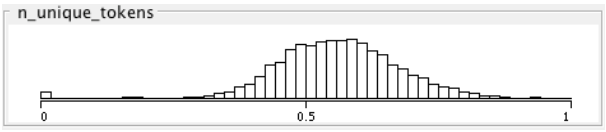
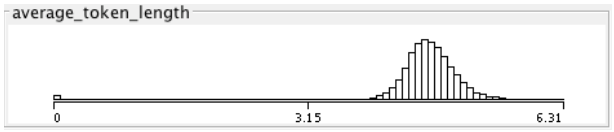
Kaley Findley
CS 4641 Assignment 3
4/2/2017

Unsupervised Learning

Introduction:

For this assignment I will be exploring the concepts of unsupervised learning. I will be exploring two data clustering algorithms, K-Means and Expectation Maximization (EM). I also used four dimensionality algorithms to find correlations between the attributes in my data: Principal Components Analysis (PCA), Independent Components Analysis (ICA), Random Projections (RP), and Random Subset (RS). To illustrate the advantages and disadvantages of each algorithm I used two different data sets. The first data set was also used in my analysis for assignment 1. It had 12 attributes (including a class attribute) that described different characteristics of wine and how those characteristics related to the overall quality of the wine. I used the largest training set from assignment 1, which was 3k instances. The other data set is a new data set than the other data sets I've used in previous assignments. This data set is about online news popularity; it describes details of a webpage (number of images, links, content type, unique words, etc.) and is intended to relate those characteristics to how many times that webpage is shared. Both data sets have continuous values, which is required for the dimensionality algorithms like ICA to run. Online News data has a total of 59 attributes with only about 39k instances. I chose this data set because I knew the curse of dimensionality was a big problem for this data set since 2^59 is approximately 5.76e17. However, I hypothesized that this data would be an expressive data set for the dimensionality algorithms. I chose to use 10k instances from the total 40k to potentially save time running the algorithms. In addition, most of the attribute values, besides the boolean values, were close to a normal distribution so that told me there was not a lot of obvious noise or corrupted data. Some examples of the distributions are shown below.





Test Process:

For each part of the assignment I used Weka except for part two regarding ICA, which required further implementation in Abagail.

Part 1:

The first part of the assignment was to delve into the clustering algorithms, K-Means and EM, and how they behaved for the different data sets. I used the same process for each data set. K-Means required the most parameter manipulation. I experimented with different clustering values (2, 3, 5, 7, 9), initialization methods (Random and FarthestFirst), and distance methods (Manhattan and Euclidean). I initially looked into changing the values for the maximum number of iterations, however during some preliminary test runs I noticed that this value, unless set to extremely small integer numbers, would have no effect on the results. For EM I changed two parameter values, number of clusters (2, 3, 5, 7, 10) and the number of K-Means runs (5, 10, 20, 30).

I ran these test by first starting with K-Means, iterated through each distance metric and ran the algorithm on each of the different cluster amounts for each of the initialization methods. I did a similar process for EM by iterating through the different K-Means run values and ran EM for each of the different cluster amounts. I decided to experiment with the different K-Means runs values to see if running K-Means more times contributed to a more concise clustering.

Part 2:

The second part of the assignment was exploring the details of the dimensionality algorithms. I ran PCA, ICA, RP, and RS, for each data set. I chose RS, or Random Subset, because in terms of reducing the dimensions of your data set, this is probably the simplest and most intuitive method that I could fathom. It simply chooses a random number of your attributes and creates a new data set with your now smaller subset. I hypothesized this would be a good comparison for the more complex algorithms. The question I sought to answer was whether or not the extra time or space complexity from running PCA, ICA, or RP, was worth the results considering Random Subset is very fast.

For PCA on the online news data, I only looked at the effects of keeping a different number of attributes (1-5, 10, 13). The reason I chose 13 was because it was the greatest number of attributes I could have, according to the curse of dimensionality, where my data set could possibly represent. After running PCA with 13 attributes, I looked at the eigenvalues and noticed that the top 10 eigenvalues were above 2, so I decided to run an additional test with 10 attributes. For wine data, I ran PCA with 1-5, and 11 attributes. I chose 11 for the same reason I chose 13 attributes for online news data.

ICA was particularly tricky to run on my data sets. It was having a lot of issues running in Weka and sifting through the massive amounts of data spat out from Abagail was difficult. However, I eventually found that Abagail was my best option. By the time I figured it out, I was short on time and could only run the test with one and two attributes. When I went up to three attributes, it took at least a day before I had to move on with the data that I had.

For Random Projections and Random Subset, I created a pseudo random seed number generator to make RP and RS appear to be 'more random.' The random seed values I got was 1, 34, and 47. For each of the different seed values I ran each algorithm with each of the different numbers of attributes (1-5 for wine data and 1-5 and 13 for online data). Note, if reproducing the results for this algorithm, it will be very unlikely to get the same results, however, there shouldn't be any drastic changes.

Part 3:

This part of the assignment built upon my results for part 2 and was by far the longest testing process of the entire project. I took the results from part 2 and ran K-Means and EM on them to compare to how they performed with the results from part 1. Due to the massive amounts of parameter manipulation, I looked into methods of choosing the 'best' number of attributes, and the number of seed values from the random algorithms, to use for each of the dimensionality algorithms from part 2.

The process for choosing the 'best' attributes for PCA was briefly mentioned above in part 2. For online data, I chose the top 10 features and for wine data I chose the top 4 based on their eigenvalues. Then I ran K-Means and EM on the 10 features with the same parameter changes as described in part 1. ICA form part 2 only had 2 different options for the number of attributes so I simply did a brute force comparison for each of the values. Otherwise, I would have calculated the average kurtosis for all of the sources and pick the number of attributes that contributes to an absolute value of kurtosis that was the furthest from 0. Excel expressed a Gaussian distribution as a kurtosis of 0 instead of 3 to simplify calculations.

For Random Projections and Random Subset, I chose 13 attributes for online data based on the previous justification based on the curse of dimensionality. To choose the 'best' number of attributes for wine data required a bit more calculation. I used the standard deviation (for each attribute across all seeds) divided by the range. I then chose the smallest average of that number to choose the optimal number of attributes and seed values. For RP it was a seed value of 34 and 3 attributes. For RS, it was a seed of 34 and 2 attributes. For online data I simply compared the average standard deviations for each seed and attribute value. For RP that seed value was 47 and for RS it was 1.

Part 4:

This part of the assignment was relatively simple to execute. It took the parts from part 2, but only for wine data, and ran them through the best Neural Network I found using grid search in assignment 1. This NN had the values: hiddenLayers = t, learningRate = 0.1, momentum = 0.4, trainingTime = 400. I used the same values to make a more direct comparison of how using the dimensionality algorithms changed the results. I ran my NN for each dimensionality reduction algorithm and for every parameter I changed along the way.

Part 5:

Similar to part 4, I only used wine data for this part of the experiment. I took the best results from part 3 and ran them through the same NN from part 4, except I used the clusters the algorithms from part 3 found as features in the NN. For each dimensionality algorithm I chose the best clustering based on different error metrics produced in the results.

For K-Means I chose the best Euclidean and Manhattan K-Means algorithms since they produced different error metrics. For Euclidean I chose the clustering with the lowest intra cluster sum of squared error and for Manhattan I chose the clustering with the lowest intra cluster sum of distances across the different clusterings and initialization methods.

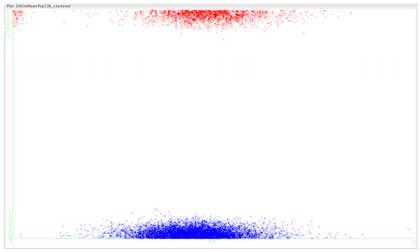
I chose the best EM cluster based on which clustering had the maximum log-likelihood across all the parameter changes. I believed this to be a good metric because the log-likelihood function is a monotonically increasing function, meaning that using this function rather than just a likelihood function will not change the results we get and using log-likelihood makes the background calculations easier. This overall process produced 3 'best' clusters for each algorithm, which was quick to run in comparison to running NN on the original data set before dimensionality reduction.

Data Results:

Online News Data:

Part 1:

I will begin with EM for this analysis. The first plot illustrates the clustering for 5 K-Means runs with a specified 2 clusters. This plot is an example of the behavior I saw throughout the different features. The resulting clustering had lots of crossover between them. In other words, the same attribute value could end up in either cluster. There was nothing to indicate why a value would be in one cluster over another. The clusters appeared to almost mirror each other.

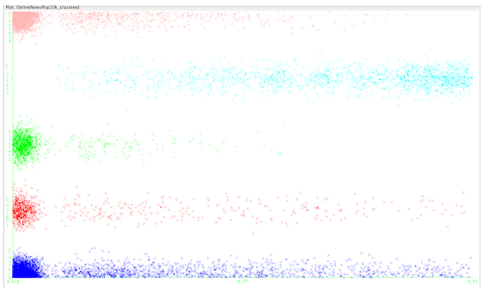


When I increased the clustering to 3 clusters it appeared to have more outlier values for some attributes in the new cluster. I also began to see differences in the binary features. The blue, red, and green clusters are cluster 0, 1, and 2 respectively. This particular plot illustrates how the clusters are distributed for the attribute 'is_entertainment.' A more specific clustering begins to appear. Cluster 0 (blue) is almost exclusively 1, which means if a data point has a 1 for 'is_entertainment' then you can be relatively sure that that particular data point is in cluster 0.

In the non-binary attributes, there begins to be more, albeit very little, separation in the clusters as the number of clusters increase. Once the number

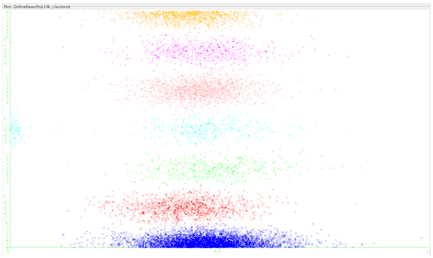
of clusters reach 5, the LDA attributes (describe the closeness of topic of the webpage to some topic described by Latent Dirichlet allocation) begin to reveal small clues that a dimensionality reduction processes could make some improvements.

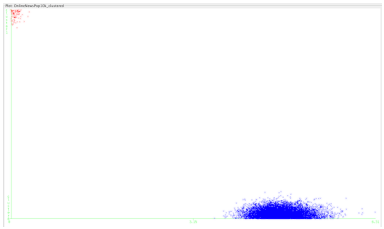
With 7 clusterings, the 'isWorld' and 'isBusiness' attributes, including the aforementioned LDA attributes, were the attributes that revealed the most hints that further exploration would result in some interesting data. However, with 10 clusterings, it appeared the



data was 'over-clustered.' In other words, the data was spread too thin among clusters and began to illustrate worse results than the smaller clusters. This behavior was apparent in the LDA attribute results. When those attributes began to cluster poorly then it was apparent there was a problem. The plot above illustrates one of the LDA attributes. As you can see the clusterings are still not the best one could have, however you begin to see some of the data begin to separate.

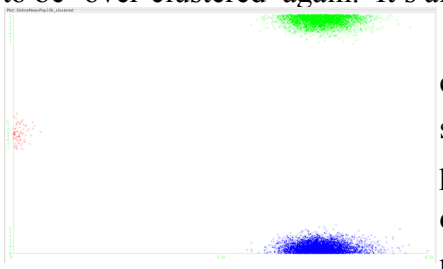
As I increased the number of K-Means runs there is very little improvement in the cluster separation. The little improvements were not significant enough to be interesting. Overall, EM on the online news data set clustered very poorly and it will soon be apparent that K-Means was not much of an improvement.





For K-Means, I explored the plots for the Euclidean and Manhattan distance metrics with the 2 different initialization methods. For Euclidean and Random on 2 clusters, there was good clustering on some of the attributes especially 'average_token_length' as shown in the left plot.

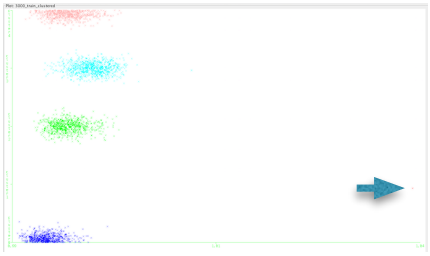
However, simply going up to 3 clusters, there already appears to be problems. It appears to be 'over-clustered' again. It's almost as if the third cluster doesn't really matter very much.



Using FarthestFirst initialization did not improve the clusters at all for any number of clusters. If it did show minor separation it was not any more than what has already been previously discussed. The same goes for using Manhattan for distance. Based on the plots shown above, Manhattan would not be an improvement. Most of the plots look very similar.

Wine Data:

This data set performed very poorly in both EM and K-Means despite the different parameters for each algorithm. The only interesting piece of information in the entirety of wine data was when I ran K-Means with Euclidean/Manhattan Distance and FarthestFirst initialization. Apparently, somewhere in wine data is an extreme outlier that is so far away from the rest of the data points that it was clustered into its own cluster even when I specified 10 clusters. This interesting event is shown in the plot.



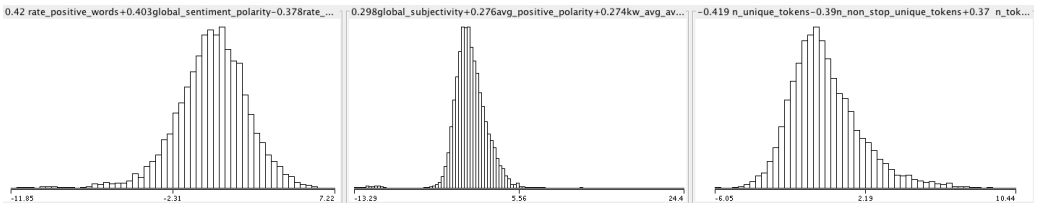
Part 2:

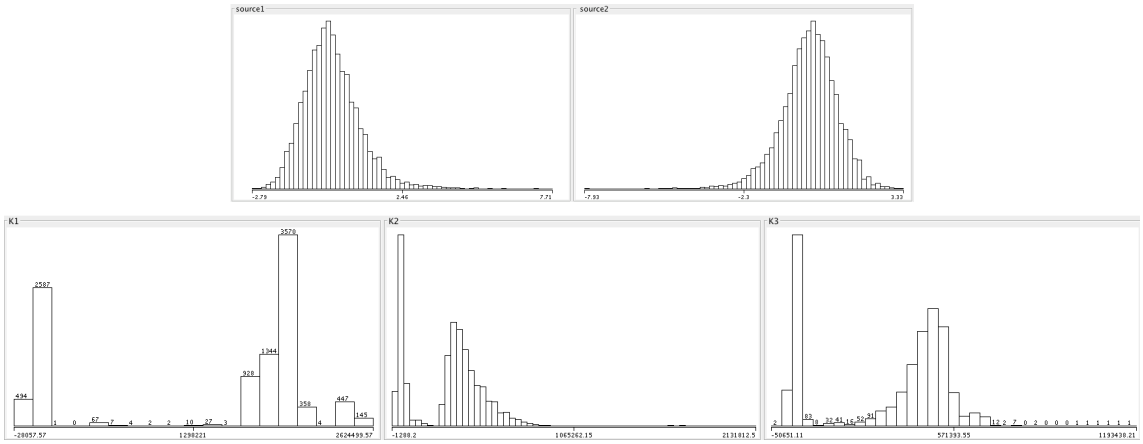
This part is a bit more difficult to visualize since no clustering is involved in this section. For ICA for both wine and online news data, it took quite a bit of time to figure out how to run this algorithm without error. Once I got it running correctly, there was not enough time to run tests that potentially took a few days to run all of them in Abagail. So due to those constraints, I simply ran ICA keeping 1 and 2 attributes. In later sections I will discuss how these performed.

For PCA, I used 1, 2, 3, 4, and 5 principal components for wine data and 1-5, 10, and 13 for online data. PCA gave me eigenvalues for each of these principal components in descending order. I used these values to pick the optimal number of components out of the ones I chose to run. I found that when I ran successive tests with a larger number of components, the eigenvalues would not change, only the new values corresponding to the added components would appear. For example, when I ran PCA with 5 components, the eigenvalues for the previous 4 components were the same. According to these values I decided to set a threshold of 2 for online data and 1 for wine data. The values above the respective thresholds, I would keep for part 3 tests.

As for Random Projection and Random Subset the process for selecting the seed number and the number of features (1-5) was previously explained in the Test Process section. I only considered 1-5 for wine data because it would have reduced the dimensions greatly and I hoped to still preserve the data.

Below are the plots of the distributions for online news data. The first plot is of some of the components of PCA, second is ICA distribution with 2 features, third is for RP.

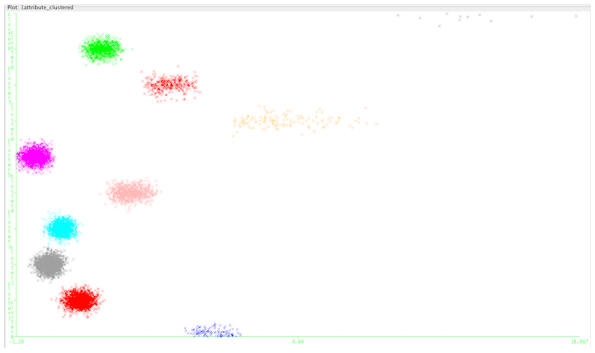




Part 3:

For the analysis of Part 3, I have already chosen the 'best' number of features (and seeds for the random algorithms) as explained in the previous parts, but for clarity I will repeat the values I found. For wine data PCA, the best number of features was 4. For online news data PCA, the best number of attributes was 10. Number of features for ICA for both data sets was 1 and 2 (reasons explained earlier). The seed for online news data RP and RS was 47 and 1 respectively. The seed for wine data for RP and RS was 34. The number of attributes for RP and RS for wine data was 3 and 2 respectively. For online data the number of attributes was 13.

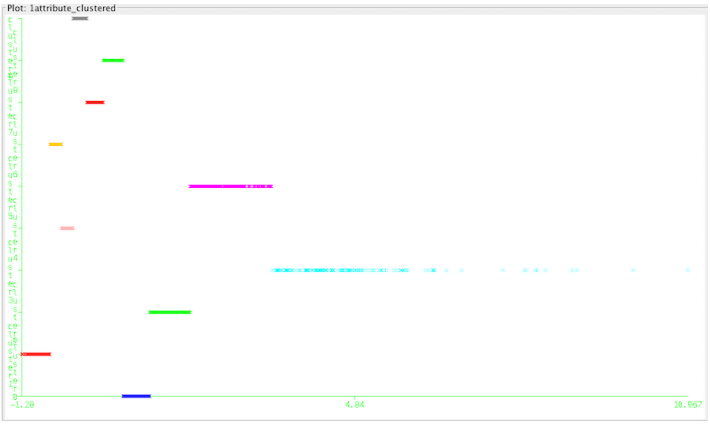
I will start with ICA with EM, on online news data, as the clustering algorithm. I began the analysis on 5 K-Means runs and immediately saw a vast improvement from part 1 clustering. As shown in the plots below (even with a fair amount of jitter) there is very definitive separation between each cluster.



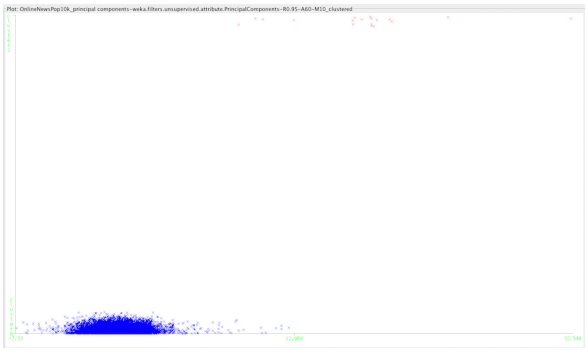
However, once I increase the number of K-Means runs to 10, the clean separation, as shown in the plots to the left, disappear and become very blended and do not represent any correlation between features. My hypothesis for this behavior is that the data points could be too close to each other to be able to commit to a single cluster. It could be due to the soft clustering in EM. The distributions for each of the data points hover in the middle between clusters, unable to commit. In contrast, wine data was far more consistent across multiple values of K-Means runs. It retained a strict separation of clusters as illustrated in the color bar above.

On the other hand, K-Means worked remarkably well for online news data and was very inconsistent for wine data. Using Euclidean distance and both initialization methods contributed

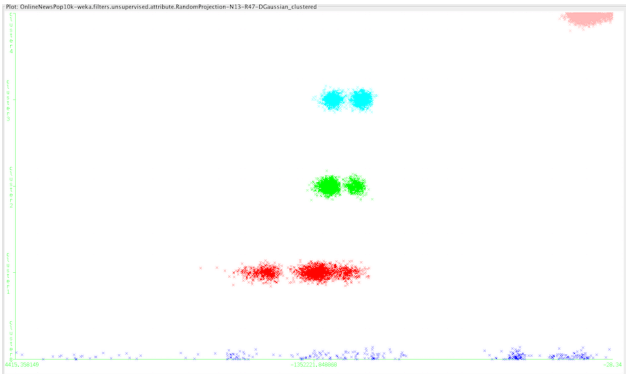
to very cleanly separated clusters. Also, as the number of clusters increased to 10, it captured outlier data in their own separate cluster, which made for a very clean graph. Using the Manhattan distance performed similarly expect it captured more concise clusters as shown in the plot.



The next dimensionality reduction algorithm is PCA. Using EM for online news data and wine data performed very poorly. It did not show any inter cluster separation. There was no indication that could tell me why a data point was clustered the way it was. K-Means also performed poorly for both data sets, the only difference was that there appeared to be one dominant cluster that contained the majority of the data points while the others had only a few. PCA for both clustering algorithms on wine data, resulted in extremely poor clustering. I believe this substandard clustering for PCA is the fact that the attributes in both data sets are in reality not dependent on one another, which translates to poor performance in PCA. I believe my hypothesis is correct after seeing how well ICA performed. The distributions of the features were good indicators that it had a good chance of performing well in ICA.



The clusters for Random Projections looked bizarre in comparison to the other algorithms. With online news data for EM and 5 K-Means runs with 2 clusters resulted in what appeared to be one cluster in between the other cluster and its outlier. Wine data in this same situation performed better as in the clusters aren't in between each other. This arrangement does not necessarily mean the clusters are well separated. As the number of clusters and K-Means runs increases, the clusters start to look like two nodules with a space in the middle. This clustering is illustrated in the plot above.



As for K-Means, the behavior seems to be less consistent than say ICA. Using Euclidean distance and Random initialization, 2 clusters was the best at cluster separation. It performed better than a higher number of clusters with online news data. As for wine data, the clustering got worse as the number of clusters increased for any combination of distance and initialization methods. I am starting to notice a trend that K-Means is a very poor algorithm choice for wine data. I have also begun to notice that the distance metric plays a smaller role in the clustering as I initially thought. It turns out that the initialization method seems to play a big role in terms of clustering, which makes sense if you think about the methodology of K-Means. It relies heavily on where you start.

Finally, there is Random Subset. The only interesting trend from these round of tests was that it heavily depends on the seed and what set of attributes it randomly picks. Theoretically, RS could be lucky and grab all the relative attributes and perform extremely well. However, the flip side of this situation could happen as well; it could be particularly unlucky as well.

Part 4:

For this part I used the reduced and projected values from part 2 and ran them through the same 'optimal' Neural Net I found in assignment 1. The goal was to see how well the NN can learn with a reduced dimension. My initial thoughts was that it would greatly decrease the time for the test to run since there are less attributes. I ran the NN for each of the hyper-parameters used in part 2. I compared the root mean squared error (RMSE) using cross validation. The best runs in terms of RMSE are as follows: ICA was 0.9307 (one attribute), PCA was 0.7695 (11 attributes), RP was 0.8095 (seed=47, 5 attributes), and RS was 0.8029 (seed=1, 5 attributes). Clearly the best dimensionality reduction algorithm was PCA, however, the RMSE for PCA was still very high.

The RMSE for cross validation for the NN from assignment 1 was 0.2902. This means that PCA (the rest of the algorithms) failed to relate the clusters in terms of the quality of the wine. It turns out that the correlations between the attributes does not directly relate to the quality of the wine. If time was not an issue I would delve into the online news data set. It would have been particularly interesting if my original hypothesis at the beginning of the paper was correct; whether or not the curse of dimensionality would be avoided once reducing the dimensions and whether that translated into a lower RMSE error in NN.

Part 5:

This part is where the data is particularly interesting. I used the best clusterings from part 3 as an added feature in the data set and then ran it through the same NN as part 4. I used wine data again to directly compare it to how the NN performed in part 4. The best clusters were chosen using the methods I described in the Test Process section. The data results for ICA are as follows: K-Means (Manhattan, 10 clusters) was 0.3347, EM (5 K-Means runs, 10 clusters) was

0.9232. The results for PCA were: K-Means (Euclidean, Random, 10 clusters) was 0.3838, EM (7 clusters, 5 K-Means runs) was 0.2638. The results for Random Projection were: K-Means (Euclidean, Random, 10 clusters) was 0.3575, EM (5 clusters, 10 K-Means runs) was 0.5799. The results for Random Subset were: K-Means (Euclidean, Random, 10 clusters) was 0.3364, EM (10 clusters, 5 K-Means) was 0.5163. The best clustering, dimensionality reduction pair was EM using PCA with 7 clusters and 5 K-Means runs. The RMSE for the NN in assignment was 0.2902 and EM clustering with PCA had a lower RMSE. I thought this was particularly interesting considering how clustering with PCA performed so poorly. This may have been due to the fact that when I chose the 'best' parameters for PCA I didn't use kurtosis. That might have led me to a more accurate set of parameter values, which could have translated to better clusters.

Conclusion:

After all the tests and analyses, I have concluded that using dimensionality reduction algorithms along with clustering algorithms actually perform better (in the case of Neural Nets) than simply running a NN on the original data set. However, I was not expecting PCA to be the winning algorithm. Based on the running performance throughout the paper, ICA appeared to be performing the best in terms of cluster distributions and consistency. However, I think ICA fell victim to time constraints and if I had some more time I could have found even better hyper-parameters which would have surely translated into better results in terms of sum squared error (K-Means) or log-likelihood (for EM). Also, due to time constraints I was unable to perform another grid search for the Neural Net again to find the best parameters for my new reduced data sets. I believe that would have made a difference in the results in Part 4 and 5. As far as the clustering algorithms, I would have like to explore the effects of the other parameters as well. In particular, I would have liked to experiment with the Canopy initialization method in K-Means. There were several different Canopy specific parameters in K-Means that I was unable to explore. In EM, I wanted to see if changing some of the minimum setting parameters played a significant role in the clustering. In particular, I hypothesize that the minimum improvement for log-likelihood would play a significant role, considering I used that metric to make important decisions.