Unsupervised Learning and Dimensionality Reduction

Data Overview

The Mushroom data set I chose came from the UCI Machine Learning Repository and includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. The definitely poisonous and unknown edibility classifications were combined to create a binary class variable. The data set contains 22 different attributes about the mushrooms, some of which I used in my analysis based on a number of factors. The data cleansing process I used for the dataset included removing nulls and checking for singularity as these both could affect my ability to create an accurate model. The original mushroom data set contains 8,124 instances, but after removing rows with nulls there still are 5,644 instances that I was able to use in my models. I also removed the column veil.type as there was only one value in the column making it somewhat useless in including it in the analysis. This left me with 21 viable attributes to use in my models that I changed from categorical to dummy variables in order to use them accurately in model creation. There is no simple rule for determining the edibility of a mushroom; no rule like "leaflets three, let it be"' for Poisonous Oak and Ivy, so as a result, using this dataset for modeling is actually quite helpful in determining other ways to identify possibly poisonous mushrooms based off the mushrooms attributes.

The UCI Credit Card data set I chose came from Kaggle. It includes a variety of attributes such as default payments, demographic factors, credit data, history of payment, and bill statements. The data set includes a total of 30,000 instances and 24 variables (all of which are numeric, so I scaled the data) which I used in the creation of the models (there were no null values to remove in the original data set). The target variable is whether they have defaulted on their credit card payment. The UCI Credit Card data set was interesting to me because this is a very common type of data set to use for classification models. It also has a lot of data which I thought would be a good thing to look into given the mushroom data set had below 10,000 rows.

Mushroom Data Analysis

K-Means Clustering

K-Means Clustering is a common algorithm used in unsupervised learning to organically find groups in the data with those groups represented by the chosen K value. The algorithm runs iteratively through the data to assign each data point to one of the K groups based on feature similarity, which in this case is defined by Euclidean distance or closeness to the cluster centroid. These centroids of the K clusters can then be used to label new data. I used the kmeans function in R to test K values 2 through 10 to determine which performed best. Below is the resulting elbow plot in Figure 1 that I used to determine the best value of K, which normally occurs where the bend in the graph commences. In this case it seemed to be at K=4, however the smoothness of the graph makes it somewhat difficult to interpret. In order to compare the accuracy of the K-Means clustering model I choose to use K=2 as that represents the true number of labels in this dataset (given this is already known as it's labeled data). I did this in subsequent algorithm analyses throughout this assignment for comparison purposes. A value of K=2 seems to provide very distinct clusters with the data with the model having an accuracy of 85.37%. Although there seem to be some outliers in the Figure 2 cluster graph, I still think the model does a fairly good job in classifying the data. The model itself took a fairly short amount of time for each cluster as shown in Figure 3.

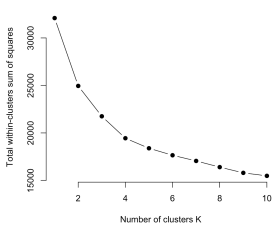


Figure 1

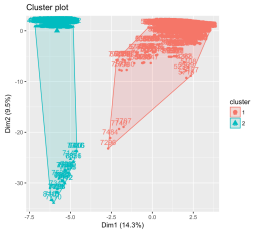


Figure 2

Figure 3

Expectation Maximization

Similar to K-Means, Expectation Maximization (EM) is an iterative method used in unsupervised learning to create soft clusters based on a data points probabilistic features of being in the cluster. It looks to maximize the likelihood of observed data by estimating means and standard deviations of the different clusters. This can be evaluated to find the most appropriate number of clusters for the data using a BIC graph which shows the value of the maximized loglikelihood. Similar to the K-Means evaluation, I tested 9 different clusters in order to compare the models more closely. Figure 4 shows the highest BIC to be 1 component which seems somewhat odd as the true labeling is 2 classes. Both K-Means and EM did not choose clusters that match the actual classification of the data nor did they match each other's K selections. This could be explained by the way distance is defined by each algorithm, with K-Means using Euclidean distance to determine centroids and resulting clusters and EM using more statistical methods. The timing of the EM algorithm also did not take too long to run, though it did take a quite a bit longer than the K-Means algorithm as shown in Figure 5.

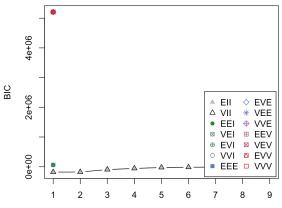


Figure 4

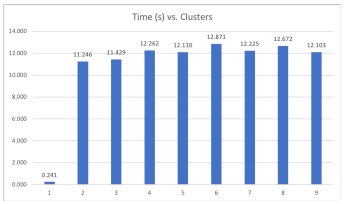
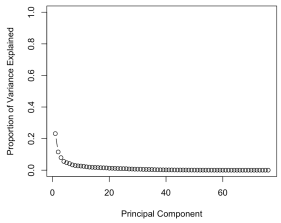


Figure 5

Principal Component Analysis (PCA)

Principal Component Analysis (PCA) works to reduce the dimensions of your feature space by focusing on the variables that matter the most providing you with fewer variables and thus less of a chance to overfit your data. It also helps this data set in particular because dummy variables created a large number of variables to process meaning the amount of data needed to support results grows exponentially. PCA will thus provide a more manageable data set to run through K-Means and EM with only the variables that explain the most amount of variance in the data. In Figures 6 and 7, we can see that not many variables have a significantly high variance explanatory power. The PCA chosen variables that explain approximately 75% of the variance would include the top 20 PCA variables to be used in re-running K-Means and EM.

Figure 6



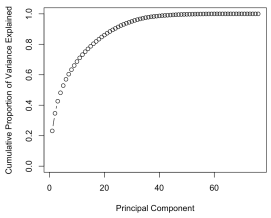


Figure 7

PCA and K-Means Clustering

I used the top 20 PCA variables to re-run the K-Means clustering algorithm again testing K values 2 through 10. Below is the resulting elbow plot in Figure 8, which seems to show a bend at K=2, though again, the smoothness of the graph makes it somewhat difficult to interpret. A value of K=2 seems to provide very distinct clusters with the data with the model having actually the same accuracy as the original K-Means model at 85.37%. The clustering does seem to show a more reasonable grouping though using PCA as shown in Figure 9 in comparison to the original K-Means model. The model itself took a much shorter amount of time to run than the original K-Means as well, most likely due to the fewer number of variables as shown in Figure 10. Though the accuracy is not changed, the quicker time and more grouped clusters seem to show PCA may be a good option to use with this dataset.

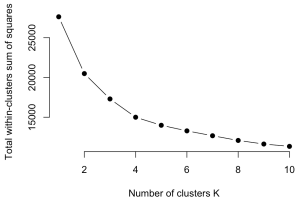


Figure 8

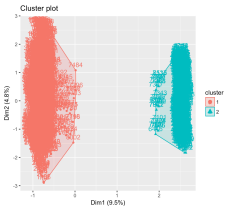


Figure 9

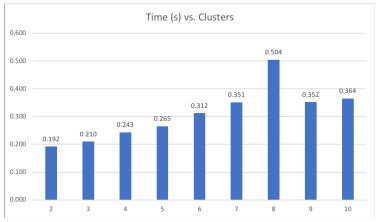


Figure 10

PCA and Expectation Maximization

I used the top 20 PCA variables to re-run the EM algorithm again testing K values 1 through 9. The peak of the BIC seems to be at 7 components as shown in Figure 11, which since we know the true number of labels to be 2, this seems quite high. The cluster plot in Figure 12 would also imply that this may not be a very accurate number of clusters as there do not seem to be very distinct cluster patterns. The timing did go down though from the original EM model as shown in Figure 13, which again makes sense because of the fewer variables. Outliers or overfitting could be possible explanations for such a high component value. In general, it does not seem that PCA with EM improved the model's performance by much.

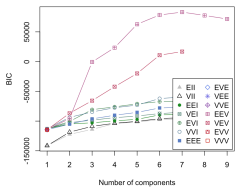


Figure 11

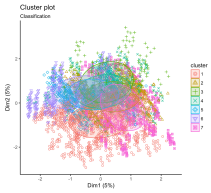


Figure 12

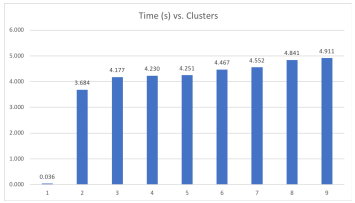


Figure 13

Independent Component Analysis (ICA)

Independent Component Analysis (ICA) seeks to transform the data by maximizing the statistical independence of each component which can be found through maximization of non-Gaussianity. In this case we can see that the kurtosis peak in Figure 14 shows an optimal number of components to be 5. This is again rather high since we know the true value to be 2. This kurtosis value represents the shape of a probability distribution and thus seems to show that according to ICA 5 components will best maximize non-Gaussianity.

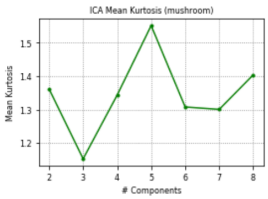


Figure 14

ICA and K-Means Clustering

I used the ICA transformed data to re-run K-Means Clustering on K values 2 through 10. The elbow graph in Figure 15 shows that a K=2 value is most appropriate as the graph starts to bend there. However, from looking at the cluster plot in Figure 16, you can see that the clusters are not very well defined as they were in the original K-Means cluster plot. It does not seem that using ICA transformed data works very well with this data set as it doesn't seem to improve model performance by much. In fact, the accuracy is much lower at 49%. Although the time did decrease quite a bit as shown in Figure 17, I do not believe that would be enough to warrant using this method for transforming the data in a K-Means clustering.

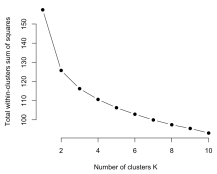


Figure 15

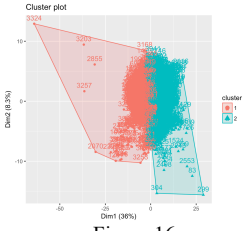


Figure 16

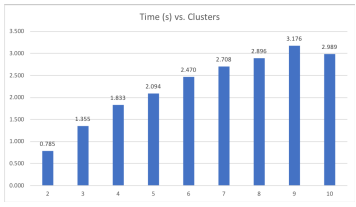


Figure 17

ICA and Expectation Maximization

I next used the ICA transformed data to re-run the EM algorithm on K values of 1 through 9. The BIC graph in Figure 18 shows a peak at 9 components, which is again quite high. Similarly, the cluster plot in Figure 19 seems to show very weak clustering with most of the clusters overlapping. This would indicate that the ICA data does not work well in this model to improve performance from the original EM model. Perhaps the high number of components ICA is transforming causes the model to be overfit as it produces an even higher recommended number of components than the kurtosis plot. The time again went down with this model though only slightly as shown in Figure 20, and that would not be enough in my

opinion to make up for the inability to cluster appropriately. Interestingly, the time is much lower at 5 components, which is the suggested component value according to kurtosis. Perhaps using 5 clusters would create a better performing model, though it would not accurately classify the data according to the labels it already has.

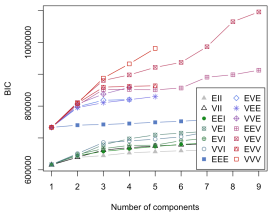


Figure 18

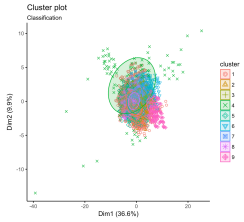


Figure 19

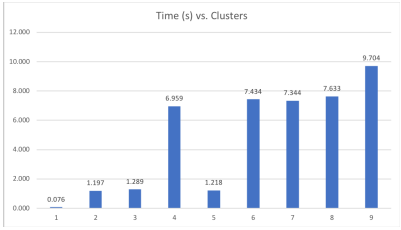


Figure 20

Randomized Projections

The goal of Randomized Projections (RP) is to find the most efficient way to reduce the dimensionality of your data set trading a set amount of error for speedier processing times and more manageable model sizes. The dimensions and distribution of random projection matrices are set so as to preserve as best as possible the pairwise distances between any two samples of the dataset. Performing RP on the Mushroom data set yielded Figures 21 and 22 showing that the reconstruction error decreases with the number of components. They also show that the pairwise distance correlation increases the with number of components though the curve does flatten out around 40 components, slightly higher than what PCA suggested. In order to get the most accurate model it would seem that 40 components would do well to provide a lower reconstruction error while maintaining over 80% of the pairwise distance correlation, which is what we want to do.

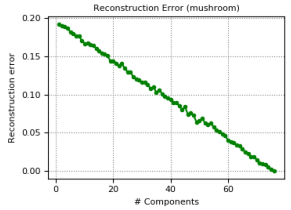


Figure 21

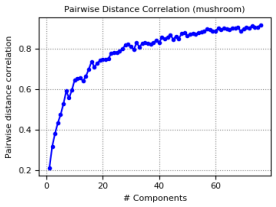


Figure 22

Randomized Projections and K-Means Clustering

I used the data from the RP output to re-run K-Means Clustering testing it with 2 through 10 clusters, which yielded an extremely low accuracy of 14.63%, by far the lowest accuracy so far. This would indicate that the RP model's basis of reducing time and error to gain efficiency and a smaller model do not work well for this data set. The elbow curve in Figure 23 seems to show a K=4 value as being optimal, though when it's tested with K=2 to compare to the true labeling of the data, the clusters are quite distinct. Although the cluster plot in Figure 24 does seem to show that the model is able to cluster well, the accuracy does not reflect this as it is not classifying the data points correctly. Even the decrease in time shown in Figure 25 does not make up for the terrible accuracy of this model.

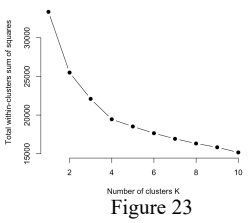


Figure 23

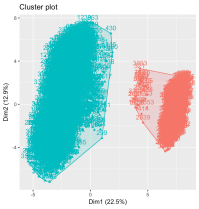


Figure 24

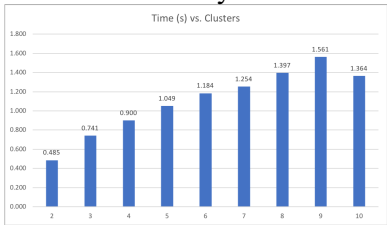


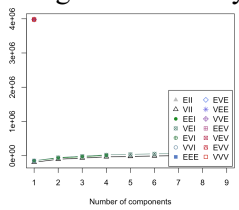
Figure 25

Randomized Projections and Expectation Maximization

I used the RP data again to re-run the EM algorithm on clusters of 1 through 9. The BIC graph in Figure 26 shows a peak at 1 component which does not reflect an accurate labeling of the data. This is lower than the suggested K value from K-Means perhaps because the RP dimensionality reduction method does not work well with the statistical methods for choosing K used by EM. In general, the

models that have suggested only a value of 1 for K seem unrealistic as this provides no use in clustering at all. The unsupervised technique in a sense is not working quite correctly. After the first cluster, the time increases significantly as shown in Figure 27 to even higher times than the original EM model indicating that the efficiency RP hopes to gain would only be realized by having simply one cluster.

Figure 26



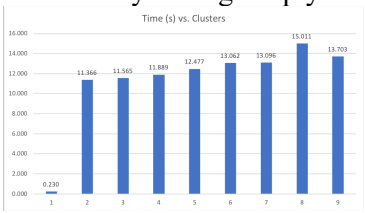


Figure 27

Feature Selection – Random Forest

I choose to use a Random Forest model for my feature selection technique, which works by examining a large number of decision trees by generating a random sample of the initial data through bootstrapping and using a set number of variables to determine node splitting. Multiple sets of trees are created and the importance of each variable in each decision is noted. In unsupervised learning, the clusters are formed organically by trying some clusters, in this case 2, and measuring the closeness between data points. The graph in Figure 28 shows the cumulative percent of variable importance by feature. From here I chose the variables that accounted for 75% or more of the variance in the data to use in my K-Means and EM algorithms, which ended up being 16 of the original variables. I then transformed the data back to its dummy variable state to perform the K-Means and EM analysis.

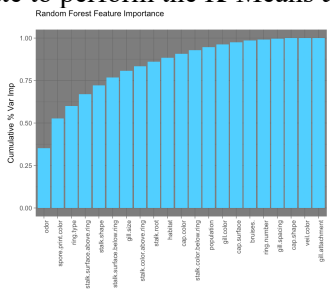


Figure 28

Feature Selection and K-Means Clustering

I again tested K values 2 through 10 yielding the below elbow plot in Figure 29, which seems to show a bend at K=3, though the smoothness of the graph makes it somewhat difficult to interpret. I used a value of K=2 for accuracy comparison which created decent clusters as shown in Figure 30, though there is some overlap and outliers. The accuracy of the model was 57.25% which is rather low, perhaps as a result of those outliers, which if removed could lead to better model performance. In general, this accuracy does not indicate that the model performance of K-Means is improved by using this feature selection technique. Although time did decrease in this model in comparison to the original K-Means as shown in Figure 31, it would not be enough for me to use this method in the future on this data set.

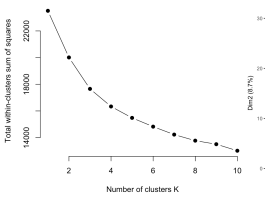


Figure 29

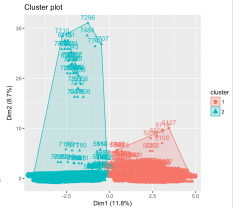


Figure 30

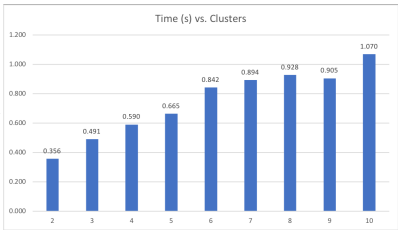


Figure 31

Feature Selection and Expectation Maximization

Using the Random Forest data to test K values 1 through 9 for EM yielded the below BIC graph in Figure 32, which shows a peak at 1 cluster. Similar to the results from RP, the time increases significantly after this first cluster as shown in Figure 33, and again, it would seem that the unsupervised technique is not working very well here to identify the data correctly. Removing the outliers here too may cause the model to improve in future iterations.

Figure 32

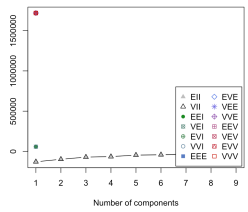


Figure 33

UCI Credit Card Data Analysis

K-Means Clustering

I ran K-Means Clustering on the UCI Credit Card data set testing K values of 2 through 10, which yielded the elbow plot in Figure 34. From this I found the bend to be at K=3 clusters as the optimal cluster amount, but I used K=2 for comparison sake. The cluster plot of this analysis in Figure 35 shows that the data is quite widely dispersed thus making it difficult to cluster into only 2 forms. There seem to be many outliers that could lead to this distribution, which if removed could improve model performance. The accuracy of this model was 68.38%, which is much worse than the performance of the mushroom data set using this algorithm. The time also increased as shown in Figure X in comparison to the mushroom data set which is odd considering there are fewer variables and rows.

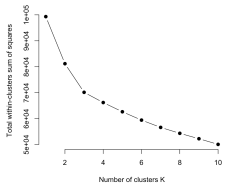


Figure 34

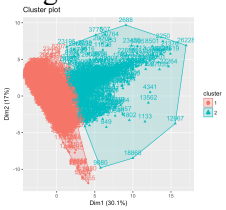


Figure 35

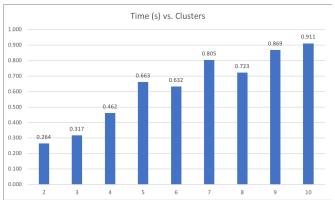
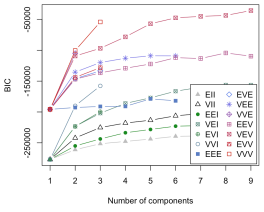


Figure 36

Expectation Maximization

I then ran the EM algorithm with this dataset which yielded the BIC chart in Figure 37 with a peak at 9 components. This is much different than the suggested 1 component given for the mushroom data set and might imply that the UCI data set is more complex. Similarly, the cluster plot in Figure 38 shows many possible outliers with large clusters in one general location. There seems to be a lot of overlapping and indistinct regions of clusters indicating that this data set does not perform very well with this soft clustering method. The runtime for this model was also surprisingly high as shown in Figure 39, another possible indicator of poor performance as efficiency is definitely not gained in this method. Although this may have to do with the nature of EM vs. K-Means calculation methods, it could also be a sign that the model does not work well with this data set.

Figure 37



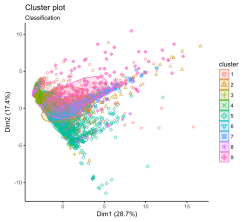


Figure 38

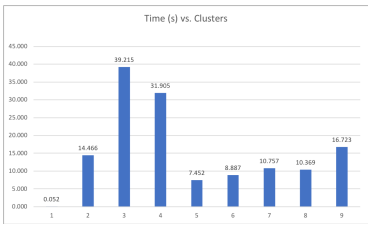
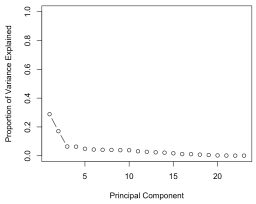


Figure 39

Principal Component Analysis

I then ran the first dimensionality reduction algorithm on the UCI data set which produced Figures 40 and 41, where we can again see that not many variables have a significantly high variance explanatory power. The PCA chosen variables that explain approximately 75% of the variance would include the top 10 PCA variables to be used in re-running K-Means and EM.

Figure 40



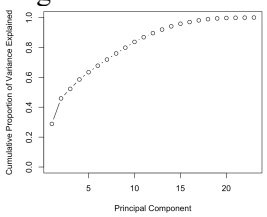


Figure 41

PCA and K-Means Clustering

I used the PCA variables to re-run the K-Means clustering algorithm again testing K values 2 through 10. Below is the resulting elbow plot in Figure 42, which seems to show a bend at K=3. A value of K=2 seems to provide very distinct clusters with the data with the model having a much lower accuracy than the original K-Means model at 31.52%. The clustering does seem to show a more reasonable grouping though using PCA as shown in Figure 43 in comparison to the original K-Means model. The model itself took a shorter amount of time to run than the original K-Means as well, most likely due to the fewer number of variables as shown in Figure 44. Though the accuracy is so low that I do not believe this is a good technique to use with this data set for improving the model.

Figure 42

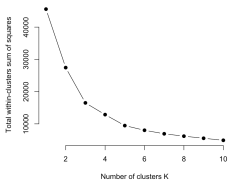


Figure 43

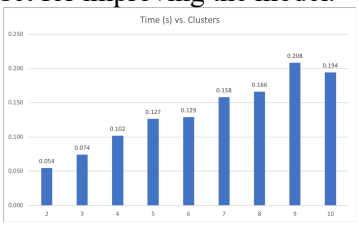
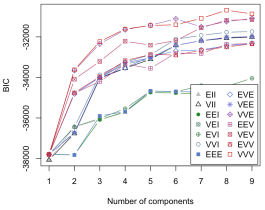


Figure 44

PCA and Expectation Maximization

I again used the top PCA variables to re-run the EM algorithm testing K values 1 through 9. The peak of the BIC seems to be at 8 components as shown in Figure 45, which since we know the true number of labels to be 2, this seems quite high. The cluster plot in Figure 46 shows somewhat distinct clusters which may indicate that EM with PCA could be a potential candidate for EM model improvement. The timing did go down quite substantially from the original EM model as shown in Figure 47, which makes sense because of the fewer variables. Interestingly, we are seeing the same jumpy from K=1 to the following in terms of time. Outliers or overfitting may again be possible explanations for such a high component value. In general, it does seem that EM with PCA improved the model's performance a good bit.

Figure 45



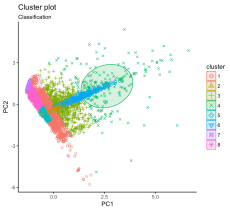


Figure 46

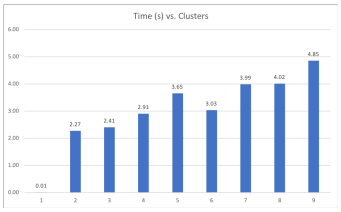


Figure 47

Independent Component Analysis

Next, I ran ICA on the data set which yielded the kurtosis graph in Figure 48, which shows a peak at 8 components. This is rather high since we know the true value to be 2, though the graph does interestingly peak at 5 as well indicating another possible option. The ICA model ran in 0.226 seconds, extremely fast in comparison to the time it took for the mushroom dataset to run. This would lead me to believe that the number of variables has a significant impact on the efficiency of the ICA model though not necessarily on the recommended number of components, which may have to do more with data set complexity. I saw this in assignment 1, that the UCI data set often performed worse than mushroom.

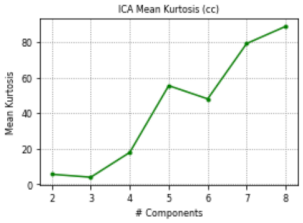


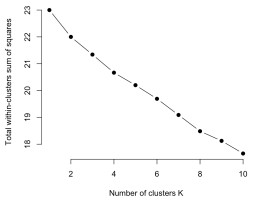
Figure 48

ICA and K-Means Clustering

I used the ICA transformed data to re-run K-Means Clustering on K values 2 through 10. The elbow graph in Figure 49 shows that a K=2 value is most appropriate though the extreme smoothness of

the line makes interpretation very difficult. However, unlike the mushroom data set, the UCI data set cluster plot in Figure 50 shows two very distinct clusters when run on K=2 indicating that it might be an appropriate cluster value. The model performance did not confirm that thought though as it decreased to 58.12%, about 10% lower than the original K-Means model. Given the purpose of ICA, this may be a difficult model to use on a more complex data set as it is attempting to maximize the statistical independence of each component. The run time also increased slightly as shown in Figure 51, adding yet another reason why ICA does not improve on the original K-Means algorithm for this data set.

Figure 49



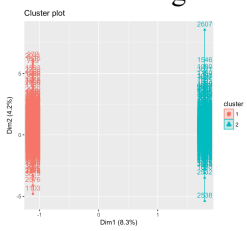


Figure 50

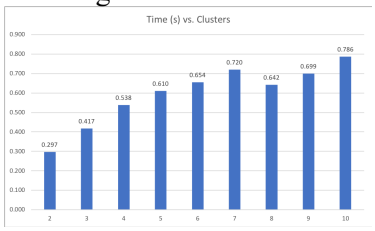
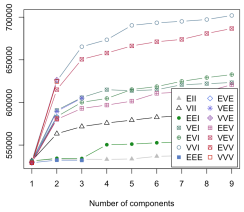


Figure 51

ICA and Expectation Maximization

I next used the ICA transformed data to re-run the EM algorithm on K values of 1 through 9. The BIC graph in Figure 52 shows a peak at 9 components, which is again quite high. Similarly, the cluster plot in Figure 53 seems to show very weak clustering with most of the clusters overlapping. This is similar to what I found with the mushroom data set. This would indicate that the ICA data does not work well in this model to improve performance from the original EM model. Interestingly, the time went down as the number of clusters increased as shown in Figure 54, which I have not seen with many of the models. The EM time in general has been high for this dataset making K-Means seem like more of a suitable algorithm to use on it.

Figure 52



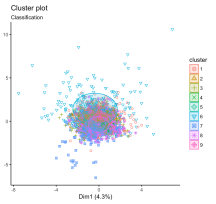


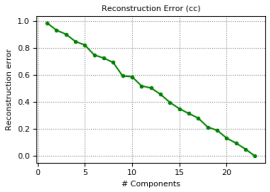
Figure 53

Figure 54

Randomized Projections

I then performed the next dimensionality reduction algorithm on the UCI data set which yielded Figures 55 and 56 showing similar results as the RP performed on the mushroom set. The reconstruction error decreases with the number of components and the pairwise distance correlation increases the with number of components though the curve does flatten out around 10 components, much fewer than we saw with the mushroom and actually the same as what was suggested by PCA. In order to get the most accurate model it would seem that 10 components would do well to provide a lower reconstruction error while maintaining over 90% of the pairwise distance correlation, which is what we want to do. There is also less variance in Figure 56 than there seemed to be with the mushroom set which may suggest that the pairwise distance correlation is smoother for this data set.

Figure 55



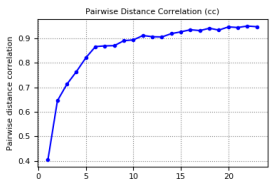


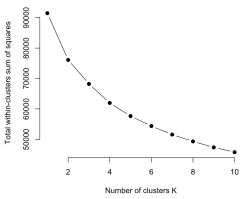
Figure 56

Randomized Projections and K-Means Clustering

I used the data from the RP output to re-run K-Means Clustering testing it with 2 through 10 clusters, which yielded an accuracy of 51.62%. This is worse than the original K-Means and accuracy and would indicate that the RP model's basis of reducing time and error to gain efficiency and a smaller model do not work as well for this data set. The elbow curve in Figure 57 seems to show a K=2 value as

being optimal, though the curve here is also quite smooth. When it's tested with K=2 to compare to the true labeling of the data, the clusters are not very distinct at all, Figure 58. Neither the accuracy nor the visual clustering shows this model improving upon the original K-Means. Even the decrease in time shown in Figure 59 does not make up for the terrible accuracy of this model. The run time for RP is similar though to the normal K-Means, thus no added efficiency is gained in that sense.

Figure 57



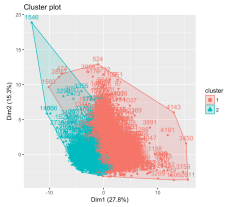


Figure 58

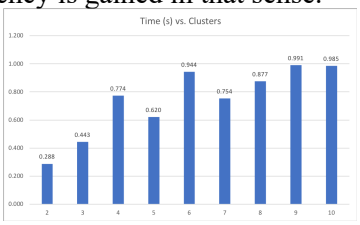
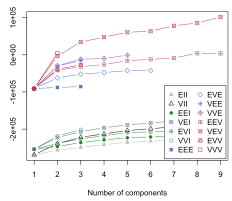


Figure 59

Randomized Projections and Expectation Maximization

I used the RP data again to re-run the EM algorithm on clusters of 1 through 9. The BIC graph in Figure 60 shows a peak at 9 components, which does not reflect an accurate labeling of the data. This could be another indication of the complexity of the data set as it's difficult to cluster into only a few groups. Interestingly, K values of 2, 3, and 4 have the highest run time (Figure 61) while it significantly decreases as K increases after then making the suggested higher number of clusters more understandable. It seems the algorithm has a hard time grouping this data set into only a few clusters. RP's goal of finding the most efficient way to reduce the dimensionality of a data set trading a set amount of error for speedier processing times and more manageable model sizes may not be best if you have a complex data set.

Figure 60



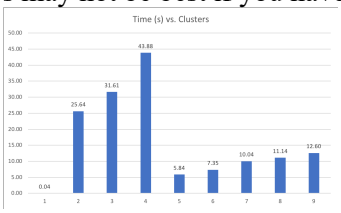


Figure 61

Feature Selection – Random Forest

I ran the same Random Forest model as my feature selection technique for this data set which produced the Figure 62 graph of the cumulative percent of variable importance by feature. From here I chose the variables that accounted for 75% or more of the variance in the data to use in my K-Means and EM algorithms, which ended up being 11 of the original variables, which I used for K-Means and EM.

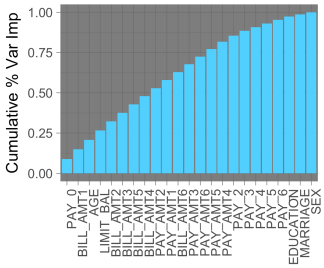


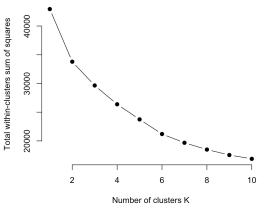
Figure 62

Feature Selection and K-Means Clustering

I again tested K values 2 through 10 yielding the below elbow plot in Figure 63, which seems to show a slight bend at K=2. I used this value for accuracy comparison which created decent clusters as shown in Figure 64, though there are some outliers. The accuracy of the model was 54.83% which is rather low, perhaps as a result of those outliers, which if removed could lead to better model performance. In general, this accuracy does not indicate that the model performance of K-Means is improved by using this feature selection technique. Although time did decrease in this model in comparison to the original K-Means as shown in Figure 65, it would not be enough for me to use this method in the future on this data.

Figure 64

Figure 63



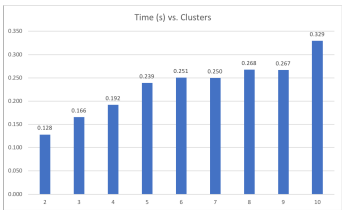
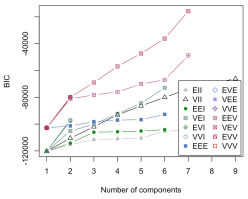


Figure 65

Feature Selection and Expectation Maximization

Using the Random Forest data to test K values 1 through 9 for EM yielded the below BIC graph in Figure 66, which shows a peak at 7 clusters. Similar to previous iterations of the EM algorithm, a higher number of clusters is suggested, again suggesting the complexity of the data and the necessity for a lot of clusters to fully split the data. However, the cluster plot in Figure 67 shows a very bad clustering with most clusters almost fully on top of one another. Interestingly, the time here increases significantly (Figure 68) after the first cluster, possibly indicating that the larger clusters are increasing difficult to create.

Figure 66



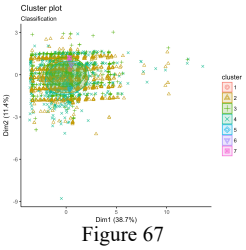


Figure 67

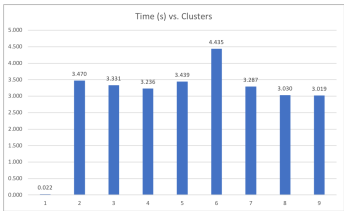


Figure 68

Neural Network Analysis and Conclusion

I re-ran my Backpropagation Neural Network from the previous assignments with the UCI Credit Card data set with the test accuracies for each of the models as well as the original shown in the table below. The original Neural Network was running using the scaled UCI Credit Card data set with a 10-fold CV and 15 different sample sizes. The original model performed best with 1 hidden layer as the testing accuracy did not increase significantly with more layers, but if anything, it decreased. I thus used these parameters for the subsequent 6 Neural Networks for comparison sake.

Model	Original	K-Means	EM	PCA	ICA	RP	RF
Accuracy	81.3%	78.2%	33.6%	78.3%	78.6%	80.1%	78.3%

Interestingly, the originally backpropagation model with no change to the original features of the UCI data set performed the best at 81.3%, marginally beating out K-Means, PCA, ICA, RP, and RF (Random Forest). The EM model by far performed the worst with an accuracy of 33.6% indicating it does not handle the data set well for clustering. The most likely reason for this is due to the nature of how it calculates its soft clusters and the definition of distance which uses statistical methods to find the clusters it thinks are the most correct. The difference in speed was insignificant for the 7 different neural networks. In general, the K-Means algorithm performed the best amongst all of the different models for both data sets indicating that this method of hard assigning clusters worked best in comparison to EM's soft assigning method. It did not seem that dimensionality reduction had much of a positive impact in this case other than in the K-Means with PCA for the mushroom data set. This kept the same accuracy while decreasing runtime and the number of variables creating a more manageable model. K-Means with ICA worked the best for the UCI Credit Card data set (second to the original K-Means model) and could be used to increase efficiencies and create an overall smaller more manageable model with that data if you were willing to forgo almost 10% of the accuracy.