# I. Assignment 3 Unsupervised Learning and Dimensionality Reduction

## II. About Data

### A. *Data Set 1*

This dataset was sourced from Kaggle and poses a binary classification challenge. The target variable takes on values of 0 or 1, indicating whether an individual's income falls below or exceeds $50,000. The dataset includes a range of demographic, education, and work-related attributes for each individual, which can be leveraged to predict their income level.

Upon initial examination, there were 8 categorical columns and 5 numerical columns. After cleaning the data by removing rows with missing values, we were left with a total of 30,162 instances for analysis.

To prepare the data for modeling, we transformed the categorical variables into dummy or indicator variables, expanding the feature set to a total of 105 variables.

Regarding the target variable distribution, approximately 75.9% of the instances are labeled 'Yes' (1), indicating income exceeding $50,000, while the remaining 24.1% are labeled 'No' (0), representing income less than or equal to $50,000.

In Assignment 1, I achieved a test score of 0.9009 for Dataset1. I hope that in this assignment, I will be able to improve this score. Dataset 1, with its imbalanced nature and 105 variables, will serve as a good test for various dimension reduction and clustering algorithms.

### B. *Data Set 2*

This dataset was obtained from Kaggle, and it presents a binary classification problem where the target variable takes on values 0 or 1, representing 'male' and 'female,' respectively. The dataset comprises information related to facial features, including attributes like forehead size, nose characteristics, and lip size and shape. In total, the dataset contains seven features and a label column.

The dataset consists of 5,001 instances, all of which have complete data without any null values. The target variable's distribution is perfectly balanced, with 50% of the instances labeled as 'Male' (1) and the other 50% as 'Female' (0).

In Assignment 1, I achieved a test score of 0.977 for Dataset2. However, this dataset is relatively simple, with balanced data and only 7 features, which makes it less attractive for use in the later parts of this assignment.

## III. Hypotheses

By employing unsupervised learning, especially through the application of clustering and dimensionality reduction algorithms, we can enhance the performance of clustering algorithms, such as neural networks

## IV. Part 1 Clusters

In Part 1, we use Kmeans and Gaussian Mixture (GMM) to apply clustering to our datasets. We utilize the Silhouette Score to measure the performance of clustering algorithms.

TABLE I
Clustring

| | DataSet1 | DataSet2 |
|---|---|---|
| Kmeans silhouette score | 0.16829 | 0.42804 |
| Kmeans num of clusters | 3 | 2 |
| GussianMixture silhouette score | 0.168251 | 0.43644 |
| GussianMixture num of clusters | 3 | 2 |

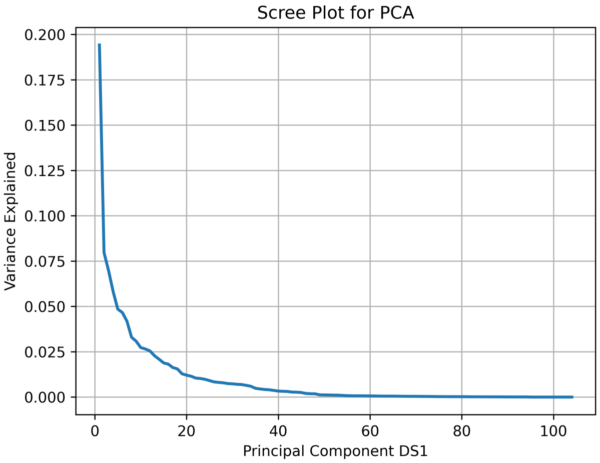

Fig. 1. Principal Component Analysis DS1

The Silhouette Score is easy to understand, applicable to different algorithms, and can be calculated without the need for ground truth labels.

Table 1 shows that both clustering algorithms produce nearly identical results. We can hypothesize that both datasets represent binary classification problems, with all features almost entirely separated into binary groups. Both algorithms yield a score of approximately 0.42 for Dataset2. This high score suggests that the clusters are well-separated and coherent.

## V. Part 2 Dimensionality Reduction

In this section, we will explore different algorithms to reduce the data into lower dimensions while retaining as much information as possible. This is helpful for eliminating unnecessary data or noise from the dataset.

### A. *Principal Component Analysis (PCA)*

PCA is a popular unsupervised linear algorithm used to reduce the dimensionality of a dataset. It assumes that variables are linear with each other and orthogonal to each other. To determine the suitable number of components that can explain the maximum data, we draw a graph between the number of components and the percentage of variance explained by each of the selected components, also known as the explained variance ratio. We then select the appropriate number of components using the elbow method.

When looking at the graph of Dataset1 in Figure 1, it becomes apparent that selecting the number of components

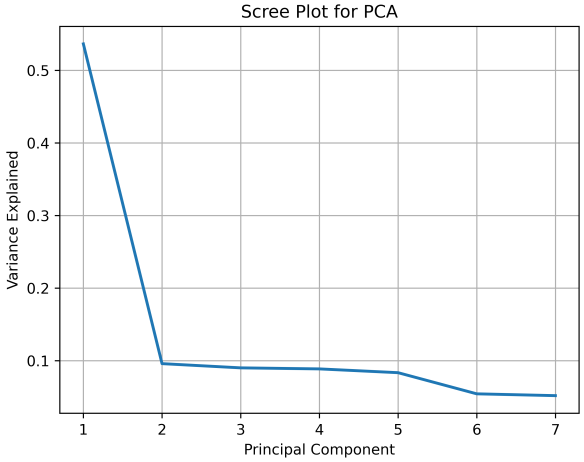

Fig. 2. Principal Component Analysis DS2

using the elbow method is challenging. To address this, I experimented with different numbers of components and used Kmeans to calculate the Silhouette score for each number. I ultimately selected 2 as the number of components because it produces the highest Silhouette score.

For Dataset 2, it's clear that two components can explain 80% of the data. This implies that we can reduce the data to lower dimensions without significant information loss.

### B. *Independent Component Analysis (ICA)*

This is also an unsupervised linear algorithm used to reduce the dimensionality of the dataset. This algorithm assumes that the data is linearly independent and non-Gaussian.

To select the best number of components, we use Kurtosis. Figures 3 and 4 show the average Kurtosis vs. the number of components for Dataset1 and Dataset2.

From Figure 3, we can observe that as the number of components increases, the average Kurtosis value also increases. This implies that we are unable to apply the elbow method, suggesting that the data may not be linearly independent or follows a Gaussian distribution (probability distributions following a normal distribution).

To proceed with ICA, I calculated the absolute value of Kurtosis for each column while providing the maximum value of components in the algorithm. By analyzing the output, I determined that out of 104 columns, 33 have Kurtosis values above average. This indicates that these 33 columns have heavier tails or outliers. Similar to PCA, I experimented with different numbers of components and used Kmeans to calculate the Silhouette score for each number. I selected 3 as n_components that produced the highest Silhouette score.

For Dataset 2, Figure 4 reveals that the Kurtosis value increases with the number of components and reaches a minimum at 5. Therefore, we can choose 5 as the number of components for ICA.

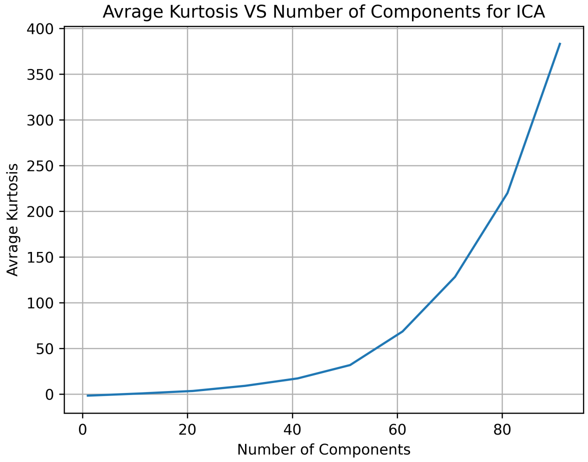

Fig. 3. Independent Component Analysis DS1

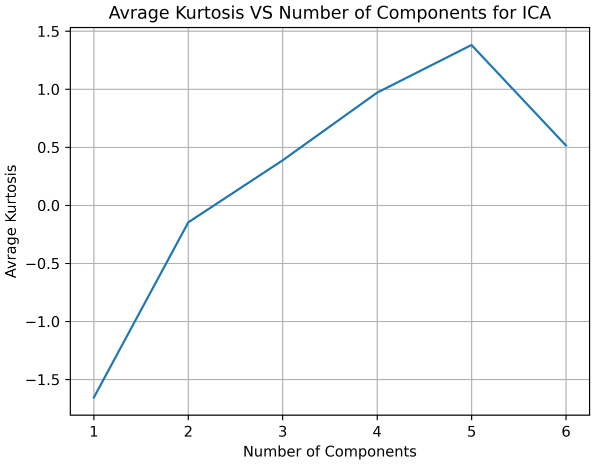

Fig. 4. Independent Component Analysis DS2

### C. *Random Projection (RP)*

GaussianRandomProjection uses randomized projections to reduce the dimensionality of data. It assumes that the data is linearly correlated. Random projections can also, to some extent, reduce the impact of noise in the data. In this section, we are utilizing GaussianRandomProjection from scikit-learn and Mean Square Error (MSE) to determine suitable components.

For Dataset 1, Figure 5 shows that MSE decreases as we increase the number of components. It demonstrates a positive correlation between MSE and the number of components, indicating that using more components results in a better reconstruction of the original data. This behavior is expected, as increasing the dimensionality of the reduced data space can lead to a closer approximation of the original data. However, it also makes it challenging to find a balance between the number of components and MSE. just like PCA and ICA I experimented with different numbers of components and used Kmeans to calculate the Silhouette score for each number. I

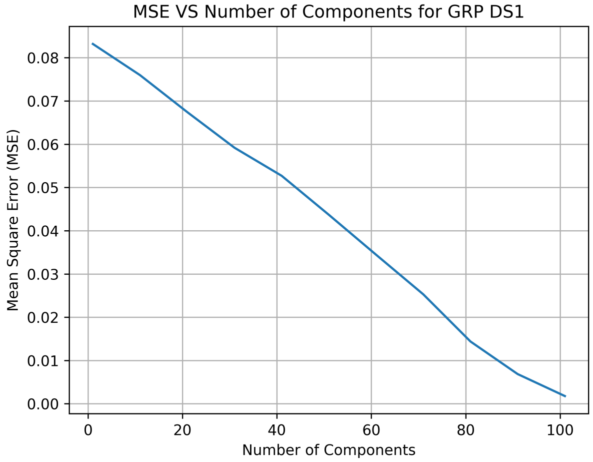

Fig. 5. Random Projection DS1

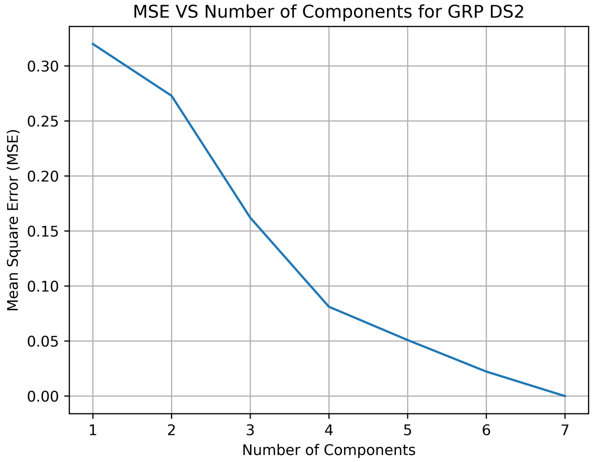

Fig. 6. Random Projection DS2

ultimately selected 4 as the number of components because it produces the highest Silhouette score.

In the case of Dataset 2, Figure 6 reveals that MSE reduces significantly until component 4, after which the rate of decrease diminishes notably. This suggests that we can reconstruct the original data with minimal error if we use 4 components.

### D. *Locally Linear Embedding (LLE)*

For the non-linear Manifold Learning Algorithm, I selected the Locally Linear Embedding (LLE) algorithm. According to the scikit-learn website, Locally Linear Embedding seeks a lower-dimensional projection of the data that preserves distances within local neighborhoods. It can be thought of as a series of local Principal Component Analyses. Since PCA produced better results than other dimensionality reduction algorithms for our datasets, I assumed that this algorithm is better suited for this assignment.

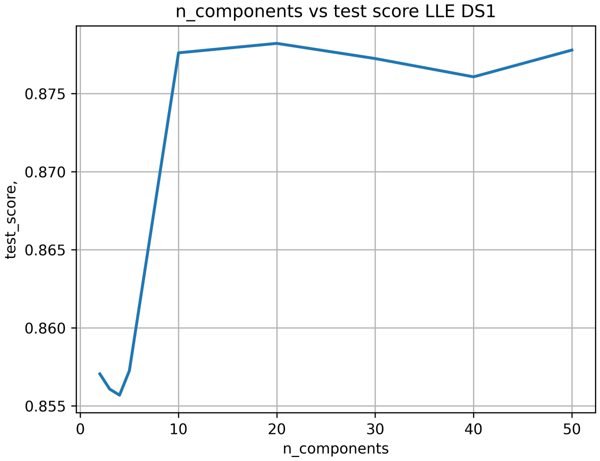

Fig. 7. n_components vs test score LLE DS1

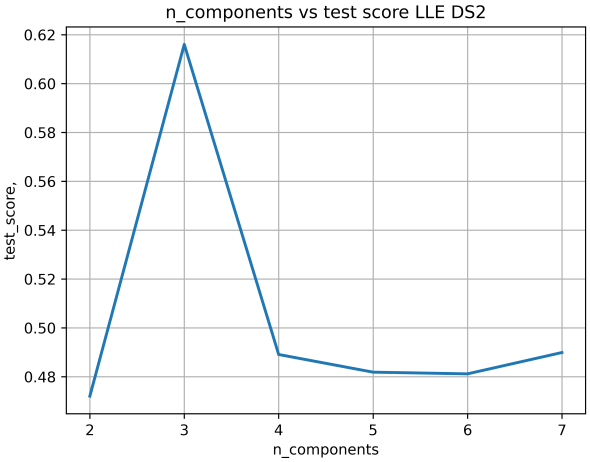

Fig. 8. n_components vs test score LLE DS2

This algorithm requires an appropriate value of K (k nearest neighbors) for the lower-dimensional projection of the data. We provided different values of K to this algorithm and ran a dummy classification algorithm (MLPClassifier) on its output.

In Figure 7, we can observe the values of K and MLPClassifier test scores using Dataset 1. It's evident that the test score increases significantly at K=10.

In Figure 8, we see the values of K and MLPClassifier test scores for Dataset 2. There is a noticeable and significant increase in the test score at K=3.

## VI. Part 3 Clusters after Dimensionality Reduction

In this section we apply clustering algorithms using results of DR algorithms

### A. *Principal Component Analysis (PCA)*

In this section, we will delve into clustering after applying PCA, as it significantly improves the Silhouette score compared to other linear algorithms.

As mentioned earlier in Part 2, subsection PCA, the scree plot of variance and the number of components show a smooth curve. Consequently, we experiment with various values for the hyperparameter n_components to maximize the Silhouette score. Table II displays the PCA components and their respective values for Kmeans and GMM.

When we compare the Silhouette scores of Kmeans and GMM in Table 1, a substantial improvement in the Silhouette score becomes evident. PCA diminishes the data's dimensionality by capturing the most significant variance, resulting in a more focused representation of the data. This reduction in dimensionality aids in eliminating noise and irrelevant information, which might adversely impact clustering algorithms. Furthermore, it is observable that as we increase the number of PCA components, the Silhouette score decreases. As the number of PCA components grows, the dimensionality of the feature space increases. In high-dimensional spaces, data points tend to distribute more uniformly, making it challenging for clustering algorithms to identify well-defined clusters. This phenomenon is commonly referred to as the ”curse of dimensionality.” Additionally, an increase in PCA components corresponds to an increase in the number of clusters. Having more clusters often results in smaller, less meaningful clusters or overlapping clusters, ultimately leading to a decreased Silhouette score

TABLE II
CLUSTER AFTER PCA DATASET1

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| PCA components | Silhouette | Clusters | Silhouette | Clusters |
| 2 | 0.67843 | 4 | 0.653913 | 4 |
| 5 | 0.422188 | 15 | 0.388014 | 17 |
| 10 | 0.227375 | 17 | 0.186843 | 17 |
| 15 | 0.168940 | 17 | 00.166208 | 6 |
| 20 | 0.1792397 | 8 | 0.173979 | 11 |
| 25 | 0.186242758 | 15 | 0.1677358 | 11 |

Table III showing Silhouette score from kmeans and GM algorithems after applying PCA.

TABLE III
CLUSTER AFTER PCA DATASET 2

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| PCA components | Silhouette | Clusters | Silhouette | Clusters |
| 2 | 0.80456 | 9 | 0.8901 | 17 |

TABLE IV
CLUSTER AFTER ICA DATASET 1

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| ICA components | Silhouette | Clusters | Silhouette | Clusters |
| 3 | 0.483005 | 7 | 0.447710 | 7 |
| 5 | 0.42777 | 17 | 0.38517 | 17 |
| 10 | 0.22852 | 16 | 0.189583 | 3 |
| 15 | 0.17045 | 4 | 0.1647316 | 5 |
| 20 | 0.18792 | 9 | 0.167969 | 6 |
| 25 | 0.17111 | 15 | 0.1611934 | 12 |

TABLE V
CLUSTER AFTER ICA DATASET 2

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| ICA components | Silhouette | Clusters | Silhouette | Clusters |
| 5 | 0.83050 | 15 | 0.8874458 | 19 |

TABLE VI
CLUSTER AFTER RP DATASET 1

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| RP components | Silhouette | Clusters | Silhouette | Clusters |
| 4 | 0.2469402 | 2 | 0.24073 | 2 |
| 5 | 0.213468 | 2 | 0.194118 | 2 |
| 10 | 0.15607 | 2 | 0.075502 | 3 |
| 15 | 0.1335421 | 4 | 0.02509 | 2 |
| 20 | 0.131256 | 4 | 0.040959 | 2 |
| 25 | 0.1396422 | 3 | 0.080201 | 2 |

### B. *Locally Linear Embedding (LLE)*

Table VIII displays the Silhouette scores of Kmeans and Gaussian Mixture on Dataset1 after applying LLE to the dataset. When we compare these scores with the Kmeans and GM scores on the raw Dataset1 (Table1), we observe that the Kmeans score decreases, but the GM score significantly increases. . LLE may have altered the overall structure of the dataset while preserving the local linear relationships. This suggests that Kmeans is more sensitive to the overall structure of the dataset compared to GMM.In contrast, GMM uses different Gaussian distributions, enabling it to adapt to more complex data shapes. The improvement in the GMM score also suggests that LLE has preserved the Gaussian distributions while reducing noise.

Table IX displays the Silhouette scores of Kmeans and Gaussian Mixture (GaussianMixture) on Dataset2 after applying LLE to the dataset. When we compare these scores with the Kmeans and GM scores on the raw Dataset2 (Table1), we can observe that both the Kmeans and GM scores have significantly increased. This indicates that LLE has preserved and possibly even improved the underlying structure of the data, making it more suitable for clustering.

LLE may have separated data points that were previously overlapping or mixed, resulting in more distinct clusters.

LLE performs better than all other dimension reduction algorithms for both datasets, except for Kmeans in Dataset

TABLE VII
CLUSTER AFTER RP DATASET 2

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| RP components | Silhouette | Clusters | Silhouette | Clusters |
| 5 | 0.83050 | 15 | 0.8874458 | 19 |

TABLE VIII
CLUSTER AFTER LLE DATASET 1

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| LLE components | Silhouette | Clusters | Silhouette | Clusters |
| 10 | 0.124001 | 4 | 0.69735 | 4 |

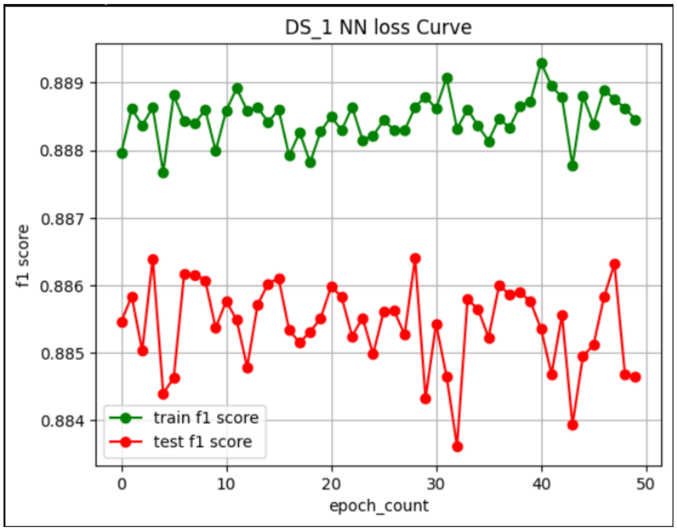

Fig. 9. NN PCA

1. Its only drawback is that it requires a significant amount of processing time.

TABLE IX
CLUSTER AFTER LLE DATASET2

| | Kmeans | | GaussianMixture | |
|---|---|---|---|---|
| LLE components | Silhouette | Clusters | Silhouette | Clusters |
| 3 | 0.960637 | 2 | 0.957012 | 2 |

## VII. PART 4 NEURAL NETWORK AFTER DIMENSIONALITY REDUCTION

### A. *Principal Component Analysis (PCA)*

Figure 9 illustrates the loss curve of the Neural Network on Dataset1 after applying PCA. In Table X, we find a comparison of results between Assignment 1 and Neural Network tuning after applying PCA. It's evident that the test score decreases at a point, but the time is significantly reduced.

PCA effectively reduces the dimensionality of the data by selecting the most important features while discarding less informative ones. This reduction in dimensionality leads to a smaller neural network model, which can be trained faster due to fewer parameters to optimize. Although PCA simplifies the data and reduces the model's complexity, it may result in a small decrease in test performance. This decrease can occur because some information is lost during dimensionality reduction.

TABLE X
NN AFTER PCA

| | Assignment 1 | PCA |
|---|---|---|
| Time | 39.2756 | 22.91351 |
| Test Score | 0.9009 | 0.8988 |
| Hidden Layers | 7,2 | 8,100 |

TABLE XI
NN AFTER LLE

| | Assignment 1 | LLE |
|---|---|---|
| Time | 39.2756 | 3.1706907 |
| Test Score | 0.9009 | 0.85622 |
| Hidden Layers | 7,2 | 5,70 |

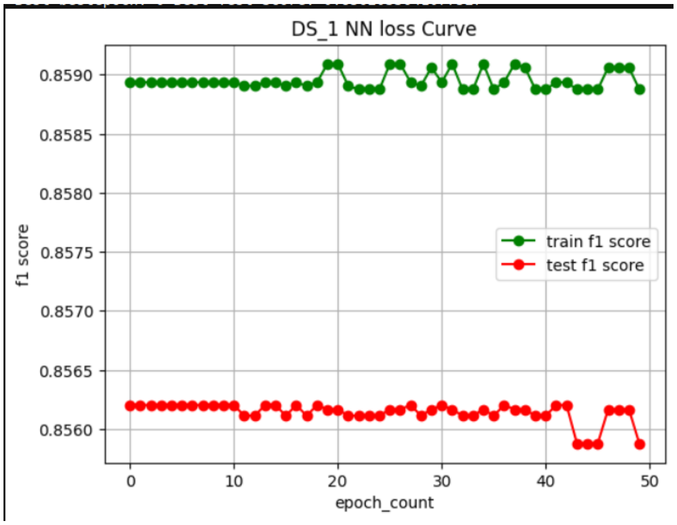

Fig. 10. NN LLE

### B. *Locally Linear Embedding (LLE)*

Figure 10 illustrates the loss curve of the Neural Network on Dataset1 after applying PCA. In Table XI, we find a comparison of results between Assignment 1 and Neural Network tuning after applying LLE. It's evident that the test score decreases at a point, but the time is significantly reduced.

LLE, like other dimensionality reduction techniques, reduces the dimensionality of the data by emphasizing local relationships among data points. This reduction in dimensionality results in a simplified feature space, leading to a smaller neural network model with fewer parameters. The reduced model complexity allows for faster training, significantly reducing the wall clock time.

The small decrease in the test score after applying LLE is typically attributed to the fact that some information is lost during dimensionality reduction. While LLE effectively captures local structures in the data, it may not fully preserve global patterns.

## VIII. PART 5 NEURAL NETWORK AFTER CLUSTERS

Figures 11 and 12, along with Table XII, showcase the Neural Network results after applying Kmeans and GMM clustering algorithms. Table XII also provides a comparison with Assignment 1 or the Neural Network tuning done before applying any clustering algorithm. When comparing the results, we observe that both clustering methods reduce wall clock time but do not significantly improve the test score.

K-Means clustering reduces the dimensionality of the data by assigning each data point to a cluster centroid, while

TABLE XII
NN AFTER CLUSTERS

| | Assignment 1 | Kmeans | GaussianMixture |
|---|---|---|---|
| Time | 39.2756 | 22.91351 | 25.23497 |
| Test Score | 0.9009 | 0.8988 | 0.90270 |
| Hidden Layers | 7,2 | 8,100 | 14,2 |

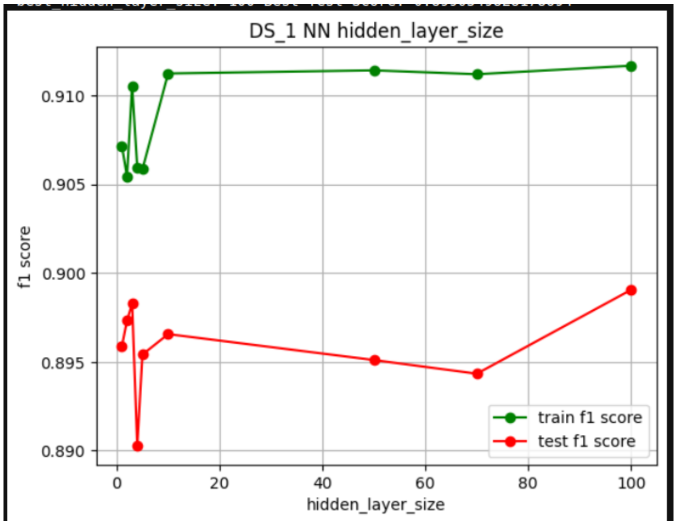

Fig. 11. NN Kmeans

GMM clustering reduces dimensionality by modeling the data distribution using Gaussian components. This reduction in dimensionality leads to a more compact and efficient data representation. As a result, the neural network model becomes simpler with fewer parameters to train, resulting in a significant reduction in wall clock time.

The fact that the test score remains largely unchanged after applying clustering indicates that the clustering process has preserved the essential information in the data. Cluster centroids and Gaussian components effectively represent the underlying patterns and structure, allowing the neural network to learn from this structure without losing predictive power. Clustering abstracts the data into cluster assignments, making the data more suitable for neural network training and potentially improving the neural network's ability to generalize from the clustered representations.

## Conclustion

In the course of this assignment, I embarked on an exploration of unsupervised learning algorithms using two distinct datasets. My analysis encompassed both clustering techniques and dimensionality reduction methods, with a subsequent application of a neural network algorithm to one of the datasets The comparison between clustering and dimension reduction methods revealed that, overall, clustering algorithms delivered superior performance. K-Means and Gaussian Mixture Models (GMM) emerged as the dominant contenders. A notable exception occurred when these algorithms were employed in conjunction with Locally Linear Embedding (LLE), where GMM demonstrated a pronounced advantage over K-Means.

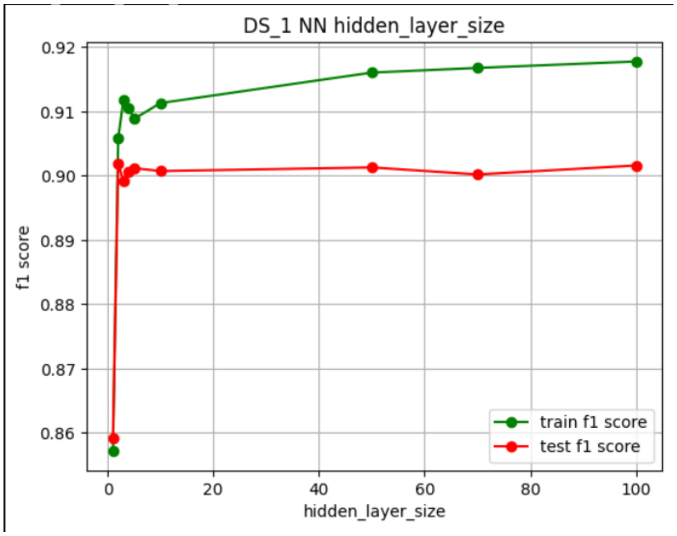

Fig. 12. NN GMM

Among the linear dimension reduction techniques, Principal Component Analysis (PCA) emerged as the most effective. However, the introduction of the non-linear algorithm, Locally Linear Embedding (LLE), presented a compelling case. LLE not only enhanced the Silhouette scores for GMM in Dataset 1 but also significantly reduced the processing time of the neural network algorithm, underscoring its dual benefit in improving both computational efficiency and clustering quality. Moreover, in Dataset 2, LLE emerged as the optimal choice, improving the Silhouette scores for both K-Means and GMM. In summary, this exploration reaffirms the pivotal role of clustering algorithms in unsupervised learning tasks. K-Means and GMM stood out as robust choices, particularly in conjunction with appropriate dimensionality reduction techniques. PCA demonstrated reliability among linear methods, while the introduction of LLE brought forth significant advantages, including enhanced clustering quality and notable reductions in computational processing time. These findings underscore the importance of careful algorithm selection, highlighting the impact on both accuracy and efficiency in unsupervised learning scenarios.

## References

Other then course lecture and book

- https://www.kaggle.com/
- https://towardsdatascience.com/
- https://medium.com/
- https://datascience.stackexchange.com/
- https://chat.openai.com/