# week10_kmeans_overall_sample

Yuhe

2022-11-29

```
tinytex::install_tinytex(force=TRUE)
```

```
## tlmgr install path
```

Helper packages

```
library(dplyr)       # for data manipulation
```

```
##
## 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
```

```
library(ggplot2)     # for data visualization
library(stringr)     # for string functionality
library(gridExtra)   # for manipulaiting the grid
```

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

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

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

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v tibble  3.1.8     v purrr   0.3.4
## v tidyr   1.2.1     v forcats 0.5.2
## v readr   2.1.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x dplyr::filter()      masks stats::filter()
## x dplyr::lag()         masks stats::lag()
```

```r
library(cluster)      # for general clustering algorithms
library(factoextra)   # for visualizing cluster results
```

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

```r
data("iris")
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
##                 
```

#To remove any missing value that might be present in the data, type this:

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

#we start by scaling/standardizing the data

```r
df <- scale(df[c(1:4)])
head(df)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1   -0.8976739  1.01560199    -1.335752   -1.311052
## 2   -1.1392005 -0.13153881    -1.335752   -1.311052
## 3   -1.3807271  0.32731751    -1.392399   -1.311052
## 4   -1.5014904  0.09788935    -1.279104   -1.311052
## 5   -1.0184372  1.24503015    -1.335752   -1.311052
## 6   -0.5353840  1.93331463    -1.165809   -1.048667
```

#start at 2 clusters

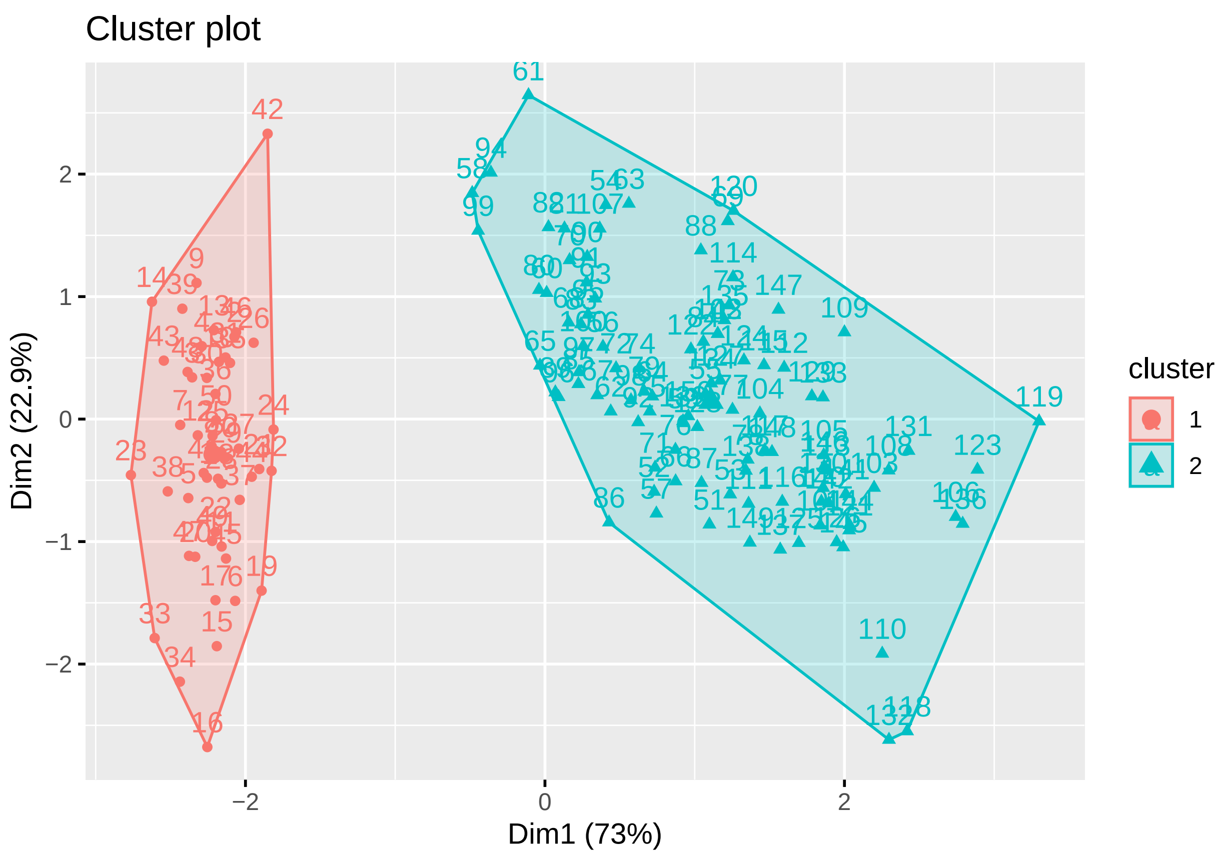
```r
k2 <- kmeans(df, centers = 2, nstart = 25)
str(k2)
```

```
## List of 9
##  $ cluster     : Named int [1:150] 1 1 1 1 1 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:150] "1" "2" "3" "4" ...
##  $ centers     : num [1:2, 1:4] -1.011 0.506 0.85 -0.425 -1.301 ...
##   ..- attr(*, "dimnames")=List of 2
```

```
##   .. ..$ : chr [1:2] "1" "2"
##   .. ..$ : chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"
##  $ totss       : num 596
##  $ withinss    : num [1:2] 47.4 173.5
##  $ tot.withinss: num 221
##  $ betweenss   : num 375
##  $ size        : int [1:2] 50 100
##  $ iter        : int 1
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

#plot the 2 clusters

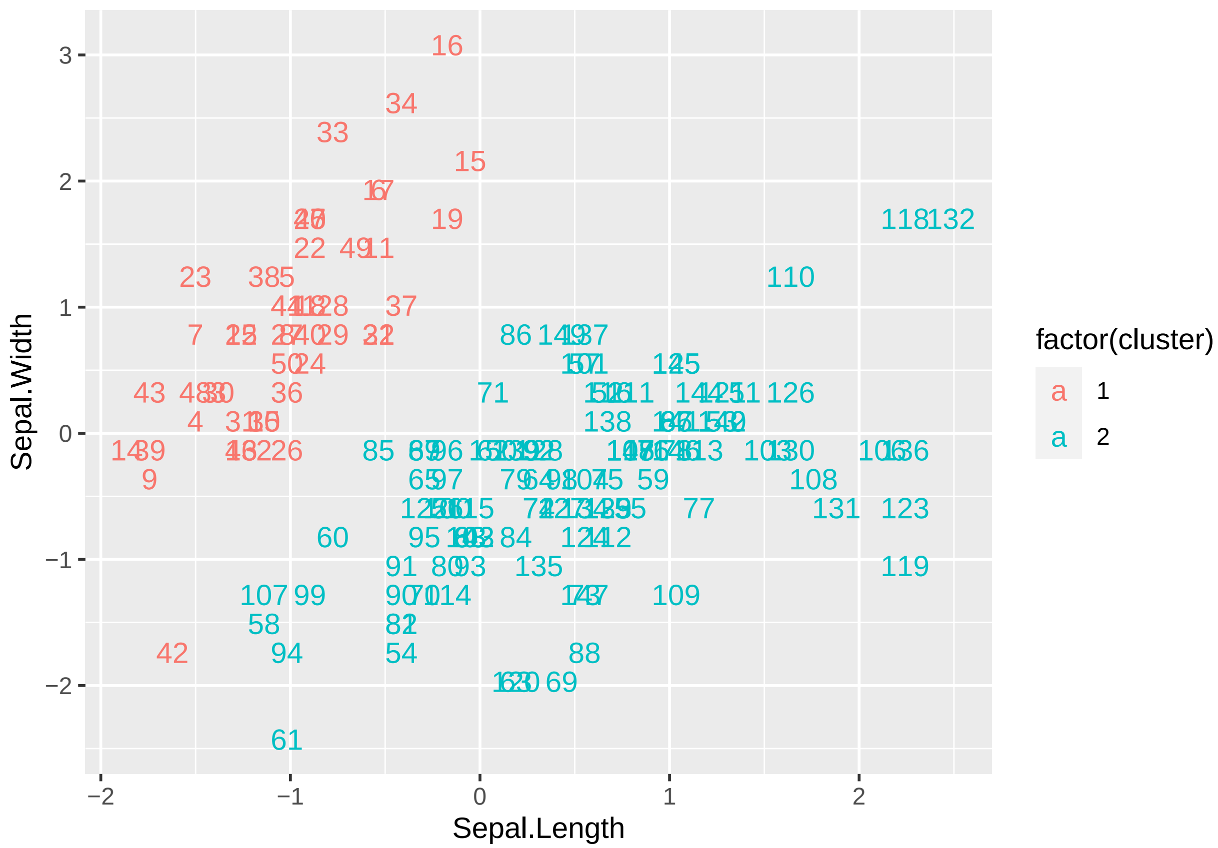
```
fviz_cluster(k2, data = df)
```




get the each clsuter's data

```
df %>%
  as_tibble() %>%
  mutate(cluster = k2$cluster,
         Species = row.names(iris)) %>%

  ggplot(aes(Sepal.Length, Sepal.Width, color = factor(cluster), label = Species)) +
  geom_text()
```

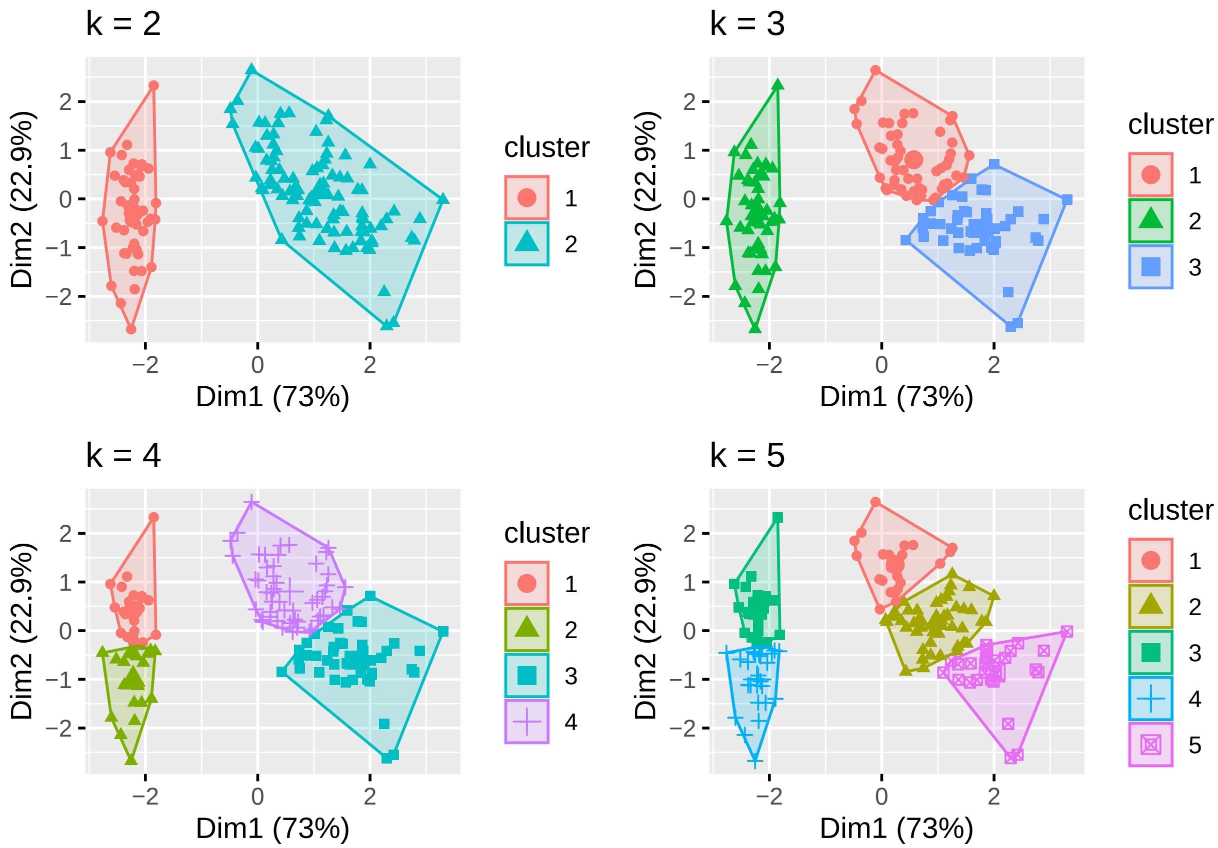



```
k3 <- kmeans(df, centers = 3, nstart = 25)
k4 <- kmeans(df, centers = 4, nstart = 25)
k5 <- kmeans(df, centers = 5, nstart = 25)
```

## 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")

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




#Determining Optimal Number of Clusters

```
set.seed(123)
```

#function to compute total within-cluster sum of square

```
wss <- function(k) {
  kmeans(df, k, nstart = 10 )$tot.withinss
}
```

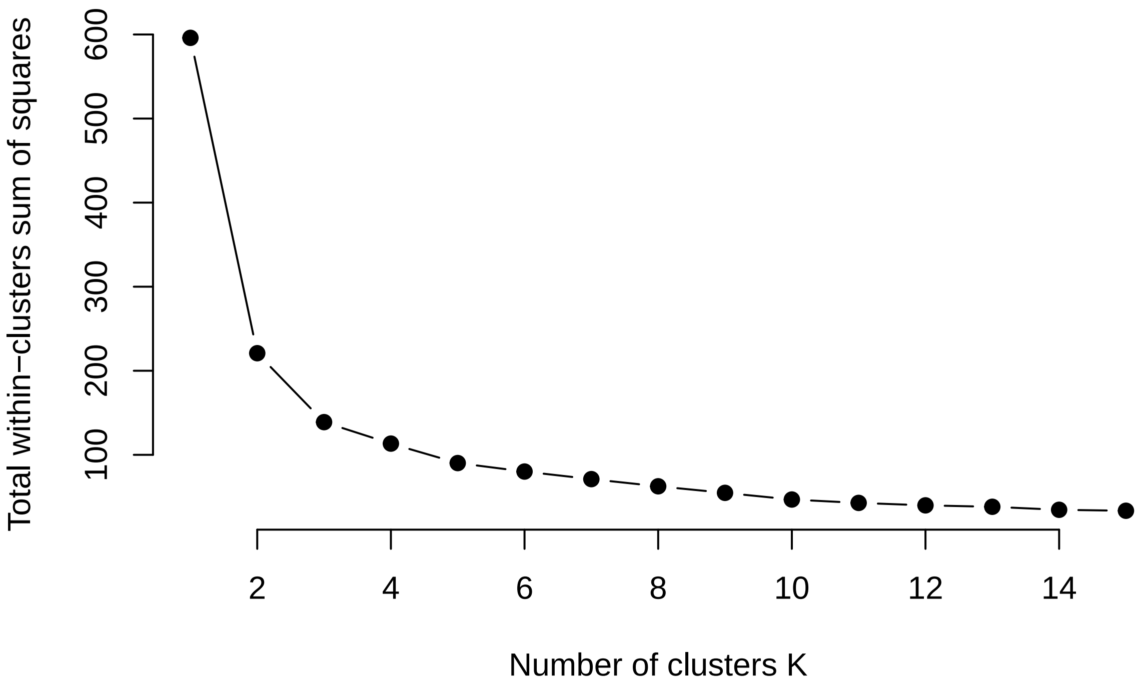
# Compute and plot wss for k = 1 to k = 15

```
k.values <- 1:15
```

# extract wss for 2-15 clusters

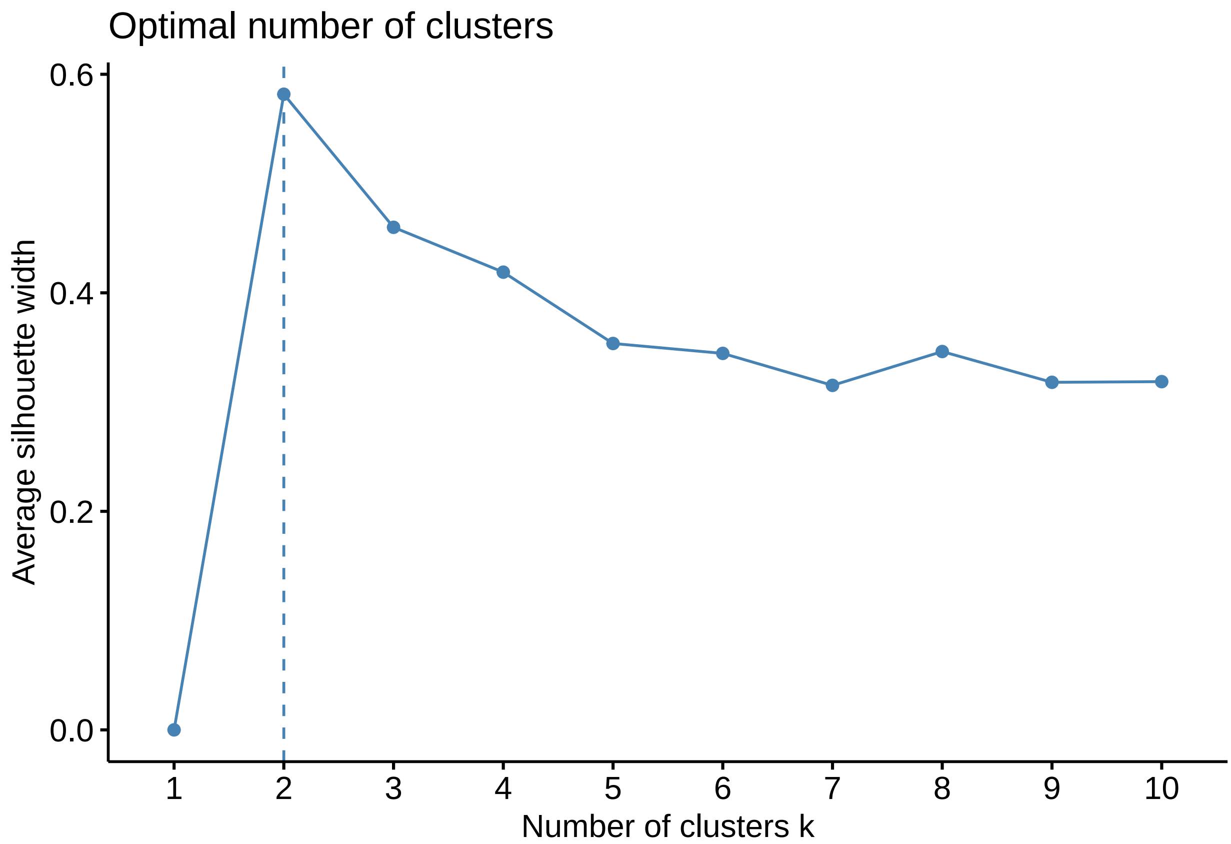
```
wss_values <- map_dbl(k.values, wss)
```

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




#or use this

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




# compute gap statistic

```r
set.seed(123)
gap_stat <- clusGap(df, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
```

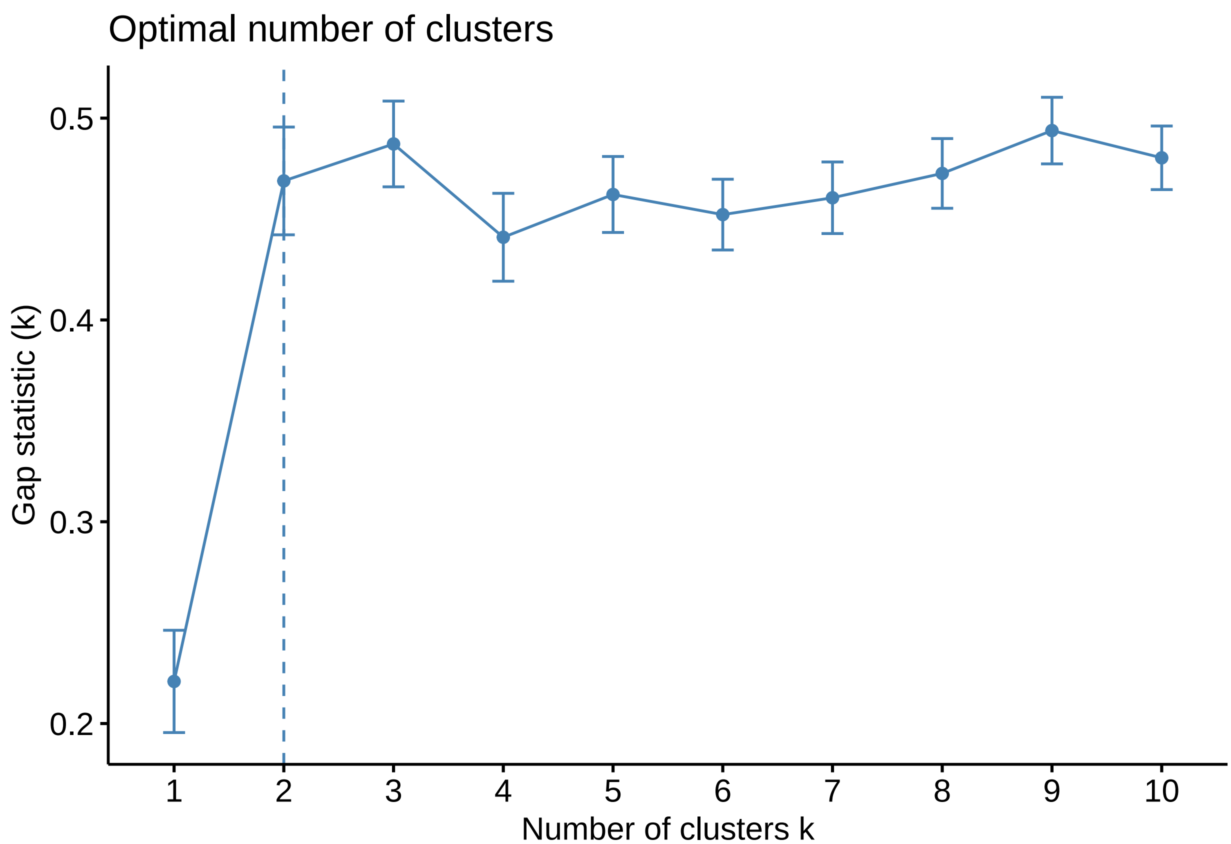
## Print the result

```r
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'): 3
##           logW   E.logW       gap     SE.sim
##  [1,] 4.534565 4.755428 0.2208634 0.02534324
##  [2,] 4.021316 4.490212 0.4688953 0.02670070
##  [3,] 3.806577 4.293793 0.4872159 0.02124741
##  [4,] 3.699263 4.140237 0.4409736 0.02177507
##  [5,] 3.589284 4.051459 0.4621749 0.01882154
##  [6,] 3.522810 3.975009 0.4521993 0.01753073
##  [7,] 3.448288 3.908834 0.4605460 0.01774025
##  [8,] 3.379870 3.852475 0.4726054 0.01727207
```

```
##  [9,] 3.310088 3.803931 0.4938436 0.01649671
## [10,] 3.278659 3.759003 0.4803440 0.01576050
```

```
fviz_gap_stat(gap_stat)
```




# Compute k-means clustering with k = 2

```
set.seed(123)
final <- kmeans(df, 2, nstart = 25)
print(final)
```

```
## K-means clustering with 2 clusters of sizes 50, 100
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1   -1.0111914   0.8504137    -1.300630  -1.2507035
## 2    0.5055957  -0.4252069     0.650315   0.6253518
##
## Clustering vector:
##   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   2   2   2   2   2   2   2   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   2   2   2   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   2
## 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   2   2   2   2   2   2
## 141 142 143 144 145 146 147 148 149 150
##   2   2   2   2   2   2   2   2   2   2
##
## Within cluster sum of squares by cluster:
## [1]  47.35062 173.52867
##  (between_SS / total_SS =  62.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

#final data

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