K-means Clustering

Xinyuan Min 950829573070

1. Cluster the data in the file 2dtest.csv, depicted in Fig. 1, into three clusters and print the output. Run multiple times k-means on this dataset (with k=3). Do you always get the same result? How many iterations are needed for converge?

   Use the same k for the same dataset, I run the algorithm for 5 times and get different result every time. This is because the initial centroids are selected randomly, the different initial centroids lead to slightly different clustering results.

   The number of iterations needed for converge are as follows:

	Iterations needed for converge
1st run	5
2nd run	2
3rd run	4
4th run	2
5th run	6

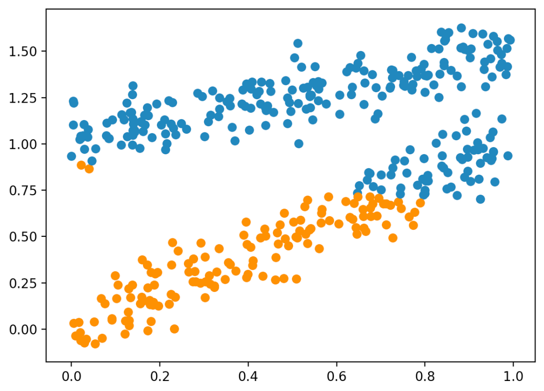
2. The data in the file LargeSet 1.csv represents measurements of 100 genes in 24 conditions. Run k-means multiple times for each k 2 {2, 3, 4, 5, 6}. (a) Do you always get the same result? How many iterations are needed for converge? (b) Which cluster sizes (number of elements in each cluster) do yo get for each of the multiple runs.

   Different clustering results were obtained every time

K value	1st run/ *iteration, cluster size*	2nd run	3rd run	4th run	5th run
2	4; [75, 25]	3; [75, 25]	4; [75, 25]	2; [66, 34]	9; [46, 54]
3	4; [25, 25, 50]	3; [10, 15, 75]	4; [25, 50, 25]	5; [52, 25, 23]	2; [14, 11,75]
4	4; [50, 25,10, 25]	6; [25, 25, 30, 20]	3; [50, 25, 10, 15]	4; [25, 30, 20, 25]	3; [11, 23, 52, 14]
5	5; [20, 25, 30, 10, 15]	2; [10, 30, 15, 25, 20]	2; [25, 50, 7, 8, 10]	5; [30, 17, 20, 8, 25]	7; [25, 25, 2, 28, 20]
6	2; [25, 3, 12, 30, 10, 20]	4; [30, 25, 2, 10, 13, 20]	10; [25, 16, 10, 4, 30, 15]	6; [30, 10, 14, 15, 11, 20]	5; [15, 20, 10, 10, 30, 15]

3. The data in the *file LargeSet 2.csv* is depicted in Fig. 2. It contains 400 points in two dimensions. Data are arranged so that the first 200 are in one cluster and the second 200 in another cluster. Run the k-means algorithm with k=2 multiple times. You will notice that it fails to recover the correct cluster structure. Explain why do you think this is happening.

   Because the optimization standard of k-means is to maximize within cluster homogeneity and between cluster heterogeneity, and the similarity is measured upon the Euclidian distance of each data point to the centroid it belongs.
   Though the pattern of this dataset is pretty obvious to us, k-means would still simply cluster it based on Euclidian distance, and leads to a cluster result that closely located points being assigned to the same cluster.

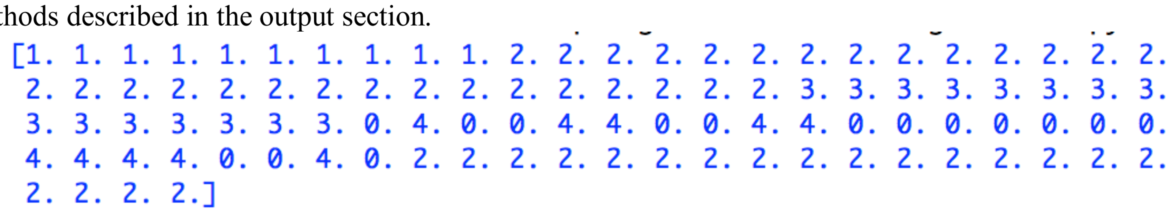


k-means cluster result of file *LargeSet_2.csv*

4. Find the optimal number of clusters for the data in LargeSet 1.csv. Run k-means multiple times for each k 2 {2, 3, 4, 5, 6}. Make sure that you run the algorithm a suciently high number of times (meaning that if you run it more times it is unlikely that the results improve).
   (a) What is the lowest W=WGSS/BGSS value that you find?
   The lowest wgss/bgss value I found is 0.21%

   (b) What is the k value that leads to the best clustering?
   K =5 leads to the most occurrences of the lowest wgss/bgss ratio

   (c) What is the best set of clusters that you find? Describe your output using any of the methods described in the output section.



```
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2.
 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 3. 3. 3. 3. 3. 3. 3. 3.
 3. 3. 3. 3. 3. 3. 3. 0. 4. 0. 0. 4. 4. 0. 0. 4. 4. 0. 0. 0. 0. 0. 0. 0.
 4. 4. 4. 4. 0. 0. 4. 0. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2.
 2. 2. 2. 2.]
```

k-means clustering result using the first output method

   (d) For the optimal k value, what is the variability in the WGSS/BGSS ratio that you find (mean and standard deviation, you don't have to code this in python, you can use other tools)
   Mean (k=5): 0.374%
   Standard deviation (k=5): 0.149

5. Is the k-means algorithm deterministic? Justify your answer.
   Deterministic algorithm is an algorithm which, given a particular input, will always produce the same output, with the underlying machine always passing through the same sequence of states.
   k-means starts with randomly chosen centroids and is sensitive to how the algorithm is initialized, in this sense, it's not deterministic. But except for the initialization, the algorithm is deterministic because it relies on exact mathematical expressions.

6. What is the memory complexity of the k-means algorithm you implemented?
   *the algorithm stores all data points and centroids, as well as an array of cluster assignment result of all data points for each iteration. The space complexity is O((np + k)*d + np)*

7. How does the running time depend on k, d and np (number of points)? How do these affect N iter (the number of iterations)?
   *The value of k determines the number of clusters/ centroids. In the assignment step, the Euclidian distance of each data points to each centroid is calculated, np*k times of operation are needed for each iteration. And according to the formula, the time needed for calculating Euclidian distance is proportional to d (dimensions), because you need to do one subtraction*

*and one power operation on each attribute. So, the time complexity for each iteration is O(np*k*d). Given Niter, the total time complexity is O(np*k*d*Niter)*

According to my answer of question 1 &2, I didn't observe an relationship between k and number of iterations. But intuitively, when the size of dataset (np & d) grows larger, it takes more iterations to converge.