Machine Learning Homework 5

Name: SunnerLi (李政哲) ID: 0656011

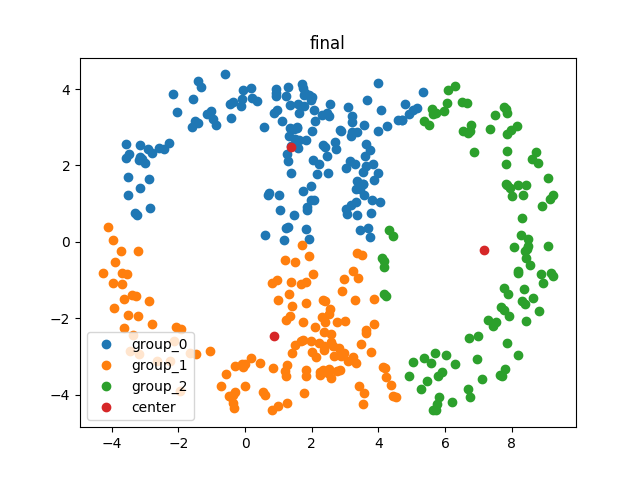
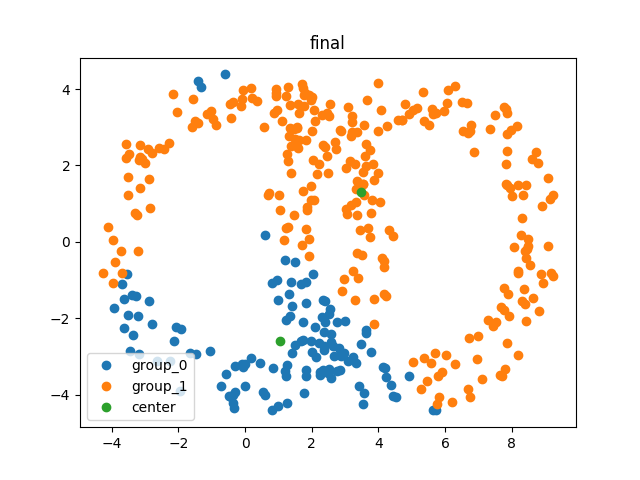
**Question 1: You need to make a video showing the clustering procedure of your kmeans/kernel kmeans program (excluding spectral clustering)  You can refer to the following webpage to see what kind of visualization you want to have.**

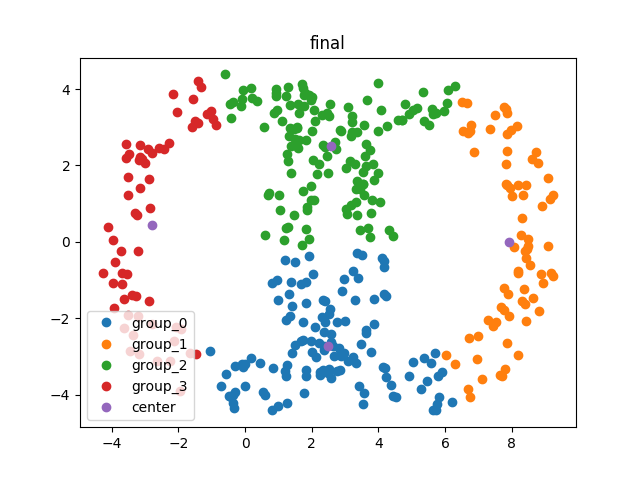
Ans: For my experiments, the recursive procedure will conduct up to 20 times (20 iteration). If the label of each point didn’t change, I will end the program earlier. In the kernel k means and spectral clustering, I adopt RBF kernel as the similarity measure. For the hyper parameter, the gamma I use in kernel k means is 0.01, but the gamma in spectral clustering is 0.0001. This changing can raise the performance of clustering. The formula of RBF kernel is shown in below:

**Question 2: In addition to cluster data into 2 clusters, try more clusters and show results.**

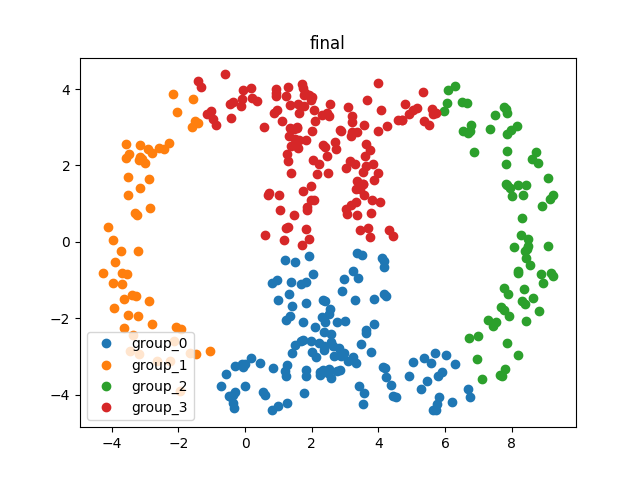
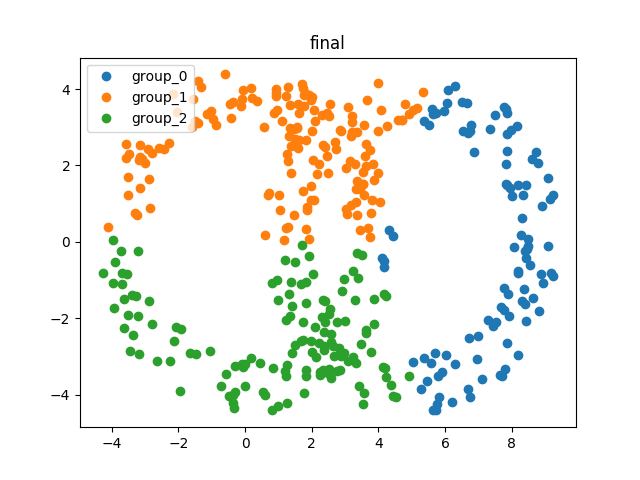
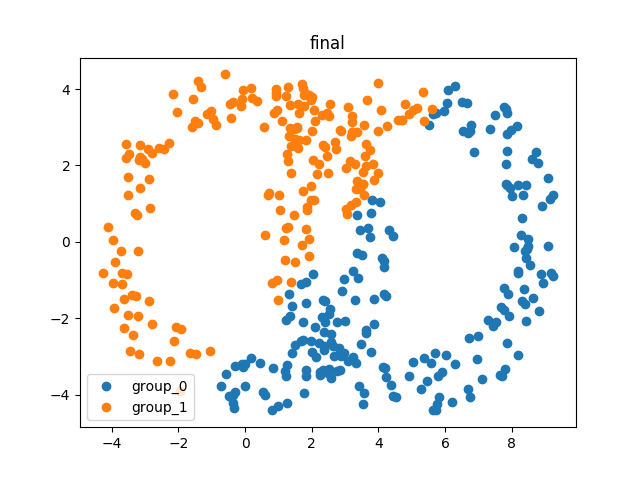
Ans:

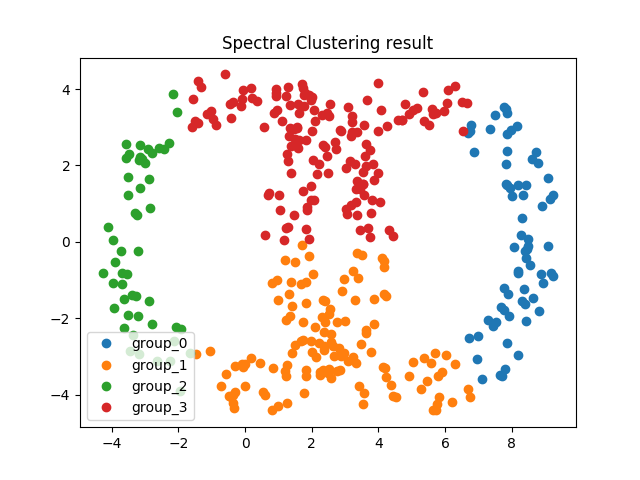
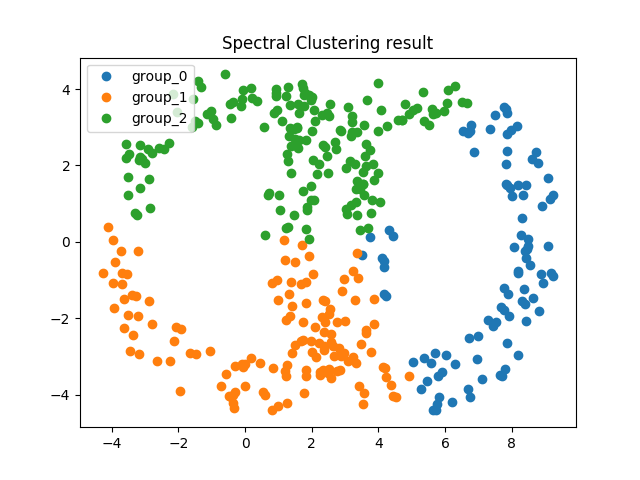
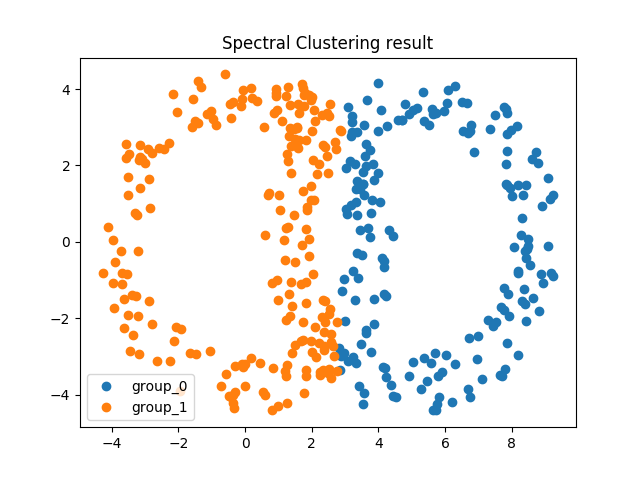
The following three image shows the result that using usual k-means to deal with the test data 1. As you can see, the performance isn’t very well.





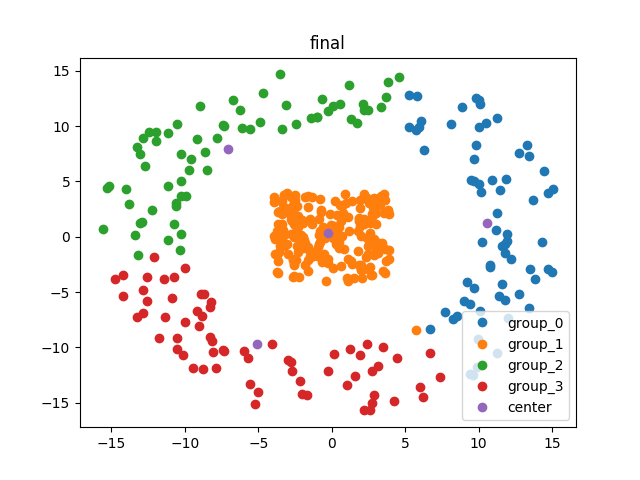
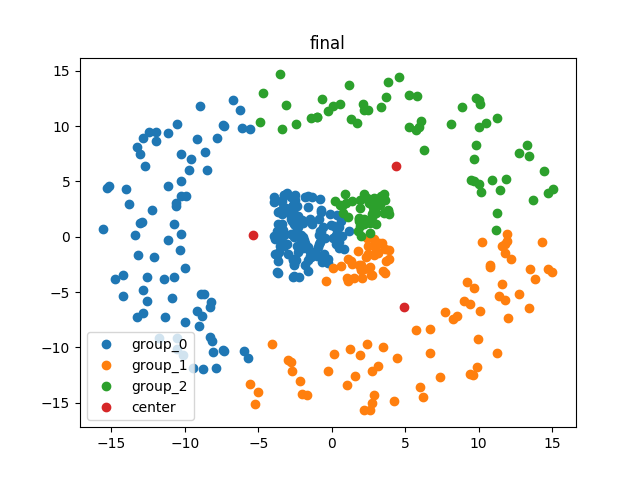
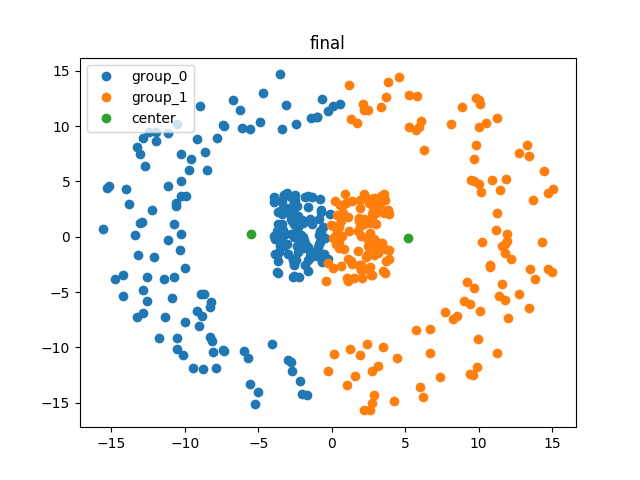
The results are similar if using kernel k-means and spectral clustering.



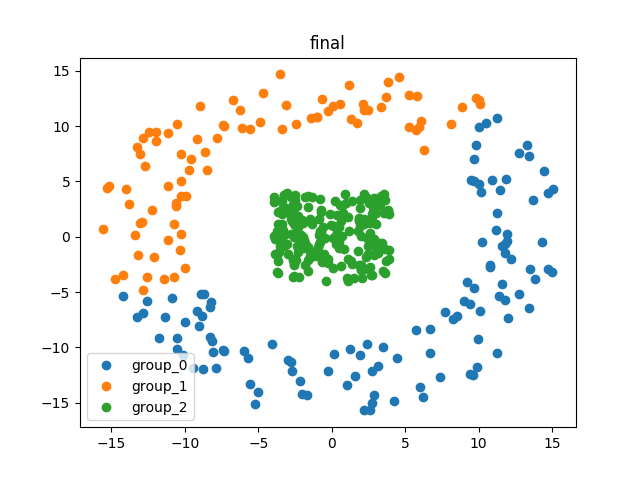
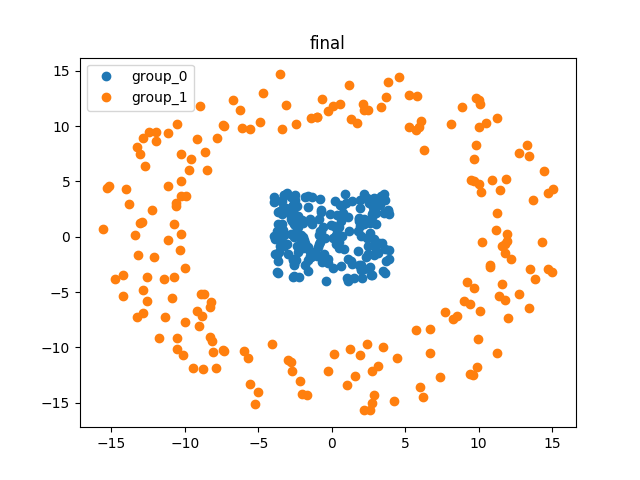


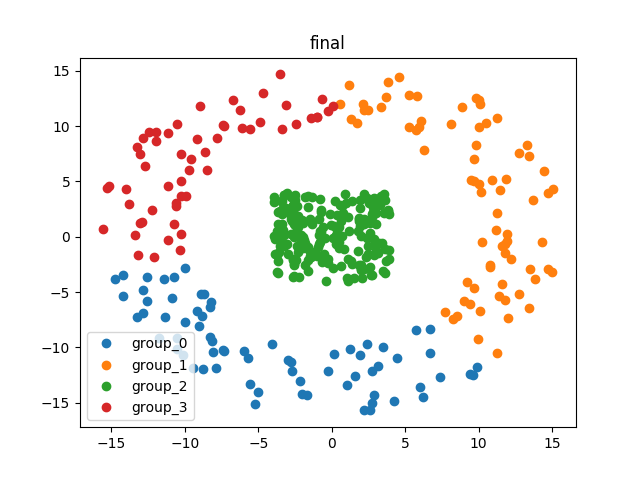
Because the both group data have very serious transversality. Even we use kernel trick to project the data points, it cannot split the group clearly.

Next, we consider the case in test data 2. By using usual k means, we can get these result.

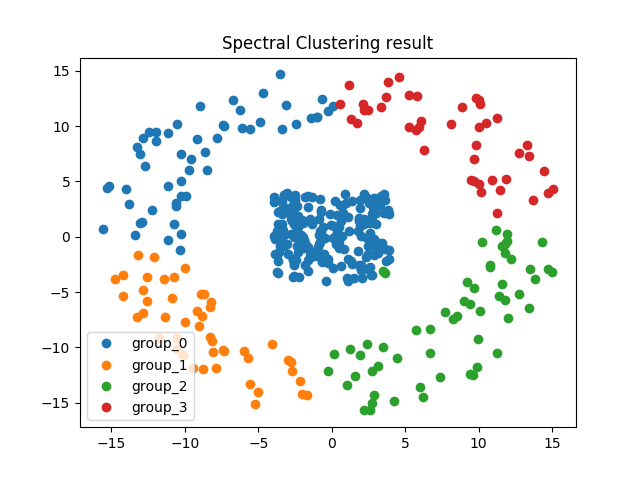
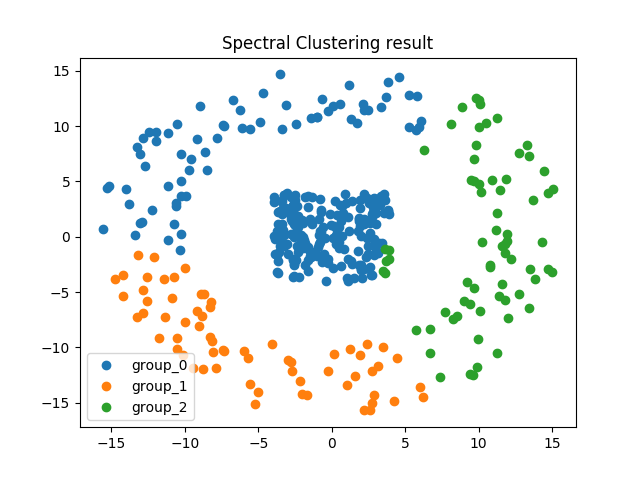
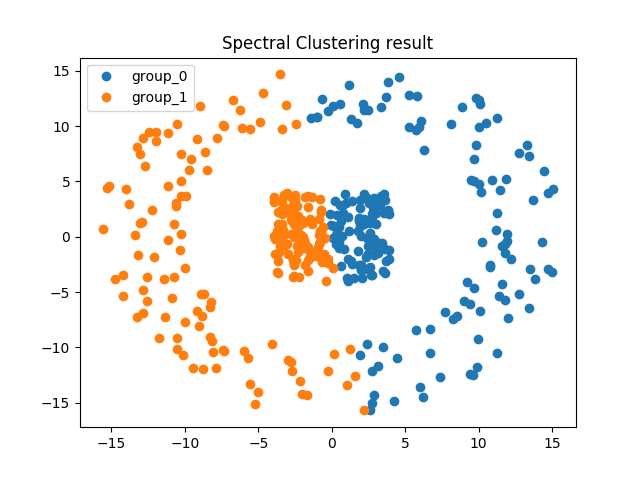


However, if we adopt kernel trick into the k means, the program can get very great performance! As you can see in the following image, the center group can be separated correctly whatever the number of cluster it is.





On the other hand, the spectral clustering also uses kernel trick to project the data to the feature space which is spam by eigen vector with k-smallest eigen value. However, we cannot obtain the great result eventually.

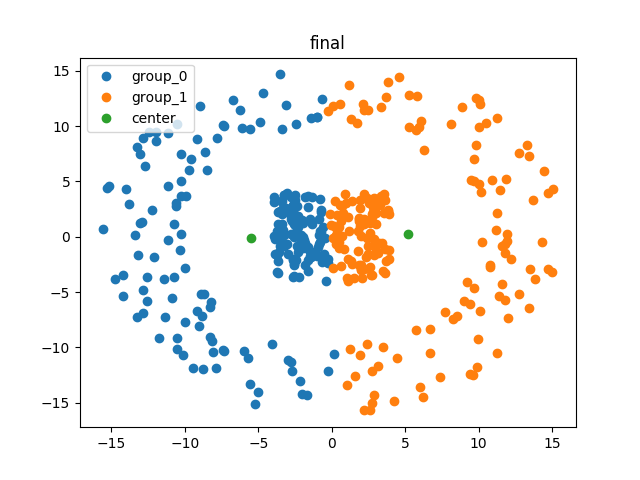
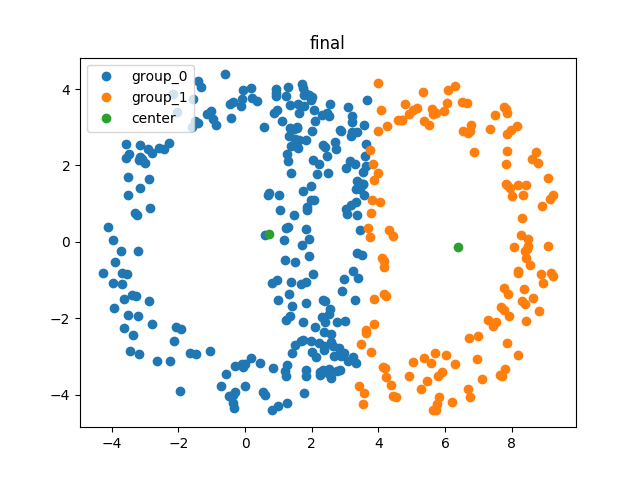


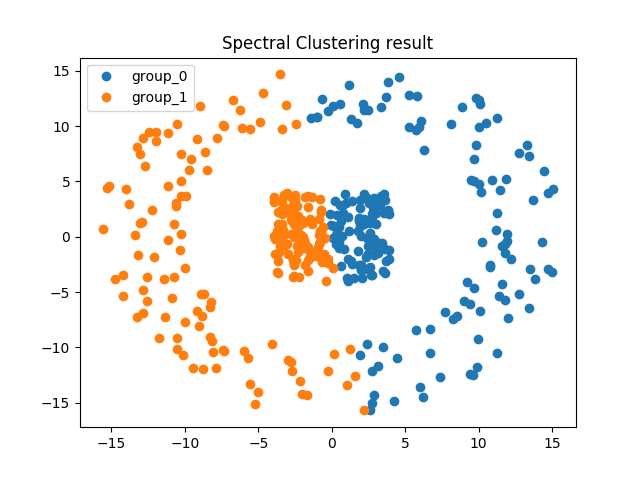
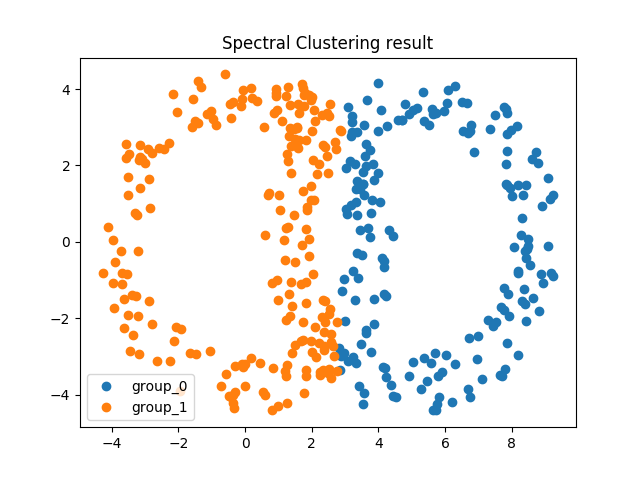
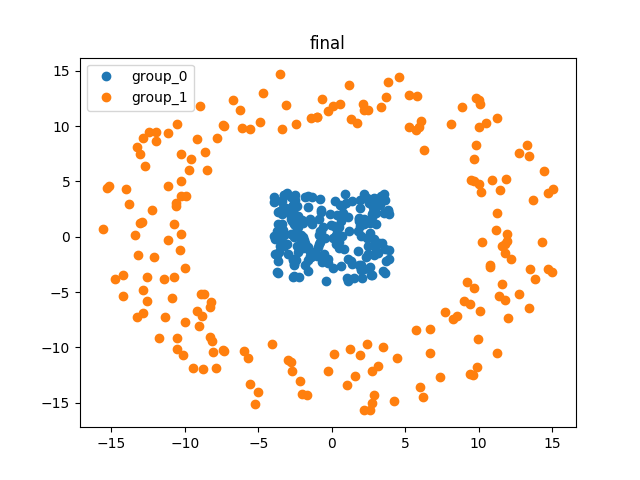
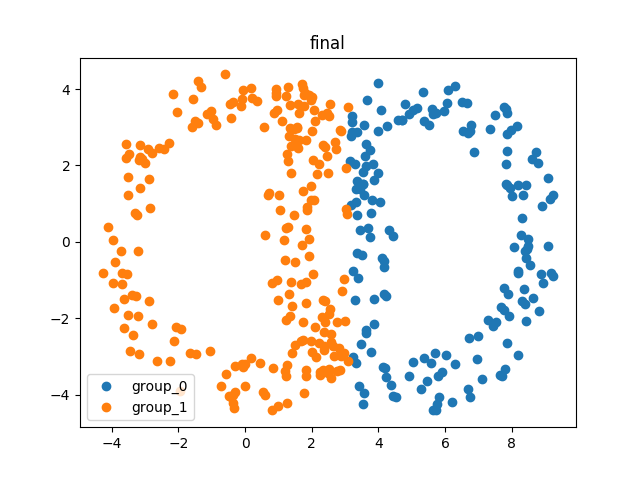
**Question 3: for the initialization of k means clustering used in k-means/kernel k-means/ spectral clustering, you can try different ways and show corresponding results. You will get bonus points if you do so**

Ans: Since the biggest disadvantage of the k-means is the initialize problem. The bad initialize strategy will lead the result stuck in the local minimum. There’re many different initialize methods which had been purposed. The most popular one is hierarchical clustering. The idea is that we can use bottom-up way to compose the cluster. After getting the result of hierarchical clustering, we can assume the tag by the result, and conduct the k-means secondly. It’s the general way and common used. However, the hierarchical clustering should spend time to construct the group.

Another method is k-means++ which is also a creative idea. The concept of initialization in k-means++ is to find the point with the maximum distance between each known centroids, and determine the centroids one by one. The advantage is that it’s not time consuming than the hierarchical clustering. As the result, I adopt this idea finally.

In the following six images, I should the result after adopting the initialization of k-means++. The two images in 1st row are the result of usual k-means; the two images in 2nd row are the result of kernel k-means; the two images in 3rd row are the result of spectral clustering. As you can see, the whole three method can get the good performance.

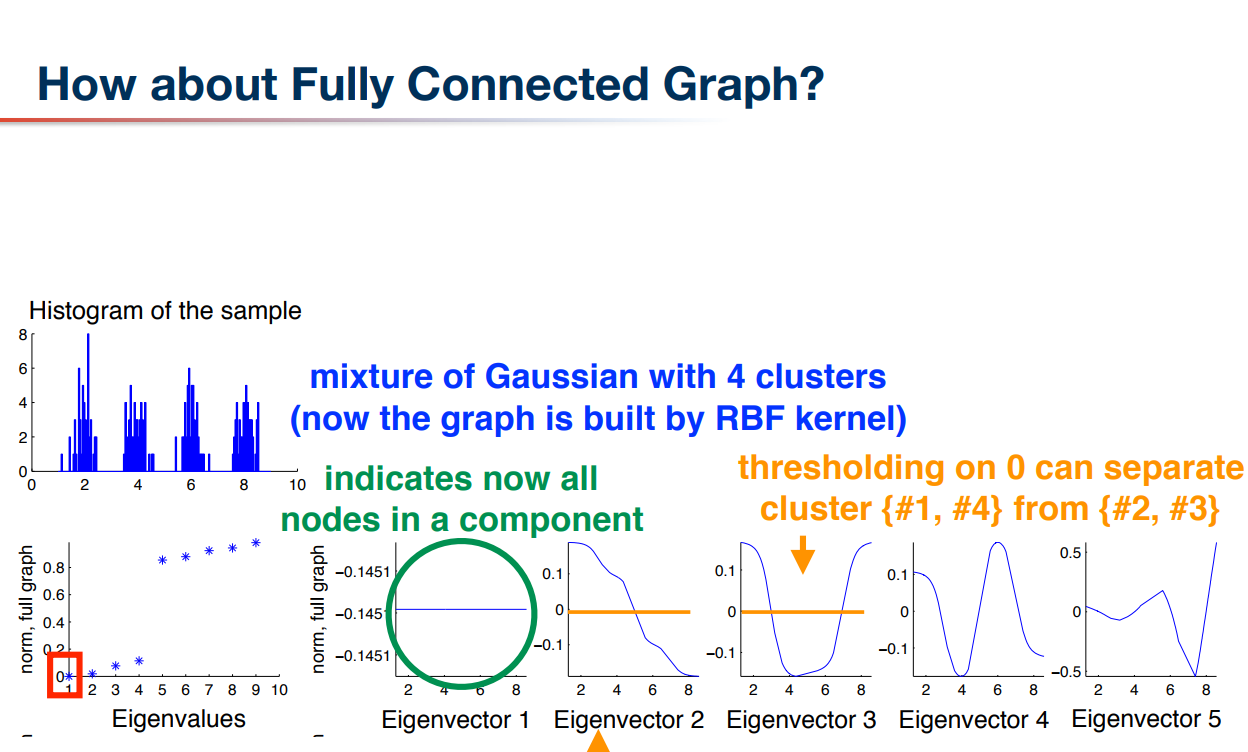




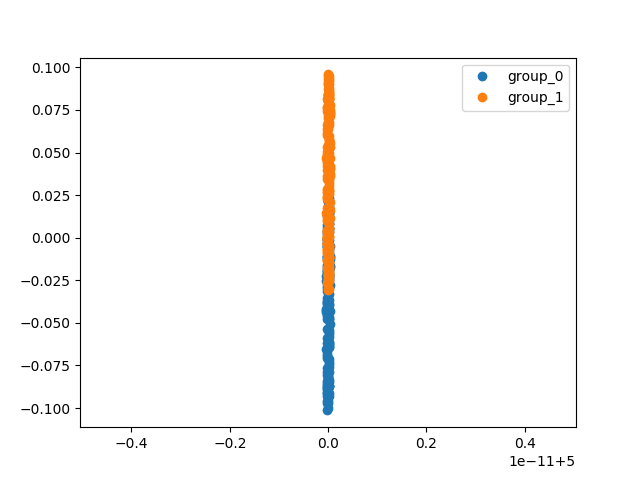
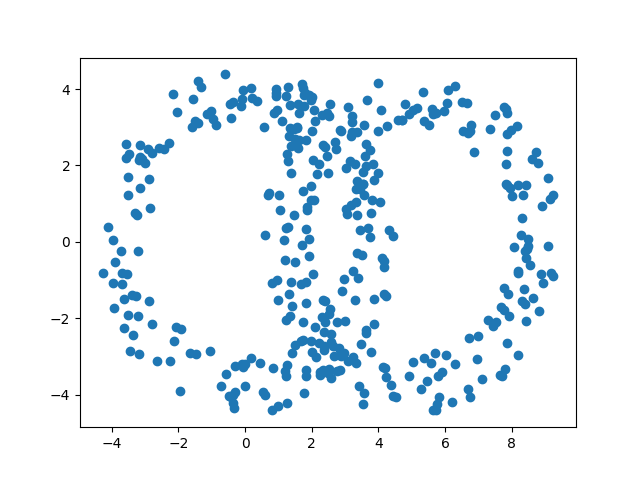
However, this initialization method isn’t the best one. It also have the disadvantage! Since it has the rules to generate the centroid, the rigid rules cannot adapt the whole situation of data distribution. In my experiment in kernel k-means toward test data 2, it just use 4 iteration to converge. However, the program should spend up to 10 epoch to converge while implementing k-means++.

**Question 4: For spectral clustering, you can see if data points within the same cluster do have the same coordinates in the eigenspace of graph Laplacian, discuss in the report to get bonus points.**

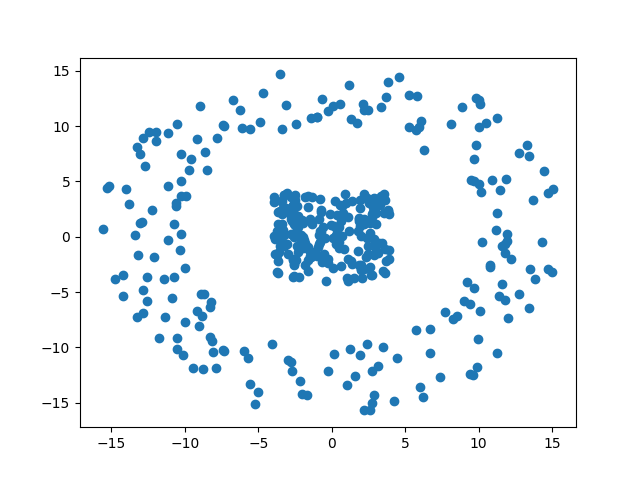
Ans: I also shows the project result in my experiment. Since we adopt the RBF kernel, the whole similarity graph only has one connected component. It means that it’s the similarity graph is fully connected, and we just can get one eigen value which is zero.



The above image is captured from the PowerPoint of professor Walon. As you can see, the value In the first eigen vector is almost the same. We can realize the difference between each point after observing the 2nd vector or the vectors with bigger eigen value. By this observation, we can assume that the value of first coordinate will be the same for whole data points if we project the vector into the 2-dimension space.



The above image shows the case of test data 1. The right part shows the project phenomenon which are render by the tag information in test ground 1 file. As you can see, the two bunch of data can be separated easily.



However, the above image shows the case in test data 2. And the projection result is shown in below. As you can see, the two groups of data are duplicated in the eigen vector space! The orange dots cover the blue points in the image. By this illustration, the not-perfect result I encounter in spectral clustering can be explained because the two groups of data are mutually distributed in the eigen space. As the result, even though the kernel trick is adopted, the best clustering result cannot be obtained at all.

