DSO562 Fraud Analytics

Project 1 Report

Minjoo Sung

Table of Contents

Table of Contents....2
Section 1. Executive Summary....3
Section 2. Description of the data....4
1. Summary Statistics....4
1.1 Numerical Table....4
1.2 Categorical Table....4
2. Important Field Distributions....5
2.1 Amount Field Distribution....5
2.2 Date Field Distribution....5
2.3 Cardnum Field Distribution....6
Section 3. Data Cleaning....7
1. Exclusions....7
2. Outlier Treatment....8
3. Imputation process....8
3.1 Field: Merchnum....8
3.2 Field: Merch state....9
3.3 Field: Merch zip....9
Section 4. Variable Creation....10
Section 5. Feature Selection....12
1. Comparison of Models: RandomForestClassifier vs. LGBMClassifier....12
2. Comparison of Forward and Backward Selection....14
3. Comparison of LGBMClassifier with num_filter 200, 300, and 500....15
Section 6. Preliminary Models Exploration....18
1. Model Exploration Table....19
2. Performance Plot....20
Section 7: Final Model Performance....21
1. Best Model Choice - Non-default Hyperparameter Choices....21
2. Three Summary Tables....22
Section 8. Financial Curves and Recommended Cutoff....24
1. Plot for three curves and Recommendation for a cutoff at 5%....24
Section 9. Summary....25
Section 10. Appendix - DQR....26

Section 1. Executive Summary

This project addresses a critical business challenge: the detection of fraudulent transactions within a large set of card transaction data from 2010. By building an advanced fraud detection system, the goal was to improve fraud identification while minimizing false positives, which directly impacts financial losses and customer experience. Through careful data preparation, including cleaning and feature selection, a machine learning model was developed to effectively flag potentially fraudulent transactions.

The implemented model, a LightGBM classifier, achieved a fraud detection rate of 57.24% in the top 3% of flagged transactions in out-of-time (OOT) validation. This performance corresponds to significant financial savings, estimated to exceed $40 million annually, by reducing undetected fraudulent transactions while minimizing the decline of legitimate ones. This approach allows the business to improve fraud prevention efforts without negatively affecting customer transactions, striking a balance between operational efficiency and revenue protection.

Section 2. Description of the data

The dataset contains detailed records of card transactions for the year 2010. There are 97,852 transaction records across 10 fields, including both numerical and categorical variables. These fields provide essential insights into each transaction, including the card number, transaction date, merchant details, transaction types, and fraud status.

The table below provides summary statistics for the numerical and categorical fields.

1. Summary Statistics

1.1 Numerical Table

Field Name	# Records with value	% Populated	% Zeros	Min	Max	Mean	Std	Most Common value
Amount	97,852	100.00%	0	0.01	3,102,045.53	425.47	9,949.80	3.62

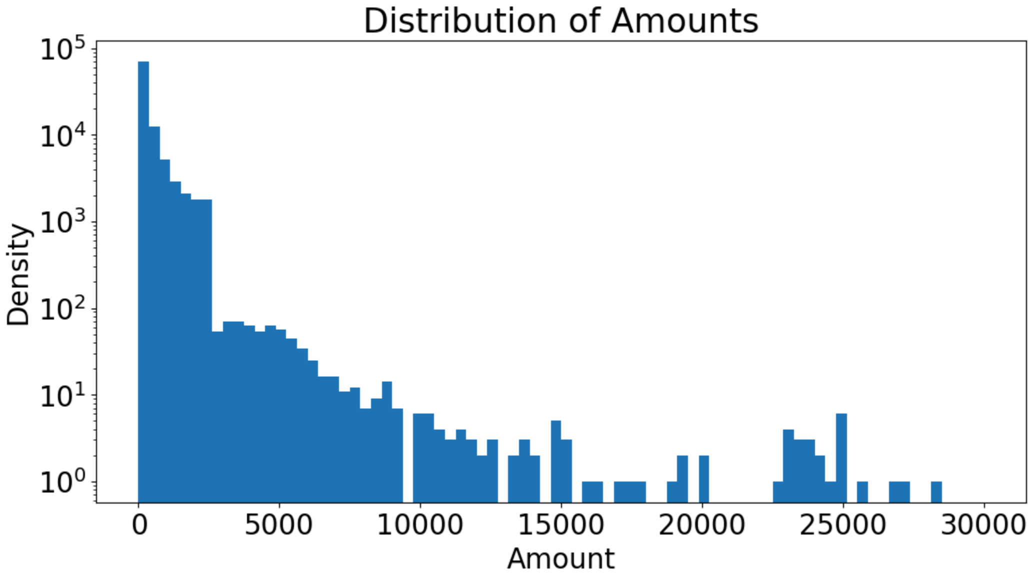
1.2 Categorical Table

Field Name	# Records with Values	% Populated	% Zeros	# Unique Values	Most Common Value
Recnum	97,852	100.00%	0	97,852	1
Cardnum	97,852	100.00%	0	1,645	5,142148452
Date	97,852	100.00%	0	365	2/28/10
Merchnum	94,455	96.53%	0	13,091	930090121224
Merch description	97,852	100.00%	0	13,126	GSA-FSS-ADV
Merch state	96,649	98.77%	0	227	TN
Merch zip	93,149	95.20%	0	4,567	38118
Transtype	97,852	100.00%	0	4	P
Fraud	97,852	100.00%	95,805	2	0

2. Important Field Distributions

2.1 Amount Field Distribution

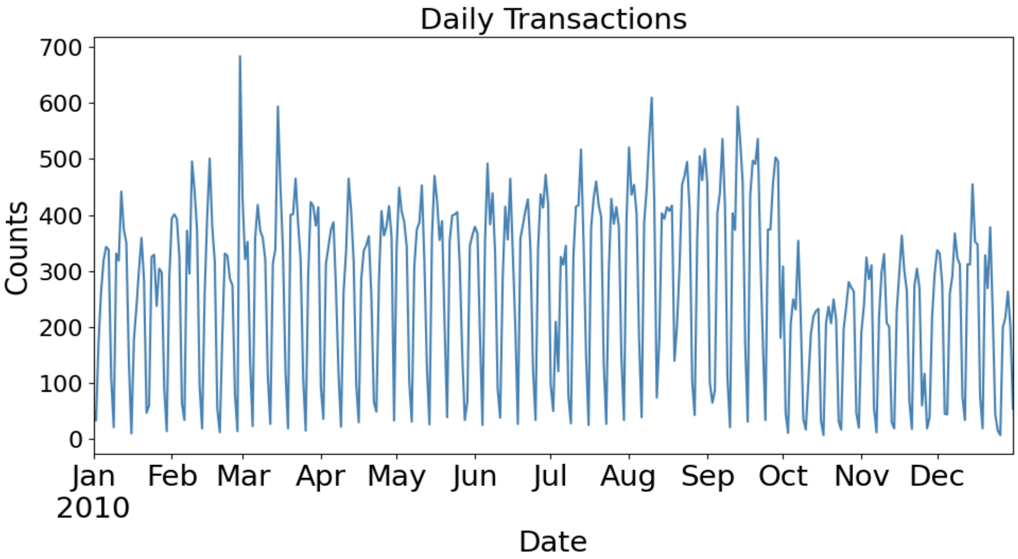
The Amount field represents the monetary value of each transaction. As shown in the distribution plot, the majority of transactions fall below 5,000 USD, with a sharp decline in frequency as the transaction amounts increase. Interestingly, there is a noticeable concentration of transactions below 2,500 USD, after which the distribution becomes more sporadic. The majority of the data is concentrated in the lower transaction amounts, with only a small number of transactions exceeding 10,000 USD. This right-skewed distribution indicates that most of the transactions are of relatively low value, which is typical in many financial datasets. However, the presence of larger transactions, though less frequent, could signal the need for heightened scrutiny, as these may correlate with fraudulent behavior.



2.2 Date Field Distribution

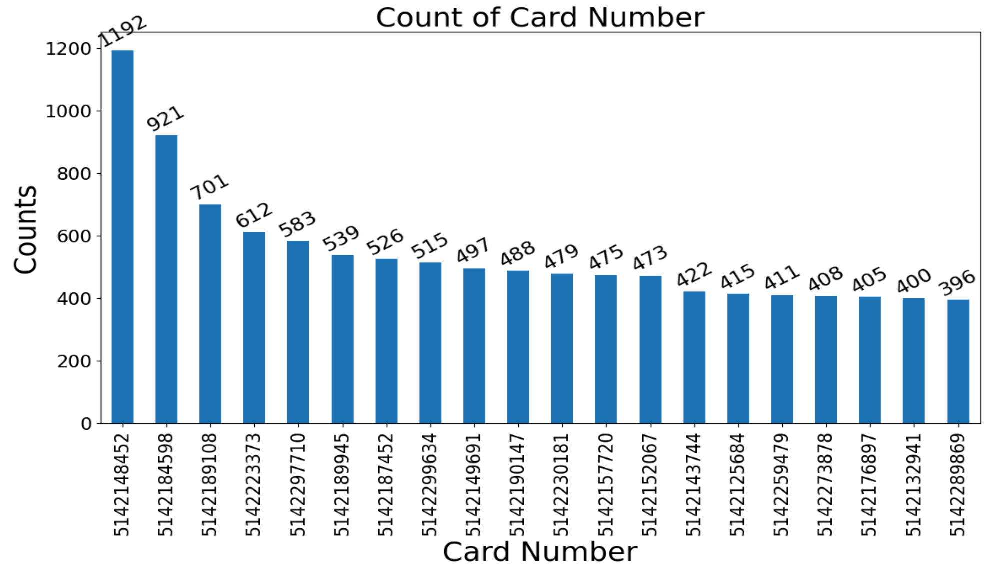
The Date field captures the specific day each transaction occurred. The plot of daily transaction counts reveals consistent fluctuations in activity throughout the year, with noticeable peaks and valleys. There is a particularly high level of transaction activity in February and September, followed by a significant drop-off in the number of transactions at the end of September. This sudden decline could indicate an operational or seasonal factor affecting transaction volumes.

Monitoring these trends can help financial institutions better predict and manage periods of high transaction volume and potential fraud spikes.



2.3 Cardnum Field Distribution

The Cardnum field represents unique card numbers used in the transactions. As shown in the distribution, the card number 5142148452 has the highest frequency, with 1,192 transactions, followed by 512485598 with 921 transactions. The distribution shows that some card numbers have been used significantly more frequently than others, which could indicate potential fraud patterns or repeat customers. Monitoring these high-frequency card numbers is crucial for detecting possible suspicious behavior.

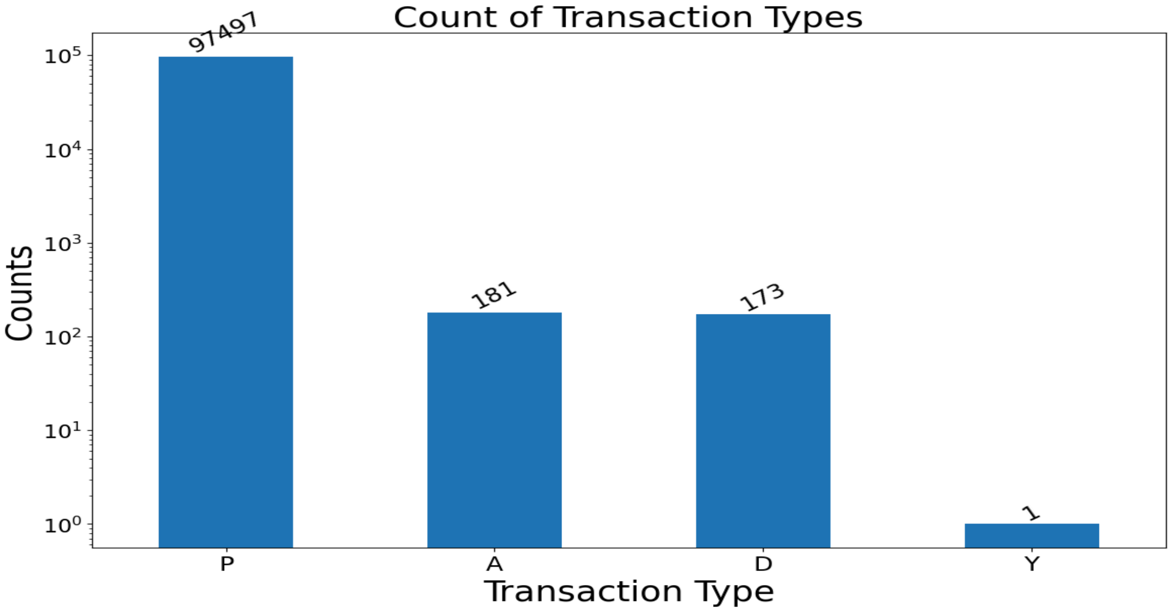


Section 3. Data Cleaning

As part of the data cleaning process for the fraud detection model, we identified specific records to exclude from the analysis, as well as outliers that required special treatment. The goal was to ensure that the model focuses on relevant data while excluding unusual or non-representative records that could distort the model's learning process.

1. Exclusions

We identified four different transaction types in the dataset: "P", "A", "D", and "Y". After reviewing the data, we found that the majority of transactions—99.64% (97,497 out of 97,852)—were labeled as type "P", which likely represents purchases. The other transaction types ("A", "D", and "Y") made up only a small fraction of the total volume, collectively accounting for just 0.36% of all transactions. The breakdown is as follows:



["P": 97,497 transactions (99.64%), "A": 181 transactions (0.18%), "D": 173 transactions (0.18%), "Y": 1 transaction (0.001%)]

Given that the overwhelming majority of transactions are of type "P", and the business manager's uncertainty regarding the exact meanings of the other types, we decided to exclude all non-"P" transactions from our analysis. This exclusion ensures that our model focuses on the most relevant and consistent data, which is crucial for accurate fraud detection.

2. Outlier Treatment

In our dataset, we discovered one significant outlier: a transaction with an unusually high amount of $3,102,045.53, while the next highest transaction was $47,900. After further investigation, we found that this transaction was associated with a legitimate purchase at a Mexican retailer and was not labeled as fraudulent. Given the extreme value of this transaction and the fact that it was legitimate, we decided to remove this outlier from our modeling process to avoid skewing the results.

Recnum	Cardnum	Date	Merch num	Merch description	Merch state	Merch zip	Trans type	Amount	Fraud
53179	5142189135	7/13/2010	NaN	INTERMEXICO	NaN	NaN	P	3,102,045.53	0

In general, for this project, we followed a "push" approach for outliers, adjusting values that fell far outside normal ranges by capping them at a reasonable threshold. This technique helped to minimize the impact of extreme values without discarding large amounts of data.

By implementing these exclusions and treating outliers effectively, we ensured that our model focused on detecting patterns in the majority of the dataset, leading to more reliable and accurate results.

3. Imputation process

In the imputation process, the goal was to fill missing values in a way that preserved the integrity of the dataset while making reasonable assumptions about the missing data. This is crucial for minimizing information loss and ensuring that the model can generalize well to unseen data. The imputation logic is designed to maximize data usability by leveraging available fields such as Merchnum, Merch description, and Merch zip. This approach not only fills missing values but also ensures consistency across the dataset, which is crucial for developing a robust fraud detection model. By carefully treating exclusions, outliers, and missing data, we've ensured that our model is trained on the most relevant and complete data possible. This process ultimately enhances the model's ability to accurately detect fraudulent transactions, reducing both false positives and missed frauds, and leading to significant cost savings for the business.

3.1 Field: Merchnum

Step 1: There were 3,279 records where the Merchnum field was missing. We needed to impute these missing values with reasonable guesses.

Step 2: First, we used the Merch description field to assign appropriate Merchnum values, leaving 2,115 records with blank Merchnum.

Step 3: For transactions with the description “retail transaction adjustment,” we assigned a value of ‘unknown’ to the Merchnum field, leaving 1,421 missing values.

Step 4: The remaining 1,421 records had 515 unique values for Merch description. To resolve this, we assigned a new and unique Merchnum for each unique Merch description. After this step, there were 0 missing values in the Merchnum field.

3.2 Field: Merch state

Step 1: There were 1,028 records where the Merch state field was missing.

Step 2: We used a mapping from the Merch zip field to fill in the Merch state, leaving 954 remaining missing values.

Step 3: Next, we used a mapping from the Merchnum field to impute the state. This step filled 1 record, reducing the number of missing values to 953.

Step 4: We then used a mapping from the Merch description field to further impute the state. This step filled 1 additional record, reducing the missing values to 952.

Step 5: For transactions with the description “retail transaction adjustment,” we assigned a value of ‘unknown’ to the Merch state field. After these steps, 297 records were still missing.

Step 6: For the remaining 297 records, we assigned the state as ‘foreign’ for non-U.S. transactions or ‘unknown’ if the state couldn’t be determined. After this step, there were 0 missing values in the Merch state field.

3.3 Field: Merch zip

Step 1: There were 4,347 records where the Merch zip field was missing.

Step 2: We created a mapping between the Merchnum and Merch zip fields using records where both were available, leaving 2,625 remaining missing values.

Step 3: For transactions with the description “retail transaction adjustment,” we assigned a value of ‘unknown’ to the Merch zip field, reducing the number of missing values to 1,940.

Step 4: For records still missing zip codes, we used the most common zip code for the known Merch state as a reasonable estimate, leaving 531 still missing zip codes.

Step 5: For the remaining 531 records, we assigned ‘unknown’ for the Merch zip field. After this step, there were 0 missing zip code values.

Section 4. Variable Creation

Fraud often occurs when transactional behaviors deviate from expected patterns, which can be identified by analyzing irregularities in frequency, timing, and amounts. To capture these anomalies, we created variables that assess transactional behavior across different dimensions. For instance, fraud might be concentrated on certain days of the week (Dow_Risk), involve suspicious locations (Target Encoded Fraud Risk), or display unnatural number distributions (Benford's Law). Additionally, variables like "New Entities" and "Day-Since" track new or infrequent participants in transactions, while "Frequency/Amount" and "Velocity Ratios" monitor spikes or sudden changes in transaction behavior. Measures of variability and unique counts help identify inconsistencies in transaction patterns, while square ratios highlight extreme discrepancies over short and long time windows. These variables were designed to capture the different ways fraud occurs by analyzing key transactional features over time.

Description of Variables	# Variables Created
Dow_Risk: Indicates the day of the week and filters the most fraudulent day of the week.	1
Target Encoded: Fraud risk scores for Merch state, Merch zip and Dow.	3
Benford's Law: Detection of potential fraud by analyzing the distribution of Cardnum and Merchnum to identify deviations from expected Benford's Law patterns.	2
New Entities: Unique identifiers for transactions (e.g., Cardnum, Merchnum) used to track behavior across multiple time windows.	21
Day-Since: Number of days since a transaction with the given entity has been observed.	23
Frequency and Amount: For each entity, frequency and amount variables were created across different time windows (0, 1, 3, 7, 14, 30, 60 days). These include count, average, max, median, total transaction amounts, and ratios comparing the current transaction to these statistics.	1,449
Count and Total Amount Ratio: Ratios comparing transaction counts and total amounts over shorter periods (0 and 1 day) to longer periods (7, 14, 30, 60 days).	368
Velocity Ratios:	184

Velocity ratios compare the frequency of transactions for a given entity over short time frames (0 and 1 day) to longer windows (7, 14, 30, 60 days).	
Variability: Measures of variability (average, maximum, and median) for each entity across multiple time windows (0, 1, 3, 7, 14, and 30 days).	414
Unique Count (Mapper Function): The unique count of entity combinations over various time windows (1, 3, 7, 14, 30, 60 days).	696
Square Ratios: Squared ratios of transaction counts comparing short time frames (0, 1 day) with longer windows (7, 14, 30, 60 days).	184
Amount Bins: Categorizes the ‘Amount’ field into five bins based on quantile distribution.	1
Foreign Zip Code: Identifies whether a merchant’s zip code is foreign or domestic by checking against a U.S. zip code database	1
Additional New Variables	
Transaction Velocity: Time difference between consecutive transactions for each card**.**	1
Merchant Transaction Freq: Time difference between consecutive transactions for each merchant.	1
Amount Variation: Rolling standard deviation of transaction amounts over 3 transactions for each card.	1
Avg Merchant Time: Average time between transactions at each merchant.	1
Merchant Cluster: Categorizes merchants into 5 clusters based on transaction amounts using quantiles,	1

Section 5. Feature Selection

The goal of the feature selection process was to identify the most relevant features that improve model performance while reducing computational complexity. By selecting the top features, we ensure that the model focuses on the variables that contribute most to fraud detection, improving both accuracy and processing speed. Our performance goal was to achieve a score above 0.70 for the selected model.

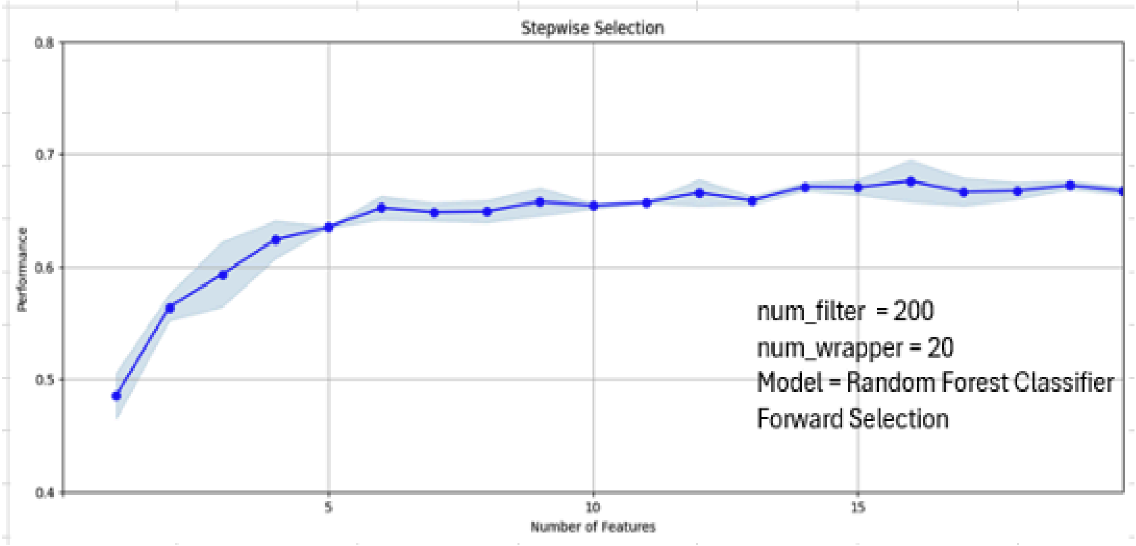
We used two main classifiers for feature selection: RandomForestClassifier and LGBMClassifier. For both classifiers, we initially set the num_filter = 200 and num_wrapper = 20, using a forward selection method. This allowed us to rank and select the most important variables based on their contribution to the model's performance. The forward selection process incrementally adds features and evaluates model performance as each new feature is added, helping us identify when adding more features no longer provides significant improvements.

1. Comparison of Models: RandomForestClassifier vs. LGBMClassifier

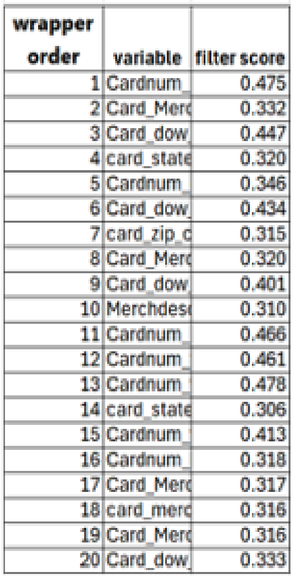
Model	num_filter	num_wrapper	Selection Type
RandomForestClassifier	200	20	Forward
LGBMClassifier	200	20	Forward

1.1 RandomForestClassifier (num_filter = 200, num_wrapper = 20, forward selection)

The performance plot shows that the model performance increased steadily as more features were added, peaking at around 15 features before plateauing, with performance stabilizing around 0.67.



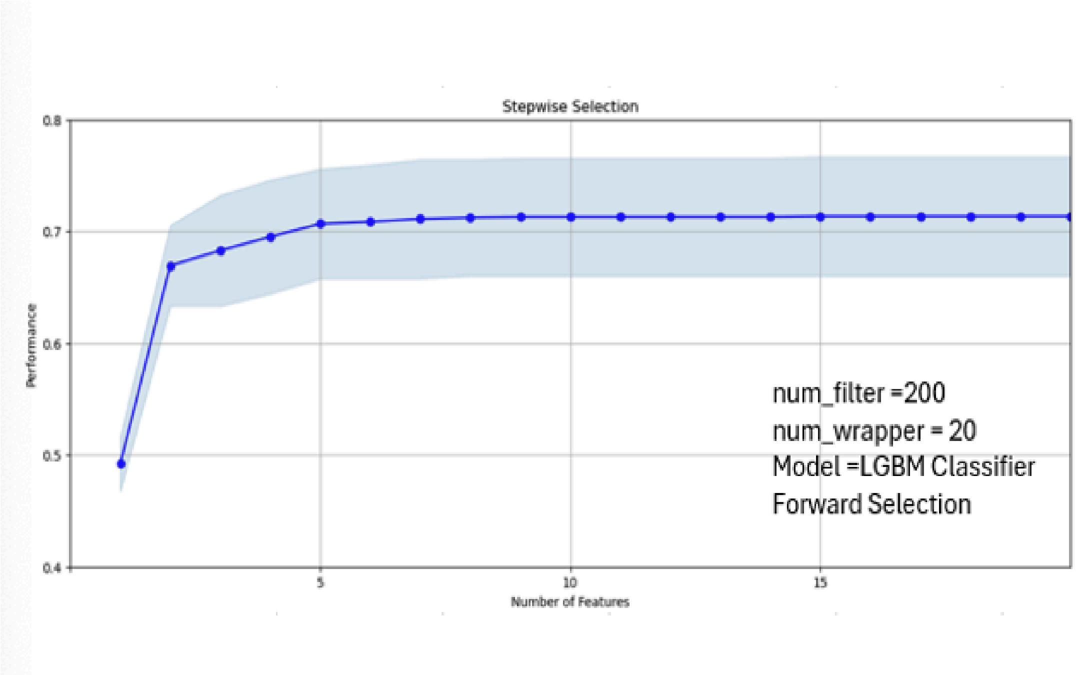
num_filter = 200
num_wrapper = 20
Model = Random Forest Classifier
Forward Selection



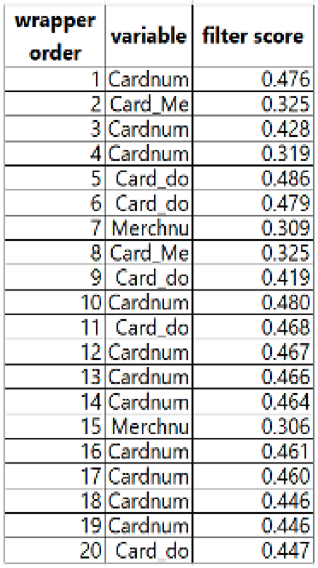
wrapper order	variable	filter score
1	Cardnum_	0.475
2	Card_Merc	0.332
3	Card_dow	0.447
4	card_state	0.320
5	Cardnum_	0.346
6	Card_dow	0.434
7	card_zip_c	0.315
8	Card_Merc	0.320
9	Card_dow	0.401
10	Merchdes	0.310
11	Cardnum_	0.466
12	Cardnum_	0.461
13	Cardnum_	0.478
14	card_state	0.306
15	Cardnum_	0.413
16	Cardnum_	0.318
17	Card_Merc	0.317
18	card_merc	0.316
19	Card_Merc	0.316
20	Card_dow	0.333

1.2 LGBMClassifier (num_filter = 200, num_wrapper = 20, forward selection)

The performance of LGBMClassifier was more stable and slightly better than RandomForestClassifier, with smoother improvements as features were added, reaching a performance above 0.70.



num_filter =200
num_wrapper = 20
Model =LGBM Classifier
Forward Selection



wrapper order	variable	filter score
1	Cardnum	0.476
2	Card_Me	0.325
3	Cardnum	0.428
4	Cardnum	0.319
5	Card_do	0.486
6	Card_do	0.479
7	Merchnu	0.309
8	Card_Me	0.325
9	Card_do	0.419
10	Cardnum	0.480
11	Card_do	0.468
12	Cardnum	0.467
13	Cardnum	0.466
14	Cardnum	0.464
15	Merchnu	0.306
16	Cardnum	0.461
17	Cardnum	0.460
18	Cardnum	0.446
19	Cardnum	0.446
20	Card_do	0.447

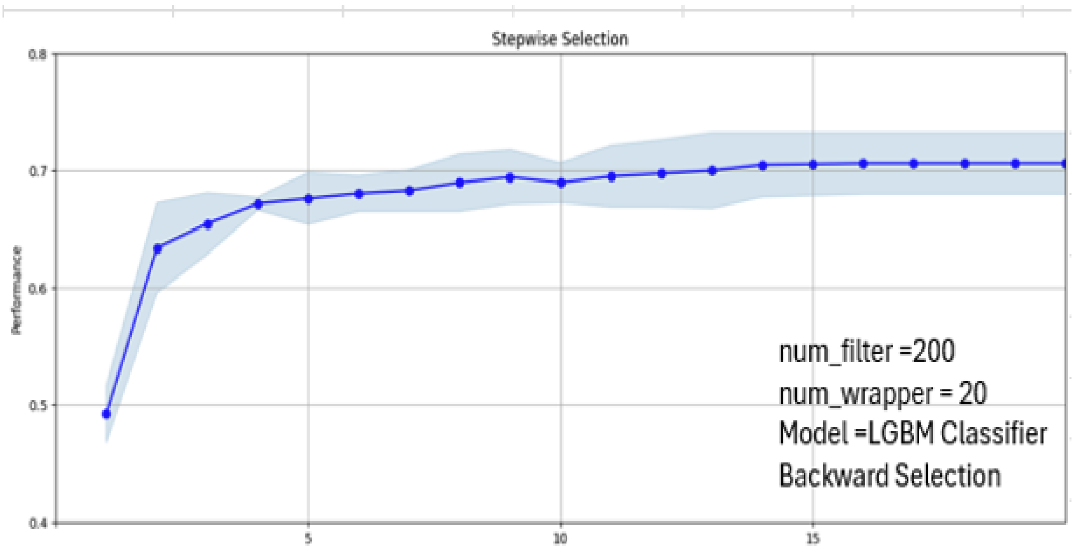
Based on the comparison, I chose LGBMClassifier as the final model for further optimization due to its better overall performance and ability to consistently achieve a score above 0.70.

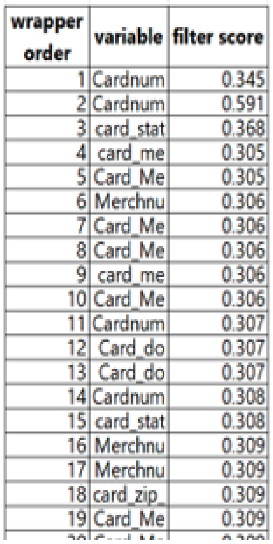
2. Comparison of Forward and Backward Selection

Model	num_filter	num_wrapper	Selection Type
LGBMClassifier	200	20	Forward
LGBMClassifier	200	20	Backward

2.1 LGBMClassifier (num_filter = 200, num_wrapper = 20, backward selection)

The backward selection method yielded similar performance to forward selection but was slightly slower in processing and more prone to overfitting with smaller feature sets, with performance stabilizing just below 0.70.





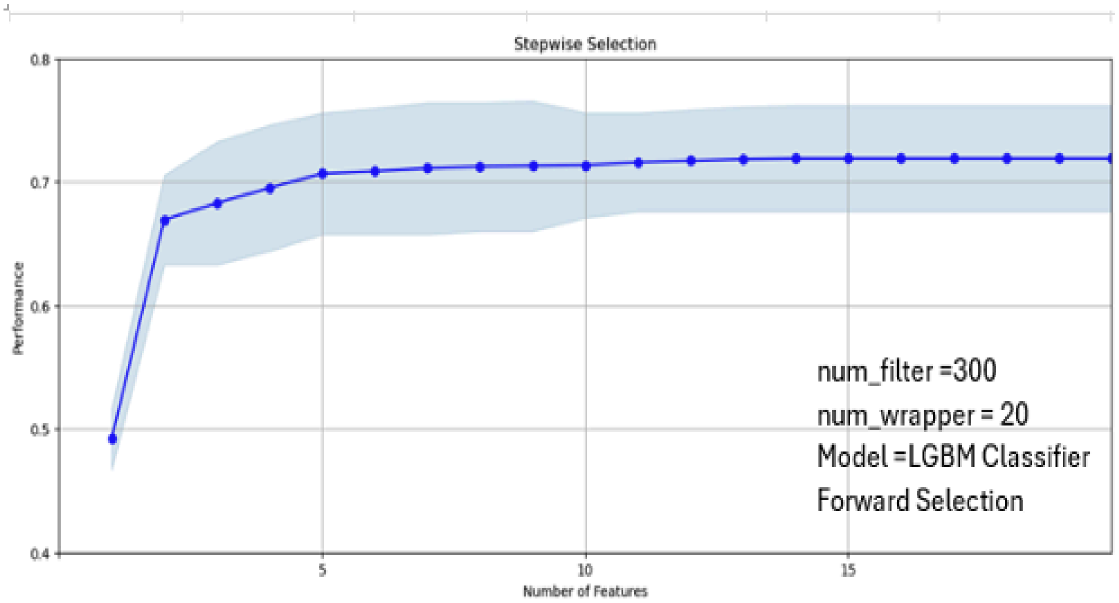
wrapper order	variable	filter score
1	Cardnum	0.345
2	Cardnum	0.591
3	card_stat	0.368
4	card_me	0.305
5	Card_Me	0.305
6	Merchnu	0.306
7	Card_Me	0.306
8	Card_Me	0.306
9	card_me	0.306
10	Card_Me	0.306
11	Cardnum	0.307
12	Card_do	0.307
13	Card_do	0.307
14	Cardnum	0.308
15	card_stat	0.308
16	Merchnu	0.309
17	Merchnu	0.309
18	card_zip_	0.309
19	Card_Me	0.309

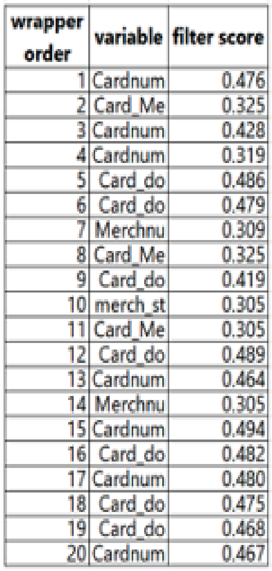
I selected forward selection due to its faster processing time and smoother feature ranking progression, which was more reliable for achieving consistent performance above 0.70.

3. Comparison of LGBMClassifier with num_filter 200, 300, and 500

Model	num_filter	num_wrapper	Selection Type
LGBMClassifier	200	20	Forward
LGBMClassifier	300	20	Forward
LGBMClassifier	500	20	Forward

3.1 LGBMClassifier (num_filter = 300, num_wrapper = 20, forward selection)
Performance for num_filter = 300 also reached a stable score above 0.70, showing solid improvement as more features were added.

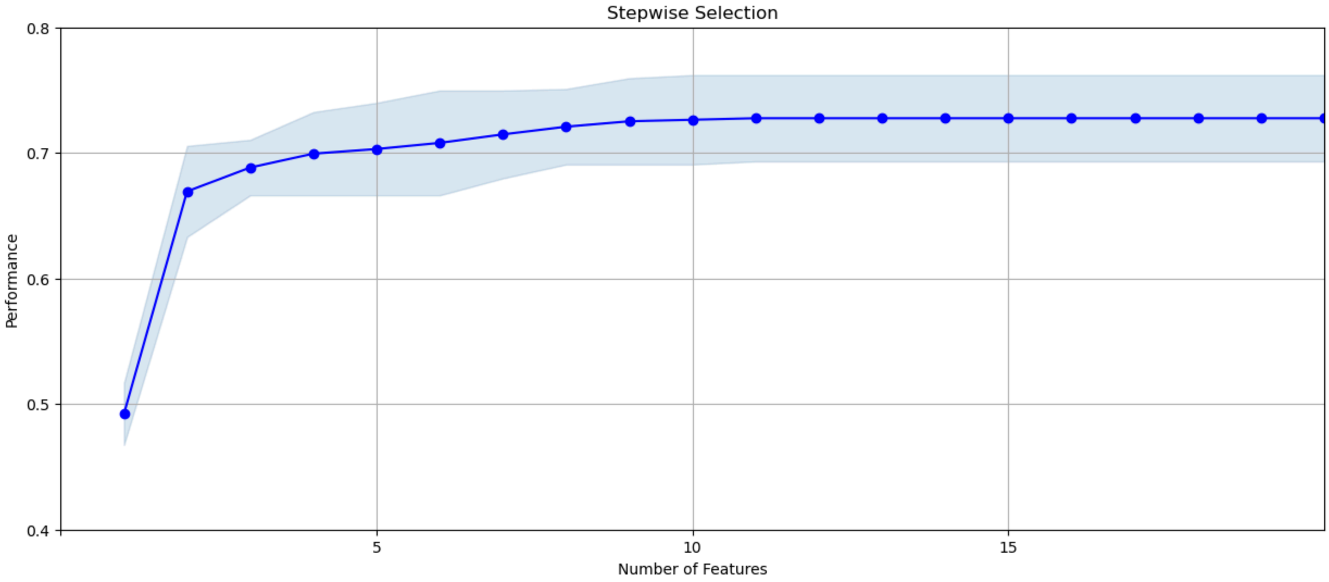




wrapper order	variable	filter score
1	Cardnum	0.476
2	Card_Me	0.325
3	Cardnum	0.428
4	Cardnum	0.319
5	Card_do	0.486
6	Card_do	0.479
7	Merchnu	0.309
8	Card_Me	0.325
9	Card_do	0.419
10	merch_st	0.305
11	Card_Me	0.305
12	Card_do	0.489
13	Cardnum	0.464
14	Merchnu	0.305
15	Cardnum	0.494
16	Card_do	0.482
17	Cardnum	0.480
18	Card_do	0.475
19	Card_do	0.468
20	Cardnum	0.467

3.2 LGBMClassifier (num_filter = 500, num_wrapper = 20, forward selection)

LGBM, num_filter = 500, Forward Selection => Final Choice of Model



The final LGBM model with num_filter = 500 showed the highest performance, consistently maintaining a score above 0.70. Although the performance with num_filter = 300 also reached the target, the 500-filter model demonstrated a slightly higher score and better stability with 20 selected variables. The performance of the model reached a saturation point at around 10 variables, where adding additional variables did not significantly improve performance.

I selected LGBMClassifier with num_filter = 500, num_wrapper = 20, and forward selection as the final model. While num_filter = 300 also met the goal, the num_filter = 500 model showed a slight improvement in performance, making it the best choice for balancing feature selection complexity with accuracy.

Top 20 Variables Selected: These variables were selected based on their importance scores calculated using the KS filter, followed by wrapper selection. The final variables were:

wrapper order	variable	filter score
1	Cardnum_unique_count_for_card_state_1	0.476067
2	Card_Merchdesc_State_total_7	0.324668
3	card_merch_day_since	0.266699
4	Cardnum_count_1_by_30	0.428229
5	Cardnum_max_14	0.318826
6	Cardnum_count_3	0.563356
7	card_zip_total_14	0.332039
8	Card_dow_unique_count_for_merch_state_1	0.447357
9	card_merch_count_1_by_60	0.280599
10	state_des_total_3	0.31554
11	Card_Merchdesc_day_since	0.268933
12	Cardnum_count_7	0.526897
13	Card_dow_total_14	0.511203
14	Cardnum_total_14	0.494375
15	Card_dow_count_7	0.482384
16	Cardnum_actual/total_0	0.47955
17	Cardnum_variability_max_1	0.477836
18	Card_dow_total_30	0.474759
19	Card_dow_max_14	0.470975
20	Card_dow_vdratio_0by7	0.467961

Section 6. Preliminary Models Exploration

I explored five machine learning algorithms—one linear and four non-linear—to detect fraudulent transactions. Each model was tuned to optimize performance while balancing complexity and avoiding overfitting.

The **Logistic Regression** model, a linear algorithm, served as a baseline due to its simplicity. I applied regularization using the C parameter to prevent overfitting and tested both L1 and L2 penalties. Although simple, Logistic Regression provided a useful benchmark against more complex models.

For the **Decision Tree**, a non-linear algorithm, I experimented with varying the maximum depth (max_depth) from 2 to 20 to understand its impact on overfitting. The hyperparameters adjusted included max_depth, min_samples_split, and min_samples_leaf to fine-tune the model's complexity.

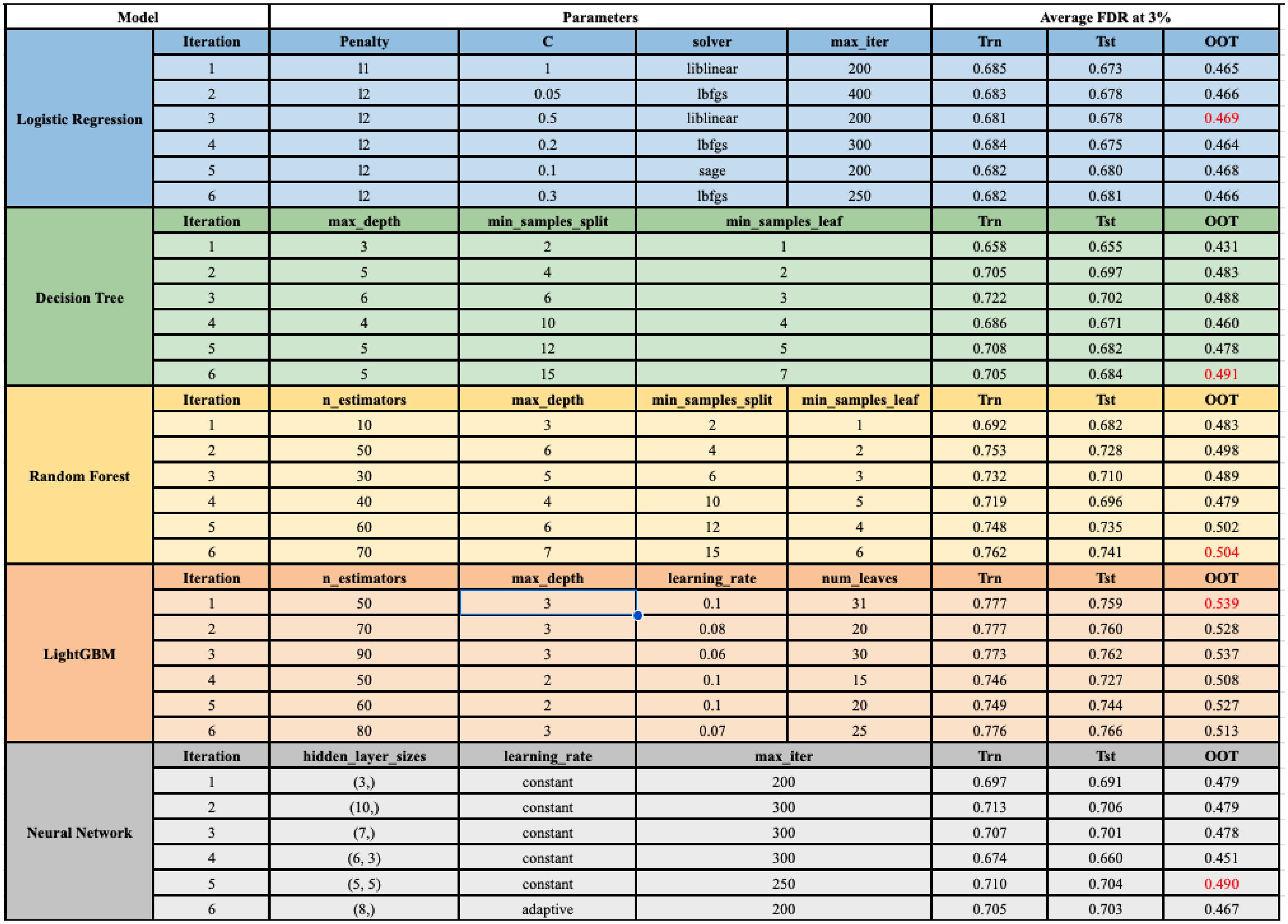
The **Random Forest**, an ensemble method, also a non-linear algorithm, was built using 200 estimators. I adjusted hyperparameters like n_estimators, min_samples_split, and max_depth. Random Forest performed well in avoiding overfitting by combining the predictions of multiple trees.

The **LightGBM** model, another non-linear algorithm, demonstrated excellent performance. Tuning focused on hyperparameters such as n_estimators, max_depth, learning_rate, and min_samples_leaf, allowing LightGBM to handle large datasets efficiently while reducing overfitting and maintaining high accuracy.

Finally, I explored the **Neural Network** (MLPClassifier), a complex non-linear algorithm, by increasing the number of hidden layers (ranging from (10, 10) to (100, 100)) to improve model complexity. Regularization, controlled by the alpha parameter, was applied to prevent overfitting, with adjustments made to hidden_layer_sizes and max_iter.

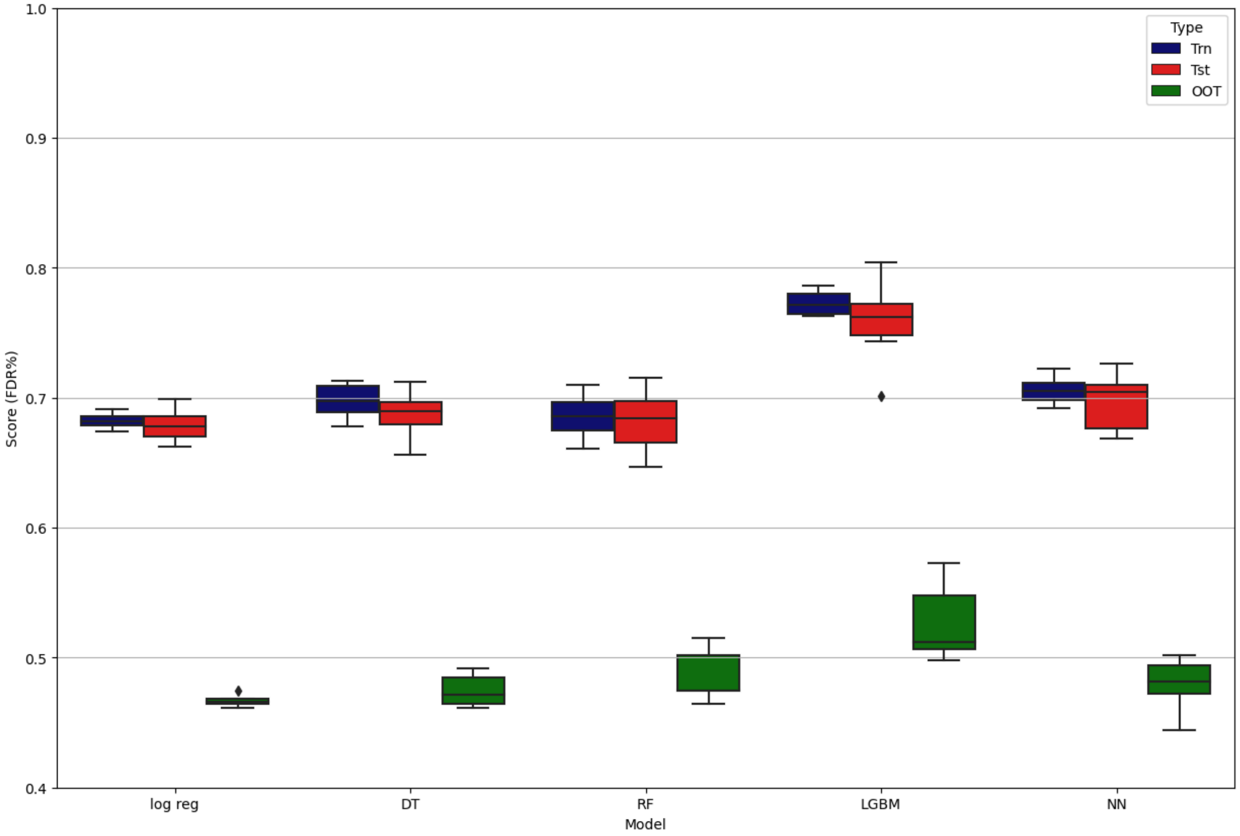
Each model was evaluated on training, test, and out-of-time (OOT) datasets, assessing performance, overfitting risks, and consistency. The performance of the models was visualized using box plots, with a table summarizing the hyperparameter settings and performance metrics for each model.

1. Model Exploration Table



Model		Parameters				Average FDR at 3%		
Logistic Regression	Iteration	Penalty	C	solver	max_iter	Trn	Tst	OOT
	1	l1	1	liblinear	200	0.685	0.673	0.465
	2	l2	0.05	lbfgs	400	0.683	0.678	0.466
	3	l2	0.5	liblinear	200	0.681	0.678	0.469
	4	l2	0.2	lbfgs	300	0.684	0.675	0.464
	5	l2	0.1	sage	200	0.682	0.680	0.468
	6	l2	0.3	lbfgs	250	0.682	0.681	0.466
Decision Tree	Iteration	max_depth	min_samples_split	min_samples_leaf		Trn	Tst	OOT
	1	3	2	1		0.658	0.655	0.431
	2	5	4	2		0.705	0.697	0.483
	3	6	6	3		0.722	0.702	0.488
	4	4	10	4		0.686	0.671	0.460
	5	5	12	5		0.708	0.682	0.478
	6	5	15	7		0.705	0.684	0.491
Random Forest	Iteration	n_estimators	max_depth	min_samples_split	min_samples_leaf	Trn	Tst	OOT
	1	10	3	2	1	0.692	0.682	0.483
	2	50	6	4	2	0.753	0.728	0.498
	3	30	5	6	3	0.732	0.710	0.489
	4	40	4	10	5	0.719	0.696	0.479
	5	60	6	12	4	0.748	0.735	0.502
	6	70	7	15	6	0.762	0.741	0.504
LightGBM	Iteration	n_estimators	max_depth	learning_rate	num_leaves	Trn	Tst	OOT
	1	50	3	0.1	31	0.777	0.759	0.539
	2	70	3	0.08	20	0.777	0.760	0.528
	3	90	3	0.06	30	0.773	0.762	0.537
	4	50	2	0.1	15	0.746	0.727	0.508
	5	60	2	0.1	20	0.749	0.744	0.527
	6	80	3	0.07	25	0.776	0.766	0.513
Neural Network	Iteration	hidden_layer_sizes	learning_rate	max_iter		Trn	Tst	OOT
	1	(3,)	constant	200		0.697	0.691	0.479
	2	(10,)	constant	300		0.713	0.706	0.479
	3	(7,)	constant	300		0.707	0.701	0.478
	4	(6, 3)	constant	300		0.674	0.660	0.451
	5	(5, 5)	constant	250		0.710	0.704	0.490
	6	(8,)	adaptive	200		0.705	0.703	0.467

2. Performance Plot



Model Selection: LightGBM

After evaluating the performance of all models using the training, test, and out-of-time (OOT) datasets, I selected LightGBM as the final model due to its superior performance. Among all models, LightGBM achieved the highest out-of-time (OOT) score, with an FDR@3% of 0.539. This was significantly higher compared to other algorithms, such as Random Forest, which reached 0.504, and Logistic Regression, which only managed 0.469. Both Decision Tree and Neural Network performed below 0.50 in the OOT set. In addition to the tabulated comparison, the box plot further supported the selection of LightGBM. It demonstrated the highest OOT range and maintained strong performance across the training and test datasets, indicating that the model was not overfitting. Considering both the quantitative performance and the visual analysis, LightGBM was the best choice, delivering accurate and reliable fraud detection while maintaining generalization over time.

Section 7: Final Model Performance

For the final model, I selected LightGBM due to its strong performance during the preliminary explorations. The model delivered the highest FDR@3% in the out-of-time (OOT) set, reaching 0.539, along with training and test scores of 0.777 and 0.759, respectively. These results were superior to other algorithms explored, including Random Forest and Logistic Regression, which showed lower stability and performance across the datasets.

1. Best Model Choice - Non-default Hyperparameter Choices

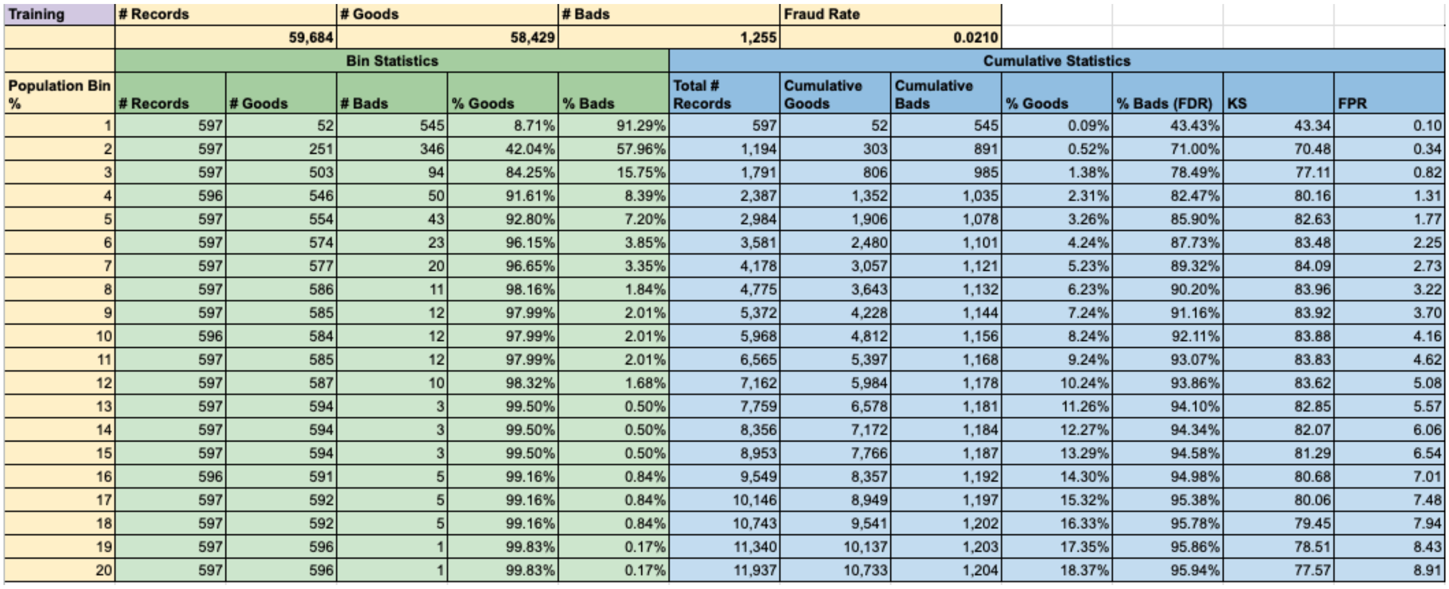
Final Model	n_estimators	max_depth	learning_rate	num_leaves
LightGBM	50	3	0.1	31

The LightGBM model was fine-tuned using specific non-default hyperparameters to optimize its performance. I set the n_estimators to 50, which controls the number of boosting iterations or trees. The max_depth was limited to 3, ensuring the model did not overfit by growing too complex. The learning_rate was set to 0.1, balancing the speed of learning and risk of overfitting by controlling the step size at each iteration. Lastly, the num_leaves was set to 31, determining the number of leaf nodes in each tree to control the granularity of the splits. These choices were made to ensure a balance between model complexity and generalization capability.

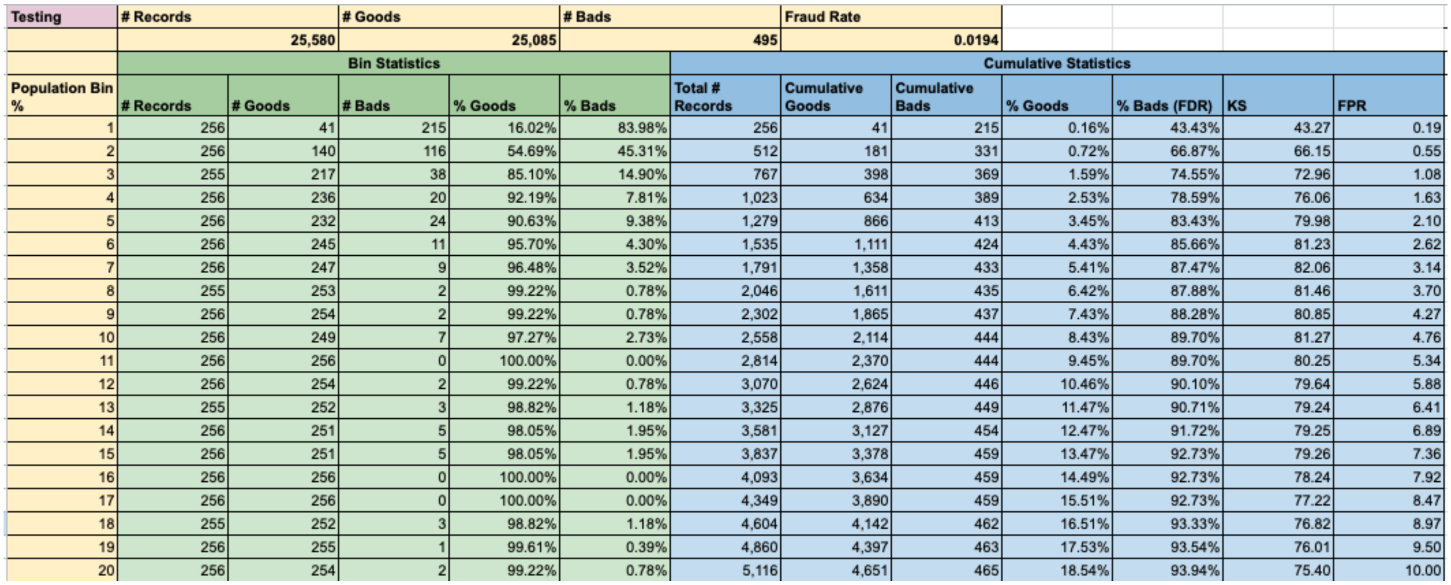
In the model selection process, my goal was to achieve an FDR@3% of at least 0.57. I ran the model iteratively with the fixed hyperparameters and looped it multiple times until a suitable model was found that met the desired FDR in the OOT set. During the final iteration, the model yielded strong results, achieving an FDR@3% of 0.7849 on the training set, 0.7455 on the test set, and 0.5724 on the OOT set. This confirmed that the model met the target performance in the OOT set while maintaining strong generalization on both the training and test data.

These results highlight the model's stability and effectiveness, as it consistently performed well across all datasets. The final OOT FDR exceeded the target of 0.57, ensuring the model's ability to generalize well to unseen data. This LightGBM model, with its optimized hyperparameters, represents a solid final choice, offering high performance and minimal variance between the training, test, and OOT results.

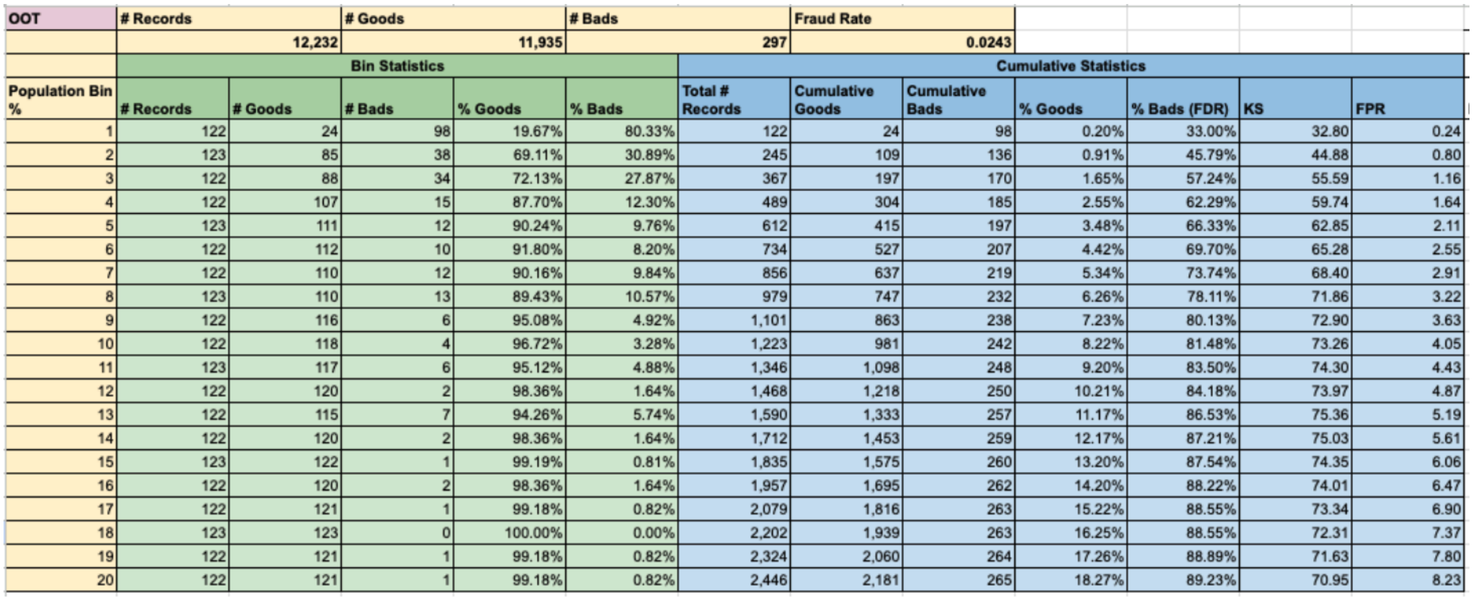
2. Three Summary Tables



Training	# Records		# Goods		# Bads		Fraud Rate					
	59,684		58,429		1,255		0.0210					
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	597	52	545	8.71%	91.29%	597	52	545	0.09%	43.43%	43.34	0.10
2	597	251	346	42.04%	57.96%	1,194	303	891	0.52%	71.00%	70.48	0.34
3	597	503	94	84.25%	15.75%	1,791	806	985	1.38%	78.49%	77.11	0.82
4	596	546	50	91.61%	8.39%	2,387	1,352	1,035	2.31%	82.47%	80.16	1.31
5	597	554	43	92.80%	7.20%	2,984	1,906	1,078	3.26%	85.90%	82.63	1.77
6	597	574	23	96.15%	3.85%	3,581	2,480	1,101	4.24%	87.73%	83.48	2.25
7	597	577	20	96.65%	3.35%	4,178	3,057	1,121	5.23%	89.32%	84.09	2.73
8	597	586	11	98.16%	1.84%	4,775	3,643	1,132	6.23%	90.20%	83.96	3.22
9	597	585	12	97.99%	2.01%	5,372	4,228	1,144	7.24%	91.16%	83.92	3.70
10	596	584	12	97.99%	2.01%	5,968	4,812	1,156	8.24%	92.11%	83.88	4.16
11	597	585	12	97.99%	2.01%	6,565	5,397	1,168	9.24%	93.07%	83.83	4.62
12	597	587	10	98.32%	1.68%	7,162	5,984	1,178	10.24%	93.86%	83.62	5.08
13	597	594	3	99.50%	0.50%	7,759	6,578	1,181	11.26%	94.10%	82.85	5.57
14	597	594	3	99.50%	0.50%	8,356	7,172	1,184	12.27%	94.34%	82.07	6.06
15	597	594	3	99.50%	0.50%	8,953	7,766	1,187	13.29%	94.58%	81.29	6.54
16	596	591	5	99.16%	0.84%	9,549	8,357	1,192	14.30%	94.98%	80.68	7.01
17	597	592	5	99.16%	0.84%	10,146	8,949	1,197	15.32%	95.38%	80.06	7.48
18	597	592	5	99.16%	0.84%	10,743	9,541	1,202	16.33%	95.78%	79.45	7.94
19	597	596	1	99.83%	0.17%	11,340	10,137	1,203	17.35%	95.86%	78.51	8.43
20	597	596	1	99.83%	0.17%	11,937	10,733	1,204	18.37%	95.94%	77.57	8.91



Testing	# Records		# Goods		# Bads		Fraud Rate					
	25,580		25,085		495		0.0194					
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	256	41	215	16.02%	83.98%	256	41	215	0.16%	43.43%	43.27	0.19
2	256	140	116	54.69%	45.31%	512	181	331	0.72%	66.87%	66.15	0.55
3	255	217	38	85.10%	14.90%	767	398	369	1.59%	74.55%	72.96	1.08
4	256	236	20	92.19%	7.81%	1,023	634	389	2.53%	78.59%	76.06	1.63
5	256	232	24	90.63%	9.38%	1,279	866	413	3.45%	83.43%	79.98	2.10
6	256	245	11	95.70%	4.30%	1,535	1,111	424	4.43%	85.66%	81.23	2.62
7	256	247	9	96.48%	3.52%	1,791	1,358	433	5.41%	87.47%	82.06	3.14
8	255	253	2	99.22%	0.78%	2,046	1,611	435	6.42%	87.88%	81.46	3.70
9	256	254	2	99.22%	0.78%	2,302	1,865	437	7.43%	88.28%	80.85	4.27
10	256	249	7	97.27%	2.73%	2,558	2,114	444	8.43%	89.70%	81.27	4.76
11	256	256	0	100.00%	0.00%	2,814	2,370	444	9.45%	89.70%	80.25	5.34
12	256	254	2	99.22%	0.78%	3,070	2,624	446	10.46%	90.10%	79.64	5.88
13	255	252	3	98.82%	1.18%	3,325	2,876	449	11.47%	90.71%	79.24	6.41
14	256	251	5	98.05%	1.95%	3,581	3,127	454	12.47%	91.72%	79.25	6.89
15	256	251	5	98.05%	1.95%	3,837	3,378	459	13.47%	92.73%	79.26	7.36
16	256	256	0	100.00%	0.00%	4,093	3,634	459	14.49%	92.73%	78.24	7.92
17	256	256	0	100.00%	0.00%	4,349	3,890	459	15.51%	92.73%	77.22	8.47
18	255	252	3	98.82%	1.18%	4,604	4,142	462	16.51%	93.33%	76.82	8.97
19	256	255	1	99.61%	0.39%	4,860	4,397	463	17.53%	93.54%	76.01	9.50
20	256	254	2	99.22%	0.78%	5,116	4,651	465	18.54%	93.94%	75.40	10.00



OOT	# Records		# Goods		# Bads		Fraud Rate					
	12,232		11,935		297		0.0243					
	Bin Statistics					Cumulative Statistics						
Population Bin %	# Records	# Goods	# Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Goods	% Bads (FDR)	KS	FPR
1	122	24	98	19.67%	80.33%	122	24	98	0.20%	33.00%	32.80	0.24
2	123	85	38	69.11%	30.89%	245	109	136	0.91%	45.79%	44.88	0.80
3	122	88	34	72.13%	27.87%	367	197	170	1.65%	57.24%	55.59	1.16
4	122	107	15	87.70%	12.30%	489	304	185	2.55%	62.29%	59.74	1.64
5	123	111	12	90.24%	9.76%	612	415	197	3.48%	66.33%	62.85	2.11
6	122	112	10	91.80%	8.20%	734	527	207	4.42%	69.70%	65.28	2.55
7	122	110	12	90.16%	9.84%	856	637	219	5.34%	73.74%	68.40	2.91
8	123	110	13	89.43%	10.57%	979	747	232	6.26%	78.11%	71.86	3.22
9	122	116	6	95.08%	4.92%	1,101	863	238	7.23%	80.13%	72.90	3.63
10	122	118	4	96.72%	3.28%	1,223	981	242	8.22%	81.48%	73.26	4.05
11	123	117	6	95.12%	4.88%	1,346	1,098	248	9.20%	83.50%	74.30	4.43
12	122	120	2	98.36%	1.64%	1,468	1,218	250	10.21%	84.18%	73.97	4.87
13	122	115	7	94.26%	5.74%	1,590	1,333	257	11.17%	86.53%	75.36	5.19
14	122	120	2	98.36%	1.64%	1,712	1,453	259	12.17%	87.21%	75.03	5.61
15	123	122	1	99.19%	0.81%	1,835	1,575	260	13.20%	87.54%	74.35	6.06
16	122	120	2	98.36%	1.64%	1,957	1,695	262	14.20%	88.22%	74.01	6.47
17	122	121	1	99.18%	0.82%	2,079	1,816	263	15.22%	88.55%	73.34	6.90
18	123	123	0	100.00%	0.00%	2,202	1,939	263	16.25%	88.55%	72.31	7.37
19	122	121	1	99.18%	0.82%	2,324	2,060	264	17.26%	88.89%	71.63	7.80
20	122	121	1	99.18%	0.82%	2,446	2,181	265	18.27%	89.23%	70.95	8.23

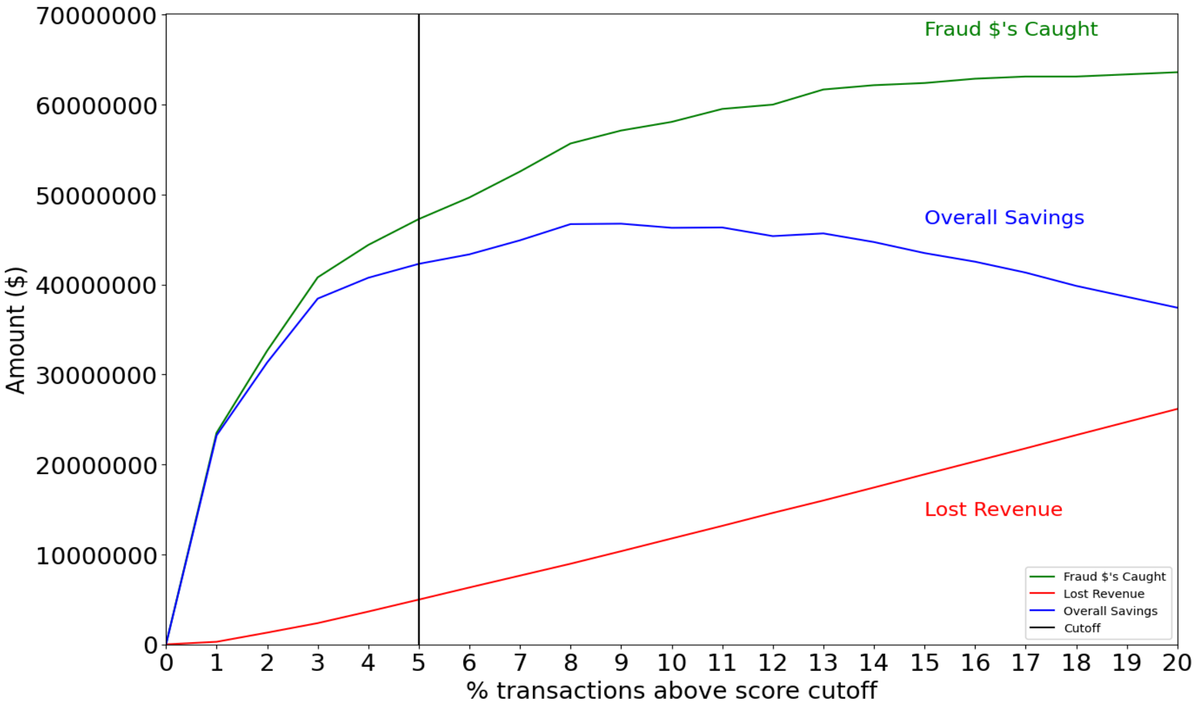
Section 8. Financial Curves and Recommended Cutoff

The green line on the plot represents Fraud $'s Caught—the deeper the cutoff, the more fraudulent transactions are captured. However, extending the cutoff also starts declining legitimate transactions, leading to Lost Revenue (red line). The blue line represents Overall Savings, which balances the fraud dollars caught with revenue lost from false positives.

Banks aim to balance fraud detection with customer experience, avoiding aggressive cutoffs that decline too many legitimate transactions. While maximizing fraud detection is important, operating at a point where savings decline sharply is risky. For instance, a 2% cutoff may capture more fraud, but it risks too many false positives, making it unstable. A 3% cutoff is more conservative but still near the edge of diminishing returns.

After analyzing the financial curves, I recommend setting the cutoff at 5%. This strikes an optimal balance between maximizing fraud detection (green line) and minimizing lost revenue (red line), leading to the highest overall savings (blue line). The 5% cutoff ensures most transactions are processed while achieving fraud prevention savings of over $40 million per year, aligning with the bank's goal of minimizing disruption while maintaining effective fraud detection.

1. Plot for three curves and Recommendation for a cutoff at 5%



Section 9. Summary

This project focused on developing a robust fraud detection system by utilizing advanced machine learning techniques. The dataset contained 97,852 card transaction records from 2010, with various fields capturing transaction-specific details. The key goal was to build a model that accurately detects fraudulent transactions while minimizing false positives, thereby optimizing both fraud prevention and customer experience.

The data preparation phase involved extensive cleaning, exclusion of irrelevant transaction types, and treatment of outliers to ensure model accuracy. Notably, outliers such as a $3 million legitimate transaction were removed to prevent skewing the model. Missing data were imputed logically using fields such as merchant descriptions, ensuring data completeness without compromising integrity.

A wide range of variables were engineered, including frequency, velocity, and behavioral metrics, to capture transactional irregularities indicative of fraud. The most critical variables—such as unique card counts and transaction velocities—proved highly effective in identifying fraudulent patterns.

For feature selection, we employed both Random Forest and LightGBM models. After comparing different configurations, the LightGBM model with 500 filters and 20 wrappers emerged as the optimal choice. The forward selection method helped refine the feature set, ensuring the model focused on the most predictive variables. LightGBM consistently outperformed other models, achieving a Fraud Detection Rate (FDR) of 57.24% in the top 3% of flagged transactions on the out-of-time (OOT) dataset.

In terms of financial impact, the recommended cutoff of 5% ensured a balance between fraud detection and revenue preservation. This cutoff was projected to save the client over $40 million annually by minimizing fraud losses while keeping legitimate transaction declines low. Overall, the LightGBM model provided a highly stable and accurate solution, delivering strong generalization across training, test, and OOT datasets.

In conclusion, the comprehensive approach—spanning data cleaning, variable creation, model exploration, and performance tuning—led to a highly effective fraud detection system. The final model not only exceeded the target FDR but also ensured significant financial savings, making it a valuable asset for the business. Further improvements could involve exploring additional feature engineering techniques or incorporating more advanced ensemble methods to push the model’s performance even higher.

Section 10. Appendix - DQR

1. Data Description

This dataset provides detailed records of card transactions, capturing key identifying information for each transaction. Spanning the entire year of 2010, it contains 10 fields, comprising both categorical and numerical data, with a total of 97,852 transaction records. These fields provide insights into transaction details such as card numbers, transaction dates, merchant information, transaction types, and fraud indicators.

2. Summary Statistics

2.1 Numerical Table

Field Name	# Records with value	% Populated	% Zeros	Min	Max	Mean	Std	Most Common value
Amount	97,852	100.00%	0	0.01	3,102,045.53	425.47	9,949.80	3.62

2.2 Categorical Table

Field Name	# Records with Values	% Populated	% Zeros	# Unique Values	Most Common Value
Recnum	97,852	100.00%	0	97,852	1
Cardnum	97,852	100.00%	0	1,645	5,142148452
Date	97,852	100.00%	0	365	2/28/10
Merchnum	94,455	96.53%	0	13,091	930090121224
Merch description	97,852	100.00%	0	13,126	GSA-FSS-ADV
Merch state	96,649	98.77%	0	227	TN
Merch zip	93,149	95.20%	0	4,567	38118
Transtype	97,852	100.00%	0	4	P
Fraud	97,852	100.00%	95,805	2	0

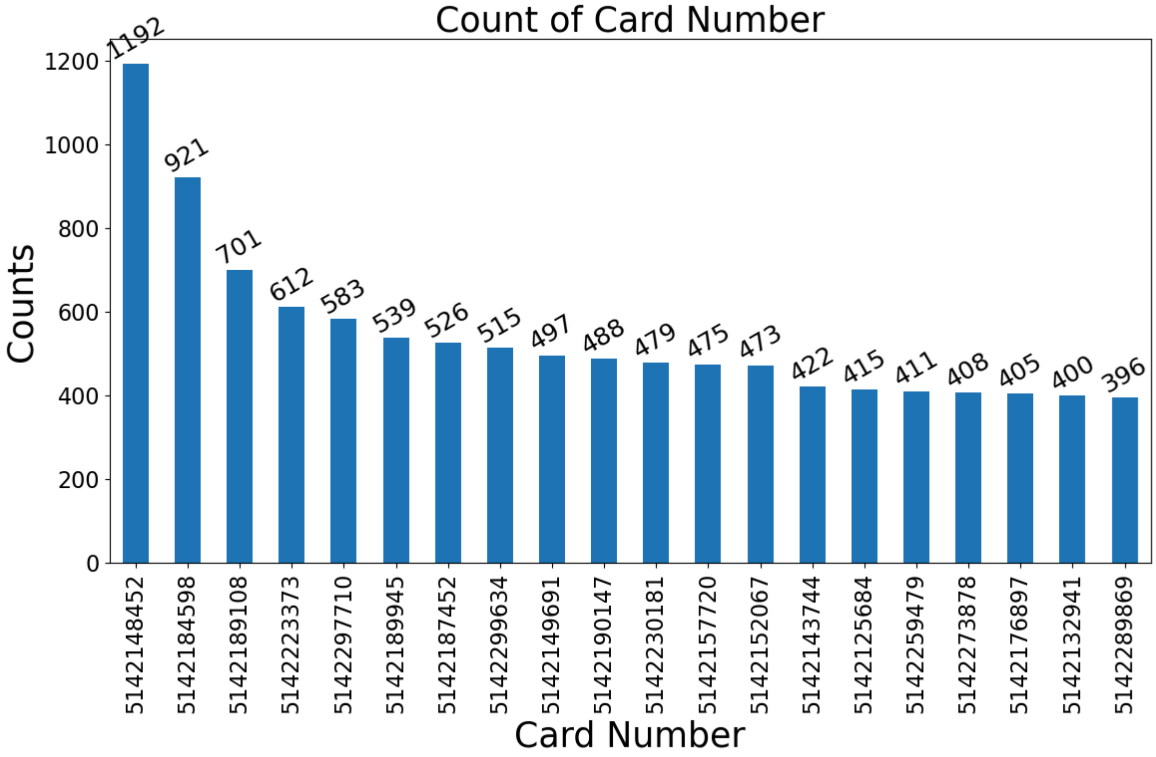
3. Visualization

3.1 Field Name: Recnum

Description: The Recnum field represents a unique identifier for each transaction in the dataset. Since each value in this field is distinct, it serves as a record number or index for tracking individual transactions. This field does not require a distribution plot, as it functions as a primary key, ensuring that every record can be uniquely identified. Its purpose is to maintain uniqueness and enable tracking of transactions.

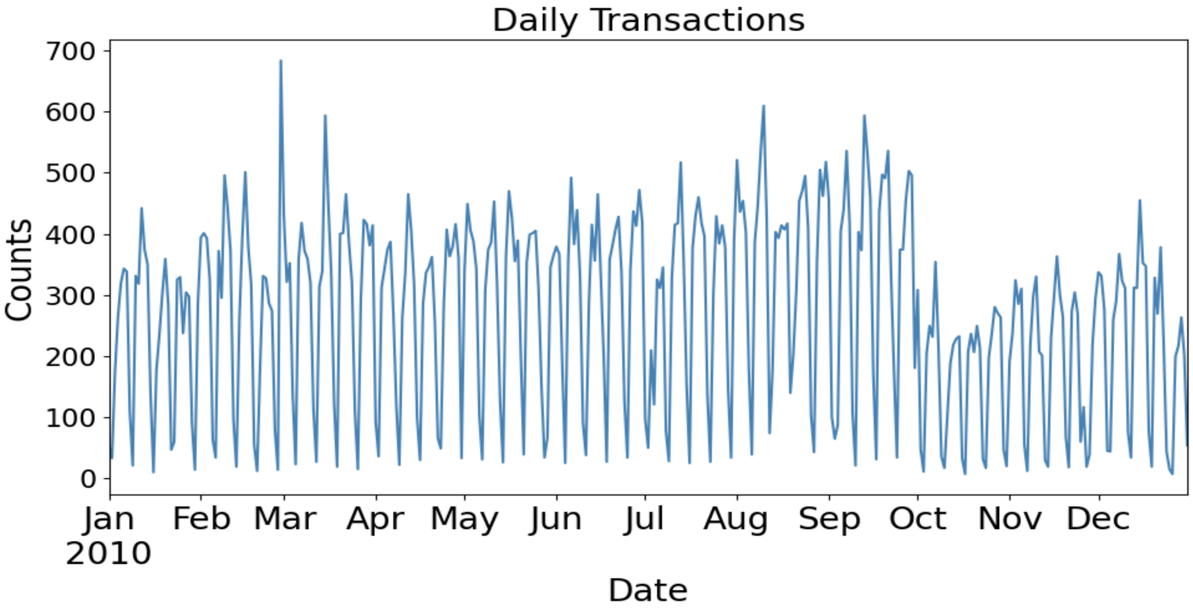
3.2 Field Name: Cardnum

Description: The Cardnum field represents unique card numbers associated with transactions. This bar chart displays the distribution of the top 20 most frequent card numbers. The card number 5142148452 appears most frequently with 1,192 transactions.



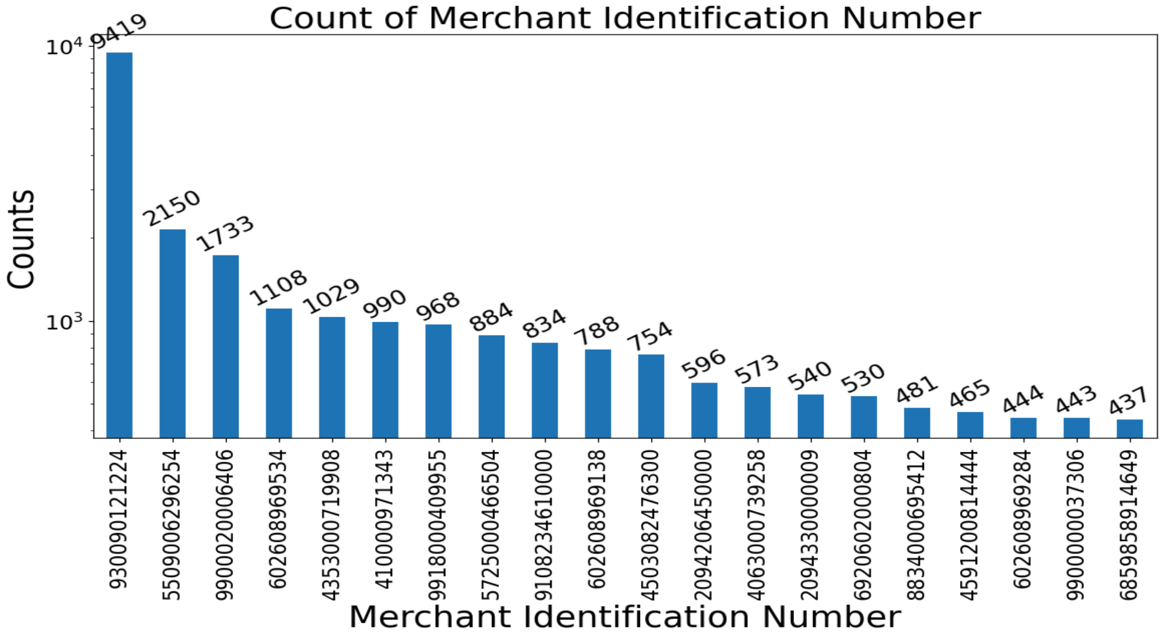
3.3 Field Name: Date

Description: The Date field represents the specific day on which each transaction occurred. This time series plot illustrates the daily number of transactions throughout the year 2010. Transaction volumes appear to fluctuate regularly, with some noticeable peaks, particularly in the first quarter and late summer.



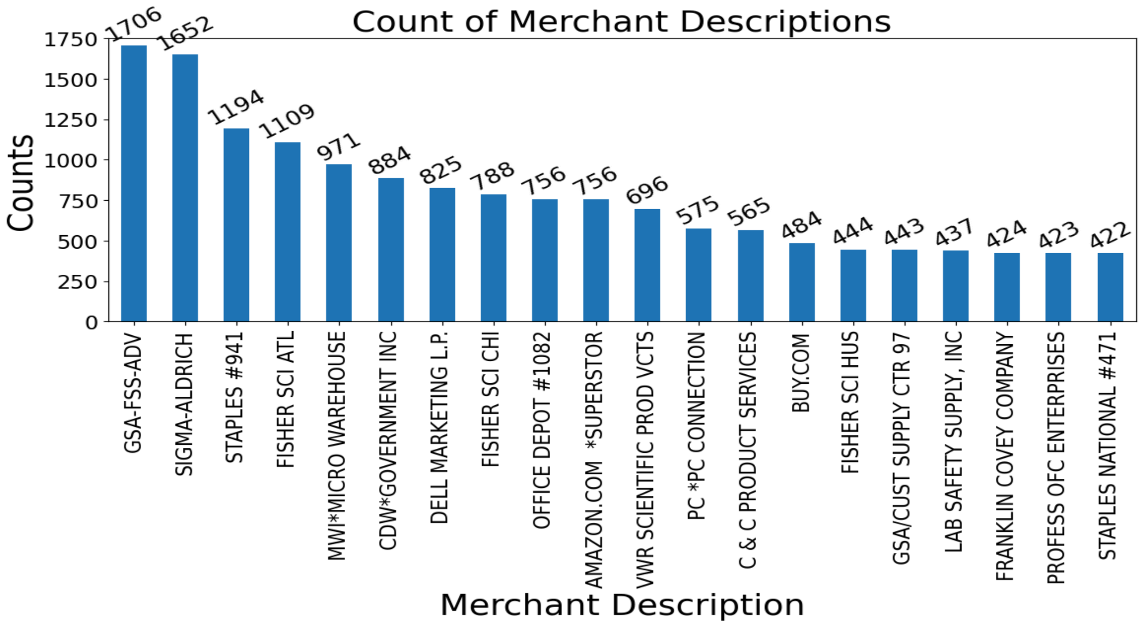
3.4 Field Name: Merchnum

Description: The Merchnum field represents unique merchant identification numbers associated with transactions. This histogram shows the distribution of the top 20 most frequent merchant numbers. The most frequent merchant number is 930090121224 with 9,419 transactions.



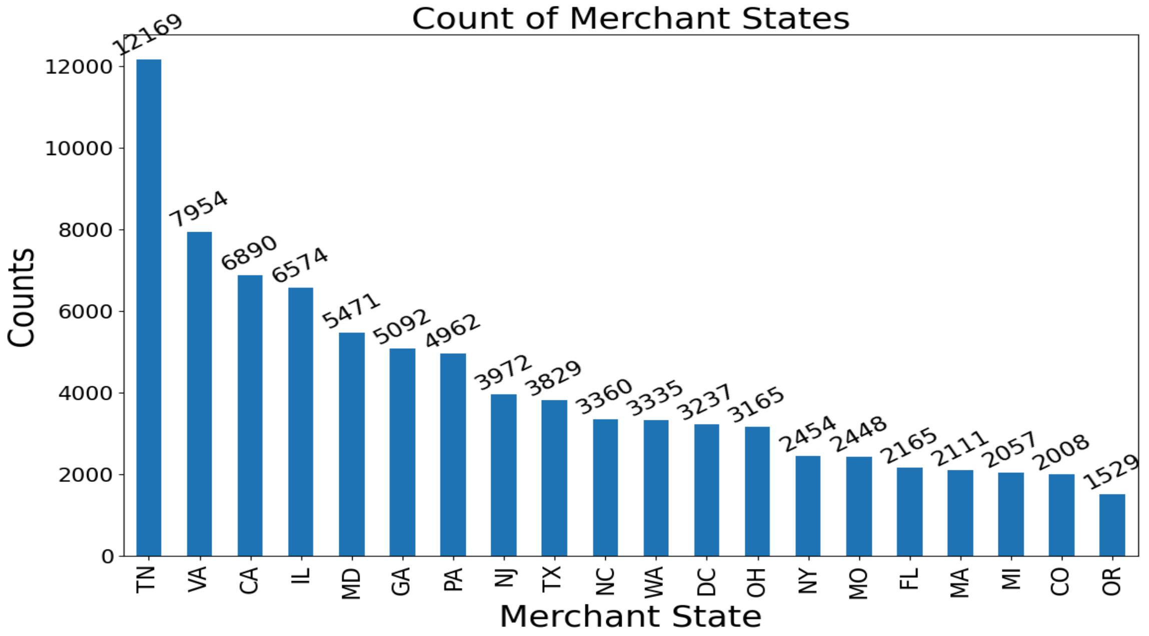
3.5 Field Name: Merch description

Description: The Merch description field represents the description or name of merchants associated with transactions. This histogram shows the distribution of the top 20 most frequent merchant descriptions. The most common merchant description is GSA-FSS-ADV, with 1,706 transactions, followed by SIGMA-ALDRICH with 1,652 transactions.



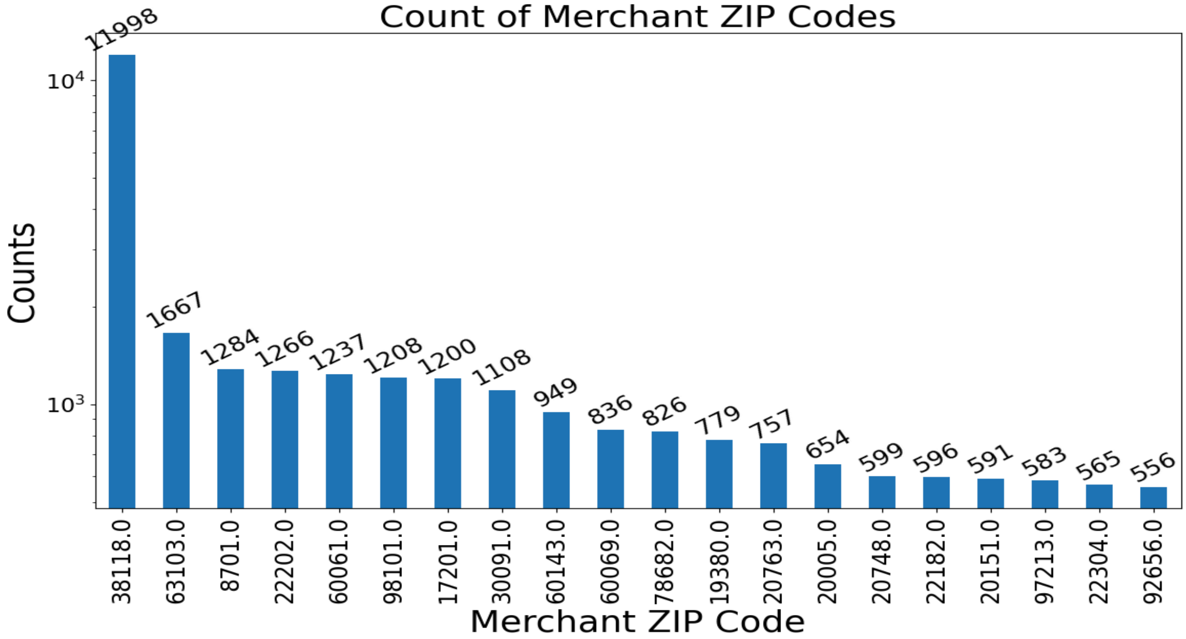
3.6 Field Name: Merch state

Description: The Merch state field represents the U.S. states where the merchants are located. This histogram displays the distribution of the top 20 most frequent merchant states. Tennessee (TN) has the highest number of transactions with 12,169 counts.



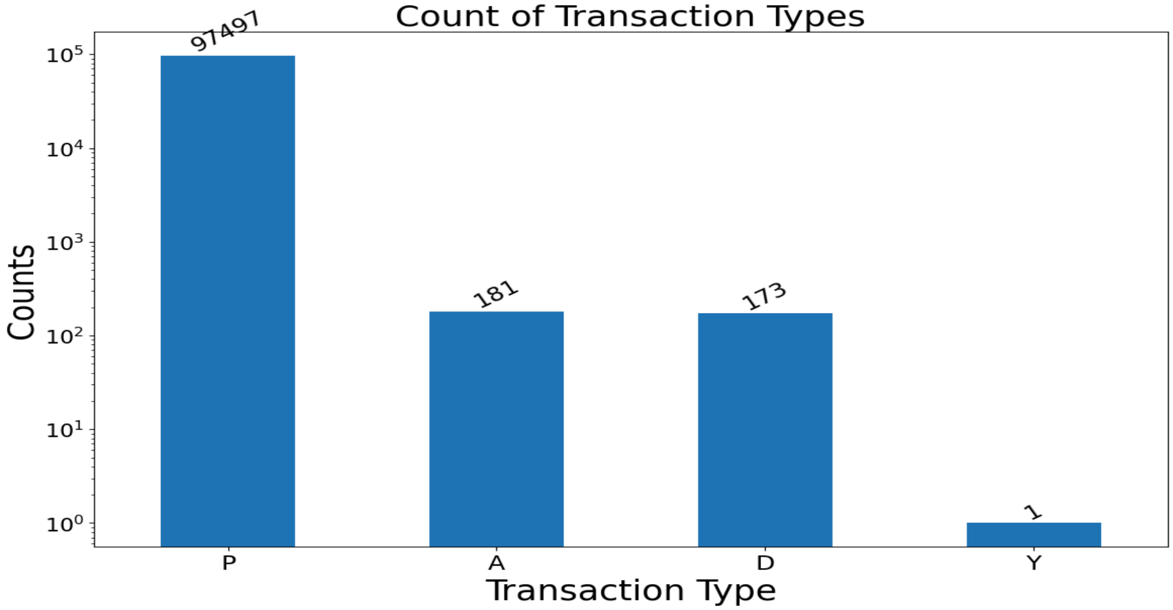
3.7 Field Name: Merch zip

Description: The Merch zip field represents the ZIP codes where the merchants are located. This histogram displays the distribution of the top 20 most frequent merchant ZIP codes. The most common ZIP code is 38118, with 11,998 transactions.



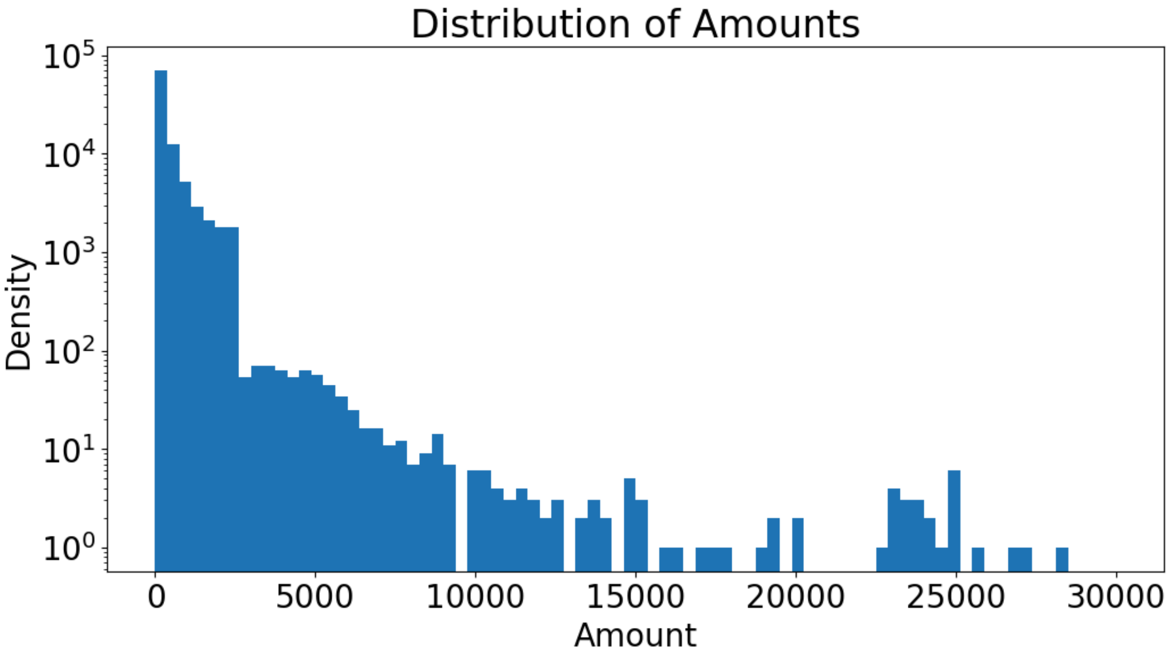
3.8 Field Name: Transtype

Description: The Transtype field represents the type of transaction. This bar chart shows the distribution of transaction types. The most frequent transaction type is “P”, with 97,497 transactions, indicating it is the dominant type.



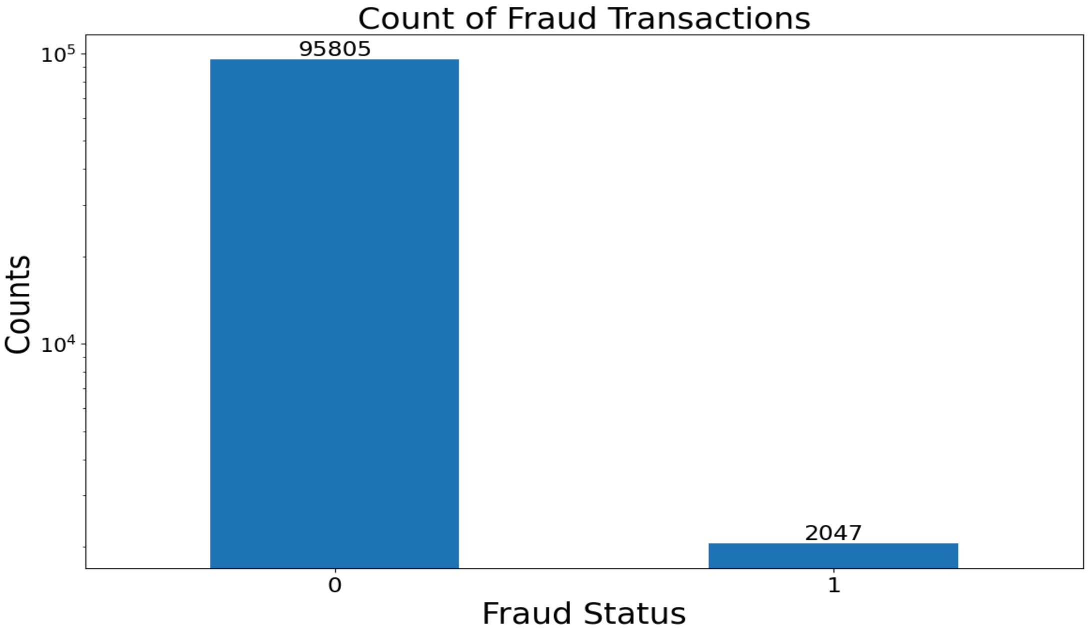
3.9 Field Name: Amount

Description: The Amount field represents the monetary value of each transaction in the dataset. The majority of transactions fall between 0 and 5,000, with a sharp decline in frequency as the amounts increase. Most of the data is concentrated in lower transaction amounts, creating a right-skewed distribution.



3.10 Field Name: Fraud

Description: The Fraud field indicates whether a transaction is fraudulent (1) or non-fraudulent (0). This bar chart shows the distribution of fraud status across all transactions. The vast majority of transactions, 95,805, are non-fraudulent, while 2,047 transactions are identified as fraudulent.


Count of Fraud Transactions
10^5
10^4
Counts
Fraud Status