SVM Assignment

Brandon Maness

**Introduction**

This is a project using 3 SVM kernels, Linear, Radial Basis Function (RBF) and Polynomial (poly) to analyze and classify objects in a dataset based on their feature. Going into this project, I expected the implementation of the kernels to be the most difficult, but I was surprised at how much harder the graphs were than the kernel.

**Dataset**

This dataset is a set of 284,807 credit card transactions, one of which is the 'initial' transaction. Of these transactions, 492 are flagged as fraudulent.

This dataset is definitely obscurified using a transformation of the features.

The features are as follows:

Time: This is the amount of time in second between the current transaction and the first transaction in the set.

V1-V28: These are features that have to do with the transaction. I want to give more info but after a glance at the Kaggle page, this info has been obscurified using a Principle Component Analysis (PCA) transformation. These values are relevant, but I'm not sure how.

Amount: The total of the transaction.

Class: This is a true/false binary. 1 if fraudulent, 0 if not.

**Read Data**

The dataset is made up of various transactions of fraudulent or not fraudulent transactions. While it wasn’t and ideal set because there were very few fraud transactions, I was able to work around it and make the set work.

**Data Preprocessing**

I had to balance the dataset first and foremost, it wasn’t filled with enough fraud so I had to do a resample and repeat some of the cases so the model could recognize them. I split the data into 2 sets, X, the Dataset minus Class, and y, the class column. I ran the train/test/split method to split the dataset to 2 different training/testing sets. One for linear/RBF and a separate, smaller set for poly as it has a horrible time complexity to it.

**Observations**

The data didn’t turn out how I thought. Linear split it well, but RBF seemed to do a lot better than the other two kernels as it had the least false positives and false negatives, showing it was the realest of the three. Considering a false negative is missing a fraud transaction on someone’s account, this further strengthens RBF’s results.

**Conclusion**

All kernels were effective, but of them all RBF was the most effective and poly seemed to be the least. While it balanced the dataset well, the confusion matrix showed too many false values, so it was not very compatible with this dataset. RBF takes the cake with it’s false-negative minimalization and the non-existence of false positives.