**Credit Card Fraud Detection**

Using metadata to detect fraudulent transactions

Brendan McDonald
McDonaBP@colorado.edu

Kevin Corboy
Kevin.Corboy@colorado.edu

## PROBLEM STATEMENT/MOTIVATION

The credit card industry is responsible for processing billions of dollars in transactions on a daily basis, which makes timely and reliable detection of fraud paramount. An entire industry supports the banking and credit card transaction market in fraud detection, using all variants of regression analysis, decision trees and neural networks, and all types of AI/ML analysis to describe and make decisions using data.

- Non cash payments increased at a rate of 9.5 percent per year since 2018 with payments value reaching $128.51 trillion in 2021 [1]
- The value of card payments rose 10% annually since 2018 with card payments value reaching $9.43 trillion in 2021 [1]
- In person card fraud decreased due to adoption of embedded chips in cards [2]
- Remote fraud increased, accounting for $4.57 billion in 2016 and growing since then [2]
- In 2016, the fraud rate for credit cards alone was 14.9 “basis points” meaning for every $10,000, $1.49 was fraudulent [2]

## SURVEY OF PRIOR LITERATURE

This dataset was found on Kaggle and there are a number of analysis notebook submissions from different developers available. Since the authors are unknown and the quality is variable, we are not referencing these analyses in our work. We have provided a review of some of the most popular Kaggle analyses that used this dataset. We also reviewed two additional studies related to credit card fraud analysis that rely on alternative datasets.

### Kaggle Analysis With This Dataset:

Mohamed Alabasy

- Alabasy’s work is primarily a descriptive analysis of the data
- They remove many of data categories prior to checking if they are significant predictors of fraud, including order and merchant location data, transaction date and time, and other potentially significant descriptors
- Their logistic regression results are not highly predictive of fraudulent transactions
- https://www.kaggle.com/code/mohamedalabasy/credit-card-transactions

Omar Wagih

- Wagih’s work is a classification model using three separate techniques;
  - Logistic regression
  - K-nearest neighbors (KNN) classification
  - Decision tree classification
- They retain most of the data in preprocessing, only dropping a few fields that are incomplete and not required for the analysis
- Like the Alabasy analysis, the performance of the classification models is highly biased towards non-fraudulent transactions, and did not have a single case of accurately detecting true fraudulent transactions
- :https://www.kaggle.com/code/owm4096/credit-card-fraud-eda-w-ml

Shivam Chaurasia

- Chaurasia’s work is similar to Alabasy’s in that they remove a large number of fields prior to investigating for correlation
- Like the Alabasy analysis, the performance of the classification models is highly biased towards non-fraudulent transactions, but did have some success in detecting fraudulent transactions using random forest classification techniques

- The geospatial aspect of Chaurasia's analysis was interesting, and we intend to review this data in a different way to see if there is a variability in fraud based on distance
- https://www.kaggle.com/code/shivjichaurasia/credit-card-fraud-detection-eda-prediction

**Other Studies on Card Fraud:**

"Enhancing Credit Card Fraud Detection: An Ensemble Machine Learning Approach"

- This paper proposes an ensemble machine learning model that combines various algorithms (SVM, Random Forest, KNN, etc.) to tackle challenges w/ data imbalance and concept drift in fraud detection. This model significantly improved accuracy, recall, and F1-scores when compared to traditional machine learning approaches
- https://www.mdpi.com/2504-2289/8/1/6

"A systematic review of literature on credit card cyber fraud detection using machine and deep learning"

- This paper aligns well on credit card fraud detection, with an emphasis on the importance of accurate and timely fraud detection within the credit card industry. It highlights how the industry heavily relies on advanced techniques such as regression analysis, decision trees, neural networks, and various AI/ML methods to analyze data and improve fraud detection systems. The paper's focus on machine learning approaches could provide valuable insights for our project, as it explores the methods and models already in use for detecting fraud.
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10280638/

**PROPOSED WORK**

1. Load data into a suitable coding environment
    a. The university furnished CSEL coding environment is well suited to a medium sized dataset like this one
2. Clean data to populate missing fields and remove observations that are not able to be completed
    a. Determine if we replace the missing merchant zip code data via geocoded lat/long lookup (all other fields are complete)
    b. Imputation: For small amounts of missing data
        i. Mean/median imputation could be used for numeric fields
        ii. Mode for categorical fields
        iii. Predictive Imputation: Machine learning models can predict missing values based on relationships between other variables
    c. Forward/Backward Fill:
        i. For time series data
        ii. Can be used to propagate previous/subsequent values
    d. Exclusion:
        i. If the missing data or anomalies are too significant, then remove
    e. Flagging Anomalies:
        i. To determine if further analyses indicate fraud
3. Perform data reduction on unnecessary fields to reduce analysis workload
    a. Review the credit card number column; they should all pass the Luhn algorithm as valid (otherwise they would have been rejected at the point of purchase), and can likely be removed from the dataset since they are randomly assigned by the card issuer
    b. We may not need merchant zip if we're going to use coordinates for geolocation instead
    c. Confirm that the unique transaction category is indeed unique, and the memory usage associated with the

unique transaction category vs an incrementing integer as a unique identifier instead
    d. Are there variations in correlation between spending categories, transaction values, geospatial variations, seasonal volatility, etc
4. Review results of correlation analysis and visualize data via plot to better detect patterns that aren't obvious
5. Test various correlations to implement ML model to detect fraudulent transactions
    a. Set precondition flags for specific cases; ie. if there tends to be a higher incidence of fraud when the merchant and the customer are separated by a far distance, flag that transaction as "LONG_DISTANCE" to notify the processor that this may be a fraudulent transaction
    b. If a specific number of preconditional flags are established, the transaction may be denied by the processor before being place
6. Evaluate model performance compared to test sample set
7. Tune model performance as needed

**DATA SET**

The dataset includes ~1.8M rows and 24 columns of detailed credit card transaction data. Each transaction includes purchase information about time and value, customer and merchant details (age, location, market segment, demographic data), and a binary value indicating whether the transaction is fraudulent or not. The dataset spans ~18 months, which also allows for analysis of seasonality and year-over-year spending trends.

The uncompressed CSV file is 337 MB, which is larger than the Github upload limit, but easy to ingest in most other environments. The CU-furnished CSEL environment accepts the uncompressed CSV as an upload, or you can pull the data from the Kaggle link below. [3]

**EVALUATION METHODS**

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.

Here are some metrics we intend to evaluate:

1. Accuracy: Evaluate how well (we would need to pick a scale) your model predicts both fraudulent transactions correctly. This could be misleading considering non fraud cases dominate most datasets (avgs above in the federal reserve numbers)
2. Precision: Calculate the ratio of true positives (correct fraud predictions) to all positive predictions (both true and false) – High precision would indicate flagged transactions are *truly* fraudulent i.e.not a false positive
3. Recall (Sensitivity): Calculate the ratio of true positives to **all** fraud cases (true positives + false negatives) – High recall means fewer false negatives.
4. F1 Score: Calculate the mean of precision and recall, providing a balance between these two metrics
5. ROC-AUC (Receiver Operating Characteristic - Area Under Curve): Evaluates a model's performance by plotting true positive rates against false positive rates – A high AUC score means the model distinguishes more easily between fraudulent and non-fraudulent transactions
    a. AUC score closer to 1 = better performance

**TOOLS**

Our primary tool for this analysis will be the Python programming language. Within Python, we will source a handful of useful libraries and packages to complete our analysis:

- Pandas: open source data manipulation and analysis library that makes processing large datasets quick and easy using Python
    - https://pandas.pydata.org/
- Numpy: another opens source data manipulation and analysis library with Python
    - *https://numpy.org/doc/stable/index.html*
- SciKit Learn: *"Simple and efficient tools for predictive data analysis built on the Numpy package"*
    - https://scikit-learn.org/stable/
- MatPlotLib and Seaborn: Both of the libraries are useful for data visualization purposes
    - https://matplotlib.org/
    - https://seaborn.pydata.org/
- We will manage our analysis within an iPython Notebook so that we can easily integrate markdown text and explain our work as we go

**MILESTONES**

IN PROGRESS: Data ingested and summarized

1. Ingest of the dataset is as simple as reading the raw CSV file into a Pandas dataframe. The raw CSV file is too large to upload to Gitbhub in a single file, but the CSEL environment has worked well for data processing
2. Summary:
    a. 24 columns of data; merchant zip code is missing ~195K observations, otherwise all 1,296,675 rows of data are non-null in all columns (~99.6% "complete" data)
    b. Column data types are a mix of integer, string, and float values
    c. When read into a Pandas dataframe, the dataset occupies ~1.16GB of memory, which will serve as our starting point for data reduction measurement

IN PROGRESS: Data cleaning and reconciliation (including details about the justification for removal of data)

3. Data cleaning
    a. Credit card numbers: since credit card numbers are random and issued by the bank/merchant, it is unlikely that the number itself is useful in determining fraud. We did a quick check to confirm that all the card numbers were indeed valid card numbers based on the Luhn algorithm, but as expected, there were no errors found. Removing the credit card number column saved ~10MB of memory.
    b. Transaction Num: since the transaction Num column is a unique hexadecimal string, it is unlikely that the string value is useful for fraud detection. The primary use is to serve as a unique identifier for each transaction. However, using a hexadecimal string is unnecessary when we can instead use an integer value to identify each transaction. Additionally, our dataset already contains an incrementing integer value for each transaction (the column originally titled "Unnamed: 0" in the raw data). Removing the transaction number column frees up ~110 MB of memory.
    c. Other columns like customer first, last, and street/address data were removed as they don't add anything unique/interesting that the other latitude longitude data doesn't provide. Removal of these columns resulted in a memory savings of ~254 MB.
    d. We will review correlations of other customer data like city pop, job, and DOB to determine if they are safe to remove
4. Data reconciliation/completion
    a. Merchant zip code: this is the only column that is not completely populated in the dataset. We will

review initial correlations before making a determination on whether to fill the missing values, or remove the column entirely. We have a proof of concept to perform a geocode look-up of the missing zip code data using the latitude and longitude data that is available. A zip code can be useful for an approximation of distance between two points, but it is imperfect, and we can get a similar approximation using a great circle distance estimate between both sets of coordinates

b. Imputed missing values in merch_zipcode using the most frequent value strategy fills ~150k empty entries
c. Confirmed that no other columns had significant missing data

IN PROGRESS; Initial correlations investigated, training methods selected

5. Functions created :
    a. Created a feature by calculating the distance between customer and merchant locations/zips + set a threshold (50 km) to flag non local transactions as potentially fraudulent:

*#function to add a not_local flag to determine if further distance causes more fraud --ie Fraud while Traveling*

*def calculate_distance(row):*

*return np.sqrt((row['lat'] - row['merch_lat'])**2 + (row['long'] - row['merch_long'])**2)*

*df['not_local'] = df.apply(calculate_distance, axis=1).apply(lambda x: 1 if x > 50 else 0)*

    b. Determines if fraud happens more frequently when people are away from home (ie traveling or maybe in places that are not typical)
6. Visualized transaction locations using a scatter geoplot for geospatial insights
    a. Random sample used for rendering
    b. Can see if there are hotspots of fraud
7. Data exploration still pending:
    a. Review how fraudulent transactions are distributed across 24 hours; if there is a significant difference in one part of the day versus others, that can be used as a precondition for our model
    b. Review fraudulent transaction values to determine if a particular price range has a higher probability for fraud than others
    c. Investigate customer fraud data to determine if there any customers with a significantly higher rate of fraudulent transactions than others
    d. Exploratory correlation analysis and data visualization to detect interesting patterns and trends

NOT STARTED: Model trained and tested

8. Determining a hold-back method and data testing size will be an important aspect in handling the significant class imbalances inherent in many fraud datasets. There are a few techniques that we are reviewing to help combat this problem:
    a. Class weight adjustment is a method to increase the weights in minority class observations
    b. Evaluation metrics that may work for balanced datasets (like five number summary statistics) often are less useful with imbalanced datasets, so we will observe different metrics like the ROC Curve and F1 scores that can be used to assess datasets with significant size differences in classes
9. Data addition and augmentation is another area that we'd like to explore:
    a. There are some basic customer demographic data points included within the dataset that can be augmented with additional demographics information that may be useful in flagging transactions.

b. There is also a subset of data that spans more than 12 months, which could prove useful for year-over-year trend analysis that is applicable for fraud detection purposes.
c. Customer segmentation and predictive modeling may support flagging mechanism for future buyer behaviors

NOT STARTED: Verification of results

10. Establishing a data holdout set will be a primary method for verification of our model. By holding back a portion of the dataset, we can use that subset to evaluate our models performance in a test scenario. We can also make this subset selection random to conduct repeated tests on different subsets of data for further verification (Monte Carlo validation strategy). Stratified sampling may be useful in this scenario, but we haven't tested it yet.

**REFERENCES**


[1] *Federal Reserve Payments Study (FRPS)* (no date) *Federal Reserve Board - Federal Reserve Payments Study (FRPS)*. Available at: https://www.federalreserve.gov/paymentsystems/fr-payments-study.htm (Accessed: 15 October 2024)

[2] *Changes in U.S. payments fraud from 2012 to 2016*. Available at: https://www.federalreserve.gov/publications/files/changes-in-us-payments-fraud-from-2012-to-2016-20181016.pdf (Accessed: 16 October 2024).

[3] *Credit Card Transactions Dataset*, *Kaggle*. Choksi, P. (2024) Available at: https://www.kaggle.com/datasets/priyamchoksi/credit-card-transactions-dataset?resource=download (Accessed: 15 October 2024).