Credit Card Transaction Analysis

Mining transaction data to detect fraudulent purchases

By: Kevin Corboy & Brendan McDonald

Intro and Problem Statement

Credit cards are part of most of our daily lives

Payment convenience = security cost, and fraud rates are on the rise

Early and accurate fraud detection is crucial to reduce costs

Credit Card Transactions Dataset

Overview:

Dataset includes ~1.3M rows x 24 columns of detailed transaction data

- Transaction time, value, customer and merchant details, demographics
- Flag for ~7K known fraud transactions

Questions:

- What patterns can we identify for fraud?
- Do some categories have a higher probability and prevalence of fraud?
- How accurately can we predict fraud?

Completed Work and Tools

1. Loaded data into dev environment
2. Reviewed summary stats and EDA
3. Developed additional features to train the model
4. Reviewed correlations
5. Performed data reduction
6. Implemented predictive model for fraudulent transactions
7. Evaluated model performance

Tools:

- Analysis: Pandas, NumPy
- ML algorithms: Scikit Learn
- Visualization: Matplotlib, Seaborn

Dev Env: Jupyter Notebook
Github: github.com/brendan-mcdonald-cspb/4502GROUP10

Exploratory Data Analysis (EDA)

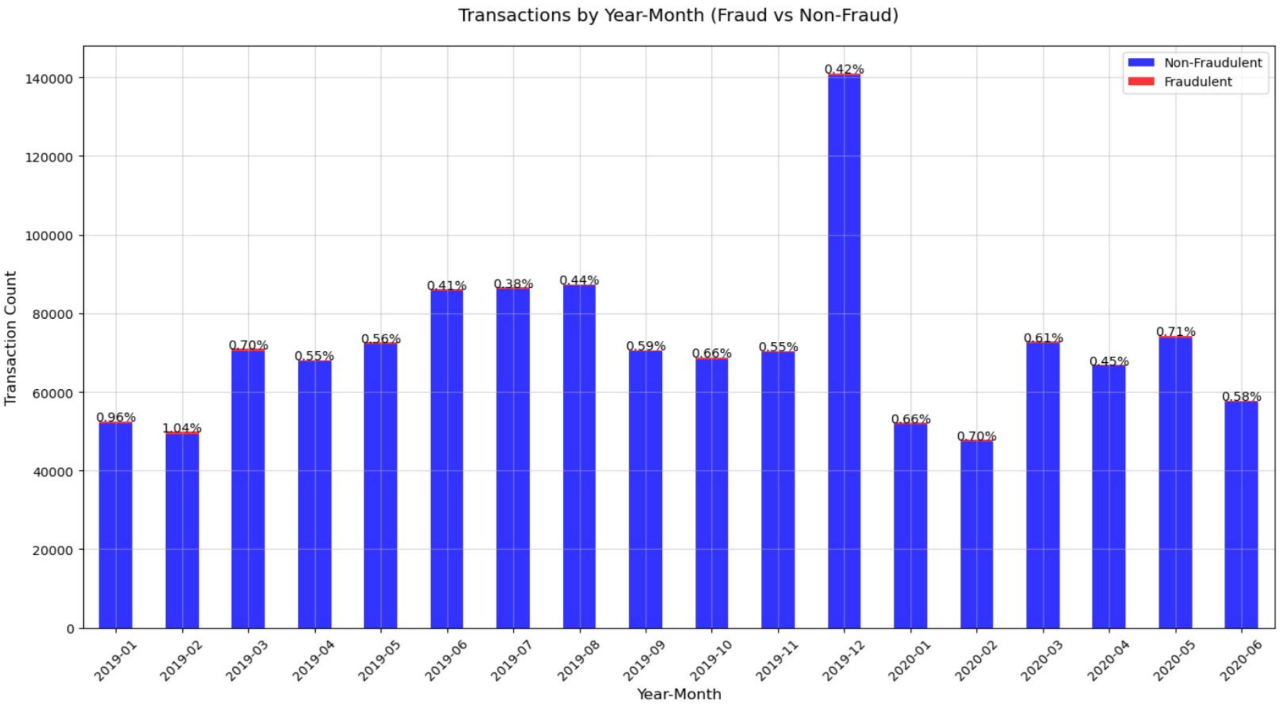
We centered our exploratory data analysis around a few key metrics in the dataset:

- Transaction time
- Transaction value
- Transaction distance
- Are certain merchant categories prone to fraudulent transactions?

EDA: Fraud Counts and Rate by Month

- Monthly transactions vary presumably due to consumer spending habits

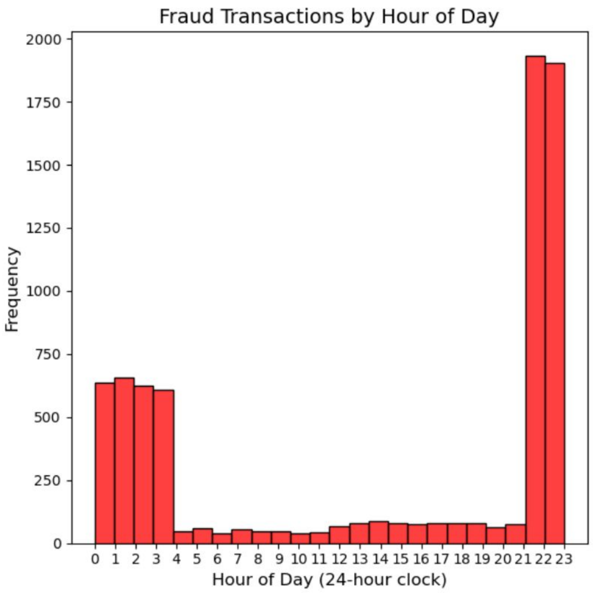
- Monthly fraud rates vary between ~.5% and ~1% of total transactions

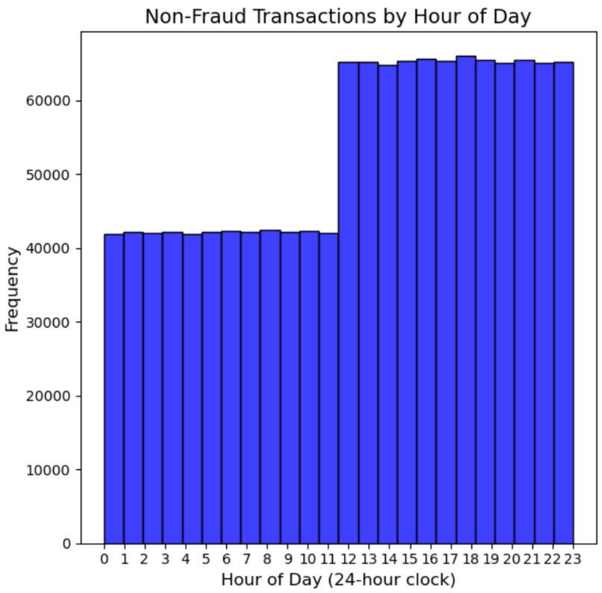


EDA: Fraud Frequency by Hour

- Fraudulent transactions peak between 10PM and midnight, then are reduced slightly through early morning

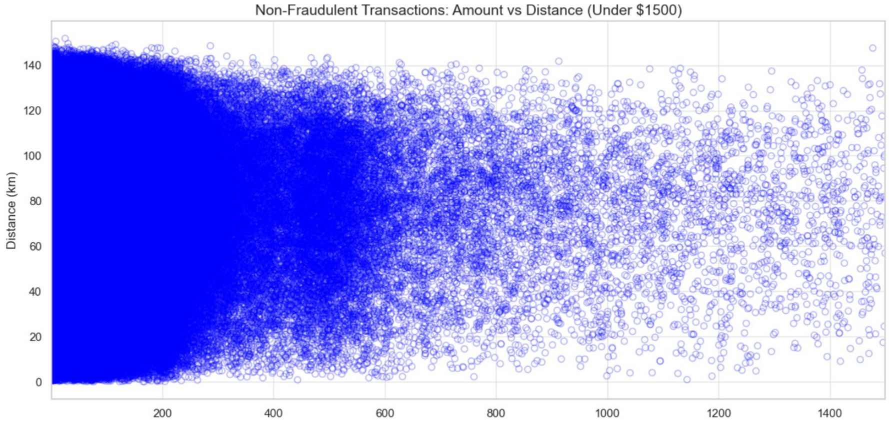
- This unique distribution may prove useful, so we will develop a feature flag for transactions :
    - 10PM - 12AM = high risk (“2”)
    - 12AM - 4AM = medium risk (“1”)
    - Else low risk (“0”)

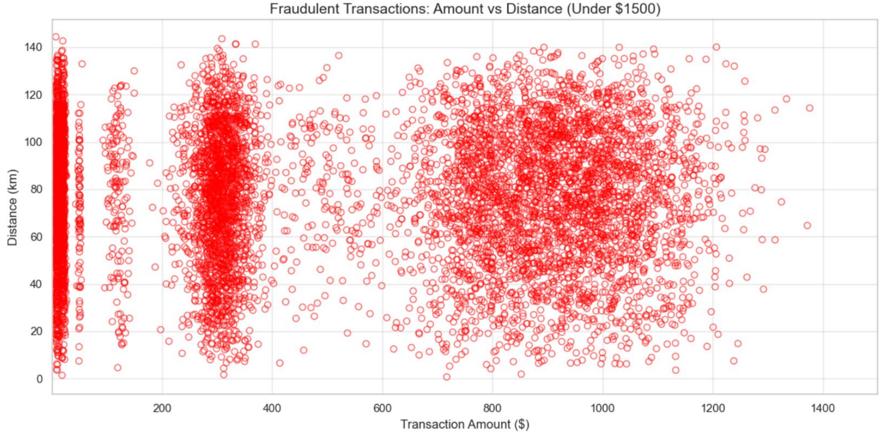




EDA: Transaction Distance and Value

- Fraud rates increase slightly as the distance between customer and merchant increases:
  - Local (<10 km): 0.511%
  - Regional (<50 km): 0.562%
  - Long Distance (> 50 km): 0.583%
- We developed a feature to add the distance in km to each transaction, as well as a flag for long distance transactions
- Clustering mentioned on previous slide is very apparent here as well

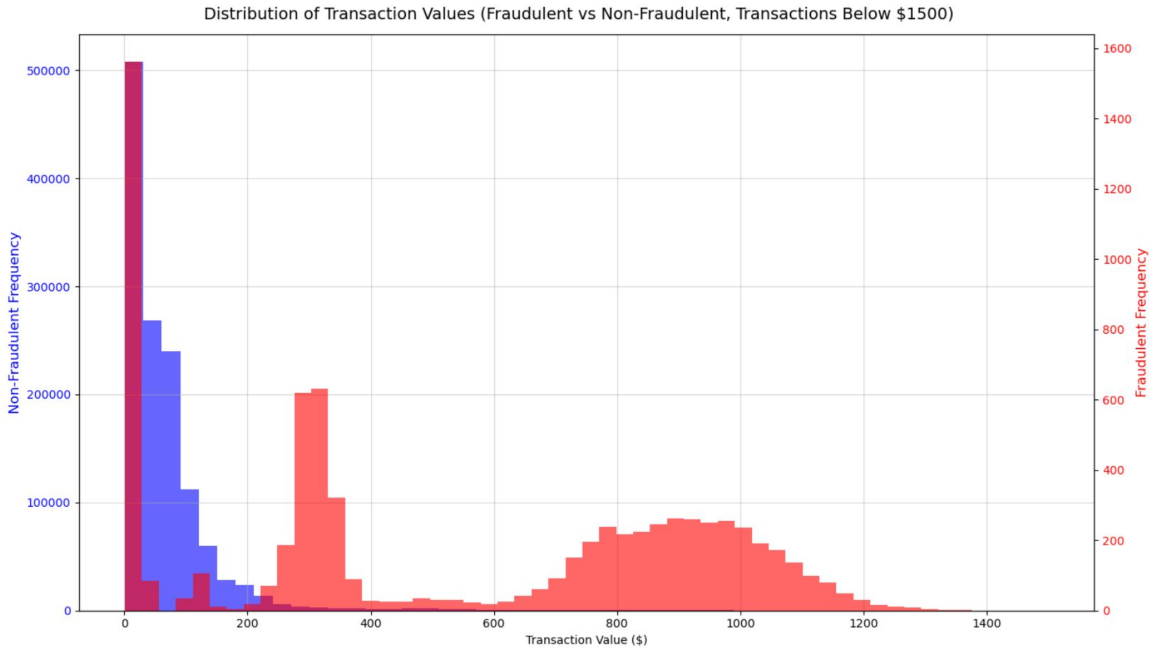




EDA: Transaction Value by Category

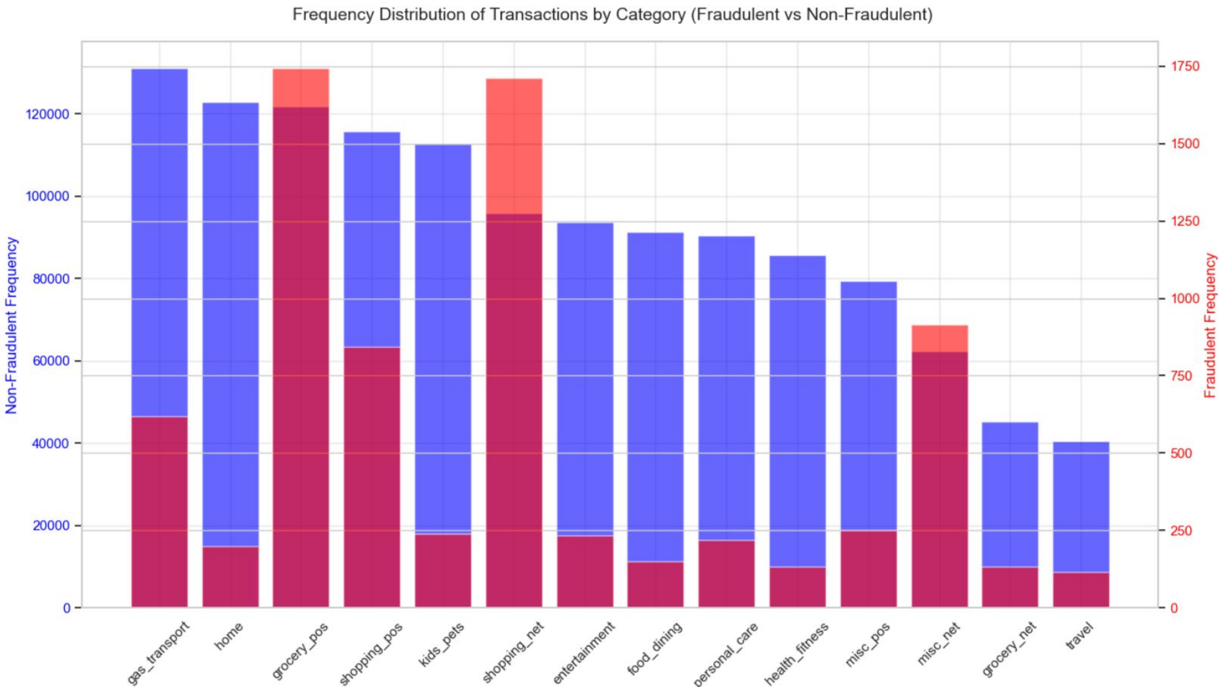
- Fraudulent transactions have two noticeable bulges in the distribution of transaction value that are not present for non-fraudulent transactions

- This unique distribution may be helpful for our predictive model, so we will develop a flag for transaction values within these two ranges



EDA: Merchant Categories

- Transaction counts by merchant category are distributed slightly differently in fraud vs non-fraud transactions
- Fraud transactions are significantly more frequent in five categories:
  - Grocery point of sale
  - Online shopping
  - Shopping point of sale
  - Gas/transport
  - Online misc
- We created a feature to flag transactions in these categories

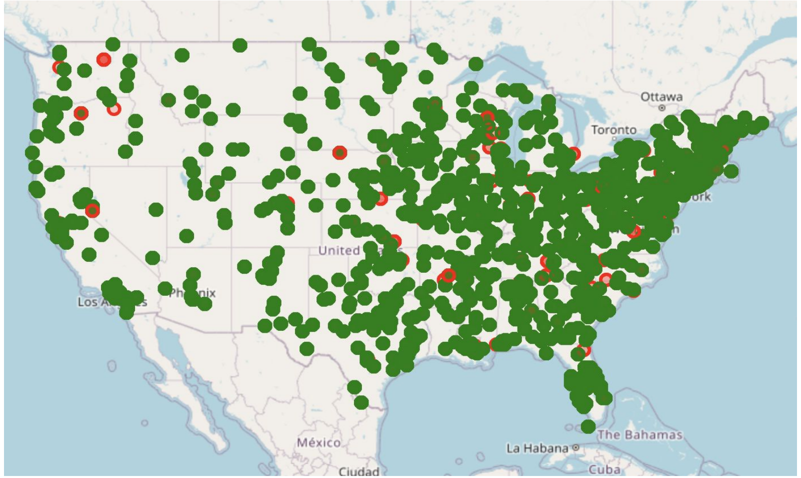


Feature Flag

- **Engineered features:**
  - `distance_km`: Calculated using geospatial coordinates.
  - `not_local`: Flag for transactions <50 km.
  - Distance categories: Local, Regional, Long Distance.
  - Additional Risk Encodings:
    - Transaction value risk levels (High/Medium/Low)
    - Transaction time risk levels (based on transaction hour)
    - Categorical risk flags for transactions with specific merchant types

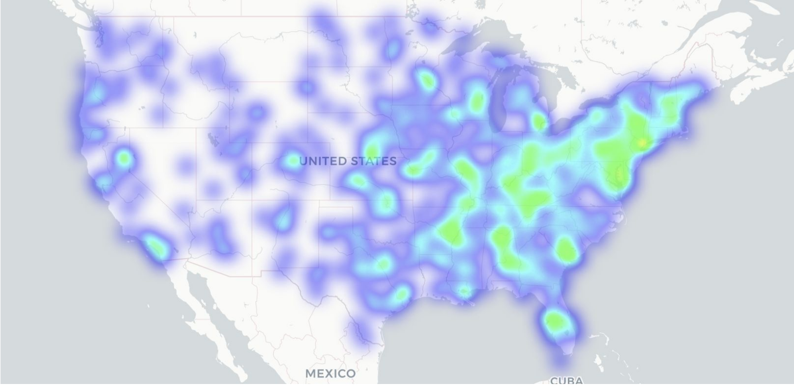
Feature Flag: Geospatial Map

- Maps built with folium
- Different parameters can be turned on/off with the code
- Geospatial clustering of fraudulent transactions
    - Hotspots
- No interesting trends for this dataset
    - Ex. Zoom in and most are on cities

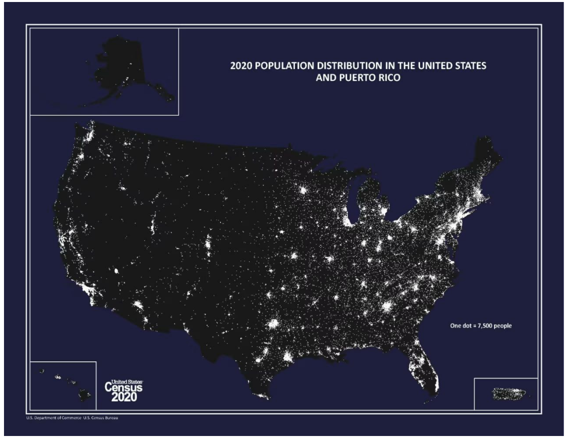


Feature Flag : Heat Map

- Higher frequency of fraud :
    - Higher Population
    - Higher frequency of transactions (more stores)



- Population Density map



https://www.census.gov/library/visualizations/2021/geo/population-distribution-2020.html

Data Cleaning

Datetime Feature Extraction

- Transformed the `trans_date_trans_time` column to datetime format
  - Extracted features `hour`, `day`, `month`, `year`, and weekday for temporal analysis
- Calculated the customer's age at the time of the transaction by combining `dob` + `trans_date_trans_time`.

Dropped Irrelevant Columns

- Unnecessary columns
  - Personal info
  - Redundant identifiers
  - Other irrelevant or unique fields

Data Cleaning

Encoded Categorical Variables

- Converted categorical features into numerical encodings for machine learning models:
  - Columns like `category`, `job`, `distance_category` were label encoded .

```
...  <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 1296675 entries, 0 to 1296674
     Data columns (total 24 columns):
      #   Column                 Non-Null Count    Dtype
     ---  ------                 --------------    -----
      0   Unnamed: 0             1296675 non-null  int64
      1   trans_date_trans_time  1296675 non-null  object
      2   cc_num                 1296675 non-null  int64
      3   merchant               1296675 non-null  object
      4   category               1296675 non-null  object
      5   amt                    1296675 non-null  float64
      6   first                  1296675 non-null  object
      7   last                   1296675 non-null  object
      8   gender                 1296675 non-null  object
      9   street                 1296675 non-null  object
      10  city                   1296675 non-null  object
      11  state                  1296675 non-null  object
      12  zip                    1296675 non-null  int64
      13  lat                    1296675 non-null  float64
      14  long                   1296675 non-null  float64
      15  city_pop               1296675 non-null  int64
      16  job                    1296675 non-null  object
      17  dob                    1296675 non-null  object
      18  trans_num              1296675 non-null  object
      19  unix_time              1296675 non-null  int64
     ...
      22  is_fraud               1296675 non-null  int64
      23  merch_zipcode          1100702 non-null  float64
     dtypes: float64(6), int64(6), object(12)
     memory usage: 237.4+ MB
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

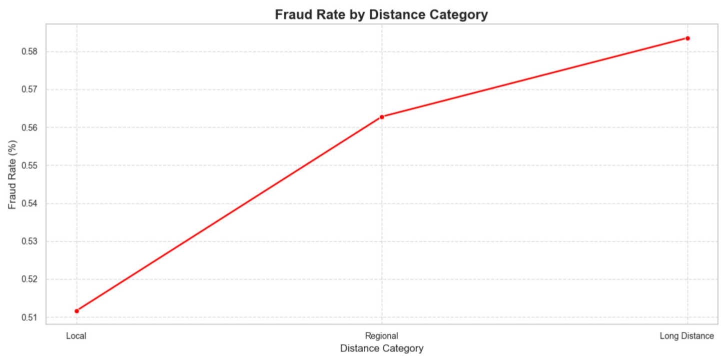
Data Correlations

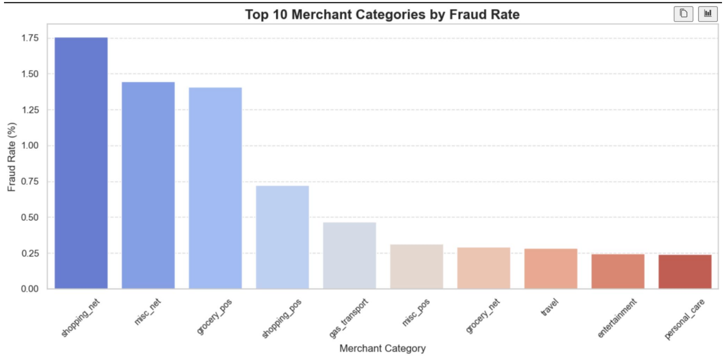
Geospatial Features:

- **`distance_km`**:
    - + correlation with `fraud - greater distance`
- **`not_local`**:
    - “Non-local" (> 50 km) = higher likelihood of fraud

Merchant Features:

- Categorized the fraud by seller





Data Correlations

Engineered Risk Indicators:

- **`transaction_value_risk_encoded`**:
  - Encoded transaction amounts into risk levels (High, Medium, Low).
  - Fraud was more common in **High Risk**:
    1. $250–$400 and $650–$1200.
- **`category_risk`**:
  - Flag for high risk transaction categories
  - Categories with higher fraud rates:
    1. Gas / online shopping / etc.

Transaction Timing Features:

- **`transaction_time_risk_encoded`**:
  - Fraud rates were higher during specific time windows:
  - Temporal correlations reflect patterns where fraud was exploited during periods of lower “vigilance” (e.g., nighttime)

Evaluation

To evaluate our results, we will hold-back a subset of the data prior to performing our analysis. This subset will serve as our sample to conduct testing on once we have our model developed.

If our model is working, it should be able to detect fraudulent transactions using customer data provided at the point of purchase.

We can compare model estimates to the binary flag that tells us whether or not the transaction was fraudulent, and tune our model as needed.

Model Performance

```
Random Forest:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95    386334
           1       0.94      0.96      0.95    387168

    accuracy                           0.95    773502
   macro avg       0.95      0.95      0.95    773502
weighted avg       0.95      0.95      0.95    773502

XGBoost:
              precision    recall  f1-score   support

           0       0.92      0.90      0.91    386334
           1       0.90      0.92      0.91    387168

    accuracy                           0.91    773502
   macro avg       0.91      0.91      0.91    773502
weighted avg       0.91      0.91      0.91    773502

Logistic Regression:
              precision    recall  f1-score   support

           0       0.80      0.96      0.87    386334
           1       0.95      0.76      0.84    387168
...
   macro avg       0.87      0.86      0.86    773502
weighted avg       0.87      0.86      0.86    773502

ROC-AUC Scores: RF=0.9868863973122182, XGB=0.9723194690982926, LogReg=0.8310847664811623
```

- **Random Forest – Best**
- **XGBoost – Solid Alternative**
- **Logistic Regression**

Accuracy and ROC-AUC:

- RF had highest accuracy (95%) and ROC-AUC score (0.9868)
- XGB 91% accuracy and 0.9723 ROC-AUC.
- LogReg 86% accuracy and 0.8310 ROC-AUC

Conclusion : Lessons Learned

- **Data Cleaning Complexity:**
  - Extensive data preprocessing was required to work with this dataset
  - Creating, parsing, and encoding categorical variables and temporal features is time consuming
  - There is a balance to find between trimming too many data points and reducing your workload

- **Handling Imbalanced Data**
  - Fraudulent transactions represented a small fraction of this dataset, so it would be interesting to find alternative datasets with a larger sample of fraudulent transactions to train with

Conclusion – Future Projects

Enhance Feature Engineering:

- Explore external data sources, such as customer profiles or merchant reliability scores.
- Incorporate dynamic variables like transaction history trends.

Real-Time Fraud Detection:

- Develop models optimized for real time prediction to enhance responsiveness and usage

Continuous Improvement:

- Regularly update models with new data to adapt to evolving fraud patterns / scam differences

Questions??