Project 2

Finding Fraud Faster

Bryce Drynan

# Executive Summary

At Dragon’s Hoard Credit Union in The Lonely Mountain in South Erabor we pride ourselves in having the most protection against fraud, while also helping those who need money. In order to provide even better service to our clientele we are developing a model to predict fraudulent accounts. It would be a shame for all Middle Earth if they could not use our fantastic services. I will use the data provided to me by management to assess current transactions and provide predictions on data using that model that are yet to be confirmed as fraudulent or legit.

## Analysis

First, we will conduct data cleaning and EDA on the data. After that is finished, I will create transforms on the data, so it is passable to machine learning models with scalers and one hot encoding. Finally, I will run a baseline Logistic Regression, baseline random forest, tuned random forest, baseline XGBoost and tuned XGBoost. From this process it was determined that a tuned XGBoost model is best.

## Recommendations

After performing my analysis there are a few things that I could recommend. The first piece of advice is that billing zip and email domain are too specific and too high cardinality to include in our model. The result would be something that focuses too much on statistics that do not capture a large amount of the data resulting in models that are too specific and not general. My next recommendation is to implement a warning system. People often do not commit crimes if they simply know they are getting watched. A notification on fraudulent accounts/purchases could curb a lot of those issues and help handle a potential misclassification of a legitimate account. Finally, I would recommend using my tuned XGBoost model to make predictions since it is better than the blanket prediction of 95% legit and 5% fraud.

## METHODOLOGY

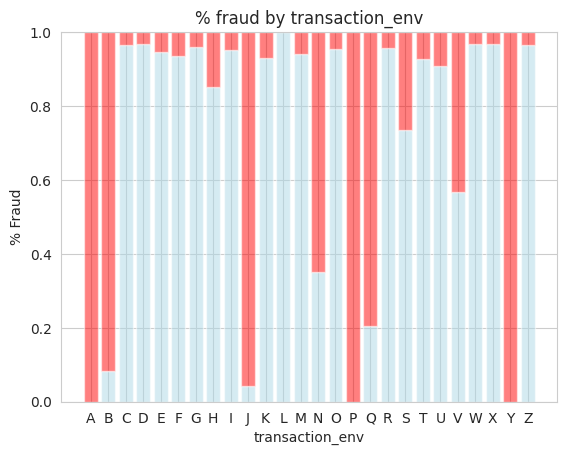
**Data Exploration and Preprocessing**

1. **Exploratory Data Analysis (EDA) & Feature Screening**:

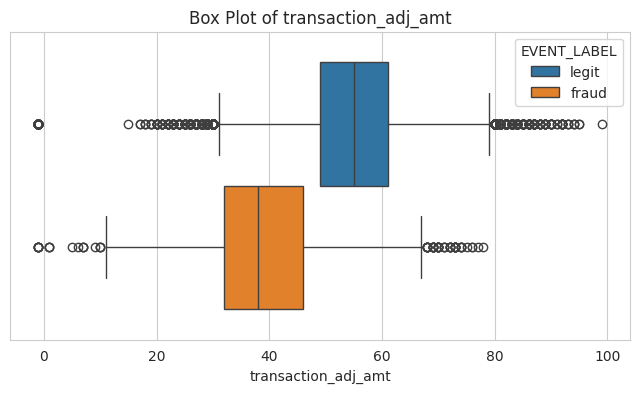
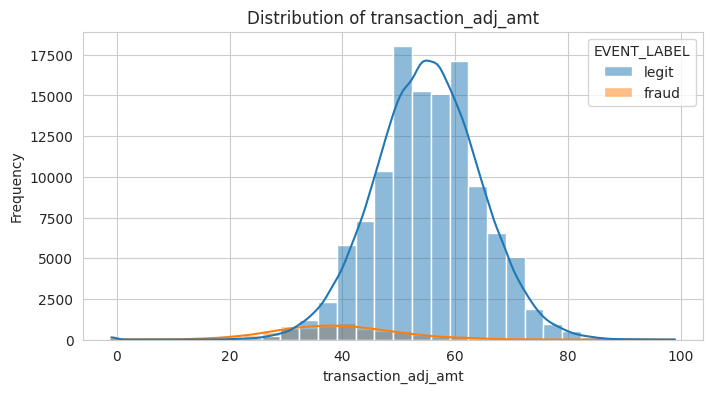
A close up of a number

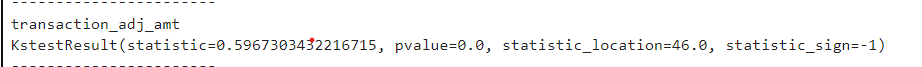
Description automatically generated

The data contains roughly 95% legit and 5% fraud. This is an important baseline for us when modeling. If we were to blanket assign based on this ratio, we would be right roughly 95% of the time that we have a fraudulent account. All of our models need to beat this benchmark to be considered viable.



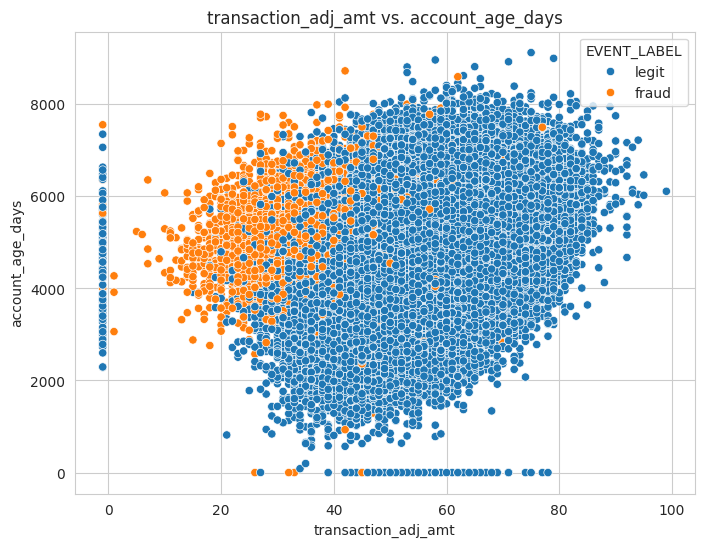
All of my categorical variables were passed through these 100% stacked bar charts for the purpose of visualizing the ratio of fraud by each option within the category that I could have. From these charts we are able to lightly infer some of the predictive power of the categories and eliminate columns from our model accordingly. For instance, the tranaction\_intitiate column saw no variance in fraudulent percentage across its classes so we will not be using that column. Whereas the above plot is useable, we can see there are certain types of environments that seem to produce fraudulent accounts more than others.

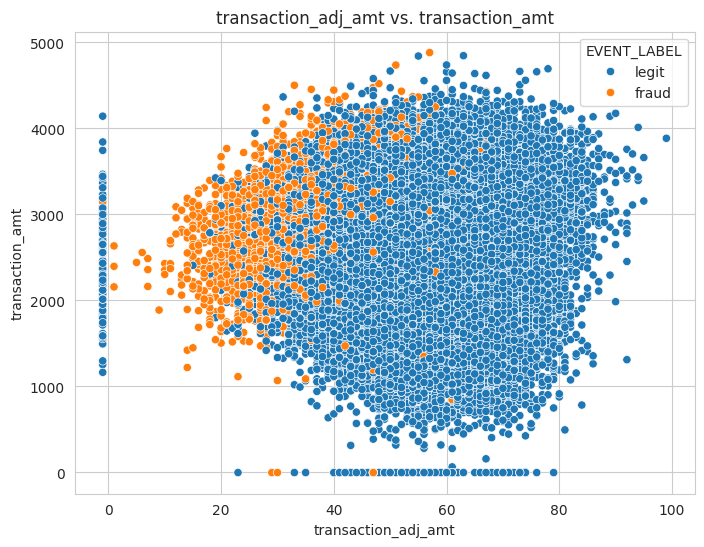




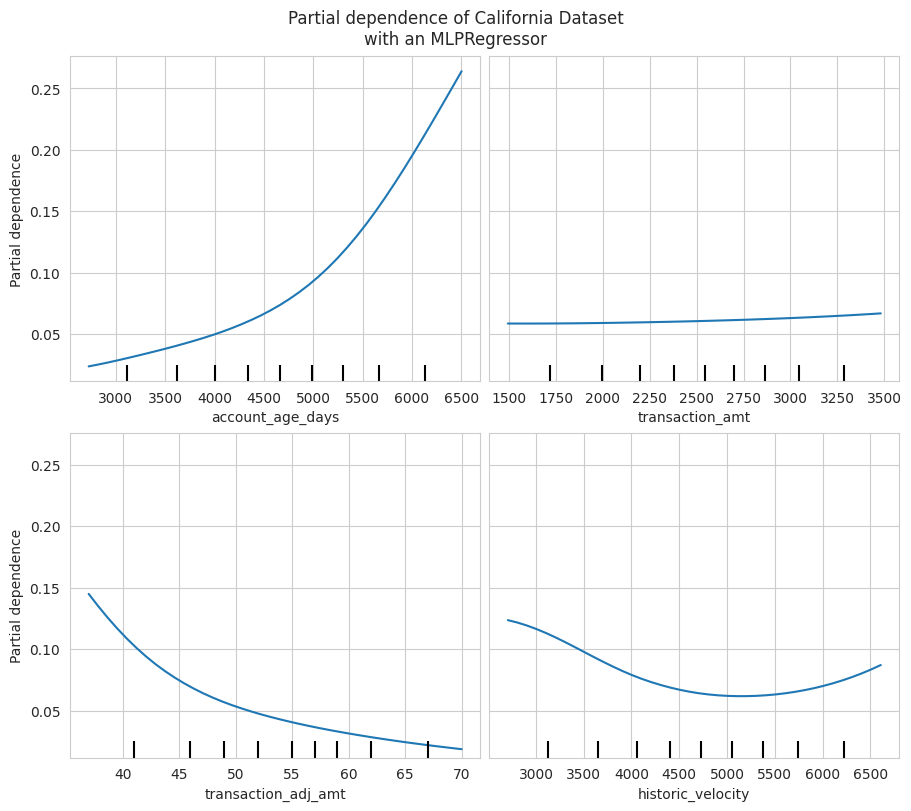
For the numeric variables I create boxplots as well as distributions of accounts flagged as fraudulent vs legit. The boxplots help us identify outliers, which we have a healthy amount of. Additionally, we can see that visually our fraudulent data and legit data seem to come from two different distributions which could indicate higher predictive power. To confirm this a 2 way KS test was computed. For 3 out of 4 numeric variables, its fraudulent and legit data come from two different distributions. Those were transaction\_adj\_amt, transaction\_amt, and account\_age\_days.

A diagram of a diagram

Description automatically generated with medium confidence



Scatter plots were also computed across all variables to check for possible interactions. We can see some clear definitions above when transaction\_adj\_amt is plotted against the other three variables suggesting that the lower adjusted amount that is transferred it has an extremely high probability of being fraudulent.



I was also interested in developing Partial Dependence Plots for numeric variables. The ones above were the only ones with non-flat lines. They were developed from an MLP model that was run over the data. We can see some of the behavior explained from the scatter plots made above as well. Especially with high account age and low transaction\_adj\_amt. An interesting return is the historic velocity of the accounts. It’s parabolic shape most likely comes from the density of accounts in the 5000s. We can see from the scatter plots above the density of fraudulent accounts decreases between 4000 and 5500.

1. **Data Preprocessing**:

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Transformer | Imputer Type | Scaler Type |
| Historic Velocity | Numerical | Median | Standard Scaler |
| Account Age Days | Numerical | Median | Standard Scaler |
| Transaction Amt | Numerical | Median | Standard Scaler |
| Transactions Amt Adj | Numerical | Median | Standard Scaler |
| Transaction Type | Categorical | Most\_frequent (Mode) | One Hot Encoder |
| CVV | Categorical | Most\_frequent (Mode) | One Hot Encoder |
| Currency | Categorical | Most\_frequent (Mode) | One Hot Encoder |
| Signature Image | Categorical | Most\_frequent (Mode) | One Hot Encoder |
| Transaction Environment | Categorical | Most\_frequent (Mode) | One Hot Encoder |
| Billing State | Categorical | Most\_frequent (Mode) | One Hot Encoder |

No columns saw over a 10% null rate which was promising. Since there was skew in the data I decided to use the median to impute the numeric, while using the mode to impute the categorical. Additionally, a Standard Scalar was used to normalize the numeric data to make sure that everyone was on the same scale. This way our model will not be affected by the different dimensionalities. One hot encoder was used to transform my categoricals to that they could be numeric and pass through the model.

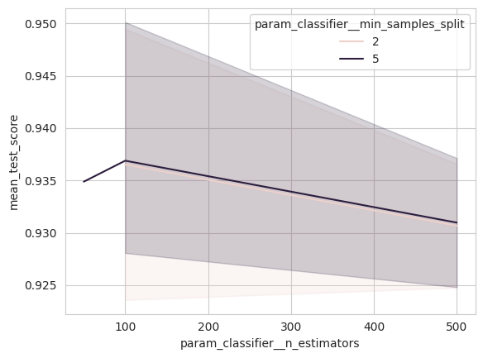
**Model Development**

1. **Model Training**:

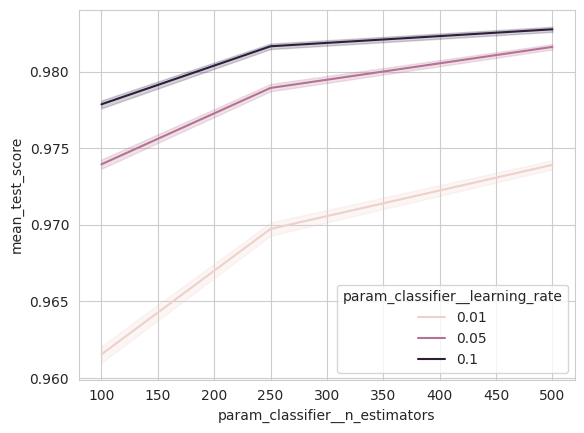
We begin with a Logistic Regression model to use as a baseline. This is one of the most simplistic classifiers so it should function as a good starting point. From there we will build a Random Forest Model. This model’s main function is to give us more insight into the data with feature importance and create a model that understands more of the nuances than a LogReg. After that we will build a Gradient Boosted model which will help us to understand further underlying complexities in the data that a random forest cannot. For this I used the XGBoost package.

1. **Parameter Tuning**:

Since the logistic regression was a baseline no tuning was applied to the model. For the random forest a baseline was run with 50 trees at no max length. This model was then tuned on a random search model where I adjusted min leaf samples, max tree size and n\_estimators. My best model have 5 min samples per leaf, no max depth, and 100 estimators. This is visualized with the following figure:



Next I ran a baseline XGBoost model baseline with a learning rate of .1 and 100 estimators. From there the model was tuned with a grid search algorithm. The best model had 500 estimators, and a learning rate of .1. We can see the model progression below. Additionally, all tuned models were cross validated three times to further improve the generalization and predictability of the model.



1. **Feature Selection**:

The selection of my features was performed iteratively. Starting with some statistical analysis above with the KS testing and % fraud by class for each categorical column. Additionally, I performed a Permutation Importance and PDP charts that revealed more about what columns would be worthwhile using.

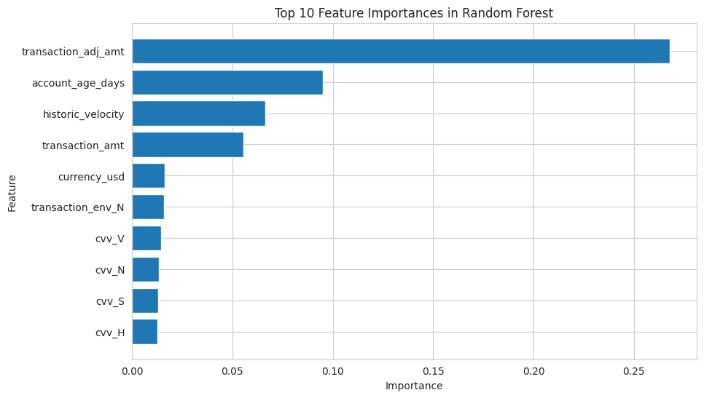
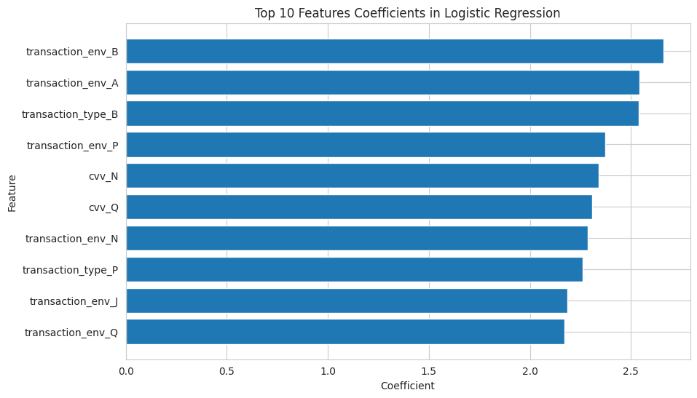
**Model Evaluation**

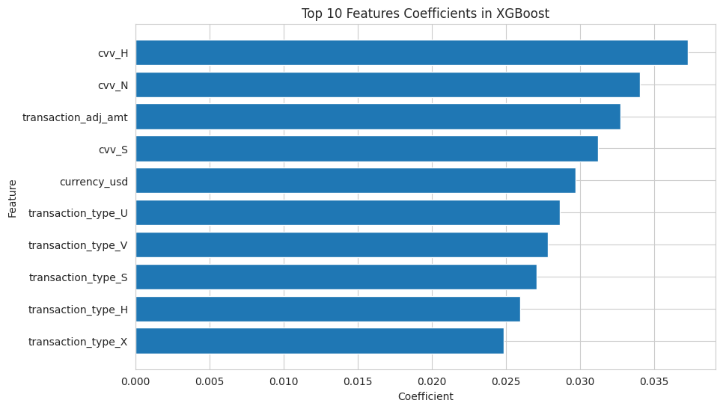
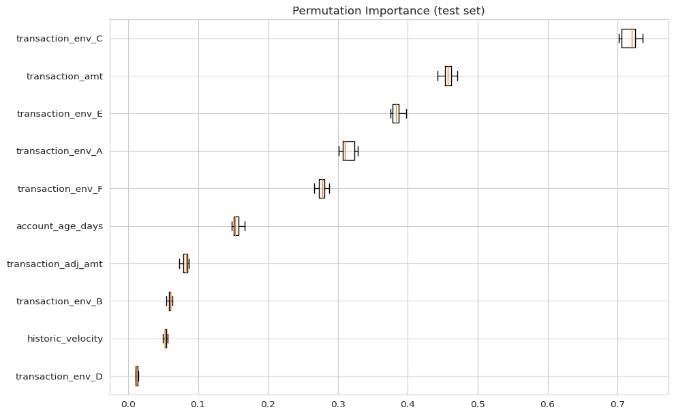
1. **Performance Metrics**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **param\_classifier\_\_max\_depth** | **param\_classifier\_\_n\_estimators** | **param\_classifier\_\_min\_samples\_split** | **mean\_test\_score** | **rank\_test\_score** |
| **8** | None | 100 | 5 | 0.95012 | 1 |
| **0** | None | 100 | 2 | 0.94946 | 2 |
| **3** | 10 | 500 | 5 | 0.93714 | 3 |
| **1** | 10 | 100 | 5 | 0.93687 | 4 |
| **4** | 10 | 500 | 2 | 0.93659 | 5 |
| **9** | 10 | 50 | 5 | 0.93488 | 6 |
| **2** | 5 | 500 | 5 | 0.92481 | 7 |
| **6** | 5 | 500 | 2 | 0.92475 | 8 |
| **5** | 5 | 100 | 5 | 0.92366 | 9 |
| **7** | 5 | 100 | 2 | 0.92358 | 10 |

The table above is an example of how I selected the best model. The models were assessed based on accuracy. The top model having the best accuracy is the one that appears as my tuned random forest model. This model performed much better on my test set than my train set which I found to be very interesting.

1. **Feature Importance Analysis**:



Combining the top 10 most impactful features from my baseline and tuned models with my permutation importance there are a few interesting pieces to note. CVV seems to appear in all my models, but not in my permutation test which is interesting. Unsurprisingly we see that transaction\_adj\_amt is a top player in every model except the logistic regression which begs the question of high dimensionality having an impact on its prediction power with how simple of a model it is.

1. **Model FPR/TPR/Threshold Table**

**Tuned XGB:**

|  |  |  |
| --- | --- | --- |
| Target False Positive Rate | True Positive Rate (TPR) | Prob Threshold |
| 1% | 81.30% | 0.2451 |
| 2% | 85.10% | 0.1303 |
| 3% | 87.07% | 0.0831 |
| 4% | 88.03% | 0.0615 |
| 5% | 88.75% | 0.0479 |
| 6% | 89.14% | 0.0391 |

**Insights and Recommendations**

1. **Model Comparison**: Compare the models based on their performance and feature importance scores to identify the most effective model.
   1. A screenshot of a graph

      Description automatically generated**Table of ROC**-AUC, Precision, Recall, and F1 on TEST set.

The Tuned XGBoost model that I created is by far the best model that I created, I am roughly 1% above the baseline in accuracy. More importantly I have an improvement of roughly 7% on my recall and 5% improvement on my F1 score. In the context of the business problem these are the most important metrics because high recall means we are not missing fraudulent accounts and loosing money. Below I have also displayed ROC-AUC and a PR curve. We can see the tuned XGB is the best performing in both scenarios.

* 1. A graph of a line

     Description automatically generated with medium confidence**ROC Charts for each model on Test Set.**

1. **Feature Evaluation**:

I have found that neither email domain, nor zip-code are good predictors of fraud. Their cardinality is too high resulting in models that are too specific to one dataset. Going forward we should find a way to reduce the cardinality of these variables while retaining their information. One solution to this would be to explore rank order binning these down to say 10-15 bins. These would be determined by the average amount of fraud in each zip-code.

1. **Operational Strategy at 5% FPR**:

If we operate this model at a 5% false positive rate a few things will change in the metrics of the model. First, we will have to adjust our threshold, which means the minimum probability that our model assigns the account as fraudulent will move to .049. In terms of predictive power this is extremely weak and risky. If I told someone that I am predicting something with roughly 5% confidence you would never believe me. On top of that we would be increasing our precision to .9412. However, our recall would increase by about 14% which would be the most important feature from all of this. In other words we will be somehow increasing the overall effectiveness of the model by doing this and not experience a tradeoff.

**Plain Language Explanations**

1. **Random Forest vs. GBM/XGBoost**:

Lets pretend that you're a security guard at a bank, trying to identify suspicious people and you have two teams dedicated to helping you catch them.

* 1. The Random Forest Team:

Each analyst works independently, examining different aspects of transactions (amount, location, time, etc.). They don't share information and make separate judgements on whether a transaction is fraudulent. If most analysts flag a transaction, it's likely suspicious. But if they disagree, it's harder to decide. They might miss complex fraud patterns if they're not looking in the right places individually.

* 1. The Gradient Boosting Team:

Analysts work in a chain. The first one analyzes a transaction, identifying potential red flags. The second one builds on that, focusing on areas the first analyst missed or got wrong, trying to refine the suspicion level. This continues with each analyst learning from the previous one's mistakes. They get better at identifying fraud as they collaborate and learn from each other's insights. This can lead to more accurate detection, but they might be too focused on specific patterns and miss new types of fraud.

In fraud terms:

Random Forest: Good at catching obvious fraud based on individual red flags, but might miss complex schemes or be fooled by disguised transactions.

Gradient Boosting: Adapts to new fraud patterns quickly as analysts learn from each other but might be overly reliant on past patterns and miss completely new types of fraud.

1. **Understanding 5% False Positive Rate**: Describe what operating at a 5% false positive rate means in practical terms, emphasizing its impact on customer experience and fraud detection accuracy.

A 5% false positive rate means that 5% of accounts that get marked as fraudulent are legit accounts that committed no bad activity. This could be very damaging to the relationship with those clients, but at that rate 5% of 5% is .0025% of the data or 32 of 12500 people. Additionally, it could be used as a learning opportunity for them to change their spending habits moving forward. I know that I personally have been flagged by my bank as a fraudulent purchase when I make one that is very out of the ordinary for my account.