Assignment 3

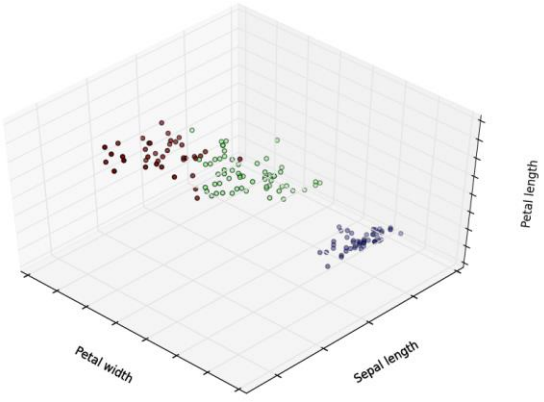
Kuan-E Chao

Unsupervised Learning and Dimensionality Reduction

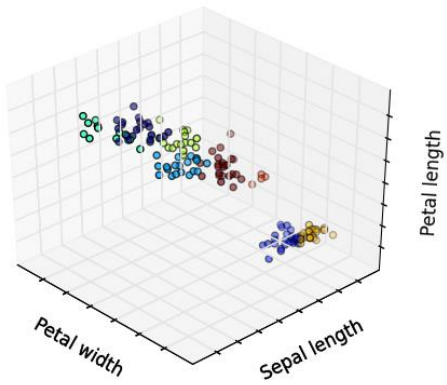
For this assignment, I decided to choose the iris data and one of the data I used for assignment 1 to perform this experiment which is the wine dataset. I use scikit learn for this assignment in python.

The reason I choose the Iris dataset is because this dataset is very easy to observe clusters. This dataset is very easy to differentiate by human therefore machine should also expect some high prediction rate. The reason I chose wine dataset is because this type of dataset is harder to distinguish than iris dataset, I want to see how clustering and dimensionality reduction apply on the wine dataset. Also want to see how dimension reduction works in dataset which contains a lot of noisy columns.

First thing I chose is to run the k-means on the iris dataset. The distance I used is the actual distance in the dataset as the distance in the graph. The reason I chose this as the actual measurement is because the difference for the dataset is not too large, and the data appears normal so that we don't need other types of measurement to describe it.



Iris dataset for 3-means

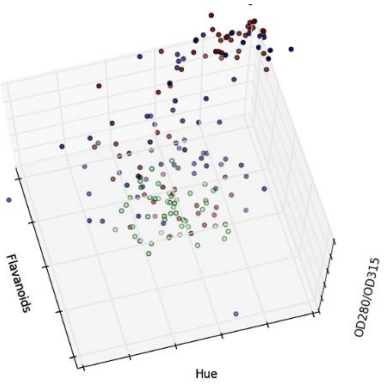


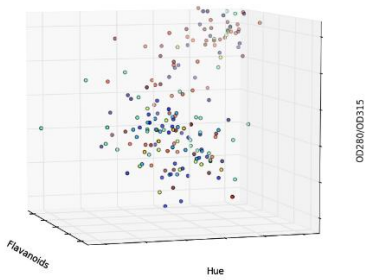
Iris dataset for 8-means

When I run the iris dataset, I noticed that using 3 means is very good because clearly there are 3 clusters. When using 8 means, we can see that it is not a good way to form clusters. Three clusters are distributed very evenly and does not have any special shape. The 8 means clusters looks connected on some of clusters which makes this clusters not as good as I want.

Then I run my dataset from assignment 1, which is the wine dataset. This wine dataset is trying to predict wine's location from the information given in the dataset. I use the actual distance for this data. I decided to choose 3 as k, because without looking the actual datasets, it appears to be 3 clusters. And if we look at the target values, there are 3 types of iris, which are setosa,

versicolor and virginica. Therefore with and without looking at the target value, I would choose 3 as k values.



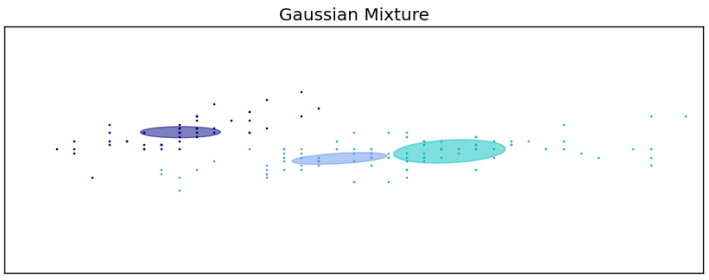


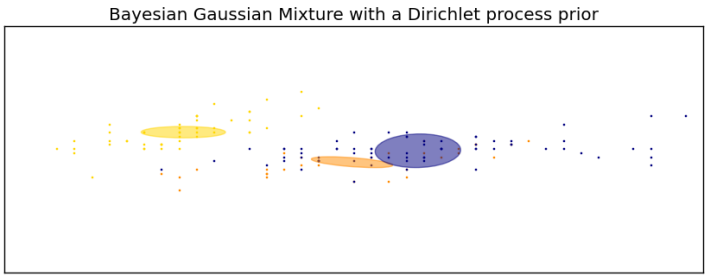
Wine dataset for 3-means

Wine dataset for 8-means

When I run the wine dataset, I realized that this dataset is harder to see the clusters. However, we can quite still see the pattern where 3 clusters are distributed. When I used the 8-means. I cannot see the pattern. Therefore, we can see that using 3 means is better. Therefore, I also use 3 for these datasets. I decided to use 3-means for both datasets.

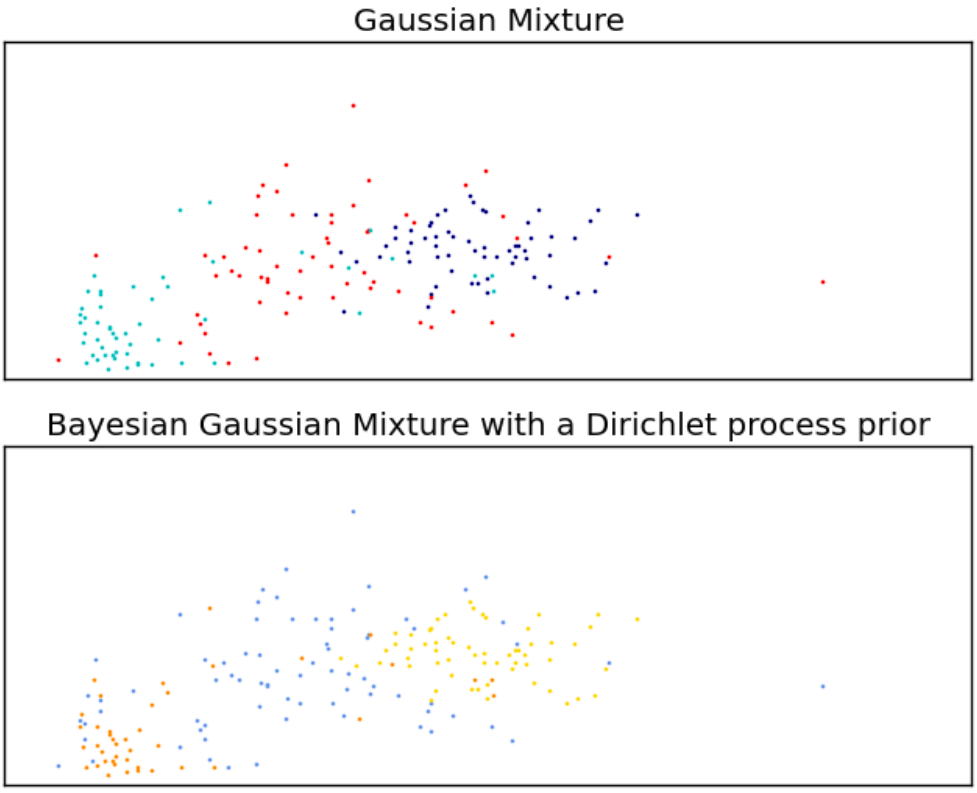
And then I run Expectation Maximization on those two datasets. I used two-dimensional plot on those 2 datasets. First thing I run is the iris data for the Expectation Maximum using Gaussian Mixture in the scikit learn. I used 3 clusters (3 colors):





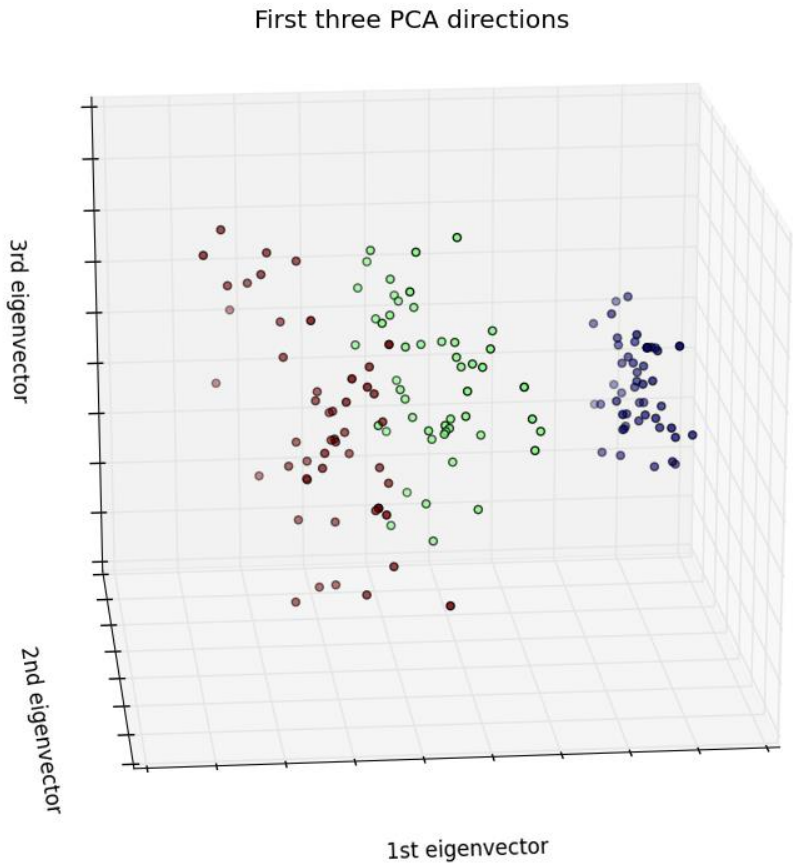
We can see that the expectation maximization works well on this type of data because there are ellipse which can show the maximum likelihood for each particular data point. We can see there are three clusters.

And then I run the expectation maximization on the wine dataset with 3 colors:



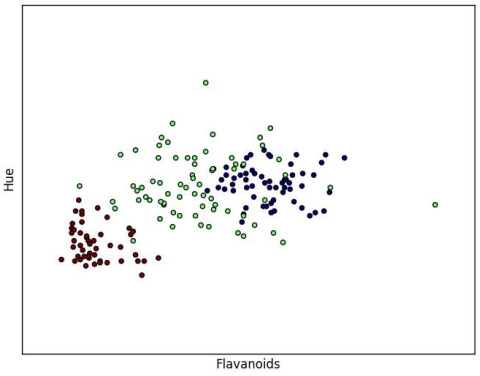
We can see that the Expectation Maximization works well on the wine dataset compare to k-means clustering. Clusters are harder to observe compare to using k-means.

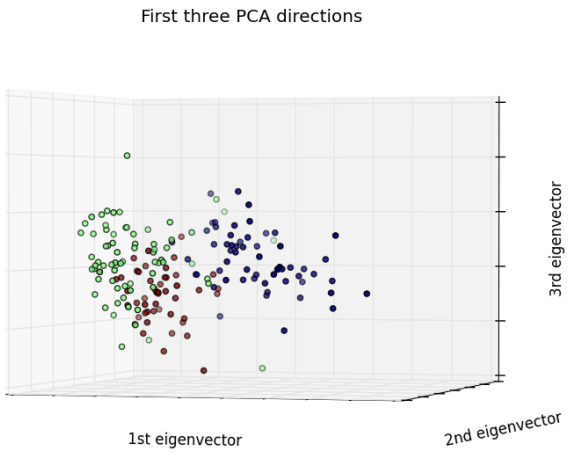
And then I start doing dimension reduction on these datasets. The first one I did is PCA, which is Principal component analysis. Here is my result when I run PCA for the iris dataset:



From this Iris data set we can see that the PCA distinguish the clusters very well, especially the first eigenvector. The first eigenvector can easily distinguish this dataset very well into 3 clusters. We can also see that 2nd and 3rd eigenvector also helps to distinguish the clusters very well.

And then I run the PCA for my wine dataset. First thing I do is to plot the training points as scatterplot. We can see 3 clusters separated clearly:

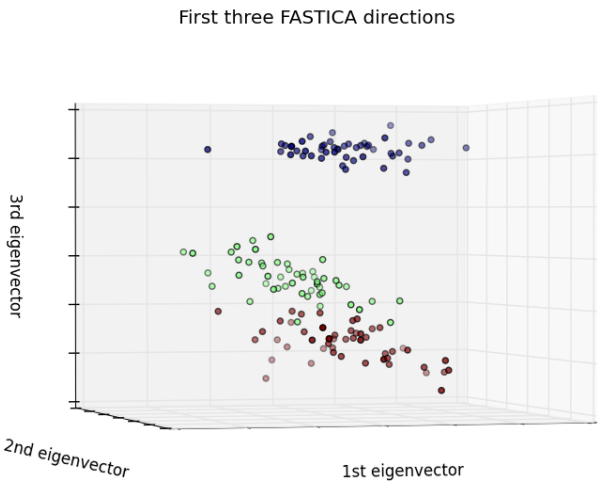




PCA for wine dataset

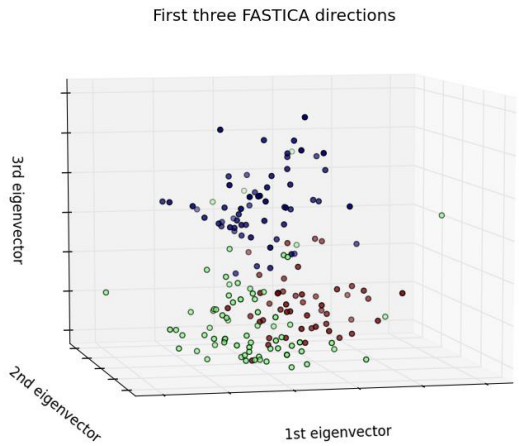
We can see that the first eigenvector separate this dataset very well. Second and third eigenvector also help to separate clusters.

And then I run the ICA (Independent component analysis) for the iris dataset, in scikit learn, it is called FASTICA:



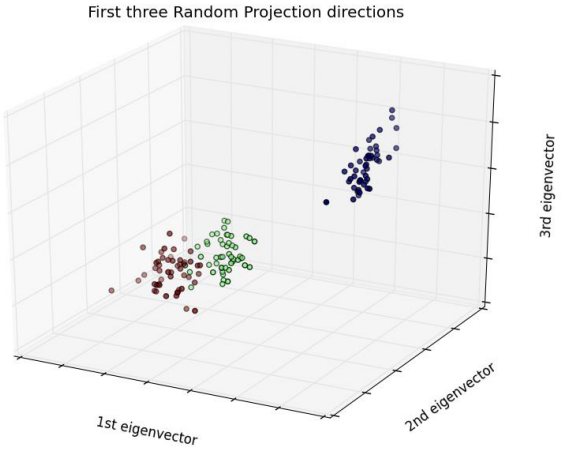
We can see that ICA can separate the data very well by treating each subcomponents as independent dataset to each other.

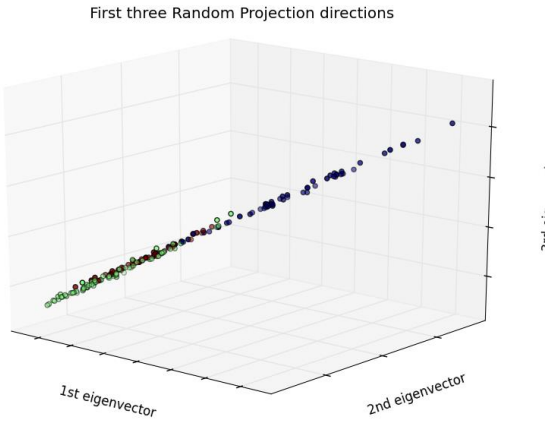
And then I run the ICA for the wine dataset:



We can also see 3 clusters in the scatterplots. This can separate one of the dataset, but not the purple one and the green one. ICA can barely separate the brown and green dots.

Then I run randomized projection on both datasets:

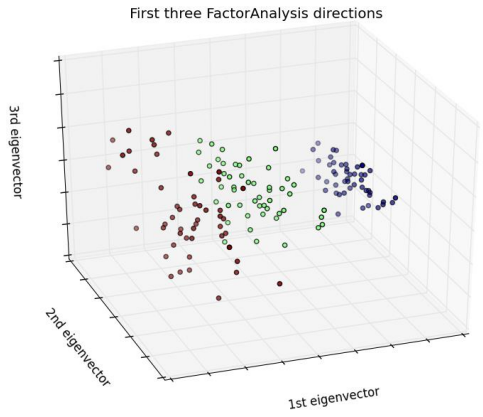


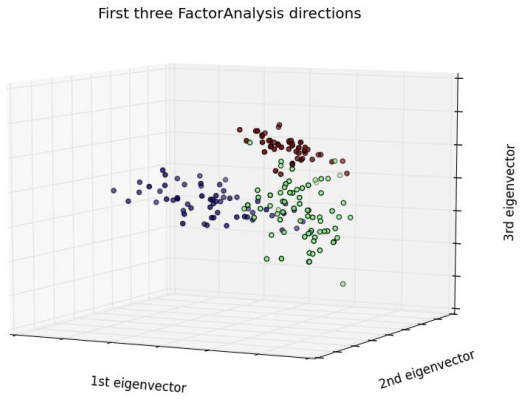


Randomized projection for Iris dataset Randomized projection for wine dataset

We can see that for randomized projection, it works well on the Iris dataset because each distinguish subcomponent can distinguish the dataset well. However, for the wine dataset, we can see that the randomized projection does not work too well because the there are too many noisy subcomponent, therefore, we can only get a straight line when we run randomized projection.

And the for the last projection, I chose factor analysis:





Factor Analysis for Iris dataset Factor Analysis for Wine dataset

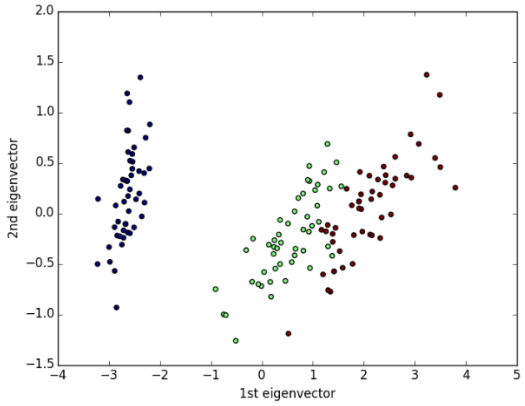
Factor Analysis performs a maximum likelihood estimate of the so-called loading matrix, the transformation of the latent variables to the observed ones, using expectation-maximization.

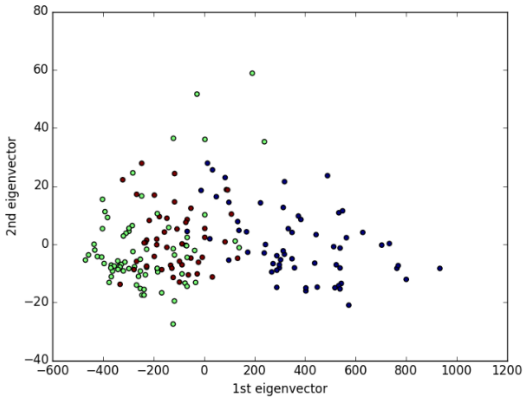
We can see that the factor analysis separates both datasets very well. We can clearly see the pattern from the graph.

Applying dimension reduction

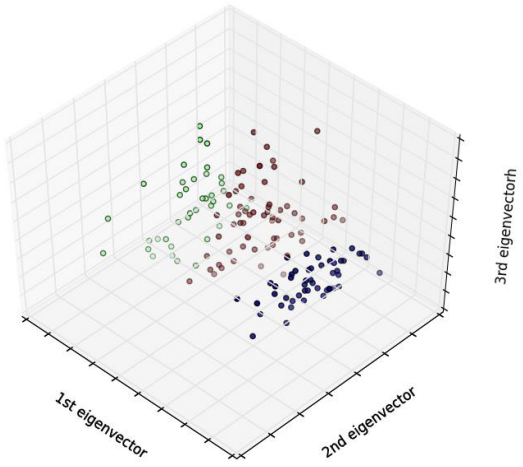
I decided to plot into a 2-dimensional scatterplot to see if the dataset fits well.

First, I run PCA for both datasets.

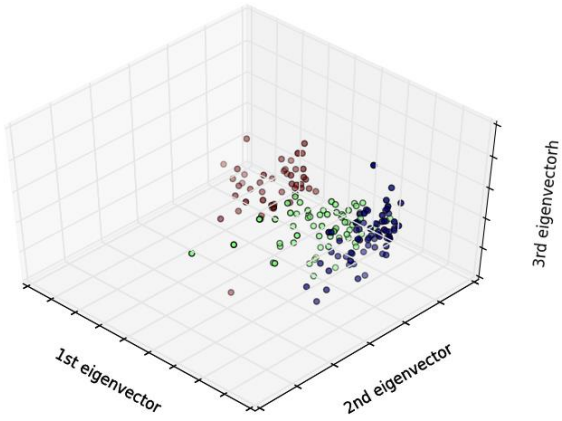




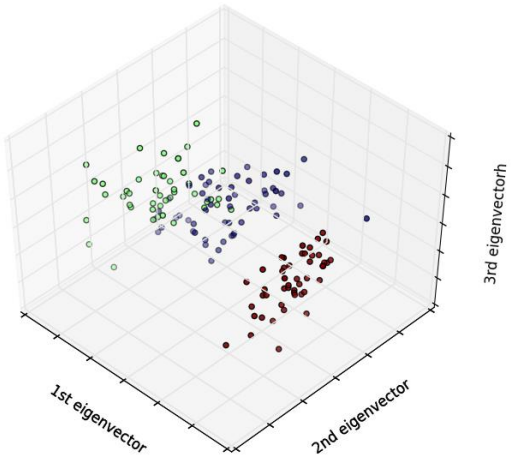
Then I run k-means for Reduced matrix.



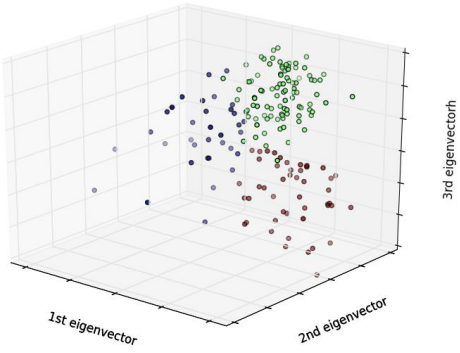
K-mean after PCA reduced for iris dataset



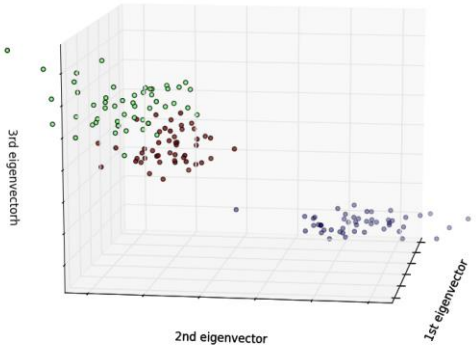
K-mean after PCA reduced for wine dataset



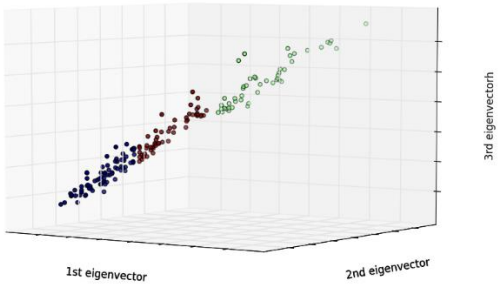
K-mean after ICA reduced for iris dataset



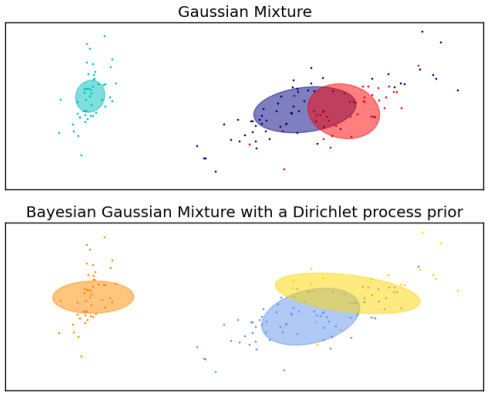
K-mean after ICA reduced for wine dataset

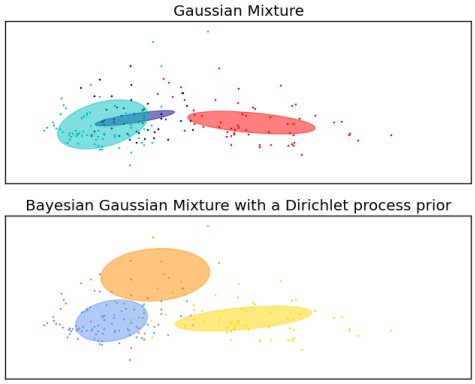


K-mean Randomized Projection for iris dataset



K-mean Randomized projection for wine dataset





EM for Factor analysis for Iris dataset

EM for Factor analysis for Wine dataset

When I run dimension reduction, I can see that clusters are easy to tell with colors. However, for most of the dataset, we can see that it is really hard for human to tell the cluster without coloring for the wine datasets. We can also see that randomized projection dimension reduction is not a good method for wine dataset since it contains too many noisy columns.

ICA can capture meaningful output by choosing the correct columns during the reduction algorithms. Else, it will be hard to observe when it get a lot of noisy data, but still has better effect than randomized projection.

The clusters I got will not be the same as before for some reduction algorithm because they might not choosing the right column that is not noisy.

Neural Network for reduced dataset.

	Training scores:	Testing scores:
Without reduction	0.61538462	0.398876404494
PCA	0.90769231	0.398876404494
ICA	0.61538462	0.33333333
Randomized projection	0.61538462	0.398876404494
FactorAnalysis	0.96923077	0.69444444

Conclusion:

When I do the clusters, I see that the cluster line up with the labels. However, without the cluster color, it is hard to distinguish the cluster without the color. We can see that PCA and Factor

Analysis can improve performance very well. Randomized project cannot improve performance, but it can increase the running time.

For PCA, we can see that the first eigenvector can separate the both data very well. 2nd and 3rd eigenvectors can also help to distinguish data better. For the wine dataset, PCA is probably the only good dataset that works well with it because it can help to choose the correct eigenvector to distinguish the datasets.

For ICA, we can see that ICA works very well on Iris dataset because all the subcomponents are quite distinguishable. However, ICA doesn't perform too well on wine dataset because there are too many noisy data.

For Factor Analysis, it works very well and very accurate on both datasets. However, it is the running time is very high for factor analysis.

For Randomized projection, I rerun it for like 10 times and realized that it will not work well unless it chooses the correct columns. However, this datasets contains too many noisy columns, this makes the dataset ending with a straight line most of the time.

When I rerun my neural network, I can see that except for FactorAnalysis, other dimension reduction performs faster due to number of datasets got reduced especially for randomized projection due to the projection is randomly chosen. However, running neural network after using dimension reduction by FactorAnalysis can improve the training and testing scores of the wine dataset.

Next time if I want to improve the performance, I will remove the noisy datasets manually to obtain better result for the wine datasets. I also believe that if this dataset contains more data, it could increase the accuracy in separating the clusters.

Citation:

Scikit learn

Numpy

Pandas