K-means Clustering

Dataset preparation:

Use dataset `q_data`. Code for loading dataset into 2D python list: here

Train:

1. K = 4
2. Load dataset into 2D list "*Data*"
3. Randomly select K <u>different</u> data points from "*Data*" and store them into 2D list "*Centers*"
4. Initialize a 2D list named "*Clusters*" which contains K 1D lists for the K centers
5. **for** each sample/ data point "S" **in** "*Data*":
6. identify the center "C_*i*" that is the closest to "S"
7. Append "S" in "*i*"th list of "*Clusters*"
8. itr = 1, "*Shift*" = 0
9. **while** True:
10. **for** each 1D list "L" in "*Clusters*":
11. Determine the average of the data points. This is the new center of this list.
12. Update the center of this list in "*Centers*"
13. **if** itr > 1 **and** "*Shift*" < **50**: **break** (convergence)
14. "*Shift*" = 0
15. Initialize a 2D list named "*Temp_Clusters*" which contains K 1D lists for the K centers
16. **for** each sample/ data point "S" **in** "*Data*":
17. identify the center "C_*i*" that is the closest to "S"
18. Append "S" in "*i*"th list of "*Temp_Clusters*"
19. **if** S belongs to different clusters in "*Clusters*" and "*Temp_Clusters*" **then**
20. "*Shift*" = "*Shift*" + 1
21. Now "*Temp_Clusters*" 2D list contains K 1D lists
22. Assign "*Temp_Clusters*" to "*Clusters*"
23. itr = itr + 1
24. "*Clusters*" will contain your desired clusters and "*Centers*" will contain your desired centers at the end of loop
25. **Plot them with appropriate color**
26. "*inertia*" = 0
27. **for** each 1D list "L" **in** "*Clusters*":
28. "*inertia*" = "*inertia*" + sum of distances-square of data points of "L" from the center

Report:

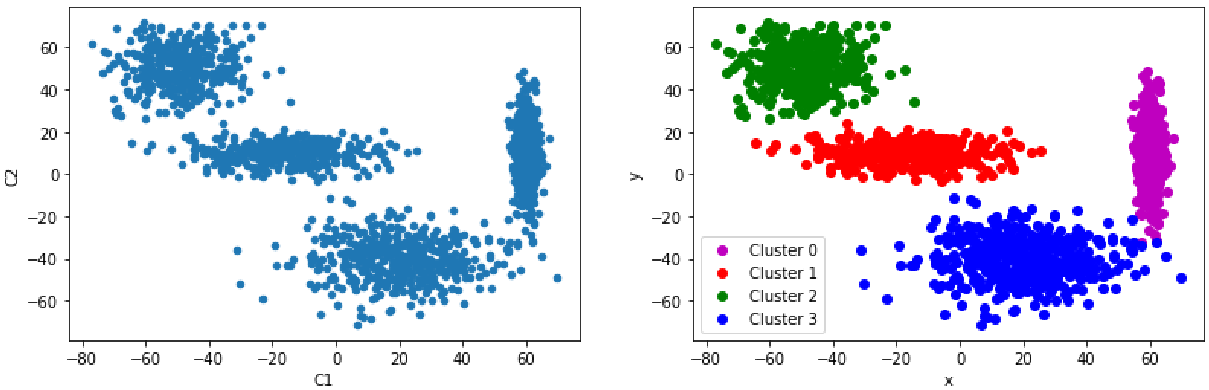
- [ ] Plot the data for K = 2, 4, 6, 7 and note down inertia.

Instruction

- Submit a .ipynb file and a report (report template) .pdf file.
- **You must follow the given algorithm**
- **DO NOT USE LIBRARIES SUCH AS: "Sklearn", "Scikit learning" or "pandas" for this assignment**
- **Use your student id as seed**
- **Copying will result in -100% penalty**
- **Your marks will fully depend on your viva and understanding.**
    - Full Algorithm: 16
    - Plotting: 4

Resources

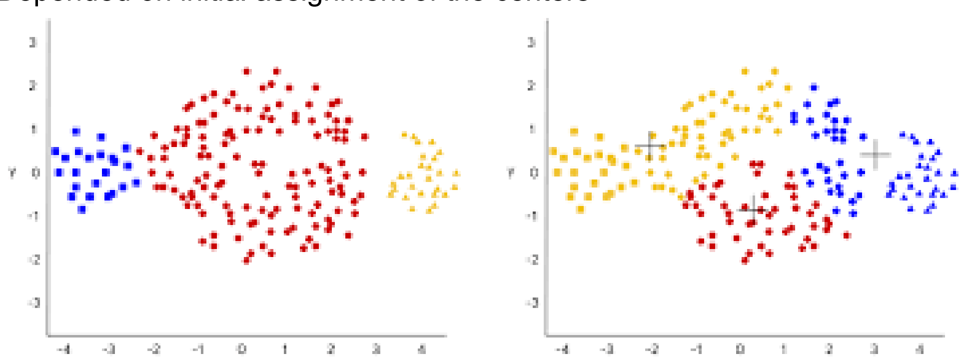
k-means clustering



1. Select K random data points as the centers of K clusters
2. Assign each datapoint to the closest clusters (by calculating the distance from centers).
3. **While True**:
4. Recalculate the center of the clusters (which is the mean of the data points)
5. Reassign each datapoint to the closest cluster
6. **If** no datapoint changes cluster **then**
7. **break**

Limitations:

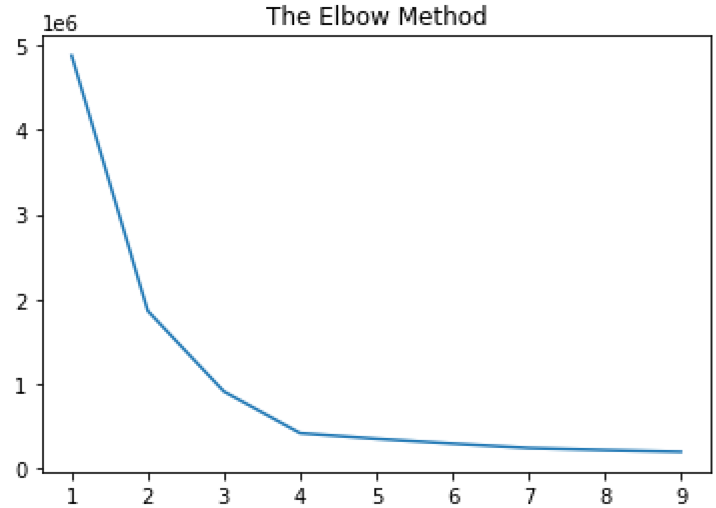
- Need to know K in advance
- Depended on initial assignment of the centers



-

How to choose the K?

- Inertia measures how well a dataset was clustered by K-Means. It is calculated by measuring the distance between each data point and its centroid, squaring this distance, and summing these squares across one cluster. A good model is one with low inertia AND a low number of clusters (K).



-

6. Use K-Means clustering on the following dataset where K=2 and the initial centers are [1, 1] and [5, 5]. **Show the first three (3) iterations. Stop iterating as soon as the clusters converge completely.** **[5]**

x	y	Label
3	4	No
5	6	Yes
5	-1	Yes
6	4	Yes
6	6	No

centers = [(1,1) , (5,5)]

clusters = [0, 0, 0, 0, 0]

(1)

min_dist = ∞

dist = (3,4) → (1,1) = $\sqrt{2^2 + 3^2}$ = $\sqrt{13}$

∴ min_dist = $\sqrt{13}$

∴ clusters [0] = 0

dist = (3, 4) → (5,5) = $\sqrt{2^2 + 1^2}$ = $\sqrt{5}$

∴ min_dist = $\sqrt{5}$

∴ clusters [0] = 1

Similarly,

cluster [1] = 1

cluster [2] = 0

cluster [3] = 1

cluster [4] = 1.

centers = [(5, -1) , (5,5)]

(2)

clusters [0] = 1

cluster [1] = 1

cluster [2] = 0

cluster [3] = 1

cluster [4] = 1

∴ Centers = [(5, -1) , (5,5)]

unchanged in 2nd iteration.

∴ Clustering complete.

2