Données breast tumors

Julien JACQUES

28 septembre, 2022

```
library(mixOmics)
data(breast.tumors)
X=breast.tumors$gene.exp
colnames(X)=breast.tumors$genes$name
Y=as.factor(breast.tumors$sample$treatment)
```

On va chercher à partir des niveaux d'expression de gènes à prédire si l'échantillon a été mesuré avant (BE) ou après (AF) le traitement.

Imputations des données manquantes

On va commencer par imputer les données manquantes par la moyenne des individus de la même classe

```
for (j in 1:ncol(X)){
  for (i in 1:nrow(X)){
    if (is.na(X[i,j])) X[i,j]=mean(X[Y==Y[i],j],na.rm=TRUE)
  }
}
```

Random Forest

```
library(randomForest)
```

Estimons un modèle random forest.

```
model <- randomForest(X,Y,ntree=2000,cutoff = c(.42,.58))
model$confusion
```

```
##    AF BE class.error
## AF 13  7   0.3500000
## BE  8 19   0.2962963
```

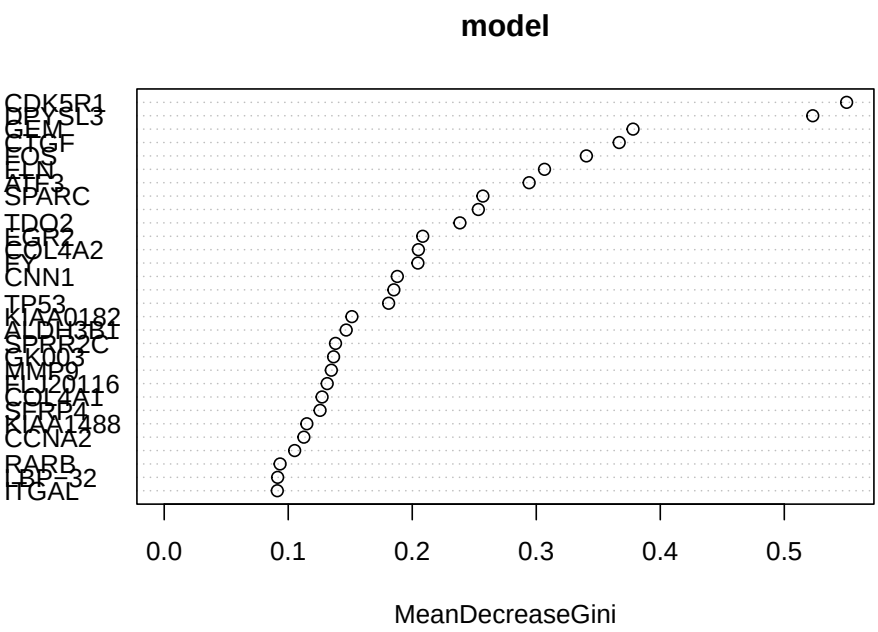
il y a une classe que l'on prédit plutôt bien (les BE), l'autre est par contre pas bien prédite du tout. Le taux de bon classement global est :

```
sum(diag(model$confusion))/sum(model$confusion)
```

```
## [1] 0.6716157
```

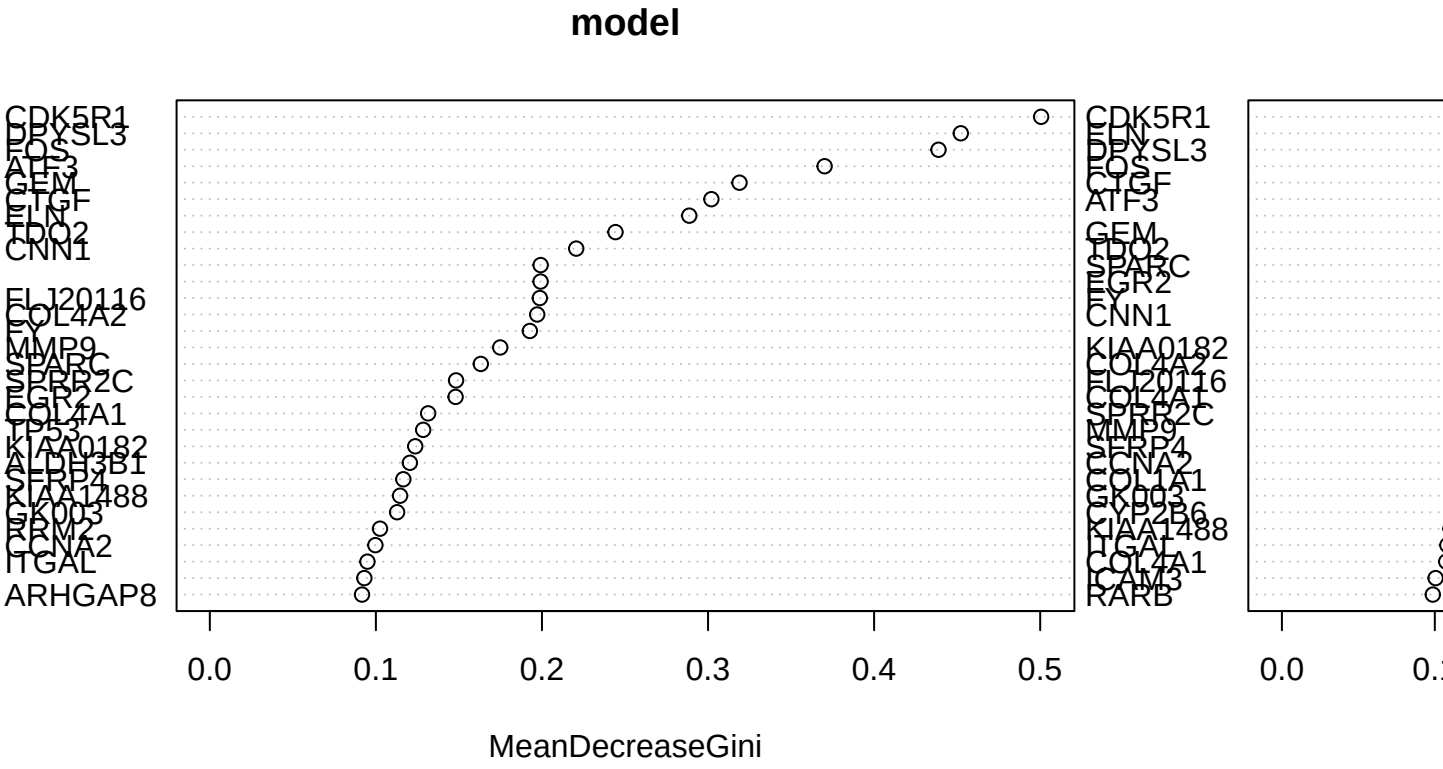
On peut représenter l'importance des variables, où l'on voit que quelques unes se détachent

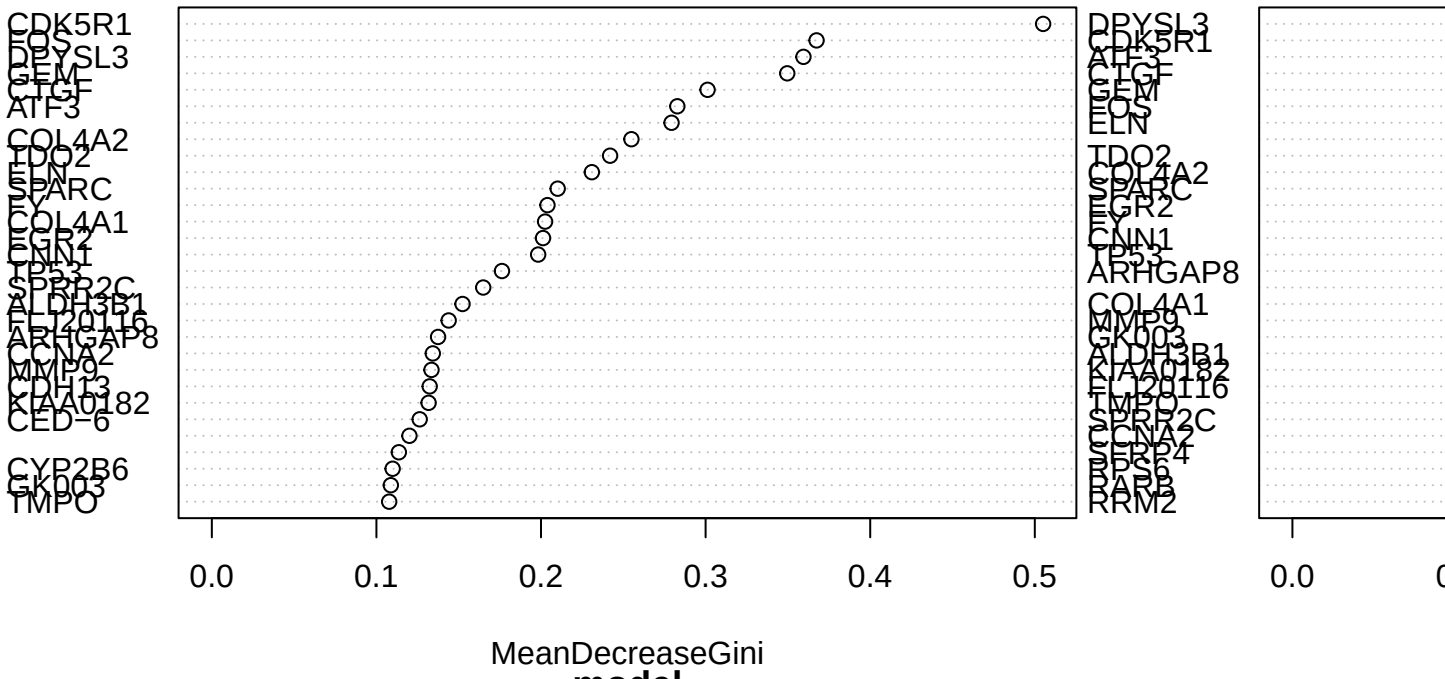
```
varImpPlot(model)
```

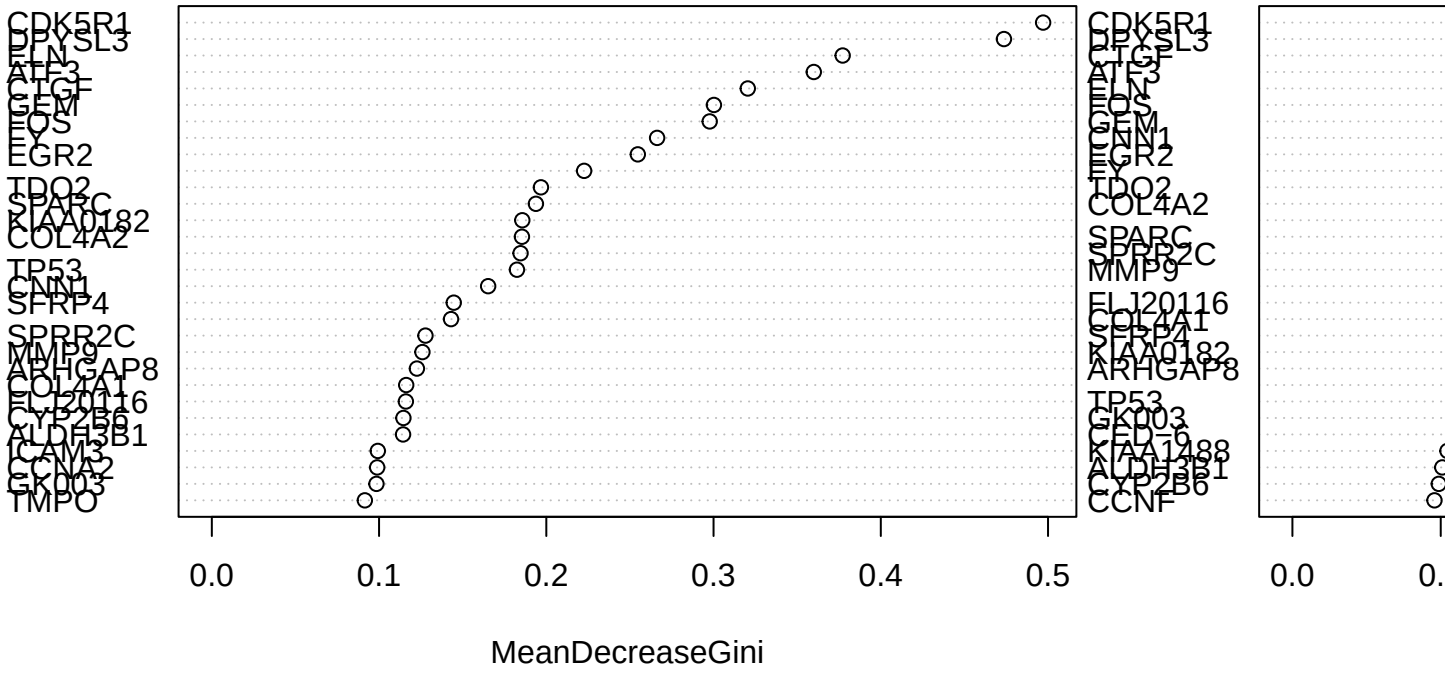


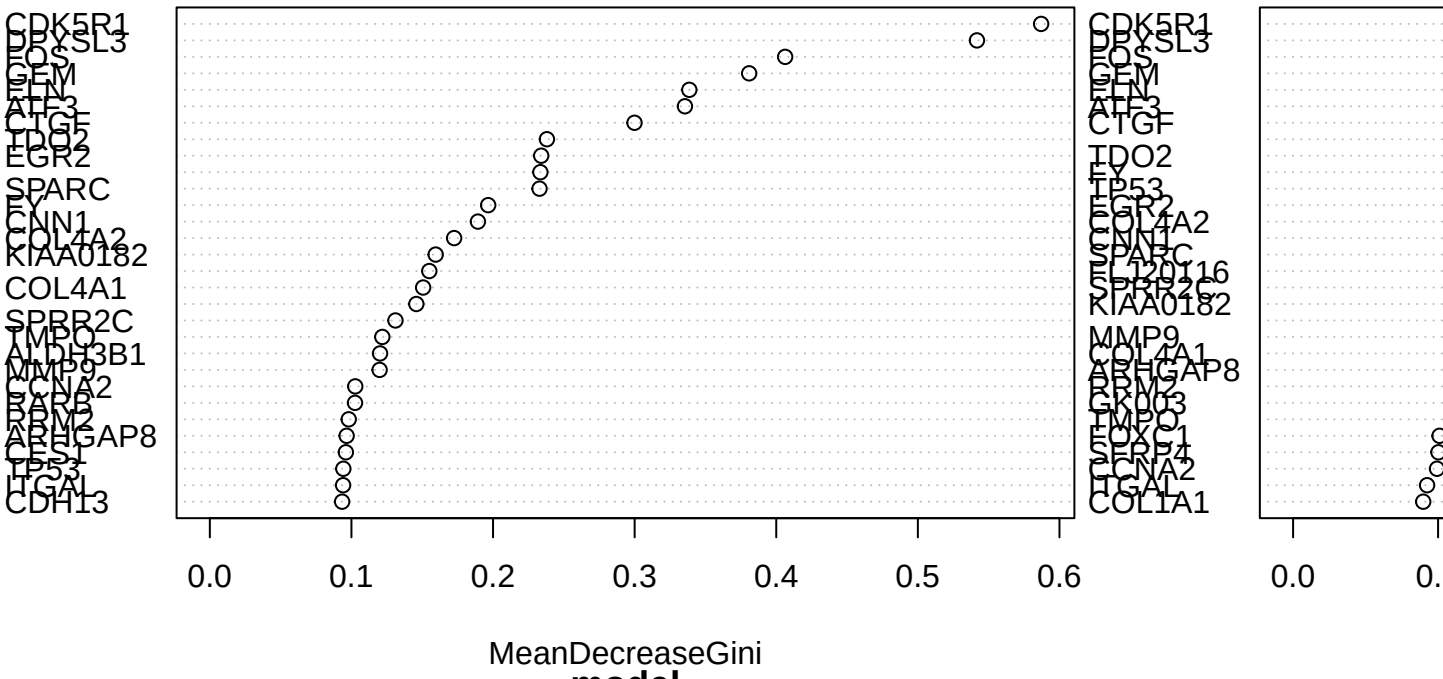
Pour évaluer la stabilité de l'importance des variables, je vais réaliser un certain nombre de forêt aléatoires, et je vais regarder si ce sont toujours les mêmes variables qui sont les plus importantes

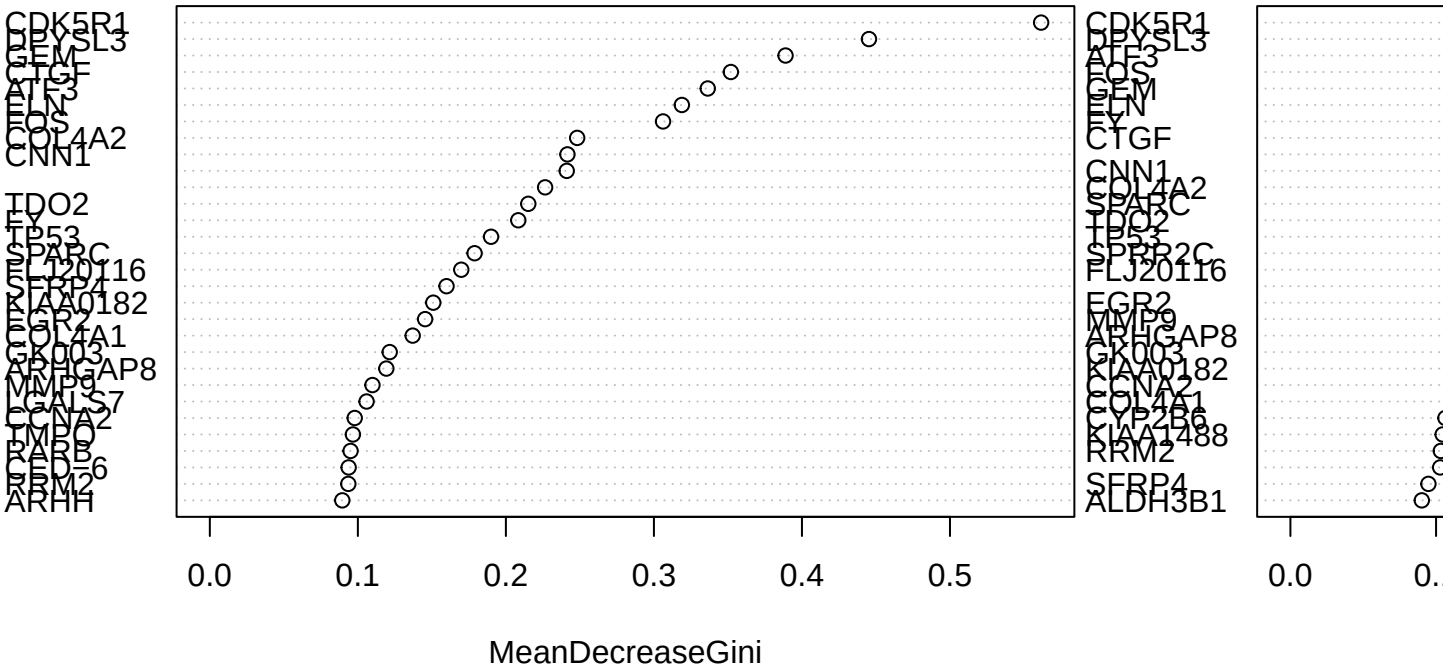
```
tmp2=NULL
for (i in 1:20){
  model <- randomForest(X,Y,ntree=2000,cutoff = c(.42,.58))
  tmp=varImpPlot(model)
  # je récupère les indices des variables dont l'importance est >0.2
  tmp2=c(tmp2,which(tmp>0.2))
}
```

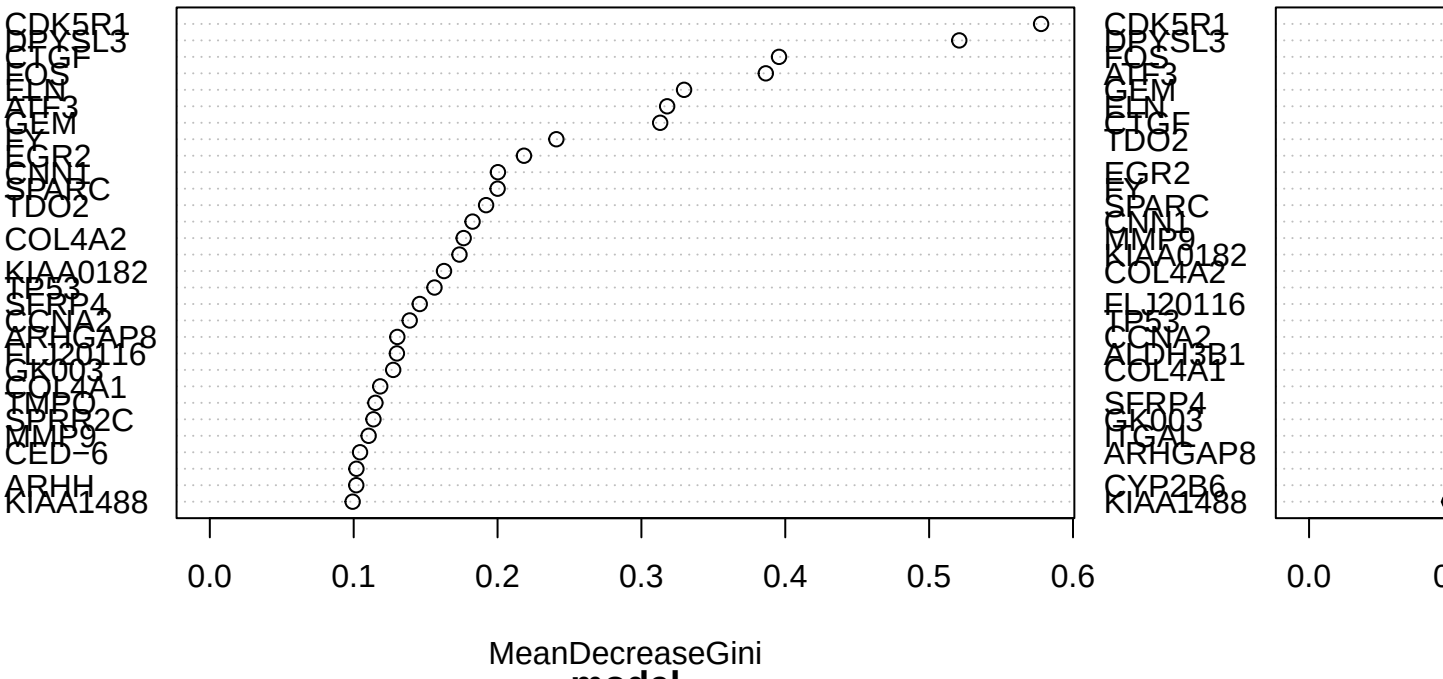


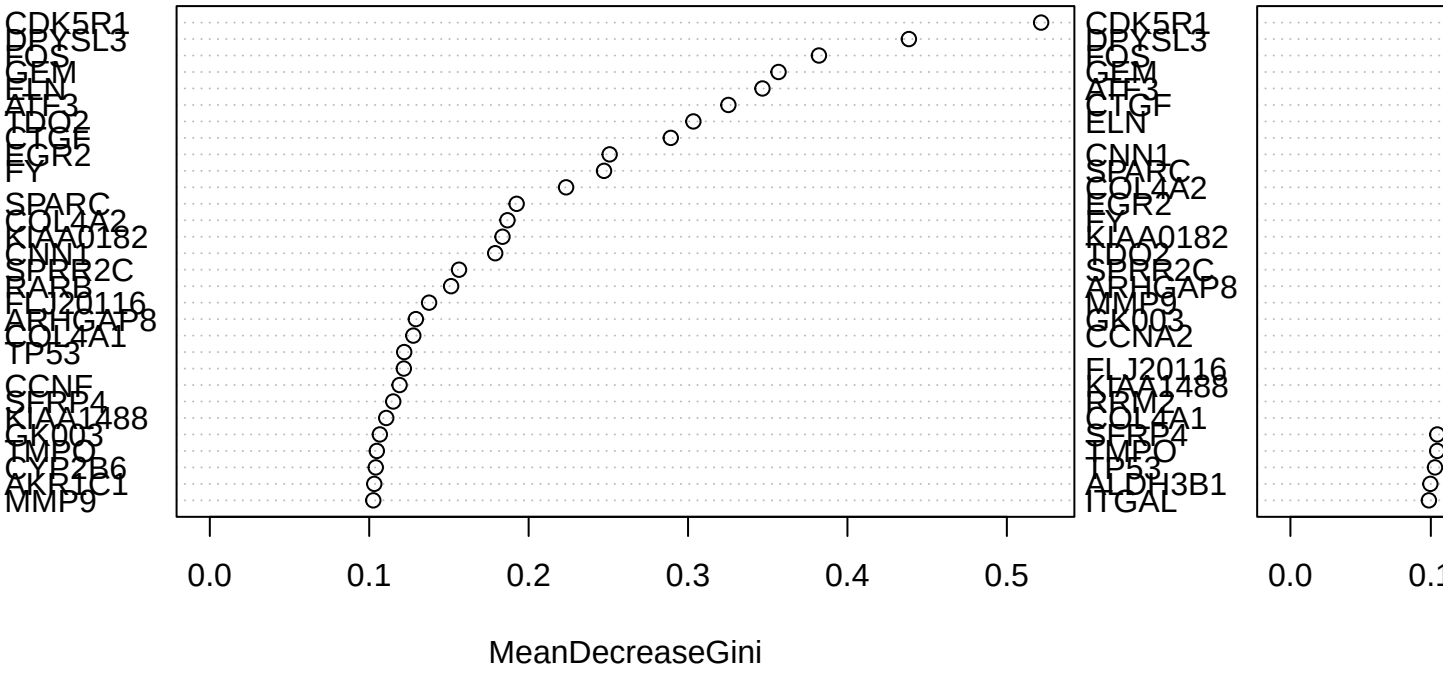

model
CDK5R1
EOS
DPYSL3
GEM
CTGF
ATF3
COL4A2
TDO2
ELN
SPARC
EY
COL4A1
EGR2
CNN1
TP53
SPRR2C
ALDH3B1
FLJ20116
ARHGAP8
CCNA2
MMP9
CDH13
KIAA0182
CED−6
CYP2B6
GK003
TMPO
DPYSL3
CDK5R1
ATF3
CTGF
GEM
EOS
ELN
TDO2
COL4A2
SPARC
EGR2
EY
CNN1
TP53
ARHGAP8
COL4A1
MMP9
GK003
ALDH3B1
KIAA0182
FLJ20116
TMPO
SPRR2C
CCNA2
SFRP4
RPS6
RARB
RRM2
0.0
0.1
0.2
0.3
0.4
0.5
MeanDecreaseGini

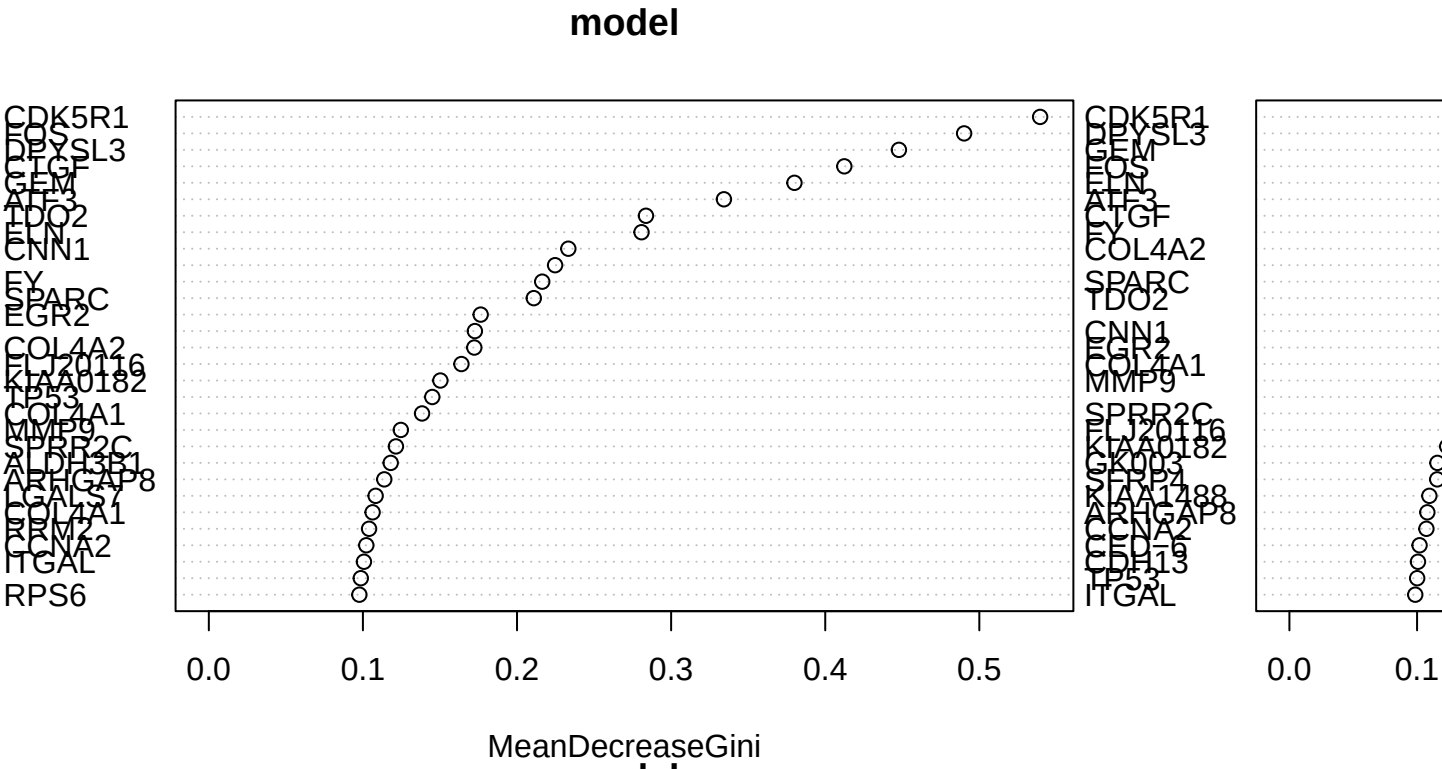

model
CDK5R1
DPYSL3
ELN
ATF3
CTGF
GEM
EOS
EY
EGR2
TDO2
SPARC
KIAA0182
COL4A2
TP53
CNN1
SFRP4
SPRR2C
MMP9
ARHGAP8
COL4A1
FLJ20116
CYP2B6
ALDH3B1
ICAM3
CCNA2
GK003
TMPO
CDK5R1
DPYSL3
CTGF
ATF3
ELN
EOS
GEM
CNN1
EGR2
EY
TDO2
COL4A2
SPARC
SPRR2C
MMP9
FLJ20116
COL4A1
SFRP4
KIAA0182
ARHGAP8
TP53
GK003
CED−6
KIAA1488
ALDH3B1
CYP2B6
CCNF
0.0
0.1
0.2
0.3
0.4
0.5
MeanDecreaseGini

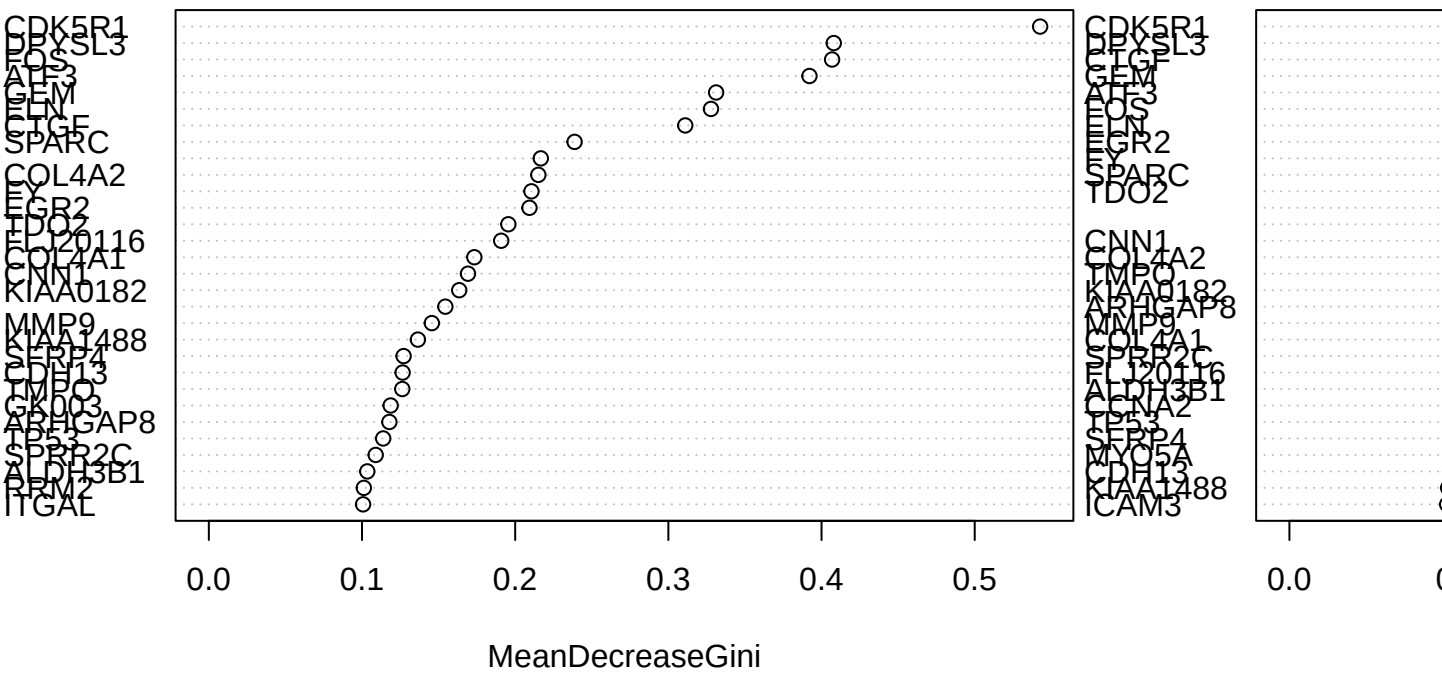

model
CDK5R1
DPYSL3
FOS
GEM
ELN
ATF3
CTGF
IDO2
EGR2
SPARC
FY
CNN1
COL4A2
KIAA0182
COL4A1
SPBB2C
TMPO
ALDH3B1
MMP9
CCNA2
RARB
RRM2
ARHGAP8
CES1
TP53
ITGAL
CDH13
0.0
0.1
0.2
0.3
0.4
0.5
0.6
MeanDecreaseGini

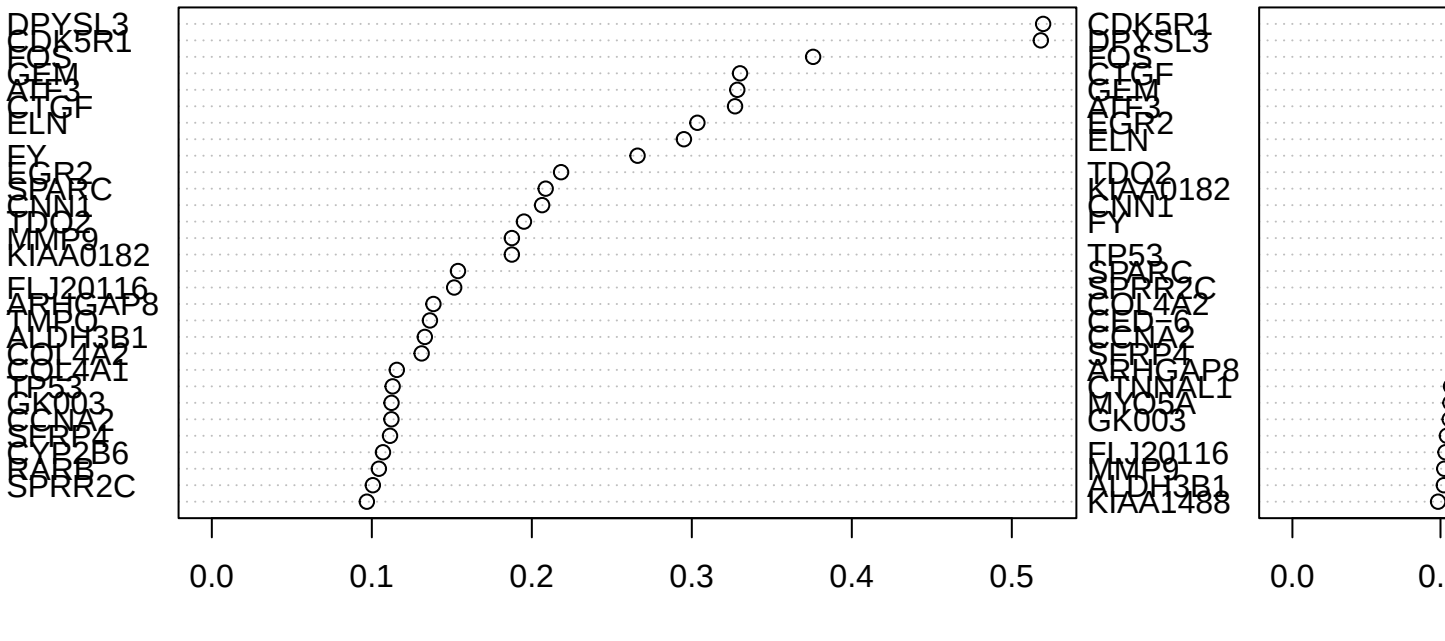

model
CDK5R1
DPYSL3
GEM
CTGF
ATF3
ELN
FOS
COL4A2
CNN1
IDO2
FY
TP53
SPARC
FLJ20116
SFRP4
KIAA0182
EGR2
COL4A1
GK003
ARHGAP8
MMP9
LGALS7
CCNA2
TMPO
RARB
CES6
RRM2
ARHH
0.0
0.1
0.2
0.3
0.4
0.5
MeanDecreaseGini


model
CDK5R1
DPYSL3
CTGF
EOS
ELN
ATF3
GEM
FY
EGR2
CNN1
SPARC
TDO2
COL4A2
KIAA0182
TP53
SERP4
CCNA2
ARHGAP8
FLJ20116
GK003
COL4A1
TMPO
SPRR2C
MMP9
CED−6
ARHH
KIAA1488
0.0
0.1
0.2
0.3
0.4
0.5
0.6
MeanDecreaseGini


model
CDK5R1
DPYSL3
EOS
GEM
ELN
ATF3
TDO2
CTGF
EGR2
FY
SPARC
COL4A2
KIAA0182
CNN1
SPRR2C
RARB
FLJ20116
ARHGAP8
COL4A1
TP53
CCNF
SERP4
KIAA1488
GK003
TMPO
CYP2B6
AKR1C1
MMP9
0.0
0.1
0.2
0.3
0.4
0.5
MeanDecreaseGini


model
CDK5R1
EOS
DPYSL3
CTGF
GEM
ATF3
TDO2
ELN
CNN1
EY
SPARC
EGR2
COL4A2
FLJ20116
KIAA0182
TP53
COL4A1
MMP9
SPRR2C
ALDH3B1
ARHGAP8
LGALS7
COL4A1
RRM2
CCNA2
ITGAL
RPS6
0.0
0.1
0.2
0.3
0.4
0.5
MeanDecreaseGini


model
CDK5R1
DPYSL3
EOS
ATF3
GEM
ELN
CTGF
SPARC
COL4A2
EY
EGR2
TDO2
FLJ20116
COL4A1
CNN1
KIAA0182
MMP9
KIAA1488
SFRP4
CDH13
TMPO
GK003
ARHGAP8
TP53
SPRR2C
ALDH3B1
RRM2
ITGAL
0.0
0.1
0.2
0.3
0.4
0.5
MeanDecreaseGini



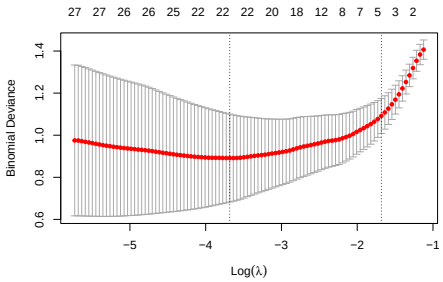

```
# je résume les indices dans variables importantes dans un tableau de contingence
tab=table(tmp2)
# je remplace ces indices par le nom des variables correspondantes
names(tab)=colnames(X)[as.integer(names(table(tmp2)))]
# j'affiche le tableau des noms des variables importantes par ordre décroissant
sort(tab,decreasing = T)
```

```
##      CTGF       GEM      ATF3   DPYSL3       FOS       ELN   CDK5R1
##        20        20        20        20        20        20        17
##        FY     TDO2     EGR2     CNN1    SPARC   COL4A2           COL4A1
##        13        12        11         9         9         7         2         1
## KIAA0182
##         1
```

On arrive à identifier un petit pool de variables qui sont à chaque fois parmi les variables les plus importantes dans les différentes forêts aléatoires construites : CTGF, GEM, ATF3, DPYSL3, FOS, ELN, CDK5R1.

Régression logistique

```
library(glmnet)
require(doMC)
registerDoMC(cores=4)
cvfit=cv.glmnet(X,Y,alpha=1,family="binomial",parallel=TRUE)
plot(cvfit)
```




```
fit=glmnet(X,Y,alpha=1,lambda=cvfit$lambda.1se,family="binomial")
# on peut regarder la performance de classement (ici optimiste car sur l'échnatillon d'apprentissage)
pfit = predict(fit, X, s = cvfit$lambda.1se, type = "response")
table(pfit>.5, Y)
```

```
##         Y
##          AF BE
##   FALSE 15  0
##   TRUE   5 27
```

```
# examinons les variables sélectionnées
fit$beta[fit$beta@i+1]
```

```
## [1] -0.48013194 -0.01699264 -0.38990702 -0.06232503 -0.15805861
```

```
fit$beta@Dimnames[[1]][fit$beta@i+1]
```

```
## [1] "CTGF"   "FY"     "GEM"    "FOS"    "CDK5R1"
```

On peut, comme pour les forêts aléatoires, ré-exécuter plusieurs fois le choix de lambda (qui peut varier du fait du découpage des folds en validation croisée)

```
numvar=NULL
for (i in 1:20){
  cvfit=cv.glmnet(X,Y,alpha=1,family="binomial",parallel=TRUE)
  fit=glmnet(X,Y,alpha=1,lambda=cvfit$lambda.1se,family="binomial")
  numvar=c(numvar,fit$beta@i+1)
}
tab=table(numvar)
names(tab)=fit$beta@Dimnames[[1]][as.integer(names(tab))]
# j'affiche le tableau des noms des variables importantes par ordre décroissant
sort(tab,decreasing = T)
```

```
##   CTGF    GEM CDK5R1    FOS     FY  PDGFB   ODC1 COL4A1
##     20     20     19     11      7      1      1      1
```

Là encore un pool de variables ressort, relativement cohérent avec ce que l'on a obtenu avec les forêts aléatoires.

En croisant ainsi plusieurs exécution de l'estimation de nos modèles, cela permet d'avoir une idée assez sûre des variables les plus importantes