Team member: Yiming Gu, Anamika Choudhary, Shenghua Yue

The dataset (dataframe, data) has 178 observations and 13 features. All features are continuous numeric data without missing values. Since features are in different ranges and Euclidean distance (used in the clustering method) is sensitive to the scaling of each feature, **preprocessing.StandardScaler()** method is implemented to standardize the data. After standardization (dataframe, scaled_df), the mean will be zero and the standard deviation will be one. Moreover, **PCA** is performed to denoise and reduce dimensionality. To achieve a cumulative explained variance ratio of above 90%, first eight new features are selected and used for kmeans calculation (dataframe, principalDf). To define an initial k value, two methods are tested. The first one is to use hierarchical clustering. From dendrogram, k value of 3 would be a good estimate. The second method is to calculate within sum of squares (WSS). As we increase the value of k, the value of WSS goes on decreasing. The location of a bend(knee) in the plot is generally considered as an indicator of the appropriate number of clusters.[4]

Our clustering method is modified based on K-Means. For the K-Means algorithm, **initCent** function is implemented to determine the initial locations of the k centroids [1]. First centroid is selected randomly from the 178 points. From the second centroid, data point is iteratively assigned to the nearest centroid and the farthest point is selected as new centroid. With this improvement, the initial k centroids are spreading out and the distance among them is as far as possible.

kMeans function is deployed to perform clustering algorithm. To improve the time efficiency and avoid redundant distance calculation between points and centroids, triangle inequality is used to accelerate KMeans [2][3]. According to the triangle inequality (the sum of the lengths of any two sides must be greater than or equal to the length of the remaining side), we can get two lemmas:

Lemma 1: Let x be a point and let b and c be centers. If d(b,c) >= 2d(x,b) then d(x,c) >= d(x,b).

Lemma 2: Let x be a point and let b and c be centers. Then d(b,c) >= max{0, d(x,b) - d(b,c)}.

Lemma 1 tells us when 2 times of the distance of a data point (x) and a center (b) is smaller than or equals to the distance of the center (b) and some other center (c), then x is closer to b. When we define the c as the closet center to b, then inequality tells us b is the closest center to x. Lemma 2 is used to update the lower bound which will be introduced more specifically in the jupyter notebook.

With bringing in this two lemmas in the algorithm, we reduced the times to calculate the distances which would potentially accelerate the process (which would be more obvious on large datasets) and the memory is appropriately allocated in the algorithm.

Additionally, there are some help functions. distEclud function is used to calculate the Euclidean distance between two vectors. shortestdis funciton is used to calculate the shortest distance between points and centroids and help to find the closest centroid for each data point. distCent function returns a hashmap, where key is the centroid and value is the distance between this centroid and its closest centroid.

Metric - Within sum of squares (wss): This metric essentially gives us the homogeneity within a group. The metric calculates the sum of squares of the distance of each point in a particular cluster from its respective centroid.

Reference:

[1] https://www.coursera.org/lecture/ml-clustering-and-retrieval/smart-initialization-via-k-means-T9ZaG

[2] Elkan. *Using the triangle inequality to accelerate k-means*. ICML, 2003..

[3] Hamerly. *Making k-means even faster*. SDM, 2010

[4] https://www.datanovia.com/en/lessons/determining-the-optimal-number-of-clusters-3-must-know-methods/