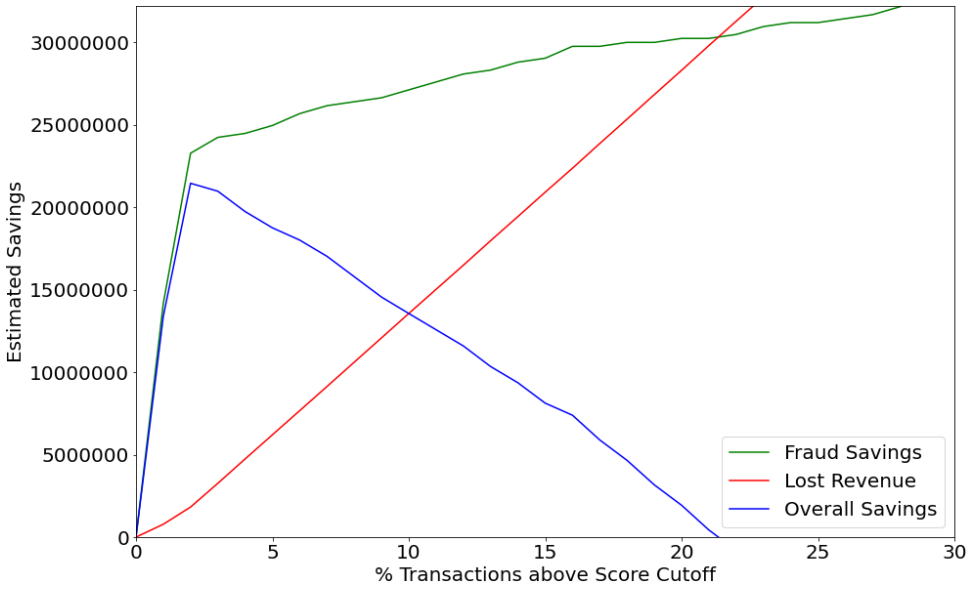
Testing Data

OOT	# Records	# Goods	# Bads	Fraud Rate								
	12427	12248	179	0.014404								
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	124	65	59	52.42%	47.58%	124	65	59	0.53%	32.96%	32.43	1.10
2	125	87	38	69.60%	30.40%	249	152	97	1.24%	54.19%	52.95	1.57
3	124	119	5	95.97%	4.03%	373	271	102	2.21%	56.98%	54.77	2.66
4	124	123	1	99.19%	0.81%	497	394	103	3.22%	57.54%	54.33	3.83
5	124	124	0	100.00%	0.00%	621	518	103	4.23%	57.54%	53.31	5.03
6	125	123	2	98.40%	1.60%	746	641	105	5.23%	58.66%	53.43	6.10
7	124	122	2	98.39%	1.61%	870	763	107	6.23%	59.78%	53.55	7.13
8	124	123	1	99.19%	0.81%	994	886	108	7.23%	60.34%	53.10	8.20
9	124	123	1	99.19%	0.81%	1118	1009	109	8.24%	60.89%	52.66	9.26
10	125	122	3	97.60%	2.40%	1243	1131	112	9.23%	62.57%	53.34	10.10
11	124	123	1	99.19%	0.81%	1367	1254	113	10.24%	63.13%	52.89	11.10
12	124	121	3	97.58%	2.42%	1491	1375	116	11.23%	64.80%	53.58	11.85
13	125	125	0	100.00%	0.00%	1616	1500	116	12.25%	64.80%	52.56	12.93
14	124	124	0	100.00%	0.00%	1740	1624	116	13.26%	64.80%	51.55	14.00
15	124	123	1	99.19%	0.81%	1864	1747	117	14.26%	65.36%	51.10	14.93
16	124	123	1	99.19%	0.81%	1988	1870	118	15.27%	65.92%	50.65	15.85
17	125	124	1	99.20%	0.80%	2113	1994	119	16.28%	66.48%	50.20	16.76
18	124	121	3	97.58%	2.42%	2237	2115	122	17.27%	68.16%	50.89	17.34
19	124	124	0	100.00%	0.00%	2361	2239	122	18.28%	68.16%	49.88	18.35
20	124	123	1	99.19%	0.81%	2485	2362	123	19.28%	68.72%	49.43	19.20

OOT Data

Testing	# Records	# Goods	# Bads	Fraud Rate								
	25191	24921	270	0.010718								
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	252	111	141	44.05%	55.95%	252	111	141	0.45%	52.22%	51.78	0.79
2	252	199	53	78.97%	21.03%	504	310	194	1.24%	71.85%	70.61	1.60
3	252	227	25	90.08%	9.92%	756	537	219	2.15%	81.11%	78.96	2.45
4	252	241	11	95.63%	4.37%	1008	778	230	3.12%	85.19%	82.06	3.38
5	252	246	6	97.62%	2.38%	1260	1024	236	4.11%	87.41%	83.30	4.34
6	251	249	2	99.20%	0.80%	1511	1273	238	5.11%	88.15%	83.04	5.35
7	252	247	5	98.02%	1.98%	1763	1520	243	6.10%	90.00%	83.90	6.26
8	252	251	1	99.60%	0.40%	2015	1771	244	7.11%	90.37%	83.26	7.26
9	252	251	1	99.60%	0.40%	2267	2022	245	8.11%	90.74%	82.63	8.25
10	252	250	2	99.21%	0.79%	2519	2272	247	9.12%	91.48%	82.36	9.20
11	252	249	3	98.81%	1.19%	2771	2521	250	10.12%	92.59%	82.48	10.08
12	252	252	0	100.00%	0.00%	3023	2773	250	11.13%	92.59%	81.47	11.09
13	252	251	1	99.60%	0.40%	3275	3024	251	12.13%	92.96%	80.83	12.05
14	252	251	1	99.60%	0.40%	3527	3275	252	13.14%	93.33%	80.19	13.00
15	252	251	1	99.60%	0.40%	3779	3526	253	14.15%	93.70%	79.55	13.94
16	252	252	0	100.00%	0.00%	4031	3778	253	15.16%	93.70%	78.54	14.93
17	251	251	0	100.00%	0.00%	4282	4029	253	16.17%	93.70%	77.54	15.92
18	252	252	0	100.00%	0.00%	4534	4281	253	17.18%	93.70%	76.53	16.92
19	252	251	1	99.60%	0.40%	4786	4532	254	18.19%	94.07%	75.89	17.84
20	252	251	1	99.60%	0.40%	5038	4783	255	19.19%	94.44%	75.25	18.76

VIII. Financial Curves and Recommended Cutoff

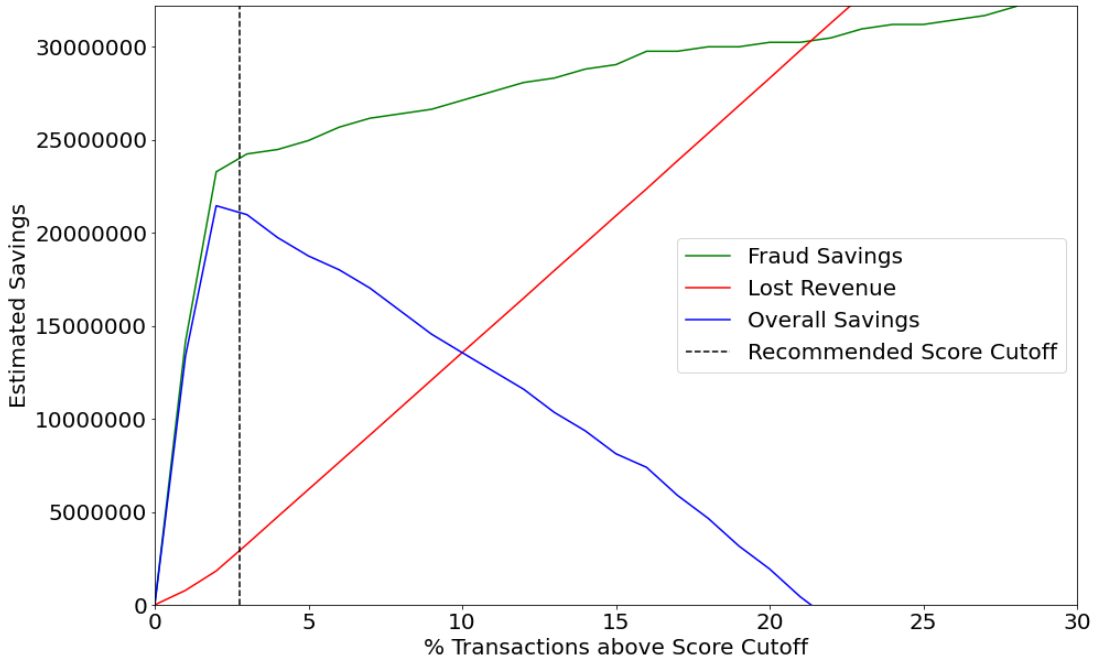


Explanation for above graph:

1. **Green curve:** Assuming $400 gain for every fraud that's caught
2. **Red curve:** Assuming $20 loss for every false positive (a good that's flagged as a bad)
3. **Blue curve:** Overall savings by the company (difference between the green curve and the red curve)
4. Basic assumption for calculation: This is based on testing on OOT data (2 months out of 12 months), so we scale our savings values by (12/2 = 6). The data available to us is also 100,000 records out of total 10 million transactions a year by the company, so we scale our savings values by (10,000,000/100,000 = 100)

Recommended Cutoff:

Maximum possible savings using our model is $21,456,000. We want to recommend a cutoff that will maximize your overall savings and at the same time keep the False Positive Rate at a minimum (reduce the number of good transactions that are denied). For this reason, we recommend a score cutoff at 3% with an anticipated savings of around $21 million per year. We show the cutoff point below for easy understanding -



IX. Summary

A. Introduction

This report analyzes the trends in credit card frauds for a U.S. government organization over the span of one year (2010) and builds a model to effectively detect fraudulent transactions. The goal is to identify fraudulent transactions while minimizing the number of legitimate transactions that are flagged as fraudulent. The report includes data cleaning, variable creation, and feature selection. We then train, test, and validate various machine learning models to select the best one. Finally, we evaluate the model's performance, estimate savings for the company, and provide recommendations.

B. Description of Data

The dataset is Card Transactions Data, which contains card transactions of a U.S. government organization for business purposes. There are 10 fields and 96,753 records covering a span of one year (2010). The variables in the dataset include Transaction Date, Average Amount/Transaction/Day, Transaction Amount, Card No, Merchant ID, Merchant Name, Merchant State, Merchant Zip, Cardholder Zip, and Fraud.

C. Data Cleaning

We excluded transactions with a transaction type other than Purchase and Amount greater than 3000000 to remove the outlier transaction that was incorrectly recorded in Mexican

pesos and hence very large when compared to USD transactions. We then performed imputations for missing values in the columns ‘Merchnum’, ‘Merch State’, and ‘Merch Zip’.

D. Variable Creation

We created variables based on signals of how fraud usually occurs. We also created target encoded variables for the day of the week and month of the transaction. These variables were created to convert categorical variables to numerical variables, which could then be used for modeling purposes. We used a smoothing formula to target encode these variables.

E. Feature Selection

We used filter and wrapper methods for feature selection. We used KS statistic to filter our top 300 variables and then for wrapper we used LGBM with forward selection model to select our top 20 variables. The variables selected were Cardnum_Last (last transaction amount), Merch_State_count_14 (number of transactions in last 14 days for a particular merchant state), Merchnum_count_14 (number of transactions in last 14 days for a particular merchant number), and several others.

F. Model Selection

We trained, tested, and validated several machine learning models, including logistic regression, decision trees, random forest, some boosting algorithms, and neural network. We selected the random forest model as our final model based on its superior performance on testing data and OOT data.

G. Recommendations

Based on our analysis, we recommend the following:

1. Continue monitoring for fraudulent transactions using the model we developed.
2. Consider implementing additional security measures such as 2-factor authentication for high-risk transactions.
3. Keep the model updated with new data to ensure its continued effectiveness in detecting fraudulent transactions.

H. Final Model Performance

Our final model is Random Forest with the following hyperparameters: max_depth=8, min_samples_split=120, min_samples_leaf=60, and n_estimators=100. The hyperparameters set for the model are designed to balance the model's complexity and accuracy, prevent overfitting, and ensure good generalization performance on new data. The final model had a Fraud Detection Rate of 56.98% at a score cutoff of 3% (FDR@3%). We estimated a savings of about $21 million/year for the company with a low false positive rate of 2.65%.

APPENDIX

Data Quality Report

1. Data Description

The dataset is **Card Transactions Data**, which contains card transactions of a U.S. government organization for business purposes. The data came from author Mark Nigrini's website and covers the company's card transactions over 1 year (2010). There are **10 fields** and **96,753 records**.

2. Summary Tables

Numeric Fields Table

Field Name	# Records With Values	% Populated	# Zeros	Min	Max	Mean	Most Common	Stdev
Date	96,753	100.00%	0	1/1/2010	12/31/2010	6/25/2010	2/28/2010	98 days 21:38
Amount	96,753	100.00%	0	0.01	3102045.53	427.88	3.62	10006.14

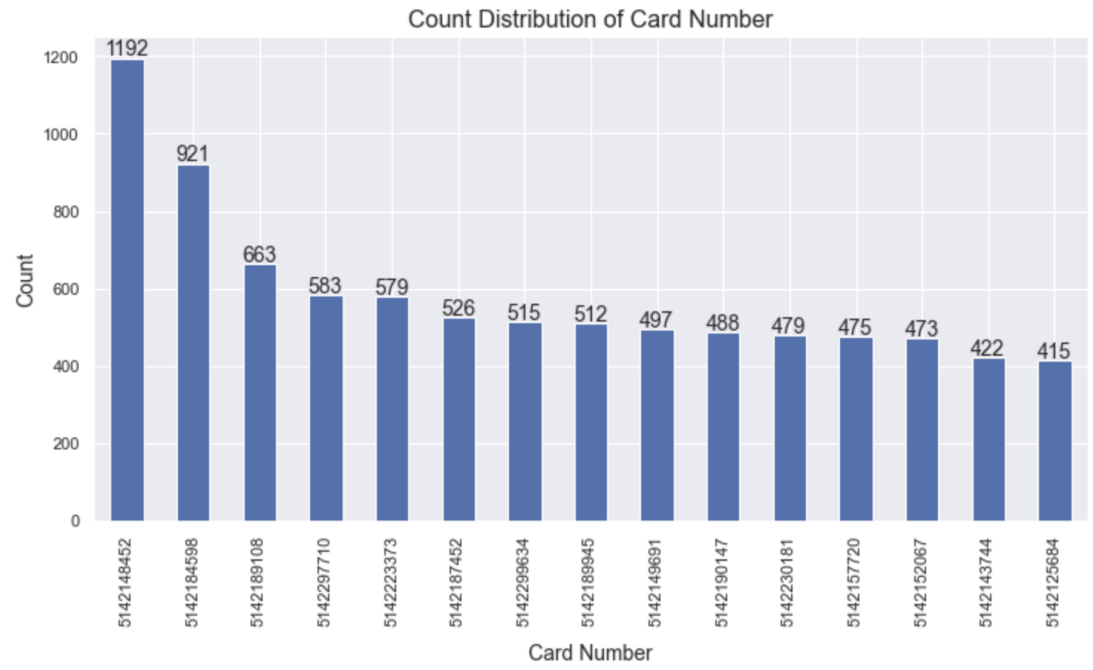
Categorical Fields Table

Field Name	# Records With Values	% Populated	# Zeros	# Unique Values	Most Common
Recnum	96,753	100.00%	0	96,753	1
Cardnum	96,753	100.00%	0	1,645	5142148452
Merchnum	93,378	96.51%	0	13,091	930090121224
Merch description	96,753	100.00%	0	13,126	GSA-FSS-ADV
Merch state	95,558	98.76%	0	227	TN
Merch zip	92,097	95.19%	0	4,567	38118
Transtype	96,753	100.00%	0	4	P
Fraud	96,753	100.00%	95,694	2	0

3. Visualization of Each Field

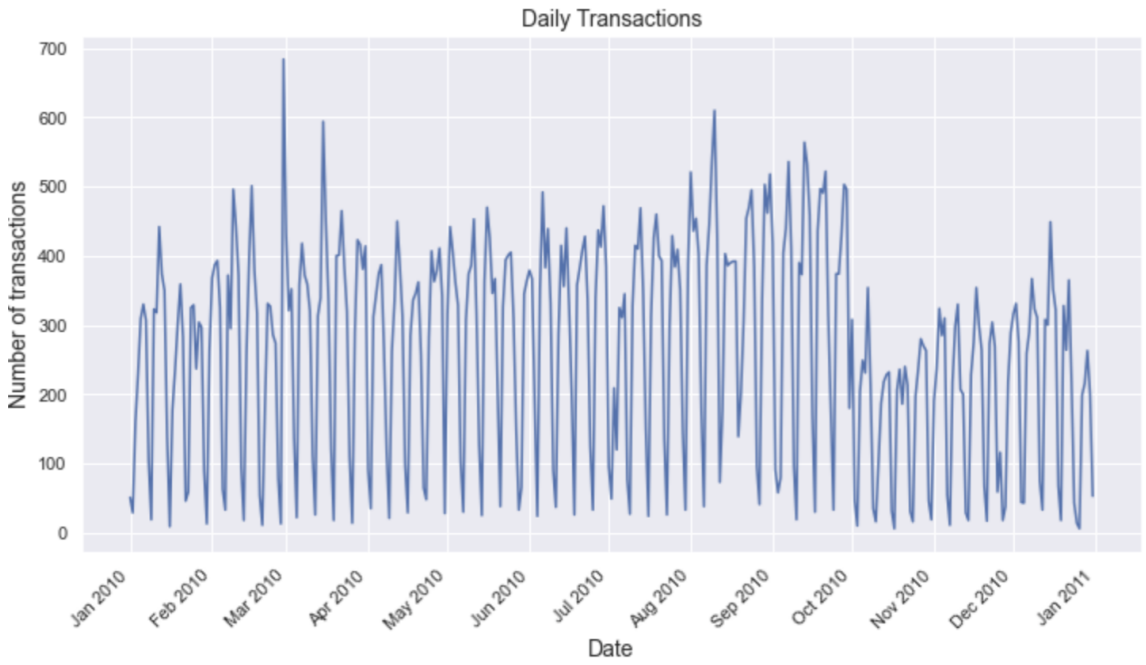
1. **Field Name: Recnum**
   Description: Ordinal unique positive integer for each application record, from 1 to 96,753.

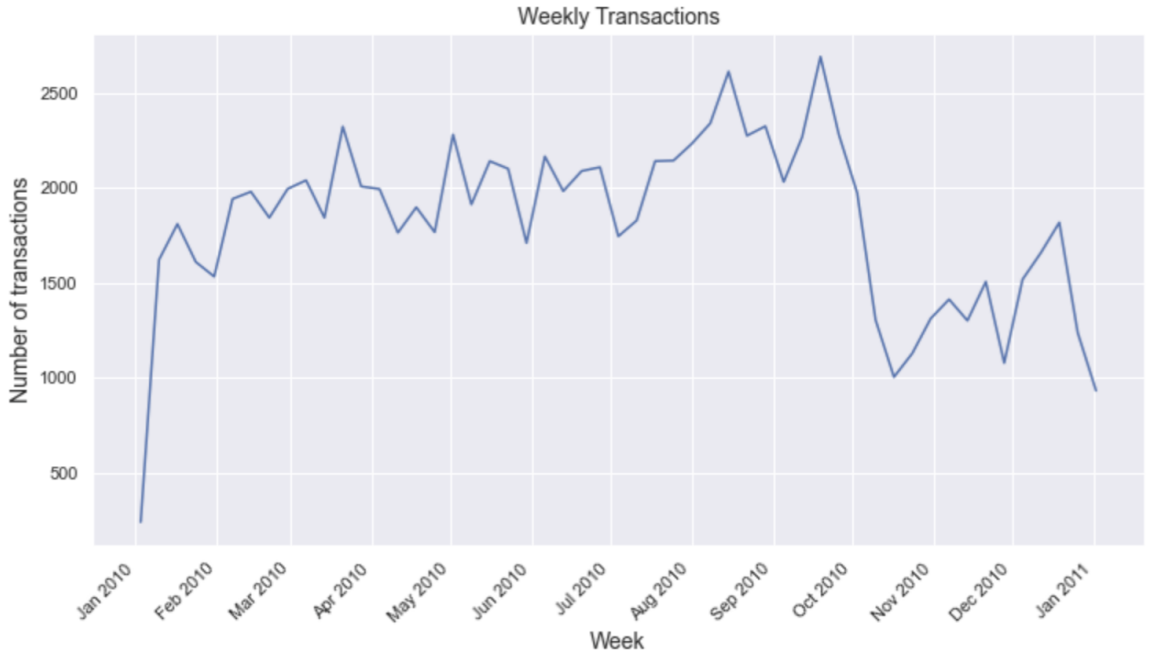
2. **Field Name: Cardnum**
   Description: Credit/Debit card number used for a particular transaction. The distribution shows the top 15 field values of card numbers. The most common card number is 514214852, with a total count of 1,192.



3. Field Name: Date

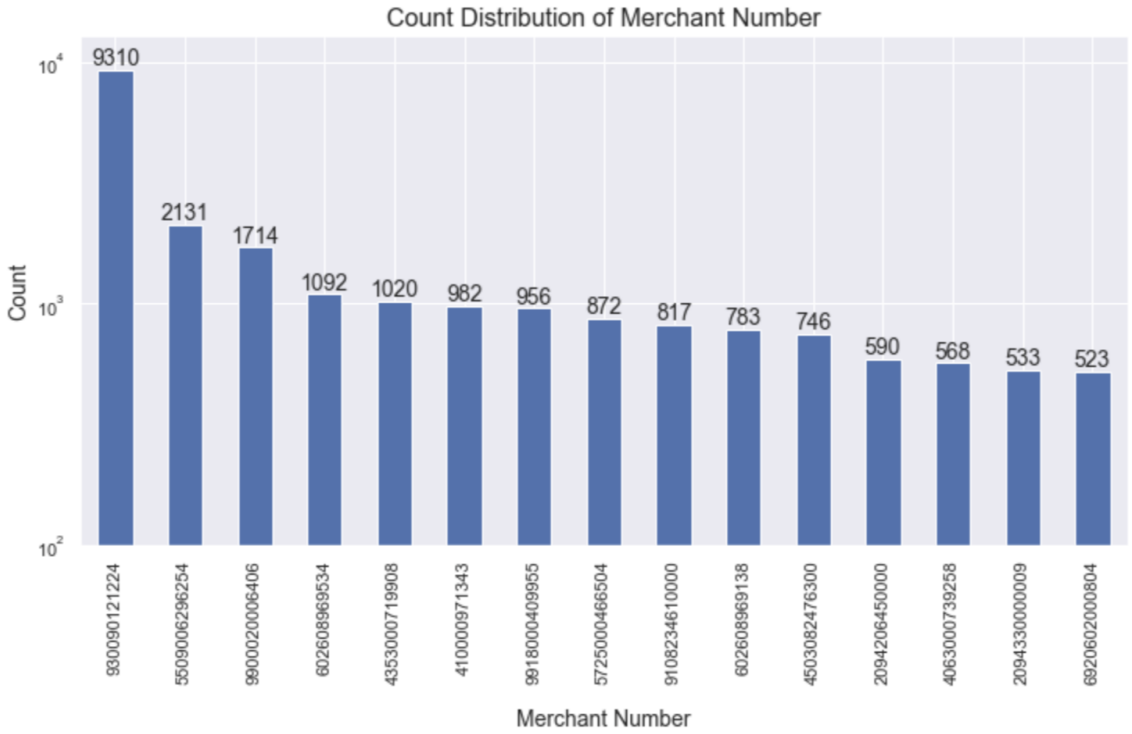
Description: Transaction date. The first distribution shows the number of daily transactions across the year. The second distribution shows the number of weekly transactions across the year.





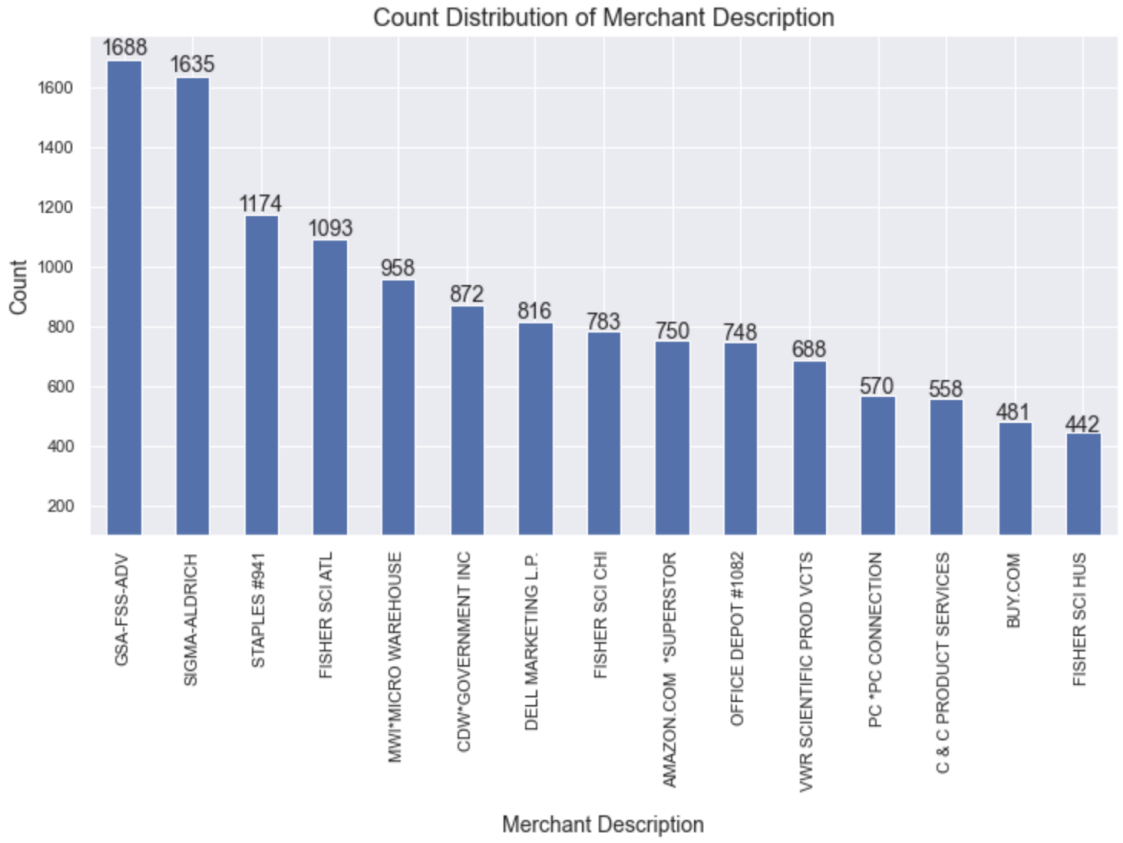
4. Field Name: Merchnum

Description: Merchant number for a particular transaction. The distribution shows the top 15 field values of merchant numbers. The most common merchant number is 930090121224, with a total count of 9,310.



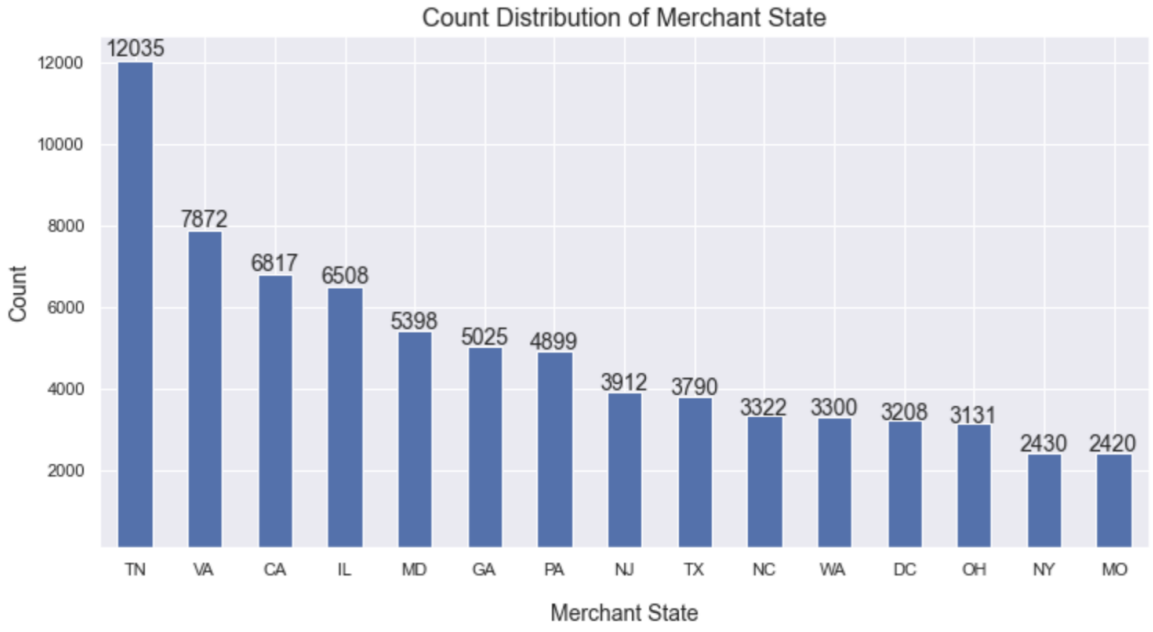
5. Field Name: Merch description

Description: Merchant description for a particular transaction. The distribution shows the top 15 field values of merchant descriptions. The most common merchant description is GSA-FSS-ADV, with a total count of 1,688.



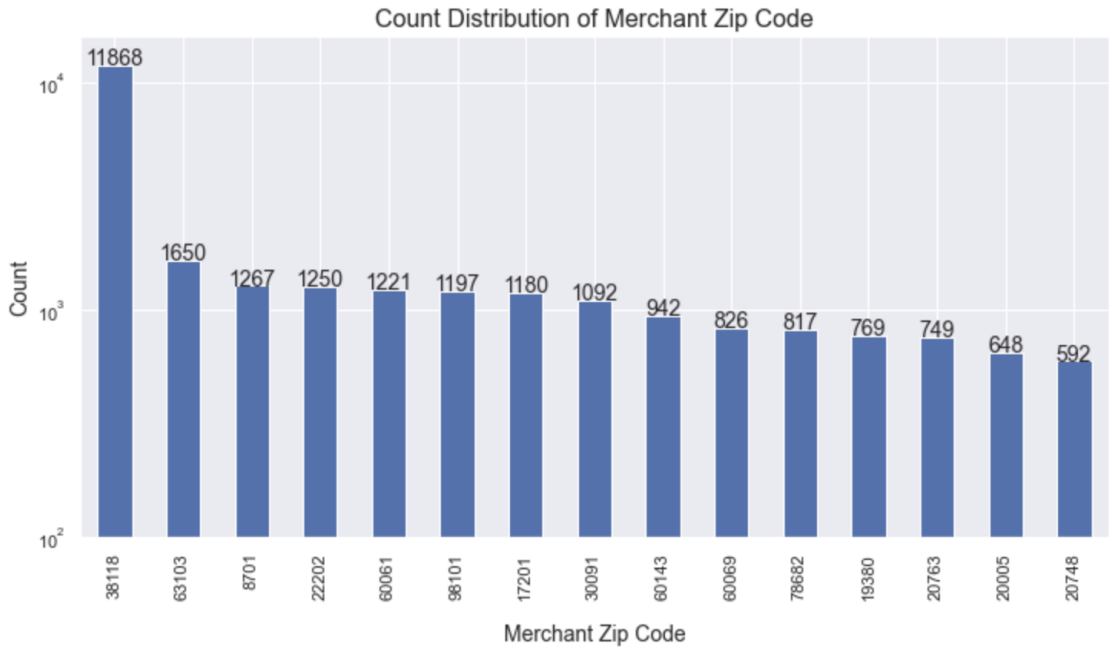
6. Field Name: Merch state

Description: Merchant's state for a particular transaction. The distribution shows the top 15 field values of merchant states. The most common merchant state is TN (Tennessee), with a total count of 12,035.



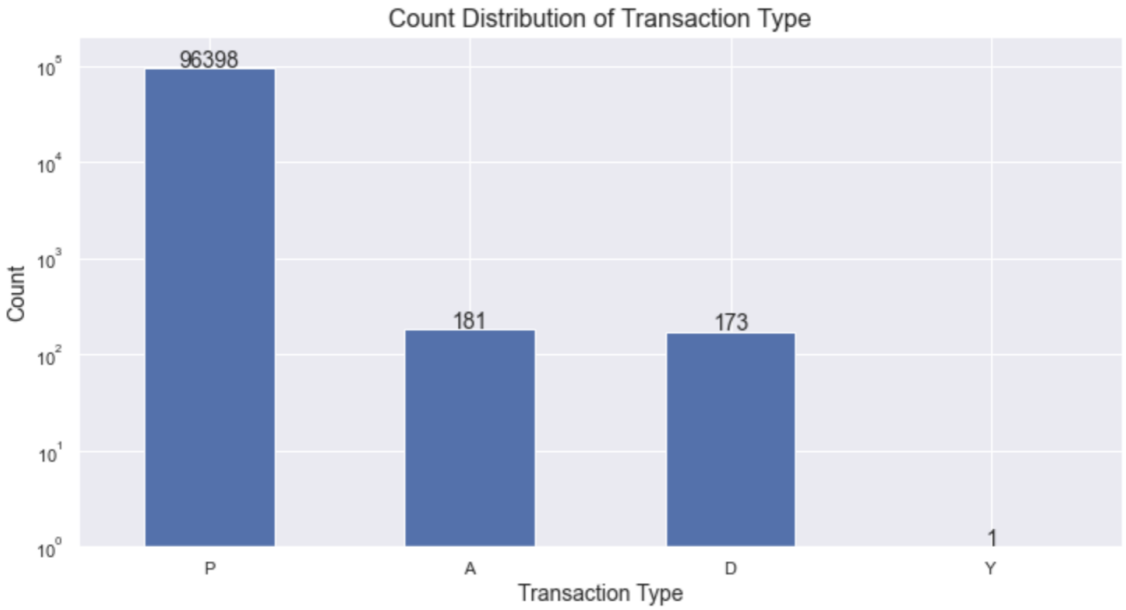
7. Field Name: Merch zip

Description: The zip code of a Merchant for a particular transaction. The distribution shows the top 15 field values of merchant zip codes. The most common merchant zip code is 38118, with a total count of 11,868.



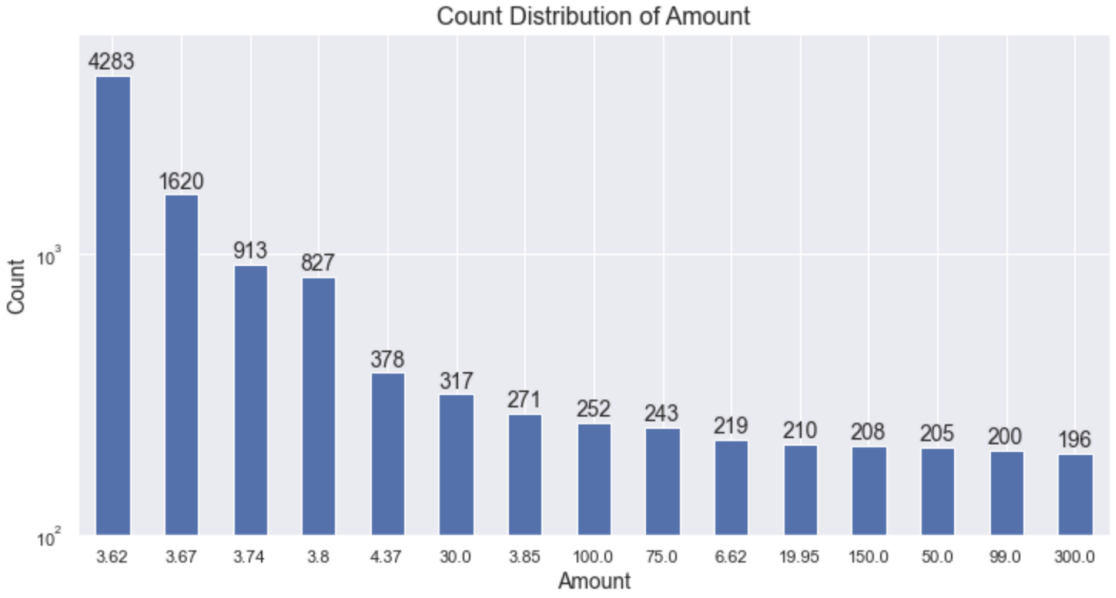
8. Field Name: Transtype

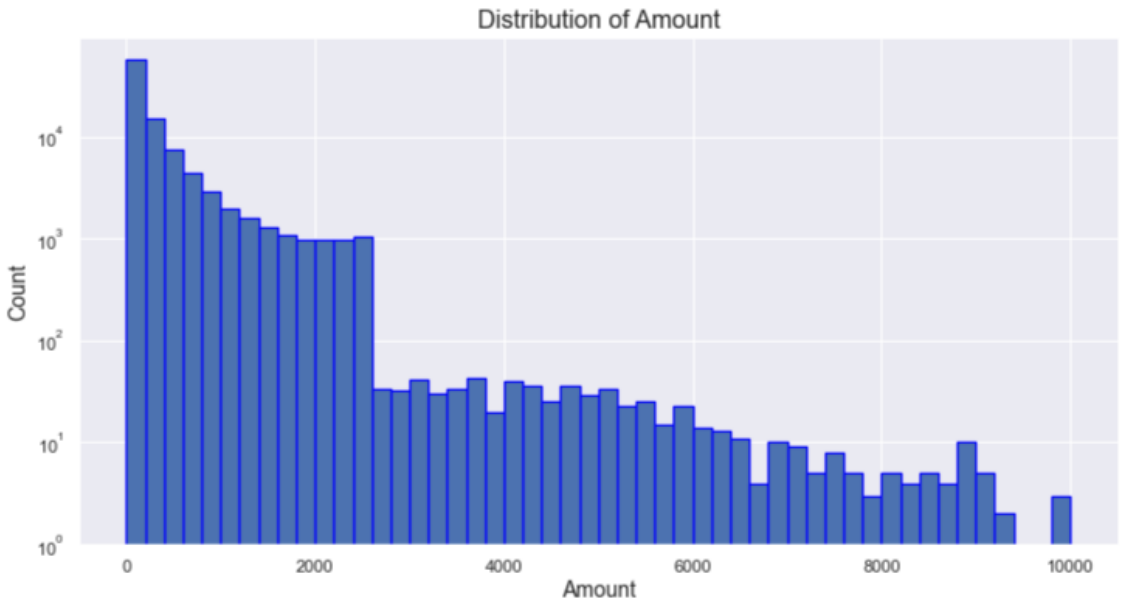
Description: Transaction type. The distribution shows all field values of transaction type. The most common type is P, with a total count of 96,398.



9. Field Name: Amount

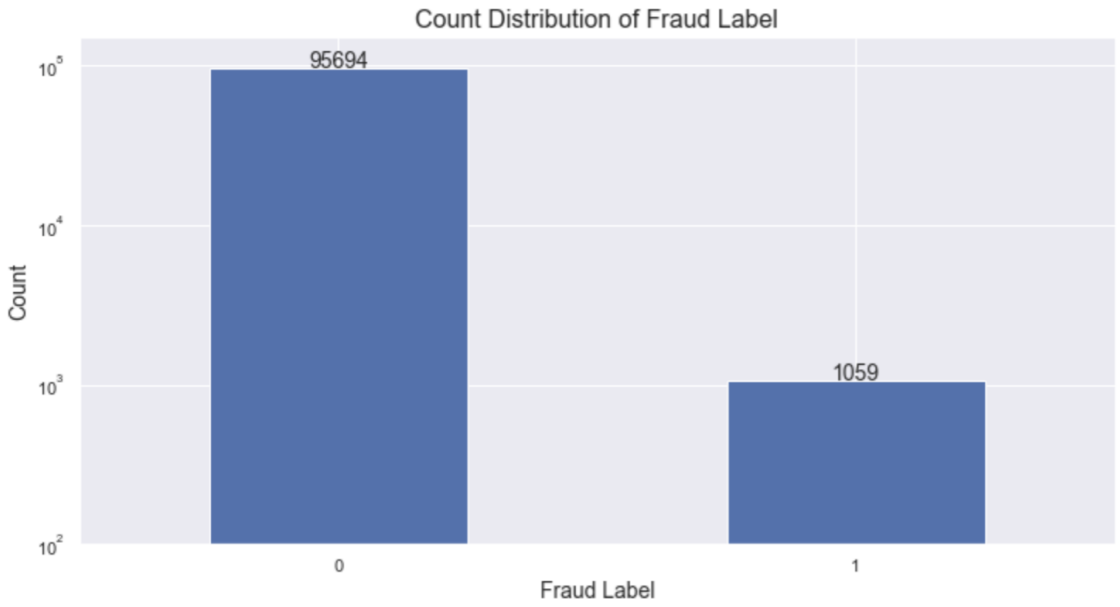
Description: The amount for a particular transaction. The first distribution shows the top 15 field values of amount. The most common value is 3.62, with a total count of 4,283. We see that most common transactions are generally for small amounts (with the highest value being 300). The second distribution shows the continuous distribution of amounts with the amounts being taken till standard deviation.





10. Field Name: Fraud

Description: Fraud identification label. Fraud = 0 (Not fraudulent), Fraud =1 (Fraud identified). The total count of fraud label 0 is 95,694. The total count of fraud label 1 is 1,059.



Report Submitted By: Neha Mittal

NOTE FOR THE GRADER:

Extra parts put in by me:

1) In the variable creation section, I have added an additional section "C" which is "Explanation of Target Encoding Variables" to explain target encoding of new variables
2) In the Financial curves section, in the recommendation for cutoff section, I have added an additional graph with the cutoff drawn for easy understanding of the client

Also under feature selection, I have put in the "what, why, how" but it is not explicitly labelled as it did not seem like a formal way of writing a report. Similarly my Model exploration includes "high level description" which isn't explicitly labelled.