Data Mining 2: Modélisation statistique et apprentissage **G. Cohen**

Travaux pratiques 03 février 2018

ETheme 3 Arbres de décision

1 Données du syndrome de Cushing

Trouble de l'hypertension associée à la sécrétion excessive de cortisol par la glande surrénale. Les observations sont les taux d'excrétion urinaire de deux métabolites de stéroïdes. Le data frame Cushings (dans library MASS) a 27 lignes e 3 colonnes:

- Tetrahydrocortisone: taux d'excrétion urinaire (mg/24hr).
- Pregnanetriol: taux d'excrétion urinaire (mg/24hr).
- Type: type du syndrome sous-jacent
  - (adenome)
  - (hyperplasie bilatérale)
  - (carcinome)
  - inconnu

```
rm(list = ls(all = TRUE))
library(rpart)
library(MASS)
#Cushings data
data(Cushings)
cush <- Cushings[Cushings$Type!="u",]
cush$Type<-factor(cush$Type)
cush[,1:2] <- log(cush[,1:2])
```

On rappelle que le découpage est basé sur une mesure de l'impureté, et que l'élagage est basé sur un autre critère , $R\alpha(T) = R(T) + \alpha \times size(T)$ par exemple. Le paramètre CP de rpart est $\alpha/R(T)$ où T est l'arbre que l'on élague. Dans notre cas on va découper sur la décroissance de l'index de Gini, et élaguer en utilisant le taux de mals classés On utilise la fonction *rpart()*. On demande un arbre de classification et puisque l'ensemble de données est petit on autorise des découpages de feuilles avec 5 ou plus vecteurs de

données dans chaque classe non vide (minsplit=5, et on peut faire un retour sur l'élagage si on a exagéré le découpage). La fonction rpart réalise une V-validation croisée, dans le cadre de son analyse, on spécifie donc V (xval).

```
cush.trer<-rpart(Type~Tetrahydrocortisone+Pregnanetriol,cush,
cp=-Inf,control = rpart.control(minsplit=2,xval=10),method="class")
```

On utilise des outils de visualisations

```
windows(); plot(cush.trer); text(cush.trer);
```

On examine la sortie

```
summary(cush.trer)
```

et maintenant l'élagage. On élague au plus petit arbre avec xerror entre 1 std. dev. de la valeur de CP qui donne la xerror minimum atteinte.

```
(cush.cpt<-printcp(cush.trer))
windows(); plotcp(cush.trer)
cush.pt<-prune(cush.trer,cp=0.07)
windows(); plot(cush.pt); text(cush.pt);
```

On peut alors prédire les classes (ou vérifier les classes de l'ensemble d'apprentissage)

```
cush.ctp<-predict(cush.pt)
```

Exercice: Comment est estimée $p(c|x)$? Verifier que la fréquence des classes aux feuilles détermine $p(c|x)$:

```
predict(cush.pt,data.frame(Tetrahydrocortisone=1.6,Pregnanetriol=0.7))
```

A la feuille Tetrahydrocortisone>= 1.575727, Pregnanetriol>= 0.6931472 on 6 exemples dont 5 sont de type 'c' et 1 de type 'b'. L'erreur tend vers 0 au fur et à mesure que l'on découpe (et on force le découpage même pour les feuilles avec relativement peu d'exemples). L'erreur x-val identifie l'overfit.

```
windows();
par(xaxt="n")
plot(1:nrow(cush.cpt),cush.cpt[,3],type='l',xlab="CP",ylab="error,xerror")
par(xaxt="s")
points(1:nrow(cush.cpt),cush.cpt[,4],type='b')
axis(1, at = 1:nrow(cush.cpt), labels = formatC(cush.cpt[,1], format="fg"))
axis(3, at = 1:nrow(cush.cpt), labels = formatC(cush.cpt[,2]+1, format="fg"))
```

Exercice: les plotcp x-labels correspondent à la moyenne géométrique des cp-values adjacentes. Pourquoi?
On peut réaliser les frontières de décision en prédisant les types pour un treillis de coté m

```
m<-100
x<-seq(0,4,length.out=m)
y<-seq(-3,2.5,length.out=m)
z<-data.frame(expand.grid(Tetrahydrocortisone=x,Pregnanetriol=y))
plot(cush[,1:2],pch=as.character(cush$Type))
```

Tracer les données et la frontière de décision. Les classes sont a,b,c =1,2,3 donc paramétrer les contours à 1.5 et 2.5. Pour l'arbre non élagué:

```
cush.trerb<-predict(cush.trer,z)
contour(x,y,matrix(max.col(cush.trerb),m,m),levels=c(1.5,2.5),
add=T,d=F,lty=1,col=2)
```

Pour l'arbre élagué:

```
cush.ptb<-predict(cush.pt,z)
contour(x,y,matrix(max.col(cush.ptb),m,m),levels=c(1.5,2.5),
add=T,d=F,lty=1,col=3)
```

L'impureté des critères de l'entropie et de Gini donnent ici les mêmes découpages:

```
cush.tre<-rpart(Type~Tetrahydrocortisone+Pregnanetriol,cush,
cp=-Inf,control = rpart.control(minsplit=2,xval=10),
method="class",parms = list(split="information"))
plot(cush.tre); text(cush.tre);
printcp(cush.tre)
```

2 Jeux de données Verres Forensic

Mesures (indice de réfraction et pour cent en poids d'oxydes de Na, Mg, Al, Si, K, Ca, Ba, et Fe) sur 214 fragments de verres. Type de verre

1. Window foloat glass (70)
2. Window non-float glass (76)
3. Vehicle window glass (17)

4. Containers (13)

5. Tableware (9)

6. Vehicle headlamps (29)

Maintenant, comme exercice, refaire cette analyse pour les données et decider combien de noeuds élaguer.

```
data(fgl)
names(fgl)[10]<-"Type"
fgl.trer<-rpart(Type~.,fgl,
cp=-Inf,control = rpart.control(minsplit=2,xval=10),method="class")
```

Afficher l'arbre non élagué

```
windows(); plot(fgl.trer); text(fgl.trer);
(fgl.xv<-printcp(fgl.trer))
```

Dessiner les graphiques de xerror et de l'erreur apparente

```
windows(); plotcp(fgl.trer)
windows();
par(xaxt="n")
plot(1:nrow(fgl.xv),fgl.xv[,3],type='l',xlab="CP",ylab="error,xerror")
par(xaxt="s")
points(1:nrow(fgl.xv),fgl.xv[,4],type='b')
axis(1, at = 1:nrow(fgl.xv), labels = formatC(fgl.xv[,1], format="fg"))
axis(3, at = 1:nrow(fgl.xv), labels = formatC(fgl.xv[,2]+1, format="fg"))
```

Elaguer,

```
fgl.pt<-prune(fgl.trer,cp=0.018)
and plot the pruned tree
windows(); plot(fgl.pt); text(fgl.pt);
Predict class (or check classes on training data)
predict(fgl.pt)
table(fgl$Type,predict(fgl.pt,type="class"))
```