Notes de cours et codes sur la classification non supervisée (clustering)

Zaineb Smida

Table des matières

Packages nécessaires :

```
install.packages("cluster")
```

1 Rappels de statistique bivariée

On s'intéresse ici au lien entre une variable Y quantitative et une variable qualitative X.

On considère le jeu de données `iris` qui est constituée de 150 observations concernant des fleurs sur lesquelles on observe 4 variables quantitative (exprimées en cm) et une variable qualitative (3 espèces).

```
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
```

```
4          4.6          3.1          1.5          0.2  setosa
5          5.0          3.6          1.4          0.2  setosa
6          5.4          3.9          1.7          0.4  setosa
```

On considère la variable `Petal.Width` comme étant la variable Y quantitative et la variable `Species` comme étant la variable X qualitative.

Objectif :

1. présenter des outils de statistique descriptive
2. décomposer la variance en variance inter-classe et variance intra-classe
3. calculer le rapport de corrélation η^2

1.1 Outils de statistique descriptive

On peut dans un premier temps chercher à calculer la moyenne/variance de Y en fonction des classes de X. Pour cela on va utiliser la fonction `tapply()`. Le premier argument contient la variable quantitative, le deuxième la variable qualitative et le troisième la fonction statistique qu'on va utiliser sur chaque sous-groupe.

```
n_gr <- tapply(X = iris$Petal.Width, INDEX = iris$Species, FUN = length)
moy_gr <- tapply(X = iris$Petal.Width, INDEX = iris$Species, FUN = mean)
var_gr <- tapply(X = iris$Petal.Width, INDEX = iris$Species, FUN = var)
```

On peut ainsi représenter le tableau descriptif suivant :

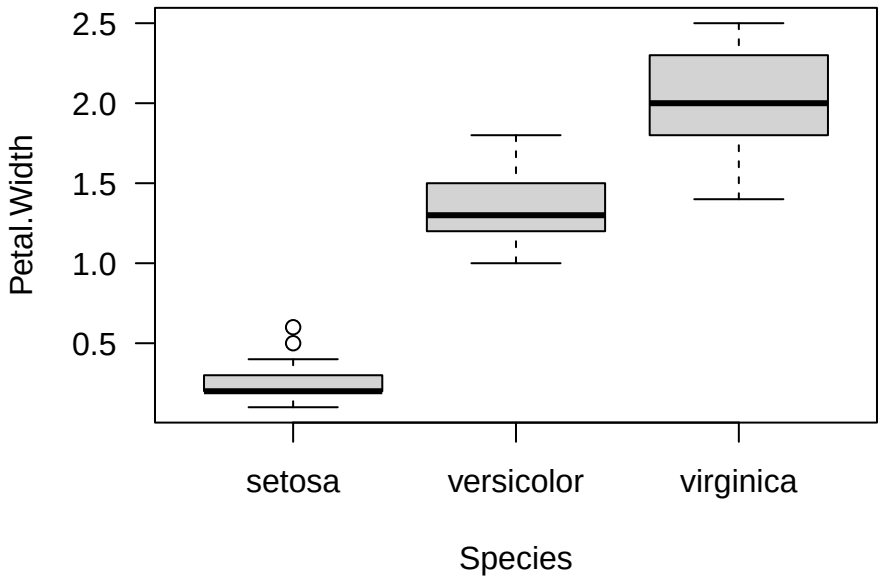
Tableau : SD1

```
kableExtra::kbl(data.frame(effectifs = n_gr, moyenne = moy_gr,
                           variance = var_gr))
```

	effectifs	moyenne	variance
setosa	50	0.246	0.0111061
versicolor	50	1.326	0.0391061
virginica	50	2.026	0.0754327

Un graphique très utile est la boîte à moustache parallèle, qui représente la sous-distribution de la variable Y pour chaque classe. Dans une boîte à moustache, la distribution est résumée par la médiane, les 1er et 3ème quartiles, ainsi que les valeurs "extrêmes", i.e. les valeurs situées au-delà d'un seuil supérieur (resp. inférieur), représenté ici par les moustaches.

```
par(las = 1)
boxplot(Petal.Width ~ Species, data = iris)
```



Commentaires : on observe que la distribution de Y est très différente selon les modalités de X.

1.2 Décomposition de la variance

Calcul de la variance inter-classe

On utilise le calcul précédent de la moyenne des groupes :

Résultat : SD2

```
n <- nrow(iris)
var_inter <- sum(n_gr * (moy_gr - mean(iris$Petal.Width)) ^ 2) / n
var_inter
```

```
[1] 0.5360889
```

Calcul de la variance intra-classe

On utilise le calcul précédent de la variance par groupe :

Résultat : SD3

```
n <- nrow(iris)
var_intra <- sum((n_gr - 1) * var_gr) / n
var_intra
```

```
[1] 0.041044
```

On vérifie que la somme de la variance intra et variance inter est égale à la variance totale :

Résultat : SD4

```
var_inter + var_intra
```

```
[1] 0.5771329
```

```
var(iris$Petal.Width) * (n - 1) / n
```

```
[1] 0.5771329
```

Remarque : par défaut, le logiciel R divise par $(n-1)$ et non n, ce qui explique les ajustements ci-dessus.

1.3 Calcul du rapport de corrélation

Résultat : SD5

```
var_inter / (var_inter + var_intra)
```

```
[1] 0.9288829
```

Interprétation : le rapport de corrélation étant proche de 1, on en déduit que les moyennes conditionnelles de Y sont très différentes selon les groupes définis par les modalités de X

Alternative : pour comparer les moyennes des groupes entre elles, on peut utiliser un test d'analyse de variance (sous hypothèse de normalité des observations) :

```r
anova(lm(Petal.Width ~ Species, data = iris))
```

```
Analysis of Variance Table

Response: Petal.Width
           Df Sum Sq Mean Sq F value    Pr(>F)
Species     2 80.413  40.207  960.01 < 2.2e-16 ***
Residuals 147  6.157   0.042
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Ici, comme la statistique de test est très grande, on peut en conclure que les moyennes ne sont pas toutes égales dans les groupes.

2 Méthode d'Agrégation autour de Moyennes Mobiles (AMM) ou k-means

On considère le jeu de données suivant :

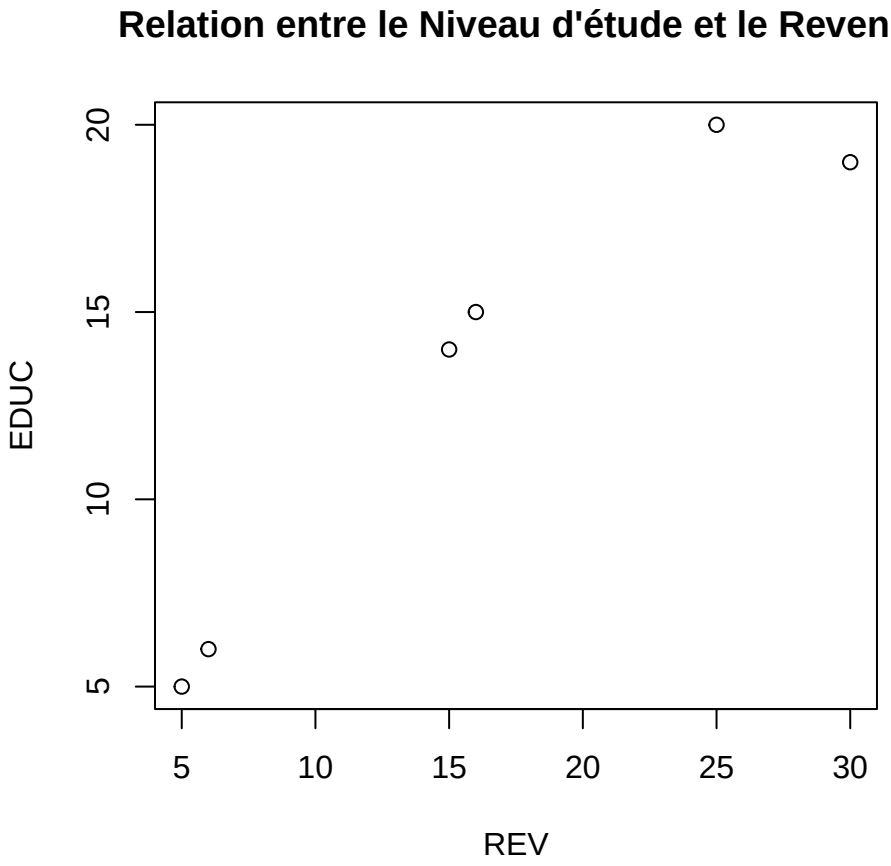
```r
my_df <- data.frame(
  REV = c(5, 6, 15, 16, 25, 30),
  EDUC = c(5, 6, 14, 15, 20, 19)
)
my_df
```

```
  REV EDUC
1   5    5
2   6    6
3  15   14
4  16   15
5  25   20
6  30   19
```

Pour visualiser la relation entre le niveau d'étude et le revenu, on trace le graphique correspondant :

Figure : Données

```r
plot(my_df$REV, my_df$EDUC, main = "Relation entre le Niveau d'étude et le Revenu",
     xlab = "REV", ylab = "EDUC")
```

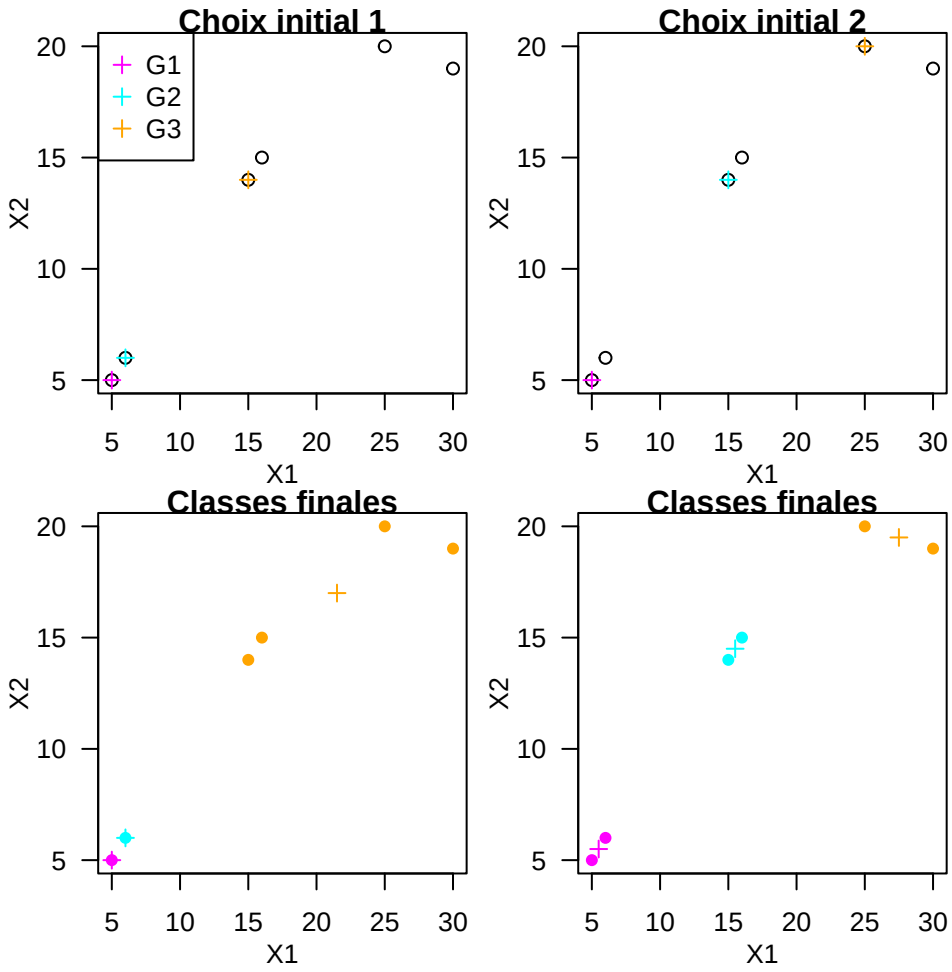


La méthode AMM utilise la fonction `kmeans()`, où le premier argument est le jeu de données et le second le nombre de classes souhaité.

Si la fonction est répétée plusieurs fois, les résultats peuvent varier en raison du choix aléatoire des centres initiaux.

L'illustration suivante montre que des centres différents (figure de gauche vs figure de droite) donnent des classifications distinctes.

Figure CL0



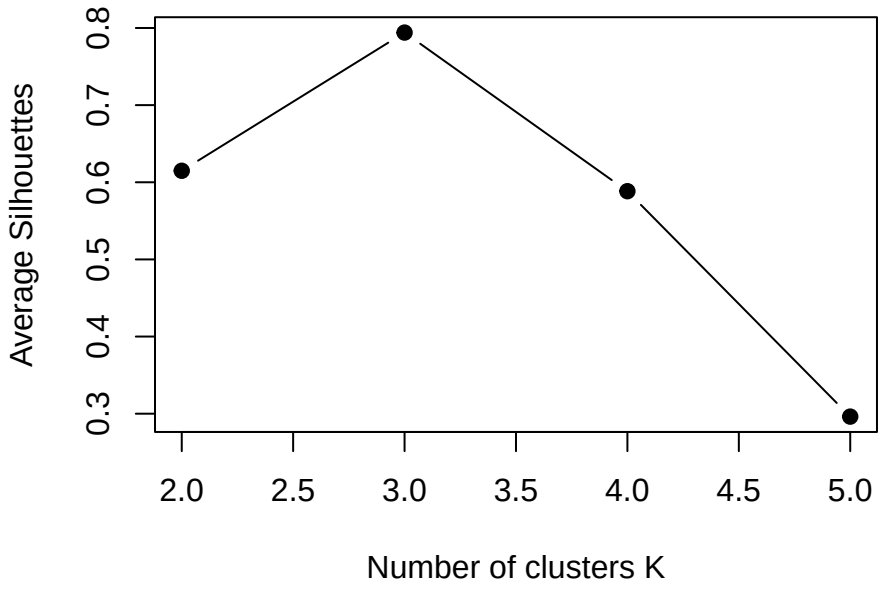
Choix du nombre de classe : La fonction `silhouette()` du package **cluster** permet de retourner le coefficient silouhette des observations pour un cluster donné. Ici, on calcule la moyenne des coefficients silhouette en faisant varier le nombre de classes. Ici, on voit que le choix optimal du nombre de classes est 3.

```
library(cluster)
```

```
Warning: le package 'cluster' a été compilé avec la version R 4.4.2
```

```
avgsil = function(k) {
    km.res = kmeans(my_df, centers = k, nstart = 100)
    ss = silhouette(km.res$cluster, dist(my_df))
    mean(ss[, 3])
}
```

```
k.values <- 2:5
avgsilvalues <- sapply(k.values, avgsil)
plot(k.values, avgsilvalues, type = "b", pch = 19,
     xlab = "Number of clusters K", ylab = "Average Silhouettes")
```



Pour obtenir des résultats stables, l'argument `nstart=` permet de tester plusieurs centres initiaux et la fonction retourne les classes les plus stables :

Tableau CL1

```
classif <- kmeans(my_df, 3, nstart = 10)
names(classif)
```

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

Pour obtenir les résultats de la classification. On obtient les numéros du groupe auquel chaque observation est affectée :

Tableau CL2

```
classif$cluster
```

```
[1] 1 1 2 2 3 3
```

Les centres (ou moyennes) des groupes, dont les coordonnées doivent être commentées pour réaliser la typologie, sont donnés par :

Tableau CL3

```
classif$centers
```

```
   REV EDUC
1  5.5  5.5
2 15.5 14.5
3 27.5 19.5
```

Le nombre d'individus dans chaque groupe est donné par :

Tableau CL4

```
classif$size
```

```
[1] 2 2 2
```

Les sommes des carrés associées à la classification sont présentées ci-dessous :

Tableau CL5

- somme des carrés inter-groupes :

```
classif$betweenss
```

```
[1] 686.6667
```

- somme des carrés totale :

```
classif$totss
```

```
[1] 701.6667
```

- somme des carrés intra-groupes :

```
classif$tot.withinss
```

```
[1] 15
```

- somme des carrés pour chaque groupe :

```
classif$withinss
```

```
[1]  1  1 13
```

On calcule le rapport inertie inter-classes / inertie totale ainsi :

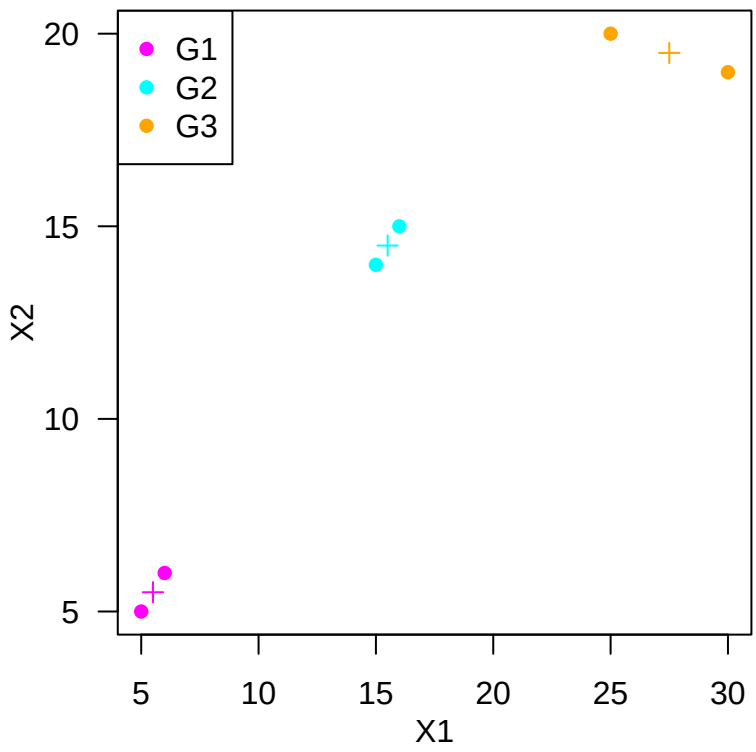
Tableau CL6

```
R2 <- classif$betweenss/classif$totss
R2
```

```
[1] 0.9786223
```

Comme il n'y a que deux variables ici, on peut représenter les groupes sur le nuage de points :

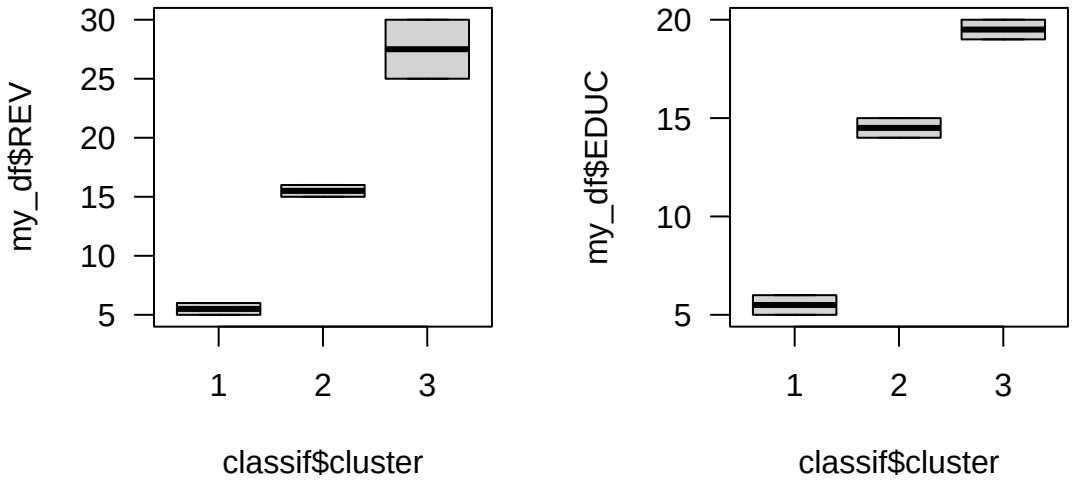
```
my_col <- c("magenta", "cyan", "orange")
par(las = 1,  mar = c(3, 3, 0.75, 0.5), mgp = c(2., 1, 0))
plot(my_df$REV, my_df$EDUC, main = "", xlab = "X1", ylab = "X2",
     col = my_col[classif$cluster], pch = 16)
points(classif$centers, col = my_col, pch = 3)
legend("topleft", legend = paste0("G", 1:3), pch = 16,
       col = c("magenta", "cyan", "orange"))
```



On représente les boîtes à moustaches parallèles des deux variables utilisées en fonction des classes obtenues. On présente l'illustration, même s'il n'y a pas suffisamment d'observations pour obtenir un graphique convenable.

Figure CL7

```
par(mfrow = c(1, 2), las = 1)
boxplot(my_df$REV ~ classif$cluster)
boxplot(my_df$EDUC ~ classif$cluster)
```



Les rapports de corrélations entre les deux variables dans ce cas sont donnés par :

Tableau CL8

```
summary(lm(my_df$REV ~ as.factor(classif$cluster)))$r.squared
```

```
[1] 0.9729369
```

```
summary(lm(my_df$EDUC ~ as.factor(classif$cluster)))$r.squared
```

```
[1] 0.9926048
```

3 Classification Ascendante Hiérarchique (CAH)

D'abord, on calcule la matrice de distance entre les observations à l'aide de la fonction `dist()` :

```
my_dist <- dist(my_df)
my_dist
```

```
          1         2         3         4         5
2  1.414214
3 13.453624 12.041595
4 14.866069 13.453624  1.414214
5 25.000000 23.600847 11.661904 10.295630
6 28.653098 27.294688 15.811388 14.560220  5.099020
```

Commentaires : on voit que la distance la plus faible est celle entre les observations 1 et 2. Ces deux observations seront donc rassemblées à l'étape 1 de l'algorithme.

A partir de cette matrice de distance, la fonction `hclust()` va faire la CAH. L'argument `method=` permet d'indiquer quelle méthode il faut utiliser pour calculer les distances entre deux groupes. Parmi les méthodes possibles, on utilise ici la méthode de Ward :

Tableau CL9

```
classif2 <- hclust(my_dist, method = "ward.D2")
```

Plusieurs valeurs sont retournées par la fonction :

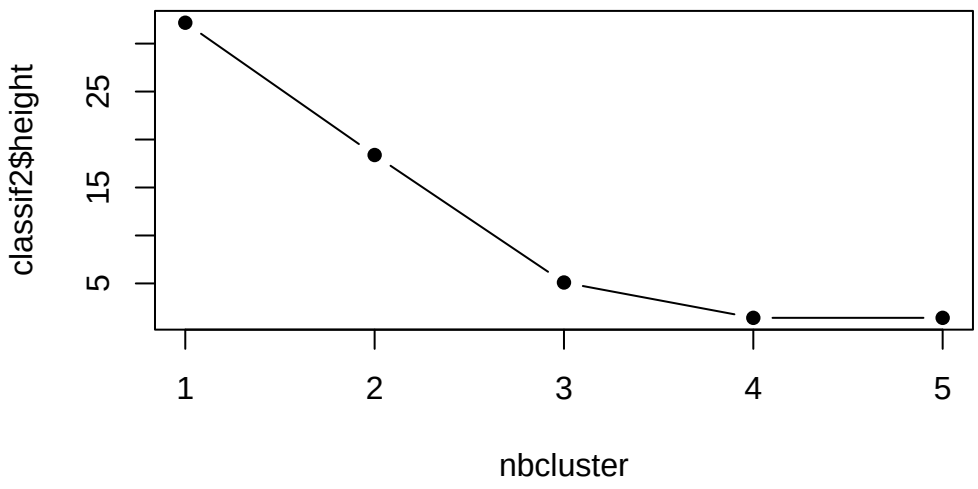
```
names(classif2)
```

```
[1] "merge"       "height"      "order"       "labels"      "method"
[6] "call"        "dist.method"
```

On repésente le graphique des hauteurs :

Figure CL10

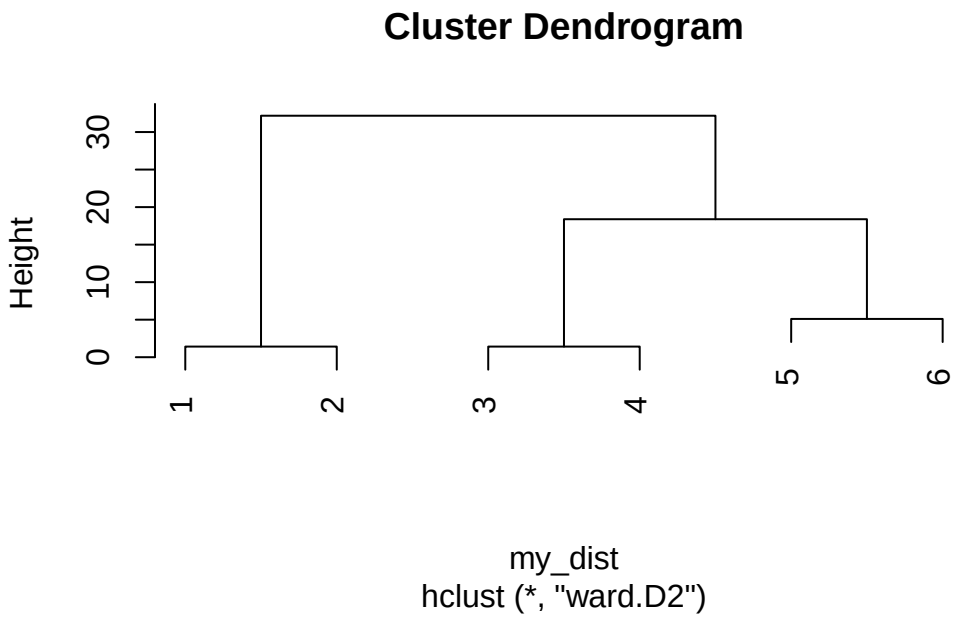
```
nbcluster <- 5:1
plot(nbcluster, classif2$height, type = "b", pch = 16)
```



On représente le dendrogramme directement avec la fonction `plot()`.

Figure CL11

```
plot(classif2)
```



Commentaire : on observe un saut important en choisissant 3 classes.

Pour récupérer les numéros de groupe auquel chaque observation est affectée, on utilise la fonction `cutree()` en indiquent le nombre de classes choisi. **Tableau CL12** :

```
groupe2 <- cutree(classif2, k = 3)
groupe2
```

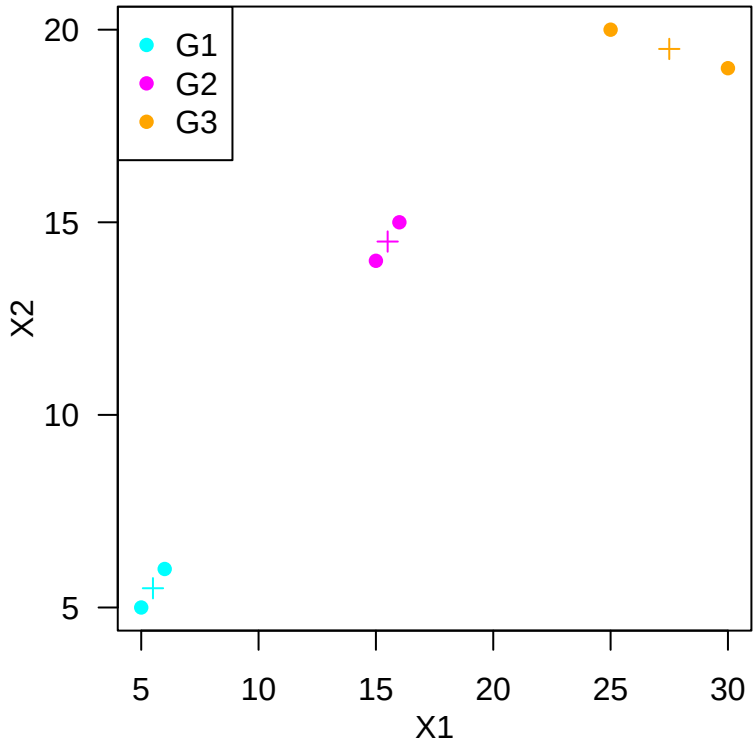
```
[1] 1 1 2 2 3 3
```

Commentaires : on trouve les mêmes classes qu'en utilisant la méthode des k-means

On peut également afficher les groupes sur le nuage de points:

Figure CL13

```
my_col <- c( "cyan", "magenta", "orange")
par(las = 1,  mar = c(3, 3, 0.75, 0.5), mgp = c(2., 1, 0))
plot(my_df$REV, my_df$EDUC, main = "", xlab = "X1", ylab = "X2",
     col = my_col[groupe2], pch = 16)
centers <- aggregate(my_df, list(groupe2), mean)[,-1]
points(centers, col = my_col, pch = 3)
legend("topleft", legend = paste0("G", 1:3), pch = 16,
       col = c( "cyan", "magenta", "orange"))
```


G1
G2
G3
20
15
10
5
X2
5
10
15
20
25
30
X1