Assignment 4

Due: Nov. 21

Submission Instructions

- **Your program must run on erdos.dsm.fordham.edu**
- **Create a README file, with simple, clear instructions on how to compile and run your code. *If the TA cannot run your program by following the instructions, you will receive 50% of programing score.***
- **Zip all your files (code, README, written answers, etc.) in a zip file named** $\{firstname\}\_\{lastname\}\_CS6930\_HW4.zip$ **and upload it to Blackboard**

1. (40 points) For this question you will be implementing the k-means clustering algorithm and investigating the effects of different starting configurations.

   First you will need to implement the k-means clustering algorithm. You should be able to reuse a lot of code from the previous assignment (data input, normalization, distance measure, etc). We will work with the `segment.arff` dataset distributed with this assignment. This dataset is based on a set of images taken in color around the UMASS campus to which low-level image processing operators were applied. The goal is to find clusters in the data which define different types of objects (buildings, trees, sky etc). But you need not be concerned with understanding the meaning of each cluster.

   You should z-score normalize the data as a preprocessing step before proceeding with the clustering. Again, k is a tuneable parameter and should be abstracted from your core clustering subroutine; you will vary k and observe the effects.

   Random Starting Positions

   k-means is sensitive to the starting positions of the cluster centroids. To try to overcome this, we can run k-means 25 times with randomized starting positions for the cluster centroids. For an actual application you would select the centroids through your own randomization process. For this excercise, we are providing 300 instance numbers to use (counting to start at the first instance in the dataset). To illustrate the approach, consider 5-means. You will need 5 centroid instances for each of 25 trials or a total of 125 indices into the dataset. You will select items 775, 1020, 200, 127, and 329 for the first iteration, then 1626, 1515, 651, 658, 328 for the second and so on. The 300 indices are the following:

[775, 1020, 200, 127, 329, 1626, 1515, 651, 658, 328, 1160, 108, 422, 88, 105, 261, 212, 1941, 1724, 704, 1469, 635, 867, 1187, 445, 222, 1283, 1288, 1766, 1168, 566, 1812, 214, 53, 423, 50, 705, 1284, 1356, 996, 1084, 1956, 254, 711, 1997, 1378, 827, 1875, 424, 1790, 633, 208, 1670, 1517, 1902, 1476, 1716, 1709, 264, 1, 371, 758, 332, 542, 672, 483, 65, 92, 400, 1079, 1281, 145, 1410, 664, 155, 166, 1900, 1134, 1462, 954, 1818, 1679, 832, 1627, 1760, 1330, 913, 234, 1635, 1078, 640, 833, 392, 1425, 610, 1353, 1772, 908, 1964, 1260, 784, 520, 1363, 544, 426, 1146, 987, 612, 1685, 1121, 1740, 287, 1383, 1923, 1665, 19, 1239, 251, 309, 245, 384, 1306, 786, 1814, 7, 1203, 1068, 1493, 859, 233, 1846, 1119, 469, 1869, 609, 385, 1182, 1949, 1622, 719, 643, 1692, 1389, 120, 1034, 805, 266, 339, 826, 530, 1173, 802, 1495, 504, 1241, 427, 1555, 1597, 692, 178, 774, 1623, 1641, 661, 1242, 1757, 553, 1377, 1419, 306, 1838, 211, 356, 541, 1455, 741, 583, 1464, 209, 1615, 475, 1903, 555, 1046, 379, 1938, 417, 1747, 342, 1148, 1697, 1785, 298, 1485, 945, 1097, 207, 857, 1758, 1390, 172, 587, 455, 1690, 1277, 345, 1166, 1367, 1858, 1427, 1434, 953, 1992, 1140, 137, 64, 1448, 991, 1312, 1628, 167, 1042, 1887, 1825, 249, 240, 524, 1098, 311, 337, 220, 1913, 727, 1659, 1321, 130, 1904, 561, 1270, 1250, 613, 152, 1440, 473, 1834, 1387, 1656, 1028, 1106, 829, 1591, 1699, 1674, 947, 77, 468, 997, 611, 1776, 123, 979, 1471, 1300, 1007, 1443, 164, 1881, 1935, 280, 442, 1588, 1033, 79, 1686, 854, 257, 1460, 1380, 495, 1701, 1611, 804, 1609, 975, 1181, 582, 816, 1770, 663, 737, 1810, 523, 1243, 944, 1959, 78, 675, 135, 1381, 1472]

Running k-means entails iteratively move the centroids to the best possible position. For each value of k and for the 25 initial centroid sets, you will run k-means until either the clusters no longer change or your program has conducted 50 iterations over the data set, whichever comes first.

To evaluate the results, compute the sum of squared errors (SSE) for each of the 25 clustering runs. SSE measures the deviation of points from their cluster centroid and gives a simple measure of the cluster compactness:

$$SSE = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - m_j\|^2$$

where the clusters are C_j $(j = 1 \dots k)$, the final centroid for C_j is m_j, the x_i's are all the points assigned to C_j and $\|a - b\|$ is the distance from point a to point b.

Note: Weka users should apply the "SimpleKMeans" algorithm under the clustering algorithms. Vary the k values as $k = 1, 2, \dots, 12$, and make 5 clustering runs for each k value. For each clustering run, set the "seed" parameter as 1, 2, 3, 4, 5 respectively to obtain different starting points. Report the SSE values of each clustering run for each k value and use the results to answer question 1(a) and 1(b).

(a) For each $k = 1, 2, \ldots, 12$ compute the mean SSE, which we denote μ_k and the sample standard deviation of SSE, which we denote σ_k, over all 25 clustering runs for that value of k. Generate a line plot of the mean SSE (μ_k) as a function of k. Include error bars that indicate the 95% confidence interval: ($\mu_k - 2\sigma_k$ to $\mu_k + 2\sigma_k$).

(b) Produce a table containing the 4 columns: k, μ_k, $\mu_k - 2\sigma_k$ and $\mu_k + 2\sigma_k$ for each of the values of $k = 1, 2, \ldots, 12$.

(c) As k increases and approaches the total number of examples N, what value does the SSE approach? What problems does this cause in terms of using SSE to choose an optimal k?

(d) Can you suggest another measure of cluster compactness and separation that might be more useful than SSE?

2. (15 points) Consider the following dataset:

`{ 0, 4, 5, 20, 25, 39, 43, 44 }`

(a) Build a dendrogram for this dataset using the **single-link, bottom-up** approach. Show your work.

(b) Suppose we want the two top level clusters. List the data points in each cluster.

3. (15 points) Given two clusters

$$C_1 = \{(1,1), (2,2), (3,3)\} \qquad C_2 = \{(5,2), (6,2), (7,2), (8,2), (9,2)\}$$

compute the values in (a) - (f). Use the definition for scattering criteria presented in class. Note that tr in the scattering criterion is referring to the trace of the matrix.

(a) The mean vectors m_1 and m_2

(b) The total mean vector m

(c) The scatter matrices S_1 and S_2

(d) The within-cluster scatter matrix S_W

(e) The between-cluster scatter matrix S_B

(f) The scatter criterion $\frac{tr(S_B)}{tr(S_W)}$

4. (15 points) Consider density-based clustering algorithm DBSCAN with parameters $\epsilon = \sqrt{2}, MinPts = 3$, and Euclidean distance measures. Given the following points:

$$(0,0), (1,2), (1,6), (2,3), (3,4), (5,1), (4,2), (5,3), (6,2), (7,4)$$

(a) List the clusters in terms of their points.

(b) What are the density-connected points?

(c) What points (if any) does DBSCAN consider as noise?

5. (15 points) A Naive Bayes classifier gives the predicted probability of each data point belonging to the positive class, sorted in a descending order:

Instance #	True Class Label	Predicted Probability of Positive Class
1	P	0.95
2	N	0.85
3	P	0.78
4	P	0.66
5	N	0.60
6	P	0.55
7	N	0.43
8	N	0.42
9	N	0.41
10	P	0.4

Suppose we use 0.5 as the threshold to assign the predicted class label to each data point, i.e., if the predicted probability ≥ 0.5, the data point is assigned to positive class; otherwise, it is assigned to negative class. Calculate the *Confusion Matrix, Accuracy, Precision, Recall, F1 Score* and $Specificity$ of the classifier.