ashishsa_hw2_p3

```
my_data <- read.delim("seeds_dataset.txt",header = TRUE)
head(my_data)
```

```
##    Area Perimeter Compactness Length.Kernel Width.Kernel Asymmetry
## 1 15.26     14.84      0.8710         5.763        3.312     2.221
## 2 14.88     14.57      0.8811         5.554        3.333     1.018
## 3 14.29     14.09      0.9050         5.291        3.337     2.699
## 4 13.84     13.94      0.8955         5.324        3.379     2.259
## 5 16.14     14.99      0.9034         5.658        3.562     1.355
## 6 14.38     14.21      0.8951         5.386        3.312     2.462
##   Length.Kernel.Grove Seed.Group
## 1               5.220          A
## 2               4.956          A
## 3               4.825          A
## 4               4.805          A
## 5               5.175          A
## 6               4.956          A
```

Here we observe that the data contains very large values in length for Area and Perimeter while the length of each attribute of the other data is very small in case of other variables. So we need to scale the data. Without scaling if we perform the distance function we get NaN values.

```
## Warning: package 'ggdendro' was built under R version 3.6.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Warning: package 'dendextend' was built under R version 3.6.3
```

```
##
## ---------------------
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
## The following object is masked from 'package:ggdendro':
##
##     theme_dendro
## The following object is masked from 'package:stats':
##
##     cutree
## Warning: package 'philentropy' was built under R version 3.6.3
## Warning: package 'ape' was built under R version 3.6.3
##
## Attaching package: 'ape'
## The following objects are masked from 'package:dendextend':
##
##     ladderize, rotate
## Warning: package 'fossil' was built under R version 3.6.3
## Loading required package: sp
## Warning: package 'sp' was built under R version 3.6.3
## Loading required package: maps
## Warning: package 'maps' was built under R version 3.6.3
##
## Attaching package: 'maps'
## The following object is masked from 'package:cluster':
##
##     votes.repub
## Loading required package: shapefiles
## Warning: package 'shapefiles' was built under R version 3.6.3
## Loading required package: foreign
##
## Attaching package: 'shapefiles'
## The following objects are masked from 'package:foreign':
##
##     read.dbf, write.dbf
##
## Attaching package: 'fossil'
## The following objects are masked from 'package:philentropy':
##
##     euclidean, jaccard, manhattan
```

We now scale the data

```
m <- apply(my_data[,-8],2,mean)
s <- apply(my_data[,-8],2,sd)
z <- scale(my_data[,-8],m,s)
```

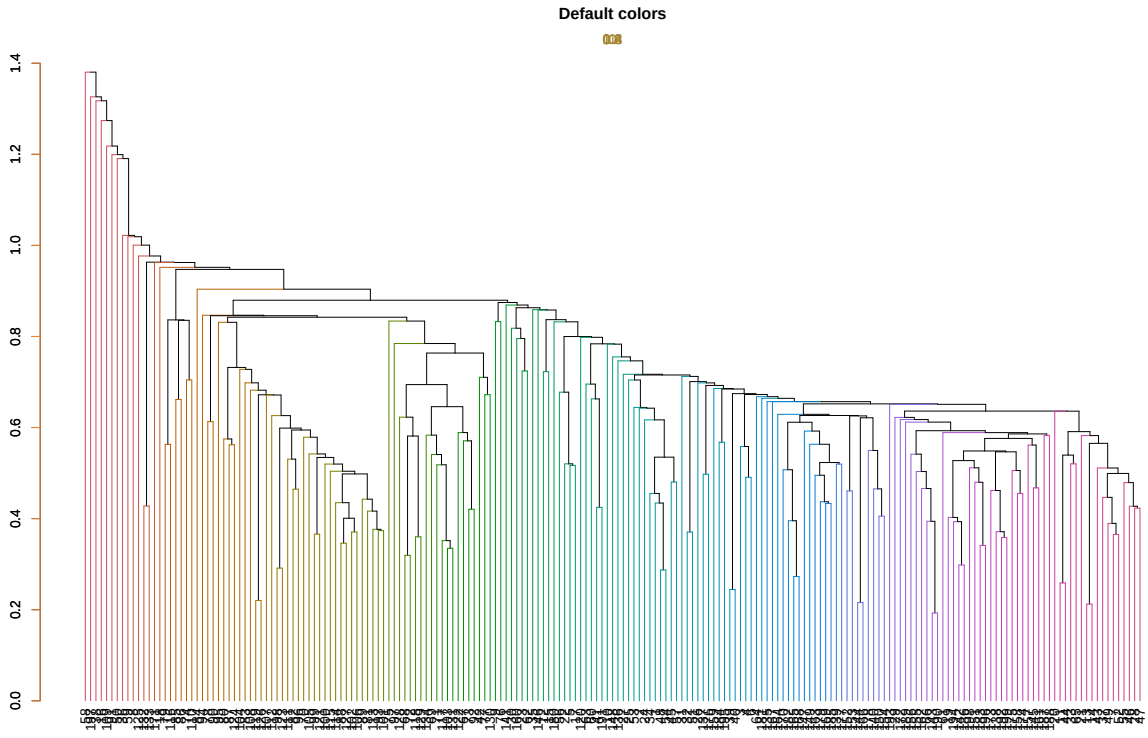
Calculate the Eucledian Distance.

```
distance1 <- dist(z,method = "euclidean")
```

##SINGLE LINKAGE

We cluster the data using Single Linkage

```
hclust1 <- hclust(distance1,method="single")
dend1 <- as.dendrogram(hclust1)
dend1 %>% set("branches_k_color") %>%
  plot(main = "Default colors") %>%
  axis(side = 2,col = "#F38630",labels = TRUE) %>%
  mtext(col = "#A38630")
```



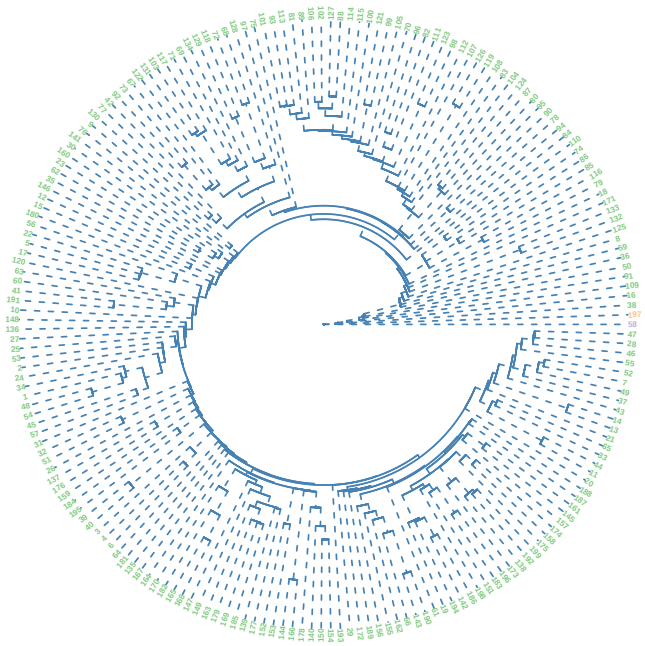
Incase of Single Linkage of Heirarchical Clustering we observe that there are a large number of clusters. We find the Minimal Inter-Cluster Similarity which is used to calculate the inter-cluster similarity. We now plot the clusters using Fan-Plot.

```
cutree1 <- cutree(dend1,3)
cutree1
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   1   1
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

```
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   3   1   1
```

```
plot(as.phylo(hclust1), type = "fan", cex = 0.6,
    tip.color = brewer.pal(3, 'Accent')[cutree1],
      font = 2,
     edge.color = 'steelblue', edge.width = 2, edge.lty = 2)
```



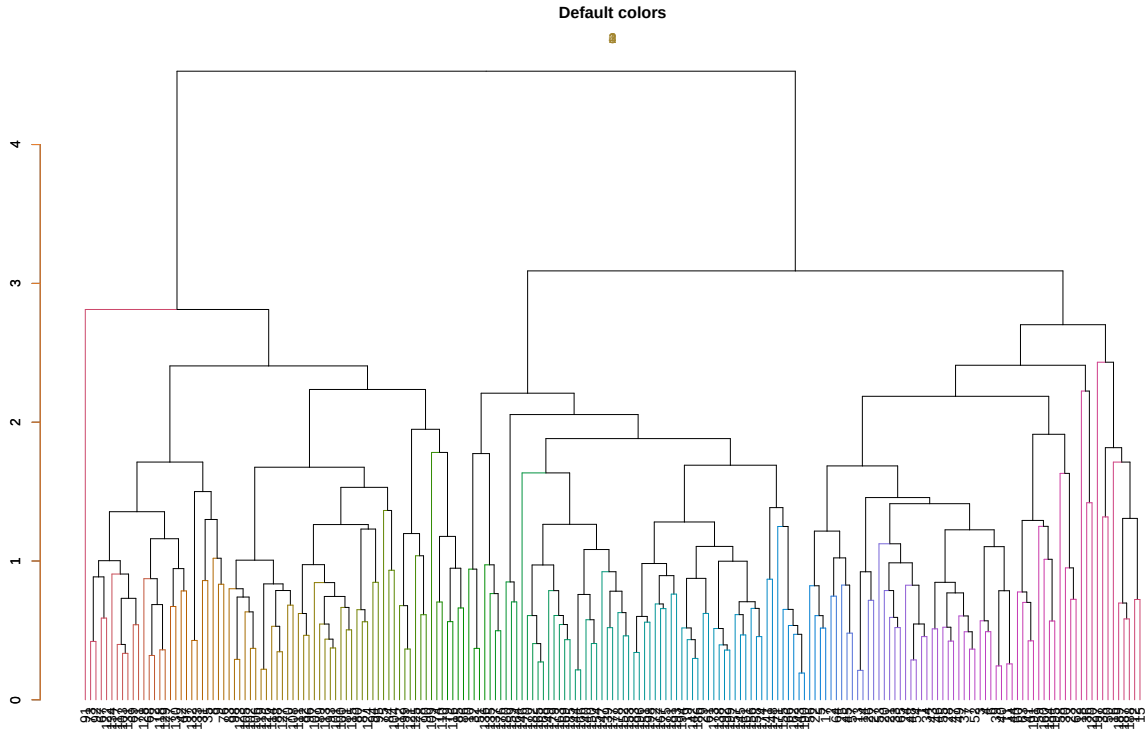
As we can observe from the fan plot most of thee data gets clustered into the same class.

##AVERAGE LINKAGE

We cluster the data using Average Linkage.

```
hclust2 <- hclust(distance1,method="average")
dend2 <- as.dendrogram(hclust2)
dend2 %>% set("branches_k_color") %>%
  plot(main = "Default colors") %>%
  axis(side = 2,col = "#F38630",labels = TRUE) %>%
```

```
mtext(col = "#A38630")
```

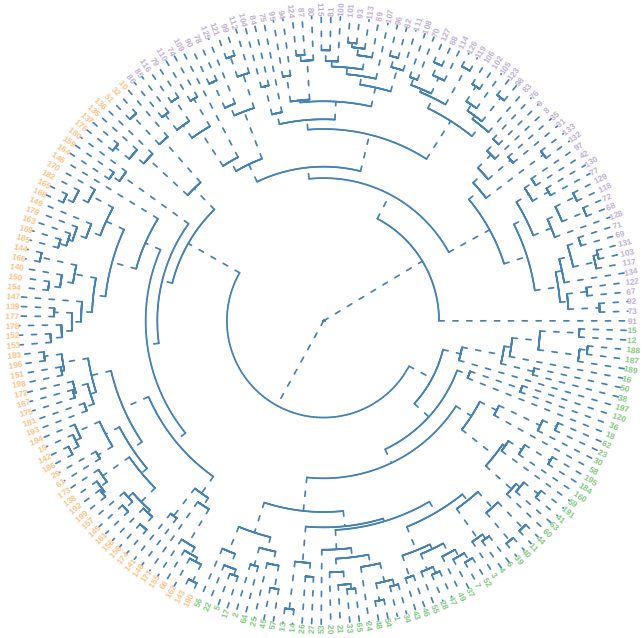


We compute Mean intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the average of these dissimilarities.

```
cutree2 <- cutree(dend2,3)
cutree2
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   1   1   1   1   1   1   1   2   2   3   1   1   1   1   1   1   1   1   3   1
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   1   1   1   1   1   1   1   1   3   1   2   3   1   1   2   1   1   1   1   1
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   1   2   1   1   1   1   1   1   1   1   3   1   1   1   1   1   1   1   1   1
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   3   1   1   1   1   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   3   3   3   3   3   3
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199
##   3   3   3   1   3   3   1   1   1   3   1   3   3   3   1   3   1   3   3
```

```
plot(as.phylo(hclust2), type = "fan", cex = 0.6,
    tip.color = brewer.pal(3, 'Accent')[cutree2],
      font = 2,
     edge.color = 'steelblue', edge.width = 2, edge.lty = 2)
```

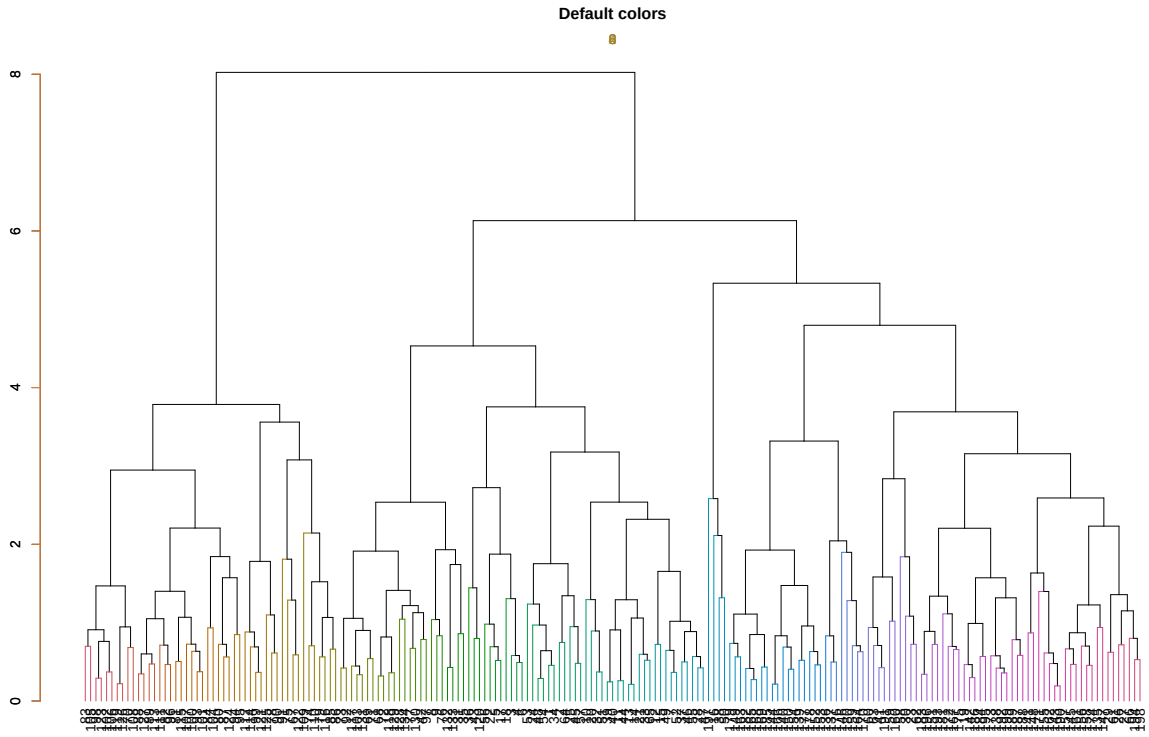


As we can observe from the fan plot the data gets clustered into three classes.

##COMPLETE LINKAGE

We cluster the data using Complete Linkage.

```
hclust3 <- hclust(distance1,method="complete")
dend3 <- as.dendrogram(hclust3)
dend3 %>% set("branches_k_color") %>%
  plot(main = "Default colors") %>%
  axis(side = 2,col = "#F38630",labels = TRUE) %>%
  mtext(col = "#A38630")
```



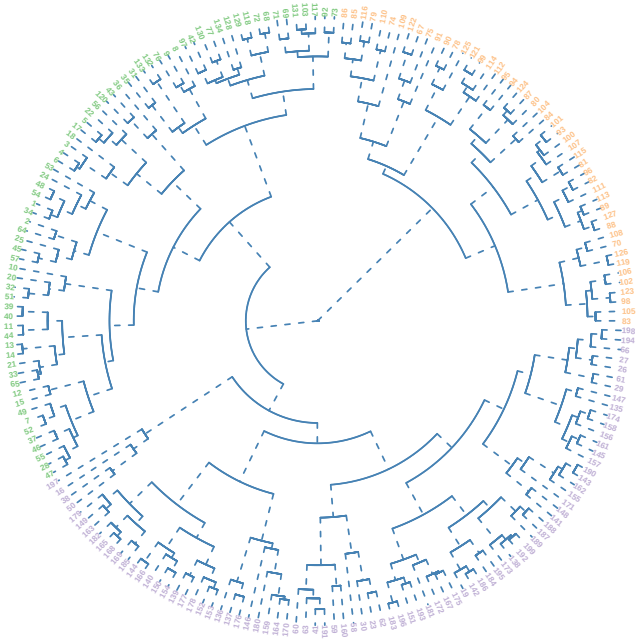
We compute Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the largest of these dissimilarities

```
cutree3 <- cutree(dend3,3)
cutree3
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   1   1   2   1
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   1   1   2   1   1   2   2   1   2   2   1   1   1   1   1   1   1   2   1   1
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   2   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1   1   2   2   2
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   2   2   2   1   1   2   3   1   1   3   1   1   1   3   3   1   1   3   3   3
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   3   3   3   3   3   3   3   3   3   3   3   1   3   3   3   3   1   3   3   3
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   3   3   1   3   3   3   3   3   3   3   3   3   3   3   3   3   1   1   3   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   3   3   3   3   3   3   3   1   1   1   1   1   1   1   2   2   2   2   2   2
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

As we can observe from the fan plot the data gets clustered into three classes.

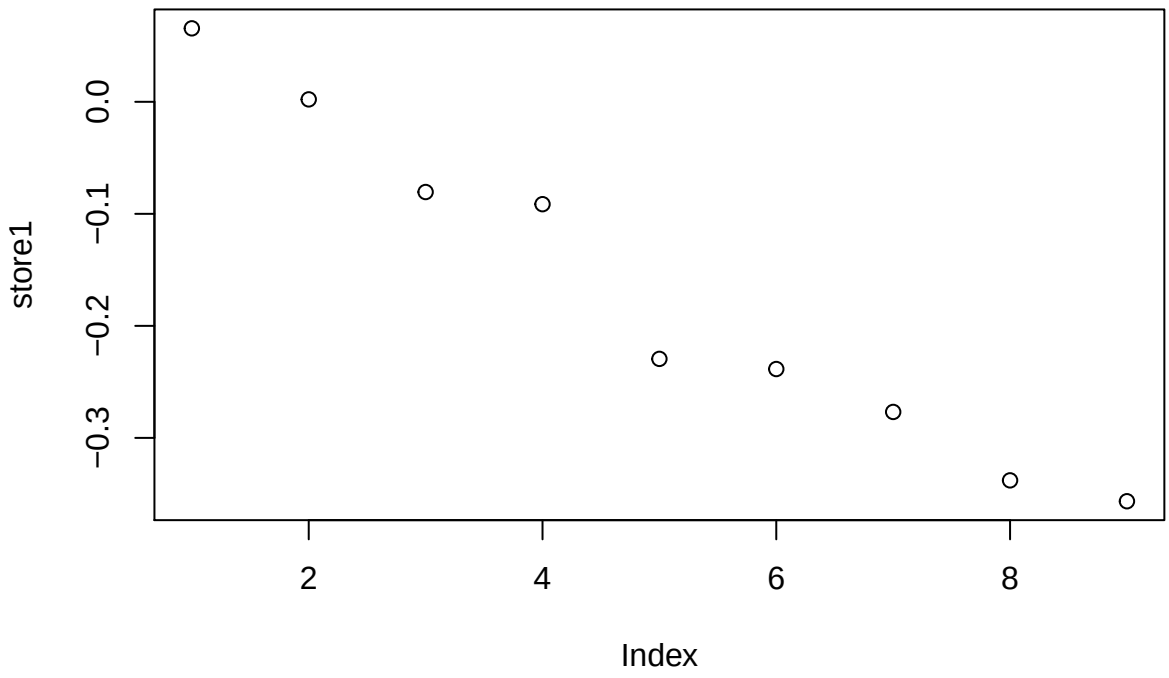
```
plot(as.phylo(hclust3), type = "fan", cex = 0.6,
    tip.color = brewer.pal(3, 'Accent')[cutree3],
      font = 2,
     edge.color = 'steelblue', edge.width = 2, edge.lty = 2)
```



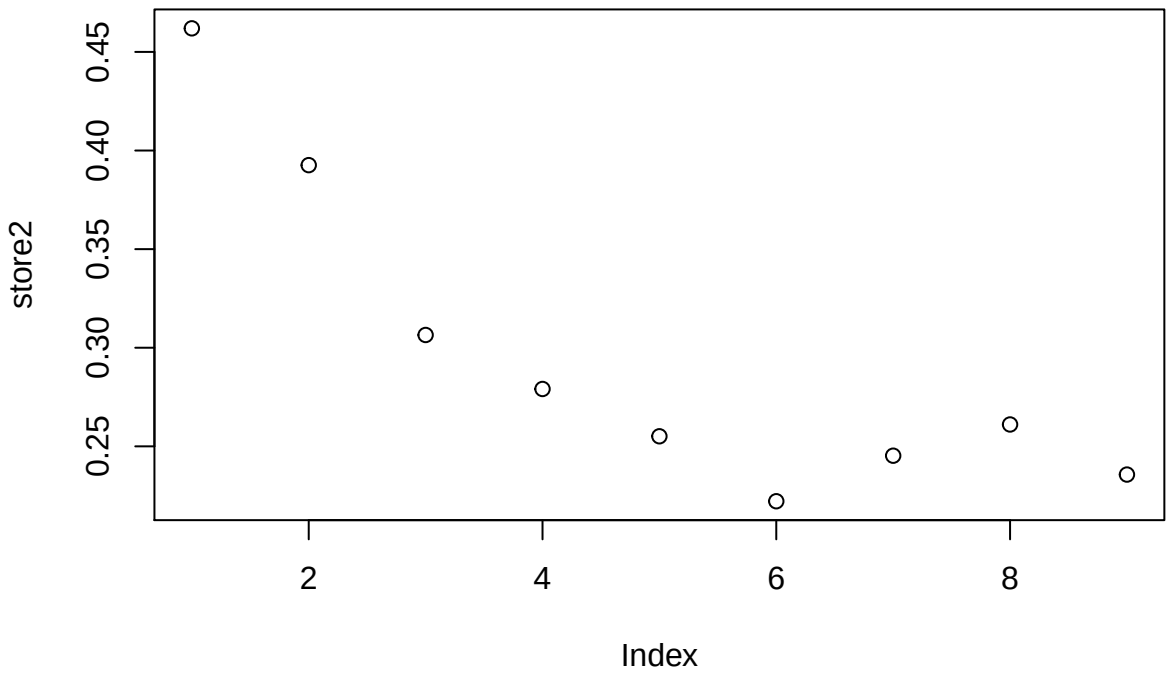
We use the Average silhouette plot to compare the performance of each of the above methods for various values of k for each of the clustering method.

##ERROR RATES FOR VARIOUS METHODS OF HEIRARCHICAL CLUSTERING The plot that has the lowest error for the best k-value gives us the best result.

```
store1 <- c()
for (i in 2:10){
  ct <- cutree(hclust1,k=i)
  si <- silhouette(ct,dist = distance1)
  avg_width <- summary(si)$avg.width
  store1 <- c(store1,avg_width)
}
plot(store1)
```

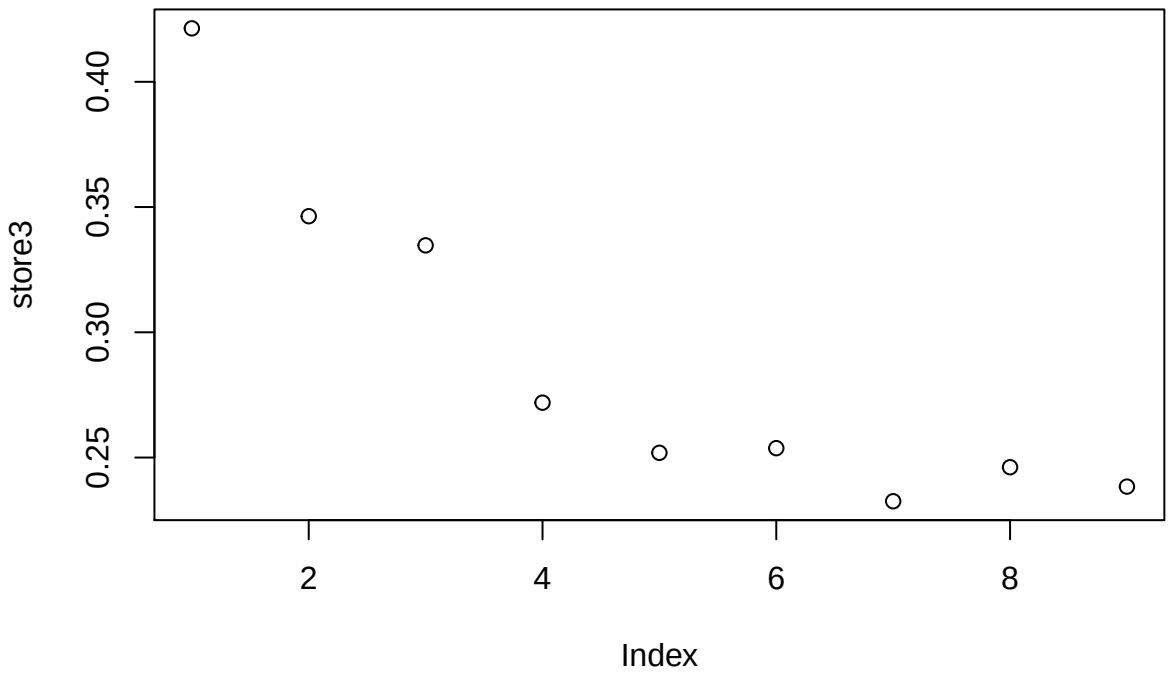



```
store2 <- c()
for (i in 2:10){
  ct <- cutree(hclust2,k=i)
  si <- silhouette(ct,dist = distance1)
  avg_width <- summary(si)$avg.width
  store2 <- c(store2,avg_width)
}
plot(store2)
```




```
library(ggplot2)
store3 <- c()
for (i in 2:10){
  ct <- cutree(hclust3,k=i)
  si <- silhouette(ct,dist = distance1)
  avg_width <- summary(si)$avg.width
  store3 <- c(store3,avg_width)
}

plot(store3)
```



From the above plots we can estimate that

1)complete linkage has the error of 0.35 for k=3(which is estimated to be best cluster size)

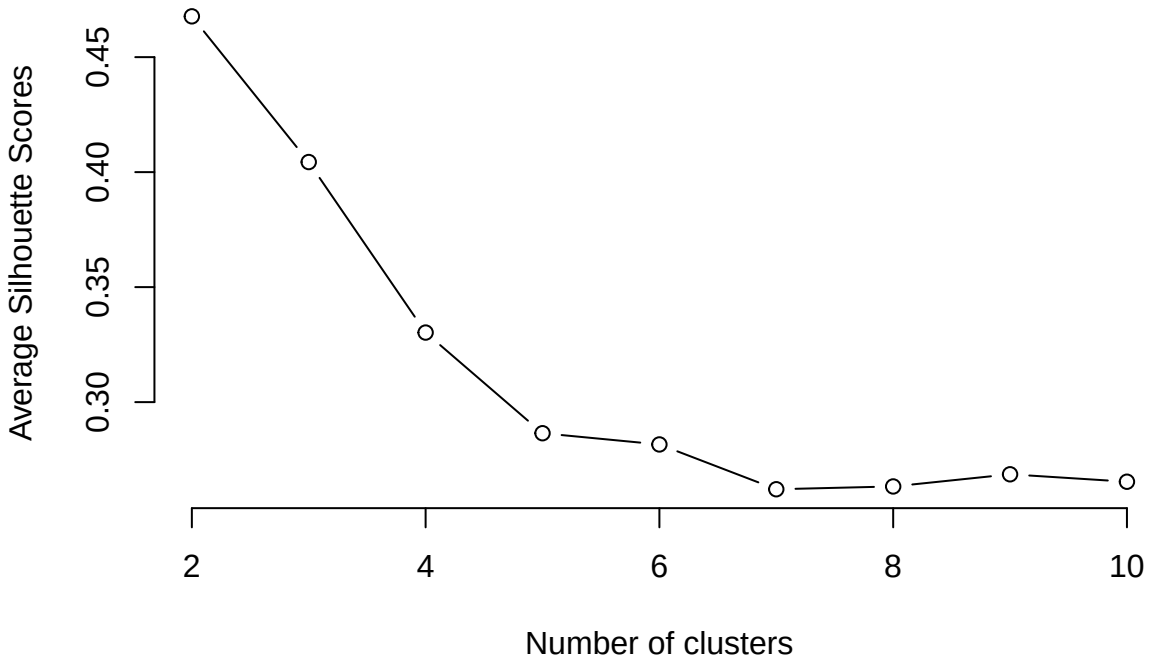
2)average linkage has the error of 0.30 for k=3(which is estimated to be best cluster size)

3)single linkage has the error of 0.1 for k=3(which is estimated to be best cluster size)

From the ablve observation we can conclude that Average and Complete has a better performance and alo they are preferred over single linkage as they give a balanced dendrogram.

##K-MEANS Kmeans – we complute the errors for various k values and then use the optimal k value to find the optimal k-size.

```
silhouette_score <- function(k){
  km <- kmeans(z, centers = k, nstart=10)
  ss <- silhouette(km$cluster, dist(z))
  mean(ss[, 3])
}
k <- 2:10
avg_sil <- sapply(k, silhouette_score)
plot(k, type='b', avg_sil, xlab='Number of clusters', ylab='Average Silhouette Scores', frame=FALSE)
```



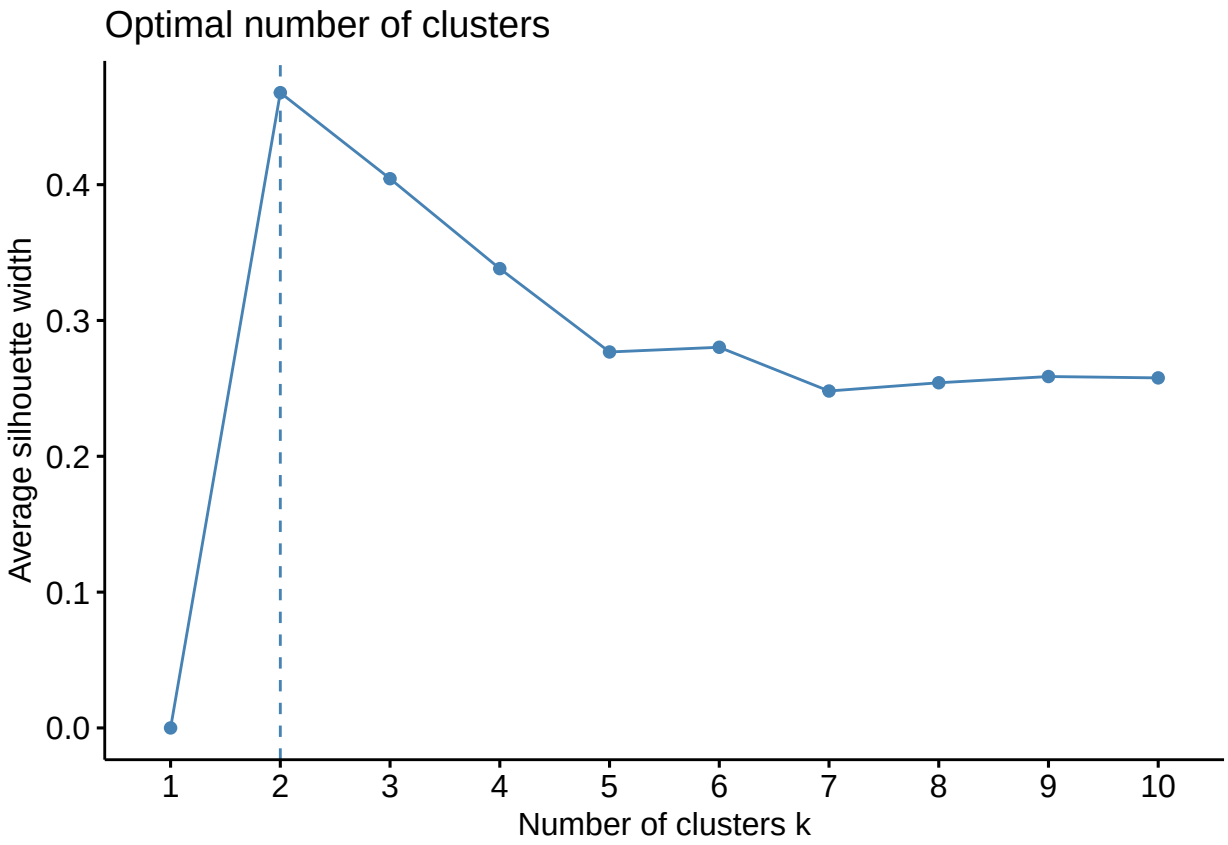
As it is difficult to find the optimum k size we use fviz_nbclust to find the right k.

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 3.6.3
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
fviz_nbclust(z, kmeans, method='silhouette')
```



We observe that the optimal number of cluster is obtained when k=2.

```
ka2 <- kmeans(z,centers=3,nstart = 10)
```

we calculate rand index and adjusted rand index to calculate the accuracy.

```
rand.index(ka2$cluster,as.numeric(my_data[,8]))
```

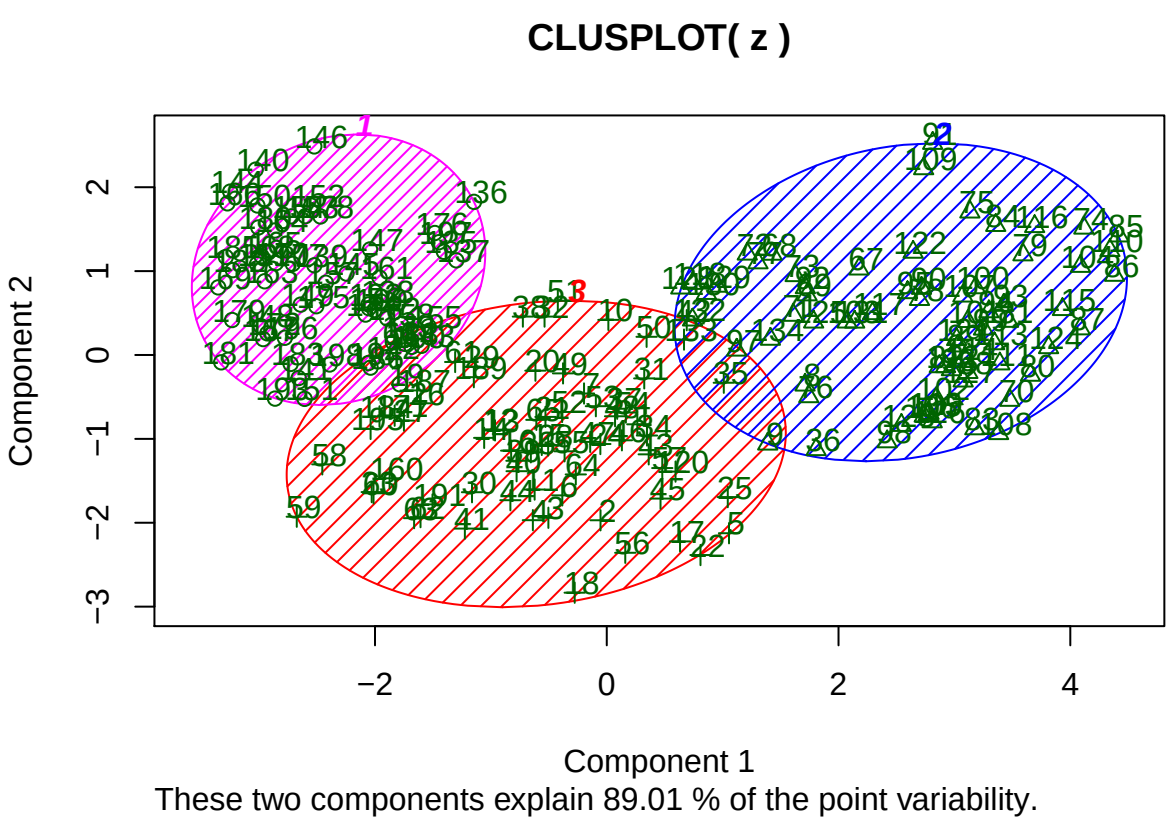
```
## [1] 0.9123902
```

```
adj.rand.index(ka2$cluster,as.numeric(my_data[,8]))
```

```
## [1] 0.8021194
```

The values of Rand and Adjusted Rand confirm that the optimal cluster size=3.

```
clusplot(z, ka2$cluster, color=TRUE, shade=TRUE,
   labels=2, lines=0)
```



##K-MENOIDS

A medoid can be defined as the point in the cluster, whose dissimilarities with all the other points in the cluster is minimum.

We initially find a good medoid size

```
library(fpc)
```

```
## Warning: package 'fpc' was built under R version 3.6.3
```

```
kmed <- pamk(z)
kmed$nc
```

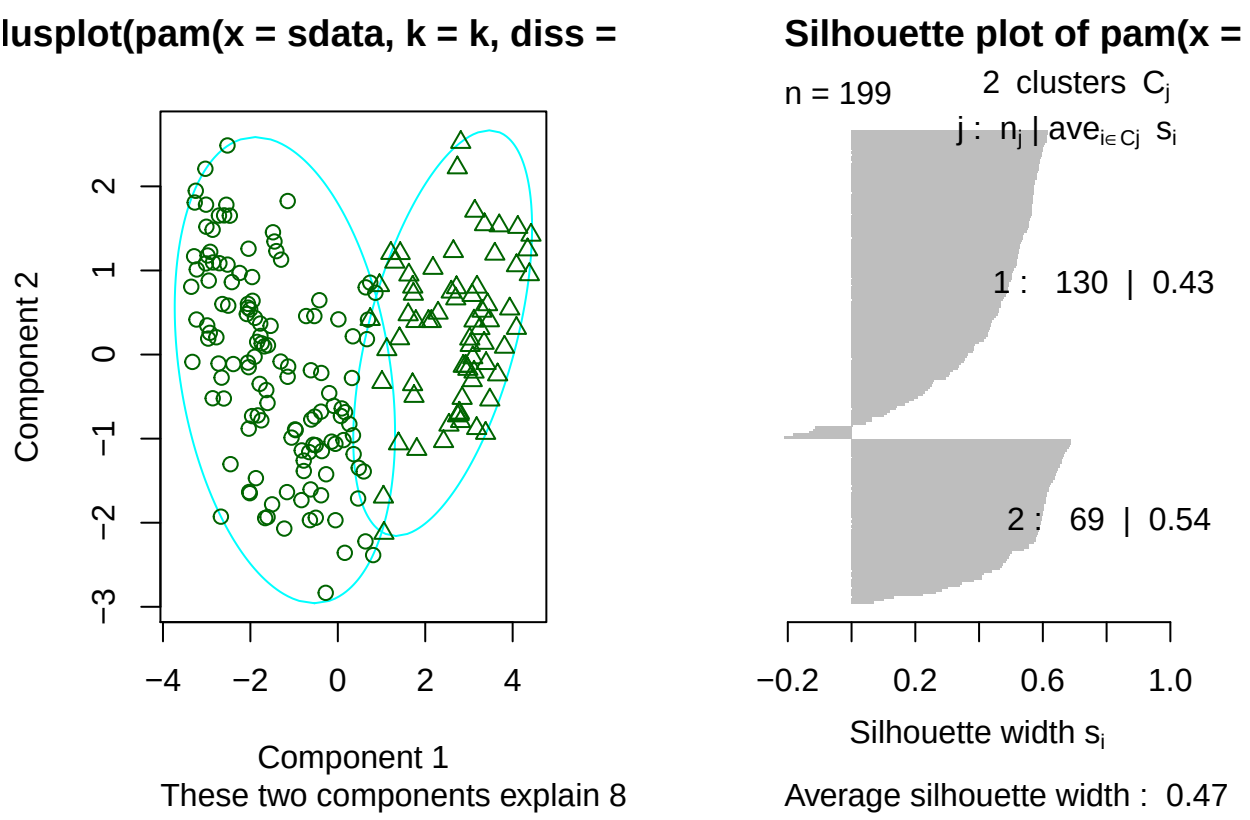
```
## [1] 2
```

##K-MENOIDS-2-K

```
table(kmed$pamobject$clustering,my_data$Seed.Group)
```

```
##
##      A  B  C
##   1 60  5 65
##   2  6 63  0
```

```
layout(matrix(c(1,2),1,2))
plot(kmed$pamobject)
```

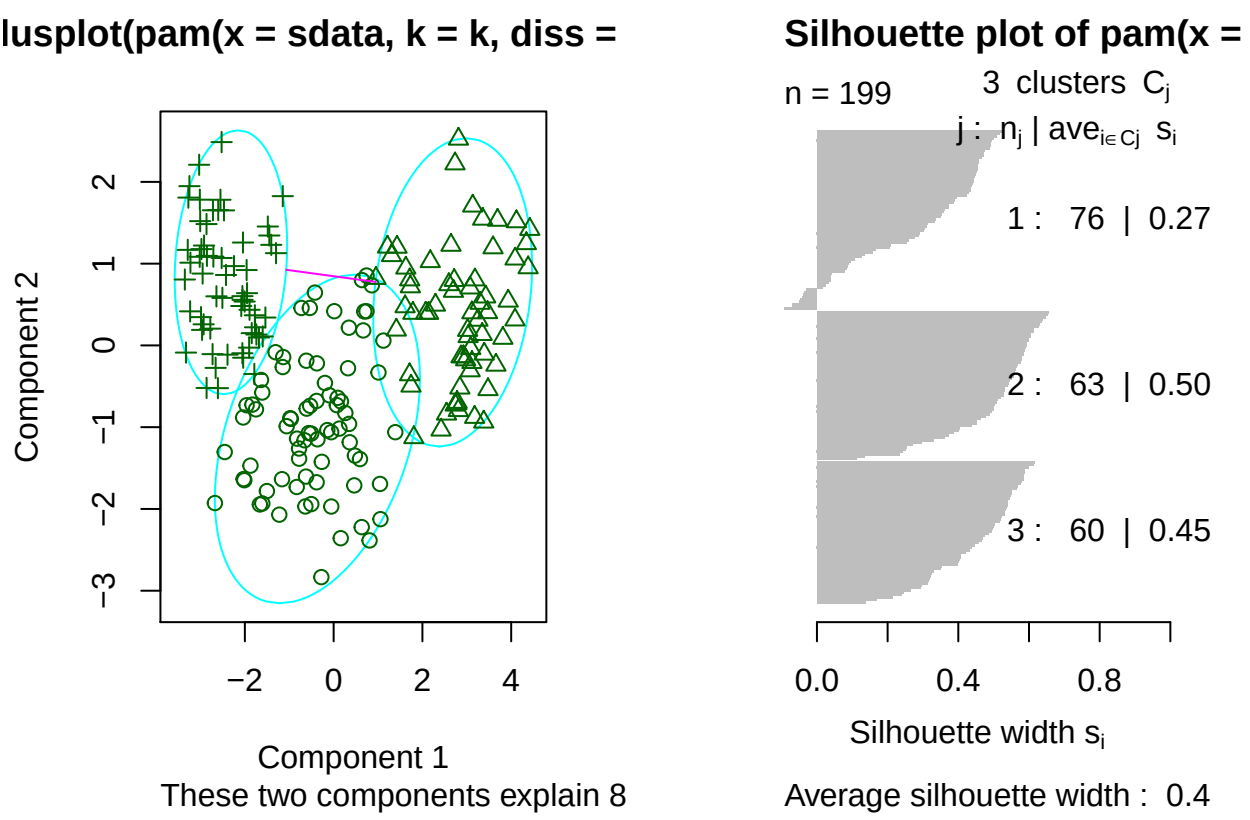


##K-MENOIDS-3-K

```
library(fpc)
kmed3 <- pamk(z,3)
table(kmed3$pamobject$clustering,my_data$Seed.Group)
```

```
##
##      A  B  C
##   1 62  7  7
##   2  2 61  0
##   3  2  0 58
```

```
layout(matrix(c(1,2),1,2))
plot(kmed3$pamobject)
```




```
library(bootcluster)
```

```
## Warning: package 'bootcluster' was built under R version 3.6.3
```

```
## Registered S3 method overwritten by 'sets':
##   method        from
##   print.element ggplot2
```

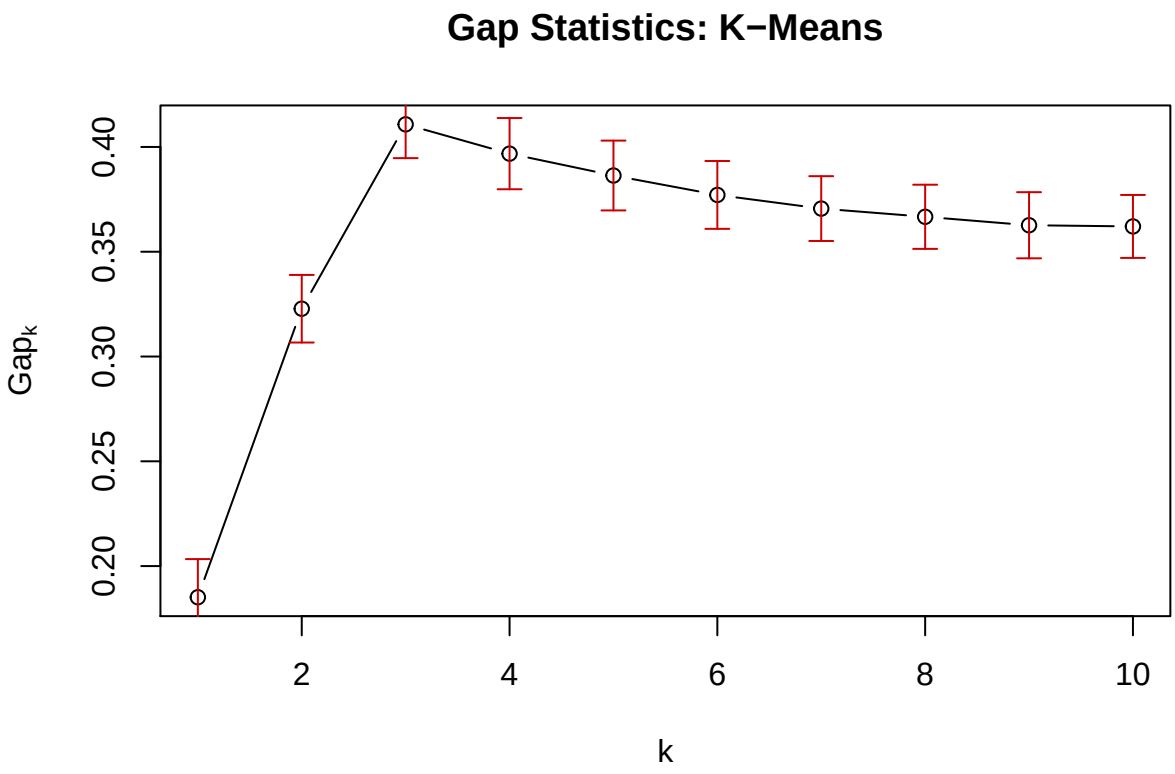
```
k.select(z,range=2:6,B=50,r=5,scheme_2=TRUE)
```

```
## $profile
##         2         3         4         5         6
## 0.9422927 0.8299274 0.5316617 0.4071184 0.2806451
##
## $k
## [1] 3
```

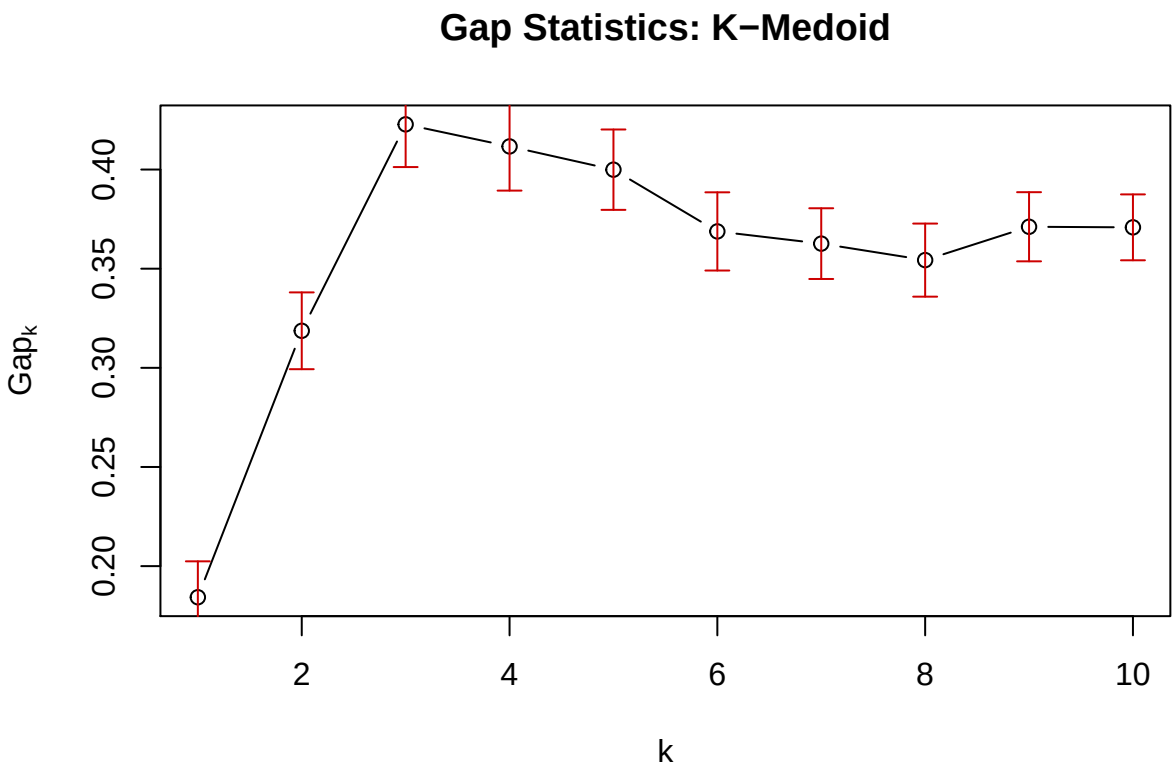
```
k.select(z,range=2:6,B=50,r=5,scheme_2=FALSE)
```

```
## $profile
##         2         3         4         5         6
## 0.9377294 0.8166799 0.6037502 0.3495511 0.2963327
##
## $k
## [1] 3
```

```
gap_kmeans <- clusGap(z,kmeans,nstart=20,K.max=10,B=100)
plot(gap_kmeans,main="Gap Statistics: K-Means")
```




```
gap_kmedoid <- clusGap(z,pam,K.max=10,B=100)
plot(gap_kmedoid,main="Gap Statistics: K-Medoid")
```



From the above methods we can coclude that in the above problem K-Means perform better than Heirarchical Clustering