Predicting Fraudulent Transaction Using Binary Classification Algorithms

Giorgos Tzimas

May 2023

Abstract

Using customers' transactional dataset, we built two supervised learning machine learning algorithms, Logistic Regression and Random Forest, to classify each instance as either fraudulent or non-fraudulent transactions. After comparing the performance of both models on important metrics such as recall and False Positive Rate, we picked Random Forest as the final binary classification model to be used.

1 Dataset Description

1.1 Background

The panel dataset contains commercial customers' financial information and days past due indicator from 2000 to 2020 provided by the Capital One Data Scientist Recruiting process.

Dataset Link: **https://github.com/CapitalOneRecruiting/DS**

1.2 Observations and Attributes

The dataset comprises a comprehensive collection of **786,363** entries, encompassing **29** distinct features. These features contain a range of information pertaining to both the customer and the transactional context. Customer-related attributes include identifiers, credit card balance, credit limit, and more. Similarly, business-related attributes encompass details like business name, category, and point-of-sale information.

	type
accountNumber	int64
customerId	int64
creditLimit	int64
availableMoney	float64
transactionDateTime	object
transactionAmount	float64
merchantName	object
acqCountry	object
merchantCountryCode	object
posEntryMode	object
posConditionCode	object
merchantCategoryCode	object
currentExpDate	object
accountOpenDate	object
dateOfLastAddressChange	object

	type
cardCVV	int64
enteredCVV	int64
cardLast4Digits	int64
transactionType	object
echoBuffer	object
currentBalance	float64
merchantCity	object
merchantState	object
merchantZip	object
cardPresent	bool
posOnPremises	object
recurringAuthInd	object
expirationDateKeyInMatch	bool
isFraud	bool

Table 1: Dataset attributes and their types

1.3 Data Types

Out of the 29 features in the dataset:

- 17 are of type "object" or text
- 6 are of type "integer"
- 3 are of type "float"
- 3 are of type "boolean"

2 Data Cleaning

Some of our features are missing values either due to not being entered or that feature containing sensitive information. Features **recurringAuthInd**, **posOnPremises**, **merchantZip**, **merchantState**, **merchantCity** and **echoBuffer** will be removed from the dataset. The rest of the features with missing values will be imputed using a "most frequent" approach due to all of them being categorical in nature.

	total_missing
recurringAuthInd	786363
posOnPremises	786363
merchantZip	786363
merchantState	786363
merchantCity	786363
echoBuffer	786363
acqCountry	4562
posEntryMode	4054
merchantCountryCode	724
transactionType	698
posConditionCode	409

Table 2: Features with missing values

3 Feature Extraction

There are three datetime features in the dataset: **transactionDateTime**, **accountOpenDate**, **dateOfLastAddressChange** and **currentExpDate**. We can extract transactional information from these features to include in the dataset.

From **transactionDateTime**, we can get the month, day and hour that the transaction occurred.

	transactionDateTime	trans_month	trans_day_name	trans_hour
0	2016-08-13 14:27:32	8	Saturday	14
1	2016-10-11 05:05:54	10	Tuesday	05
2	2016-11-08 09:18:39	11	Tuesday	09
3	2016-12-10 02:14:50	12	Saturday	02
4	2016-03-24 21:04:46	3	Thursday	21

Table 3: Features extracted from transactionDateTime

Additionally, we can get the month and year from the expiration date.

	currentExpDate	exp_month	exp_year
0	2023-06-01 00:00:00	6	2023
1	2024-02-01 00:00:00	2	2024
2	2025-08-01 00:00:00	8	2025
3	2025-08-01 00:00:00	8	2025
4	2029-10-01 00:00:00	10	2029

Table 4: Features extracted from currentExpDate

We can also extract date differences in days from each datetime feature, such as how long the account had been active, how many days since the last address change and how close to the expiration date the account was at the time of the transaction.

	transactionDateTime	dateOfLastAddressChange	accountOpenDate	trans_day_addr_change_diff	trans_day_open_date_diff	exp_trans_day_diff
0	2016-08-13 14:27:32	2015-03-14 00:00:00	2015-03-14 00:00:00	518	518	2482
1	2016-10-11 05:05:54	2015-03-14 00:00:00	2015-03-14 00:00:00	577	577	2668
2	2016-11-08 09:18:39	2015-03-14 00:00:00	2015-03-14 00:00:00	605	605	3187
3	2016-12-10 02:14:50	2015-03-14 00:00:00	2015-03-14 00:00:00	637	637	3155
4	2016-03-24 21:04:46	2015-08-06 00:00:00	2015-08-06 00:00:00	231	231	4938

Table 5: Date differences extracted

Another feature which might be useful is whether the credit card's CVV code matched the entered CVV at the time of the transaction.

	cardCVV	enteredCVV	matchingCVV
0	414	414	True
1	486	486	True
2	486	486	True
3	486	486	True
4	885	885	True

Table 6: Caption

The last feature to be extracted is the brand name from the merchant feature allowing us to view which store brands have the largest fraudulent transaction counts.

	merchantName	brandName
0	Uber	Uber
1	AMC 191138	AMC
2	Play Store	Play Store
3	Play Store	Play Store
4	Tim Hortons 947751	Tim Hortons
5	In-N-Out 422833	In-N-Out
6	Krispy Kreme 685312	Krispy Kreme
7	Shake Shack 968081	Shake Shack
8	Burger King 486122	Burger King
9	Five Guys 510989	Five Guys

Table 7: Brand name extracted from merchant feature

4 Exploratory Data Analysis

In this section, we will visually explore our dataset to gain some insight into the relationships between different features.

4.1 Total Counts by Fraud Status

Approximately 98% of the records are non-fraudulent. This is a very large dataset imbalance which might heavily skew the results of the model. This imbalance will be addressed with an under-sampling method during the model creation stage.

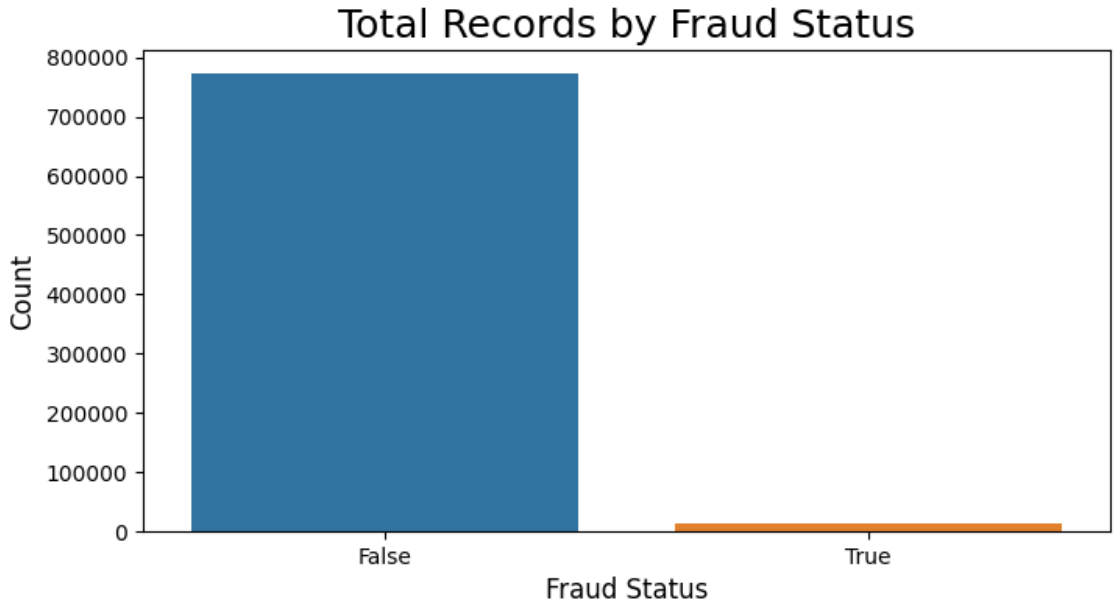


Figure 1: Caption

4.2 Numeric Feature Distribution

All of the numerical features of the dataset have a right-skewed distribution. Although some of these extreme values could be outliers, it is possible that they are valid due to fraudulent transactions differing from non-fraudulent ones on a bigger scale.

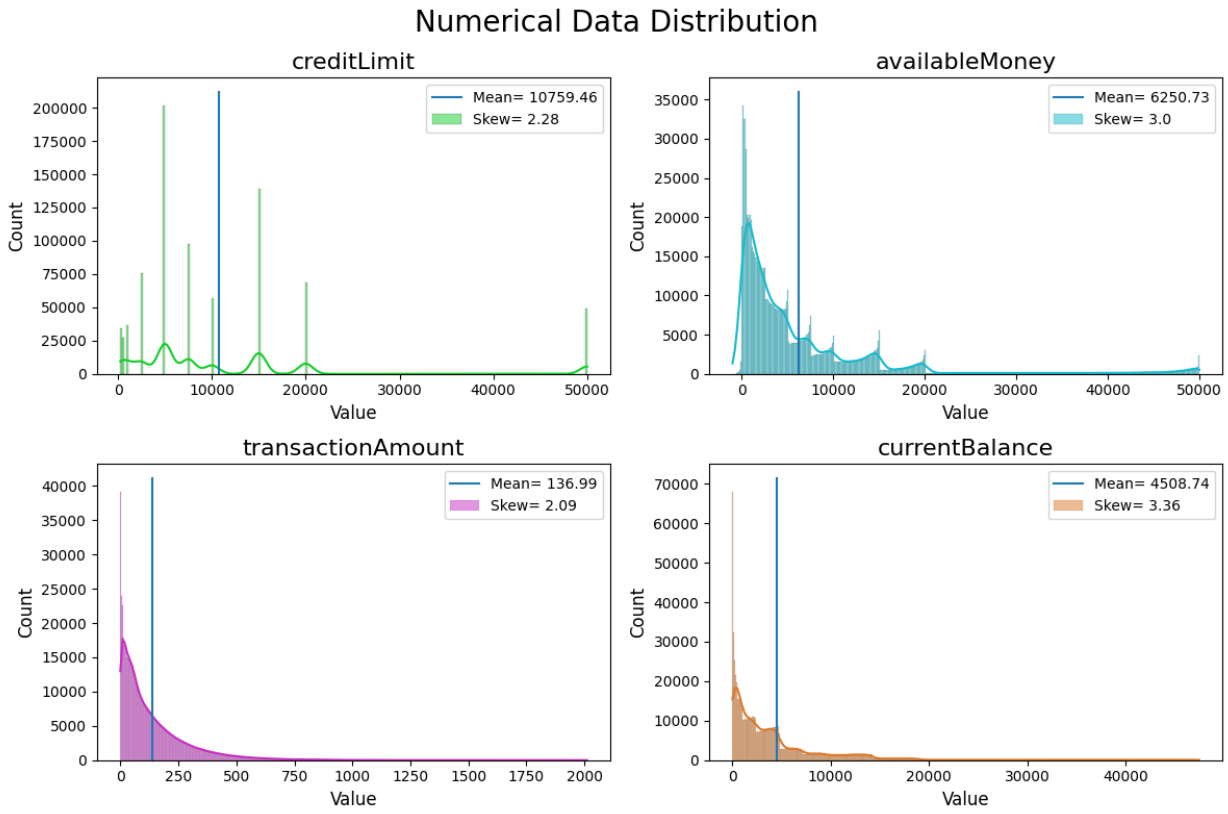


Figure 2: Distributions and mean values of numerical features

4.3 Numeric Feature Boxplot by Fraud Status

The biggest difference in numeric feature distributions when grouped by fraud status is visible for transaction amount. Median transaction amount is larger for fraudulent transactions compared to non-fraudulent. The interquartile range is also larger for fraudulent transactions.

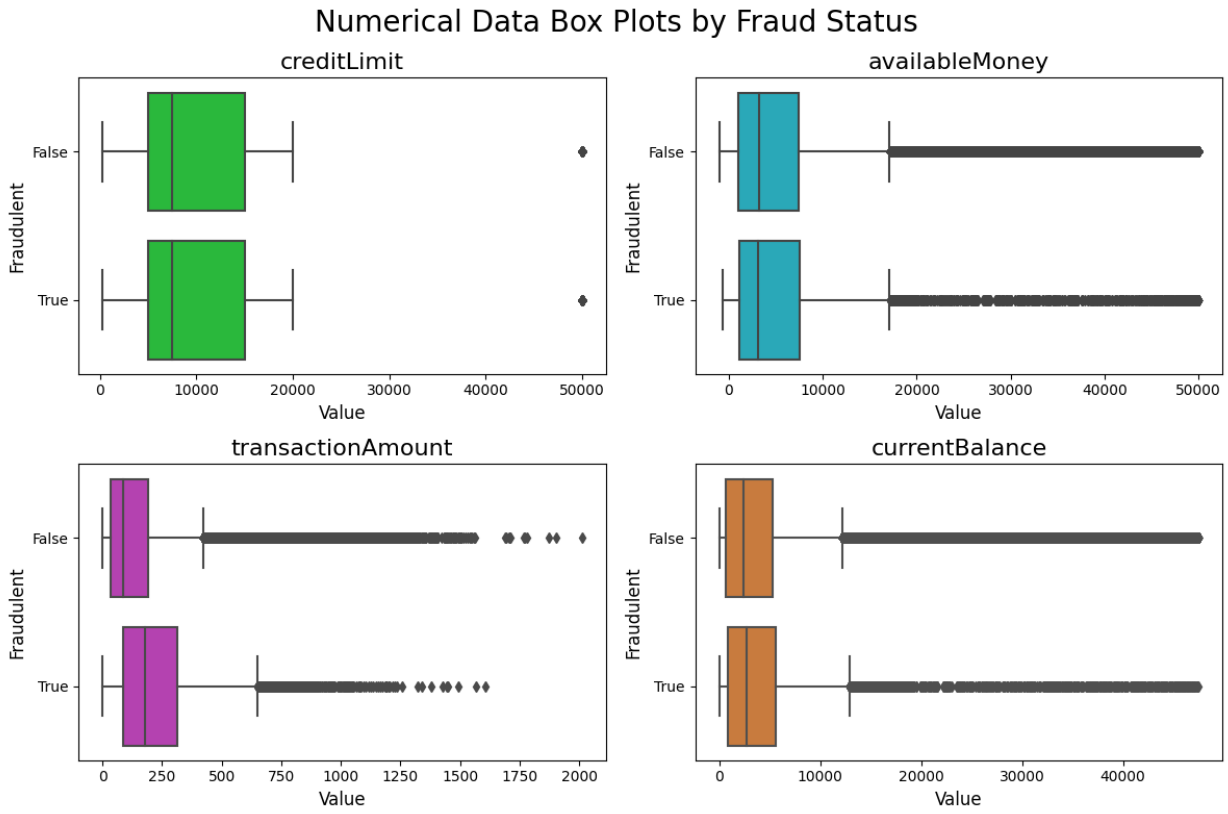


Figure 3: Numeric feature boxplots by fraud status

4.4 Transaction Amount by Fraud Status

The median transaction amount for fraudulent transactions is approximately $176. For non-fraudulent transactions it is approximately $86. Fraudulent transactions tend to cost almost double the amount of non-fraudulent ones in dollars.

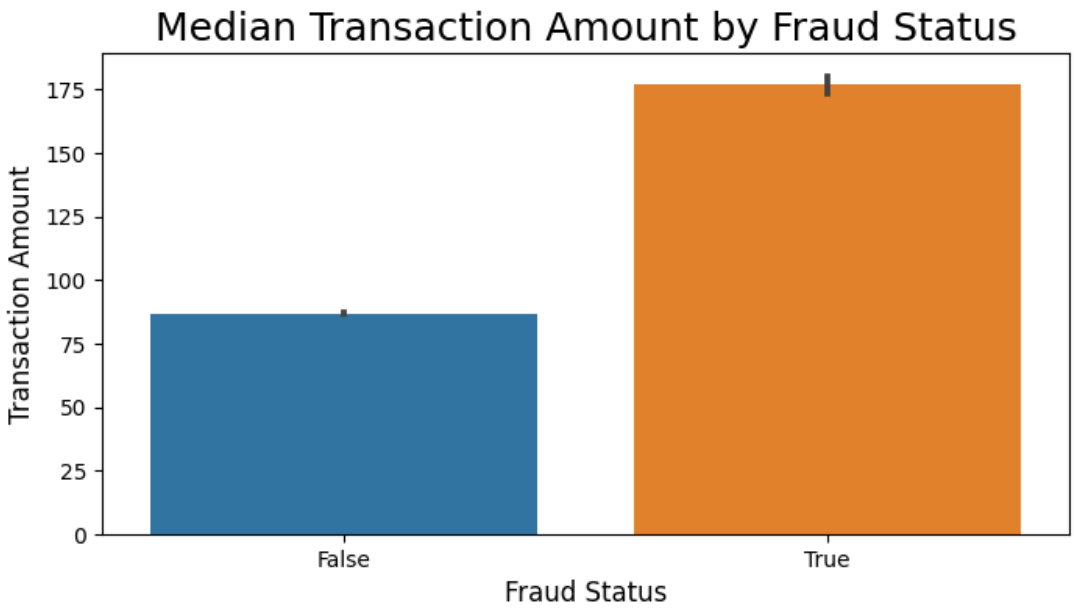
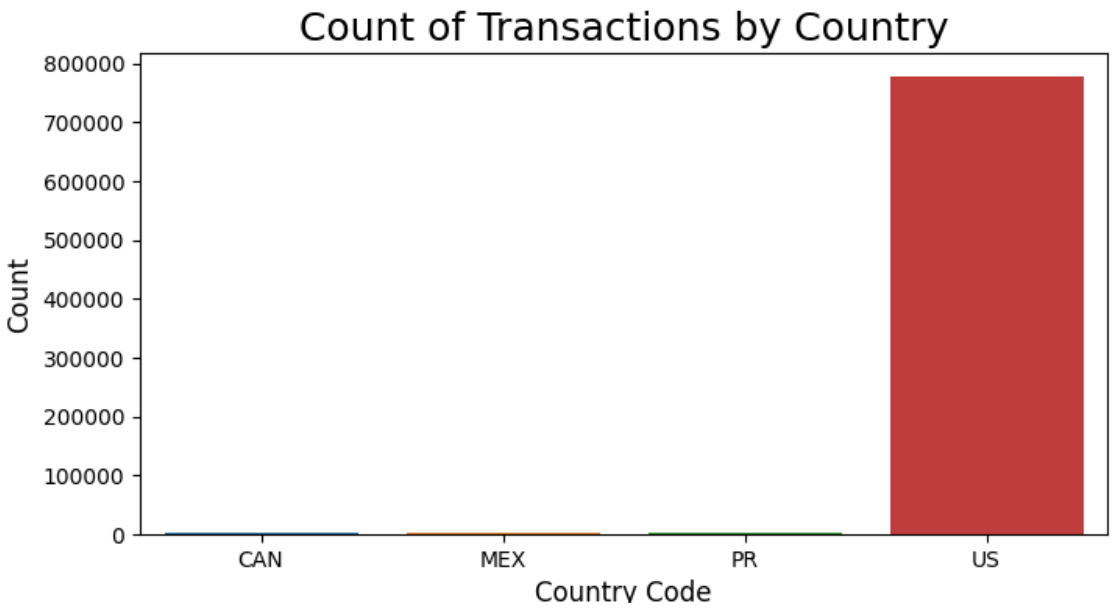


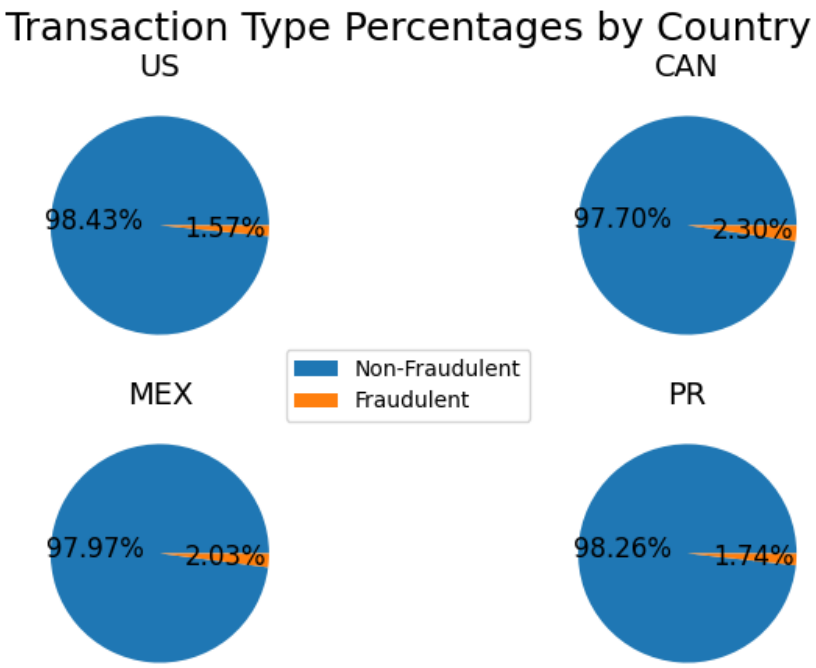
Figure 4: Median transaction amount by fraud status

4.5 Transactions by Country

Approximately 99% of the transactions in the dataset are made inside the United States. When broken down into percentages by fraud status and by country, there does not seem to be any significant difference in proportions.



(a) Transaction count by country



(b) Transaction type percentages by country

Figure 5: Transaction counts and proportions by status

4.6 Fraudulent Transaction Amount by Country

Canada is the country with the largest median fraudulent transaction amount ($229), followed by Puerto Rico ($198), the United States ($176) and Mexico ($162).

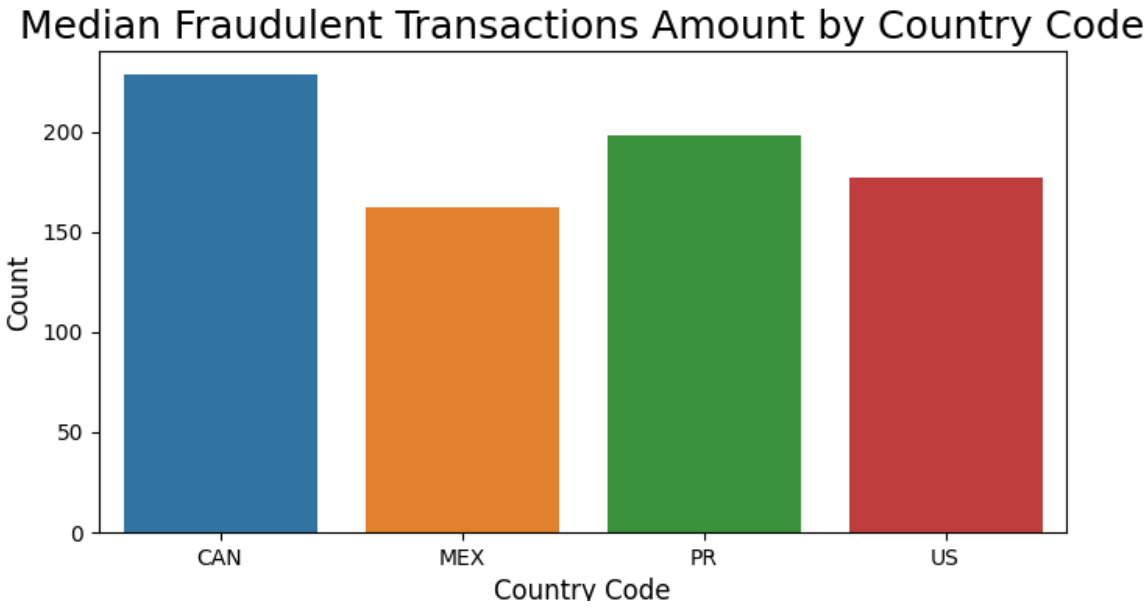
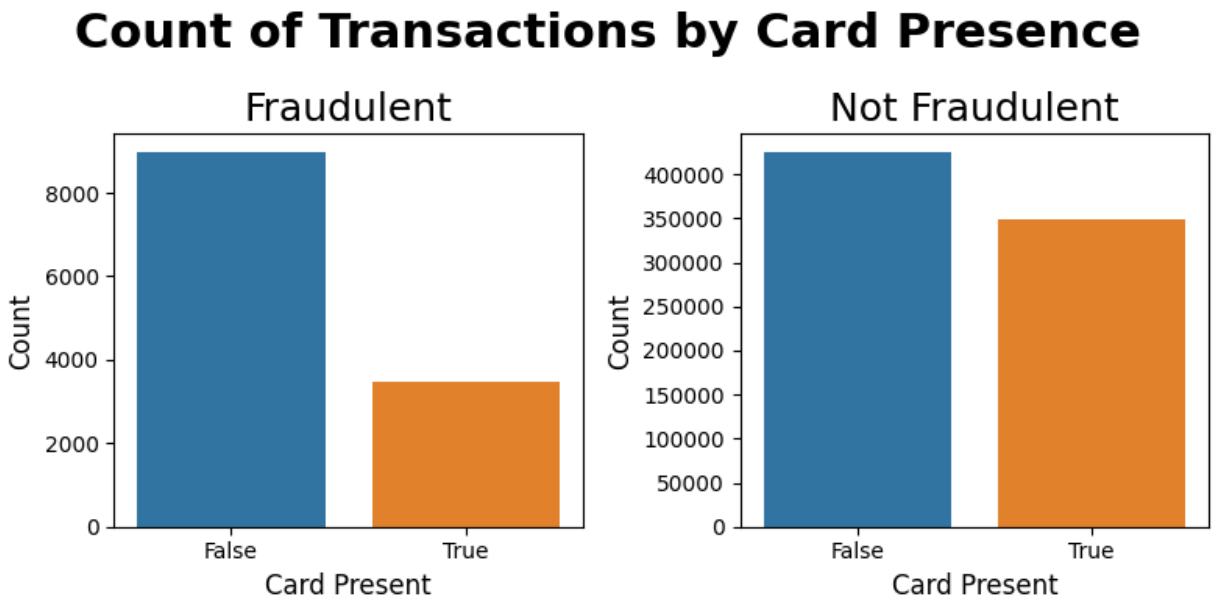


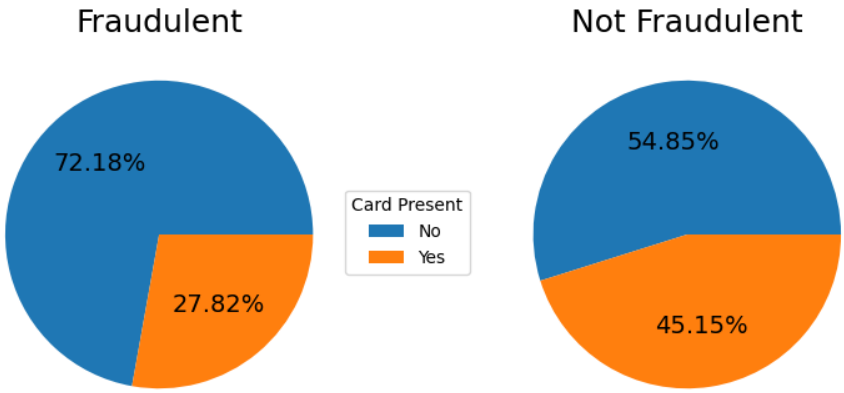
Figure 6: Median fraudulent transaction amount by country code

4.7 Transactions by Card Presence

There is a great difference in count of transactions by card presence when separated by fraud status. Fraudulent transactions tend to not have the credit card present at the time, presumably due to most of them occurring online where the credit card information can be inputted manually.



(a) Transaction count by card presence and fraud status



(b) Card presence percentages by fraud status

Figure 7: Transaction counts and proportions by status

4.8 Top 20 Brands by Transaction Count

The brand with the most transactions in the dataset is **AMC** with 37,942 total transactions.

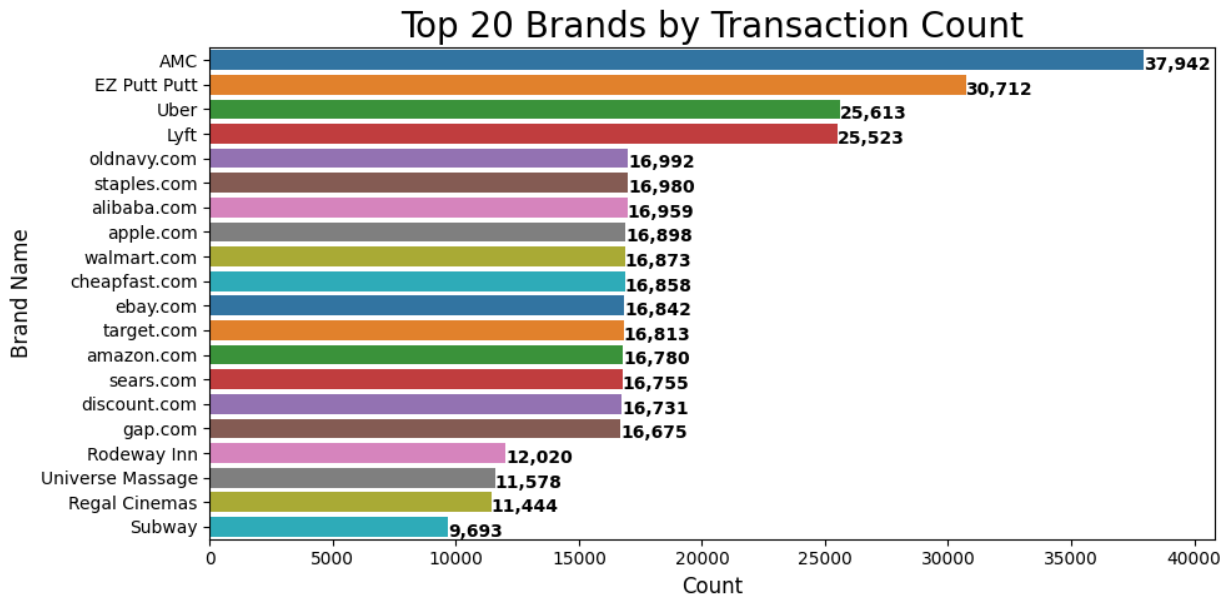


Figure 8: Top 20 brand by number of total transactions

4.9 Top 20 Brands by Fraudulent Transaction Count

The brand with the most fraudulent transactions in the dataset is **Lyft** with 760 total transactions.

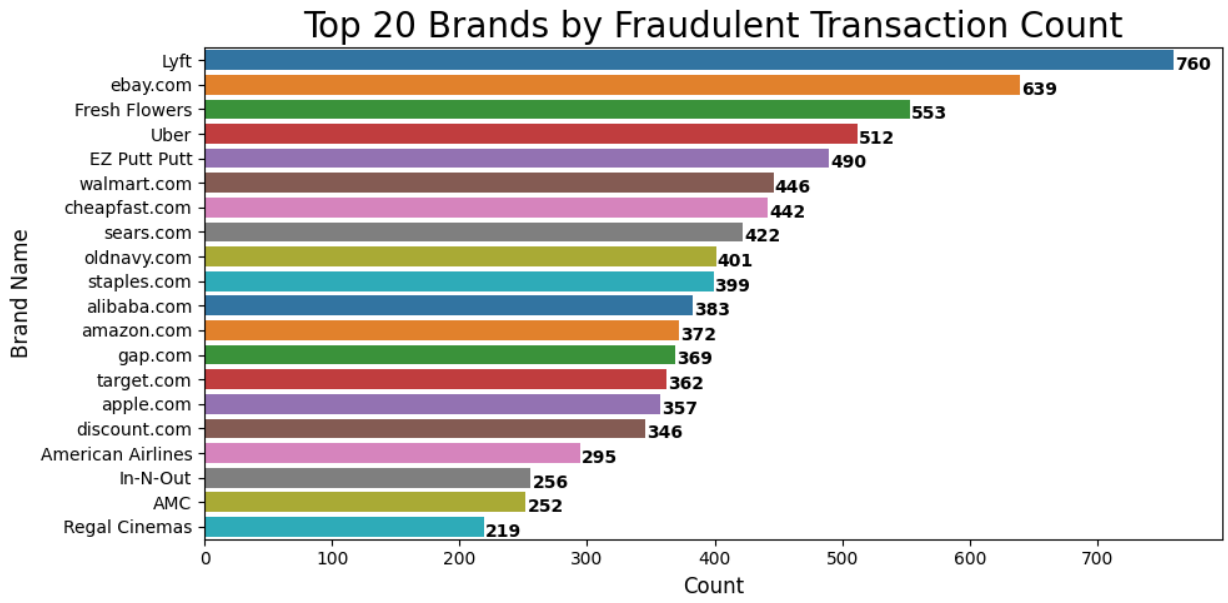


Figure 9: Top 20 brand by number of total fraudulent transactions

4.10 Top 20 Brands by Average Fraudulent Transaction Amount

Marriott Hotels has the largest average fraudulent transaction amount at $444.10

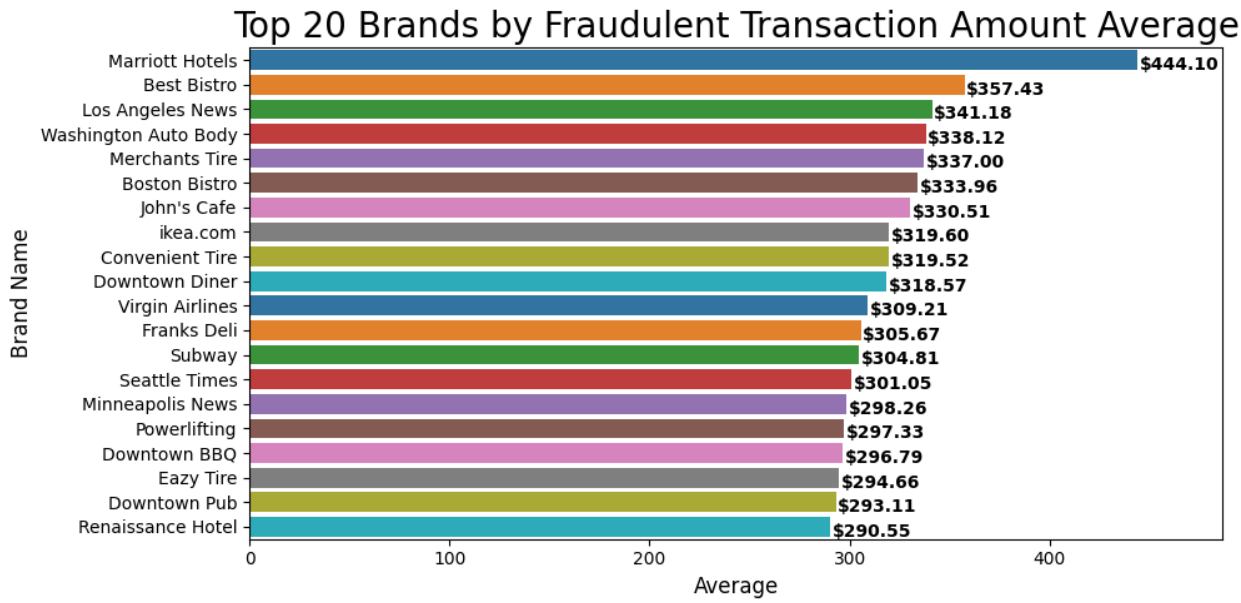


Figure 10: Top 20 brand by average fraudulent transaction amount

5 Unsupervised Learning

In this section we will extract patterns and structures from the dataset by grouping data points into clusters based on their inherent characteristics, enabling the discovery of natural groupings within the dataset.

5.1 Dimensionality Reduction: Singular Value Decomposition

Our preprocessed and transformed dataset has a total of 266 features. In order to more effectively cluster all these data points, we will perform dimensionality reduction using scikit-learn's **TruncatedSVD**, a dimensionality reduction technique commonly used in machine learning and data analysis that is particularly effective when dealing with high-dimensional data. The total number of dimenionally-reduced components for the model will be decided using the total explained variance ratio, a metric used to assess the amount of information or variability in the data.

With 12 components we can retain approximately 74% of the variance of the original dataset. In addition, we drastically reduced the number of features from 266 to only 12.

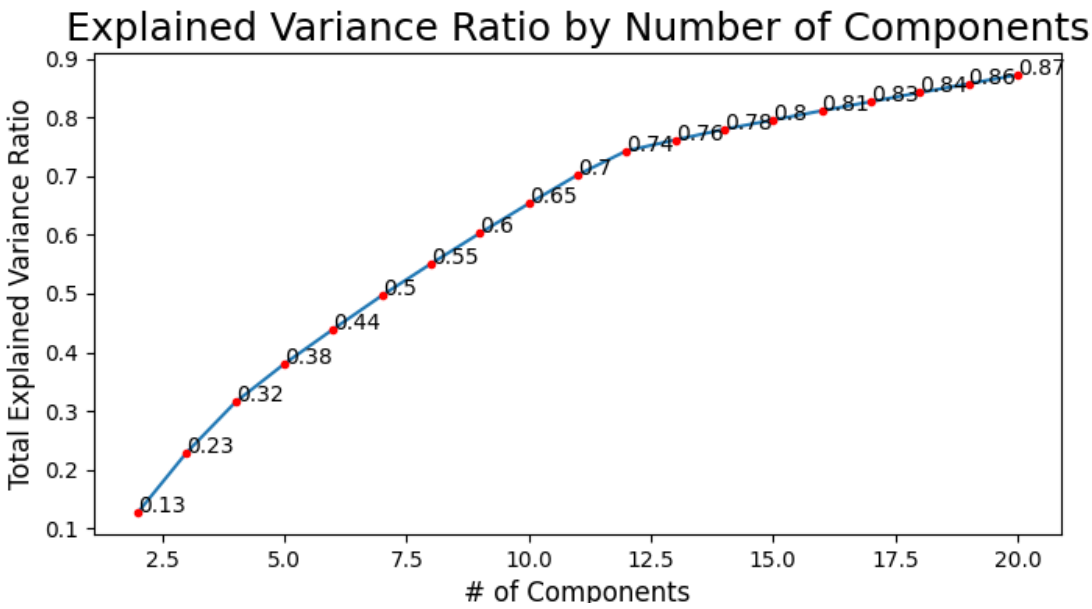


Figure 11: Total Explained Variance Ratio based on number of components

5.2 Clustering: KMeans

With our dimensionally-reduced dataset, we will build a KMeans clustering model to group those datapoints. The optimal number of clusters will be determined by calculating the within-cluster sum of squares for the model, a metric used in K-means clustering to evaluate the quality of the clustering solution.

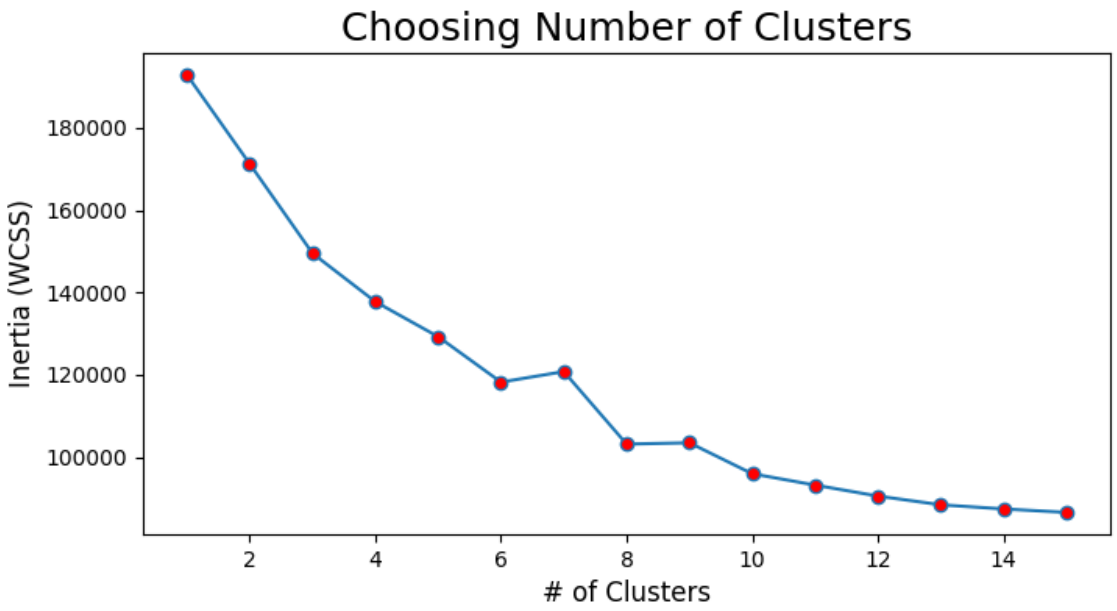


Figure 12: Within-Cluster Sum of Squares based on number of clusters

We will use 4 clusters for the KMeans algorithm, with a total within-cluster sum of squares of 140,216.

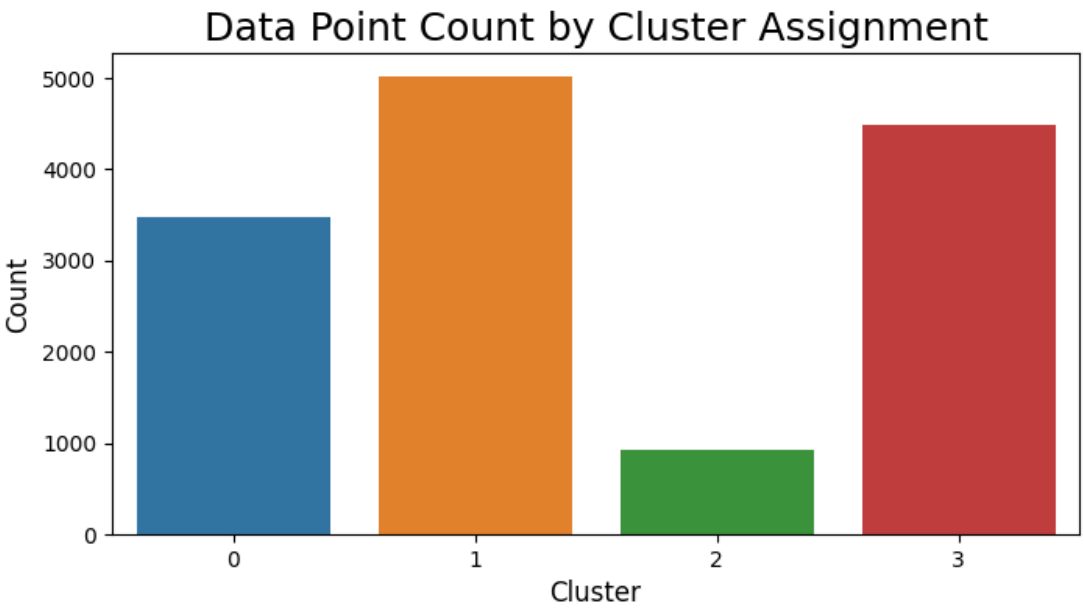


Figure 13: Caption

Cluster 1 has the largest count of data points, followed by cluster 3, cluster 0 and cluster 2.

6 Supervised Learning

Supervised learning is a machine learning approach where a model learns to make predictions or decisions based on labeled training data. In supervised learning, we have a dataset consisting of input features and corresponding target labels or outputs. The goal is to train a model that can generalize from the provided examples and accurately predict the target labels for new, unseen data. We will create two different machine learning model to classify the data as fraudulent or non-fraudulent.

6.1 Dataset Resampling

Due to our dependent variable being drastically imbalanced, we will need to resample the dataset to eliminate that imbalance. Since our minority class contains 12,417 records, we will use under-sampling, which is a method that aims to reduce the imbalance between classes by randomly removing samples from the majority class until the desired balance is achieved.

isFraud	Original	Under-Sampled
False	773946	12417
True	12417	12417

Table 8: Transaction by fraud status before and after resampling

Now both fraudulent and non-fraudulent transactions are equal in count, which will significantly increase our model performances.

6.2 Hyperparameter Tuning

For both the Logistic Regression and Random Forest models, hyperparameter estimation and tuning will be done using scikit-learn's **GridSearchCV**. GridSearchCV systematically searches and evaluates the performance of a model across a grid of possible hyperparameter values and returns the best-tuned model that will yield the greatest performance.

	param	value
0	C	0.500000
1	class_weight	NaN
2	dual	False
3	fit_intercept	True
4	intercept_scaling	1
5	l1_ratio	NaN
6	max_iter	500
7	multi_class	auto
8	n_jobs	NaN
9	penalty	l1
10	random_state	NaN
11	solver	saga
12	tol	0.000100
13	verbose	0
14	warm_start	False

(a) Logistic Regression

	param	value
0	bootstrap	True
1	ccp_alpha	0.000000
2	class_weight	NaN
3	criterion	gini
4	max_depth	5
5	max_features	auto
6	max_leaf_nodes	NaN
7	max_samples	NaN
8	min_impurity_decrease	0.000000
9	min_samples_leaf	4
10	min_samples_split	2
11	min_weight_fraction_leaf	0.000000
12	n_estimators	100
13	n_jobs	NaN
14	oob_score	True
15	random_state	NaN
16	verbose	0
17	warm_start	False

(b) Random Forest

Figure 14: Hyperparameters for both models

6.3 Logistic Regression

Logistic Regression models the relationship between the input features and the probability of belonging to a particular class. The input features are linearly combined with weights or coefficients, and then passed through a sigmoid or logistic function. The sigmoid function maps the linear combination to a value between 0 and 1, representing the probability of the input belonging to the positive class.

Using the best estimator result from GridSearchCV, we fit the training and testing dataset to the model and get the probabilities for the positive class. We then compute the ROC-AUC Curve to find the most appropriate threshold value for the model.

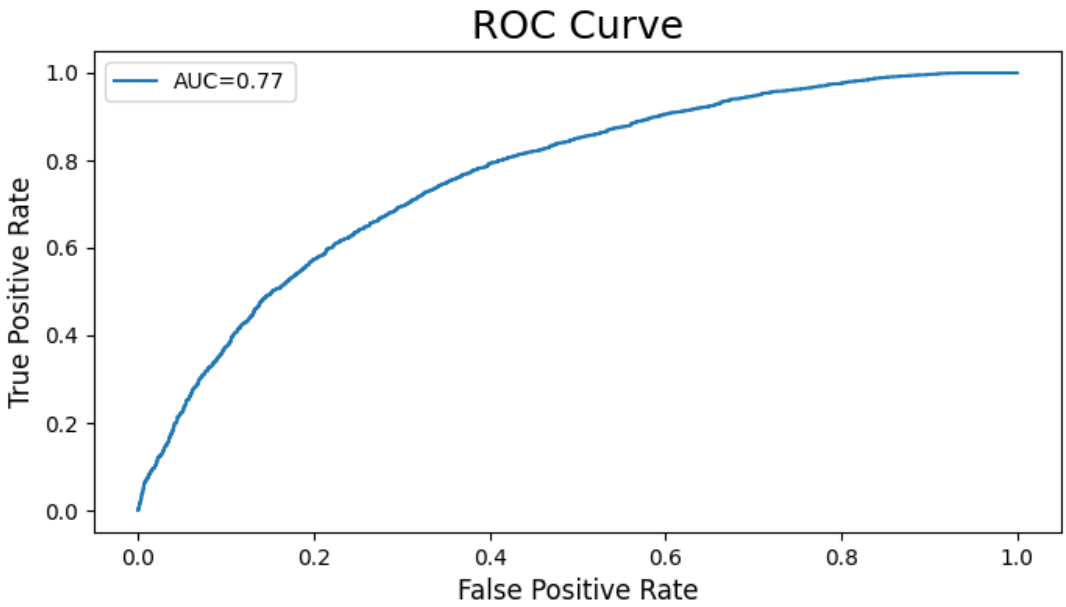


Figure 15: ROC-AUC curve for Logistic Regression

	thresh=0.3	thresh=0.35	thresh=0.4	thresh=0.45	thresh=0.5	thresh=0.55	thresh=0.6
Accuracy	0.634290	0.658782	0.679919	0.695689	0.699379	0.694682	0.683946
Recall/Sensitivity	0.930560	0.890641	0.840322	0.790003	0.729621	0.653137	0.561892
False Positive Rate	0.662081	0.573154	0.480537	0.398658	0.330872	0.263758	0.193960
Precision	0.584369	0.608526	0.636271	0.664691	0.688073	0.712404	0.743453
Specificity	0.337919	0.426846	0.519463	0.601342	0.669128	0.736242	0.806040
F1 Score	0.717909	0.723039	0.724198	0.721950	0.708238	0.681484	0.640046

Table 9: Model metrics for different threshold values

In our case, a threshold of 0.45 seems to yield the best results for our purpose. We want to correctly classify as many actual positive cases as possible but not at the cost of a very high False Positive rate. This threshold offers a good balance in this case, with a harmonic mean value (F1 Score) of 0.72198.

		Predicted	
		0	1
Actual	0	1792	1188
	1	626	2355

(a) Confusion matrix

	thresh=0.45
Accuracy	0.695689
Recall/Sensitivity	0.790003
False Positive Rate	0.398658
Precision	0.664691
Specificity	0.601342
F1 Score	0.721950

(b) Model metrics

Figure 16: Logistic Regression results with threshold equal to 0.45

Our final Logistic Regression model has an accuracy of 0.6957, True Positive Rate of 0.79 and a False Positive Rate of 0.3987. We will cross-validate the model performance by training it 50 times to get the mean metrics, distributions and confidence intervals for each metric.

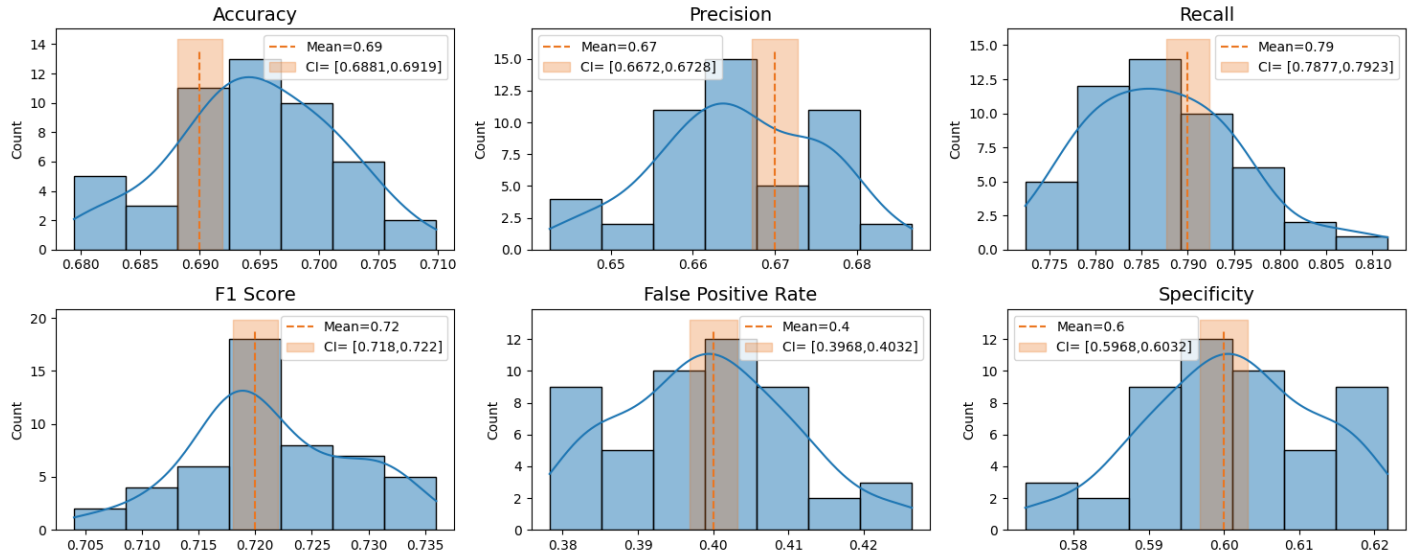


Figure 17: Logistic Regression metrics (n_fits=50)

Results: If we were to repeat the estimation process n number of times, in 95% of cases our metrics would fall in the following ranges:

- Accuracy: [0.6881, 0.6919]
- Precision: [0.6672, 0.6728]
- Recall: [0.7877, 0.7923]
- F1 Score: [0.7180, 0.7220]
- False Positive Rate: [0.3968, 0.4032]
- Specificity: [0.5968, 0.6032]

6.4 Logistic Regression Feature Importances

In order to determine what the most important features for the Logistic Regression model are, we will look at the top 20 features with the largest absolute coefficients. The magnitude of the coefficient indicates the strength of the relationship between the independent variable and the log-odds of the event. A larger magnitude suggests a stronger influence on the outcome.

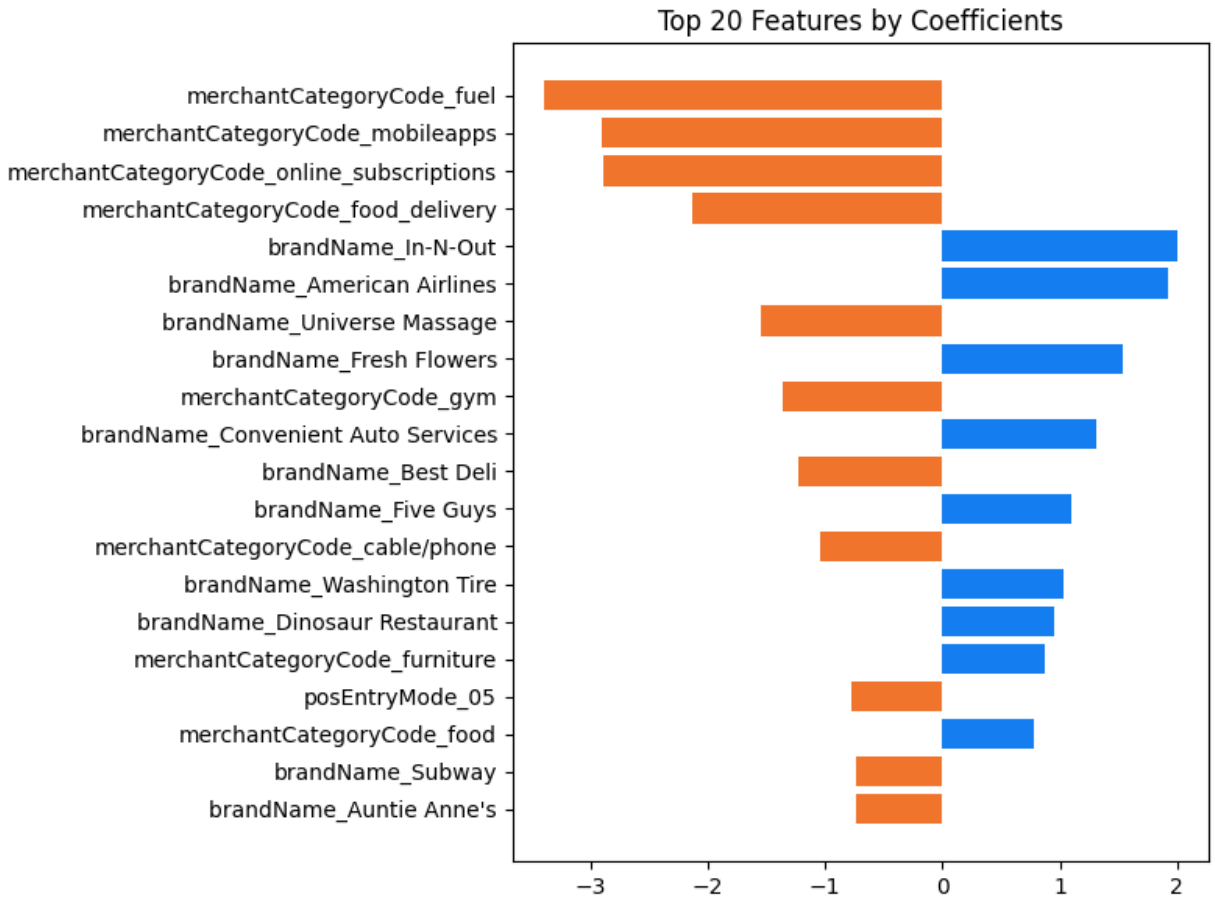


Figure 18: Most important features for Linear Regression model

The most important feature by coefficient effect is **merchantCategoryCode_fuel**.

Feature	Coef.
merchantCategoryCode_fuel	-3.393160
merchantCategoryCode_mobileapps	-2.907693
merchantCategoryCode_online_subscriptions	-2.887401
merchantCategoryCode_food_delivery	-2.127403
brandName_In-N-Out	1.998216
brandName_American Airlines	1.917929
brandName_Universe Massage	-1.545704
brandName_Fresh Flowers	1.531638
merchantCategoryCode_gym	-1.364648
brandName_Convenient Auto Services	1.311114
brandName_Best Deli	-1.224450
brandName_Five Guys	1.104633
merchantCategoryCode_cable/phone	-1.039582
brandName_Washington Tire	1.028438
brandName_Dinosaur Restaurant	0.947981
merchantCategoryCode_furniture	0.876960
posEntryMode_05	-0.782440
merchantCategoryCode_food	0.777277
brandName_Subway	-0.737239
brandName_Auntie Anne's	-0.730191

Table 10: Features with largest absolute coefficients for Logistic Regression model

6.5 Random Forest

Random Forest is an ensemble machine learning algorithm that is composed of multiple decision trees, with each tree trained independently on a random subset of the training data. The class with the most votes across all the trees is selected as the predicted class. It is less prone to overfitting and able to handle data with a large number of features, both categorical and numerical.

Same as with our Linear Regression model, we will use the best estimator resulting from GridSearchCV and fit preprocessed dataset.

	thresh=0.3	thresh=0.35	thresh=0.4	thresh=0.45	thresh=0.5	thresh=0.55	thresh=0.6
Accuracy	0.517698	0.532293	0.571213	0.651065	0.687301	0.657608	0.580607
Recall/Sensitivity	1.000000	0.998658	0.969473	0.877893	0.723247	0.454881	0.210332
False Positive Rate	0.964765	0.934228	0.827181	0.575839	0.348658	0.139597	0.048993
Precision	0.509051	0.516751	0.539683	0.603970	0.674804	0.765237	0.811125
Specificity	0.035235	0.065772	0.172819	0.424161	0.651342	0.860403	0.951007
F1 Score	0.674663	0.681080	0.693378	0.715614	0.698187	0.570587	0.334044

Table 11: Model metrics for different threshold values

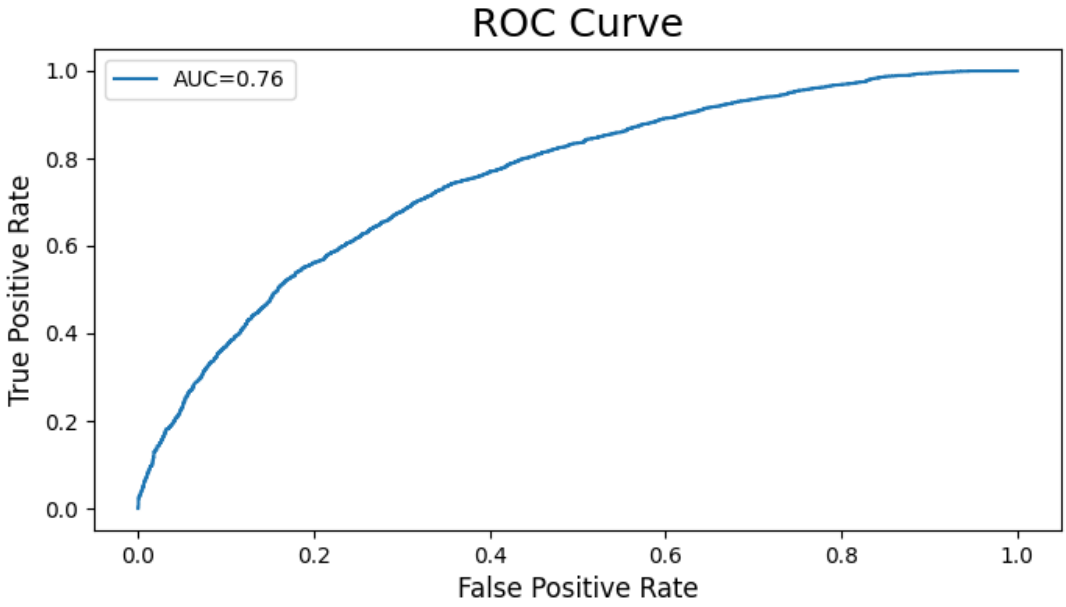


Figure 19: ROC-AUC curve for Random Forest

For this model, a threshold of 0.5 seems to be the most appropriate, with a harmonic mean value (F1 Score) of 0.698187.

		Predicted	
		0	1
Actual	0	1992	988
	1	865	2116

(a) Confusion matrix

	thresh=0.5
Accuracy	0.689146
Recall/Sensitivity	0.709829
False Positive Rate	0.331544
Precision	0.681701
Specificity	0.668456
F1 Score	0.695481

(b) Model metrics

Figure 20: Random Forest results with threshold equal to 0.50

Our final Random Forest model has an accuracy of 0.6891, True Positive Rate of 0.71 and a False Positive Rate of 0.33. As with the Linear Regression model, we will cross-validate the model performance by training it 50 times to get the mean metrics, distributions and confidence intervals for each metric.

Results: If we were to repeat the estimation process n number of times, in 95% of cases our metrics would fall in the following ranges:

- Accuracy: [0.6879, 0.6921]
- Precision: [0.6866, 0.6934]
- Recall: [0.6942, 0.7058]

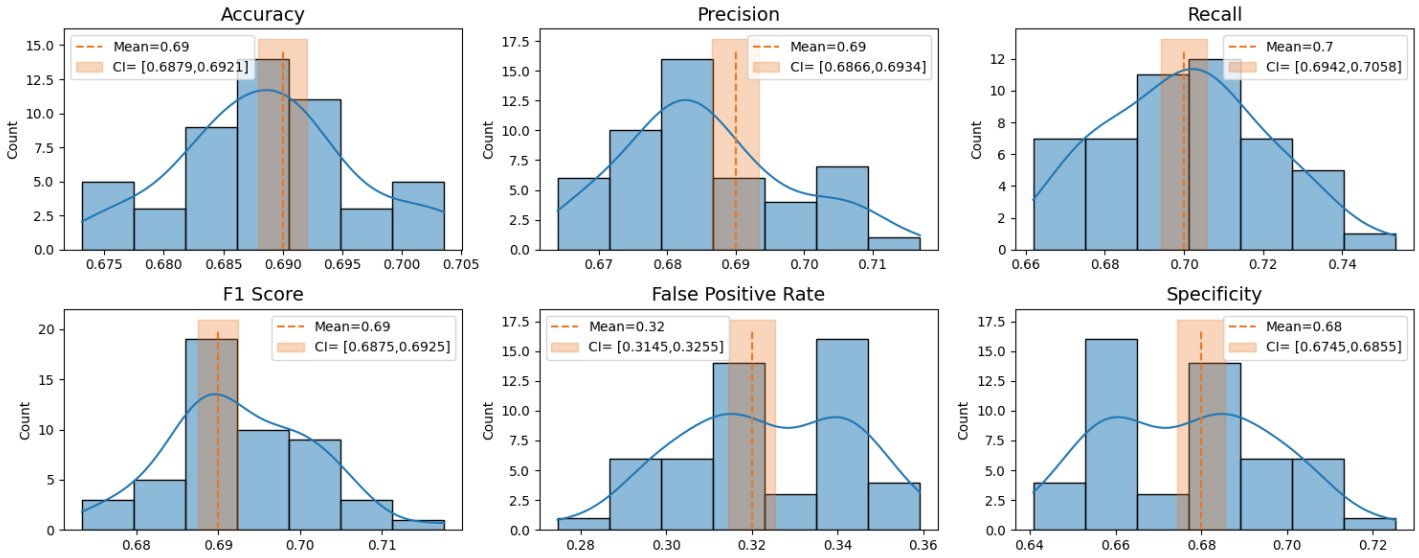


Figure 21: Logistic Regression metrics (n_fits=50)

- F1 Score: [0.6875, 0.6925]
- False Positive Rate: [0.3145, 0.3255]
- Specificity: [0.6745, 0.6855]

6.6 Random Forest Feature Importances

Random Forest feature importances provide a measure of the relative importance of each feature in a Random Forest model. The feature importances are calculated based on the impurity reduction or information gain that each feature contributes to the overall performance of the Random Forest. The total sum of all values equals to 1, so each value gives us the proportion of importance in the total model.

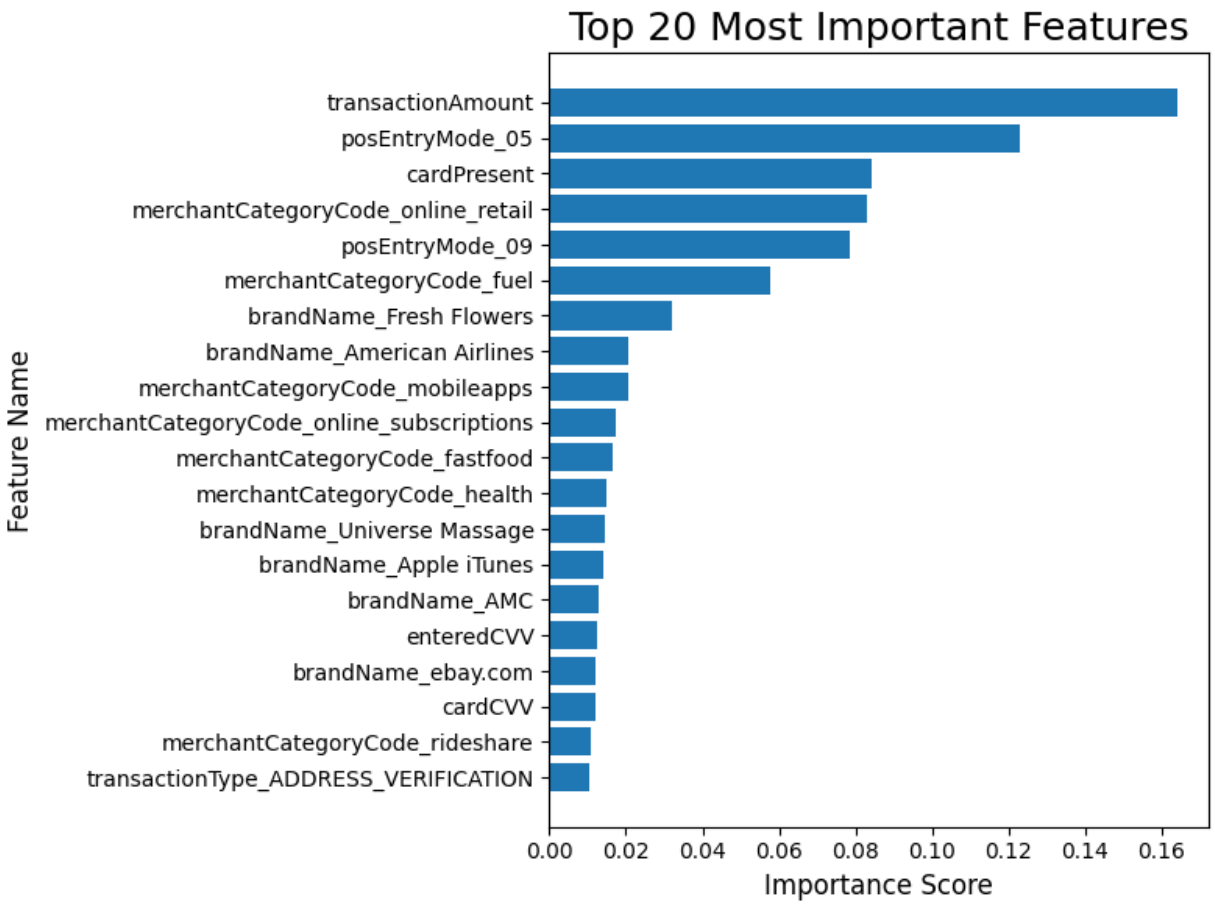


Figure 22: Most important features for Random Forest model

The most relevant feature in determining whether a transaction is fraudulent is **transactionAmount** (0.16), followed by **posEntryMode_05** (0.12) and **cardPresent** (0.08).

Feature	Proportion
transactionAmount	0.164092
posEntryMode_05	0.123055
cardPresent	0.084142
merchantCategoryCode_online_retail	0.082937
posEntryMode_09	0.078340
merchantCategoryCode_fuel	0.057539
brandName_Fresh Flowers	0.032164
brandName_American Airlines	0.020716
merchantCategoryCode_mobileapps	0.020647
merchantCategoryCode_online_subscriptions	0.017387
merchantCategoryCode_fastfood	0.016681
merchantCategoryCode_health	0.014733
brandName_Universe Massage	0.014493
brandName_Apple iTunes	0.014050
brandName_AMC	0.012724
enteredCVV	0.012533
brandName_ebay.com	0.012169
cardCVV	0.011947
merchantCategoryCode_rideshare	0.010692
transactionType_ADDRESS_VERIFICATION	0.010458

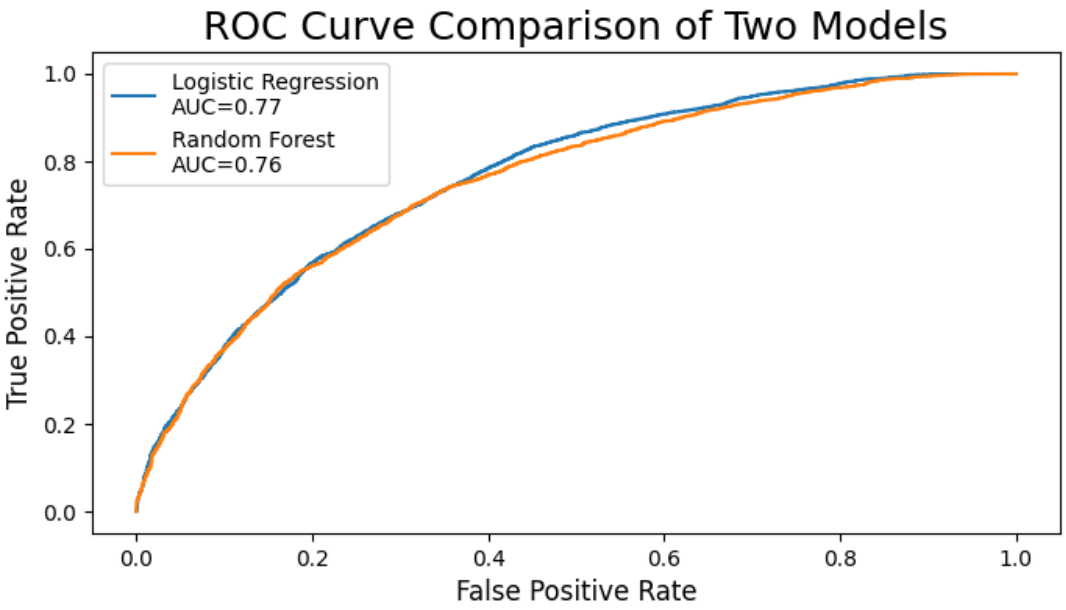
Table 12: Features with largest importance for Random Forest model

7 Model Comparison

Out of the two classification models, Random Forest seems to have the best performance for our purposes, with a recall score of 0.715137 and a False Positive Rate of 0.332624. Althouth Logistic Regression is outperforming the Random Forest model when it comes to recall, it also has a relatively high False Positive Rate in comparison.

	Logistic Regression (thresh=0.45)	Random Forest (thresh=0.5)
Accuracy	0.691585	0.691048
Recall/Sensitivity	0.786623	0.715137
False Positive Rate	0.401809	0.332624
Precision	0.657984	0.678746
Specificity	0.598191	0.667376
F1 Score	0.716576	0.696466

(a) Model metrics



(b) ROC-AUC Curves

Figure 23: Comparison of both classification algorithms