James Liu
CS 4641, Spring 2014, Isbell
Unsupervised Learning and Dimensionality Reduction Assignment

Dataset Description

The two datasets used in this assignment are the same two used in assignment 1: spambase and breast-cancer. spambase is a dataset of word and character frequencies in emails and whether or not the email was considered spam by the user. It is the exact kind of dataset one would expect great improvement from dimensionality reduction algorithms: it has a large number of attributes, 58 in total, that are discrete and highly kurtotic in distribution, and many of them may be redundant/irrelevant. The classification of whether it is spam or not is directly dependent on a small subset of attributes, each of which provides a significant amount of information on the classification of the instance. Breast-cancer is a smaller dataset with only 10 attributes with discrete/binary values, most of which are well distributed and have very little skew in their distributions. This serves to be a standard of comparison for spambase. Clustering algorithms are expected to perform better on it than spambase, but dimensionality reduction algorithms should worsen performance on it, as there should be little covariance between each attribute. spambase has 4601 instances. Breast-cancer has 286.

Initial Clustering Experiments and Results

For the purposes of this analysis and to maintain consistency for the experiments done, both k-means and expectation maximization used Euclidean distance measure as a similarity measure between instances. Manhattan distance may have worked better for breast-cancer, due to

For the purposes of testing k-means and EM, I chose to use a k equal to the number of possible values the nominal class attribute for each dataset. This will test to see how well the clusters created by k-Means and EM matches the given classes. The clusters were compared to classes by use of Weka's "classes to cluster" evaluation, which simply looks for the best matched class for each cluster, which does not run well with a large number of clusters/classes. Thus why I chose to use datasets with binary classifications. I had also tried datasets like a large number of classes, which took too long to test using this method and was also difficult to visualize the result due to the increased number of clusters. This test method's only result is how many of the best fit cluster match the given class as a percentage of the database, which will be used as the main performance metric for this analysis.

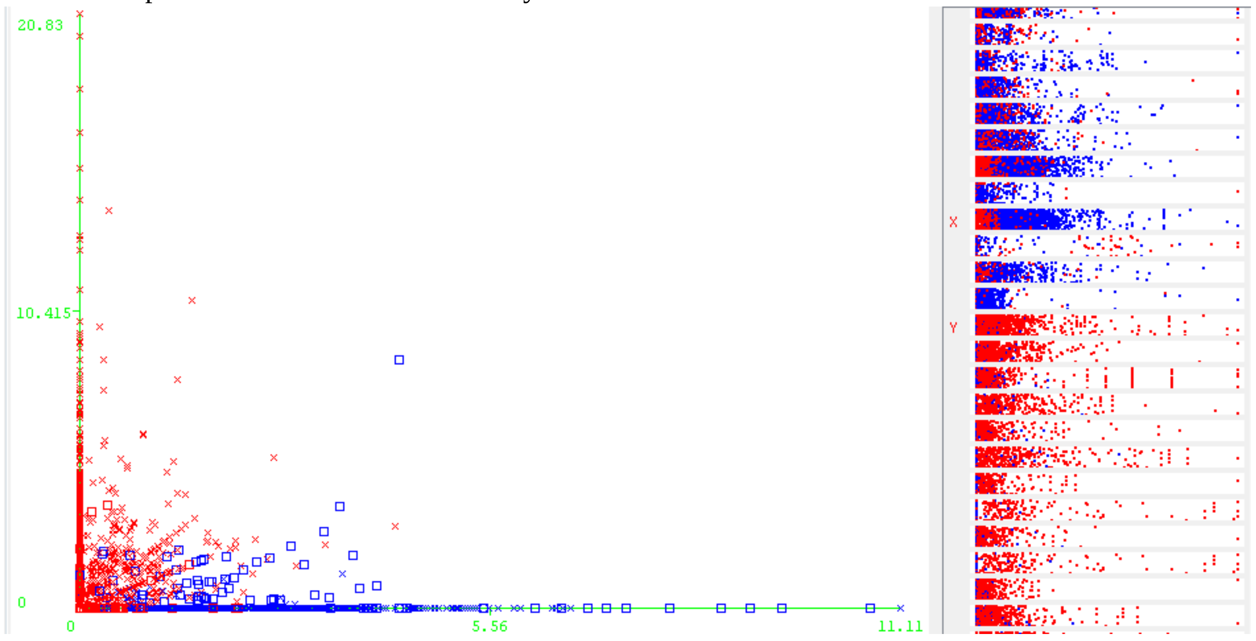


Figure 1. The k-Means results plotting instances based on the frequency of the word “your” (X axis), and the frequency of the word “hp” (Y axis). Red are those in cluster 1, blue are those in cluster 2. Note the clear distinction between both clusters and little overlap. On the left is the relative distribution, and cluster assignment of instances relative to certain attributes. Note the distribution of each cluster relative to said attributes.

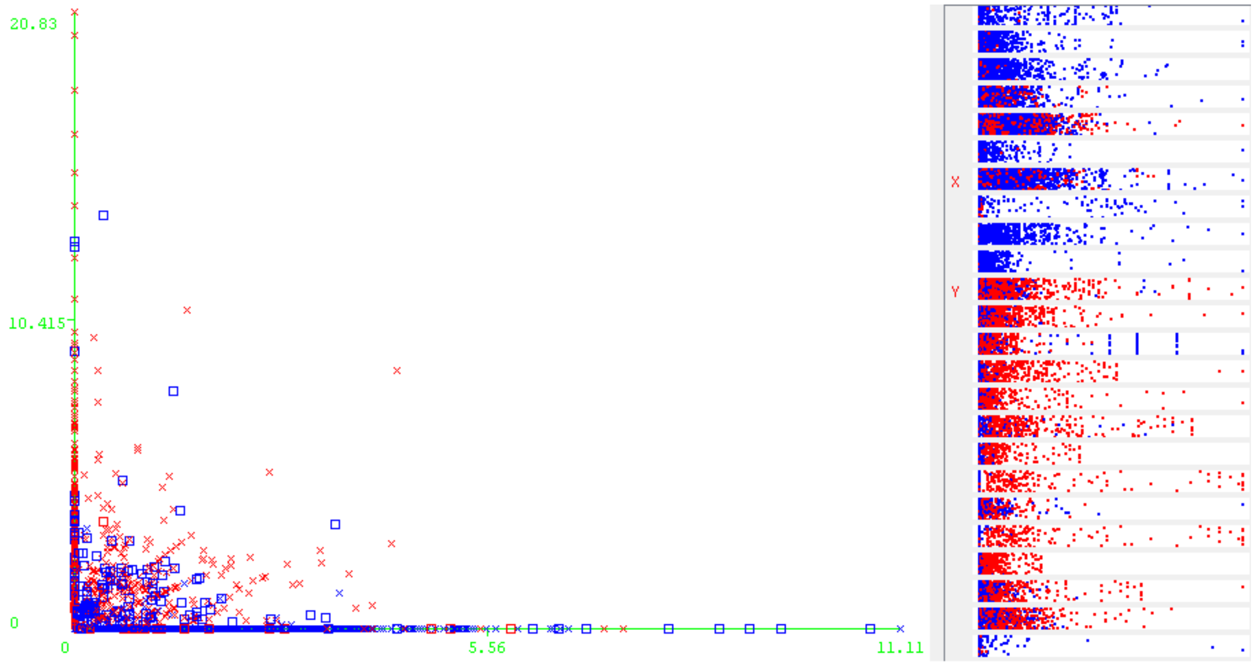


Figure 2. The results of expectation maximization plotted in the same format as the one for k-means above. Unlike k-means, there is no clear distinction between either cluster in this plot and the attribute-class distributions to the right, particularly the attribute labeled with “X”.

Dataset	Clustering Algorithm	Time Elapsed	% Incorrectly Clustered
spambase	k-Means	0.39s	20.0826%
spambase	EM	4.51s	21.7996%
Breast-cancer	k-Means	<0.01s	25.5245%
Breast-cancer	EM	0.03s	49.3007%

Dataset size and number of attributes directly affects the runtime of these algorithms, as evidenced by the difference in time taken to process spambase and breast-cancer.

Spambase’s results were unexpected. Since most of the attributes have a high kurtosis and are heavily skewed to the right, I expected it to be difficult for either algorithm to generate clusters that matched up with the classes so well, since most of the instances have very high similarity and thus are very close together, save for a few major outliers. K-means also did slightly better than EM did in matching clusters to classes. As seen in Figure 1, K-Means seems to have made clear distinctions between each cluster. The plot between “hp” frequency and “your” frequency shows a clear line of division between the two clusters with very little overlap of the clusters. EM, on the other hand shows no significant distinction and significant overlap in not only the two plotted attributes, but many others as well, as seen in Figure 2. This is because of the way EM handles whether an instance is in a cluster or not. Unlike K-Means which explicitly sets an instance to a cluster, EM calculates the probability that each instance is in an each cluster and chooses the most likely one. K-Mean’s better accuracy and overall way it clustered the dataset suggests that spam mail can be determined fairly easily by the presence of only a small subset of keywords, or complete lack thereof.

Breast-cancer's performance is also unexpected. However it can be explained by the fact that the tests I ran look for a total of 2 clusters, when in actuality there are far more than two easily separable clusters in the dataset due to the discrete and evenly distributed nature of it attributes, whereas spambase really only has one main cluster and a large number of outliers. Our domain knowledge applied to it was in a sense not well suited for the problem, and it resulted in lower performance. Expectation maximization did particularly worse than k-Means for this very reason.

Both algorithms should work better when attributes with small variance and small covariance with other attributes, basically redundant or irrelevant attributes, are removed. As this reduces the amount of work needed to compute distances between instances, and potentially accuracy of the clustering algorithms.

Dimensionality Reduction Experiments and Clustering Results

Principle Component Analysis

Dataset	Clustering Algorithm	Variance Covered	# of result attributes	Log Likelihood	Time Elapsed	% Incorrectly Clustered
spambase	k-Means	0.25	6	-	0.08s	19.0828%
spambase	k-Means	0.5	17	-	0.28s	17.9092%
spambase	k-Means	0.75	32	-	0.42s	18.5829%
spambase	k-Means	0.95	49	-	0.81s	20.7564%
spambase	EM	0.25	6	-7.52479	0.85s	48.1852%
spambase	EM	0.5	17	-21.31495	4.59s	47.3375%
spambase	EM	0.75	32	-37.4544	7.82s	43.1645%
spambase	EM	0.95	49	-55.24697	10.63s	42.2734%
Breast cancer	k-Means	0.25	5	-	<0.01s	28.3217%
Breast cancer	k-Means	0.5	12	-	<0.01s	27.972%
Breast cancer	k-Means	0.75	20	-	0.01s	27.6624%
Breast cancer	k-Means	0.95	29	-	0.01s	36.3636%
Breast cancer	EM	0.25	5	-7.11398	0.02s	46.1538$%
Breast cancer	EM	0.5	10	-18.26259	0.06s	44.0559%
Breast cancer	EM	0.75	20	-28.29255	0.09s	36.014%
Breast cancer	EM	0.95	29	-38.86653	0.11s	36.3636%

For spambase, the eigenvalues generated by PCA were very heavily skewed, with 3 linear combinations of attributes in the 2-6 range and the rest in the 0-1 range. From this we can see that only a few axes for which the dataset shows actual variance. The rest show very little variance at all, and thus are redundant or irrelevant, which confirms our initial assumptions on the dataset.

The results for breast-cancer seem odd, as it seems to have increased the number of attributes rather than decreased them. This is likely because of how Weka's PCA implementation handles nominal attributes: it transforms them into a set of binary attributes before performing PCA. This results in a large number of extra attributes that weren't there before. Since almost all of the attributes in breast-cancer were nominal, the number of actual attributes used was 35. With this in mind, the distribution of breast-cancer was rather close to that of spambase's, albeit a bit less skewed.

Reducing the number of dimensions directly decreased the amount of time it took for the clustering algorithms to run. While it did improve some runs, PCA generally worsened the performance the clustering algorithms had on spambase and breast-cancer, especially the EM tests on spambase after using PCA. The decreased in performance in EM after PCA can be attributed to PCA reducing the effect extraneous attributes had on the distance calculation. With PCA, differences in certain attributes are more pronounced, which made it harder for EM to assign the correct probabilities that a certain instance belonged to a certain cluster.

Random Projection

Dataset	Clustering Algorithm	# final attributes	Time Elapsed	Error (seed=42)	Error (seed=82)	Error (seed=100)
spambase	k-Means	10	0.22s	42.12%	19.13%	36.49%
spambase	k-Means	20	0.28s	17.90%	35.82%	36.45%
spambase	k-Means	40	0.42s	18.58%	44.16%	36.32%
spambase	EM	10	3.2s	29.58%	32.23%	31.86%
spambase	EM	20	4.59s	47.33%	29.88%	30.38%
spambase	EM	40	7.82s	43.16%	29.82%	30.17%
Breast cancer	k-Means	4	<0.01s	49.3007%	45.4545%	47.5524%
Breast cancer	k-Means	8	<0.01s	49.3007%	41.958%	25.5245%
Breast cancer	EM	4	0.01s	46.1538%	47.9011%	47.2028%
Breast cancer	EM	8	0.03s	49.3007%	46.1538%	48.951%

The random projection tests are interesting. I tried projecting the result of random projection back into the original space and the resultant attributes' distributions have similar kurtosis and skew, but the mean of data has definitely been translated and the variance has changed significantly.

Once again, like PCA, randomized projections tends to, on average, produce worse results in terms of the output of clustering algorithms. The performance seems to vary depending on the initial number of attributes, as evidenced by how much larger the variance of the spambase results are compared to those of the breast-cancer tests, save for one outlier. The variance in clustering performance after RP seems to increase as the number of final attributes after RP decreases. EM also seems to be more consistent than k-Means due to how it clusters, but does worse, on average, than k-Means with the same datasets.

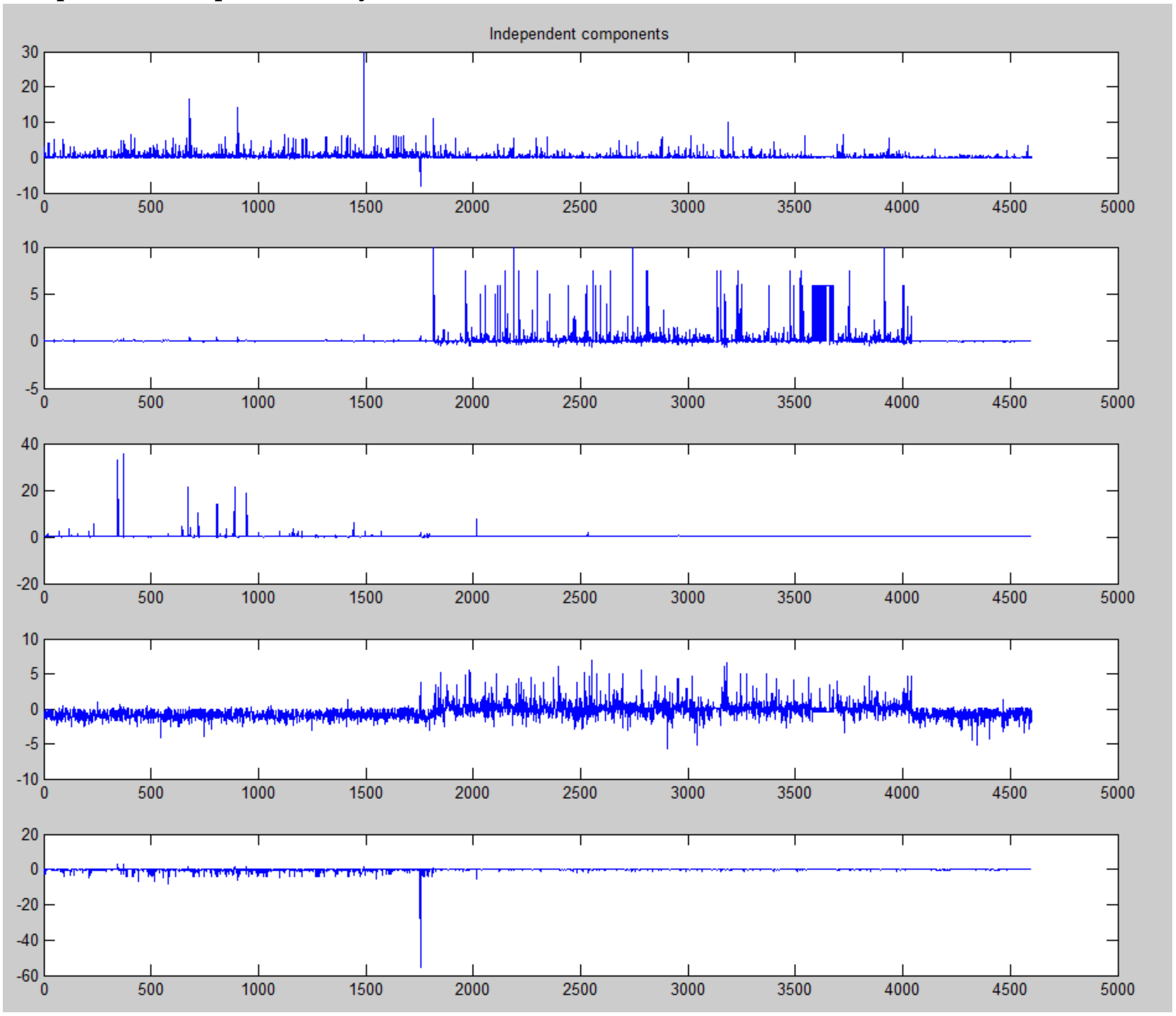
Information Gain

Dataset	Clustering Algorithm	Variables Removed	# of result attributes	Log Likelihood	Time Elapsed	% Incorrectly Clustered
spambase	k-Means	5	53	-	0.65s	22.9298%
spambase	k-Means	15	43	-	0.52s	22.9298%
spambase	k-Means	25	33	-	0.41s	22.9298%
spambase	k-Means	35	23	-	0.22s	22.8221%
spambase	EM	5	6	-29.65018	3.84s	19.9522%
spambase	EM	15	17	-28.37427	3.25s	26.1682%
spambase	EM	25	32	-25.56246	2.74s	15.2358%
spambase	EM	35	49	-23.19093	1.81s	14.8011%
Breast cancer	k-Means	2	8	-	<0.01s	25.5245%
Breast cancer	k-Means	4	6	-	<0.01s	27.664%
Breast cancer	k-Means	6	4	-	<0.01s	30.0699%
Breast cancer	k-Means	8	2	-	<0.01s	27.972%
Breast cancer	EM	2	8	-7.58981	0.01s	27.972%
Breast cancer	EM	4	6	-4.7803	0.01s	27.972%
Breast cancer	EM	6	4	-4.03002	0.01s	27.2727%
Breast cancer	EM	8	2	-1.06491	0s	44.4056%

For this section, I used Weka's InfoGain evaluation function to sort the attributes of each dataset by how much information gain each gives, and then removed the a number of the ones that gave least information gain.

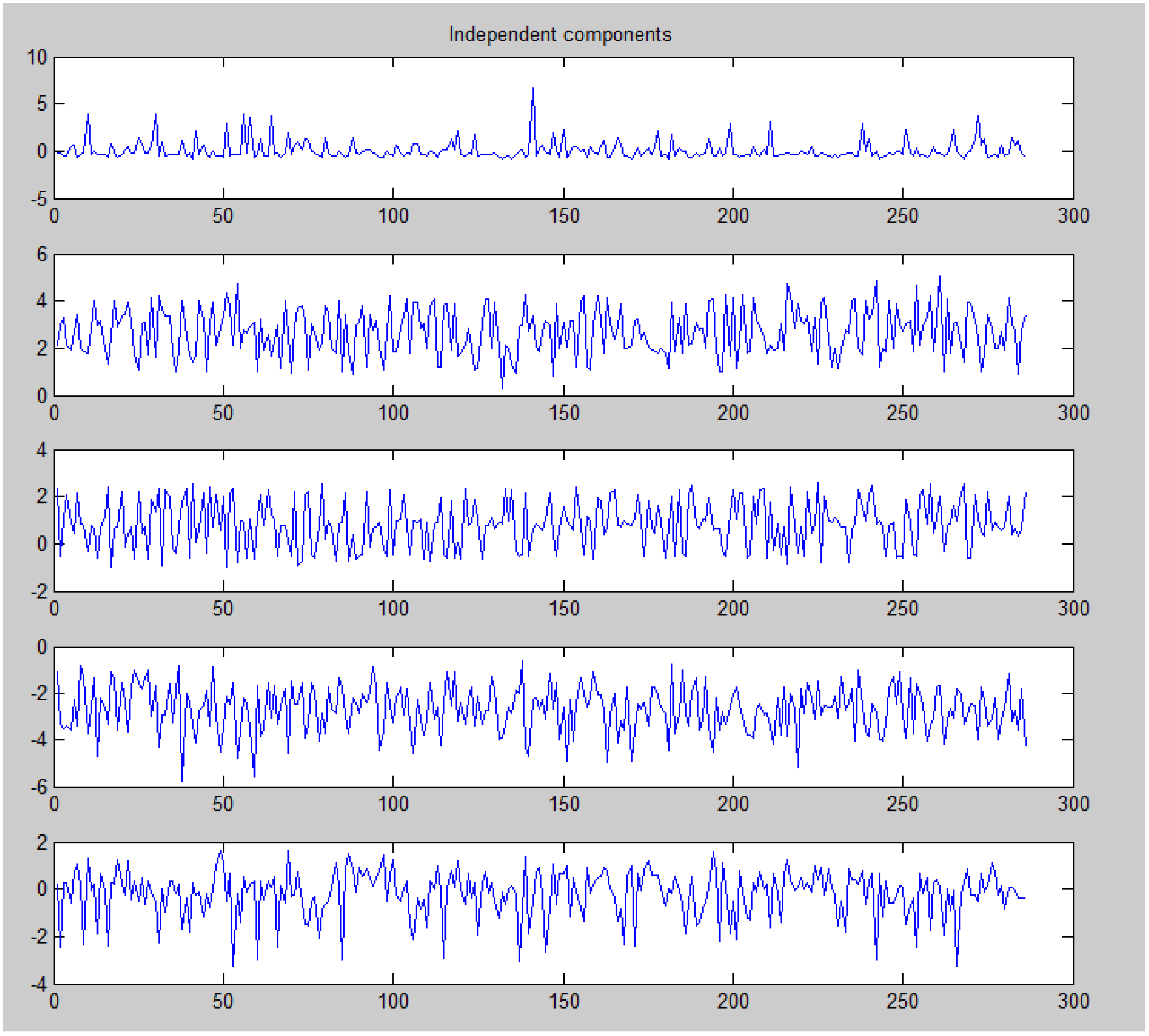
There are a few things interesting to note here, first is the how the number of incorrectly clustered instances when using k-Means on spambase after removing the attributes does not change significantly at all. In fact, the model and clustering did not change until the 35 attribute test. This further supports the idea that there are a good number of irrelevant/redundant attributes in spambase. The second is the sudden jump in error breast cancer with EM at 8 removed attributes. This is because the class attribute is included in the number of remaining attributes, thus there is only one actual attribute used in that test. However, it should also be noted that the k-Means test on the same resultant dataset didn't suffer such a spike in inaccuracy. This is explainable by the fact that the last remaining attribute is nominal and only has three values, two of which are completely one class. K-Means completely took the aforementioned two clusters, but EM split the middle cluster and ended with an increased error.

Independent Component Analysis

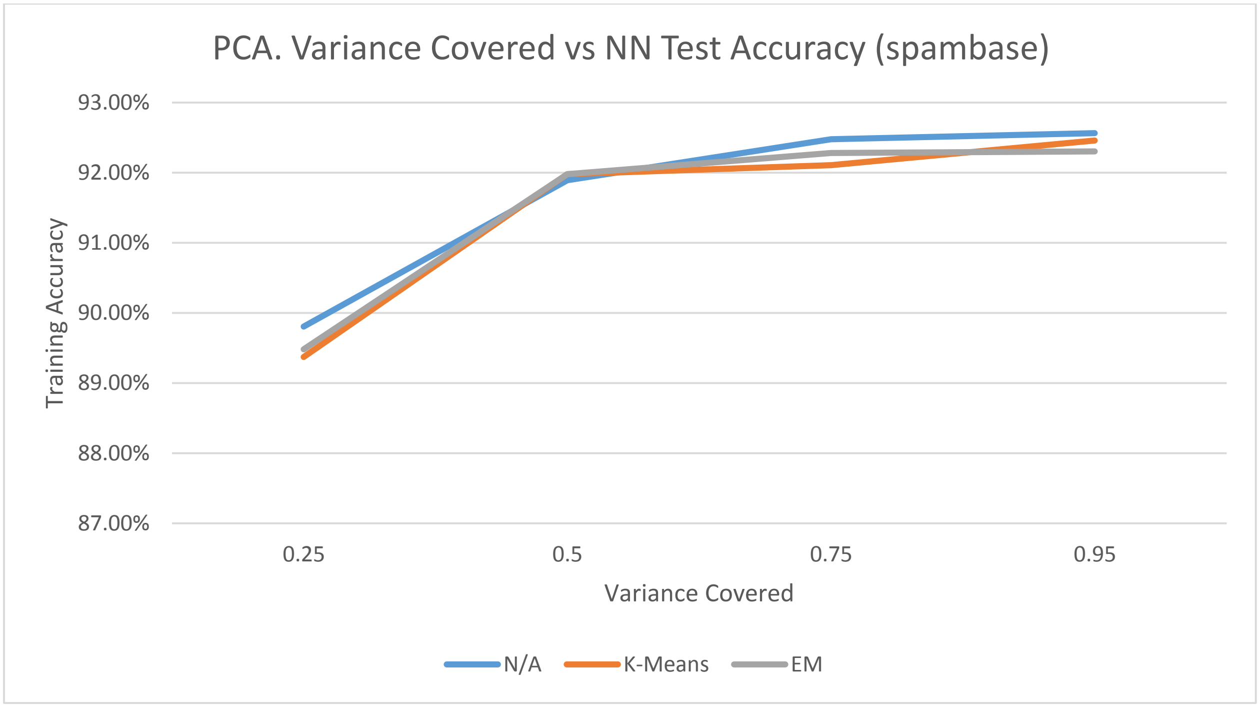


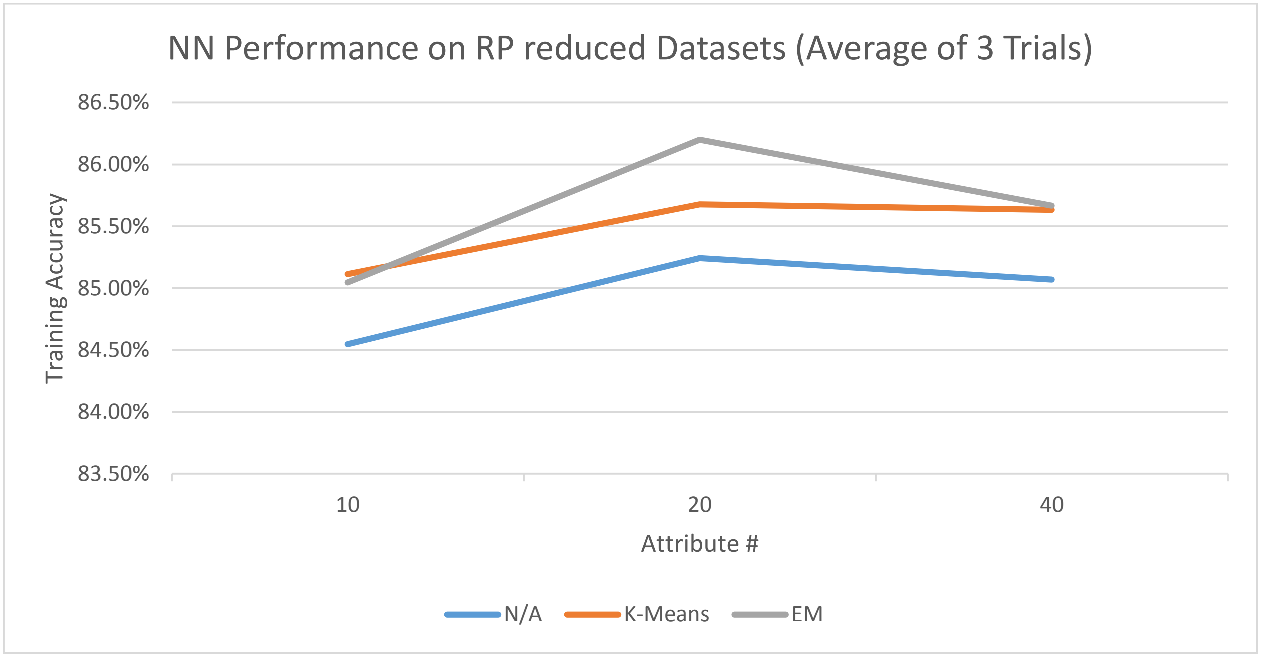
spambase's ICA results graphed against instance #. (5 ICs)

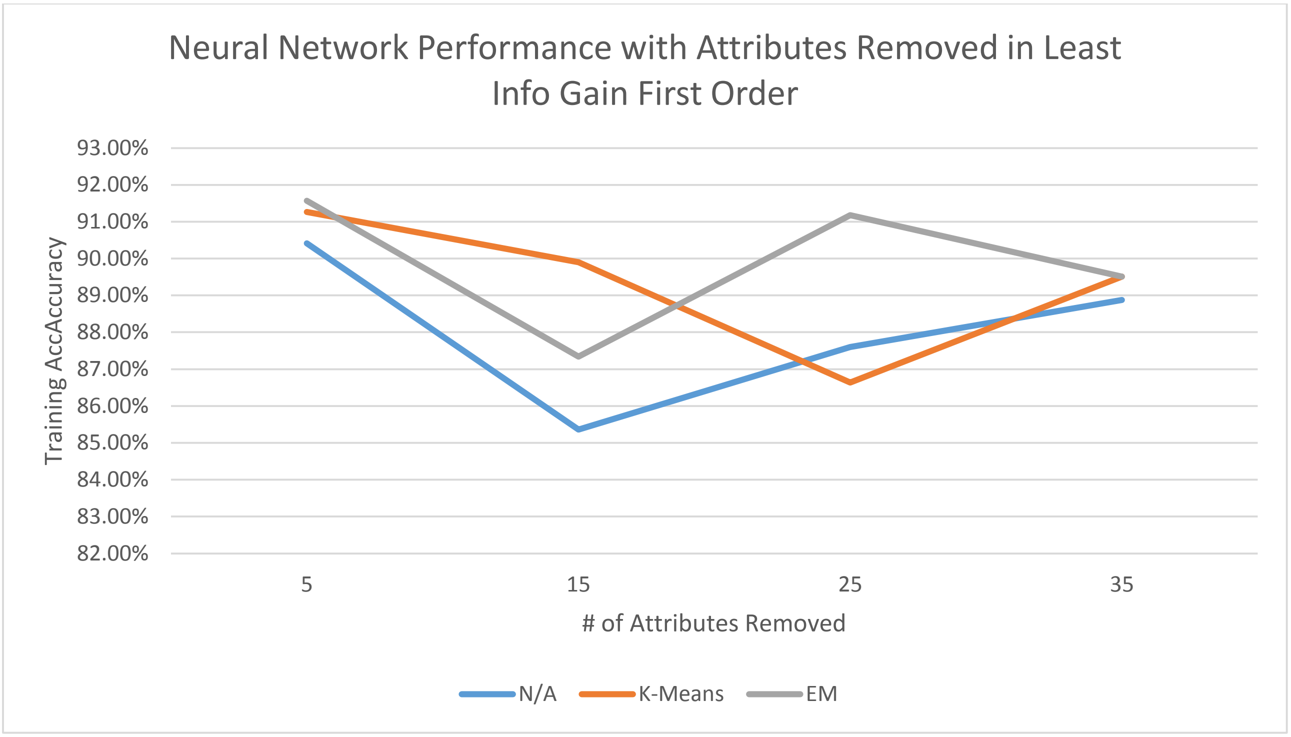
For ICA, I used the FastICA library for MATLAB. As seen above, a few of the ICs had highly kurtotic distributions, with a large number of instances measuring close to zero, and a small minority that is of high value. As the number of ICs computed increased (the results of the 5 ICs test are shown above for spambase), the number of high kurtosis ICs increased. Conversely as the number of ICs being computed decreased, the number ICs with approximately Gaussian distributions like the 1st and 4th ICs shown above increased. When performed on breast-cancer, FastICA tended to produce ICs with distributions that were approximately Gaussian or uniform in nature. This is expected: spambase had a good number of already kurtotic attributes that varied often alongside another attribute, and breast-cancer had uniformly distributed attributes that generally varied independent of each other.



Neural Network Training with Dimensionally Reduced Datasets with and without Clusters as Features







For this section, I used Weka's MultilayerPerceptron with the standard of 500 training epochs, learning rate of 0.3, and momentum of 0.2. For the sake of comparison, the results from that test were a test accuracy of 81.9565% and a training time of 87.6s. These tests were performed with 10-fold cross validation for more accurate measurement of much the cluster/dimensionality reduction algorithms affected the neural networks compared to unfiltered/clustered tests. Unfortunately, as I could not find a way to return the transformed data from MATLAB to Weka, I could not perform NN tests on ICA transformed datasets. The above plots show how well a transformed spambase performed in NN testing. Each line denotes whether there was a cluster attribute added and which algorithm generated the clusters.

General Observations

- Training time is proportional to the number of resultant attributes after dimensionality reduction.
    - For example, PCA with a covariance covered of 0.25 runs for 1.74s (6 attributes). A similar test with a covariance covered of 0.95 took 52.46s (49 attributes) to complete.
    - Adding the clusters did also signfigantly increase the amount of time it takes for the neura
    - All of the run times were shorter than that of the assignment 1 neural network tests of similar parameters (on spambase)
- Testing accuracy tends to (but doesn't always) increase as the aggressiveness of dimensionality reduction algorithms increased, to a point, on spambase. This is likely because of the number of redundant attributes the dataset has. By reducing the number of attributes it has, the performance of spambase with neural nets increased because it decreased the size of the hypothesis space and thus decreased the time needed to converge to a better result. However being too aggressive started to hurt performance, as it starts to remove significant amounts of useful information from the data. On spambase, all of the dimensionally reduced dataset tests performed better than the Assignment 1 test did in terms of both accuracy and training time.
- As I could not find a way to return the transformed data from MATLAB to Weka, I could not perform NN tests on ICA transformed datasets

Principle Components

- The addition of the cluster attribute didn’t do much to affect the performance of the neural network with principle components, which is expected, since PCA already generates attributes in which the dataset is varies most upon. Additional attributes would not contribute much, and may hinder it, as it often did according to the graph.

Random Projections

- It is expected that it performed subpar compared to PCA, as its projections are not optimal due to its random nature. However it still did better overall than the original backpropogation network, as it convereged faster due a lower number of attributes to work with.

Info Gain

- While it did not perform worse than the Assignment 1 standard, the results from info gain are not very consistent. I expected a more consistent change in performance due to only removing the items with low info gain. It may have been a more informative test if I had tested with a smaller interval of removed attributes.