Assignment 3: Unsupervised Learning and Dimensionality Reduction

Ahmad Aldabbagh

November 6, 2016

1. Introduction

This assignment has several parts that explore unsupervised learning. One of the main parts is related to clustering where two main clustering techniques are examined. The second main part is related to dimensionality reduction where four feature transformation algorithms are used to reduce the dimensions of the data sets and apply the clustering algorithms on the reduced data. Other parts include, testing the dimensionally reduced data on neural networks as well as using clustering as a feature selection method.

2. Data Set of Choice

The first data set I have chosen is the same set used in assignment 1 which is the Character Data Set with 20,000 instances. It has been acquired from the UCI Machine Learning Repository and it has 16 features extracted from it. The second data set, has also been used in assignment 1 which is the 'Breast Cancer Wisconsin (Diagnostic) Data Set with 569 instances. Where it has also been acquired from the UCI Machine Learning Repository and it has 30 features extracted from it.

Both data sets have been pre-processed to match the requirements of the programming language of choice and related packages (e.g. order of features & classes), which shall be explained further in the following section. Pre-processing started with changing class labels from strings to numbers. It also included normalizing features which resulted in yielding better results. This would also be noticeable when using algorithms that leverage gradient-descent (many algorithms) to minimize the cost function resulting in a harder path to find the global minimum. Further, the used normalization/scaling was mean-normalization where the mean for a given feature (list of one kind of features) was subtracted from the feature and then the feature has been divided by the standard deviation.

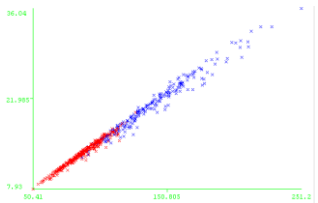
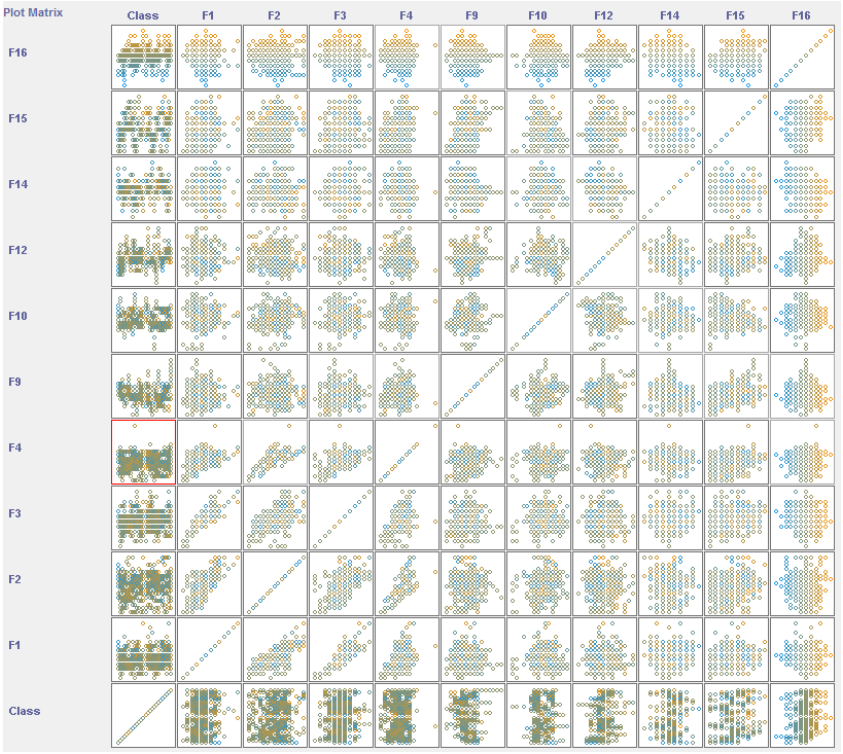


Figure 1: example of correlated two features for the cancer data set

I first started by looking at the features of both of my data sets to see how correlated some of the features are. If a lot of them were highly correlated (as shown in figure below), I would expect that the feature reduction algorithm can effectively reduce the data. Resulting in (k) features that are substantially less than the original (m) features.



This figure is a subset of the features of the character data set. It can be seen that there are some correlated features that possibly can be removed without affecting the performance of the learning algorithm. For example, F3 and F1 as can be seen are correlated with a distribution along the diagonal. Such relations will be of interest when examining feature selection.

3. Clustering of Data sets

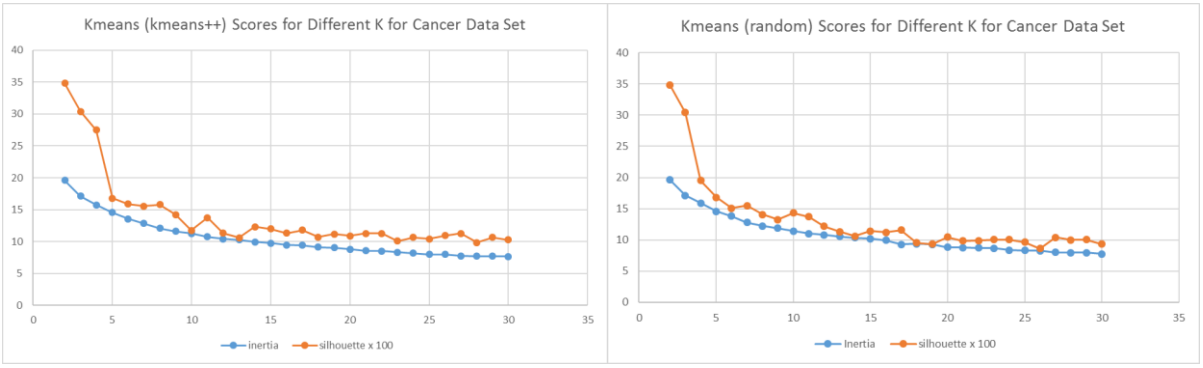
The development language that have been used is Python. Specifically, sklearn has been used for both clustering algorithms. For both data sets, kmeans and expectation maximization clustering algorithms have been used with a different number of clusters. Various metrics have been looked to evaluate the clustering results.

3.1 K-means Clustering

This algorithm clusters the data into 'k' groups of the same variance while trying to minimize the sum of squares of the cluster or the inertia. In my experiments, I have varied the value of 'k' for both data sets. I have also tested it on data sets with different sizes and noticed that it still can perform well with large data sets as well as larger values of k. As with any other algorithm, k-means has both advantages and disadvantage. For example, it does well if the clusters are convex. However, if they were stretched out or irregular, it is not expected to perform that well.

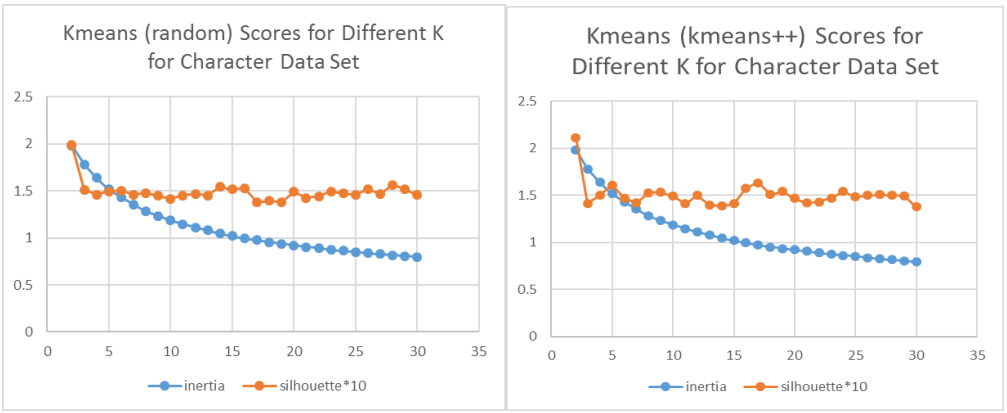
Different metrics have been used to evaluate the performance of kmeans for different values of 'k' on both data sets. The metrics include: inertia, homogeneity score, completeness score, V measure, adjusted Rand index, adjusted mutual information and silhouette coefficient.

For the cancer data, the value of k has been varied from 2 to 30 as well as using two different methods to choose the cluster centers where one speeds up convergence (k-means++) and the other chooses randomly from the data (random). The graphs below shows their performance for different values of 'k' evaluated using two metrics which are inertia and silhouette score. Both these metrics are unsupervised in the sense that they are both not accessing the labels of the data to evaluate the clustering.



It can be seen in the graph above that the highest scores achieved are at about k =2 and k=3 then there is an abrupt negative change in the performance. However, following the elbow method a value of k=10 or 11 is more reasonable. For such a data set with only 2 classes it would be expected to have the best results when k=2 in case the data is linearly separable and the classes aren't intertwined. However, upon looking at the purity of the clusters in the training set it is only about 30-40%.

Similar evaluation metrics have been used for the character data set. As for the results, k=2 also yielded the highest scores where the performance stabilizes for values beyond k=3. The graphs below show this performance.



In the case of the character data, it would have been assumed that k would be optimum for the value of k=26 if samples of the same class are close to each other with some separation/distance between points of different classes. However, given the large number of samples and their closeness to each other this is not the case. This could have also

been inferred through examining the complexity of the models used in assignment 1 to model the function with the lowest error which is definitely non-linear. As shown above the best value is at around k=5 where there is some sort of an 'elbow'. Runtimes are as follows for both k-means++ and random k-means:

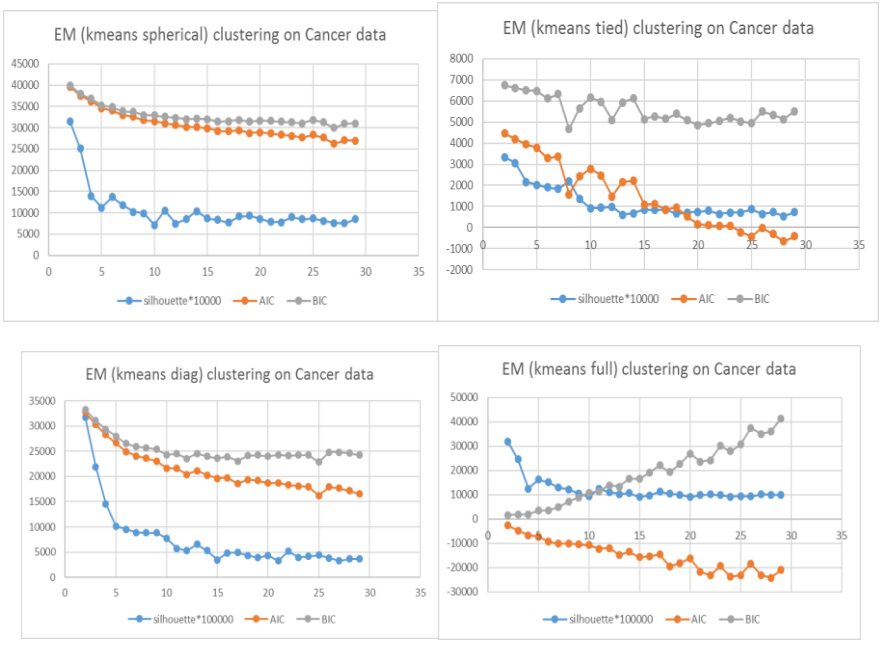
The runtimes show a linear trend with respect to the number of clusters as well as the number of samples (569 for cancer vs. 20,000 for character).

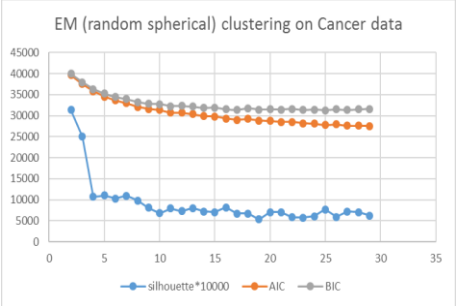
3.1 Expectation Maximization (EM) Clustering

For EM clustering, experiments similar to that of k-means have been run by changing various parameters of the algorithm. 'k' has been changed from 2 to 30 for both data sets, the method to initialize the weights and the type of covariance parameters to use:

- Full: each component has its own general covariance matrix
- Tied: all components share the same general covariance matrix
- Diag: each component has its own diagonal covariance matrix
- Spherical: each component has its own single variance

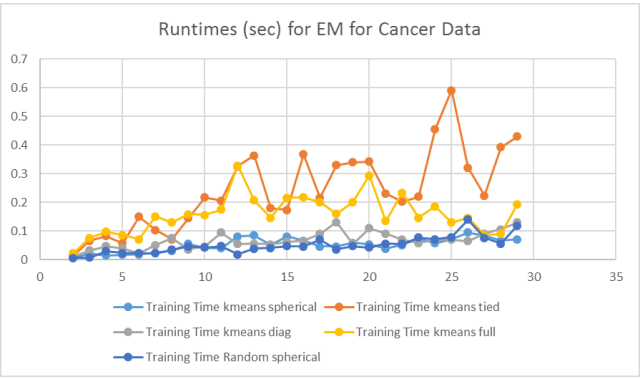
The following results were obtained for the cancer data set:





As seen above, a similar overall value of k is observed when looking at the silhouette metric which is around k=5 or k=6. This is almost the case for BIC (Bayesian information criterion) and AIC (Akaike information criterion) but in some cases they are not consistent. As for the runtimes, they seem to be linear where some parameters take more time than others. Also, the increase in number of clusters did not increase the times significantly. Due to the consistency of silhouette over BIC, I decided to use kmeans over EM for clustering my data after feature reduction.

The runtimes are shown below:

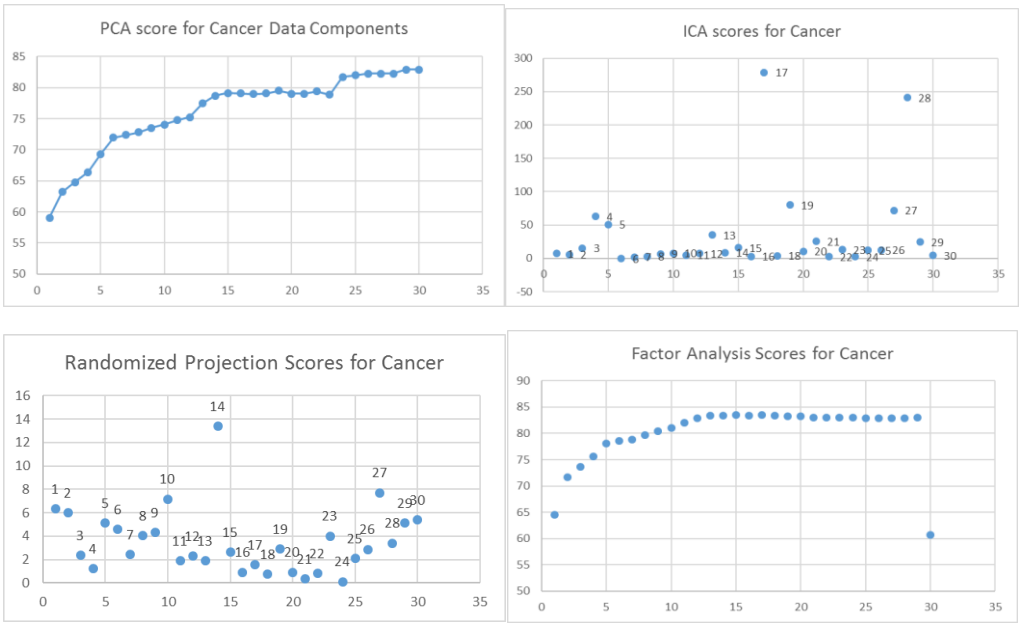


For conciseness and the assignment limit, I had to omit the graphs pertaining to the character data set. However, I will comment on it. For the various parameters, It was noticed that best k was at around 15 with a less apparent elbow. Runtimes followed a trend similar to that for the cancer data set but it took longer due to the larger size of the second data set. The main outtake other than seeing some trend for two different algorithms is determining a suitable number of clusters in an unsupervised fashion(no labels). I however tried to look at how the data aligns with the clusters for the cancer data set when k=2 which resulted in 40% alignment.

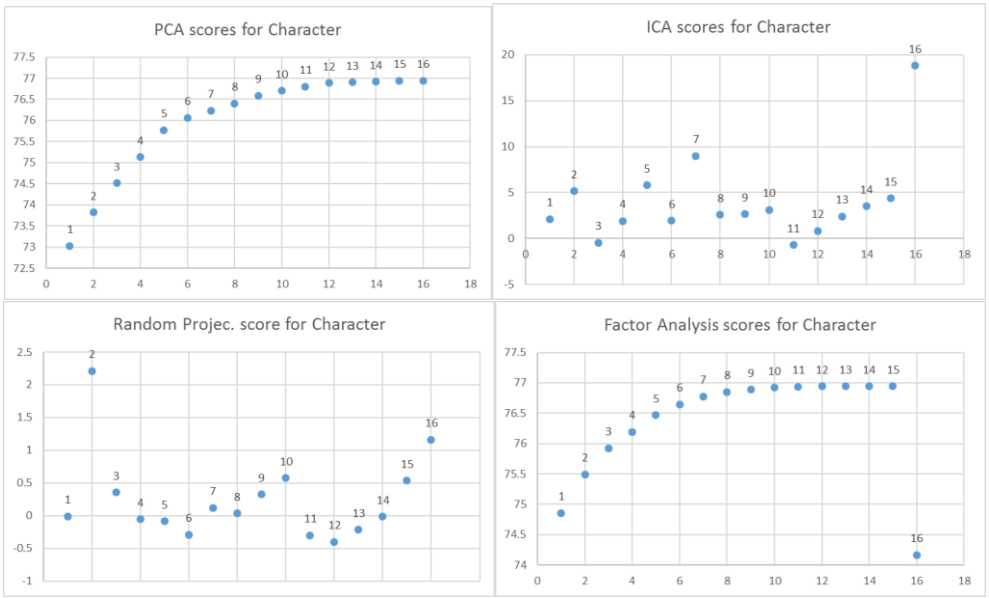
4. Dimensionality Reduction

Four algorithms have been applied on both data sets to reduce the dimensions of the data and they are: PCA (log likelihood), ICA(kurtosis), Randomized Projections (kurtosis) and Factor Analysis (log likelihood) which is a "simple generative model with Gaussian latent variables" (sklearn definition). Each algorithm was run for 'n' times equal to the number of features in the data set where each iteration one component is added.

Different criteria have been used to evaluate different algorithms (as shown between brackets above) and the following shows the metric for each component of each of the algorithms:



It can be seen above how each component performs where higher values correspond to better components. The runtimes for all algorithms was pretty small except for factor analysis which took substantially longer time to run. The following shows the results for the character data set:



In this data set, the runtime for factor analysis was also significantly higher than the other algorithms.

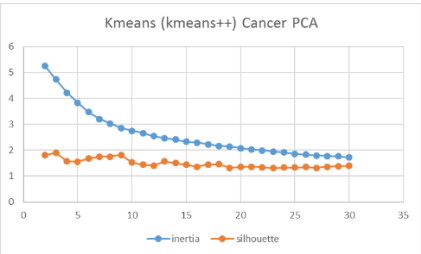
To determine the number of components in each method, I decided to look for gaps between the scores of each component (as shown in the graphs above) and select those

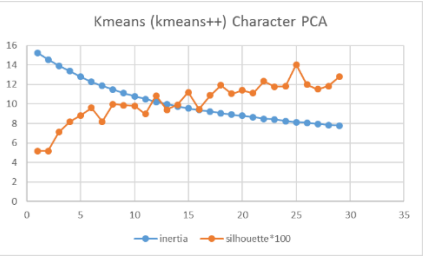
components with highest scores. In the case no clear gap is present like in factor analysis, I chose the top 6 features except for ICA where 10 were used. I found previously in the cancer data using information gain that 4 features are sufficient to produce good results which suggests that maybe 6 is not a bad choice. An important observation here is the resemblance of PCA's results to that of Factor Analysis and this is expected since both algorithms are similar except that one looks at the variances and other looks at the covariances or correlations between features.

5. Clustering on Reduced Data

In this section, I decided to stick with k-means for clustering the reduced data which resulted in similar results to what was obtained previously. Runtimes were generally pretty insignificant except for the case of Factor Analysis which took more time than other data sets. The clustering algorithms have been applied on the reduced data sets after choosing the top 'n' feature/components which are denoted next to each data reduction below.

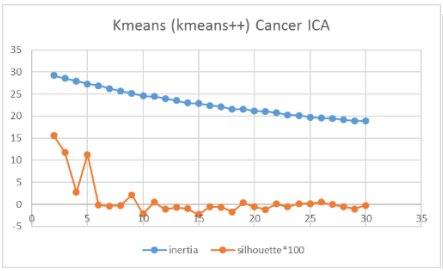
PCA (6 components for both):

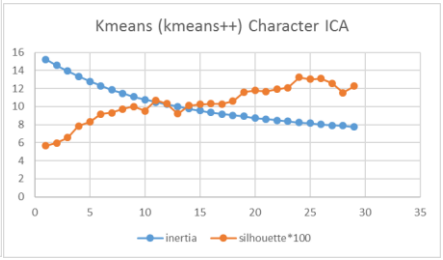




The steady state is achieved in about k=5 for the cancer data which was similar to what was observed earlier. However, it is not as apparent in PCA as it was previously. In the case of the character data set, k still appear to somewhat stabilize at 15 (this can be clear if we smoothed the signal using MA filter).

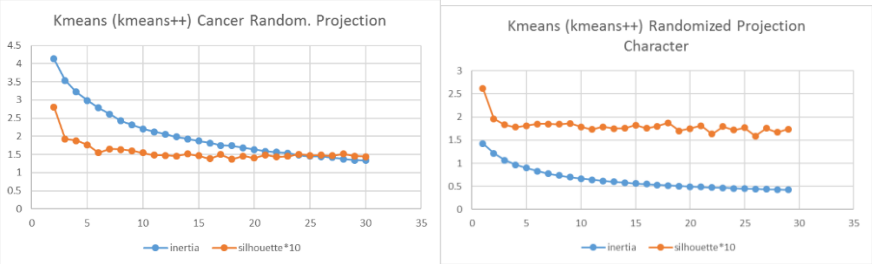
ICA (10 components & 10 components for char):





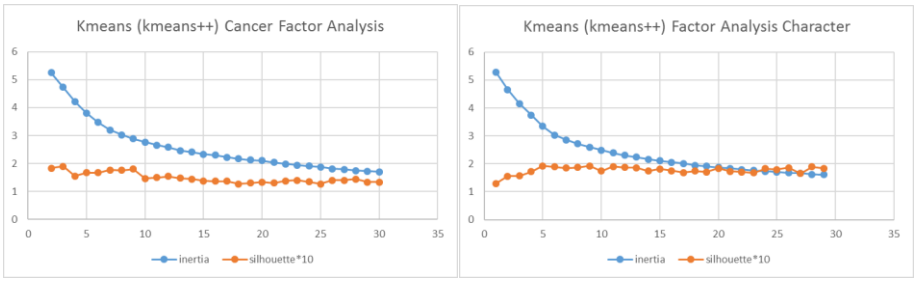
For ICA, a value of k=6 is needed for the cancer data to reach the elbow point which still pretty close to what we got earlier. As for the character data, it is still around 15.

Randomized Projection (6 components 7 components for char):



Again, randomized projection also requires a value of k = 6 to reach the steady state for the cancer data. For the character data set, it had an interesting change, where k got reduced significantly from around 15 to 5 due to the reduction of the dimensions in the new data set.

Factor Analysis (6 components for both):



Finally, factor analysis exhibits a behavior that is similar to PCA which was also observed in the previous section. The other data set also exhibited the same behavior where it seems to stabilize at k=10.

6. Neural Networks with Reduced Data sets

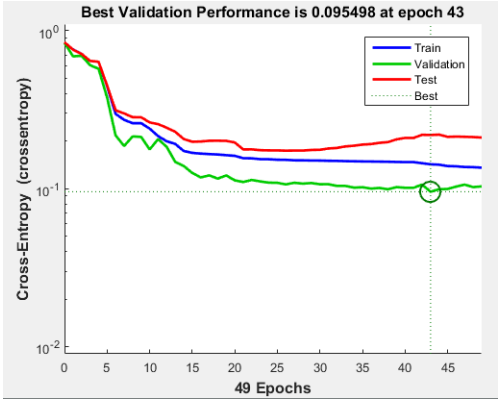
The data set I decided to use is the cancer data set where I run model selection by changing the single layered network size from (2 to 30) and change the regularization parameter alpha along the use of cross validation. The reason I went with one hidden layer is because that proved sufficient on the original data set which only required a single layer. Furthermore, learning curves for the best model were plotted to ensure that no over fitting has taken place. The neural network was modeled again for the original data set since I used sklearn in this assignment instead of MATLAB which was used in assignment 1.
The table below illustrates a summary of the results:

Data Set	Score	Configuration	Train Time	Number of Features
Original Cancer	93.42 %	8 hidden / 2 output	0.303 s	30
PCA Cancer	95.15 %	5 hidden / 2 output	0.031 s	6
ICA Cancer	93.60 %	9 hidden / 2 output	0.086 s	10
RP Cancer	91.63 %	9 hidden / 2 output	0.026 s	6
FA Cancer	92.95 %	8 hidden / 2 output	0.063 s	6

As shown in the summary of results, the reduction of dimensions resulted in significant improvements in training time which in some case were reduced to $1/10^{th}$ the time required for the original data set. As for classification score, some algorithms improved the score while others resulted in a slightly lesser score but with smaller train time. This tradeoff is extremely useful if we were dealing with a large data set and will run many

iterations and being able to reduce runtime by 1/10 with significantly affecting performance is extremely beneficial.

Another experiment I decided to run is it to evaluate my dimensionally reduce data sets in MATLAB to double check the consistency of my solutions. For example The following learning curve was obtained for the PCA data set (note that the same was calculated during training the models above but was omitted for conciseness):



The performance of the network was quite close to what I got using sklearn where my cross validation error was %3.5 and my testing accuracy (classification score) was 94.37% which was close to the performance obtained in sklearn. A possible difference could be due to the initial conditions. Also, MATLAB's stopping is a bit different than sklearn.

7. Neural Networks with Clustered Reduced Data sets

In this section the clustering algorithms were applied on the cancer data set after applying the dimensionality reduction algorithms. The number of adequate number of clusters have been identified in the previous section for the cancer data as k=5. This has been used for clustering. After that, clusters were appended to the previous features as new features.

After the new sets were created, the neural network learner was applied on the data set with cross validation and model selection to determine the best configuration. The following results were obtained:

Data Set	Score	Configuration	Train Time	Number of Features
PCA Cancer w/clustering	96.47 %	9 hidden / 2 output	0.08 s	6+5
ICA Cancer w/clustering	96.03 %	6 hidden / 2 output	0.297 s	10+5
RP Cancer w/clustering	90.74 %	11 hidden / 2 output	0.069 s	6+5
FA Cancer w/clustering	95.15 %	12 hidden / 2 output	0.095 s	6+5

As can be seen from the table, in general the performance increased in all algorithms except for the case of RP. While the accuracy increased, it is important to note the increase in train time due to the additional features but it still is smaller than the original data. This shows a neat technique to remodel the data set by transforming the

original data set and trying to capture the sparsity of it using clustering. By combining both techniques, a good balance between high accuracy and speed can be achieved.

References

[1] https://archive.ics.uci.edu/ml/index.html
[2] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29
[3] http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation
[4] http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.FactorAnalysis.html
[5] http://stats.stackexchange.com/questions/50537/should-one-remove-highly-correlated-variables-before-doing-pca