Unsupervised Learning Exploration

Yongxing NIE
College of Engineering
Northeastern University
Toronto, ON
nie.yo@northeastern.edu

Abstract

The objective of this project is to explore how will unsupervised learners perform in clustering labelled dataset. I studied how KMeans and GMM cluster datasets before and after datasets going through dimension reduction. I explored how dimension reduction will affect the performance of both supervised and unsupervised learners.

1. Introduction

In this project, I will reuse the datasets I have analyzed in assignment 2. One dataset is the benign or malignant Breast Cancer, the other one is the iris flower. The reason I choose Breast Cancer dataset is that delays in access to cancer treatment will result in 80–90% of the cases, while early diagnosis of Breast Cancer can improve the prognosis and chance of survival significantly. As for the Iris Flower dataset, it is a classic dataset that almost every beginner more or less has to analyze it. And as a botanical enthusiast, I enjoy exploring whatever information revealed during the learning process.

In assignment 2, I applied 5 supervised learning algorithms on both datasets. Except for the neural network learner, the other 4 algorithms can implement training with accuracy score above 90%. In addition to that, support vector machine learner can produce stable and consistent performance on both datasets.

In this project, I will ignore the datasets target and explore how unsupervised learners perform in clustering the dataset. I will compare the unsupervised learning clusters with the real targets to see the logic behind the unsupervised learners. Furthermore, I will apply dimension reduction algorithms to explore how dimension reduction will affect the supervised and unsupervised learners. I use different libraries like Numpy, Pandas, Matplotlib in this project.

2. Experiments

(1) Datasets Inspection

In this project, Breast Cancer dataset can be downloaded at https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data. The dataset contains 569 cases and 30 attributes in total. The 30 attributes are actually three categories, each category contains the same 10 features but were computed in their mean, standard error and worst. Therefore, at this time, I use only the mean value of each case in the model. Before the exploring the dataset, 10 standard error columns and 10 worst columns were dropped.
Iris Flower dataset can be downloaded at https://www.kaggle.com/datasets/arshid/iris-flower-dataset/code. The dataset contains a set of 150 records under 5 attributes - Petal Length, Petal Width, Sepal Length, Sepal width and Class (Species). There are 3 species to be categorized (setosa, versicolor, and virginica), each has 50 entries.

Table 1: The Breast Cancer dataset with 569 entries.

```
df.describe()
```

	radius	texture	perimeter	area	smoothness	compactness	concavity	concave_points	symmetry	fractal_dimension
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440

Table 2: The iris flower dataset with 150 entries.

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

(2) Data cleaning

There were no null values in both datasets. For the outliers, since KMeans is sensitive to outliers, and there are outliers in Breast Cancer dataset, I dropped outliers before model training.

(3) Data characteristics

Both of the two datasets are labeled. I explored the distribution of each feature with respect to the labeled target on both datasets.

For the Iris Flower dataset, one species (setosa) is always clearly separatable from other species. While the other two species do not find clear boundaries. For the Breast Cancer dataset, it seems none of the features can be identified separatable with respect to the malignant or benign target.

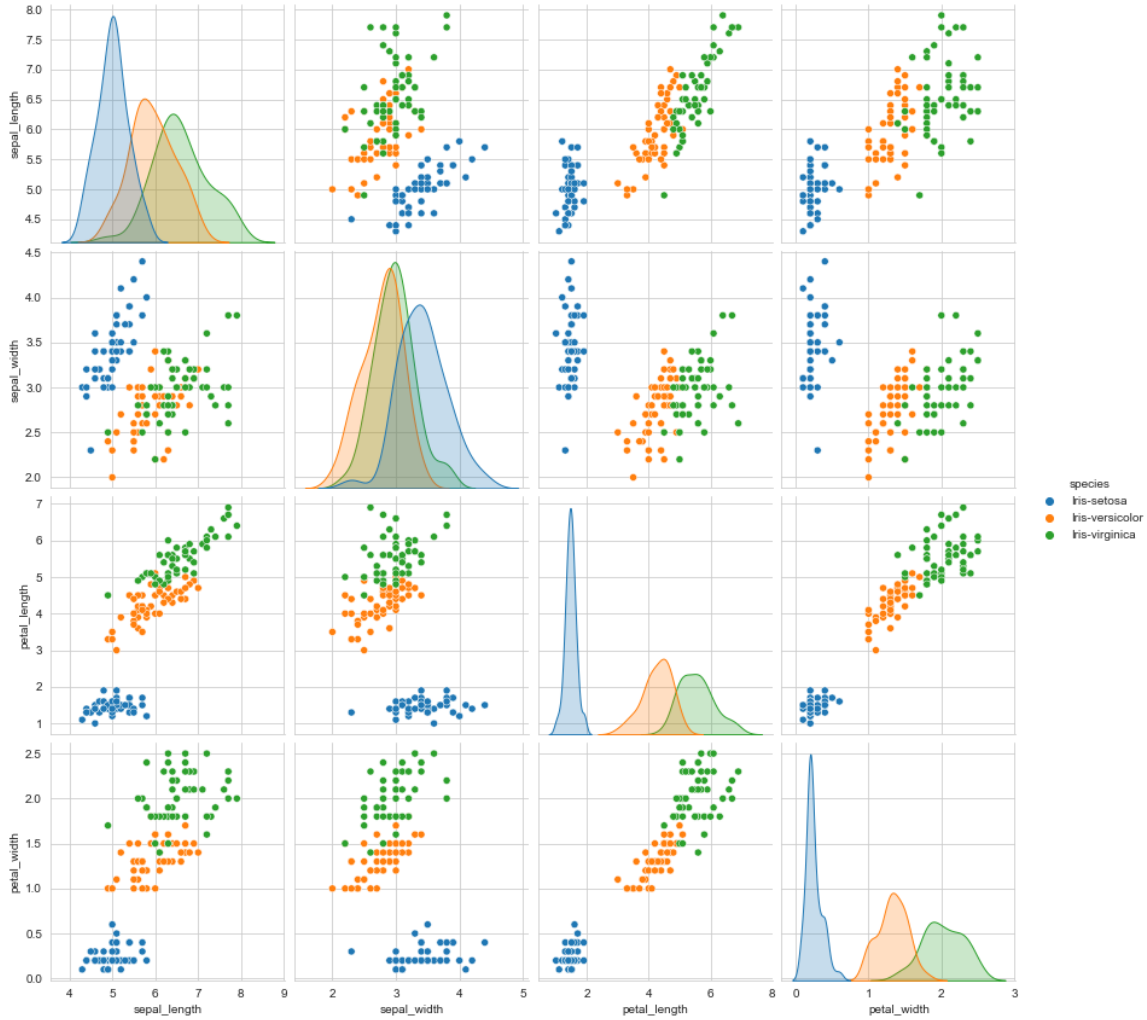


Figure1: The scattering distribution of each iris feature to the labeled target.

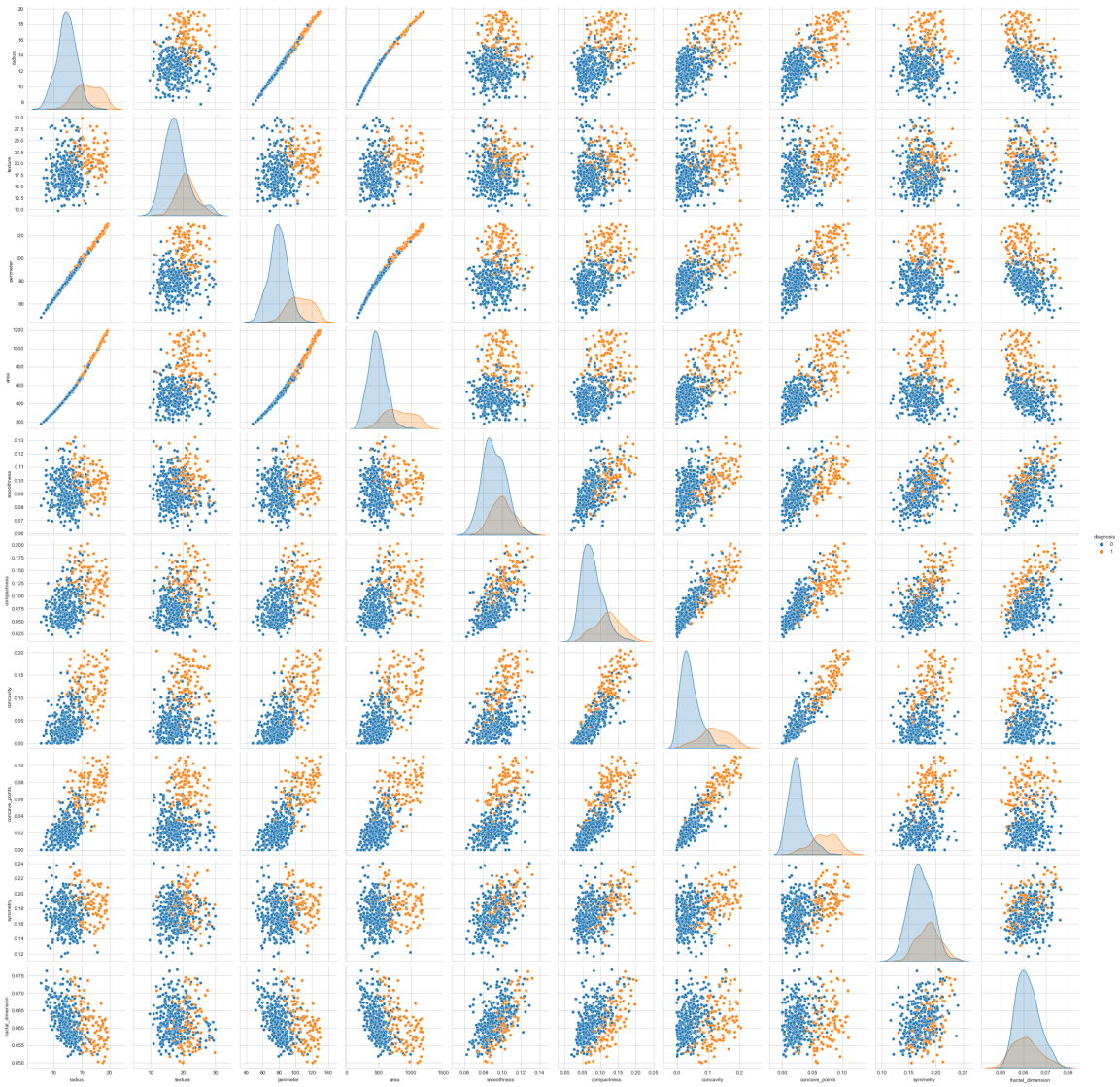


Figure2: The scattering distribution of each Breast Cancer feature to the labeled target.

(4) Methods

In this project, I have:

(1) Applied KMeans and Gaussian Mixture Model (GMM) to both datasets.

(2) Applied dimension reduction algorithms (PCA, ICA, and TSNE(t-Distributed Stochastic Neighbor Embedding)) to both datasets.

(3) Applied KMeans and GMM to the dimension decreased datasets.

(4) Applied supervised learning algorithms to the dimension decreased datasets and compared the wall time of learning and the accuracy with respect to with/without dimension reduction.

3. Results

(1) The optimum cluster numbers for KMeans in Iris dataset is 2.

Although I already know the labelled target, I assume that the label is unknown. The elbow method graph shows that when n_clusters=2, the overall WCSS (Within-Cluster Sum of Square) changed rapidly and thus creating an elbow shape.

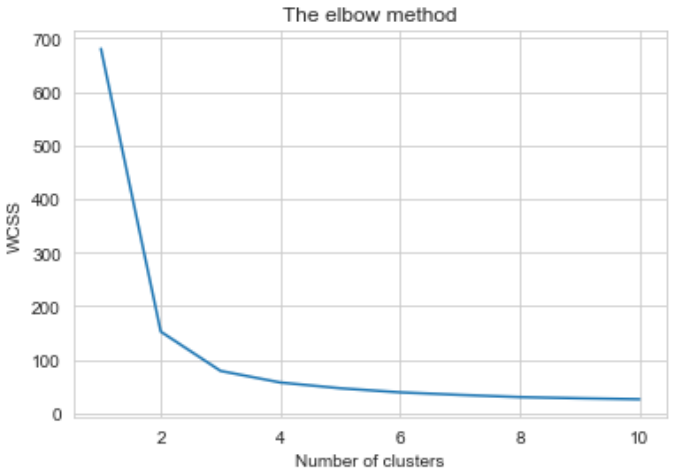


Figure 3: The elbow method graph, WCSS with respect to number of clusters.

(2) In the Iris dataset, KMeans naturally clusters the dataset into 2 groups. While if following the real labels and clustering the data into 3 groups, overfitting appears. Although it is closer to the real situation and it has higher accuracy score, it does present a worse cluster configuration as signified by the silhouette score.

I compared the real cases, which has 3 labelled species thus 3 clsuters, with the KMeans 2 clusters. The result shows that n_cluster=2 is a better clustering configuration but has lower prediction accuracy. While n_clusters=3 is closer to the real situation thus has high prediction accuracy, but the configuration score is lower.

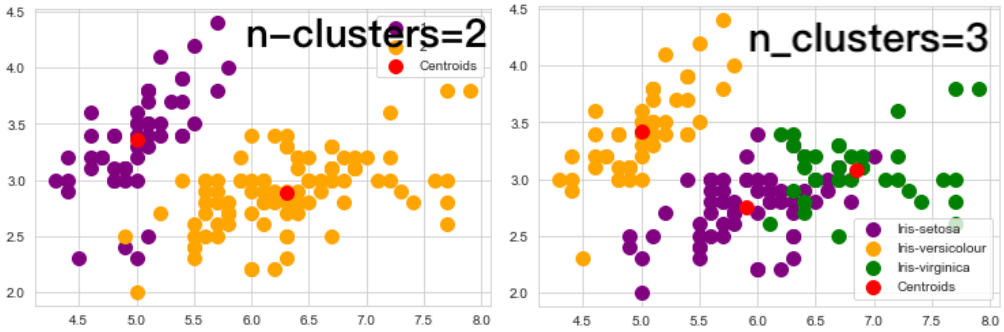


Figure 4: the KMeans scattering with respect n_clusters=2 (left), n_clsuters=3 (right).

Table 2: Comparison between n_clusters =2 and n_clusters =3.

	n_clusters=2	n_clusters=3
silhouette score	0.681	0.553
kmeans_score	0.540	0.730

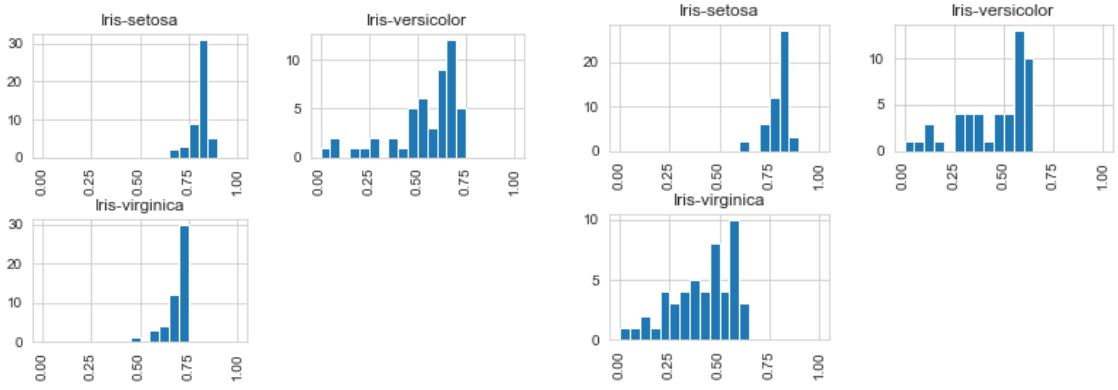
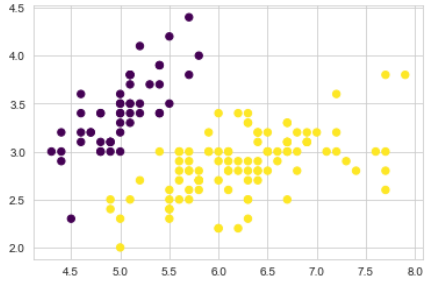


Figure 5: the distribution of silhouette score for each species with respect KMeans n_clusters=2 (left), n_clsuters=3 (right).

(3) GMM tells the same trend in the Iris dataset, 2 clusters are the better clustering configuration while 3 clusters bring up overfitting.

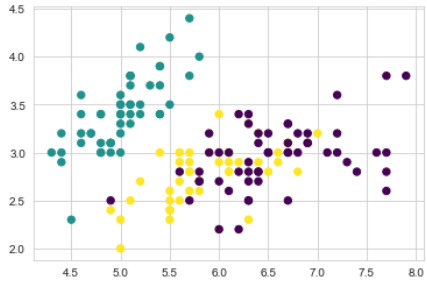
The probability analysis of any point belonging to a given cluster shows that when n_clusters=2, a sample has either 100 or 0 probability to a given cluster, the two clusters has clear boundary. But the prediction accuracy is only 0.568.

While when n_clusters=3, the prediction accuracy can be 0.904. But there are samples shared by two clusters.



```
# The probability of any point belongs to the given cluster.
probs = gmm.predict_proba(x)
print(probs[:5].round(3))
```

```
[[0. 1.]
 [0. 1.]
 [0. 1.]
 [0. 1.]
 [0. 1.]]
```



```
# The probability of any point belongs to the given cluster.
probs = gmm.predict_proba(x)
print(probs[:5].round(3))
```

```
[[1.    0.    0.   ]
 [1.    0.    0.   ]
 [0.999 0.    0.001]
 [1.    0.    0.   ]
 [1.    0.    0.   ]]
```

Figure 6: the GMM scattering and the probability display with respect n_clusters=2 (upper), n_clsuters=3 (lower).

(4) The above result in Iris dataset shows that if there are visible and separatable clusters in the original dataset, KMeans and GMM can produce reasonable and consistent clusters.

(5) The result after PCA dimension reduction in Iris dataset shows the effect of PCA in the new space depends on the clustering configuration. PCA helps improve performance in natural KMeans/GMM clusters. If the KMeans/GMM clustering are overfitted, PCA decrease the performance.

The PCA's explained variance ratio result shows that 2 principal components account for 95.8% of the variance. Because KMeans and GMM are all distance to clusters centers models, we can say we have projected 95.8% of original dataset information into the new space. In the other others, after PCA dimension reduction, we will get the almost exact same clusters.

After feeding the 2 principal components data to KMeans and GMM, the results show that when n_clusters=2, PCA does improve the prediction accuracy. And another discovery is that we are having the exact same clusters as the accuracy of GMM stays the same after PCA.

While when n_clusters=3, since the clustering configuration is a bit overfitted, PCA does not help both KMeans and GMM performance.

Table 3: Comparison of KMeans and GMM score before and after PCA with respect to n_clusters.

		n_clusters=2	n_clusters=3
Before PCA	kmeans_score	0.540	0.730
	GMM score	0.568	0.904
After PCA	kmeans_score	0.568	0.620
	GMM score	0.568	0.507

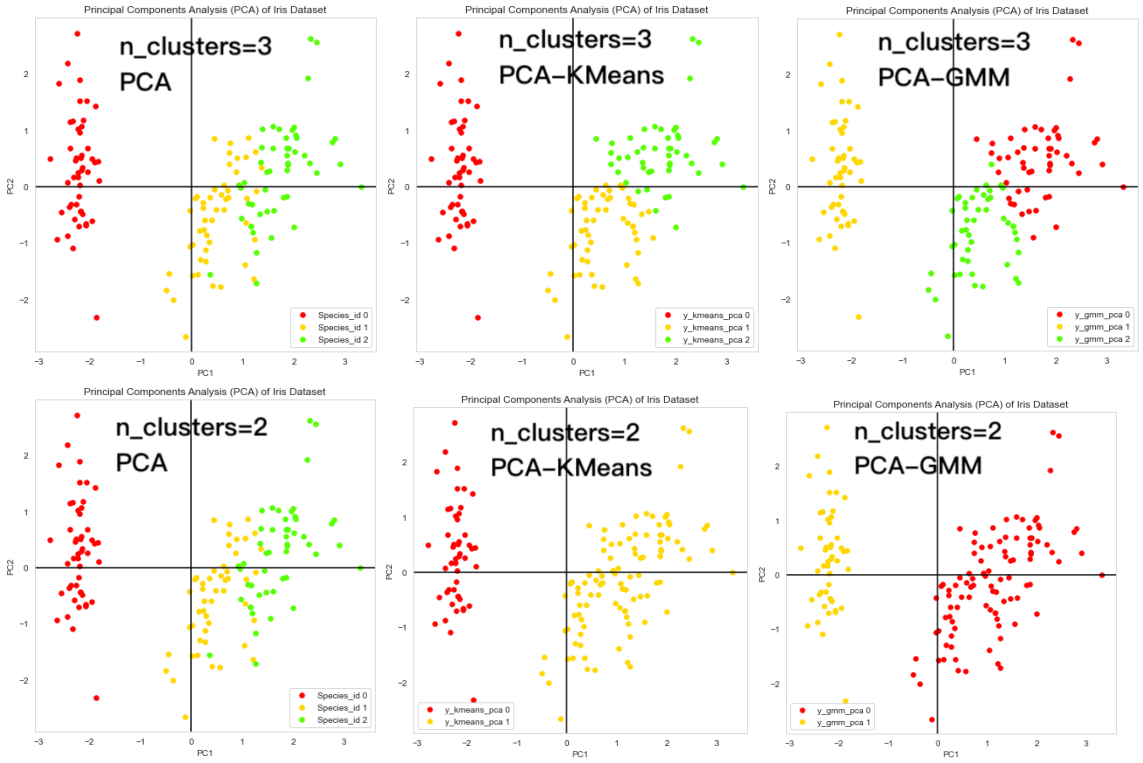


Figure 7: PCA components scattering with respect n_clusters=3 (upper), n_clsuters=2 (lower).

(6) The same conclusion holds in the breast cancer dataset. PCA helps improve the accuracy. But we should control the amount of information projection in PCA to avoid overfitting.

The elbow method result shows the elbow locates where n_clusters=2, which happens to be the same number of the labelled target.

The accuracy score listed in table 3 shows that, after applying PCA, the accuracy scores increase on both KMeans and GMM. PCA do help improve the clustering accuracy.

An interesting point is that, after applying PCA dimension reduction as illustrated in Figure 5, when the number of the principal components is 5, the variance ratio is almost 96.74%, but the prediction accuracy is lower. When the principal components number is 2 or 3, in which the variance ratio is around 80%, both the KMeans and GMM produce highest accuracies. It tells that when projecting data into a new space with more principal components, overfitting appears. Therefore, the more information projected, higher possibility of overfitting. We should control the information projection at around 80% to avoid overfitting.

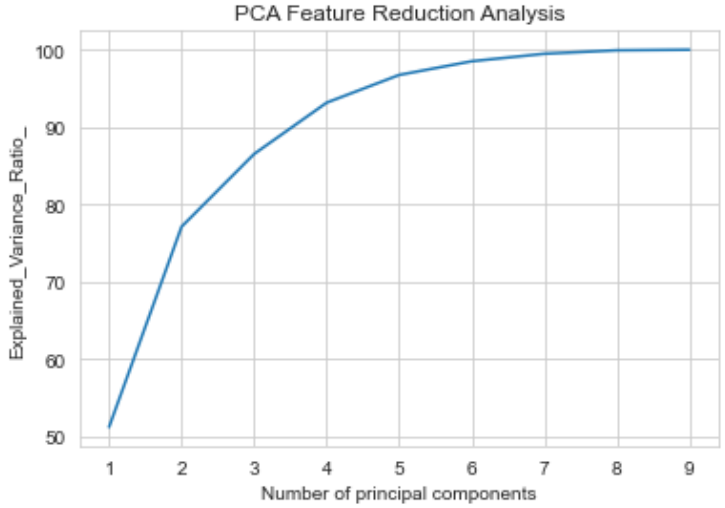


Figure 8: PCA explained variance ratio with respect to number of principal components.

Table 4: Comparison of KMeans and GMM score before and after PCA with respect to PCA components.

		n_clusters=2	PCA components (Explained_Variance_Ratio)
Before PCA	kmeans_score	0.584	N/A
	GMM score	0.538	
After PCA	kmeans_score	0.584	n=5 (96.74%)
	GMM score	0.597	
After PCA	kmeans_score	0.591	n=4 (93.20%)
	GMM score	0.590	
After PCA	kmeans_score	0.591	n=3 (86.53%)
	GMM score	0.617	
After PCA	kmeans_score	0.591	n=2 (77.13%)
	GMM score	0.617	

(7) The number of independent components generated by the ICA is equal to the number of observed mixtures. After ICA, both KMeans and GMM's performance decrease because there are components in both datasets that are not separated by ICA.

According to Ulykbek[1], "The output of ICA depends on a fundamental parameter: the number of components (factors) to compute. The optimal choice of this parameter, related to determining the effective data dimension, remains an open question in the application of blind source separation techniques to transcriptomic data." Therefore, in this project, I will not explore the optimal number of independent components of ICA this time. Since I already know the labels of the datasets, the number of independent components generated by the ICA is equal to the number of targets.

After ICA, the performance of KMeans and GMM on both datasets decrease. The Kurtosis result in the Breast Cancer dataset shows that the kurtosis of each independent component is close to 0, which means the two components are Gaussian and will not be separated by ICA.

The Kurtosis result in the iris dataset shows one component is non-Gaussian while the other two components are Gaussian, therefore, ICA can only separate one component as shown in Figure 6 (more 3D pictures can be checked in code).

Another interesting point is that, for both datasets, when decreasing the numbers of independent components, their KMeans and GMM performance keep increasing until there is only one independent component.

Table 5: Comparison of KMeans and GMM score before and after ICA with respect to datasets.

		Breast cancer dataset	Iris dataset (n_clusters=3)
Before ICA	kmeans_score	0.584	0.730
	GMM score	0.538	0.904
After ICA	kmeans_score	0.564	0.571
	GMM score	0.493	0.611
	kurtosis	[0.016, 0.03]	[-1.421, 0.261, -0.233]

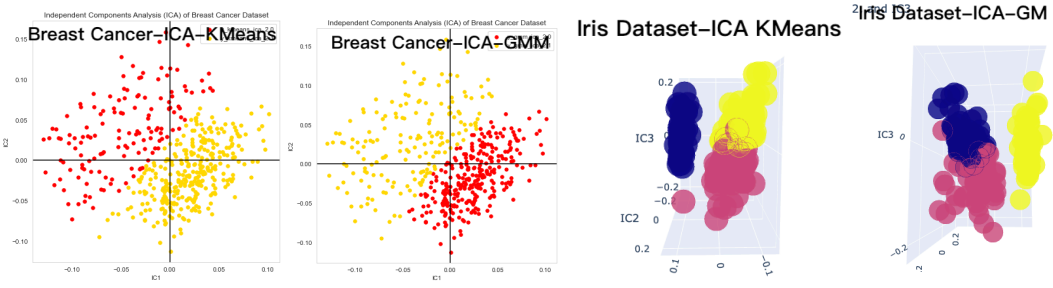


Figure 9: KMeans and GMM results after ICA with respect Breast Cancer dataset (left), Iris dataset (right).

1 Kairov, U., Cantini, L., Greco, A., Molkenov, A., Czerwinska, U., Barillot, E. and Zinovyev, A., 2017. Determining the optimal number of independent components for reproducible transcriptomic data analysis. BMC genomics, 18(1), pp.1-13.

(8) I explored the t-SNE (t-Distributed Stochastic Neighbor Embedding). Because it is usually used to solve the curse of dimension and is more suitable for datasets with hundreds of features, thus the performance does not get improved after t-SNE.

During t-SNE reduction, we should pay attention to the iterations. t-SNE uses gradient descent with Kullback-Leibler divergence as the cost function, the algorithm is stochastic, therefore multiple executions with different random seeds will yield different results. We should run the algorithms round by round to pick the one with lowest KL divergence.

Table 6: Comparison of KMeans and GMM score before and after t-SNE with respect to datasets.

		Breast cancer dataset	Iris dataset (n_clusters=3)
Before t-SNE	kmeans_score	0.584	0.730
	GMM score	0.538	0.904
After t-SNE	kmeans_score	0.422	0.267
	GMM score	0.540	0.272

(9) How will dimension reduction algorithm improve the time efficiency and performance of supervised learning algorithms. The answer is it depends.

Before feeding the PCA dimension reduction data to supervised learning algorithm, I thought it may PCA may help improve the wall time as well as the accuracy since PCA has already picked out the most important information from the original dataset. However, it turns out that it depends on the situation. Sometimes it will help improve the accuracy, sometimes not. At the same time, it does not help improve the wall time. But I still think that PCA may help with the wall time when the dataset face dimension curse.

Table 7: Comparison of wall time before and after PCA with respect to different supervised learning algorithms.

		Breast cancer dataset	Iris dataset (n_clusters=3)
neural network	Before PCA	2.64s (0.9211)	4.56 s (0.80)
	After PCA	2.64s (0.9298)	4.59s (0.80)

4. Conclusions

(1) KMeans and GMM can produce reasonable and consistent clusters for the datasets which has already visible clusters.

(2) It happens that KMeans and GMM cluster dataset not in accordance with the real labels. At this situation, if following the real labels and let KMeans and GMM unnaturally clustering the data, overfitting appears. Although it is closer to the real situation and it has higher accuracy score, it does present a worse cluster configuration in the KMeans and GMM perspective.

(3) When doing PCA and deciding the optimal number of principal components, take 80% as the benchmark of the explained variance ratio. Lower than 80% may project inadequate information into the new space, while higher than 80% may bring up overfitting.

(4) The effect of PCA in the new space depends on the clustering configuration. PCA helps improve performance in natural KMeans/GMM clusters. If applying KMeans/GMM according to the original labels, PCA do further deteriorate the performance.

(5) The number of independent components generated by the ICA should be equal to the number of observed mixtures.

(6) t-SNE (t-Distributed Stochastic Neighbor Embedding) is more suitable for datasets with hundreds of features.

(7) The impact of dimension reduction algorithms on supervised learning varies, it really depends on the dataset, the dimension reduction algorithms and the supervised learning algorithms.

References

[1] Kairov, U., Cantini, L., Greco, A., Molkenov, A., Czerwinska, U., Barillot, E. and Zinovyev, A., 2017. Determining the optimal number of independent components for reproducible transcriptomic data analysis. BMC genomics, 18(1), pp.1-13.

[2] "Dealing with Highly Dimensional Data using Principal Component Analysis (PCA)", https://towardsdatascience.com/dealing-with-highly-dimensional-data-using-principal-component-analysis-pca-fea1ca817fe6

[3] Tharwat, A., 2020. Independent component analysis: An introduction. Applied Computing and Informatics.

[4] “Independent Component Analysis (ICA)”, https://towardsdatascience.com/independent-component-analysis-ica-a3eba0ccec35
[5] “Dimension Reduction”, https://goldinlocks.github.io/Basic-Dimensionality-Reduction/
[6] “What is tSNE and when should I use it?”, https://sonraianalytics.com/what-is-tsne/#:~:text=T%2Ddistributed%20Stochastic%20Neighbourhood%20Embedding,in%202%20or%203%20dimensions.