Analysis of Clustering Methods for Data Mining

Xuhui Wang

March 2021

Introduction

In this report I introduce clustering methods. Specifically, I will be running two types of clustering methods, Lloyd's algorithm ("k-means") and hierarchical agglomerative clustering, on two datasets. To other motivation of this project is to get experience running clustering methods and interpreting the results, including how to go from results to selecting a good number of clusters.

Algorithms

I will primarily use two algorithms for the analysis:

- Lloyd's algorithm (k-means) with two forms of initialization: uniform random initialization and k-means++ initialization.
- Hierarchical agglomerative clustering with two forms of dissimilarity measure: single linkage and average linkage.

Clustering Methods for Dataset 1

Method

The first dataset, **dataset1.csv**, consists of 3500 two-dimensional examples. These examples were generated by a Gaussian mixture model. [1]

On this dataset, for each version of Lloyd's algorithm, we try different values of k (starting from 2 and increasing) and, for each value of k, run the method multiple times and pick the best result. We would make a plot of the cost as k increases. Then, we using the plot, decide how many clusters to use, and our the choice.[1]

On this dataset, for each version of hierarchical agglomerative clustering, we somehow decide what final clustering to pick (by making a cut in the dendrogram). we would explain our reasoning for deciding where to make a cut in the dendrogram. We would use the function **dendrogram** from the scipy package **scipy.cluster.hierarchy**. Since the number of examples is very large, we would use the option **truncate_mode = "lastp"**, in order to avoid showing too many lower levels of the dendrogram.[1]

Experiment - Lloyd's algorithm

Figure 1 respectively shows how costs of Lloyd's algorithm with uniform random initialization and k-means++ initialization change, as the k value increases in the range [2, 12]. From the figure, we could observe that the results of the different two methods are considerably similar, with small difference when k = 6 and 11. The costs remain a declining trend throughout the domain. It is theoretically reasonable that the algorithm with greater k values have less cost, because the overall Euclidean distances to the centroid in each cluster would be smaller as the number of clusters increases. So, for this problem, it seems that with large k values comes less cost.

On the other hand, after the harsh decrease from k =2 to k = 4, the costs transit to a relatively smooth descent. Because the costs stop rapid decreasing at k = 4 and the corresponding costs at k = 4 are acceptable, despite of better costs at k >= 4, we still work on the further analysis at k = 4. This is a typically classic strategy to pick k values for this algorithm.

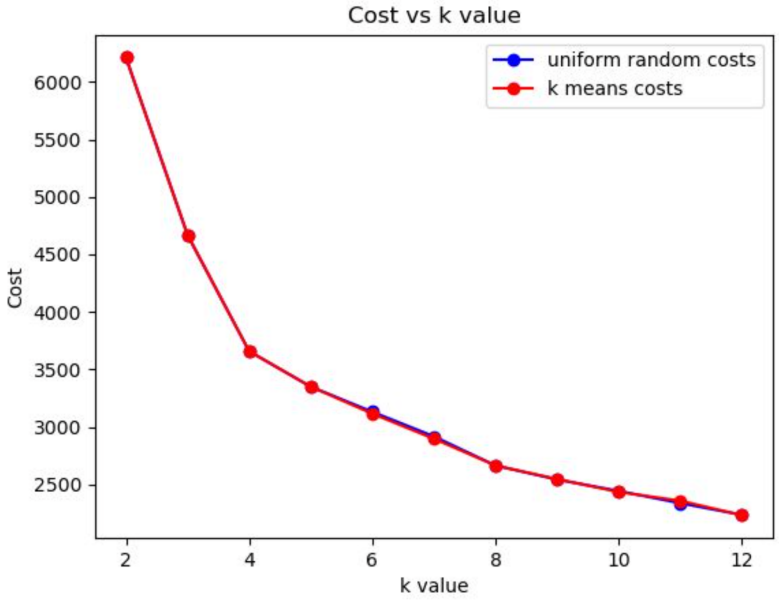
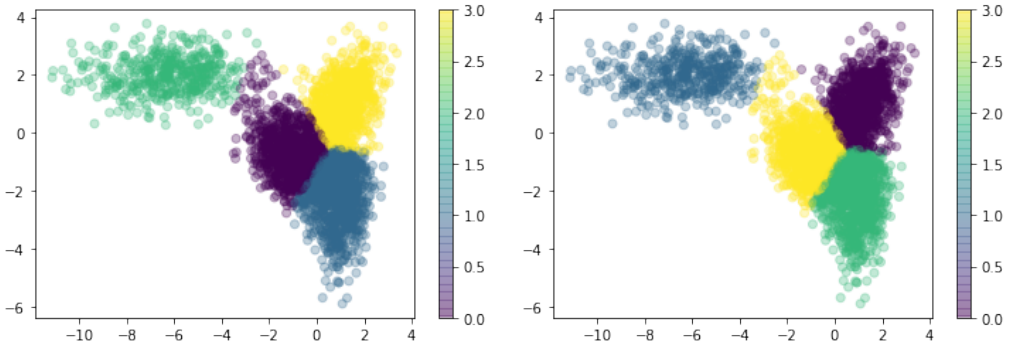


Figure 1: Lloyd's algorithm

Figure 2 respectively shows the scatter plots for Lloyd's algorithm with uniform random initialization and k-means++ initialization with k = 4. From the figure, we could observe that the clustering results for two different methods are similar to each other.



(a) uniform random initialization

(b) k-means++ initialization

Figure 2: Lloyd's algorithm scatter plot with k = 4

The points are separated into 4 different clusters spherically. It could be expected that the points on the plane would be separated into more spherically-shaped clusters if greater k values were chosen to perform the algorithm.

Experiment - Hierarchical agglomerative clustering

Figure 3 shows the dendrograms for hierarchical agglomerative clustering with single linkage. From the figure, we could observe that the distances between different merging nodes of the dendrograms are very close, and the far left node in the figure already contains as many as 3490 points. It could be expected that we do not obtain good results whatever final clustering to pick (by making a cut in the dendorgrams). Given these characteristics, we may want to guess that this method does not work very well for this problem.

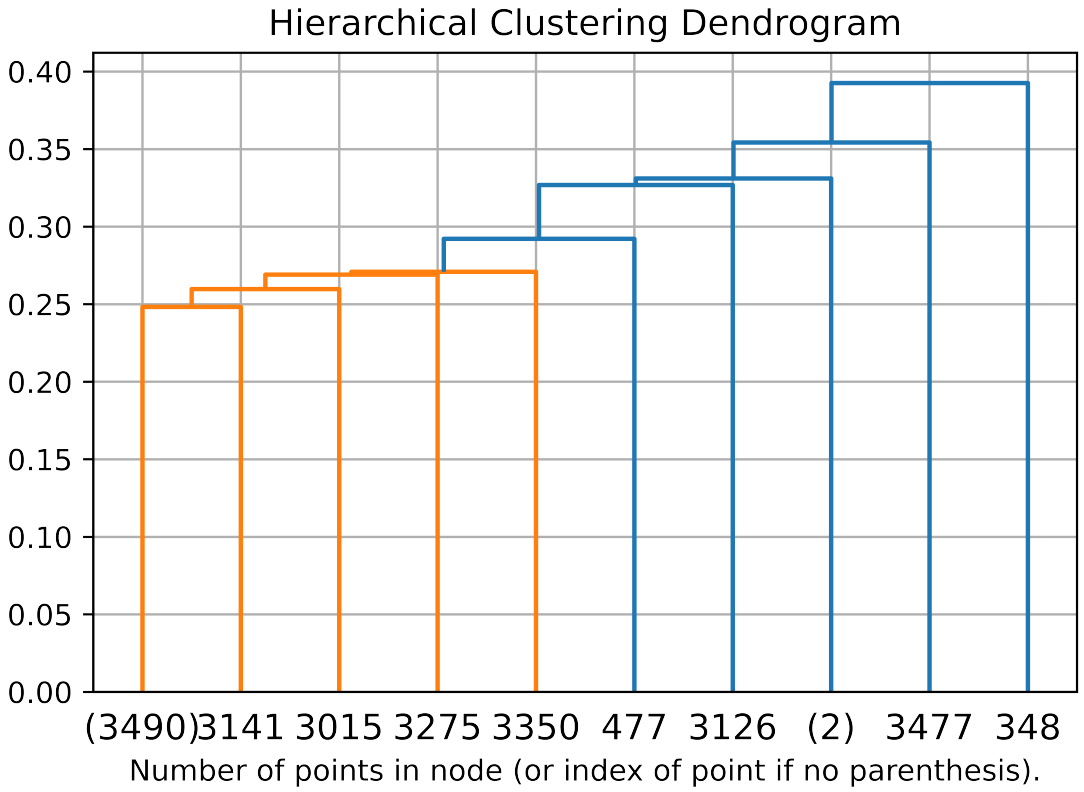
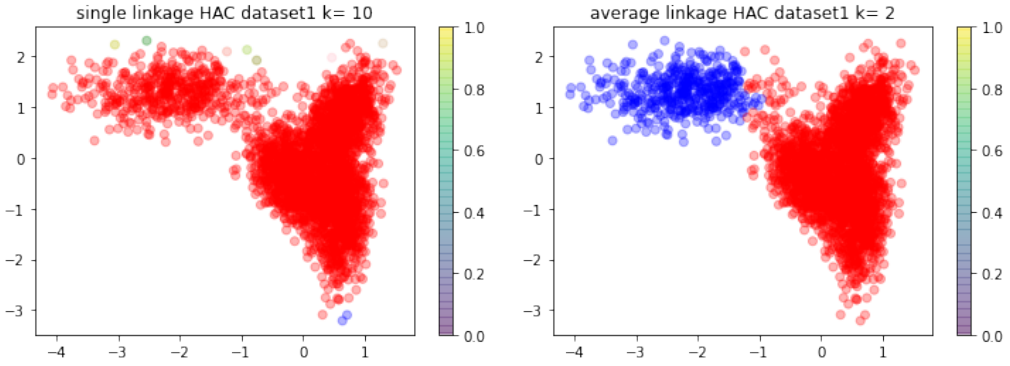


Figure 3: dendrograms for hierarchical agglomerative clustering with single linkage

Figure 4(a) shows how the algorithm works with single linkage and $k = 10$. As we assumed in the above, an overwhelmingly large number of points are separated into one cluster while only a few points are left outside.

Figure 5 shows the dendrograms for hierarchical agglomerative clustering with average linkage. From the figure, we could observe that, for the last two times of mergings, the distances between the nodes are much longer than those of previous mergings, so, it is safe to say that this method gives better results than using single linkage does.



(a) single linkage with $k = 10$ (b) average linkage with $k = 2$

Figure 4: hierarchical agglomerative clustering scatter plot

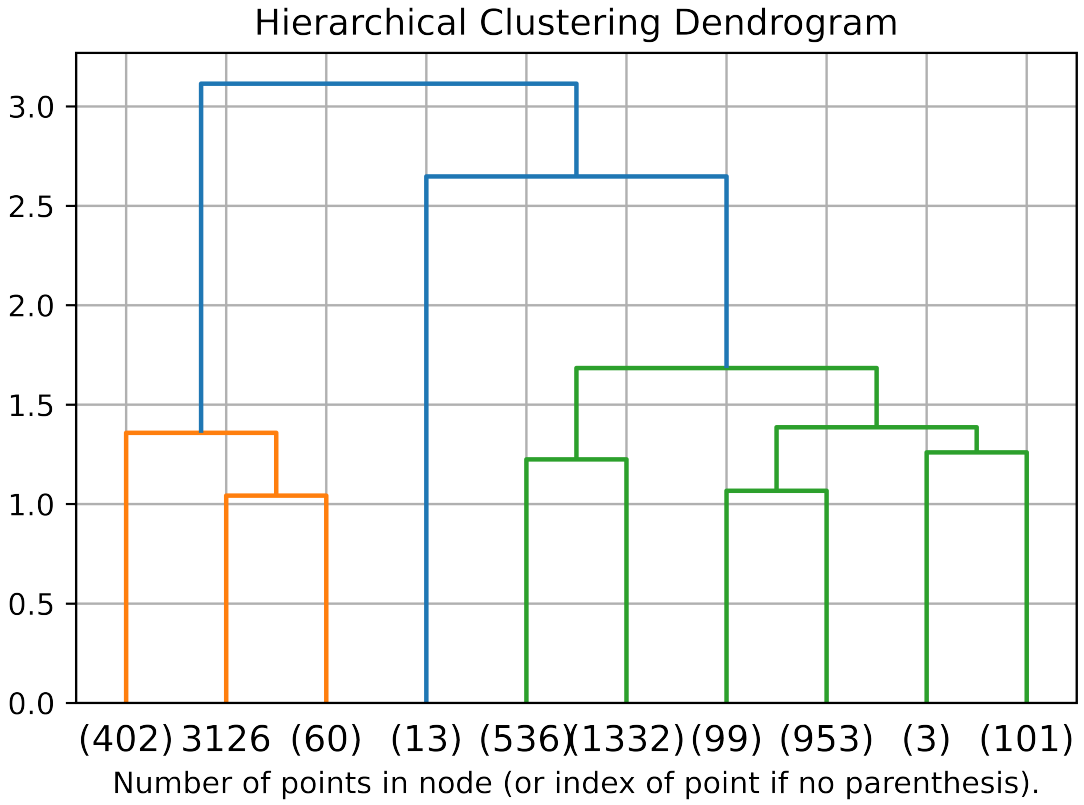


Figure 5: dendrograms for hierarchical agglomerative clustering with average linkage

It is intuitive to make a cut at *height* = 2.0 as a threshold. However, we notice that this cut would result in an independent cluster with only 13 points. Similarly, a cut at *height* = 1.6 receives four clusters but also includes that cluster with small size. Given that the distribution

of the raw data points looks more like binary on an Cartesian coordinate, we could make a cut at $height = 2.8$, which would divide the dataset into two clusters.

Figure 4(b) shows how the algorithm works with average linkage and $k = 2$. As we assumed in the above, the algorithm separates the data points into two different clusters correctly. This result is not as good as that of Lloyd's but acceptable.

Conclusion

For random or spherically-shaped data, we conclude that Lloyd's algorithm, including both uniform random initialization and k-means++ initialization, accomplishes more flexible and accurate classification results than hierarchical agglomerative clustering does. Meanwhile, hierarchical agglomerative clustering using average linkage gives better performance than using single linkage.

Clustering Methods for Dataset 2

Method

The second dataset, **dataset2.csv**, consists of 14,801 three-dimensional examples.[1]

On this dataset, for each version of Lloyd's algorithm, we try different values of k (starting from 2 and increasing) and, for each value of k, run the method multiple times and pick the best result. We would make a plot of the cost as k increases. Then, we using the plot, decide how many clusters to use, and our the choice.[1]

On this dataset, for each version of hierarchical agglomerative clustering, we somehow decide what final clustering to pick (by making a cut in the dendrogram). we would explain our reasoning for deciding where to make a cut in the dendrogram. We would use the function **dendrogram** from the scipy package **scipy.cluster.hierarchy**. Since the number of examples is very large, we would use the option **truncate_mode = "lastp"**, in order to avoid showing too many lower levels of the dendrogram.[1]

Experiment - Lloyd's algorithm

Figure 6 respectively shows how costs of Lloyd's algorithm with uniform random initialization and k-means++ initialization change, as the k value increases in the range [2, 10]. From the figure, we could observe that the results of the different two methods are considerably similar, with small difference when k = 9. The costs remain a declining trend throughout the domain. It is theoretically reasonable that the algorithm with greater k values have lesser cost, because the overall Euclidean distances to the centroid in each cluster would be smaller as the number of clusters increases. So, for this problem, it seems that with large k values comes less cost.

On the other hand, after the harsh decrease from k =2 to k = 4, the costs transit to a relatively smooth descent. Because the costs stop rapid decreasing at k = 4 and the corresponding costs at k = 4 are acceptable, despite of better costs at k >= 4, we still work on the further analysis at k = 4. This is a typically classic strategy to pick k values for this algorithm.

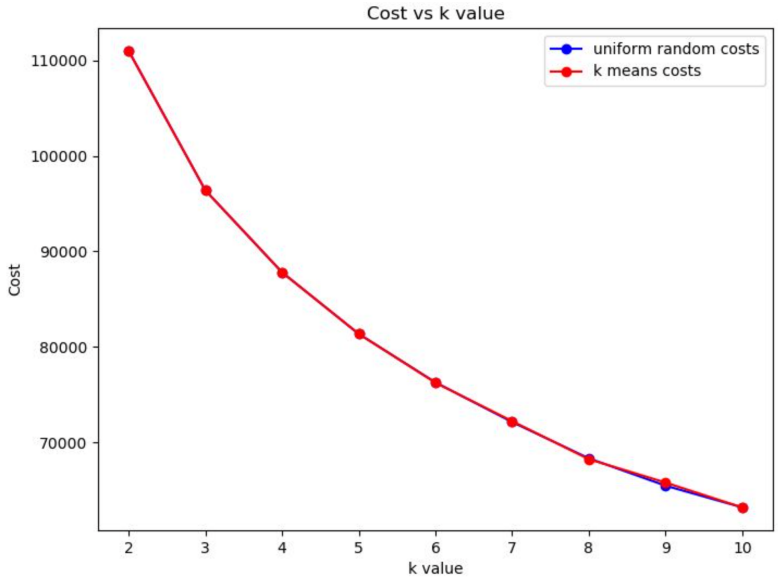
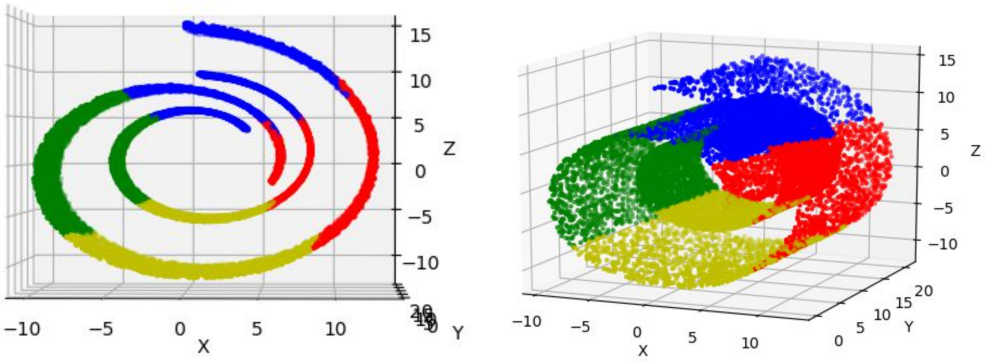


Figure 6: Lloyd's algorithm

Figure 7 shows the scatter plots for Lloyd's algorithm with uniform random initialization and k = 4. As we know, k-means clustering is found to work well when the structure of the clusters is hyper spherical. However, from the figure, we could observe that the data points are non-spherical in 3D and they are more like a couple of curved surfaces entangled. K-means separates the non-spherical dataset into 4 clusters based on the rule that for each cluster the overall Euclidean distances to the centroid are locally minimum, and gives a neat result. However, sometimes the aim of classification tasks is to figure out the governing connections that manipulate the scattered data (like, it would be great if the algorithm could recognize the two different curved surfaces as two different clusters). So, this geometrically-good result is not ideal for this problem.

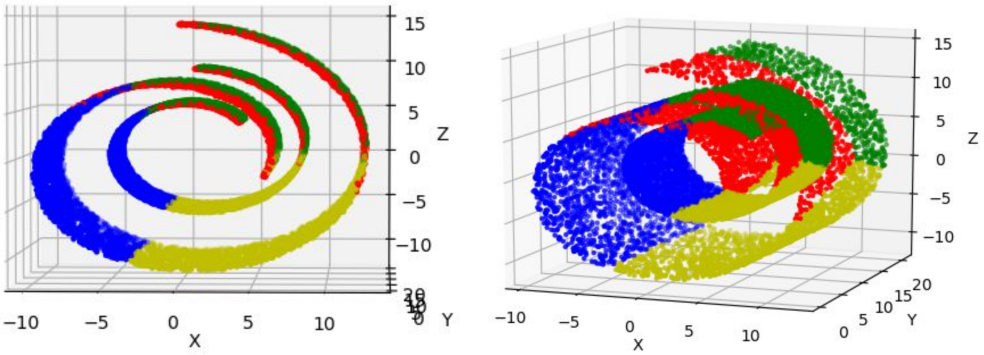


(a) projection of data on z-plane

(b) 3D viewpoint

Figure 7: Lloyd's algorithm with uniform random initialization and k = 4 3D scatter plot

Figure 8 shows the scatter plots for Lloyd's algorithm with k-means++ initialization and k = 4. The same problems as those of the previous method happens here, too.



(a) projection of data on z-plane

(b) 3D viewpoint

Figure 8: Lloyd's algorithm with k-means++ initialization and k = 4 3D scatter plot

Experiment - Hierarchical agglomerative clustering

Figure 9 shows the dendrograms for hierarchical agglomerative clustering with single linkage. From the figure, we could observe that the far left node has 6369 points and the far right node has 8329 data points. If we make a cut at *height* at 0.24, it is expected that we would obtain two relatively even clusters.

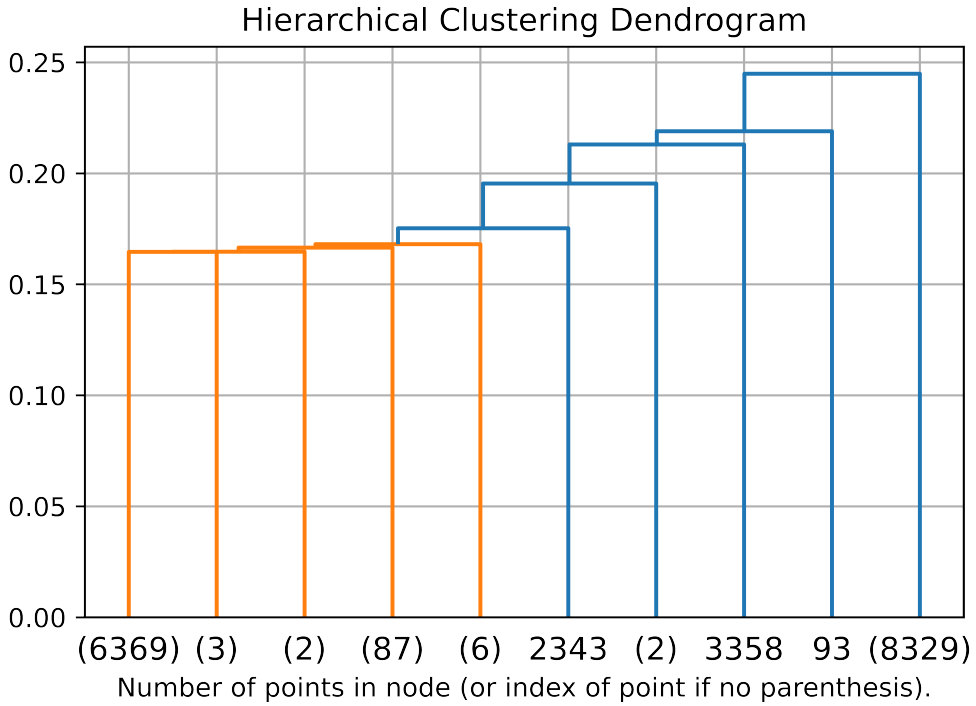
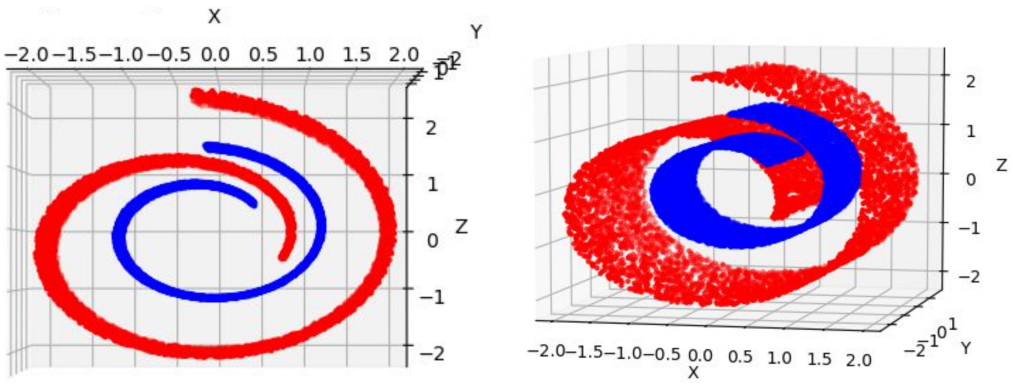


Figure 9: dendrograms for hierarchical agglomerative clustering with single linkage



(a) projection of data on z-plane

(b) 3D viewpoint

Figure 10: hierarchical agglomerative clustering with single linkage scatter plot

Figure 10 shows the scatter plots for hierarchical agglomerative clustering with single linkage and k = 2. Given that the distribution of the raw data points looks more like binary on an Cartesian coordinate, we make a cut at $height = 0.24$, which would divide the dataset into two clusters. As we assumed in the above, the algorithm separates the data points into two different clusters and it successfully recognizes the true patterns of two different data point clusters respectively. In other words, the governing connections that manipulate the scattered data (the two different curved surfaces) are correctly recognized.

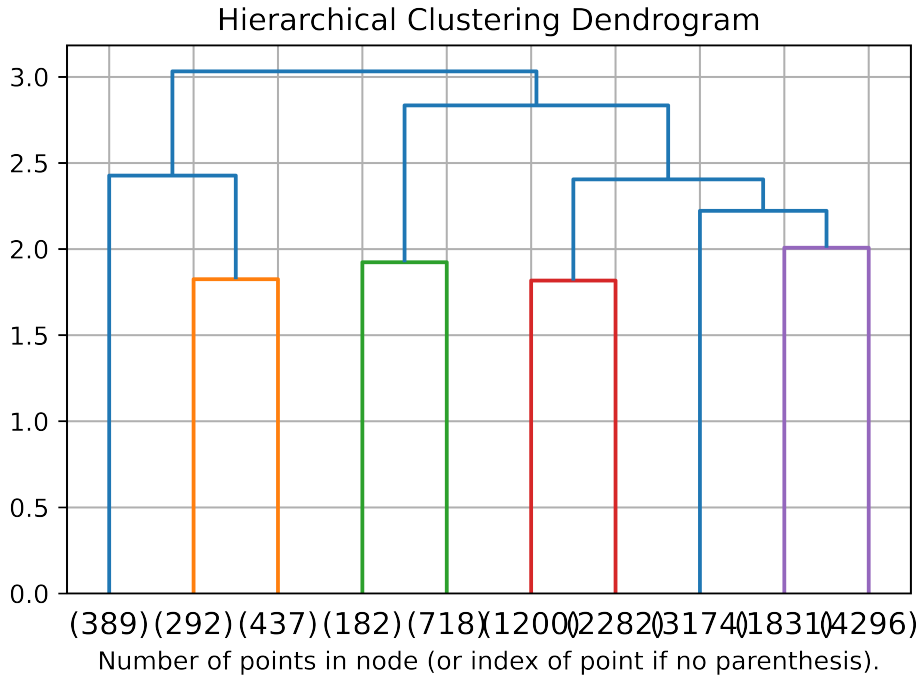
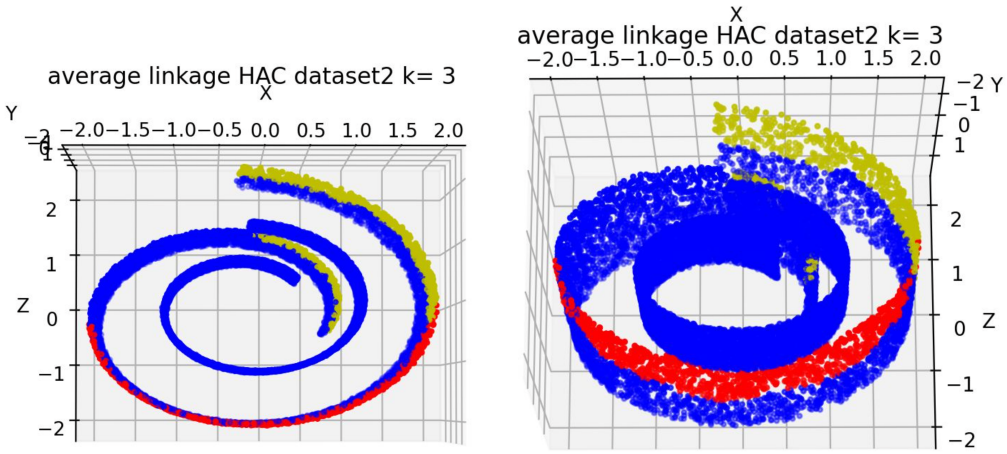


Figure 11: dendrograms for hierarchical agglomerative clustering with average linkage

Figure 11 shows the dendrograms for hierarchical agglomerative clustering with average linkage. From the figure, it is expected that the algorithm could separate the data points into three relatively even clusters if we intuitively make a cut at $height = 2.6$. And, each cluster would have a considerably large amount of data points.



(a) projection of data on z-plane

(b) 3D viewpoint

Figure 12: hierarchical agglomerative clustering with average linkage scatter plot

Figure 12 shows how the algorithm works with average linkage and $k = 3$. As we assumed in the above, the algorithm separates the data points into three different clusters correctly. One of the three clusters have much more data points than others. From the figure, we could observe that the three different clusters do not have very distinct features and the clustering is also not perfect geometrically. So this results is not ideal enough for this problem.

Conclusion

For non-spherically-shaped dataset, we conclude that though Lloyd's algorithm, including both uniform random initialization and k-means++ initialization, accomplishes good results geometrically, they neglect the intrinsic connections of data points (the governing connections that manipulate the scattered data) to some extent. In comparison, hierarchical agglomerative, especially using single linkage, gives better performance.

References

[1] N. Mehta, *Seng474: Data mining course materials.*