Assignment 2: K-means clustering algorithm
Total marks: 100
Due date: 29th of Sep, 2023. (Midnight)

You need to implement the K-means algorithm using Python. You are **not allowed** to use any **off-the-shelf K-means function** in Python or any other environment. However, you can use any package such as 'numpy' to create an array, matrix, or any other data type. Also, you might find the following packages useful for this assignment. 'matplotlib' for plotting the data, 'math' for mathematical operations. The assignment has the following components. Please use Google Colab for this. For sharing the code, download all the relevant files on your local machine and upload those on Moodle. **DO NOT** share a link of your colab project.

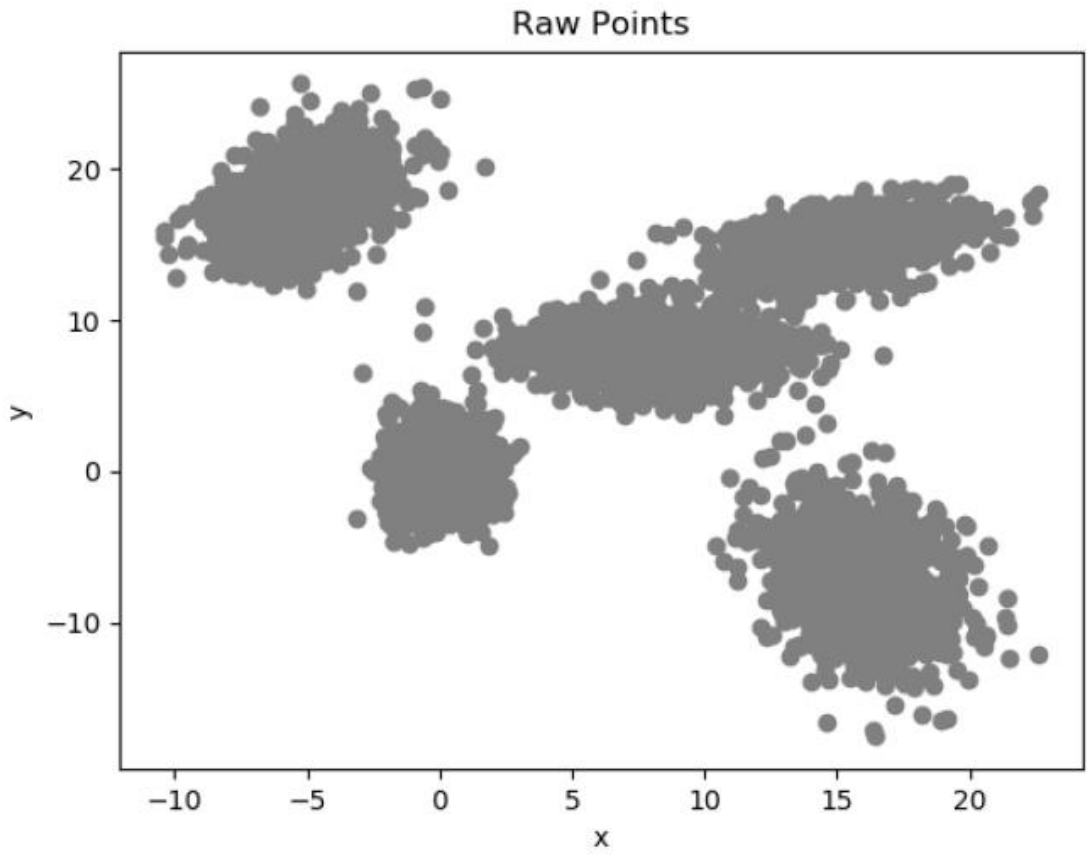
a) Load the data file ('data.txt') provided with the assignment and pass it to your function 'kmeans_fall2022()' along with the parameter defining number of clusters. The function should be able to perform the k-means clustering on the data provided. You need to code the following steps inside the function with proper comments describing each steps. For you convenience, the data is only two dimensional so that you can plot the data and manually decide the number of clusters.

1. Random initialization of the cluster centers
2. Assign points to their nearest centers (use Euclidean distance measure)
3. Re-calculate centers using the points assigned in each of the clusters
4. Compute the steps 2-3 inside a loop until the stopping criteria are reached. As described in the class, the stopping criteria need to be defined by yourselves and it can be a combination of displacement of centroid positions compared to the previous step and the maximum number of iterations you want the function to execute.

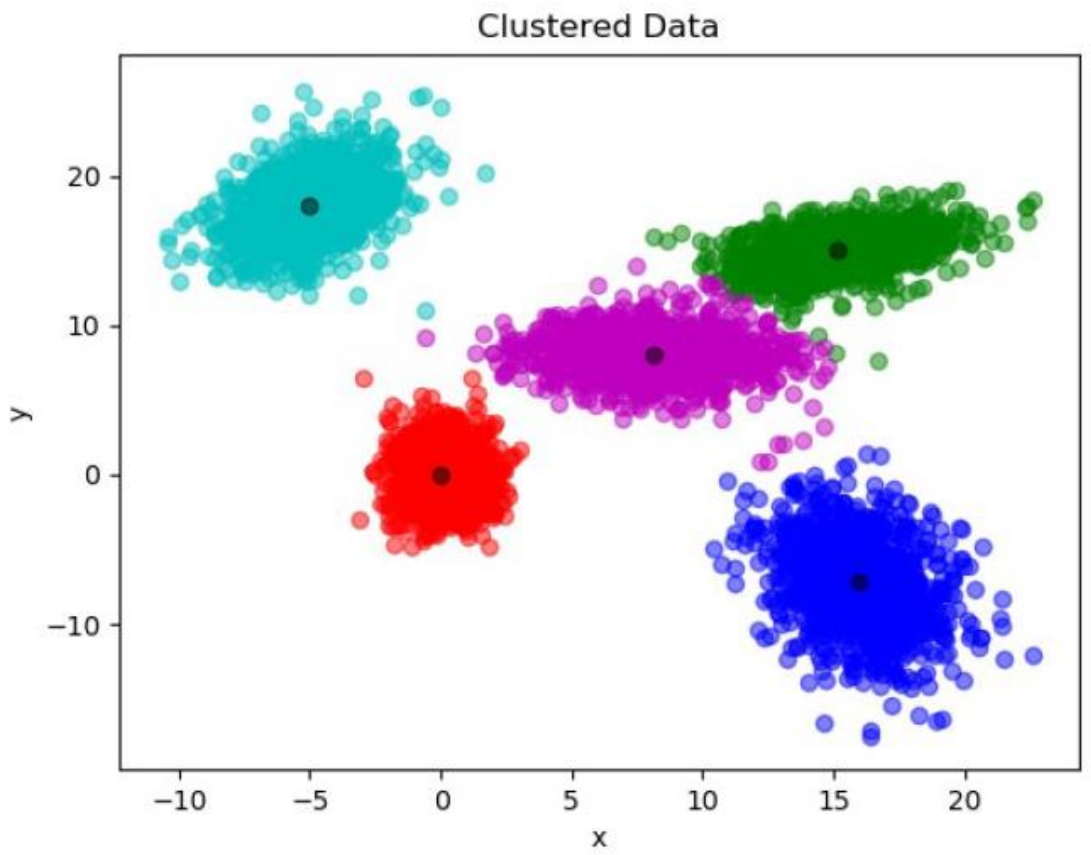
Deliverable:

- A python script that reads data.txt, and call kmeans_fall2021() function. For this go to Google colab editor, file -> download and download both .ipynb and .py formats. Upload both the formats on Moodle.
- kmeans_fall2022() function
- Your code **must plot** the data points supplied in data.txt **before and after clustering**. Assign different colors to display the points in different clusters. Example plots are given in the next page.

Example plots your code should display



Plot before clustering



Plot after clustering

b) One of the drawbacks of the k-means algorithm is the user needs to specify the number of clusters ‘k’. One of the possible approaches to fix this problem is the ‘Elbow method’ that automatically finds the best value for ‘k’ (not covered in the class, find out on your own). Please implement the ‘Elbow method’, and run your ‘kmeans_fall2022()’ function with the ‘Elbow method’ on the data.txt. For this, you need to call ‘kmeans_fall2022()’ in a loop varying the ‘k’ value. For each iteration of the loop, you need to evaluate the ‘optimal k value criteria’. At the end, return the best ‘k’ value and plot the clustered data for the best ‘k’ value. You are welcome to try any other method that finds the best ‘k’ values. Again, remember to **not use an off-the-shelf method** for this.

Deliverable:

- A Python script that implements ‘Elbow method’ which call ‘kmeans_fall2022() function in a loop. Again, download both .ipynb and .py formats of your code. Upload both the formats on Moodle.
- Make sure your code returns (and prints) the best ‘k’ value and plots the clustered data with the best ‘k’ value
- A report describing the elbow method and how you select the best ‘k’ value.

40

Grading Criteria

1. Your code should run without error. If it doesn’t run or part of the code doesn’t run you will loose 30% of marks. For example, if part b) of your code doesn’t run, you can receive a maximum 28 marks (instead of 40) for part b).
2. Late submission: 10% of the awarded marks will be deducted if you are late by one day. 20% for two days. Assignment submission will not be considered if you are more than two days late.