Real-Time Credit Card Transaction Fraud Scoring Model

Table of Contents

1.0 Executive Summary

Credit card fraud poses a significant threat to financial integrity and operational security for institutions. Our project was initiated in response to the rising incidences of fraudulent transactions affecting a U.S. government agency in Tennessee. The agency's transaction dataset comprised approximately 100,000 records, each a potential vulnerability point for financial loss.

1.1 Objective: Our goal was to refine fraud detection mechanisms to mitigate risks efficiently and protect the agency's financial assets. This required a sophisticated analysis of transaction patterns and the implementation of advanced analytical techniques to enhance detection accuracy.

1.2 Approach: We employed a comprehensive methodology that involved:

- **Data Segmentation:** Partitioning the dataset into training, testing, and validation sets to ensure robust model training and unbiased evaluation.
- **Variable Development and Feature Selection:** Crafting and selecting predictive features to improve the model's sensitivity to fraudulent activities.
- **Model Implementation:** Applying a supervised machine learning algorithm specifically tuned to recognize and predict fraudulent transactions based on historical data and transaction characteristics.

1.3 Results: Our model demonstrated robust performance, achieving an out-of-time (OOT) fraud detection rate (FDR) of 55.7912%, marking a significant advancement in our ability to detect and prevent fraud efficiently. This level of accuracy estimates potential savings of approximately $44.66 million for the agency. Through detailed data quality reviews, cleansing, and strategic model development, we have established a new standard in fraud detection that not only delivers considerable financial benefits but also reinforces transaction security.

2.0 Data Description

The dataset comprises **transaction records** from company-issued credit cards used for business-related expenses within a U.S. government organization in Tennessee. It encompasses a total of **97,852 records**, each characterized by **10 distinct fields** pertinent to the transactional details and cardholder information.

2.1 Summary Tables:

Numeric Fields Table

Field Name	Field Type	# Records Have Values	% Populated	# Zeros	Min	Max	Mean	Stdev	Most Common
Cardnum	numeric	97,852	100.00%	0	5142110002	5142847398	5142201750	55560	5142148452
Merch zip	numeric	93,149	95.20%	0	1	99999	44684	28371	38118
Amount	numeric	97,852	100.00%	0	0.01	3,102,045	425	9949	3.62

Categorical Fields Table

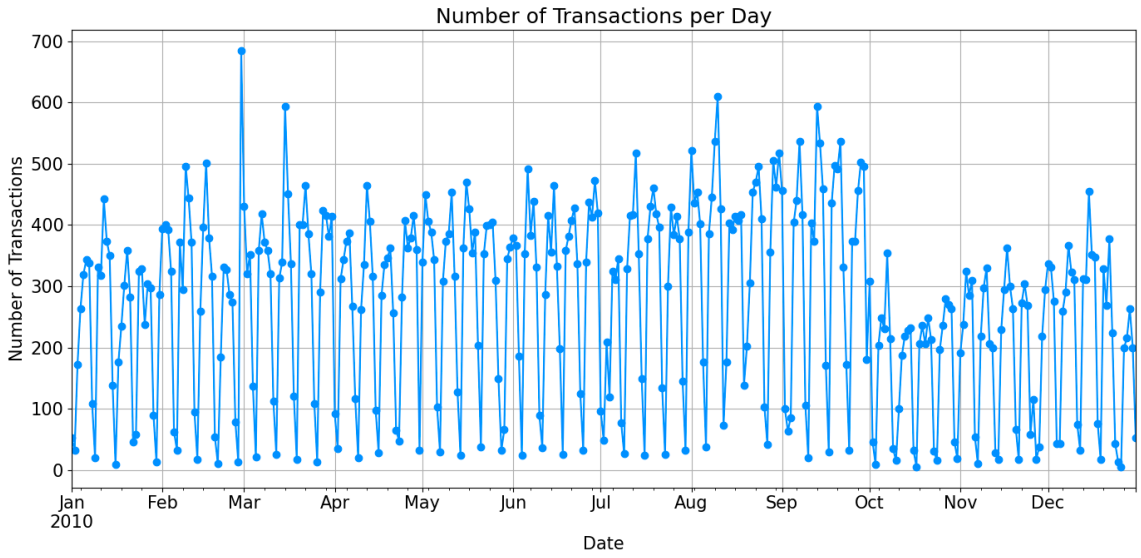
Field Name	# Records Have Values	% Populated	# Zeros	# Unique Values	Most Common
Date	97,852	100.00%	0	365	2/28/10
Merchnum	94,455	96.50%	0	13,091	930090121224
Merch description	97,852	100.00%	0	13,126	GSA-FSS-ADV
Merch state	96,649	98.80%	0	227	TN
Transtype	97,852	100.00%	0	4	P
Recnum	97,852	100.00%	0	97,852	1
Fraud	97,852	100.00%	95805	2	0

2.2 Some Important Field Distributions

The analysis of key field distributions is critical in understanding the underlying patterns and anomalies within the data, which can significantly inform our fraud detection strategies. Below are the distributions for some of the crucial fields from the dataset

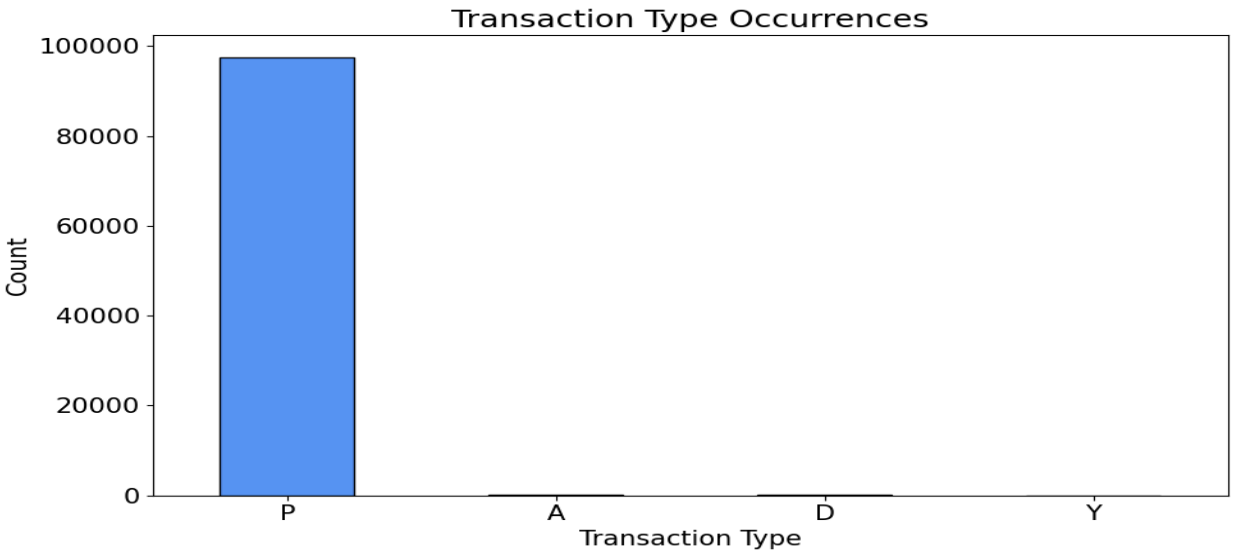
1) Field Name: Date

The transaction date can provide insights into cyclical trends, peak activity periods, and any correlations between transaction timing and fraud incidence. Analysis may reveal patterns such as higher fraud on specific days of the week or times of the month, possibly correlating with lower staffing levels or other operational vulnerabilities.



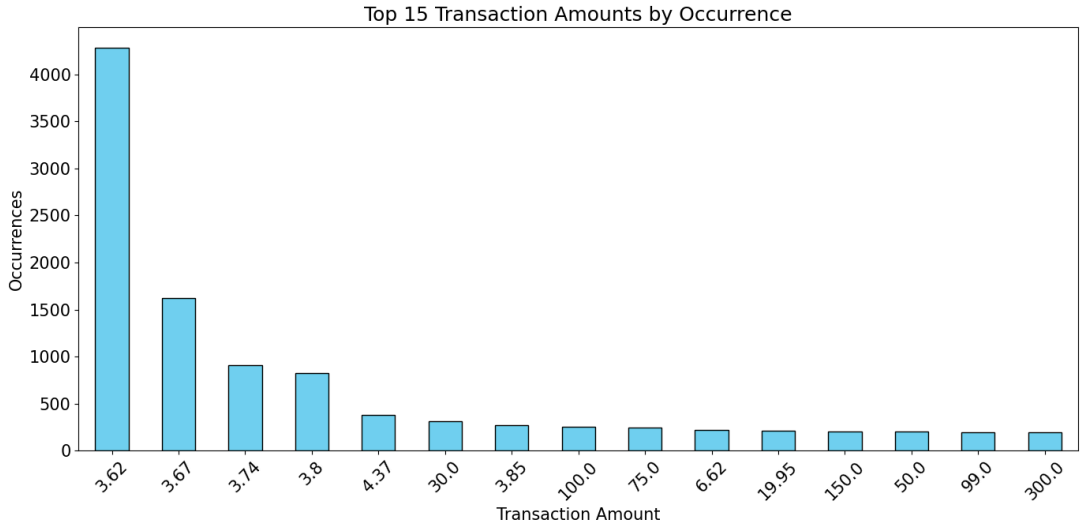
2) Field Name: Transtype

This field categorizes the nature of each transaction processed on the company's credit cards. The types of transactions include purchases, refunds, and others, which are pivotal for understanding spending behaviors and identifying patterns that may indicate fraud.
The dominant transaction type is 'P' for purchases, totaling 97,497 transactions, which is typical for company-issued cards used primarily for business expenses. Analyzing transaction types can reveal fraud indicators, such as unusual patterns of refunds or other non-purchase transactions. For example, a high number of refunds could suggest potential fraud through schemes like over-purchasing followed by returns.



3) Field Name: Amount

This field indicates the monetary value of the transaction. Analyzing the amounts can help identify outliers or transactions that deviate from typical spending patterns. Large transactions, or those above a certain threshold, could be automatically flagged for review. Similarly, an aggregation of small, repetitive transactions from the same card or merchant might indicate a 'smurfing' type of fraud.
The monetary value of the transaction. The most common being 3.62 with a total count of 4283.



3.0 Data Cleaning

In the process of preparing a transactional dataset for analysis, it is critical to ensure the cleanliness and completeness of the data. This involves a series of steps to handle invalid entries, missing values, and potential outliers. Below, we delve into the specific procedures implemented for cleaning and imputing missing data across several key fields: **Merchnum**, **Merch state**, and **Merch zip**. Each field presents its unique challenges, ranging from the straightforward replacement of invalid entries to the more complex task of generating new unique identifiers. The outlined methodology employs both deterministic and inferential techniques to maintain data integrity and facilitate robust downstream analyses.

To begin we:

1. **Remove Empty Columns**: Deletes any columns fully consisting of missing values.
2. **Convert Date Format**: Transforms the 'Date' column into a datetime format to facilitate date-based operations.
3. **Filter Data**: Keeps only transactions with type 'P' (likely purchases). Excludes transactions exceeding $3,000,000 to remove outliers or errors.

3.1 Field Specific Cleaning

3.1.1 Merchnum Field

1) Exclusions:
Invalid Merchnum values that were represented as '0' are treated as missing data by replacing them with NaN.

2) Outlier Treatment:
There was no outlier treatment needed for the Merchnum field, as the focus was on handling missing data.

3) Imputation Process:
Imputation began by creating a mapping from the Merch description to the corresponding Merchnum for records where both fields were non-null. Missing Merchnum values were initially imputed by looking up this mapping with the Merch description. Where the Merch description was associated with adjustment transactions, the Merchnum was set to a placeholder value of 'unknown'. For Merch description values without a direct mapping, unique Merchnum identifiers were created and assigned, ensuring each unique description had a corresponding merchant number.

3.1.2 Merch State Field

1) Exclusions:
There were no exclusions applied to the Merch state field, as the goal was to fill in missing data without removing any records.

2) Outlier Treatment:
Outlier treatment was not directly addressed for the Merch state field; the process was more about ensuring data completeness and standardization.

3) Imputation Process:
A dictionary mapping from Merch zip to Merch state was created for cases where the zip code was available but the state was not. This allowed for the imputation of state values based on zip codes. Missing Merch state values were also attempted to be filled using mappings from both Merchnum and Merch description to Merch state, but this had limited success in reducing the number of missing states. For retail adjustment transactions, the Merch state was set to 'unknown'. Non-U.S. state abbreviations or unrecognized state values in the Merch state field were reclassified as 'foreign'. Any remaining missing Merch state values after these steps were filled in as 'unknown' to complete the field.

3.1.3 Merch Zip Field

1) Exclusions:
No exclusions were applied to the Merch zip field; all records were retained for cleaning.

2) Outlier Treatment:
The outlier treatment process was not applicable to the Merch zip field.

3) Imputation Process:
The process of imputation for Merch zip included creating and utilizing mappings from both Merchnum and Merch description to the non-null Merch zip values to fill in gaps where possible. For records with a known Merch state but a missing Merch zip, the most populous zip code for the given state was imputed, using an external source for the most populous zip codes. After applying these imputation methods, any remaining missing Merch zip values were set to 'unknown' to ensure no missing values remained in the dataset.

Across these fields, the data cleaning process involved detailed attention to missing data, applying a mix of direct mapping, conditional imputation, and the assignment of placeholder values where direct imputation was not possible. This careful approach was designed to preserve the integrity of the dataset while preparing it for further analysis or modeling.

4.0 Variable Creation

The creation of new variables is a critical step in enhancing the model's ability to detect fraudulent transactions by enriching the dataset with more informative features. These variables were designed to encapsulate patterns and anomalies in transaction data that are typically indicative of fraudulent activity. By encoding complex relationships and historical data trends, these variables significantly improve the model's predictive accuracy.

This structured approach ensures clarity and comprehensiveness, allowing stakeholders to understand the rationale and effectiveness of each variable type in combating fraud. If you have visualizations or specific statistics that illustrate the impact of these variables, including them can further enhance the understanding of their importance.

4.1 Summary of Created Variable

Description	# Variables created

Target encoded Variables: Merch state_TE: Represents the average fraud rate associated with each merchant state. Merch zip_TE: Represents the average fraud rate associated with each merchant zip code. Dow_TE: Represents the average fraud rate for each day of the week.	3
Day-Since Variables: the number of days since the last recorded transaction for each entity.	23
Frequency Variables: count of the number of transactions involving each entity within specified time windows (0, 1, 3, 7, 14, 30, and 60 days).	161
Amount Variables: The monetary values associated with transactions involving each entity, calculated over the same specified time windows as the frequency variables.	1288
Ratio Variables: Count Ratio Variables: comparison of the daily number of transactions on a specific day (d) to the average over a longer period (dd) Total Amount Ratio Variables: daily spending against the average over more extended periods	368
Velocity Ratios: The number of transactions with the same entity over the last {0,1} day divided by the number of transactions with those same entities over the last {7,14,30,60} days, adjusted for the number of days since the last transaction plus one.	184
Variability Metrics: the average, maximum, and median differences in transaction amounts for the same entity over the last {0,1,3,7,14,30} days	414
Unique Entity Interaction Counts: Calculates the number of unique instances of one entity with another within given time windows (1, 3, 7, 14, 30, 60 days).	880
Amount Categorization: Segments transaction amounts into five quantiles	1
Foreign Merchant Indicator: Identifies whether the merchant's ZIP code is outside of the standard US ZIP codes.	1

5.0 Feature Selection

Objective:
The feature selection process was designed to optimize the predictive model's efficiency and effectiveness by identifying the most influential variables. This optimization aimed to achieve an ideal balance between model complexity and performance, ensuring high accuracy while maintaining manageable computational demands.

Methodology Overview:
Feature selection was executed using a combination of forward and backward stepwise selection techniques. These methods were applied across three distinct classifiers: LightGBM, Random Forest, and CatBoost. The choice of these methods and classifiers was strategic, intended to rigorously test each feature's contribution under various modeling approaches and settings.

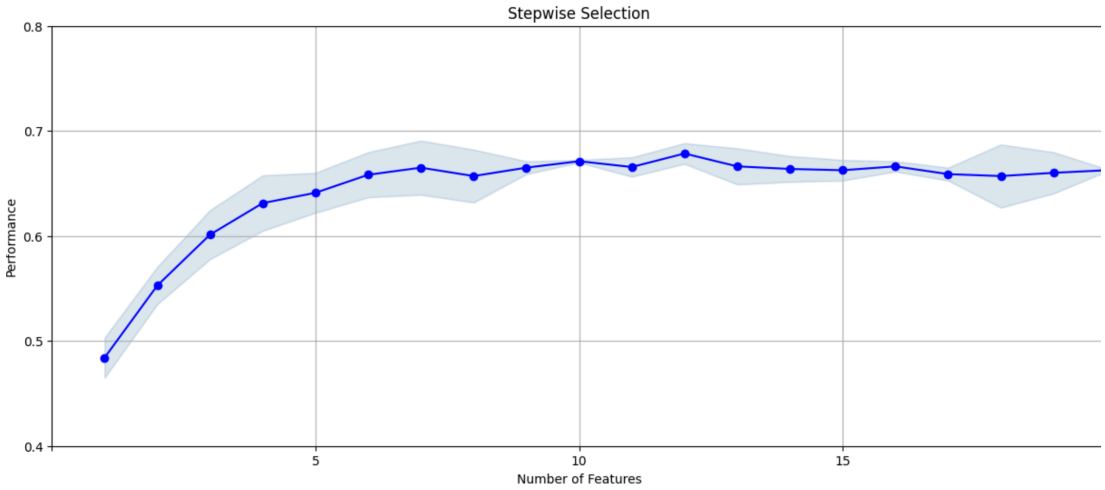
- **Forward Stepwise Selection**: Begins with no variables and adds them one by one, each time adding the variable that provides the most significant improvement to the model performance until no further significant improvement is noted.
- **Backward Stepwise Selection**: Starts with all potential variables and systematically removes the least significant, assessing the impact on the model's performance to retain only those that contribute positively.

This feature selection exercise aimed to identify a set of 20 variables that could provide an optimal balance between complexity and performance for a predictive model. The performance metric of choice was the False Discovery Rate (FDR), aiming to maximize the predictive accuracy while minimizing false positives.

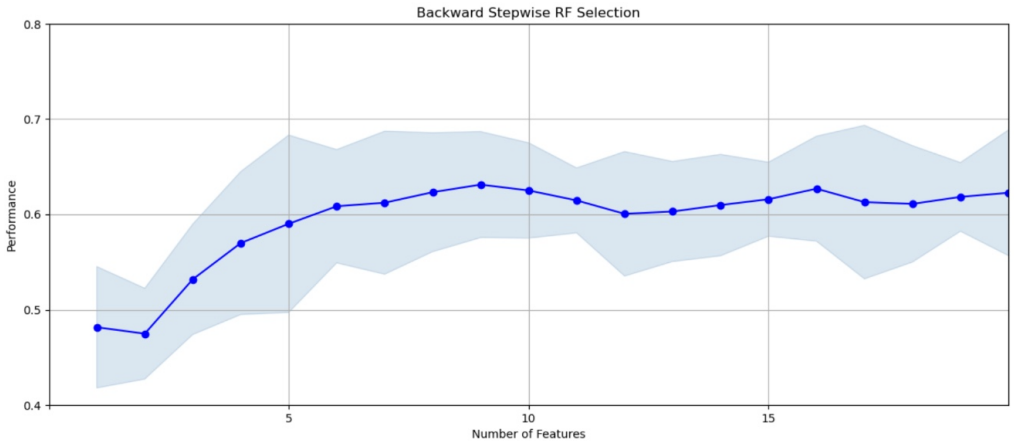
5.1 Detailed Performance Analysis

Random Forest Classifier:

- **Forward Selection**: Demonstrated a rapid improvement in performance with the initial addition of features, reaching a plateau relatively quickly. This indicates that a few features have substantial predictive power.



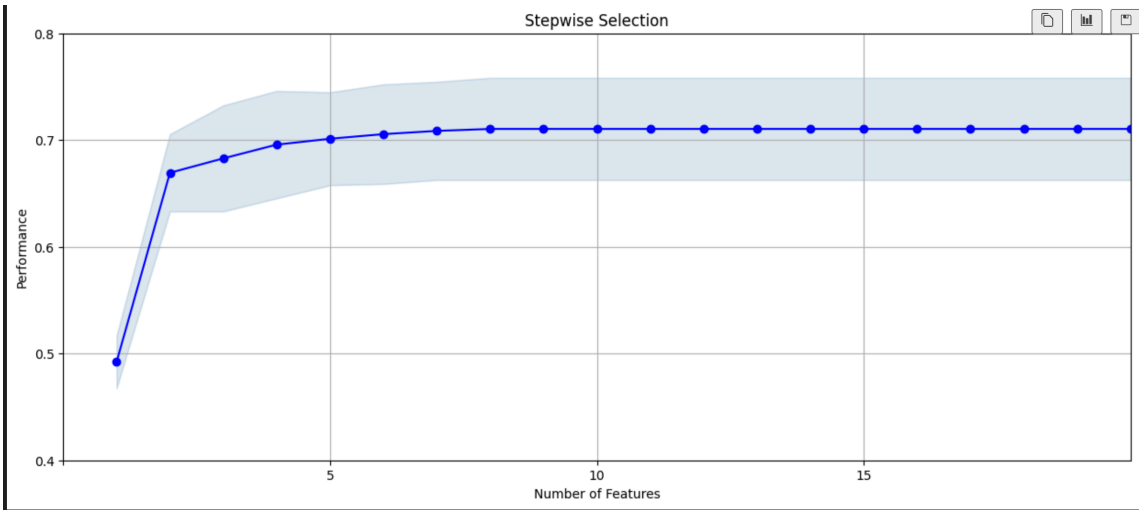
- **Backward Selection**: Presented a more gradual increase in performance as less informative features were removed. The plateau was broader, suggesting that while the most predictive features were crucial, additional features contributed to the model's performance, albeit to a lesser extent.



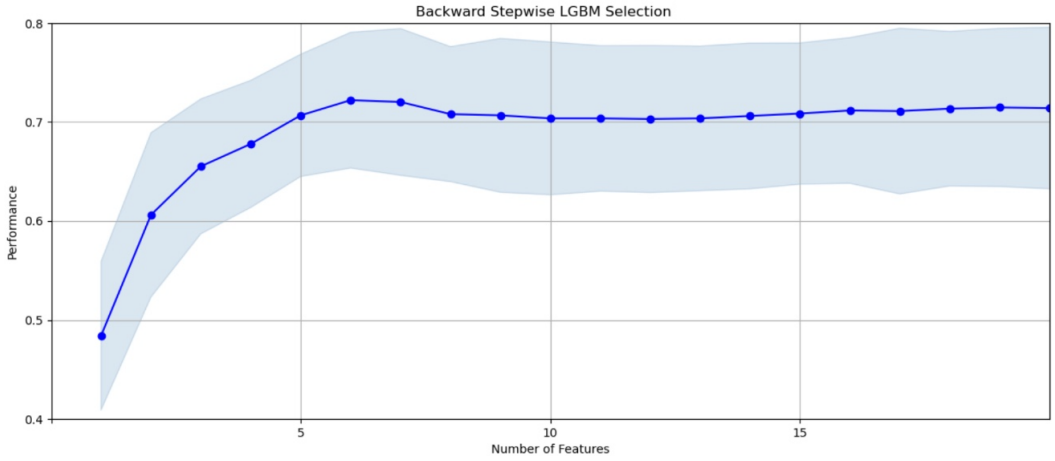
The Random Forest model, while robust, was not selected due to the plateauing of performance gains, indicating that after a certain point, additional features did not significantly enhance the model's predictive power.

LightGBM Classifier:

- **Forward Selection**: The performance increased sharply with the inclusion of the first few features and then leveled off. This was indicative of the LightGBM's efficiency in identifying and leveraging the most predictive features early in the selection process.



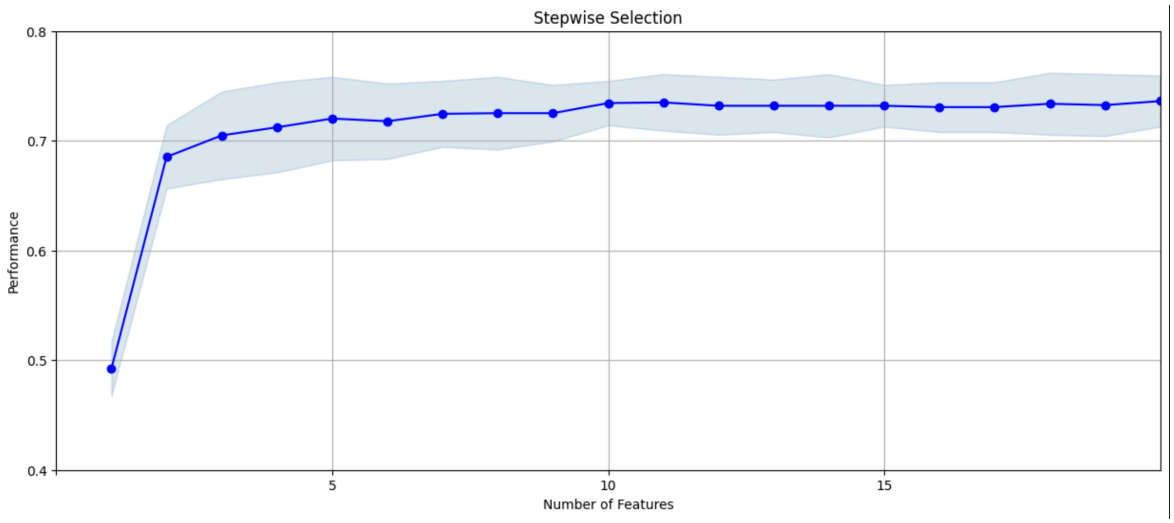
- **Backward Selection**: Showed an analogous pattern to Random Forest, with a strong start and a tapering performance, reinforcing the findings of the most impactful features.



Despite its speed and efficiency, LightGBM was not chosen due to a slightly lower performance plateau compared to CatBoost, suggesting that while it quickly captures feature importance, it may not be as effective in utilizing the full spectrum of features for maximum prediction accuracy.

CatBoost Classifier:

- **Forward Selection**: Showcased a consistent and smooth improvement in performance across the feature selection process. The model continued to benefit from the inclusion of additional features beyond the point where other models' performance stabilized.



The CatBoost model's higher overall performance reflects its capability to handle a wide variety of feature interactions, including categorical data without extensive preprocessing. Its algorithm minimizes overfitting without a significant need for hyperparameter tuning, making it well-suited for datasets with complex patterns.

5.2 Model for Feature Selection

Given the consistent and balanced improvement in model performance with an increasing number of features, along with the highest FDR, the CatBoost model was selected as the most suitable for the final feature set. Its ability to extract value from a wider range of features makes it a robust choice for the predictive model, suggesting that it is well-tuned to handle both strong and subtle patterns within the data.

5.3 Final Variable Selection

Based on the analysis, the following 20 variables were selected from the CatBoost model, ordered by their importance as determined by the wrapper method and accompanied by their filter scores:

wrapper order	variable	filter score
1	Cardnum_unique_count_for_card_state_1	0.476066612
2	Card_Merchdesc_total_7	0.32463085
3	card_state_max_3	0.341323382
4	Card_dow_count_0_by_30_sq	0.327031284
5	merch_state_total_1	0.304893208
6	card_state_max_14	0.305945894
7	Cardnum_count_1_by_14	0.41385987
8	card_zip_count_1_by_60_sq	0.314821538

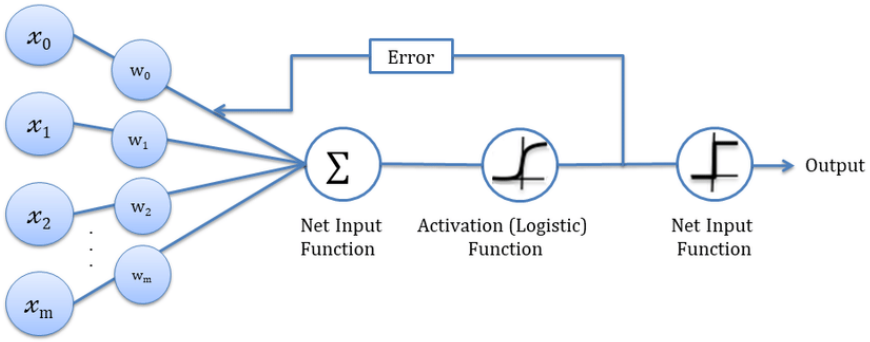
9	Card_dow_unique_count_for_merch_state_1	0.447357298
10	Cardnum_total_3	0.621931866
11	Merchnum_desc_total_1	0.304969274
12	Merchdesc_dow_total_7	0.286146645
13	Card_dow_count_0_by_60_sq	0.354881813
14	Cardnum_vdratio_1by14	0.485430712
15	Card_dow_unique_count_for_merch_zip_1	0.447156912
16	Cardnum_day_since	0.432169134
17	Cardnum_total_0	0.591317765
18	Cardnum_count_0	0.516123246
19	Card_dow_actual/max_14	0.289426732
20	Card_dow_unique_count_for_merch_zip_7	0.418930521

6.0 Preliminary Model Exploration

We explored various machine learning algorithms to identify the most effective approach for detecting credit card fraud. Below are brief descriptions and visual representations for each model tested. In the preliminary phase of our analysis, various machine learning algorithms were tested to identify the most effective approach for detecting fraudulent transactions. Each model was evaluated based on its ability to generalize well to unseen data and its robustness against overfitting. Below is a summary of the models explored along with hyperparameters adjustments and the rationale behind each choice.

6.1 Brief Descriptions of Each Machine Learning Algorithm

1. Logistic Regression

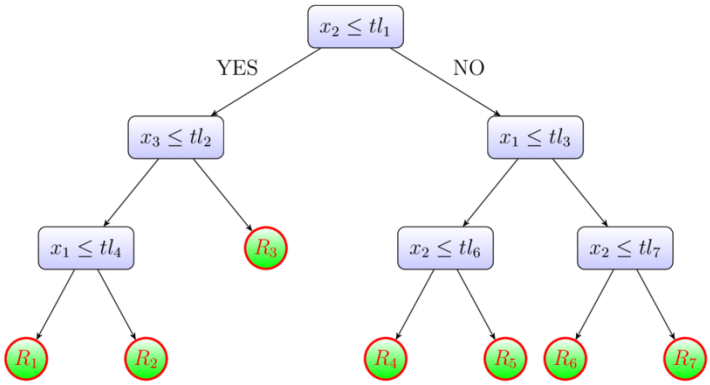


Logistic Regression is a statistical model that predicts outcomes for a binary dependent variable based on one or more independent variables. It is used to estimate the probability of an event occurring by fitting data to a logistic curve. The model calculates the odds of the dependent variable in terms of the predictors using

a logistic function, making it especially useful for binary classification tasks like fraud detection.

- **Key Variables**: Utilizes penalty types (L1, L2) and different solvers (lbfgs, saga) to explore the effects of regularization and optimization.
- **Hyperparameters**: The 'C' parameter was varied to adjust the strength of regularization, influencing the model's complexity and its ability to generalize.

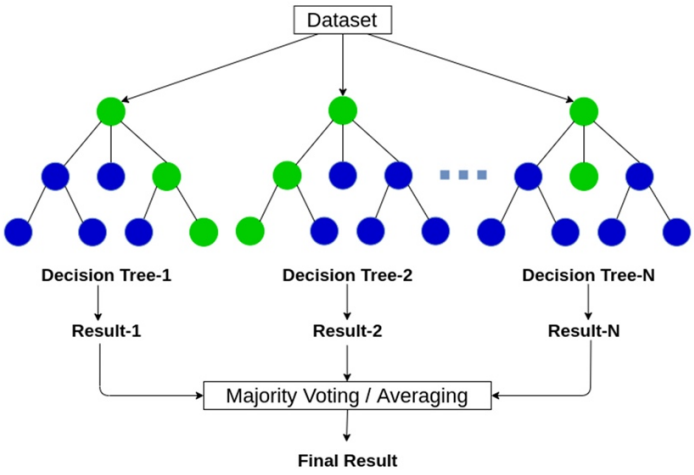
2. Decision Trees



A Decision Tree is a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. It splits the dataset into branches to form a tree structure. Each node in the tree acts as a decision point that splits the data based on a certain condition, leading to highly interpretable decision-making processes.

- **Key Variables**: Focuses on split quality metrics (gini, entropy) and the best strategies for node splitting.
- **Hyperparameters**: Adjusted 'max_depth', 'min_samples_split', and 'min_samples_leaf' to control tree complexity and prevent overfitting.

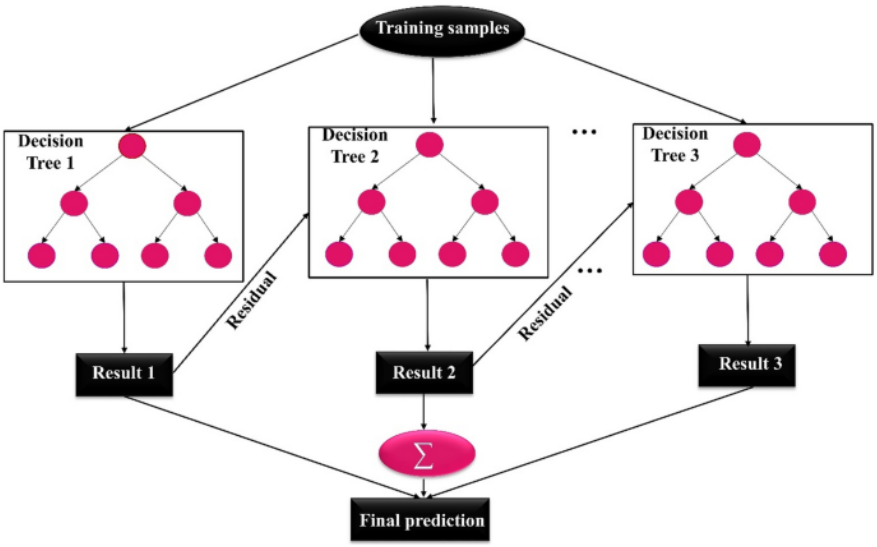
3. Random Forests



Random Forest is an ensemble learning technique for classification and regression that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the Random Forest is the class selected by most trees. It improves upon the simplicity of decision trees by adding randomness in the tree generation process, which helps in reducing overfitting and enhancing the generalization capabilities of the model.

- **Key Variables**: Modifies 'n_estimators' to define the forest size and 'criterion' for assessing split quality.
- **Hyperparameters**: Incorporates adjustments similar to Decision Trees but applies them across a forest to enhance stability and accuracy.

4. LightGBM (Boosted Trees)

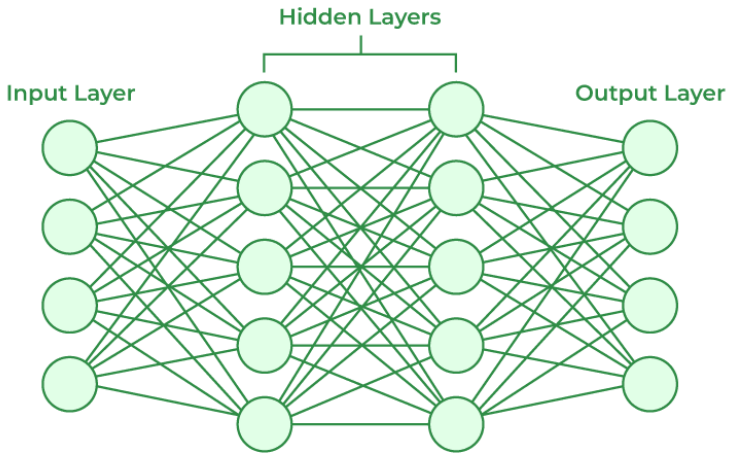


LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is designed for distributed and efficient training, particularly with

large datasets. LightGBM improves upon traditional gradient boosting techniques by using a histogram-based algorithm for speed and efficiency, and it handles large amounts of data with a lower memory footprint. The model produces highly accurate results, even on imbalanced data, making it a powerful tool for complex datasets like transaction data.

- **Key Variables**: Manages 'num_leaves', 'max_depth', and 'learning_rate' to moderate the growth rate and pace of learning.
- **Hyperparameters**: 'n_estimators' is varied to manage the number of boosting rounds, affecting the model's performance and vulnerability to overfitting.

5. **Neural Networks**



Neural Networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling, or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text, or time series, must be translated. Neural Networks help to cluster and classify information as they can learn to model non-linear and complex relationships between inputs and outputs.

- **Key Variables**: Configures 'hidden layers' and 'activation' types (relu, tanh) to explore different network architectures.
- **Hyperparameters**: Alters 'solver' for optimization, 'alpha' for L2 regularization strength, and 'max_iter' for the number of training iterations.

6.2 Exploring Model Hyperparameters

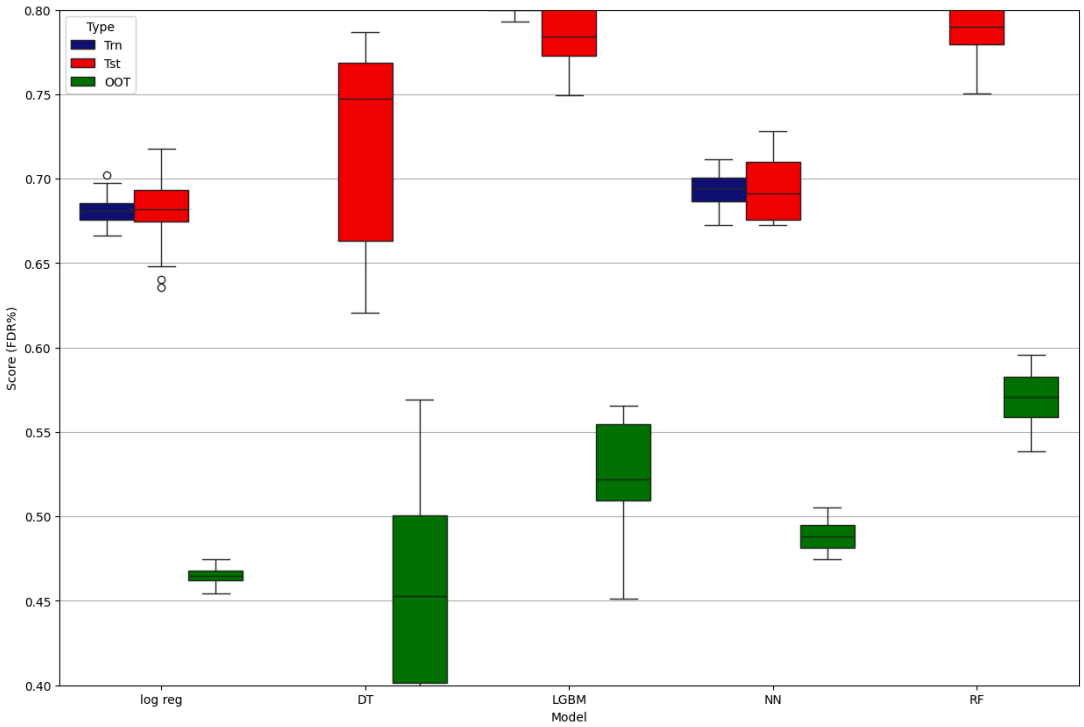
For each algorithm, hyperparameters were meticulously tuned to enhance model performance and mitigate overfitting. The table below summarizes the selected hyperparameters for the final models:

Algorithm	Key Hyperparameters
Logistic Regression	Penalty: L2, Solver: saga, C: 1.0
Decision Trees	Criterion: gini, Max Depth: 7, Min Samples Split: 20, Min Samples Leaf: 5
Random Forests	n_estimators: 60, Criterion: gini, Max Depth: 7, Min Samples Split: 20, Min Samples Leaf: 10
LightGBM	num_leaves: 31, Max Depth: 7, Learning Rate: 0.05, n_estimators: 100
Neural Networks	Hidden Layers: (50,), Activation: tanh, Solver: sgd, Alpha: 0.001, Max Iterations: 300

Hyperparameters selected for the final models are pointed out.

Model	Parameter						Average FDR at 3%		
Logistic Regression	Number of variables	penalty	C	solver		L1 ratio	Train	Test	OOT
1	5	l2	1	lbfgs		None	0.684227	0.676117	0.466667
2	5	l1	1	saga		None	0.67687	0.689793	0.46532
3	10	l1	1	saga		None	0.678354	0.654017	0.4781
4	10	l2	1	lbfgs		None	0.679138	0.685175	0.46532
Decision Tree	Number of variables	criterion	splitter	max_depth	min_samples_split	min samples leaf	Train	Test	OOT
1	5	gini	best	3	2	1	0.617263	0.629102	0.461728
2	5	entropy	best	10	10	5	0.828524	0.683879	0.46734
3	5	gini	best	25	20	10	0.871936	0.671836	0.462812
4	10	gini	best	3	2	1	0.667192	0.627182	0.461812
5	10	entropy	best	10	10	5	0.889101	0.718922	0.402917
6	10	gini	best	25	20	10	0.886104	0.760739	0.463973
Random Forest	Number of variables	n_estimators	criterion	max_depth	min_samples_split	min samples leaf	Train	Test	OOT
1	5	3	gini	3	2	1	0.657794	0.660059	0.442761
2	5	30	entropy	10	10	5	0.837958	0.72062	0.476768
3	5	60	gini	25	20	10	0.853594	0.726999	0.480471
4	10	3	gini	3	2	1	0.668258	0.667974	0.46229
5	10	30	entropy	10	10	5	0.882585	0.794887	0.552189
6	10	60	gini	25	20	10	0.894019	0.795464	0.557912
LGBM	Number of variables	num_leaves	max_depth	learning_rate		n_estimators	Train	Test	OOT
1	5	4	2	0.1		10	0.657794	0.689156	0.468687
2	5	10	5	0.05		100	0.710108	0.722297	0.443294
3	5	10	5	0.01		300	0.779543	0.722837	0.466709
4	10	4	2	0.1		10	0.706757	0.699015	0.482492
5	10	10	5	0.05		100	0.805064	0.777988	0.525253
6	10	10	5	0.01		300	0.738439	0.700318	0.50233
NN	Number of variables	Hidden layers	activation	solver	alpha	max iter	Train	Test	OOT
1	5	(3,)	relu	adam	0.0001	200	0.680609	0.688668	0.453199
2	5	(30,)	tanh	adam	0.0001	100	0.758933	0.760739	0.463973
3	5	(30,30)	relu	sgd	0.001	300	0.685689	0.699015	0.482828
4	10	(3,)	relu	adam	0.0001	200	0.69677	0.697519	0.473064
5	10	(30,)	tanh	adam	0.0001	100	0.758933	0.760739	0.525253
6	10	(30,30)	relu	sgd	0.001	300	0.739161	0.681056	0.46129

6.3 Optimal Model Selection



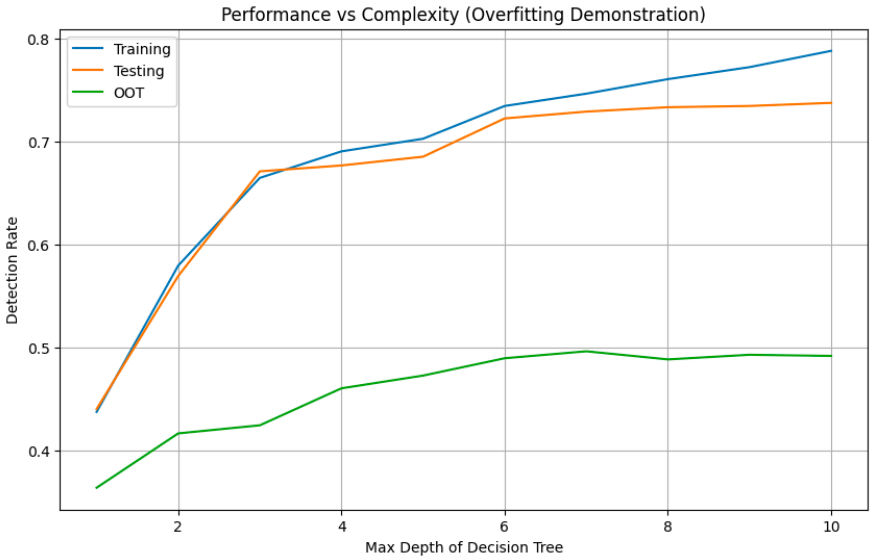
Given the importance of a model that not only performs well on known data but also generalizes effectively to new data, **the Random Forest (RF) model** would be the advisable choice. It shows the highest OOT performance, which is indicative of strong predictive power on unseen data. The consistency across different evaluations also suggests that it's a robust model for practical applications, balancing performance and stability well.

6.4 Addressing Overfitting

1. Decision Tree

```
model = DecisionTreeClassifier(criterion='gini', splitter='best', max_depth=max_depth,
min_samples_split=20, min_samples_leaf=5)
```

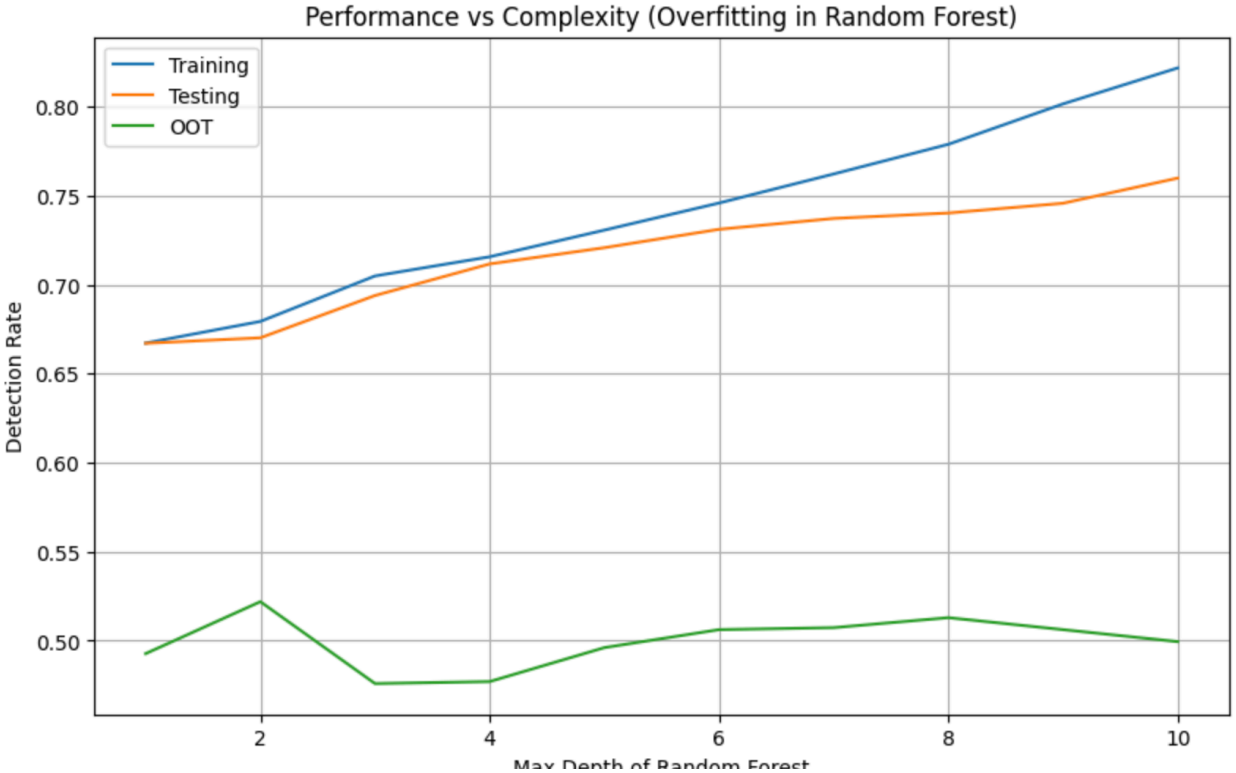
the max depths range is (1, 11)



2. Random Forest

```
model = RandomForestClassifier(n_estimators=60, criterion='gini', max_depth=max_depth,
min_samples_split=20, min_samples_leaf=10)
```

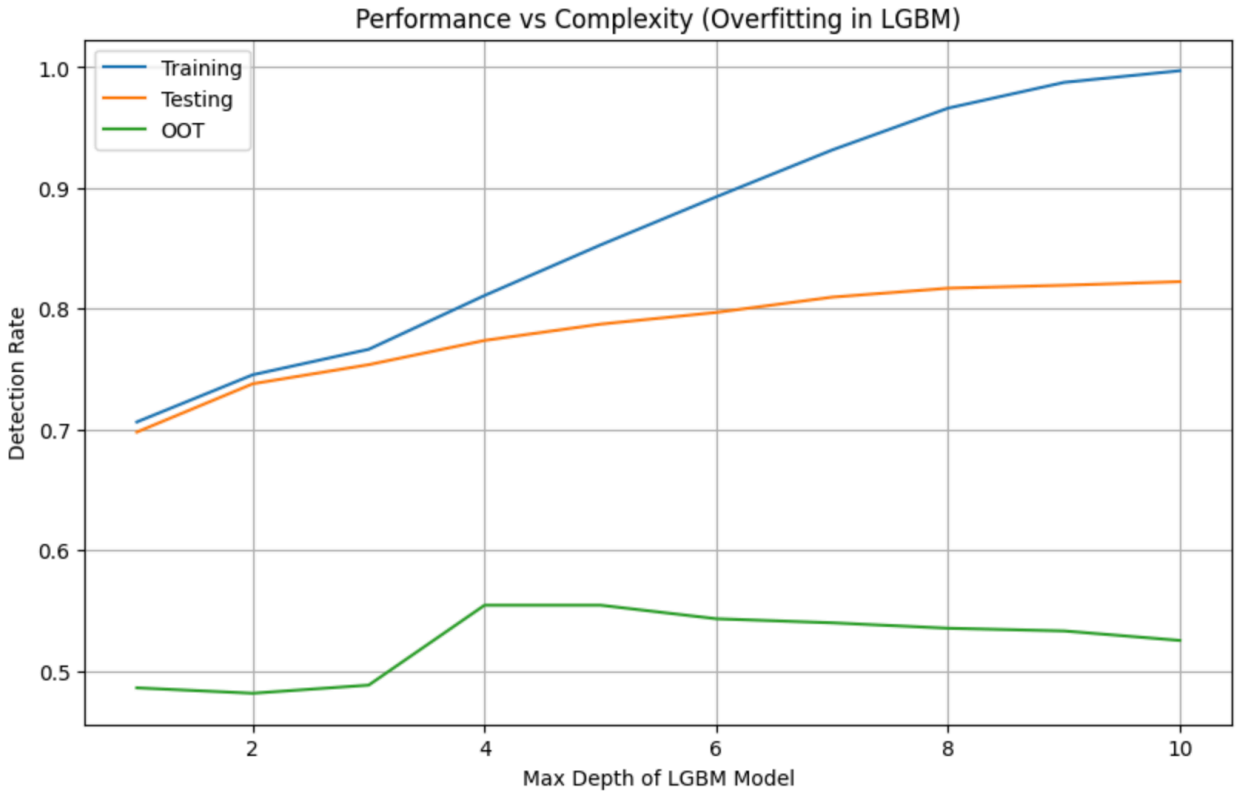
range= (1,11)



3. LGBM

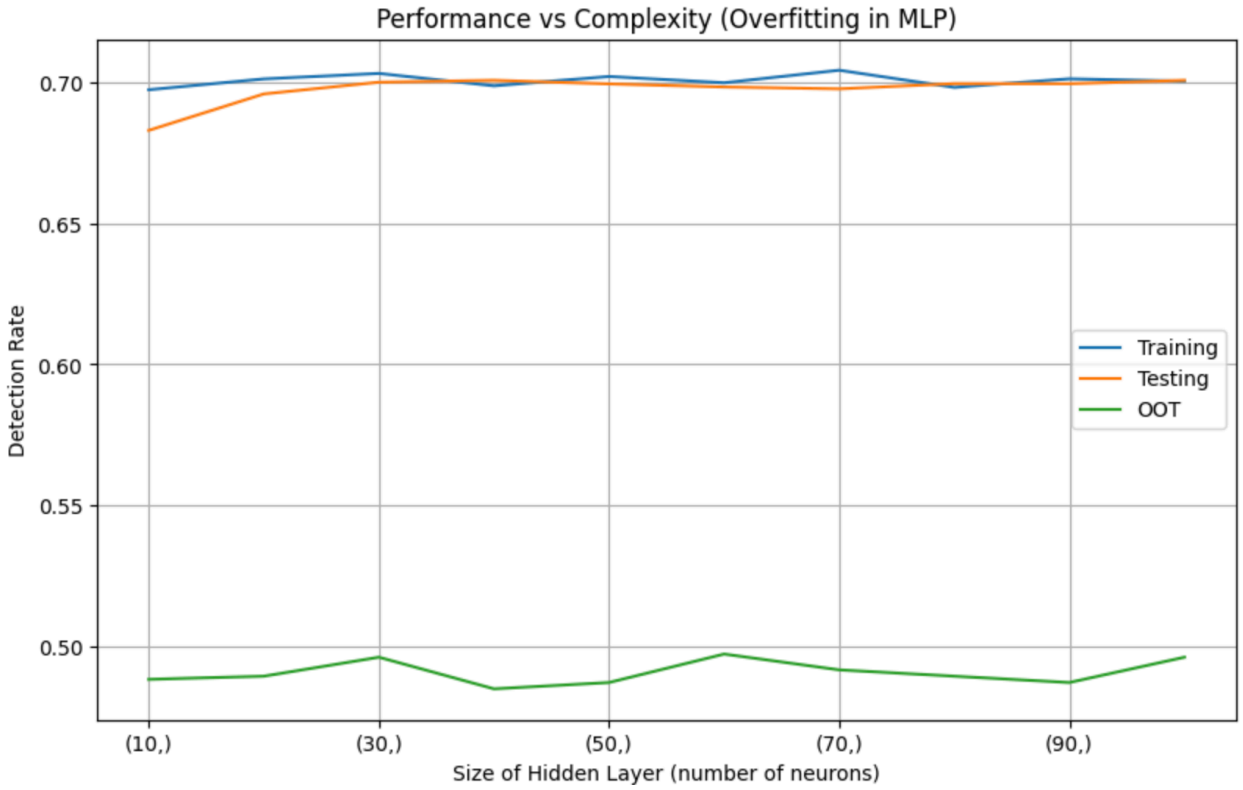
- `model = lgb.LGBMClassifier(num_leaves=2**max_depth, max_depth=max_depth, learning_rate=0.05, n_estimators=100)`

depth range= (1,11)



4. Neural Networks

```
model = MLPClassifier(
        hidden_layer_sizes=size,  # Varying size of the hidden layer
        activation='tanh',        # Activation function
        solver='sgd',             # Solver for weight optimization
        alpha=0.001,              # L2 regularization strength
        max_iter=300              # Maximum iterations
    )
Hidden_layer_sizes = [(10,), (20,), (30,), (40,), (50,), (60,), (70,), (80,), (90,), (100,)]  # Varying sizes of the hidden layer
```



7.0 Final model performance

7.1 Model: RandomForestClassifier

- **n_estimators=60**: Specifies the number of trees in the forest. The choice of 60 trees balances between adequate model complexity and computational efficiency, ensuring robust performance without excessive training time.
- **criterion='gini'**: Utilizes the Gini impurity as the function to measure the quality of a split. Gini impurity is a popular choice for classification tasks and helps in maximizing the homogeneity of nodes.
- **max_depth**: This parameter limits the maximum depth of the trees. While the specific depth isn't provided in your description, setting a max depth helps prevent the model from becoming overly complex and overfitting, which is common in highly branched trees.
- **min_samples_split=20**: This setting requires a node to have at least 20 samples before it can be split further. This helps in avoiding overfitting by ensuring that splits that create very specific rules to capture outliers are not allowed.
- **min_samples_leaf=10**: Ensures that a leaf node must contain at least 10 samples. This setting further aids in smoothing the model by avoiding overly granular classification rules, enhancing generalization.

7.2 Model Performance

Typically, the performance of the model is evaluated across three data subsets to ensure robustness and predictability:

- **Training (trn)**: Evaluates how well the model fits the data it was trained on, which provides insight into its learning capability.
- **Test (tst)**: Assesses the model's performance on a separate set of data not seen during training, offering a measure of generalization.
- **Out-of-time (oot)**: Tests the model on data collected in a different time period than the training and test sets, which is critical for understanding how the model will perform in future or under different conditions.

Train	#Records	# Goods	#Bads	Fraud Rate
	597	584	122	0.204355109

Bin Statistics						Cumulative Statistics						
Population Bin %	# Records	#Goods	#Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Cumulative Goods	% Bads (FDR)	KS	FPR
0	0	0	0	0.00	0	0	0	0	0	0	0	0
1	597	15	582	2.51	97.49	597	15	582	0.03	47.39	47.37	0.03
2	597	238	359	39.87	60.13	1194	253	941	0.43	76.63	76.20	0.27
3	597	448	149	75.04	24.96	1791	701	1090	1.20	88.76	87.56	0.64
4	596	500	96	83.89	16.11	2387	1201	1186	2.05	96.58	94.53	1.01
5	597	562	35	94.14	5.86	2984	1763	1221	3.02	99.43	96.41	1.44
6	597	590	7	98.83	1.17	3581	2353	1228		100.00	95.97	1.92
7	597	597	0	100	0	4178	2950	1228	5.05	100.00	94.95	2.40
8	597	597	0	100	0	4775	3547	1228	6.07	100.00	93.93	2.89
9	597	597	0	100	0	5372	4144	1228	7.09	100.00	92.91	3.37
10	596	596	0	100	0	5968	4740	1228	8.11	100.00	91.89	3.86
11	597	597	0	100	0	6565	5337	1228	9.13	100.00	90.87	4.35
12	597	597	0	100	0	7162	5934	1228	10.15	100.00	89.85	4.83
13	597	597	0	100	0	7759	6531	1228	11.17	100.00	88.83	5.32
14	597	597	0	100	0	8356	7128	1228	12.19	100.00	87.81	5.80
15	597	597	0	100	0	8953	7725	1228	13.22	100.00	86.78	6.29
16	596	596	0	100	0	9549	8321	1228	14.23	100.00	85.77	6.78
17	597	597	0	100	0	10146	8918	1228	15.26	100.00	84.74	7.26
18	597	597	0	100	0	10743	9515	1228	16.28	100.00	83.72	7.75
19	597	597	0	100	0	11340	10112	1228	17.30	100.00	82.70	8.23

Test	#Records	# Goods	#Bads	Fraud Rate
	597	250	52	0.087102178

Bin Statistics						Cumulative Statistics						
Population Bin %	# Records	#Goods	#Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Cumulative Goods	% Bads (FDR)	KS	FPR
0	0	0	0	0.00	0.00	0	0	0	0.00	0.00	0.00	0.00
1	256	28	228	10.94	89.06	256	28	228	0.11	43.68	43.57	0.12
2	256	113	143	44.14	55.86	512	141	371	0.56	71.07	70.51	0.38
3	255	215	40	84.31	15.69	767	356	411	1.42	78.74	77.31	0.87
4	256	227	29	88.67	11.33	1023	583	440	2.33	84.29	81.96	1.33
5	256	242	14	94.53	5.47	1279	825	454	3.29	86.97	83.68	1.82
6	256	246	10	96.09	3.91	1535	1071	464	4.27	88.89	84.61	2.31
7	256	252	4	98.44	1.56	1791	1323	468	5.28	89.66	84.38	2.83
8	255	251	4	98.43	1.57	2046	1574	472	6.28	90.42	84.14	3.33
9	256	249	7	97.27	2.73	2302	1823	479	7.28	91.76	84.49	3.81
10	256	252	4	98.44	1.56	2558	2075	483	8.28	92.53	84.25	4.30
11	256	255	1	99.61	0.39	2814	2330	484	9.30	92.72	83.42	4.81
12	256	255	1	99.61	0.39	3070	2585	485	10.32	92.91	82.60	5.33
13	255	253	2	99.22	0.78	3325	2838	487	11.33	93.30	81.97	5.83
14	256	254	2	99.22	0.78	3581	3092	489	12.34	93.68	81.34	6.32
15	256	255	1	99.61	0.39	3837	3347	490	13.36	93.87	80.51	6.83
16	256	254	2	99.22	0.78	4093	3601	492	14.37	94.25	79.88	7.32
17	256	255	1	99.61	0.39	4349	3856	493	15.39	94.44	79.06	7.82
18	255	253	2	99.22	0.78	4604	4109	495	16.40	94.83	78.43	8.30
19	256	254	2	99.22	0.78	4860	4363	497	17.41	95.21	77.80	8.78
20	256	254	2	99.22	0.78	5116	4617	499	18.43	95.59	77.17	9.25

OOT	#Records	# Goods	#Bads	Fraud Rate
	112	119	29	0.258928571

Bin Statistics						Cumulative Statistics						
Population Bin %	# Records	#Goods	#Bads	% Goods	% Bads	Total # Records	Cumulative Goods	Cumulative Bads	% Cumulative Goods	% Bads (FDR)	KS	FPR
0	0	0	0	0.00	0.00	0	0	0	0.00	0.00	0.00	0.00
1	122	24	98	19.67	80.33	122	24	98	0.20	33.00	32.80	0.24
2	123	86	37	69.92	30.08	245	110	135	0.92	45.45	44.53	0.81
3	122	89	33	72.95	27.05	367	199	168	1.67	56.57	54.90	1.18
4	122	102	20	83.61	16.39	489	301	188	2.52	63.30	60.78	1.60
5	123	113	10	91.87	8.13	612	414	198	3.47	66.67	63.20	2.09
6	122	113	9	92.62	7.38	734	527	207	4.42	69.70	65.28	2.55
7	122	116	6	95.08	4.92	856	643	213	5.39	71.72	66.33	3.02
8	123	118	5	95.93	4.07	979	761	218	6.38	73.40	67.02	3.49
9	122	114	8	93.44	6.56	1101	875	226	7.33	76.09	68.76	3.87
10	122	117	5	95.90	4.10	1223	992	231	8.31	77.78	69.47	4.29
11	123	116	7	94.31	5.69	1346	1108	238	9.28	80.13	70.85	4.66
12	122	116	6	95.08	4.92	1468	1224	244	10.26	82.15	71.90	5.02
13	122	116	6	95.08	4.92	1590	1340	250	11.23	84.18	72.95	5.36
14	122	118	4	96.72	3.28	1712	1458	254	12.22	85.52	73.31	5.74
15	123	121	2	98.37	1.63	1835	1579	256	13.23	86.20	72.97	6.17
16	122	120	2	98.36	1.64	1957	1699	258	14.24	86.87	72.63	6.59
17	122	121	1	99.18	0.82	2079	1820	259	15.25	87.21	71.96	7.03
18	123	123	0	100.00	0.00	2202	1943	259	16.28	87.21	70.93	7.50
19	122	122	0	100.00	0.00	2324	2065	259	17.30	87.21	69.90	7.97

8.0 Financial curves and recommended cutoff

To determine the optimal cutoff for the fraud detection model and ensure maximum financial savings, we analyze the impact of different cutoff points on three financial aspects: the Fraud Dollars Caught, Lost Revenue due to False Positives, and Overall Savings.

Financial Impact Curves and Cutoff Recommendation

Assumptions for Calculations:

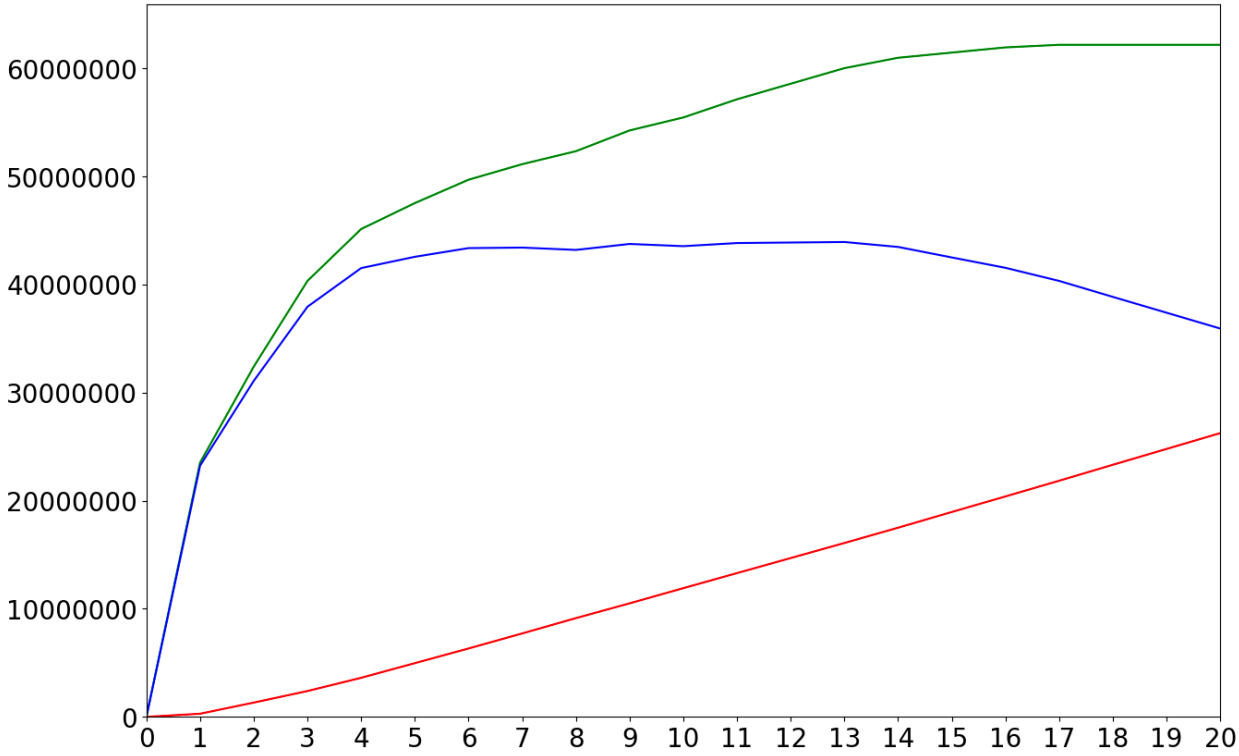
- Fraud Gain: $400 for every correctly identified fraud transaction (Green Curve).
- Loss per False Positive: $20 for every legitimate transaction incorrectly flagged as fraud (Red Curve).
- Sample Size: Data is based on a sample of 100,000 records from a portfolio of 10 million transactions per year.

Financial Curve Calculations:

1. Fraud Dollars Caught (Green Curve): Represents the total value recovered from preventing frauds by correctly identifying fraudulent transactions.
2. Lost Revenue (Red Curve): Represents the total cost incurred from legitimate transactions wrongly classified as fraud.
3. Overall Savings (Blue Curve): Calculated as the difference between Fraud Dollars Caught and Lost Revenue. It provides a net view of the financial impact of using different cutoff points.

Adjustment for Annual Projection:

To estimate the annual savings, multiply the out-of-time (OOT) savings by the factor (12/2)×(10,000,000/100,000)(12/2)×(10,000,000/100,000). This adjustment accounts for the biannual period and scales up the sample to the entire portfolio.



Plot Interpretation and Cutoff Setting:

- **Objective**: The goal is to set the cutoff at a point where Overall Savings (Blue Curve) are maximized, while minimizing the number of legitimate transactions denied (minimizing False Positives).
- **Recommended Cutoff**: From the plot, the recommended cutoff is the point where the Overall Savings curve (Blue) begins to plateau or just before it starts to decrease. This point ensures that we are capturing the maximum possible savings without incurring significant losses from False Positives.

Recommendation:

Based on the curves, a cutoff point where the blue line peaks or just before it starts to dip should be chosen. This point will provide a balance between capturing most of the fraud dollars and keeping the loss from false positives low.
By following this methodology, you can make an informed decision on setting the score cutoff that optimizes financial outcomes for the fraud detection system, taking into account the specific cost-benefit dynamics of your operation.

9.0 Summary and Conclusions

In this project, we aimed to enhance fraud detection for a dataset of approximately 100,000 credit card transactions from a U.S. government agency in Tennessee. The initiative began with a thorough analysis of the data, which included 10 distinct transactional fields. Key steps in data preparation involved rigorous cleaning processes such as removing empty columns, converting dates to a consistent format, and filtering out outliers. A critical component was addressing missing data in essential fields like 'Merchnum', 'Merch state', and 'Merch zip' through advanced imputation methods.

To improve the model's predictive accuracy, extensive feature engineering was performed. We developed a wide range of variables, including day-since-last-transaction counters, frequency measures, and transaction amount metrics, tailored to capture the unique behaviors indicative of fraud. The selection of predictive features was conducted using forward and backward stepwise methods across various classifiers including LightGBM, Random Forest, and CatBoost. The RandomForestClassifier was ultimately chosen for its robust performance, configured with key hyperparameters like 60 trees, and settings to control tree depth and node splitting to prevent overfitting.

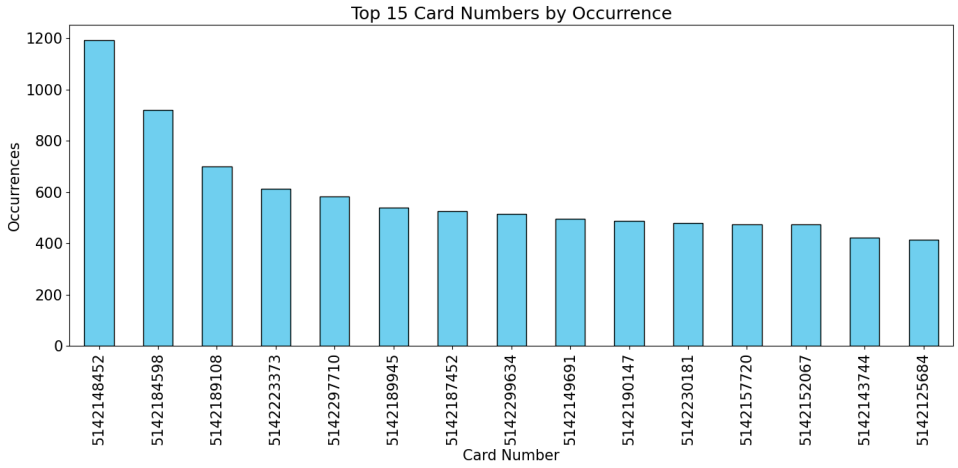
The model's effectiveness was quantitatively assessed by the False Discovery Rate (FDR) at 3% in an out-of-time (OOT) validation, focusing on its ability to accurately identify fraudulent transactions. The financial impact was significant; the projected annual savings from implementing this model, assuming an operational scale based on the agency’s full transaction volume, were estimated to be around $44.66 million. This projection was based on the model’s ability to reduce fraud while minimizing the costs associated with false positives.

Future recommendations for this project include integrating more diverse data sources to enrich the transaction context, continuous model parameter tuning to adapt to new fraud patterns, and exploring more advanced algorithms like deep learning for potentially enhanced detection capabilities.

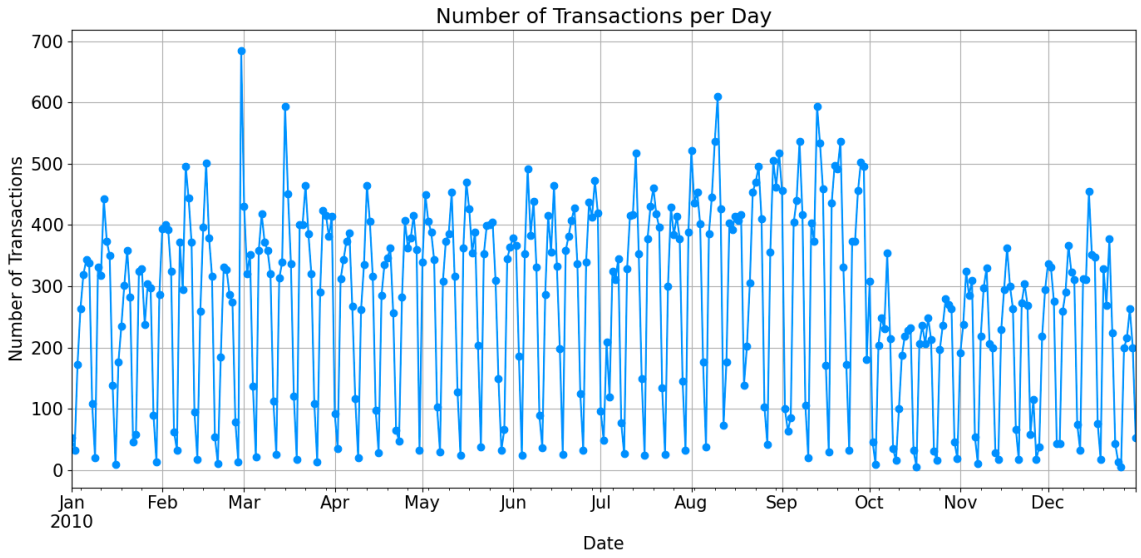
10.0 Appendix

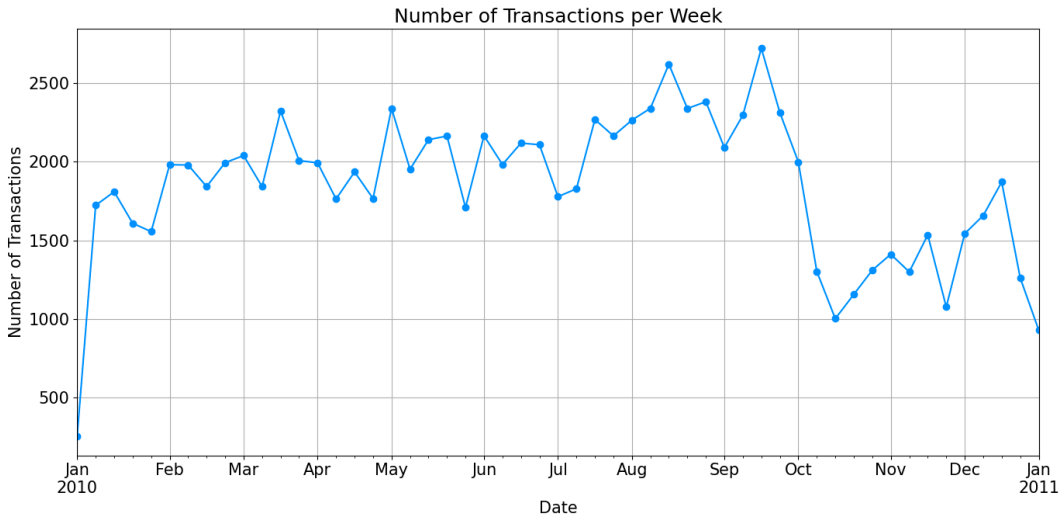
1) **Field Name: Recnum**
   Description: Ordinal unique positive integer for each application record, from 1 to 97,852.

2) **Field Name: Cardnum**
   Description: The credit card number associated with each transaction. It identifies the company card that was used for the transaction. The distribution shows the top 15 card numbers by occurrence. The most common card number is 5142148452, with a total count of 1192.



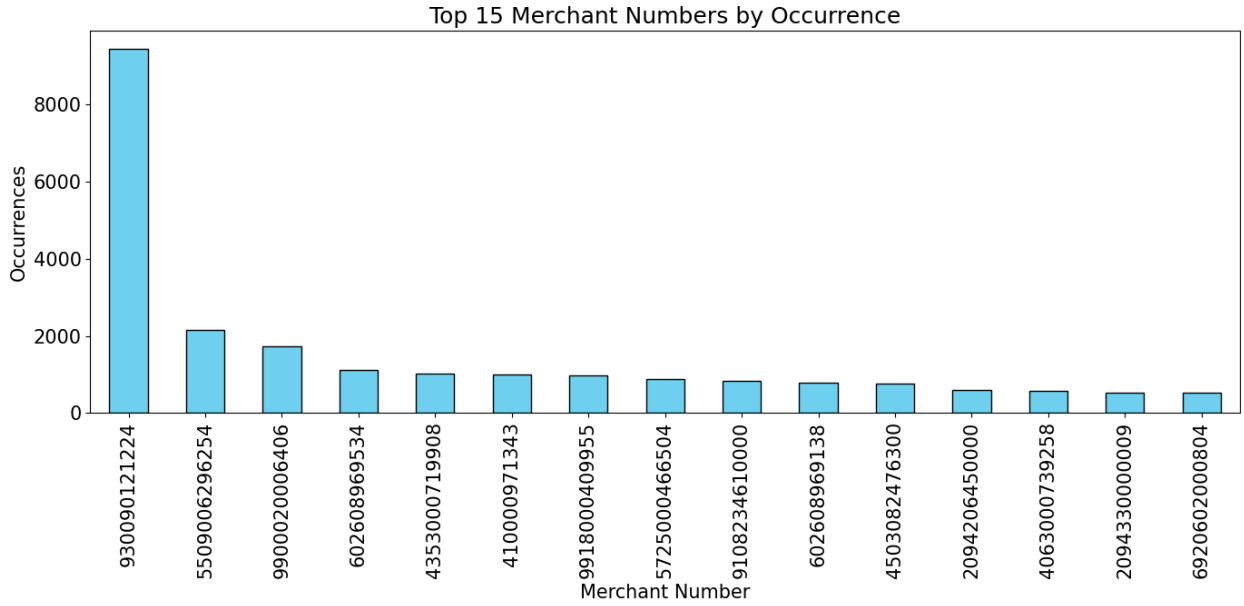
3) **Field Name: Date**
   Description: The date on which the transaction was executed. It is formatted in MM/DD/YYYY. The first graph shows the number of transactions per day and the second graph shows the distribution of transactions per week.





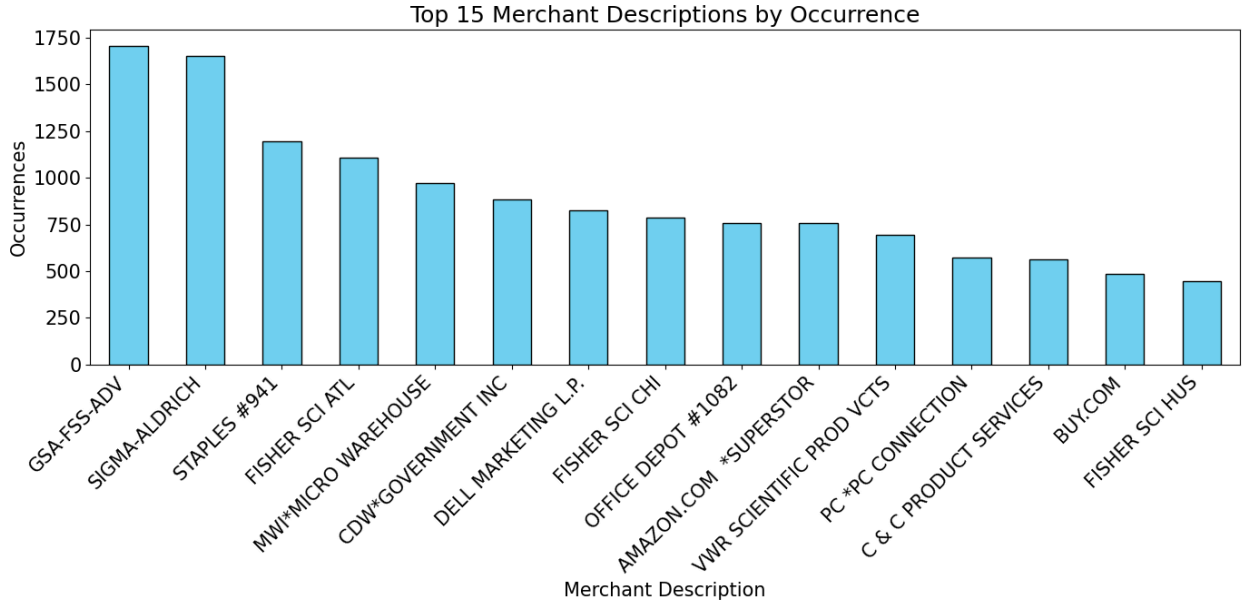
4) Field Name: Merchnum

Description: Merchant's unique identification number. This number categorizes transactions by the merchant. The distribution shows the top 15 card Merchant numbers by occurrence. The most common being 930090121224, with a total count of 9419.



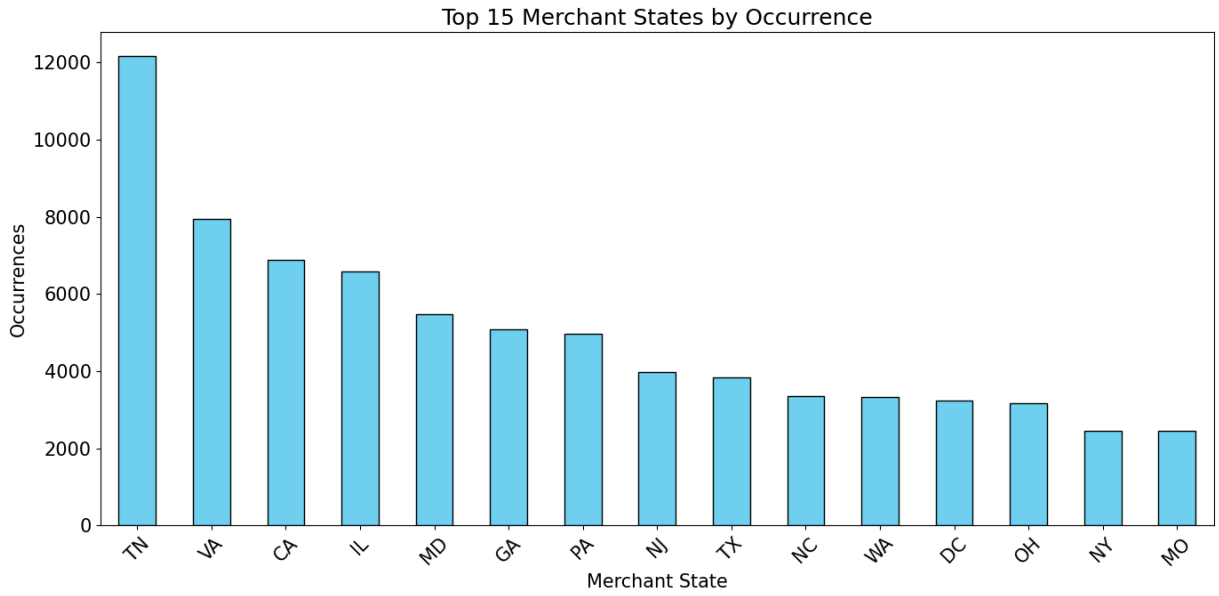
5) Field Name: Merch description

Description: A text description of the merchant where the transaction took place. The distribution shows the top 15 card Merch descriptions by occurrence. The most common being GSA-FSS-ADV, with a total count of 1706.



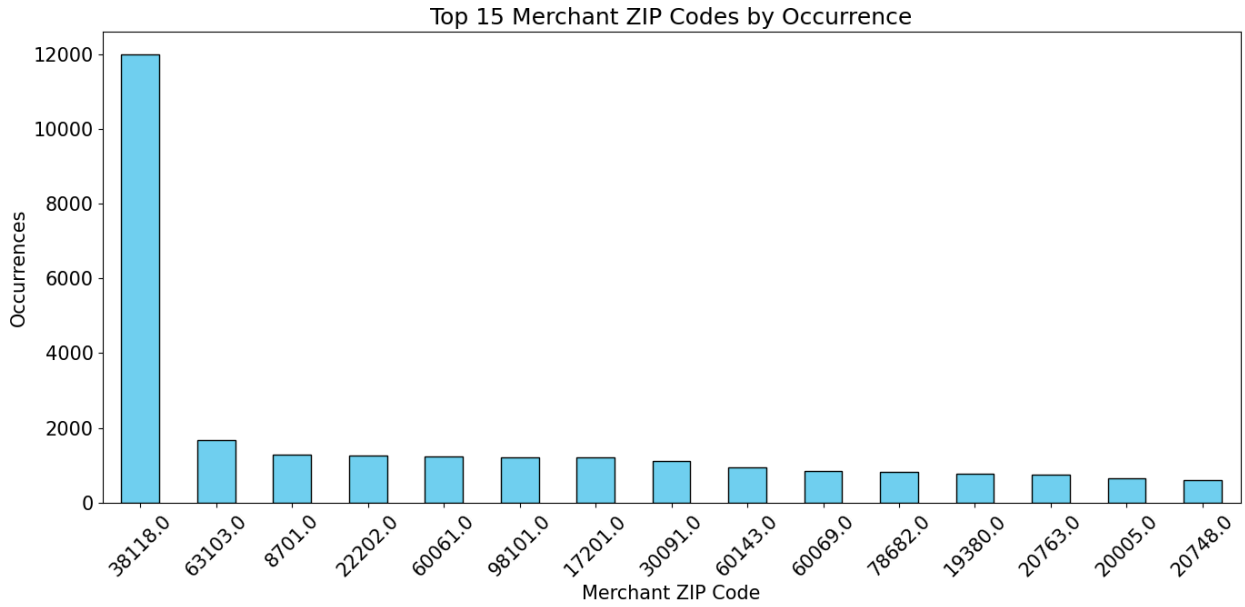
6) Field Name: Merch state

Description: The state within the United States where the merchant is located. The distribution shows the top 15 card Merchant states by occurrence. The most common being TN, with a total count of 12169.



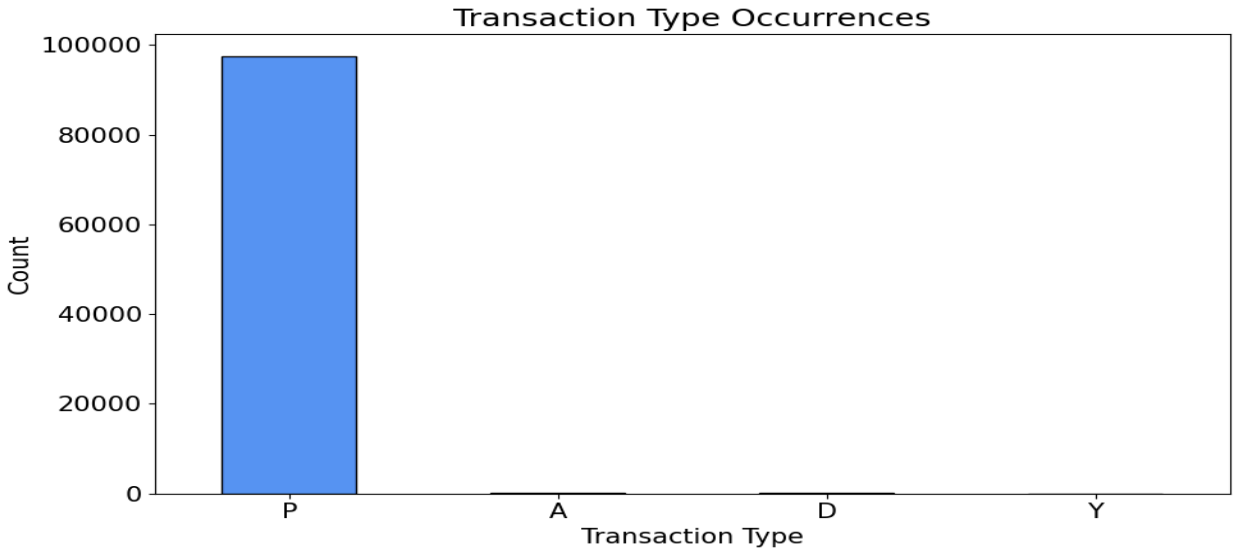
7) **Field Name: Merch zip**

Description: The ZIP code associated with the merchant's location. The distribution shows the top 15 card Merchant zip codes by occurrence. The most common being 38118, with a total count of 11998.



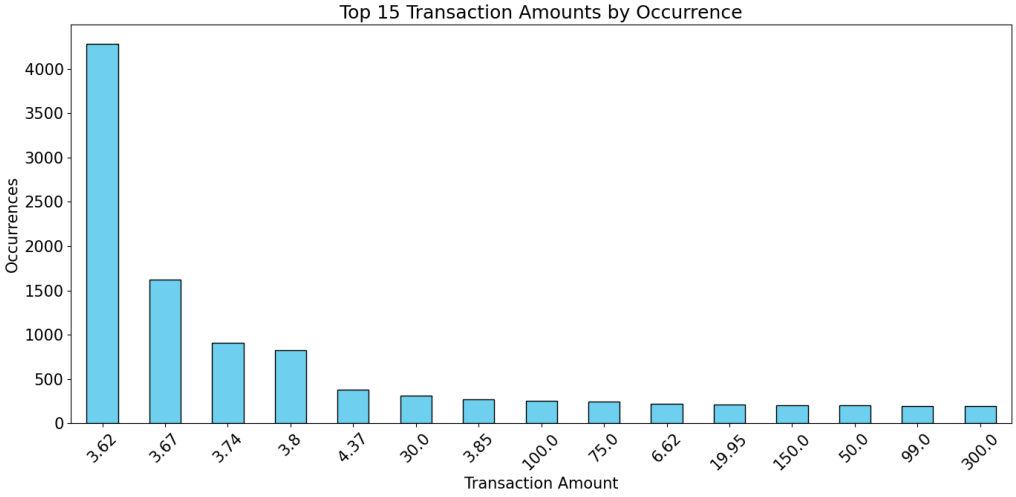
8) **Field Name: Transtype**

Description: The type of transaction that took place, such as purchase, refund, etc. The most common being P (purchases), with a total count of 97497.



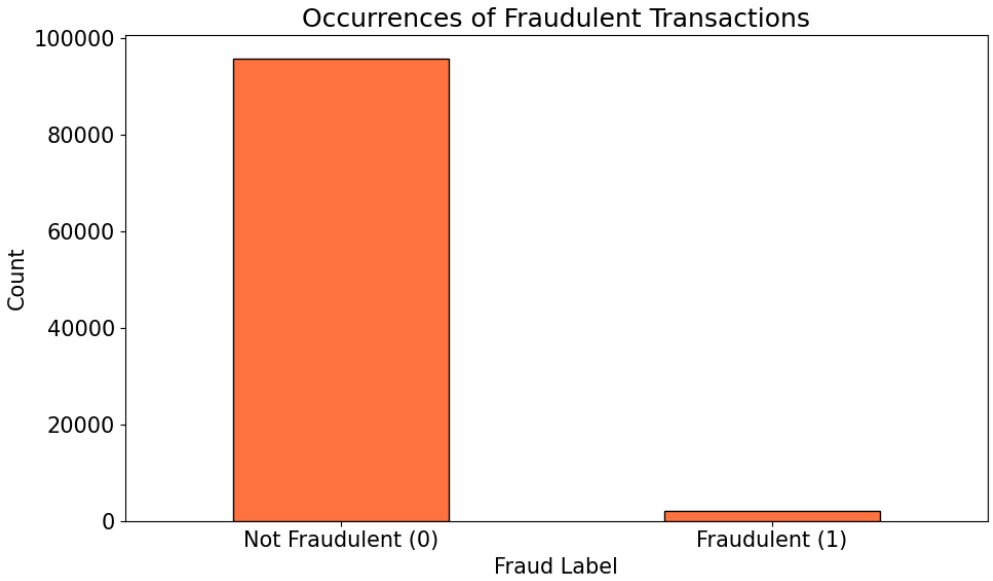
9) Field Name: Amount

Description: The monetary value of the transaction. The most common being 3.62 with a total count of 4283.



10) Field Name: fraud

Description: Fraud identification label. Fraud = 0 (Not fraudulent), Fraud =1 (Fraud identified). The total count of fraud_label = 0 is 95,805. The total count of fraud_label = 1 is 2047.


Occurrences of Fraudulent Transactions
100000
80000
60000
40000
20000
0
Count
Not Fraudulent (0)
Fraudulent (1)
Fraud Label