CS 657 Final Project

Rohit Kate

David Krampert

I chose to build a machine learning program to detect credit card fraud from a dataset of over 284,000 transactions I found on the dataset website Kaggle. I used anomaly detection methods from the sklearn package, the first is the local outlier factor to calculate anomaly scores and then second an isolation forest algorithm. I used these to comb through all of the transactions to build models to predict whether a transaction was valid or fraudulent. The dataset contains 31 parameters including the time the transaction took place, the amount of the transaction, the class of the transaction, and 28 hidden features that are the result of a PCA dimensionality reduction to protect the sensitive information of the credit cards. Having the ability to detect fraudulent charges on a credit card can help save people money and help catch criminals stealing credit card information so using machine learn to build prediction models can help credit card companies get ahead of the bad guys.

From the dataset I was able to determine the max, min, and mean of some of the parameters but because a lot of the feature are hidden for confidentiality reasons, I wasn’t able to explore the other data in great depth. Looking at graph 1 on the left side above the time elapsed between the first transaction in the data set and the last transaction in seconds 172,792 or about 48 hours. Graph 2 has the amount charged to the credit card during this time with the max being $25,691.16 and the mean being around $88.35. Also, from the dataset the number of fraudulent and valid transactions can be found at 284,315 valid cases and 492 fraud cases. In order to build a model that will work for actual fraud detection, using real world data is important which is why the number of valid to fraud cases is drastically different. A problem with having so little fraudulent cases is the models won’t have a large dataset to train and test with which makes the machine learning models difficult to build.

A close up of a logo

Description automatically generated

Looking that the above heatmap the correlation between parameters can be better understood, with the map showing most of the V values being close to zero the data from this set doesn’t differ much other than the time, amount and class. Having such a large dataset can make up for the fact that most of the hidden feature the model will be learning from are very similar. Other information gained from this map is that both time and amount do not have a strong correlation with the class, the class being whether it was a valid or fraud case, seeing that both are close to zero. Looking at the class there is a strong relationship with V17 in the negative and V11 in the positive range. Not being able to know what V17 and V11 are it is difficult to say why the relationship is there, but it will be useful for building the models. There also aren’t many one-to-one correlations, if any, so there is no need to pull out any of the columns before building the models besides the class column.

To format the dataset, I obtained all of the columns from the dataset and then filtered them to remove the class column, so the machine learning program wasn’t able to see whether the case was valid or fraudulent. This is unsupervised learning because if the model knew what the classes were the predictions could be skewed. The variable being predicted on, the target, is the Class with the rest of the columns being used to train the model. I used metrices from sklearn to report how successful the outlier detection was, I used Local Outlier Factor and Isolation Forest for the algorithms when building my models. The Local Outlier Factor algorithm works by calculating the anomaly score of each sample and it measures the local deviation density of a given sample with respect with its neighbors, it is local in that anomaly score depends on how isolated the object is with respect with the surrounding neighborhood. The neighbors in this method are calculated in the same way as K nearest neighbors’ method. The Isolation Forest algorithm works a bit different by returning the anomaly score of each sample, it does this by isolating the observations by randomly selecting a feature and then randomly selecting a split value between the min and max values of the selected feature. In this case all of the columns are the features, and since recursive partitioning can be represented by a tree structure the number of splitting required to isolate a sample is equivalent to the path length from the root node to the terminating node.

From the first run the Local Outlier Factor model, as seen it Graph 3 and Image 1, there were 935 errors reported and there was an accuracy rating of 0.9967 but looking at the precision I could tell that the model was far from that accurate with only a 0.05 precision for class 1, class 1 indicates a fraudulent detection and class 0 indicates a valid case. The recall was also very low at 0.05 but after changing the n\_neighbor variable in my code I was able to increase the precision and recall by 100% as seen in Graph 5. Moving to Isolation Forest I was able to achieve a much greater precision and recall for a class 1 case, getting a 0.34 precision and a 0.35 in recall. Some other notable data from the model was a macro avg of 0.67 for both precision and recall, there were 645 errors reported, and an accuracy rating of 0.9977 but from the precision and recall I know that is not a good judge to how well my model is correctly labeling fraud.

In conclusion the Isolation Forest algorithm worked better for detecting fraudulent transactions on a credit card but the models I could create would not meet any real-world business applications because the predictions were filled with false-positives and false-negatives. None the less, the models were able to label some of the fraud charges and give me feedback to base the next models on. In the future building fraud prediction models would be easier if all the features were known but due to this being real world data, I wasn’t able to see the private information about the cards and card holders. Working for an actual credit card company would give me access to this information but that will require me to apply for a job at one.

Image 1

A screen shot of a computer

Description automatically generated

Image 2

A screen shot of a computer

Description automatically generated

Image 3

A screenshot of a computer

Description automatically generated

Image 4

A close up of text on a white background

Description automatically generated