Machine Learning – Unsupervised Learning
November 2017
Timothy Baba, Georgia Institute of Technology

Abstract - The purpose of this project is to explore and analyze 5 unsupervised learning algorithms divided into two clustering algorithms (K-means(KM) and Expected Maximization(EM)) and three dimensionality reduction algorithms namely: Principal Component Analysis (PCA), Autoencoders (AE) and Randomized Projection(RP). This paper is divided into 5 Experiments. In experiment 1, we run the two clustering algorithms above on Pima Indian Diabetes and Wisconsin Cancer Datasets. In experiment 2, we apply the dimensionality reduction algorithms above on our datasets. In experiment 3, we reproduce our clustering experiment from experiment 1 on the newly projected data from experiment 2. In experiment 4, we run a neural network learner on the newly projected Wisconsin Cancer dataset from experiment 2. And lastly in experiment 5, we add our cancer data cluster labels from experiment 3 as new features to the newly reduced cancer data from experiment 2, and run a neural network learner on the resulting data.

A. Introduction

The datasets used in these experiments are same as those used in our supervised learning analysis from Project 1. As a reminder, the Pima Indians Diabetes dataset has in total 768 instances with 8 attributes and two classes where class value 1 is interpreted as "tested positive for diabetes" and 0 as "tested negative for diabetes". There are in total 500 instances labelled 0 and 268 labelled 1. This dataset is interesting because all instances have been carefully chosen to include only females of pima Indian heritage and at least 21 years of age. A focus such as this will enable us to eliminate 'noise' and help us improve the accuracy of our models. On the other hand, the Breast Cancer Wisconsin (Diagnostic) dataset has in total 569 instances with 30 attributes and a class variable (0 or 1) where class value 0 is interpreted as "WDBC- Malignant cancer" and class value 1 is interpreted as "WDBC-Benign cancer". There are in total 212 instances labelled as WDBC Malignant Cancer and 357 total instances labelled as WDBC-Benign cancer. More information about both datasets are available and can be obtained from the UCI Machine learning repository.

In our experiments, we split our data into 70% training samples and 30% testing samples. The data was also preprocessed using a standard scaler library from sklearn. Similarly, all clustering, dimensionality reduction and neural nets training were done using python's scikit learn library.

The features used in both datasets were narrowed down to just three that represent the most important features in each dataset. These were determined using an ExtraTreesClassifier to make estimates on the importance of each feature. Focusing on the import features will enable us reduce overfitting (as our classifiers will make decisions on less redundant /noisy data), improve the accuracy of out classifier and also reduce training time as less data will be used. Below is an exploratory visualization of how each of our dataset classes distribute along our selected features.

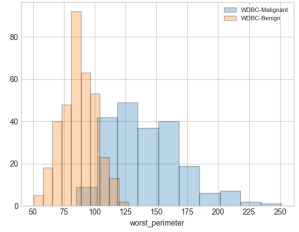

Figure 1: worst perimeter

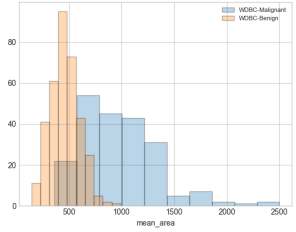

Figure 2: mean area

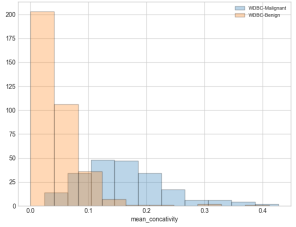

Figure 3: mean concavity

From figures 1, 2 and 3 above, we can observe that among the Wisconsin Cancer dataset instances classified as WDBC-Malignant cancer (more dangerous type), majority had worst perimeter of around 125cm, mean area of around 750 square cm and mean concavity ranging between 0.1 and 0.19.

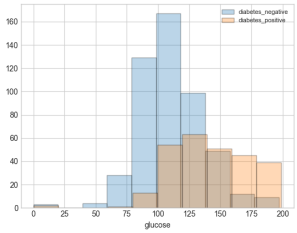

Figure 4: glucose

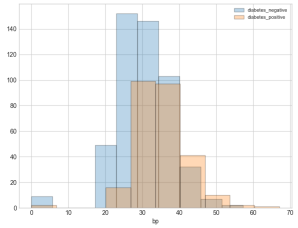

Figure 5: blood pressure

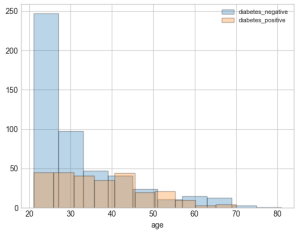

Figure 6: age

Similarly, we can also observe from figures 4, 5, and 6 that among the Pima Indian Diabetes Dataset instances classified as diabetes positive, majority had glucose concentration of around 125 mg/dl, blood pressure of around 30 mm Hg and were of age range between 20 and 30.

1. Analysis of unsupervised clustering algorithm

The clustering algorithms used are K-Means and Expected maximization from sklearn. The metric used for closeness or similarity of data points was the Euclidean distance. We didn't use the Manhattan distance because our data was both pre-processed and continuous. Also, Minkowski distance metric wasn't also appropriate here as the dimensions of our attributes and classes were not very high.

1.1 K-Means Clustering on Pima Indian Diabetes

k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

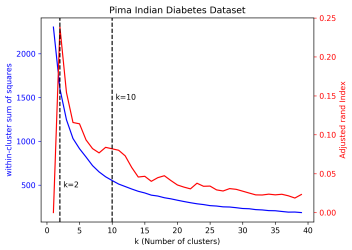

Figure 7: Pima Indian diabetes dataset

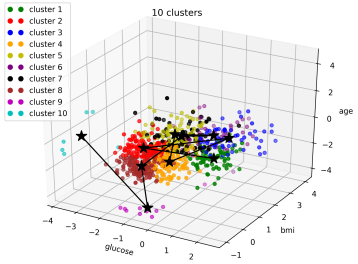

Figure 8: Pima Indian diabetes dataset

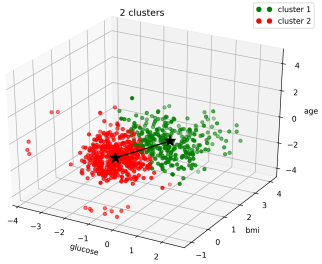

Figure 9: Pima Indian diabetes dataset

Using the elbow analysis, we see from figure 7 that we require 10 clusters to minimize the within cluster sum of squares (WCSS). Requiring this high number of clusters tells us our data have values far apart from each other, and 10 clusters could help us visualize the different subgroups as seen in figure 8. Table1 shows us the mean attribute values of the instances in each cluster (M/clusters) and the mean attribute values of the instances grouped by their ground truth labels (M/class). Using the domain knowledge of M/class, we can infer that clusters 1, 6, 3, 5 are diabetic

Table 2(mean values)

ARI = 0.23828

clusters	Glucose	Bmi	age
1	146.363	35.283	42.337
2	104.298	29.848	27.314
trueClass			
0	109.98	30.304	31.19
1	141.257	35.1425	37.067

Table 1(mean values)

ARI = 0.0819

clusters	Glucose	Bmi	age
1	176.104	33.832	32.477
2	99.500	36.236	25.417
3	162.633	33.977	50.433
4	128.851	29.994	27.462
5	105.081	34.424	43.455
6	125.085	26.447	59.978
7	140.563	44.824	29.380
8	95.889	25.159	25.617
9	101.100	0.000	26.600
10	7.333	31.567	29.833
trueClass			
0	109.98	30.304	31.19
1	141.257	35.1425	37.067

positive while clusters 10, 9, 8, 2, 4, 5 are diabetic negative. And if we observe clusters 10, 9, 8, 2, 4, & 5 closely again, we see that more individuals testing negative to diabetes were below age 30 while clusters 1, 6, 3, & 5 suggests the contrary. This age distribution does not match that of our ground truth. If we look back at the introduction section, we know that majority of individuals tested positive for diabetes were below the age 30. Hence from this point, we can begin to doubt the Adjusted Rand Index (ARI, a measure of the similarity between the clustering labels and our ground/actual class labels) of K-means on this data. As we can see, figure 7 shows that the 10 clusters we get with K-means have a low Adjusted Rand Index of just 8%.

However, in our experiments, we would want to maximize AR since we have knowledge of our ground truth. Hence by using just 2 clusters as seen in figure 7, we could achieve clusters with optimum ARI of 23%. From Table 2 above, comparing the M/cluster to the M/class, we can infer that cluster 1 is diabetic positive since it is closer to the M/class of trueClass label 1 and on the other hand cluster 2 is diabetic negative. We see from figure 9 that our data could be almost perfectly linearly separable which explains why we obtained optimum ARI with just two clusters. However, because 23% is still very low, using K-means alone on our dataset is not efficient.

1.2 K-Means Clustering on Wisconsin Cancer Dataset

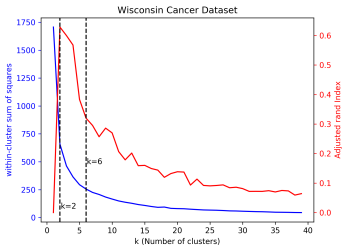

Figure 10: KM - Wisconsin Cancer dataset

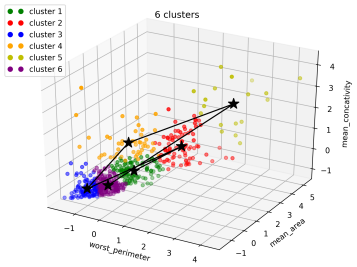

Figure 11:KM- Wisconsin Cancer dataset

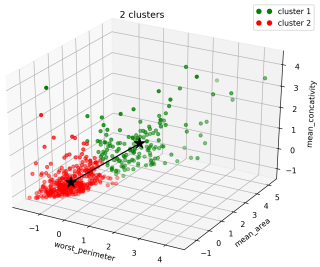

Figure 12: KM-Wisconsin Cancer dataset

Table 3: (mean values)

ARI = 0.31996

clusters	Worst perimeter	Mean concavity	Mean area
1	111.386	0.091	718.97
2	158.158	0.169	1168.000
3	72.431	0.032	323.796
4	114.15	0.205	634.396
5	199.372	0.303	1737.22
6	91.430	0.0398	505.499
trueClass			
0	141.370	0.161	978.3764
1	87.0059	0.046	462.790

Here we see from figure 10 that with 6 clusters we can minimize WCSS. We can also observe from figure 11 that clusters 3, 6, and 1 have values much closer to their centroids than the remaining clusters. This is a depiction of stronger relationship among instances in the cluster. Also, comparing to the ground truth from Table 3, we see that clusters 3, 6, and 1 have instances with similar mean-attributes compared to those of trueClass 1. Hence, we infer that clusters 3, 6 and 1 are Benign cancer patients while cluster 2, 4, and 5 are Malignant cancer patients. While 6 clusters were able to help us visualize more subgroupings, the obtained ARI of 32% is still low. With 2 clusters however, we could gain an ARI of 63%. This implies our data is linearly separable (as we also see above) from figure 12 and can be represented in two clusters with 63% accuracy. Hence, we conclude that K-Means behaves better on Wisconsin dataset than on pima Indian Diabetes Dataset.

1.3 Expected Maximization (EM) on Pima Indian Diabetes

EM in its simplest case can be used in finding number of optimum cluster just like k-means, but unlike k-means, EM could also be used in making probabilistic cluster assignments. As we can see below, EM requires only 3 clusters to both maximize ARI and minimize the Bayesian information criterion (BIC is an

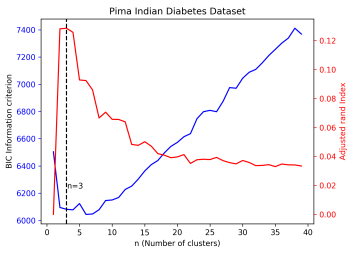

Figure 13: EM- Pima Indian Diabetes dataset

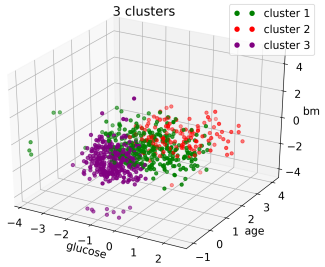

Figure 14: EM- Pima Indian Diabetes dataset

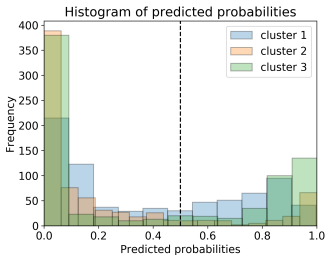

Figure 15: EM- Pima Indian Diabetes dataset

estimator of the relative information lost on a given model). If we look at the clusters formed, we see that it is practically impossible to linearly separate the clusters which explains why we need more than just 2 clusters in the first place to be able to classify our data. The clusters are noisy as we see that some of the data fall in more than one clusters hence making it difficult to really classify. Similarly, we see from the histogram above that the same number of instances (around 360) have equal probabilities of falling in both cluster 3 and cluster 2. All the above are due to the nature of Expected Maximization which in our case makes our clusters not only difficult to describe but also very not similar to our ground truth. As a result, EM on Pima Indian performed poorly with an ARI score of just 11.2%.

1.4 Expected Maximization (EM) on Wisconsin Cancer data

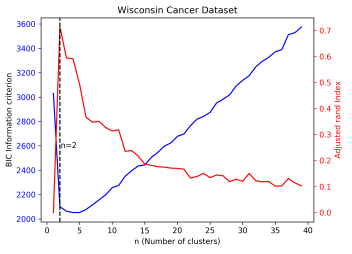

Figure 16: EM - Wisconsin Cancer dataset

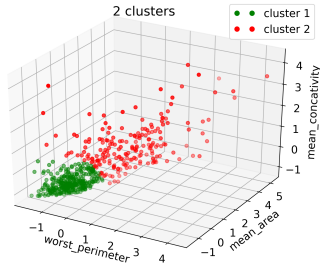

Figure 17: EM - Wisconsin Cancer dataset

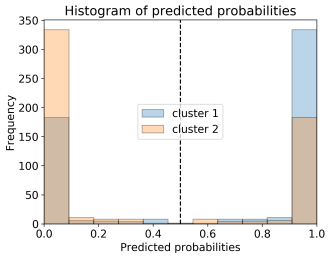

Figure 18: EM - Wisconsin Cancer dataset

ARI = 0.7068

clusters	Worst perimeter	Mean concavity	Mean area
1	87.500	0.043	462.709
2	141.812	0.169	990.972
trueClass			
0	141.370	0.161	978.3764
1	87.0059	0.046	462.790

We also see here that EM could maximize ARI and minimize BIC with just only 2 clusters on the Wisconsin cancer dataset. Unlike EM on our previous dataset, the clusters here appear to be linearly separable. From the table to the right we observe that the mean of the attributes of cluster 1 appear to be closer to the mean of the attributes of class 1. Hence, we can infer that cluster 1 are Malignant Cancer patients while cluster 2 can be inferred in a similar manner to be benign cancer. From the histogram above, we see a more symmetric probability distribution suggesting that if 350 instances had high probabilities of falling into cluster 1, those same instances will have low probabilities of falling into cluster 2. This symmetric behavior ensures we get less noisy and more linearly separable clusters. As a result, EM on Wisconsin dataset performed best with a very high ARI score of 70.68%

2. Analysis of unsupervised dimensionality Reduction algorithms

To get the number of principal components that our datasets could be reduced into without losing too much information, we use the "explained variance" measure which can be calculated from the eigenvalues. To obtain the Eigen values, we perform an

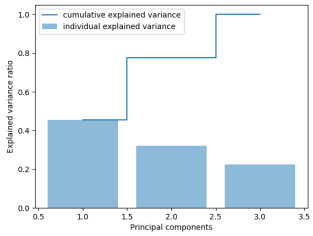

Figure 19: Diabetes Dataset. Eigen value distribution: [1.36, 0.96, 0.673]

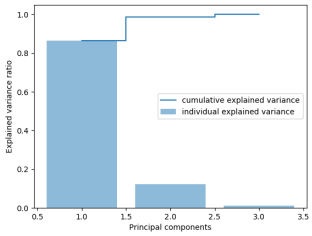

Figure 20: Cancer Dataset Eigen value distribution: [2.59, 0.37, 0.034]

eigendecomposition of the covariance matrix of our dataset where each element represents the covariance between two features. We sort and then choose the principal components (which correspond to the Eigen vectors) with the highest Eigen value distribution. Figure 19 shows us that 45% of the variance can be explained by the first principal component alone. The second component still bears 25% of information while the third component can safely be dropped without losing much information. Hence, we could reduce the Pima Indian Diabetes dataset into two components and still retain 80% of information. Similarly, figure 20 suggests that the Wisconsin Cancer dataset can be reduced into two principal components containing 97% of information.

2.1 Principal Component Analysis (PCA) on Pima Indian Diabetes

PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

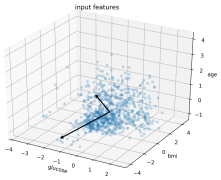

Figure 21

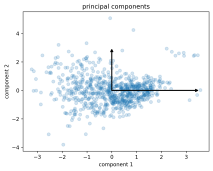

Figure 22

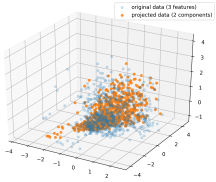

Figure 23

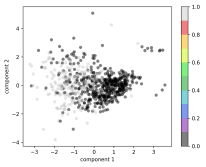

Figure 24

Figure 21 above is a vector visualization of PCA's principal components and explained variance over the scatter plot of the Pima Indian Diabetes dataset. The principal components defined the direction of the vector while the explained variance defined the squared-length of the vector. The vectors seen in figure 21 represent the principal axes of our data where the length of the vector along an axis is a measure of the variance of our data when projected on that axis. The projection of each data point onto the principal axes are the principal components of our data and if we take a look at figure 22, we see a visualization of how our data lay on its principal components. Furthermore, figure 23 shows us a plot of our projected data along our original data. The fraction of the variance cut out (which is proportional to the spread of points about the cone shaped plot of our projected data in figure 23) is roughly a measure of how much information PCA discards. Figure 24 is similar to figure 22 in the sense that they are both scatter plots of our principal

components, however figure 24 attempts to gain information of our reduced data by associating each data point in our reduced data to its ground truth label. Hence, we see 2-dimension representation with same class labels as those of our original 3-dimensioanl data. More information about our data will be discussed in experiment 3 when we reproduce our clustering experiments on it. Lastly, it is important to note that while PCA was used here for dimensionality reduction, it could also be used as a tool for visualization, noise filtering, and feature extraction.

2.2 Principal Component Analysis (PCA) on Wisconsin Cancer Data

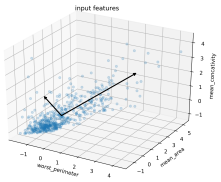

Figure 25

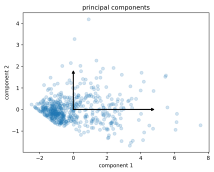

Figure 26

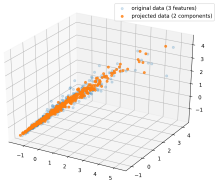

Figure 27

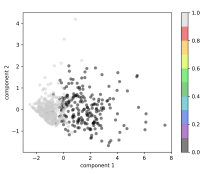

Figure 28

PCA was performed on our Wisconsin Cancer dataset and as can we can observe from figure 25 and 27, our data features appear to exhibit a more linear relationship. Also, if we observe figure 27 closely, we see that very little information was lost from reducing our data which is because using two clusters as earlier discussed above retains 97% of information, hence we see very minimal spread of points about our newly projected data. Finally, figure 28 shows us a 2-dimension PCA reduced version of our dataset with same class labels as those of our original 3-dimensioanl data.

2.3 Autoencoders (AE) on Pima Indian Diabetes dataset VS AE on Wisconsin dataset

AE are neural networks that aim at approximating our input data and eventually try to generate an output that is well trained very similar to our input. AE consists of encoders and decoders. The encoders on one hand try to compress our input data into a

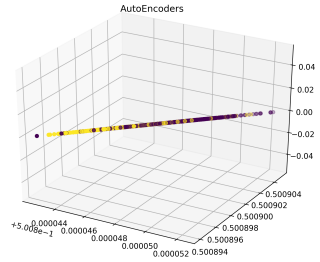

Figure 29

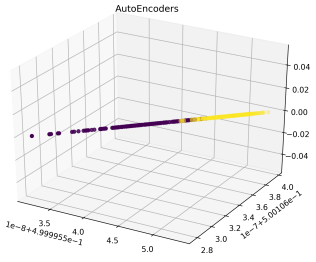

Figure 30

low-dimensional representation while the decoders try to reconstruct the input data based on the low dimensional representations generated by the encoders. To achieve this, AE begins with some random low representations and will gradually gradient descend towards an optimal solution by changing the weights that connect the input layers to the hidden layer and of those connecting the hidden layer to the output layer. AE operates on a more automatic approach with no linearity assumption of our features. Hence if linear activations or only a single sigmoid layer is used, then AE will tend to produce a solution that is much similar to PCA. With cross validation, the hyper parameters found to optimize AE's performance on our data include 3 sigmoid layers (3 units for the first two layers and 2 units for the third), and a learning rate of 0.002. Figure 29 and 30 show a 2-dimensional AE reduced version our diabetes and cancer datasets respectively with same class labels as those of their original 3-dimensioanl data. If you notice Fig 29 and 30, you'll see that we plotted the newly projected 2-D data on a 3-D space, this was because AE was able to compress our data (without losing much information) to very small values that we needed an extra depth axes to visualize the class distribution. While these plots may not provide us with much information at this time (of course other than a reduced dimension and minimal noise), we will later be able to analyze the

effects of AE by observing the performance of a neural net classifier on an AE reduced version of our datasets.

2.4. Randomized Projection (RP) on Pima Indian Diabetes dataset

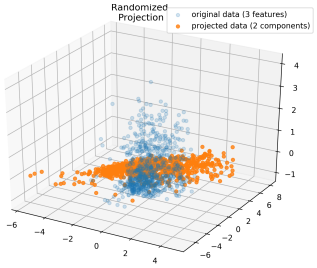

Figure 31

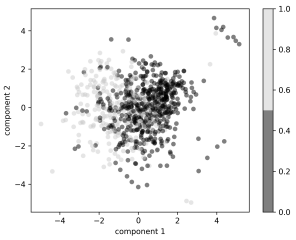

Figure 32

The main theory behind random projections is the Johnson-Lindenstrauss lemma which states that a small set of points in a high-dimensional space can be embedded into a space of much lower dimension in such a way that that the distance between the points are preserved within some epsilon variance. The Johnson-Lindenstrauss lemma (JLL) makes no assumptions about our data which makes it more flexible on different datasets than the other dimensionality reduction algorithms. Given that the number of instances in our Pima Indian Diabetes dataset is 768, running JLL's minimum dimension method shows that our projection matrix would have to contain 5694 components in order to preserve the Euclidean distances within a tolerance of 0.1 epsilon. However, our dataset only has 8 features (which we already reduced to 3), hence, we can't preserve the pairwise distance between our data points. However, since we care more about the accuracy of the NN classifier on an RP reduced data (as we will see in experiment 4 & 5), we can care less about the perseverance of the pairwise distances between our data points. For our experiments, we will use the dense Gaussian random projection matrix which reduces dimensionality by projecting the original input space on a randomly generated matrix. In Figure 31, we see a reduced 2-dimensional projection of our data along the original data and a visualization of how much information is cut off. The newly projected data appears a little bit noisy. The distribution of our projected data over the ground truth label is also shown in figure 32.

2.4. Randomized Projection (RP) on Wisconsin Cancer dataset

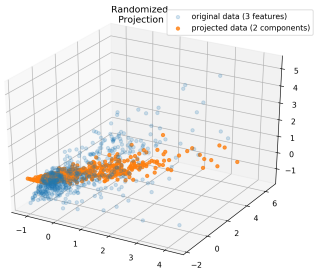

Figure 33

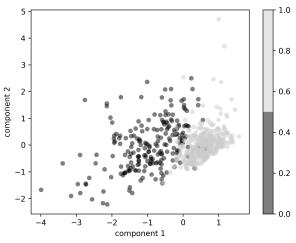

Figure 34

Similarly, our experiments from 2.3 were repeated on the Wisconsin Cancer dataset and we can see plots of our newly projected data. Here also we observe that our newly projected data appear to be a little noisy. More information about this data will be carefully seen when we dive into experiments 3 and 4 as we begin to see what clusters can be formed, how these clusters relate to our ground truth and how the data impact the accuracy of our neural net classifier.

3. Analysis of clustering algorithms on dimensionally reduced data from 2

Here we apply both out K-Means and Expected Maximization clustering algorithms on our reduced datasets from experiment 2 and see how these clusters compare to those formed on our raw data.

3.1 K-Means vs. EM on PCA-reduced dataset (Pima Indian Diabetes)

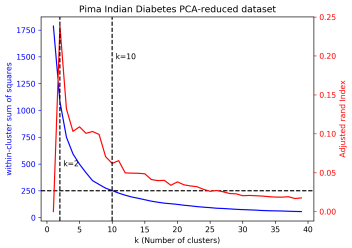

Figure: 35 KM + PCA

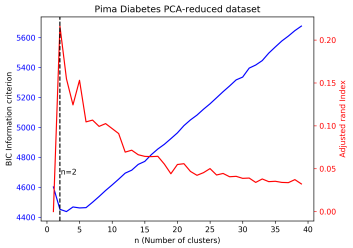

Figure 36: EM + PCA

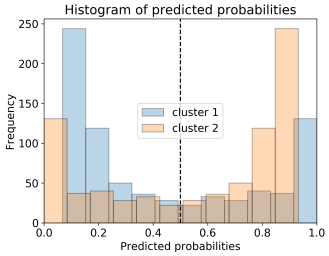

Figure 37: predicted probabilities

We can see from figure 35 that PCA dimensionality reduction on the Pima Indian diabetes dataset did not have any effect on the number of clusters required to minimize WCSS and maximize ARI (10 and 2 respectively), and neither did it change the ARI score which still remained at 23%. However, we notice that the WCSS at 10 clusters greatly reduced from 500 to 250. Hence, we see that PCA was able to help penalize the WCSS by 50%. This implies we can now observe stronger relationships among our clusters. Similarly, PCA on Wisconsin diabetes dataset had no effect on the optimum number of clusters. However, here we recorded two changes. First was that the WCSS at 2 clusters decreased from 6100 to 4450. And second, the ARI increased from 11.2% to 17%. This therefore implies that not only are we now able to observe stronger relationships among clusters, we now have clusters that look a bit more similar to the ground truth. Lastly figure 37 shows us a symmetric probability distribution of our data points in their respective clusters. This behavior indicates that our clusters are linearly separable and all points are associated with a particular cluster. As a result, we see that EM clusters of our diabetes dataset benefit more from PCA reduction.

3.2 K-Means vs. EM on PCA-reduced dataset (Wisconsin Cancer)

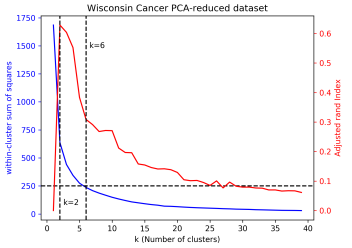

Figure 38: KM + PCA

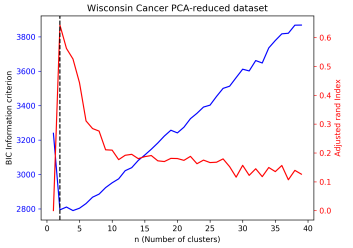

Figure 39: EM + PCA

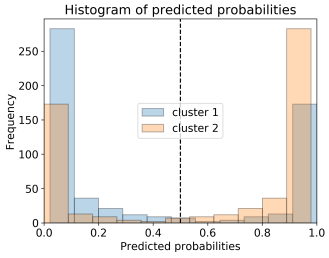

Figure 40 EM + PCA

With k-means on PCA-reduced Wisconsin dataset, the number of clusters to maximize ARI and minimize WCSS remained the same and the WCSS also never changed. However, the ARI increased from 32% to 63%. Hence, we see that PCA was able to make our data points in our clusters more similar to the ground truth. On the other hand, PCA had a worst effect on EM clusters. In fact, the only thing good was the symmetric behavior seen in figure 40. Not only did we experience a slight jump from 2100 to 2800 WCSS at 2 clusters, the ARI (with 2 clusters) also decreased from 70.68% to 66%. This therefore indicate that KM clusters of the cancer dataset benefited more from PCA reduction.

3.3 K-Means vs. EM on AE-reduced dataset (Pima Indian Diabetes)

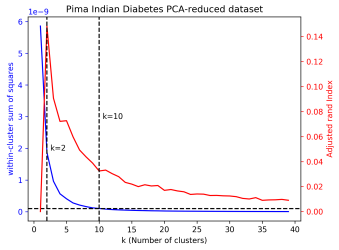

Figure 41: KM +AE

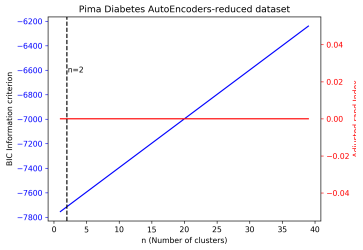

Figure 42 EM + AE

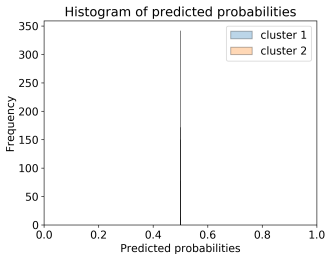

Figure 43

The results of the experiments here from figure 41 show that the number of clusters needed to minimize WCSS and maximize ARI decreased from 3 to 2. While KM + AE greatly minimized WCSS, the ARI however decreased from 23%(KM w/o AE) to 15%. For EM + AE, more WCSS was greatly penalized but resulted in a 0% ARI as shown in figure 42. An explanation could be found from the histogram in figure43. The histogram in figure 43 shows that all our data points have a 50-50 chance of falling in cluster 1 or 2. This is bad as we may find data points in more than one cluster due to the high level of classification uncertainty. It appears AE does not really improve but instead worsened the results of our clustering.

3.4 K-Means vs. EM on AE-reduced dataset (Wisconsin Cancer)

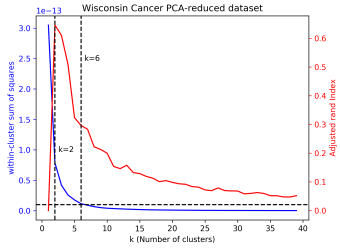

Figure 44: KM + AE

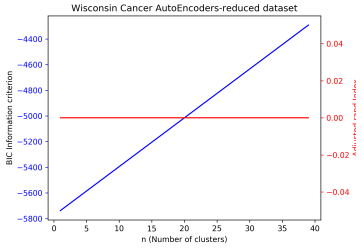

Figure 45: EM + AE

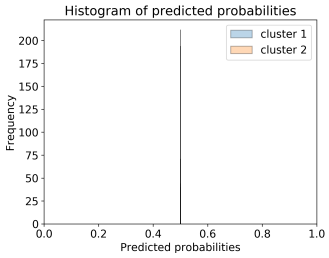

Figure 46

Interestingly also, reducing the Wisconsin cancer dataset with Autoencoders seemed to yield similar results as those of section 3.3 for the most part. As can be seen from figure 44, the number of clusters that maximized ARI and minimized WCSS (2 and 6 respectively) did not change. However, the ARI (2 clusters) was improved from 63% to 65%. On the other hand, EM + AE resulted in a 0% ARI as we can observe from figure 45. The histogram also showed a high level of classification uncertainty as observed also in the previous section. Therefore, we conclude that KM clusters of the cancer dataset benefit more from AE reduction.

3.5 K-Means vs. EM on RP-reduced dataset (Pima Indian Diabetes)

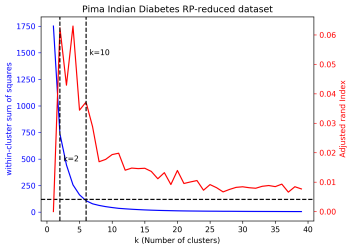

Figure 47: KM + RP

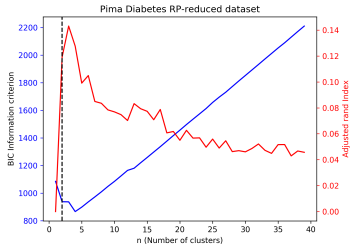

Figure 48: EM + RP

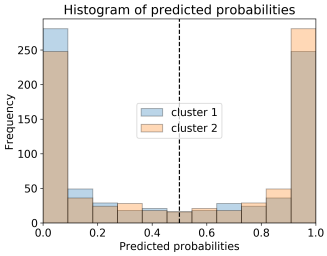

Figure 49

WCSS decreased from 500 to 125 for KM + RP. While a decrease from this indicates a stronger relationship among our clusters, it did not result in clusters similar to our ground truth as ARI was recorded to be just 6.2%. Also, EM + RP exhibited a decrease in WCSS from 6100 to 900, and the histogram in figure 49 suggests our data was somewhat separable, however, the ARI score of only 6.4% indicates the clusters had many data points inaccurately (based on our ground truth) classified.

3.6 K-Means vs. EM on RP-reduced dataset (Wisconsin Cancer)

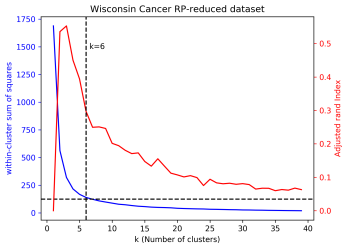

Figure 50: KM + RP

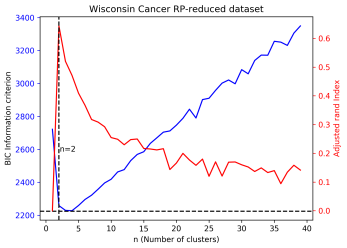

Figure 51: EM + RP

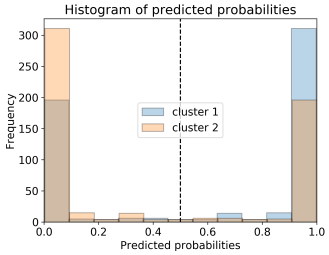

Figure 52

We see also a general decline in WCSS in figure 50 and figure 51 if compared to their initial values when EM and KM were run on our raw data. The ARI of KM + RP (with 2 clusters) was recorded to be 54% which is still lower than that of the raw data. EM + RP seem to however yield an ARI of 67% which is 4% higher that the ARI of EM clusters on our raw data. We can see also that the histogram suggests a symmetric probability distribution which indicates all data points fall in only one cluster alone. And with the information that these clusters are more similar to the ground truth, we conclude that EM clusters benefited more with RP reduced Wisconsin dataset.

4. Analysis of Neural Net on dimensionally reduced data from 2

The maximum number of hidden layers and maximum number of nodes in each layer were set to be equal to the number of features (for our raw data) or number of components (for our reduced data).

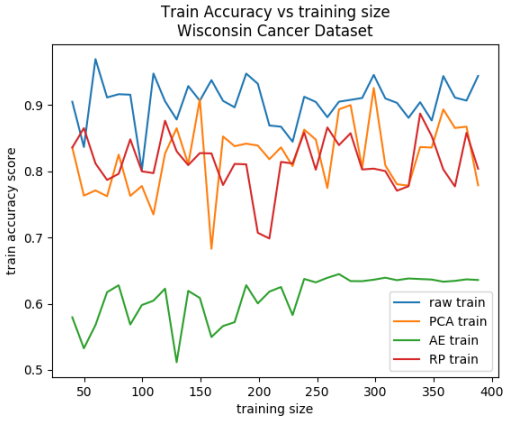

Figure 53

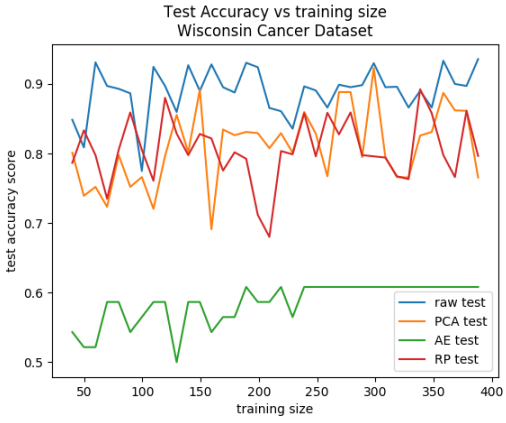

Figure 54

From figure 53 and 54, we can see that NN on our raw data (without reduction) appear to yield the highest training and testing accuracies of around 90%. We also see that the train and test accuracies of NN on PCA, AE and RP reduced data tend to oscillate around 80%. On the other hand, the optimum training and testing accuracies of our NN classifier that can be achieved on our AE reduced dataset appear to be 65% and 60%

respectively. The non-convexity of AE (just like our multi-layer neural network) could explain why NN + AE performed worst. AE is a non-convex technique that suffers from the presence of local optima and as such, we are never guaranteed of a global optimum. Though NN's accuracy with AE was low, we can however observe that it easily gained stability and convergence over the others. This is because AE is non-linear in nature and hence tend to easily learn more complicated relationship between visible and hidden units. The operation of encoders and decoders in AE tend not to only reduce data dimension but also to minimize noise as we can also observe from the graphs above.

5. Analysis of Neural Net on dimensionally reduced data from 2 with clusters from 3 as new features

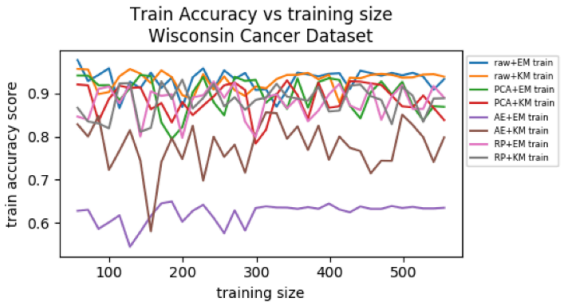

Figure 55

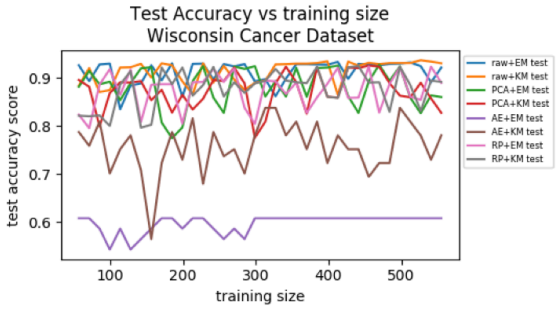

Figure 56

In this experiment, we added our cluster labels from experiment 3 to the reduced Wisconsin data from experiment 2 as new features and run a neural nets learner on the new data. We see from figures 55 and 56 that it didn't matter what cluster labels we added to our raw data as the training and testing accuracies with KM and EM clusters both tend to reach about 95% and 92% respectively. Generally, a slight improvement to the accuracies of NN on our raw data from 90 to 95% (training) and 90 to 92% (testing) were achieved with these clusters. Similarly, we see that it also didn't matter what clusters we added to the RP and PCA reduced datasets as their training and testing accuracies with KM and EM clusters both also tend to reach around 85%. We generally observe an 80% to 85% increase in the training and testing accuracies of our NN classifier on the RP and PCA reduced datasets. On the other hand, we see that EM clusters added to our RP reduced dataset led to a significant increase on NN's training and testing accuracies from 65% to around 80% and from 60% to around 80% respectively. However, there was no effect to adding KM clusters to our AE reduced dataset. Therefore, we see that the type of cluster labels added to our data might contribute to whether NN improves or not.

B. Conclusion

We've been able to see that generally reducing our datasets can help reduce the Within cluster sum of squares between points in our datasets. We also saw that adding clustering labels to our reduced datasets improved the performance of our neural nets classifier. We observed the best ARI score of 70.68% when EM was used on our raw Wisconsin data (without reduction) to generate two clusters. We also saw that no combination of our clustering algorithms on our reduced Pima Indian Diabetes dataset was able to generate clusters that exceeded the ARI score of 23% which was initially achieved when KM was used on our raw Pima Indian Diabetes data to generate two clusters. And lastly, we were able to see that the accuracy of our NN classifier improved to 95% when cluster labels (EM or KM doesn't matter) were added as features to our raw dataset.

Future work that could be done would include training our models with more training sample. As the learning curves above suggests, we might need more training samples to be able to actually observe the exact point of convergence of our models. This will provide clarity as to how we analyze the performance of our neural nets classifier. Also, in our experiments with the random projection algorithm, we used a dense Gaussian random projection matrix which reduces dimensionality by projecting the original input space on a randomly generated matrix. In future experiments, we could use the sparse random matrix instead which guarantees a similar embedding quality but is much more memory efficient and computationally fast. Lastly, in the beginning we performed feature extraction to select the top three most important features. In the future, we could reproduce our experiments with the original number of features. This will allow us to completely gauge the effect of our reduction algorithms on the entire dataset and to observe if feature extraction added any value to the performance of our model.

References

[1] Maaten, Laurens van der, et al. “Dimentionality Reduction: A Comparative Review.” 26 Oct. 2009, pp. 25–26.

[2] Tran, Kenneth. “How Is Autoencoder Compared with Other Dimensionality Reduction Algorithms?” Dimensionality Reduction: Machine Learning, 7 Dec. 2013, www.quora.com/How-is-autoencoder-compared-with-other-dimensionality-reduction-algorithms.

[3] VanderPlas, Jacob T. Python Data Science Handbook: Essential Tools for Working with Data. O'Reilly, 2017.

[4] Liu, WenSui. “Autoencoder for Dimensionality Reduction.” Yet Another Blog in Statistical Computing, 2 Apr. 2017, statcompute.wordpress.com/2017/01/15/autoencoder-for-dimensionality-reduction.

[5] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.