

# Introduction au Data Mining - TD Complet

Julien JACQUES

14/03/2019

## Pré-traitements sur les données

Commençons par importer les données

```
data=read.table("VisaPremier.txt",header=TRUE,na.strings = ".",stringsAsFactors = TRUE)
```

Identifions les variables catégorielles et quantitatives, puis créons deux jeux de données, quali. et quanti.

```
ind_categ_feature=c(1,2,3,4,6,8,9,45,46,47)
data_categ=data[,ind_categ_feature]
data_continuous=data[,-ind_categ_feature]
```

Commençons par regarder les variables catégorielles

```
summary(data_categ)
```

```
##      matricul      departem      ptvente      sexe      sitfamil
## Min.   : 113333 Min.   : 6.00 Min.   :1.000 Sfem:405 F.   : 32
## 1st Qu.: 860436 1st Qu.:31.00 1st Qu.:1.000 Shom:668 Fcel:360
## Median :1948586 Median :31.00 Median :1.000      Fdiv: 86
## Mean   :2489307 Mean   :41.18 Mean   :1.664      Fmar:547
## 3rd Qu.:3901594 3rd Qu.:61.25 3rd Qu.:2.000      Fsep: 14
## Max.   :7589439 Max.   :97.00 Max.   :7.000      Fuli: 25
##      NA's      :7      Fveu: 9
##      csp      codeqlt      cartevp      sexer      cartevpr
## Pcad   :446 A      :207 Cnon:714 Min.   :0.0000 Min.   :0.0000
## Pemp   :288 B      :303 Coui:359 1st Qu.:0.0000 1st Qu.:0.0000
## Psan   :199 C      :218      Median :0.0000 Median :0.0000
## Pouv   : 85 D      :168      Mean   :0.3774 Mean   :0.3346
## Part   : 31 E      : 44      3rd Qu.:1.0000 3rd Qu.:1.0000
## Pret   : 21 NA's:133      Max.   :1.0000 Max.   :1.0000
## (Other): 3
```

On peut voir qu'il y a des données manquantes, et certaines sont enregistrées comme des variables quantitative (integer)

On va remplacer les départements manquants par le département le plus fréquent. Pour trouver ce dernier ce n'est pas si simple. On va calculer un tableau de contingence que l'on va trier par ordre décroissant. Le département le plus fréquent arrive en premier et on peut l'utiliser pour remplacer les données manquantes.

```
tmp=sort(table(data_categ$departem),decreasing=TRUE)
data_categ$departem[is.na(data_categ$departem)]=as.integer(names(tmp)[1])
data_categ$departem=factor(data_categ$departem)
```

Pour la variable codeqlt, comme il y a plus de NA, on va remplacer en tirant aléatoirement le codeqlt en fonction des fréquences observées chez les non NA

```
data_categ$codeqlt=as.factor(data_categ$codeqlt)
nbNA=length(data_categ$codeqlt[is.na(data_categ$codeqlt)])
data_categ$codeqlt[is.na(data_categ$codeqlt)]=sample(levels(data_categ$codeqlt),nbNA,
                                                    prob=table(data_categ$codeqlt),replace=TRUE)
data_categ$codeqlt=as.factor(data_categ$codeqlt)
```

Et on transforme les variables matricul et ptvente en variables catégorielles (factor)

```
data_categ$matricul=factor(data_categ$matricul)
data_categ$ptvente=factor(data_categ$ptvente)
```

On a deux variables redondantes qui sont le recodage de la variable sexe et de la variable possession de la carte Visa Premier. Je les supprime. Je supprime également le matricule du client dont je n'ai pas besoin ici

```
data_categ$sexe=NULL
data_categ$cartevpr=NULL
data_categ$matricul=NULL
```

On va maintenant remplacer automatiquement les NA des variables manquantes par la moyenne

```
summary(data_continuous)
```

```
##      age      anciente      nbimpaye      mtrejet
##  Min.   :18.00   Min.    : 1.0   Min.    :0   Min.    : -51.00000
##  1st Qu.:33.00   1st Qu.: 45.0   1st Qu.:0   1st Qu.: 0.00000
##  Median :43.00   Median :136.0   Median :0   Median : 0.00000
##  Mean   :42.53   Mean    :157.1   Mean    :0   Mean    : -0.07269
##  3rd Qu.:52.00   3rd Qu.:216.0   3rd Qu.:0   3rd Qu.: 0.00000
##  Max.   :65.00   Max.    :870.0   Max.    :0   Max.    : 0.00000
##
##      nbopguic      moycred3      aveparmo      endette
##  Min.    : 0.000   Min.     : 0.00   Min.     : 0   Min.     : 0.000
##  1st Qu.: 0.000   1st Qu.: 3.00   1st Qu.: 0   1st Qu.: 0.000
##  Median : 1.000   Median : 12.00   Median : 6017   Median : 0.000
##  Mean    : 1.505   Mean     : 47.63   Mean     : 57249   Mean    : 5.457
##  3rd Qu.: 2.000   3rd Qu.: 27.00   3rd Qu.: 57818   3rd Qu.: 6.000
##  Max.    :28.000   Max.    :19579.00   Max.    :970000   Max.    :99.000
##
##      engagemt      engagemc      engagemm      nbcptvue
##  Min.     : 0   Min.     : 0   Min.     : 0   Min.     :0.000
##  1st Qu.: 0   1st Qu.: 0   1st Qu.: 0   1st Qu.:1.000
##  Median : 0   Median : 0   Median : 0   Median :1.000
##  Mean    : 77316   Mean    : 4199   Mean    : 20230   Mean    :1.028
##  3rd Qu.: 34927   3rd Qu.: 0   3rd Qu.: 0   3rd Qu.:1.000
##  Max.    :3472938   Max.    :500780   Max.    :1618242   Max.    :4.000
##
##      moysold3      moycredi      agemvt      nbop
##  Min.     : -70050   Min.     : 0.00   Min.     : 0.00   Min.     : 0
##  1st Qu.: 434   1st Qu.: 2.00   1st Qu.: 13.00   1st Qu.: 6
##  Median : 4371   Median : 11.00   Median : 13.00   Median : 25
##  Mean    : 10674   Mean     : 25.91   Mean     : 19.06   Mean     : 29
##  3rd Qu.: 11034   3rd Qu.: 24.00   3rd Qu.: 14.00   3rd Qu.: 43
##  Max.    :241827   Max.    :4079.00   Max.    :944.00   Max.    :262
##
##      mtfactor      engageml      nbvie      mtvie
##  Min.     : 0   Min.     : 0   Min.     : 0.0000   Min.     : 0
##
##      NA's :6
```

```
## 1st Qu.:      0 1st Qu.:      0 1st Qu.: 0.0000 1st Qu.:      0
## Median :      0 Median :      0 Median : 0.0000 Median :      0
## Mean   : 23379 Mean   : 52888 Mean   : 0.2395 Mean   : 35915
## 3rd Qu.: 3500 3rd Qu.:      0 3rd Qu.: 0.0000 3rd Qu.:      0
## Max.   :1331530 Max.   :3472938 Max.   :13.0000 Max.   :5449561
##
##      nbeparmo      mteparmo      nbeparlo      mteparlo
## Min.   :0.000 Min.   :      0 Min.   :0.0000 Min.   :      0
## 1st Qu.:0.000 1st Qu.:      0 1st Qu.:0.0000 1st Qu.:      0
## Median :1.000 Median :    6017 Median :0.0000 Median :      0
## Mean   :1.473 Mean   :   75442 Mean   :0.6524 Mean   : 32184
## 3rd Qu.:2.000 3rd Qu.:   58390 3rd Qu.:1.0000 3rd Qu.: 23854
## Max.   :9.000 Max.   :19508920 Max.   :4.0000 Max.   :579603
##
##      nbllivret      mtlivret      nbeparlt      mteparlt
## Min.   :0.0000 Min.   :      0 Min.   :0.00000 Min.   :      0
## 1st Qu.:0.0000 1st Qu.:      0 1st Qu.:0.00000 1st Qu.:      0
## Median :1.0000 Median :    127 Median :0.00000 Median :      0
## Mean   :0.7586 Mean   : 20740 Mean   :0.05871 Mean   : 4325
## 3rd Qu.:1.0000 3rd Qu.: 15544 3rd Qu.:0.00000 3rd Qu.:      0
## Max.   :4.0000 Max.   :970000 Max.   :6.00000 Max.   :559559
##
##      nbeparte      mteparte      nbbon      mtbon
## Min.   :0.000000 Min.   :    0.00 Min.   :0.000000 Min.   :      0
## 1st Qu.:0.000000 1st Qu.:    0.00 1st Qu.:0.000000 1st Qu.:      0
## Median :0.000000 Median :    0.00 Median :0.000000 Median :      0
## Mean   :0.002796 Mean   :   19.71 Mean   :0.000932 Mean   : 18173
## 3rd Qu.:0.000000 3rd Qu.:    0.00 3rd Qu.:0.000000 3rd Qu.:      0
## Max.   :1.000000 Max.   :21149.00 Max.   :1.000000 Max.   :19500000
##
##      nbpaiecb      nbcb      nbcbptar      avtscpte
## Min.   : 0.0 Min.   :0.00 Min.   :0.0000 Min.   :      0
## 1st Qu.: 0.0 1st Qu.:0.00 1st Qu.:0.0000 1st Qu.:   3184
## Median : 9.0 Median :1.00 Median :0.0000 Median :  23993
## Mean   :11.5 Mean   :1.07 Mean   :0.1361 Mean   : 146819
## 3rd Qu.:18.0 3rd Qu.:2.00 3rd Qu.:0.0000 3rd Qu.: 114807
## Max.   :69.0 Max.   :5.00 Max.   :4.0000 Max.   :19856243
## NA's   :278
##      aveparfi      nbjdebit
## Min.   :      0 Min.   : 0.00
## 1st Qu.:      0 1st Qu.: 0.00
## Median :      0 Median : 0.00
## Mean   : 50727 Mean   : 12.08
## 3rd Qu.:    500 3rd Qu.: 10.00
## Max.   :7066619 Max.   :134.00
##
```

```
for (j in 1:ncol(data_continuous)){
  if (sum(is.na(data_continuous[,j]))>0){
    indiceNA=which(is.na(data_continuous[,j]))
    data_continuous[indiceNA,j]=median(data_continuous[,j],na.rm=TRUE)
  }
}
summary(data_continuous)
```

##	age	anciente	nbimpaye	mtrejet
##	Min. :18.00	Min. : 1.0	Min. :0	Min. : -51.00000
##	1st Qu.:33.00	1st Qu.: 45.0	1st Qu.:0	1st Qu.: 0.00000
##	Median :43.00	Median :136.0	Median :0	Median : 0.00000
##	Mean :42.53	Mean :157.1	Mean :0	Mean : -0.07269
##	3rd Qu.:52.00	3rd Qu.:216.0	3rd Qu.:0	3rd Qu.: 0.00000
##	Max. :65.00	Max. :870.0	Max. :0	Max. : 0.00000
##	nbopguic	moycred3	aveparmo	endette
##	Min. : 0.000	Min. : 0.00	Min. : 0	Min. : 0.000
##	1st Qu.: 0.000	1st Qu.: 3.00	1st Qu.: 0	1st Qu.: 0.000
##	Median : 1.000	Median : 12.00	Median : 6017	Median : 0.000
##	Mean : 1.505	Mean : 47.63	Mean : 57249	Mean : 5.457
##	3rd Qu.: 2.000	3rd Qu.: 27.00	3rd Qu.: 57818	3rd Qu.: 6.000
##	Max. :28.000	Max. :19579.00	Max. :970000	Max. :99.000
##	engagemt	engagemc	engagemm	nbcpvue
##	Min. : 0	Min. : 0	Min. : 0	Min. :0.000
##	1st Qu.: 0	1st Qu.: 0	1st Qu.: 0	1st Qu.:1.000
##	Median : 0	Median : 0	Median : 0	Median :1.000
##	Mean : 77316	Mean : 4199	Mean : 20230	Mean :1.028
##	3rd Qu.: 34927	3rd Qu.: 0	3rd Qu.: 0	3rd Qu.:1.000
##	Max. :3472938	Max. :500780	Max. :1618242	Max. :4.000
##	moysold3	moycredi	agemvt	nbop
##	Min. : -70050	Min. : 0.00	Min. : 0.00	Min. : 0
##	1st Qu.: 434	1st Qu.: 2.00	1st Qu.: 13.00	1st Qu.: 6
##	Median : 4371	Median : 11.00	Median : 13.00	Median : 25
##	Mean : 10674	Mean : 25.91	Mean : 19.03	Mean : 29
##	3rd Qu.: 11034	3rd Qu.: 24.00	3rd Qu.: 14.00	3rd Qu.: 43
##	Max. :241827	Max. :4079.00	Max. :944.00	Max. :262
##	mtfactur	engageml	nbvie	mtvie
##	Min. : 0	Min. : 0	Min. : 0.0000	Min. : 0
##	1st Qu.: 0	1st Qu.: 0	1st Qu.: 0.0000	1st Qu.: 0
##	Median : 0	Median : 0	Median : 0.0000	Median : 0
##	Mean : 23379	Mean : 52888	Mean : 0.2395	Mean : 35915
##	3rd Qu.: 3500	3rd Qu.: 0	3rd Qu.: 0.0000	3rd Qu.: 0
##	Max. :1331530	Max. :3472938	Max. :13.0000	Max. :5449561
##	nbeparmo	mteparmo	nbeparlo	mteparlo
##	Min. :0.000	Min. : 0	Min. :0.0000	Min. : 0
##	1st Qu.:0.000	1st Qu.: 0	1st Qu.:0.0000	1st Qu.: 0
##	Median :1.000	Median : 6017	Median :0.0000	Median : 0
##	Mean :1.473	Mean : 75442	Mean :0.6524	Mean : 32184
##	3rd Qu.:2.000	3rd Qu.: 58390	3rd Qu.:1.0000	3rd Qu.: 23854
##	Max. :9.000	Max. :19508920	Max. :4.0000	Max. :579603
##	nblivret	mtlivret	nbeparlt	mteparlt
##	Min. :0.0000	Min. : 0	Min. :0.00000	Min. : 0
##	1st Qu.:0.0000	1st Qu.: 0	1st Qu.:0.00000	1st Qu.: 0
##	Median :1.0000	Median : 127	Median :0.00000	Median : 0
##	Mean :0.7586	Mean : 20740	Mean :0.05871	Mean : 4325
##	3rd Qu.:1.0000	3rd Qu.: 15544	3rd Qu.:0.00000	3rd Qu.: 0
##	Max. :4.0000	Max. :970000	Max. :6.00000	Max. :559559
##	nbeparte	mteparte	nbbon	mtbon
##	Min. :0.000000	Min. : 0.00	Min. :0.000000	Min. : 0
##	1st Qu.:0.000000	1st Qu.: 0.00	1st Qu.:0.000000	1st Qu.: 0
##	Median :0.000000	Median : 0.00	Median :0.000000	Median : 0
##	Mean :0.002796	Mean : 19.71	Mean :0.000932	Mean : 18173

```
## 3rd Qu.:0.000000 3rd Qu.: 0.00 3rd Qu.:0.000000 3rd Qu.: 0
## Max. :1.000000 Max. :21149.00 Max. :1.000000 Max. :19500000
## nbpaiecb nbc b ncbptar avtscpte
## Min. : 0.00 Min. :0.00 Min. :0.0000 Min. : 0
## 1st Qu.: 3.00 1st Qu.:0.00 1st Qu.:0.0000 1st Qu.: 3184
## Median : 9.00 Median :1.00 Median :0.0000 Median : 23993
## Mean :10.85 Mean :1.07 Mean :0.1361 Mean : 146819
## 3rd Qu.:14.00 3rd Qu.:2.00 3rd Qu.:0.0000 3rd Qu.: 114807
## Max. :69.00 Max. :5.00 Max. :4.0000 Max. :19856243
## aveparfi nbjdebit
## Min. : 0 Min. : 0.00
## 1st Qu.: 0 1st Qu.: 0.00
## Median : 0 Median : 0.00
## Mean : 50727 Mean : 12.08
## 3rd Qu.: 500 3rd Qu.: 10.00
## Max. :7066619 Max. :134.00
```

On enlève également la variable nbimpaye qui est constante

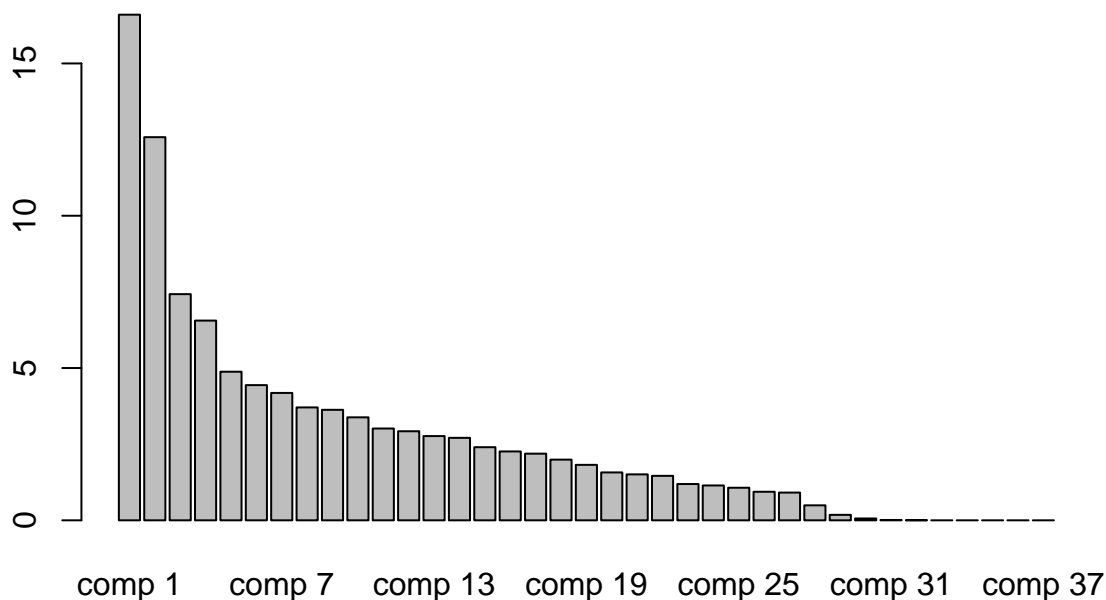
```
data_continuous$nbimpaye=NULL
```

## Analyse en composantes principales

Maintenant que le jeu de données est “propre”, on peut faire une ACP

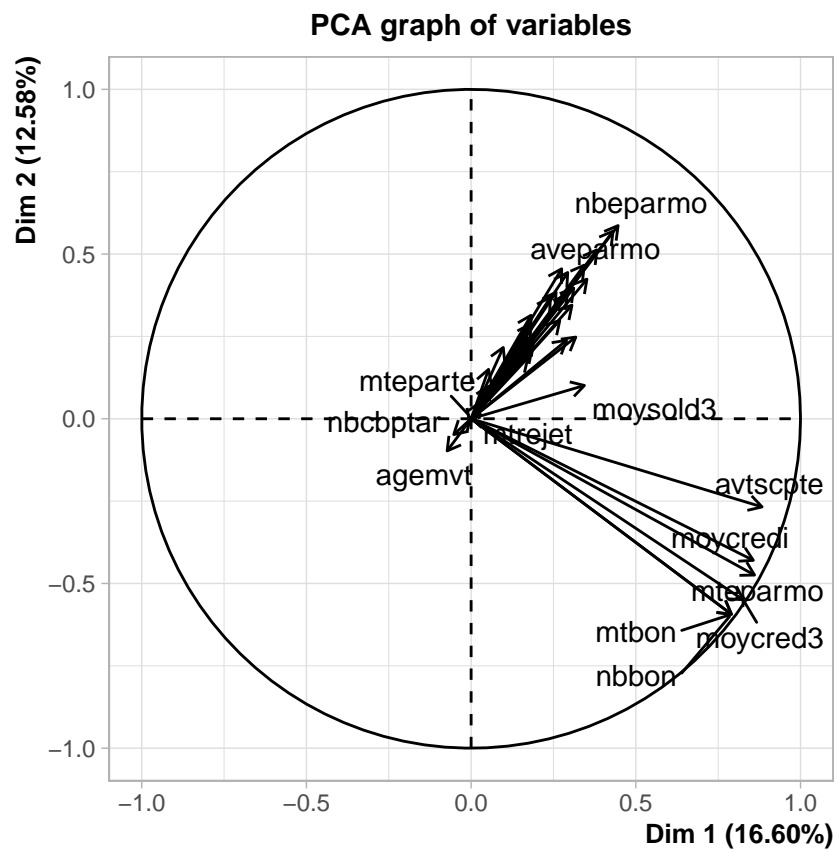
```
library(FactoMineR)
res.pca <- PCA(data_continuous, graph = FALSE)
barplot(res.pca$eig[,2],main="Pourcentage de variance expliquée")
```

### Pourcentage de variance expliquée

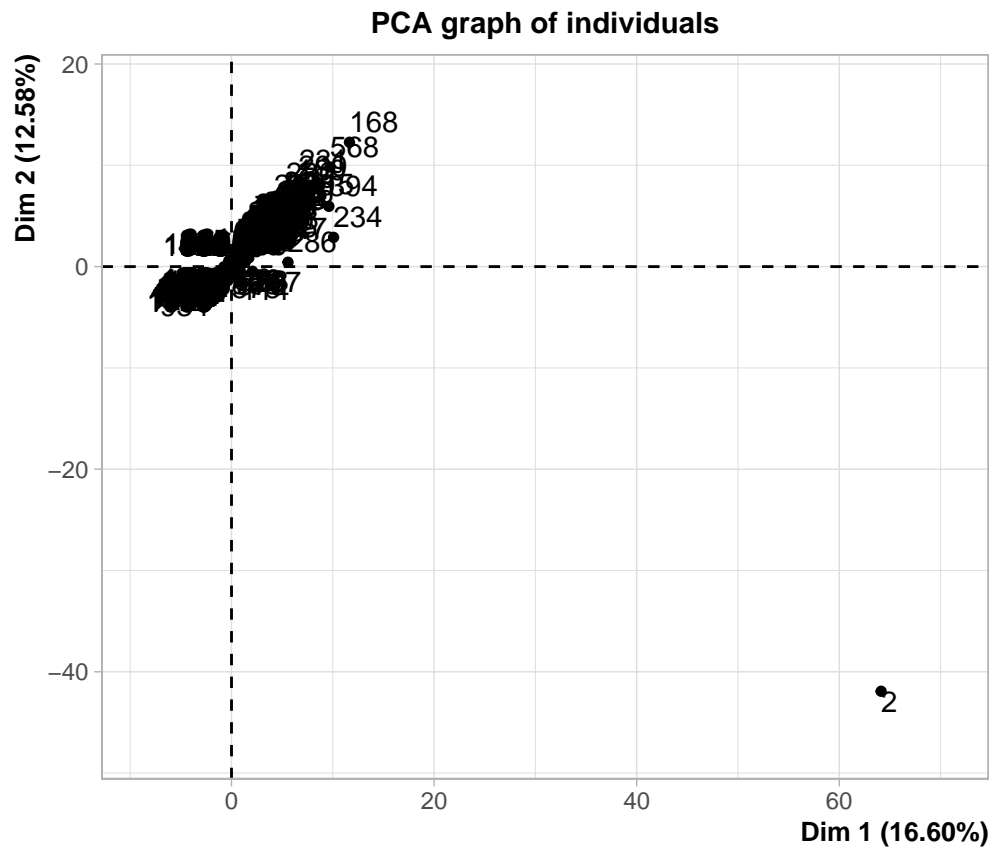


```
plot(res.pca,choix = "var")
```

```
## Warning: ggrepel: 24 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
plot(res.pca,choix = "ind")
```



On remarque un individu (n°2) atypique. Après examen on remarque que c'est un individu très très riche, qui a des produits d'épargne dont il est le seul à en posséder parmi la base. Mettons cet individu de coté et offrons lui autant de carte VP qu'il veut.

```
data_continuous<-data_continuous[-2,]
data_categ<-data_categ[-2,]
```

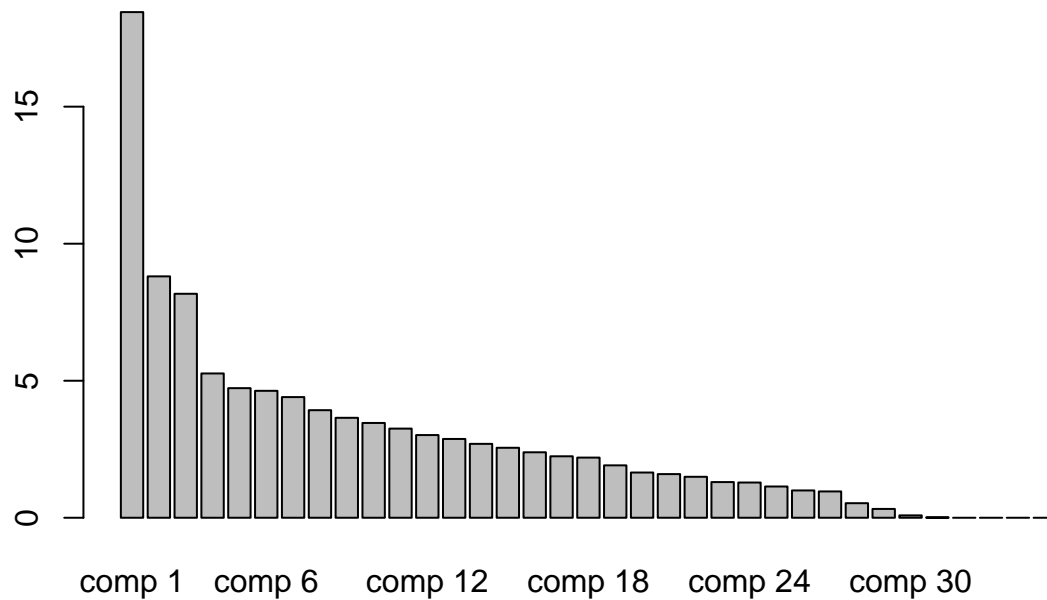
Du coup on enlève les variables nbbon et mtbon qui deviennent constante une fois que l'on a enlevé cet individu

```
data_continuous$nbbon=NULL
data_continuous$mtbon=NULL
```

On peut recommencer l'ACP (qui a bien changé sans cet individu)

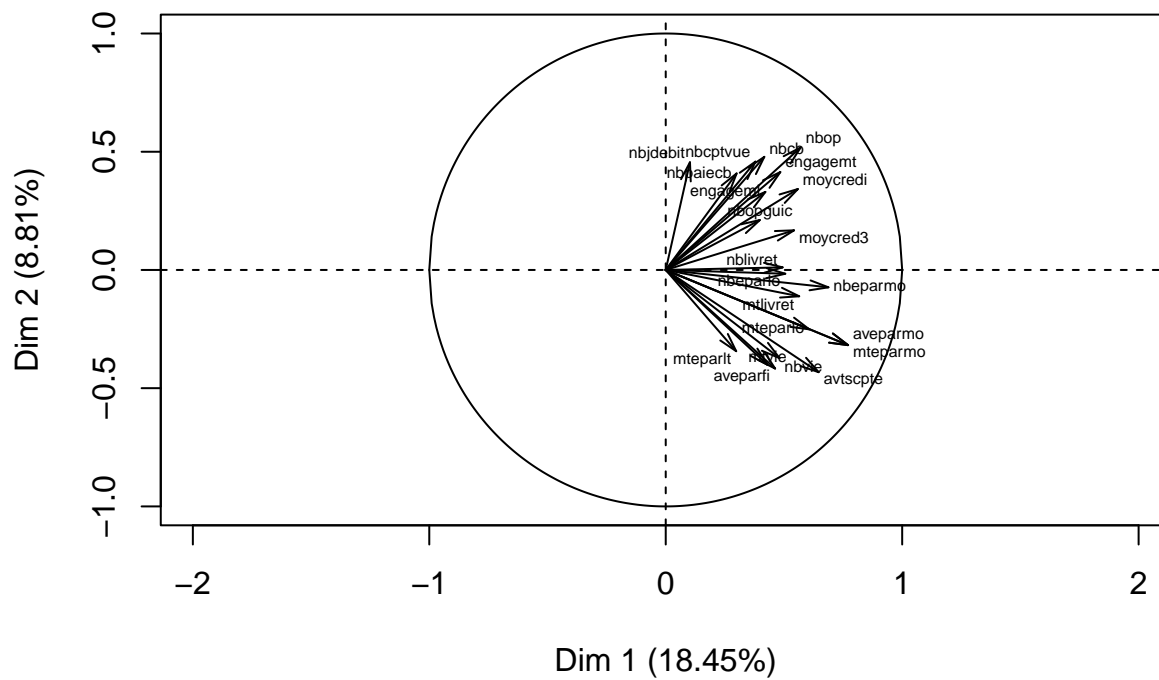
```
library(FactoMineR)
res.pca <- PCA(data_continuous, graph = FALSE)
barplot(res.pca$eig[,2],main="Pourcentage de variance expliquée")
```

### Pourcentage de variance expliquée



```
plot(res.pca, choix = "var", cex=.5, lim.cos2.var = 0.2, graph.type = 'classic')
```

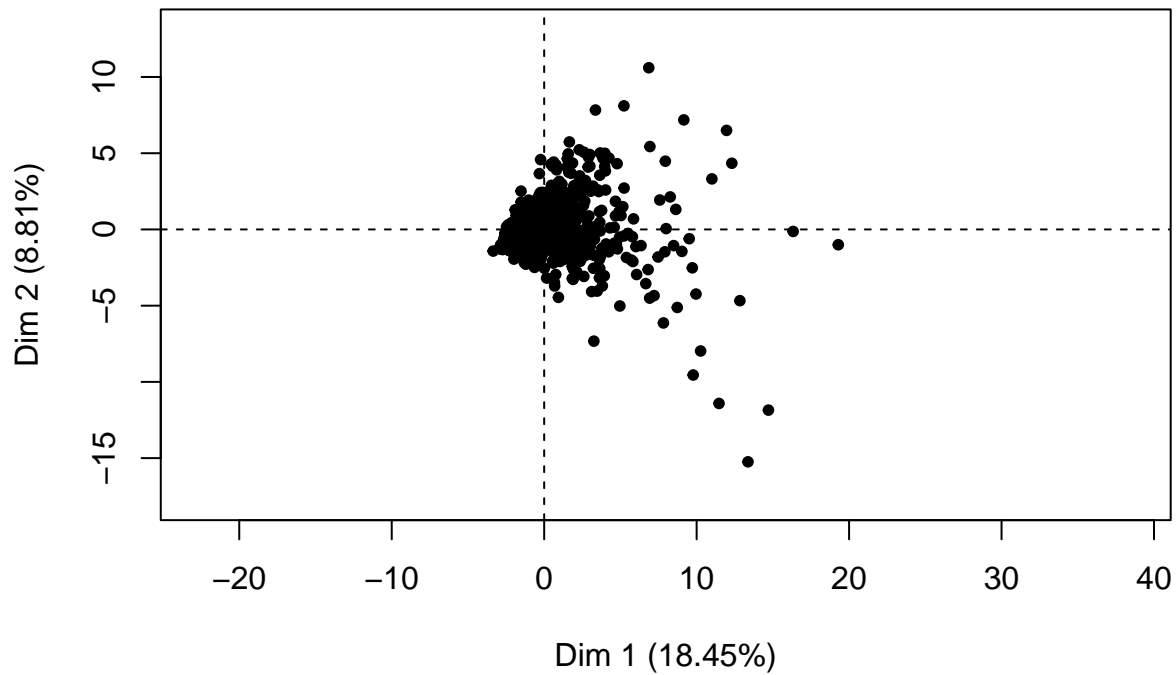
## PCA graph of variables



```
plot(res.pca, choix = "ind", label='none', graph.type = 'classic')
```



## PCA graph of individuals

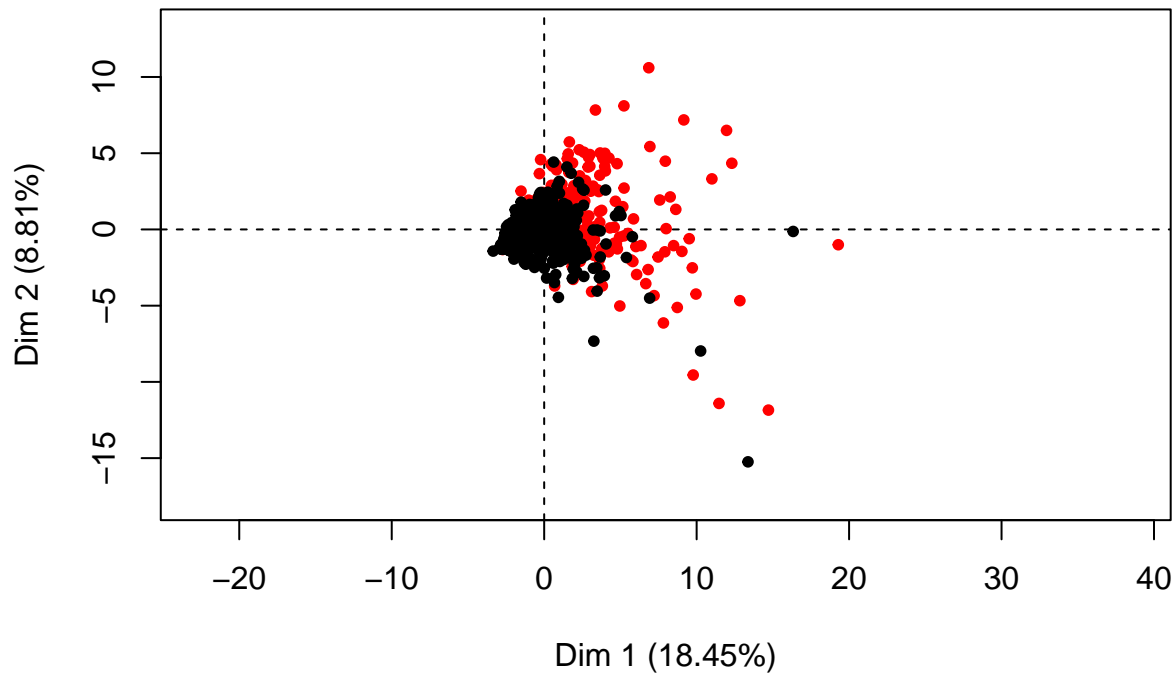


On remarque que le premier axe distingue les clients plutôt riches (à droite) des clients qui le sont moins (à gauche). Le second axe oppose les épargnants (en bas) aux dépensiers (en haut).

Finalement, on peut représenter sur ce graphique quels sont les individus qui possèdent la carte Visa Premier :

```
plot(res.pca,choix = "ind",label='none',graph.type = 'classic',col.ind = (data_categ$cartevp=="Oui")+1,
```

## PCA graph of individuals



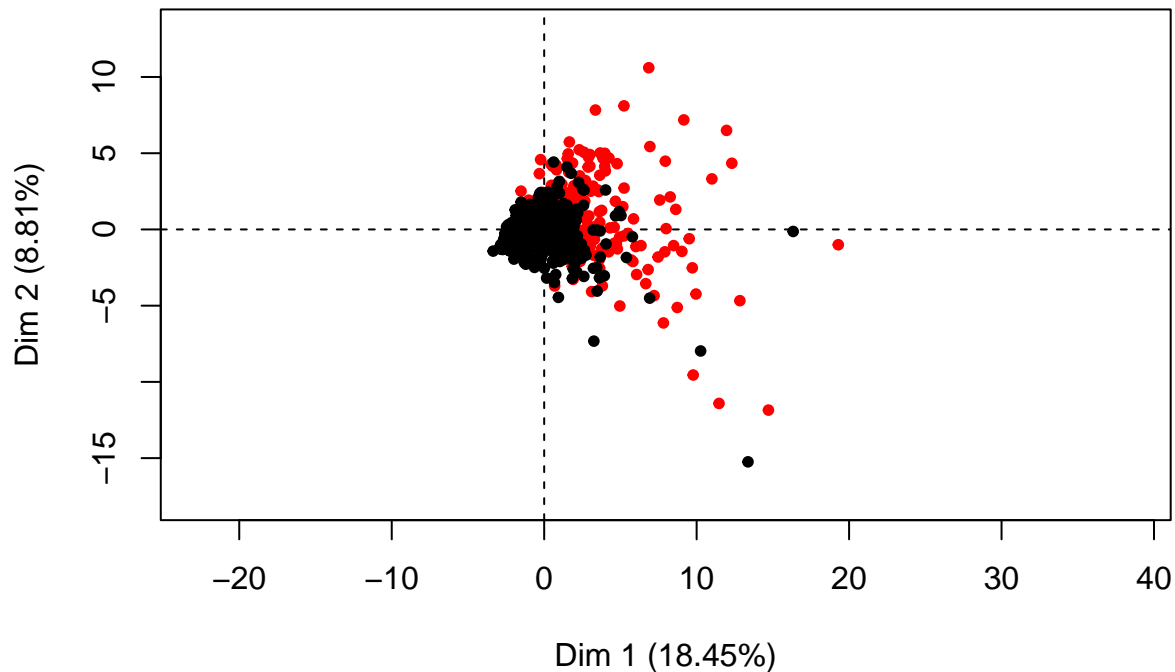
On constate que ce sont les clients riches, et plutôt dépensiers. De plus, la distinction est assez nette entre ceux qui possèdent ou non la carte, les modèles de classification que l'on va mettre en oeuvre devrait être plutôt performants.

### Prédiction de l'appétence à la carte VP

Afin d'avoir une idée de la difficulté de la tâche, nous pouvons représenter les individus dans le plan de l'ACP en les différenciant (avec leur couleur) suivant s'ils possèdent ou non la carte VP

```
plot(res.pca,choix = "ind",label='none',col.ind = (data_categ$cartevp=="Coui")+1,graph.type = 'classic')
```

## PCA graph of individuals



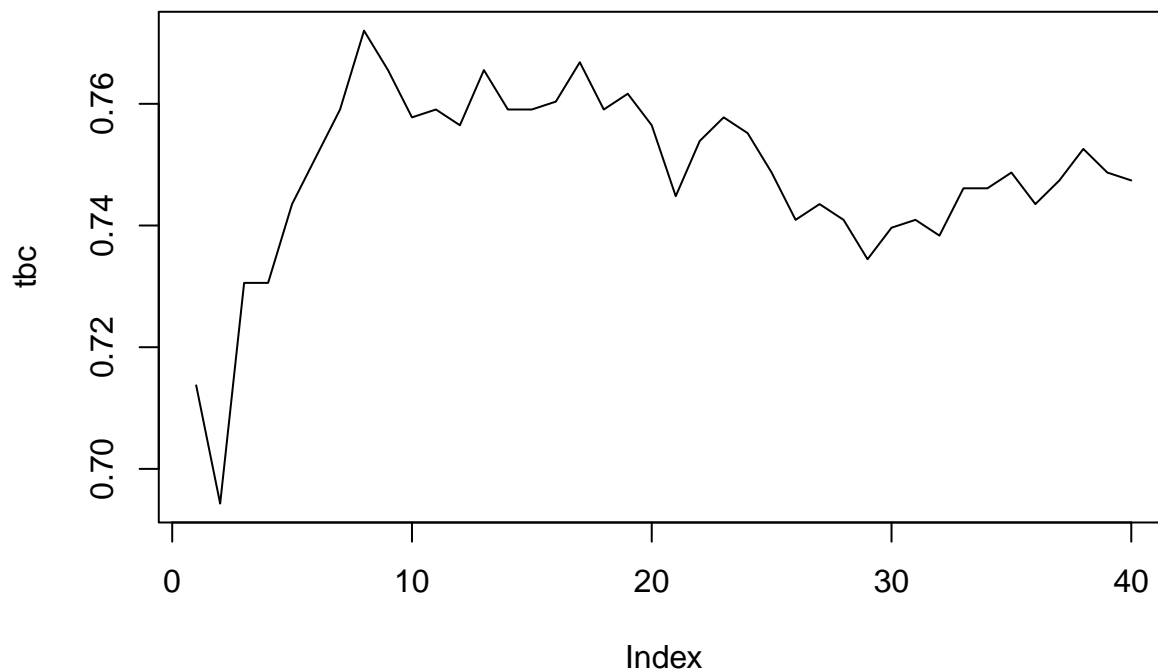
Commençons par construire une base de test, en tirant 300 individus au hasard. Je fixe la graine de mon ordinateur à 1 pour que tout le monde ait le même échantillon de test

```
set.seed(1)
ind_test=sample(1:nrow(data_continuous),300)
dco_test=data_continuous[ind_test,]
dco_app=data_continuous[-ind_test,]
dca_test=data_categ[ind_test,]
dca_app=data_categ[-ind_test,]
```

## k-plus proches voisins sur variables quantitatives

Pour choisir k, on va utiliser la validation croisée qui est implémentée en interne dans la fonction knn.cv

```
library(class)
tbc=NULL
for (k in 1:40){
  tmp=knn.cv(dco_app,dca_app$cartevp,k=k)
  tbc=c(tbc,mean(tmp==dca_app$cartevp))
}
plot(tbc,type='l')
```



Puis on relance les knn avec la valeur qui maximise le taux de bon classement.

```
print(which.max(tbc))
```

```
## [1] 8
```

```
mod1=knn(dco_app,dco_test,dca_app$cartevp,k=which.max(tbc))
```

Et on évalue le taux de bon classement sur l'échantillon test :

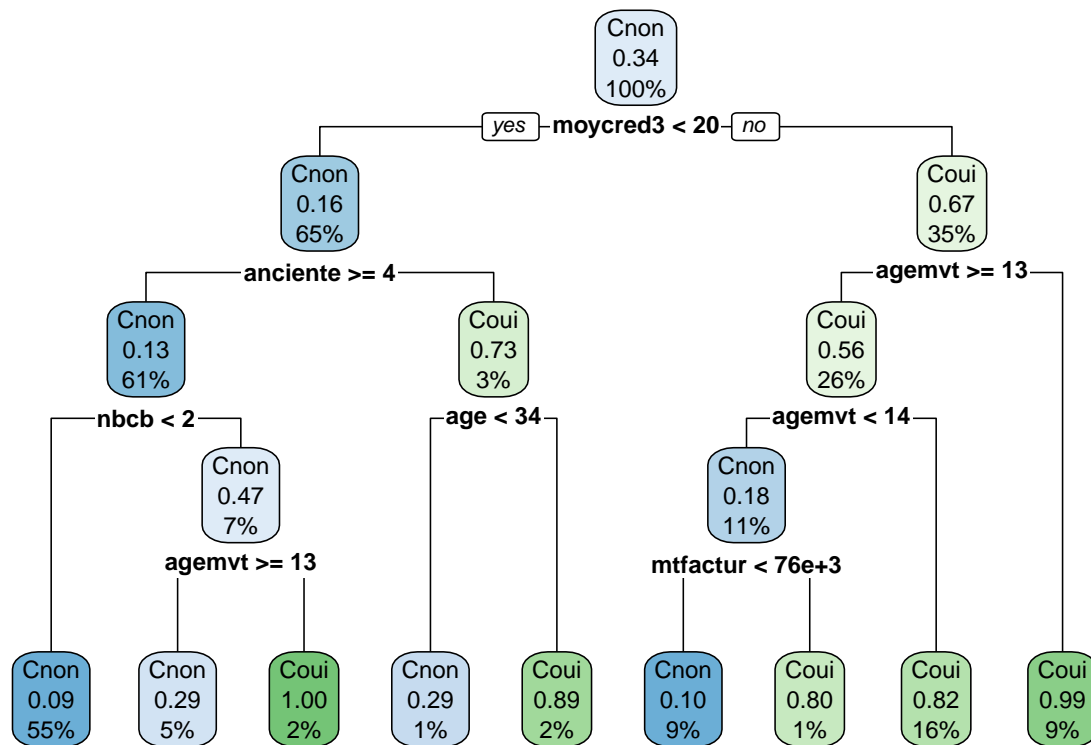
```
mean(mod1==dca_test$cartevp)
```

```
## [1] 0.77
```

## Arbre (CART) sur variables quantitative

Utilisons un arbre binaire sur les données quantitatives. Pour cela il faut utiliser un dataframe et donc rapatrier la variable à prédire cartevp dans le même dataframe que les variables quanti

```
library('rpart');library('rpart.plot')
dfco_app=data.frame(dco_app,cartevp=as.factor(dca_app$cartevp))
mod2=rpart(cartevp~.,data=dfco_app)
p=predict(mod2,newdata = dco_test,type='class')
rpart.plot(mod2)
```



Taux de bon classement par Arbre de classification sur les variables quanti :

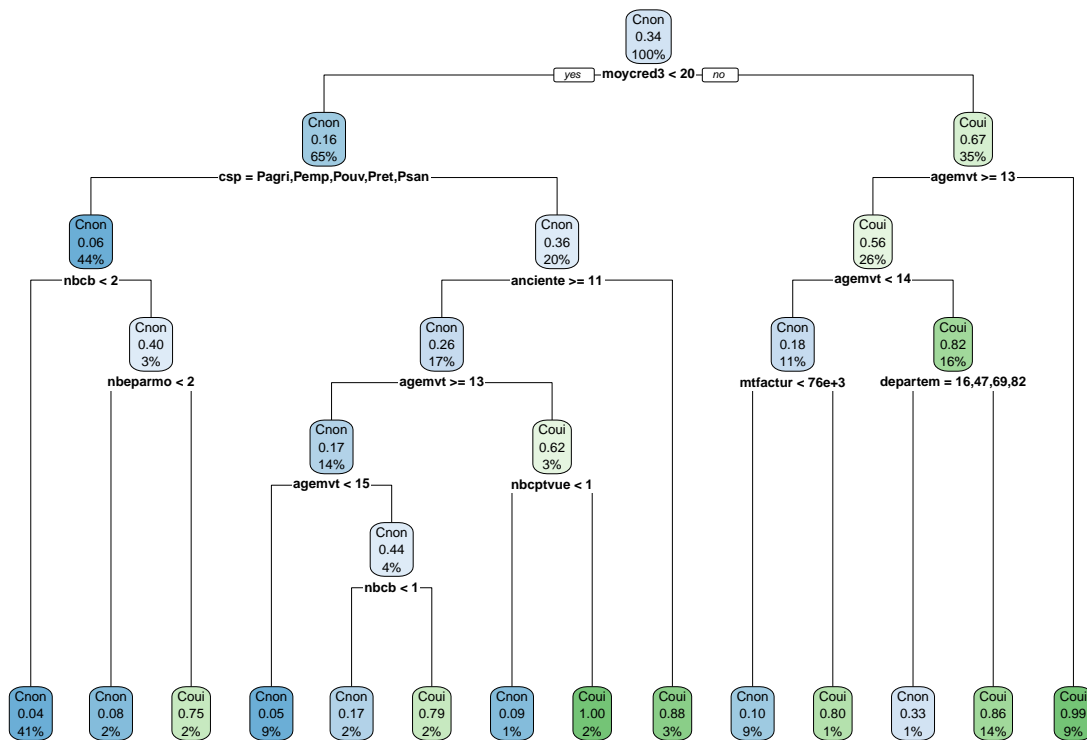
```
mean(p==dca_test$cartevp)
```

```
## [1] 0.8933333
```

## Arbre (CART) sur variables quantitative + catégorielle

Utilisons un arbre binaire sur les données quantitatives. Pour cela il faut utiliser un dataframe et donc rapatrier la variable à prédire cartevp dans le même dataframe que les variables quanti

```
library('rpart'); library('rpart.plot')
df_app = data.frame(dco_app, dca_app)
df_test = data.frame(dco_test, dca_test)
mod3 = rpart(cartevp ~ ., data = df_app)
p = predict(mod3, newdata = df_test, type = 'class')
rpart.plot(mod3)
```



```
mean(p==dca_test$cartevp)
```

```
## [1] 0.8866667
```

## Forêt aléatoire sur les variables quantitatives

Utilisons une forêt aléatoire, en laissant les paramètres (ntree : nombre d'arbre et mtry : nombre de variables candidates à chaque coupure) par défaut

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.4.1
```

```
## randomForest 4.7-1.2
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
mod4=randomForest(cartevp~.,data=dfco_app)
```

```
p=predict(mod4,newdata = dco_test,type='class')
```

Le taux d'erreur est

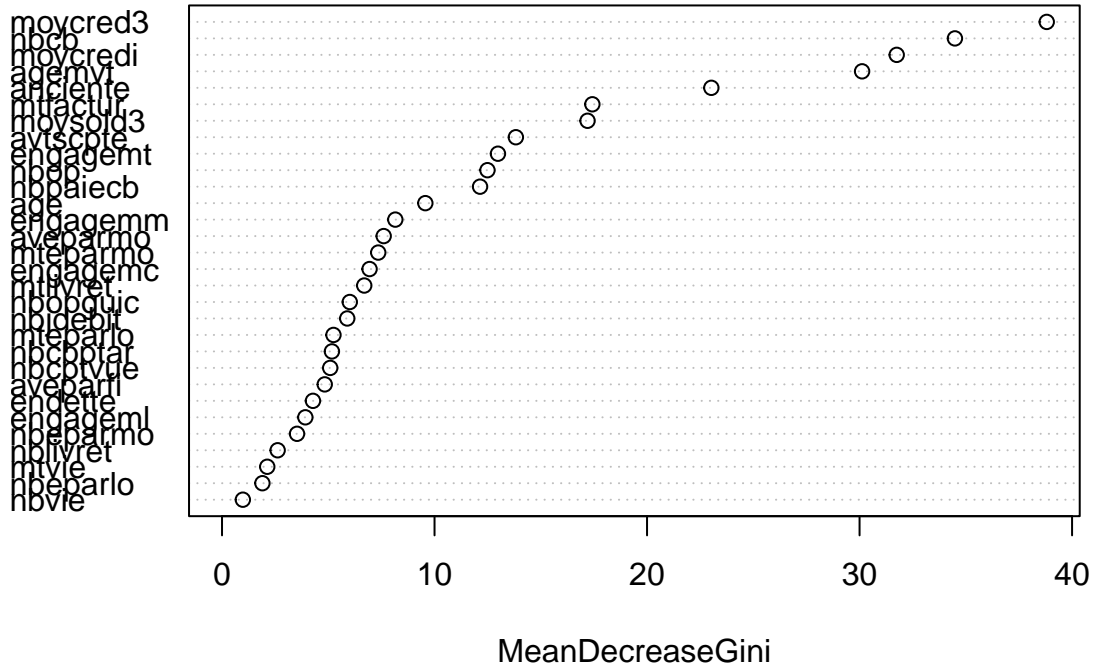
```
mean(p==dca_test$cartevp)
```

```
## [1] 0.9233333
```

On peut regarder l'importance des variables :

```
varImpPlot(mod4)
```

## mod4



logistique

# Régression

Effectuons une régression logistique sur les variables continues. Ajouter les variables catégorielles ne va pas forcément beaucoup aidé (au vu des résultats précédents), et va poser des soucis d'estimation du fait du grand nombre de département notamment (et du peu d'observation pour certains départements)

```
mod5=glm(carvevp~.,data=dfco_app,family=binomial(link = "logit"))
summary(mod5)
```

```
##
## Call:
## glm(formula = cartevp ~ ., family = binomial(link = "logit"),
##      data = dfco_app)
##
## Coefficients: (4 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.107e+00  6.402e-01  -6.416  1.40e-10 ***
## age          1.785e-02  1.127e-02   1.583  0.113358
## ancienne    -2.033e-03  9.126e-04  -2.228  0.025913 *
## mtrejet      8.605e-02  7.183e-02   1.198  0.230914
## nbopguic     2.593e-01  7.326e-02   3.539  0.000401 ***
## moycred3     -1.328e-03  3.620e-03  -0.367  0.713800
## aveparmo     7.971e-06  6.947e-06   1.147  0.251204
## endette     -1.871e-02  1.069e-02  -1.750  0.080129 .
## engagemt     1.373e-06  7.547e-07   1.819  0.068961 .
## engagemc     2.838e-05  1.303e-05   2.179  0.029356 *
## engagemm     9.629e-06  2.721e-06   3.539  0.000401 ***
## nbcptvue     4.335e-01  3.409e-01   1.272  0.203512
## moysold3     1.229e-05  5.734e-06   2.143  0.032141 *
## moycredi     2.941e-02  8.213e-03   3.581  0.000343 ***
## agemvt      -6.611e-03  9.728e-03  -0.680  0.496746
```

```
## nbop          -5.216e-02  1.043e-02  -5.000  5.73e-07 ***
## mtfactor      7.954e-06  3.049e-06   2.609  0.009093 **
## engageml      NA          NA          NA      NA
## nbvie         1.737e-03  2.642e-01   0.007  0.994755
## mtvie        -3.802e-06  2.340e-06  -1.625  0.104199
## nbeparmo      2.207e+12  5.856e+13   0.038  0.969935
## mteparmo      NA          NA          NA      NA
## nbeparlo     -2.207e+12  5.856e+13  -0.038  0.969935
## mteparlo      2.470e-06  6.595e-06   0.375  0.708026
## nblivret     -2.207e+12  5.856e+13  -0.038  0.969935
## mtlivret      4.099e-06  6.651e-06   0.616  0.537655
## nbeparlt     -2.207e+12  5.856e+13  -0.038  0.969935
## mteparlt      NA          NA          NA      NA
## nbeparte     -2.207e+12  5.856e+13  -0.038  0.969935
## mteparte      NA          NA          NA      NA
## nbpaiecb      1.595e-02  1.704e-02   0.936  0.349200
## nbcb          2.539e+00  2.507e-01  10.127  < 2e-16 ***
## nbcbptar     -3.120e+00  5.115e-01  -6.099  1.07e-09 ***
## avtscpte     -9.180e-06  3.393e-06  -2.706  0.006816 **
## aveparfi      1.208e-05  4.285e-06   2.819  0.004812 **
## nbjdebit      4.309e-03  5.590e-03   0.771  0.440861
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 990.45  on 771  degrees of freedom
## Residual deviance: 518.77  on 740  degrees of freedom
## AIC: 582.77
##
## Number of Fisher Scoring iterations: 25
```

L'effet de certaines variables ne peut être estimé (ce sont les variables quasiment constante, où seuls quelques individus ont des valeurs non nuls). Une sélection de variable stepwise devrait solutionner cela

```
library(MASS)
mod5_2=step(mod5,direction = "both",trace = F)
summary(mod5_2)

##
## Call:
## glm(formula = cartevp ~ age + anciente + mtrejet + nbopguic +
##      moycred3 + aveparmo + endette + engagemt + engagemc + engagemm +
##      nbcptvue + moysold3 + moycredi + agemvt + nbop + mtfactor +
##      mtvie + nbeparmo + nbeparlo + mteparlo + nblivret + mtlivret +
##      nbeparlt + nbeparte + nbpaiecb + nbcb + nbcbptar + avtscpte +
##      aveparfi, family = binomial(link = "logit"), data = dfco_app)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.168e+00  6.325e-01  -6.591 4.38e-11 ***
## age          1.723e-02  1.105e-02   1.559 0.119066
## anciente     -2.150e-03  9.122e-04  -2.357 0.018399 *
## mtrejet       8.668e-02  6.865e-02   1.263 0.206721
## nbopguic      2.527e-01  7.221e-02   3.500 0.000465 ***
```



```
## moycred3      -1.187e-03  3.430e-03  -0.346  0.729338
## aveparmo      7.968e-06  6.939e-06   1.148  0.250859
## endette      -1.617e-02  1.059e-02  -1.527  0.126700
## engagemt      1.340e-06  7.476e-07   1.792  0.073099 .
## engagemc      3.017e-05  1.279e-05   2.359  0.018315 *
## engagemm      9.895e-06  2.723e-06   3.634  0.000279 ***
## nbcptvue      4.499e-01  3.373e-01   1.334  0.182259
## moysold3      1.203e-05  5.617e-06   2.142  0.032185 *
## moycredi      2.916e-02  8.085e-03   3.606  0.000311 ***
## agemvt       -6.526e-03  9.397e-03  -0.694  0.487396
## nbop         -5.164e-02  1.028e-02  -5.023  5.10e-07 ***
## mtfactor      8.059e-06  3.035e-06   2.655  0.007935 **
## mtvie        -3.859e-06  2.340e-06  -1.649  0.099146 .
## nbeparmo      4.977e+13  5.599e+13   0.889  0.374032
## nbeparlo     -4.977e+13  5.599e+13  -0.889  0.374032
## mteparlo      2.599e-06  6.613e-06   0.393  0.694327
## nblivret     -4.977e+13  5.599e+13  -0.889  0.374032
## mtlivret      4.078e-06  6.675e-06   0.611  0.541284
## nbeparlt     -4.977e+13  5.599e+13  -0.889  0.374032
## nbeparte     -4.977e+13  5.599e+13  -0.889  0.374032
## nbpaiecb      1.689e-02  1.684e-02   1.003  0.315846
## nbcbb         2.531e+00  2.483e-01  10.194  < 2e-16 ***
## nbcbbptar    -3.097e+00  5.057e-01  -6.124  9.13e-10 ***
## avtschte     -9.273e-06  3.389e-06  -2.736  0.006220 **
## aveparfi      1.225e-05  4.280e-06   2.861  0.004224 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 990.45  on 771  degrees of freedom
## Residual deviance: 520.24  on 742  degrees of freedom
## AIC: 580.24
##
## Number of Fisher Scoring iterations: 25
Beaucoup de variables restent non significatives...
```

Effectuons une pénalisations LASSO

```
require(doMC)
```

```
## Loading required package: doMC
## Loading required package: foreach
## Loading required package: iterators
## Loading required package: parallel
```

```
registerDoMC(cores=2)
library(glmnet)
```

```
## Loading required package: Matrix
## Loaded glmnet 4.1-8
```

```
fit=cv.glmnet(as.matrix(dco_app),dca_app$cartevp,alpha=1,family="binomial",parallel=TRUE)
p=predict(fit, newx = as.matrix(dco_test),s="lambda.min",type ="class")
```

```
print(mean(p==dca_test$cartevp))
```

```
## [1] 0.84
```

Estimons maintenant une régression logistique sur les variables sélectionnées par la régression LASSO

```
mod6=glmnet(as.matrix(dco_app),dca_app$cartevp,alpha=1,family="binomial",lambda = fit$lambda.1se)
varindex=which((abs(mod6$beta)>0))
mod7=glm(cartevp~.,data=dfco_app[,c(varindex,ncol(dfco_app))],family=binomial(link = "logit"))
summary(mod7)
```

```
##
```

```
## Call:
```

```
## glm(formula = cartevp ~ ., family = binomial(link = "logit"),
```

```
##      data = dfco_app[, c(varindex, ncol(dfco_app))])
```

```
##
```

```
## Coefficients:
```

```
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.421e+00  2.493e-01 -13.718  < 2e-16 ***
## nbopguic     1.316e-01  4.492e-02   2.930  0.00339 **
## moycred3     1.603e-03  2.743e-03   0.584  0.55891
## engagemt     8.117e-07  5.476e-07   1.482  0.13826
## engagemc     2.005e-05  9.673e-06   2.073  0.03816 *
## engagemm     6.379e-06  2.327e-06   2.742  0.00611 **
## moycredi     8.834e-03  4.893e-03   1.805  0.07101 .
## nbcb         1.984e+00  1.858e-01  10.681  < 2e-16 ***
## nbcbptar    -2.798e+00  4.486e-01  -6.238  4.44e-10 ***
```

```
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
```

```
##      Null deviance: 990.45  on 771  degrees of freedom
```

```
## Residual deviance: 600.95  on 763  degrees of freedom
```

```
## AIC: 618.95
```

```
##
```

```
## Number of Fisher Scoring iterations: 6
```

```
p=predict(mod7,newdata = dco_test[,varindex],type = "response")
```

```
y_predit=rep("Cnon",length(p));y_predit[p>.5]="Coui"
```

```
print(mean(y_predit==dca_test$cartevp))
```

```
## [1] 0.82
```