COMPITI A CASA 1- Casadio Sara

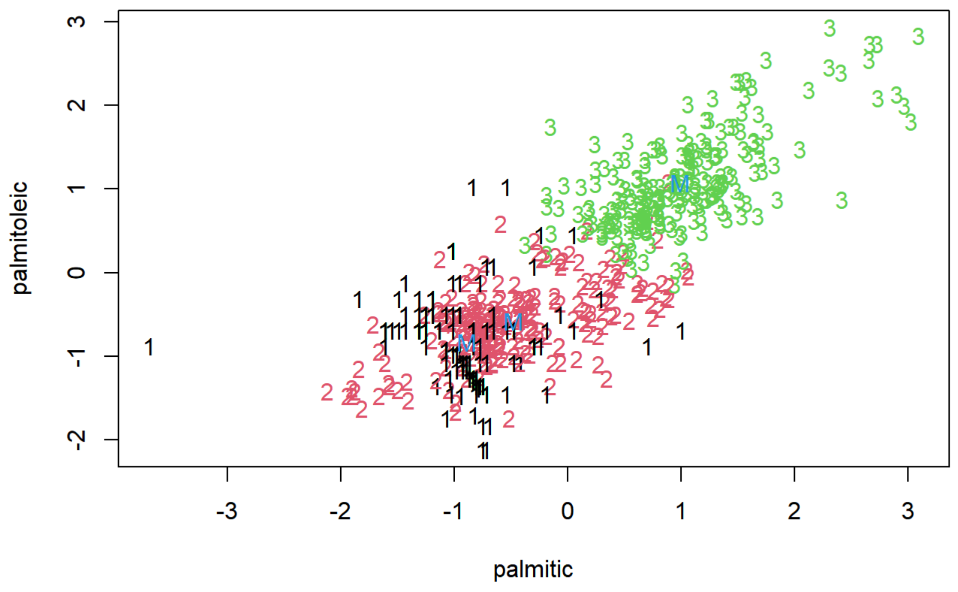
04-10-2022

1. Esegui K-mean per i dati dell'olio d'oliva con K = 3 e K = 9 con dati in scala e non in scala. Assumendo che le macroaree siano i "cluster per K = 3, utilizzare la tabella per confrontare le macroaree con il clustering e confrontare la qualità dei due clustering con K = 3. Fare lo stesso per le regioni e il K = 9 cluster.

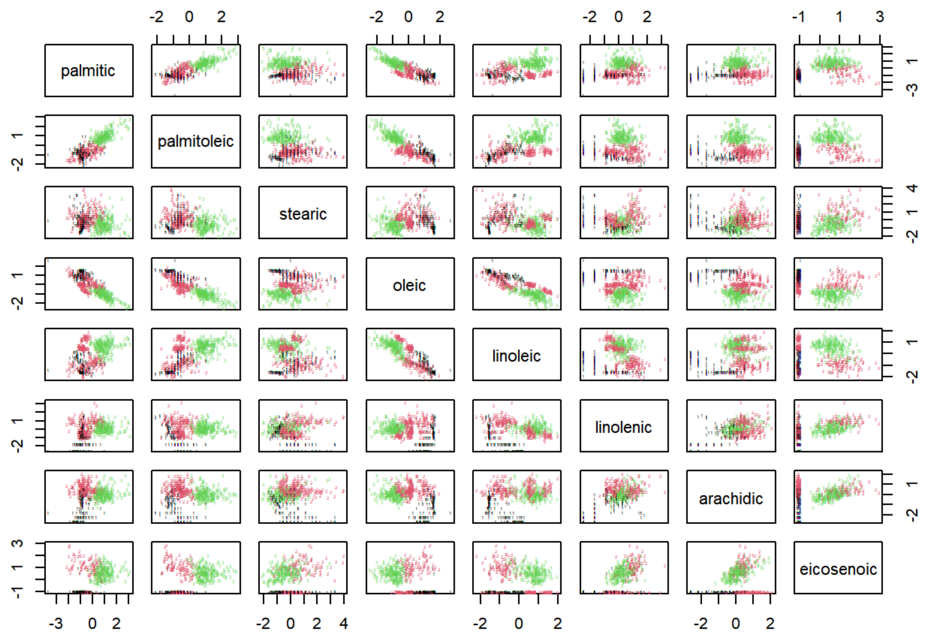
```
#install.packages("pdfCluster")
library(pdfCluster)
```

```
## pdfCluster 1.0-3
```

```
data("oliveoil")
olive<-oliveoil[,3:10] #consider only the numerical variables in order to run the kmeans
#pairs(olivescal, cex=0.3, col=oliveoil[,1])
olivescal<-scale(olive)

#install.packages("fpc")
library(fpc)

#scaled dataset and k=3
set.seed(665544)
kolive.3<-kmeans(olivescal,centers = 3, nstart=100)
plot(olivescal, col=kolive.3$cluster, pch=clusym[kolive.3$cluster])
points(kolive.3$centers,pch="M", cex=1, col=4)
```



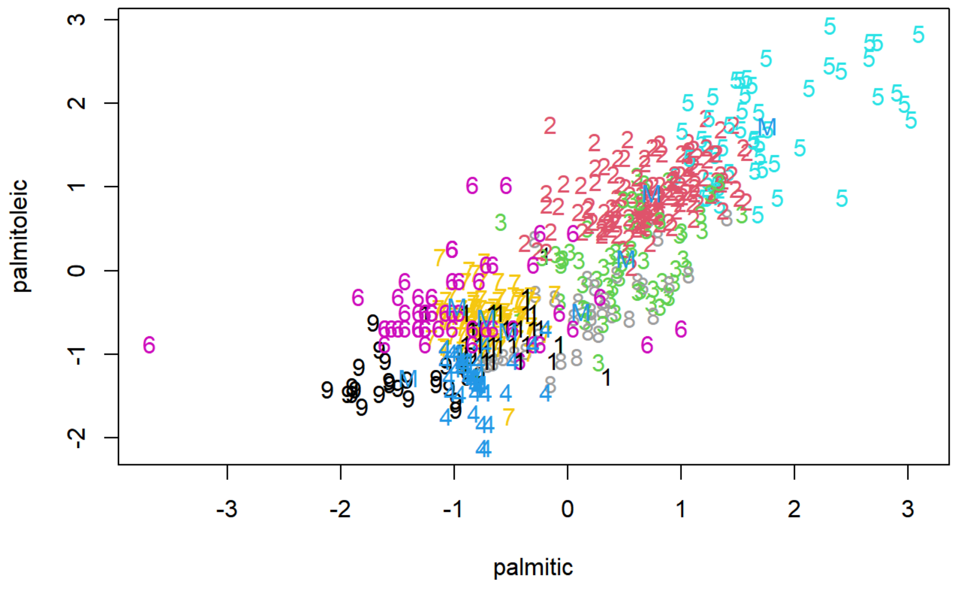
```
pairs(olivescal, cex=0.3, col=kolive.3$cluster, pch=clusym[kolive.3$cluster])
```



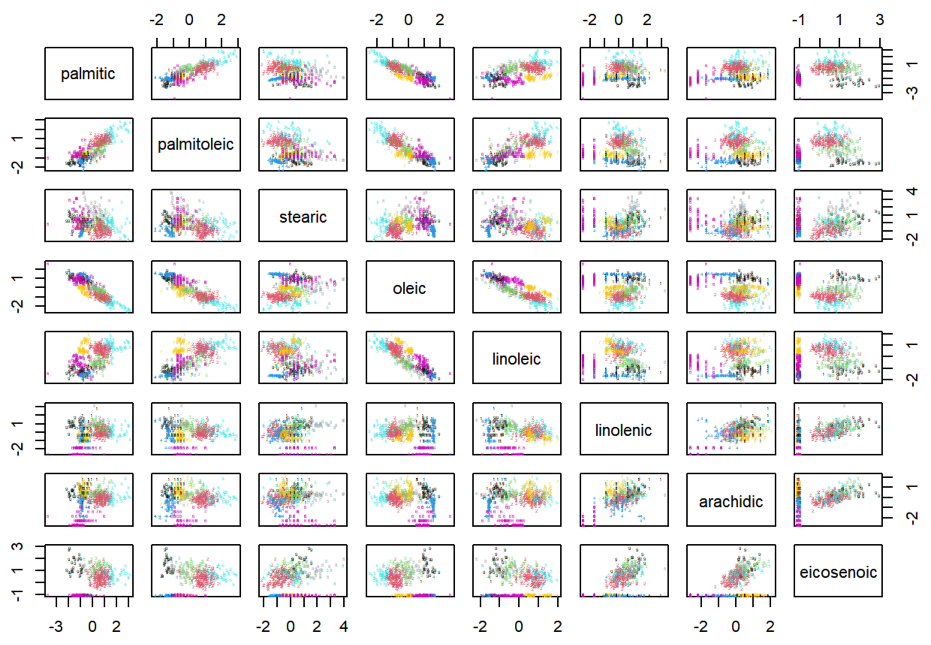
```
str(kolive.3)
```

```
## List of 9
##  $ cluster     : int [1:572] 2 2 2 2 2 2 2 2 2 1 ...
##  $ centers     : num [1:3, 1:8] -0.888 -0.473 0.996 -0.832 -0.581 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 4568
##  $ withinss    : num [1:3] 581 1174 564
##  $ tot.withinss: num 2320
##  $ betweenss   : num 2248
##  $ size        : int [1:3] 123 230 219
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#scaled dataset and k=9
set.seed(665544)
kolive.9<-kmeans(olivescal,centers = 9, nstart=100)
plot(olivescal, col=kolive.9$cluster, pch=clusym[kolive.9$cluster])
points(kolive.9$centers,pch="M", cex=1, col=4)
```



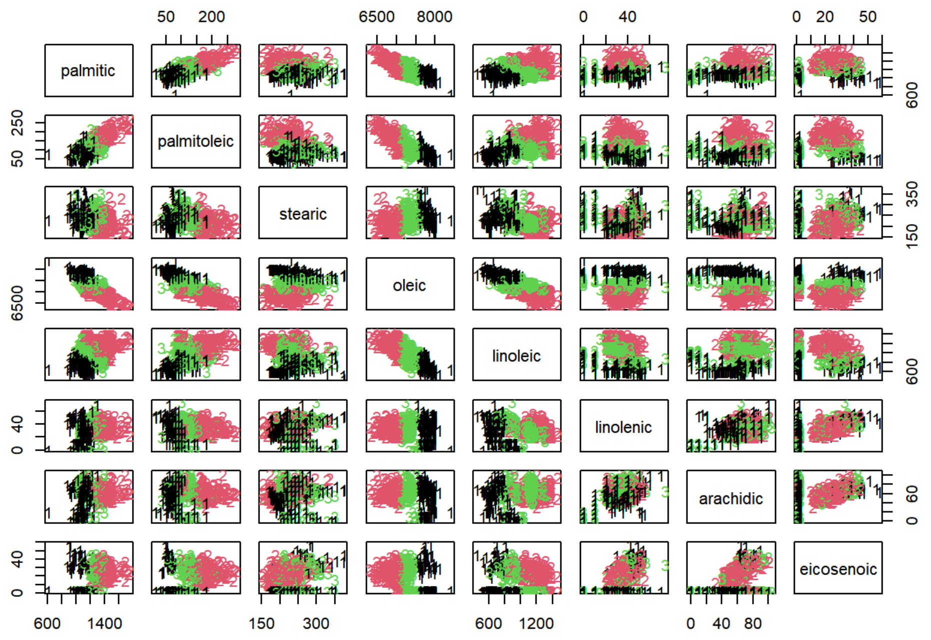
```
pairs(olivescal, cex=0.3, col=kolive.9$cluster, pch=clusym[kolive.9$cluster])
```



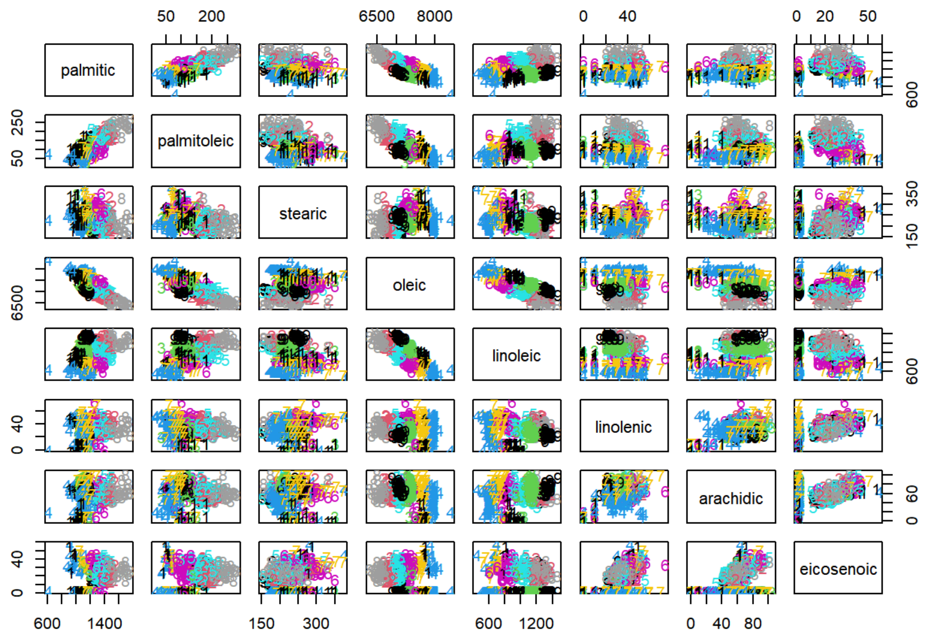
```
str(kolive.9)
```

```
## List of 9
##  $ cluster     : int [1:572] 9 9 9 9 9 9 9 9 9 9 ...
##  $ centers     : num [1:9, 1:8] -0.524 0.74 0.52 -0.814 1.763 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:9] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 4568
##  $ withinss    : num [1:9] 69.2 207.1 119.7 87.1 135.4 ...
##  $ tot.withinss: num 1020
##  $ betweenss   : num 3548
##  $ size        : int [1:9] 35 144 62 58 51 60 98 36 28
##  $ iter        : int 6
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#unscaled dataset and k=3
set.seed(665544)
olive.3<-kmeans(olive,centers = 3, nstart=100)
plot(olive, col=olive.3$cluster, pch=clusym[olive.3$cluster])
points(olive.3$centers,pch="M", cex=1, col=4)
```



```
#unscaled dataset and k=9
set.seed(665544)
olive.9<-kmeans(olive,centers = 9, nstart=100)
plot(olive, col=olive.9$cluster, pch=clusym[olive.9$cluster])
points(olive.9$centers,pch="M", cex=1, col=4)
```



```
#with the function table we can see where a specimen of olive oil is classified based on the value of the variables(chemical measurements on the acid components)

table(kolive.3$cluster,oliveoil$macro.area)
```

```
##
##     South Sardinia Centre.North
##   1     2        0          121
##   2   102       98           30
##   3   219        0            0
```

```
#for the first cluster we can find 2 specimens of olive oil in the macro-area south anche 121 in the centre.north
table(olive.3$cluster,oliveoil$macro.area)
```

```
##
##     South Sardinia Centre.North
##   1    42        0          134
##   2   190       22            0
##   3    91       76           17
```

```
#for the first cluster we have 42 specimens in south and 134 in centre.north
table(kolive.9$cluster,oliveoil$region)
```

```
##
##     Apulia.north Calabria Apulia.south Sicily Sardinia.inland Sardinia.coast
##   1            0        1            0      0               0              0
##   2            0        0          144      0               0              0
##   3            2       32           12     16               0              0
##   4            1        0            0      0               0              0
##   5            0        0           49      2               0              0
##   6            0        0            0      0               0              0
##   7            0        0            0      0              65             33
##   8            0       23            1     12               0              0
##   9           22        0            0      6               0              0
##
##     Liguria.east Liguria.west Umbria
##   1           33            0      1
##   2            0            0      0
##   3            0            0      0
##   4            7            0     50
##   5            0            0      0
##   6           10           50      0
##   7            0            0      0
##   8            0            0      0
##   9            0            0      0
```

```
#for the fisrt cluster we can find 1 specimens in region calabria,33 in liguria.east and 1 in umbria
table(olive.9$cluster,oliveoil$region)
```

```
##
##     Apulia.north Calabria Apulia.south Sicily Sardinia.inland Sardinia.coast
##   1            2        0            0      5               0              0
##   2            0        0           75      2               0              0
##   3            0        0            3      1              65              0
##   4           13        0            0      2               0              0
##   5            0        3           77      7               0              0
##   6            1       48            5     11               0              0
##   7            9        5            0      7               0              0
##   8            0        0           30      1               0              0
##   9            0        0           16      0               0             33
##
##     Liguria.east Liguria.west Umbria
##   1            0           39      0
##   2            0            0      0
##   3            0            3      0
##   4           16            2     51
##   5            0            0      0
##   6            2            1      0
##   7           32            5      0
##   8            0            0      0
##   9            0            0      0
```

```
#for the first cluster we have 2 specimens in Apulia.north, 5 in sicily and 39 in liguria.west

#we can't compare k=3 and k=9 since the are relative different things: in the first we see from which macro.area the oil com
es, and in the second from which region.

#for the scaled olive with k=3
str(kolive.3)
```

```
## List of 9
##  $ cluster     : int [1:572] 2 2 2 2 2 2 2 2 2 1 ...
##  $ centers     : num [1:3, 1:8] -0.888 -0.473 0.996 -0.832 -0.581 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 4568
##  $ withinss    : num [1:3] 581 1174 564
##  $ tot.withinss: num 2320
##  $ betweenss   : num 2248
##  $ size        : int [1:3] 123 230 219
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 2320
#totss       : num 4568


#for unscaled olive with k=3
str(olive.3)
```

```
## List of 9
##  $ cluster     : int [1:572] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:8] 1079.9 1384.8 1200.6 78.2 176.5 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 1.47e+08
##  $ withinss    : num [1:3] 7631152 12636160 10226254
##  $ tot.withinss: num 30493566
##  $ betweenss   : num 1.16e+08
##  $ size        : int [1:3] 176 212 184
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 30493566
#totss       : num 146755255


#for the scaled olive with k=9
str(kolive.9)
```

```
## List of 9
##  $ cluster     : int [1:572] 9 9 9 9 9 9 9 9 9 9 ...
##  $ centers     : num [1:9, 1:8] -0.524 0.74 0.52 -0.814 1.763 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:9] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 4568
##  $ withinss    : num [1:9] 69.2 207.1 119.7 87.1 135.4 ...
##  $ tot.withinss: num 1020
##  $ betweenss   : num 3548
##  $ size        : int [1:9] 35 144 62 58 51 60 98 36 28
##  $ iter        : int 6
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 1020
#totss       : num 4568



#for unscaled olive with k=9
str(olive.9)
```

```
## List of 9
##  $ cluster     : int [1:572] 4 7 4 4 7 4 4 7 7 4 ...
##  $ centers     : num [1:9, 1:8] 1020 1407 1103 1070 1365 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:9] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 1.47e+08
##  $ withinss    : num [1:9] 917598 1051934 576972 1423081 1509159 ...
##  $ tot.withinss: num 9336608
##  $ betweenss   : num 1.37e+08
##  $ size        : int [1:9] 46 77 72 84 87 68 58 31 49
##  $ iter        : int 5
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 9336608
#totss       : num 146755255



#we have to use the scaled datasets because their total within sum of square are less than in the unscaled datasets,
#and we know that the more less is the tot.withinss, the better is the model
#so the clustering is more precise in the first case.
```

2.

```
library(pdfCluster)
data("oliveoil")
olive<-oliveoil[,3:10]

#scaled dataset
library(fpc)
olivescal<-scale(olive)
olivescal.mat<-data.matrix(olivescal)

set.seed(665544)
kolive.3<-kmeans(olivescal.mat,centers = 3, nstart=100)
str(kolive.3)
```

```
## List of 9
##  $ cluster     : int [1:572] 2 2 2 2 2 2 2 2 2 1 ...
##  $ centers     : num [1:3, 1:8] -0.888 -0.473 0.996 -0.832 -0.581 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 4568
##  $ withinss    : num [1:3] 581 1174 564
##  $ tot.withinss: num 2320
##  $ betweenss   : num 2248
##  $ size        : int [1:3] 123 230 219
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
q=12345
olivescal1<-olivescal.mat*q

set.seed(665544)
kolive.3.1<-kmeans(olivescal1,centers = 3, nstart=100)
str(kolive.3.1)
```

```
## List of 9
##  $ cluster     : int [1:572] 2 2 2 2 2 2 2 2 2 1 ...
##  $ centers     : num [1:3, 1:8] -10966 -5839 12292 -10268 -7176 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 6.96e+11
##  $ withinss    : num [1:3] 8.86e+10 1.79e+11 8.60e+10
##  $ tot.withinss: num 3.54e+11
##  $ betweenss   : num 3.43e+11
##  $ size        : int [1:3] 123 230 219
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
kolive.3$centers
```

```
##     palmitic palmitoleic     stearic      oleic   linoleic  linolenic
## 1 -0.8883305  -0.8317752 -0.02819753  1.2477891 -1.0234607 -1.0940902
## 2 -0.4730087  -0.5813223  0.42040240  0.2684474 -0.1323053  0.3462298
## 3  0.9956925   1.0776826 -0.42568153 -0.9827441  0.7137711  0.2508687
##    arachidic eicosenoic
## 1 -1.3913028 -0.9834754
## 2  0.6117049 -0.0632071
## 3  0.1389869  0.6187448
```

```
kolive.3.1$centers
```

```
##     palmitic palmitoleic    stearic      oleic   linoleic  linolenic  arachidic
## 1 -10966.440  -10268.265  -348.0985  15403.956 -12634.622 -13506.544 -17175.633
## 2  -5839.292   -7176.424  5189.8676   3313.983  -1633.309   4274.206   7551.497
## 3  12291.823   13303.991 -5255.0385 -12131.976   8811.505   3096.974   1715.793
##    eicosenoic
## 1 -12141.0038
## 2   -780.2916
## 3   7638.4043
```

```
table(kolive.3$cluster)
```

```
##
##   1   2   3
## 123 230 219
```

```
table(kolive.3.1$cluster)
```

```
##
##   1   2   3
## 123 230 219
```

```
#unscaled dataset
olive.mat<-data.matrix(olive)
olive1<-olive.mat*q

set.seed(665544)
olive.3<-kmeans(olive.mat,centers = 3, nstart=100)
str(olive.9)
```

```
## List of 9
##  $ cluster     : int [1:572] 4 7 4 4 7 4 4 7 7 4 ...
##  $ centers     : num [1:9, 1:8] 1020 1407 1103 1070 1365 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:9] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 1.47e+08
##  $ withinss    : num [1:9] 917598 1051934 576972 1423081 1509159 ...
##  $ tot.withinss: num 9336608
##  $ betweenss   : num 1.37e+08
##  $ size        : int [1:9] 46 77 72 84 87 68 58 31 49
##  $ iter        : int 5
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
set.seed(665544)
olive.3.1<-kmeans(olive1,centers = 3, nstart=100)
str(olive.3.1)
```

```
## List of 9
##  $ cluster     : int [1:572] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:8] 13331267 17095962 14821179 964944 2178776 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 2.24e+16
##  $ withinss    : num [1:3] 1.16e+15 1.93e+15 1.56e+15
##  $ tot.withinss: num 4.65e+15
##  $ betweenss   : num 1.77e+16
##  $ size        : int [1:3] 176 212 184
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
olive.3$centers
```

```
##   palmitic palmitoleic  stearic    oleic  linoleic linolenic arachidic
## 1 1079.892    78.16477 236.2500 7812.699  697.1477  28.15341  46.89773
## 2 1384.849   176.49057 216.8113 6890.741 1197.9717  34.07547  62.53774
## 3 1200.582   113.87500 235.6902 7317.652 1001.0543  32.94022  63.69565
##   eicosenoic
## 1   10.06818
## 2   22.42453
## 3   15.14674
```

```
olive.3.1$centers
```

```
##   palmitic palmitoleic stearic    oleic linoleic linolenic arachidic eicosenoic
## 1 13331267    964944.1 2916506 96447767  8606289  347553.8  578952.4   124291.7
## 2 17095962   2178776.0 2676536 85066192 14788961  420661.7  772028.3   276830.8
## 3 14821179   1405786.9 2909596 90336416 12358016  406647.0  786322.8   186986.5
```

```
table(olive.3$cluster)
```

```
## 
##   1   2   3 
## 176 212 184
```

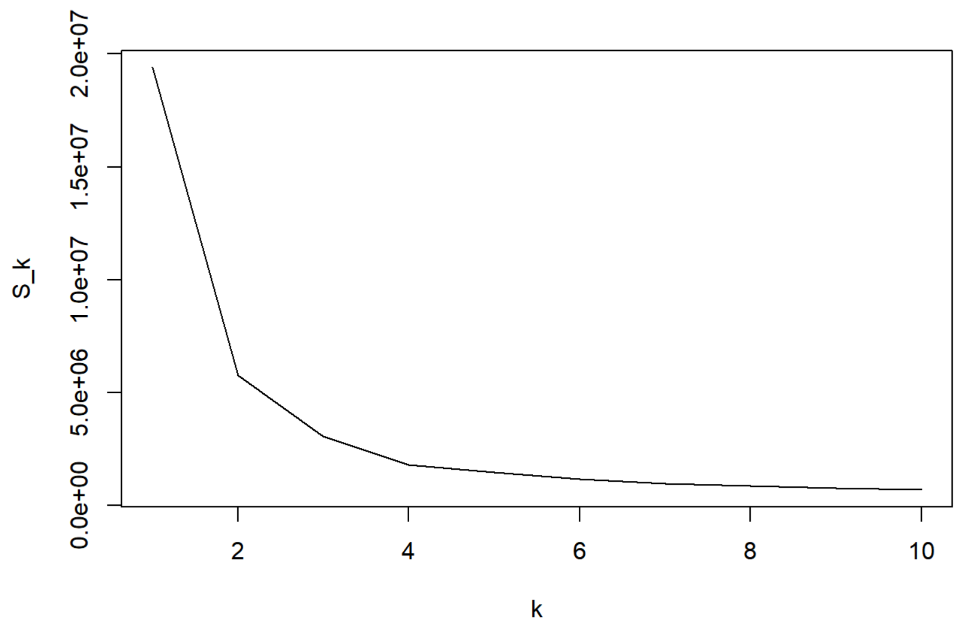
```
table(olive.3.1$cluster)
```

```
## 
##   1   2   3 
## 176 212 184
```

```
#by multiplying the dataset matrix by a constant we obtain the same clustering of the original one (we can see it from by th
e table statement),
#but different center.
#This holds both for the scaled and for the unscaled dataset.
#In fact in both datasets, the centers differs for the constant q for which I multiplied the original dataset.
```

(3) Visualizza i dati, crea un raggruppamento di questo set di dati che ti sembra ragionevole e spiega i motivi per cui hai scelto questo e ritieni che sia ragionevole. Puoi utilizzare altri metodi di clustering che conosci diversi da K-mean, ma se usi solo K-mean, va bene anche questo. Nota che alcune delle variabili sono molto discrete, il che potrebbe creare problemi con il clustering, quindi potresti decidere di lasciare fuori una o più variabili.

```
Boston <- read.csv("C:/Users/Utente/OneDrive/Desktop/LM/bigData/Boston.dat", sep="")
boston<-data.matrix(Boston)
boston1<-boston[,-c(4)]

#find the good k
sk<-numeric(0)
kclusterings<-list()
for (k in 1:10) {
  kclusterings[[k]]<-kmeans(boston1,k,nstart=100)
  sk[k]<-kclusterings[[k]]$tot.withinss
}
plot(1:10,sk,xlab="k", ylab="S_k",type="l")
```



```
kclusterings[[2]]$tot.withinss
```

```
## [1] 5764962
```

```
#5764962
kclusterings[[3]]$tot.withinss
```

```
## [1] 3068467
```

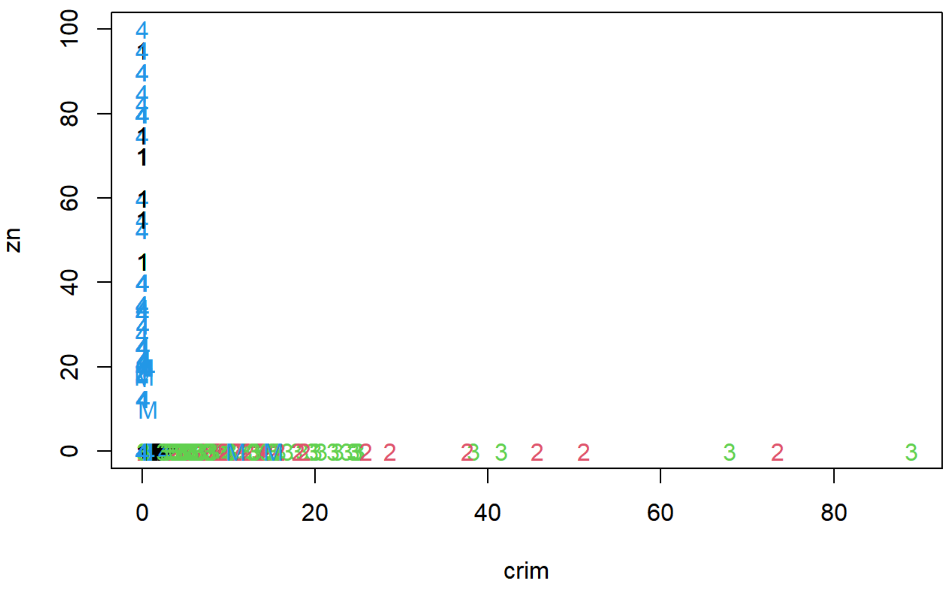
```
#3068467
kclusterings[[4]]$tot.withinss
```

```
## [1] 1814405
```

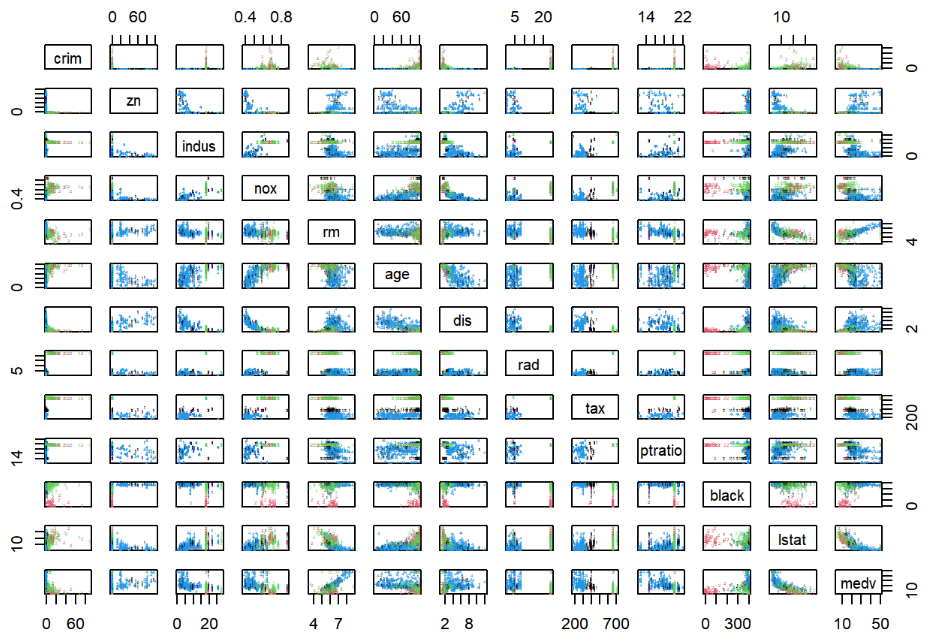
```
#1814405
kclusterings[[5]]$tot.withinss
```

```
## [1] 1475517
```

```
#1475517

#The last great difference of tot.withinss is between k=3 and k=4 while the improvement between k=4 and k=5 is negligible .
#So the best solution is to take the number of centres k=4.
kboston.4<-kmeans(boston1,centers = 4, nstart=100)
plot(boston1, col=kboston.4$cluster, pch=clusym[kboston.4$cluster])
points(kboston.4$centers,pch="M", cex=1, col=4)
```



```
pairs(boston1, cex=0.3, col=kboston.4$cluster, pch=clusym[kboston.4$cluster])
```



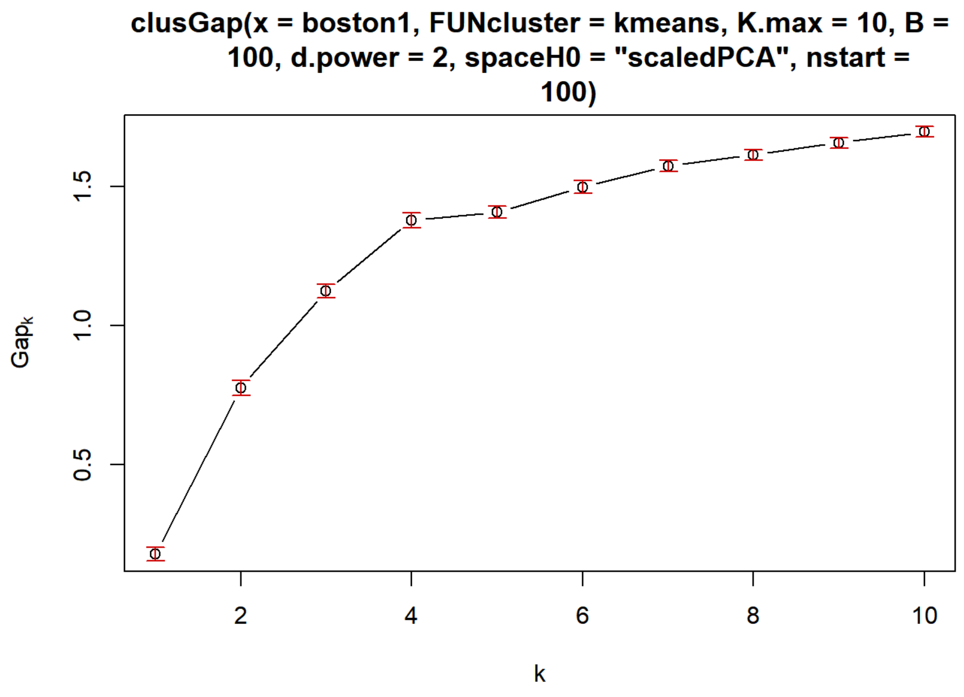
```
#Elbow method
#elbow<-fviz_nbclust(boston1, kmeans, method = "wss") +
  #geom_vline(xintercept = 4, linetype = 2) +
  #labs(subtitle = "Elbow method")

#with the plot resulting from the application of the elbow method shows that the optimal number of cluster is 4.

#Gap Method
#install.packages("cluster")
library(cluster)
set.seed(12345)
gap<- clusGap(boston1,kmeans,K.max=10,B=100,d.power=2,spaceH0="scaledPCA",nstart=100)
```

```
## Warning: non converge in 10 iterazioni
```

```
print(gap,method="globalSEmax",SE.factor=2)
```

```
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = boston1, FUNcluster = kmeans, K.max = 10, B = 100, d.power = 2, spaceH0 = "scaledPCA", nstart = 100)
## B=100 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
##  --> Number of clusters (method 'globalSEmax', SE.factor=2): 10
##           logW   E.logW       gap     SE.sim
##  [1,] 16.08769 16.26729 0.1795963 0.02489975
##  [2,] 14.87416 15.65081 0.7766494 0.02715961
##  [3,] 14.24354 15.36786 1.1243164 0.02451128
##  [4,] 13.71812 15.09520 1.3770771 0.02676528
##  [5,] 13.51137 14.91745 1.4060826 0.02157747
##  [6,] 13.27647 14.77326 1.4967948 0.02226517
##  [7,] 13.11278 14.68565 1.5728643 0.02074077
##  [8,] 12.99367 14.60579 1.6121199 0.01919923
##  [9,] 12.87852 14.53455 1.6560314 0.01901579
## [10,] 12.77679 14.47129 1.6945038 0.01874866
```

```
plot(gap)
```



```
#with the gap method results that the optimal number of clusters is 4 as is shown by applying the elbow method.
#the improvement that we can see after k=4 is negligible.

###
#we can cluster the units basing the clustering method on the variables that result form the application of principal compon
ents.
#In this way we retain only the principal components that explain the biggest part of the variance of the dataset.
prolive<-princomp(boston1)
summary(prolive)
```

```
## Importance of components:
##                              Comp.1     Comp.2      Comp.3     Comp.4
## Standard deviation     175.6386229 78.9839276 28.65026817 16.3330343
## Proportion of Variance   0.8045735  0.1627058  0.02140834  0.0069576
## Cumulative Proportion    0.8045735  0.9672794  0.98868773  0.9956453
##                              Comp.5      Comp.6       Comp.7       Comp.8
## Standard deviation     8.768608846 6.825771401 4.1265499050 3.6784846027
## Proportion of Variance 0.002005336 0.001215148 0.0004441196 0.0003529097
## Cumulative Proportion  0.997650662 0.998865810 0.9993099299 0.9996628396
##                               Comp.9      Comp.10      Comp.11      Comp.12
## Standard deviation     2.9818846230 1.6478926312 1.048723e+00 4.663176e-01
## Proportion of Variance 0.0002319035 0.0000708245 2.868452e-05 5.671388e-06
## Cumulative Proportion  0.9998947432 0.9999655677 9.999943e-01 9.999999e-01
##                             Comp.13
## Standard deviation     5.413879e-02
## Proportion of Variance 7.644390e-08
## Cumulative Proportion  1.000000e+00
```

(4)++” is the name of a method to initialise the k-means algorithm that has been proposed in the literature (for really big datasets it may be problematic to run Lloyd’s or similar algorithms a lot of times from random starting points, and having just one well picked starting point will be much faster and hopefully not much worse or even better). Do some research on the internet, find out and explain how this works. It can be run by the following function kmpp, where X is a data matrix and k is the number of clusters. The output is of the same format as kmeans. Run this on one or more of the example data sets and run kmeans as well, both with nstart=1 and nstart=100, and compare the achieved values of the objectivefunction (tot.withinss-component of the output).

Both the Kmeans and the Kmeans++ start by allocating cluster centers randomly. The k-means++ is an algorithm for choosing the initial values for the k-means algorithm: the first looks for the better solution after this initial step, while the ++ searches for other centers given the first one. Kmenas++ speeds up the convergence, and theorically it provides better results. The k-means aim is to find cluster centers that minimize the sum of squared distances from each data point in a cluster to its cluster center. It is used widely and often finds good solutions quickly but it is sensitive to the initialization of the centroids so if we start with very bad centroids, we can quickly arrive to a poor solution. K-means ++ overcomes this obstacles, due to its ability to do a smart initialization of the centroids. It specifies an initialization procedure for the cluster centres before applying the kmeans optimizaion algorithm.

The steps involved: 1.select at random the first centroid from the data points. 2.compute for each data point its distance from the nearest centroid chosen previously. 3.select the next centroid from the data points such that its probability to be chosen as centroid is directly proportional to its distance from the nearest centroid chosen before. 4.Repeat steps 2 and 3 until k centroids have been sampled 5.apply the kmeans algorithm.

```r
#install.packages("pracma")
library(pracma)

Boston <- read.csv("C:/Users/Utente/OneDrive/Desktop/LM/bigData/Boston.dat", sep="")
boston1<-data.matrix(Boston)

set.seed(665544)
kboston.100<-kmeans(boston1,centers = 3, nstart=100)
#plot(boston1, col=kboston.100$cluster, pch=clusym[kboston.100$cluster])
#points(kboston.100$centers,pch="M", cex=1, col=4)
#pairs(boston1, cex=0.3, col=kboston.100$cluster, pch=clusym[kboston.100$cluster])
str(kboston.100)
```

```
## List of 9
##  $ cluster     : int [1:506] 2 2 2 2 2 2 2 2 2 2 ...
##  $ centers     : num [1:3, 1:14] 10.911 0.375 15.219 0 15.71 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:14] "crim" "zn" "indus" "chas" ...
##  $ totss       : num 19401064
##  $ withinss    : num [1:3] 181892 2573399 313209
##  $ tot.withinss: num 3068499
##  $ betweenss   : num 16332565
##  $ size        : int [1:3] 102 366 38
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 3068499

set.seed(665544)
kboston.1<-kmeans(boston1,centers = 3, nstart=1)
#plot(boston1, col=kboston.1$cluster, pch=clusym[kboston.1$cluster])
#points(kboston.1$centers,pch="M", cex=1, col=4)
#pairs(boston1, cex=0.3, col=kboston.1$cluster, pch=clusym[kboston.1$cluster])
str(kboston.1)
```

```
## List of 9
##  $ cluster     : int [1:506] 2 2 2 2 2 2 2 2 2 2 ...
##  $ centers     : num [1:3, 1:14] 10.911 0.375 15.219 0 15.71 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:14] "crim" "zn" "indus" "chas" ...
##  $ totss       : num 19401064
##  $ withinss    : num [1:3] 181892 2573399 313209
##  $ tot.withinss: num 3068499
##  $ betweenss   : num 16332565
##  $ size        : int [1:3] 102 366 38
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 4460944

kmpp <- function(X, k) {
  n <- nrow(X)
  C <- numeric(k)
  C[1] <- sample(1:n, 1)
  for (i in 2:k) {
    dm <- distmat(X, X[C, ])
    pr <- apply(dm, 1, min); pr[C] <- 0
    C[i] <- sample(1:n, 1, prob = pr)
  }
  kmeans(X, X[C, ])
}

set.seed(665544)
kboston.pp<-kmpp(boston1, 3)
str(kboston.pp)
```

```
## List of 9
##  $ cluster     : int [1:506] 3 3 3 3 3 3 3 3 3 3 ...
##  $ centers     : num [1:3, 1:14] 12.299 0.781 0.241 0 9.653 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:14] "crim" "zn" "indus" "chas" ...
##  $ totss       : num 19401064
##  $ withinss    : num [1:3] 2896224 640601 924119
##  $ tot.withinss: num 4460944
##  $ betweenss   : num 14940120
##  $ size        : int [1:3] 137 101 268
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 4460944

#in terms of tot.withinss the best solution is the kmeans with the number of random initialization for starting the algorith
m equal to 100
#Known the kmeans iterative nature and that the initialization of centroids at the start of the algorithm is random, differe
nt initializations may lead to different clusters since kmeans algorithm tends to find only a local optimum, not the global
 as the kmean++ tends to do.
kboston.1$centers
```

```
##         crim       zn     indus       chas       nox       rm      age      dis
## 1 10.9105113  0.00000 18.572549 0.07843137 0.6712255 5.982265 89.91373 2.077164
## 2  0.3749927 15.71038  8.359536 0.07103825 0.5098626 6.391653 60.41339 4.460745
## 3 15.2190382  0.00000 17.926842 0.02631579 0.6737105 6.065500 89.90526 1.994429
##        rad      tax  ptratio     black    lstat     medv
## 1 23.01961 668.2059 20.19510 371.80304 17.87402 17.42941
## 2  4.45082 311.2322 17.81776 383.48981 10.38866 24.93169
## 3 22.50000 644.7368 19.92895  57.78632 20.44868 13.12632
```

```
kboston.100$centers
```

```
##         crim       zn     indus       chas       nox       rm      age      dis
## 1 10.9105113  0.00000 18.572549 0.07843137 0.6712255 5.982265 89.91373 2.077164
## 2  0.3749927 15.71038  8.359536 0.07103825 0.5098626 6.391653 60.41339 4.460745
## 3 15.2190382  0.00000 17.926842 0.02631579 0.6737105 6.065500 89.90526 1.994429
##        rad      tax  ptratio     black    lstat     medv
## 1 23.01961 668.2059 20.19510 371.80304 17.87402 17.42941
## 2  4.45082 311.2322 17.81776 383.48981 10.38866 24.93169
## 3 22.50000 644.7368 19.92895  57.78632 20.44868 13.12632
```

```
kboston.pp$centers
```

```
##         crim        zn     indus       chas       nox       rm      age
## 1 12.2991617  0.000000 18.451825 0.05839416 0.6701022 6.006212 89.96788
## 2  0.7807617  9.653465 13.070594 0.06930693 0.5873366 6.182515 73.70594
## 3  0.2410479 17.817164  6.668582 0.07462687 0.4833981 6.465448 55.70522
##        dis       rad      tax  ptratio    black     lstat     medv
## 1 2.054470 23.270073 667.6423 20.19635 291.0391 18.674526 16.27226
## 2 3.294209  4.831683 405.8020 17.63960 363.0747 12.750990 22.18218
## 3 4.873560  4.313433 276.5485 17.87313 387.8141  9.538022 25.86530
```

```
olive.mat<-data.matrix(olive)

set.seed(665544)
olive.100<-kmeans(olive.mat,centers = 3, nstart=100)
#plot(olive.mat, col=olive.100$cluster, pch=clusym[olive.100$cluster])
#points(olive.100$centers,pch="M", cex=1, col=4)
str(olive.100)
```

```
## List of 9
##  $ cluster     : int [1:572] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:8] 1079.9 1384.8 1200.6 78.2 176.5 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 1.47e+08
##  $ withinss    : num [1:3] 7631152 12636160 10226254
##  $ tot.withinss: num 30493566
##  $ betweenss   : num 1.16e+08
##  $ size        : int [1:3] 176 212 184
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 9336608

set.seed(665544)
olive.1<-kmeans(olive.mat,centers = 3, nstart=1)
#plot(olive.mat, col=olive.1$cluster, pch=clusym[olive.1$cluster])
#points(olive.1$centers,pch="M", cex=1, col=4)
str(olive.1)
```

```
## List of 9
##  $ cluster     : int [1:572] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:8] 1079.9 1384.8 1200.6 78.2 176.5 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 1.47e+08
##  $ withinss    : num [1:3] 7631152 12636160 10226254
##  $ tot.withinss: num 30493566
##  $ betweenss   : num 1.16e+08
##  $ size        : int [1:3] 176 212 184
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
#tot.withinss: num 9339261

set.seed(665544)
olive.pp<-kmpp(olive.mat, 3)
str(olive.pp)
```

```
## List of 9
##  $ cluster     : int [1:572] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:8] 1079.9 1200.6 1384.8 78.2 113.9 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:8] "palmitic" "palmitoleic" "stearic" "oleic" ...
##  $ totss       : num 1.47e+08
##  $ withinss    : num [1:3] 7631152 10226254 12636160
##  $ tot.withinss: num 30493566
##  $ betweenss   : num 1.16e+08
##  $ size        : int [1:3] 176 184 212
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
olive.pp$tot.withinss
```

```
## [1] 30493566
```

```
#tot.withinss: 9637187

#in terms of tot.withiniss both the kmeans are better than the kmeans++, and with nstart=1 the clustering is the best.
olive.1$centers
```

```
##   palmitic palmitoleic  stearic    oleic  linoleic linolenic arachidic
## 1 1079.892    78.16477 236.2500 7812.699  697.1477  28.15341  46.89773
## 2 1384.849   176.49057 216.8113 6890.741 1197.9717  34.07547  62.53774
## 3 1200.582   113.87500 235.6902 7317.652 1001.0543  32.94022  63.69565
##   eicosenoic
## 1   10.06818
## 2   22.42453
## 3   15.14674
```

```
olive.100$centers
```

```
##   palmitic palmitoleic  stearic    oleic  linoleic linolenic arachidic
## 1 1079.892    78.16477 236.2500 7812.699  697.1477  28.15341  46.89773
## 2 1384.849   176.49057 216.8113 6890.741 1197.9717  34.07547  62.53774
## 3 1200.582   113.87500 235.6902 7317.652 1001.0543  32.94022  63.69565
##   eicosenoic
## 1   10.06818
## 2   22.42453
## 3   15.14674
```

```
olive.pp$centers
```

```
##   palmitic palmitoleic  stearic    oleic  linoleic linolenic arachidic
## 1 1079.892    78.16477 236.2500 7812.699  697.1477  28.15341  46.89773
## 2 1200.582   113.87500 235.6902 7317.652 1001.0543  32.94022  63.69565
## 3 1384.849   176.49057 216.8113 6890.741 1197.9717  34.07547  62.53774
##   eicosenoic
## 1   10.06818
## 2   15.14674
## 3   22.42453
```

```
olive.1$cluster
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3
##  [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 1 3 1 3 3 3 3 3 3 3 2 3 3 3 3
##  [75] 3 3 3 3 3 3 3 2 3 2 3 3 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [112] 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2
## [149] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2
## [186] 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 2 2
## [223] 2 2 2 3 3 2 2 3 2 2 3 2 2 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2
## [260] 2 2 3 1 1 1 1 1 1 3 3 1 1 3 2 3 2 2 2 2 2 3 3 3 1 1 3 1 1 1 1 3 3 3 2 3 3
## [297] 2 3 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3
## [334] 3 2 2 2 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [371] 2 3 2 2 3 2 3 3 2 2 3 3 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [408] 3 3 3 2 3 2 2 2 3 2 3 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [445] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3
## [482] 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [519] 1 1 1 1 3 3 3 3 3 1 1 1 3 1 3 1 1 1 3 1 1 3 1 1 1 1 3 1 3 1 1 1 1 3 1 1 1
## [556] 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1
```

```
olive.100$cluster
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3
##  [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 1 3 1 3 3 3 3 3 3 3 2 3 3 3 3
##  [75] 3 3 3 3 3 3 3 2 3 2 3 3 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [112] 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2
## [149] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2
## [186] 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 2 2
## [223] 2 2 2 3 3 2 2 3 2 2 3 2 2 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2
## [260] 2 2 3 1 1 1 1 1 1 3 3 1 1 3 2 3 2 2 2 2 2 3 3 3 1 1 3 1 1 1 1 3 3 3 2 3 3
## [297] 2 3 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3
## [334] 3 2 2 2 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [371] 2 3 2 2 3 2 3 3 2 2 3 3 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [408] 3 3 3 2 3 2 2 2 3 2 3 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [445] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3
## [482] 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [519] 1 1 1 1 3 3 3 3 3 1 1 1 3 1 3 1 1 1 3 1 1 3 1 1 1 1 3 1 3 1 1 1 1 3 1 1 1
## [556] 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1
```

```
olive.pp$cluster
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 2 1 2 2 2 2 2 2 2 3 2 2 2 2
##  [75] 2 2 2 2 2 2 2 3 2 3 2 2 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [112] 3 3 3 3 3 3 3 2 3 2 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3
## [149] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3
## [186] 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 2 3 3 3 3 3
## [223] 3 3 3 2 2 3 3 2 3 3 2 3 3 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3
## [260] 3 3 2 1 1 1 1 1 1 2 2 1 1 2 3 2 3 3 3 3 3 2 2 2 1 1 2 1 1 1 1 2 2 2 3 2 2
## [297] 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2
## [334] 2 3 3 3 2 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [371] 3 2 3 3 2 3 2 2 3 3 2 2 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [408] 2 2 2 3 2 3 3 3 2 3 2 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [445] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2
## [482] 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [519] 1 1 1 1 2 2 2 2 2 1 1 1 2 1 2 1 1 1 2 1 1 2 1 1 1 1 2 1 2 1 1 1 1 2 1 1 1
## [556] 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1
```