CS 7641 Assignment 3

Student: Yinglin Li (yli973)   Email: allamli@gatech.edu

1. Data Description

1.1. Description of Dataset 1 Digits Recognition

This dataset was used in the assignment1. It was made by NIST dataset (a large written digits dataset) through extracting the normalized bitmaps of handwritten digits from a preprinted form. This dataset contains 5620 instances (3822 training instances and 1798 test instances) and each instance has 8*8=64 attributes with an input of the integers between 0 to 16. The labels of each instance are 0 to 9 totally 10 labels, which means the corresponding written digits.

1.2. Description of Dataset 2 User Modeling Data Set

This dataset was a real dataset about the students' knowledge status about the subject of Electrical DC Machines which had been obtained from PHD thesis. There are 5 features for each instance, they are STG(the degree of study time for goal objects), SCG(the degree of repetition number of user for goal objects), STR(the degree of study time of user for related objects), LPR(the exam performance of user for related objects), PEG(the exam performance of user for goal objects). According to these 5 attributes, the candidates were classified into 4 levels(high, middle, low, very low) of related knowledge by intuitive knowledge classifiers.

2. Evaluation Explanation

2.1. Silhouette Coefficient Score

The Silhouette Coefficient value of metrics.silhouette_score of ***Scikit-learn*** to describe the clustering of the dataset with the ground truth labels unknown. A higher Silhouette relates to a model with better defined(denser) clusters.

Silhouette Score is bound in range [-1, 1], where -1 means incorrect clustering and 1 is for highly dense clustering so higher score means clusters denser and better separated.

The silhouette Coefficient *s* for a single sample is:

$$s = \frac{b - a}{max(a, b)}$$

Where *a* means the mean distance between a sample and all other points in the same class, *b* is the mean distance between a sample and other points in the next nearest cluster. And the Silhouette Coefficient score we use is the mean of the Silhouette Coefficient for every samples.

Here, we use Silhouette score to determine the best clustering number k for the clustering algorithm.

2.2. Adjusted Rand Index(ARI) Score

Adjusted Rand Index is a function in metrics.adjusted_rand_score of ***Scikit-learn*** to measure how similar the two clustering assignments are, ignoring the permutations of the assignments and with chance normalization for different kinds of the assignments.

ARI's bounded range is [-1, 1], where negative values show the bad result with independent labelings. Similar clusters assignments have a positive ARI and 1.0 is the perfect match score.

The unajusted(raw) Rand Index value is given by:

$$\mathbf{RI} = \frac{a+b}{C_2^n}$$

Where *a* is the number of pairs of elements that are in the same set in true set and the predict set, and *b* is the number of pairs of elements that are in different sets in true set from predict set. *C* is the total number of possible pairs in the dataset with *n* samples.

To counter the effect that RI score does not always guarantee the random input will get a value close to 0, it defines the adjusted Rand Index value as:

$$\mathbf{ARI} = \frac{\mathbf{RI} - E[\mathbf{RI}]}{max(\mathbf{RI}) - E[\mathbf{RI}]}$$

In this paper, we use ARI score to evaluate the performance of the clustering assignment comparing to the its true label.

3. Experiment 1: Clustering Algorithm

3.1. Description and Overview

In this experiment, we need apply the two clustering algorithms: k-means and Expectation Maximization to dataset1 and dataset2. Here, I used KMeans and Gaussian Mixture Model in ***Scikit-learn*** to do the experiment.

Although both dataset1 and dataset2 had grounded truth label for each instance, I still used Silhouette Coefficient Score to test to get the best number of clustering for each dataset. At the same time, we used ARI score to evaluate whether it is suitable to choose such a k value.

3.2. Detailed Analysis

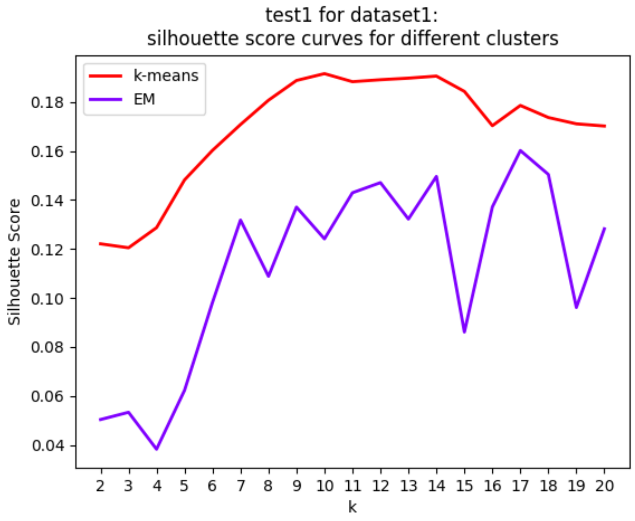

Fig 3.1. Silhouette score of dataset1

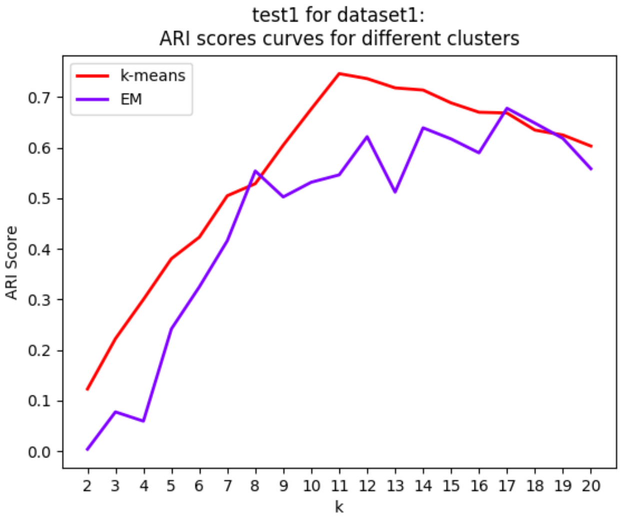

Fig 3.2. ARI score of dataset1

Dataset1 originally had 10 different kinds of labels, so we tested the k from 2 to 20 in k-means and EM clustering algorithms. In figure 3.1, we could see that when k = 10, the silhouette score was the best, but the values around 8-14 were very close. When we checked the figure 3.2 about the ARI score, there was an obvious mountain peak where k = 12 scored the best ARI value. As the result, we should choose k=12 as the best k value.

For EM algorithm, we could see that when k ranged in [8, 18], it was very unstable in Silhouette value. It was the same in figure 3.2 for ARI score. In this situation, we would pick k = 10, the original labelings of the dataset for EM as the best k.

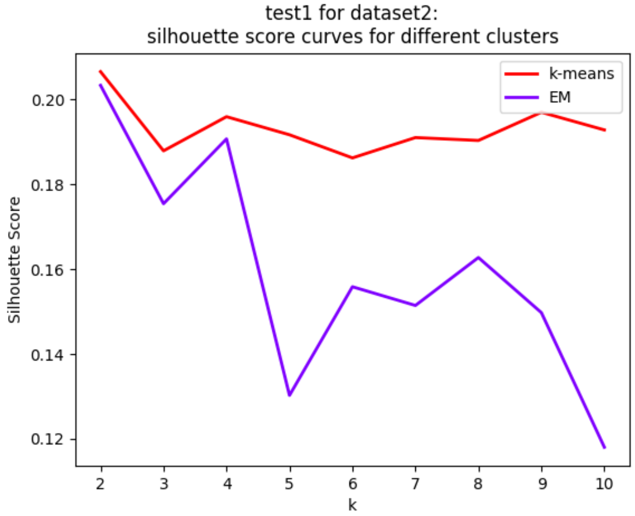

Fig 3.3. Silhouette score of dataset2

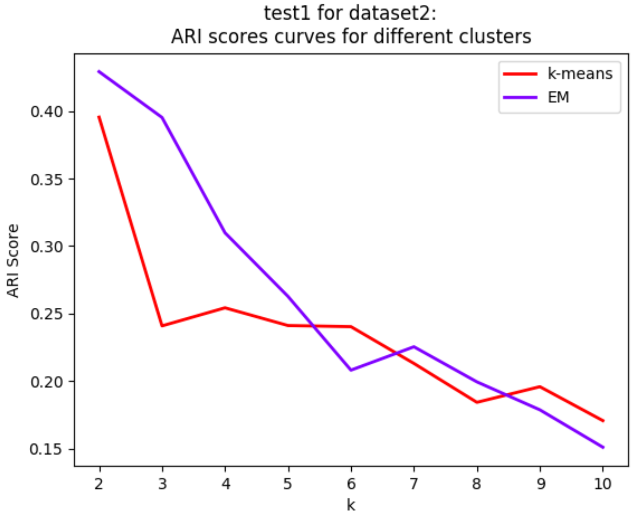

Fig 3.4. ARI score of dataset2

Dataset2 had 4 labels in its raw data. But however, both Silhouette score and ARI score showed that k = 2 was the best clusters number for this dataset. It was very possible because the raw data had 4 labels, it was quiet close to the number we got. However, a not high ARI score showed that maybe this data was not suitable being clustered by k-means and EM.

3.3. Algorithms Summary

3.3.1. K-means Clustering

The K-means algorithm tries to cluster the data by separating the samples into k groups of equal variance by minimizing the sum of squares within each cluster. K-means needs to specify the number of clusters k to get the clustering separation. However, some degree like KNN, it must need all the attributes of an instance to contribute to the clustering algorithm equally. As the result, PCA, ICA or other features selection algorithm would help a lot in k-means to get well-separated clusters.

3.3.2. Expectation Maximization(EM)

The EM algorithm is aimed to find the maximum likelihood of an instance's partition. Therefore, it uses probability of which partition each instance belongs to instead of an exact cluster. This model describes the likelihood model and latency with Gaussian Mixture Model to minimize the means and covariance of each.

4. Experiment 2: Dimensionality Reduction

4.1. Description and Overview

In experiment2, we need use PCA, ICA, Randomized Projection and choose one of the other feature selection algorithm to apply in the two datasets to reduce their dimensionality and describe the result. Here, I chose Truncated Singular Value Decomposition(TruncatedSVD) in ***scikit-learn*** as the fourth feature selection algorithm. The difference between PCA and truncated SVD will be explained below.

For PCA, sklearn.decomposition.PCA of ***scikit-learn*** was used. For ICA, in sklearn.decomposition.FastICA of ***scikit-learn*** was used. For RP, sklearn.random_projection.GaussianRandomProjection of ***scikit-learn*** was used. For SVD, sklearn.decomposition.TruncatedSVD of ***scikit-learn*** was used.

Since it's difficult to measure or describe the data after dimensionality reduction and feature selection, we used the error of data after reconstruction through there four algorithms to evaluate their performance.

4.2. Detailed Analysis

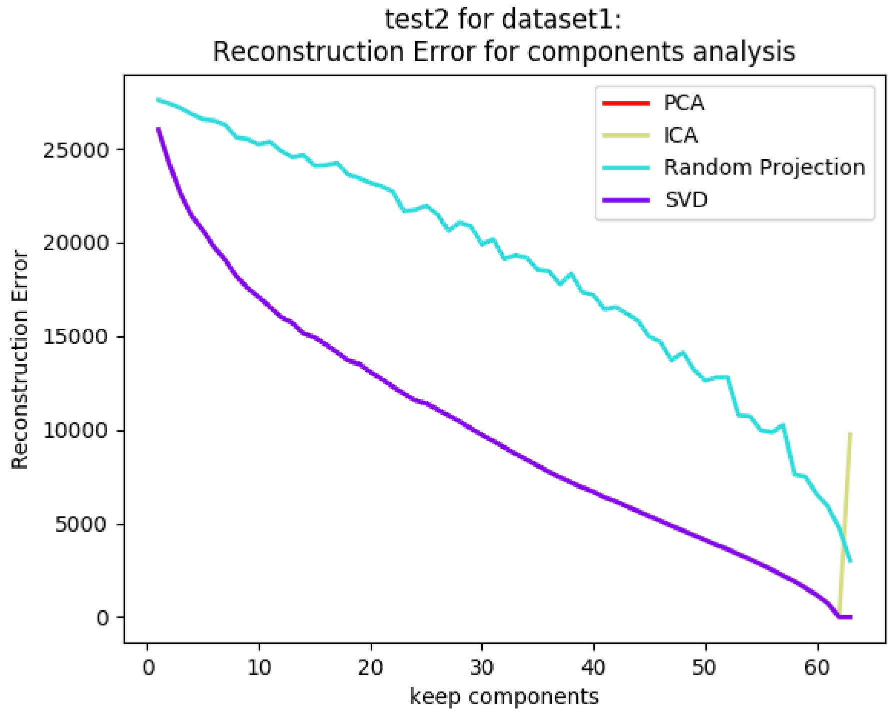

Fig 4.1. Reconstruction error curves of dataset1

In this experiment, we could see that in both dataset1 and dataset2, the results of PCA, Randomized Projection and SVD were so close to each other that we could only see one curve! It seemed very strange and surprising that the reconstruction error of these three dimensionality reduction algorithms were so similar!

But in fact, they had a bit difference but the scale was so large that we could not see this kind of difference. Moreover, in ***scikit-learn***, the implementation of these three algorithms were based on different kinds of SVD transformation but they were still SVD transformation so the reconstruction result would be such similar that we could nearly see the difference.

And obviously, due to its random process, Randomized Projection would have a larger error comparing to the other three algorithms.

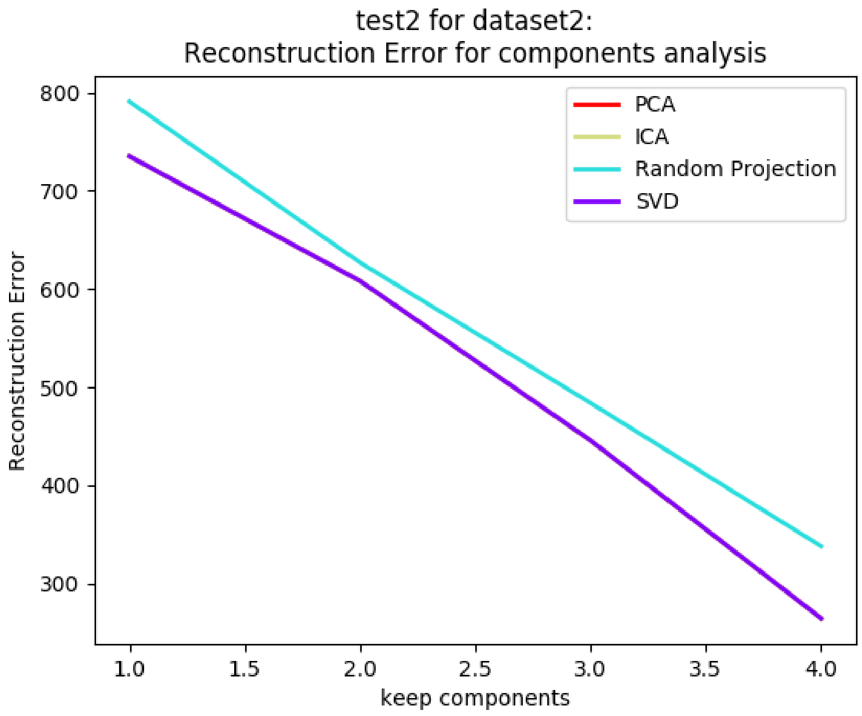

Fig 4.2. Reconstruction error curves of dataset1

4.3. Algorithms Summary

4.3.1. Principal Components Analysis(PCA)

PCA is used to decompose a multivariate dataset in a set of successive orthogonal vectors that could maximize the variance of these vectors. It produced an orthogonal transformation to convert a matrix into an orthogonal matrix so that maximize the variance of kept features. The vectors of the matrix are uncorrelated but PCA is sensitive to relative scaling of the original data.

4.3.2. Independent Components Analysis(ICA)

As its name means, ICA is used to find the most independent components of the given features. It could reveal the hidden factors which are under different kinds of situations such like random values, measurements. However, as an extension of PCA to find the features that are unrelated to the data, it is strong enough to find all the underlying factors but it works badly when all the features contributing to the model.

4.3.3. Randomized Projection

Here, we used Gaussian Random Projection as a process of randomized projection. It reduces the dimensionality by projecting the original data on a randomly generated Gaussian distribution matrix.

4.3.4. Truncated Singular Value Decomposition(Truncated SVD)

SVD is a matrix factorization in Linear Algebra. It is to generate the eigen decomposition of a positive semi-define normal matrix to m*n matrix via an extension of polar decomposition. However, PCA could not deal with the sparse matrix but Truncated SVD could deal with it by not centering the data before computing the SVD. As the result, here we could see that the result was similar to PCA because all the inputs were not a sparse matrix.

5. Experiment 3: Apply Dimensionality Reduction to Clustering

5.1. Description and Overview

In this experiment, we need to apply the four dimensionality reduction algorithms in experiment 2 for each dataset and then reproduce the clustering algorithms of k-means and EM in experiment 1.

Here, we used adjusted Rand Index (ARI) Value mentioned in 2.2 to evaluate the performance of the clustering algorithm for the data after dimensionality(features) reduction. Since we knew the truth labels of every instances for each dataset, so we compared the ARI value before and after dimensionality reduction for each dataset.

5.2. Detailed Analysis

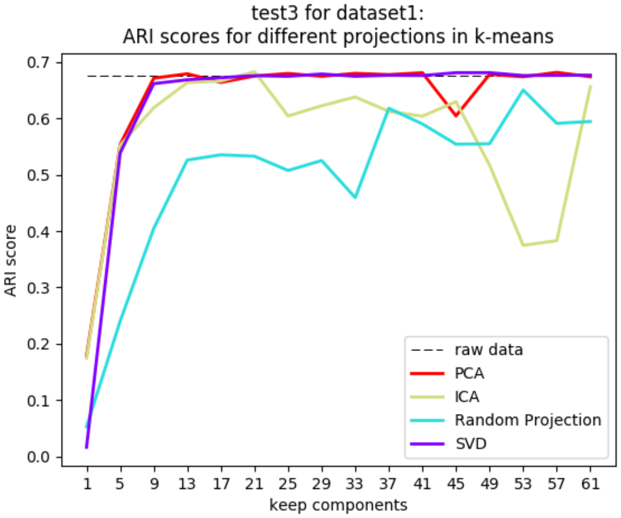

Fig 5.1. ARI scores with dimension reduction in k-means of dataset1

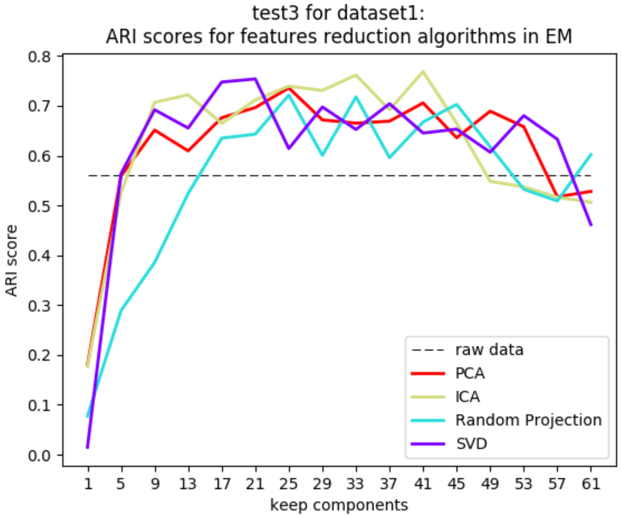

Fig 5.2. ARI scores with dimension reduction in EM of dataset1

In dataset1 for k-means algorithm, we could see that all curves of the four dimensionality reduction algorithms could converge very fast. With k equaled about 13, they had converged to the best ARI values for each algorithm.

However, PCA and SVD had the best ARI score and they converged fastest. This means that PCA and SVD needed a small number of components to converge. You may notice the PCA score in 45 was a bit strange and I think it's the reason of random state that made it smaller than its neighbors.

But you can see that, after k = 25, the score of ICA became unstable, that was because most of the features in this dataset1 of each instance mattered a lot to each other. Therefore, it became normal when the features close to the original number.

For EM algorithm. The result was close to the k-means algorithm but they became unstable after k = 13. That was the reason of the random state used in the EM algorithm.

At last, notice that this dataset was made of images of 8*8=64 features, which meant that, most of the features were 0 for each instance. As the result, PCA and SVD were perfect to get reduce the dimensionality of features, and ICA would have a lower peak. Moreover, from figure 5.1 and 5.2, we could see that when k = 10, the number of dataset's original label number, PCA and ICA reached the best ARI score.

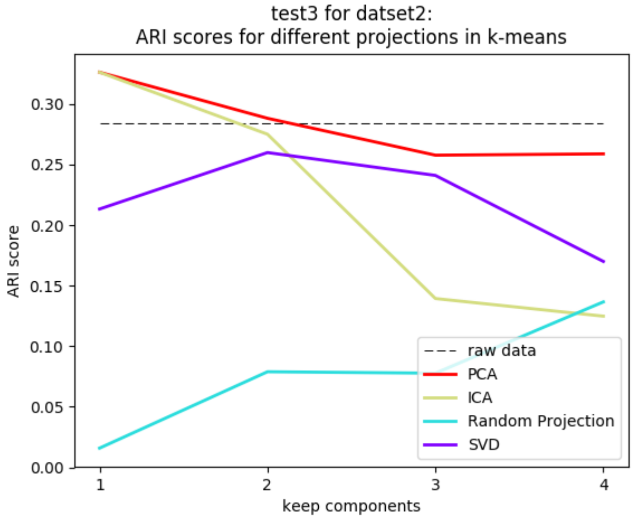

Fig 5.3. ARI scores with dimension reduction in k-means of dataset2

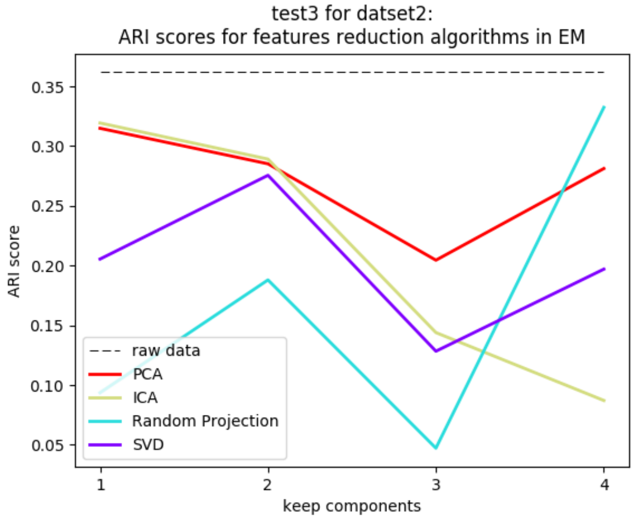

Fig 5.4. ARI scores with dimension reduction in EM of dataset2

In dataset2, since the features number was only 5, it was difficult to see the tendency of how the dimensionality reduction algorithms made effect to the clustering algorithms. But we could still find some interesting result from these two figures.

In general, EM algorithm got a better ARI score that k-means algorithm. In figure 5.3, we could see PCA always got the best ARI score for all the x values. But in figure 5.4, we could see that Randomized Projection got the best score with components = 4 because its “random” process (random not always made a bad result).

However, since it was difficult to describe the data after dimensionality reduction so I the data projected to 2D dimension would be helpful for us to understand how it worked for the data. Following were the graphs of PCA dimensionality reduction with component = 2 for dataset1 and dataset2 clustering in k-means.

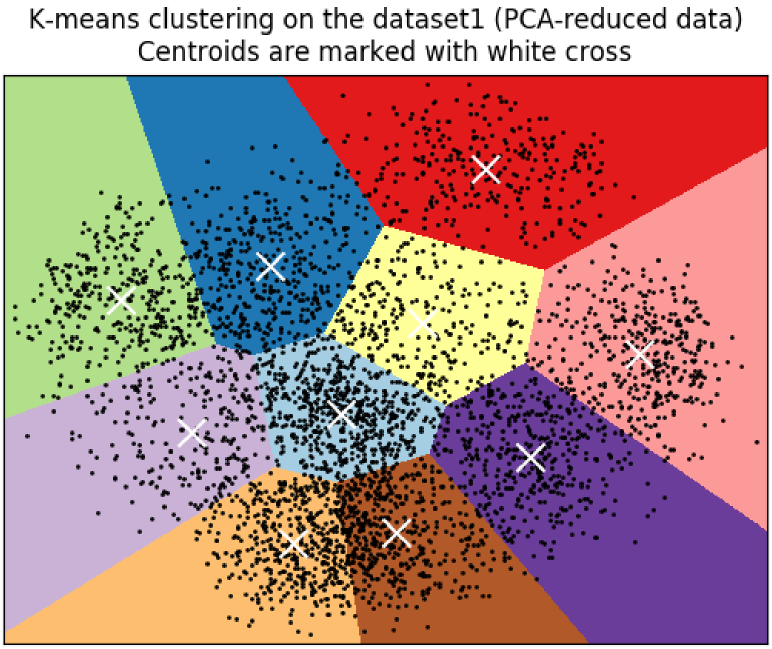

Fig 5.5. k-means with PCA(n=2) on dataset1

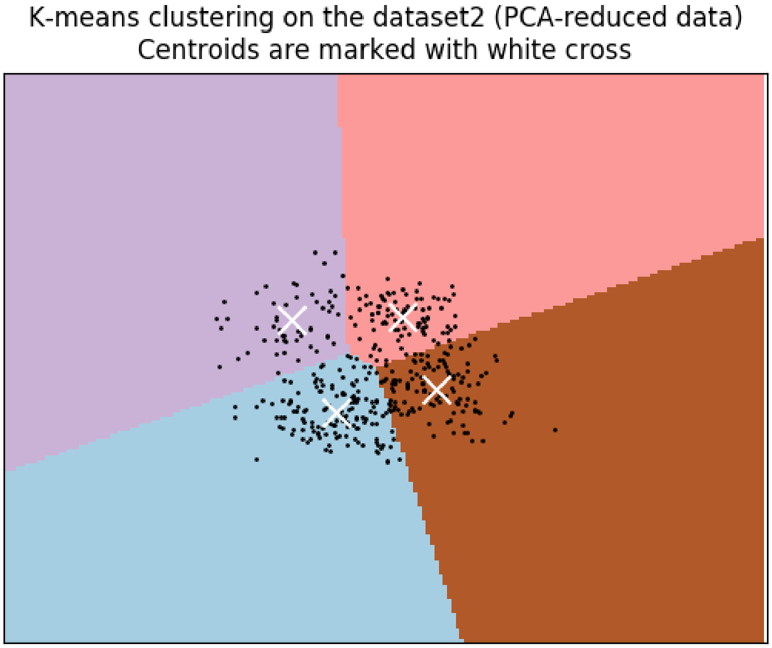

Fig 5.6. k-means with PCA(n=2) on dataset2

In figure 5.5, although the label (clustering) ARI score was 0.3, it was clear to divide the samples into 10 different areas to make them well-separated.

In figure 5.6, the samples were close to each other and it was hard to tell 4 labels were a good clustering separation. However, from the figure we could see that the ARI score was only 0.3 so we could conclude that the k-means clustering result was not quite suitable for this dataset.

6. Experiment 4 and 5: Apply Dimensionality Reduction and Clustering to Neural Network

6.1. Description and Overview

For experiment 4, we need to apply dimensionality reduction algorithms to a dataset and return its Neural Network learner to see the differences in performance.

For experiment 5, apply the clustering algorithm for the same dataset first and then treat the clusters as a new feature for each instance and reproduce the experiment 4.

Here in the experiment 4 and 5, we used the dataset1 digits recognition which was also used in assignment 1. Since experiment 4 and 5 were similar, so we did the tests and analyzed the results together to see what differences dimensionality reduction and clusters would bring to the Neural Network training.

In assignment1, the training accuracy of this dataset was 100% for training data and testing accuracy was 96.60%, and the total running time was 0.87s.

6.2. Detailed Analysis

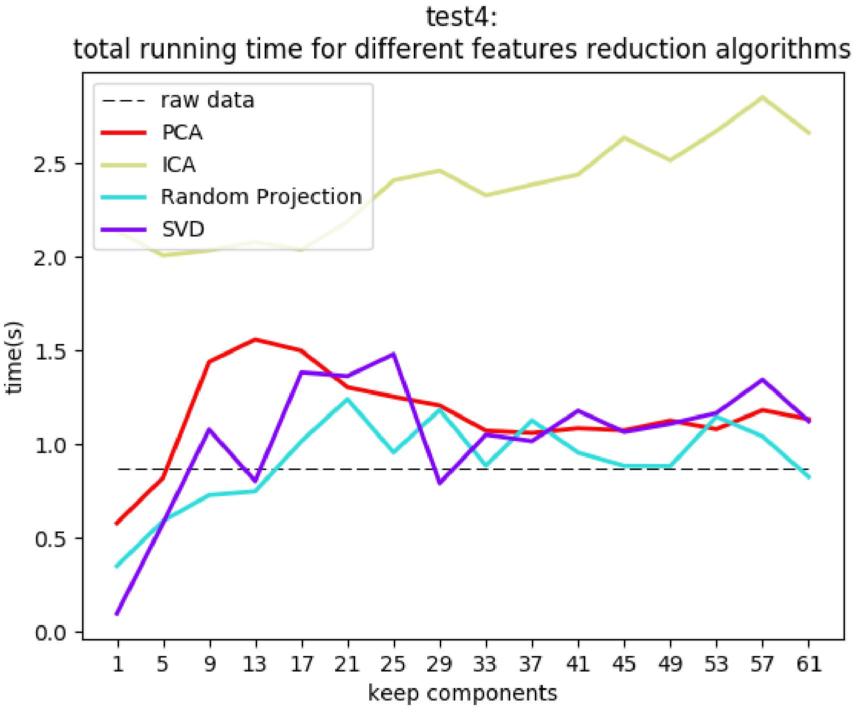

Fig 6.1. Time complexity of experiment 4

Let's start with the time performance improvement. In the figure 6.1, we could see that when the kept components number was small(about 10), PCA, RP and SVD spent time faster than the raw data. When the number of kept components was larger, these three algorithms of features selection was a bit slower than the original one but just very close to it. That was because the number of iterations to make the Neural Network to converge was larger.

However, the ICA method was much slower than the raw Neural Network and it seemed that it has a linear relation to the kept components. The reason why ICA was so slow was that each of the neural network could not converge with in max iterations.

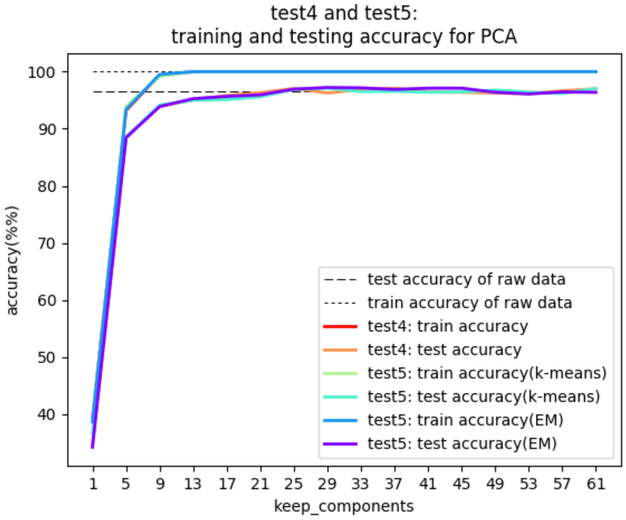

Fig 6.2. Training curves of experiment4 and 5 of PCA

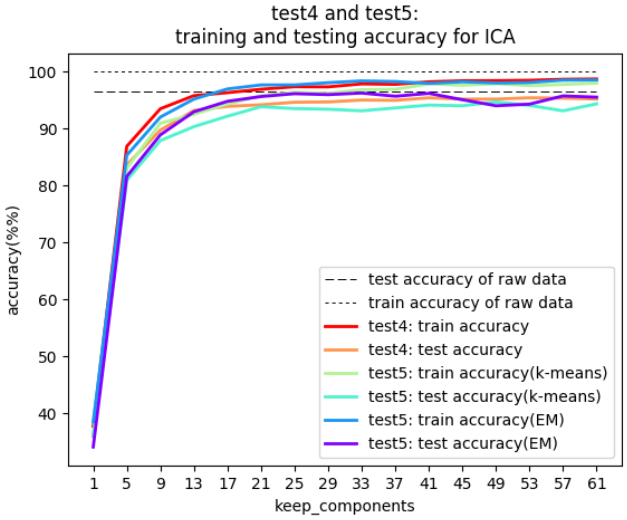

Fig 6.3 Training curves of experiment4 and 5 of ICA

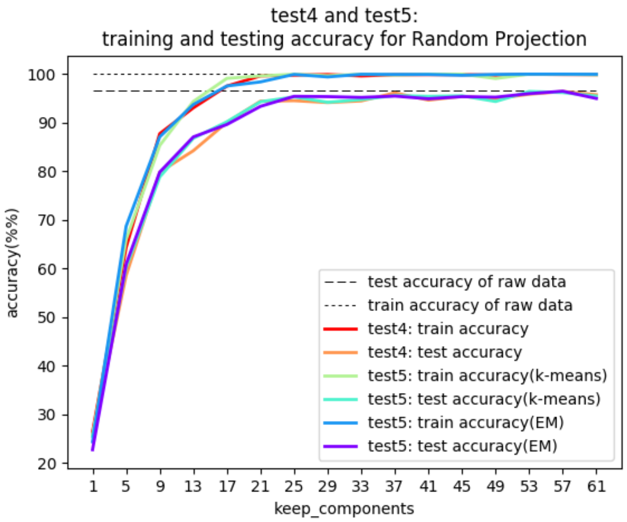

Fig 6.4. Training curves of experiment4 and 5 of RP

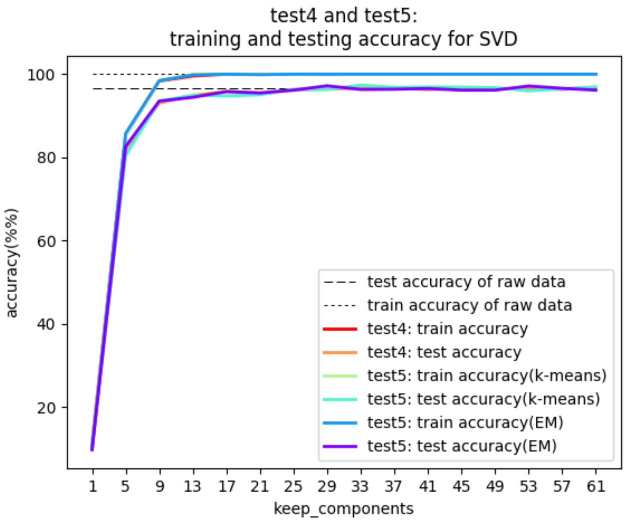

Fig 6.5 Training curves of experiment4 and 5 of SVD

In figure 6.3 for ICA, because the Neural Network didn't converge all the time, both training accuracy and testing accuracy in experiment 4 and 5 could not reach the accuracy of the raw data. In experiment 5, the performance of k-means and EM clustering result showed a similar effect to the Neural Network.

Then, in the figure 6.2, 6.3 and 6.5, we could see that the values of experiment 4 and 5 with PCA, RP and SVD would converge to the values of raw data. In the experiment 2 and 3, we could conclude that PCA and SVD could show a similar performance on this dataset. As the result, they also performance similarly to the neural network but PCA would converge a little faster. For Randomized Projection, it converged much slower than the other 3 methods but it could also get converged to the same result as the Neural Network of raw data. However, all the four algorithms could converge in a small number kept components. It means that most of the features were useless when the instances were applied to the Neural Network.

From the four figures, we could find that both the two clustering algorithms showed no difference when they were applied to get a new feature for the data with four different kinds of dimensionality reduction algorithms. With the previous experiments, we could conclude that the raw data of this dataset has a good clustering property (easy to separate) itself with all the features could contribute to clustering equally, which also means the clustering algorithm nearly didn't help the classification.

7. Arguments and Problems

7.1. Clustering

In clustering part, the both k-means and EM algorithm could get a similar result for both two datasets. Comparing to k-means, EM algorithm was more unstable because of its random state of Gaussian kernel process. However, the key difference between k-means and EM is that EM is a model to measure the probability of the cluster which each sample belongs to but k-means is to compute the exact clustering label but this difference is difficult to show in the above experiments.

7.2. Dimensionality Reduction

PCA and ICA were strong tools to reduce dimensionality of the features. Most of the implementations were based on the SVD transformation. Moreover, we can see ICA as an extension. However, it's difficult to make use of the inductive bias from learning estimator without testing through this estimator but you know, there are no free lunch. You have to sacrifice time to get to know such inductive bias from learner and make use of them.