TP3 : Classification automatique

Zakaria Laabsi & Chloé Cordoval

Importez le fichier villes.csv dans R (read.table("villes.csv",header=T)) et examinez-le.

```
library(readr)
VillesBrut <- read.table("villes.csv",header=T,sep = "\t")
#View(VillesBrut)
```

Nous avons donc un tableau de données qui réunit des données décrivant 100 villes de France selon 54 variables représentant plusieurs indicateurs socio-économiques, écologiques, culturels.

On désire mettre en oeuvre une classification thématique complétée par un treillis de Galois. Pour ce faire, on doit regrouper des thèmes afin de les rendre moins nombreux. On décide le regroupement suivant (ce n'est qu'un exemple, que vous pouvez remettre en question si un autre vous paraît plus pertinent ou vous intéresse davantage):

- Économie: Chomage, ChomageJeunes, ChomageLong, EvoluEmploiCree, Activite,EmploiFeminin, EmploiCommune, DefaillEntreprise, SalaireAnnuel, ImpotRevenu, ImpotFortune, Imposables, MetreCarreAncien, TaxeHabitation, FoncierBati, MetreCubeEau,EvolDemographique, Vieillissement, AttiranceGlobale,AttiranceActifs, Proprietaires, LogtSup4pieces, LogtInsalubre, LogtVacant, LogtConstruction
- Risques: Criminalite, EvolutionCrimes, SecuriteRoutiere, Inondations, TerrainsPollues, UsinesRisques, MortaliteInfantile, MortaliteCancerPoumon, MortaliteAlcool, DecesInfarctus,TauxSuicide, MortaliteGlobale, TailleClassesPrimaires, Retard6eme, Retard3eme, RetardTerminale
- Nature: Mer, Ski, Soleil, Pluie, Temperature, MarcheAPied
- Culture: Musees, Cinema, MonumHistoriques, PretLivres, RestaurDistingues, Presse, Etudiants

I- Préparation des données.

a) Chargez les données dans un dataframe (ci-dessous Villesbrut).
b) Standardisez les variables.

NB: Vous pouvez utiliser la fonction scale sur le dataframe entier, mais il faut en corriger le résultat en le multipliant par le bon facteur qui est içi = (N/N+1)

```
Villes = data.frame(scale(VillesBrut[,2:55]) * sqrt(54/53))
VillesBrut[49,2]<-0.113
```

II-CAH de l'ensemble des variables.

a) création de la matrice des distances euclidiennes:

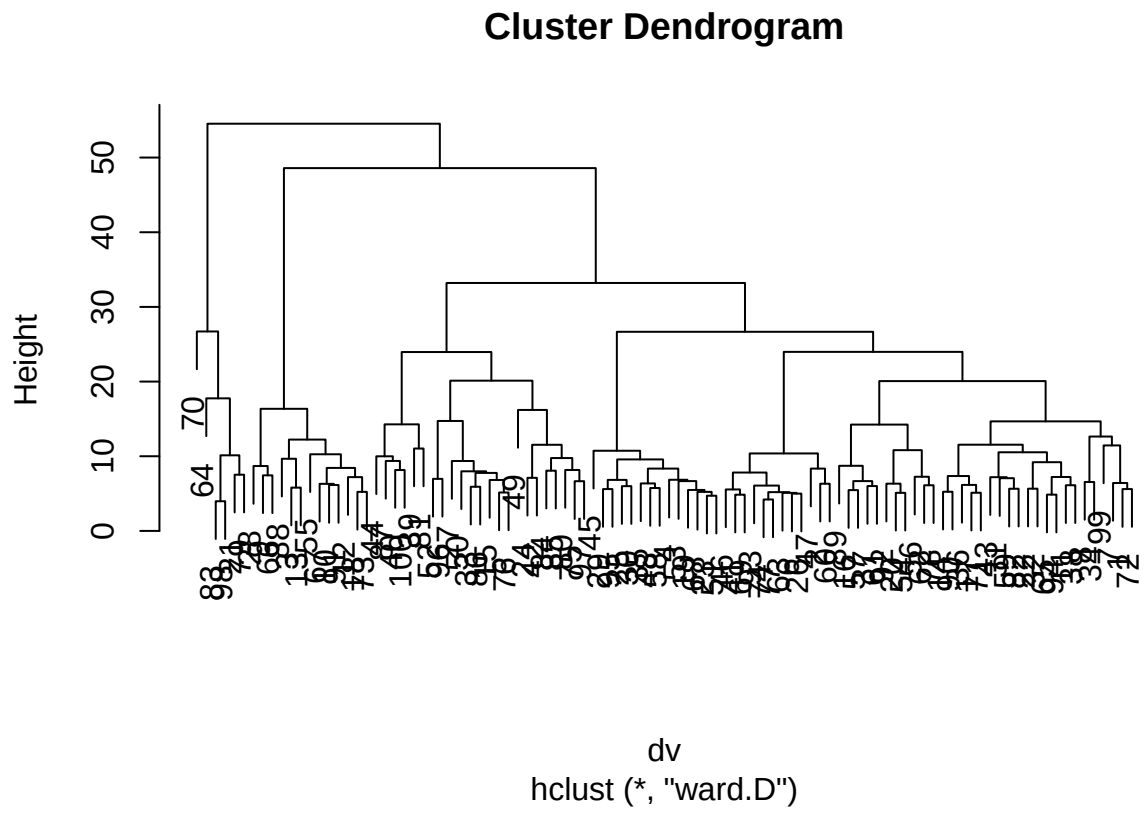
```
dv=dist(Villes, method="euclidean")
```

b) CAH avec Ward (vous pouvez essayer plusieurs indices pour juger de la stabilité des résultats):

```
CAHV = hclust(d=dv, method = "ward.D")
```

c) Dendrogramme de la hiérarchie indicée:

```
plot(CAHV)
```



d) Coupure de l'arbre et fabrication de la variable de classe correspondant à la partition obtenue, par exemple k=2 classes:

```
PV2 = cutree(tree = CAHV, k=2)

print(PV2)
```

```
##   [1] 1 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 1 1 1 1 1 1 1 1
##  [38] 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 1 1 1 1 1 2 1 1 1 1
##  [75] 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
```

e) Calcul du R2 des variables avec la variable de classe. On va stocker tous les R2 dans un seul vecteur: R2.

```
R2_PV2 = cbind(rep(0 , ncol(Villes)))
```

Puis, on calcule les R2 de toutes les variables avec la variable de classe et on met les résultats dans R2:

```
for (i in cbind(1:ncol(Villes))) {

  R2_PV2[i] =
    summary(lm(Villes[,i]~as.factor(PV2)))$r.squared
}
```

On peut réassigner les noms des variables aux éléments de ce vecteur:

```
row.names(R2_PV2) = colnames(Villes)
```

f) Calcul du R2 de la partition:

```
##   [1] 1 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 1 1 1 1 1 1 1 1
##  [38] 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 1 1 1 1 1 2 1 1 1 1
##  [75] 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
```

Démontrez que, les variables ayant toutes le même poids, le R2 de la partition est égal à la moyenne des R2 des variables. On se sert de cette propriété pour le calculer:

La démonstration a été faite aux TP précédents :

$$R^2 = \frac{\sigma_{inter}}{\sigma^2} = \sum_{j=1}^{J} w^j (\overline{x^j})^2$$

Les variables ont le même poids, d'où

$$\sum_{i=1}^{K} w_i = \sum_{i=1}^{K} \frac{1}{K}$$

Par la suite, on fait la moyenne arithémtique

$$\sum_{i=1}^{K} w_i \sum_{j=1}^{J} w^j \sum_{i=1}^{K} (\overline{x^j})^2$$

$$\sum_{i=1}^{K} w_i \sum_{j=1}^{J} w^j \sum_{i=1}^{K} (\overline{x^j})^2$$

$$\frac{1}{K} \sum_{j=1}^{J} w^j \sum_{i=1}^{K} (\overline{x^j})^2$$

Finalement, on a

$$\frac{1}{K} \sum_{j=1}^{J} w^j \|\overline{x^j}\|^2 = R^2_{part}$$

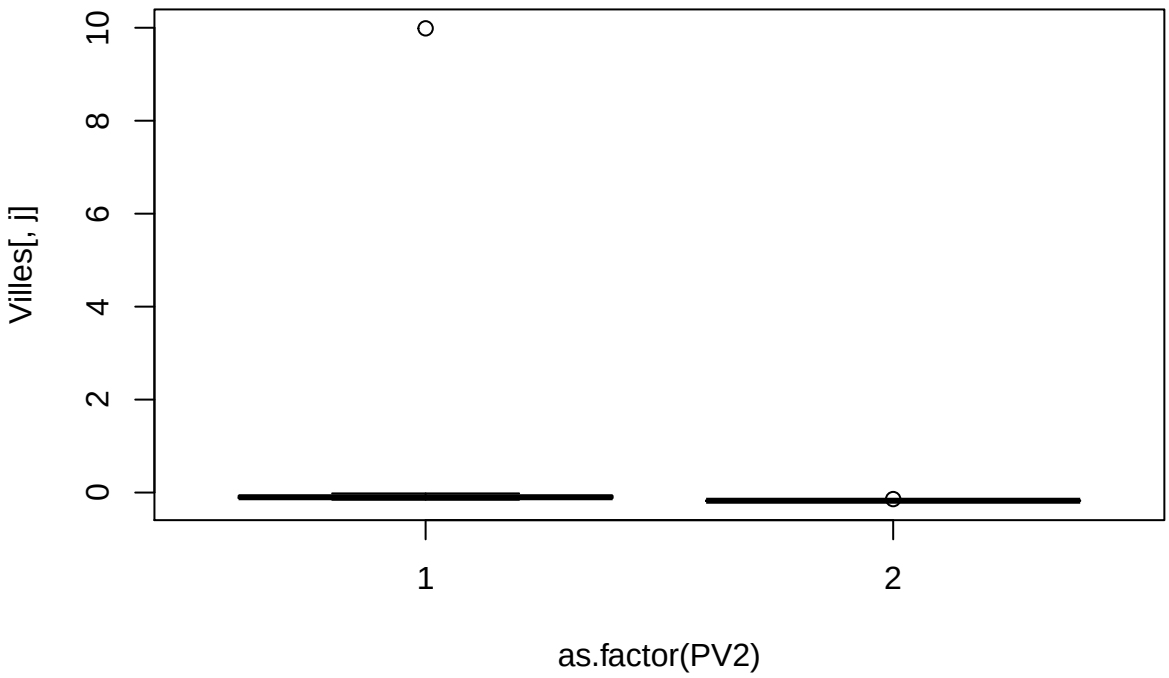
, ce qui démontre la preuve

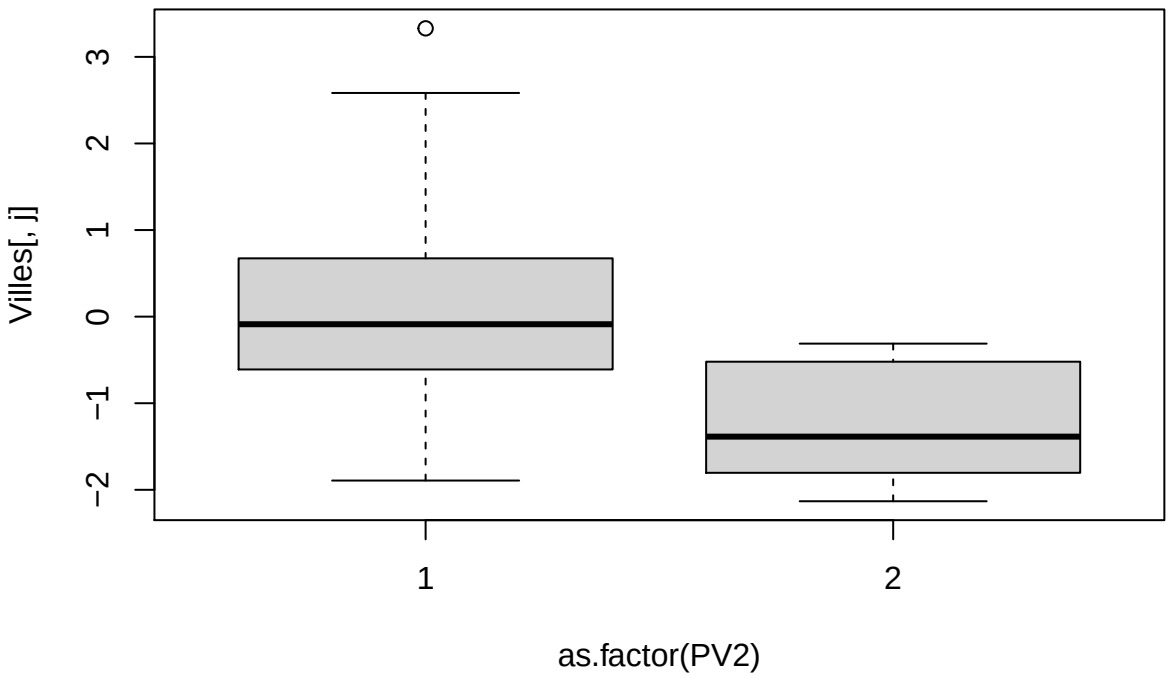
```
R2G_PV2 = mean(R2_PV2)

print(R2G_PV2)
```

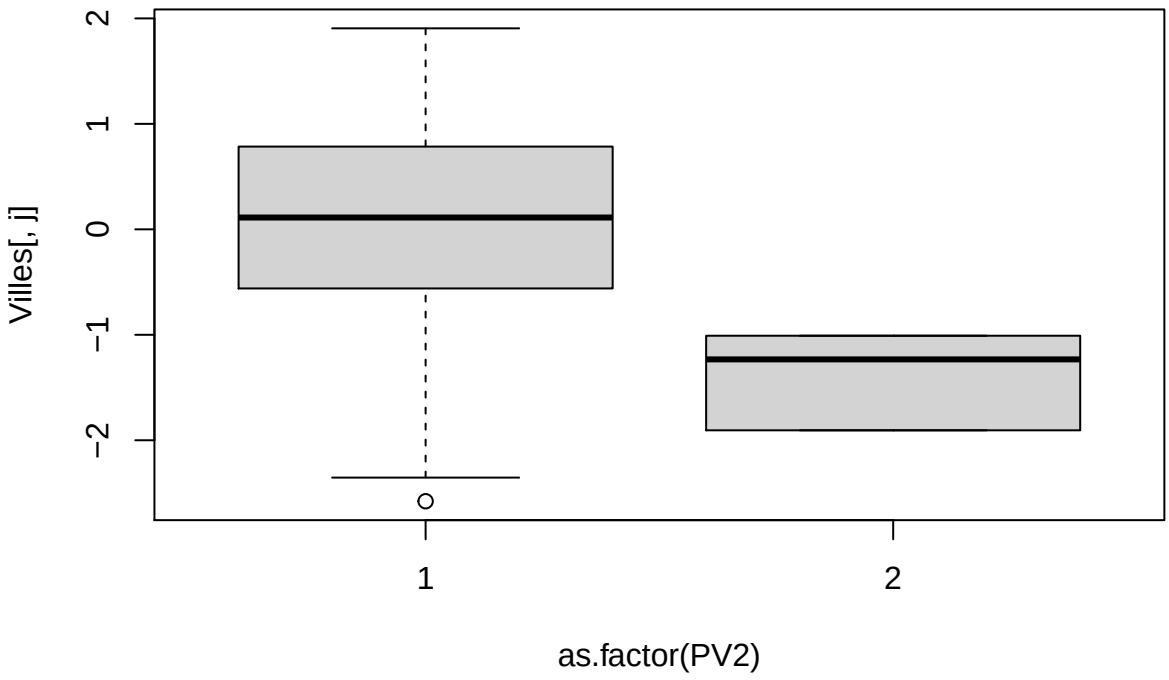
```
## [1] 0.1150262
```

g) Boxplot d'une variable x^{j} conditionnellement à la variable de classe:

```
for (j in 1:3){
boxplot(Villes[,j]~as.factor(PV2))
}
```


Villes[, j]
0
2
4
6
8
10
1
2
as.factor(PV2)


Villes[, j]
3
2
1
0
−1
−2
1
2
as.factor(PV2)



h) Optimisation d'une partition avec les K-means, e.g. pour celle en 2 classes:

- Transformation d'une variable qualitative en matrice d'indicatrices:

```
IC2Vil = data.frame(model.matrix(~as.factor(PV2)-1))
```

- Calcul matriciel des centres de gravité de classes de la CAH: Démontrez que les centres de gravité (CentresC2) sont obtenus par la formule programée ci-dessous:

```
mIC2Vil = as.matrix(IC2Vil)

mVil = as.matrix(Villes)

CentresC2 = solve(t(mIC2Vil) %*% mIC2Vil) %*% t(mIC2Vil)%*% mVil

#print(CentresC2)
```

- K-means à partir de ces centres initiaux:

```
KMV2 = kmeans(Villes, CentresC2)
```

- La variable de classe ainsi produite est dans:

```
KMV2$cluster
```

```
##   [1] 1 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 1 1 1 1 1 1 1 1
##  [38] 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 1 1 1 1 1 2 1 1 1 1
##  [75] 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
```

III- CAH par thèmes.

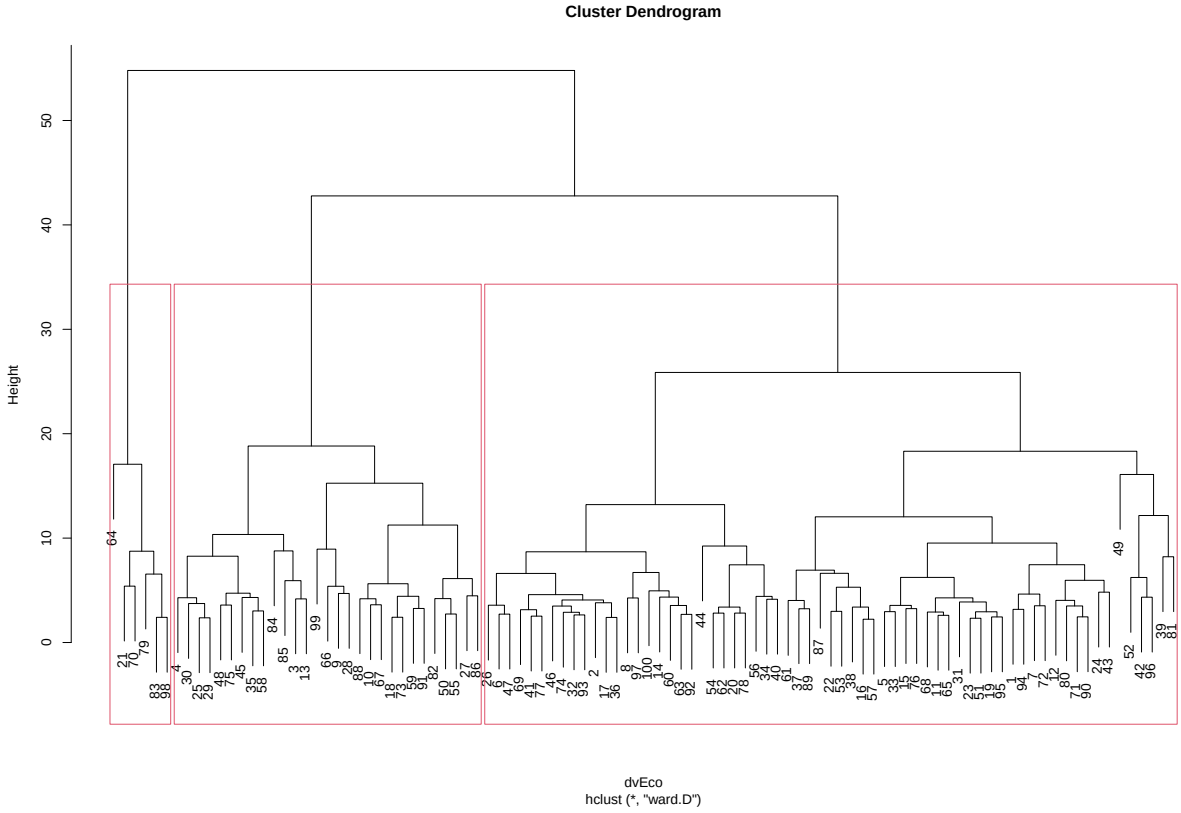
À l'aide des acquits précédents..

1-

a) Procédez à une CAH par thème.
b) Choisissez une "bonne" partition n'ayant pas plus de 3 classes (seulement deux quand c'est possible) pour chaque thème, et optimisez-la à l'aide de K-means. Interprétez les classes de la partition optimisée, et trouvez un qualificatif pour chacune.

On va regrouper les thèmes afin de les rendre moins nombreux Ce qui donnera :

- Economie



- Optimisation Économie

```
#Partition en 2 classes

P2Eco = cutree(tree = CAHVEco, k=2)

R2_P2Eco = cbind(rep(0 , ncol(Economie)))

for (i in cbind(1:ncol(Economie))) {

  R2_P2Eco[i] =
    summary(lm(Economie[,i]~as.factor(P2Eco)))$r.squared
}

row.names(R2_P2Eco) = colnames(Economie)

IC2Eco = data.frame(model.matrix(~as.factor(P2Eco)-1))
```

```r
mIC2Eco = as.matrix(IC2Eco)

mEco = as.matrix(Economie)

CentresC2Eco = solve(t(mIC2Eco) %*% mIC2Eco) %*% t(mIC2Eco)%*% mEco

KM2Eco = kmeans(Economie, CentresC2Eco)

print(KM2Eco)
```

```
## K-means clustering with 2 clusters of sizes 94, 6
##
## Cluster means:
##   Villes.Chomage Villes.ChomageJeunes Villes.ChomageLong Villes.EvoluEmploiCree
## 1     0.01094307           0.08021319         0.08824228              0.0175522
## 2    -0.17144145          -1.25667337        -1.38246241             -0.2749845
##   Villes.Activite Villes.EmploiFeminin Villes.EmploiCommune
## 1     -0.09525613          -0.06548767            0.1168436
## 2      1.49234600           1.02597345           -1.8305490
##   Villes.DefaillEntreprise Villes.SalaireAnnuel Villes.ImpotRevenu
## 1               0.04151495           -0.2227755         -0.1978207
## 2              -0.65040092            3.4901487          3.0991915
##   Villes.ImpotFortune Villes.Imposables Villes.MetreCarreAncien
## 1          -0.1827696         -0.196942              -0.2125872
## 2           2.8633903          3.085424               3.3305325
##   Villes.TaxeHabitation Villes.FoncierBati Villes.MetreCubeEau
## 1             0.1212111            0.14127         -0.01467898
## 2            -1.8989742           -2.21323          0.22997072
##   Villes.EvolDemographique Villes.Vieillissement Villes.AttiranceGlobale
## 1              0.007542201           0.002336805             -0.06115536
## 2             -0.118161153          -0.036609939              0.95810065
##   Villes.AttiranceActifs Villes.Proprietaires Villes.LogtSup4pieces
## 1            -0.08000543          -0.01519532            0.06427342
## 2             1.25341848           0.23806002           -1.00695025
##   Villes.LogtInsalubre Villes.LogtVacant Villes.LogtConstruction
## 1          -0.09524332      0.0009490186              0.04356013
## 2           1.49214537     -0.0148679574             -0.68244208
##
## Clustering vector:
##   [1] 1 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 1 1 1 1 1 1 1 1
##  [38] 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 1 1 1 1 1 2 1 1 1 1
##  [75] 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
##
## Within cluster sum of squares by cluster:
## [1] 1844.7191  191.4178
##  (between_SS / total_SS =  19.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
#Partition en 3 classes
```

```r
P3Eco = cutree(tree = CAHVEco, k=3)

R2_P3Eco = cbind(rep(0 , ncol(Economie)))

for (i in cbind(1:ncol(Economie))) {

  R2_P3Eco[i] =
    summary(lm(Economie[,i]~as.factor(P3Eco)))$r.squared
}

row.names(R2_P3Eco) = colnames(Economie)

IC3Eco = data.frame(model.matrix(~as.factor(P3Eco)-1))

mIC3Eco = as.matrix(IC3Eco)

mEco = as.matrix(Economie)

CentresC3Eco = solve(t(mIC3Eco) %*% mIC3Eco) %*% t(mIC3Eco)%*% mEco

KM3Eco = kmeans(Economie, CentresC3Eco)

print(KM3Eco)
```

```
## K-means clustering with 3 clusters of sizes 56, 38, 6
##
## Cluster means:
##   Villes.Chomage Villes.ChomageJeunes Villes.ChomageLong Villes.EvoluEmploiCree
## 1     0.07581079           -0.1607579         -0.2562092            -0.01859882
## 2    -0.08465146            0.4353286          0.5958551             0.07082739
## 3    -0.17144145           -1.2566734         -1.3824624            -0.27498445
##   Villes.Activite Villes.EmploiFeminin Villes.EmploiCommune
## 1       0.1847225            0.3350944           -0.2314573
## 2      -0.5078562           -0.6558192            0.6301291
## 3       1.4923460            1.0259735           -1.8305490
##   Villes.DefaillEntreprise Villes.SalaireAnnuel Villes.ImpotRevenu
## 1               0.12682995           -0.1474390         -0.1454841
## 2              -0.08421241           -0.3337975         -0.2749484
## 3              -0.65040092            3.4901487          3.0991915
##   Villes.ImpotFortune Villes.Imposables Villes.MetreCarreAncien
## 1          -0.1511659        0.01553263              -0.1205524
## 2          -0.2293435       -0.51006248              -0.3482174
## 3           2.8633903        3.08542450               3.3305325
##   Villes.TaxeHabitation Villes.FoncierBati Villes.MetreCubeEau
## 1            0.09865289          0.0260677          -0.1224061
## 2            0.15445482          0.3110418           0.1440768
## 3           -1.89897420         -2.2132297           0.2299707
##   Villes.EvolDemographique Villes.Vieillissement Villes.AttiranceGlobale
## 1                0.3095311           -0.11992306               0.5600981
## 2               -0.4374940            0.18250924              -0.9766868
## 3               -0.1181612           -0.03660994               0.9581006
##   Villes.AttiranceActifs Villes.Proprietaires Villes.LogtSup4pieces
## 1              0.4895529           -0.5951500            -0.3473396
## 2             -0.9193546            0.8394748             0.6708610
```

```
## 3              1.2534185            0.2380600            -1.0069503
##   Villes.LogtInsalubre Villes.LogtVacant Villes.LogtConstruction
## 1          -0.05257972        0.07009180             -0.05513876
## 2          -0.15811600       -0.10094561              0.18901114
## 3           1.49214537       -0.01486796             -0.68244208
##
## Clustering vector:
##   [1] 1 1 2 2 2 1 1 1 1 2 1 2 2 1 2 1 1 2 1 1 3 1 2 2 2 1 2 2 2 2 2 1 2 1 2 1 1
##  [38] 1 1 1 1 2 1 1 2 1 1 2 1 2 2 1 1 1 2 1 1 2 2 1 1 1 1 3 1 2 2 2 1 3 1 1 2 1
##  [75] 2 1 1 1 3 1 1 2 3 2 2 2 1 2 1 1 2 1 1 1 2 1 1 3 1 1
##
## Within cluster sum of squares by cluster:
## [1] 975.9185 594.9753 191.4178
##  (between_SS / total_SS =  30.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

##Interpretation des classes "Economie"

#Classes

```
#Eco <- cbind.data.frame(Villes[,1],Economie)
Eco <- cbind.data.frame(VillesBrut$Ville,Economie)

Ceco1 <- vector()
Ceco2 <- vector()
Ceco3 <- vector()

a<-1
b<-1
c<-1

for (i in 1:100) {

  if (KM3Eco$cluster[i] == 1){
    Ceco1[a] <- as.matrix(Eco[i,1])
    a <- a+1
  }

  if (KM3Eco$cluster[i] == 2){
   Ceco2[b] <- as.matrix(Eco[i,1])
   b <- b+1
  }

  if (KM3Eco$cluster[i] == 3){
    Ceco3[c] <- as.matrix(Eco[i,1])
    c <- c+1
  }
}
```

On aura donc :

```
print(Ceco1)
```

```
##  [1] "Agen"            "AixEnProvence"   "Angers"          "Angouleme"
##  [5] "Annecy"          "Antibes"         "Auxerre"         "Bayonne"
##  [9] "Belfort"         "Besancon"        "Blois"           "Bordeaux"
## [13] "BourgEnBresse"   "Caen"            "Chambery"        "Chartres"
## [17] "ClermontFerrand" "Colmar"          "Compiegne"       "CorbeilEssonnes"
## [21] "Creteil"         "Dijon"           "Epinal"          "Evry"
## [25] "Grenoble"        "La Rochelle"     "Laval"           "Lille"
## [29] "Limoges"         "Lyon"            "Melun"           "Metz"
## [33] "Montpellier"     "Mulhouse"        "Nancy"           "Nantes"
## [37] "Nevers"          "Orleans"         "Pau"             "Perigueux"
## [41] "Poitiers"        "Reims"           "Rennes"          "Rouen"
## [45] "SaintBrieuc"     "SaintDenis"      "Sarcelles"       "Strasbourg"
## [49] "Tarbes"          "Toulouse"        "Tours"           "Troyes"
## [53] "Valenciennes"    "Vannes"          "Vichy"           "Villeurbanne"
```

puis,

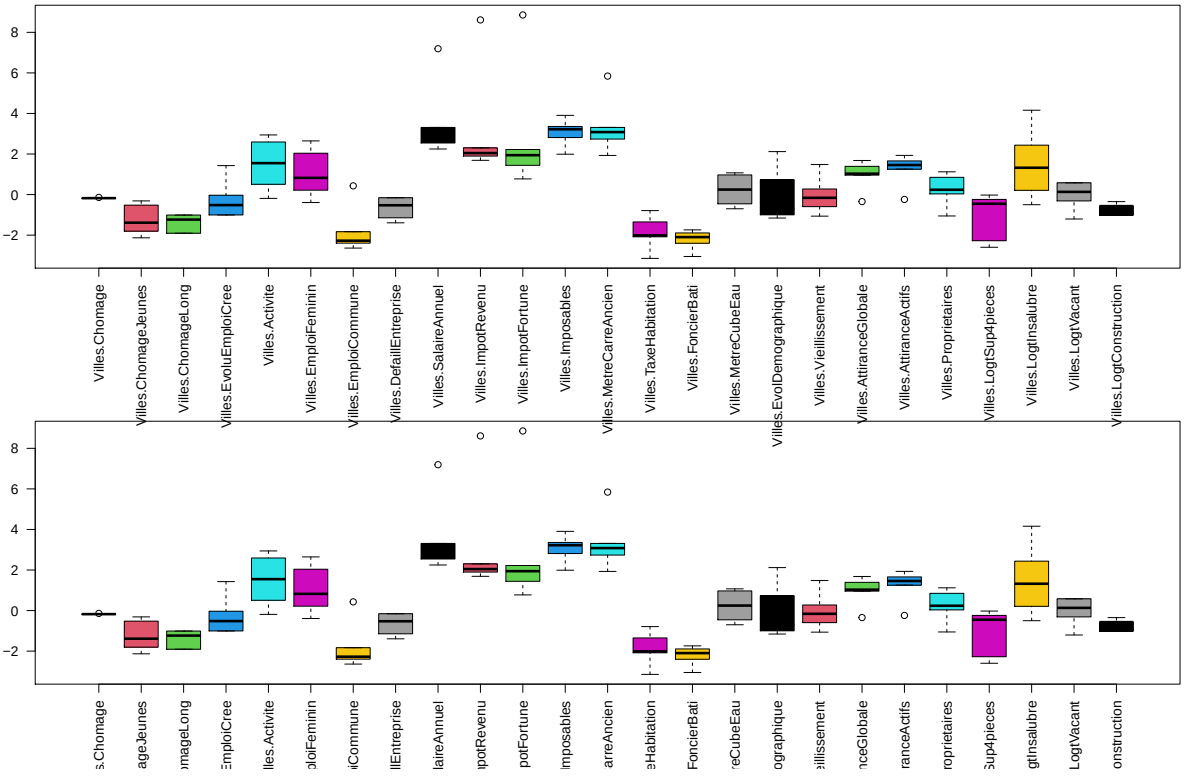
```
print(Ceco2)
```

```
##  [1] "Ajaccio"             "Albi"                "Amiens"
##  [4] "Arles"               "Avignon"             "Bastia"
##  [7] "Beauvais"            "Beziers"             "Bourges"
## [10] "Brest"               "BriveLaGaillarde"    "Calais"
## [13] "Cannes"              "Carcassonne"         "Castres"
## [16] "ChalonSurSaone"      "CharlevilleMezieres" "Cholet"
## [19] "Dunkerque"           "Gap"                 "La RocheSurYon"
## [22] "LeHavre"             "LeMans"              "Marseille"
## [25] "Montauban"           "Montlucon"           "Nice"
## [28] "Nimes"               "Niort"               "Perpignan"
## [31] "Quimper"             "SaintEtienne"        "SaintMalo"
## [34] "SaintNazaire"        "SaintQuentin"        "Sete"
## [37] "Toulon"              "Valence"
```

et,

```
print(Ceco3)
```

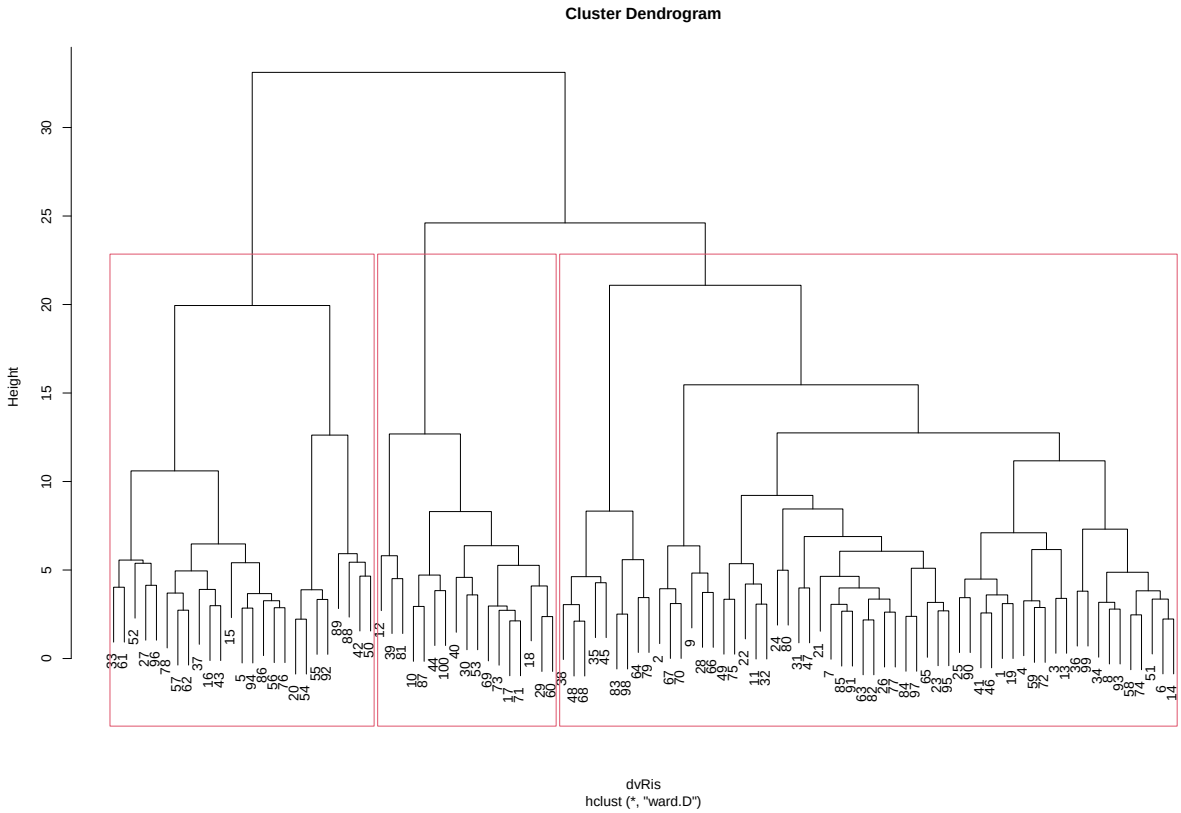
```
## [1] "BoulogneBillancourt" "NeuillySurSeine"     "Paris"
## [4] "RueilMalmaison"      "SaintGermainEnLaye"  "Versailles"
```



##Interprétation sur les classes :

- La classe 1 contient les villes pauvres de la France
- La classe 2 nous avons des villes comme Chambery ou bien Strasbourg qui sont assez actifs sur le plan économique mais aussi des villes comme Anger qui ne sont pas spécialement actifs sur le plan économique que les autres mais qui ont la particularité d'avoir des indicateurs plutôt élevés sur l'implication de la femme dans le monde du travail.
- La classe 3 contient spécialement certaines villes de l'ouest de île de France avec une population à forte présence de cadres supérieurs, aux revenus les plus élevés, là où se concentre les principaux sièges des sociétés (exemple le quartier de la Défense dans les Haut-de-Seine, certaines villes également des Haut-de-Seine comme Neuilly-sur-Seine, Rueil-Malmaison, Boulogne-Billancourt dont la population a les revenus les plus élevés de France). Ce sont des villes avec une croissance économique importante, et un cadre de vie élevé.

##Risques




```
#Partition en 2 classes

P2Ris = cutree(tree = CAHVRis, k=2)

R2_P2Ris = cbind(rep(0 , ncol(Risques)))

for (i in cbind(1:ncol(Risques))) {

  R2_P2Ris[i] =
    summary(lm(Risques[,i]~as.factor(P2Ris)))$r.squared
}

row.names(R2_P2Ris) = colnames(Risques)

IC2Ris = data.frame(model.matrix(~as.factor(P2Ris)-1))

mIC2Ris = as.matrix(IC2Ris)

mRis = as.matrix(Risques)

CentresC2Ris = solve(t(mIC2Ris) %*% mIC2Ris) %*% t(mIC2Ris)%*% mRis

KM2Ris = kmeans(Risques, CentresC2Ris)

print(KM2Ris)
```

```
## K-means clustering with 2 clusters of sizes 65, 35
```

```
##
## Cluster means:
##   Villes.Criminalite Villes.EvolutionCrimes Villes.SecuriteRoutiere
## 1         0.05281342             0.08522099               0.1050281
## 2        -0.09808206            -0.15826755              -0.1950521
##   Villes.Inondations Villes.TerrainsPollues Villes.UsinesRisques
## 1        -0.09575866             -0.2563895           -0.1979678
## 2         0.17783751              0.4761519            0.3676545
##   Villes.MortaliteInfantile Villes.MortaliteCancerPoumon Villes.MortaliteAlcool
## 1                -0.1868984                   -0.3903430             -0.4846342
## 2                 0.3470970                    0.7249227              0.9000349
##   Villes.DecesInfarctus Villes.TauxSuicide Villes.MortaliteGlobale
## 1            -0.4042999         -0.1865952              -0.5421862
## 2             0.7508428          0.3465339               1.0069172
##   Villes.TailleClassesPrimaires Villes.Retard6eme Villes.Retard3eme
## 1                    0.08095563       -0.01689239        -0.1319263
## 2                   -0.15034618        0.03137159         0.2450059
##   Villes.RetardTerminale
## 1             -0.1970923
## 2              0.3660285
##
## Clustering vector:
##   [1] 1 1 1 1 2 1 2 1 1 1 1 1 1 1 2 2 1 1 1 2 1 1 1 2 1 1 2 1 1 1 2 1 2 1 1 2 2
##  [38] 1 2 1 1 2 1 1 1 1 1 1 1 2 1 2 1 1 2 2 2 1 2 1 2 2 1 1 2 1 1 1 1 1 1 1 1 1
##  [75] 1 2 1 2 1 2 2 1 1 2 2 2 2 2 2 1 1 1 1 2 1 2 1 1 2 1
##
## Within cluster sum of squares by cluster:
## [1] 844.9229 562.3234
##  (between_SS / total_SS =  12.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
#Partition en 3 classes

P3Ris = cutree(tree = CAHVRis, k=3)

R2_P3Ris = cbind(rep(0 , ncol(Risques)))

for (i in cbind(1:ncol(Risques))) {

  R2_P3Ris[i] =
    summary(lm(Risques[,i]~as.factor(P3Ris)))$r.squared
}

row.names(R2_P3Ris) = colnames(Risques)

###

IC3Ris = data.frame(model.matrix(~as.factor(P3Ris)-1))

mIC3Ris = as.matrix(IC3Ris)
```

```r
mRis = as.matrix(Risques)

CentresC3Ris = solve(t(mIC3Ris) %*% mIC3Ris) %*% t(mIC3Ris)%*% mRis

KM3Ris = kmeans(Risques, CentresC3Ris)

print(KM3Ris)
```

```
## K-means clustering with 3 clusters of sizes 47, 25, 28
##
## Cluster means:
##   Villes.Criminalite Villes.EvolutionCrimes Villes.SecuriteRoutiere
## 1        -0.21996027            -0.09611717               0.2482376
## 2         0.07347296            -0.40683137              -0.1885141
## 3         0.30361817             0.52458184              -0.2483685
##   Villes.Inondations Villes.TerrainsPollues Villes.UsinesRisques
## 1         -0.1420112             -0.3620900           -0.2636671
## 2          0.4583481              1.0587377            0.7273165
## 3         -0.1708635             -0.3375077           -0.2068056
##   Villes.MortaliteInfantile Villes.MortaliteCancerPoumon Villes.MortaliteAlcool
## 1                -0.2356103                  -0.43355204             -0.3216265
## 2                 0.2848908                   0.83857373              0.8546406
## 3                 0.1411219                  -0.02097847             -0.2231989
##   Villes.DecesInfarctus Villes.TauxSuicide Villes.MortaliteGlobale
## 1           -0.53172340        -0.04329084             -0.56246781
## 2            0.93147716         0.02701737              1.00905210
## 3            0.06085968         0.04854412              0.04320302
##   Villes.TailleClassesPrimaires Villes.Retard6eme Villes.Retard3eme
## 1                    0.07730157        -0.4623269        -0.5303249
## 2                    0.17699512        -0.2597313        -0.1067137
## 3                   -0.28778756         1.0079517         0.9854683
##   Villes.RetardTerminale
## 1             -0.5089785
## 2              0.1747967
## 3              0.6982882
##
## Clustering vector:
##   [1] 1 1 1 1 2 1 3 1 1 3 3 3 1 1 2 2 3 3 1 2 1 3 1 2 3 1 2 1 3 3 3 1 2 1 1 3 2
##  [38] 1 3 3 1 2 1 3 1 1 1 1 1 2 1 2 3 2 2 2 2 1 1 3 2 2 1 1 3 1 1 1 1 1 3 1 3 1
##  [75] 1 2 1 2 1 3 3 3 1 3 3 2 3 2 2 1 1 2 1 2 3 2 1 1 1 3
##
## Within cluster sum of squares by cluster:
## [1] 553.8702 377.9839 354.5494
##  (between_SS / total_SS =  20.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

##Interpretation des classes “Risques”

#Classes

```
Ris <- cbind.data.frame(VillesBrut$Ville,Risques)

CRIS1 <- vector()
CRIS2  <- vector()
CRIS3  <- vector()

a<-1
b<-1
c<-1

for (i in 1:100) {

  if (KM3Ris $cluster[i] == 1){
    CRIS1[a] <- as.matrix(Ris[i,1])
    a <- a+1
  }

  if (KM3Ris $cluster[i] == 2){
   CRIS2[b] <- as.matrix(Ris[i,1])
   b <- b+1
  }

  if (KM3Ris $cluster[i] == 3){
    CRIS3[c] <- as.matrix(Ris[i,1])
    c <- c+1
  }
}
```

Nous aurons :

```
print(CRIS1)
```

```
##  [1] "Agen"                "AixEnProvence"       "Ajaccio"
##  [4] "Albi"                "Angers"              "Annecy"
##  [7] "Antibes"             "Bastia"              "Bayonne"
## [10] "Blois"               "BoulogneBillancourt" "Bourges"
## [13] "Caen"                "Cannes"              "Chambery"
## [16] "Chartres"            "Cholet"              "Compiegne"
## [19] "Dijon"               "Epinal"              "Gap"
## [22] "Grenoble"            "La Rochelle"         "La RocheSurYon"
## [25] "Laval"               "LeMans"              "Montauban"
## [28] "Montlucon"           "Nantes"              "NeuillySurSeine"
## [31] "Nice"                "Nimes"               "Niort"
## [34] "Orleans"             "Paris"               "Perigueux"
## [37] "Poitiers"            "Quimper"             "Rennes"
## [40] "RueilMalmaison"      "SaintGermainEnLaye"  "Tarbes"
## [43] "Toulon"              "Tours"               "Vannes"
## [46] "Versailles"          "Vichy"
```

puis,

```
print(CRIS2)
```

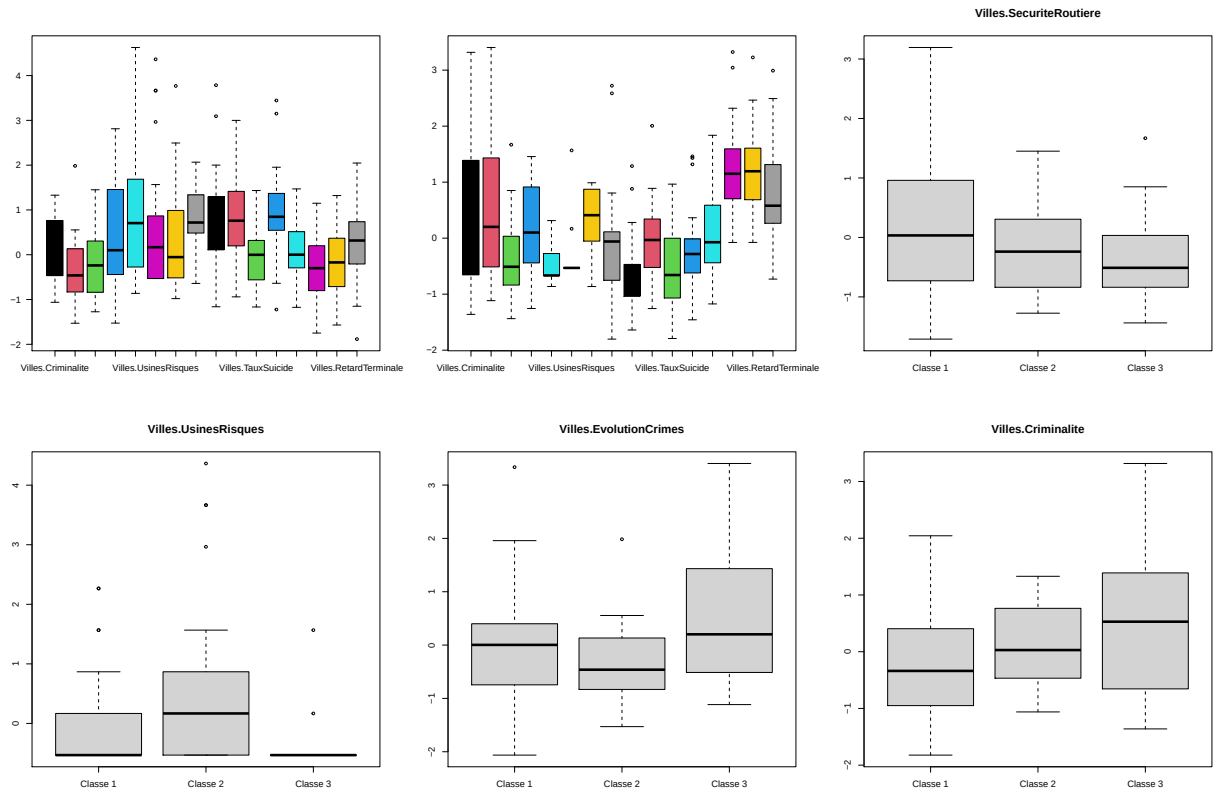
```
##  [1] "Amiens"              "Beauvais"            "Belfort"
##  [4] "Bordeaux"            "Brest"               "Calais"
##  [7] "CharlevilleMezieres" "Colmar"              "Dunkerque"
```

```
## [10] "LeHavre"              "Lille"              "Lyon"
## [13] "Marseille"            "Melun"              "Metz"
## [16] "Mulhouse"             "Nancy"              "Reims"
## [19] "Rouen"                "SaintQuentin"       "Sete"
## [22] "Strasbourg"           "Toulouse"           "Troyes"
## [25] "Valenciennes"
```

et ,

```
print(CRIS3)
```

```
##  [1] "Angouleme"         "Arles"             "Auxerre"          "Avignon"
##  [5] "Besancon"          "Beziers"           "BourgEnBresse"    "BriveLaGaillarde"
##  [9] "Carcassonne"       "Castres"           "ChalonSurSaone"   "ClermontFerrand"
## [13] "CorbeilEssonnes"   "Creteil"           "Evry"             "Limoges"
## [17] "Montpellier"       "Nevers"            "Pau"              "Perpignan"
## [21] "SaintBrieuc"       "SaintDenis"        "SaintEtienne"     "SaintMalo"
## [25] "SaintNazaire"      "Sarcelles"         "Valence"          "Villeurbanne"
```



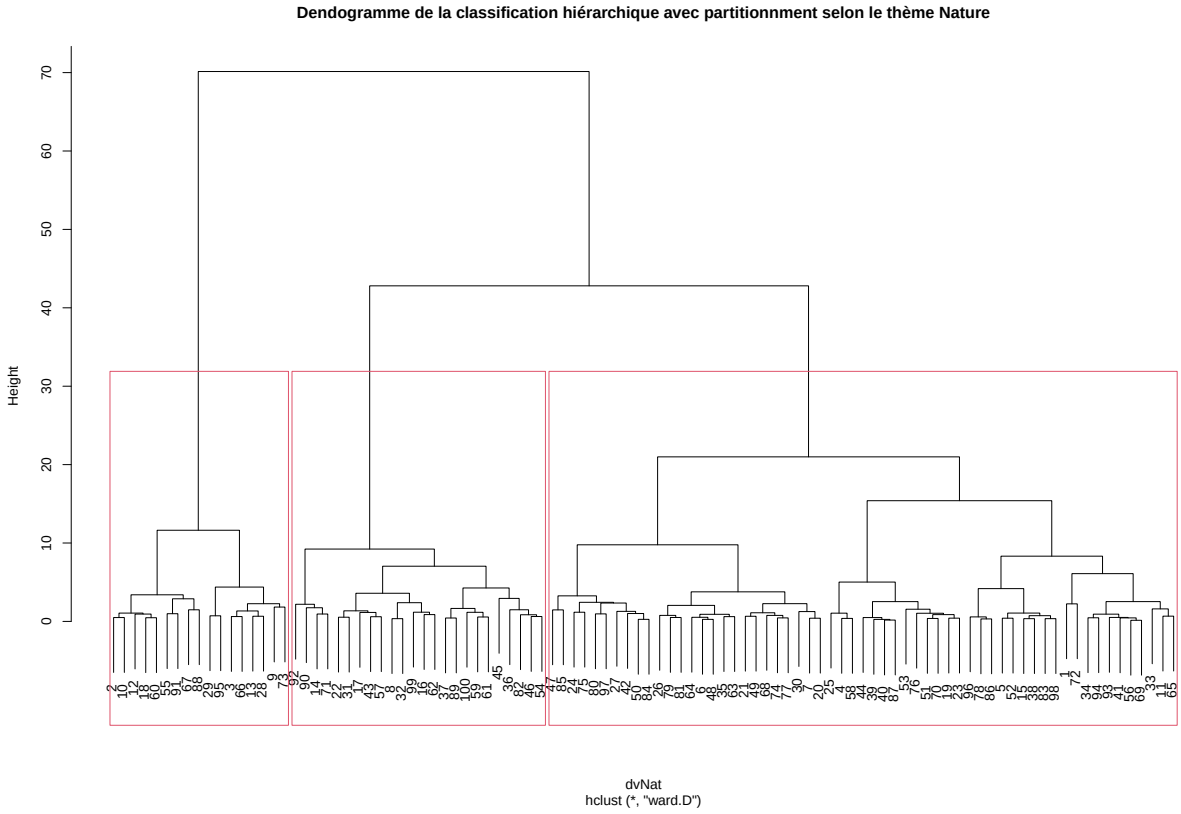
Interprétation sur les classes :

- La classe 1 contient les villes où le taux de criminalité est le moins élevé, mais également un taux de pollution élevé.
- La classe 2 contient les villes avec le taux de criminalité le plus élevé, principalement des grandes métropoles (Marseille, Toulouse, Lyon)
- La classe 3 contient les villes avec le taux le plus faible d'usines dangereuses.

Remarque : Paris n'est pas dans la classe 2 (avec le taux de criminalité le plus élévé) néanmoins, on peut justifier sa présence dans la classe 1 par son taux de pollution élevé (Paris est une ville qui subit souvent des

pic de pollution)

##Nature




```
#Partition en 3 classes

P3Nat = cutree(tree = CAHVNat, k=3)

R2_P3Nat= cbind(rep(0 , ncol(Nature)))

for (i in cbind(1:ncol(Nature))) {

  R2_P3Nat[i] =
    summary(lm(Nature[,i]~as.factor(P3Nat)))$r.squared
}

row.names(R2_P3Nat) = colnames(Nature)

IC3Nat = data.frame(model.matrix(~as.factor(P3Nat)-1))

mIC3Nat = as.matrix(IC3Nat)

mNat = as.matrix(Nature)

CentresC3Nat = solve(t(mIC3Nat) %*% mIC3Nat) %*% t(mIC3Nat)%*% mNat

KM3Nat = kmeans(Nature, CentresC3Nat)
```

```r
print(KM3Nat)
```

```
## K-means clustering with 3 clusters of sizes 58, 17, 25
##
## Cluster means:
##    Villes.Mer Villes.Ski Villes.Soleil Villes.Pluie Villes.Temperature
## 1  0.05850547 -0.6889649    -0.4693038    0.4143170         -0.3010360
## 2  0.97807385  0.3242188     1.9964480   -1.9369079          1.7791891
## 3 -0.80082292  1.3779297    -0.2687999    0.3558819         -0.5114451
##   Villes.MarcheAPied
## 1         -0.3678834
## 2          0.2126292
## 3          0.7089016
##
## Clustering vector:
##   [1] 1 2 2 1 1 1 1 3 2 2 1 2 2 3 1 3 3 2 1 1 1 3 1 1 1 1 1 2 2 1 3 3 1 1 1 3 3
##  [38] 1 1 1 1 1 3 1 3 3 1 1 1 1 1 1 1 3 2 1 3 1 3 2 3 3 1 1 1 2 2 1 1 1 3 3 2 1
##  [75] 1 1 1 1 1 1 1 1 3 1 1 1 1 1 2 3 3 2 3 1 1 2 1 1 1 3 3
##
## Within cluster sum of squares by cluster:
## [1] 136.20532  43.13738  70.55525
##  (between_SS / total_SS =  58.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
#Partition en 2 classes

P2Nat = cutree(tree = CAHVNat, k=2)

R2_P2Nat= cbind(rep(0 , ncol(Nature)))

for (i in cbind(1:ncol(Nature))) {

  R2_P2Nat[i] =
    summary(lm(Nature[,i]~as.factor(P2Nat)))$r.squared
}

row.names(R2_P2Nat) = colnames(Nature)

IC2Nat = data.frame(model.matrix(~as.factor(P2Nat)-1))

mIC2Nat = as.matrix(IC2Nat)

mNat = as.matrix(Nature)

CentresC2Nat = solve(t(mIC2Nat) %*% mIC2Nat) %*% t(mIC2Nat)%*% mNat

KM2Nat = kmeans(Nature, CentresC2Nat)

print(KM2Nat)
```

```
## K-means clustering with 2 clusters of sizes 83, 17
##
## Cluster means:
##   Villes.Mer  Villes.Ski Villes.Soleil Villes.Pluie Villes.Temperature
## 1 -0.2003284 -0.06640625     -0.408911    0.3967161         -0.3644122
## 2  0.9780739  0.32421876      1.996448   -1.9369079          1.7791891
##   Villes.MarcheAPied
## 1        -0.04355056
## 2         0.21262919
##
## Clustering vector:
##   [1] 1 2 2 1 1 1 1 1 2 2 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 1 1 2 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 1 1 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 316.08450  43.13738
##  (between_SS / total_SS =  40.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

##Interpretation des classes “Nature”

#Classes

```
Nat <- cbind.data.frame(VillesBrut$Ville,Nature)

CNat1 <- vector()
Cnat2  <- vector()
CNat3  <- vector()

a<-1
b<-1
c<-1

for (i in 1:100) {

  if (KM3Nat$cluster[i] == 1){
    CNat1 [a] <- as.matrix(Nat[i,1])
    a <- a+1
  }

  if (KM3Nat$cluster[i] == 2){
   Cnat2[b] <- as.matrix(Nat[i,1])
   b <- b+1
  }

  if (KM3Nat$cluster[i] == 3){
    CNat3 [c] <- as.matrix(Nat[i,1])
    c <- c+1
  }
}
```

Nous avons donc :

```
print(CNat1)
```

```
##  [1] "Agen"                "Albi"                "Amiens"
##  [4] "Angers"              "Angouleme"           "Auxerre"
##  [7] "Beauvais"            "Blois"               "Bordeaux"
## [10] "BoulogneBillancourt" "Bourges"             "Brest"
## [13] "BriveLaGaillarde"    "Caen"                "Calais"
## [16] "Castres"             "CharlevilleMezieres" "Chartres"
## [19] "Cholet"              "Compiegne"           "CorbeilEssonnes"
## [22] "Creteil"             "Dijon"               "Dunkerque"
## [25] "Evry"                "La Rochelle"         "La RocheSurYon"
## [28] "Laval"               "LeHavre"             "LeMans"
## [31] "Lille"               "Limoges"             "Melun"
## [34] "Montauban"           "Nantes"              "NeuillySurSeine"
## [37] "Nevers"              "Niort"               "Orleans"
## [40] "Paris"               "Poitiers"            "Quimper"
## [43] "Reims"               "Rennes"              "Rouen"
## [46] "RueilMalmaison"      "SaintBrieuc"         "SaintDenis"
## [49] "SaintGermainEnLaye"  "SaintMalo"           "SaintNazaire"
## [52] "SaintQuentin"        "Sarcelles"           "Tours"
## [55] "Troyes"              "Valenciennes"        "Vannes"
## [58] "Versailles"
```

puis,

```
print(Cnat2)
```

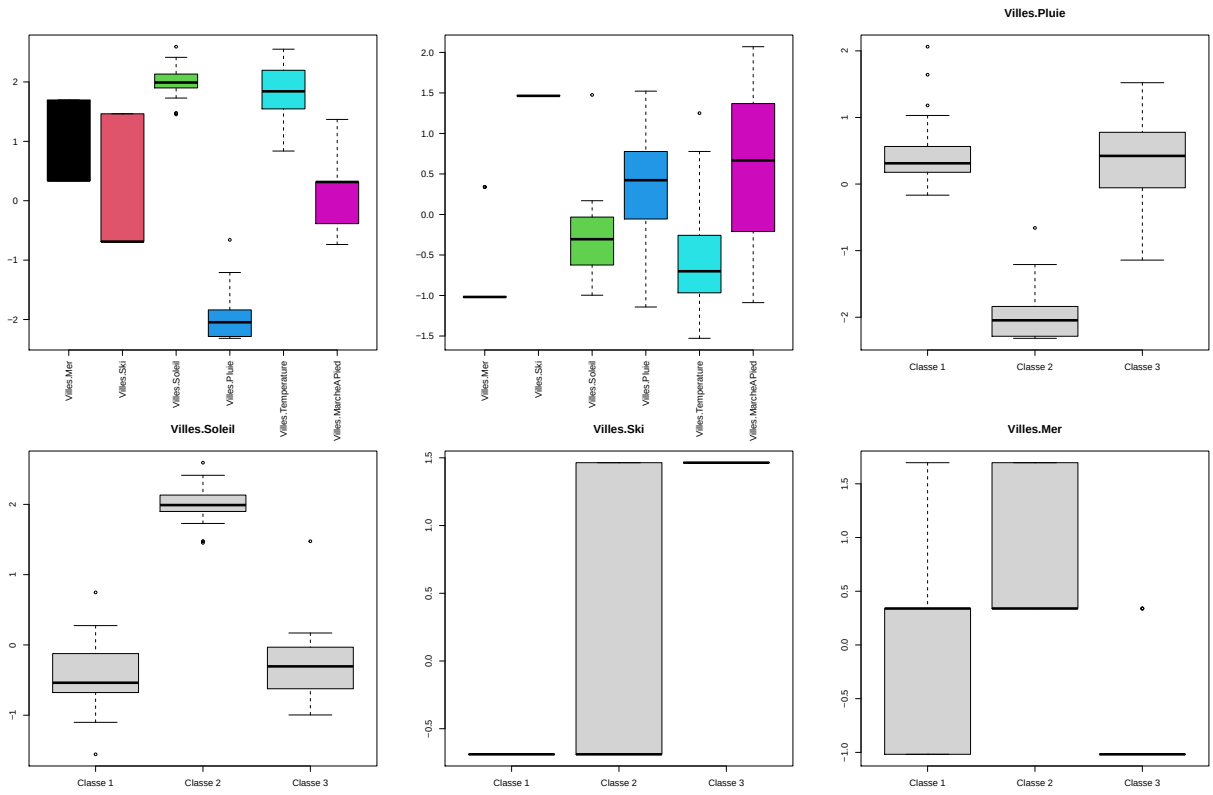
```
##  [1] "AixEnProvence" "Ajaccio"       "Antibes"       "Arles"
##  [5] "Avignon"       "Bastia"        "Beziers"       "Cannes"
##  [9] "Carcassonne"   "Marseille"     "Montpellier"   "Nice"
## [13] "Nimes"         "Perpignan"     "Sete"          "Toulon"
## [17] "Valence"
```

et,

```
print(CNat3)
```

```
##  [1] "Annecy"          "Bayonne"         "Belfort"         "Besancon"
##  [5] "BourgEnBresse"   "ChalonSurSaone"  "Chambery"        "ClermontFerrand"
##  [9] "Colmar"          "Epinal"          "Gap"             "Grenoble"
## [13] "Lyon"            "Metz"            "Montlucon"       "Mulhouse"
## [17] "Nancy"           "Pau"             "Perigueux"       "SaintEtienne"
## [21] "Strasbourg"      "Tarbes"          "Toulouse"        "Vichy"
## [25] "Villeurbanne"
```

#Interprétation sur les classes :



#Interprétation

- La classe 1 représente les villes qui se situent plutôt dans des régions froides, qui ont un climat océanique dégradé (Paris, Reims, Orléans. . .), un climat océanique aquitanien (Bordeaux, Limoges. . .), et de climat océanique nord-ouest (Tours, Nantes, Rennes. . .)
- La deuxième classe regroupe des villes avec un climat méditerranéen, principalement situé au sud de la France avec taux d'ensoleillement élévé (et de pluie faible), pour la plupart étant proche des stations balnéaires situé sur la côte méditerranéenne (Montpellier, Nice, Valence, Marseille . . .)
- La 3ème classe se compose de villes avec un climat océanique semi continental (Tarbes, Grenoble, Lyon. . .)

##Culture

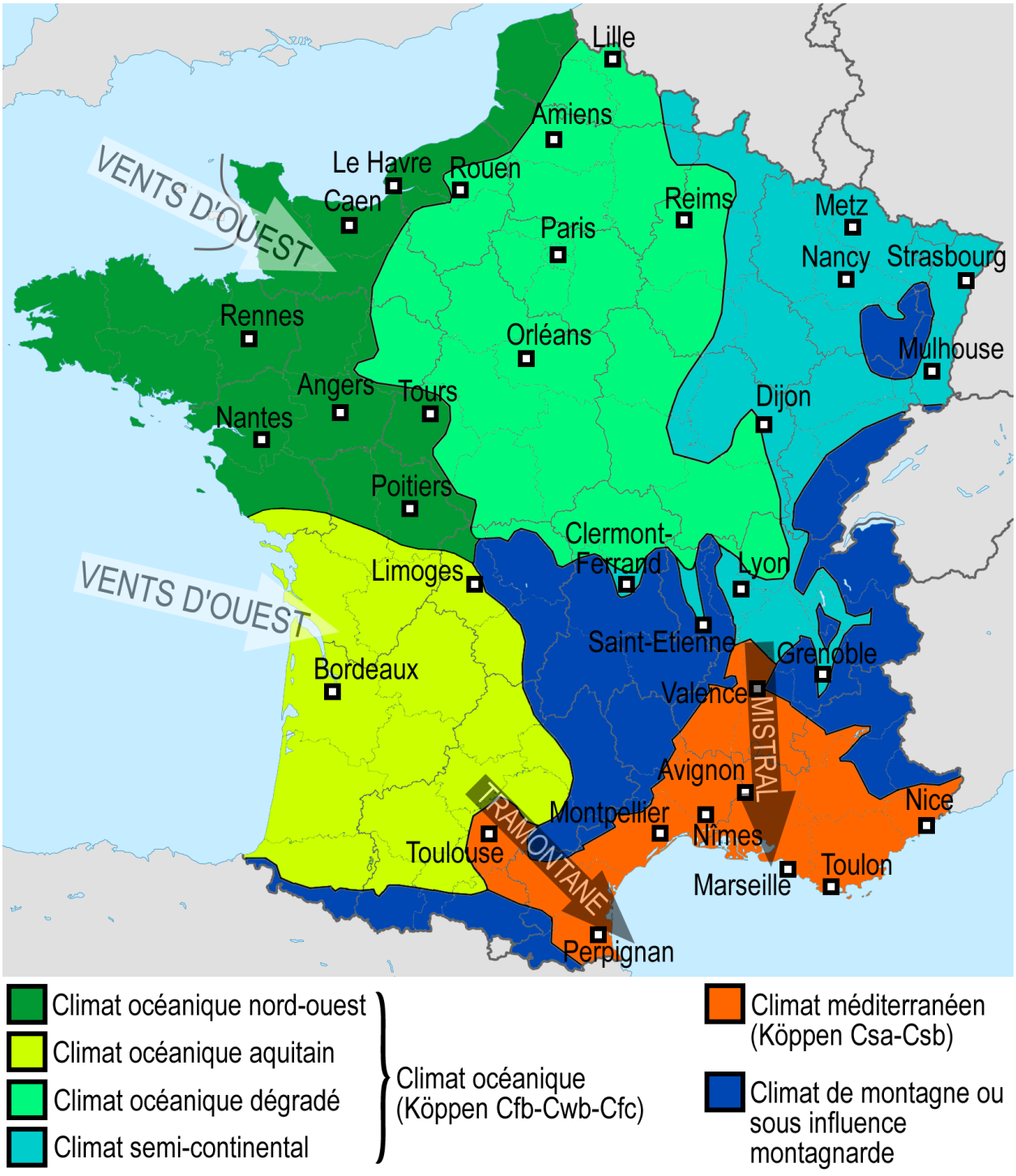
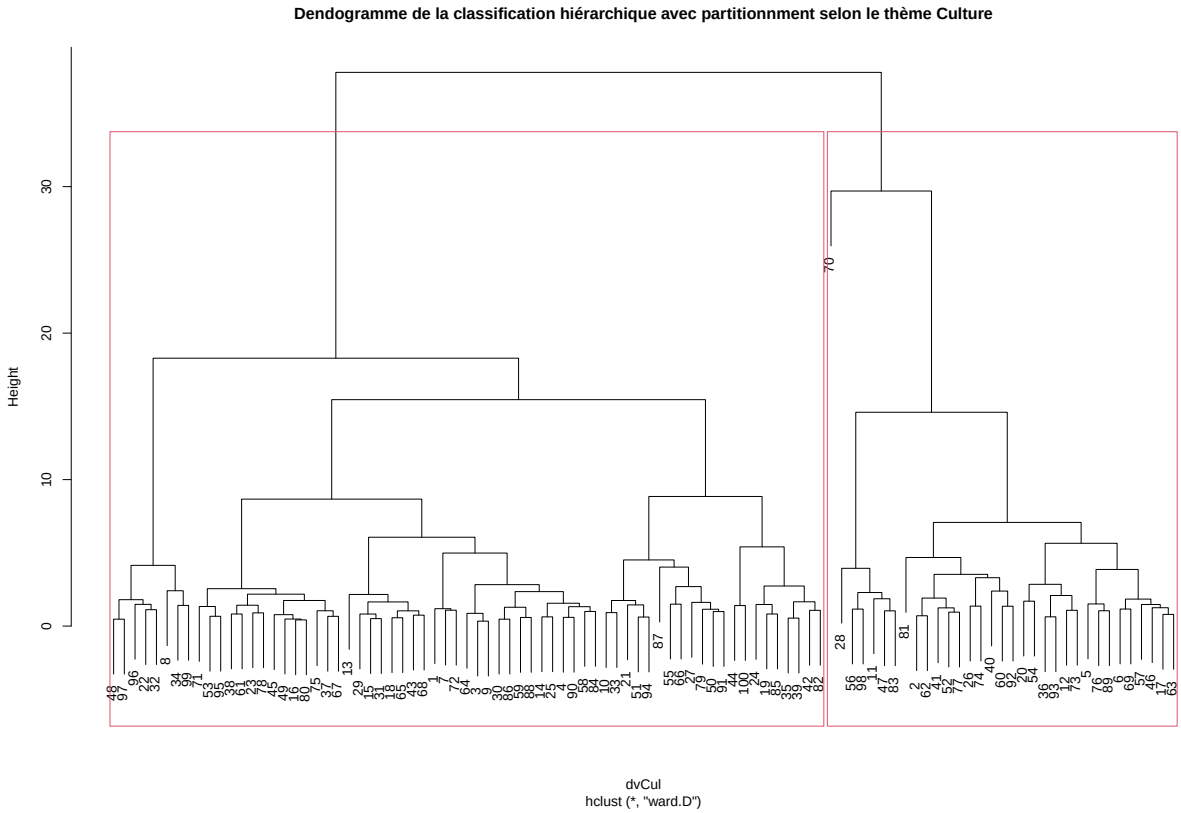


Figure 1: Climat de France , source : Wikipédia




```
#Partition en 3 classes Culture :


P3Cul = cutree(tree = CAHVCul, k=3)

R2_P3Cul = cbind(rep(0 , ncol(Culture)))

for (i in cbind(1:ncol(Culture))) {

  R2_P3Cul[i] =
    summary(lm(Culture[,i]~as.factor(P3Cul)))$r.squared
}

row.names(R2_P3Cul) = colnames(Culture)
colnames(R2_P3Cul)<- "R2 des variables du thème 'Nature' avec partition 3 classes"


IC3Cul = data.frame(model.matrix(~as.factor(P3Cul)-1))

mIC3Cul = as.matrix(IC3Cul)

mCul = as.matrix(Culture)

CentresC3Cul = solve(t(mIC3Cul) %*% mIC3Cul) %*% t(mIC3Cul)%*% mCul

KM3Cul = kmeans(Culture, CentresC3Cul)
```

```r
print(KM3Cul)
```

```
## K-means clustering with 3 clusters of sizes 67, 32, 1
##
## Cluster means:
##   Villes.Musees Villes.Cinema Villes.MonumHistoriques Villes.PretLivres
## 1    -0.1981879    -0.4539647              -0.2522720         0.1201805
## 2     0.1232856     0.9332981               0.2440957        -0.2413606
## 3     9.3334479     0.5500943               9.0911614        -0.3285522
##   Villes.RestaurDistingues Villes.Presse Villes.Etudiants
## 1             -0.142220208    0.05078316       -0.4838164
## 2             -0.009293323   -0.20214565        0.9998996
## 3              9.826140262    3.06618911        0.4189136
##
## Clustering vector:
##   [1] 1 2 1 1 2 2 1 1 1 1 2 2 1 1 1 1 2 1 1 2 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 2 1
##  [38] 1 1 2 2 1 1 1 1 2 2 1 1 1 1 2 1 2 1 2 2 1 1 2 1 2 2 1 1 1 1 1 1 3 2 1 2 2
##  [75] 1 2 2 1 1 1 2 1 2 1 1 1 1 1 2 1 1 2 2 1 1 1 1 2 1 1
##
## Within cluster sum of squares by cluster:
## [1] 203.8214 121.6342   0.0000
##  (between_SS / total_SS =  53.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
#Partition en 2 classes Culture :

P2Cul = cutree(tree = CAHVCul, k=2)
R2_P2Cul = cbind(rep(0 , ncol(Culture)))

for (i in cbind(1:ncol(Culture))) {

  R2_P2Cul[i] =
    summary(lm(Culture[,i]~as.factor(P2Cul)))$r.squared
}

row.names(R2_P2Cul) = colnames(Culture)


IC2Cul = data.frame(model.matrix(~as.factor(P2Cul)-1))

mIC2Cul = as.matrix(IC2Cul)

mCul = as.matrix(Culture)

CentresC2Cul = solve(t(mIC2Cul) %*% mIC2Cul) %*% t(mIC2Cul)%*% mCul

KM2Cul = kmeans(Culture, CentresC2Cul)

print(KM2Cul)
```

```
## K-means clustering with 2 clusters of sizes 66, 34
##
## Cluster means:
##   Villes.Musees Villes.Cinema Villes.MonumHistoriques Villes.PretLivres
## 1    -0.2151455    -0.4613638              -0.2593671         0.1459599
## 2     0.4176354     0.8955886               0.5034772        -0.2833340
##   Villes.RestaurDistingues Villes.Presse Villes.Etudiants
## 1               -0.1472329    0.06987067       -0.4828609
## 2                0.2858051   -0.13563130        0.9373182
##
## Clustering vector:
##   [1] 1 2 1 1 2 2 1 1 1 1 2 2 1 1 1 1 2 1 1 2 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 2 1
##  [38] 1 1 2 2 1 1 1 1 2 2 1 1 1 1 2 1 2 2 2 2 1 1 2 1 2 2 1 1 1 1 1 1 2 2 1 2 2
##  [75] 1 2 2 1 1 1 2 1 2 1 1 1 1 1 2 1 1 2 2 1 1 1 1 2 1 1
##
## Within cluster sum of squares by cluster:
## [1] 197.4201 390.7417
##  (between_SS / total_SS =  16.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

- Interpretation des classes “Culture”

#Classes

```
Cult <- cbind.data.frame(VillesBrut$Ville,Culture)

Ccult1 <- vector()
Ccult2  <- vector()
Ccult3  <- vector()

a<-1
b<-1
c<-1

for (i in 1:100) {

  if (KM3Cul  $cluster[i] == 1){
    Ccult1  [a] <- as.matrix(Cult[i,1])
    a <- a+1
  }

  if (KM3Cul  $cluster[i] == 2){
  Ccult2 [b] <- as.matrix(Cult[i,1])
   b <- b+1
  }

  if (KM3Cul $cluster[i] == 3){
    Ccult3 [c] <- as.matrix(Cult[i,1])
    c <- c+1
  }
}
```

Nous avons donc :

```
print(Ccult1 )
```

```
##  [1] "Agen"                "Ajaccio"             "Albi"
##  [4] "Angouleme"           "Annecy"              "Antibes"
##  [7] "Arles"               "Bastia"              "Bayonne"
## [10] "Beauvais"            "Belfort"             "Beziers"
## [13] "Blois"               "BoulogneBillancourt" "BourgEnBresse"
## [16] "Bourges"             "Brest"               "BriveLaGaillarde"
## [19] "Calais"              "Carcassonne"         "Castres"
## [22] "ChalonSurSaone"      "Chambery"            "CharlevilleMezieres"
## [25] "Chartres"            "Cholet"              "Colmar"
## [28] "Compiegne"           "CorbeilEssonnes"     "Dunkerque"
## [31] "Epinal"              "Evry"                "Gap"
## [34] "La RocheSurYon"      "Laval"               "LeHavre"
## [37] "LeMans"              "Limoges"             "Marseille"
## [40] "Montauban"           "Montlucon"           "Mulhouse"
## [43] "NeuillySurSeine"     "Nevers"              "Nice"
## [46] "Nimes"               "Niort"               "Orleans"
## [49] "Perigueux"           "Quimper"             "Rouen"
## [52] "RueilMalmaison"      "SaintBrieuc"         "SaintEtienne"
## [55] "SaintMalo"           "SaintNazaire"        "SaintQuentin"
## [58] "Sarcelles"           "Sete"                "Tarbes"
## [61] "Toulon"              "Troyes"              "Valence"
## [64] "Valenciennes"        "Vannes"              "Vichy"
## [67] "Villeurbanne"
```

puis,

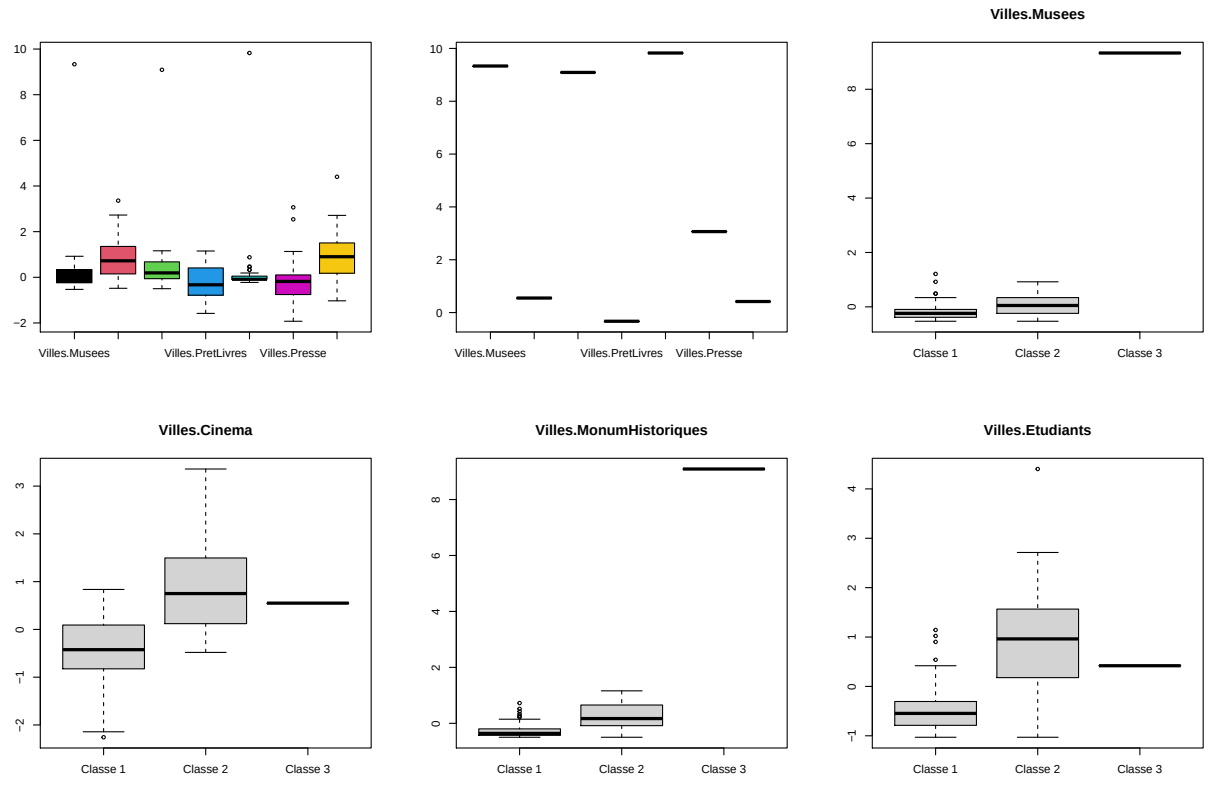
```
print(Ccult2)
```

```
##  [1] "AixEnProvence"      "Amiens"             "Angers"
##  [4] "Auxerre"            "Avignon"            "Besancon"
##  [7] "Bordeaux"           "Caen"               "Cannes"
## [10] "ClermontFerrand"    "Creteil"            "Dijon"
## [13] "Grenoble"           "La Rochelle"        "Lille"
## [16] "Lyon"               "Melun"              "Metz"
## [19] "Montpellier"        "Nancy"              "Nantes"
## [22] "Pau"                "Perpignan"          "Poitiers"
## [25] "Reims"              "Rennes"             "SaintDenis"
## [28] "SaintGermainEnLaye" "Strasbourg"         "Toulouse"
## [31] "Tours"              "Versailles"
```

et,

```
print(Ccult3)
```

```
## [1] "Paris"
```



Interprétation sur les classes :

- La classe 1 se compose de villes plutot faibles sur le plan culturel (Albi, Limoges, Mulhouse...).
- La classe 2 se compose des villes moyennement riche culturellement, mais aussi des villes étudiantes (Toulouse, Lyon, Bordeaux, Montpellier, Grenoble).
- La classe 3 est composée de Paris qui posède le plus de monuments historiques (Tour Eiffel, Basilique du Sacré-Coeur, Cathédrale Notre Dame) et de musées (Musée du Louvre, Musée Pompidou) sur le territoire, une proportion d'étudiants assez conséquent via ses nombreuses universités.

Néanmoins, on remarque grâce aux boxplots qu'il est préférable de garder une partition en 2 classes. Le treillis de Galois nous le confirmera bien après.

#Treillis de Galois

On va créer le tableau logique disjonctif complet formé par les quatre variables de classe : Economie, Culture, Risque et nature sur Galicia

```
vect0<- rep(0,100) #Sur Galicia, un problème survient si la première colonne n'est pas composé de 0

KMClasses <- data.frame(mIC3Eco,mIC3Nat,mIC3Nat,mIC3Cul)


#Exportation du tableau en format texte (.txt) pour l'importation sur Galicia


write.table(cbind(vect0, KMClasses), "tabdij_galicia1.txt", sep="\t", row.names = FALSE, col.names = FA
```

Treillis de Galois.

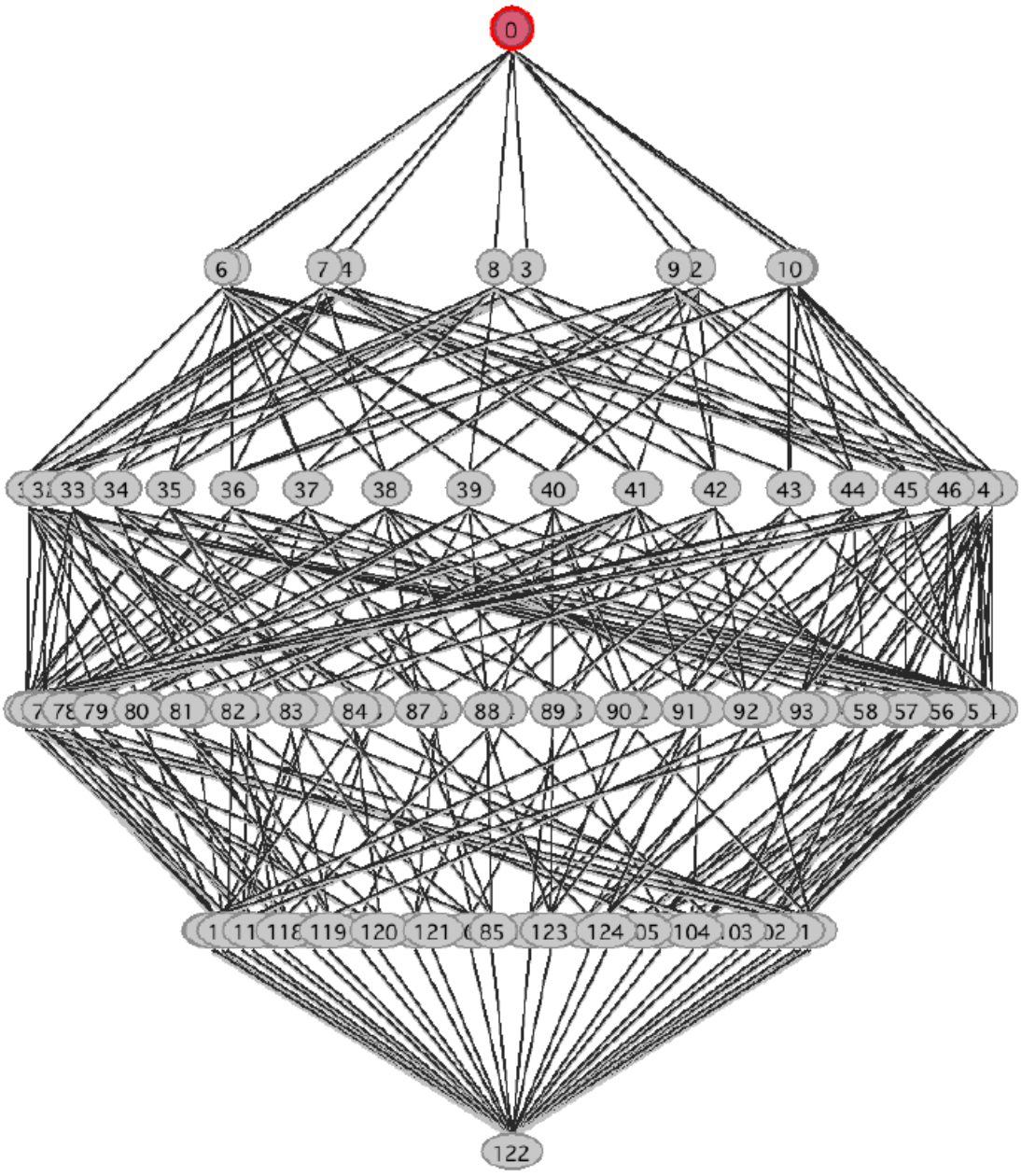


Figure 2: Treillis de Galois

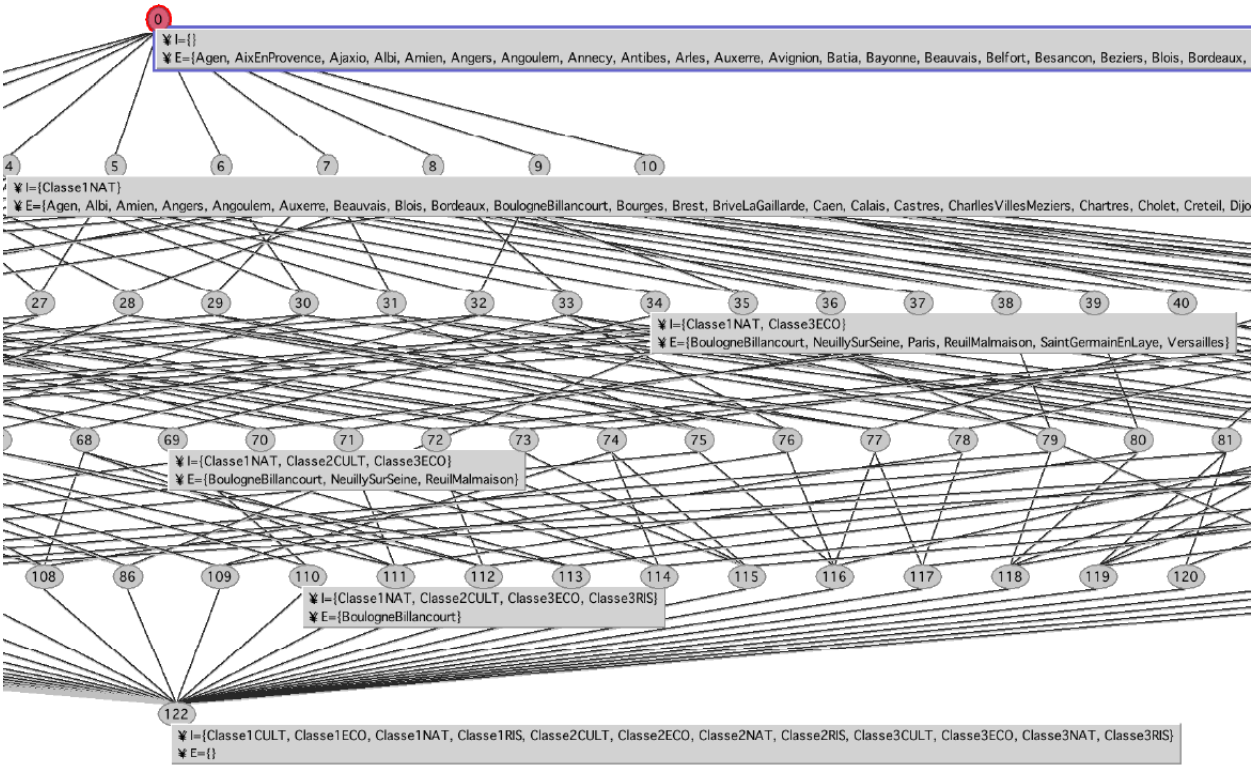


Figure 3: Visualisation des classes dans le treillis de Galois

Tout d'abord, On a dans le noeud 122, les 12 classes. Ce noeud ne contient logiquement aucune ville (aucune ville est dans toutes les classes à la fois) .

Lorsqu'on remonte vers le noeud 110, on s'aperçoit que la ville Boulogne-Billancourt appartient à la fois à la classe 1 du thème "Nature", à la classe 2 "Culture", à la classe 3 "Economie" et finalment à la classe 3 du thème "Risque".

Au noeud 69, on trouve alors les villes des Haut-de-Seine (92) qui sont dans les classes : Classe 1 "Nature", classe 2 "Culture", classe 3 "Economie"

Par ensuite, on observe au noeud 34,que les villes sont ceux du noeud précédent, avec l'ajout néanmoins des villes de Paris de Versailles et Saint Germain En Laye ces villes appartiennent à la fois à la première classe "Nature" et à la troisièle classe "Economie", ce sont donc les villes au climat froid et situé dans le département des Haut-de-Seine (à l'ouest de Paris). Ce noeud est ainsi plus précis que le précédent

Finalement, on s'aperçoit que le nombre de classe avant la génération de notre treillis de Galois était de 12 (3 classes x 4 thèmes).

Le treillis de Galois nous a permis de réduire de 12 à 10 classes, ce qui n'est guère pas étonnant par rapport à ce qu'on observait dans les boxplots notamment celui du thème "Culture" et "Economie" qui indiquaient une utilisation plus judicieuse de K=2 classes à la place de 3 classes. En effet, le treillis a regroupé les classe 3 Economie correspondant aux villes aisées de la banlieue ouest parisienne, et la classe 3 Culture qui correspond uniquement à Paris. Le nombre de 3 classes pour les thèmes "Nature" et "Risque" reste judicieuse et cela est confirmé par le treillis

##IV - Classification de variables

#a) Procédez à une classification des indicateurs à l'aide du package ClustOfVar.

" Les fonctions du package ClustOfVar permettant de faire de la classification hiérarchique des variables

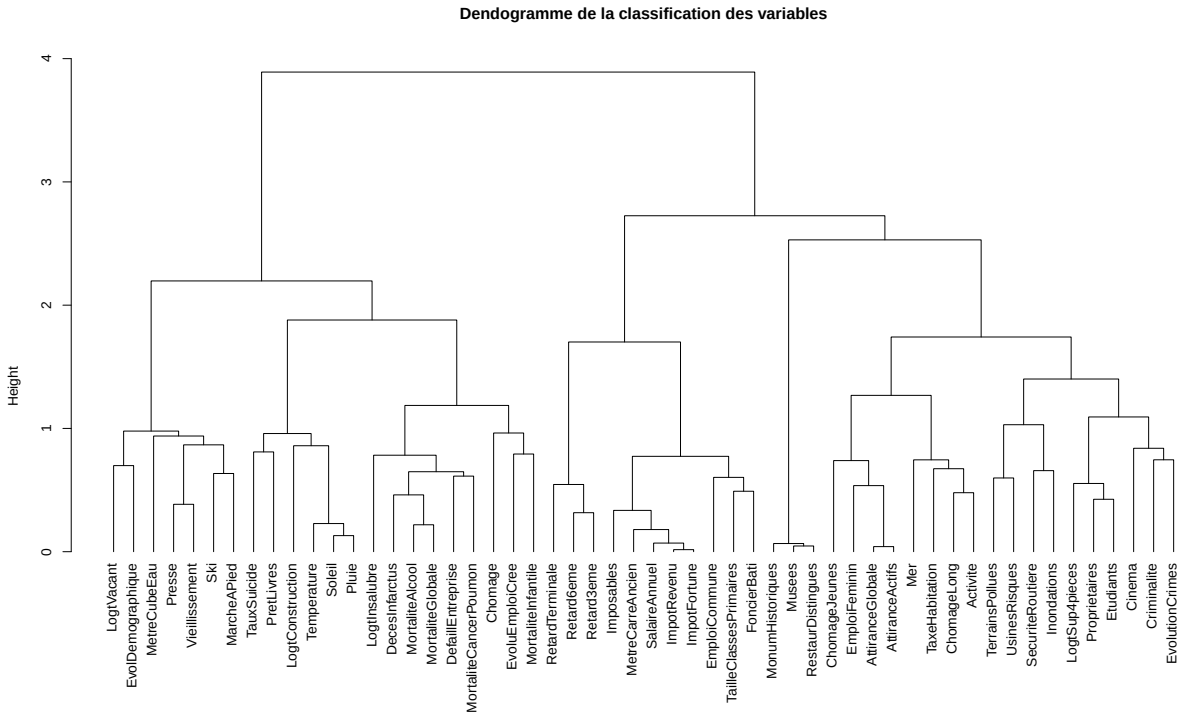
La fonction hclustvar construit la hiérarchie. – La fonction plot permet d'obtenir le dendrogramme de cette hiérarchie. – La fonction cutreevar coupe la hiérarchie et extrait une partition en K classes où le nombre K de classes est donné par l'utilisateur. – La fonction rect.hclust dessine des rectangles autour des branches du dendrogramme permettant de mettre en évidence les K classes obtenues. " [1]

Méthode ClustOfVar :

Source : [1] *La méthode ClustOfVar pour mesurer les conditions de vie 'a l'´echelle communale Vanessa Kuentz-Simonet 1, Amaury Labenne 1,2 & Tina Rambonilaza 1*

##Utilisation de la classification hiérarchique

```
## Warning in data(Villes): data set 'Villes' not found
```



Via le graphique des stabilités des partitions, on remarque une petite baisse de la stabilité au cluster = 7 . Ainsi, on choisira de faire une partition avec K=7 classes dans notre arbre de classification hiérarchique.

La description des k = 7 classes de cette partition est obtenue via la fonction summary

```
summary(partCHVAR)
```

```
##
## Call:
## cutreevar(obj = CHVar, k = 7)
##
```

```
##
##
## Data:
##    number of observations:  100
##    number of variables:  54
##    number of clusters:  7
##
## Cluster  1 :
##                        squared loading correlation
## MortaliteGlobale                 0.848       -0.92
## MortaliteAlcool                  0.658       -0.81
## DecesInfarctus                   0.518       -0.72
## MortaliteCancerPoumon            0.493       -0.70
## DefaillEntreprise                0.448       -0.67
## LogtInsalubre                    0.278       -0.53
## EvoluEmploiCree                  0.044        0.21
## MortaliteInfantile               0.031       -0.18
## Chomage                          0.015        0.12
##
##
## Cluster  2 :
##                   squared loading correlation
## AttiranceActifs              0.74        0.86
## AttiranceGlobale             0.74        0.86
## Activite                     0.44        0.67
## EmploiFeminin                0.44        0.67
## ChomageLong                  0.43       -0.66
## ChomageJeunes                0.31       -0.56
## Mer                          0.25       -0.50
## TaxeHabitation               0.17       -0.42
##
##
## Cluster  3 :
##                         squared loading correlation
## SalaireAnnuel                      0.92        0.96
## MetreCarreAncien                   0.88        0.94
## ImpotRevenu                        0.87        0.93
## ImpotFortune                       0.82        0.90
## Imposables                         0.73        0.85
## TailleClassesPrimaires             0.46        0.68
## FoncierBati                        0.43       -0.66
## EmploiCommune                      0.25       -0.50
## RetardTerminale                    0.25       -0.50
## Retard6eme                         0.19       -0.43
## Retard3eme                         0.18       -0.42
##
##
## Cluster  4 :
##                   squared loading correlation
## LogtSup4pieces             0.5965       0.772
## Proprietaires              0.4783       0.692
## Etudiants                  0.4627      -0.680
## TerrainsPollues            0.2981      -0.546
## Criminalite                0.2942      -0.542
```

```
## Inondations               0.1807      -0.425
## Cinema                    0.1790      -0.423
## SecuriteRoutiere          0.1353      -0.368
## UsinesRisques             0.0252      -0.159
## EvolutionCrimes           0.0068       0.082
##
##
## Cluster  5 :
##                   squared loading correlation
## Soleil                       0.87       -0.93
## Pluie                        0.83        0.91
## Temperature                  0.80       -0.90
## PretLivres                   0.20        0.45
## LogtConstruction             0.17       -0.42
## TauxSuicide                  0.14        0.37
##
##
## Cluster  6 :
##                    squared loading correlation
## Vieillissement               0.628       -0.79
## MarcheAPied                  0.506       -0.71
## Presse                       0.496       -0.70
## LogtVacant                   0.296       -0.54
## Ski                          0.278       -0.53
## EvolDemographique            0.245        0.49
## MetreCubeEau                 0.051        0.22
##
##
## Cluster  7 :
##                    squared loading correlation
## Musees                        0.97        0.98
## RestaurDistingues             0.96        0.98
## MonumHistoriques              0.96        0.98
##
##
## Gain in cohesion (in %):  33.19
```

b) Chacune des classes obtenues dans la partition que vous retenez finalement semble-t-elle liée à un thème particulier, au sens où ses variables proviennent majoritairement de ce thème ? Dans quelle(s) classe(s) se retrouvent la majorité des variables de chacun des thèmes?

Les 7 classes obtenues sont assez cohérentes : par exemple le "Cluster 1" regroupe les différentes mortalités comme la mortalité globale et infantile, la mortalité dû à l'alcool, à l'infarctus et au cancer. Elle contient également le logement insalubre qui peut occasioner des risques sur la santé. Nous avons également l'évolution de l'emploi et le chomage qui contiennent des facteurs de risque. Le Custer 1 est ainsi associé au thème "Risque". Nous avons ensuite le Cluster 5 qui regroupe les variables Soleil, Pluie et Temperature, qui sont présents également dans le thème "Nature". Néanmoins, ce cluster ne contient pas la variable "Ski" qui est quant à lui présent dans le Cluster 6, cluster qui peut représenter les Vacances d'une certaine classe sociale et d'âge (classe moyenne supérieur et assez vieille). Le Cluster 7 partage les mêmes variables que le thème "Culture". Au vu des variables contenues (Musées, Restaurants distingués et monuments historiques), ce cluster peut être associé à la ville parisienne.

Finalement, on remarque la présence d'un petit groupe de variables quasi-présent à la fois dans les clusters obtenus et aux thèmes. Cependant, on peut remarquer que des variables présentes dans un certain thème (exemple : Suicide du thème "Risque") se retrouve dans le cluster qui partage le plus de variables communes

avec le thème “Nature”. Ainsi, cette partition regroupe les variables selon d’autres critères que ceux des thèmes que nous avons traités précedemment : le suicide peut être lié au climat météorologique (cf : *The Effect of Season and Weather on Suicide Rates in the Elderly in B. C., Stephen Marion MD, Revue canadienne de santé publique*)

#Utilisation de la classification de type k-means

Un algorithme d’optimisation du partitionnement (de type k-means).

Classification de variables kmeans à partir de la fonction kmeansvar

```
partKMVar <- kmeansvar(Villes,init = 3, nstart = 200)
```

Le paramètre init de notre fonction kmeansvar correspond au nombre de classes choisies et nstart quant à lui correspond au nombre d’initialisations.

```
table(partCHVAR$cluster,partKMVar$cluster)
```

```
##
##      1  2  3
##   1  5  2  2
##   2  2  0  6
##   3 10  0  1
##   4  0  4  6
##   5  0  0  6
##   6  1  2  4
##   7  0  3  0
```

```
rand(partCHVAR$cluster,partKMVar$cluster,adj=FALSE)
```

```
## [1] 0.6694619
```

On a ensuite comparé les deux partitions (celle via classification hiérarchique et celle par kmeans) via le tableau de contingence. On remarque que l’indice de Rand n’est pas égale à 1 donc les deux partitions (celle via classification hiérarchique et celle par kmeans) ne sont totalement pas identiques

Source : *Classification de variables et analyse multivariée de données mixtes issues d’une étude BCI, Jérôme Saracco1, Marie Chavent, Liliana Garcia, Véronique Lespinet-Najib, Ricardo Ron-Angevin*