SENG 474 Assignment 3

Zheng Yin

March 24, 2021

1. K-Means Algorithm Analysis

In assignment 3 of SENG 474, we need to use the K-means algorithm (Lloyd's algorithm). K-means algorithm is an iterative algorithm that tries to separate the data into different subsets that do not overlap with each other. Each data point belongs to only one subset. The algorithm tries to make the data point in one subset as same as possible and make the subsets as different as possible from other subsets. At the same time, the sum of the squared distance between all data points and the cluster's centroid (the mean of all the data points that belong to that cluster) is the smallest as possible. Less different in a cluster, the data pints are more similar within the same cluster ("K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks", Imad Dabbura, Sep 17,2018, https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a)

$$C_i = \frac{1}{||S_i||} \sum_{x_j \epsilon S_i} x_j$$

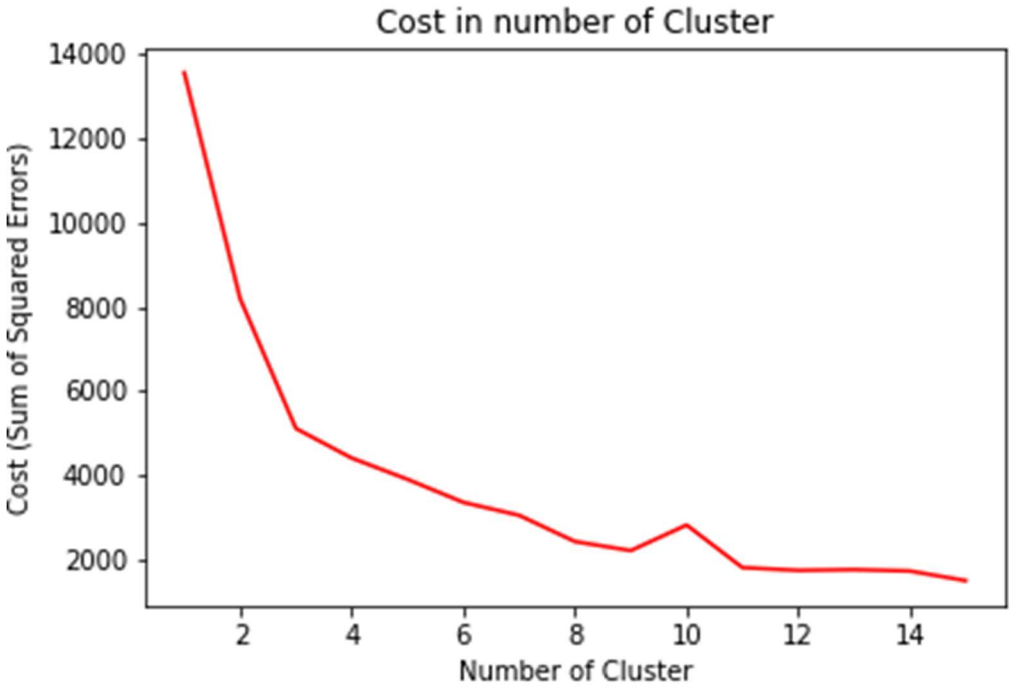
("Understanding K-Means, K-Means++ and, K-Medoids Clustering Algorithms", Satyam Kumar, Jun 10, 2020, https://towardsdatascience.com/understanding-k-means-k-means-and-k-medoids-clustering-algorithms-ad9c9fbf47ca)

This is the function that calculates the centroid point in a cluster. Which C_i is the I'th centroid, S_i are all points in the data which have a centroid as C_i. X_j is the j's point from the dataset.

In some cases, the K-mean algorithm may cause the problem of initialization sensitivity which will affect the final clusters due to the randomization of picking k-centroids points. K-mean++ algorithm picks the first centroid point randomly and uses the first centroid point to find the distance to every other point to define the new centroid.

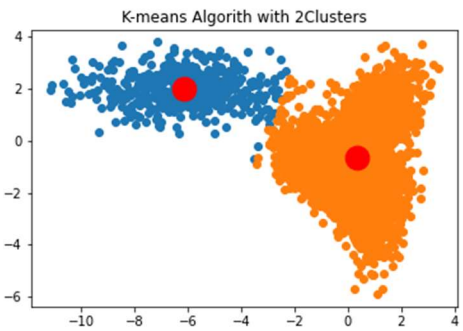
1.1 2D dataset

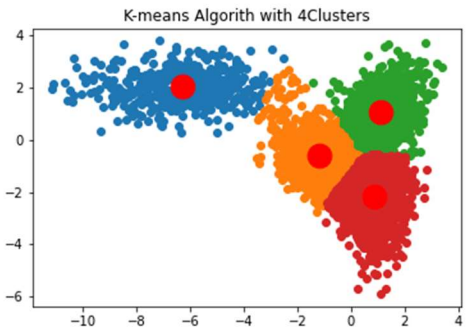
Provided dataset1.csv is a 2d dataset that was generated by a Gaussian mixture model. Based on the K-mean algorithm, the first test should be uniform random initialization which I chose K value is between 1 to 16 as shown as in figure 1, in which the cost is the sum of squared errors.

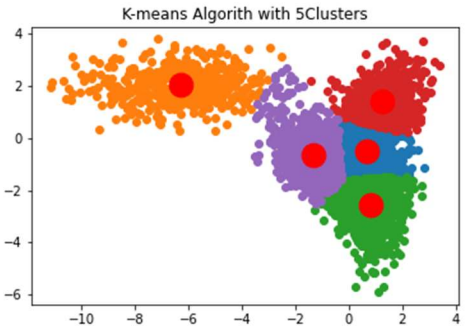


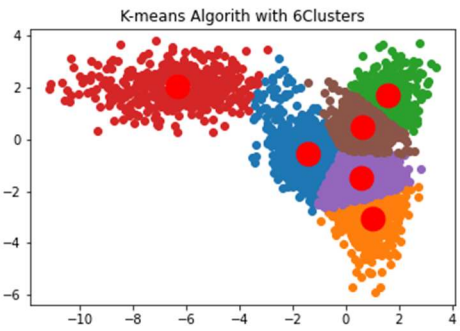
As we can see from the figure 1, using the elbow method, the optimal number of clusters is K = 3.

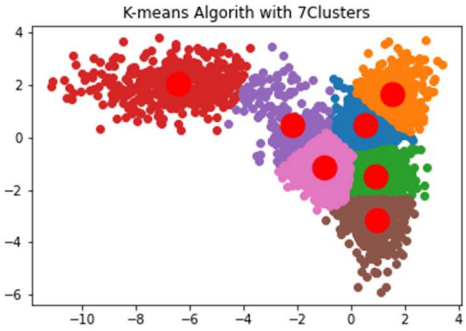
Now we have the K value from 2 to 17 to produce plots which are the figures below shows:

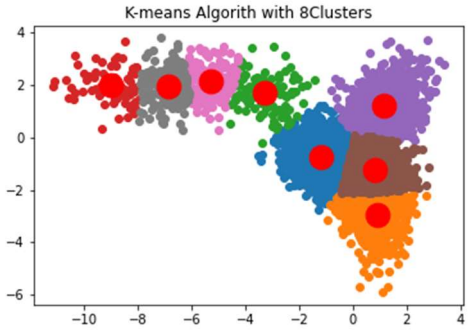


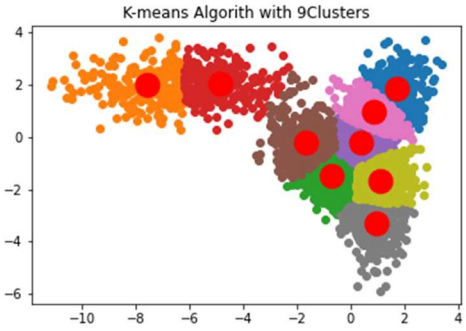


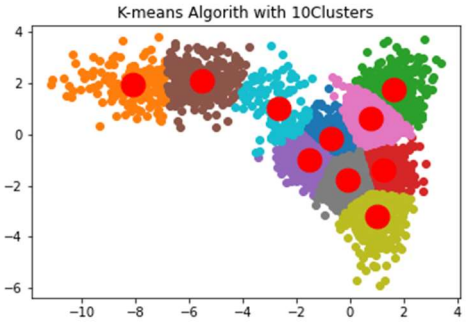

K-means Algorith with 5Clusters

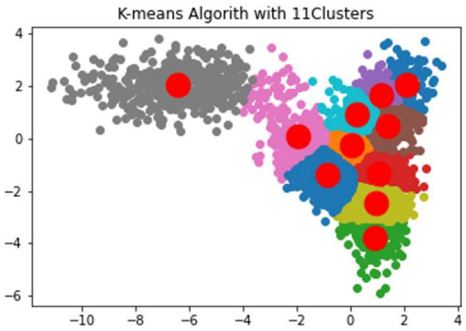

K-means Algorith with 6Clusters

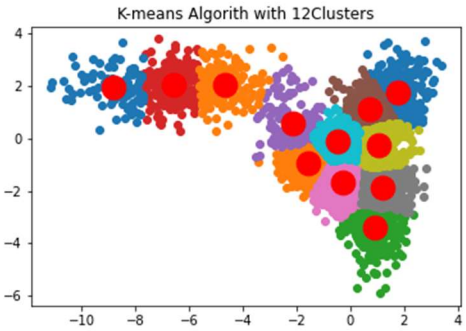

K-means Algorith with 7Clusters

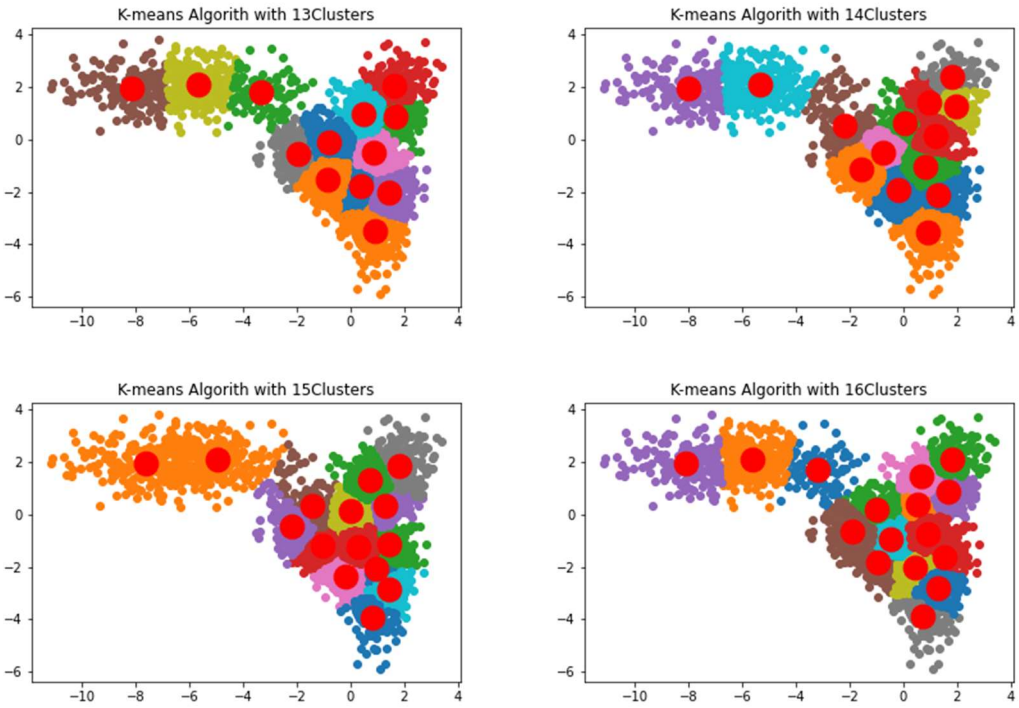

K-means Algorith with 8Clusters


K-means Algorith with 9Clusters


K-means Algorith with 10Clusters


K-means Algorith with 11Clusters

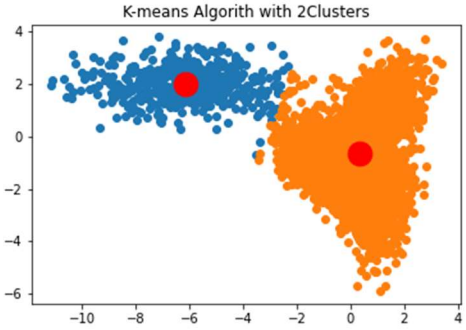

K-means Algorith with 12Clusters

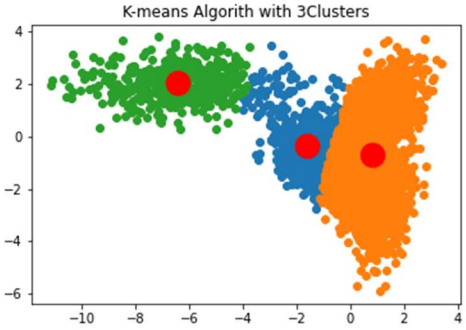


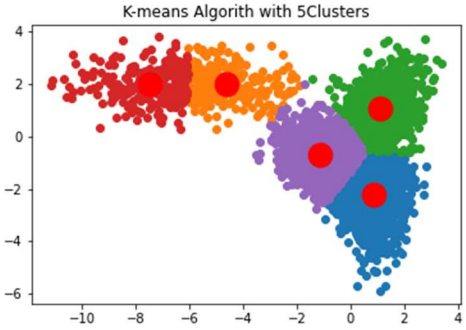
The second initialization method was based on the k-means++ algorithm. The K value is the same as the one that is above. But the optimal number of clusters is 4.

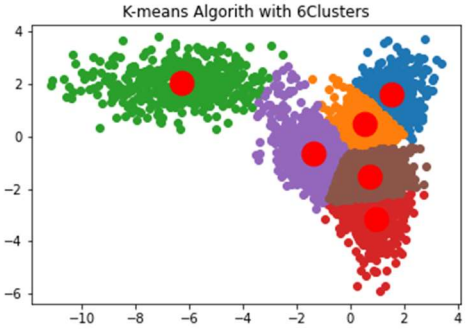
Now there is the same number of the chart with the k-mean. Which K value is from 2 to 16. The plot is displayed below.

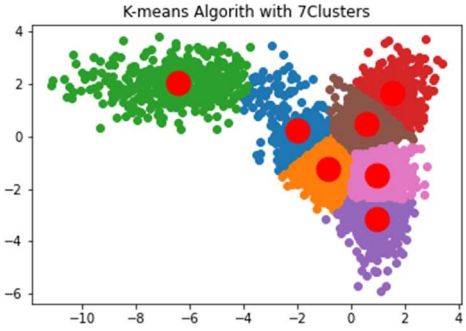
Now there is the same number of the chart with the k-mean, which K value is from 2 to 16. The plots are displayed below.

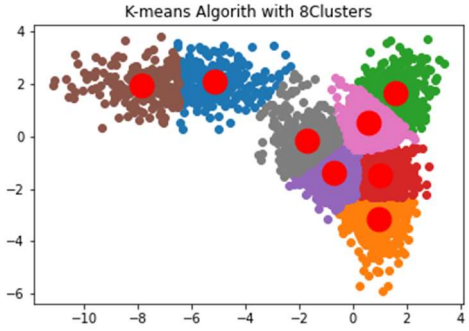


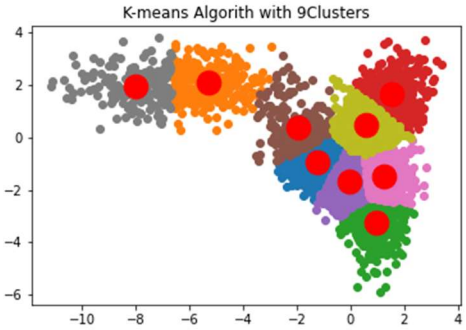


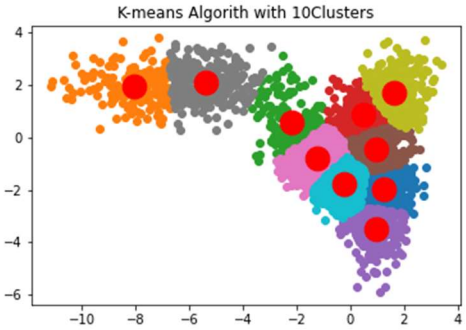


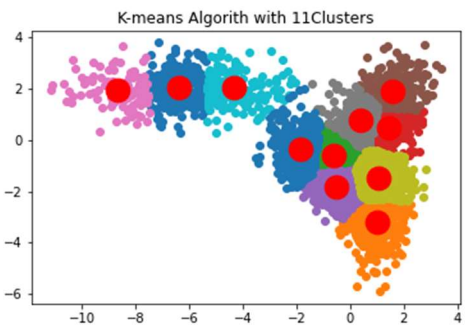


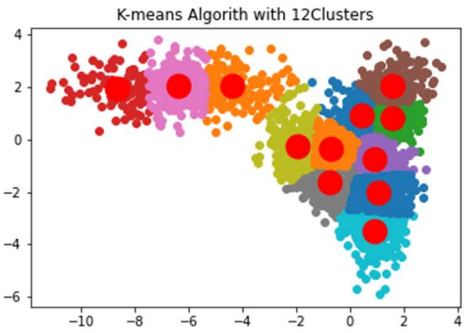


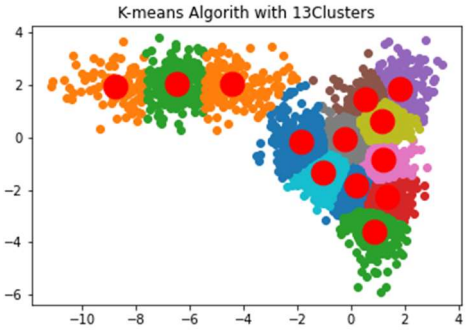


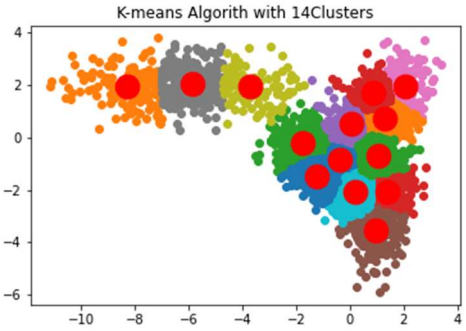


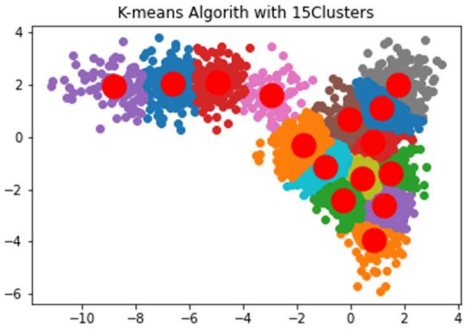


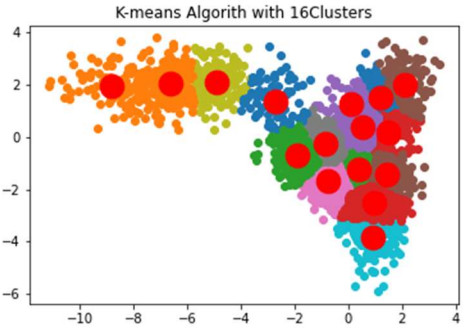

K-means Algorith with 11Clusters


K-means Algorith with 12Clusters


K-means Algorith with 13Clusters


K-means Algorith with 14Clusters

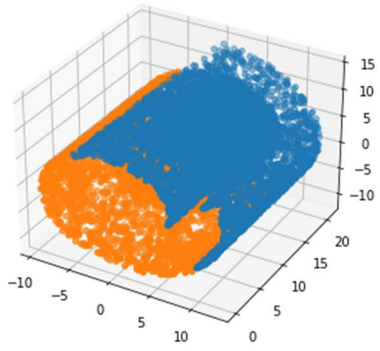

K-means Algorith with 15Clusters


K-means Algorith with 16Clusters

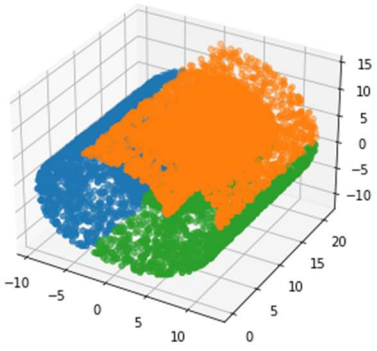
1.2 3D dataset

Dataset2.csv is three-dimensional data, like the function we do for the 2d dataset, we need to initial the uniform random initialization first which K-value is same with the

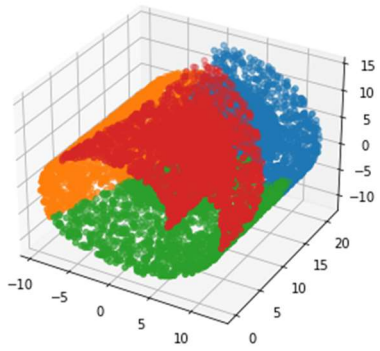
By the chart we got from cost, we can consider the optimal number of clusters is about 8. The chart is shown below that K value is from 2 to 16 which 8 is not include.



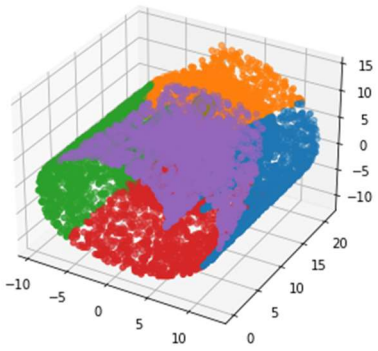
K =2



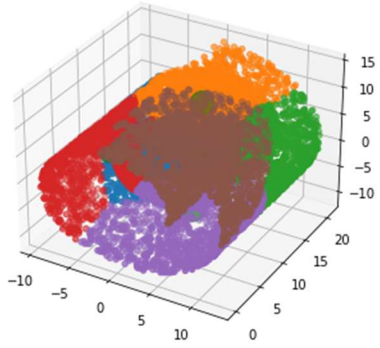
K=3



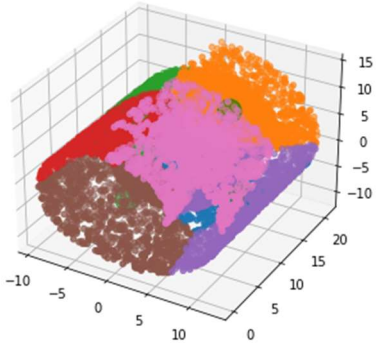
K=4



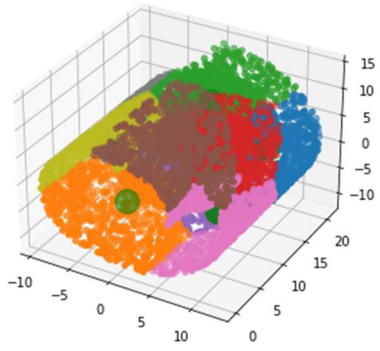
K=5



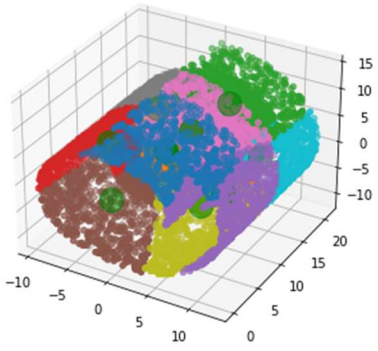
K=6



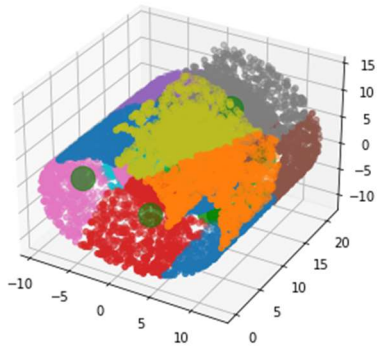
K=7



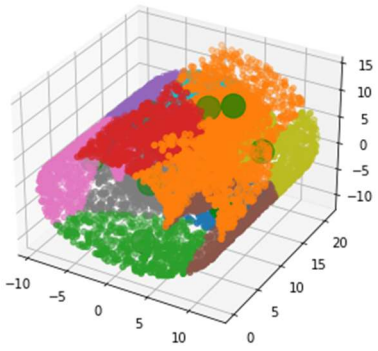
K = 9



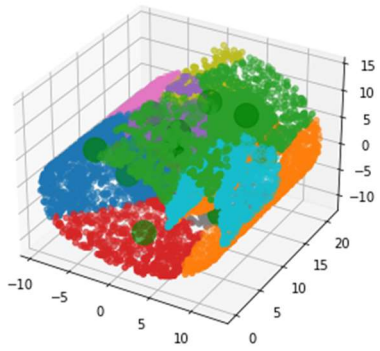
K=10



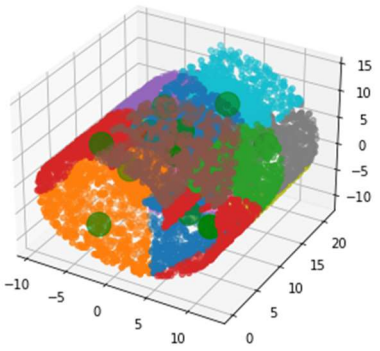
K=11



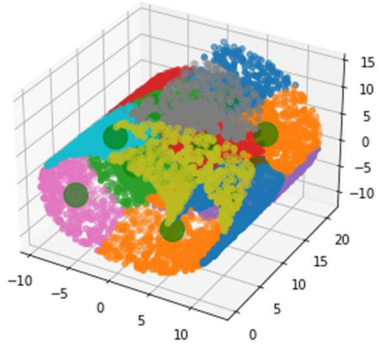
K=12



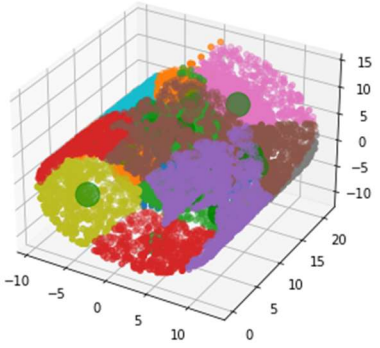
K=13



K=14

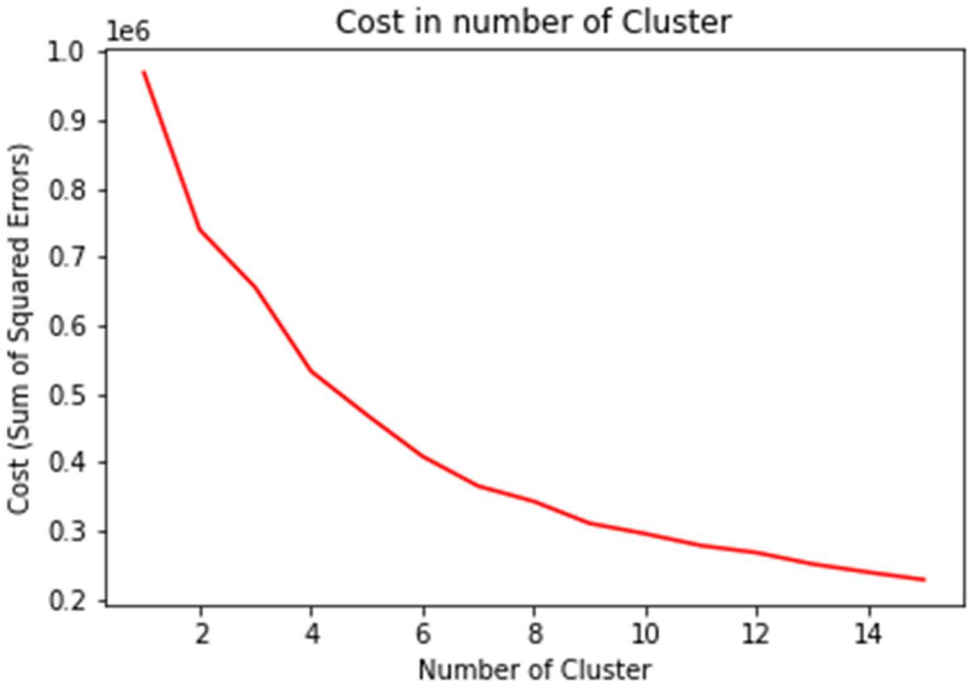


K=15

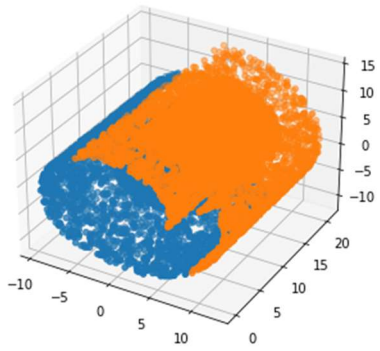


K=16

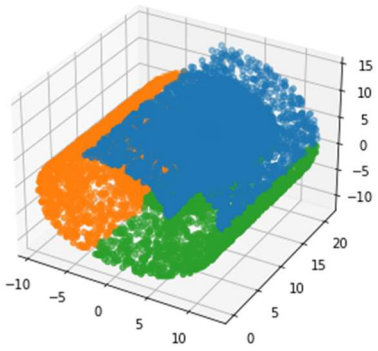
The second method that apply on the dataset 2 is K-mean++, which the parameter is like the experience in K-mean. And we can get an optimal number of clusters about 8.



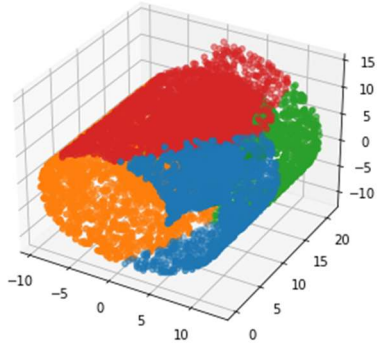
By the optimal number 8, we can get the chart below:



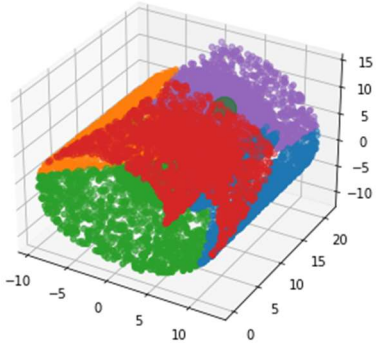
K=2



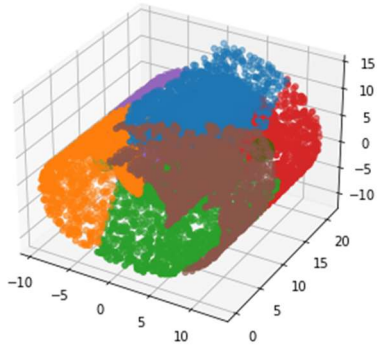
K=3



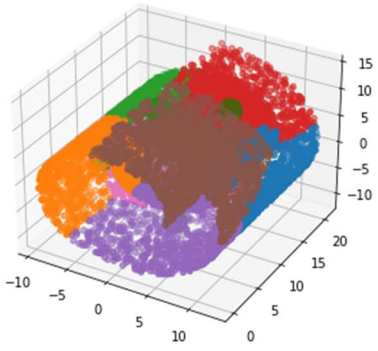
K=4



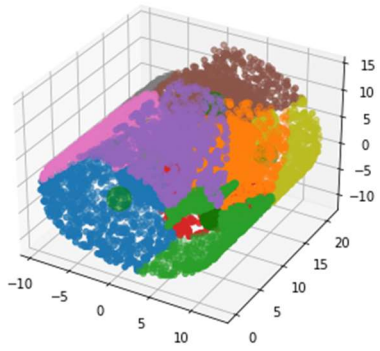
K=5



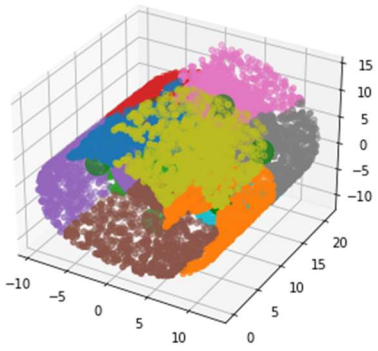
K=6



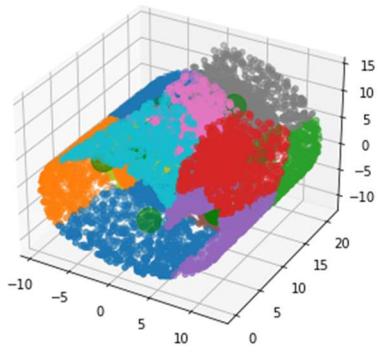
K=7



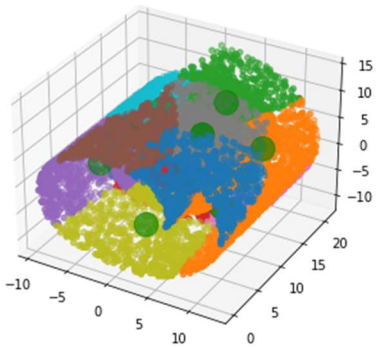
K=9



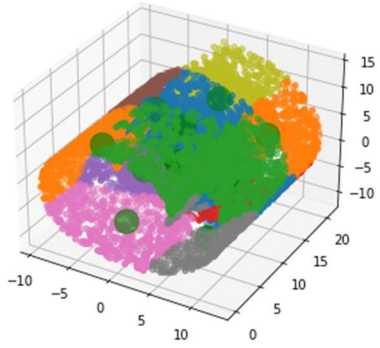
K=10



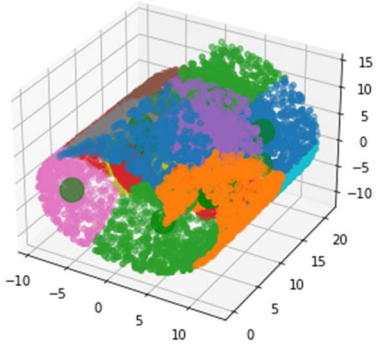
K=11



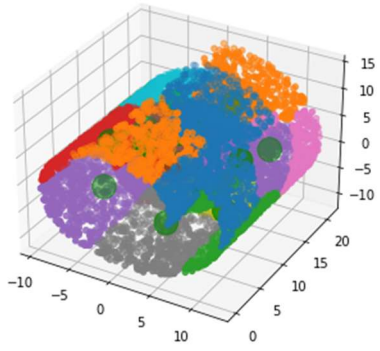
K=12



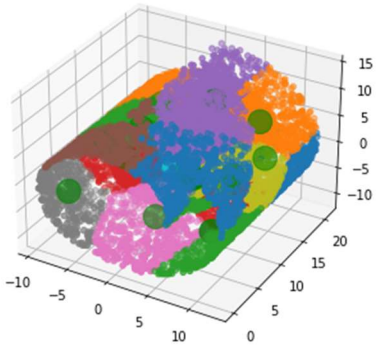
K=13



K=14



K=15



K=16

By the experiments using the K-means algorithm, K-mean algorithm seems worked better on 2D situation which the cost on the y-axis of both 2d cost vs cluster chart is larger and easy to fin the optimal number, but the line in 3d cost vs cluster chart is smoother which is hard to tell the right optimal point.

On the other side, there is a sensitivity problem in the uniform random initialization which mentioned at beginning. In the 2d uniform random initialization chart, there is a unnormal point on cluster at 10. That happened many times in experiments.

2. Hierarchical Agglomerative Clustering Analysis

Hierarchical agglomerative clustering (HAC) separates the similar object into clusters. We can use dendrograms to visualize the history of groupings and the optimal number of clusters. Normally, the optimal number is the longest vertical line in the chart.

In this assignment, we have 2d dataset and 3d dataset to test by HAC. In the HAC algorithm, there are two linkage dissimilarity measures: single linkage and average linkage.

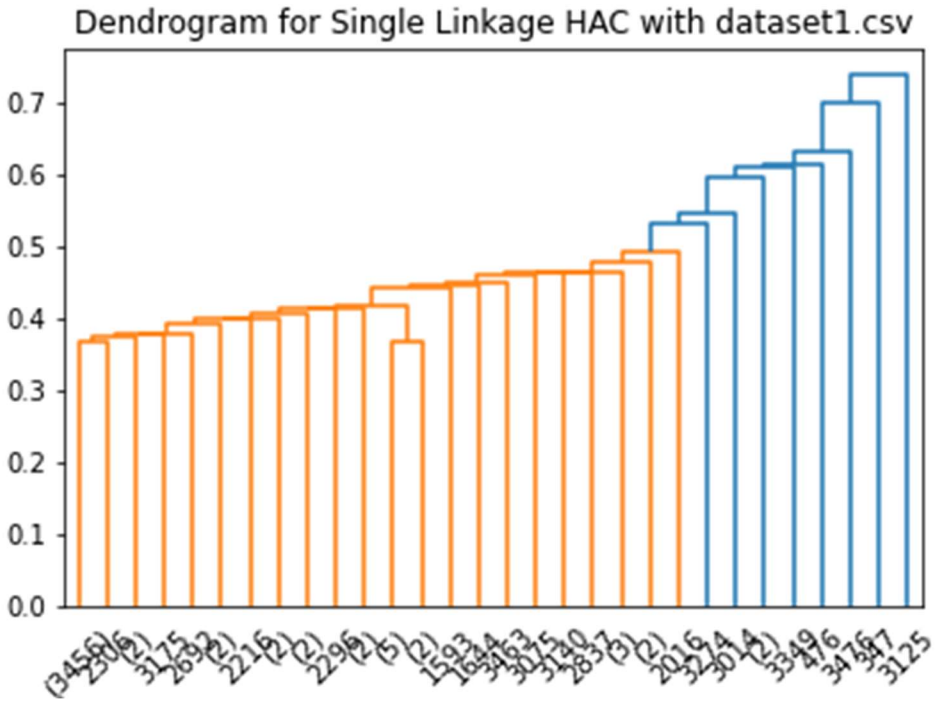
To figure out the optimal number of clusters, there are three steps:

1. Determine the largest vertical distance that does not intersect any of the other clusters.
2. Draw a horizontal line at both extremities.
3. The optimal number of clusters is equal to the number of vertical lines going through the horizontal line.

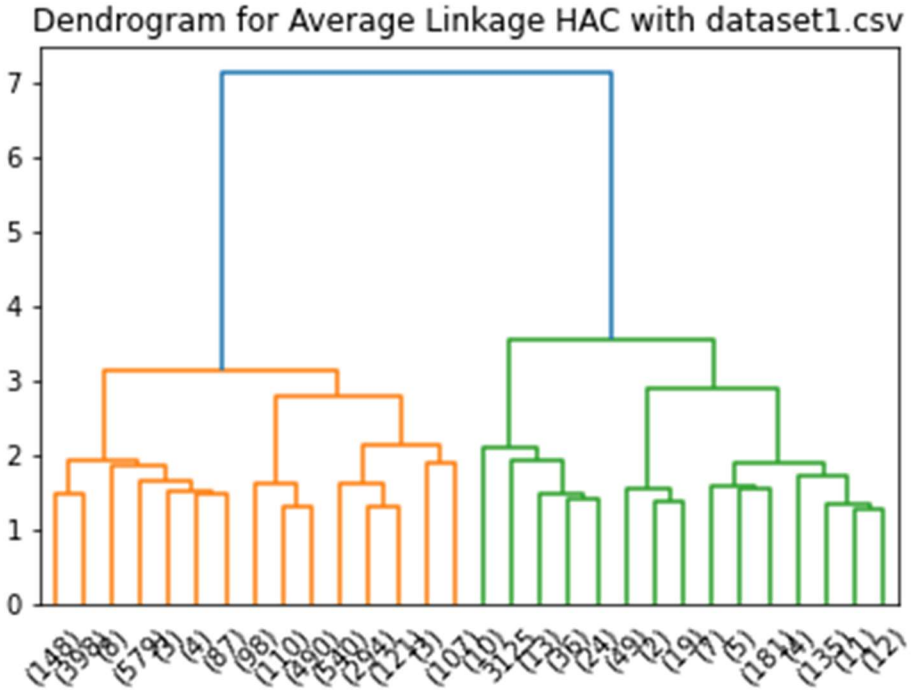
("Hierarchical Agglomerative Clustering Algorithm Example in Python", Cory Maklin, Dec 31, 2018, https://towardsdatascience.com/machine-learning-algorithms-part-12-hierarchical-agglomerative-clustering-example-in-python-1e18e0075019)

2.1 2D dataset

The first dataset of HAC testing is on 2-dimensional data in dataset1.csv, which were generated by the Gaussian mixture model. The chart below shows the dendrogram for HAC.

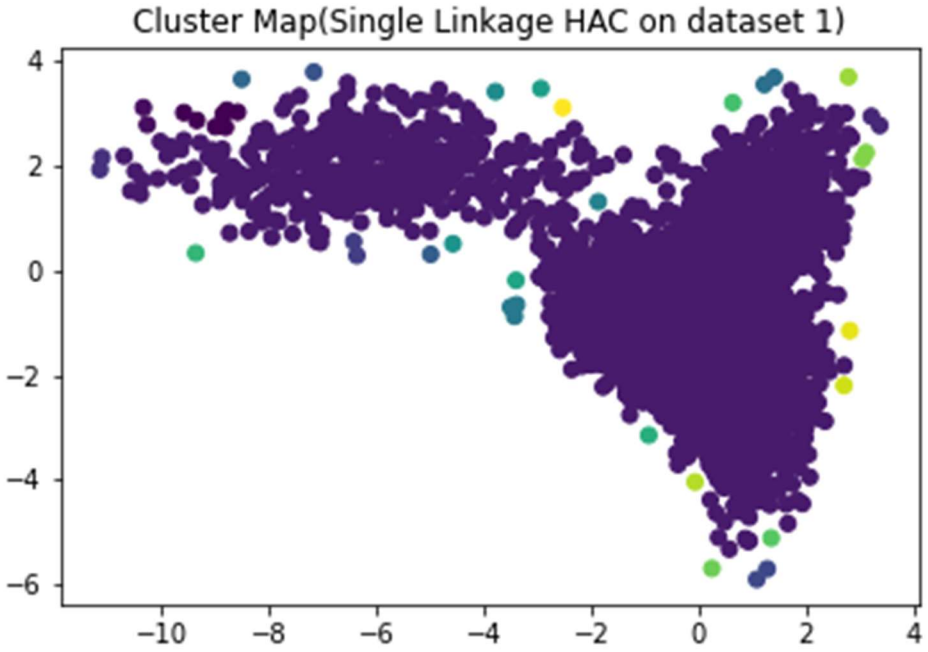


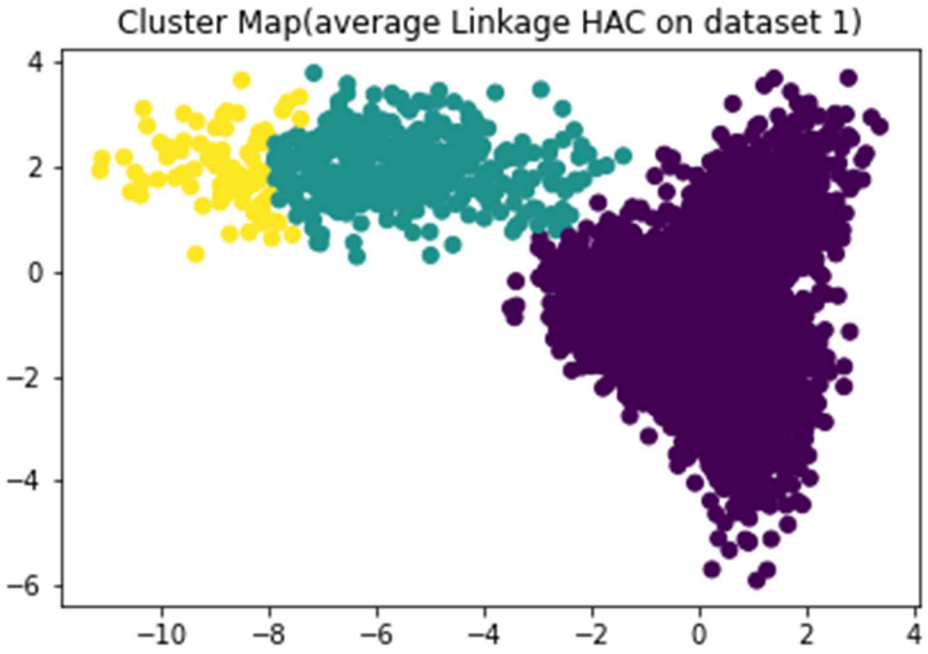
For the dendrogram of single linkage, the longest vertical distance is below 0.36 which there are 30 lines in that area, so the optimal number of clusters should be 30.



The chart above shows the dendrogram for average linkage, and we can cut between 3.5 and 7 which that is the longest vertical line and there are two line in that area, so the optimal number of clusters is 2.

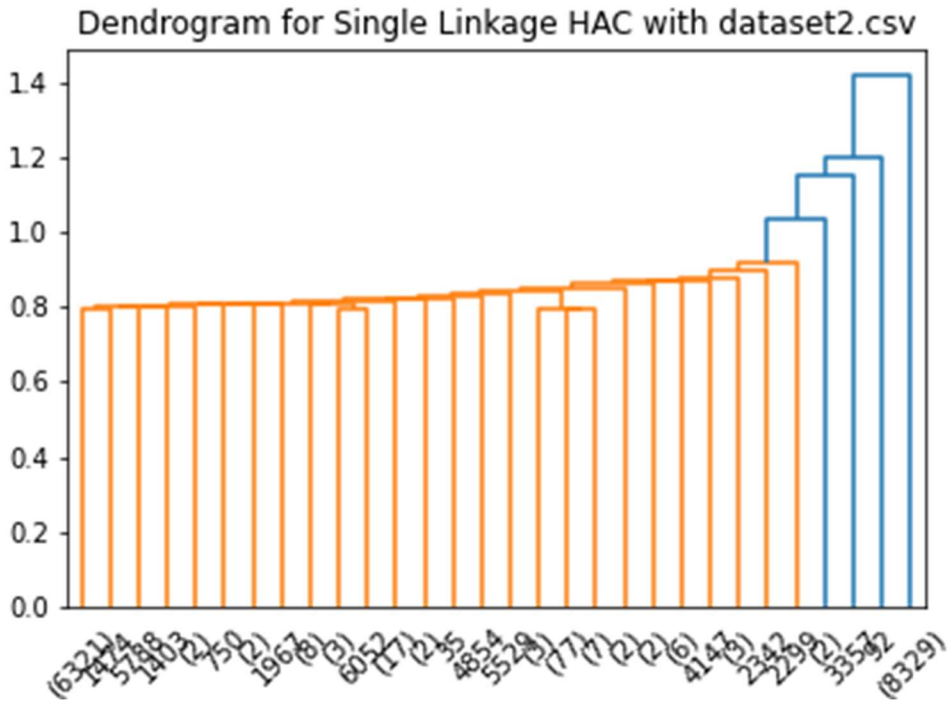
Then we get the cluster map below to show the cluster.



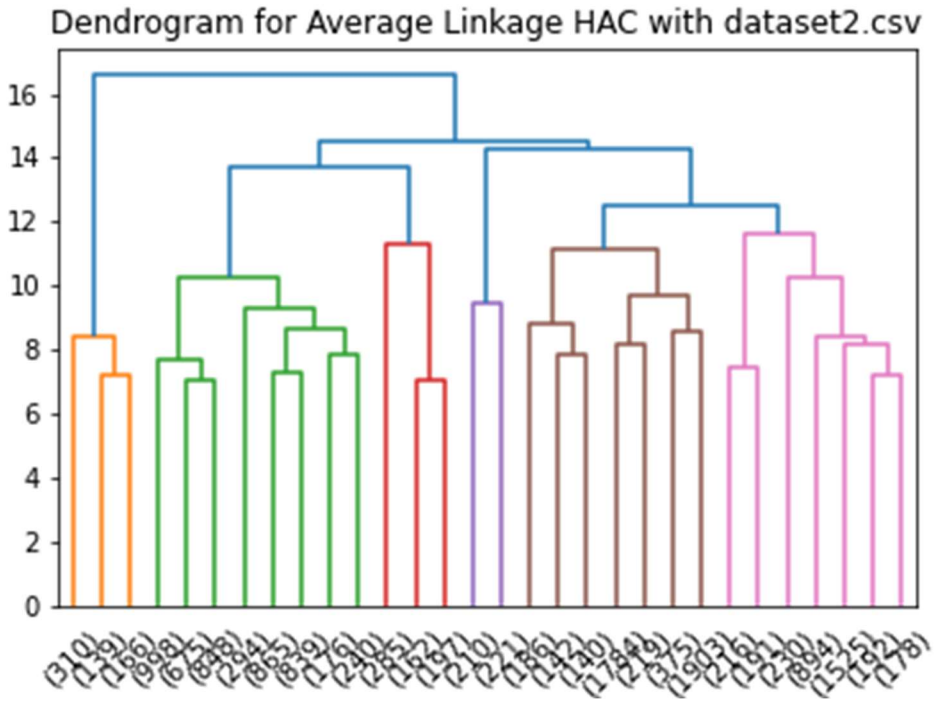


2.2 3D dataset

The three-dimensional data is available in dataset2.csv, the chart below shows the dendrogram for HAC.



We can cut the dendrogram of single linkage to find K value. The longest vertical line is below 0.8 which include 30 lines, so the optimal number of clusters should be 30.



I choose 30 as the K value to cut the plot which the longest vertical line is at 0 to 8.

In conclusion, the experiments by HAC algorithm show HAC have better performance in average linkage which single linkage always produce only two group of data. For 2d dataset, HAC did not perform as well as in 3d dataset which only have two group of data.

For opinion, single linkage in HAC may be useful for getting the data that not in main category, and average linkage can give a group separation scheme