Chloe Zeller

Machine Learning Project 1

Technical Report

**Description of Data and Research Question:**

Two datasets were considered in this project, Data Banknote file (denoted DB), and Swiss Banknote data file (denoted SW). Both datasets contain data on real and fake currency. Each column represented a different feature of the banknotes that was measured. Around half of the rows in each data file corresponded to “real” currency, and the other half to “fake” currency. In DB, there were four features to compare real and fake currency, we do not know what these features actually are. There was a fifth column to denote whether or not the data was for a real or fake bill. In SW on the other hand, we have six features to compare real and fake currency:

* Length
* Left Width
* Right Width
* Bottom Margin
* Top Margin
* Diagonal Length

and the first column denoted whether or not the row represented real or fake data. It is also important to note that DB is a significantly larger dataset, it has almost 140 rows compared to SW which only has about 200.

The purpose of this research was to compare three algorithms’ ability to cluster and disseminate real and fake data. We considered K Means, K Nearest Neighbor, and Naïve Bayes as our three clustering algorithms, and compared rates and instances of error in the execution of each algorithm over each dataset. I provided each algorithm with two features from the datasets that had maximized confidence intervals for detecting counterfeit currency, hopeful that it would aid the algorithm in clustering.

**Feature Selection Rationale**

Before executing any of the algorithms, I conducted data analysis on the two datasets to compare features against one another. I calculated confidence intervals while comparing each feature in a given dataset. This confidence intervals translates to how good of a measure those two features were in detecting real versus fake currency. For example, comparing the real and fake measurements of top margin in SW, I found a confidence interval of -0.3584, indicating that top margin measurements were a good measure for comparing between real and fake currency. The confidence interval for diagonal length was -0.282, another strong indicator. This led me to use the top margin and diagonal length as my choice features in the algorithms to come.

In the DW, the confidence interval for column 1 was -0.4176, and the confidence interval for column 2 was -1.121, so again, because of strong indication, columns 1 and 2 were chosen as the features to be used in the three clustering algorithms. I believe that it is important to note that because DW was so much larger than SW, the confidence intervals might have been stronger across the board.

**Description of Classification Techniques**

The first classification algorithm we used was K means. Given an array of data, and a k value of 2 (because we wanted the data to be classified as real or fake), the algorithm will find two centroids, and then pair each data point with one of the two centroids. Then the centroids will be readjusted, the data will be reclassified, and this process continues until the centroids have settled. Choosing a k value of 2 wasn’t so much a choice but rather understanding that we wanted our data clustered into two groups: expected real data and expected fake data (relative to the model). K Means had an interesting effect that I did not initially expect. Because K Means operates off of these two centroids, there was a “flip flopping” effect. Essentially, K Means will assign one cluster to be the “real” cluster and the other to be the “fake” cluster. However, whether or not the first cluster is the real on or the fake one was subjective, based on each individual trial, and chosen at random. I ended up correcting for this later with a few additional lines of code. This was not an issue for the other two algorithms because they didn’t have centroids.

After K Means, we used K Nearest Neighbor (KNN) to cluster the data. KNN allows you to choose a value for the number of neighbors. Through experimentation, I found that using around 4 neighbors (K = 4) yielded a minimized error, without overfitting to an error of 0. I tried various types of distances, including city block and the default Euclidean. I was surprised to find that cosine yielded the lowest error out of the three. For this reason, I chose cosine as my final distance heuristic for KNN.

Finally, we used Naïve Bayes to classify the data. Naïve Bayes doesn’t have any alterable model choices.

**Results of Each Technique**

Naïve Bayes proved to be the best clustering technique, followed by K Nearest Neighbor and finally K Means. I will discuss the basic tendencies in error in this section, and then discuss the inferences in the **Analysis** section. All of the tendencies discussed below were seen across about 10 trials of the experiment overall, comparing different rates of error in each case. There were no cases where the tendencies I mention below did not occur in those 10 trials.

* Beginning with K Means, across both data sets, the test error is consistently higher than the error found in the training data. In the case of DB, the discrepancy between the errors was relatively small, whereas in SW, the discrepancy was much larger (the test data error was often 50%).
* In KNN, the lowest error was found when all features of the training data were fed to the algorithm, followed by when only two features were fed to the model. Highest instances of error were seen when the test data was compared with the feature model. This was true across both data sets.
* Finally, in Naïve Bayes, an interesting effect occurred that was inconsistent with previous results from KNN. The training error when the model was trained on all features was the highest of all the errors, followed by the model trained on two features, and then the lowest error was with the test set. This was true for both SW and DB.

To see sample plots from a trial of the experiment, proceed to the Appendix.

**Analysis of Results**

In K Means, Db showed a smaller discrepancy between the error in test data and training data than in SW. This makes sense given that DB is a much larger dataset than SW, so the model was trained over a much larger range of values, and therefore probably more able to classify the test data. SW on the other hand, was much smaller, so even though the model had been trained already, it was less widely applicable.

In KNN, it’s no surprise that the lowest error was found when the model was trained on all features, considering that it took into account the largest set of data. When the model was trained on fewer features, there is less data to work with, therefore the model will have to essentially make more inferences to create a cluster, it will have to jump a larger Euclidean distance to other data points to keep exploring neighbors. Again, it’s not surprising that the test data error was larger, considering that the model wasn’t trained on the test data, it’s just a prediction for where they will end up.

Finally, for Naïve Bayes, the error seen is a bit odd. In SW, I saw that the error decreased between general training and feature training (2 features used), and then further between feature training and test error. To be fair, this difference is not major, and the largest error margins are still under 20%. Another thing to note is that this effect is more pronounced in SW when compared to DB. To reiterate, this is probably because SW is a much smaller data set than DB.

**Future Work**

In addition to the classification work completed for this project, there are a number of viable extensions. It would be interesting to compare other clustering algorithms beyond K Means, K Nearest Neighbor and Naïve Bayes, like Hierarchical Clustering, and then comparing rates of error. There were also a couple of other feature pairings that I considered because they had comparable confidence intervals. Given more time, it would have been nice to compare the error between different permutations of feature pairings, to try and find the best two features to distinguish error for each data set, as opposed to simple assuming based on confidence intervals.

**Appendix**











