R\_KMeans\_Clustering\_Zoo

Yang

2/25/2021

## Introduction

This document serves as an introduction to the k-means clustering method.

## Load R Packages

Install required R packages:

#install.packages('RWeka')  
#install.packages('tidyverse') # data manipulation  
#install.packages('cluster') # clustering algorithms  
#install.packages('factoextra') # clustering algorithms & visualization  
#install.packages('gridExtra')

library(RWeka)  
library(tidyverse) # data manipulation

## -- Attaching packages -------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3 v purrr 0.3.3  
## v tibble 3.0.0 v dplyr 1.0.4  
## v tidyr 1.0.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ----------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

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

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

library(gridExtra) # subfigure layout package

##   
## Attaching package: 'gridExtra'

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

## Data Preparation

Setup the working environment and load the zoo.csv file

setwd("C:/Users/Yang/R\_Workspace/Week\_4")  
zoo <- read.csv("zoo.csv")  
str(zoo)

## 'data.frame': 101 obs. of 18 variables:  
## $ animal.name: Factor w/ 100 levels "aardvark","antelope",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ hair : int 1 1 0 1 1 1 1 0 0 1 ...  
## $ feathers : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ egg : int 0 0 1 0 0 0 0 1 1 0 ...  
## $ milk : int 1 1 0 1 1 1 1 0 0 1 ...  
## $ airborne : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ aquatic : int 0 0 1 0 0 0 0 1 1 0 ...  
## $ predator : int 1 0 1 1 1 0 0 0 1 0 ...  
## $ toothed : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ backbone : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ breathes : int 1 1 0 1 1 1 1 0 0 1 ...  
## $ venomous : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ fins : int 0 0 1 0 0 0 0 1 1 0 ...  
## $ legs : int 4 4 0 4 4 4 4 0 0 4 ...  
## $ tail : int 0 1 1 0 1 1 1 1 1 0 ...  
## $ domestic : int 0 0 0 0 0 0 1 1 0 1 ...  
## $ catsize : int 1 1 0 1 1 1 1 0 0 0 ...  
## $ type : int 1 1 4 1 1 1 1 4 4 1 ...

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

zoo <- na.omit(zoo)

Remove the label information

zoo\_unlabeled <- zoo[,c(2:17)]  
head(zoo\_unlabeled)

## hair feathers egg milk airborne aquatic predator toothed backbone breathes  
## 1 1 0 0 1 0 0 1 1 1 1  
## 2 1 0 0 1 0 0 0 1 1 1  
## 3 0 0 1 0 0 1 1 1 1 0  
## 4 1 0 0 1 0 0 1 1 1 1  
## 5 1 0 0 1 0 0 1 1 1 1  
## 6 1 0 0 1 0 0 0 1 1 1  
## venomous fins legs tail domestic catsize  
## 1 0 0 4 0 0 1  
## 2 0 0 4 1 0 1  
## 3 0 1 0 1 0 0  
## 4 0 0 4 0 0 1  
## 5 0 0 4 1 0 1  
## 6 0 0 4 1 0 1

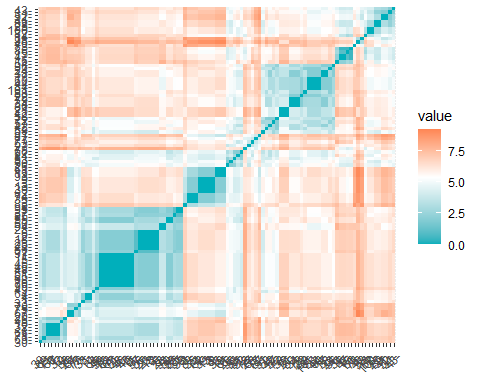
As we don’t want the clustering algorithm to depend to an arbitrary variable unit, we start by scaling/standardizing the data using the R function scale:

optional for zoo data.

zoo\_unlabeled <- scale(zoo\_unlabeled)  
head(zoo\_unlabeled)

## hair feathers egg milk airborne aquatic predator  
## 1 1.1556307 -0.494438 -1.1793445 1.2037132 -0.5555198 -0.740515 0.8919727  
## 2 1.1556307 -0.494438 -1.1793445 1.2037132 -0.5555198 -0.740515 -1.1100105  
## 3 -0.8567607 -0.494438 0.8395333 -0.8225373 -0.5555198 1.337041 0.8919727  
## 4 1.1556307 -0.494438 -1.1793445 1.2037132 -0.5555198 -0.740515 0.8919727  
## 5 1.1556307 -0.494438 -1.1793445 1.2037132 -0.5555198 -0.740515 0.8919727  
## 6 1.1556307 -0.494438 -1.1793445 1.2037132 -0.5555198 -0.740515 -1.1100105  
## toothed backbone breathes venomous fins legs tail  
## 1 0.8057576 0.4633792 0.5098049 -0.2918387 -0.4476351 0.5696983 -1.689987  
## 2 0.8057576 0.4633792 0.5098049 -0.2918387 -0.4476351 0.5696983 0.585862  
## 3 0.8057576 0.4633792 -1.9421137 -0.2918387 2.2118440 -1.3974651 0.585862  
## 4 0.8057576 0.4633792 0.5098049 -0.2918387 -0.4476351 0.5696983 -1.689987  
## 5 0.8057576 0.4633792 0.5098049 -0.2918387 -0.4476351 0.5696983 0.585862  
## 6 0.8057576 0.4633792 0.5098049 -0.2918387 -0.4476351 0.5696983 0.585862  
## domestic catsize  
## 1 -0.3824456 1.1325318  
## 2 -0.3824456 1.1325318  
## 3 -0.3824456 -0.8742351  
## 4 -0.3824456 1.1325318  
## 5 -0.3824456 1.1325318  
## 6 -0.3824456 1.1325318

distance <- get\_dist(zoo\_unlabeled)  
fviz\_dist(distance, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"))



## RWeka SimpleKMeans Clustering

model\_rweka <- SimpleKMeans (zoo\_unlabeled, control = Weka\_control(N = 7, I=500, S=100))  
model\_rweka

##   
## kMeans  
## ======  
##   
## Number of iterations: 5  
## Within cluster sum of squared errors: 94.70379505535756  
##   
## Initial starting points (random):  
##   
## Cluster 0: -0.856761,-0.494438,0.839533,-0.822537,-0.55552,1.337041,0.891973,0.805758,0.463379,-1.942114,-0.291839,2.211844,-1.397465,0.585862,-0.382446,1.132532  
## Cluster 1: 1.155631,-0.494438,-1.179344,1.203713,-0.55552,-0.740515,0.891973,0.805758,0.463379,0.509805,-0.291839,-0.447635,0.569698,0.585862,-0.382446,1.132532  
## Cluster 2: -0.856761,-0.494438,0.839533,-0.822537,-0.55552,1.337041,0.891973,-1.22878,-2.136693,-1.942114,-0.291839,-0.447635,1.55328,-1.689987,-0.382446,-0.874235  
## Cluster 3: -0.856761,-0.494438,0.839533,-0.822537,-0.55552,1.337041,0.891973,0.805758,0.463379,0.509805,3.392624,-0.447635,0.569698,-1.689987,-0.382446,-0.874235  
## Cluster 4: -0.856761,2.002474,0.839533,-0.822537,-0.55552,-0.740515,0.891973,-1.22878,0.463379,0.509805,-0.291839,-0.447635,-0.413883,0.585862,-0.382446,-0.874235  
## Cluster 5: -0.856761,-0.494438,0.839533,-0.822537,-0.55552,-0.740515,-1.11001,-1.22878,0.463379,0.509805,-0.291839,-0.447635,0.569698,0.585862,-0.382446,1.132532  
## Cluster 6: -0.856761,-0.494438,-1.179344,1.203713,-0.55552,1.337041,0.891973,0.805758,0.463379,0.509805,-0.291839,2.211844,-1.397465,0.585862,-0.382446,1.132532  
##   
## Missing values globally replaced with mean/mode  
##   
## Final cluster centroids:  
## Cluster#  
## Attribute Full Data 0 1 2 3 4 5 6  
## (101.0) (13.0) (37.0) (18.0) (8.0) (14.0) (7.0) (4.0)  
## ===================================================================================================  
## hair 0 -0.8568 1.1556 -0.4096 -0.8568 -0.8568 -0.8568 0.1494  
## feathers -0 -0.4944 -0.4944 -0.4944 -0.4944 2.0025 1.6458 -0.4944  
## egg 0 0.8395 -1.1248 0.7274 0.5872 0.8395 0.8395 -1.1793  
## milk 0 -0.8225 1.2037 -0.8225 -0.8225 -0.8225 -0.8225 1.2037  
## airborne -0 -0.5555 -0.4292 0.2238 -0.5555 1.6153 0.4464 -0.5555  
## aquatic 0 1.337 -0.6282 -0.048 0.558 -0.1469 -0.1469 1.337  
## predator -0 0.276 -0.1361 -0.109 0.6417 -0.252 -0.252 0.892  
## toothed 0 0.8058 0.7508 -1.2288 0.8058 -1.2288 -1.2288 0.8058  
## backbone 0 0.4634 0.4634 -2.1367 0.4634 0.4634 0.4634 0.4634  
## breathes -0 -1.9421 0.5098 -0.4437 0.2033 0.5098 0.5098 0.5098  
## venomous 0 -0.0084 -0.2918 0.5269 1.0898 -0.2918 -0.2918 -0.2918  
## fins 0 2.2118 -0.4476 -0.4476 -0.4476 -0.4476 -0.4476 2.2118  
## legs 0 -1.3975 0.4102 0.9249 -0.168 -0.4139 -0.2734 -1.1516  
## tail 0 0.5859 0.2783 -1.5636 -0.2676 0.5859 0.5859 0.0169  
## domestic 0 -0.1539 0.26 -0.2174 -0.3824 0.2543 -0.3824 -0.3824  
## catsize 0 -0.2568 0.6444 -0.7627 -0.8742 -0.8742 1.1325 1.1325

## KMeans Clustering in R

Since it is difficult to interpret and visualize the clustering results with RWeka, we will introduce some built-in R functions, as well as visualization packages.

model\_r = kmeans(zoo\_unlabeled, centers = 7, nstart = 25)  
model\_r

## K-means clustering with 7 clusters of sizes 37, 7, 21, 8, 4, 13, 11  
##   
## Cluster means:  
## hair feathers egg milk airborne aquatic predator  
## 1 1.1556307 -0.494438 -1.1247802 1.2037132 -0.4291516 -0.6282147 -0.1360727  
## 2 -0.8567607 -0.494438 0.8395333 -0.8225373 -0.5555198 1.0402473 0.8919727  
## 3 -0.8567607 1.883573 0.8395333 -0.8225373 1.2256708 -0.1469276 -0.2520177  
## 4 -0.8567607 -0.494438 0.5871736 -0.8225373 -0.5555198 0.5579575 0.6417248  
## 5 0.1494350 -0.494438 -1.1793445 1.2037132 -0.5555198 1.3370410 0.8919727  
## 6 -0.8567607 -0.494438 0.8395333 -0.8225373 -0.5555198 1.3370410 0.2759779  
## 7 -0.1249820 -0.494438 0.6559990 -0.8225373 0.7196507 -0.7405150 -0.7460135  
## toothed backbone breathes venomous fins legs  
## 1 0.7507701 0.4633792 0.5098049 -0.291838667 -0.4476351 0.4101986  
## 2 -1.2287803 -2.1366929 -1.9421137 0.234513214 -0.4476351 0.6399541  
## 3 -1.2287803 0.4633792 0.5098049 -0.291838667 -0.4476351 -0.3670462  
## 4 0.8057576 0.4633792 0.2033150 1.089835020 -0.4476351 -0.1679880  
## 5 0.8057576 0.4633792 0.5098049 -0.291838667 2.2118440 -1.1515697  
## 6 0.8057576 0.4633792 -1.9421137 -0.008418423 2.2118440 -1.3974651  
## 7 -1.2287803 -2.1366929 0.5098049 0.713014924 -0.4476351 1.1061974  
## tail domestic catsize  
## 1 0.27831492 0.25999939 0.6443993  
## 2 -1.68998664 -0.38244559 -0.5875541  
## 3 0.58586203 0.04202699 -0.2053128  
## 4 -0.26758122 -0.38244559 -0.8742351  
## 5 0.01689987 -0.38244559 1.1325318  
## 6 0.58586203 -0.15388343 -0.2567683  
## 7 -1.48309130 -0.11232668 -0.8742351  
##   
## Clustering vector:  
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20   
## 1 1 6 1 1 1 1 6 6 1 1 3 6 2 2 2 3 1 6 5   
## 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40   
## 3 3 1 3 7 4 4 1 1 1 7 1 1 3 6 1 1 3 6 7   
## 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60   
## 7 3 7 3 1 1 2 1 1 1 1 7 4 2 1 1 3 3 3 3   
## 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80   
## 6 6 4 1 1 1 5 1 1 1 1 3 7 6 5 5 4 2 3 3   
## 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100   
## 4 7 6 3 1 2 6 3 7 4 3 4 6 1 1 3 1 7 1 7   
## 101   
## 3   
##   
## Within cluster sum of squares by cluster:  
## [1] 173.592410 32.538758 108.285404 62.847609 8.659909 42.931467 94.013674  
## (between\_SS / total\_SS = 67.3 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"

print the centroids

model\_r$centers

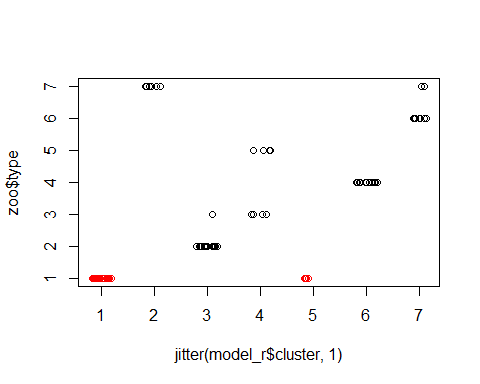
## hair feathers egg milk airborne aquatic predator  
## 1 1.1556307 -0.494438 -1.1247802 1.2037132 -0.4291516 -0.6282147 -0.1360727  
## 2 -0.8567607 -0.494438 0.8395333 -0.8225373 -0.5555198 1.0402473 0.8919727  
## 3 -0.8567607 1.883573 0.8395333 -0.8225373 1.2256708 -0.1469276 -0.2520177  
## 4 -0.8567607 -0.494438 0.5871736 -0.8225373 -0.5555198 0.5579575 0.6417248  
## 5 0.1494350 -0.494438 -1.1793445 1.2037132 -0.5555198 1.3370410 0.8919727  
## 6 -0.8567607 -0.494438 0.8395333 -0.8225373 -0.5555198 1.3370410 0.2759779  
## 7 -0.1249820 -0.494438 0.6559990 -0.8225373 0.7196507 -0.7405150 -0.7460135  
## toothed backbone breathes venomous fins legs  
## 1 0.7507701 0.4633792 0.5098049 -0.291838667 -0.4476351 0.4101986  
## 2 -1.2287803 -2.1366929 -1.9421137 0.234513214 -0.4476351 0.6399541  
## 3 -1.2287803 0.4633792 0.5098049 -0.291838667 -0.4476351 -0.3670462  
## 4 0.8057576 0.4633792 0.2033150 1.089835020 -0.4476351 -0.1679880  
## 5 0.8057576 0.4633792 0.5098049 -0.291838667 2.2118440 -1.1515697  
## 6 0.8057576 0.4633792 -1.9421137 -0.008418423 2.2118440 -1.3974651  
## 7 -1.2287803 -2.1366929 0.5098049 0.713014924 -0.4476351 1.1061974  
## tail domestic catsize  
## 1 0.27831492 0.25999939 0.6443993  
## 2 -1.68998664 -0.38244559 -0.5875541  
## 3 0.58586203 0.04202699 -0.2053128  
## 4 -0.26758122 -0.38244559 -0.8742351  
## 5 0.01689987 -0.38244559 1.1325318  
## 6 0.58586203 -0.15388343 -0.2567683  
## 7 -1.48309130 -0.11232668 -0.8742351

get cluster assignment

cluster\_assignment <- data.frame(zoo,model\_r$cluster)  
#View(cluster\_assignment)

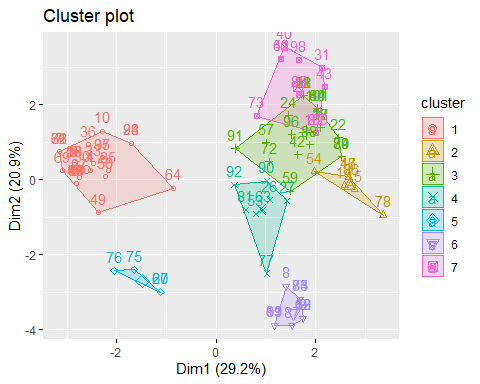
visualize animal types and clusters by specific features, red = milk yes, black = milk no.

plot(zoo$type ~ jitter(model\_r$cluster, 1), pch=21,col=as.factor(zoo$milk))



We can also view our results by using fviz\_cluster. If there are more than two variables fviz\_cluster will perform principal component analysis (PCA) and plot the data points according to the first two principal components that explain the majority of the variance.

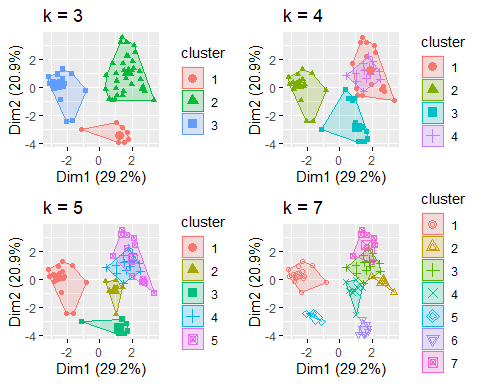
fviz\_cluster(model\_r, data = zoo\_unlabeled)



## Select the best K

It is often advantageous to use several different values of k and examine the differences in the results. We can execute the same process for 3, 4, and 5 clusters, and the results are shown in the figure:

k3 <- kmeans(zoo\_unlabeled, centers = 3, nstart = 25)  
k4 <- kmeans(zoo\_unlabeled, centers = 4, nstart = 25)  
k5 <- kmeans(zoo\_unlabeled, centers = 5, nstart = 25)  
  
# plots to compare  
p1 <- fviz\_cluster(k3, geom = "point", data = zoo\_unlabeled) + ggtitle("k = 3")  
p2 <- fviz\_cluster(k4, geom = "point", data = zoo\_unlabeled) + ggtitle("k = 4")  
p3 <- fviz\_cluster(k5, geom = "point", data = zoo\_unlabeled) + ggtitle("k = 5")  
p4 <- fviz\_cluster(model\_r, geom = "point", data = zoo\_unlabeled) + ggtitle("k = 7")  
  
library(gridExtra)  
grid.arrange(p1, p2, p3, p4, nrow = 2)

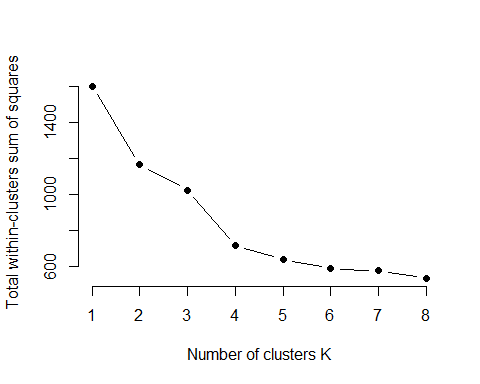


Preferably we would like to use the optimal number of clusters. The following explains the three most popular methods for determining the optimal clusters, which includes:

* Elbow method
* Silhouette method
* Gap statistic

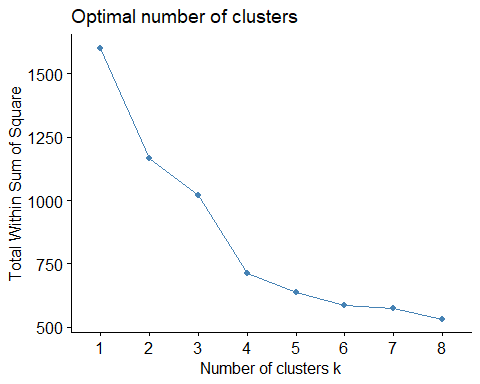
### Elbow method

set.seed(123)  
  
# set up a list that records cluster size from 1 to 8   
k.values <- 1:8  
  
# set up an empty list, which will be used to record within-cluster SSE for each k  
sse.values <- vector(mode = "list", length = length(k.values))  
  
# compute total within-cluster sum of square for each possible k  
for (i in k.values) {  
 sse.values[i] = kmeans(zoo\_unlabeled, i)$tot.withinss  
}  
  
# plot the relationship between k and sse  
plot(k.values, sse.values,  
 type="b", pch = 19, frame = FALSE,   
 xlab="Number of clusters K",  
 ylab="Total within-clusters sum of squares")



Fortunately, this process to compute the “Elbow method” has been wrapped up in a single function (fviz\_nbclust):

set.seed(123)  
fviz\_nbclust(zoo\_unlabeled, kmeans, method = "wss", k.max=8)



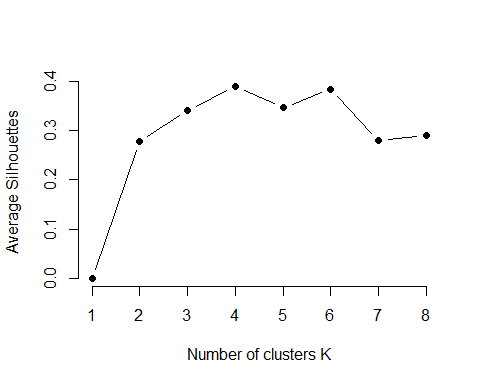
### Average Silhouette Method

In short, the average silhouette approach measures the quality of a clustering. That is, it determines how well each object lies within its cluster. A high average silhouette width indicates a good clustering. The average silhouette method computes the average silhouette of observations for different values of k. The optimal number of clusters k is the one that maximizes the average silhouette over a range of possible values for k.

set.seed(123)  
# set up a list that records cluster size from 2 to 8, because silhouette does not apply to k=1  
k.values <- 1:8  
  
# set up an empty list, which will be used to record average Silhouette  
sil.values <- vector(mode = "list", length = length(k.values))  
  
# compute average Silhouette for each possible k  
for (i in k.values) {  
 if (i == 1) {  
 sil.values[i] = 0  
 } else {  
 km.res = kmeans(zoo\_unlabeled, i)  
 ss <- silhouette(km.res$cluster, dist(zoo\_unlabeled))  
 print(i)  
 sil.values[i] = mean(ss[,3])  
 }  
}

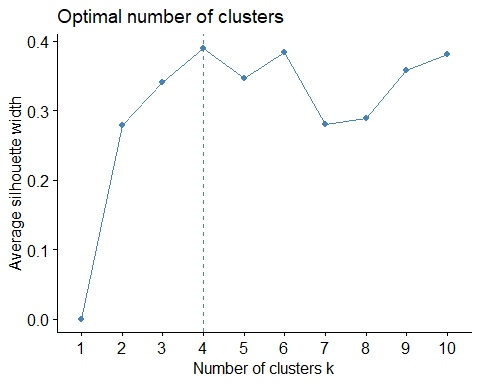
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5  
## [1] 6  
## [1] 7  
## [1] 8

# plot the relationship between k and Silhouette  
plot(k.values, sil.values,  
 type = "b", pch = 19, frame = FALSE,   
 xlab = "Number of clusters K",  
 ylab = "Average Silhouettes")



Similar to the elbow method, this process to compute the “average silhoutte method” has been wrapped up in a single function

set.seed(123)  
fviz\_nbclust(zoo\_unlabeled, kmeans, method = "silhouette")



### Gap Statistic Method

The gap statistic has been published by R. Tibshirani, G. Walther, and T. Hastie (Standford University, 2001). The approach can be applied to any clustering method (i.e. K-means clustering, hierarchical clustering). The gap statistic compares the total intracluster variation for different values of k with their expected values under null reference distribution of the data (i.e. a distribution with no obvious clustering).

Choose the number of clusters as the smallest k such that where is the standard deviation.

set.seed(123)  
gap\_stat <- clusGap(zoo\_unlabeled, FUN = kmeans,   
 K.max = 8, B = 50)  
  
set.seed(123)  
fviz\_gap\_stat(gap\_stat)

