TP Analyse de données - Apprentissage non supervisé

Introduction

Nous allons utiliser le logiciel `R` (documentation ; possibilité d'utiliser Rstudio). Dans un dossier *AnalyseDeDonnees* créer deux sous dossiers :

- Code - dans lequel vous placerez vos fichiers de code
- Data - dans lequel vous placerez les fichiers du répertoire "Data" (à télécharger via Moodle)

Les 3 TP *Analyse de données* présentent différentes méthodes d'analyse de données : le but n'est pas de finir les TP le plus vite possible mais d'analyser les résultats ! Un rapport de 1 page **maximum** vous est demandé pour chacun des 3 TP : ne choisir que les résultats les plus intéressants et les **commenter**.

Ne pas hésiter à utiliser l'aide de `R` grâce à la commande :

```
help(...)
```

Analyse en composantes principales

0. Télécharger le cours sur l'Analyse en Composantes Principales.

1. Créer un fichier *ACP.R* dans le dossier *Code*. Charger les packages d'intérêt en ajoutant dans le fichier :

```
# Adresse du dossier où vous travaillez
setwd("/Users/.../AnalyseDonnees/TP/TP/Code")
# Packages utilisés dans la suite
library("FactoMineR")
library(PCAmixdata)
```

2. Charger les données et les afficher :

```
# Chargement des données
load("../Data/eaux.RData")
# Affichage des données
print(data,digits=4)
```

3. Calculer la moyenne et l'écart type de chacune des variables :

```
# Calcul de la moyenne et de l'écart type des variables
mean <- apply(data,2,mean)
std <- apply(data,2,sd) #standard deviation
stat <- rbind(mean,std)
# Affichage
print(stat,digits=4)
```

4. Afficher les données centrées-réduites :

```
# Création des données centrées ...
datanorm <- sweep(data,2,mean,"-")
# ... et réduites
datanorm <- sweep(datanorm,2,std,"/")
# Affichage des données centrées - réduites
print(datanorm,digits=4)
```

5. Visualiser la description bivariée des 5 premières variables :

```
# Visualisation des données en description bivariée
pairs(data[,1:5])
# Afficher la matrice de corrélation
ggcorr(data[,1:5])
# Aller encore plus loin avec ggpairs
ggpairs(data[,1:5])
```

6. Afficher la matrice des distances entre les individus et les corrélations entre les variables :

```
# Matrice des distances entre les individus
dist(data)
# Corrélation entre les variables
cor(data[,1:5])
```

7. Faire l'Analyse en Composantes Principales sur les données d'origine :

```
# Analyse en composantes principales sur les données d'origine
# (scale.unit=FALSE)
res <- PCA(data,graph=FALSE,scale.unit=FALSE)
# Figure individus
plot(res,choix="ind",cex=1.5,title="")
# Figure variables
plot(res,choix="var",cex=1.5,title="")
```

8. Faire l'Analyse en Composantes Principales sur les données centrées-réduites :

```
# Analyse en composantes principales sur les données centrées-réduites
# (par défaut: scale.unit=TRUE)
resnorm <- PCA(data,graph=FALSE)
# Figure individus
plot(resnorm,choix="ind",cex=1.5,title="")
# Figure variables
plot(resnorm,choix="var",cex=1.5,title="")
```

9. Interprétation des résultats : combien de composantes peut-on retenir ? Utiliser les règles de Kaiser et du coude.

```
# Inertie (variance) des composantes principales
resnorm$eig
barplot(resnorm$eig[,1])
```

10. Interprétation des résultats : qualité de la projection des individus (angle entre l'individu et les composantes principales)

```
# Projection des individus
resnorm$ind$cos2
# Somme avec les 2 premières
resnorm$ind$cos2[,1]+resnorm$ind$cos2[,2]
# Et les 3 premières ?
resnorm$ind$cos2[,1]+resnorm$ind$cos2[,2]+resnorm$ind$cos2[,3]
```

11. Interprétation des résultats : contribution des individus (à regarder en même temps que le graphe de projection)

```
# Contribution des individus
resnorm$ind$contrib
```

12. Interprétation des résultats : qualité de la projection et contribution des variables

```
# Projection des variables
resnorm$var$cos2
# Somme avec les 2 premières
resnorm$var$cos2[,1]+resnorm$var$cos2[,2]
# Et les 3 ?
resnorm$var$cos2[,1]+resnorm$var$cos2[,2]+resnorm$var$cos2[,3]
# Contribution des variables
resnorm$var$contrib
```

En plus - Quand vous faites de l'ACP il y a deux erreurs à éviter :

- Attention aux données très asymétriques : par exemple beaucoup de très petites valeurs et quelques très grandes (dans ce cas là une transformation des données peut être utile ...).
- Attention à l'effet *taille* (toutes les variables ont des contributions positives sur un axe) quand les données sont corrélées entre elles.

0. Télécharger le cours sur le partitionnement.

Partitionnement kmeans

1. Créer un fichier *partitionnement.R* dans le dossier *Code*. Charger les packages d'intérêt en ajoutant dans le fichier :

```
# Adresse du dossier où vous travaillez
setwd("/Users/.../AnalyseDonnees/TP/TP/Code")
# Packages utilisés dans la suite
library("FactoMineR")
```

2. Charger les données et les afficher :

```
# Données sur les fromages
X<-read.table("../Data/fromage.txt",sep="",header=TRUE,row.names=1)
print(X)
```

3. Calculer la moyenne et l'écart type de chacune des variables :

```
# Calcul de la moyenne et de l'écart type des variables
mean <- apply(X,2,mean)
std <- apply(X,2,sd) #standard deviation
stat <- rbind(mean,std)
# Affichage
print(stat,digits=4)
```

4. Créer et afficher les données centrées-réduites :

```
# Création des données centrées ...
Xnorm <- sweep(X,2,mean,"-")
# ... et réduites
Xnorm <- sweep(Xnorm,2,std,"/")
# Affichage des données centrées - réduites
print(Xnorm,digits=4)
```

5. Fixer le nombre de clusters souhaité (faire varier ce paramètre à la fin de l'exercice !)

```
# Nombre de clusters souhaité
numcluster <- 5
```

6. Appliquer l'algorithme des kmeans sur les données brutes ... puis sur les données centrées réduites. étudier les résultats obtenus :

```
## KMEANS
# Algorithme des kmeans (avec affichage)
km <- kmeans(X,numcluster,nstart=50)
print(km)
# Algorithme des kmeans sur données centrées-réduites (avec affichage)
kmnorm <- kmeans(Xnorm,numcluster,nstart=50)
print(kmnorm)
```

7. Comme il y a plus de 2 variables dans cet exemple nous ne pouvons pas visualiser facilement les résultats ... L'ACP peut nous y aider : on va afficher les clusters sur les premières directions principales. Tout d'abord concaténer aux données le résultat obtenu par l'algorithme des kmeans pour les 2 cas considérées :

```
# Concatenation des données avec leur résultat de cluster
cluster <- as.factor(km$cluster)
clusternorm <- as.factor(kmnorm$cluster)
XplusCluster <- data.frame(X,cluster=cluster)
XnormplusCluster <- data.frame(Xnorm,cluster=clusternorm)
colclust <- length(X)+1
print(XplusCluster)
print(XnormplusCluster)
```

8. Mettre en place l'ACP sur les données brutes et afficher les résultats. Les couleurs correspondent aux différents clusters. Interpréter les résultats.

```
# ACP sur les données brutes
rPCA <- PCA(XplusCluster,scale.unit=FALSE,graph=FALSE,quali.sup=colclust)

# Nuage des individus et des variables dans le premier plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCA,axes=c(1,2),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCA,axes=c(1,2),choix="var")
# Nuage des individus et des variables dans le deuxième plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCA,axes=c(1,3),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCA,axes=c(1,3),choix="var")
```

9. Mettre en place l'ACP sur les données centrées-réduites et afficher les résultats. Les couleurs correspondent aux différents clusters. Interpréter les résultats.

```
# ACP sur les données centrées-réduites
rPCAnorm <- PCA(XnormplusCluster,graph=FALSE,quali.sup=colclust)

# Nuage des individus et des variables dans le premier plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCAnorm,,axes=c(1,2),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCAnorm,axes=c(1,2),choix="var")
# Nuage des individus et des variables dans le deuxième plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCAnorm,axes=c(1,3),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCAnorm,axes=c(1,3),choix="var")
```

Partitionnement hiérarchique (distance de Ward)

10. Mettre en place la classification hiérarchique de Ward sur les données brutes ... :

```
#Classification hiérarchique de Ward sur données brutes
d <- dist(X)
tree <- hclust(d^2,method="ward.D2")
par(mfrow=c(1,1))
plot(tree)
```

11. ... ainsi que sur les données centrées-réduites :

```
#Classification hiérarchique de Ward sur données centrées-réduites
dnorm <- dist(Xnorm)
treenorm <- hclust(dnorm^2,method="ward.D2")
plot(treenorm)
```

12. Concaténer les données avec leurs résultats de cluster en vue de l'ACP :

```
# Concatenation des données avec leur résultat de cluster
clusterW <- as.factor(cutree(tree,numcluster))
XplusClusterW <- data.frame(X,cluster=clusterW)
print(XplusClusterW)
clusternormW <- as.factor(cutree(treenorm,numcluster))
XnormplusClustW <- data.frame(Xnorm,cluster=clusternormW)
print(XnormplusClustW)
```

13. Mise en place de l'ACP sur les données brutes :

```
# ACP sur les données brutes
rPCAW <- PCA(XplusClusterW,scale.unit=FALSE,graph=FALSE,quali.sup=colclust)
# Nuage des individus et des variables dans le premier plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCAW,axes=c(1,2),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCAW,axes=c(1,2),choix="var")
# Nuage des individus et des variables dans le deuxième plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCAW,axes=c(1,3),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCAW,axes=c(1,3),choix="var")
```

14. Mise en place de l'ACP sur les données centrées-réduites :

```
# ACP sur les données centrées-réduites
rPCAnormW <- PCA(XnormplusClustW,scale.unit=FALSE,graph=FALSE,quali.sup=colclust)
# Nuage des individus et des variables dans le premier plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCAnormW,axes=c(1,2),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCAnormW,axes=c(1,2),choix="var")
# Nuage des individus et des variables dans le deuxième plan factoriel
par(mfrow=c(1,2))
plot.PCA(rPCAnormW,axes=c(1,3),choix="ind",habillage=colclust,invisible="quali")
plot.PCA(rPCAnormW,axes=c(1,3),choix="var")
```