##### **Executive summary**

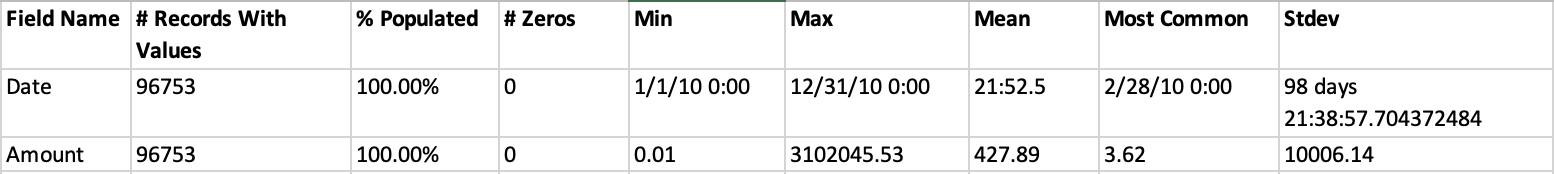
Due to the rise of fraudulent credit card transactions, this project aims to detect possible fraudulent transactions using past data to train supervised machine learning models, compare the performances of different models, and utilize the model with the best performance to detect credit card transaction fraud. This is done with FDR 3% to limit the false discoveries to 3% of the positive predictions. The final maximum estimated savings from this project are 21,480,000 USD.

##### **Description of the data**

The dataset is Credit Card Transaction Data, which contains Personal Identifying Information of credit cards. The dataset was provided in class and consists of 10 fields and 96,753 records.

Summary Tables:

Numeric Fields Table -

****

Categorical Fields Table -

**Table

Description automatically generated**

Where:

% Populated : percentage of records present

# Zeros : Number of zeros

# Unique Values : Number of unique values

Data Types of the Fields -

**Text, table

Description automatically generated**

##### **Data cleaning**

We selected only ‘P’ Transaction Type Records and removed an outlier where the amount was greater than or equal to 3000000. The number of missing values for each field was computed as seen below.

**A picture containing text, screenshot, font, algebra

Description automatically generated**

For Merchnum, a dictionary was created that mapped the existing non-NaN merchant description (the key) to the non-NaN merchant number (the value) and was used to impute the records that had the same merchant description. Additionally, some records that had merchant description as ‘RETAIL CREDIT ADJUSTMENT’ were masked as unknown since these are adjustment transactions. Finally, the maximum value for Merchnum was found and was incremented to form new Merchnums for the records that did not still have a Merchnum.

For Merch state, a dictionary was created that mapped the non-NaN zip(the key) to the non-NaN merchant state(the value) and in the same dictionary, non-NaN zipcodes that didn’t have a non-NaN merchant state were manually assigned the state the zip code belonged to. Additional dictionaries were created where one dictionary mapped the non-NaN merchant number (the key) to the non-NaN merchant state(the value) and another that mapped the non-NaN merchant description(the key) to the non-NaN merchant state(the value). The missing values in Merch state were then imputed by the mappings in these 3 dictionaries. Additionally, some records that had merchant description as ‘RETAIL CREDIT ADJUSTMENT’ were masked as unknown since these are adjustment transactions. Lastly, records with non-US states were labeled as 'foreign' as they might be useful since foreign transactions could be fraudulent. The remaining missing values were imputed as ‘unknown’.

For Merch zip, dictionaries were created that mapped the non-NaN merchant number (the key) to the non-NaN merchant zip(the value) and another that mapped the non-NaN merchant description(the key) to the non-NaN merchant zip(the value). The missing values in Merch zip were then imputed by the mappings in these 2 dictionaries. Some records that had merchant description as ‘RETAIL CREDIT ADJUSTMENT’ were masked as unknown since these are adjustment transactions. The remaining missing values were imputed as ‘unknown’.

Lastly, the datatypes of the fields were changed to more appropriate data types.

##### **Variable creation**

Fraud can be detected by looking for unusual things about a certain event and this consists of field values that would generally be considered outside of norms. This also includes odd interrelationships of field values. By keeping this in mind, these variables are built looking at the event by itself and we look for unusual connections /links between events. For example: a common contact (phone, address, email, etc), a common checking/savings account, a common device behind many events. We also consider the event along with many past events for variable building.

Additionally, we make use of Benford’s Law to create more variables. Benford’s Law states the nonintuitive fact that the first digit of many measurements isnotuniformly distributed, and the first digit “1” appears about 30% of the time. The digit “9” only appears less than 5% of the time. Fraudsters do not know Benford’s law, so they usually make up uniform distribution numbers. However, this is useful for fraud detection when a potential fraudster is making up a large number of transactions and one can reasonably examine the resulting distribution. It is not applicable for situations when numbers are generated by many people, most of whom are not committing fraud.

Possible types of fraudulent credit card transactions can include:

* Tax fraud – a fraudulent person makes up many numbers to support their own return (like purchase amounts)
* Tax fraud – a preparer makes up many numbers across many returns
* Merchant fraud – a merchant makes up many fraudulent purchase amounts
* Accounting fraud - a fraudulent financial person makes up many amounts, for example, costs, transactions…
* Healthcare fraud – a fraudulent person makes up a bunch of claim amounts

As explained earlier some of the 1500+ new variables were created using the logic as per the below picture:

**A picture containing text, screenshot, font, document

Description automatically generated**

Variable Table: These variables can be categorized as -

**A screenshot of a computer

Description automatically generated with medium confidence**

##### **Feature selection**

Due to the curse of dimensionality, data becomes sparse very quickly and all data points become outliers. Consequently, nonlinear models struggle in high dimensions, necessitating reduction in dimensions as much as possible before modeling. This can be done by including variables with substantial information and minimizing the number of variables. This allows the exploration/consideration of several candidate variables, essentially without limit and it results in a sorted list, so we know which variables to add/remove in our model explorations. It also allows much faster nonlinear model runs to optimize the model architecture and hyperparameters.

With the above in mind, a univariate KS filter was run first, then a wrapper to reduce the number of variables to around 20. Since the transactions can be flagged as either fraud or not fraud, this is a binary classification problem, and the univariate KS filter is best suited for this. The variables with the top KS filter scores were then fed to the LGBM wrapper with forward selection to identify the best suitable variables for model input.

The univariate KS (Kolmogorov-Smirnov) filter is a technique that identifies suspicious transactions by measuring the discrepancy between the distributions of legitimate and fraudulent transactions for a single variable (univariate analysis). It evaluates the differences in the cumulative distributions of a specific variable, such as transaction amount or time, between the two classes: genuine transactions and fraudulent transactions. Thereby it provides a simple yet effective way to identify potential anomalies in transaction variables that may indicate fraudulent activity and it is a useful initial screening method that helps prioritize relevant variables.

The wrapper used is stepwise selection, also referred to as stepwise variable selection, which is a method employed in statistical modeling and feature selection to determine the most important variables for inclusion in a predictive model. It follows a systematic approach of iteratively adding or removing variables based on their statistical significance or impact on the model's performance.

In this analysis, we investigate the two steps of the stepwise selection process: forward selection and backward elimination. We use the LightGBM and Random Forest Classifier algorithms with different hyperparameters. Performance plots are generated to identify the optimal wrapper along with the best parameters that yield the highest model performance. The final choice is Forward Selection with the LGBMClassifier since the performance graph for stepwise selection is relatively more stable.

The optimal parameters from explorations were:

num\_filter (is a fraction of the generated variables) = 200,

num\_wrapper (expected number of output variables for the model) = 23,

n\_estimators (number of boosting iterations or decision trees to be built in the LightGBM model) = 50, num\_leaves (maximum number of leaves or terminal nodes in each decision tree of the LightGBM model) = 5

Chart, line chart

Description automatically generated

Table

Description automatically generated

##### **Preliminary model explorations**

Initially, we utilize regularization to reduce the complexity of the models and prevent them from fitting noise or irrelevant patterns. Regularization is an ML technique used to prevent overfitting and improve the generalization ability of a model. By adding regularization terms, the model is incentivized to find a balance between minimizing the error on the training data and keeping the model parameters small.

Then post regularization, all the ML models are forced to overfit. In general, overfitting occurs when the training performance is very good and much better than testing performance, while OOT does poorly. As the model becomes too complex, it will start to fit to the noise in the training data, resulting in reduced performance on testing data. We overfit the models to optimize their architecture and hyperparameters and identify the optimal hyperparameters. Below are some models that were explored along with their performance at FDR 3%.

Note: The best models for each ML model have been highlighted in green.

**Logistic Regression**

Table

Description automatically generated

Box Plot of the best model:

Chart

Description automatically generated

**Single Decision Tree**

Table

Description automatically generated

Box Plot of the best model:

Chart, bar chart

Description automatically generated

**Random Forest**

Table

Description automatically generated

Box Plot of the best model:

Chart

Description automatically generated

**Neural Network**

Table

Description automatically generated

Box Plot of the best model:

Chart, box and whisker chart

Description automatically generated

**LGBM**

Table

Description automatically generated

Box Plot of the best model:

Chart

Description automatically generated

**LGBM with SMOTE**

Table

Description automatically generated

Box Plot of the best model:

A picture containing screenshot, text, line, number

Description automatically generated

**Gradient Boosting Classifier**

Table

Description automatically generated

Box Plot of the best model:

Chart, box and whisker chart

Description automatically generated

**CatBoost**

Table

Description automatically generated

Box Plot of the best model:

Chart

Description automatically generated

**XGBoost**

Table

Description automatically generated

Box Plot of the best model:

Chart

Description automatically generated

##### **Final Model**

The final model chosen is a Neural Network with 3 hidden layers, all having 10 neurons each. A neural network is a computational model inspired by the human brain that is widely used in machine learning, particularly in solving complex problems like credit card transaction fraud detection. It consists of interconnected artificial neurons organized in layers, which can learn and extract meaningful features from raw data. By adjusting the weights between neurons through a process called backpropagation, neural networks can automatically optimize their performance. In the context of credit card fraud, neural networks can analyze transaction data, learn patterns indicative of fraudulent behavior, and make accurate predictions to identify and prevent fraudulent transactions.

Final Model Performance:  
A table of numbers

Description automatically generated with low confidence

Training:

A screenshot of a spreadsheet

Description automatically generated with medium confidence

Test:

A screenshot of a spreadsheet

Description automatically generated with medium confidence

OOT:

A picture containing text, number, screenshot, receipt

Description automatically generated

##### **Financial curves and recommended cutoff**

Plot of the 3 financial curves:

A picture containing text, line, plot, diagram

Description automatically generated

Recommendation for cutoff location: Since the we want to identify as many frauds as possible, but also get good overall savings, a score cutoff at 3% is recommended. **The black line in the graph denotes the location of the cutoff.** This cutoff must be as far to left as possible but away from the sharp increase as seen in the graph above, so that it is still close to the highest savings.

##### **Summary**

Due to the rise of digital technology and online shopping there are more opportunities for fraudsters to engage in different types of credit card fraud. These include:

1. Card Skimming: Fraudsters use devices to illegally obtain credit card information from ATMs, gas pumps, or other payment terminals.
2. Phishing and Online Scams: Fraudsters deceive individuals through deceptive emails, fake websites, or phone calls to trick them into providing their credit card details.
3. Data Breaches: Cybercriminals target organizations to steal significant volumes of credit card data, which they can later use for fraudulent purposes.
4. Card-Not-Present Fraud: Unauthorized transactions occur when the physical credit card is not physically present, such as in online or over-the-phone transactions.
5. Identity Theft: Fraudsters gain access to personal information of individuals and use it to open fraudulent credit card accounts or make unauthorized transactions.

With these issues in mind, this project aims to detect possible fraudulent transactions using past data to train supervised machine learning models, compare their performance, and utilize the model with the best performance to detect credit card transaction fraud. New variables were created using existing dataset fields and the best variables for the models were chosen as per the top KS filter scores and wrapper results.

The optimal count of variables for model input was identified from the plot of Stepwise Selection wrapper where performance was plotted against increasing number of variables, and the saturation point i.e where the graph plateaus is chosen as the optimal variable count. Stepwise selection, also known as stepwise variable selection, is a technique used in statistical modeling and feature selection to identify the most relevant variables for inclusion in a predictive model. It is a systematic process that iteratively adds or removes variables based on their statistical significance or contribution to the model's performance. In this case, we are using the LGBM Classifier with Forward Selection.

Regularization is utilized to prevent overfitting and improve the generalization ability of a model, post which, the models are forced to overfit i.e., when a model becomes too closely tailored to the training data, to the extent that it performs poorly on new, unseen data. In other words, the model learns the noise and random fluctuations in the training data rather than capturing the underlying patterns that generalize well to unseen data. This is done to identify the best hyperparameters for the models. A neural network with 3 hidden layers and 10 neurons in all layers is found to the optimal model which is why it was chosen and then it was trained and tested on the dataset. The average performance values of the model for training data were 0.77, for test data it was 0.72 and it had oot 0.37, with FDR 3%. The maximum estimated savings were 21,480,000 USD.

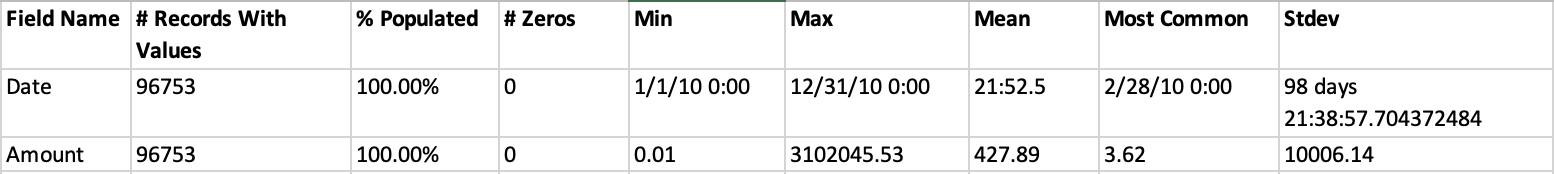
##### **Appendix: Data Quality Report**

**1. Data Description**

The dataset is **Credit Card Transaction Data**, which contains **Personal Identifying Information** of credit cards. The dataset was provided in class and consists of **10 fields** and **96,753 records**.

**2. Summary Tables**

**Numeric Fields Table**

****

**Categorical Fields Table**

**Table

Description automatically generated**

Where:

% Populated : percentage of records present

# Zeros : Number of zeros

# Unique Values : Number of unique values

**Data Types of the Fields**

**Text, table

Description automatically generated**

**3. Visualization of Each Field**

1. **Field Name: Recnum**

Description: Ordinal unique positive integer for each application record, from 1 to 96753.

1. **Field Name: Cardnum**

Description: Unique positive integer for credit card numbers, that has 1645 unique values in the dataset. The first number is known as the Major Industry identifier (MII) and it reflects the type of company a card is affiliated with. This could be an airline, a banking institution, or a retailer. It also indicates which major card network the card belongs to:

3 - American Express,

4 – Visa,

5 – Mastercard,

6 - Discover

Chart, bar chart

Description automatically generated

1. **Field Name: Merchnum**

Description: Unique positive integer for Merchant Identification number assigned to merchants by their acquiring bank or the payment processor. This field consists of 13091 unique values.

**Chart, bar chart

Description automatically generated**

1. **Field Name: Date**

Description: Refers to the credit card transaction date. The first histogram reflects the top 15 dates when most of the transactions occurred. The second histogram reflects the bottom 15 dates with the least number of transactions.

The line plots reflect the number of credit card transactions on a daily, weekly and monthly basis for the year 2010.

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**A picture containing text, screenshot, plot, line

Description automatically generated**

**A picture containing line, plot, text, diagram

Description automatically generated**

**A picture containing line, plot, diagram

Description automatically generated**

1. **Field Name: Merch description**

Description: The text field is for the different merchant descriptions and has 13126 unique values.

**Chart, bar chart

Description automatically generated**

1. **Field Name: Merch state**

Description: The text field is for the different states in USA and has 227 unique values.

Chart, bar chart

Description automatically generated

1. **Field Name: Merch zip**

Description: The float field reflects the different zip codes in USA and has 4567 unique values.

Chart, bar chart

Description automatically generated

1. **Field Name: Transtype**

Description: The field reflects the different credit card transaction types and the dataset provided has 4 such unique values for this.

**Table

Description automatically generated**

1. **Field Name: Fraud**

Description: Fraud identification label. Fraud = 0 (Not fraudulent), Fraud =1 (Fraud identified). The total count of Fraud = 0 is 95,694. The total count of Fraud = 1 is 1059.

**Table

Description automatically generated**

1. **Field Name: Amount**

Description: Refers to the credit card transaction amount in USD. The first plot reflects the occurrences of the amounts grouped as per the x axis labels. The second table reflects the descriptive stats of the field.

**Chart, bar chart

Description automatically generated**

A picture containing text, screenshot, font, number

Description automatically generated