



Inspiring Excellence

Project Title

Fraud Detection Using Machine Learning

Group:09

Group member 1:

Name: Abu Sufian Mahin

ID : 23101013

Group member 2:

Name: Mahmudul Hasan Jim

ID : 24241067

Department :

Computer Science and Engineering

Course: CSE422

Section	Page No.
1. Introduction	3
2. Dataset Description	2
3. Imbalanced Dataset	8
4. Exploratory Data Analysis	9
5. Dataset Preprocessing	12
6. Model Training and Testing	13
7. Final score of classifier models	19
8. Challenges faced	20

Introduction

The goal of this project is to determine fraudulent financial transactions by means of data analysis. With increased online payments, fraud has become a major concern. The aim is to develop a model that will detect fraudulent transactions and avoid financial loss. The driving force is enhanced security and protection of users in online financial systems.

Dataset Description

- **Features details:**

The dataset contains 11 features represented as columns. These are the features -

1.step

2.type

3.amount

4.nameOrig

5.oldbalanceOrg

6.newbalanceOrig

7.nameDest

8.oldbalanceDest

9.newbalanceDest

10.isFraud

11.isFlaggedFraud

- **Problem Type:**

This is a classification problem. Target variable isFraud can have two values (0 or 1), where:

0 = Authentic transaction

1 = Fraudulent transaction

Since we are predicting on categories, it is clearly classification. Also as we are predicting discrete labels (fraud or not fraud), it falls under binary classification.

- **Number of Data Points:**

The dataset has 6362620 data points represented as rows in the dataset.

- **Type of Features:**

The data includes both Quantitative and Categorical features:

- Quantitative Features (continuous/numerical):

There are a total of 8 quantitative features.

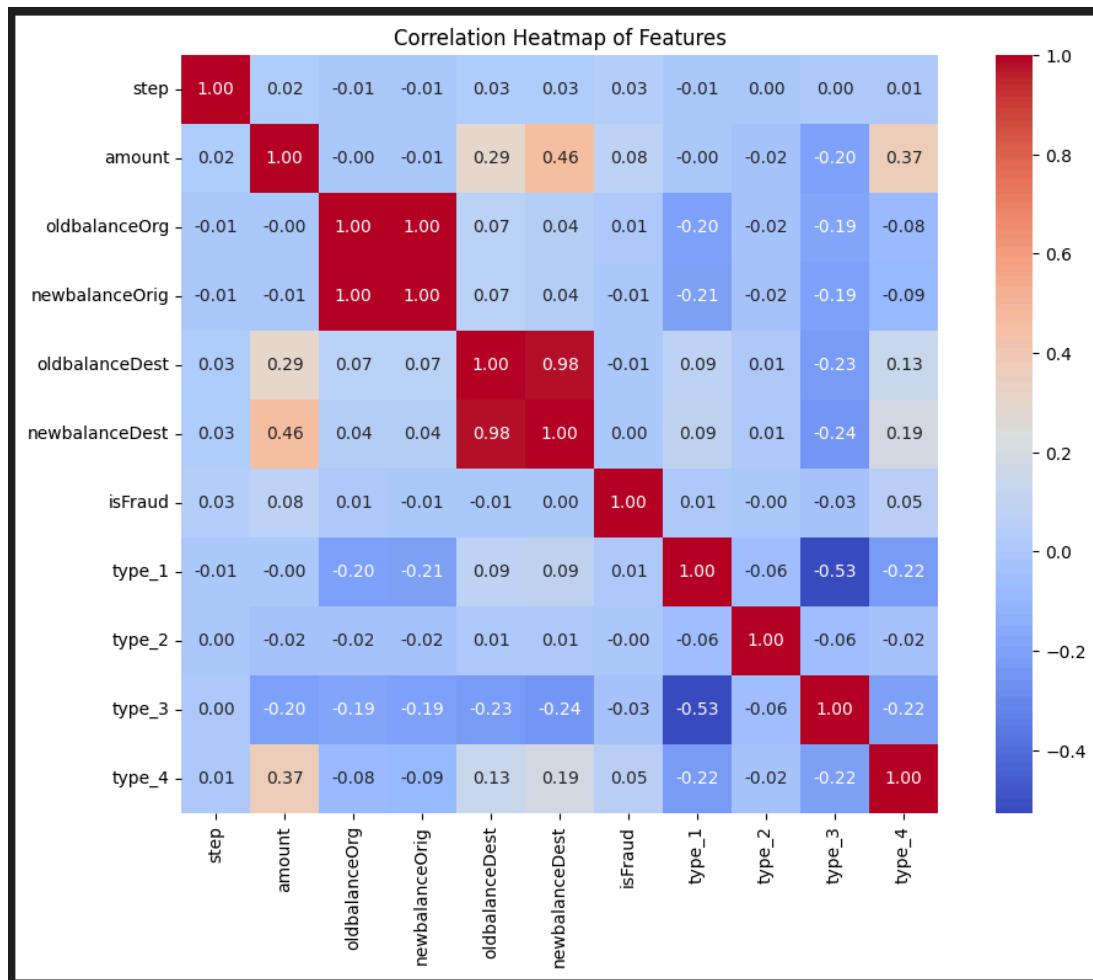
amount, oldbalanceOrg, newbalanceOrig, oldbalanceDest, newbalanceDest, step

- Categorical Features:

There are a total of 3 categorical features.

type, nameOrig, nameDest

- **Correlations : (heatmap)**



- **Observations from the correlations:**

1. Strong correlations:

- "oldbalanceOrig" and "newbalanceOrig" these two attributes are in perfect correlation (1.00). This indicates these two values are identical.

- Similarly, "oldbalanceDest" and "newbalanceDest" also possess almost perfect correlation (0.98), indicating they are almost the same.

2. Moderate correlations between Amount and Balances Destination:

- "amount" is in moderate correlation with both "oldbalanceDest" (0.29) and "newbalanceDest" (0.46), which indicates higher transaction amounts are partly related with higher destination balances.

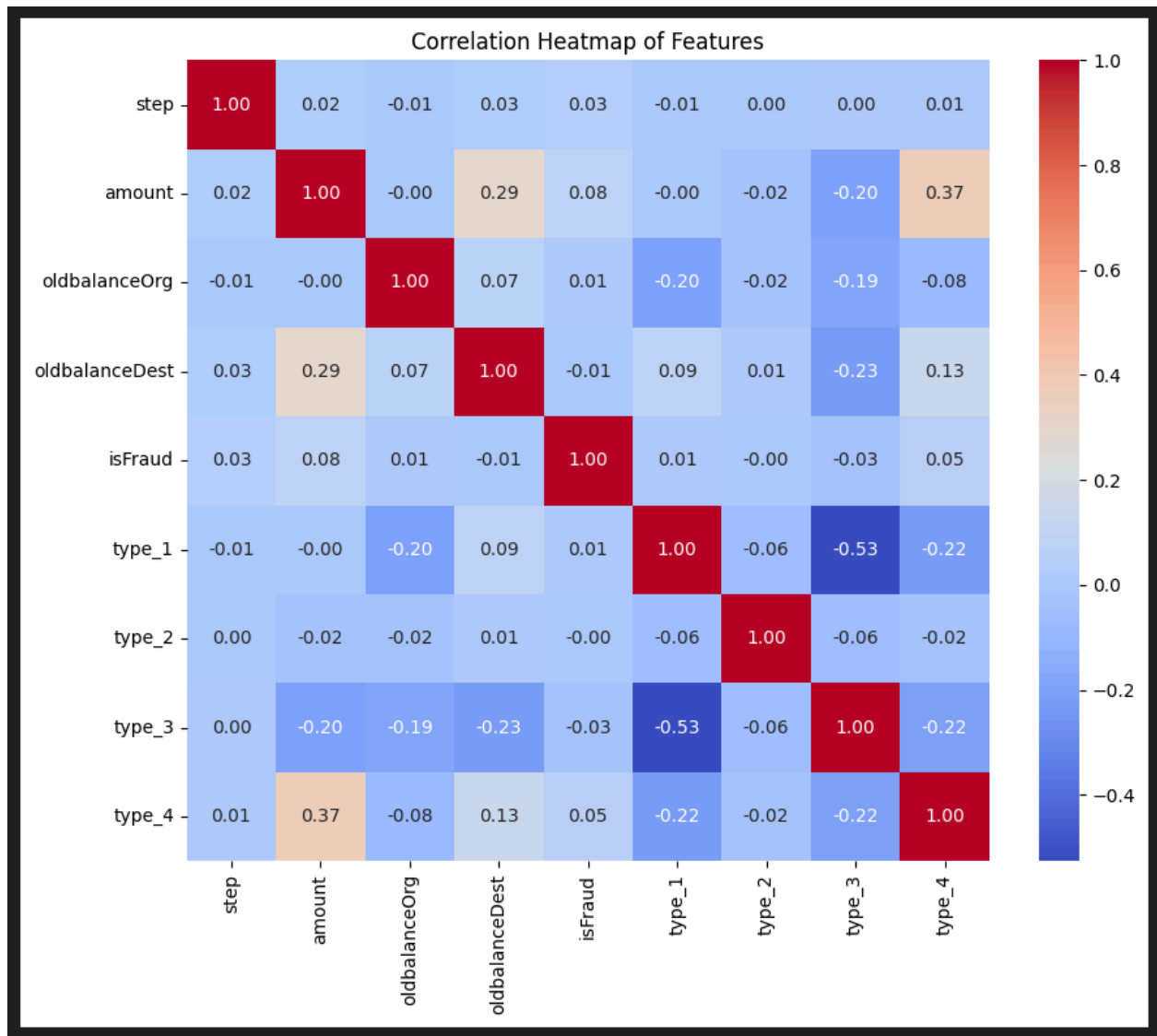
3. Correlation between Transaction Types:

- There is strong negative correlation between some transaction types (type_1 and type_3, with -0.53). This simply shows that a transaction can be only of one type at a time, which is natural.
- type_4 has weak positive correlation with amount (0.37), showing that this type of transaction is of larger amounts.

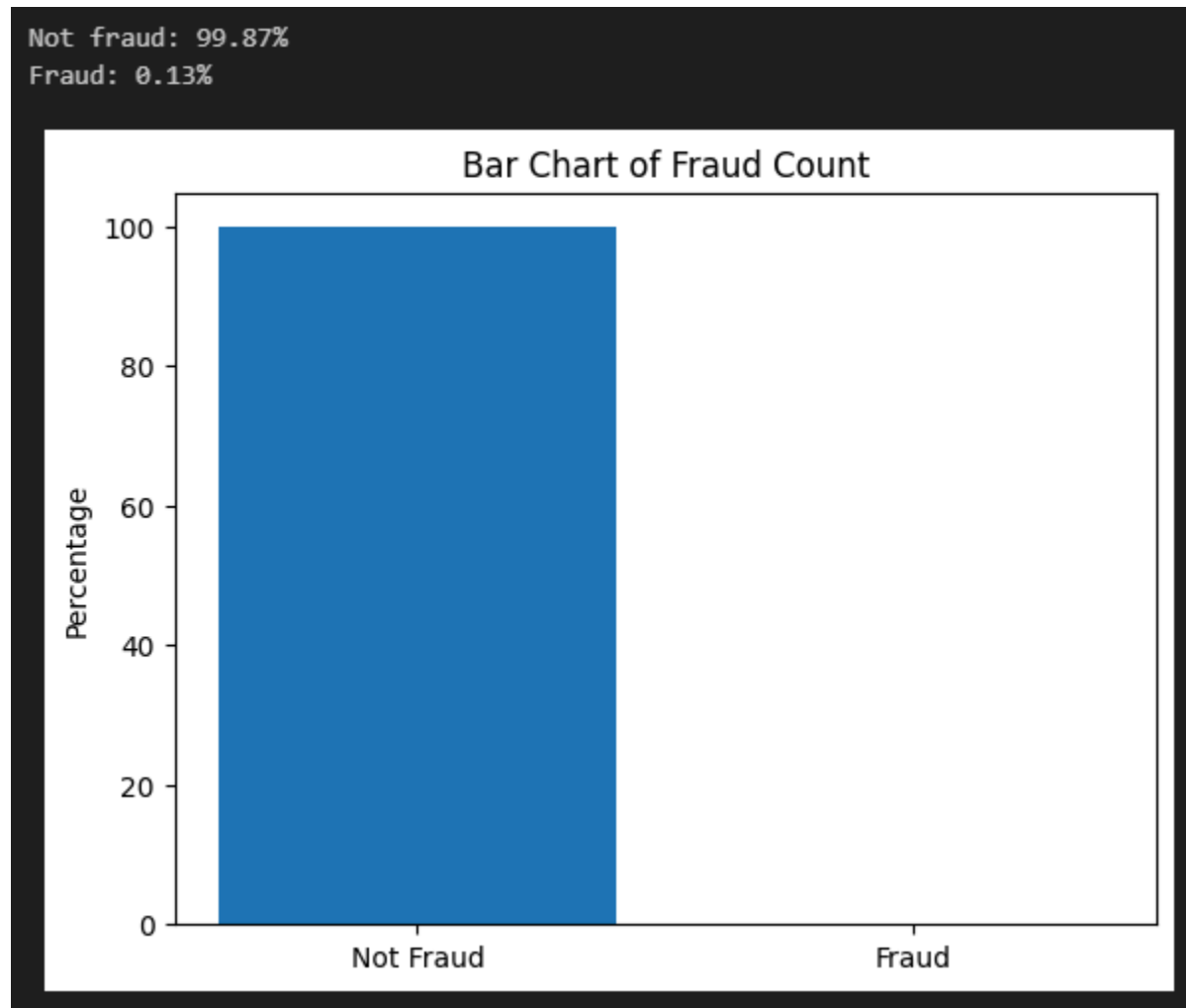
4. Target feature ("isFraud") is not Strongly Correlated With Any One Feature

- isFraud is poorly correlated with all other features. This suggests that fraud detection likely depends on complex combinations or patterns, not a single variable.

After dropping “newbalanceOrig” and “newbalanceDest”



Imbalanced Dataset

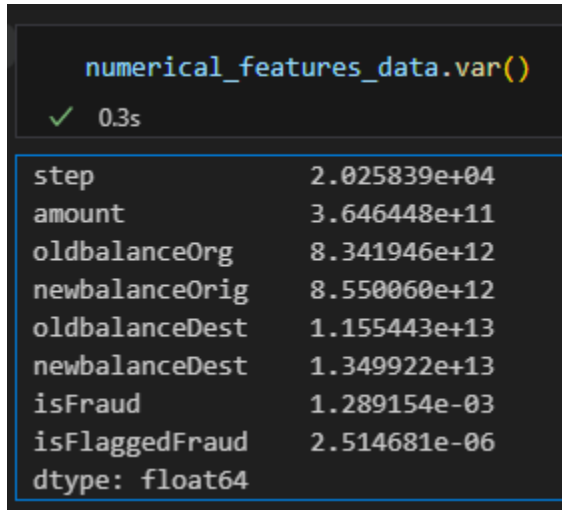


There is a massive class imbalance in the dataset. This biases machine learning models towards the majority class. The model becomes good at predicting the majority class but bad at the minority one. Overall accuracy can be misleadingly high.

Detection of the rare, valuable class becomes difficult. Fixing this imbalance is crucial for a helpful model. The model will be biased towards 'not fraud' results.

Exploratory Data Analysis

1. Variance



```
numerical_features_data.var()
```

✓ 0.3s

step	2.025839e+04
amount	3.646448e+11
oldbalanceOrg	8.341946e+12
newbalanceOrig	8.550060e+12
oldbalanceDest	1.155443e+13
newbalanceDest	1.349922e+13
isFraud	1.289154e-03
isFlaggedFraud	2.514681e-06
dtype: float64	

step (2.03×10^4): Low variance — steps (likely time steps) are tightly clustered with minor spread.

amount (3.65×10^{11}): High variance — transaction amounts vary significantly, indicating a wide range of values from small to very large.

oldbalanceOrg (8.34×10^{12}): Very high variance — original account balances differ greatly across transactions.

newbalanceOrig (8.55×10^{12}): Very high variance — similar to original balances, post-transaction values vary widely.

oldbalanceDest (1.16×10^{13}): Extremely high variance — destination account balances vary drastically, suggesting large-scale transactions in some cases.

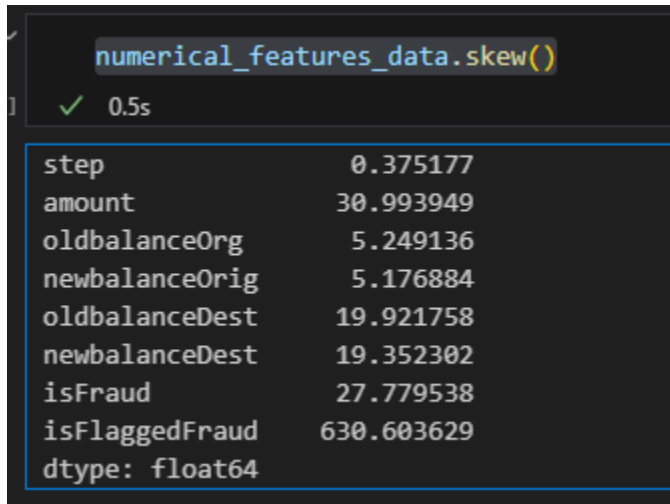
newbalanceDest (1.35×10^{13}): Extremely high variance — post-transaction balances of destination accounts also show a very large spread.

isFraud (1.29×10^{-3}): Very low variance — class imbalance exists; most values are 0 (non-fraud).

isFlaggedFraud (2.51×10^{-6}): Extremely low variance — almost all entries are 0 (not flagged), with very rare occurrences of flagged fraud.

So, this dataset needs to be scaled. In our case, we use Robust scaling.

2. Skewness



The screenshot shows a Jupyter Notebook cell with the code `numerical_features_data.skew()` executed. Below the code, a green checkmark and '0.5s' indicate successful execution. The output is a table of skewness values for various features.

step	0.375177
amount	30.993949
oldbalanceOrig	5.249136
newbalanceOrig	5.176884
oldbalanceDest	19.921758
newbalanceDest	19.352302
isFraud	27.779538
isFlaggedFraud	630.603629
dtype: float64	

step (0.375): Fairly symmetric distribution; no significant skew.

amount (30.99): Extremely positively skewed — most transactions are small, with a few very large ones.

oldbalanceOrig (5.25): Highly positively skewed — many original account balances are low; a few are very high.

newbalanceOrig (5.18): Highly positively skewed — same as above, post-transaction.

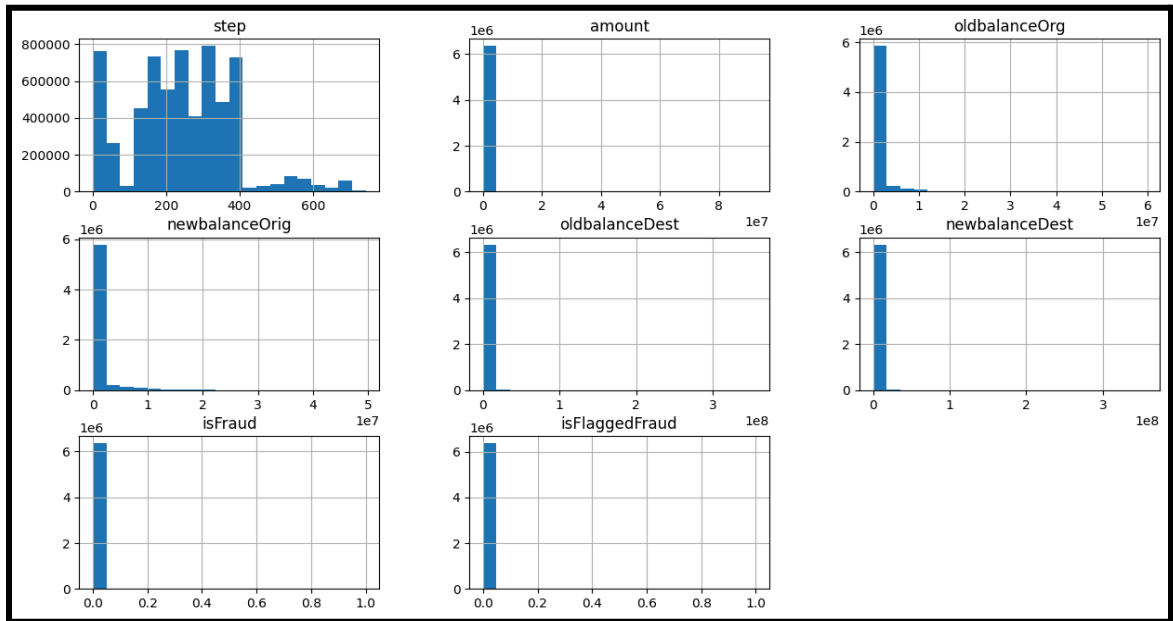
oldbalanceDest (19.92): Extremely positively skewed — most destination accounts have low balances, with a few outliers.

newbalanceDest (19.35): Extremely positively skewed — similar to oldbalanceDest.

isFraud (27.78): Extremely skewed due to class imbalance — very few fraud cases.

isFlaggedFraud (630.60): Extremely skewed — almost all entries are 0 (not flagged), making flagged fraud cases very rare.

3. Histogram



step: The distribution is fairly uniform with some variation. This indicates transactions are spread across different time steps, with certain periods showing more activity than others.

amount: The distribution is highly right-skewed. Most transactions involve small amounts, while a few transactions involve extremely large amounts.

oldbalanceOrig and **newbalanceOrig:** Both are extremely right-skewed. The majority of original account balances are low, with a small number of accounts having very high balances.

oldbalanceDest and **newbalanceDest:** These also show extreme right skewness. Most destination accounts have low balances, while a few receive large transactions.

isFraud: Almost all values are 0, confirming the presence of strong class imbalance. Fraudulent transactions are very rare in the dataset.

isFlaggedFraud: Nearly all values are 0, with very few instances flagged as fraud. This feature is extremely sparse.

Dataset Preprocessing

1. Null values

```
initial_null_values = payment_fraud_df.isnull().sum()
initial_null_values
```

✓ 0.5s

step	0
type	0
amount	0
nameOrig	0
oldbalanceOrig	0
newbalanceOrig	0
nameDest	0
oldbalanceDest	0
newbalanceDest	0
isFraud	0
isFlaggedFraud	0
dtype: int64	

As There are no null values in the dataset, No imputation or row/column removal is needed for missing data.

2. Categorical values

	step	type	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud
0	1	3	9839.64	170136.00	160296.36	0.00	0.00	0
1	1	3	1864.28	21249.00	19384.72	0.00	0.00	0
2	1	4	181.00	181.00	0.00	0.00	0.00	1
3	1	1	181.00	181.00	0.00	21182.00	0.00	1
4	1	3	11668.14	41554.00	29885.86	0.00	0.00	0

```
label_mapping = dict(zip(label_encoder.classes_, label_encoder.transform(label_encoder.classes_)))

print("Categorical values are encoded like this: ")
for key, value in label_mapping.items():
    print(f"{key} = {value}")
```

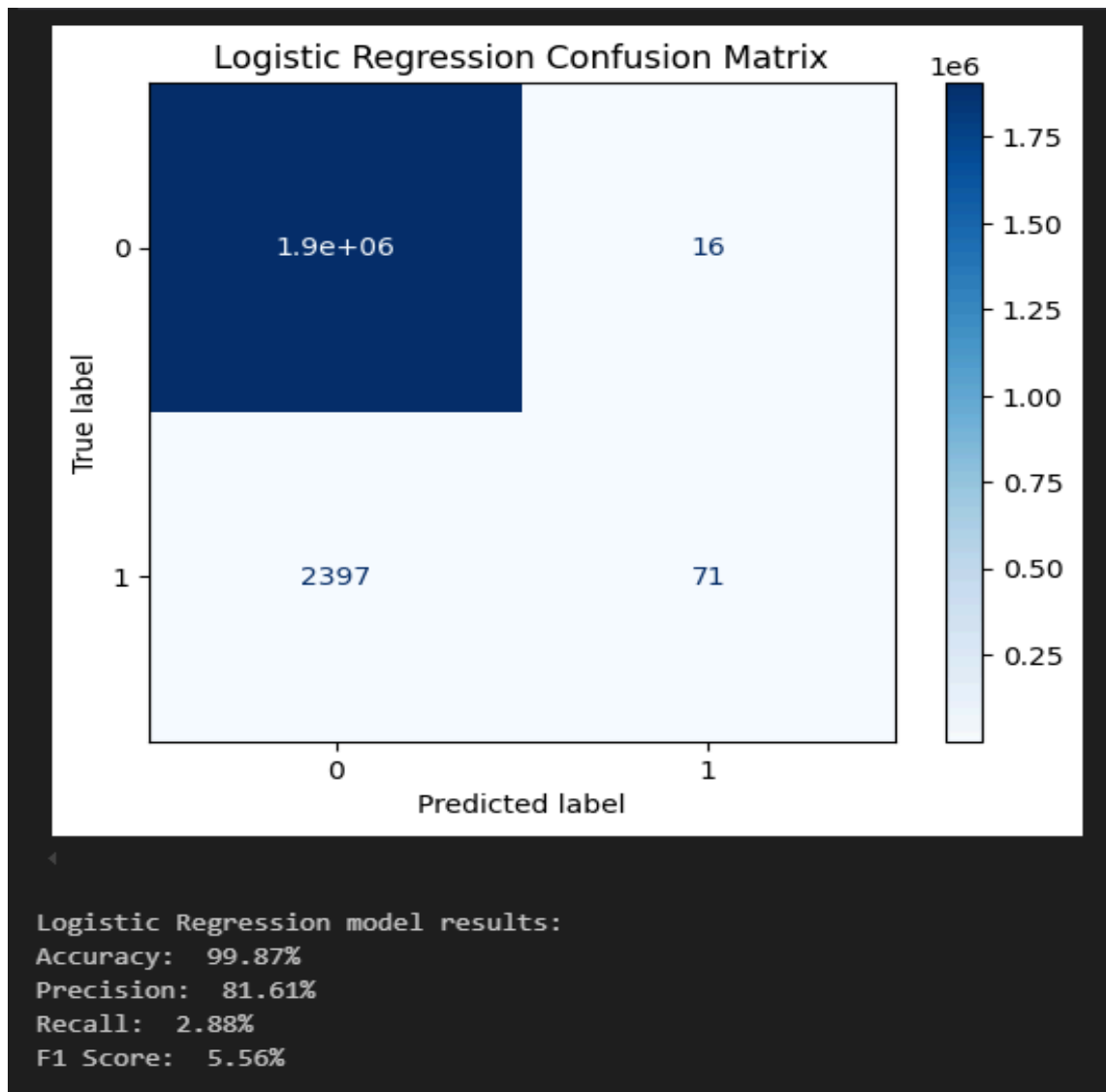
✓ 0.0s

```
Categorical values are encoded like this:
CASH_IN = 0
CASH_OUT = 1
DEBIT = 2
PAYMENT = 3
TRANSEER = 4
```

Machine learning models require numerical input, so we need to convert this categorical column into a numerical format using encoding to make it usable for training.

Model Training and Testing

1. Logistic regression



Conclusion:

- Interpreting the Results:

The model shows high accuracy but significantly low F1 and recall scores, especially in the minority class. This reflects the fact that the model over-fits toward the majority class and is failing to detect minority class instances sufficiently.

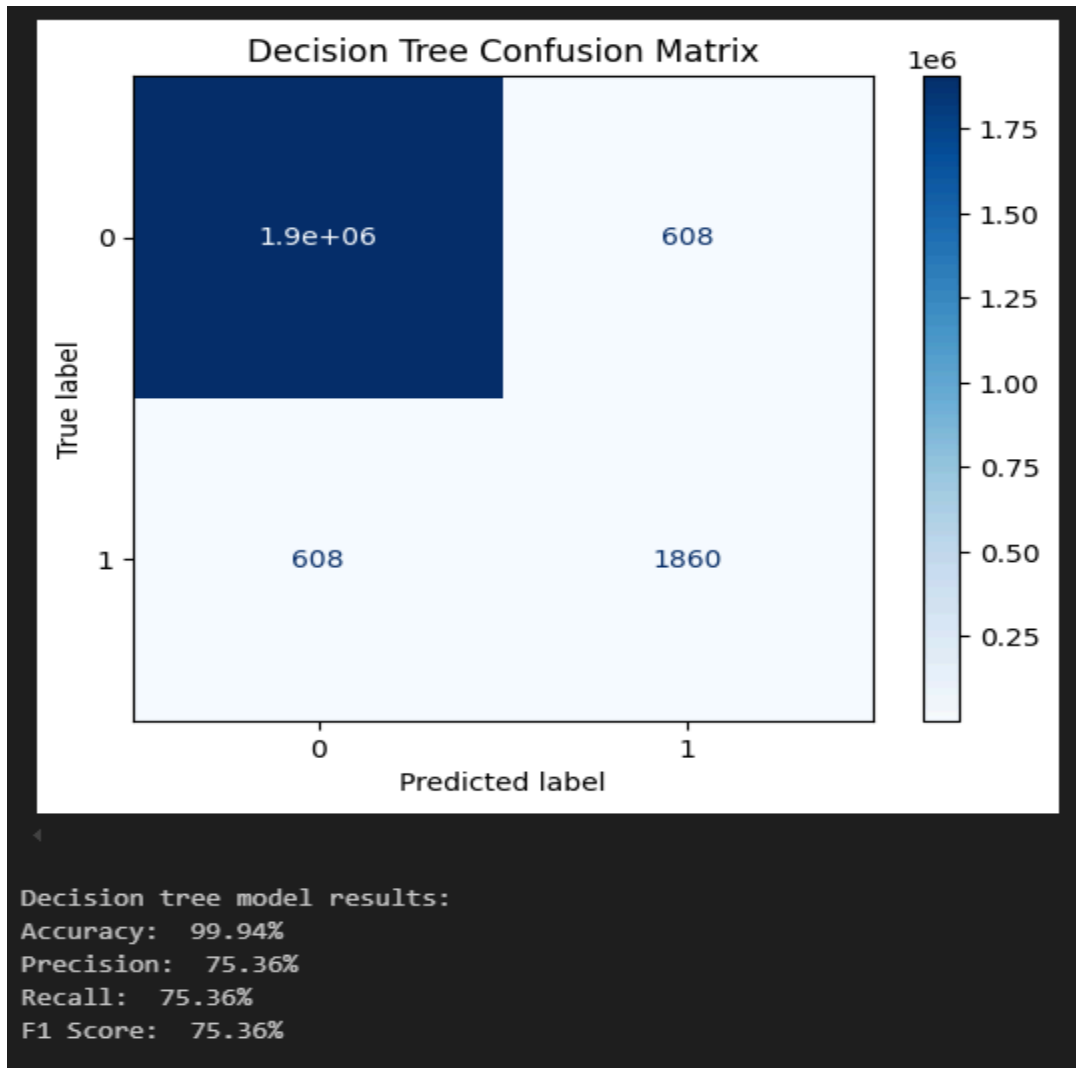
- Model Performance Comments

Even if total accuracy looks decent, it is deceptive because there is a very large class imbalance in the data. The model is bad at identifying the minority class, which is important in a fraud detection application. This renders the model inappropriate for actual use where fraud detection matters more than total accuracy.

- Explanation for These Results:

The reason the F1 and recall scores are low is mainly because of the class imbalance being extremely high. Because of the class imbalance the model has learned to favor the majority class because that provides more accuracy during training. That, of course, at the cost of missing minority class predictions, and that hurts precision and recall significantly.

2. Decision Tree



Conclusion:

- Interpreting the Results:

Decision Tree has extremely high accuracy (99.94%) but relatively low precision (74.92%), recall (75.77%), and F1 score (75.34%). This indicates that the model is overall good but is not good at predicting minority class instances.

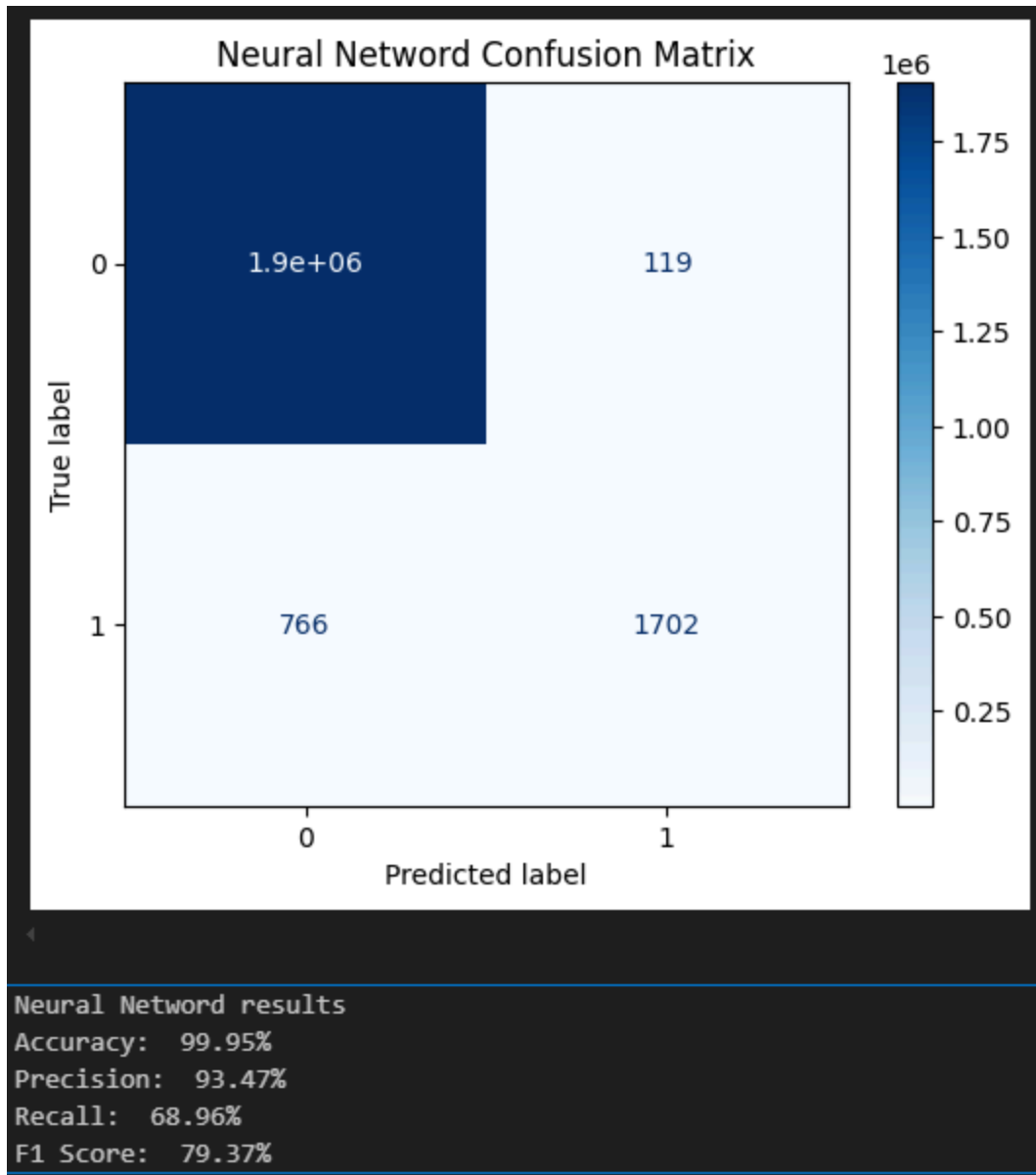
- Model Performance Comments:

The high accuracy is deceptive because of the extreme class imbalance of the data. The model may not be able to identify fraud cases, the rare but important ones. In a fraud detection system, such a trade-off is unacceptable since failing to detect fraudulent transactions is a critical failure.

- Explanation for These Results:

These performance problems are due primarily to severe class imbalance. The model is optimizing for accuracy by predicting the majority class more frequently, thus with low precision and recall on the minority class. This bias reduces the usefulness of the model in fraud detection.

3. Neural Network



- Interpreting the Results:

The Neural Network model has very high accuracy (99.95%) and precision (93.47%), but low recall (68.96%) and F1 score (79.37%). This means that although the model is extremely good at being right when it predicts fraud, it fails to capture a high number of actual fraud cases.

- Model Performance Comments:

Even with the high accuracy and precision, the low recall implies a lot of fraudulent transactions are not detected. This makes the model unsafe for use in fraud detection systems, where detecting as many cases of fraud as possible is more vital than evading false alarms.

- Explanation for These Results:

The disparity between recall and precision indicates the lasting effect of class imbalance. The model is conservative in predicting transactions as fraud, which leads to high precision but low recall. The conservativeness minimizes false positives but lets through a high number of fraudulent cases, lowering recall and F1 score.

Final score of Classifier models

Logistic Regression:							
				precision	recall	f1-score	support
			0	1.00	1.00	1.00	1906318
			1	0.82	0.03	0.06	2468
		accuracy				1.00	1908786
		macro avg		0.91	0.51	0.53	1908786
		weighted avg		1.00	1.00	1.00	1908786
Decision Tree							
				precision	recall	f1-score	support
			0	1.00	1.00	1.00	1906318
			1	0.75	0.76	0.75	2468
		accuracy				1.00	1908786
		macro avg		0.87	0.88	0.88	1908786
		weighted avg		1.00	1.00	1.00	1908786
Neural Network							
				precision	recall	f1-score	support
			0	1.00	1.00	1.00	1906318
			1	0.93	0.69	0.79	2468
		accuracy				1.00	1908786
		macro avg		0.97	0.84	0.90	1908786
		weighted avg		1.00	1.00	1.00	1908786

Challenges Faced:

Class Imbalance:

The class imbalance being extremely high made it a major challenge. In the dataset fraud cases made up only 0.13% of the data, making it difficult for the model to learn minority patterns.

Irrelevant Features:

One challenge I faced was handling irrelevant features like “nameOrig”, “nameDest” and “isFlaggedFraud”, which added noise and risked overfitting, reducing model performance.