IDA Assignment-3

Vivek Roshan Suggala (M10725902)

1. Perform k-means clustering with this dataset for values of k to be 5, 6, 7, 8, 9 and 10. For each case of k run the clustering algorithm with three different initial cluster centers and select the one with the lowest total SSE value of all clusters in the clustering. Report the following in the submitted work: (Use Matlab kmeans function or any other similar toolbox)

a. Show the cluster centers, SSE values of the clusters, and the total SSE value for the clustering for each value of k.

STEPS FOLLOWED:

- Execute the K-means algorithm for 3 iterations (given in the question) by specifying "3 after 'Replicates' in the code".
- Run the K-means algorithm for different k values (number of clusters) that is k = 5, 6, 7, 8, 9,10.
- The K-means algorithm returns 3 parameters [idx,C,sumd].
  - Idx contains the indexes of the clusters where the data points belong to.
  - C contains the centroids for a given K value (for the minimum SSE within the 3 iterations).
  - sumd contains the SSE values for each cluster.
- Hence, sum(sumd) gives us the total SSE for a given k value.
- C contains the centroid values for the given **K.**
- 'final' represents to display the results of the final iteration.
- 'Replicates' represents the number of time to repeat the clustering using the new initial centroids. The name-value pair we specified, gives us the minimum SSE's obtained in all of the three iterations performed.

MATLAB code:

```
data = xlsread('C:\Users\suggalvn\Downloads\StudentData2.xlsx');
d = data(1:57,2:5);
minimumSSE = 0;
for k = 1:6
    fprintf('\nK value: %d\n',(k + 4));
    [idx,C,sumd] = kmeans(d,(k+4),'Replicates',3,'Display','final');
    fprintf('Cluster Centers are:\n');
    disp(C);
    fprintf('SSE values for each clusters are\n');
    disp(sumd)
    sse(1,k) = sum(sumd);
    fprintf('Sum of SSE values are:\n');
    disp(sse(1,k));
```

Results Obtained:

```
K value: 5
Replicate 1, 6 iterations, total sum of distances = 28211.
Replicate 2, 5 iterations, total sum of distances = 27463.7.
Replicate 3, 3 iterations, total sum of distances = 30016.5.
Best total sum of distances = 27463.7
Cluster Centers are:
   90.2857   94.1429   53.5714   58.8571
   56.1250   90.5000   56.2500   80.8750
   91.0000   85.6250   78.9375   73.5000
```

```
   40.0833   60.5000   86.8333   86.5833
   56.0714   43.2857   55.3571   64.5714

SSE values for each clusters are
   1.0e+04 *

    0.1079
    0.4321
    0.5345
    0.4998
    1.1720

Sum of SSE values are:
   2.7464e+04

mean of the silhoutte for the 5 is 4.876344e-01

K value: 6
Replicate 1, 3 iterations, total sum of distances = 26421.
Replicate 2, 4 iterations, total sum of distances = 25442.7.
Replicate 3, 3 iterations, total sum of distances = 21387.
Best total sum of distances = 21387
Cluster Centers are:
   56.1250   90.5000   56.2500   80.8750
   50.5714   36.8571   38.5714   62.5714
   93.3077   90.5385   79.0769   76.2308
   66.4545   52.2727   75.2727   67.3636
   38.5455   62.9091   86.7273   86.2727
   90.2857   94.1429   53.5714   58.8571

SSE values for each clusters are
   1.0e+03 *

    4.3213
    3.3640
    2.7412
    5.9756
    3.9060
    1.0789

Sum of SSE values are:
   2.1387e+04

mean of the silhoutte for the 6 is 5.177656e-01

K value: 7
Replicate 1, 3 iterations, total sum of distances = 19668.4.
Replicate 2, 3 iterations, total sum of distances = 19346.2.
Replicate 3, 5 iterations, total sum of distances = 18691.1.
Best total sum of distances = 18691.1
Cluster Centers are:
   90.2857   94.1429   53.5714   58.8571
   76.5000   60.6250   77.3750   64.3750
   31.0000   70.3333   81.8333   90.3333
   47.9091   48.2727   42.3636   64.4545
   63.0000   91.8333   63.1667   87.0000
   94.6667   91.5833   79.0000   77.4167
   47.4286   48.2857   89.5714   78.4286

SSE values for each clusters are
   1.0e+03 *

    1.0789
    1.1456
    0.4475
    9.9364
    0.6937
    2.0625
    3.3266
```

```
Sum of SSE values are:
   1.8691e+04

mean of the silhoutte for the 7 is 5.681668e-01

K value: 8
Replicate 1, 2 iterations, total sum of distances = 19126.
Replicate 2, 7 iterations, total sum of distances = 15234.9.
Replicate 3, 3 iterations, total sum of distances = 15997.3.
Best total sum of distances = 15234.9
Cluster Centers are:
   36.6667   43.3333   81.0000   58.6667
   56.1250   90.5000   56.2500   80.8750
   76.5000   60.6250   77.3750   64.3750
   94.6667   91.5833   79.0000   77.4167
   90.2857   94.1429   53.5714   58.8571
   50.5714   36.8571   38.5714   62.5714
   31.7143   68.2857   82.4286   89.5714
   57.6000   50.4000   84.6000   86.8000

SSE values for each clusters are
   1.0e+03 *

    1.0400
    4.3213
    1.1456
    2.0625
    1.0789
    3.3640
    0.6843
    1.5384

Sum of SSE values are:
   1.5235e+04

mean of the silhoutte for the 8 is 5.955032e-01

K value: 9
Replicate 1, 3 iterations, total sum of distances = 20727.1.
Replicate 2, 4 iterations, total sum of distances = 14428.9.
Replicate 3, 3 iterations, total sum of distances = 14276.7.
Best total sum of distances = 14276.7
Cluster Centers are:
   31.3750   64.6250   84.5000   86.7500
   90.8750   86.8750   74.0000   72.6250
   63.0000   91.8333   63.1667   87.0000
   90.2857   94.1429   53.5714   58.8571
   57.5000   52.2500   91.5000   86.2500
   68.4444   55.3333   75.8889   62.4444
   35.5000   86.5000   35.5000   62.5000
   51.5000   37.6250   40.8750   65.8750
   97.2000   96.4000   87.2000   82.0000

SSE values for each clusters are
   1.0e+03 *

    2.1273
    0.9656
    0.6937
    1.0789
    0.5115
    3.5773
    0.4020
    4.3536
    0.5668

Sum of SSE values are:
   1.4277e+04
```

```
mean of the silhoutte for the 9 is 5.186110e-01

K value: 10
Replicate 1, 3 iterations, total sum of distances = 10580.8.
Replicate 2, 2 iterations, total sum of distances = 14907.2.
Replicate 3, 3 iterations, total sum of distances = 13703.4.
Best total sum of distances = 10580.8
Cluster Centers are:
   40.5000   45.5000   72.0000   54.5000
   76.5000   60.6250   77.3750   64.3750
   94.6667   91.5833   79.0000   77.4167
   74.0000   37.5000   42.5000   79.0000
   63.0000   91.8333   63.1667   87.0000
   90.2857   94.1429   53.5714   58.8571
   35.5000   86.5000   35.5000   62.5000
   31.7143   68.2857   82.4286   89.5714
   51.8000   49.6000   93.0000   82.4000
   44.0000   37.6667   40.3333   61.5000

SSE values for each clusters are
   1.0e+03 *

    0.3335
    1.1456
    2.0625
    1.1930
    0.6937
    1.0789
    0.4020
    0.6843
    1.6432
    1.3442

Sum of SSE values are:
   1.0581e+04

mean of the silhoutte for the 10 is 6.317284e-01
```

b. Plot the total SSE value against the values of k.

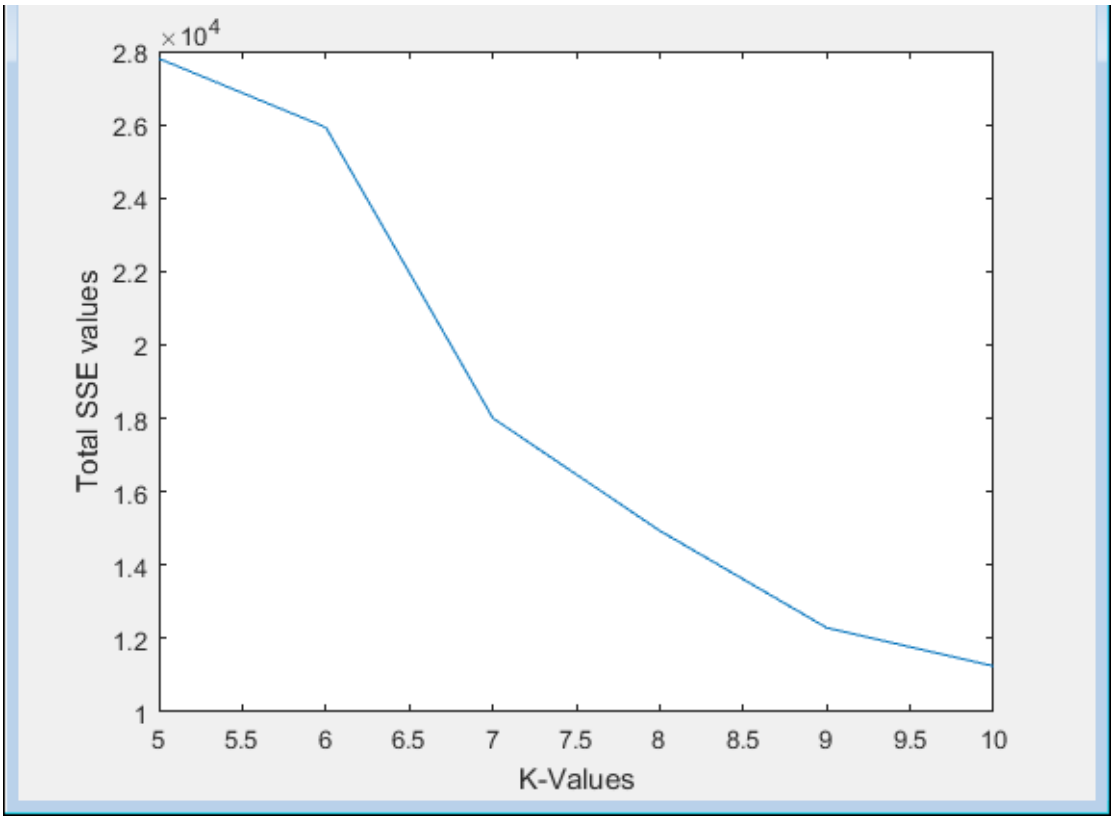
STEPS FOLLOWED:

- Plot the graph for k-values on x-axis and total SSE values on y-axis.
- The x axis is the k values and the y axis is the total SSE values.
- Graph is plotted in order to get an insight of how to select a best k.
- Best K is chosen in such a way that it has a lower SSE, higher silhouette mean and all the data points in the cluster cross the silhouette mean for the particular k (k-10)

MATLAB CODE:

```
kx = 5:10;
plot(kx,sse);
xlabel('K-Values')
ylabel('Total SSE values')
```

Results Obtained:



c. Show a plot of the silhouette coefficients for the data points in any two of the clusterings. (Each value of k results in one clustering)

STEPS FOLLOWED:

- Plot silhouette coefficients (vs) K values on a plot.
- The silhouette coefficients of the data points are returned in a vector, which we use to calculate the mean of the particular k.
- The parameters to be passed to the silhouette function are: data (d) and idx's (indexes of the clusters).
- I have plotted a graph representing silhouette coefficients mean and the k values, in order to choose the best k.
- In the code:

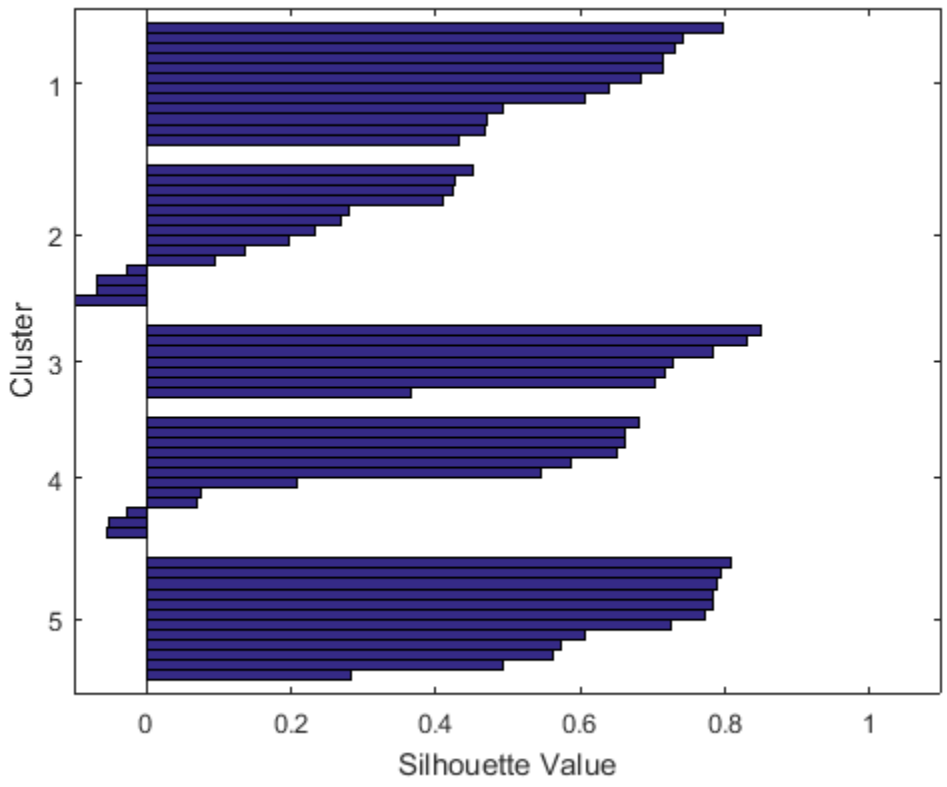
- Comparing the minimum SSE's within different k values chosen and making note of SSE, K value, centroid for the k which has a minimum SSE.

MATLAB CODE:

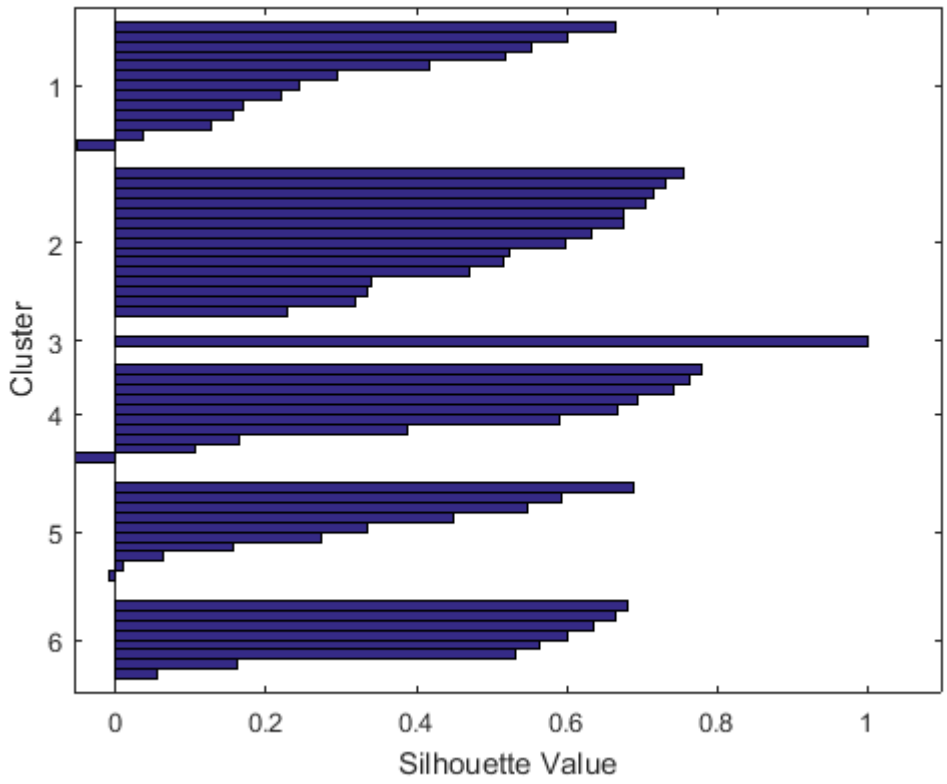
```
%1.b
    [s,n] = silhouette(d,idx);
    figure;
    fprintf('mean of the silhoutte for the %d is %d\n',k+4,sum(s)/57)
    smean(k) = sum(s)/57;
    sarray(k,1) = sse(1,k);
    sarray(k,2) = k+4;
    if (sse(1,k) <= minimumSSE) ||(minimumSSE == 0)
        minimumSSE = sse(1,k);
        minimumCentroid = C;
        minimumCluster = idx;
        minimumk = k+4;
    end
end
```

Results Obtained:

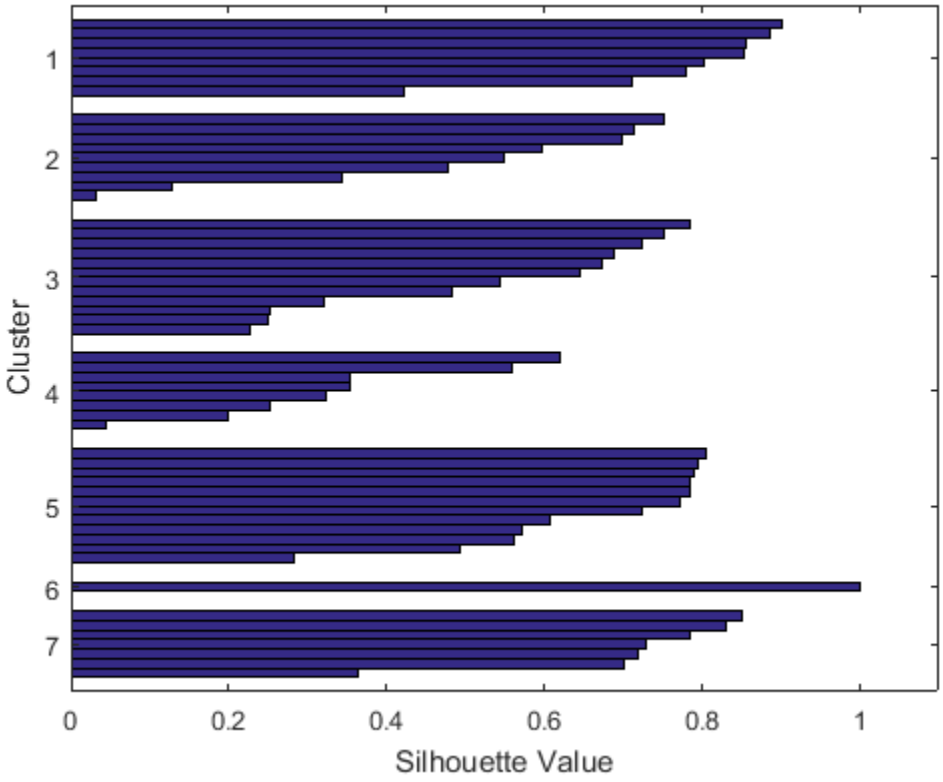
K=5:



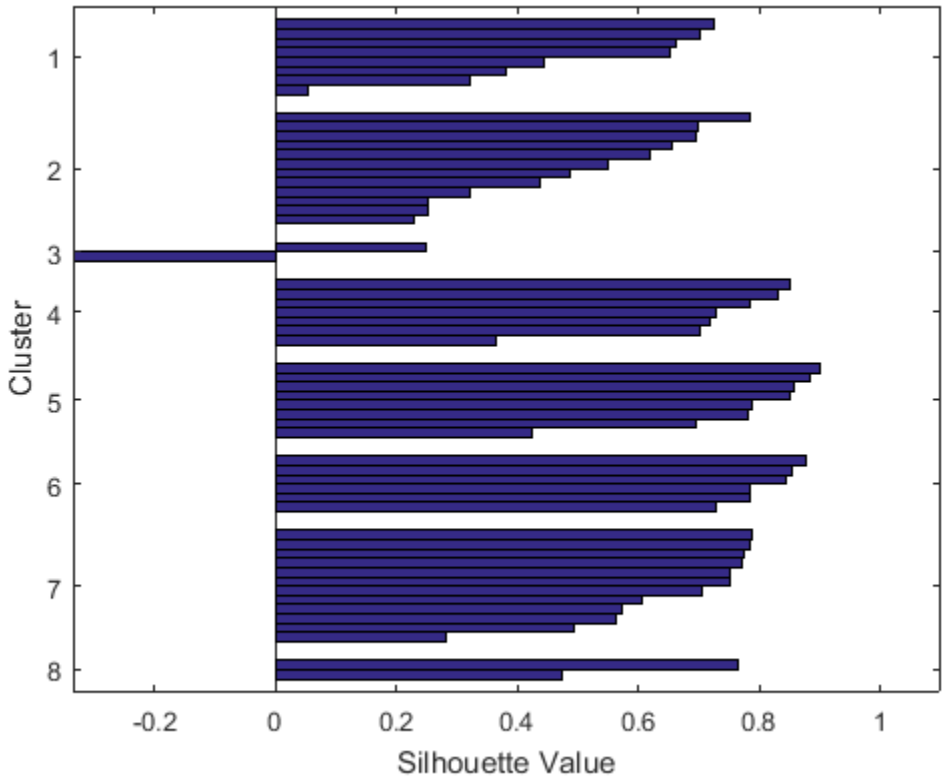
K=6


Cluster
1
2
3
4
5
6
0
0.2
0.4
0.6
0.8
1
Silhouette Value

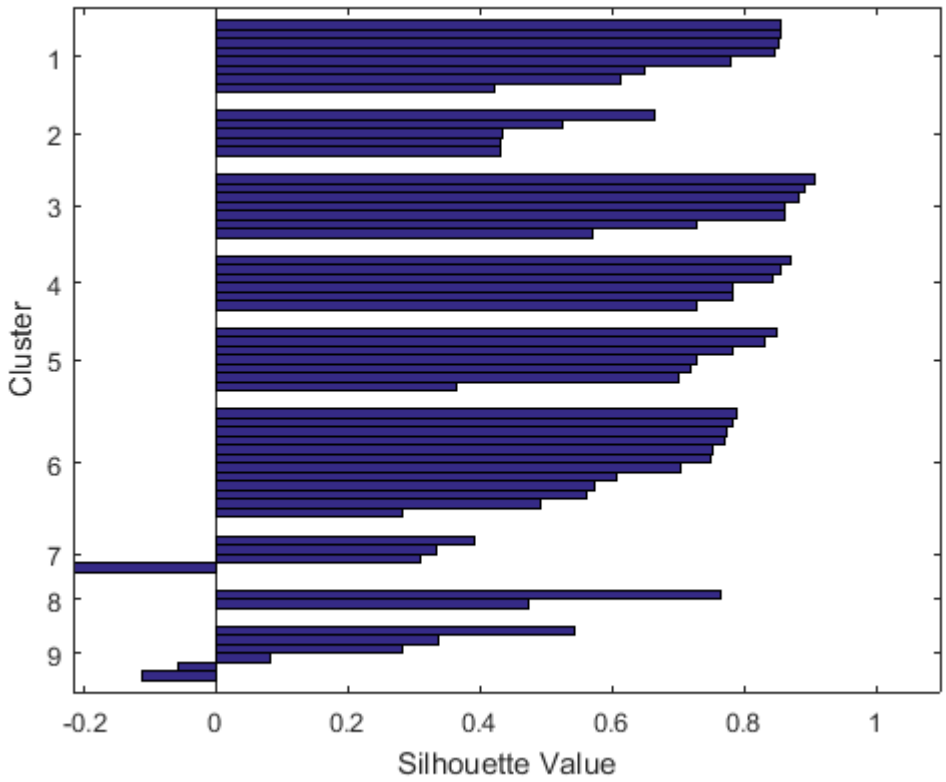
K=7


Cluster
1
2
3
4
5
6
7
0
0.2
0.4
0.6
0.8
1
Silhouette Value

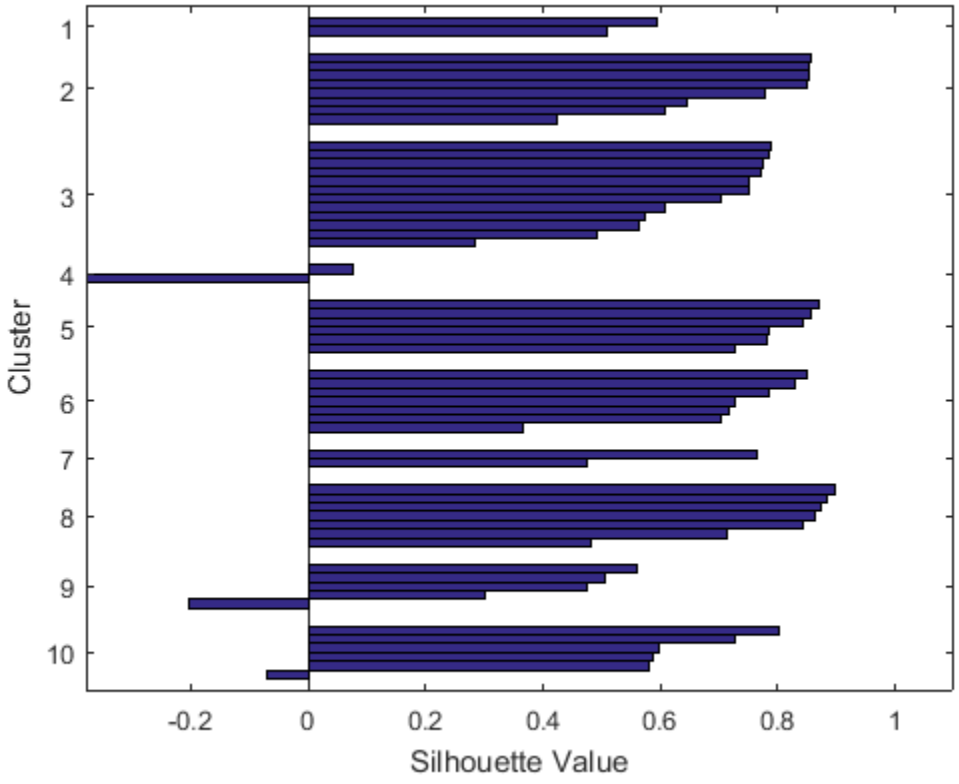
K=8


Cluster
1
2
3
4
5
6
7
8
-0.2
0
0.2
0.4
0.6
0.8
1
Silhouette Value

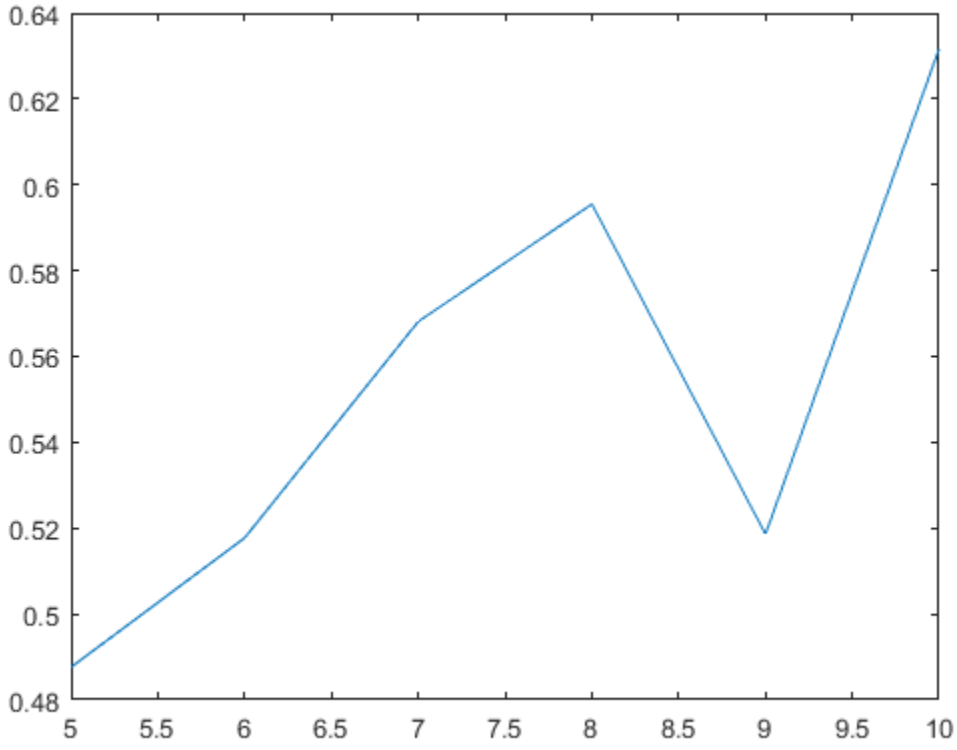
K=9


Cluster
1
2
3
4
5
6
7
8
9
-0.2
0
0.2
0.4
0.6
0.8
1
Silhouette Value

K=10



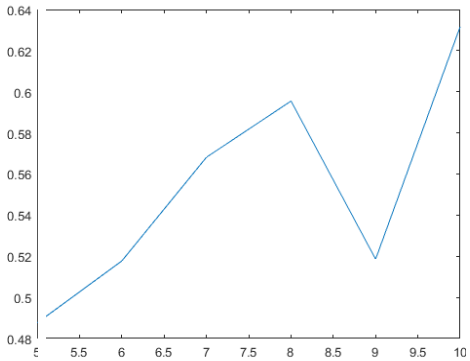
When the mean of silhoutte values against k values are plotted; the resultant graph is:

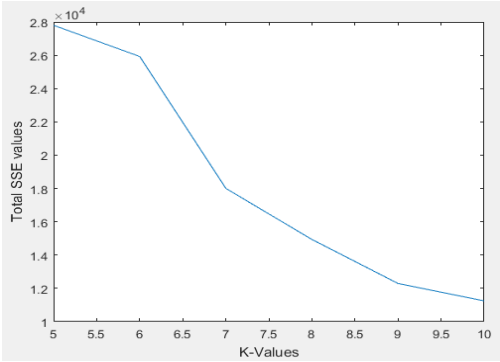


d. How many clusters would you form in this dataset? Justify your answer. For your choice of the best number of clusters, report the centroids of all the clusters and their SSE values

Solution:

- According to K-means algorithm, it is considered to be a best cluster, when the SSE value corresponding to the specific K values is minimum.
- When the algorithm is made to iterate n times, it tries to adjust the cluster centers and compute the new SSE and try to reduce the SSE value for the best cluster.
- But the problem is that, whenever the number of clusters increases for a data set, SSE value decreases gradually, hence we can say that they are inversely proportional to each other.
- But if we only consider SSE value to find out the best cluster, the algorithm makes us to divide the points till there are n clusters for n points.
- Hence we compute silhouette coefficients for the clusters in order to find out the best K value.
- For the given data, when SSE's for different k's are computed, the lowest SSE is found to be at k=10 with an SSE value of `1.0581e+04` and the mean of *silhouette coefficients* for k=10 is ***0.6317***.





- From the above two graphs obtained, we can say that k = 10 to be the best cluster.
- The first graph is the k's (vs) mean-silhouette coefficients. The mean of the silhouette coefficients for the different K's are:

```
>> smean

smean =

    0.4876    0.5178    0.5682    0.5955    0.5186    0.6317
```

- Hence with a high silhouette value and a low SSE value makes us to choose K to be the best value for the given data set.

RESULTS OBTAINED:

```
K value: 10
Replicate 1, 3 iterations, total sum of distances = 10580.8.
Replicate 2, 2 iterations, total sum of distances = 14907.2.
Replicate 3, 3 iterations, total sum of distances = 13703.4.
Best total sum of distances = 10580.8
Cluster Centers are:
   40.5000   45.5000   72.0000   54.5000
   76.5000   60.6250   77.3750   64.3750
   94.6667   91.5833   79.0000   77.4167
   74.0000   37.5000   42.5000   79.0000
   63.0000   91.8333   63.1667   87.0000
   90.2857   94.1429   53.5714   58.8571
   35.5000   86.5000   35.5000   62.5000
   31.7143   68.2857   82.4286   89.5714
```

```
   51.8000   49.6000   93.0000   82.4000
   44.0000   37.6667   40.3333   61.5000

SSE values for each clusters are
   1.0e+03 *

    0.3335
    1.1456
    2.0625
    1.1930
    0.6937
    1.0789
    0.4020
    0.6843
    1.6432
    1.3442

Sum of SSE values are:
   1.0581e+04

mean of the silhoutte for the 10 is 6.317284e-01
```

e. Generate 100 random 4-dimensional random data points such that each attribute can take values between 0 and 100. With this dataset form the same number of clusters as selected by you in (d) above. Report the centroids and populations of the clusters. Compare the total SSE for this random dataset with the SSE for the clustering of the provided dataset. Compare and comment on the differences between the two total SSE values.

STEPS FOLLOWED:

- Generate random data for 100 data points each below 100 for four columns as specified in the question(100*4 vector) .
- Run k-means algorithm algorithm for 3 iterations for a selected k value.
- K value is found to be taken as 10 (chosen k value), as it contains the minimum SSE value for k=10.
- Centroids for the clusters are present in C1, SSE values for each cluster in sumd1 and total SSE for the cluster can be found in sum(sumd1).
- Find out the compostion within the clusters using the functions unique, hist fuctions as specified in the code.
- We say that "Clustering on given data is better compared to random data as SSE is low".
- The random data is more sparsly distributed along the axis and hence it doesn't form good and sharp clusters when compared to the dataset given.
- The dataset given does forms good and sharp clusters and hence it contains a less SSE value.
- When both the SSE's are compared, SSE of random data is *9.077712e+04* and the SSE for the given data is 1.125093e+04. This clearly shows that, the random data is sparsely/randomly distributed and the given dataset is forming crisp and good clusters.

MATLAB CODE:

```
%1.e
fprintf('Random Data\n')
d1 = randi([0 100],100,4);
disp(d1);
[idx1,C1,sumd1] = kmeans(d1,minimumk,'Replicates',3,'Display','final');
```

```
sse1 = sum(sumd1);
fprintf('Centroids for the random data generated');
disp(C1);
[a,b]=hist(idx1,unique(idx1));
fprintf('Population of clusters formed in this random data from clusters 1-%d
clusters\n',minimumk);
disp(a);

%comparison
fprintf('SSE of random data for %d clusters: %d\n',minimumk,sse1);
fprintf('SSE of given data for %d clusters: %d\n',minimumk,sse(1,(minimumk - 4)));
if(sse1 <= sse(1,(minimumk - 4)))
    disp('Clustering on random data is better compared to given data as SSE is low');
else
    disp('Clustering on given data is better compared to random data as SSE is low');
end
```

RESULTS OBTAINED:

```
Random Data

    90    67    11    10
    77    94    61    96
    20    49    28    42
     4    52    51    56
    15    79    54     9
    72    57     5    48
    12    77    44    15
    22    92    22    31
    45    21    92    13
    46    61    69    34
     9    47    11    66
    51    62    36    52
     7    20    30    80
    16    68    30    58
    75     1    56     6
    33    22    80    25
    30    92    47    91
    82    77    79    89
    99    53    42    99
    97     6    74    32
    58    20    36    99
    23     1    95    62
    29    97    75    65
    68    84     1    60
    57    44    76    80
    20   100    92    61
    96    80    35    38
    33    49     7    17
    15    13    86     6
    33    21     9    15
    11    83    55    38
    77    32    92     1
    80    74    52    97
    59    16     5    32
     7    73    44    96
    19     5    69     2
    78    72    26    40
    35    12    90    28
    84    99    46    36
    13    59    23    78
    70    27    19    45
```

```
    19    43    85     5
    40    93    74     4
    34    51    66    53
    93     6    10    77
    84    66    84    62
    63    45    45    94
    87    14    65    78
    21    78    91    76
     6    53    11    32
    68     8    60    52
    54    90     1    65
    52    87     9    51
    25     5    91    66
    86    64    55    23
    83    89    65    99
     1    35    23     8
    47    67    62    41
    95    45     7    99
    46    45    65    49
    85     1    17    61
    54    87    62     1
    43    94    98    77
     2    48    67    74
    50    57    33    21
    88   100    46    40
    91    40     2    26
    28    22    52     7
    13    87    99    22
     0    69    73    48
    32    17    40    52
    53     0    59    84
    83    17    57    41
    68    20    51    40
    92    91    14    96
    76    71    29    33
    17    13    50    13
    97    42    14    70
    35    66    41    93
    60    32    59    49
    73    60    81    51
    87    21    47     0
    33    19    68    19
    46    21    76    78
    16    72    18    61
    47    25    62    80
    46    27    77    84
    12    48    42    69
    64    22    44     8
    90    35    49    81
    43    64    21    13
    17    75    84    57
    29    49    50    58
     5    62    10     1
     6    32    36    94
     2    30    82    62
    55    14    12    67
    56    85    65    42
    30    29    74    70
    76    91    71    47
Replicate 1, 5 iterations, total sum of distances = 90777.1.
Replicate 2, 5 iterations, total sum of distances = 96283.1.
Replicate 3, 8 iterations, total sum of distances = 92317.5.
Best total sum of distances = 90777.1
Centroids for the random data generated   77.1111   18.8889   39.4444   25.5556
   77.4444   81.3333   60.5556   37.7778
   81.8571   23.2857   20.7143   79.1429
   14.3077   55.8462   37.6923   74.9231
   44.4286   21.2857   70.0714   67.5714
   26.7500   79.5833   78.1667   48.0000
```

```
   21.8182   58.0000   23.8182   18.5455
   82.2857   74.7143   51.1429   95.7143
   32.1000   20.2000   76.4000   11.9000
   67.6250   73.7500   14.7500   44.8750

Population of clusters formed in this random data from clusters 1-10 clusters
     9     9     7    13    14    12    11     7    10     8

SSE of random data for 10 clusters: 9.077712e+04
SSE of given data for 10 clusters: 1.125093e+04
Clustering on given data is better compared to random data as SSE is low
```

2. Perform hierarchical clustering for the students' scores dataset. Generate and show dendrograms for the cases (i) Single-Linkage clustering (Clustering-2), and (ii) Complete-Linkage clustering (Clustering-3). Use Euclidean distance for computing distance between data points. Report the following in the submitted work: (Use Matlab functions pdist and linkage, or any other similar toolbox.) Make sure the dendrogram shows all points at its lowest level.

a. Dendrograms for the two clustering's (Clustering-2 and Clustering-3)

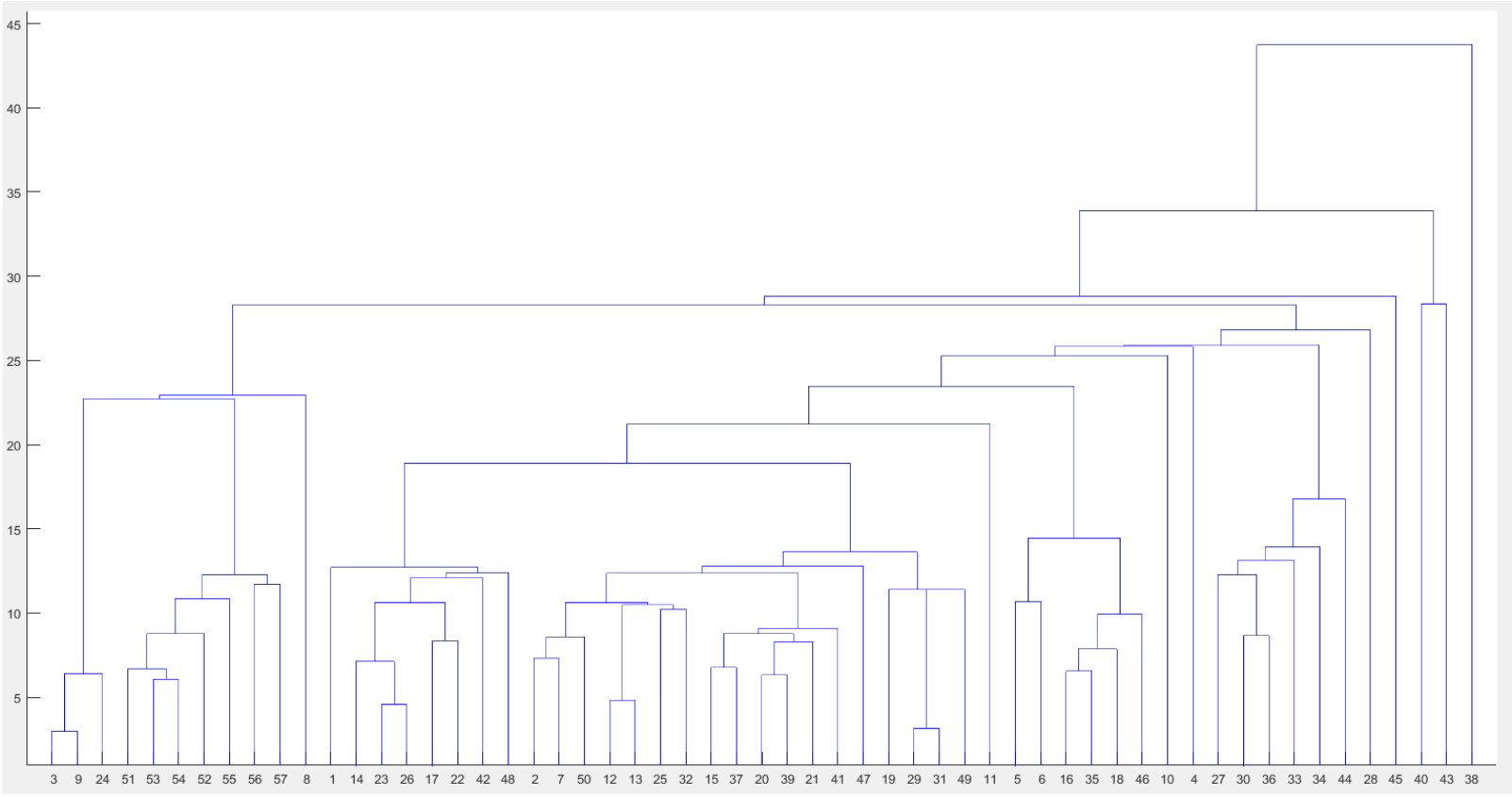
STEPS FOLLOWED:

- Single Linkage clustering is also called as min-clustering where we consider the minimum distances between the points to for the pdist matrix.
- Complete linkage clustering is also called as max-clustering where we consider the maximum distances between the points for forming the pdist matrix.
- The result of the 'pdist' values of the data (57*4) are to be passes to the function linkage.
- The result of the above function is passed to the '*dendrogram*' function and given the P value of the dendrogram function as 0.
- P value is kept 0 in order to print all the data points as leaf nodes. If not specified, dendrogram function merges the points and displays only 30 points as leaf nodes.

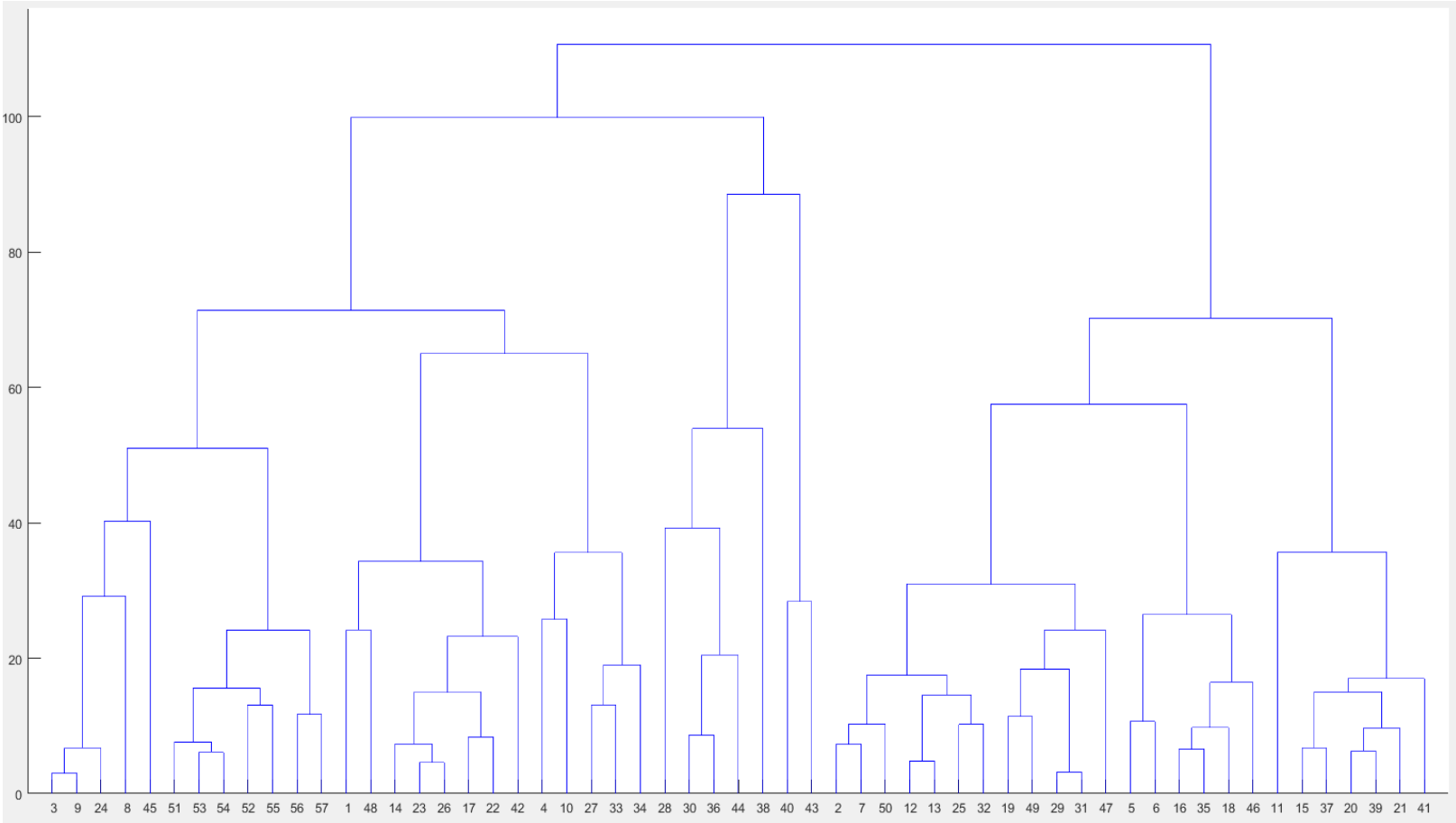
MATLAB CODE:

```
%2nd solution
%2.1
clustersRequired = 4;%given in the question
y=pdist(d);
Clustering2 = linkage(y,'single');
figure()
dendrogram(Clustering2,0); %P = 0
Clustering3 = linkage(y,'complete');
figure()
dendrogram(Clustering3,0); %P=0
```

RESULTS OBTAINED: Single Link Clustering Dendrogram:



Complete Linkage Clustering Dendrogram:



b. Cluster compositions for each case when we need only four clusters. Write ALL the data points included in each cluster and compute their centroids.

STEPS FOLLOWED:

- The clustering data should be passed as an argument for the function "*cluster*" with an argument "*maxclust*" and the number of clusters needed.
- According to the number of clusters needed, the cut on the dendrogram need to be performed (4).
- Once the Dendrograms are created and the cut, where dendrogram is made, we compute the composition of each clusters.
- Once the compositions of the clusters are identified, the centroid for the clusters can be calculated by calculating their means.

MATLAB CODE:

```
%2.2
fprintf('\nSingle linkage Points\n');
clustersDividedSingle = cluster(Clustering2,'maxclust',clustersRequired);
%ClustersPopulation(clustersRequired,rows, clustersDivided);

%population and centroids for single link
for i = 1:clustersRequired
    x = find(clustersDividedSingle == i);
    g=sprintf('%d ', x);
    fprintf('Points belonging to cluster %d: %s\n',i,g)
    %for centroids
    counter = 0;
    sum = [0 0 0 0];
    for j = 1:size(x)
        sum = sum + d(x(j),:);
        counter = counter + 1;
    end
    sum = sum / counter;
    centroidSingle(i,:) = sum;
    %disp(sum);
end

fprintf('Centroid for the Single link is \n');
disp(centroidSingle);

fprintf('\nComplete linkage Points\n');
clustersDividedComplete = cluster(Clustering3,'maxclust',clustersRequired);
%ClustersPopulation(clustersRequired,rows, clustersDivided);

%population and centroids for complete link
for i = 1:clustersRequired
    x = find(clustersDividedComplete == i);
    g=sprintf('%d ', x);
    fprintf('Points belonging to cluster %d: %s\n',i,g)
    %for centroids
    counter = 0;
    sum = [0 0 0 0];
    for j = 1:size(x)
        sum = sum + d(x(j),:);
        counter = counter + 1;
```

```
    end
    sum = sum / counter;
    centroidComplete(i,:) = sum;

    %disp(sum);
end
fprintf('Centroid for the Complete link is \n');
disp(centroidComplete);
```

RESULTS OBTAINED:

```
Single linkage Points
Points belonging to cluster 1: 45
Points belonging to cluster 2: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 39 41 42 44 46 47 48 49 50 51 52 53 54
55 56 57
Points belonging to cluster 3: 40 43
Points belonging to cluster 4: 38
Centroid for the Single link is
   29.0000   39.0000   99.0000   67.0000
   68.1698   72.4717   69.9434   73.9057
   35.5000   86.5000   35.5000   62.5000
   90.0000   32.0000   28.0000   69.0000


Complete linkage Points
Points belonging to cluster 1: 40 43
Points belonging to cluster 2: 28 30 36 38 44
Points belonging to cluster 3: 1 3 4 8 9 10 14 17 22 23 24 26 27 33 34 42 45 48 51 52
53 54 55 56 57
Points belonging to cluster 4: 2 5 6 7 11 12 13 15 16 18 19 20 21 25 29 31 32 35 37 39
41 46 47 49 50
Centroid for the Complete link is
   35.5000   86.5000   35.5000   62.5000
   55.0000   35.4000   37.4000   74.0000
   52.4400   57.0400   77.8000   72.8000
   85.8400   92.3600   68.0800   74.5200
```

c. Comment on any differences in the cluster centers and cluster compositions for the two different clusterings as performed in (b) above.

Solution:

- In Single link clustering, the clusters are formed in such a way that we tend to combine the data points or we tend to combine the clusters which has minimum distance between them and we used Euclidean distance between the points.
- In Complete link clustering or the max-link clustering, the clusters or points are combined in such a way that, the points/clusters are combined which has maximum points between them.
- The centroids for single and complete link clusterings differ because in single link, the distance between the points are minimum and this criteria differs for the complete linkage clustering.
- The centroid for the single linkage clusters are formed/combined till we get 4 resultant clusters and the combining criteria is minimum.
- The compositions obtained are in such a way that, more points are combined for cluster 2 in single link and more points are accumulated to cluster3 and 4 in complete link because of the above mentioned criteria.
- Complete link also follows a similar but takes maximum distances between the values.

- Hence the centroid values and the composition of the points differ for both single and complete link accordingly.
- The cluster compositions also differ with each other as single link tries to combine points which are nearer and complete link tries to combine the data points which are maximum in distance.

d. Compute Rand Index for the comparison of Clustering-2 and Clustering-3 and show the counts a, b, c, and d as determined for computing the Rand index. Explain the meaning of each count and why such counts have been obtained for this dataset and their clustering's.

STEPS FOLLOWED:

- Rand Index is a measure of the similarity between two data clusterings.
- Single linkage clustering supports chained data and complete link clusters are good for the globular cluster structures. If the clusters are overlapping in the two clustering algorithms, the adjusted rand index would be closer to 1.
- Higher rand index represents that good clusters are formed.
- a, the number of pairs of elements in that are in the same subset in Clustering-3 and in the same subset in Clustering-2
- b, the number of pairs of elements in that are in different subsets in Clustering-3 and in different subsets in Clustering-2
- c, the number of pairs of elements in that are in the same subset in Clustering-3 and in different subsets in Clustering-2
- d, the number of pairs of elements in that are in different subsets in Clustering-3 and in the same subset in Clustering-2
- Rand Index finds the extent of similarity between the 2 clusters under comparison.
- The value of a tells us that the clusters are similar to each other in both the clusters as an order of "a"

MATLAB CODE:

```
%for rand index 2.d
a3 = 0;
b3 = 0;
c3 = 0;
d3 = 0;
for i=1:size(d)

    for j=(i+1):size(d)
        if((clustersDividedSingle(i,1) == clustersDividedSingle(j,1)) &&
(clustersDividedComplete(i,1) == clustersDividedComplete(j,1)))
            a3 = a3+1;
        elseif((clustersDividedSingle(i,1) ~= clustersDividedSingle(j,1)) &&
(clustersDividedComplete(i,1) ~= clustersDividedComplete(j,1)))
            b3=b3+1;
        elseif((clustersDividedSingle(i,1) == clustersDividedSingle(j,1)) &&
(clustersDividedComplete(i,1) ~= clustersDividedComplete(j,1)))
            c3=c3+1;
        else
            d3=d3+1;
        end

    end
```

```
end

randindex = (a3+b3)/(a3+b3+c3+d3);
fprintf('RandIndex of Single linkage and Complete linkage\n');
fprintf('a(number of pairs that are in the same set in Clustering2 and in the same set
in Clustering 3) : %d\n',a3);
fprintf('b(number of pairs that are in the different set in Clustering2 and in the
different set in Clustering 3): %d\n',b3);
fprintf('c(number of pairs that are in the same set in Clustering2 and in the
different set in Clustering 3): %d\n',c3);
fprintf('d(number of pairs that are in the different set in Clustering2 and in the
same set in Clustering 3): %d\n',d3);
fprintf('Rand index is %d \n', randindex);
```

RESULTS OBTAINED:

```
RandIndex of Single linkage and Complete linkage
a(number of pairs that are in the same set in Clustering2 and in the same set in
Clustering 3) : 583
b(number of pairs that are in the different set in Clustering2 and in the different
set in Clustering 3): 189
c(number of pairs that are in the same set in Clustering2 and in the different set in
Clustering 3): 796
d(number of pairs that are in the different set in Clustering2 and in the same set in
Clustering 3): 28
Rand index is 4.837093e-01
```

3. Compute Rand Index for the comparison of Clustering-1 and Clustering-2 and show the counts a, b, c, and d as determined for computing the Rand index. Explain the meaning of each count and why such counts have been obtained for this dataset and these clusterings in this comparison.

Explanation:

- Rand Index is a measure of the similarity between two data clusterings.
- Rand index gives an insight about how well the clusters are overlapping. Adjusted rand index ranges from -1 to 1.
- Higher rand index represents that good clusters are formed.
- a, the number of pairs of elements in that are in the same subset in Clustering-1 and in the same subset in Clustering-2
- b, the number of pairs of elements in that are in different subsets in Clustering-1 and in different subsets in Clustering-2
- c, the number of pairs of elements in that are in the same subset in Clustering-1 and in different subsets in Clustering-2
- d, the number of pairs of elements in that are in different subsets in Clustering-1 and in the same subset in Clustering-2
- K means are good for the globular dataset while single link is for chained data sets. The clusters under comparison are of different k values and has very little to overlap between the clusters. That is the reason we get a very low value of rand index between the clustering1 and clustering2 methods.
- Rand index of cluster1 and cluster2 is lower when compared to single and complete tells us that clustering2 and clustering3 are more similar than clustering2 and clustering1.

MATLAB CODE:

```
%3rd Solution
%for rand index 2.d

a2 = 0;
b2 = 0;
c2 = 0;
d2 = 0;
for i=1:size(d)

    for j=(i+1):size(d)
        if((clustersDividedSingle(i,1) == clustersDividedSingle(j,1)) &&  (idx(i,1) ==
idx(j,1)))
            a2 = a2+1;
        elseif((clustersDividedSingle(i,1) ~= clustersDividedSingle(j,1)) &&
(idx(i,1) ~= idx(j,1)))
            b2=b2+1;
        elseif((clustersDividedSingle(i,1) == clustersDividedSingle(j,1)) &&
(idx(i,1) ~= idx(j,1)))
            c2=c2+1;
        else
            d2=d2+1;
        end
    end
end

randindex2 = (a2+b2)/(a2+b2+c2+d2);

fprintf('\nRandIndex of Single linkage and Clustering-1(with 10 clusters on
dataset):\n');
fprintf('a(number of pairs that are in the same set in Clustering2 and in the same set
in Clustering1) : %d\n',a2);

fprintf('b(number of pairs that are in the different set in Clustering2 and in the
different set in Clustering1): %d\n',b2);

fprintf('c(number of pairs that are in the same set in Clustering2 and in the
different set in Clustering1): %d\n',c2);
fprintf('d(number of pairs that are in the different set in Clustering2 and in the
same set in Clustering1): %d\n',d2);
fprintf('Rand index is %d \n', randindex2);
```

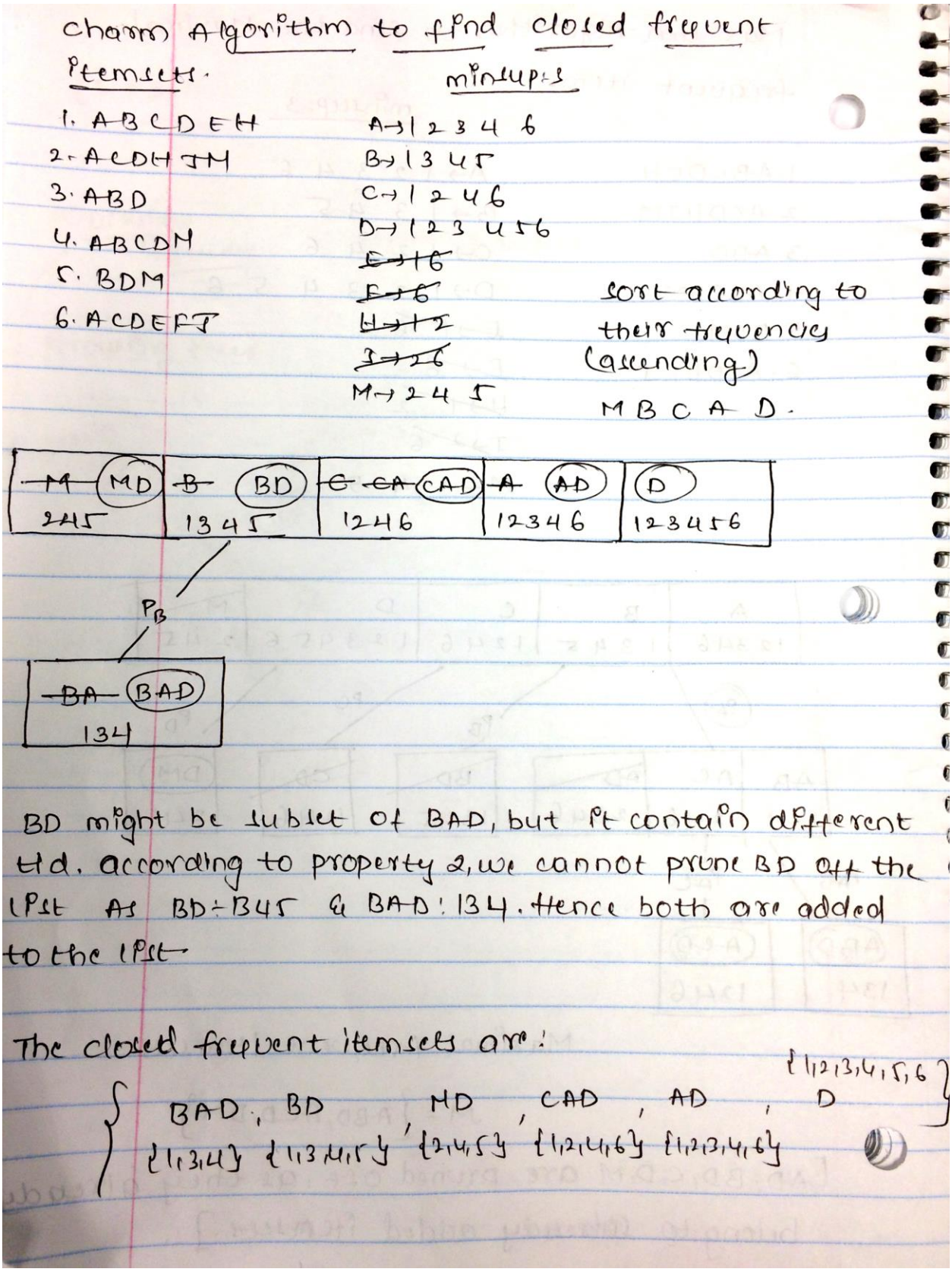
RESULTS OBTAINED:

```
RandIndex of Single linkage and Clustering-1(with 10 clusters on dataset):
```

```
a(number of pairs that are in the same set in Clustering2 and in the same set in
Clustering1) : 170
b(number of pairs that are in the different set in Clustering2 and in the different
set in Clustering1): 210
c(number of pairs that are in the same set in Clustering2 and in the different set in
Clustering1): 1209
d(number of pairs that are in the different set in Clustering2 and in the same set in
Clustering1): 7
Rand index is 2.380952e-01
```

4. Show the execution tree for the CHARM algorithm for finding all the closed itemsets for the dataset containing the following transactions: ABCDEH, ACDHJM, ABD, ABCDM, BDM, ACDEFJ.

Solution:



5. For the same data as in #4 above, show execution of the algorithm for finding all the maximal itemsets.

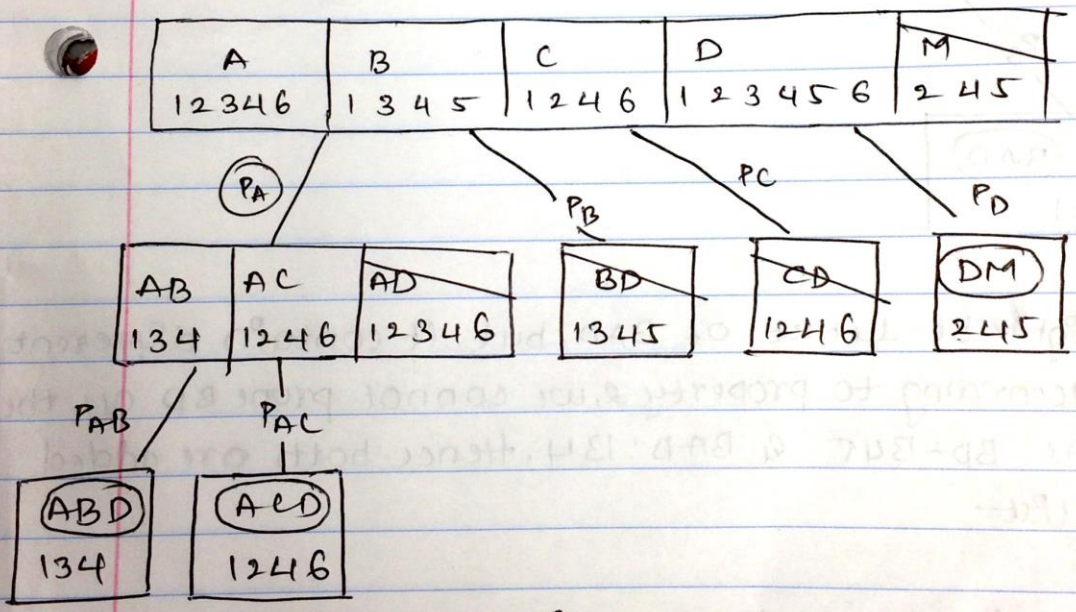
Solution:

Max-Gen Algorithm to find the Maximal frequent items.

minsup=3

1. ABCDEH
2. ACDHJM
3. ABD
4. ABCDM.
5. BDM
6. ACDEFJ

A → 1 2 3 4 6
B → 1 3 4 5
C → 1 2 4 6
D → 1 2 3 4 5 6
~~E → 6~~
~~F → 6~~
~~H → 2~~
~~I → 6~~
M → 2 4 5.



Maximal frequent Itemsets

M = {ABD, ACD, DM}

[AD, BD, CD, M are pruned off, as they already belong to already added itemsets.].