You are required to implement *k*-means algorithm in PySpark. Please be noted that all the data flow should be conducted by RDD or Spark DataFrame. You should have at least the following components:

(a) A data normalization function norm_data(X) that normalizes the training and test data. (10 marks)

(b) A distance calculation function cal_dist(x1, x2) that calculate the distance between x1 and x2 and return the value of the distance. (10 marks)

(c) Centroids initialization function centroid_init(data, k) that selects k initial centroids from the data. (10 marks)

(d) Closest centroid searching function get_closest(x, centroids) that returns the index of the centroid that is closest to the data point x. (10 marks)

(e) Centroid updating function centroid_update(X, index) that returns the updated centroid by the index of the closest centroid of each data point. (10 marks)

(f) The *k*-means main function kmeans(data, k, max_iter, tol) that returns the *k* centroids. (10 marks)

(g) The prediction function predict(X, centroids) to use the trained centroid to predict the data point X and return the predicted labels. (10 marks)

(h) The evaluation function evaluate(predictions, labels) that returns the Silhouette Coefficient of the clustering result. (10 marks)

The function arguments can be different based on your design. Finally, compare your implementation with the *k*-means in MLlib and try to analyze the difference. (20 marks)

There are two bonus questions:

(i) Implement *k*-means++ in centroid initialization. (10 marks)

(j) Implement NMI for evaluation. (10 marks)

The training dataset:

https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass/pendigits

The test dataset:

https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass/pendigits.t