CS 550 HW3- Clustering

Oytun GÜNEŞ, ID: 20901170

Part 1:

Introduction:

In this part I have implemented K-means clustering for the image sample.jpeg, when K=2, 4, 8, 16. K-means clustering works as follows:

start with randomly initialized cluster (mean) vectors
do
    for each cluster, estimate samples that belong to the mean vector (mi) (estimate Di)
    for each cluster, compute the mean vector mi that minimizes/maximizes the criterion function until there is no (or small) change in (mi).

For distance estimation I have used Euclidean distance as:

$$D_i = [x| \ \|x - m_i\|^2 = \min_j \|x - m_j\|^2]$$

For mean-vector estimation, taking the derivative of sum of squared error:

$$m_i = \frac{\sum_{x \epsilon D_i} x}{n_i}$$

Where n_i is the number of samples in D_i.

Initialization of the Mean Vectors:

I have initialized the mean vectors not totally random in order to increase the speed of convergence. As a better choice, I have divided the colour vector into K different values with increasing up to the maximum value within the image. I have done this for red, green and blue.

```
rand_r_vector=randi(length(unique(vector_red)), [1 K]);
rand_g_vector=randi(length(unique(vector_green)), [1 K]);
rand_b_vector=randi(length(unique(vector_blue)), [1 K]);

centroid_red=sort(rand_r_vector,'ascend');
centroid_green=sort(rand_g_vector,'ascend');
centroid_blue=sort(rand_b_vector,'ascend');
```

For example for K=2 if the maximum of red is 203 my mean vector would be=[102 203]. As you can see the amount of increase is the same up to the maximum value.

Cluster Vectors:

K=2:

V =

127	215
83	196
130	201

K=4:

V=

85 147 198 225
51 98 158 220
103 142 180 214

K=8:

V =

61 110 97 167 138 182 211 227
36 52 74 85 107 136 178 226
88 98 133 122 154 168 191 217

K=16:

V=

47 81 60 90 98 137 126 126 171 153 191 178 210 210 219 229
26 35 46 74 51 60 76 99 87 119 117 148 156 178 201 230
70 78 111 136 102 98 127 150 124 161 149 178 174 192 206 218

Stopping Criterion:

I have used a threshold for the absolute difference in the mean vectors. I have set the threshold to 20 because it was a good trade off between computational time and decrease in the error.

Clustering Error:

I have calculated clustering error as the mean square error of the original image and the clustered image.

```
distred=(((red_original-red_clustered)).^2);
distgreen=(((green_original-green_clustered)).^2);
distblue=(((blue_original-blue_clustered)).^2);
distance = sqrt(distred+distgreen+distblue);
err(j)=mean(mean(distance));
```

Clustering error for K=2,4,8,16 is as follows:

K=2	K=4	K=8	K=16
43.18	25.79	18.22	13.17

Images:

As it can be seen from the images below as K increases the image become clearer in terms of the color. For K=2 there are only two color where the image itself is nearly impossible to identify.

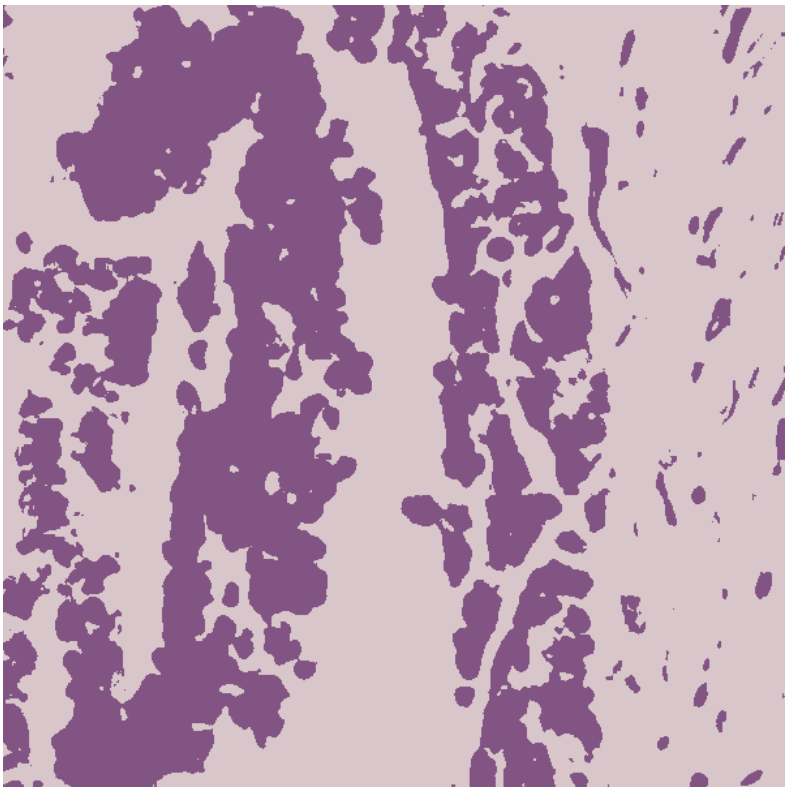


Figure 1: Clustered Image for K=2

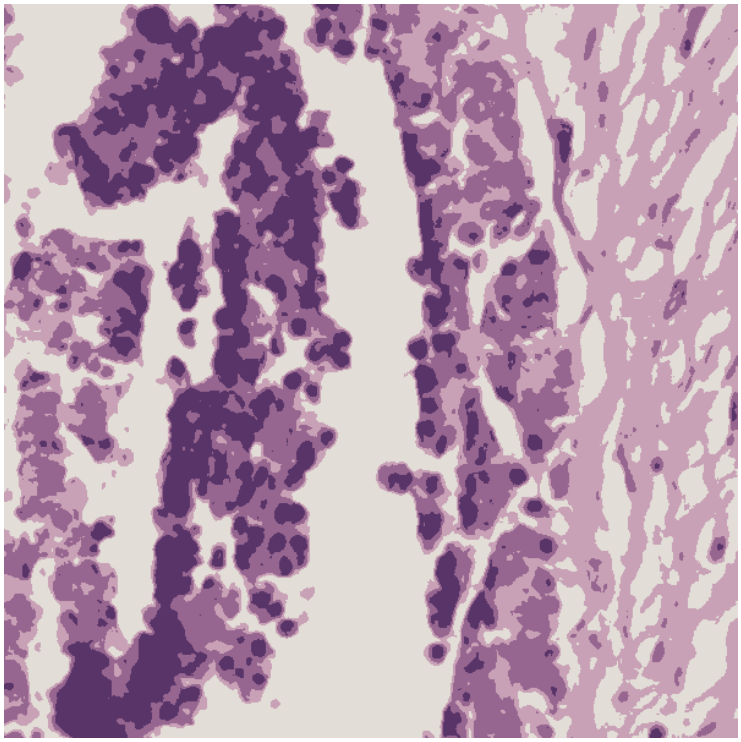


Figure 2: Clustered Image for K=4

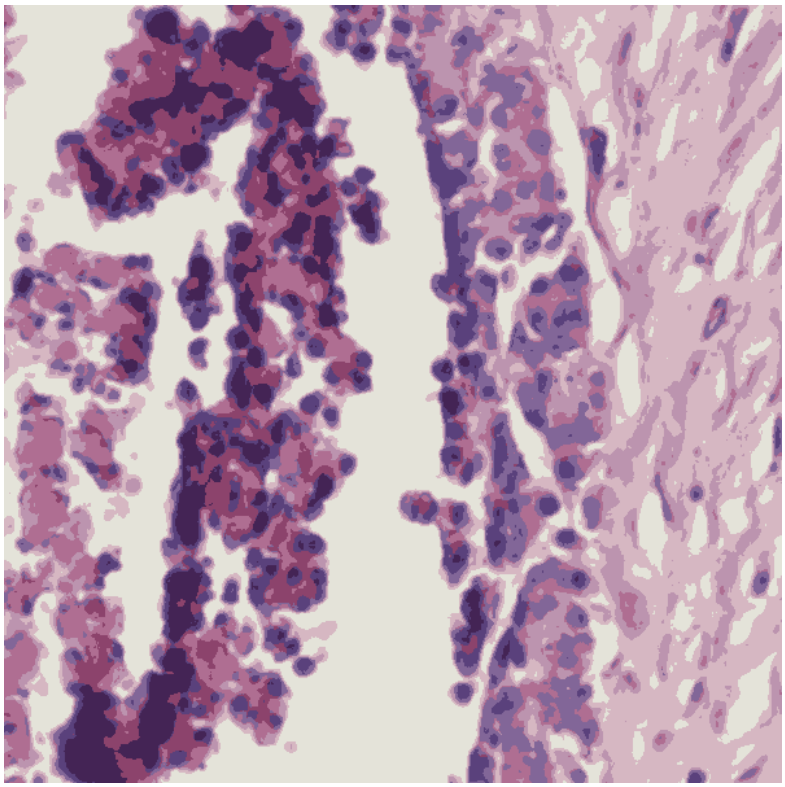


Figure 3: Clustered Image for K=8

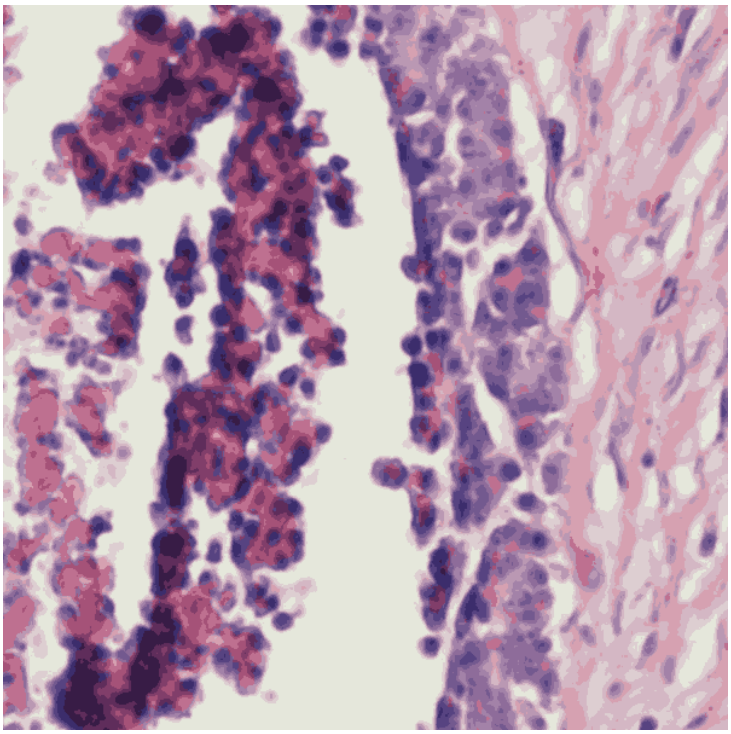
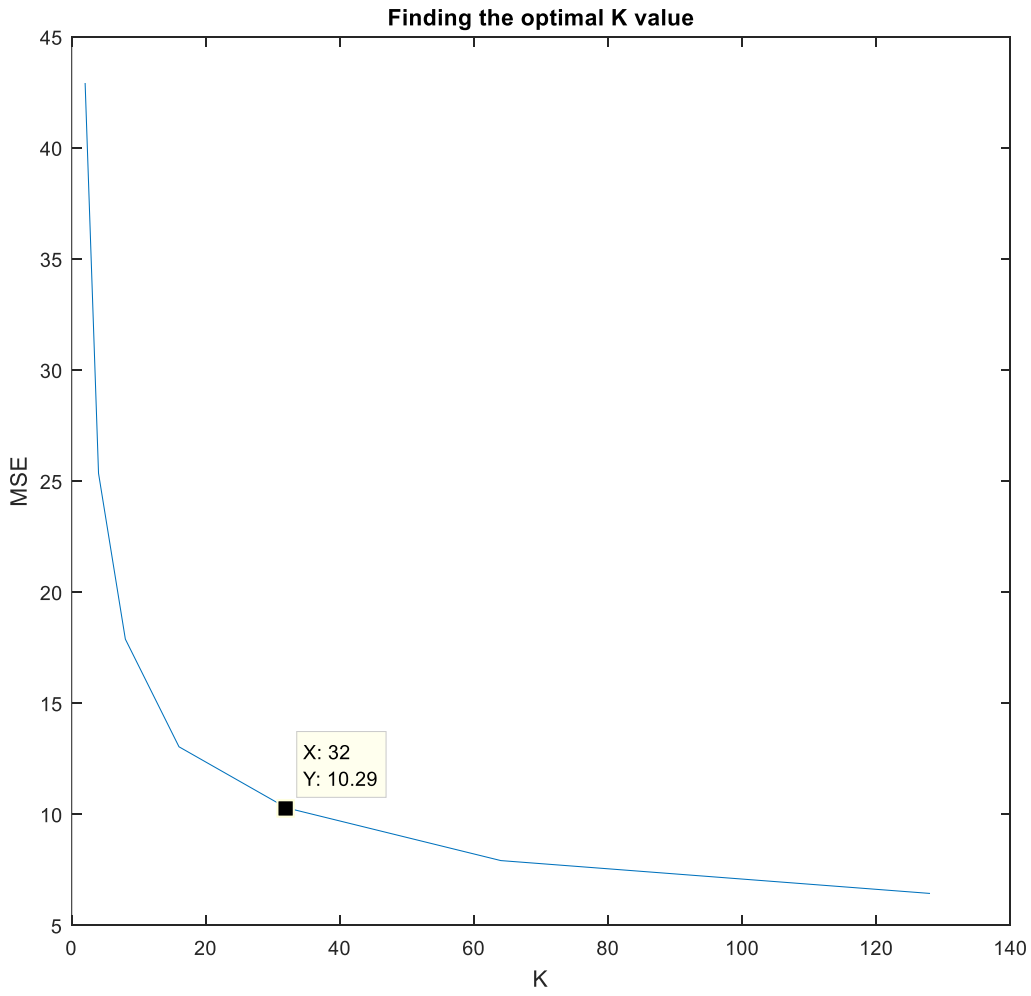


Figure 4 :Clustered Image for K=16

The optimal K value:

I have done experiments for K= 2,4,8,16,32,64,128 and calculated the MSE for each K, I have obtained the following plot of MSE as K increases:



The computation load of K value above 128 was very high therefore I did not plot them. Using the elbow rule for clustering I have obtained the optimal K=32.

Image for optimal K=32:

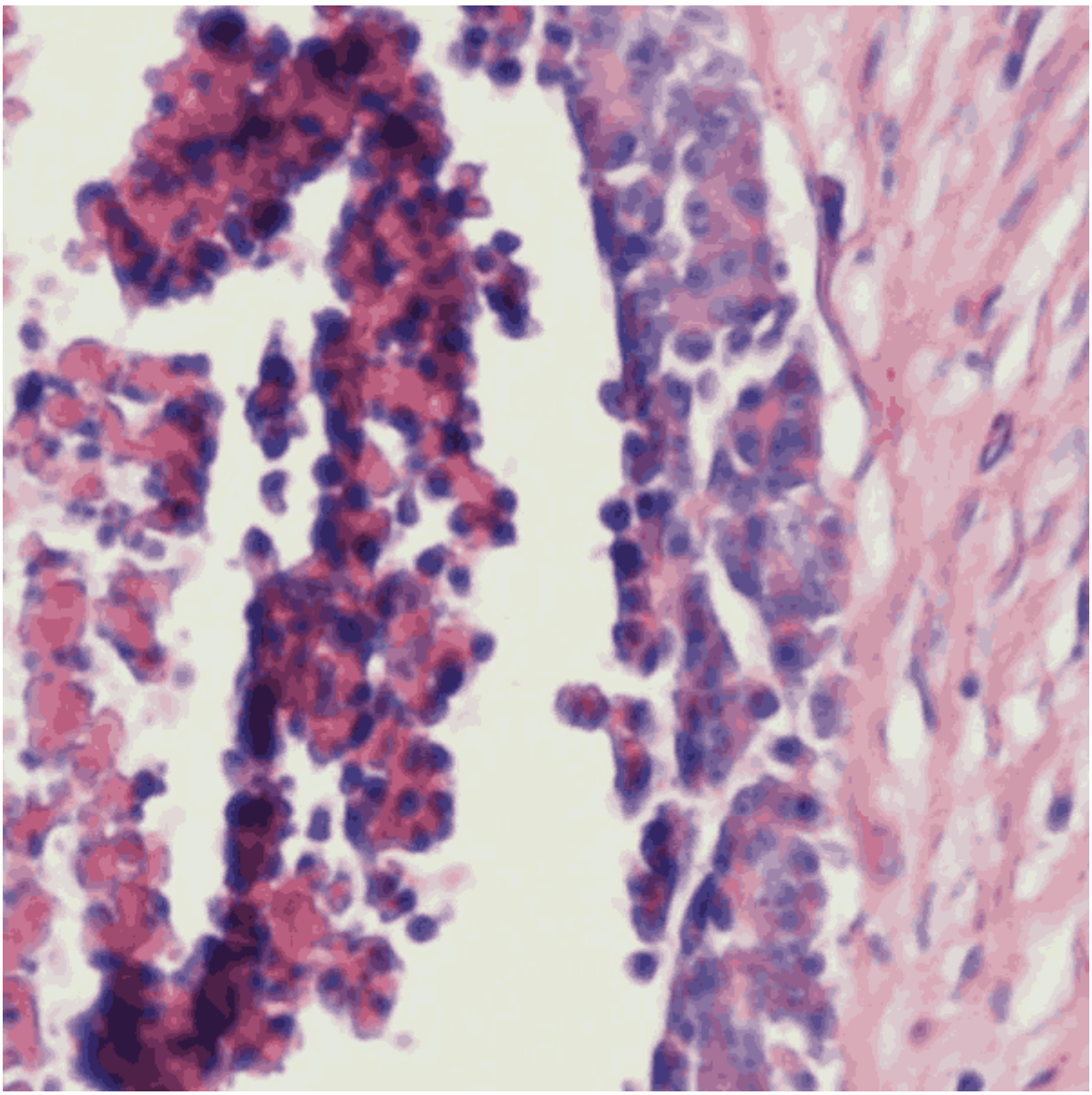


Figure 5:Clustered Image for K=32

Part 2: Hierarchical Clustering

In this part I have used another smaller size image in order to gain from computational time because by using MATLAB in first part some simulations took hours. I have selected my original image from my previous image processing course 'onion.png' as below:

Figure 6: Original Image

In aggloremative(bottom-up) clustering the aim is to start with N singleton clusters and merge the clusters successively with respect to their similarities. It can be represented with dendograms as below:

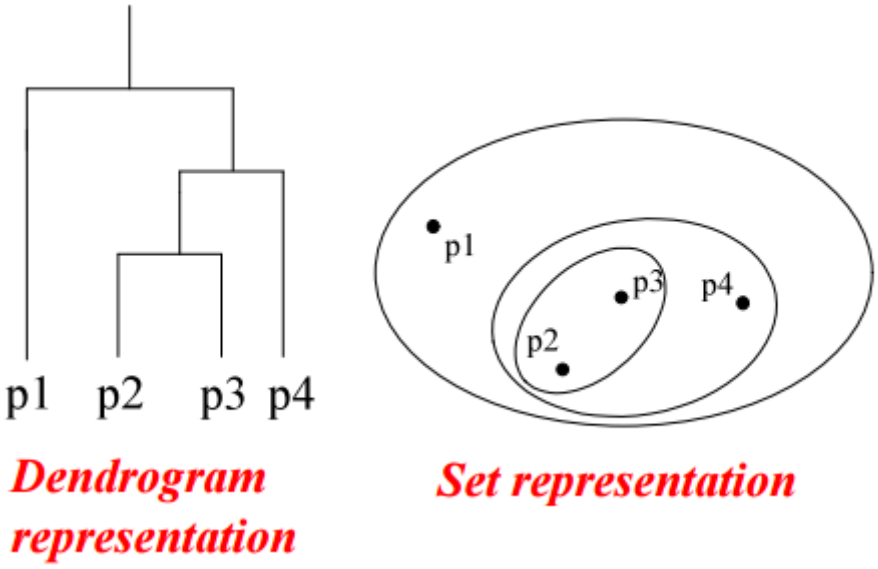


Figure 7: Dendogram representation in Hierarchical Clustering

Algorithm is as follows:

let each sample be a cluster compute the dissimilarity (distance) matrix

repeat

merge the two most similar cluster

update the distance matrix

until only a single cluster remains

As a distance matrix I have used the similarity between **mean vectors.**

Details of the Technique:

In order to decrease computational time I have first done K-means clustering up to K=128 and stored cmap and cluster vectors V in order to use in the algorithm. Otherwise I would need to store 26730*26730 if I start by assigning each sample as clusters.

Clustering Vectors:

K=2:

V=

229 155
197 101
137 34

K=4:

V=

254 195 241 108
187 135 213 59
6 104 137 42

K=8:

V=

171 223 255 96 155 237 138 64
145 70 191 157 33 208 125 38
97 57 9 165 37 153 26 49

K=16:

V=

212 153 236 185 255 215 255 254 252 161 110 226 189 108 153 58
50 128 85 176 189 112 214 229 231 35 17 190 160 101 116 42
51 77 88 53 3 12 58 123 204 43 20 181 118 35 18 37

Clustering Error:

K=2	K=4	K=8	K=16
79.86	45.01	32.38	20.71

Images for K=2,4,8,16:

Figure 8: Hierarchical Clustering K=2

Figure 9: Hierarchical Clustering K=4

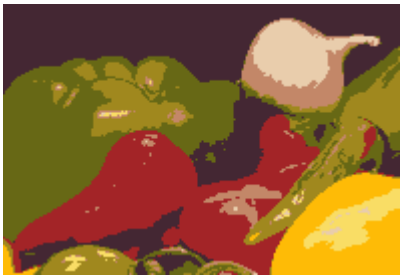
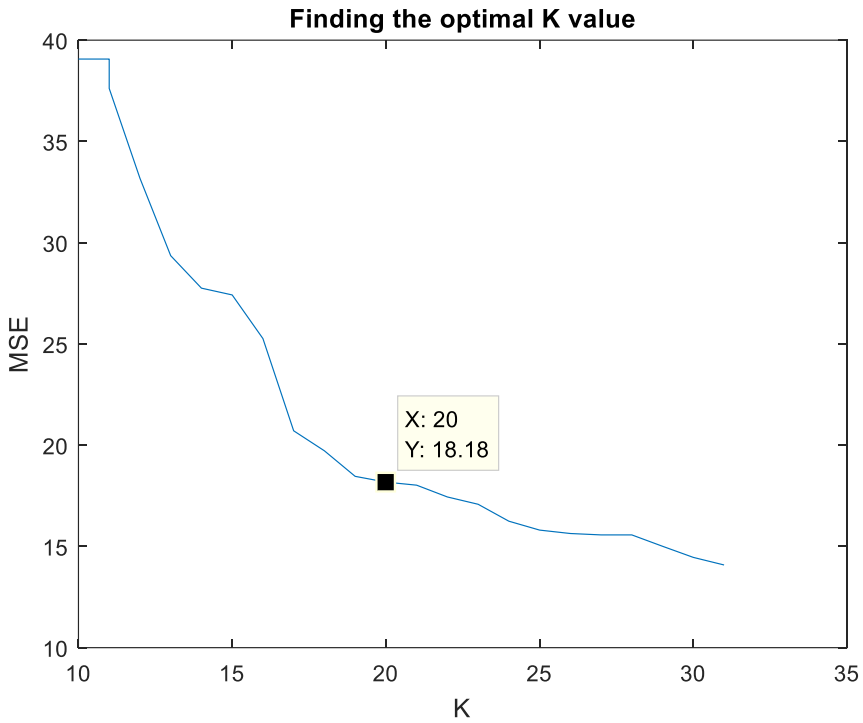


Figure 10: Hierarchical Clustering K=8

Figure 11: Hierarchical Clustering K=16

Selection of Optimal K Value:



I have found an optimal K value as K=20 using the elbow rule in clustering.

Optimal Image:

Figure 12: Optimal Image for Hierarchical Clustering, K=20

Comparison of K-means and Hierarchical clustering:

Time Complexity

K-means is linear in the number of data objects i.e. $O(n)O(n)$, where *n* is the number of data objects. The time complexity of most of the hierarchical clustering algorithms is quadratic i.e. $O(n2)O(n2)$. Therefore, for the same amount of data, hierarchical clustering will take

quadratic amount of time.

Shape of Clusters

K-means works well when the shape of clusters are hyper-spherical (or circular in 2 dimensions). If the natural clusters occurring in the dataset are non-spherical then probably K-means is not a good choice.

Repeatability

K-means starts with a random choice of cluster centers, therefore it may yield different clustering results on different runs of the algorithm. Thus, the results may not be repeatable and lack consistency. However, with hierarchical clustering, we most definitely get the same clustering results.