Ryan Reed

Unsupervised Learning – An analysis on Clustering and Dimensionality Reduction Algorithms

0 – Intro

I once again used the UCI Breast Cancer dataset, classification dataset to determine if a tumor is 'malignant' or 'benign'[1]. This dataset was interesting to me as I've had several family members with cancer. Using ML to solve hard hitting problems such as this really intrigues me and reveals the true potential that machine learning has – we can solve so many diverse and complex problems, sometimes to even save lives. The dataset has 10 initial features on a 1-10 continuous scale with a binary classifier. Around 600 records are in this dataset, so not extensively large. From my understanding it's a classic UCI ML dataset so I figured it'd be a good place to start learning these algorithms.

The second dataset was also a UCI classifier which identifies a particular class of the iris plant based on 4 attributes detailing the width and length of the petals and sepals. I chose this over my basketball dataset from assignment one because I really wanted to see if clustering could benefit a dataset with multiple classes (3 in this case). It only has 4 attributes, but should be a good control during dimensionality reduction.

1 – *K*-means Clustering and Expectation Maximization

1.1 – K-means

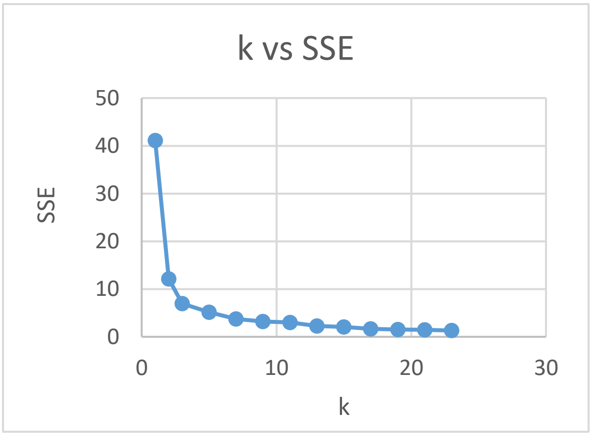


Figure 1.1a – k-means vs SSE within clusters for Iris Dataset

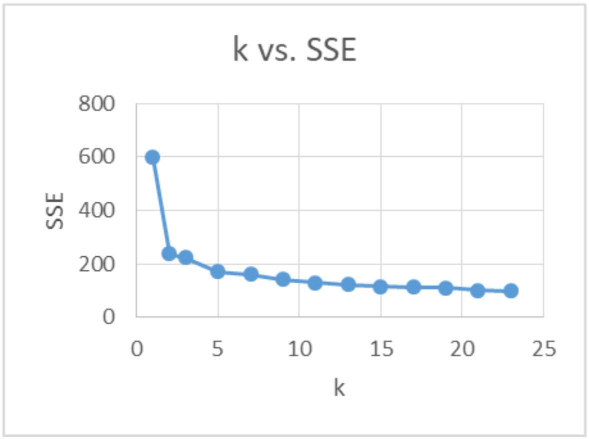


Figure 1.1b – k-means vs SSE within clusters for Cancer Dataset

Based on "elbow theory" [2], we can see that the Cancer dataset has a hard elbow at 2. This is good for our cancer dataset because it matches the number of classes, which may further hint at our classes potentially being linearly separable, make clustering very beneficial. The iris data set is a bit harder to distinguish, with k residing between 2 and 3. If the data were truly unsupervised and without labels, it would likely be beneficial look at silhouette scores to determine a good value of k. However, knowing that there are 3 classes makes a k value of 2 irrelevant because the 3rd class not included in the clusters would always be misclassified. Also, by looking at some of the clusters in two dimensions for some of the attributes, it is easy to see that any more clusters would not be very beneficial as the attributes appear to be very clustered already based on classes (most permutations of attributes looked like this – these graphs display the generated clusters, but still gets the point across as only a few points are misclassified, which are represented by boxes).

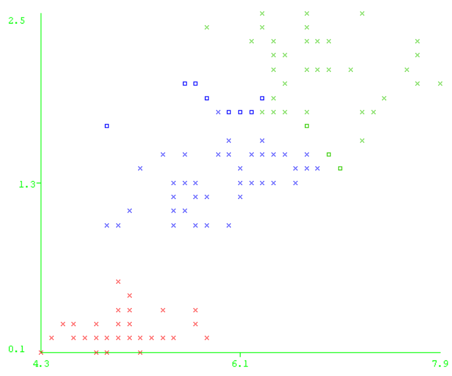


Figure 1.1c – Clusters for sepal length & petal width. Boxes represent misclassified instances (k-means)

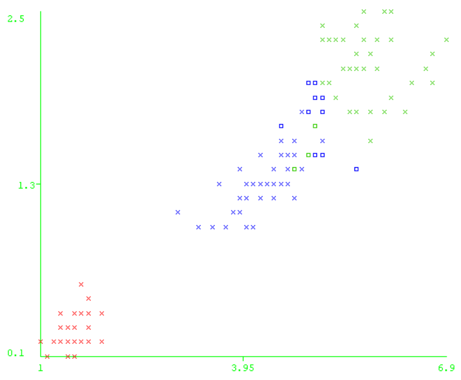


Figure 1.1d – Clusters for petal length and petal width. Boxes represent misclassified instances (k-means)

A 4% classification error for the cancer dataset (table 1.1f) led me to believe that my data *was* linearly separable, so I looked back at my HW1 to check the SVM accuracy, and it was at right around 4% (96.1% accuracy, basically identical), so k-means clustering for this data was practically the same as running an SVM. For the Iris dataset, it is easy to see *why* the classification error is 11.33%. The red cluster (Iris-setosa) is very clearly linearly separable from the other two classes, which explains the 100% accuracy. The two remaining classes do not appear to be separable and share a fringe of several points. This explains the higher classification error. The distribution of the two clusters clearly overlap slightly, so perhaps soft clustering and EM will be better for the Iris dataset. Clearly, the performance of k-means depends on the *density* of the feature space. If all 3 classes were linearly separable, then our accuracy would be at 100%.

Table 1.1e – Classification based on Clusters – Iris Dataset K-means

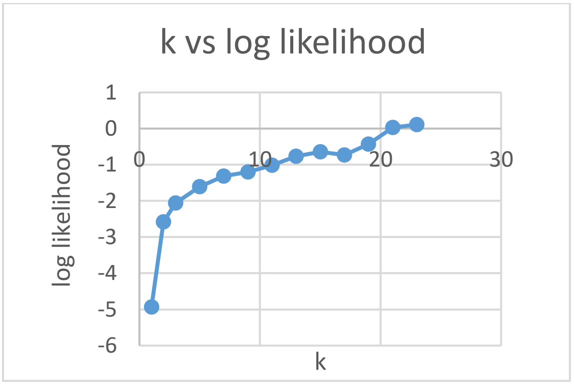
k-means			
Assigned to cluster	0	1	2
Iris-setosa	0	50	0
Iris-versicolor	47	0	3
Iris-virginica	14	0	36
Classification error:	17		**11.33%**

Table 1.1f – Classification based on Clusters – Cancer Dataset K-means

k-means		
Assigned to cluster	0	1
Malignant (0)	9	435
Benign (1)	222	17
Classification error:	26	3.8067%

It is also important to note the high SSE of the Cancer dataset in comparison to the Iris dataset. Even though it does decrease with the number of clusters (as it should – we will eventually have a cluster per data point if we keep going), the SSE still only plateaus to around 100. I believe this is a good indicator of the high dimensionality as well as range of the Cancer dataset. The data are likely very spread out and thus will likely benefit from cluster analysis as it may be easy to separate classes.

1.2 – Expectation Maximization



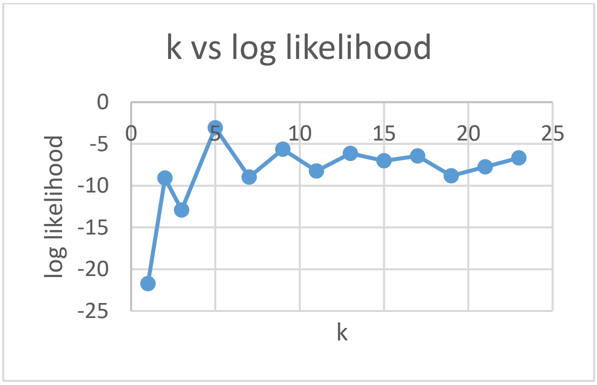


Figure 1.2a – EM k vs log likelihood for Iris Dataset

Figure 1.2b – EM k vs log likelihood for Cancer Dataset

The log likelihood functions display how the likelihood increases as we increase the number of clusters: essentially we are nearing maximal probability as each data point has their own individual cluster. However, as I explained in k-means, since we know the labels and thus number of classes, we can match k with the number of classes. If we had fewer or extra, then for each missing or additional class we would be misclassifying the entire class. So, I will be sticking with k=2 for the cancer dataset and k=3 for the iris dataset.

As predicted, EM performs better for the Iris dataset, reducing it classification error to 9.33% (Figure 9.33%). In the cluster visualizations for the same attributes we can see how our fringe points from the k-means example are no longer dependent on the Euclidean distance when determining cluster assignments, so these fringe points likely have probabilities of 50% of landing in either cluster, and now in some cases for EM they switch clusters, which gives us the slight bump in accuracy. As for the cancer dataset, as displayed further on in the analysis in Figure 2.1c, the cancer dataset appears to have a dense cluster and a cluster with a wide

Table 1.2c – Classification based on Clusters – Iris Dataset EM

EM			
Assigned to cluster	0	1	2
Iris-setosa	0	50	0
Iris-versicolor	50	0	0
Iris-virginica	14	0	36
Classification error:	14 instances		**9.33%**

Table 1.2d – Classification based on Clusters – Cancer Dataset K-means

EM		
Assigned to cluster	0	1
Malignant (0)	39	405
Benign (1)	237	2
Classification error:	41	6.0029%

distribution. This feature space likely is biased toward a k-means clustering, as the “wide” cluster will cause the EM algorithm to assign some overlapping points to the non-dense cluster.

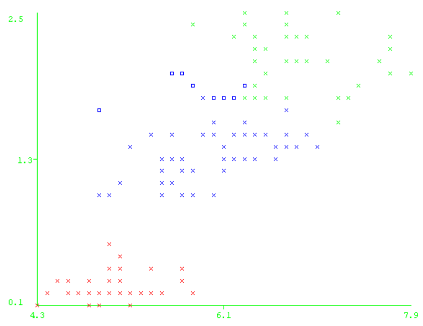


Figure 1.2e – Clusters for sepal length and petal width.
Boxes represent misclassified instances (EM)

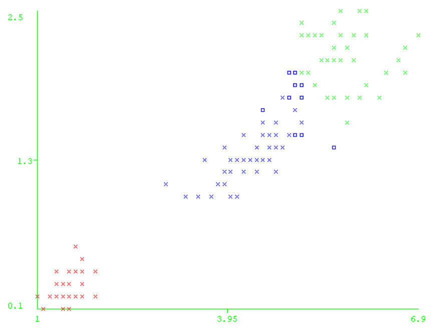


Figure 1.2f – Clusters for petal length and petal width.
Boxes represent misclassified instances (EM)

2 – Dimensionality Reduction with PCA, Autoencoders, and Randomized Projection

2.1 – PCA

Running PCA for the iris dataset with 95% of the variation covered resulted in two principle components. Looking at Figure 2.1a we can see a similar trend with one linearly separable class and two classes which share a boundary, so the data appears to be reconstructed well. However, this graph is much more informative than our previous plots. We are able to portray 95% of the variation of the data and the clusters that we saw in previous plots in with a single graph with 2 only dimensions. It is clear now that the majority of our variation is due to components that represent a high clustered form of our data like we saw in the past. Thus, we can assume that particular attributes likely explain more of the variance than others and we can remove non-necessary attributes which do not contribute to the expressiveness of the data.

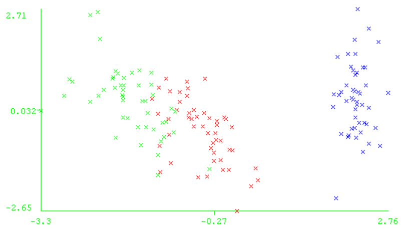


Figure 2.1a – Iris dataset, 2 component PCA graph

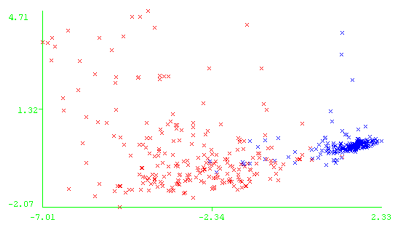


Figure 2.1b – Cancer dataset, 2 component PCA graph (#1 and #2 ranked pc’s)

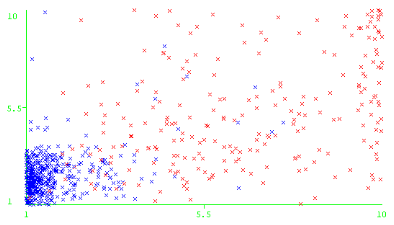


Figure 2.1c – Cancer dataset, cell shape vs cell size

For the cancer dataset, running PCA resulted in 7 principal components with 95% variation covered. With such a high dimensional space initially, this is somewhat disappointing, but looking at the eigenvalue distribution will reveal which components are the most important. Figure 2.1b displays the #1 and #2 ranked components (per Weka), and we can once again see a clear clustering of the data. One class appears to be very sparse and have a high range, while the other class likely has a more condensed distribution. This is similar to our original distribution of points (see Figure 2.1c), so PCA has again done a nice job of reconstructing the data because it picks out a linear combinations of attributes which represent the most variation.

Finally, the distribution of eigenvalues reveals the nature of PCA and the cancer dataset. One component explains ***65%*** of the variation, which is a huge amount. The remainder of the components range only from 3-8%. Clearly, the cancer data has heavy variance along one component axis and thus we can see how particular attributes were not very helpful and likely just noise.

Table 2.1d – Distribution of eigenvalues for PCA cancer

eigenvalue	proportion	cumulative	
5.8995	0.6555	0.6555	-0.381cell_size-
0.77595	0.08622	0.74172	0.906mitosis-0.
0.53925	0.05992	0.80163	0.866clump_thi
0.45963	0.05107	0.8527	-0.499bare_nuc
0.38028	0.04225	0.89496	0.69 normal_nu
0.30188	0.03354	0.9285	0.655marginal_
0.2944	0.03271	0.96121	-0.7bland_chror

2.2 – Randomized Projections

For randomized projection on the cancer dataset, I needed to analyze the ideal number of random components to include. Interestingly, as seen in Figure 2.2a, the classification error drastically reduced as I increased the number of random components to include. This could be because a higher number of random

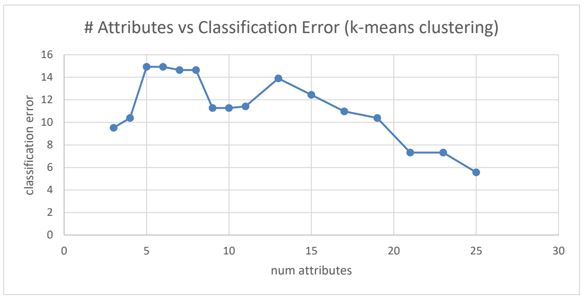


Figure 2.2a – random components vs classification error through k-means clustering for cancer dataset

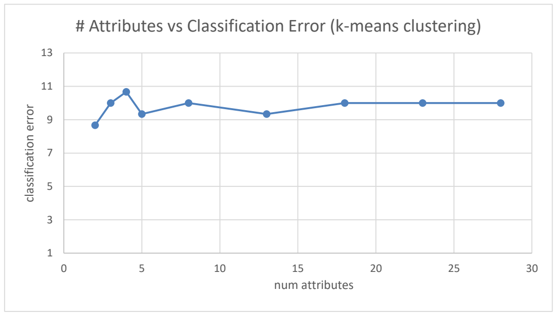


Figure 2.2b – random components vs classification error through k-means clustering for Iris dataset

components effectively covers all of the variance of the feature space, thus increasing accuracy when clustering. But since we're looking to reduce the dimensionality, not increase it, 3 random components seems to be the best choice to minimize error. For the iris dataset, however, the number of components does not aid in accuracy, and is quite stagnant. This could be due to the linear separability of the classes – adding more components does not help, likely because most of the variance resides in a smaller number of attributes. So, 2 random components were chosen to minimize error.

In order to ensure that these classification errors were not anomalies, I ran several iterations of RP with different seed values at the ideal obtained number of components for each dataset (3 for the cancer dataset, 2 for the iris). According to the mean and standard deviation of these values listed in figure, our average cancer error was worse than the original error obtained for k-means (4%), which is expected because we are reducing to 3 components from 10, thus we are likely to lose some information. The standard deviation was quite high among seeds as well at 4.14%, but with only 3 random projections, the classification error is unlikely to be similar.

Table 2.2c – Mean and variance of classification error after running k-means

	cancer (3)	iris(2)
average error	11.43485	9.00013
stdev	4.137995678	4.600087227

	cancer (25)	iris(5)
average error	5.10982	6.19981
stdev	0.591515678	2.846720536

The iris dataset error in contrast was slightly lower than its original error of 11.33% which is likely due to the structure of the iris dataset which only contains 4 attributes initially. Reducing the feature space and increasing accuracy leads me to believe that most of the variation lies within 1 or 2 attributes, but the high variance (4.6%) still remains because again, we are only using 2 random projections. In fact, we can see in Table 2.2d that the eigenvalue distribution for Iris PCA revealed only two principal components with both representing 95% of the data, and the first PC expression ***72%*** of the variance. Thus, my hypothesis was right, and the random components that I chose to minimize error likely aligned with the PC's selected by the algorithm in section 2.1.

Table 2.2d – Distribution of eigenvalues for PCA cancer

eigenvalue	proportion	cumulative	
2.91082	0.7277	0.7277	-0.581petalleng
0.92122	0.23031	0.95801	0.926sepalwidt

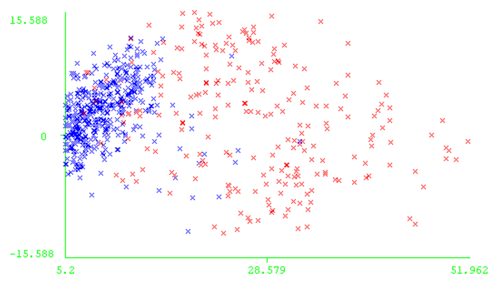


Figure 2.2e – Cancer RP3 k3k2 (increased jitter to point density)

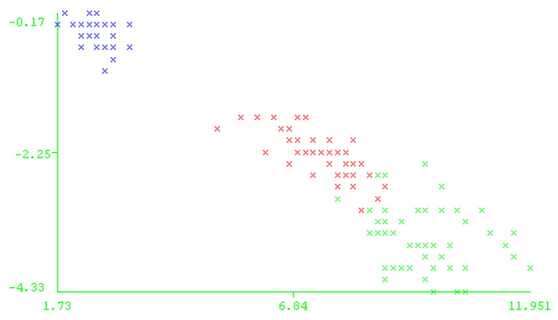


Figure 2.2f –Iris RP2 k1k2 (reduced jitter)

Plotting components for both datasets in the graphs above (while utilizing random seed which produced the projections with the lowest classification accuracies), we can see that our RP has really still maintained our feature space and the underlying structure of the clusters. Perhaps an optimization algorithm could be used to find the RPs that minimize classification error for a clustering algorithm, but I'll save that for another time. Regardless, for both the cancer and iris dataset, we can see how certain RPs produce low errors and are still able to reconstruct the feature space (after a good amount of iterations, however).

2.3 – Auto encoders

For the cancer dataset, I evaluated the sparsity of the feature vectors at 3, 5, and 7 functions for the auto encoder and obtained the following plots:

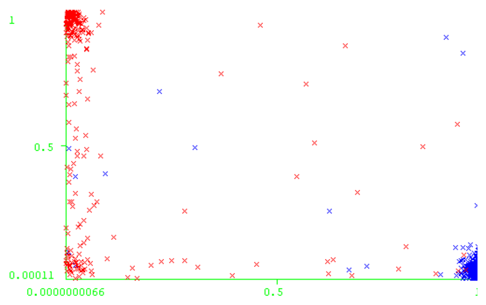


Figure 2.3a – Cancer Autoencoder feature vectors, functions = 3 (jitter increased)

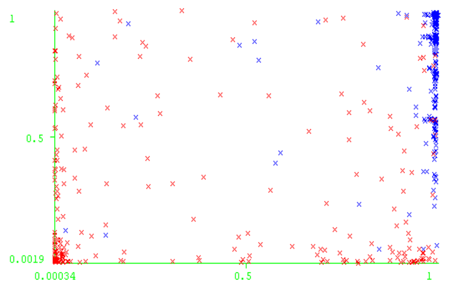


Figure 2.3b – Cancer Autoencoder feature vectors, functions = 5

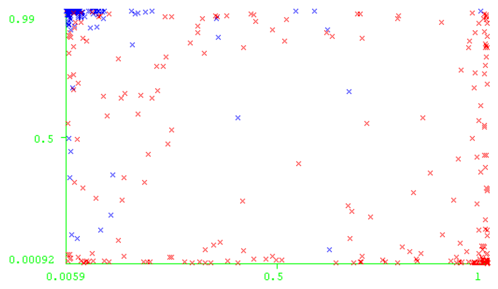


Figure 2.3c – Cancer Autoencoder feature vectors, functions = 7

It is somewhat difficult to discern, but it appears that the number of functions decreased the sparsity in the dataset. This could be because higher dimensional data is contains less expressiveness when run through auto encoders, resulting in less sparse feature vectors. In other words, the auto encoder cannot reveal as deep of an underlying structure with more dimensions. Since we want an auto encoder with sparse feature vectors, I chose 3 functions.

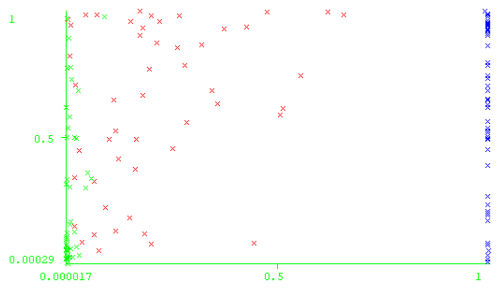


Figure 2.3d – Iris Auto encoder feature vectors, functions = 2

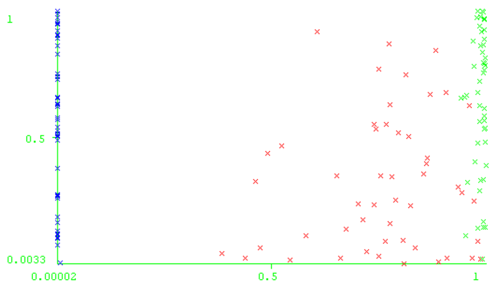


Figure 2.3e – Iris Auto encoder feature vectors, functions = 3

Running the auto encoder for the Iris data set with values of 2 and 3 functions did not result in drastically different feature vectors for either 2 or 3 functions. Thus, choosing 2 hidden outputs would be more beneficial as we can maintain the expressiveness with only 2 features instead of 3.

3 – Clustering with Dimensionality Reduction

3.1 – K-means

Once again running a k-means clustering at k=2 clusters for the cancer dataset, we can see that all RP and AE have worse errors than our original 3.8067% error with no dimensionality reduction. RP was the worst which makes sense in the case – it was reduced to 3 components from 10, and the components are randomly generated. Since RP works much better in higher dimensions, a lower accuracy in this case would be expected. Also, we see a *decrease* in error for PCA. This algorithm extracted 7 components which explained 95% of the variance and revealed an important underlying structure which allowed for an ideal clustering – PCA removed unnecessary or nondescript noise, thus allowing for a good clustering of the data.

Table 3.1a – Classification error for cancer dataset, k-means clustering (k=2)

	Classification Error
PCA7	3.5139%
RP3	4.9780%
AE3	3.9531%

Table 3.1b – Classification error for iris dataset, k-means clustering (k=3)

	Classification Error
PCA3	11.33%
RP2	4.00%
AE2	20.00%

For the iris dataset, we can see that PCA with 2 components performed identically to our original clustering with 11.33% accuracy. Since the reduction was only from 4 to 2 dimensions, the underlying structure was likely not changed significantly. This just tells us that our original data can just as easy be portrayed in 2 dimensions as 4, which will reduce run time and increase efficiency. The autoencoder performed significantly worse, which could be due to the fact that the autoencoder ouput feature space was not very interpretable from a clustering perspective. Finally, very strangely the random projection algorithm performed *significantly* better. Even with different seeds, the majority of the iterations resulted in errors lower than the original k-means clustering. Plotting this new clustering, we can see in Figure 3.1c how the new clustering seems to have widened the gap between the two clusters which were previously very close. Again, the ideal RPs selected were chosen to minimize error and increase expressiveness, so if anything the gap was part of the underlying structure and these ideal RPs helped display that.

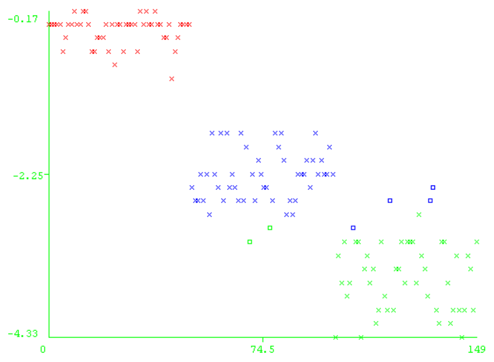


Figure 3.1c –k-means clustering (k=3) for Iris Dataset, RP with 2 components

3.2 – EM

For EM, all of the dimensionality algorithms produced worse accuracies for the cancer dataset than the original accuracy of 6%, which was expected. PCA once again did a good job of minimizing the increased error while still maintaining the underlying structure, and the same can be said for AE. However, RP for EM performed significantly worse compared to k-means. Plotting the cluster visualizations for both clustering algorithms in Figures 3.2b & c, we can see exactly how the random projections benefitted from a k-means clustering over an EM clustering. The random projections produced a higher density distribution, and clearly this heavily reduced the error for k-means. The EM algorithm assumed a large spread for the red cluster which overlapped with the high density blue cluster, thus resulting in a very low accuracy. I can see now how different dimensionality algorithms reveal a clearer underlying structure of the data, and in this case this random projection matched the high density distribution of a particular class, which is perfect for a k-means clustering. It is apparent that different combinations of dimensionality reduction and clustering techniques can improve accuracy if used in the correct way. Regardless, it is clear to see that k-means is better for the cancer dataset, but this dim-reduction algorithm just happened to significantly reduce accuracy because of the true structure of the underlying data which was revealed by these random projections.

Table 3.2a – Classification error for cancer dataset, EM clustering (k=2)

	Classification Error
PCA7	8.0527%
RP3	19.3265%
AE3	7.4671%

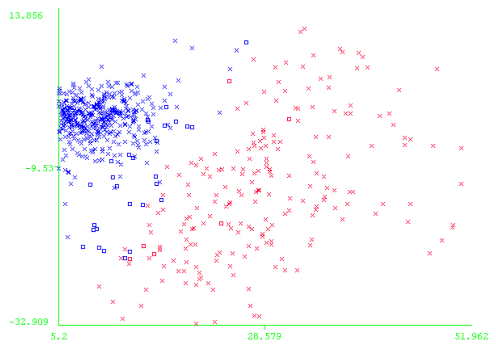


Figure 3.2b –k-means clustering (k=3) for Cancer dataset, RP components K3 and K1

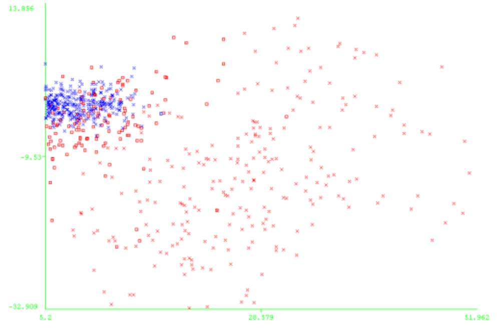


Figure 3.2c –EM clustering (k=3) for Cancer dataset, RP components K3 and K1

For the iris dataset, PCA and AE performed the same for EM clustering, and slightly worse than the original EM clustering of 9.33%. This is still a good error %, as it's still relatively low while maintaining aspects of the original feature space. RP, however, performed the same as k-means and thus again significantly lower than the original EM clustering. As I described for k-means, the random projections for the iris dataset seemed to reveal a better underlying structure for clustering, and in this case it just happened to work out well for both k-means and EM, unlike RP for the cancer

Table 3.2d – Classification error for iris dataset, k-means clustering (k=3)

	Classification Error
PCA3	14.67%
RP2	4.00%
AE2	14.67%

dataset. As Figure 3.1c displays, the random projections produced dimensions which were not extremely dense and benefitted both k-means and EM because the clusters resulting from these random projections were very separated and contained within their own clusters, which can produce high classification accuracy for both k-means and EM algorithms.

4 – Neural Network with Dimensionality Reduction

4.0 – Baseline

I switched from SKLearn to Weka in this assignment, so I will use Weka's NN output as a baseline.

4.1 – PCA, RP, and AE comparisons to NN from Assignment 1

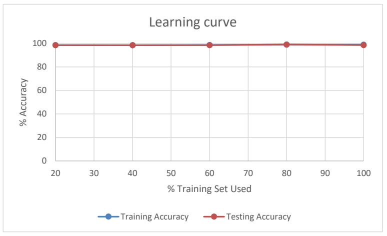


Figure 4.1a –Learning curve for NN on cancer dataset

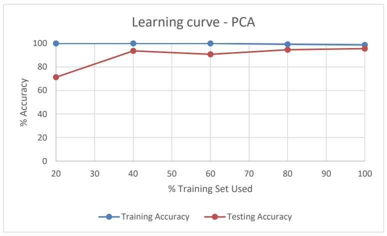


Figure 4.1b –Learning curve for PCA NN on cancer dataset

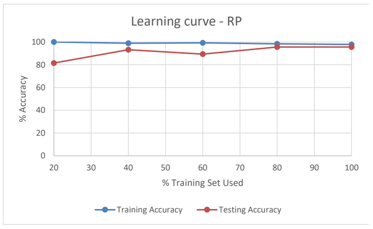


Figure 4.1c –Learning curve for RP NN on cancer dataset

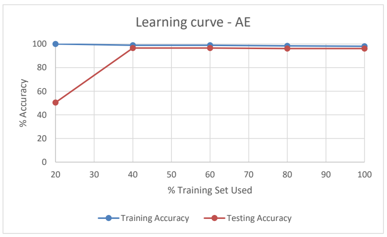


Figure 4.1d –Learning curve for AE NN on cancer dataset

	Classification Error:
Original	98.54%
PCA7	95.63%
RP3	95.63%
AE3	96.12%

Figure 4.1e – Classification errors for dimensionality-reduction algorithms

	Time to build model (s)
Original	0.61
PCA7	0.43
RP3	0.21
AE3	0.2

Figure 4.1f –Time to build model for Dimensionality Reduction Algorithms

After running the new components obtained through dimensionality reduction from the cancer dataset through a neural network, the new classification errors were revealed to be not significantly reduced compared to the original neural network from assignment 1. In figure 4.1e, we can see how PCA and our ideal RP (with 3 projections) managed to obtain the same classification error with only about a 3% reduction from the original NN. PCA and RP (through trial and error to reach the ideal # of RPs) were able to find a decent underlying structure. AE was even higher than PCA and RP, and this is likely because of the autoencoder's use of sparse neural networks to discover an even deeper understanding of the datasets. The learning curves demonstrate how dimensionality reduction algorithms require more data to obtain higher accuracies. This is likely because, since the data has fewer dimensions, the neural network needs more data to make accurate predictions. Additionally, although not significant, dimensionality reduction did reduce the amount of time to build model, and it appears that it was proportional to the total attributes in the model. Although .2 and .4 is not that significant, for very large datasets this could be very effective in increasing efficiency while maintaining a reasonable level of error.

5 – Neural Network with Dimensionality Reduction + Clustering

I had to use Weka instead of SKlearn again, and I believe Weka does randomized splits which I could not do in the previous part (I had to create manual splits), so the original accuracy is lower because of the way the data is structured initially in the cancer dataset (not random, might be autocorrelated).

Since the cancer dataset initially had a somewhat clustered and dense distribution within clusters, PCA, RP, and AE were able to extract very expressive components, and since we had more expressive attributes with less noise, the clustering assignments added an even higher level of expressiveness to the datasets. Thus, we were able to obtain higher classification errors for all of the permutations of dimensionality reduction and clustering algorithms. As I discussed before, AE performed better than PCA and RP because of the autoencoder's ability to pull even deeper meaning from the data.

	Classification Error:
Original	96.10%
PCA7 kmeans	97.56%
PCA7 EM	98.05%
RP3 kmeans	97.07%
RP3 EM	97.56%
AE3 kmeans	98.05%
AE3 EM	98.0488

Figure 5.1a – Classification errors for Dim-reduction + Clustering Algorithms

The build times for the different permutations are not that different than before we ran the clustering algorithms. However, it is important to point out how they were *slightly* slower because of the added nominal cluster attribute. This is quite insignificant however, especially if you were reducing from 400 to 15 attributes, adding one additional attribute would not really impact the build time at all.

	Time to Build Model(s)
Original	0.61
PCA7 kmeans	0.52
PCA7 EM	0.51
RP3 kmeans	0.28
RP3 EM	0.29
AE3 kmeans	0.34
AE3 EM	0.28

Figure 5.1b –Time to build model for Dim-reduction + Clustering algorithms

*Note, there is a typo in several tables – for the iris dataset, it should read "PCA2", not "PCA3"

__References__

[1]https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)
[2] https://bl.ocks.org/rpgove/0060ff3b656618e9136b