Document for heartdisease project

1. Install and load the required libraries
2. Load , clean and view the data set

heart_data

Filter

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope
1	63	1	1	145	233	1	2	150	0	2.3	3
2	67	1	4	160	286	0	2	108	1	1.5	2
3	67	1	4	120	229	0	2	129	1	2.6	2
4	37	1	3	130	250	0	0	187	0	3.5	3
5	41	0	2	130	204	0	2	172	0	1.4	1
6	56	1	2	120	236	0	0	178	0	0.8	1
7	62	0	4	140	268	0	2	160	0	3.6	3
8	57	0	4	120	354	0	0	163	1	0.6	1
9	67	1	4	160	286	0	2	108	1	1.5	2
10	63	1	4	130	254	0	2	147	0	1.4	2
11	53	1	4	140	203	1	2	155	1	3.1	3
12	57	1	4	140	192	0	0	148	0	0.4	2
13	56	0	2	140	294	0	2	153	0	1.3	2
14	67	1	4	160	286	0	2	108	1	1.5	2
15	56	1	3	130	256	1	2	142	1	0.6	2
16	44	1	2	120	263	0	0	173	0	0.0	1
17	67	1	4	160	286	0	2	108	1	1.5	2
19	57	1	3	150	168	0	0	174	0	1.6	1
20	48	1	2	110	229	0	0	168	0	1.0	3
21	54	1	4	140	239	0	0	160	0	1.2	1
22	48	0	3	130	275	0	0	139	0	0.2	1
23	49	1	2	130	266	0	0	171	0	0.6	1
24	64	1	1	110	211	0	2	144	1	1.8	2
25	58	0	1	150	283	1	2	162	0	1.0	1
26	58	1	2	120	284	0	2	160	0	1.8	2
27	58	1	3	132	224	0	2	173	0	3.2	1

Showing 1 to 26 of 300 entries, 11 total columns

3.define multiple feature sets.

```
> feature_sets <- list(
+   set1 = c("age", "trestbps", "chol", "fbs", "thalach", "oldpeak"),
+   set2 = c("age", "chol", "thalach")
+ )
```

4.initialize a result dataframe.

```
> results <- data.frame(
+   Algorithm = character(),
+   Feature_Set = character(),
+   Mean_Silhouette = numeric()
+ )
```

5. Iterate through feature sets and Check the structure of the numeric data.

```
> for (set_name in names(feature_sets)) {
+   features <- feature_sets[[set_name]]
+   heart_numeric <- heart_data[, features]
+
+   # Check the structure of the numeric data
+   print(paste("Feature Set:", set_name))
+   print(summary(heart_numeric))
```

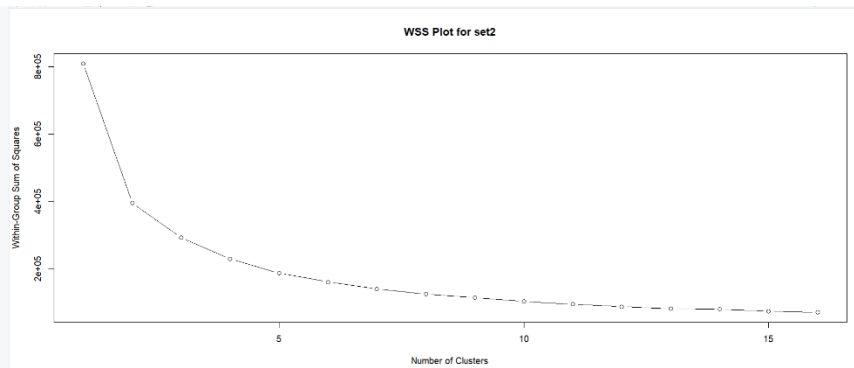
6. Handle any missing values or constants by removing them.

```
+
+   # Handle any missing values or constants by removing them
+   if (any(is.na(heart_numeric))) {
+     heart_numeric <- na.omit(heart_numeric)
+   }
+
```

Perform k-means clustering algorithm .

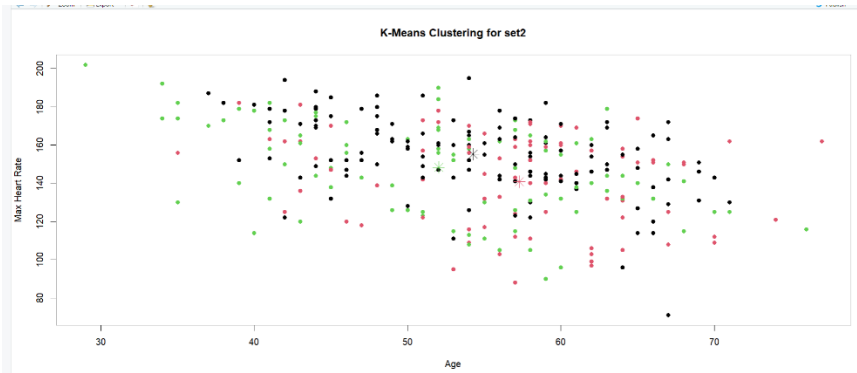
1.Find optimal k using wss and plot wss for set2.

```
wss <- numeric(16)
for (i in 1:16) {
  wss[i] <- sum(kmeans(heart_numeric, centers = i, nstart = 10)$tot.withinss)
}
plot(1:16, wss, type = 'b', xlab = 'Number of Clusters', ylab = 'Within-Group Sum of Squares',
     main = paste("WSS Plot for", set_name))
```



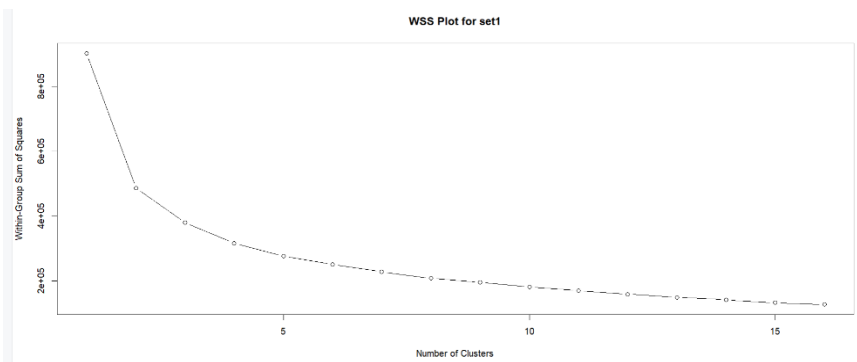
2.perform k-means and visualize the result for set2.

```
optimal_clusters <- 3  # Set manually based on WSS plot
kc <- kmeans(heart_numeric, centers = optimal_clusters, nstart = 10)

# Visualize K-Means clustering results
plot(heart_numeric$age, heart_numeric$thalach, col = kc$cluster, pch = 19,
     xlab = "Age", ylab = "Max Heart Rate",
     main = paste("K-Means Clustering for", set_name))
points(kc$centers[, c("age", "thalach")], col = 1:optimal_clusters, pch = 8, cex = 2)
```



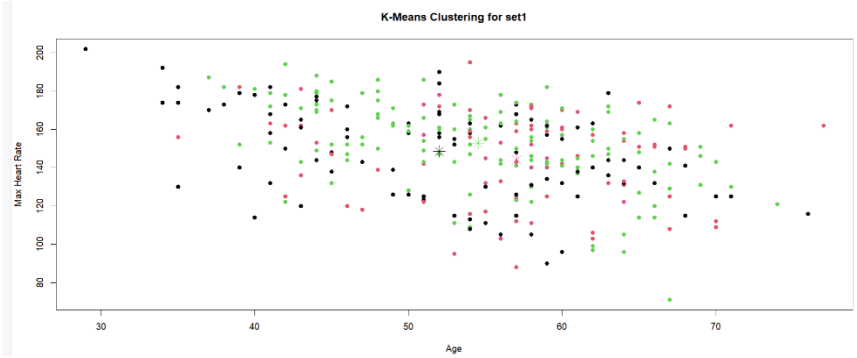
3.Find optimal k using wss and plot wss for set1.

```
wss <- numeric(16)
for (i in 1:16) {
  wss[i] <- sum(kmeans(heart_numeric, centers = i, nstart = 10)$tot.withinss)
}
plot(1:16, wss, type = 'b', xlab = 'Number of Clusters', ylab = 'Within-Group Sum of Squares',
     main = paste("WSS Plot for", set_name))
```



4.perform k-means and visualize the result for set1.

```
optimal_clusters <- 3  # Set manually based on WSS plot
kc <- kmeans(heart_numeric, centers = optimal_clusters, nstart = 10)

# Visualize K-Means clustering results
plot(heart_numeric$age, heart_numeric$thalach, col = kc$cluster, pch = 19,
     xlab = "Age", ylab = "Max Heart Rate",
     main = paste("K-Means Clustering for", set_name))
points(kc$centers[, c("age", "thalach")], col = 1:optimal_clusters, pch = 8, cex = 2)
```



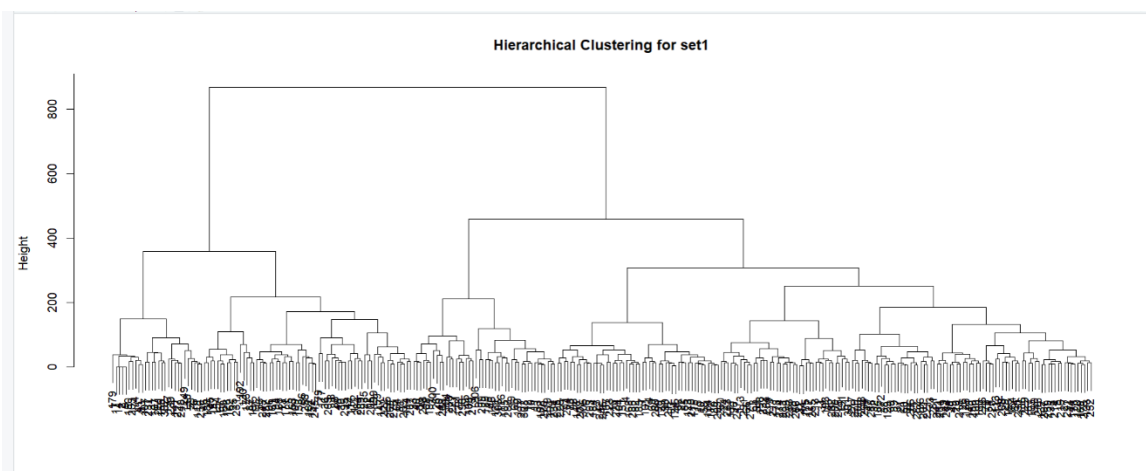
3.calculate Silhouette Score for K-Means .

```
if (length(unique(kc$cluster)) > 1) {
  silhouette_kmeans <- silhouette(kc$cluster, dist(heart_numeric))
  mean_silhouette_kmeans <- mean(silhouette_kmeans[, 3])
} else {
  mean_silhouette_kmeans <- NA
}
```

Perform Hierarchical Clustering .

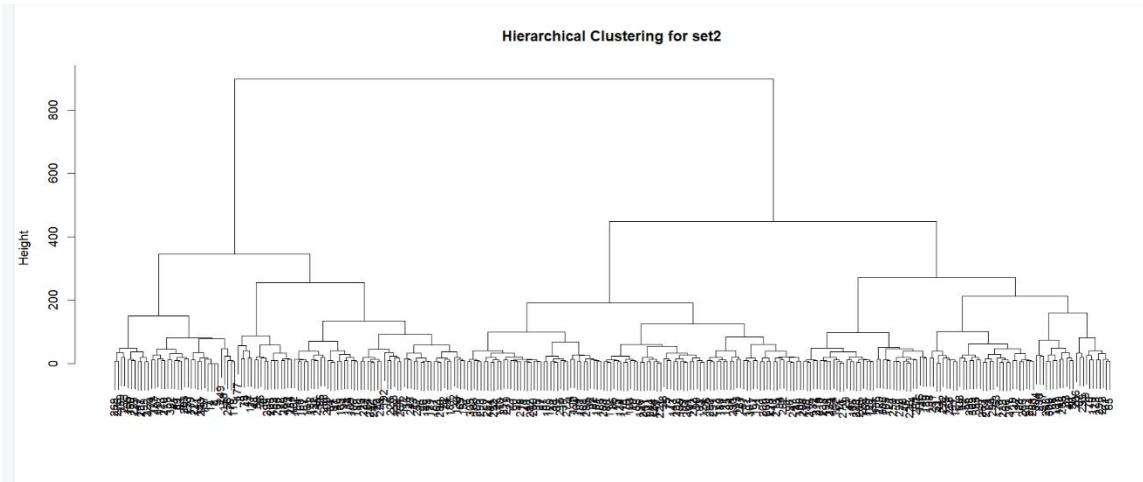
1. Perform this algorithm and plot its graph for set1.

```
dist_matrix <- dist(heart_numeric, method = "euclidean")
hclust_result <- hclust(dist_matrix, method = "ward.D2")
plot(hclust_result, main = paste("Hierarchical Clustering for", set_name), xlab = "", sub = "", cex = 0.9)

num_clusters_hierarchical <- 3
cluster_assignments_hierarchical <- cutree(hclust_result, k = num_clusters_hierar
```



2.Perform this algorithm and plot its graph for set2.

```
dist_matrix <- dist(heart_numeric, method = "euclidean")
 hclust_result <- hclust(dist_matrix, method = "ward.D2")
 plot(hclust_result, main = paste("Hierarchical Clustering for", set_name), xlab = "", sub = "", cex = 0.9)

 num_clusters_hierarchical <- 3
 cluster_assignments_hierarchical <- cutree(hclust_result, k = num_clusters_hierar
```



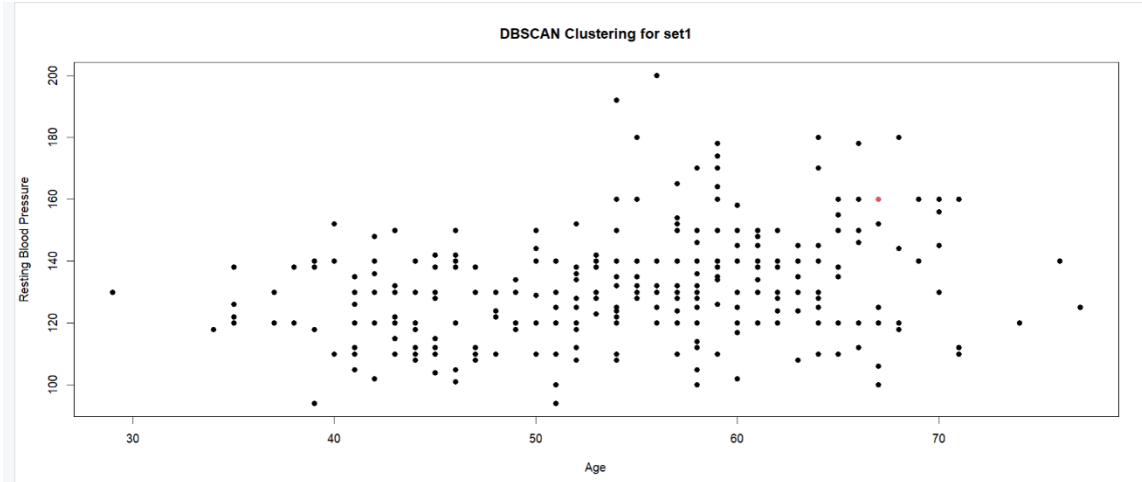
3.calculate Silhouette Score for Hierarchical Clustering .

```
# Silhouette Score for Hierarchical Clustering
if (length(unique(cluster_assignments_hierarchical)) > 1) {
  silhouette_hierarchical <- silhouette(cluster_assignments_hierarchical, dist(heart_numeric))
  mean_silhouette_hierarchical <- mean(silhouette_hierarchical[, 3])
} else {
  mean_silhouette_hierarchical <- NA
}
```

Perform DBSCAN algorithm using epslon=0.6 and ,minpts=4(adjust esplon based on dataset scale).

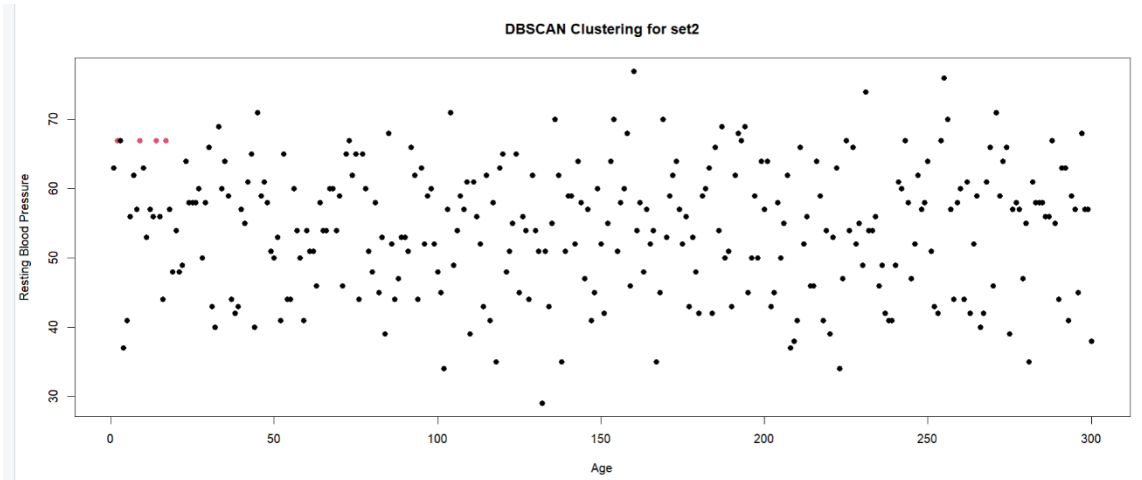
1. Perform DBSCAN algorithm and visualize its result for set1.

```
eps_value <- 0.6
 minPts_value <- 4
 dbscan_result <- dbscan(heart_numeric, eps = eps_value, minPts = minPts_value)

 # Visualize DBSCAN clustering results
 plot(heart_numeric$age, heart_numeric$trestbps, col = dbscan_result$cluster + 1, pch = 19,
      xlab = "Age", ylab = "Resting Blood Pressure",
      main = paste("DBSCAN Clustering for", set_name))
```



2.Perform DBSCAN algorithm and visualize its result for set2.

```
eps_value <- 0.6
minPts_value <- 4
dbscan_result <- dbscan(heart_numeric, eps = eps_value, minPts = minPts_value)

# Visualize DBSCAN clustering results
plot(heart_numeric$age, heart_numeric$trestbps, col = dbscan_result$cluster + 1, pch = 19,
     xlab = "Age", ylab = "Resting Blood Pressure",
     main = paste("DBSCAN Clustering for", set_name))
```



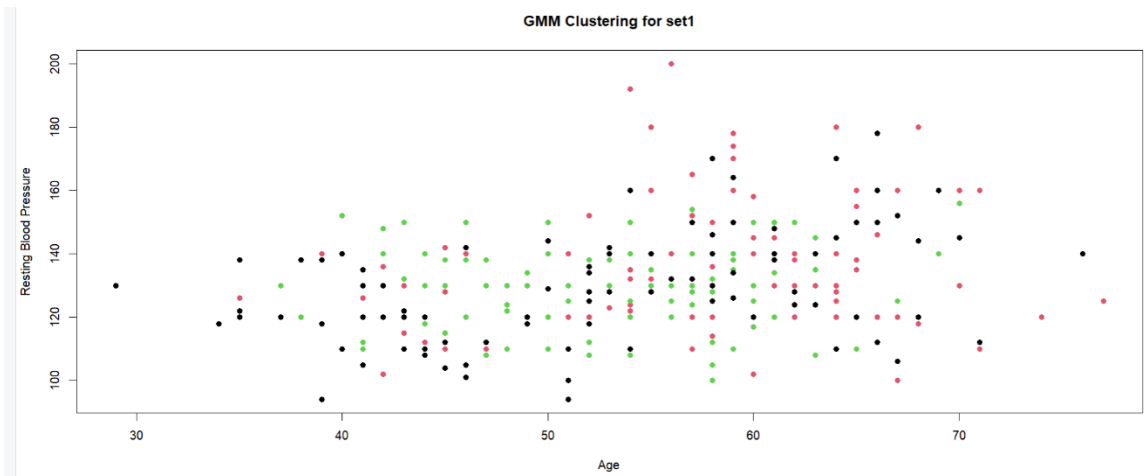
3. calculate Silhouette Score for DBSCAN.

```
> # Silhouette Score for DBSCAN
>   if (length(unique(dbscan_result$cluster)) > 1) {
+     silhouette_dbscan <- silhouette(dbscan_result$cluster, dist(heart_numeric))
+     mean_silhouette_dbscan <- mean(silhouette_dbscan[, 3])
+   } else {
+     mean_silhouette_dbscan <- NA
+   }
>
```

Perform GMM clustering with G= num_clusters_hierarchical.

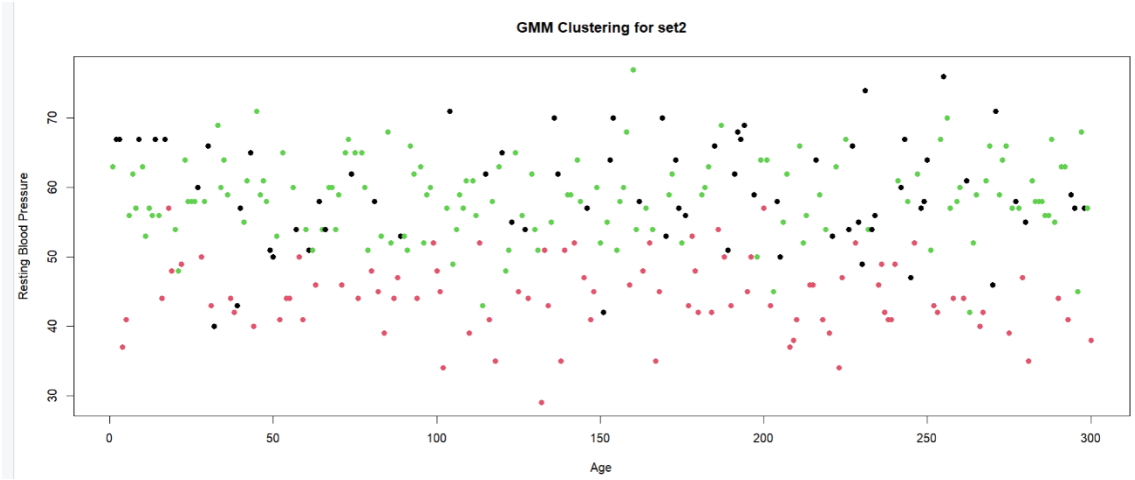
1.perform GMM algorithm and visualize its result for set1.

```
> # GMM Clustering
> gmm_result <- Mclust(heart_numeric, G = num_clusters_hierarchical)
fitting ...
  |=========================================================================================================
>
> # Visualize GMM clustering results
> plot(heart_numeric$age, heart_numeric$trestbps, col = gmm_result$classification, pch = 19,
+      xlab = "Age", ylab = "Resting Blood Pressure",
+      main = paste("GMM Clustering for", set_name))
>
```



2. perform GMM algorithm and visualize its result for set2.

```
> # GMM Clustering
> gmm_result <- Mclust(heart_numeric, G = num_clusters_hierarchical)
fitting ...
  |=========================================================================================================
>
> # Visualize GMM clustering results
> plot(heart_numeric$age, heart_numeric$trestbps, col = gmm_result$classification, pch = 19,
+      xlab = "Age", ylab = "Resting Blood Pressure",
+      main = paste("GMM Clustering for", set_name))
>
```



3. calculate Silhouette Score for GMM.

```
if (length(unique(gmm_result$classification)) > 1) {
    silhouette_gmm <- silhouette(gmm_result$classification, dist(heart_numeric))
    mean_silhouette_gmm <- mean(silhouette_gmm[, 3])
  } else {
    mean_silhouette_gmm <- NA
  }
```

Append results for Silhouette Scores for set1and set2 for each algorithm and display them.

```
  # Append results for this feature set
  results <- rbind(
    results,
    data.frame(Algorithm = "KMeans", Feature_Set = set_name, Mean_Silhouette = mean_silhouette_kmeans),
    data.frame(Algorithm = "Hierarchical", Feature_Set = set_name, Mean_Silhouette = mean_silhouette_hierarchical),
    data.frame(Algorithm = "DBSCAN", Feature_Set = set_name, Mean_Silhouette = mean_silhouette_dbscan),
    data.frame(Algorithm = "GMM", Feature_Set = set_name, Mean_Silhouette = mean_silhouette_gmm)
  )
}

# Display silhouette results for all feature sets
print(results)
```

```
> print(results)
     Algorithm Feature_Set Mean_Silhouette
1       KMeans        set1      0.27826502
2 Hierarchical        set1      0.25854666
3       DBSCAN        set1      0.11975021
4          GMM        set1      0.25144741
5       KMeans        set2      0.33089619
6 Hierarchical        set2      0.28985725
7       DBSCAN        set2      0.09370062
8          GMM        set2      0.11097923
>
```

Finally ,We foundout from this scores that k-means is the best algorithm for this dataset with best num of clusters=3.