K-Means Clustering

Task Description - Implementing a simple k-means algorithm to take in a text file specifying the data points in D dimensional space along with number of clusters demanded and to write out the centroid of the clusters in an output text file.

Brief explanation of the implementation -
The standard k-means algorithm is implemented in the given python script "k-means.py" without using any pre-defined library such as sklearn's KMeans. The program is capable of taking in a text file that consists of N number of data points in the D dimensional space. In addition, the script also provides the option for a user to evaluate clusters on a randomly generated dummy data instead of specifying an input file. In that case the user is required to specify the dimensional space and the number of points to be generated. The generated dummy data is written out in "dummy_data.txt" for reference.

The final centroids for k number of clusters, as provided by the user, are written in the "output.txt" file.

The first set of k centroids are randomly selected from the data points itself. After that, the following two steps are iterated over till the error (= difference between previous centroids and newly evaluated centroids) converges:

Step 1: Assign data points to the current centroids, i.e, form clusters.

Step 2: Assign the current centroids to old centroids and then evaluate the new set of current centroids by finding the mean of the formed clusters.
Find out the error between the old and current clusters and re-iterate if error != 0.

Running the script -
The following steps need to be followed to correctly run the **python3** script **k-means.py**:

Step 1: In the terminal navigate to the directory where the script is stored (cd <path/to/directory /where/k-means.py/is/placed>).

Step 2: Run the python script by specifying if you want the script to generate dummy data by passing Y or y as an argument. The k-means algorithm will then run on the generated dummy data. Else pass N or n. Ex. python k-means.py N

Step 3: If you passed **Y/y** as the argument, you will be asked to enter the dimension of data points and the number of points that you want to generate. The generated points will be stored in dummy_data.txt for reference.
If you passed **N/n** as the argument, you will be asked to enter the path of the input text file containing the data points.

Step 4: Finally you will be asked to input the desired number of clusters. The centroids will be computed and written in output.txt

[* Note - The python script has been carefully commented and can be referred for any clarification.]