CSE 881 Homework 2

By Abhiram Durgaraju

Problem 1)

a)

Iter	Cluster assignment of data point								Centroid Locations		
	0.10	0.20	0.80	0.90	1.00	1.30	1.80	1.90	A	B	C
0	-	-	-	-	-	-	-	-	0.00	0.30	2.00
1	A	B	B	B	B	C	C	C	0.10	0.725	1.667
2	A	A	B	B	B	C	C	C	0.15	0.90	1.667
3	A	A	B	B	B	C	C	C	0.15	0.90	1.667

b)

SSE $= (0.1 - 0.15)^2 + (0.2 - 0.15)^2 + (0.8 - 0.9)^2 + (0.9 - 0.9)^2 + (1.0 - 0.9)^2 + (1.3 - 1.667)^2 + (1.8 - 1.667)^2 + (1.9 - 1.667)^2 = 0.0025 + 0.0025 + 0.01 + 0 + 0.01 + 0.1344 + 0.0178 + 0.0544 = \mathbf{0.2316}$

c)

Iter	Cluster assignment of data point								Centroid Locations			
	0.10	0.20	0.80	0.90	1.00	1.30	1.80	1.90	A	B	B1	B2
0	-	-	-	-	-	-	-	-	0.00	2.0	-	-
1	A	A	A	A	B	B	B	B	0.50	1.5	1.0	1.9
2	A	A	A	A	B1	B1	B2	B2	0.5		1.15	1.85

SSE_A $= (0.1 - 0.5)^2 + (0.2 - 0.5)^2 + (0.8 - 0.5)^2 + (0.9 - 0.5)^2 = 0.16 + 0.09 + 0.09 + 0.16 = 0.50$

SSE_B $= (1.0 - 1.5)^2 + (1.30 - 1.5)^2 + (1.80 - 1.5)^2 + (1.90 - 1.5)^2 = 0.25 + 0.04 + 0.09 + 0.16 = 0.54$

Note in the above, I chose to assign 1.00 to cluster B since there was a tie because it had same distance to both centroid A and B. After 1 iteration, the SSE of cluster B is higher, so we must bisect this cluster. We create two clusters B1 and B2. The centroid of B1 is the smallest data point in cluster B. The centroid of B2 is the largest data point in cluster B (they are shown in table).

d)

SSE of k-means clustering in part b) = 0.2316

SSE of bisecting k-means clustering in part c) $= (0.1 - 0.5)^2 + (0.2 - 0.5)^2 + (0.8 - 0.5)^2 + (0.9 - 0.5)^2 + (1.0 - 1.15)^2 + (1.30 - 1.15)^2 + (1.80 - 1.85)^2 + (1.90 - 1.85)^2 = 0.16 + 0.09 + 0.09 + 0.16 + 0.0225 + 0.0225 + 0.0025 + 0.0025 = 0.55$

Since the SSE for k-means (0.2316) < SSE of bisecting k-means (0.55), the original k-means clustering performed better on the data-set.

Problem 2)

Prove: The SSE of a data-set when split into k clusters using k-means is always non-increasing.

Let k = N, where N is the number of the data points in our data set D. In this case, we have N centroids with location specified by each point in x_i in D. So, when k = N, the trivial solution is the optimal solution, and the SSE for this scenario is 0. We know that for k < n, SSE must be greater than 0 since at least 1 data point has finite distance to the centroid. Therefore, it can be seen that as k increases closer and closer to N, the SSE cannot increase and eventually converges to SSE = 0 at k=N.

Likewise, let us start with k=1 cluster. K-means is an iterative algorithm that forms new clusters only based on previous clusters. Therefore, only two cases can occur:

1) SSE stays the same and cluster do not change and the algorithm converges
2) SSE decreases and the algorithm recomputes the centroids

Therefore, SSE must always decrease or stay the same as k increases. This is only true for continuous variables since k-means minimizes the sum of Euclidean distance (SSE) between each point in a cluster to its centroid. The notion of Euclidean distance does not apply to categorical or discrete data.

Problem 3)

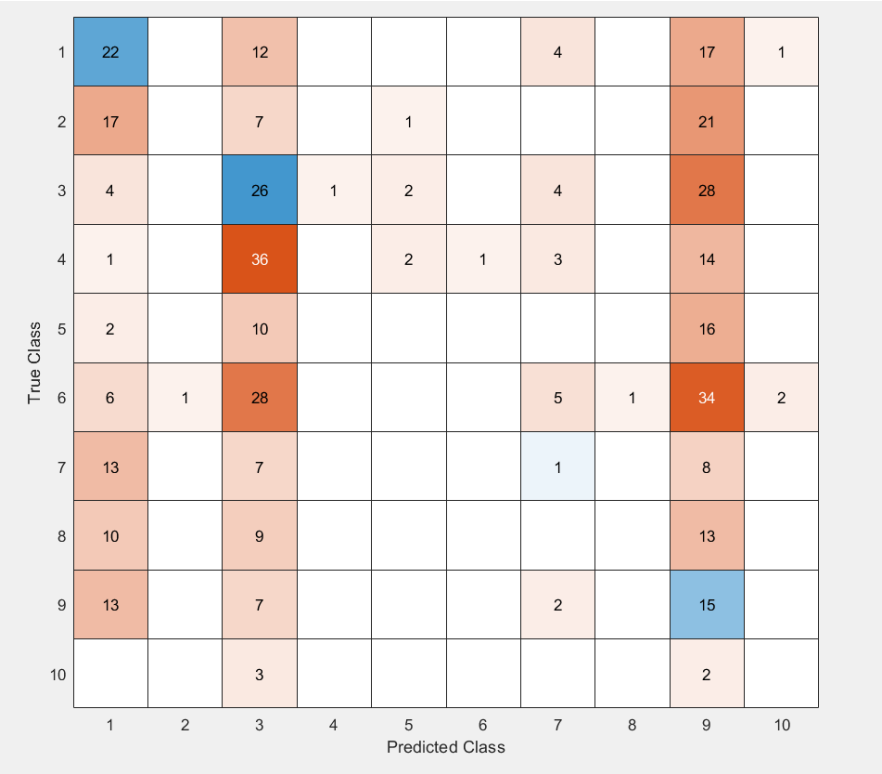
a) I have loaded the prices.txt data into MATLAB
b) I predict that Euclidean distance will be a better metric since we can only guarantee the SSE to monotonically decrease. The SSE is simply the sum of Euclidean distances between pairs of points. The centroids are recomputed based on the new clustering. Using correlation clusters points that are correlated together regardless of their absolute squared distance, and so might not cluster correctly.

c)

```
P = importdata('prices.txt');
rng(1);
[clusters,centroid] = kmeans(P, 10, 'Replicates', 500);
class_labels = importdata('sp500.class');
C = confusionmat(class_labels, clusters);
confusionchart(C);
```

d)

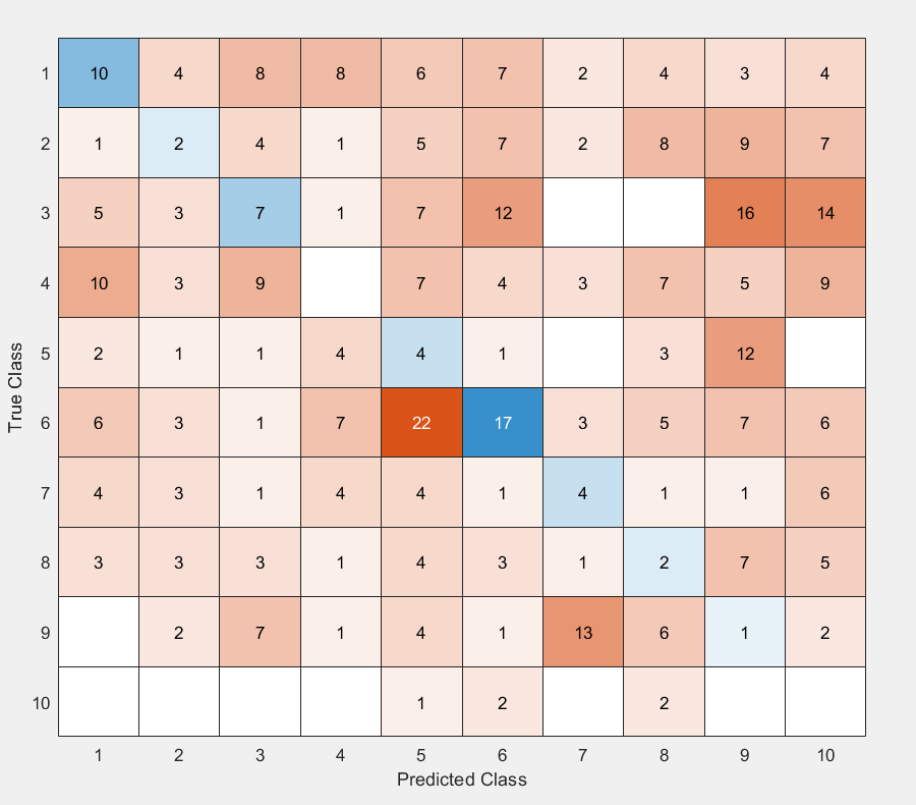
Below is a confusion matrix using Euclidean distance:



e)

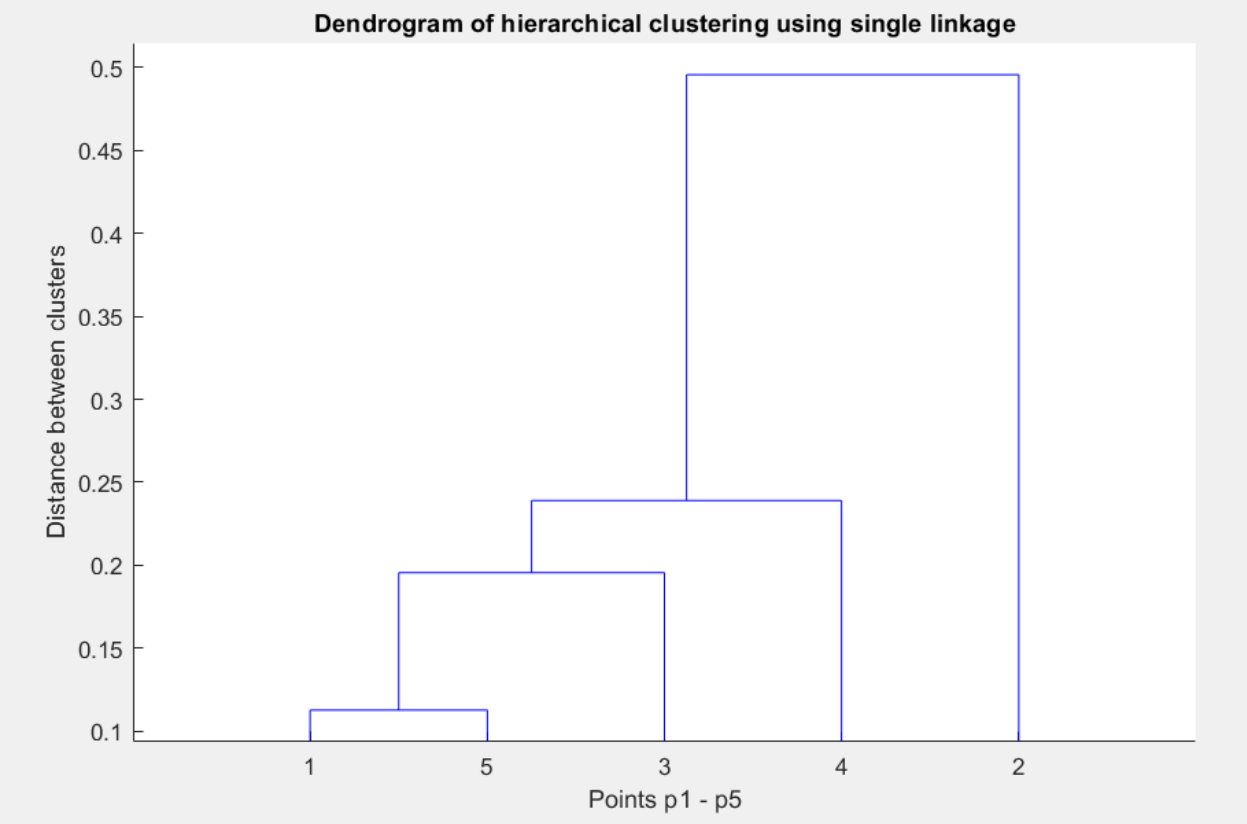
```
P = importdata('prices.txt');
rng(1);
[clusters,centroid] = kmeans(P, 10, 'Distance', 'correlation', 'Replicates', 500);
class_labels = importdata('sp500.class');
C = confusionmat(class_labels, clusters);
confusionchart(C);
```

f) Below is confusion matrix when using correlation as the proximity measure.



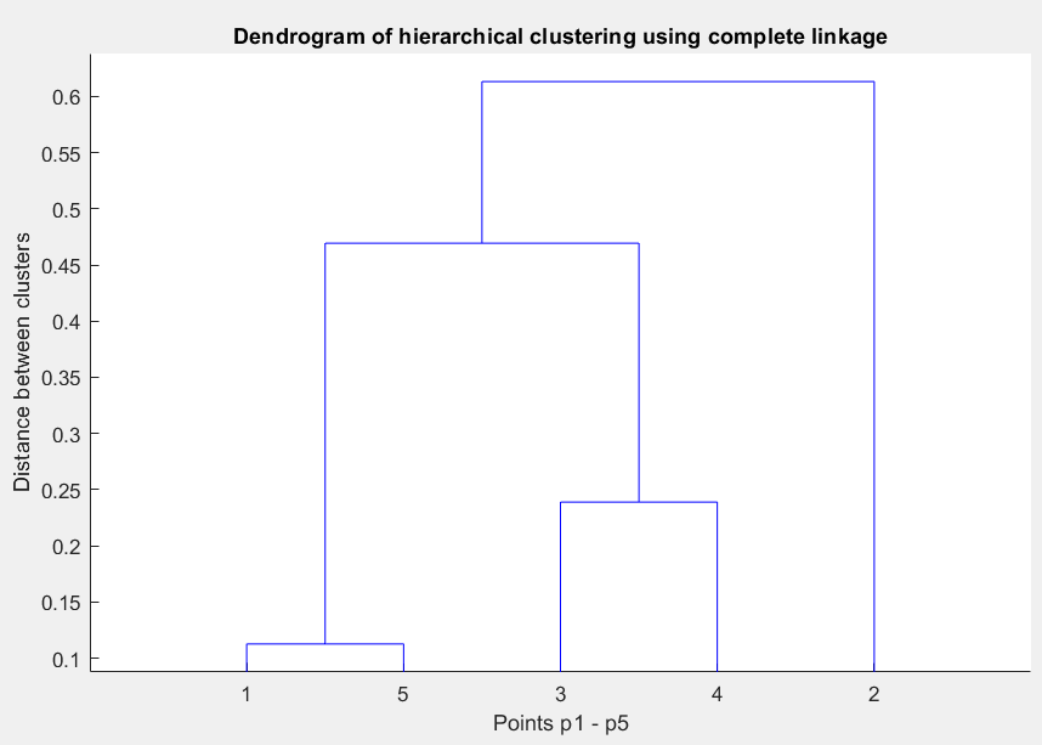
Euclidean distance resulted in a better classification accuracy (14.81%) vs. correlation which only had a classification accuracy of (10.87%). Therefore, Euclidean distances is a better proximity measure for this data set

4)



Below is a table showing the order in which each point was clustered (earliest first). The left two columns show which points were aggregated. The 3rd column shows the distance between the two clusters.

1	5	0.112700000000000
3	6	0.195500000000000
4	7	0.239000000000000
2	8	0.495600000000000



Below is a table showing the order in which each point was clustered (earliest first). The left two columns show which points were aggregated. The 3rd column shows the distance between the two clusters.

1	5	0.112700000000000
3	4	0.239000000000000
6	7	0.469400000000000
2	8	0.613200000000000

```
proximity_matrix = [0, 0.5840, 0.1955, 0.3815, 0.1127; 0.5840, 0, 0.6132, 0.4956, 0.5733; ...
    0.1955 0.6132, 0, 0.2390, 0.3067; 0.3815, 0.4956, 0.2390, 0, 0.4694; 0.1127 0.5733, 0.3067, 0.4694, 0];
v = proximity_matrix(tril(true(size(proximity_matrix)), -1))';
z = linkage(v, 'complete');
dendrogram(z);
xlabel('Points p1 - p5');
ylabel('Distance between clusters');
title('Dendrogram of hierarchical clustering using complete linkage')
```

5)

Eps = 0.15. MinPts = 3. I assume that MinPts includes the point one is trying to evaluate (basically 3 neighbors including itself).

a) Points a through y are core points

b) z is a border point

c) There are no noise points

d) Only 1 cluster can be obtained with the given epsilon and MinPts.

```
X = [0.1, 0.1; 0.2, 0.1; 0.3, 0.1; 0.1, 0.2; 0.2, 0.2; 0.3, 0.2; 0.1, 0.3; 0.2, 0.3; 0.3, 0.3;...
   0.3, 0.4; 0.3, 0.5; 0.3, 0.6; 0.4, 0.6; 0.5, 0.6; 0.6, 0.6; 0.7, 0.6; 0.8, 0.6; 0.9, 0.6; 0.7, ...
   0.7; 0.8, 0.7; 0.9, 0.7; 0.7, 0.8; 0.8, 0.8; 0.9, 0.8; 1, 0.9; 1,1];

[idx, corepts] = dbscan(X, 0.15, 3);
scatter(X(:, 1), X(:, 2));
```

6)

a)

7x7 Laplacian Matrix:

2	-1	-1	0	0	0	0
-1	2	-1	0	0	0	0
-1	-1	3	-1	0	0	0
0	0	-1	2	0	-1	0
0	0	0	0	2	-1	-1
0	0	0	-1	-1	3	-1
0	0	0	0	-1	-1	2

b)

Smallest 3 eigenvalues: -3.7470e-16, 0.2679, 1.5858

c)

Eigenvectors corresponding to the 3 smallest eigenvalues are shown in the three columns below, in order from smallest to largest:

0.3780	-0.4440	-0.2808
0.3780	-0.4440	-0.2808
0.3780	-0.3251	0.1645
0.3780	-0.0000	0.7941
0.3780	0.4440	-0.2808
0.3780	0.3251	0.1645
0.3780	0.4440	-0.2808

d) Using the entire eigenvector matrix, the clusters are as follows:

cluster 1 has points 1,2,3

cluster 2 has point 5 and 7

cluster 3 has points 4 and 6

Note that because k-means algorithm chooses a random point as the initial centroid, the results can vary.

```
G = graph([1 1 2 3 4 6 6 5], [2 3 3 4 6 5 7 7]);
%plot(G);
A = adjacency(G);
F = full(A);
deg = degree(G);
D = diag(deg);
Laplacian_matrix = D - F;
[eigenvectors, eigenvalues] = eig(Laplacian_matrix);

[clusters, centroid] = kmeans(eigenvectors(:,1:3), 3);
```

e) The normalized cut for the above clusters is as follows:

$$V_1 = \{1,2,3\}$$

$$V_2 = \{5,7\}$$

$$V_3 = \{4,6\}$$

$$Ncut(V_1, V_2, V_3) = \sum_{i=1}^{3} \frac{Cut(V_i, V - V_i)}{d(V_i)} =$$

$$\frac{Cut(V_1, V - V_1)}{d(V_1)} + \frac{Cut(V_2, V - V_2)}{d(V_2)} + \frac{Cut(V_3, V - V_3)}{d(V_3)} =$$

$$\frac{1}{3+2+2} + \frac{2}{2+2} + \frac{3}{2+3} = \frac{1}{7} + \frac{1}{2} + \frac{3}{5} = \frac{87}{70} \approx \mathbf{1.242}$$

f) The normalized cut for the supposed given clusters is as follows:

$$V_1 = \{1,2\}$$

$$V_2 = \{3,4,6\}$$

$$V_3 = \{5,7\}$$

$$Ncut(V_1, V_2, V_3) = \sum_{i=1}^{3} \frac{Cut(V_i, V - V_i)}{d(V_i)} =$$

$$\frac{Cut(V_1, V - V_1)}{d(V_1)} + \frac{Cut(V_2, V - V_2)}{d(V_2)} + \frac{Cut(V_3, V - V_3)}{d(V_3)} =$$

$$\frac{2}{2+2} + \frac{4}{3+2+3} + \frac{2}{2+2} = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = \frac{3}{2} = \mathbf{1.5}$$

This normalized cut (f) is larger than the normalized cut found in part (d).