Cluster Analysis

Réda Es-sakhi

01/05/2021

Required packages

```
#install.packages("cluster")
#install.packages("factoextra")
#install.packages("cluster.datasets")
#install.packages("xtable")
#install.packages("kableExtra")
#install.packages("knitr")
#install.packages("summarytools")
#install.packages("stargazer")
#install.packages("ade4")
#install.packages("tidymodels")
```

```
x<-c(0,0)
y<-c(6,6)
dist(rbind(x,y),method = "euclidian", diag = T, upper = T)
```

```
##          x        y
## x 0.000000 8.485281
## y 8.485281 0.000000
```

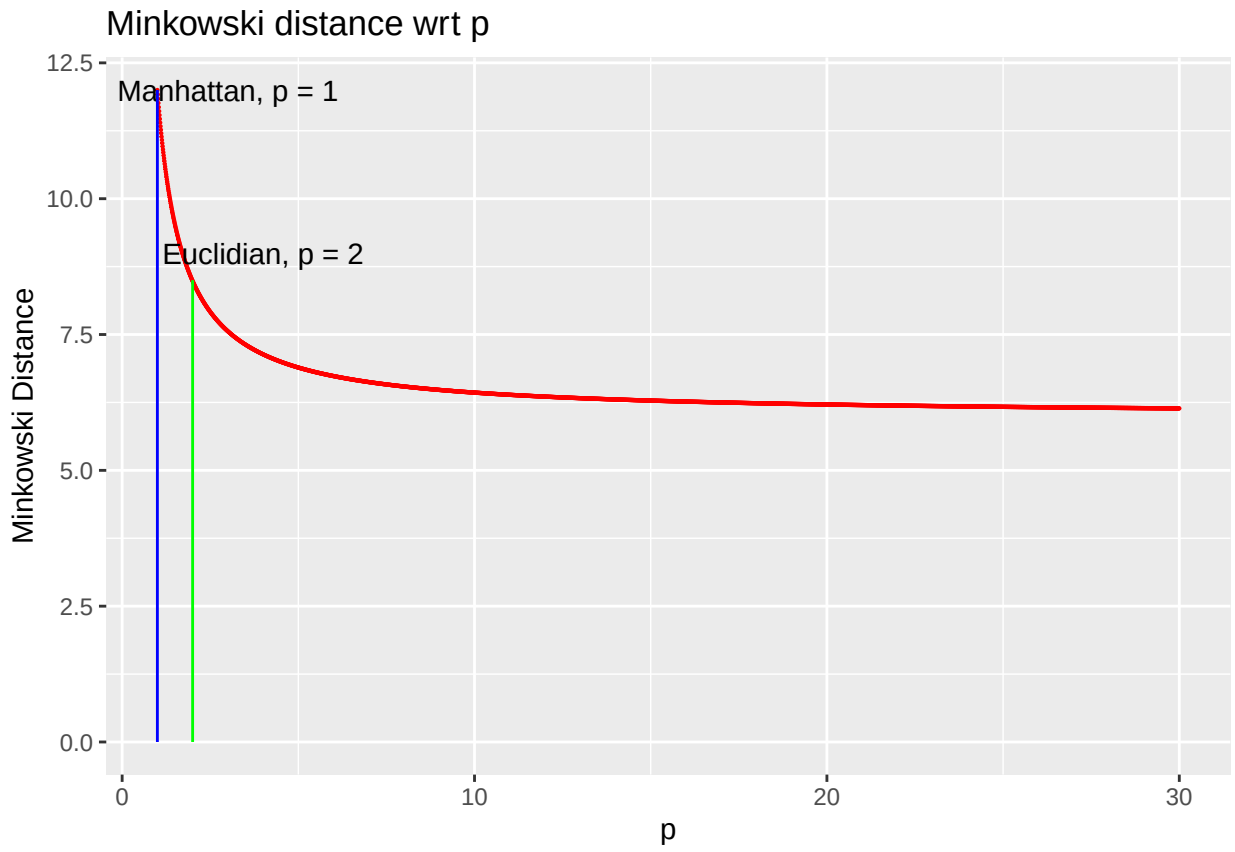
Excercie 4 minkowski distance

```
library("ggplot2")
x<-c(0,0)
y<-c(6,6)

MinkowDist=c() # Initialiser à vide la liste
for (p in seq(1,30,.01))
{
MinkowDist=c(MinkowDist,dist(rbind(x, y), method = "minkowski", p = p))
}

dist_plot <- ggplot(data =data.frame(x = seq(1,30,.01), y=MinkowDist ) , mapping = aes( x=x, y= y))+
  geom_point(size=.1,color="red")+
  xlab("p")+ylab("Minkowski Distance")+
  ggtitle("Minkowski distance wrt p")
```

```
dist_plot + annotate("text", x = 3, y = 12, label = "Manhattan, p = 1") +
  annotate("segment", x = 1, xend = 1, y = 0, yend = 12,
colour = "blue")+
  annotate("text", x = 4, y = 9, label = "Euclidian, p = 2") +
  annotate("segment", x = 2, xend = 2, y = 0, yend = 8.5,
colour = "Green")
```



exo5

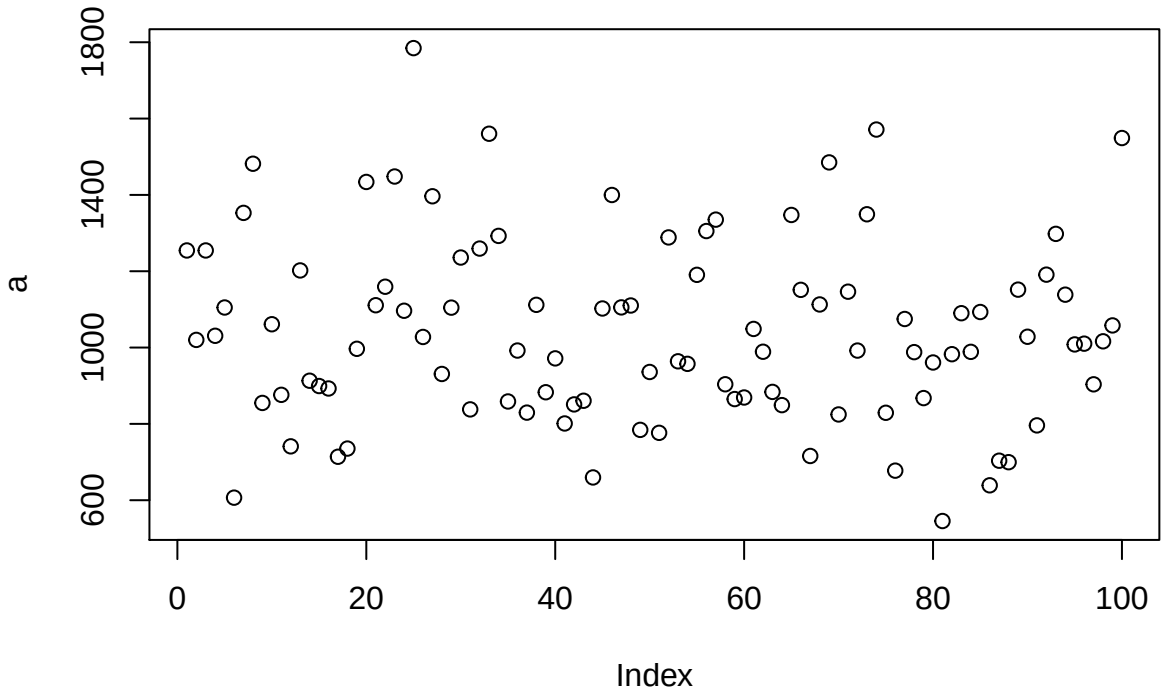
```
n = 100
m = log(2500*sqrt(2/13))
s = sqrt(log(26/25))
a <- rlnorm(n,m,s)
mean(a)
```

```
## [1] 1041.404
```

```
sd(a)
```

```
## [1] 241.0304
```

```
plot(a)
```



exo 6

```
x = c(22, 34, 1, 12, 25, 56, 7)
y = c(2,64,12,2,22,5,8)
```

ex6 The rank for each vector

```
r_x = rank(x)
r_y = rank(y)
r_x
```

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

```
r_y
```

```
## [1] 1.5 7.0 5.0 1.5 6.0 3.0 4.0
```

ex6 The rank for each vector

```
d=r_x-r_y
d
```

```
## [1]  2.5 -1.0 -4.0  1.5 -1.0  4.0 -2.0
```

```
alt_cor = 1 - 6*sum(d^2)/(7*(7^2-1))
alt_cor
```

```
## [1] 0.1696429
```

```
spear_cor = cor(x,y, method=c("spearman"))
spear_cor
```

```
## [1] 0.1621687
```

exo 7

x=(22,34,1,12,25,56,7) y=(2,64,12,2,22,5,8) 1- liste les pairs (faire un tableau) 2 - nombre de pairs Pair of coordinate (2 parmi n) soit (n*(n-1))/2

```
7*6/2
```

```
## [1] 21
```

3- for each pair compute the sign (faire un tableau)

```
tau=0
for (i in 1:7)
{
tau=tau+sign(x -x[i])%*%sign(y -y[i])
}
tau=tau/(7*6)
tau
```

```
##           [,1]
## [1,] 0.0952381
```

```
cor(x,y,method = "kendall")
```

```
## [1] 0.09759001
```

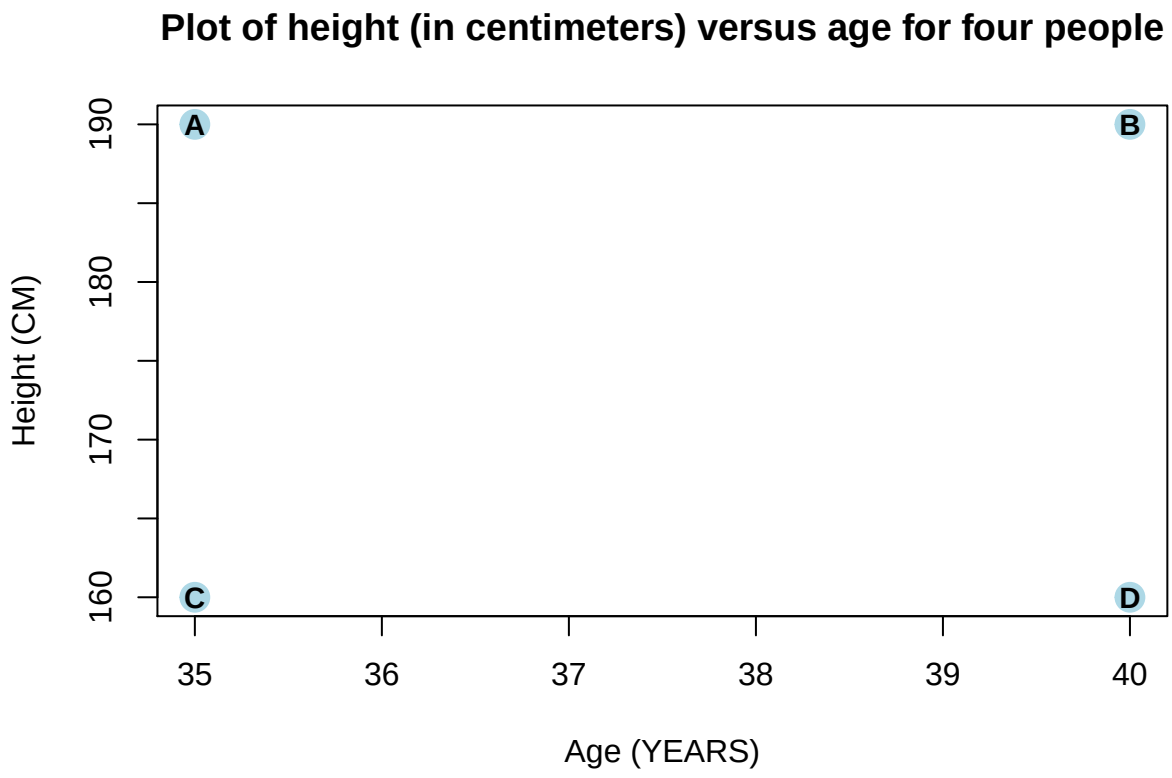
exo 8

Age in (years) and height in (cm) of fout people

```
table3 =c(
  35,190,
  40,190,
  35,160,
  40,160
)
table3=data.frame(matrix(table3, nrow=4,byrow=T))
row.names(table3)=c("A","B","C","D")
names(table3)=c("Age", "Height")
table3
```

```
##   Age Height
## A  35    190
## B  40    190
## C  35    160
## D  40    160
```

```
plot(Height~Age,
     main = "Plot of height (in centimeters) versus age for four people",
     xlab = "Age (YEARS)", ylab = "Height (CM)",
     col="lightblue", pch=19, cex=2,data=table3)
text(Height~Age, labels=rownames(table3), data=table3, cex=0.9, font=2)
```



Conversion de Height en Feet

```
yCm <-table3$Height
yCm
```

```
## [1] 190 190 160 160
```

```
yFeet <- yCm/30.48
yFeet<-round(yFeet,2)
yFeet
```

```
## [1] 6.23 6.23 5.25 5.25
```

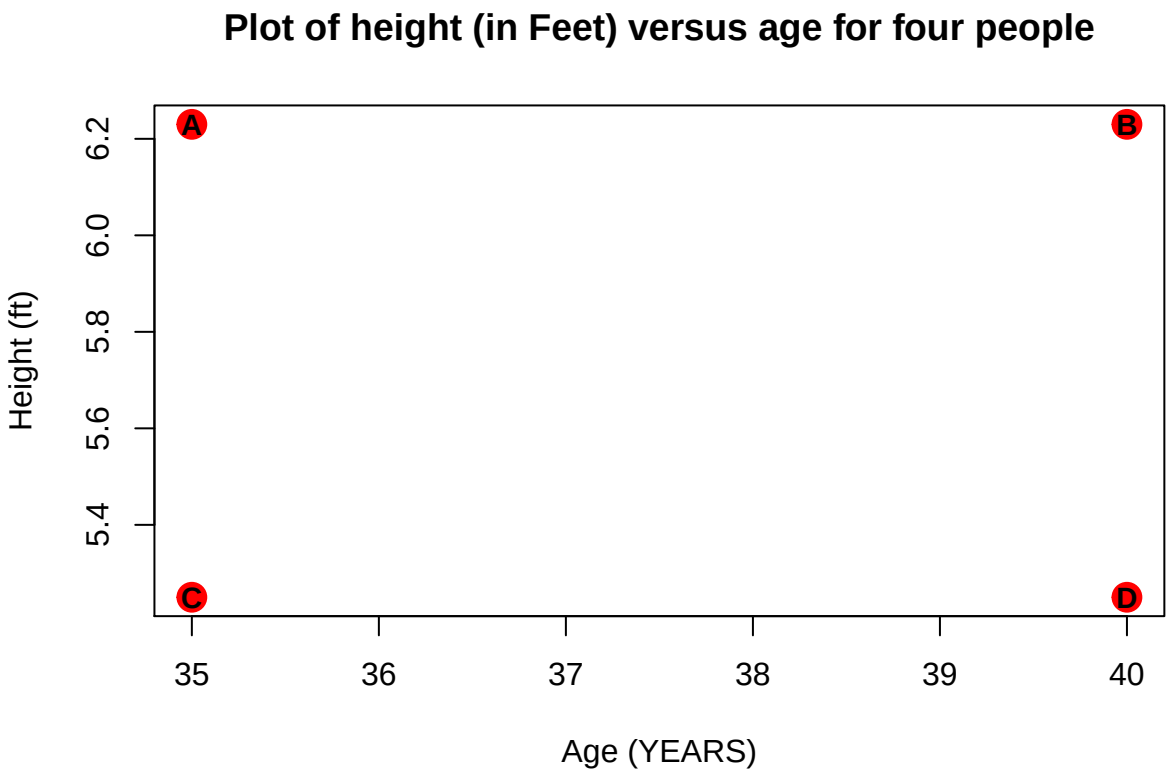
faire un tableau avec kable

```
table4 <- data.frame(table3$Age, yFeet)
row.names(table4)=c("A","B","C","D")
names(table4)=c("Age","Height")
table4
```

```
##   Age Height
## A  35   6.23
## B  40   6.23
## C  35   5.25
## D  40   5.25
```

nouveau plot height en Feet

```
plot(Height~Age,
     main = "Plot of height (in Feet) versus age for four people",
     xlab = "Age (YEARS)", ylab = "Height (ft)",
     col="red", pch=19, cex=2,data=table4)
text(Height~Age, labels=rownames(table4), data=table4, cex=0.9, font=2)
```



On remarque la même distribution sur notre plot. On transforme l'unité de mesure ne change pas la visualisation et la relation entre les deux variables

```
xSt <- table3$Age
xSt <- scale(xSt)
xSt
```

```
##            [,1]
## [1,] -0.8660254
## [2,]  0.8660254
## [3,] -0.8660254
## [4,]  0.8660254
## attr(,"scaled:center")
## [1] 37.5
## attr(,"scaled:scale")
## [1] 2.886751
```

```
ySt <- table3$Height
ySt <- scale(ySt)
ySt
```

```
##            [,1]
```

```
## [1,]  0.8660254
## [2,]  0.8660254
## [3,] -0.8660254
## [4,] -0.8660254
## attr(,"scaled:center")
## [1] 175
## attr(,"scaled:scale")
## [1] 17.32051
```

```
(table3$Age - mean(table3$Age))/sd(table3$Age)
```

```
## [1] -0.8660254  0.8660254 -0.8660254  0.8660254
```

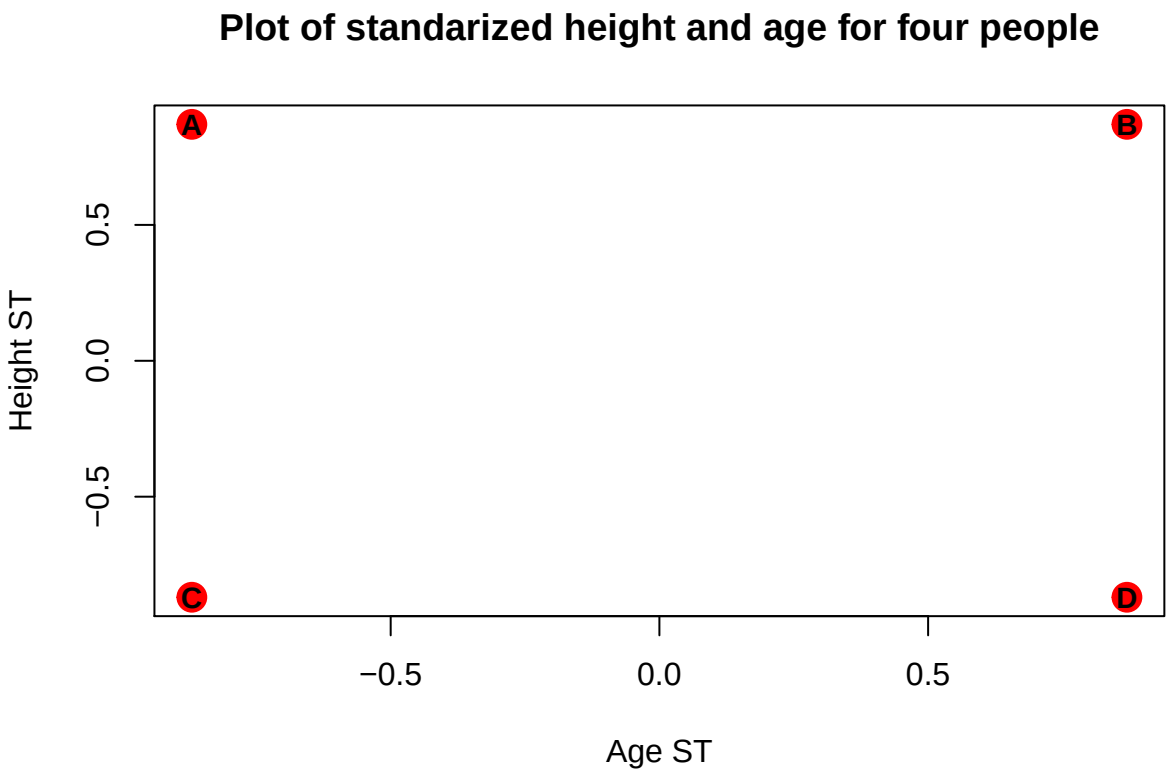
Tableau des variables standardisées

```
table5 <- data.frame(round(xSt,2),round(ySt,2))
row.names(table5) = c("A","B","C","D")
names(table5) = c("Age", "Height")
table5
```

```
##     Age Height
## A -0.87   0.87
## B  0.87   0.87
## C -0.87  -0.87
## D  0.87  -0.87
```

plot des variables standardisées

```
plot(Height~Age,
     main = "Plot of standarized height and age for four people",
     xlab = "Age ST", ylab = "Height ST",
     col="red", pch=19, cex=2,data=table5)
text(Height~Age, labels=rownames(table5), data=table5, cex=0.9, font=2)
```



La même distribution pour les variables standardisées sur notre plot

exo 9

Use the data set animals available in the package cluster. This data set was first used in this textbook KAUFMAN, Leonard et ROUSSEEUW, Peter J. Finding groups in data: an introduction to cluster analysis. John Wiley & Sons, 2009. Identify the missing measurements. Explain the way how KAUFMAN and ROUSSEEUW, pp. 296-297 treat the missing measurements. Compute a distance matrix for the completed data. Propose a graphical way to represent that distance matrix. Which group of animals look close? Change the method of calculating and observe if it has some effect of the graph

call library and animals dataset

```
library(cluster)
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
library(forcats)
data("animals")
```

finding missing values

```
head(animals)
```

```
##     war fly ver end gro hai
## ant   1   1   1   1   2   1
## bee   1   2   1   1   2   2
## cat   2   1   2   1   1   2
## cpl   1   1   1   1   1   2
## chi   2   1   2   2   2   2
## cow   2   1   2   1   2   2
```

```
is.na(animals)
```

```
##       war   fly   ver   end   gro   hai
## ant FALSE FALSE FALSE FALSE FALSE FALSE
## bee FALSE FALSE FALSE FALSE FALSE FALSE
## cat FALSE FALSE FALSE FALSE FALSE FALSE
## cpl FALSE FALSE FALSE FALSE FALSE FALSE
## chi FALSE FALSE FALSE FALSE FALSE FALSE
## cow FALSE FALSE FALSE FALSE FALSE FALSE
## duc FALSE FALSE FALSE FALSE FALSE FALSE
## eag FALSE FALSE FALSE FALSE FALSE FALSE
## ele FALSE FALSE FALSE FALSE FALSE FALSE
## fly FALSE FALSE FALSE FALSE FALSE FALSE
## fro FALSE FALSE FALSE FALSE  TRUE FALSE
## her FALSE FALSE FALSE FALSE FALSE FALSE
## lio FALSE FALSE FALSE  TRUE FALSE FALSE
## liz FALSE FALSE FALSE FALSE FALSE FALSE
## lob FALSE FALSE FALSE FALSE  TRUE FALSE
## man FALSE FALSE FALSE FALSE FALSE FALSE
## rab FALSE FALSE FALSE FALSE FALSE FALSE
## sal FALSE FALSE FALSE FALSE  TRUE FALSE
## spi FALSE FALSE FALSE  TRUE FALSE FALSE
## wha FALSE FALSE FALSE FALSE FALSE FALSE
```

```
str(animals)
```

```
## 'data.frame':    20 obs. of  6 variables:
##  $ war: int  1 1 2 1 2 2 2 2 2 1 ...
##  $ fly: int  1 2 1 1 1 1 2 2 1 2 ...
##  $ ver: int  1 1 2 1 2 2 2 2 2 1 ...
##  $ end: int  1 1 1 1 2 1 1 2 2 1 ...
##  $ gro: int  2 2 1 1 2 2 2 1 2 1 ...
##  $ hai: int  1 2 2 2 2 2 1 1 1 1 ...
```

```
animalsF<- animals
animalsF[animalsF == "1"] <- 0
animalsF[animalsF == "2"] <- 1
animalsF
```

```
##     war fly ver end gro hai
## ant   0   0   0   0   1   0
## bee   0   1   0   0   1   1
## cat   1   0   1   0   0   1
## cpl   0   0   0   0   0   1
## chi   1   0   1   1   1   1
## cow   1   0   1   0   1   1
## duc   1   1   1   0   1   0
## eag   1   1   1   1   0   0
## ele   1   0   1   1   1   0
## fly   0   1   0   0   0   0
## fro   0   0   1   1  NA   0
## her   0   0   1   0   1   0
## lio   1   0   1  NA   1   1
## liz   0   0   1   0   0   0
## lob   0   0   0   0  NA   0
## man   1   0   1   1   1   1
## rab   1   0   1   0   1   1
## sal   0   0   1   0  NA   0
## spi   0   0   0  NA   0   1
## wha   1   0   1   1   1   0
```

```
animalsF$war <- factor(animalsF$war, levels = c(0,1))
animalsF$fly <- factor(animalsF$fly, levels = c(0,1))
animalsF$ver <- factor(animalsF$ver, levels = c(0,1))
animalsF$end <- factor(animalsF$end, levels = c(0,1))
animalsF$gro <- factor(animalsF$gro, levels = c(0,1))
animalsF$hai <- factor(animalsF$hai, levels = c(0,1))

animalsF
```

```
##     war fly ver  end  gro hai
## ant   0   0   0    0    1   0
## bee   0   1   0    0    1   1
## cat   1   0   1    0    0   1
## cpl   0   0   0    0    0   1
## chi   1   0   1    1    1   1
## cow   1   0   1    0    1   1
## duc   1   1   1    0    1   0
## eag   1   1   1    1    0   0
## ele   1   0   1    1    1   0
## fly   0   1   0    0    0   0
## fro   0   0   1    1 <NA>   0
## her   0   0   1    0    1   0
## lio   1   0   1 <NA>    1   1
## liz   0   0   1    0    0   0
## lob   0   0   0    0 <NA>   0
## man   1   0   1    1    1   1
```

```
## rab   1   0   1    0    1   1
## sal   0   0   1    0 <NA>   0
## spi   0   0   0 <NA>    0   1
## wha   1   0   1    1    1   0
```

```
is.na(animalsF)
```

```
##       war   fly   ver   end   gro   hai
## ant FALSE FALSE FALSE FALSE FALSE FALSE
## bee FALSE FALSE FALSE FALSE FALSE FALSE
## cat FALSE FALSE FALSE FALSE FALSE FALSE
## cpl FALSE FALSE FALSE FALSE FALSE FALSE
## chi FALSE FALSE FALSE FALSE FALSE FALSE
## cow FALSE FALSE FALSE FALSE FALSE FALSE
## duc FALSE FALSE FALSE FALSE FALSE FALSE
## eag FALSE FALSE FALSE FALSE FALSE FALSE
## ele FALSE FALSE FALSE FALSE FALSE FALSE
## fly FALSE FALSE FALSE FALSE FALSE FALSE
## fro FALSE FALSE FALSE FALSE  TRUE FALSE
## her FALSE FALSE FALSE FALSE FALSE FALSE
## lio FALSE FALSE FALSE  TRUE FALSE FALSE
## liz FALSE FALSE FALSE FALSE FALSE FALSE
## lob FALSE FALSE FALSE FALSE  TRUE FALSE
## man FALSE FALSE FALSE FALSE FALSE FALSE
## rab FALSE FALSE FALSE FALSE FALSE FALSE
## sal FALSE FALSE FALSE FALSE  TRUE FALSE
## spi FALSE FALSE FALSE  TRUE FALSE FALSE
## wha FALSE FALSE FALSE FALSE FALSE FALSE
```

```
animals%>% is.na
```

```
##       war   fly   ver   end   gro   hai
## ant FALSE FALSE FALSE FALSE FALSE FALSE
## bee FALSE FALSE FALSE FALSE FALSE FALSE
## cat FALSE FALSE FALSE FALSE FALSE FALSE
## cpl FALSE FALSE FALSE FALSE FALSE FALSE
## chi FALSE FALSE FALSE FALSE FALSE FALSE
## cow FALSE FALSE FALSE FALSE FALSE FALSE
## duc FALSE FALSE FALSE FALSE FALSE FALSE
## eag FALSE FALSE FALSE FALSE FALSE FALSE
## ele FALSE FALSE FALSE FALSE FALSE FALSE
## fly FALSE FALSE FALSE FALSE FALSE FALSE
## fro FALSE FALSE FALSE FALSE  TRUE FALSE
## her FALSE FALSE FALSE FALSE FALSE FALSE
## lio FALSE FALSE FALSE  TRUE FALSE FALSE
## liz FALSE FALSE FALSE FALSE FALSE FALSE
## lob FALSE FALSE FALSE FALSE  TRUE FALSE
## man FALSE FALSE FALSE FALSE FALSE FALSE
## rab FALSE FALSE FALSE FALSE FALSE FALSE
## sal FALSE FALSE FALSE FALSE  TRUE FALSE
## spi FALSE FALSE FALSE  TRUE FALSE FALSE
## wha FALSE FALSE FALSE FALSE FALSE FALSE
```

```
row.names(animals)
```

```
##  [1] "ant" "bee" "cat" "cpl" "chi" "cow" "duc" "eag" "ele" "fly" "fro" "her"
## [13] "lio" "liz" "lob" "man" "rab" "sal" "spi" "wha"
```

```
missing_values <- animalsF %>%
  mutate(row_num = row_number()) %>%
  gather(key = "key", value = "value", -row_num) %>%
  filter(value %>% is.na()) %>%
  #count(row_num, sort = TRUE) %>%
  select(row_num)
```

```
missing_values
```

```
##   row_num
## 1      13
## 2      19
## 3      11
## 4      15
## 5      18
```

```
animalsF[missing_values$row_num,]
```

```
##     war fly ver  end  gro hai
## lio   1   0   1 <NA>    1   1
## spi   0   0   0 <NA>    0   1
## fro   0   0   1    1 <NA>   0
## lob   0   0   0    0 <NA>   0
## sal   0   0   1    0 <NA>   0
```

```
animalsFC <- animalsF  # animalsFC animals data completed
animalsFC
```

```
##     war fly ver  end  gro hai
## ant   0   0   0    0    1   0
## bee   0   1   0    0    1   1
## cat   1   0   1    0    0   1
## cpl   0   0   0    0    0   1
## chi   1   0   1    1    1   1
## cow   1   0   1    0    1   1
## duc   1   1   1    0    1   0
## eag   1   1   1    1    0   0
## ele   1   0   1    1    1   0
## fly   0   1   0    0    0   0
## fro   0   0   1    1 <NA>   0
## her   0   0   1    0    1   0
## lio   1   0   1 <NA>    1   1
## liz   0   0   1    0    0   0
## lob   0   0   0    0 <NA>   0
## man   1   0   1    1    1   1
## rab   1   0   1    0    1   1
```

```
## sal   0   0   1    0 <NA>   0
## spi   0   0   0 <NA>    0   1
## wha   1   0   1    1    1   0
```

```
animalsFC["lio","end"] <- 1
animalsFC["spi","end"] <- 0
animalsFC["fro","gro"] <- 0
animalsFC["lob","gro"] <- 0
animalsFC["sal","gro"] <- 0
animalsFC%>%is.na()
```

```
##       war   fly   ver   end   gro   hai
## ant FALSE FALSE FALSE FALSE FALSE FALSE
## bee FALSE FALSE FALSE FALSE FALSE FALSE
## cat FALSE FALSE FALSE FALSE FALSE FALSE
## cpl FALSE FALSE FALSE FALSE FALSE FALSE
## chi FALSE FALSE FALSE FALSE FALSE FALSE
## cow FALSE FALSE FALSE FALSE FALSE FALSE
## duc FALSE FALSE FALSE FALSE FALSE FALSE
## eag FALSE FALSE FALSE FALSE FALSE FALSE
## ele FALSE FALSE FALSE FALSE FALSE FALSE
## fly FALSE FALSE FALSE FALSE FALSE FALSE
## fro FALSE FALSE FALSE FALSE FALSE FALSE
## her FALSE FALSE FALSE FALSE FALSE FALSE
## lio FALSE FALSE FALSE FALSE FALSE FALSE
## liz FALSE FALSE FALSE FALSE FALSE FALSE
## lob FALSE FALSE FALSE FALSE FALSE FALSE
## man FALSE FALSE FALSE FALSE FALSE FALSE
## rab FALSE FALSE FALSE FALSE FALSE FALSE
## sal FALSE FALSE FALSE FALSE FALSE FALSE
## spi FALSE FALSE FALSE FALSE FALSE FALSE
## wha FALSE FALSE FALSE FALSE FALSE FALSE
```

matrice de distance

```
library(ade4)
df_animalsFC <- as.data.frame(lapply(animalsFC, as.numeric),row.names = c(row.names(animalsFC)))
df_animalsFC
```

```
##     war fly ver end gro hai
## ant   1   1   1   1   2   1
## bee   1   2   1   1   2   2
## cat   2   1   2   1   1   2
## cpl   1   1   1   1   1   2
## chi   2   1   2   2   2   2
## cow   2   1   2   1   2   2
## duc   2   2   2   1   2   1
## eag   2   2   2   2   1   1
## ele   2   1   2   2   2   1
## fly   1   2   1   1   1   1
## fro   1   1   2   2   1   1
```

```
## her   1   1   2   1   2   1
## lio   2   1   2   2   2   2
## liz   1   1   2   1   1   1
## lob   1   1   1   1   1   1
## man   2   1   2   2   2   2
## rab   2   1   2   1   2   2
## sal   1   1   2   1   1   1
## spi   1   1   1   1   1   2
## wha   2   1   2   2   2   1
```

```
df_animalsFC[df_animalsFC == "1"] <- 0
df_animalsFC[df_animalsFC == "2"] <- 1
df_animalsFC
```

```
##     war fly ver end gro hai
## ant   0   0   0   0   1   0
## bee   0   1   0   0   1   1
## cat   1   0   1   0   0   1
## cpl   0   0   0   0   0   1
## chi   1   0   1   1   1   1
## cow   1   0   1   0   1   1
## duc   1   1   1   0   1   0
## eag   1   1   1   1   0   0
## ele   1   0   1   1   1   0
## fly   0   1   0   0   0   0
## fro   0   0   1   1   0   0
## her   0   0   1   0   1   0
## lio   1   0   1   1   1   1
## liz   0   0   1   0   0   0
## lob   0   0   0   0   0   0
## man   1   0   1   1   1   1
## rab   1   0   1   0   1   1
## sal   0   0   1   0   0   0
## spi   0   0   0   0   0   1
## wha   1   0   1   1   1   0
```

```
jaccard_mat <- dist.binary(df_animalsFC, method = 1, diag = T, upper = T)
jaccard_mat
```

```
##           ant       bee       cat       cpl       chi       cow       duc
## ant 0.0000000 0.8164966 1.0000000 1.0000000 0.8944272 0.8660254 0.8660254
## bee 0.8164966 0.0000000 0.8944272 0.8164966 0.8164966 0.7745967 0.7745967
## cat 1.0000000 0.8944272 0.0000000 0.8164966 0.6324555 0.5000000 0.7745967
## cpl 1.0000000 0.8164966 0.8164966 0.0000000 0.8944272 0.8660254 1.0000000
## chi 0.8944272 0.8164966 0.6324555 0.8944272 0.0000000 0.4472136 0.7071068
## cow 0.8660254 0.7745967 0.5000000 0.8660254 0.4472136 0.0000000 0.6324555
## duc 0.8660254 0.7745967 0.7745967 1.0000000 0.7071068 0.6324555 0.0000000
## eag 1.0000000 0.9128709 0.7745967 1.0000000 0.7071068 0.8164966 0.6324555
## ele 0.8660254 0.9128709 0.7745967 1.0000000 0.4472136 0.6324555 0.6324555
## fly 1.0000000 0.8164966 1.0000000 1.0000000 1.0000000 1.0000000 0.8660254
## fro 1.0000000 1.0000000 0.8660254 1.0000000 0.7745967 0.8944272 0.8944272
## her 0.7071068 0.8660254 0.8660254 1.0000000 0.7745967 0.7071068 0.7071068
## lio 0.8944272 0.8164966 0.6324555 0.8944272 0.0000000 0.4472136 0.7071068
```

```
## liz 1.0000000 1.0000000 0.8164966 1.0000000 0.8944272 0.8660254 0.8660254
## lob 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## man 0.8944272 0.8164966 0.6324555 0.8944272 0.0000000 0.4472136 0.7071068
## rab 0.8660254 0.7745967 0.5000000 0.8660254 0.4472136 0.0000000 0.6324555
## sal 1.0000000 1.0000000 0.8164966 1.0000000 0.8944272 0.8660254 0.8660254
## spi 1.0000000 0.8164966 0.8164966 0.0000000 0.8944272 0.8660254 1.0000000
## wha 0.8660254 0.9128709 0.7745967 1.0000000 0.4472136 0.6324555 0.6324555
##           eag       ele       fly       fro       her       lio       liz
## ant 1.0000000 0.8660254 1.0000000 1.0000000 0.7071068 0.8944272 1.0000000
## bee 0.9128709 0.9128709 0.8164966 1.0000000 0.8660254 0.8164966 1.0000000
## cat 0.7745967 0.7745967 1.0000000 0.8660254 0.8660254 0.6324555 0.8164966
## cpl 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.8944272 1.0000000
## chi 0.7071068 0.4472136 1.0000000 0.7745967 0.7745967 0.0000000 0.8944272
## cow 0.8164966 0.6324555 1.0000000 0.8944272 0.7071068 0.4472136 0.8660254
## duc 0.6324555 0.6324555 0.8660254 0.8944272 0.7071068 0.7071068 0.8660254
## eag 0.0000000 0.6324555 0.8660254 0.7071068 0.8944272 0.7071068 0.8660254
## ele 0.6324555 0.0000000 1.0000000 0.7071068 0.7071068 0.4472136 0.8660254
## fly 0.8660254 1.0000000 0.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## fro 0.7071068 0.7071068 1.0000000 0.0000000 0.8164966 0.7745967 0.7071068
## her 0.8944272 0.7071068 1.0000000 0.8164966 0.0000000 0.7745967 0.7071068
## lio 0.7071068 0.4472136 1.0000000 0.7745967 0.7745967 0.0000000 0.8944272
## liz 0.8660254 0.8660254 1.0000000 0.7071068 0.7071068 0.8944272 0.0000000
## lob 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## man 0.7071068 0.4472136 1.0000000 0.7745967 0.7745967 0.0000000 0.8944272
## rab 0.8164966 0.6324555 1.0000000 0.8944272 0.7071068 0.4472136 0.8660254
## sal 0.8660254 0.8660254 1.0000000 0.7071068 0.7071068 0.8944272 0.0000000
## spi 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.8944272 1.0000000
## wha 0.6324555 0.0000000 1.0000000 0.7071068 0.7071068 0.4472136 0.8660254
##           lob       man       rab       sal       spi       wha
## ant 1.0000000 0.8944272 0.8660254 1.0000000 1.0000000 0.8660254
## bee 1.0000000 0.8164966 0.7745967 1.0000000 0.8164966 0.9128709
## cat 1.0000000 0.6324555 0.5000000 0.8164966 0.8164966 0.7745967
## cpl 1.0000000 0.8944272 0.8660254 1.0000000 0.0000000 1.0000000
## chi 1.0000000 0.0000000 0.4472136 0.8944272 0.8944272 0.4472136
## cow 1.0000000 0.4472136 0.0000000 0.8660254 0.8660254 0.6324555
## duc 1.0000000 0.7071068 0.6324555 0.8660254 1.0000000 0.6324555
## eag 1.0000000 0.7071068 0.8164966 0.8660254 1.0000000 0.6324555
## ele 1.0000000 0.4472136 0.6324555 0.8660254 1.0000000 0.0000000
## fly 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## fro 1.0000000 0.7745967 0.8944272 0.7071068 1.0000000 0.7071068
## her 1.0000000 0.7745967 0.7071068 0.7071068 1.0000000 0.7071068
## lio 1.0000000 0.0000000 0.4472136 0.8944272 0.8944272 0.4472136
## liz 1.0000000 0.8944272 0.8660254 0.0000000 1.0000000 0.8660254
## lob 0.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## man 1.0000000 0.0000000 0.4472136 0.8944272 0.8944272 0.4472136
## rab 1.0000000 0.4472136 0.0000000 0.8660254 0.8660254 0.6324555
## sal 1.0000000 0.8944272 0.8660254 0.0000000 1.0000000 0.8660254
## spi 1.0000000 0.8944272 0.8660254 1.0000000 0.0000000 1.0000000
## wha 1.0000000 0.4472136 0.6324555 0.8660254 1.0000000 0.0000000
```

Propose a graphical way to represent that distance matrix

```
jaccard_mat%>%str
```

```
##  'dist' num [1:190] 0.816 1 1 0.894 0.866 ...
##  - attr(*, "Labels")= chr [1:20] "ant" "bee" "cat" "cpl" ...
##  - attr(*, "Size")= int 20
##  - attr(*, "call")= language dist.binary(df = df_animalsFC, method = 1, diag = T, upper = T)
##  - attr(*, "Diag")= logi TRUE
##  - attr(*, "Upper")= logi TRUE
##  - attr(*, "method")= chr "JACCARD S3"
```

```
heatmap(as.matrix(jaccard_mat),symm = T,keep.dendro = FALSE,Rowv = NA)
```

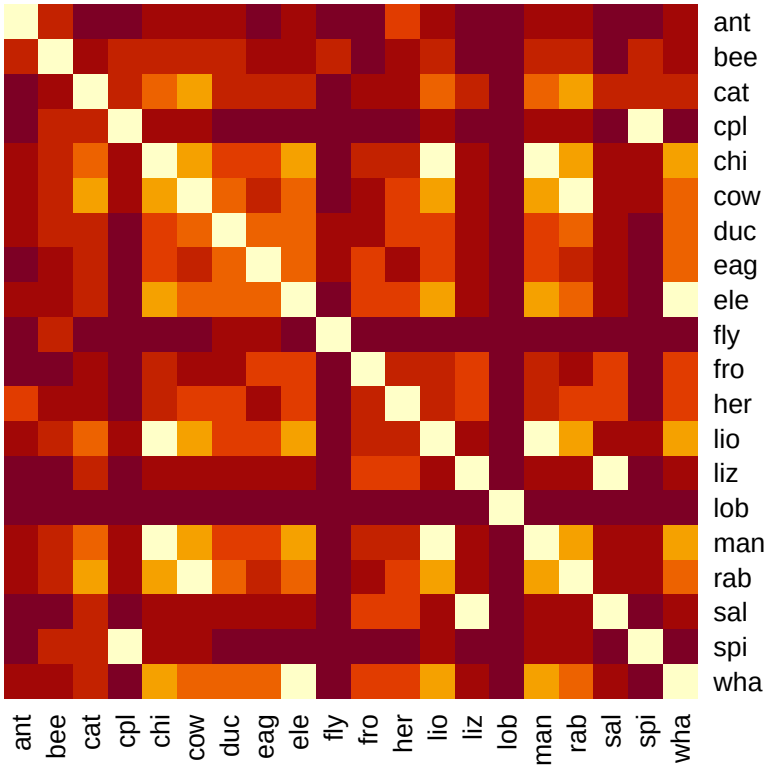


Figure 1: heatmap for distance matrix - Jaccard

simple matching matrix

```
simple_mat <- dist.binary(df_animalsFC, method = 2, diag = T, upper = T)
```

```
library(factoextra)
```

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

```
fviz_dist(simple_mat)
```

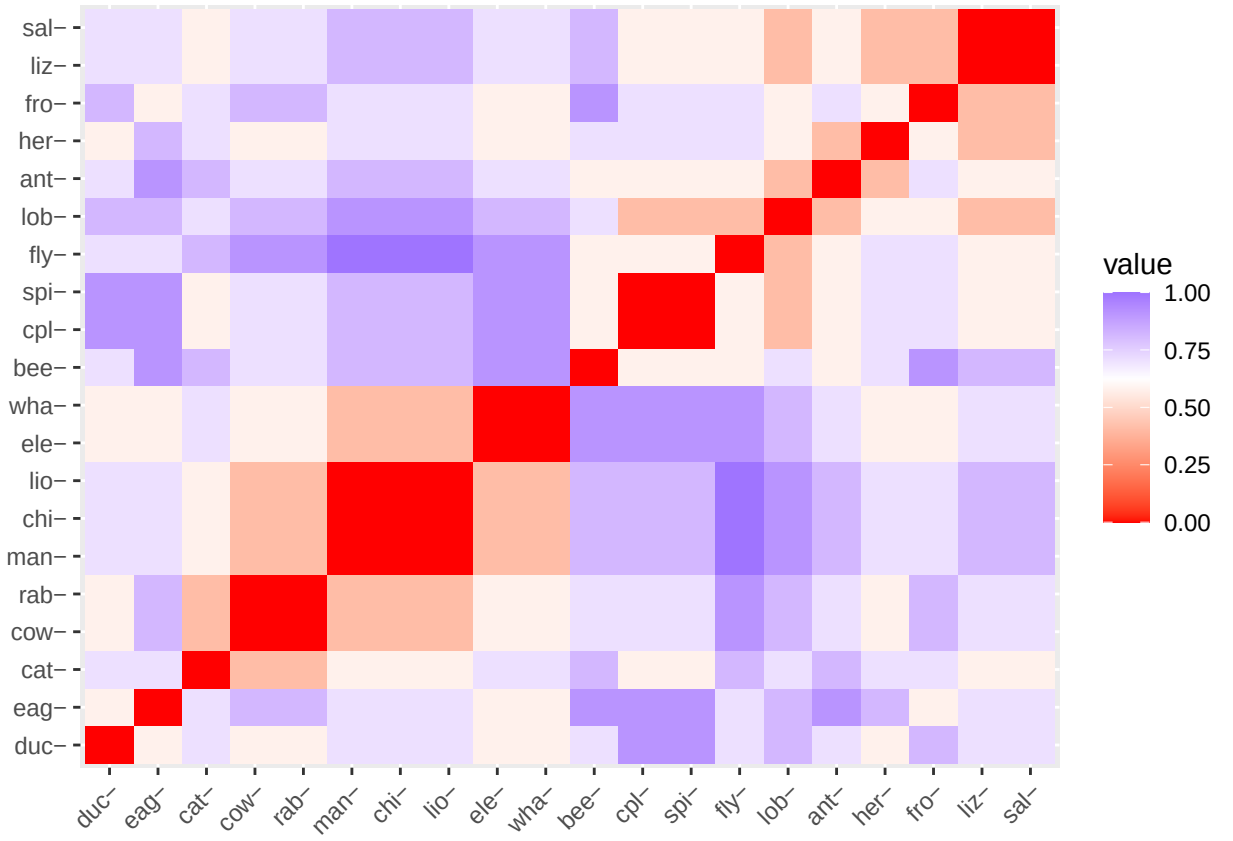


Figure 2: Heatmap for distance matrix - simple matching

exo 10

Prove that the distances based on the Simple Matching coefficient and the Jaccard coefficient satisfy A3. Prove that the distances proposed by Gower and Legendre (1986) do not satisfy A3. Hint: Proofs and counterexamples have to be adapted from in the paper: Gower, J. C., & Legendre, P. (1986). Metric and Euclidean properties of dissimilarity coefficients. Journal of classification, 3(1), 5-48.

(Inclure les preuves de : https://arxiv.org/pdf/1612.02696.pdf

https://www.researchgate.net/publication/24056133_Metric_and_Euclidean_properties_of_dissimilarity_coefficients/link/00463532856f248ec4000000/download

exo 11

The number of possible partition of n objects is given by the Stirling numbers of the second kind

$$\left\{ {n \atop k} \right\} = \frac{1}{k!}\sum_{i=1}^{k}(-1)^i \binom{n}{k}(k-i)^n$$

example function in R How many partitions of 3 objects are they in 6 object

```
#install.packages("multicool")
library(multicool)
```

```
## Loading required package: Rcpp
```

```
##
## Attaching package: 'Rcpp'
```

```
## The following object is masked from 'package:rsample':
##
##     populate
```

```
Stirling2(6,3)
```

```
## [1] 90
```

exo 13

```
df=read.csv("https://raw.githubusercontent.com/karkil2205/Cluster-Analysis/master/Hartigandata1.csv")
df%>%head
```

```
##   X            name energy protein fat calcium iron
## 1 1    Braised beef     11      29  28       1   26
## 2 2       Hamburger      8      30  17       1   27
## 3 3      Roast beef     18      21  39       1   20
## 4 4       Beefsteak     12      27  32       1   26
## 5 5     Canned beef      6      31  10       2   37
## 6 6 Broiled chicken      8      29   3       1   14
```

```
df%>%summary
```

```
##        X            name               energy          protein
##  Min.   : 1.0   Length:27          Min.   : 1.000   Min.   :10.00
##  1st Qu.: 7.5   Class :character   1st Qu.: 4.500   1st Qu.:23.50
##  Median :14.0   Mode  :character   Median : 6.000   Median :27.00
##  Mean   :14.0                      Mean   : 6.815   Mean   :27.19
##  3rd Qu.:20.5                      3rd Qu.: 8.500   3rd Qu.:31.00
##  Max.   :27.0                      Max.   :18.000   Max.   :37.00
##       fat           calcium           iron
##  Min.   : 1.00   Min.   : 1.000   Min.   : 5.00
##  1st Qu.: 5.00   1st Qu.: 1.000   1st Qu.:13.50
##  Median : 9.00   Median : 1.000   Median :25.00
##  Mean   :13.48   Mean   : 5.519   Mean   :23.81
##  3rd Qu.:22.50   3rd Qu.: 4.000   3rd Qu.:26.00
##  Max.   :39.00   Max.   :46.000   Max.   :60.00
```

```r
df<-df[1:8,c(3,4,6)]
df
```

```
##   energy protein calcium
## 1     11      29       1
## 2      8      30       1
## 3     18      21       1
## 4     12      27       1
## 5      6      31       2
## 6      8      29       1
## 7      5      36       2
## 8      5      37       2
```

```r
df[3,1]
```

```
## [1] 18
```

```r
df[3,1]<-13 # Error in line 3
df[6,1]<-4 # Error at line 6
df[7,3]<-1 # Error at line 7
df
```

```
##   energy protein calcium
## 1     11      29       1
## 2      8      30       1
## 3     13      21       1
## 4     12      27       1
## 5      6      31       2
## 6      4      29       1
## 7      5      36       1
## 8      5      37       2
```

```r
rownames(df)<-c("BB","HR","BR","BS","BC","CB","CC","BH")
colnames(df)<-c("Energy","Protein","Calcium")
df
```

```
##    Energy Protein Calcium
## BB     11      29       1
## HR      8      30       1
## BR     13      21       1
## BS     12      27       1
## BC      6      31       2
## CB      4      29       1
## CC      5      36       1
## BH      5      37       2
```

```r
km.res<-kmeans(df[1:8,],3,iter.max = 100)
km.res
```

```
## K-means clustering with 3 clusters of sizes 3, 3, 2
##
```

```
## Cluster means:
##   Energy  Protein  Calcium
## 1     12 25.66667 1.000000
## 2      6 30.00000 1.333333
## 3      5 36.50000 1.500000
##
## Clustering vector:
## BB HR BR BS BC CB CC BH
##  1  2  1  1  2  2  3  3
##
## Within cluster sum of squares by cluster:
## [1] 36.66667 10.66667  1.00000
##  (between_SS / total_SS =  81.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
k_clus <- km.res$cluster
k_clus
```

```
## BB HR BR BS BC CB CC BH
##  1  2  1  1  2  2  3  3
```

```
km.res$centers
```

```
##   Energy  Protein  Calcium
## 1     12 25.66667 1.000000
## 2      6 30.00000 1.333333
## 3      5 36.50000 1.500000
```

```
km.res$totss
```

```
## [1] 267.5
```

```
sum((df[1:8,]$Energy-mean(df[1:8,]$Energy))^2)+
  sum((df[1:8,]$Protein-mean(df[1:8,]$Protein))^2)+
  sum((df[1:8,]$Calcium-mean(df[1:8,]$Calcium))^2)
```

```
## [1] 267.5
```

```
7*var(df[1:8,]$Energy)+7*var(df[1:8,]$Protein)+7*var(df[1:8,]$Calcium)
```

```
## [1] 267.5
```

```
df
```

```
##    Energy Protein Calcium
## BB     11      29       1
## HR      8      30       1
## BR     13      21       1
## BS     12      27       1
## BC      6      31       2
## CB      4      29       1
## CC      5      36       1
## BH      5      37       2
```

rearrange with k 3 and the found clustering

```
df_ar <- df
k_clus <- as.data.frame(k_clus)
k_clus
```

```
##    k_clus
## BB      1
## HR      2
## BR      1
## BS      1
## BC      2
## CB      2
## CC      3
## BH      3
```

```
df_ar <- bind_cols(k_clus, df) %>% arrange(k_clus)
df_ar
```

```
##    k_clus Energy Protein Calcium
## BB      1     11      29       1
## BR      1     13      21       1
## BS      1     12      27       1
## HR      2      8      30       1
## BC      2      6      31       2
## CB      2      4      29       1
## CC      3      5      36       1
## BH      3      5      37       2
```

heatmap with data arranged

```
df_ar %>% select(-k_clus) %>% dist(diag = T, upper = T ) %>% as.matrix %>%
  heatmap(symm = T,keep.dendro = T, Rowv = NA, )
```

exo 15

```
k3_cluster <- km.res$centers
k3_cluster %>% as.data.frame() %>% rownames_to_column("cluster_num")
```

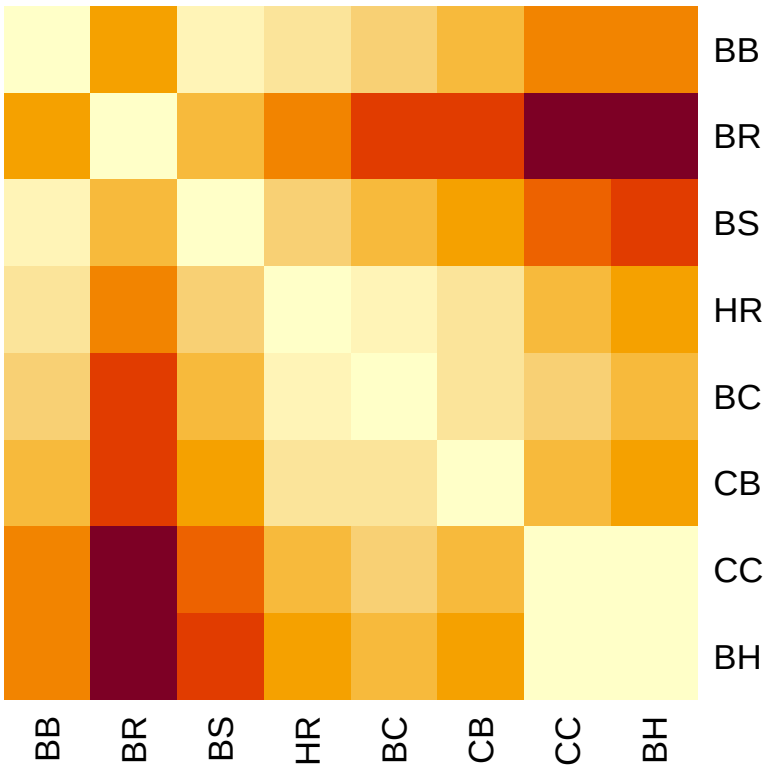


Figure 3: heatmap with data arranded according to the found clusters - k = 3

```
##   cluster_num Energy  Protein  Calcium
## 1           1     12 25.66667 1.000000
## 2           2      6 30.00000 1.333333
## 3           3      5 36.50000 1.500000
```

```
  #add_column()
```

```
k_clus %>% arrange(k_clus)
```

```
##    k_clus
## BB      1
## BR      1
## BS      1
## HR      2
## BC      2
## CB      2
## CC      3
## BH      3
```

```
cluster_nut <- data.frame(c("1","2","3"),c(("HR BC CB"),("BB BR BS"),("CC BH")))
names(cluster_nut) = c("cluster_num","cluster")
k3_cluster_num <-k3_cluster %>% as.data.frame() %>% rownames_to_column("cluster_num")
table15 <-left_join(k3_cluster_num,cluster_nut)
```

```
## Joining, by = "cluster_num"
```

```
table15 %>% select(c("cluster_num","cluster","Energy","Protein","Calcium"))
```

```
##   cluster_num  cluster Energy  Protein  Calcium
## 1           1 HR BC CB     12 25.66667 1.000000
## 2           2 BB BR BS      6 30.00000 1.333333
## 3           3    CC BH      5 36.50000 1.500000
```

```
km.res
```

```
## K-means clustering with 3 clusters of sizes 3, 3, 2
##
## Cluster means:
##   Energy  Protein  Calcium
## 1     12 25.66667 1.000000
## 2      6 30.00000 1.333333
## 3      5 36.50000 1.500000
##
## Clustering vector:
## BB HR BR BS BC CB CC BH
##  1  2  1  1  2  2  3  3
##
## Within cluster sum of squares by cluster:
## [1] 36.66667 10.66667  1.00000
##  (between_SS / total_SS =  81.9 %)
##
```

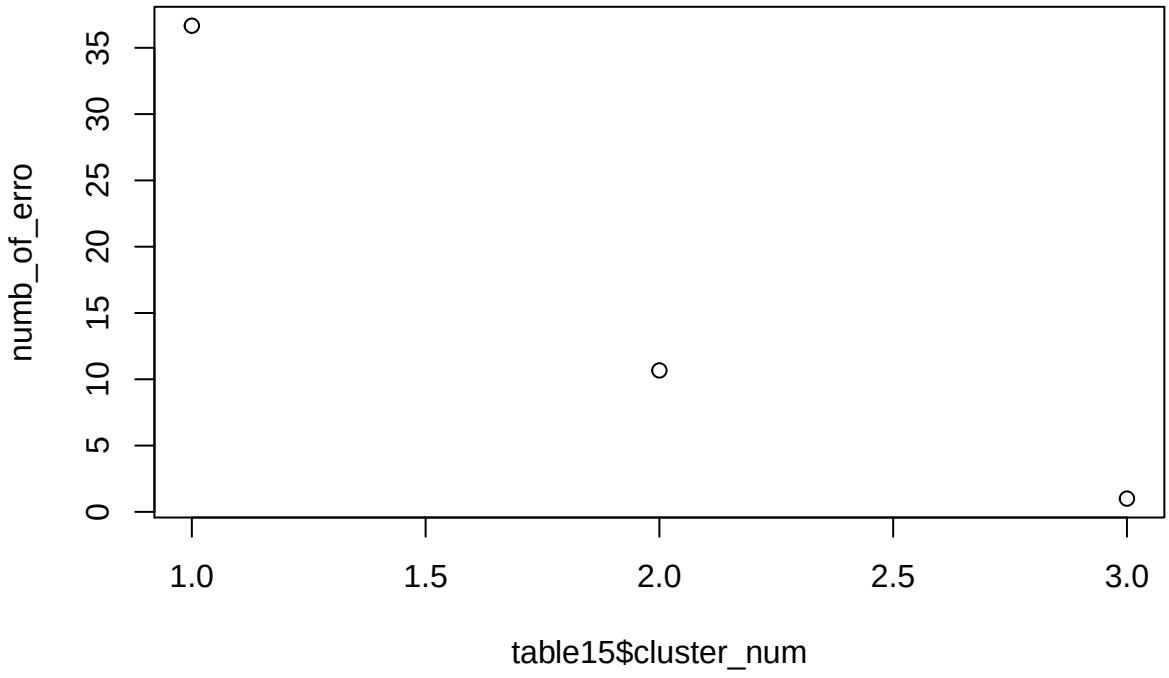
```
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
km.res$withinss
```

```
## [1] 36.66667 10.66667  1.00000
```

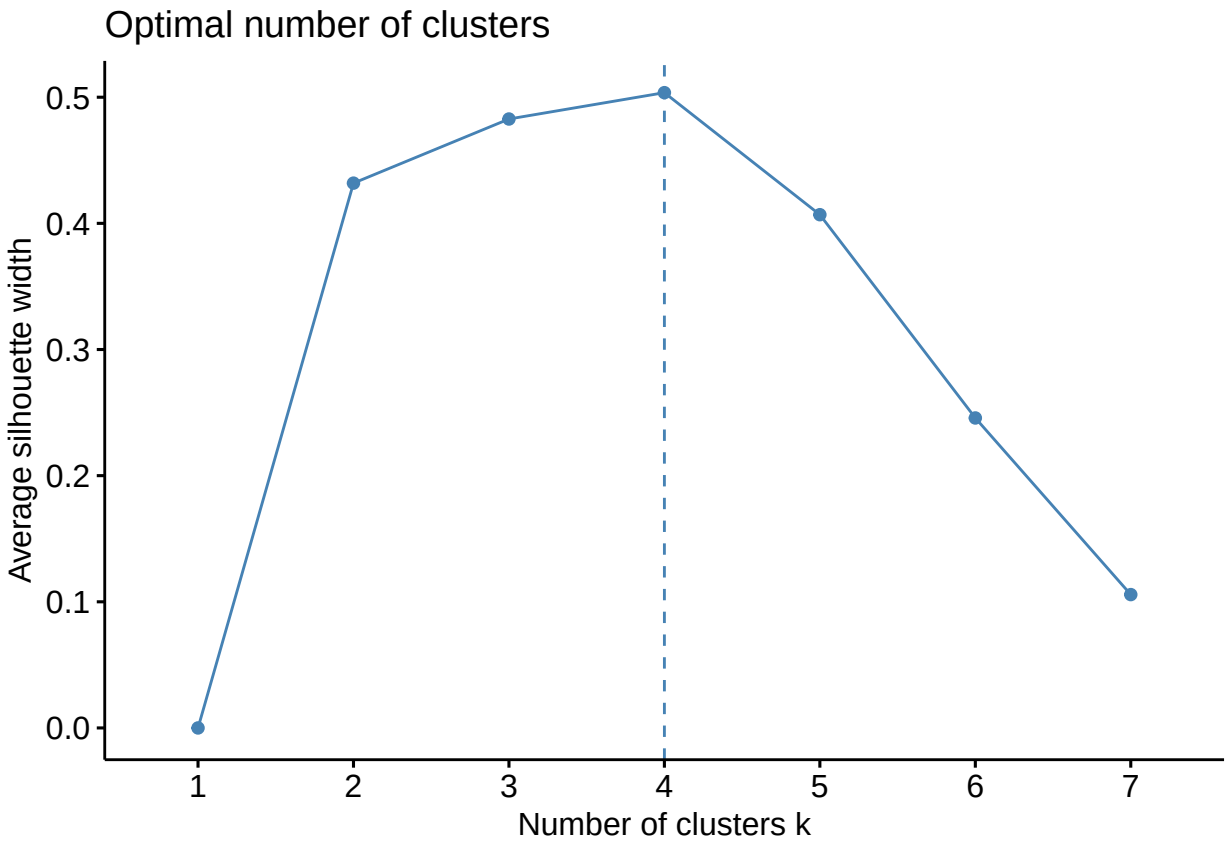
```
numb_of_erro <- km.res$withinss
```

```
plot(x = table15$cluster_num , y = numb_of_erro)
```



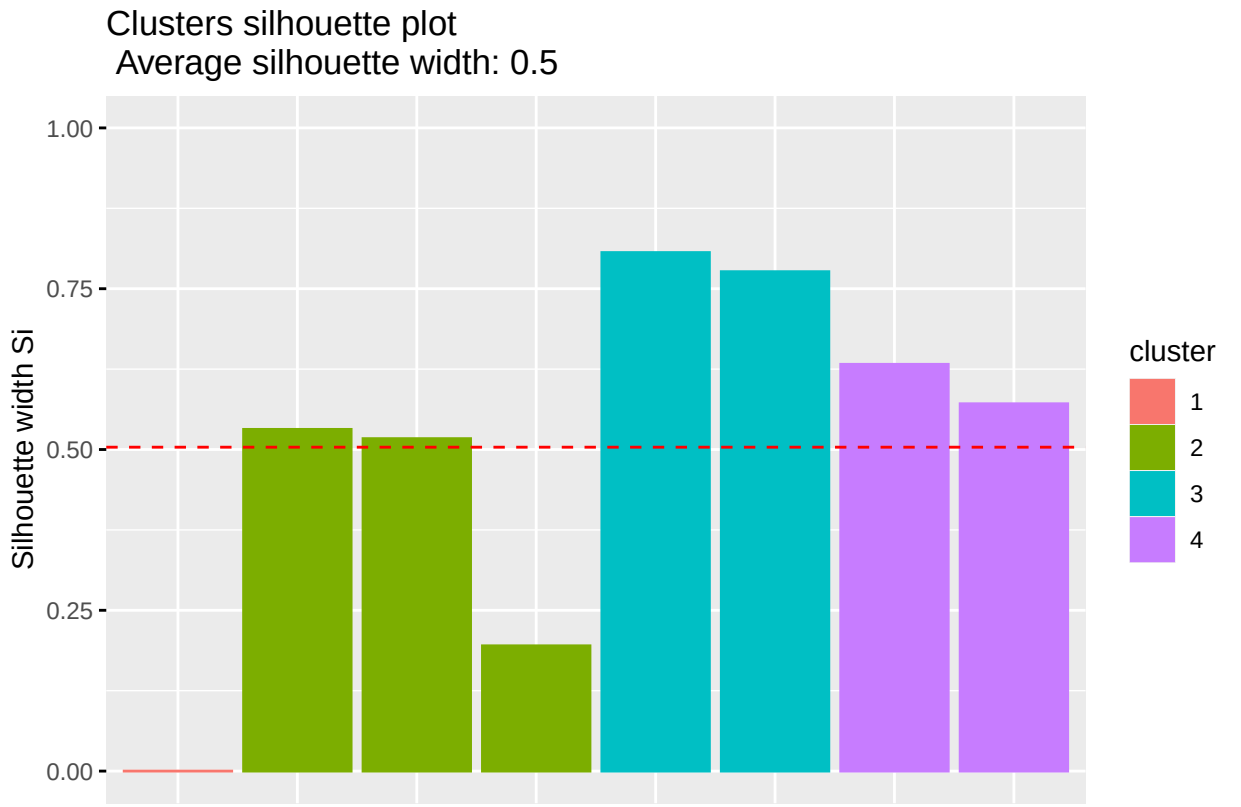
exo 18

```
fviz_nbclust(df, kmeans,method = "silhouette", k.max = 7)
```




```
k4 <- kmeans(df,4,iter.max = 100)
sil_obj <- silhouette(k4$cluster, dist(df))
sil_obj %>% fviz_silhouette(print.summary = T)
```

```
##   cluster size ave.sil.width
## 1       1    1          0.00
## 2       2    3          0.41
## 3       3    2          0.79
## 4       4    2          0.60
```

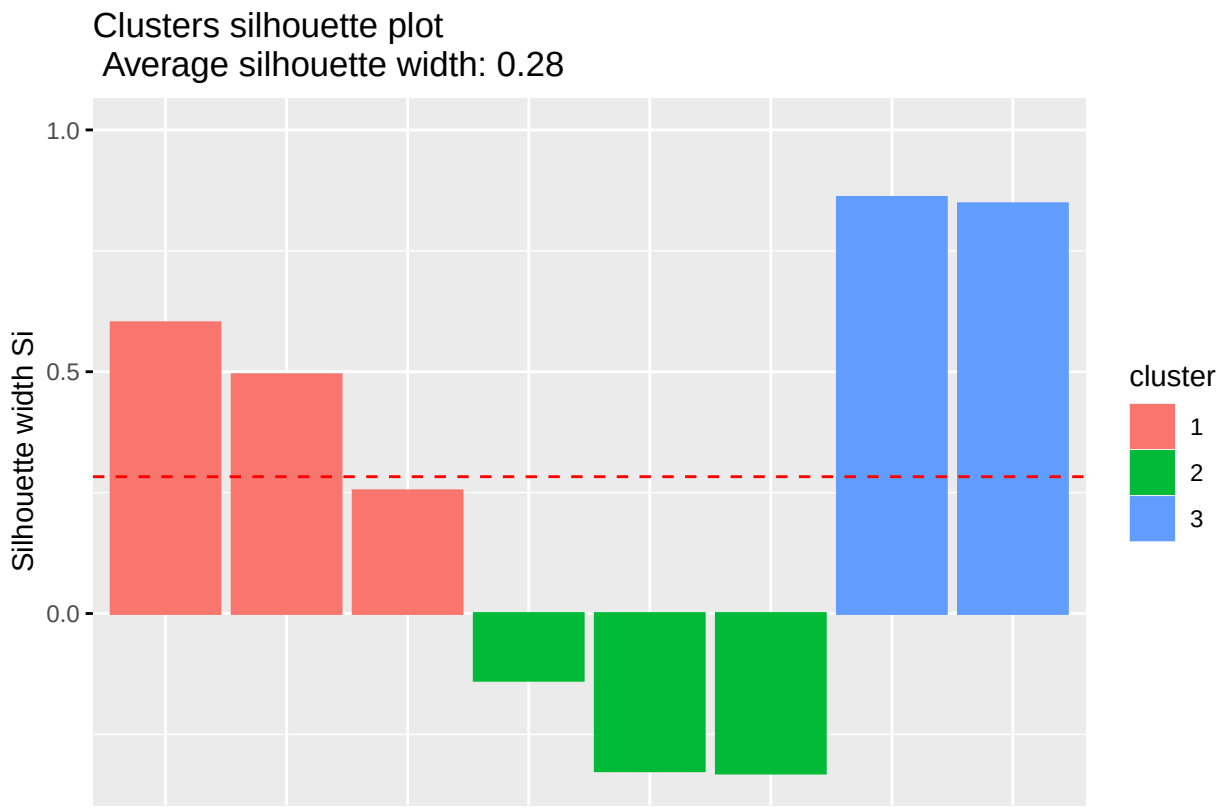



```
km.res$cluster
```

```
## BB HR BR BS BC CB CC BH
##  1  2  1  1  2  2  3  3
```

```
slobj <- silhouette(km.res$cluster, dist(df_ar))
slobj %>% fviz_silhouette()
```

```
##   cluster size ave.sil.width
## 1       1    3          0.45
## 2       2    3         -0.26
## 3       3    2          0.85
```



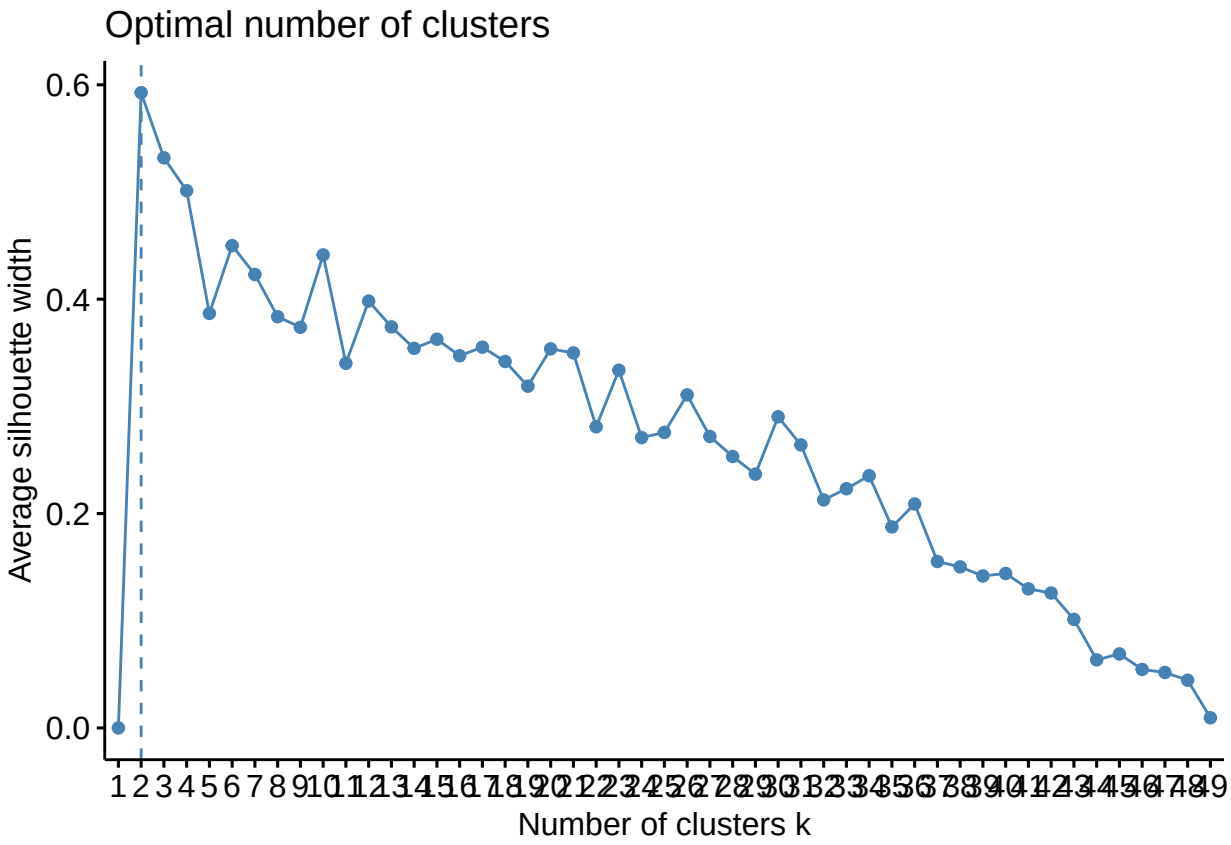
exo 19

```
library(datasets)
data("USArrests")
USArrests
```

```
##                Murder Assault UrbanPop Rape
## Alabama          13.2     236       58 21.2
## Alaska           10.0     263       48 44.5
## Arizona           8.1     294       80 31.0
## Arkansas          8.8     190       50 19.5
## California        9.0     276       91 40.6
## Colorado          7.9     204       78 38.7
## Connecticut       3.3     110       77 11.1
## Delaware          5.9     238       72 15.8
## Florida          15.4     335       80 31.9
## Georgia          17.4     211       60 25.8
## Hawaii            5.3      46       83 20.2
## Idaho             2.6     120       54 14.2
## Illinois         10.4     249       83 24.0
## Indiana           7.2     113       65 21.0
## Iowa              2.2      56       57 11.3
## Kansas            6.0     115       66 18.0
## Kentucky          9.7     109       52 16.3
```

```
## Louisiana        15.4     249       66 22.2
## Maine             2.1      83       51  7.8
## Maryland         11.3     300       67 27.8
## Massachusetts     4.4     149       85 16.3
## Michigan         12.1     255       74 35.1
## Minnesota         2.7      72       66 14.9
## Mississippi      16.1     259       44 17.1
## Missouri          9.0     178       70 28.2
## Montana           6.0     109       53 16.4
## Nebraska          4.3     102       62 16.5
## Nevada           12.2     252       81 46.0
## New Hampshire     2.1      57       56  9.5
## New Jersey        7.4     159       89 18.8
## New Mexico       11.4     285       70 32.1
## New York         11.1     254       86 26.1
## North Carolina   13.0     337       45 16.1
## North Dakota      0.8      45       44  7.3
## Ohio              7.3     120       75 21.4
## Oklahoma          6.6     151       68 20.0
## Oregon            4.9     159       67 29.3
## Pennsylvania      6.3     106       72 14.9
## Rhode Island      3.4     174       87  8.3
## South Carolina   14.4     279       48 22.5
## South Dakota      3.8      86       45 12.8
## Tennessee        13.2     188       59 26.9
## Texas            12.7     201       80 25.5
## Utah              3.2     120       80 22.9
## Vermont           2.2      48       32 11.2
## Virginia          8.5     156       63 20.7
## Washington        4.0     145       73 26.2
## West Virginia     5.7      81       39  9.3
## Wisconsin         2.6      53       66 10.8
## Wyoming           6.8     161       60 15.6
```

```
fviz_nbclust(USArrests, kmeans,method = "silhouette", k.max = 49)
```

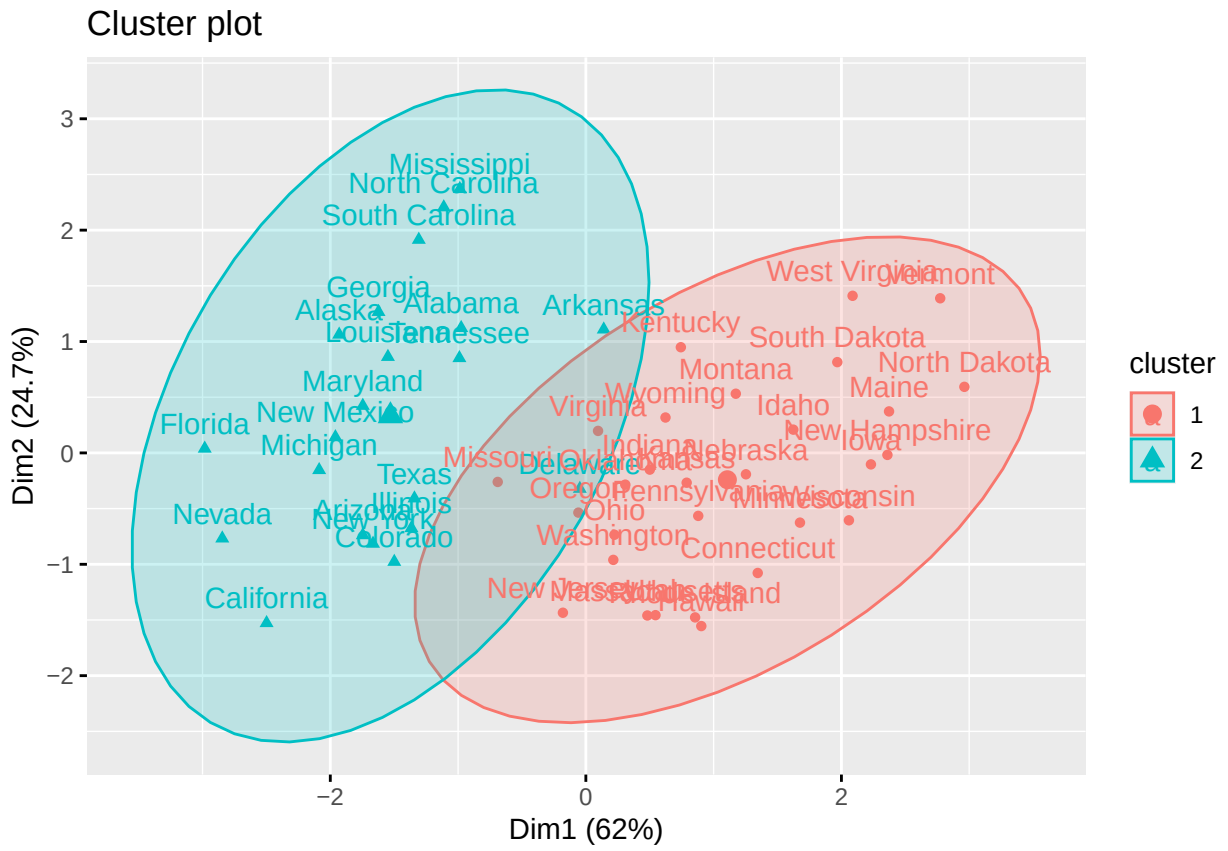



```
km.arrest <- kmeans(USArrests,2,iter.max = 100)
km.arrest
```

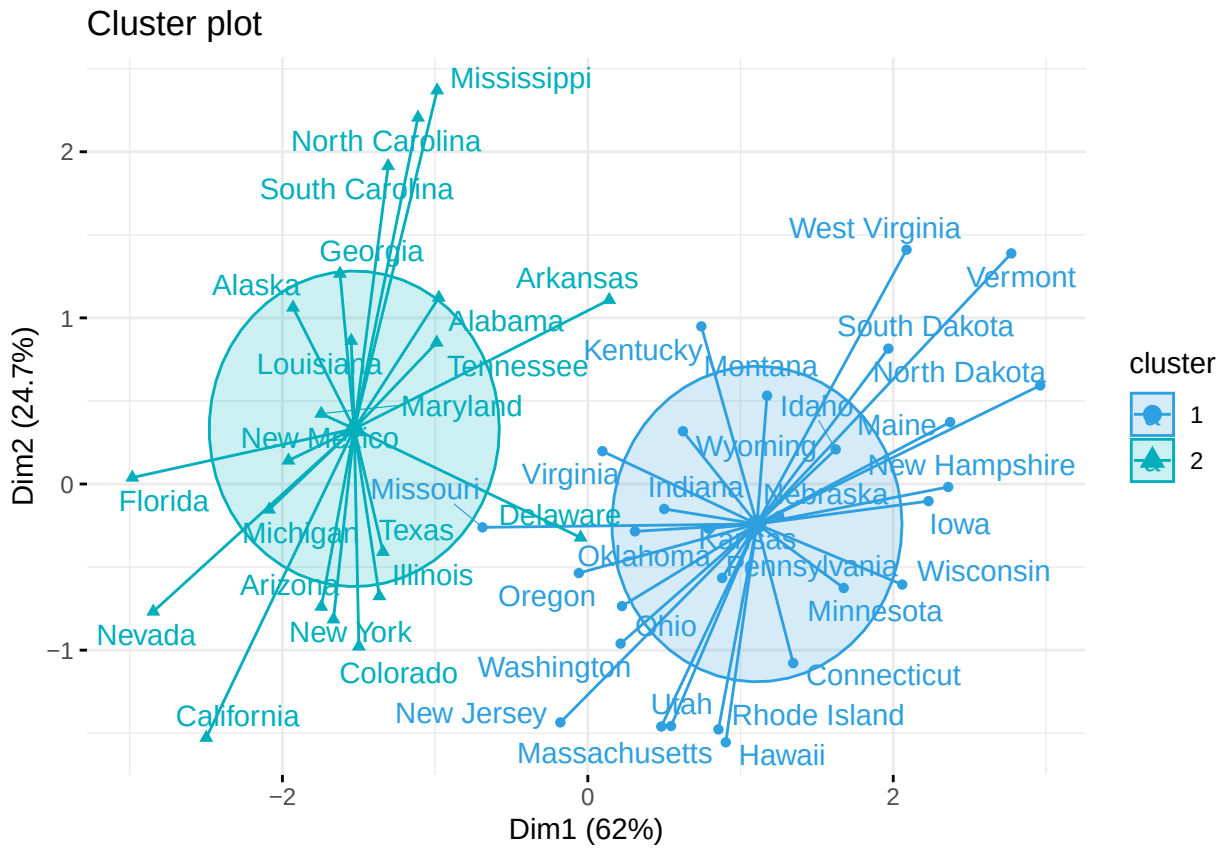
```
## K-means clustering with 2 clusters of sizes 29, 21
##
## Cluster means:
##      Murder  Assault UrbanPop     Rape
## 1  4.841379 109.7586 64.03448 16.24828
## 2 11.857143 255.0000 67.61905 28.11429
##
## Clustering vector:
##        Alabama         Alaska        Arizona       Arkansas     California
##              2              2              2              2              2
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              2              1              2              2              2
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              1              1              2              1              1
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              1              1              2              1              2
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              1              2              1              2              1
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              1              1              2              1              1
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              2              2              2              1              1
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
```

```
##              1              1              1              1              2
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              1              2              2              1              1
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              1              1              1              1              1
##
## Within cluster sum of squares by cluster:
## [1] 54762.30 41636.73
##  (between_SS / total_SS =  72.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
fviz_cluster(km.arrest, USArrests, ellipse.type = "norm")
```

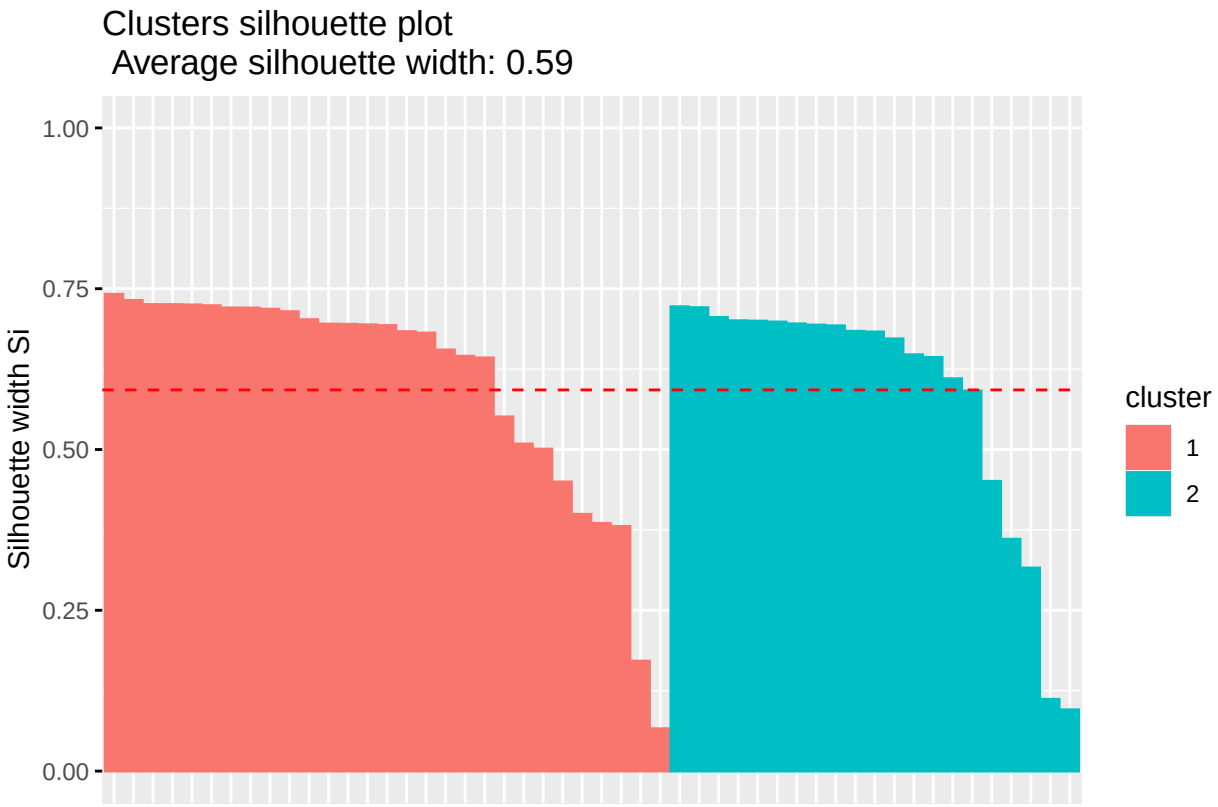



```
fviz_cluster(km.arrest, USArrests,
palette = c("#2E9FDF", "#00AFBB", "#E7B800"),
ellipse.type = "euclid", # Concentration ellipse
star.plot = TRUE, # Add segments from centroids to items
repel = TRUE, # Avoid label overplotting (slow)
ggtheme = theme_minimal()
)
```




```
silhouette(km.arrest$cluster,dist(USArrests))%>%
fviz_silhouette()
```

```
##   cluster size ave.sil.width
## 1       1   29          0.60
## 2       2   21          0.58
```


Clusters silhouette plot
Average silhouette width: 0.59
Silhouette width Si
1.00
0.75
0.50
0.25
0.00
cluster
1
2