Datasets:

Breast Cancer (BC):

My first dataset is the Breast Cancer Wisconsin (Diagnostic) Data Set. The dataset is only 569 examples and has 30 features plus the labels. I treated this dataset as a binary classification problem, to predict if the tumor was malignant or benign. The dataset has 357, or 63%, benign examples, and 212, or 37%, malignant examples.

Fashion MNIST (FM):

My second dataset is Fashion MNIST. Additionally, I only used the labels corresponding to t-shirts, trousers, coats, bags, and ankle boots. This dataset has 30,000 examples, 784 features, and each label is equally represented. Furthermore, this is a image classification problem of grayscale images. Hence, each feature column corresponds to a pixel location, and is an integer between 0 and 255.

Clustering:

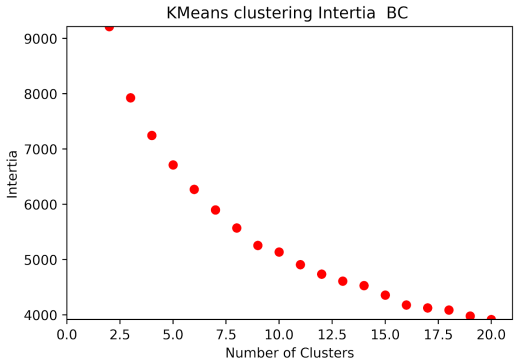
Clustering is an unsupervised learning algorithm, which groups samples based on similarity. Similarity for examples below is euclidean distance between samples. To give equal weight to all the features in a sample, the features are scaled. In addition, to run the algorithm I split the samples into training (75%) and testing (25%) sets, and found the clusters using the training set, and calculated inertia and other scoring metrics using the testing sets.

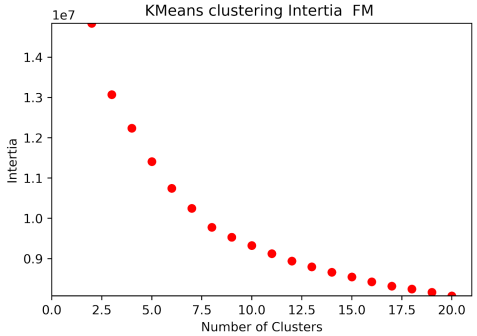
To analyze the clusters I graphed the homogeneity, completeness, Fowlkess-Mallows, and Silhouette Coefficient score. Homogeneity score is 1 if all the clusters contain only points of a single class, and the score varies between 0 and 1. If the homogeneity score is high that means the clusters line up with the labels. Completeness score is 1 if all samples of the same class fall in the same cluster, and also varies between 0 and 1. Fowlkess-Mallows is the geometric mean between precision and recall. The last metric I graphed is the Silhouette Coefficient of all the samples, which is calculated using the mean intra-cluster distance and the mean nearest-cluster distance. Unlike other metrics, which are comparing between the ground truth labels and the predicted clusters, the silhouette coefficient is scoring using only the clusters and no ground truth labels. To decide on the best cluster it should have high homogeneity score, without decreasing the completeness score too much. Additionally, it should have a high Fowlkess-Mallows score, and a high silhouette coefficient.

K-Means Clustering:

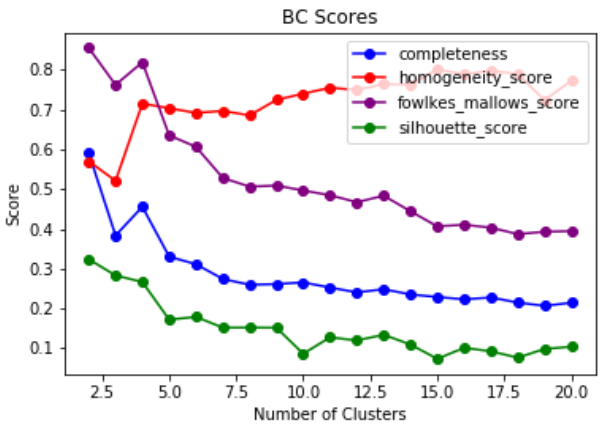
K-means clustering consists only of 3 steps. First, initializing points to be centroids. Second, assigning samples to a cluster. Third, calculate the mean of the samples that are assigned to a centroid, and make this the new centroid for step 2. Steps 2 and 3 are repeated until they reach a minimum threshold of very little movement after each iteration.

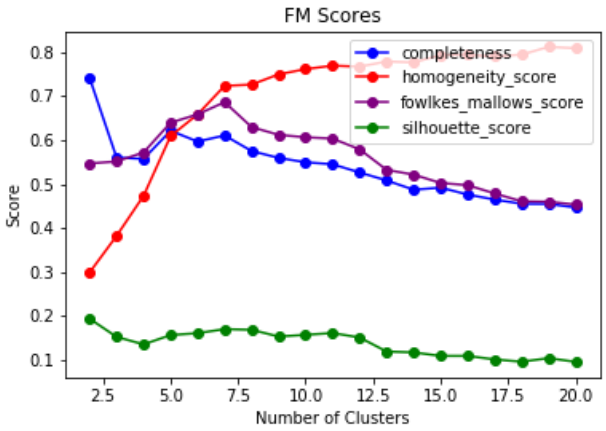
For the graphs below I score the inertia vs number of clusters. Inertia is the sum of squared errors within clusters. Intuitively, it makes sense that as the number of clusters increases the inertia decreases because there should be a cluster closer to the sample as the clusters increase. Furthermore, the inertia for FM is much larger than BC because FM has more samples and more features to account for.





It's interesting to note that in both the graphs below, homogeneity score and completeness score trends towards opposite directions as the number of clusters increases. This makes sense for completeness because as the number of clusters increase there's less of a likelihood all samples with the same label will be in the same cluster. The increase in homogeneity score as the number of clusters increases is also a positive because a cluster is consisting mostly of samples with the same label. Based on the scores below for BC, the best number of clusters seems to be 4 because it has a relatively high homogeneity score, high completeness score, and high Fowlkes-Mallows score, and high silhouette score. Hence, at 4 it's the minimum number of clusters needed to differentiate labels well. For FM, the best scores occur at 7 clusters because at this point homogeneity score starts to plateau. Furthermore, Fowlkes-Mallows score is maximized at this point, and the completeness score is relatively high.

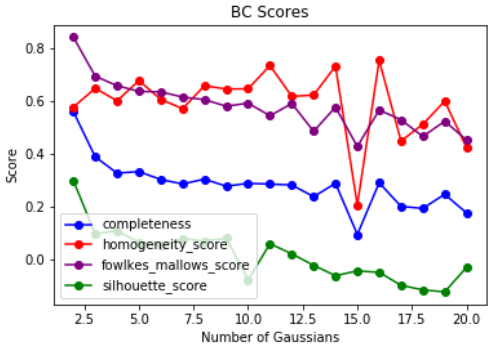


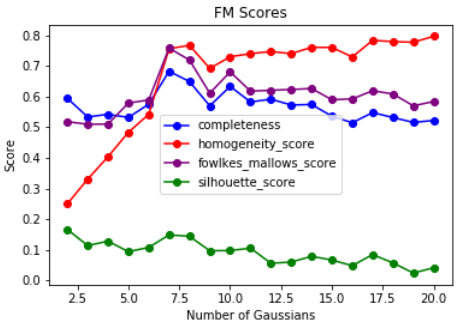


Expectation Maximization (EM):

Expectation maximization is very similar to K-means because it also varies between assigning samples to cluster centers and recalculating the centroids. However, unlike K-means, expectation maximization does soft clustering. Hence, rather than assigning samples to only one cluster, each sample is given a probability about each cluster. Additionally, we assume that the samples are generated from one of K gaussian distributions.

In the figures below, we can see that compared to K-means there are more sudden dips and increases in the scores for EM. This may have occurred because EM may have gotten stuck in local optima. Additionally, similar to K-means for BC the best scores occur around 4 gaussians and for FM the best scores occur around 7 gaussians. Furthermore, the BC and FM scores have the same general trends and values as K-means.





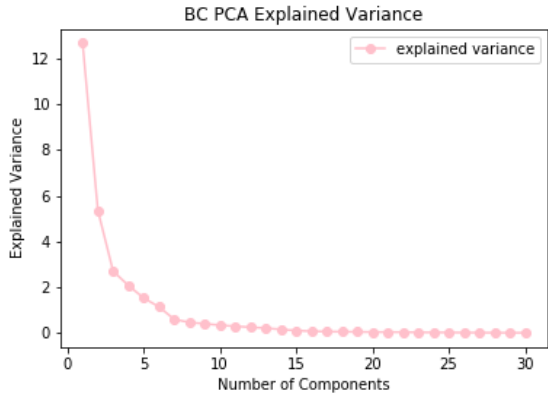
Dimensionality Reduction:

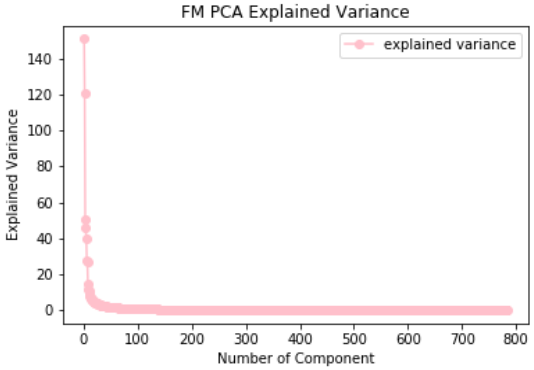
Feature transformation takes the original features and transforms them to a new, smaller set of features in such a way as to retain as much information as possible. Below we look at 3 different type of feature transformations, PCA, ICA, and randomized projections. Feature selection is a type of feature transformation that selects a subset of the original features to keep, and below I did K-select

Principal Component Analysis (PCA):

PCA works by first transforming the samples along the principal component, which is the projection that maximizes the variance of the features. Next, we can remove the components that give us less information to reduce the dimensionality of the samples. Our components are found by calculating the eigenvalues, and the largest eigenvalues maximize the variance.

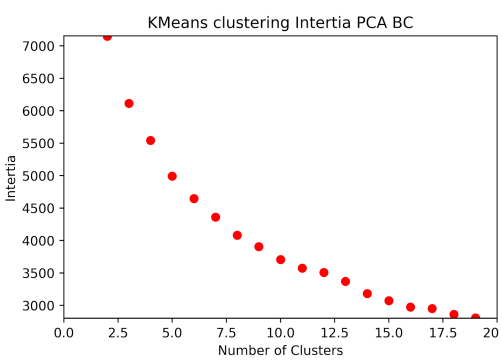
In the figures below we can see how dramatically the variance decreases as the component number increases. Hence, we can see that for BC that after 7 components the explained variance starts to plateau. For FM, the explained variance starts to plateau around 10 components. Thus for the BC graphs below I used 7 components and for the FM graphs below I used 10 components.

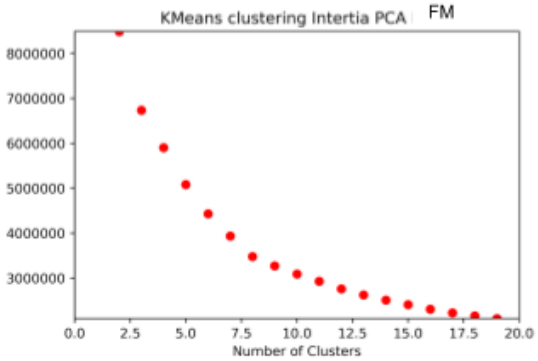




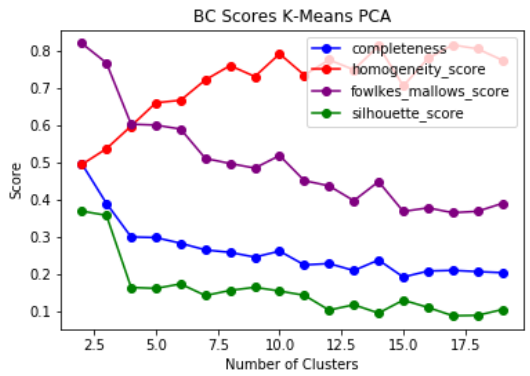
PCA with K-means:

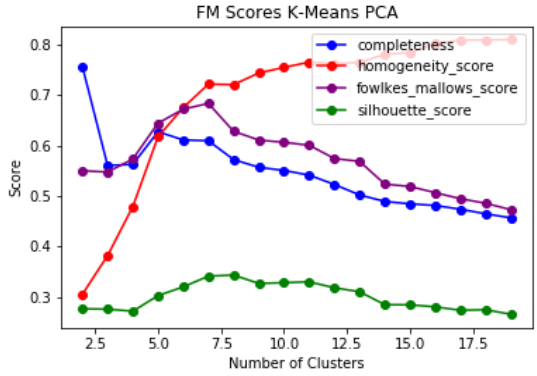
With PCA the K-means inertia decreased significantly for both BC and FM. For BC at each cluster it went down by about 2000, while for FM it went down by about 6000. This makes sense because PCA makes the features describe variance, and if I used fewer components the inertia would decrease even more.





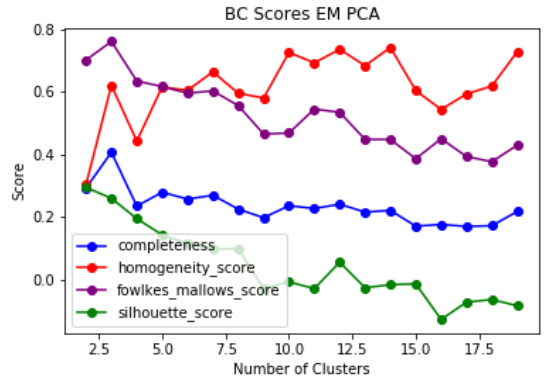
With PCA the ideal number of clusters for BC seems to occur around 8 because that's when the homogeneity score stops trending towards increasing. Hence, after applying PCA K-means for BC needs more clusters to align clusters to labels. For PCA FM the scores look very similar to the no PCA FM scores. Hence, the ideal number of clusters still is 7.

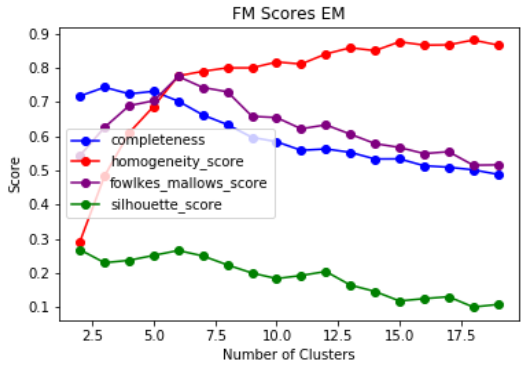




PCA with EM:

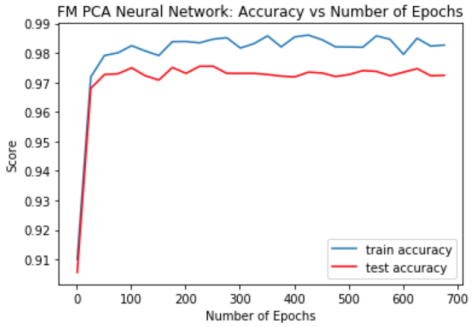
Interestingly, for BC the homogeneity scores start off much lower when PCA is applied, than without PCA. Furthermore, the ideal number of gaussians is 8, because at this point all the scores are relatively high. For FM the homogeneity score with EM achieved the highest homogeneity score that we've seen so far. Furthermore, with PCA at each number of gaussians each of the scores is slightly higher than without PCA. Hence, this may mean that for FM when applying PCA and doing EM we have better, and more compact features for clustering. Lastly, the ideal number of gaussians is still about 7.

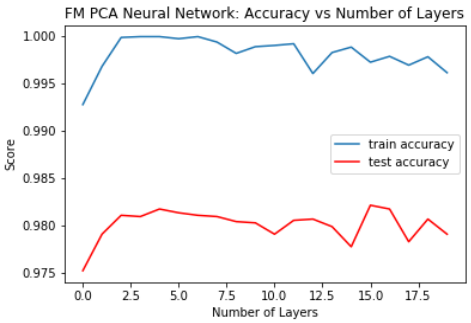




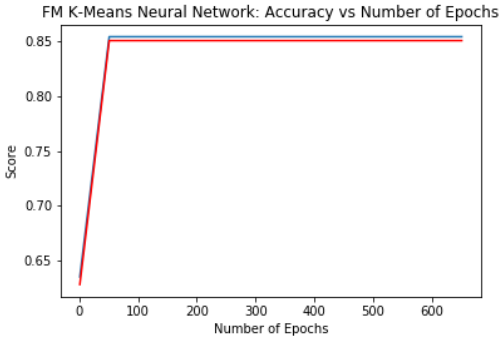
PCA with Neural Networks:

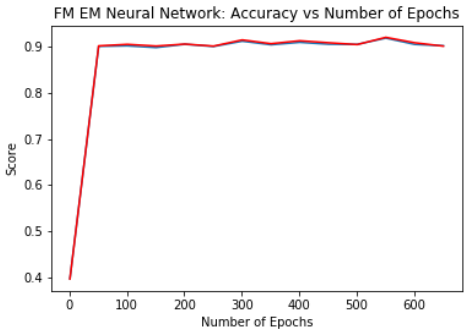
For the figure on the left I used 14 hidden layers for the FM dataset, and compared to without PCA both the train and test accuracy is about 1 percent lower. This decrease in accuracy might be able to be explained with the loss of information that was valuable to the classifier when the dimensionality was reduced. For the figure on the right, compared to without PCA up until 5 layers with PCA performs about a percentage point better.





In the graphs below I used K-means and EM as the feature for the neural network. For K-means I used the labels, which is 1,1 by feature, and I used 7 clusters. With only this one feature it did surprisingly well with 0.85 accuracy. For EM I used the probability that it belonged to a cluster as the features, and 7 gaussians. With EM we see a 5% improvement over K-means showing that it retains more valuable information. We saw this above too when we saw higher scores for EM than K-means.





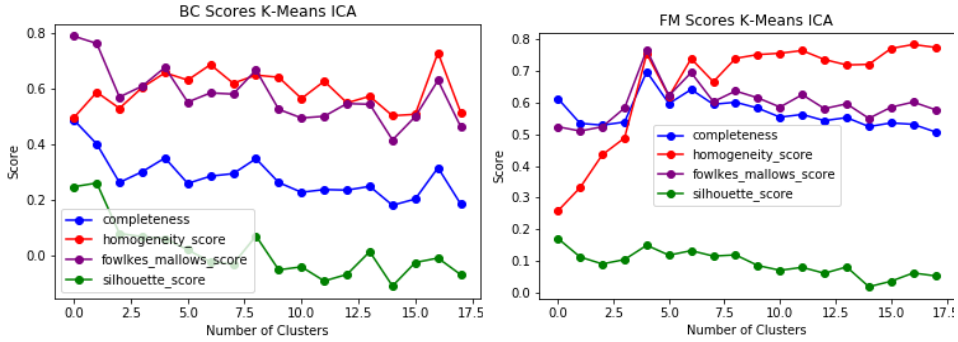
Independent Component Analysis (ICA):

The goal of ICA is to maximize the independence between the transformed features. Additionally, in the graphs below the projection axis for both BC and FM capture meaningful information because the clusters are able to correspond to ground truth labels.

ICA with K-means:

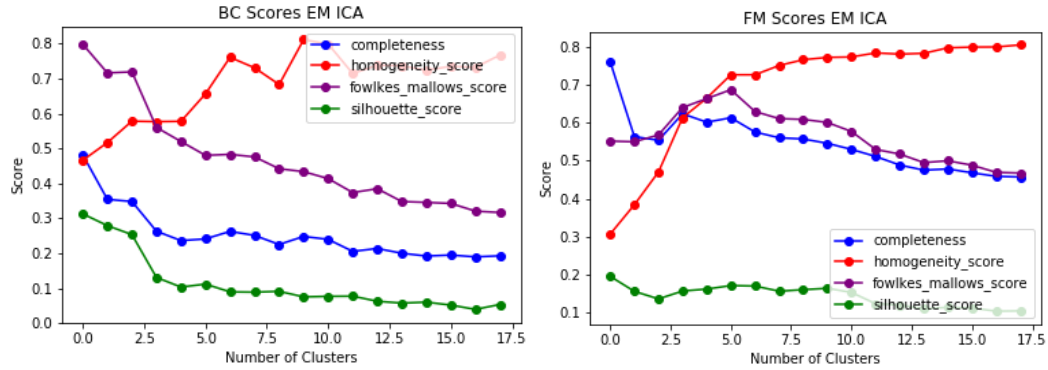
Compared to PCA for BC, ICA performs about 10 percentage points worse at each cluster for homogeneity scores. Furthermore, at higher cluster values the Fowlkes Mallows score is significantly higher than PCA, and unlike in PCA where the Fowlkes Mallows score slopes downward here it remains pretty stable. This occurs because there's a higher similarity score between clusters and higher completeness score. For BC I believe the optimal number of

clusters is 5 because that's where scores are maximized. For FM it's still around 7



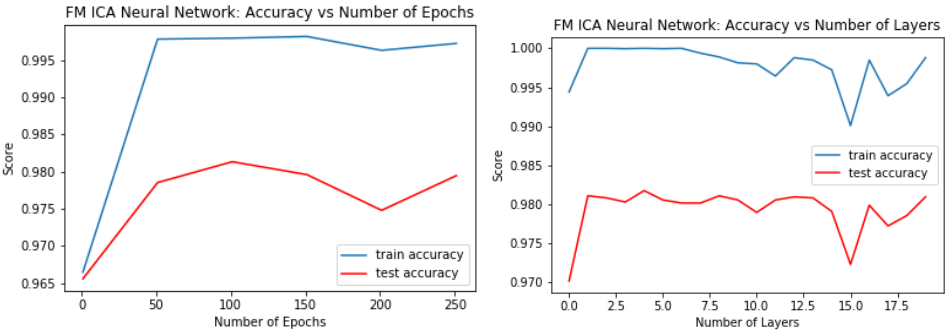
ICA with EM:

For BC and FM using ICA, the scores look very similar to using PCA because all the scores are trending in the same direction and have similar values. However, for homogeneity score for FM is a bit lower than PCA. Hence, when using ICA FM isn't able to cluster based on label as well as before

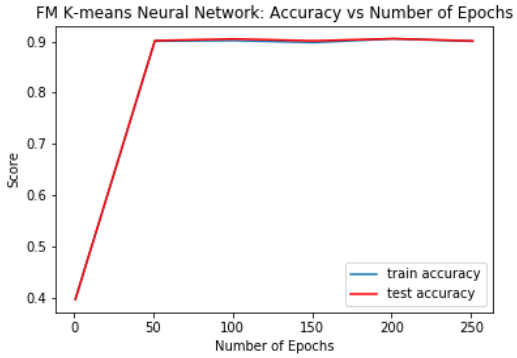


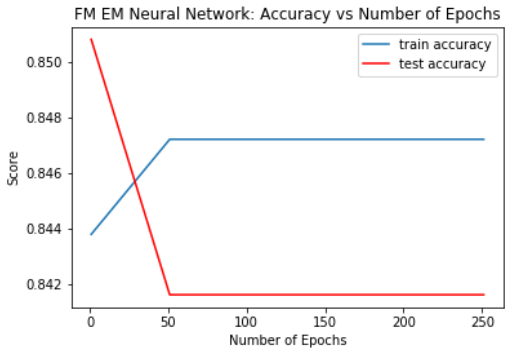
ICA with Neural Networks:

The accuracy vs number of epochs scores are very similar to the accuracy scores when doing no dimensionality reduction. This shows that compared to PCA, ICA might do a better job in preserving information that's important to build a better classifier. Additionally, higher accuracies are achieved with fewer hidden layers. Also the data is linearly separable because there's 97% accuracy with no hidden layers.



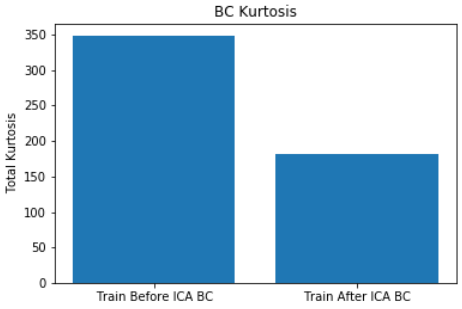
I used 7 clusters and 7 gaussians for the K-means and EM respectively. Interestingly, when doing ICA K-means performed better than EM. This means that with fewer features and less information the neural network performs better.

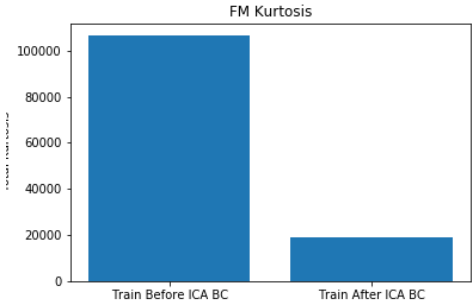




ICA Kurtosis

The kurtosis significantly decreases after applying ICA. This makes sense because the goal of ICA is to make the features more independent from each other than the original data. BC started off with more independent features than FM because the change in kurtosis is less dramatic. This makes sense because FM is image data, and pixel values that are next to each other should be similar, and not independent.



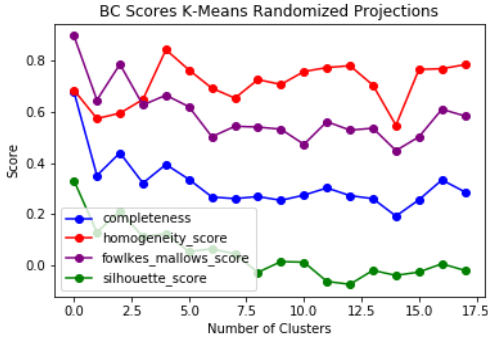


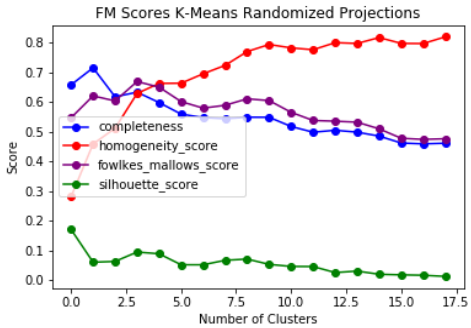
Randomized Projections:

Randomized projections project higher dimensional data to a lower dimension, but the trade off is some accuracy. Furthermore, this is computationally less expensive than PCA and ICA. For the graphs below I projected BC and FM to 10 components.

Randomized Projections with K-means:

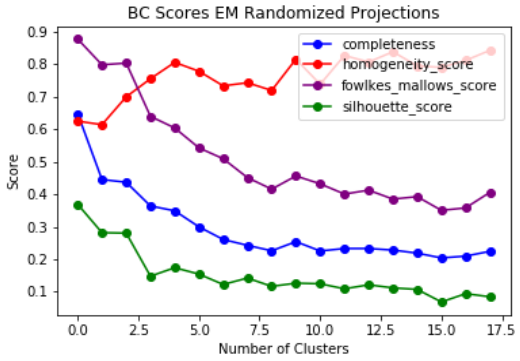
Randomized projections performed comparably to PCA and ICA. The ideal number of clusters for BC occurs around 5, which is similar to ICA. When trying projections to 7 components for BC, the results were much lower than PCA and ICA. Hence, I used 10 components for BC and 20 components for FM. The ideal number of clusters, where the clusters were separated by their labels occured, at around 9 clusters for FM.

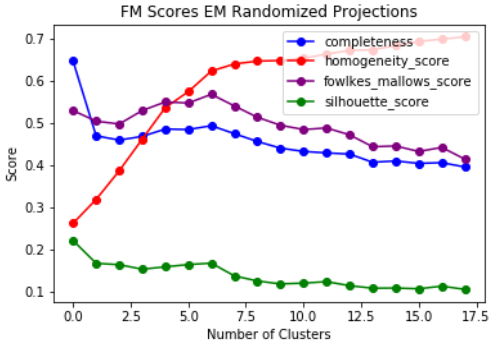




Randomized Projections with EM:

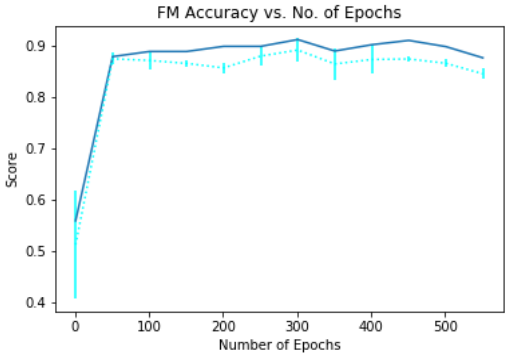
Surprisingly, the BC homogeneity scores are the highest compared to any other previously seen homogeneity scores. Also interesting to note that in this randomized projection, the homogeneity score doesn't trend downwards at higher number of gaussians, which happened in our previous EM graphs. Hence, this random projection seems very good at creating clusters that separate based on the label, while also maintaining high scores for the other measures. However, for FM the randomized projection did less well than before at creating clusters based on the label than before, which was more expected.

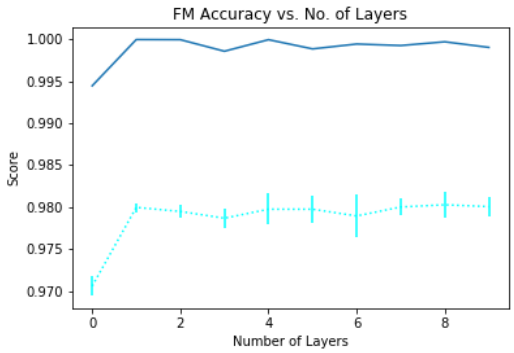




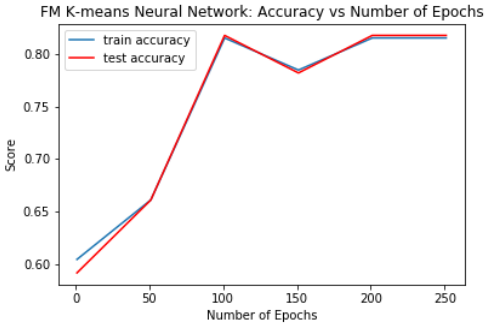
Randomized Projections with Neural Network:

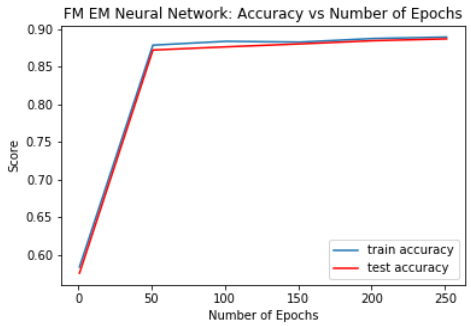
For the graphs below I ran each data point 3 times and graphed the average and standard deviation. It's interesting to note that the standard deviation trends downwards as the number of epochs increases. This might occur because as the we increase the number of epochs we are having more opportunities to do backpropagation to modify weights for randomized propagation. For the other graph the standard deviation seems to increase as the number of layers increases, which might be because there are more weights to modify as the number of layers increases.



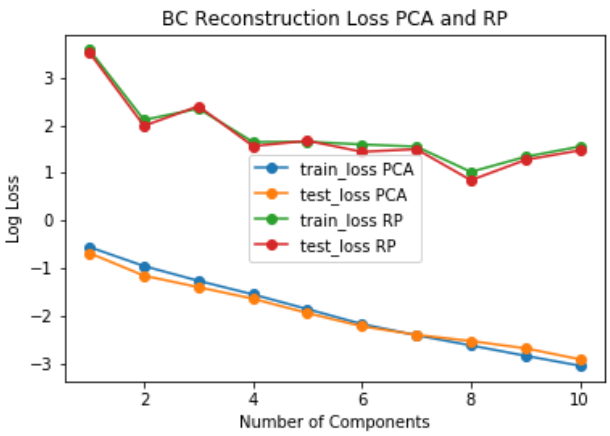


K-means on the neural network was about 10% worse , than using only randomized projections on the neural network. Hence, there's a loss of too much important information with K-means. However, EM on the neural networks had about an equal score to only randomized projection on the neural network.





Reconstruction Randomized Projections and PCA:



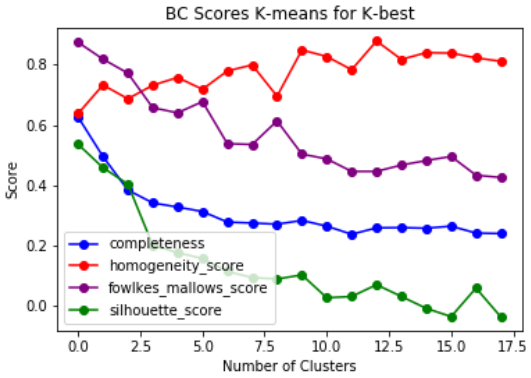
The loss is graphed for the breast cancer dataset's error in reconstruction. The PCA reconstruction significantly outperforms randomized projections. This is expected because it takes significantly longer to compute PCA, but by calculating SVD we are preserving more information than randomized projections.

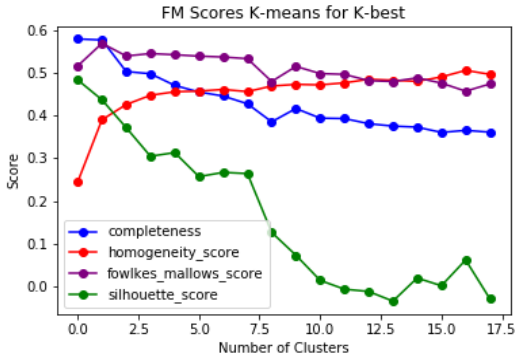
Select K-best:

For this implementation of K-best, I selected the top 10 features which had the highest ANOVA f-value.

Select K-best with K-means:

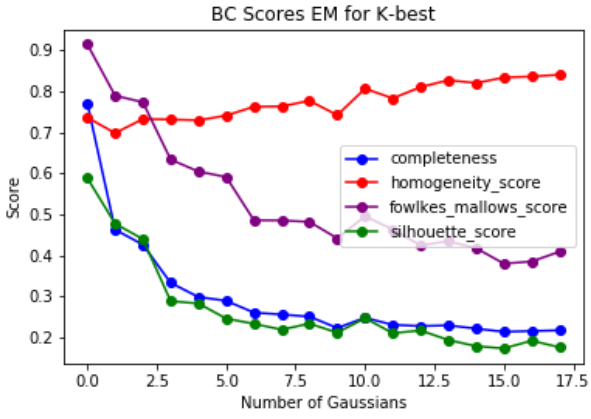
For BC, K-best performed well and is comparable to the other BC K-means scores that we've seen so far. This shows that we can maintain high scores when only using a portion of thefeatures. For FM, using only 10 features we get very low scores. This makes sense because each feature is a pixel, and it hard to glean much information from an individual pixel value.

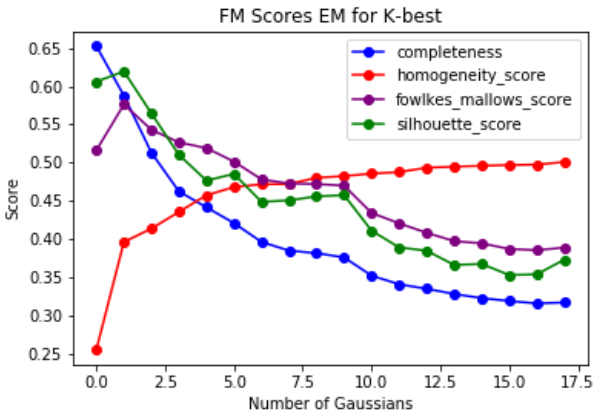




Select K-best with EM:

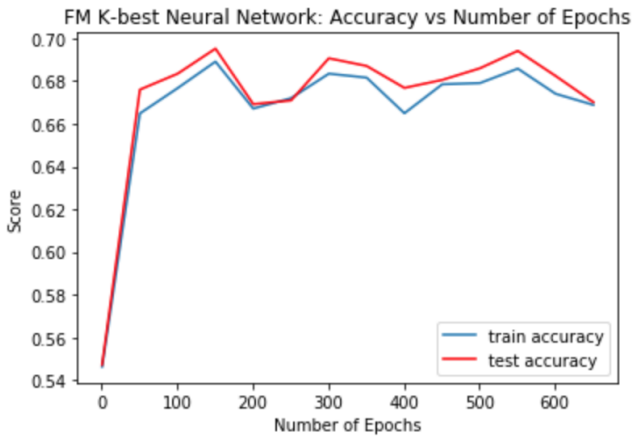
Again, K-best performed very well for BC. For FM compared to K-means the silhouette scores performed significantly better this means that either the inter-cluster distances are greater or the intra-cluster distances are smaller.

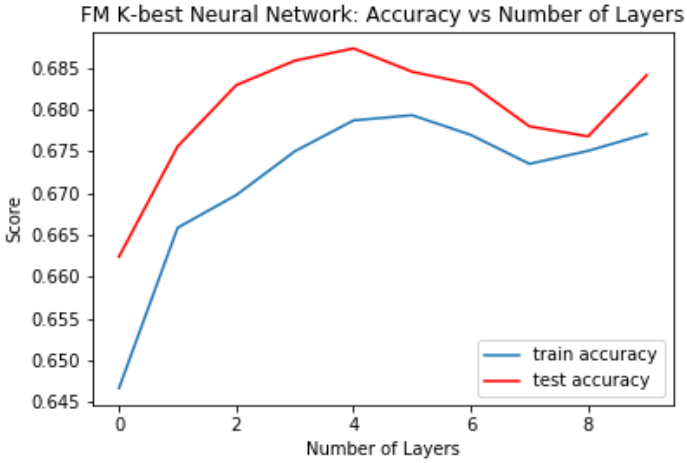




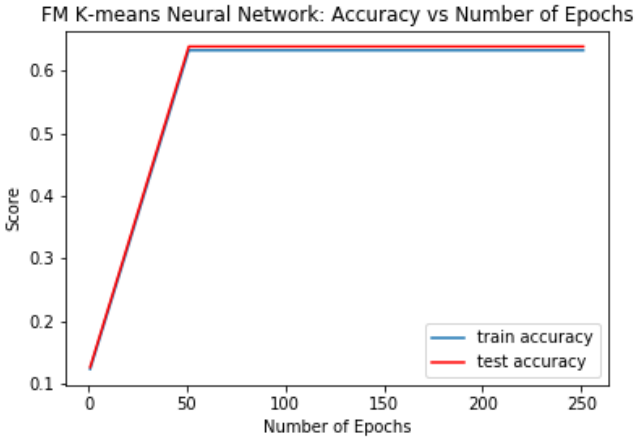
Select K-best with Neural Network:

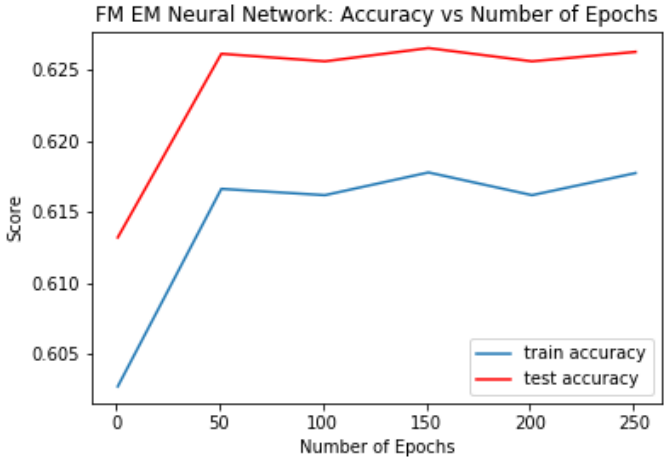
With only 10 features, the neural network performed surprisingly well with a peak about 69% accuracy.





K-means and EM for the neural network have very similar scores after 50 epochs. Additionally, K-means performed very poorly, with only one epoch. This makes sense though because it initialized with bad random weights, and wasn't able to adjust them with back propagation.





Conclusion:

Overall, dimensionality reduction proved successful in reducing the training time for the neural networks. Dimensionality reduction is also good to combat the curse of dimensionality, but the data I was working with wasn't sparse enough to see improvements due to the curse of dimensionality.

For FM, PCA, ICA, and Randomized Projections worked very well and produced results that were very similar to the non-reduced data. Furthermore, if I used more features for K-select I believe I'd have gotten results that were equally as promising. Although the results were very interesting, dimensionality reduction wasn't as beneficial for BC because the dataset was already small to begin with. However, EM scores for BC tended to be higher after dimensionality reduction.