UCA / Polytech 4A : Machine learning II. TD1-correction

Rémi Boutin

```
set.seed(150)
```

Exercice 1 : Algorithme de classification ascendante hiérarchique (CAH)

Question 1. Décrire le fonctionnement de l'algorithme CAH

Initialisation : chaque observation est seule dans son propre cluster ($K = n$) (il y a autant de clusters que d'observations) ($W = 0$).

1) les distances (selon un critère à définir) entre les clusters sont toutes calculées
2) les deux clusters les plus proches (ayant la plus petite des distances) sont fusionnés (attention : une seule fusion) ($K = K - 1$)
3) retour en 1 tant que $K > 1$

A la toute fin de l'algorithme, $K = 1$ (il n'y a plus qu'un seul cluster).

Question 2. Dans le reste de cet exercice, on utilisera le jeu de données `USArrest`. Pour obtenir des informations sur ce jeu de données, comme sur tout autre objet en R, utiliser la fonction help : `help(USArrests)`

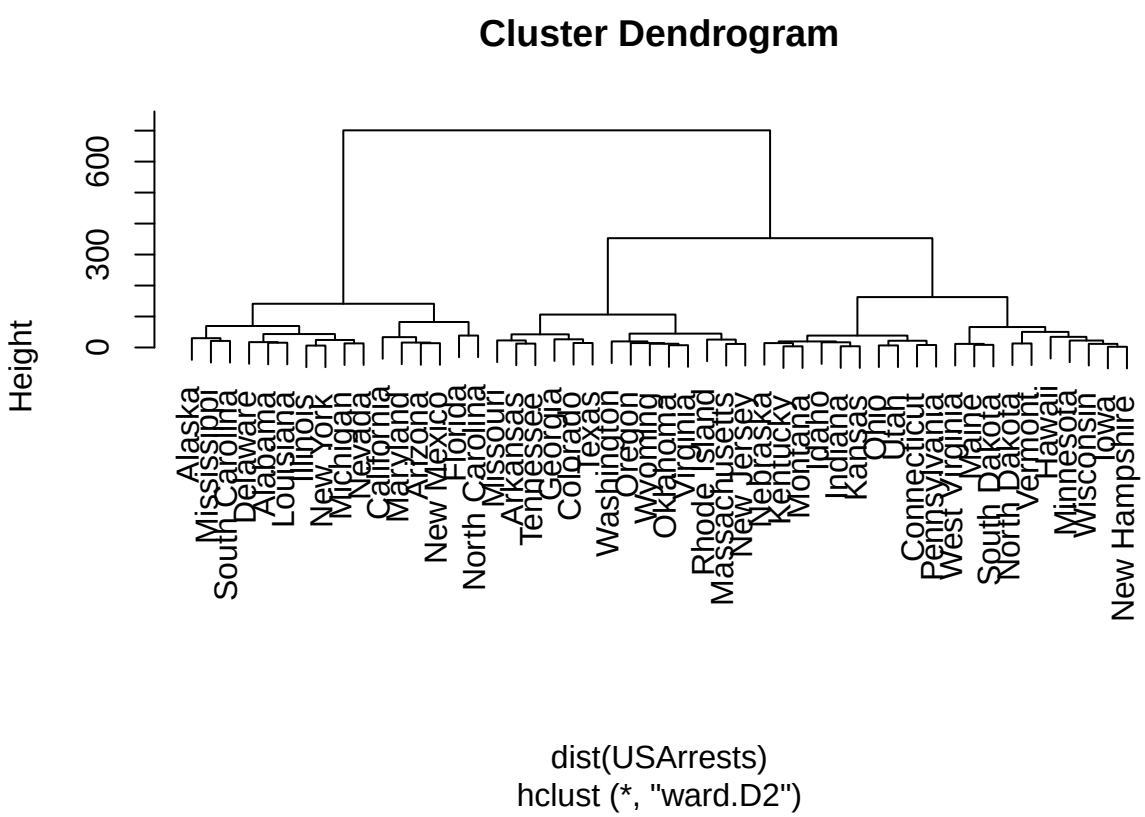
1. Charger le jeu de données en utilisant la fonction `data()`

```
data("USArrests")
str(USArrests)
```

```
## 'data.frame':    50 obs. of  4 variables:
##  $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
##  $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
##  $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
##  $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
```

2. A l'aide du package `hclust`, trouver une partition en utilisant l'algorithme CAH, avec la fonction de lien de Ward. *Attention* : bien lire le manuel !
3. Représenter les résultats

```
resCAH = hclust(dist(USArrests), method="ward.D2") # CAH avec Ward
plot(resCAH)
```



4. Donner une partition à 3 clusters et commenter les résultats. On pourra par exemple calculer des statistiques sur ces clusters et les comparer entre eux.

```
K = 3
cl_cah = cutree(resCAH, K)
cl_cah
```

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

```r
table(cl_cah) # 1) parler de la taille des clusters
```

```
## cl_cah
##  1  2  3
## 16 14 20
```

```r
mu = matrix(0, K, dim(USArrests)[2])

for(k in 1:K) {
   mu[k, ] = colMeans(USArrests[cl_cah==k,])
}
colnames(mu) = colnames(USArrests)
print(mu)
```

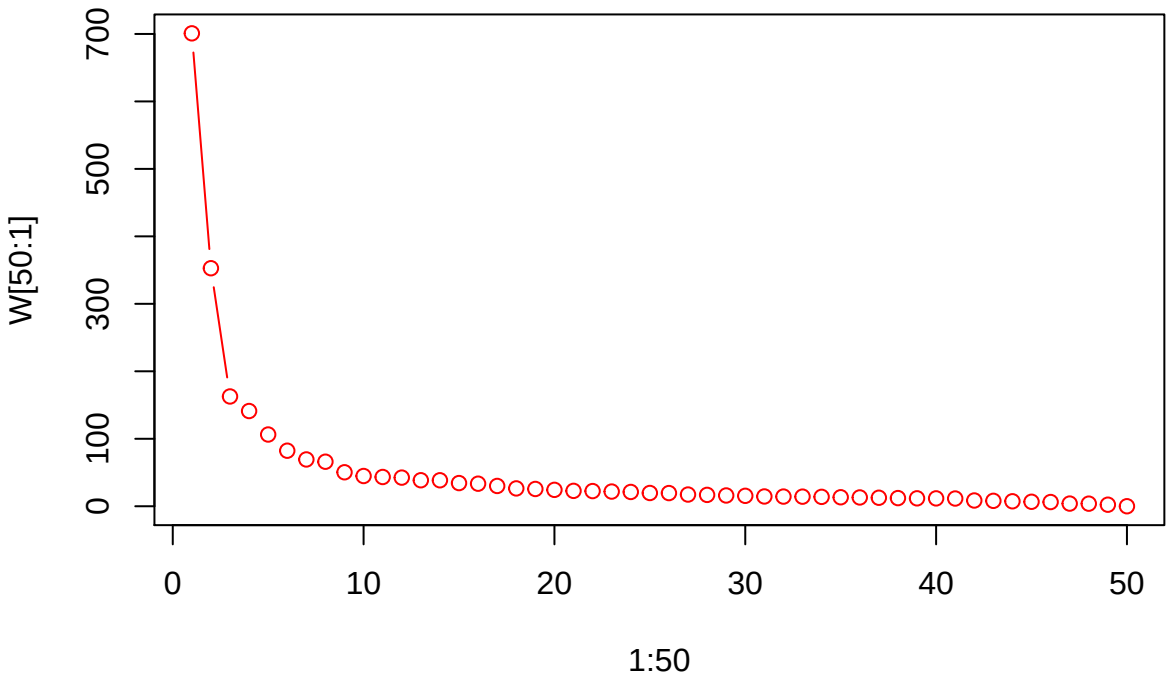
```
##         Murder  Assault UrbanPop     Rape
## [1,] 11.812500 272.5625 68.31250 28.37500
## [2,]  8.214286 173.2857 70.64286 22.84286
## [3,]  4.270000  87.5500 59.75000 14.39000
```

Question 3. Toujours avec le jeu de données `USArrest`, estimer le nombre de clusters.

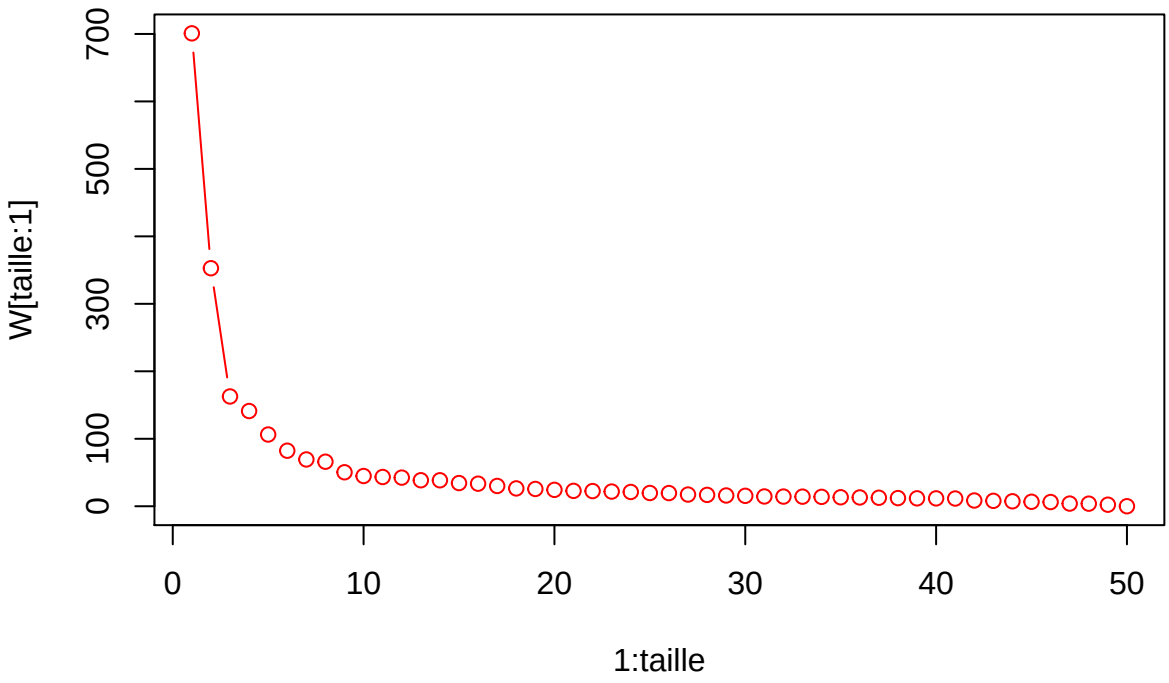
```r
resCAH$height
```

```
##  [1]   2.291288   3.834058   3.929377   6.236986   6.637771   7.355270
##  [7]   8.027453   8.537564  11.456439  11.651180  11.711106  12.089941
## [13]  12.614278  13.044922  13.297368  13.896043  14.230249  14.325967
## [19]  14.501034  15.454449  15.946368  16.826170  17.344355  19.507605
## [25]  19.622130  21.167192  21.768096  22.487219  22.810816  24.298560
## [31]  25.607616  26.415904  30.058887  33.432145  34.345378  38.493679
## [37]  38.527912  42.454525  43.362975  44.808392  50.335120  65.997831
## [43]  69.349701  82.294441 106.292266 141.133742 162.699945 352.783642
## [49] 700.878602
```

```r
W = c(0, resCAH$height)

plot(1:50, W[50:1], type="b", col="red")
```




```
# sous la forme d'une fonction
plotCAH <- function(res) {

   W = c(0, resCAH$height)
   taille = length(W)
   plot(1:taille, W[taille:1], type="b", col="red")
}

plotCAH(resCAH)
```



Exercice 2 : L'exemple des données decathlon

Cet exercice est simimaire au précédent avec moins d'étapes intérmédiaires. Il vous revient de mener la démarche de clustering adapté en gardant un esprit critique.

1. Charger les données `Decathlon` contenu dans le package `FactoMineR` et décrire ce jeu de données.

```
data("decathlon")
```

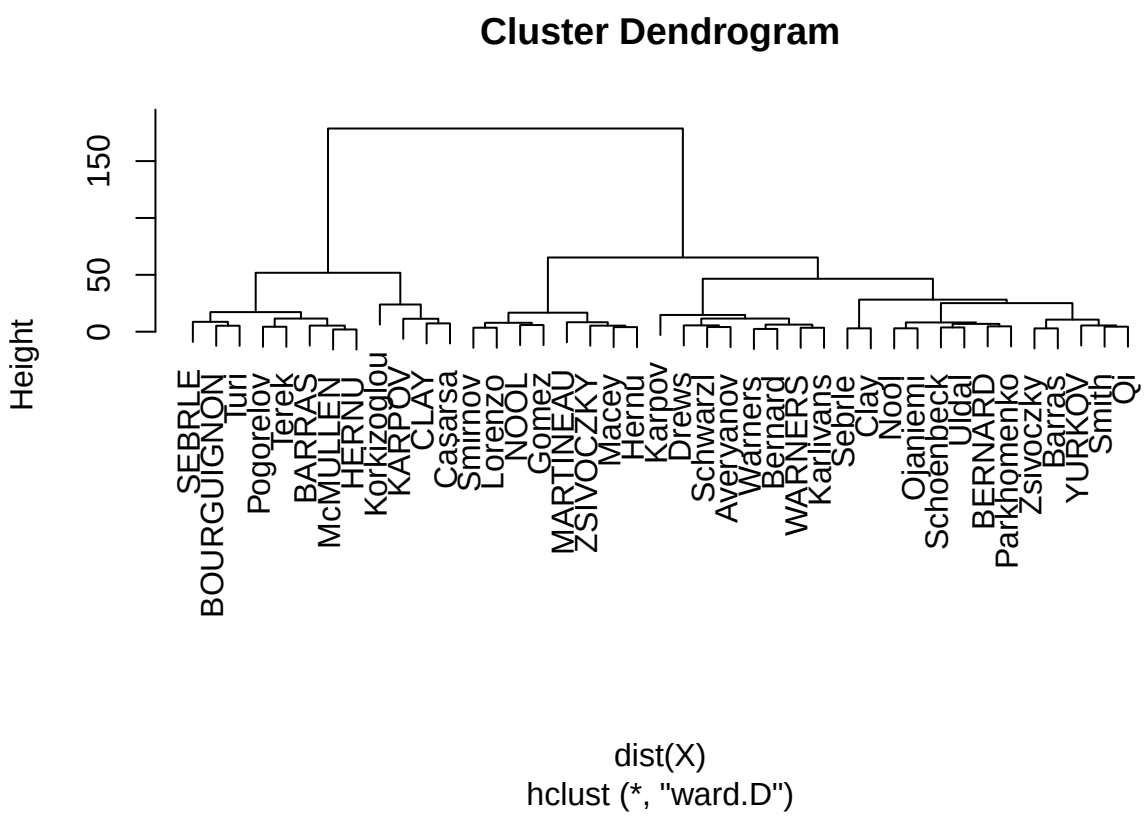
2. Partitionner les données à l'aide d'un algorithme CAH.

```
library(FactoMineR)

data("decathlon")

X = decathlon[, 1:10]
str(X)
```

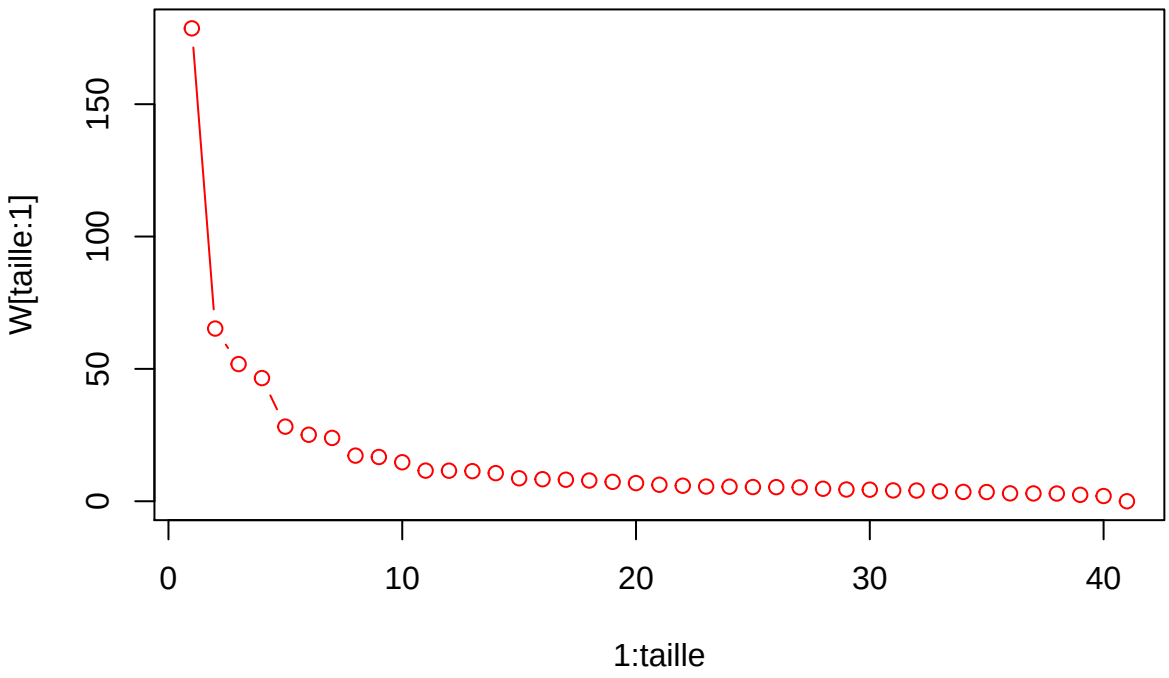
```
## 'data.frame':    41 obs. of  10 variables:
##  $ 100m       : num  11 10.8 11 11 11.3 ...
##  $ Long.jump  : num  7.58 7.4 7.3 7.23 7.09 7.6 7.3 7.31 6.81 7.56 ...
##  $ Shot.put   : num  14.8 14.3 14.8 14.2 15.2 ...
##  $ High.jump  : num  2.07 1.86 2.04 1.92 2.1 1.98 2.01 2.13 1.95 1.86 ...
##  $ 400m       : num  49.8 49.4 48.4 48.9 50.4 ...
##  $ 110m.hurdle: num  14.7 14.1 14.1 15 15.3 ...
##  $ Discus     : num  43.8 50.7 49 40.9 46.3 ...
##  $ Pole.vault : num  5.02 4.92 4.92 5.32 4.72 4.92 4.42 4.42 4.92 4.82 ...
##  $ Javeline   : num  63.2 60.1 50.3 62.8 63.4 ...
```

```
##  $ 1500m      : num  292 302 300 280 276 ...
```

```
resCAH = hclust(dist(X), method="ward.D")

plot(resCAH)
```




```
plotCAH(resCAH)
```



3. Décrire les résultats.

Nous estimons ici, grâce à la CAH, qu'il y a 5 clusters dans les données. Commenter les moyennes des groupes, les comparer, la taille des clusters et les écarts-types au sein des groupes.

```
K = 5
clCAH = cutree(resCAH, K)
clCAH
```

```
##      SEBRLE        CLAY      KARPOV     BERNARD      YURKOV     WARNERS
##           1           2           2           3           3           4
##   ZSIVOCZKY    McMULLEN   MARTINEAU       HERNU      BARRAS        NOOL
##           5           1           5           1           1           5
## BOURGUIGNON      Sebrle        Clay      Karpov       Macey     Warners
##           1           3           3           4           5           4
##   Zsivoczky       Hernu        Nool     Bernard    Schwarzl   Pogorelov
##           3           5           3           4           4           1
##  Schoenbeck      Barras       Smith   Averyanov    Ojaniemi     Smirnov
##           3           3           3           4           3           5
##          Qi       Drews Parkhomenko       Terek       Gomez        Turi
##           3           4           3           1           5           1
##     Lorenzo   Karlivans Korkizoglou       Uldal     Casarsa
##           5           4           2           3           2
```

```
table(clCAH)
```

```
## clCAH
##  1  2  3  4  5
```

```
##  8  4 13  8  8
```

Exemple pour la comparaison des moyennes :

```
mu = matrix(0, K, dim(X)[2])

for(k in 1:K) {
   mu[k, ] = colMeans(X[clCAH==k,])
}
colnames(mu) = colnames(X)
print(mu)
```

```
##          100m Long.jump Shot.put High.jump     400m 110m.hurdle   Discus
## [1,] 11.11000  7.172500 14.30250  1.987500 50.43500    14.73375 43.22375
## [2,] 11.00000  7.112500 14.69000  1.945000 50.52500    14.62250 48.60000
## [3,] 10.95077  7.256154 14.86615  1.978462 49.60385    14.67000 44.79077
## [4,] 10.83125  7.512500 14.29250  1.986250 48.89000    14.25125 43.37375
## [5,] 11.12875  7.175000 14.09750  1.970000 49.09000    14.72000 43.48625
##      Pole.vault Javeline    1500m
## [1,]   4.887500 56.28000 287.9500
## [2,]   4.735000 55.53250 303.7050
## [3,]   4.787692 63.38846 276.2315
## [4,]   4.777500 54.15625 276.0038
## [5,]   4.595000 57.66375 265.3200
```

```
#write.csv(mu, file="centers.csv", row.names = FALSE)
```

Exercice 3 : Algorithme kmeans

1. Présenter l'algorithme KMeans

Initialisation : les centres μ_k sont tirés au hasard

1) chaque observation est rangée dans le cluster dont le centre est le plus proche
2) les centres μ_k sont recalculés
3) retour en 1 tant que les centres bougent

Condition d'arrêt : nous sortons de la boucle lorsque les centres de clusters ne bougent plus

2. Charger le jeu de données 'USArrest"

```
data("USArrests")
str(USArrests)
```

```
## 'data.frame':    50 obs. of  4 variables:
##  $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
##  $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
##  $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
##  $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
```

2. Partitionner les données à l'aide de l'agorithme KMeans pour $K = 3$.

```
K = 3
res = kmeans(USArrests, K, nstart = 50)

res$cluster
```

```
##        Alabama         Alaska        Arizona       Arkansas     California
##              2              2              2              1              2
```

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

```
table(res$cluster) # 1) parler de la taille des clusters
```

```
##
##  1  2  3
## 14 16 20
```

```
res$centers # 2) interpréter les centres des clusters
```

```
##      Murder  Assault UrbanPop     Rape
## 1  8.214286 173.2857 70.64286 22.84286
## 2 11.812500 272.5625 68.31250 28.37500
## 3  4.270000  87.5500 59.75000 14.39000
```

3. Interpréter les résultats

Interprétation :

1) les clusters sont de tailles assez proches, le cluster 3 étant quand même plus important

2)

- Dans le cluster 2 : états ayant les niveaux de criminalité les plus élevés
- Dans le cluster 3 : états ayant les niveaux de criminalité les moins élevés
- Dans le cluster 1 : entre les deux clusters. Mais aussi à noter : niveau de population urbaine le plus élevé
- remarque : les états du cluster 3 ont une population urbaine plus élevée et pourtant un niveau de criminalité plus faible que ceux du cluster 1

Propriétés de l'algorithme kmeans

- dépend de l'initialisation => ne pas oublier de répéter l'algorithme plusieurs fois
- algorithme kmeans très rapide
- algorithme facilement parallélisable

4. Refaire la même démarche sur le jeu de données decathlon. Trouve-t-on les même clusters qu'avec la CAH ? Commenter.

```
library(FactoMineR)
data("decathlon")
```

```r
str(decathlon)
```

```
## 'data.frame':    41 obs. of  13 variables:
##  $ 100m       : num  11 10.8 11 11 11.3 ...
##  $ Long.jump  : num  7.58 7.4 7.3 7.23 7.09 7.6 7.3 7.31 6.81 7.56 ...
##  $ Shot.put   : num  14.8 14.3 14.8 14.2 15.2 ...
##  $ High.jump  : num  2.07 1.86 2.04 1.92 2.1 1.98 2.01 2.13 1.95 1.86 ...
##  $ 400m       : num  49.8 49.4 48.4 48.9 50.4 ...
##  $ 110m.hurdle: num  14.7 14.1 14.1 15 15.3 ...
##  $ Discus     : num  43.8 50.7 49 40.9 46.3 ...
##  $ Pole.vault : num  5.02 4.92 4.92 5.32 4.72 4.92 4.42 4.42 4.92 4.82 ...
##  $ Javeline   : num  63.2 60.1 50.3 62.8 63.4 ...
##  $ 1500m      : num  292 302 300 280 276 ...
##  $ Rank       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Points     : int  8217 8122 8099 8067 8036 8030 8004 7995 7802 7733 ...
##  $ Competition: Factor w/ 2 levels "Decastar","OlympicG": 1 1 1 1 1 1 1 1 1 1 ...
```

```r
X = decathlon[, 1:10]

str(X)
```

```
## 'data.frame':    41 obs. of  10 variables:
##  $ 100m       : num  11 10.8 11 11 11.3 ...
##  $ Long.jump  : num  7.58 7.4 7.3 7.23 7.09 7.6 7.3 7.31 6.81 7.56 ...
##  $ Shot.put   : num  14.8 14.3 14.8 14.2 15.2 ...
##  $ High.jump  : num  2.07 1.86 2.04 1.92 2.1 1.98 2.01 2.13 1.95 1.86 ...
##  $ 400m       : num  49.8 49.4 48.4 48.9 50.4 ...
##  $ 110m.hurdle: num  14.7 14.1 14.1 15 15.3 ...
##  $ Discus     : num  43.8 50.7 49 40.9 46.3 ...
##  $ Pole.vault : num  5.02 4.92 4.92 5.32 4.72 4.92 4.42 4.42 4.92 4.82 ...
##  $ Javeline   : num  63.2 60.1 50.3 62.8 63.4 ...
##  $ 1500m      : num  292 302 300 280 276 ...
```

```r
K = 3
res = kmeans(X, K, nstart = 50)

table(res$cluster)
```

```
##
##  1  2  3
## 11 21  9
```

```r
res$centers
```

```
##       100m Long.jump Shot.put High.jump     400m 110m.hurdle   Discus
## 1 11.05727  7.170000 14.31273  1.978182 49.20455    14.67909 43.47455
## 2 10.94952  7.371429 14.53333  1.978571 49.42524    14.52048 44.30667
## 3 11.03889  7.110000 14.54667  1.971111 50.56556    14.71556 45.41000
##   Pole.vault Javeline    1500m
## 1   4.623636 58.52909 266.3827
## 2   4.777143 59.22619 278.2662
## 3   4.897778 55.93444 296.2467
```

Interprétation :

Cluster 1 : 11 sportifs. Cluster 2 : 21 sportifs. Cluster 3 : 9 sportifs.

Dans le cluster 1 : **sportifs forts** au 1500m, au 400m. **Moins bons** au 100m, au poids, au disque, à la

perche. **Moyens** au saut en longueur, au saut en hauteur, au 110m haie, au javelot. En résumé : ce sont les sportifs très forts sur les exercices de course (endurance) : 1500m et le 400m.

Dans le cluster 2 : **sportifs forts** au 100m, au saut en longueur, au saut en hauteur, au 110m haie, au javelot. **Moyens** au 400m, au disque, au saut à la perche, au 1500m. Résumé : ce sont les sportifs très forts sur les exercices courts (intensité)

Dans le cluster 3 : **sportifs forts** au disque, poids, au saut à la perche. **Moins bons** au saut en longueur, au saut en hauteur, au 400m, au 110m haie, au javelot, au 1500m. Résumé : ce sont les sportifs les moins polyvalents. Ils sont uniquement forts sur les exercices de disque, de poids, et de perche.