Zach Arnold
CS 7641
Assignment 3

1 Introduction

This paper will explore several unsupervised learning and dimensionality reduction techniques on the datasets I've used for earlier assignments. It begins with a discussion of the datasets, continues with exploration of clustering, then analyzes various dimensionality reduction techniques on neural network performance and finally ends with a combination of clustering and dimensionality reduction as it pertains to NN performance. Unless otherwise stated, scikit-learn's libraries were used to generate models, results, and graphs.

2 Discussion of Datasets

2.1 Dataset 1 Census data to predict salary band

According to the dataset description [2] it contains features which attempt to classify a binary label which is whether or not the sample "makes more than $50k USD" per year. This dataset is interesting due to its size (> 40k rows) which allowed me to cover a significant portion of the space made up of the various attributes. There are 14 attributes in this dataset, 7 continuous valued attributes and 7 categorical attributes. In regards to data distribution with respect to the target attribute, there are approximately 3x as many samples in the category of "less than 50k per year" than in the other category. Given the high number of features it seems ripe for dimensionality reduction, especially because some features seem like they would not be correlated with an outcome (like feature "maritial status")

2.2 Dataset 2 Banking data to predict response

According to the dataset description this comes from the marketing campaign of a Portuguese bank in the 1990s. [2] They were intending to sell their customers on a product called "bank term deposit". The attributes describe various aspects of each customer including details such as age and employment, as well as that customers financial history with the institution and some other details like previous marketing campaign results and economic indicators such as the euribor 3 month note daily rate and consumer price index. This is also a classification problem with the outcome being measured as whether or not they eventually purchased the product in question (binary classification.) Given the high number of features it seems ripe for dimensionality reduction, especially because some features seem like they would not be correlated with an outcome (like feature "euribor3m") There is an even split between categorical features (10/20) and continuous ones (10/20) in this dataset. There are approximately 7x more examples of someone not signing up for this product (as indicated by the "y" value) than those that do.

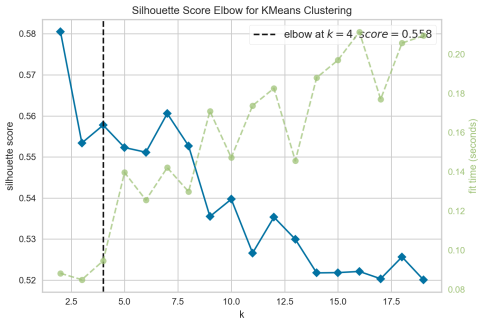

Figure 1: Timing curve vs silhouette coeff. dataset 1

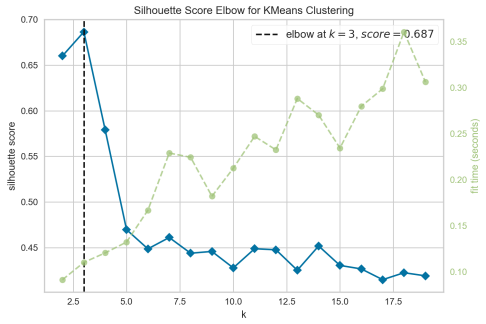

Figure 2: Timing curve vs silhouette coeff. dataset 2

3 Clustering Discussion

3.1 k-means

The two clustering techniques I explored for these datasets are K-Means clustering and Gaussian Mixture Models (Expectation Maximization) In order to choose an appropriate k to do k-means clustering, I used the elbow method [1]. Unless otherwise specified the standard distance measure I used was Euclidean distance. Comparing all cluster cohesion metrics (mean distortion, silhouette coefficient, and Calinski-Harabasz score) here were the elbow results:

	Metric	Cluster Count	Value
Dataset 1	Avg Silhouette Coefficient	4	0.558
Dataset 2	Avg Silhouette Coefficient	3	0.687

Looking at figure 1 and figure 2 you can see similar numbers in terms of the best "k". I chose silhouette coeff over the other cluster cohesion metrics because it has a bias towards preferring well separated clusters (coeffs closer to 1), which I anticipate would improve generalization and overall accuracy during a machine learning process where the clusters are inputs to the learner. Dataset 2 appears to have better separation between its 3 clusters than dataset 1's 4 clusters. Doing further silhouette analysis on DS1 reveals that the clusters when plotted on it to the surface do not appear to have a "cluster" round shape and instead are ovular, whereas data set two had similar characteristics to data set one except that cluster two is actually interspersed in between cluster zero and cluster one, however this was only for the first two features and others could have had better separation/shape. Both clustering methods On both data sets appeared to have two very strongly defined clusters, and then up to two additional weakly defined clusters. This more or less aligns with the fact that this is a binary classification problem and that samples could be placed into one of two clusters. The clusters were more well defined and well

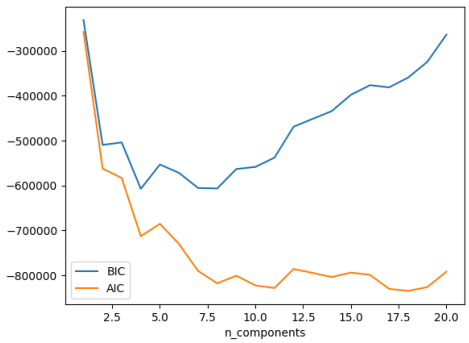

Figure 3: n_components vs AIC/BIC DS1

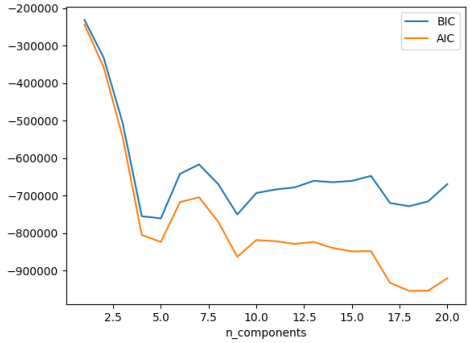

Figure 4: n_components vs AIC/BIC DS2

separated on the second dataset which also makes sense given that all supervised learning methods performed better in terms of accuracy on this dataset relative to the first. On closer inspection, the third and fourth clusters appear to be randomly dispersed throughout the dataset to capture outliers. These clusters likely are noise capturers.

	Normalized Mutual Information	Homogeneity Score	Completeness Score
Dataset 1	0.001	0.002	0.001
Dataset 2	0.189	0.236	0.158

Values closer to 1 are better than 0 in the above table for each score metric. It seems that the performance of K-means is slightly better on dataset 2 than on 1. To improve the performance of this we would likely need to include a dimensionality reduction technique, or modify the dataset so that clusters are more hyper-spherical and separated from each other in the feature space.

3.2 Expectation Maximization

For expectation maximization, I used the scikit-learn Gaussian mixture model function. Based on my exploration, I do not believe that the data are distributed in a hyper-spherical format, therefore a GMM would be able to better group samples based on its ability to accommodate non-spherical hyper-surfaces. In order to select a best value for the number of estimators in the model, I plotted the Bayesian Information Criterion vs the Akaike Information Criterion, where they begin to diverge is likely the best place to stop in terms of n_estimators. In this case looking at figures 3 and 4 it appears to be the same as the optimal K found previously (4 and 3 respectively.) After finding these and running best fits and comparing results I got:

	Normalized Mutual Information	Homogeneity Score	Completeness Score
Dataset 1	0.002	0.002	0.002
Dataset 2	0.133	0.174	0.108

These clusters more or less make sense much in the same way that K-means did. For dataset 1, GMM outperformed in terms of mutual information, homogeneity, and completeness shared by the clustering techniques and the actual labels whereas for dataset 2 K-means outperformed, which indicates that the data was slightly more spherical than I anticipated. However, neither of these clustering techniques did very well since all the values were close to 0 indicating that assigment to clusters was independent and that the clusters themselves were not very independent, nor complete.

4 Dimensionality Reduction Discussion

In this section I explore four methods of dimensionality reduction on both of my datasets.

4.1 Principal Components Analysis

For principal component analysis I chose to vary the number of components to explain variance, and collected their resulting eigenvalues to better understand this DR technique. Below is a table of results for both data sets.

	Minimum Components to Explain 99.9% Variance	Eigenvalues
Dataset 1	2	[11320901200.524, 54484006.504]
Dataset 2	4	[65088.565, 37527.175, 4525.986, 106.278]

Two features explain 100% of the variance in the data in the case of PCA for dimensionality reduction, and four features explain 99.9% of the variance in the case of the second dataset. In the case of dataset one the eigenvalues are very large which would seem to imply that the data are close together and the are being multiplied by a large number to separate. By contrast, data set two seems to have more components needed to explain the variance and each of those as a slightly smaller eigenvalue. Figures 5 and 6 show the projection of the transformed data according to their principal components for the first two features (to allow for plotting).

4.2 Independent Components Analysis

Similar to the previous I also attempted to search for the best number of independent "source components" for each dataset. According to Carsten Klein [3] "An interesting thing about two independent, non-Gaussian signals is that their sum is more Gaussian than any

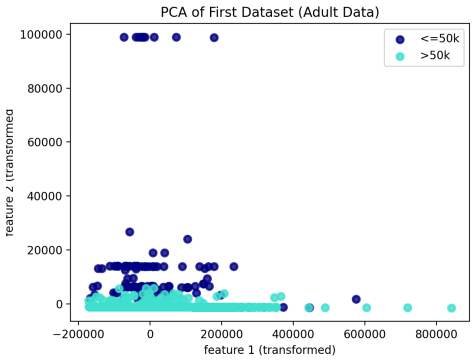

Figure 5: DS1 Projection via PCA

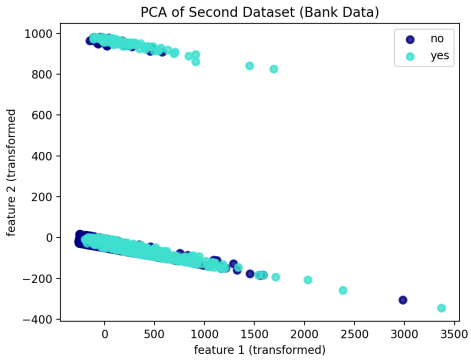

Figure 6: DS1 Projection via PCA

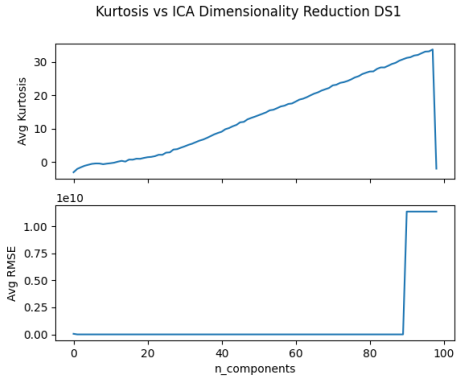

Figure 7: ICA Kurtosis vs RCError DS1

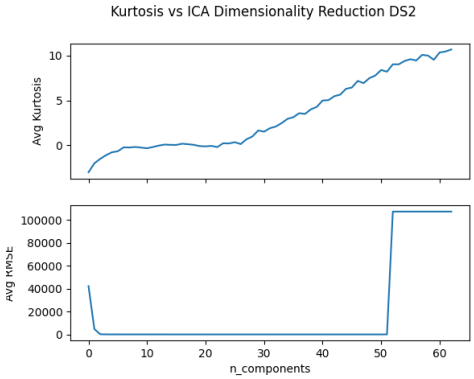

Figure 8: ICA Kurtosis vs RCError DS2

of the source signals. Therefore we need to optimize W in a way that the resulting signals of Wx are as non-Gaussian as possible." In other words, using a measurement like Kurtosis we can compare the "Gaussianity" of different ICA runs. Since the normal distribution (Gaussian) has a kurtosis of 3, we are searching for ICA components with kurtosis of < 3. In order to reduce the covariance of the feature vectors as much as possible (to in effect ensure they were independent components, I elected to whiten the feature space before processing.) Below is a table depicting the n-components selected for each dataset and the resulting Kurtosis. Looking at figures 7 and 8 we can see the relationship between the number of components (the dimensions reduced to) and both Kurtosis and Reconstruction Error (as measured by mean squared error). Similar to PCA, for small values of n_components reconstruction error quickly approaches zero where the kurtosis gradually rises for each additional component. This would seem to suggest that for $n = 2$ (Dataset 1) and $n = 4$ (Dataset 2) we reach the optimal tradeoff between treating each source attribute "as a seperate mixed signal" and reducing RC error (the obvious bias in DR being that we want

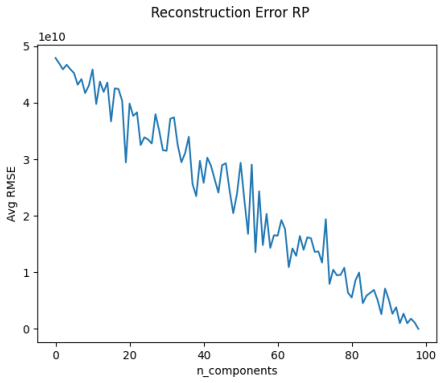

Figure 9: RCError RP DS1

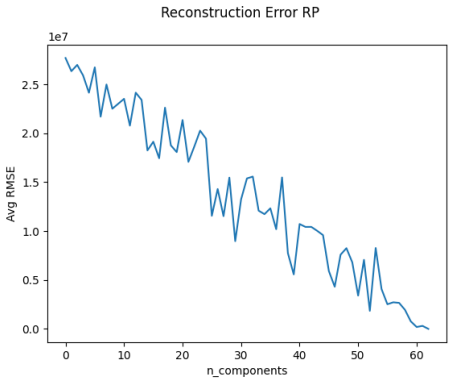

Figure 10: RCError RP DS1

n to be as small as possible.) With respect to each dataset the meaningfulness of this result is that it is possible to reconstruct the distribution with significantly less data than the many more attributes which generated the original.

4.3 Randomized Projection

Randomized projections allowed me to generate random dimensionality reduction transformations from the data. I chose to repeat this process 10 times for each dataset and to utilize a Gaussian Randomized projection such that each feature was itself equally likely to be a part of the final dimensionally reduced output. Along with repeating each experiment I chose to vary the number of components as I did in other experiments to search for an optimal reconstruction error. Perhaps unsurprisingly looking at figures 9 and 10 we can see that for larger values of `n_components` the reconstruction error was lower. The oscillations in the graph can be explained by the fact that the random projections can increase or decrease the RCError by small amounts. There was a significant amount of variance between runs as shown in the table 4.3. Dataset 1 had much more variance than dataset 2 during randomized projection runs. This is most likely due to the 30+ additional features in the space for which a random selection/transformation needed to be made.

	Variance	Standard Deviation
Dataset 1	5.4456097845336664e+16	233358303.57057506
Dataset 2	34540925361.18495	185851.89092711688

4.4 Linear Discriminant Analysis

For my final dimensionality reduction technique, I chose to use linear discriminant analysis. Due to the fact that each of my problems are binary classification, and while not

linearly seperable (otherwise accuracy would be better,) a linear decision boundary with Bayesian conditional density would allow me to reduce the dimensionality of the dataset similar to PCA in the direction of the largest discriminant. However the difference here is the introduction of a priori information in the form of class labels. The number of components had to be $min(num\_features, unique\_labels - 1)$ which in my case was 1 for both datasets given that they were both binary classification problems. Since LDA takes into account class labels I chose (similar to classifiers) to split the data between a test/training set (25/75% respectively.) After the transformation I opted to classify the remainder of instances, below is the result on test data.

	Accuracy
Dataset 1	82.86%
Dataset 2	91.17%

As you can see these results are comparable to the accuracy scores of some of the better trained classifiers from assignment 1. This I believe speaks to the power of LDA being able to consider apriori information in constructing the DR projection. Unfortunately given that there was only one value for `n_components` I cannot compare reconstruction error over time, but given the positive results, I don't think that the reconstruction error would be very high.

5 Clustering on Dimensionally Reduced Datasets

For this section I performed 16 experiments to compare each of the datasets, clustering algorithm, and dimensionality reduction technique. Below is a table of results for the output metrics I chose to compare.

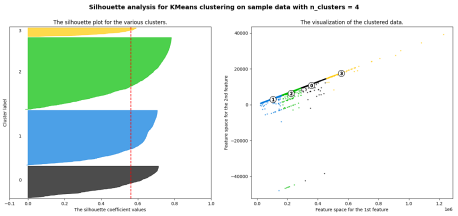

Figure 11: DS1 Kmeans RP

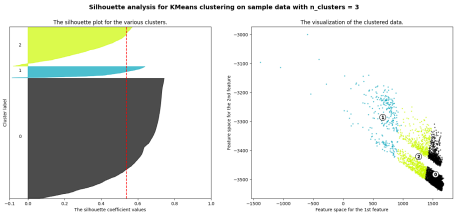

Figure 12: DS2 Kmeans RP

	Normalized Mutual Information	Homogeneity Score	Completeness Score
DS1 + KMEANS + PCA	0.001	0.002	0.001
DS1 + KMEANS + ICA	0.009	0.126	0.001
DS1 + KMEANS + RP	0.001	0.001	0.001
DS1 + KMEANS + LDA	0.230	0.370	0.167
DS1 + GMM + PCA	0.075	0.072	0.079
DS1 + GMM + ICA	0.093	0.075	0.120
DS1 + GMM + RP	0.053	0.073	0.042
DS1 + GMM + LDA	0.238	0.366	0.177
DS2 + KMEANS + PCA	0.189	0.236	0.158
DS2 + KMEANS + ICA	0.096	0.153	0.070
DS2 + KMEANS + RP	0.070	0.106	0.053
DS2 + KMEANS + LDA	0.260	0.413	0.190
DS2 + GMM + PCA	0.236	0.346	0.179
DS2 + GMM + ICA	0.096	0.153	0.070
DS2 + GMM + RP	0.234	0.345	0.178
DS2 + GMM + LDA	0.259	0.421	0.187

Comparing these values you can see that for a reasonable tolerance (relative to each output metric) we have roughly similar clusters, and in certain cases clusters which are more homogeneous and complete than the original, implying that DR was a net positive. Specifically let's look at 11 and 12 and understand a bit deeper how datasets 1 and 2 compared for Kmeans and randomized projections. I chose these (there are many more pictures in the repo) specifically because they showed the largest separation between clusters and because it demonstrates just how good a random projection can be. Look at the separation at the sillohette coeff (red dotted line) for each one, and then look at the cluster labeling for the first and second feature. The clusters appear to be (despite not having clear shapes) to be well formed.

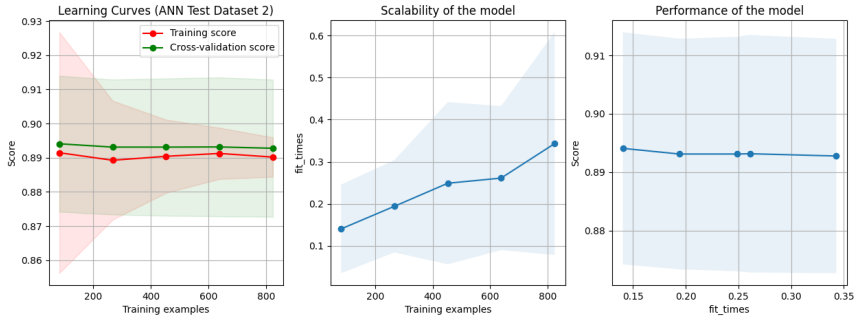

Figure 13: NN Learning Curves on DR Data

6 NN Learning on Clustered Datasets

Taking the DR discussion further I also used LDA to construct an `MLPClassifier` from dimensionally reduced data. The choice to use LDA was primarily based on its performance in the "completeness" metric during clustering comparison. Additionally I needed to make the results repeatable which would rule out RP. I chose to focus on dataset 2 given that classifiers have better performance on it and therefore DR would have to improve upon the result to be considered effective. I chose (initially and then tuned) the same parameters as assignment 1 for dataset 2 as hyperparameters for an initial comparison. In figure 13 you can see the learning and validation curves for this training run. Compared with the original neural network in assignment one for data set two, there appears to be a 2% increase in accuracy overall for both training and testing sets. Additionally, it appears that the cross validation score is very close to the training score and has more samples are produced the standard deviation grows tighter around the data. The scalability of the model appears to grow logarithmically with the number of samples, and it seems that the best score is achieved right away in terms of wall clock time. In terms of training time, and overall performance, dimensionality reduction has improved the performance of the neural network for this data set.

7 NN Learning on Clustered, Dimensionally Reduced Datasets

In this final section, I explore the relationship that adding an additional attribute which is the output of the clustering to the dimensionally reduced data set on performance of the neural network. In theory, despite increasing the dimension by one, this should either improve accuracy or lower fit time due to the fact that we are adding some a priori information to the neural network classifier. Figure 14 shows the results of the training. Compared with the previous training run, the only difference to note is that the training and cross validation scores are closer together and the standard deviation of these results

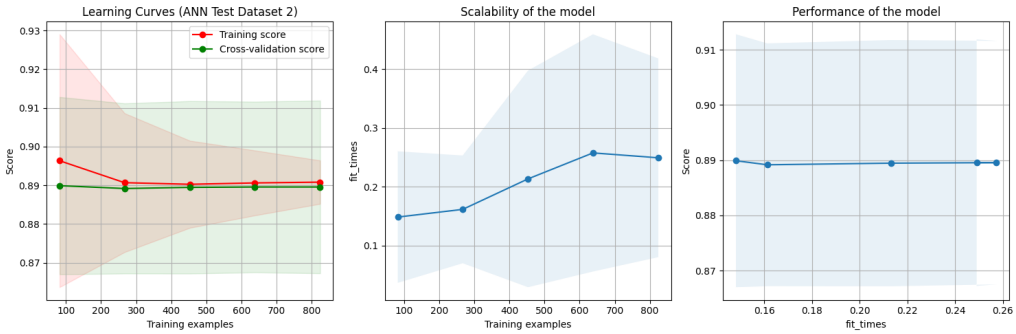

Figure 14: NN Learning Curves on DR, Clustered Data

are slightly closer. However, the scalability and performance of the model as a function of training samples or wall clock time appears to be very similar to the previous training run. Similar to the last neural network, this neural network has a 2% performance increase over assignment ones result in terms of accuracy score. We might possibly improve this result by using other dimensionality reduction techniques, beyond this in order to further improve our results we would need either more or better data that more completely describes the feature space in terms of binary classification.

8 Conclusion

In this paper, I have investigated clustering, dimensionality reduction, and binary classification of multiple data sets in order to explore the relationships between these techniques and their impact on accuracy and training times of neural networks. The results were a successfully dimensionally reduced data set and improvements on accuracy and training time relative to the original results in assignment one.

References

[1] scimitar-yb developers. Elbow method, 2020.

[2] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[3] Carsten Klein. Separating mixed signals with independent component analysis, May 2019.