Week10 R Code - Anly 506-51

Nitin

July 23, 2019

K-means Cluster Analysis

Required libraries or packages for this exercise.

```
library(tidyverse)  # data manipulation
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ------ tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.0       v purrr   0.2.5
## v tibble  2.1.3       v dplyr   0.8.0.1
## v tidyr   0.8.1       v stringr 1.3.1
## v readr   1.1.1       v forcats 0.3.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
## -- Conflicts --------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(cluster)    # clustering algorithms
```

```
## Warning: package 'cluster' was built under R version 3.5.3
```

```
library(factoextra) # clustering algorithms & visualization
```

```
## Warning: package 'factoextra' was built under R version 3.5.3
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

Data Preparation

There are three important steps in data preparation for cluster analysis in R

Step 1: Rows are observations (individuals) and columns are variables Step 2: Any missing value in the data must be removed or estimated. Step 3: The data must be standardized (i.e., scaled) to make variables comparable. Data set used for this exercise is built-in R data set of USArrests this data set contains statistics of arrests per 100,000 residents in USA. Read given data.

```
df <- USArrests
```

Remove missing value from the data.

```
df <- na.omit(df)
```

Use scale function to scaling/standardizing the data.

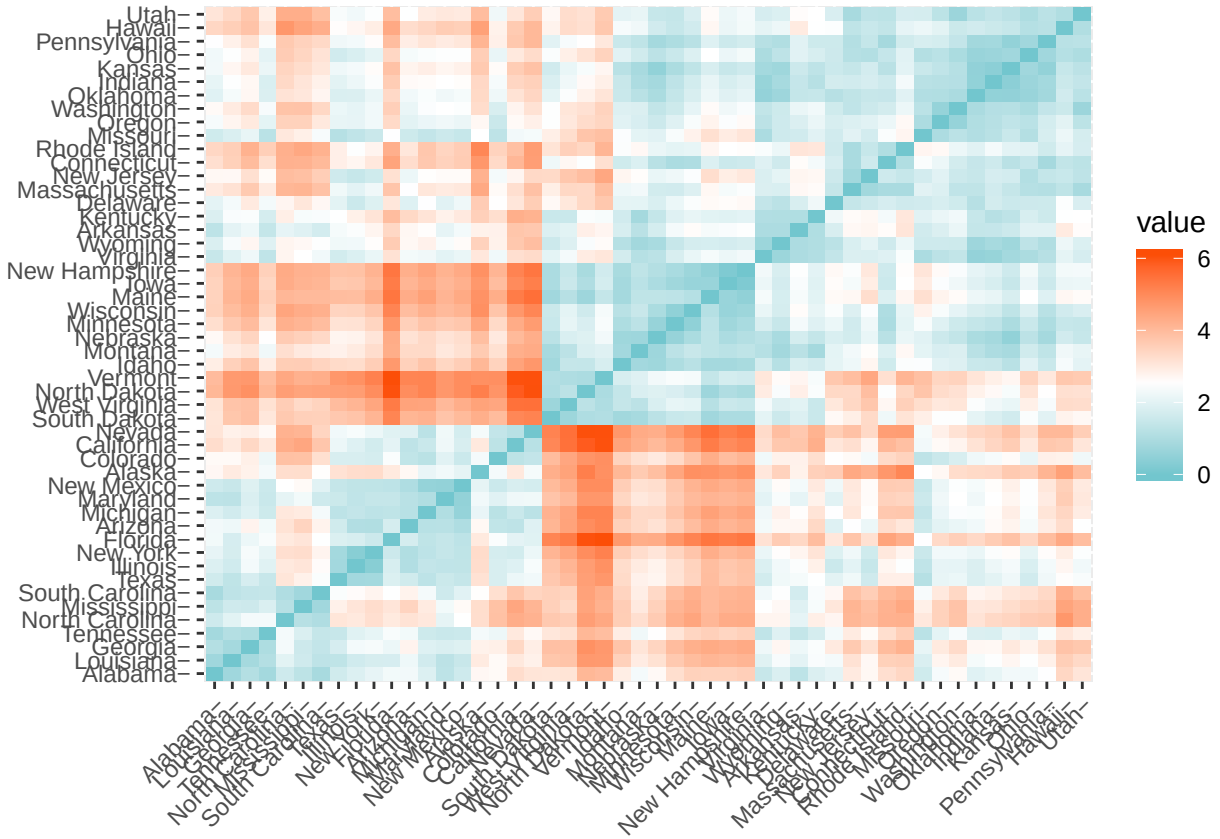
```
df <- scale(df)
head(df)
```

```
##                Murder   Assault   UrbanPop         Rape
## Alabama    1.24256408 0.7828393 -0.5209066 -0.003416473
## Alaska     0.50786248 1.1068225 -1.2117642  2.484202941
## Arizona    0.07163341 1.4788032  0.9989801  1.042878388
## Arkansas   0.23234938 0.2308680 -1.0735927 -0.184916602
## California 0.27826823 1.2628144  1.7589234  2.067820292
## Colorado   0.02571456 0.3988593  0.8608085  1.864967207
```

Clustering Distance Measures

Various methods for distance measures are Euclidean and Manhattan distances.

```
distance <- get_dist(df)
fviz_dist(distance, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"))
```



K-Means Clustering K-means Algorithm Plot two clusters by definging centers = 2

```
k2 <- kmeans(df, centers = 2, nstart = 25)
str(k2)
```

```
## List of 9
##  $ cluster     : Named int [1:50] 1 1 1 2 1 1 2 2 1 1 ...
##   ..- attr(*, "names")= chr [1:50] "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ centers     : num [1:2, 1:4] 1.005 -0.67 1.014 -0.676 0.198 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "1" "2"
##   .. ..$ : chr [1:4] "Murder" "Assault" "UrbanPop" "Rape"
##  $ totss       : num 196
```

```
##  $ withinss    : num [1:2] 46.7 56.1
##  $ tot.withinss: num 103
##  $ betweenss   : num 93.1
##  $ size        : int [1:2] 20 30
##  $ iter        : int 1
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

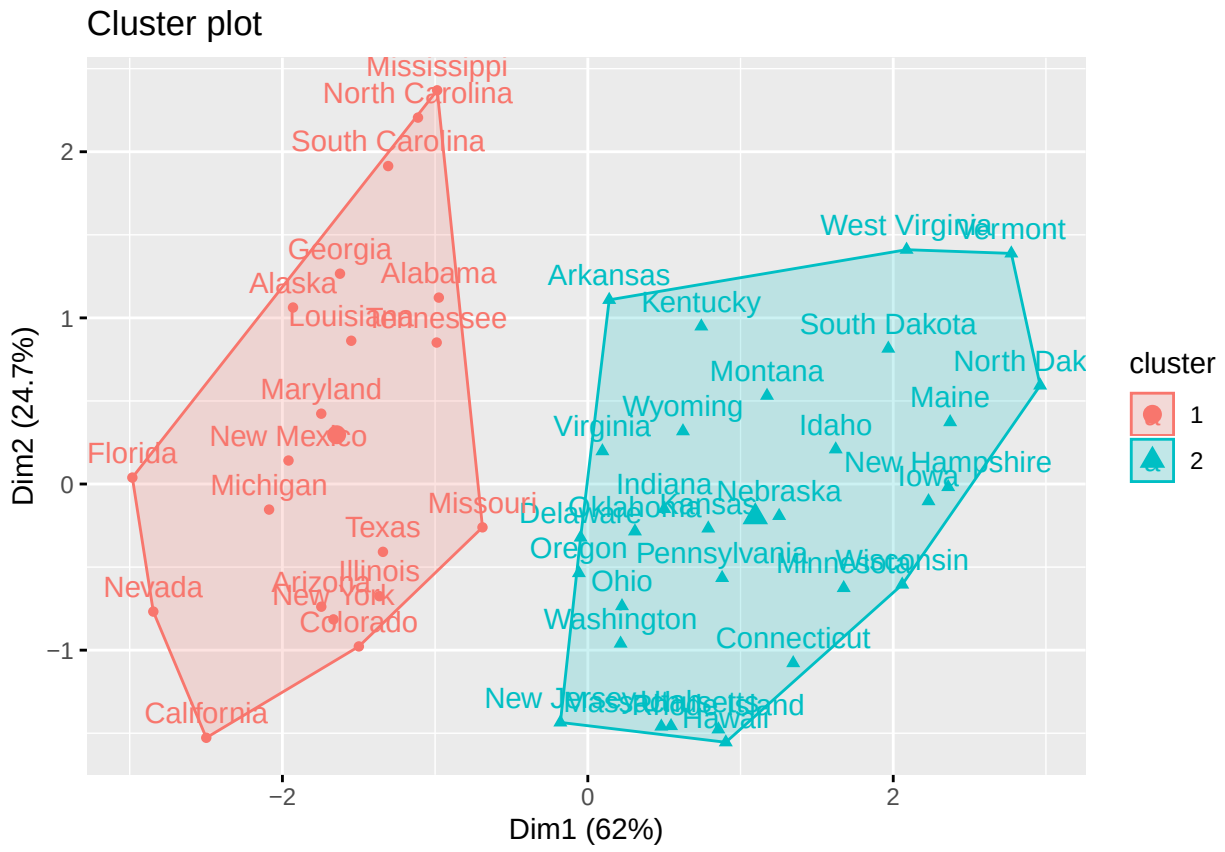
Print the results for k2

```
k2
```

```
## K-means clustering with 2 clusters of sizes 20, 30
##
## Cluster means:
##      Murder    Assault   UrbanPop       Rape
## 1  1.004934  1.0138274  0.1975853  0.8469650
## 2 -0.669956 -0.6758849 -0.1317235 -0.5646433
##
## Clustering vector:
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              1              1              2              1
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              1              2              2              1              1
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              2              2              1              2              2
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              2              2              1              2              1
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              2              1              2              1              1
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              2              2              1              2              2
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              1              1              1              2              2
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              2              2              2              2              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              2              1              1              2              2
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              2              2              2              2              2
##
## Within cluster sum of squares by cluster:
## [1] 46.74796 56.11445
##  (between_SS / total_SS =  47.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

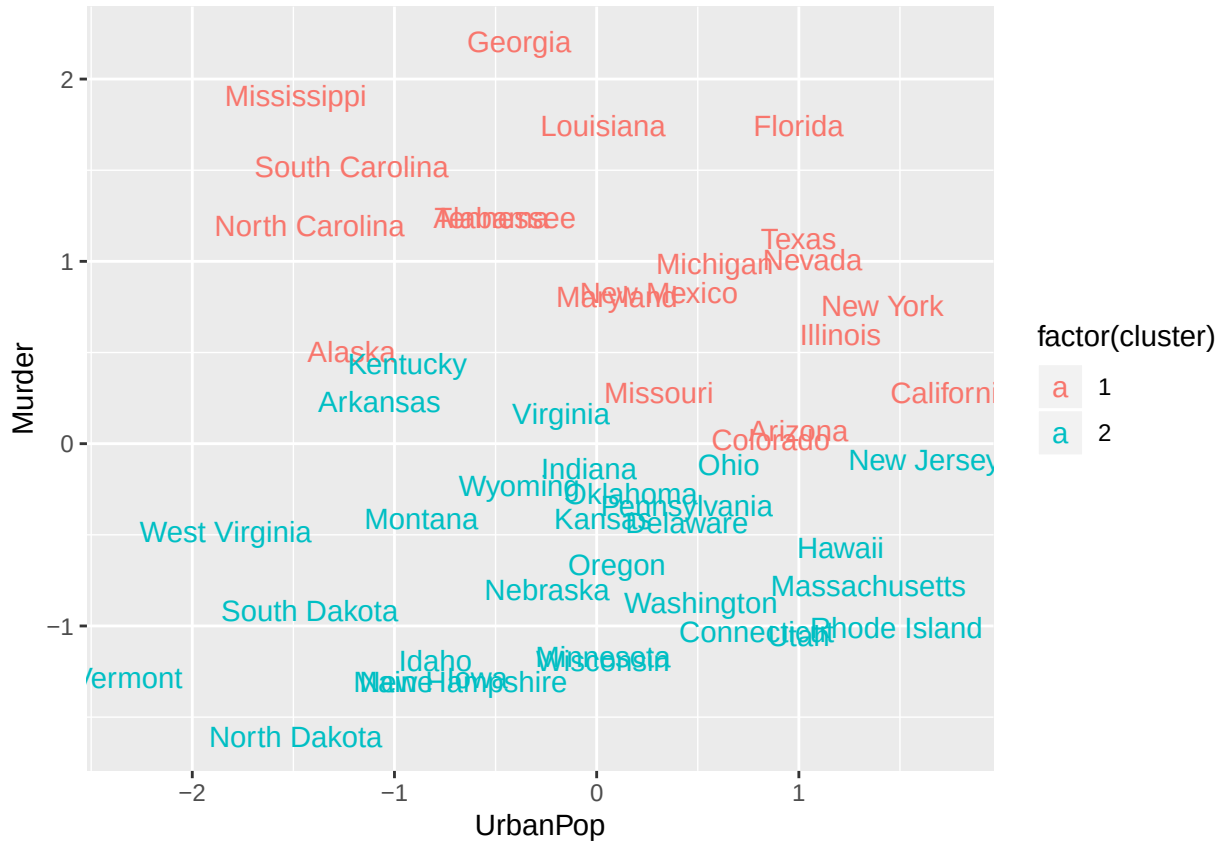
Use function fviz_cluster when there are more than two dimensions (variables) fviz_cluster will perform principal component analysis (PCA) and plot the data points.

```
fviz_cluster(k2, data = df)
```



Another way is standard pairwise scatter plots can be used show the clusters compared to the original variables.

```
df %>%
  as_tibble() %>%
  mutate(cluster = k2$cluster,
         state = row.names(USArrests)) %>%
  ggplot(aes(UrbanPop, Murder, color = factor(cluster), label = state)) +
  geom_text()
```



Execute the code for different number of cluster to see the diffrence.

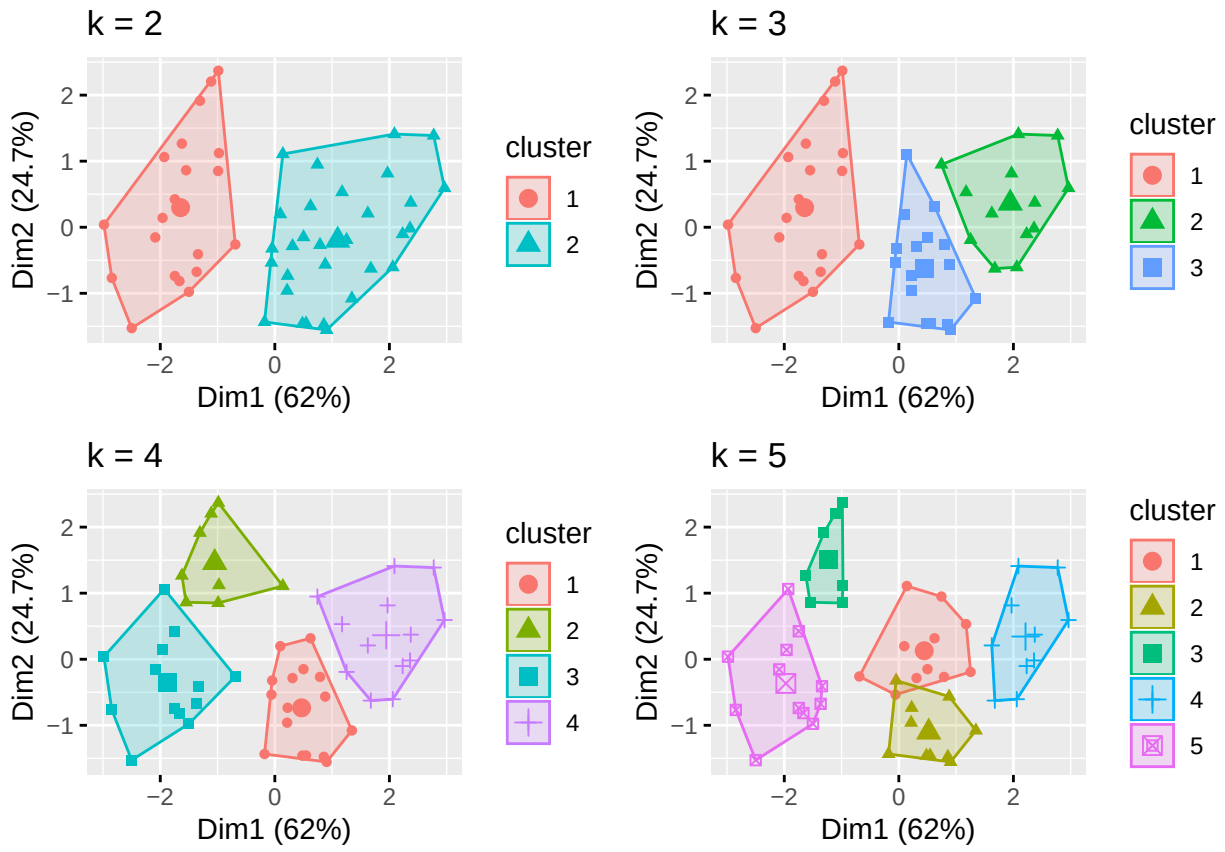
```
k3 <- kmeans(df, centers = 3, nstart = 25)
k4 <- kmeans(df, centers = 4, nstart = 25)
k5 <- kmeans(df, centers = 5, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = df) + ggtitle("k = 2")
p2 <- fviz_cluster(k3, geom = "point",  data = df) + ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = df) + ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = df) + ggtitle("k = 5")

library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 3.5.1
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

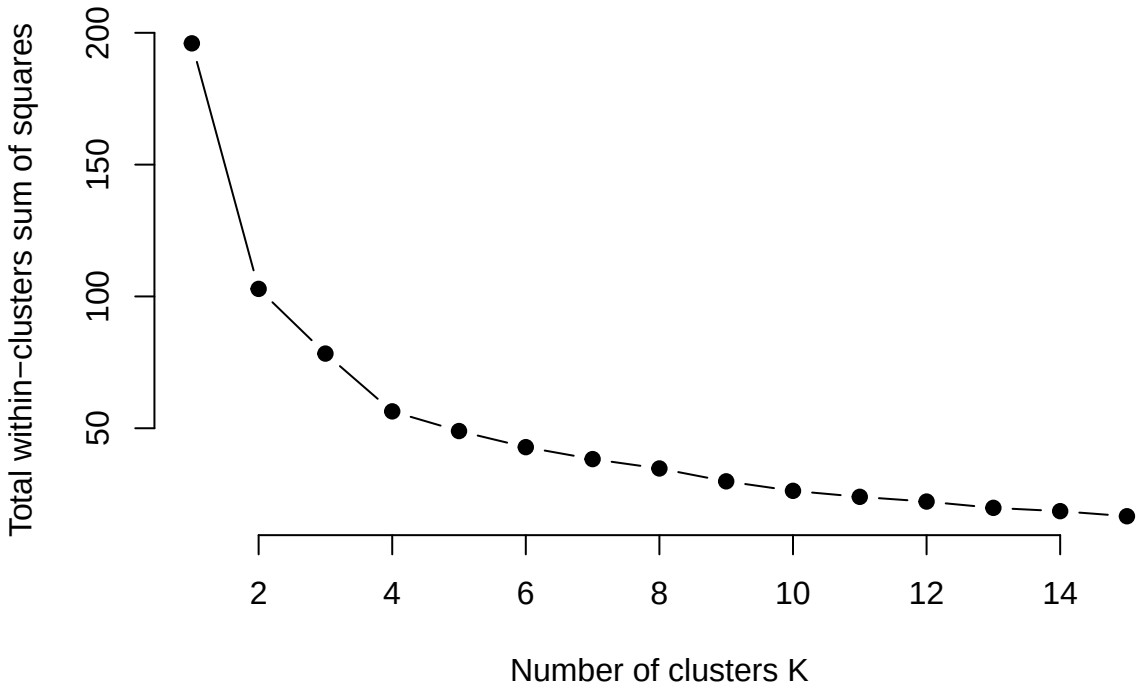
```
grid.arrange(p1, p2, p3, p4, nrow = 2)
```



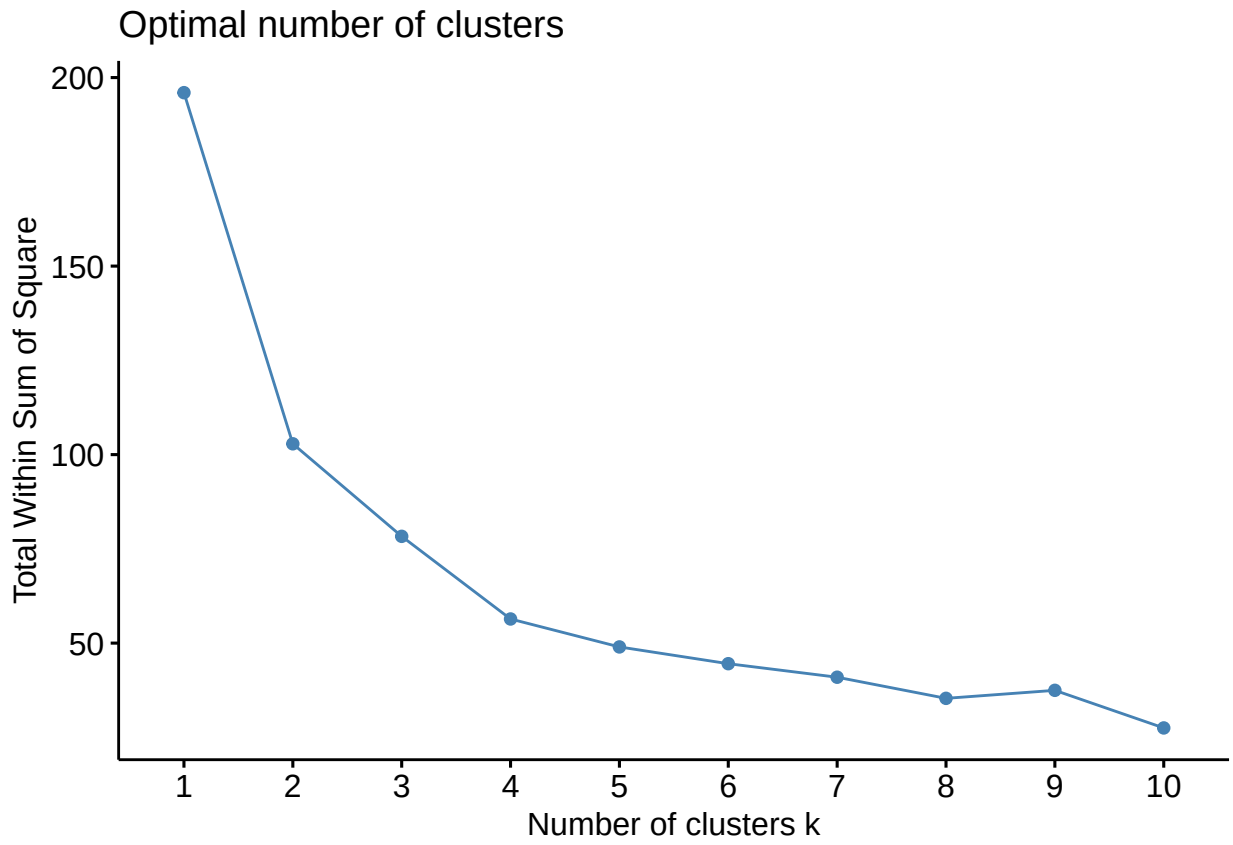
Determining Optimal Clusters

To determin optinal number of cluster we use three methods. 1. Elbow method 2. Silhouette method 3. Gap statistic Randomize data with set.seed(123) Use Elbow Method to calculate optinal number of cluster.

```
set.seed(123)

# function to compute total within-cluster sum of square
wss <- function(k) {
  kmeans(df, k, nstart = 10 )$tot.withinss
}

# Compute and plot wss for k = 1 to k = 15
k.values <- 1:15

# extract wss for 2-15 clusters
wss_values <- map_dbl(k.values, wss)

plot(k.values, wss_values,
       type="b", pch = 19, frame = FALSE,
       xlab="Number of clusters K",
       ylab="Total within-clusters sum of squares")
```

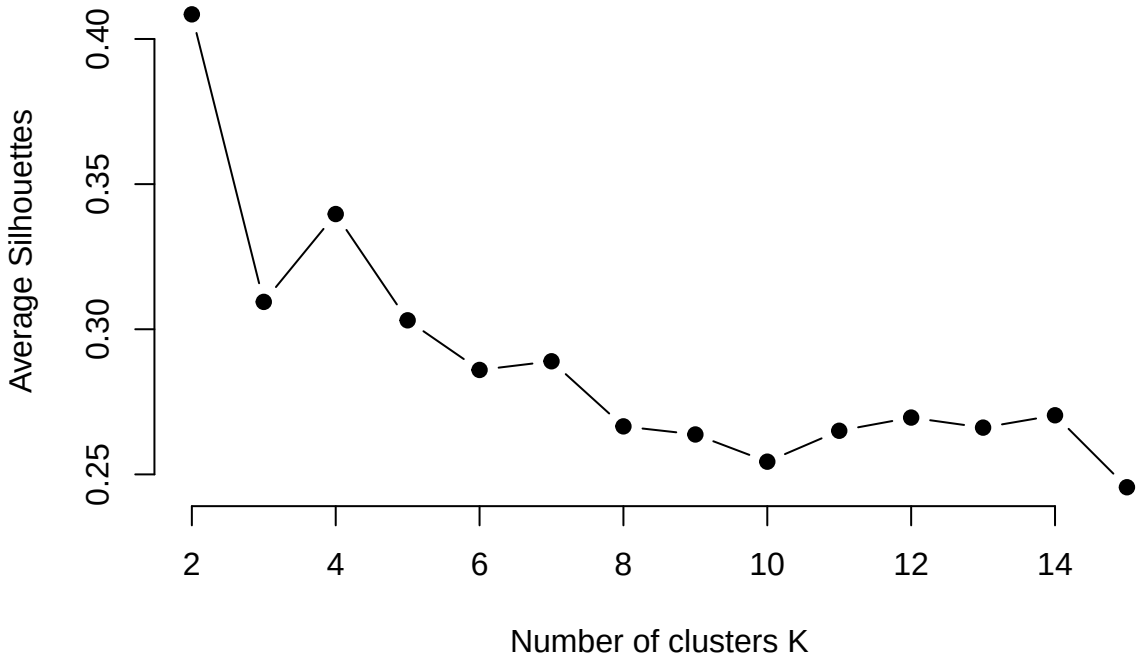



```
set.seed(123)

fviz_nbclust(df, kmeans, method = "wss")
```

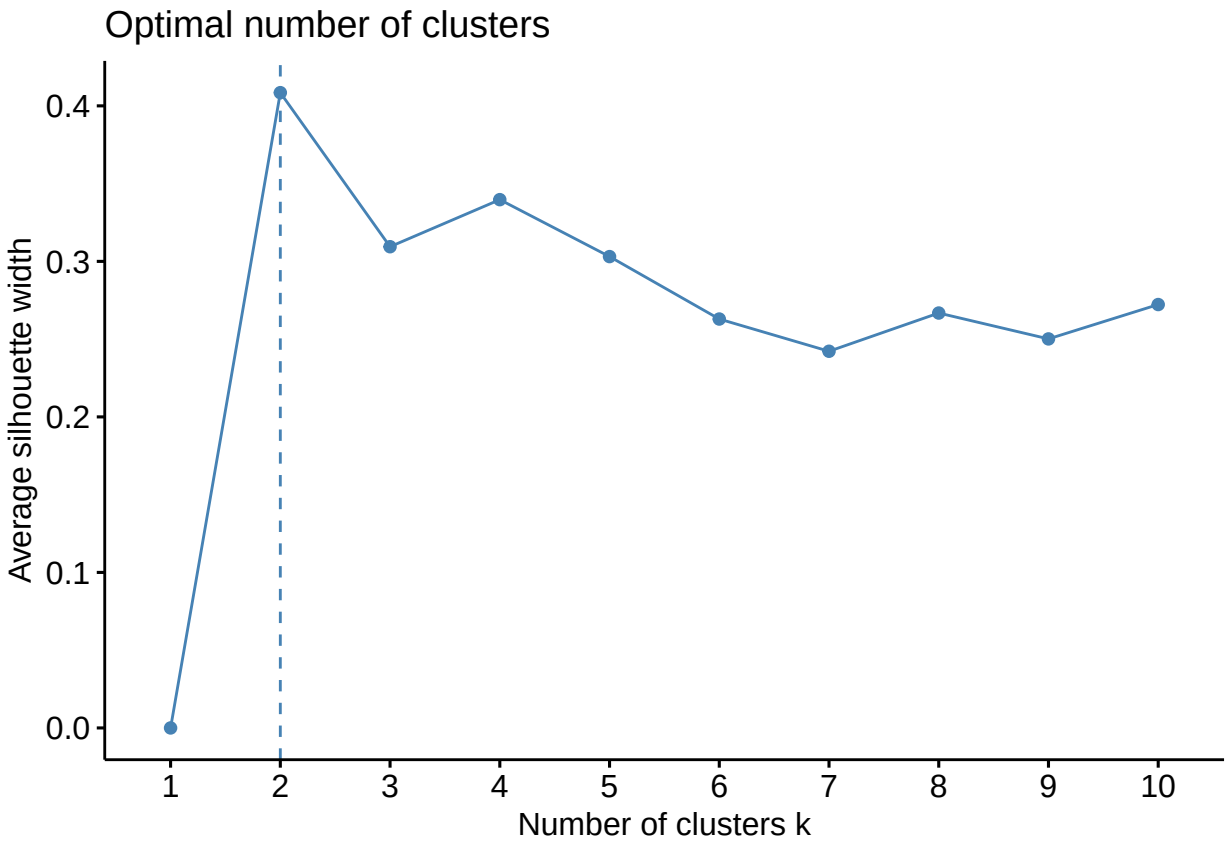


Use Silhouette method to calculate optinal number of cluster.

```
# function to compute average silhouette for k clusters
avg_sil <- function(k) {
  km.res <- kmeans(df, centers = k, nstart = 25)
  ss <- silhouette(km.res$cluster, dist(df))
  mean(ss[, 3])
}

# Compute and plot wss for k = 2 to k = 15
k.values <- 2:15

# extract avg silhouette for 2-15 clusters
avg_sil_values <- map_dbl(k.values, avg_sil)

plot(k.values, avg_sil_values,
       type = "b", pch = 19, frame = FALSE,
       xlab = "Number of clusters K",
       ylab = "Average Silhouettes")
```




```
fviz_nbclust(df, kmeans, method = "silhouette")
```



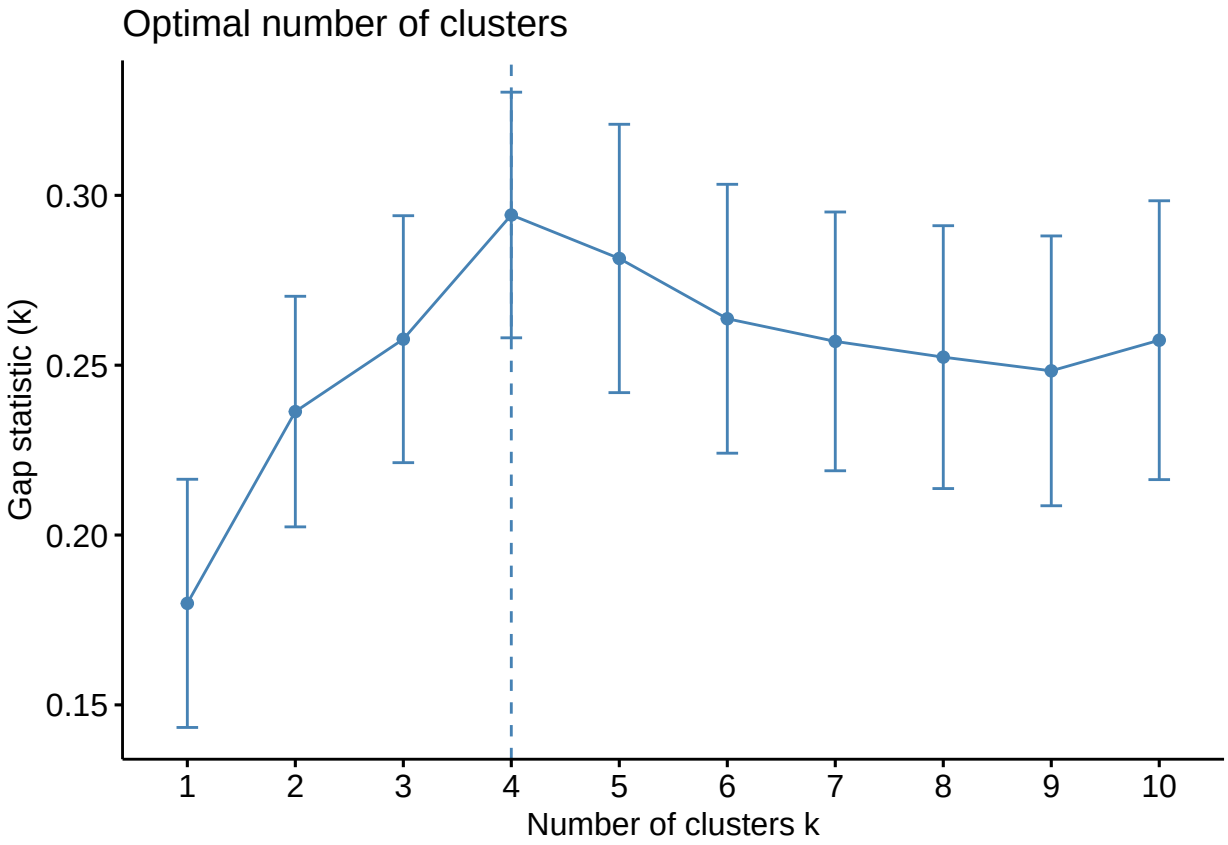
Use Gap statistic method to calculate optinal number of cluster.

```
# compute gap statistic
set.seed(123)
gap_stat <- clusGap(df, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
# Print the result
print(gap_stat, method = "firstmax")
```

```
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = df, FUNcluster = kmeans, K.max = 10, B = 50, nstart = 25)
## B=50 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
##  --> Number of clusters (method 'firstmax'): 4
##           logW   E.logW       gap     SE.sim
##  [1,] 3.458369 3.638250 0.1798804 0.03653200
##  [2,] 3.135112 3.371452 0.2363409 0.03394132
##  [3,] 2.977727 3.235385 0.2576588 0.03635372
##  [4,] 2.826221 3.120441 0.2942199 0.03615597
##  [5,] 2.738868 3.020288 0.2814197 0.03950085
##  [6,] 2.669860 2.933533 0.2636730 0.03957994
##  [7,] 2.598748 2.855759 0.2570109 0.03809451
##  [8,] 2.531626 2.784000 0.2523744 0.03869283
##  [9,] 2.468162 2.716498 0.2483355 0.03971815
## [10,] 2.394884 2.652241 0.2573567 0.04104674
```

Use fuction fviz_gap_stat to visualize the optimal number of clusters.

```
fviz_gap_stat(gap_stat)
```



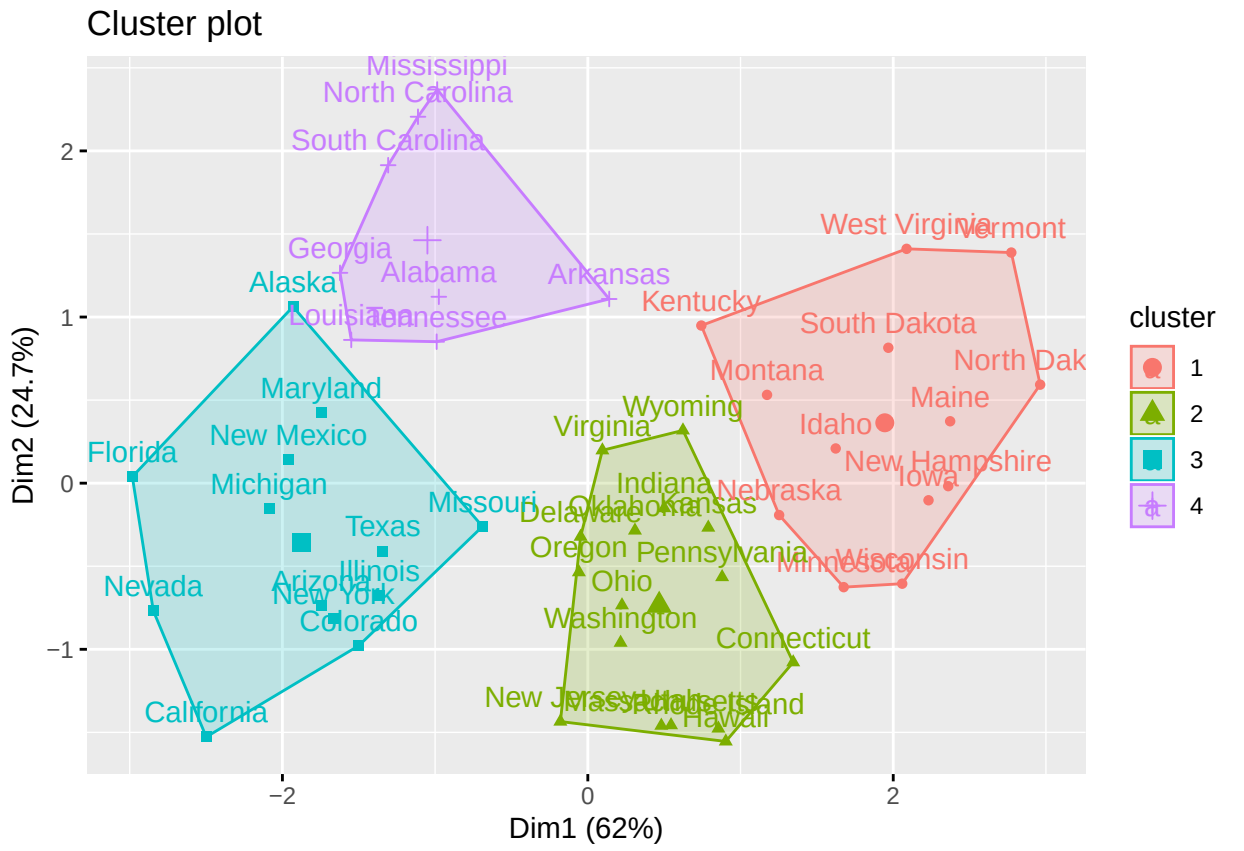

```
# Compute k-means clustering with k = 4
set.seed(123)
final <- kmeans(df, 4, nstart = 25)
print(final)
```

```
## K-means clustering with 4 clusters of sizes 13, 16, 13, 8
##
## Cluster means:
##       Murder    Assault   UrbanPop        Rape
## 1 -0.9615407 -1.1066010 -0.9301069 -0.96676331
## 2 -0.4894375 -0.3826001  0.5758298 -0.26165379
## 3  0.6950701  1.0394414  0.7226370  1.27693964
## 4  1.4118898  0.8743346 -0.8145211  0.01927104
##
## Clustering vector:
##        Alabama         Alaska        Arizona       Arkansas     California
##              4              3              3              4              3
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              3              2              2              3              4
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              2              1              3              2              1
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              2              1              4              1              3
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
```

```
##              2              3              1              4              3
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              1              1              3              1              2
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              3              3              4              1              2
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              2              2              2              2              4
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              1              4              3              2              1
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              2              2              1              1              2
##
## Within cluster sum of squares by cluster:
## [1] 11.952463 16.212213 19.922437  8.316061
##  (between_SS / total_SS =  71.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Use function fviz_cluster to visualize the results.

```
fviz_cluster(final, data = df)
```



Descriptive statistics at the cluster level can be done by extracting the cluster and adding to initial data.

```
USArrests %>%
  mutate(Cluster = final$cluster) %>%
  group_by(Cluster) %>%
  summarise_all("mean")
```

```
## # A tibble: 4 x 5
##   Cluster Murder Assault UrbanPop  Rape
##     <int>  <dbl>   <dbl>    <dbl> <dbl>
## 1       1   3.6     78.5     52.1  12.2
## 2       2   5.66   139.      73.9  18.8
## 3       3  10.8    257.      76    33.2
## 4       4  13.9    244.      53.8  21.4
```

Hierarchical Cluster Analysis

Hierarchical clustering can be divided into two main types first one is agglomerative and second one is divisive. Load required libraries and packages to these exercises.

```
library(tidyverse)  # data manipulation
library(cluster)    # clustering algorithms
library(factoextra) # clustering visualization
library(dendextend) # for comparing two dendrograms
```

```
## Warning: package 'dendextend' was built under R version 3.5.3
```

```
##
## ---------------------
## Welcome to dendextend version 1.12.0
## 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:stats':
##
##     cutree
```

Data Preparation

Read given data.

```
df <- USArrests
```

Remove missing data.

```
df <- na.omit(df)
```

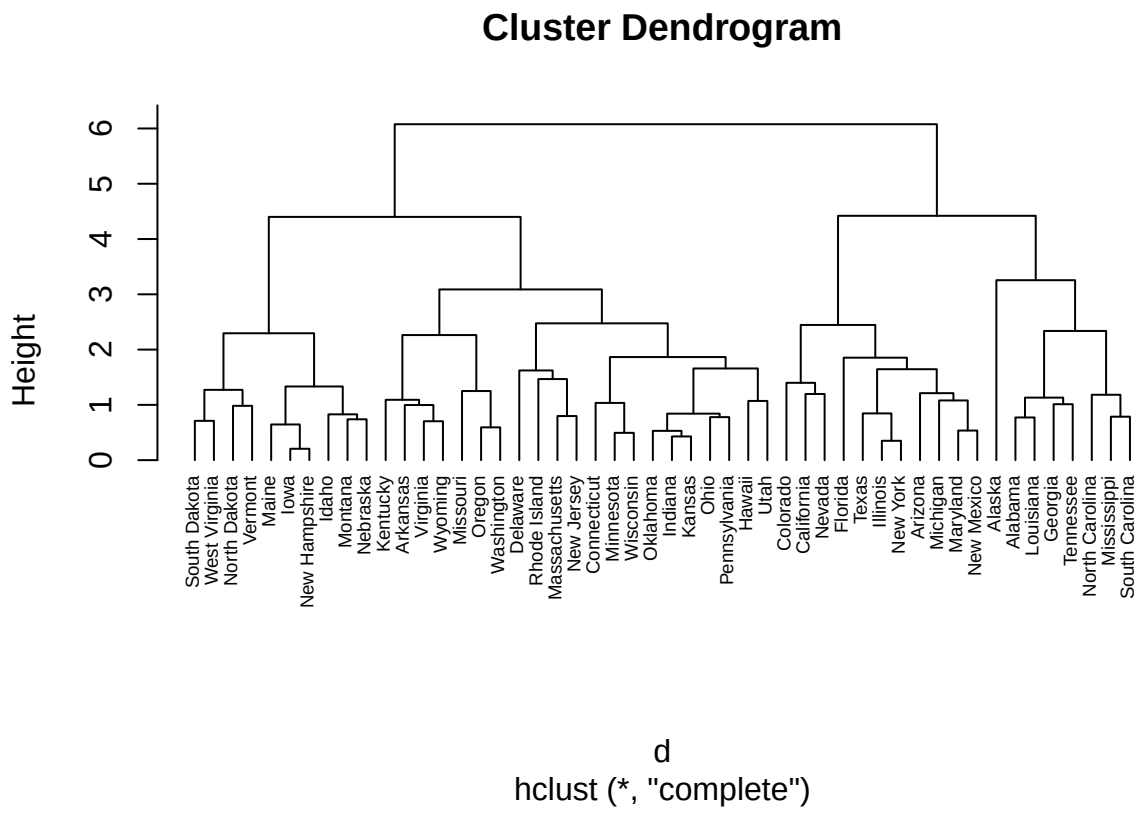
Use scale function to scaling/standardizing the data.

```r
df <- scale(df)
head(df)
```

```
##                Murder   Assault   UrbanPop         Rape
## Alabama    1.24256408 0.7828393 -0.5209066 -0.003416473
## Alaska     0.50786248 1.1068225 -1.2117642  2.484202941
## Arizona    0.07163341 1.4788032  0.9989801  1.042878388
## Arkansas   0.23234938 0.2308680 -1.0735927 -0.184916602
## California 0.27826823 1.2628144  1.7589234  2.067820292
## Colorado   0.02571456 0.3988593  0.8608085  1.864967207
```

Agglomerative Hierarchical Clustering

```r
# Dissimilarity matrix
d <- dist(df, method = "euclidean")

# Hierarchical clustering using Complete Linkage
hc1 <- hclust(d, method = "complete" )

# Plot the obtained dendrogram
plot(hc1, cex = 0.6, hang = -1)
```




```r
# Compute with agnes
hc2 <- agnes(df, method = "complete")

# Agglomerative coefficient
```

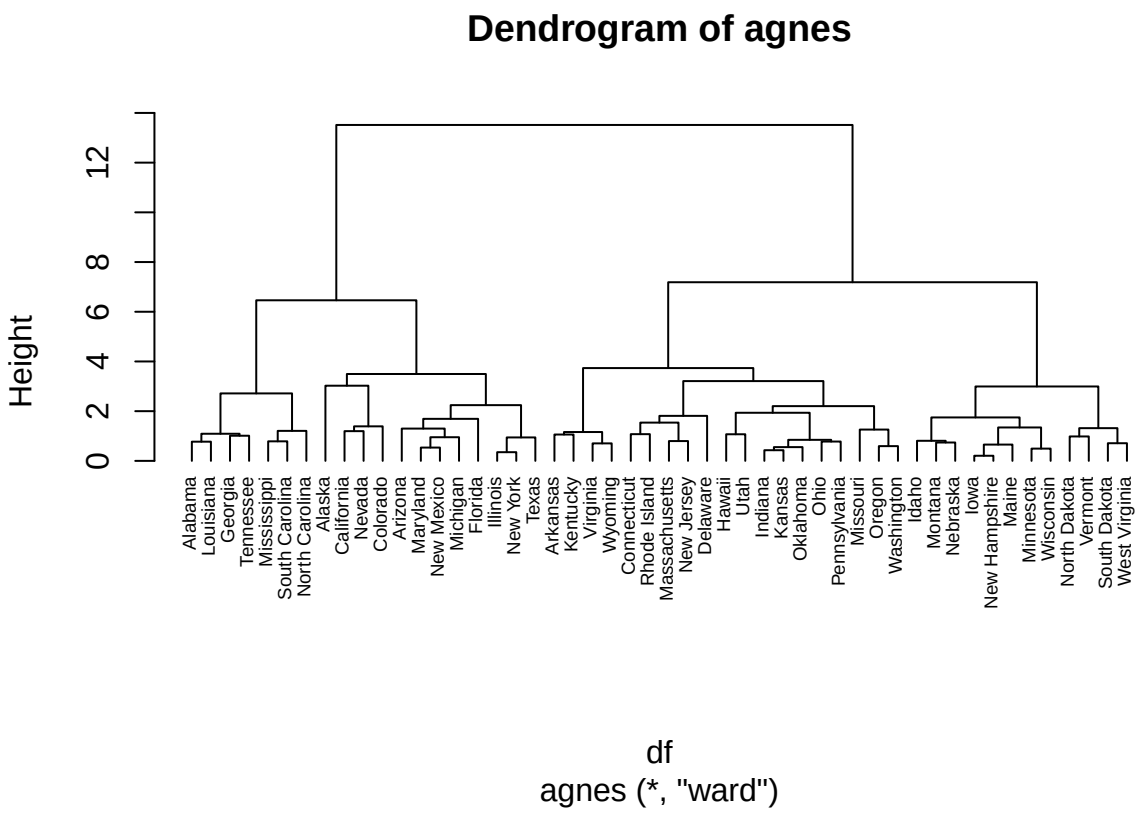
```
hc2$ac
```

```
## [1] 0.8531583
```

```
# methods to assess
m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")

# function to compute coefficient
ac <- function(x) {
  agnes(df, method = x)$ac
}

map_dbl(m, ac)
```

```
##   average    single  complete      ward
## 0.7379371 0.6276128 0.8531583 0.9346210
```

```
hc3 <- agnes(df, method = "ward")
pltree(hc3, cex = 0.6, hang = -1, main = "Dendrogram of agnes")
```



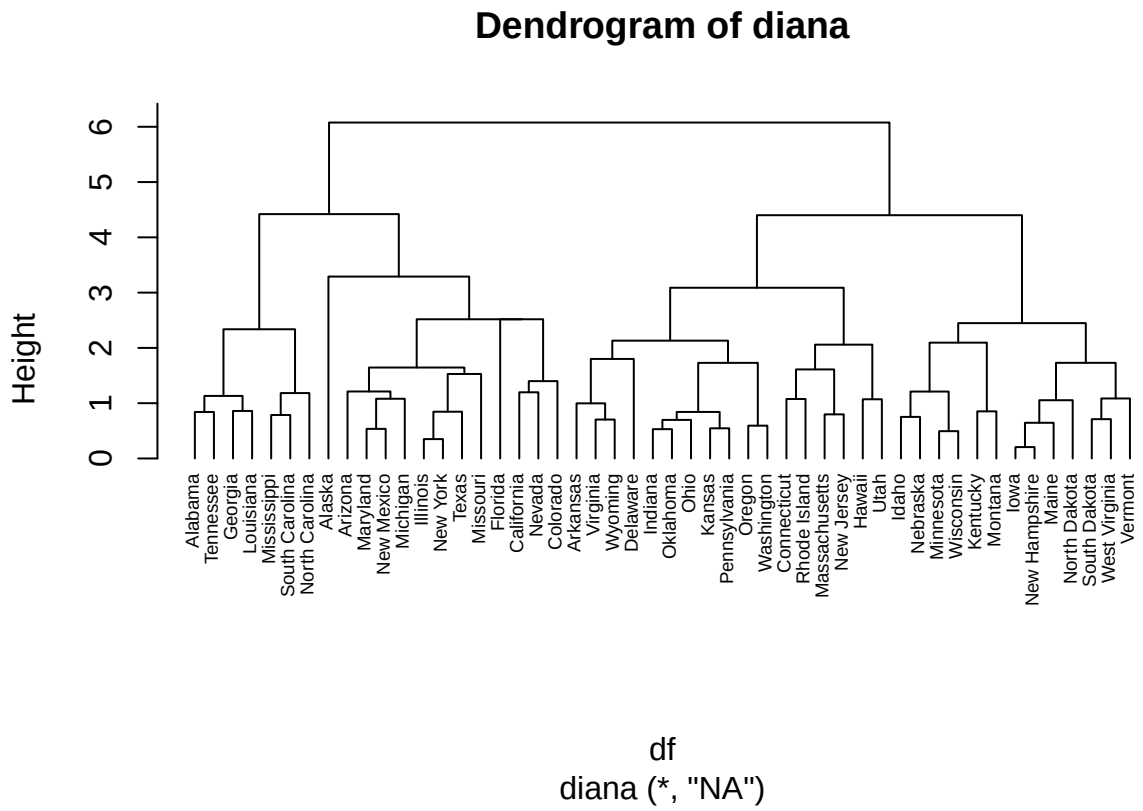
Divisive Hierarchical Clustering

```
# compute divisive hierarchical clustering
hc4 <- diana(df)

# Divise coefficient; amount of clustering structure found
```

```
hc4$dc
```

```
## [1] 0.8514345
```

```
## [1] 0.8514345

# plot dendrogram
pltree(hc4, cex = 0.6, hang = -1, main = "Dendrogram of diana")
```



Working with Dendrograms

```
# Ward's method
hc5 <- hclust(d, method = "ward.D2" )

# Cut tree into 4 groups
sub_grp <- cutree(hc5, k = 4)

# Number of members in each cluster
table(sub_grp)
```

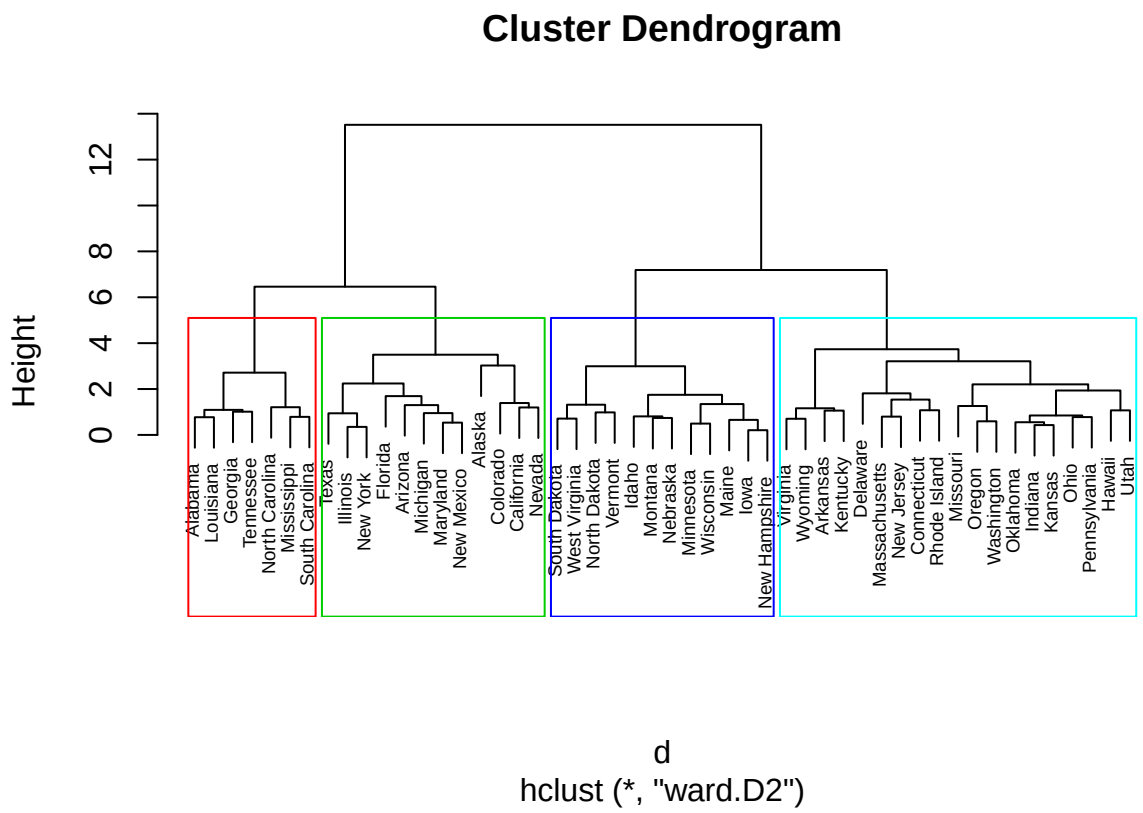
```
## sub_grp
##  1  2  3  4
##  7 12 19 12
```

```
USArrests %>%
  mutate(cluster = sub_grp) %>%
  head
```

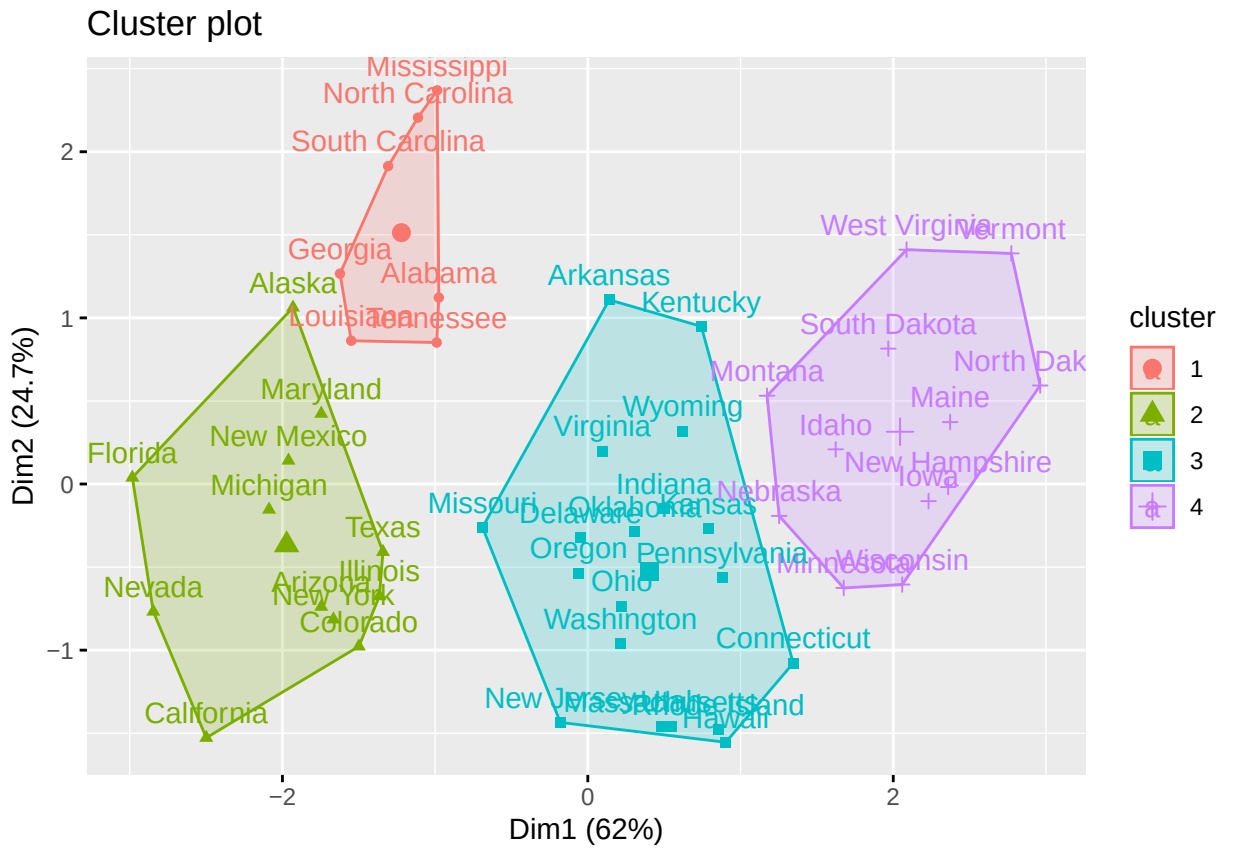
```
##   Murder Assault UrbanPop Rape cluster
```

```
## 1   13.2      236        58 21.2        1
## 2   10.0      263        48 44.5        2
## 3    8.1      294        80 31.0        2
## 4    8.8      190        50 19.5        3
## 5    9.0      276        91 40.6        2
## 6    7.9      204        78 38.7        2
```

```
plot(hc5, cex = 0.6)
rect.hclust(hc5, k = 4, border = 2:5)
```




```
fviz_cluster(list(data = df, cluster = sub_grp))
```




```
# Cut agnes() tree into 4 groups
hc_a <- agnes(df, method = "ward")
cutree(as.hclust(hc_a), k = 4)
```

```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              2              2              3              2
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              2              3              3              2              1
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              3              4              2              3              4
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              3              3              1              4              2
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              3              2              4              1              3
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              4              4              2              4              3
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              2              2              1              4              3
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              3              3              3              3              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              4              1              2              3              4
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              3              3              4              4              3
```

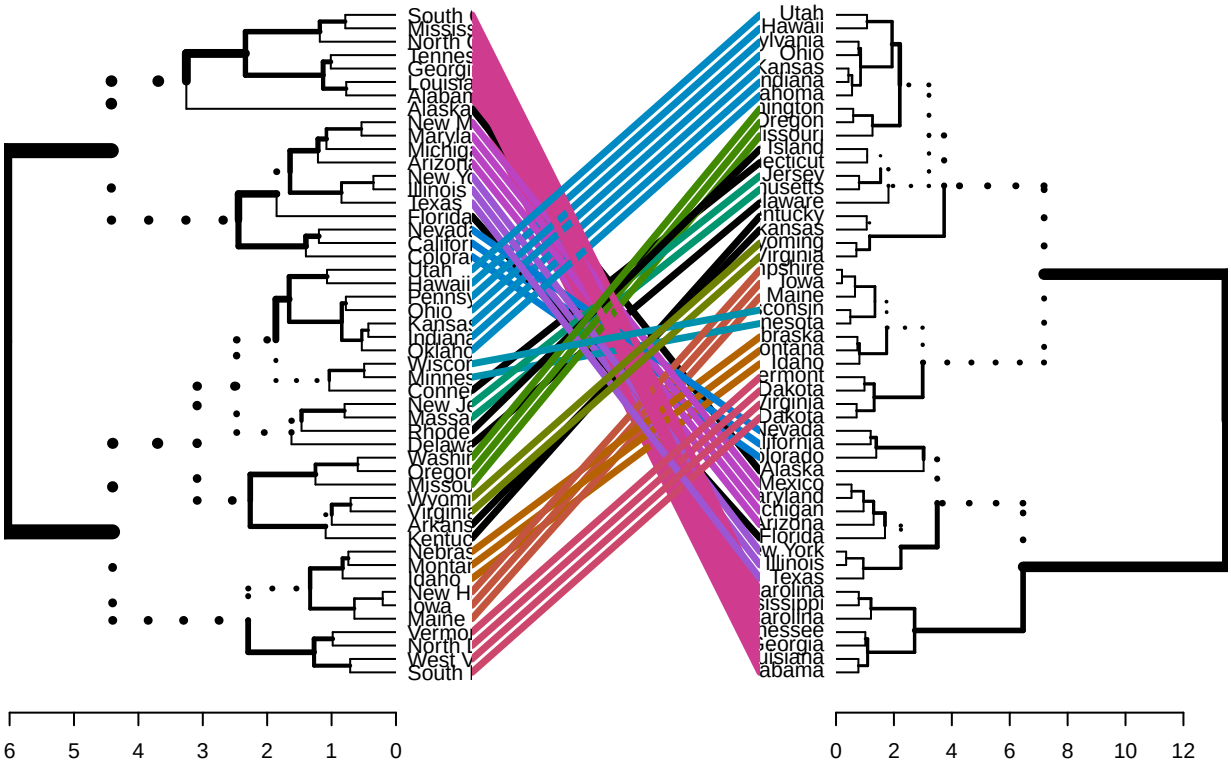
```
# Cut diana() tree into 4 groups
hc_d <- diana(df)
```

```
cutree(as.hclust(hc_d), k = 4)
```

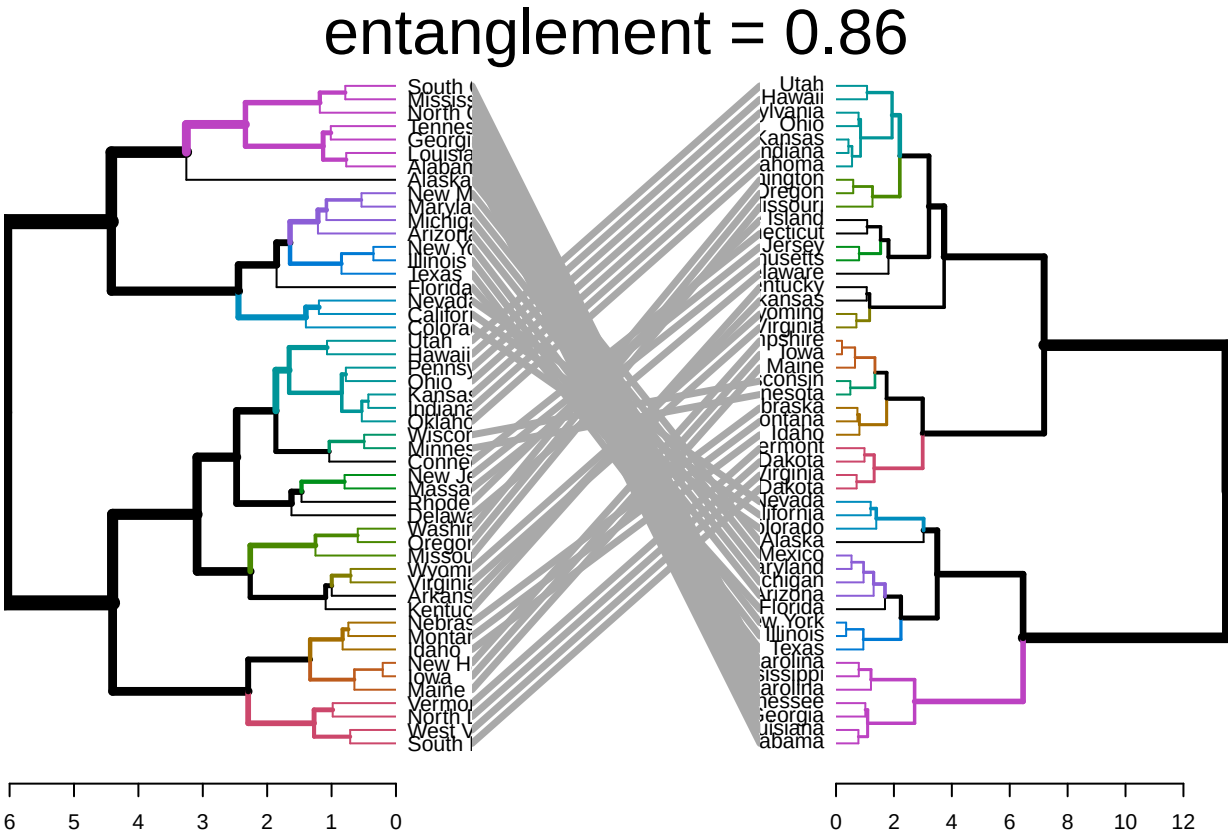
```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              2              2              3              2
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              2              3              3              2              1
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              3              4              2              3              4
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              3              4              1              4              2
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              3              2              4              1              2
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              4              4              2              4              3
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              2              2              1              4              3
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              3              3              3              3              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              4              1              2              3              4
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              3              3              4              4              3
```

Use function tanglegram to plot two dendrograms, side by side, with their labels connected by lines.

```
# Compute distance matrix
res.dist <- dist(df, method = "euclidean")

# Compute 2 hierarchical clusterings
hc1 <- hclust(res.dist, method = "complete")
hc2 <- hclust(res.dist, method = "ward.D2")

# Create two dendrograms
dend1 <- as.dendrogram (hc1)
dend2 <- as.dendrogram (hc2)
library(dendextend)

tanglegram(dend1, dend2)
```

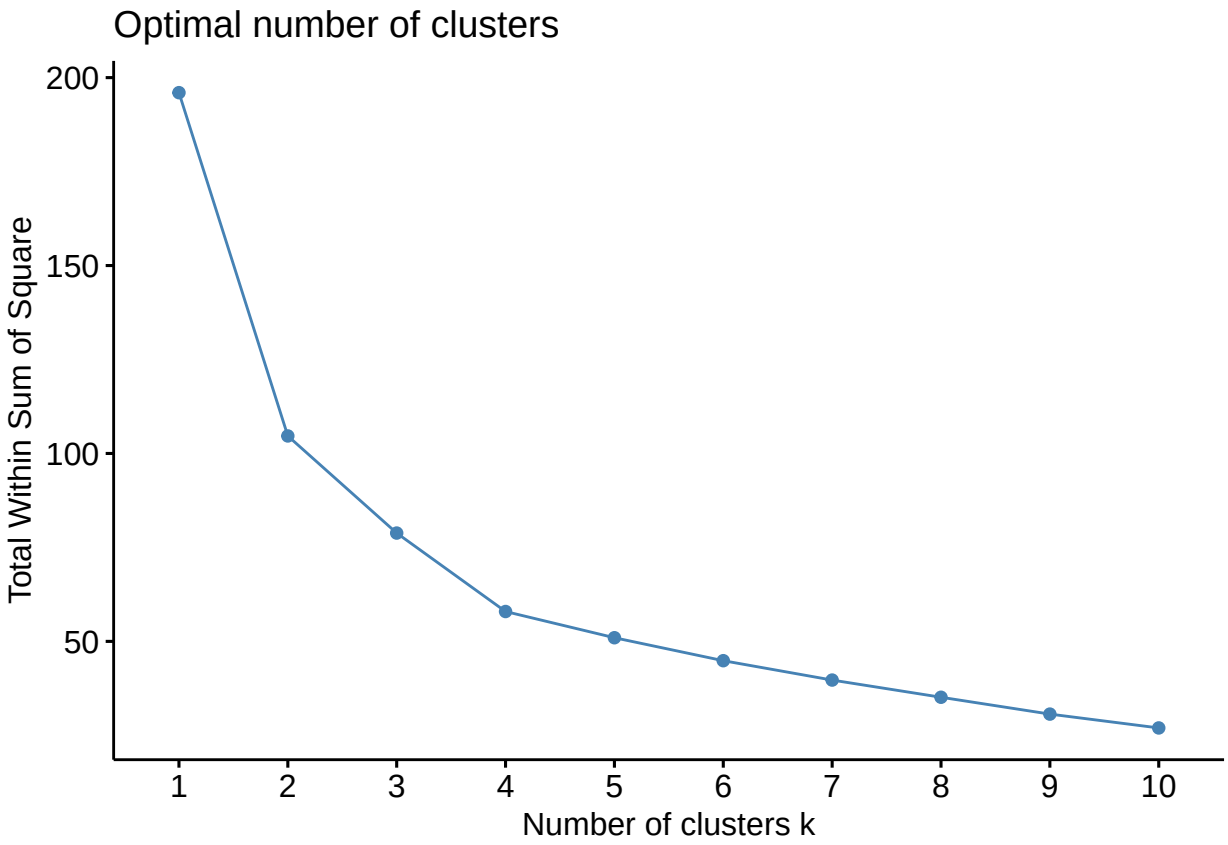



```
dend_list <- dendlist(dend1, dend2)

tanglegram(dend1, dend2,
  highlight_distinct_edges = FALSE, # Turn-off dashed lines
  common_subtrees_color_lines = FALSE, # Turn-off line colors
  common_subtrees_color_branches = TRUE, # Color common branches
  main = paste("entanglement =", round(entanglement(dend_list), 2))
  )
```



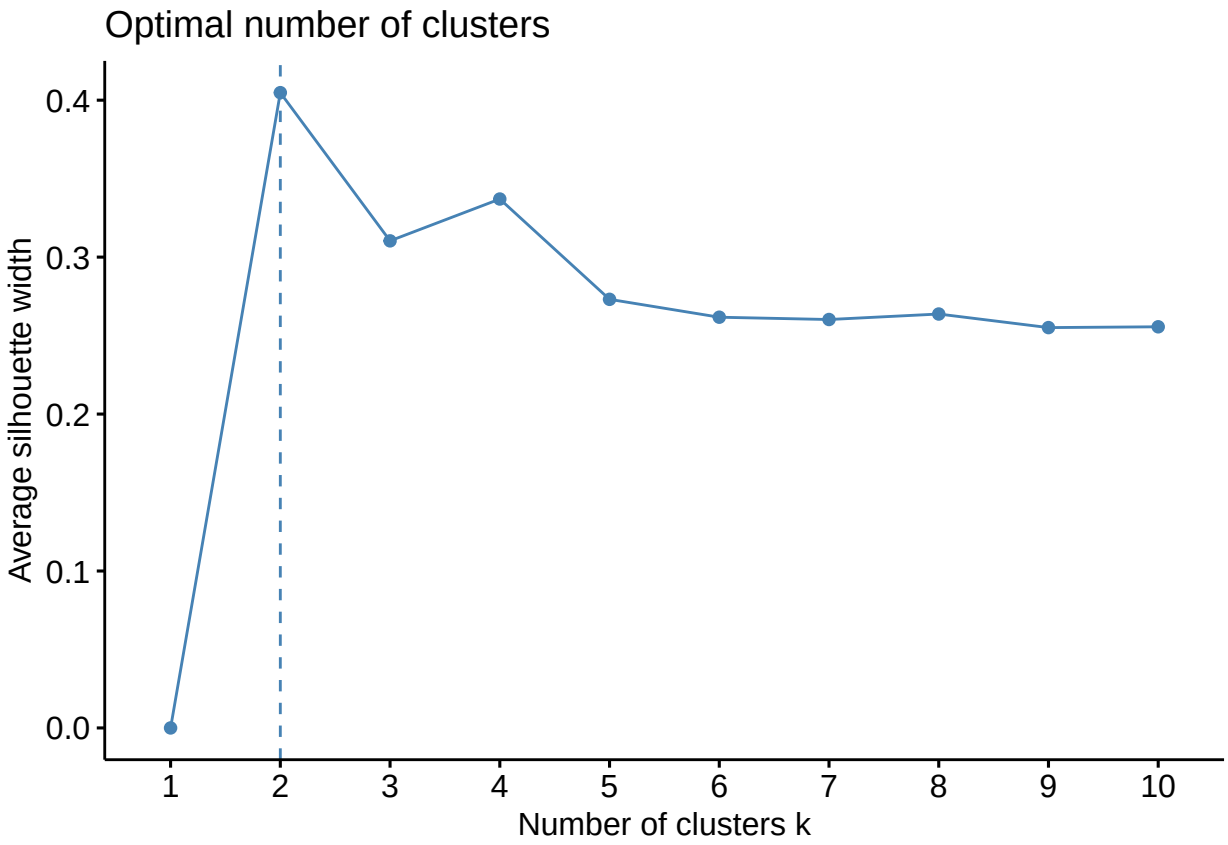
Determining Optimal Clusters Elbow Method

```
fviz_nbclust(df, FUN = hcut, method = "wss")
```



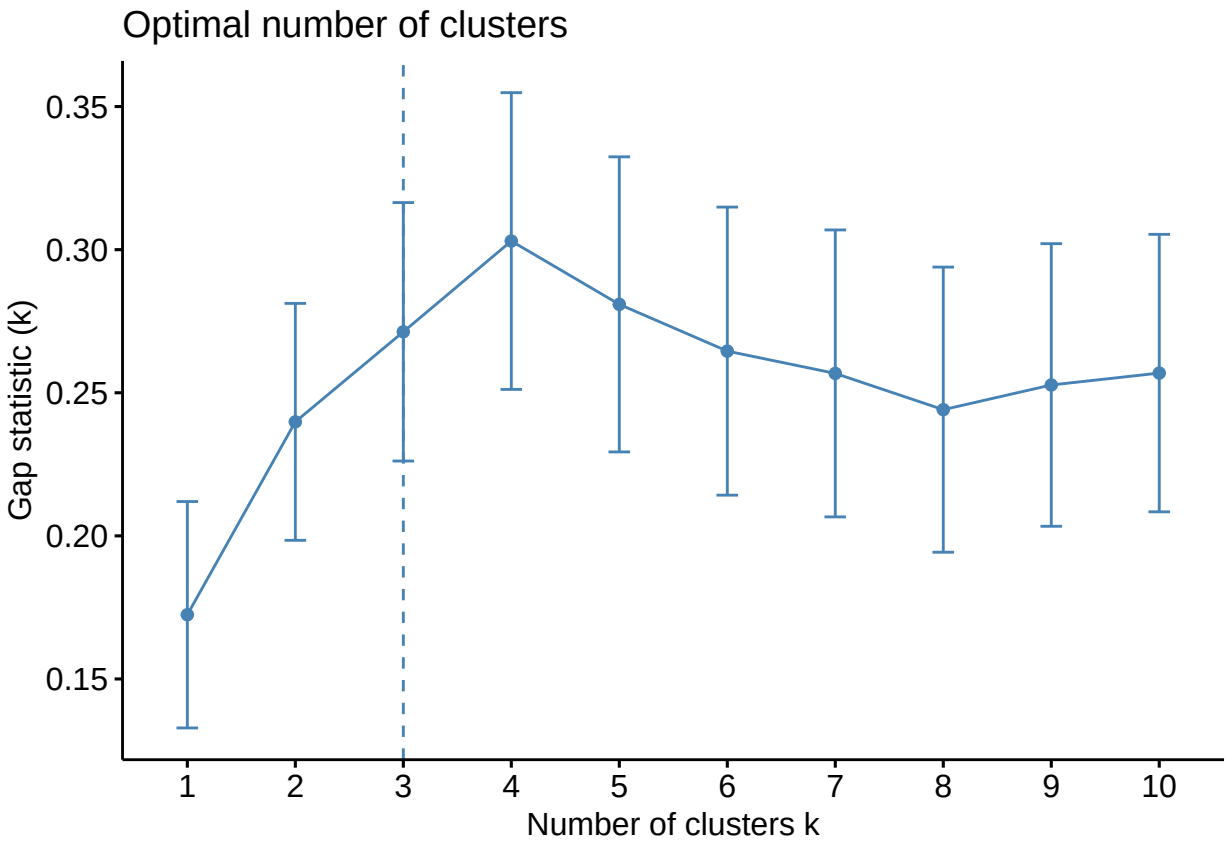
Average Silhouette Method

```
fviz_nbclust(df, FUN = hcut, method = "silhouette")
```



Gap Statistic Method

```
gap_stat <- clusGap(df, FUN = hcut, nstart = 25, K.max = 10, B = 50)
fviz_gap_stat(gap_stat)
```


Optimal number of clusters
Gap statistic (k)
0.35
0.30
0.25
0.20
0.15
1
2
3
4
5
6
7
8
9
10
Number of clusters k