Scoring & Applications – Chapitre 3 – Exercices

Exercice 1.

Les clients d'une agence se répartissent en deux classes de risque de même poids. Table 1 donne la moyenne et la variance des découverts annuels observés dans chacune des classes.

		classe	
		1	2
découvert	*moyenne*	$1,5$	$2,5$
	variance	$3,0$	$3,2$

Table 1: *moyenne et variance du découvert dans chaque classe de clients*

Le découvert annuel de M. Li vaut : $1,9$.

1. A quelle classe M. Li serait-il affecté par (i) un classifieur gaussien homoscédastique, (ii) un classifieur gaussien hétéroscédastique ?
2. Estimez l'erreur de classement théorique associée à chacun des deux classifieurs.

Exercice 2.

Table 2 indique le flux et la solvabilité de dix clients bancaires.

client	1	2	3	4	5	6	7	8	9	10
flux	2	4	6	5	4	7	8	6	5	6
solvable	N	N	N	N	O	O	O	O	O	O

Table 2: *flux (k€) et solvabilité (oui(O)/non(N)) de dix clients bancaires*

Estimez l'erreur de classement d'un classifieur gaussien (i) par resubstitution (ii) par bootstrap (iii) par Leave One Out (iv) par v-fold Cross Validation.

Exercice 3.

Table 3 donne la classe vraie et les probabilités conditionnelles obtenues sur un échantillon de test de dix sujets grâce à un classifieur basé sur un modèle paramétrique.

client	1	2	3	4	5	6	7	8	9	10
classe vraie	2	2	2	2	1	1	1	1	1	1
t_1	$0,27$	$0,18$	$0,56$	$0,61$	$0,09$	$0,27$	$0,96$	$0,34$	$0,69$	$0,13$
t_2	$0,73$	$0,82$	$0,44$	$0,39$	$0,91$	$0,73$	$0,04$	$0,66$	$0,31$	$0,87$

Table 3: *partition (classe vraie) et probabilités conditionnelles (t_1: probabilité d'appartenir à la classe 1, t_2: probabilité d'appartenir à la classe 2) obtenues sur un échantillon de test de dix clients par un classifieur paramétrique*

Tracez la courbe ROC du classifieur et calculez AUC.

Exercice 4.

M. Shan souhaite modéliser les données de Table 2 par un modèle gaussien à deux classes. Lequel du modèle hétéroscédastique ou homoscédastique est le meilleur selon BIC ?

Exercice 5.

On considère les données du fichier `client`[a]. Comparez le classifieur gaussien homoscédastique et le classifieur gaussien hétéroscédastique selon (i) le critère BIC (ii) la courbe ROC et AUC (iii) l'erreur de classement théorique (iv) l'erreur de classement apparente (v) l'erreur de classement estimée par bootstrap (vi) l'erreur de classement estimée par Leave One Out (vi) l'erreur de classement estimée par v-fold Cross Validation.

A. Lourme, Faculté d'économie, gestion & AES, Université de Bordeaux `http://alexandrelourme.free.fr`

[a]disponible sous : `http://alexandrelourme.free.fr/M2IREF/SCORING/client.csv`

Exercice 2.

```
# une fonction qui calcule l'erreur de classement du classifieur gaussien homoscédastique
library(Rmixmod)
ergauss <- function(train,trainlab,test,testlab){
  trainlab=as.factor(trainlab)
  testlab=as.factor(testlab)
  yourmodel=mixmodGaussianModel(listModels=c("Gaussian_pk_L_C"))
  out <- mixmodLearn(train, knownLabels=trainlab, models = yourmodel)
  pred <- mixmodPredict(data = test, classificationRule = out["bestResult"])
  erg = 1-sum(pred[5]==as.numeric(testlab))/length(test) # taux d'erreur sur les données de t
}

# les données
mydata <- data.frame(
  flux=c(2,4,6,5,4,7,8,6,5,6),
  solvable=c('N','N','N','N','O','O','O','O','O','O')
)
attach(mydata)
n=nrow(mydata)

############################
# estimation de l'erreur par resubstitution (erreur apparente)
############################

epsilon_ap=ergauss(flux,solvable,flux,solvable)

cat('epsilon_ap :',epsilon_ap,'\n')

## epsilon_ap : 0.3

##############################################
# estimation de l'erreur par Leave One Out (ou Cross Validation)
##############################################

comp=0

for (i in 1:n){
  comp=comp+ergauss(flux[-i],solvable[-i],flux[i],solvable[i])
}
epsilon_cv=comp/n

cat('epsilon_cv :',epsilon_cv,'\n')

## epsilon_cv : 0.6

##############################################
# estimation de l'erreur par v-fold Cross Validation
##############################################

K=5 # nbre de sous échantillons

epsilonlist <- NULL

list=sample(1:n,n,replace=FALSE)

for (j in 1:K){
  train <- mydata[-list[((2*j-1):(2*j))],] # échantillon d'apprentissage
  test <- mydata[list[((2*j-1):(2*j))],] # échantillon de test
  epsilonlist[j]=ergauss(train$flux,train$solvable,test$flux,test$solvable)
```

```
}

epsilon_vfoldcv = mean(epsilonlist)

cat('epsilon_vfoldcv :',epsilon_vfoldcv,'\n')

## epsilon_vfoldcv : 0.7

##############################################
# estimation de l'erreur par bootstrap
##############################################

beta <- NULL

N=20

#for (j in 1:N){
#  select <- sample(1:n,n,replace=TRUE)
#  locdata <- mydata[select,] # n-échantillon tiré des données avec de possibles répétitions
#  epsilonj=ergauss(locdata$flux,locdata$solvable,locdata$flux,locdata$solvable)
#  epsilon=ergauss(locdata$flux,locdata$solvable,mydata$flux,mydata$solvable)
#  beta[j]=epsilonj-epsilon
#}

#epsilon_boot = epsilon_ap + mean(beta,na.rm=TRUE)

#cat('epsilon_boot :',epsilon_boot,'\n')
```

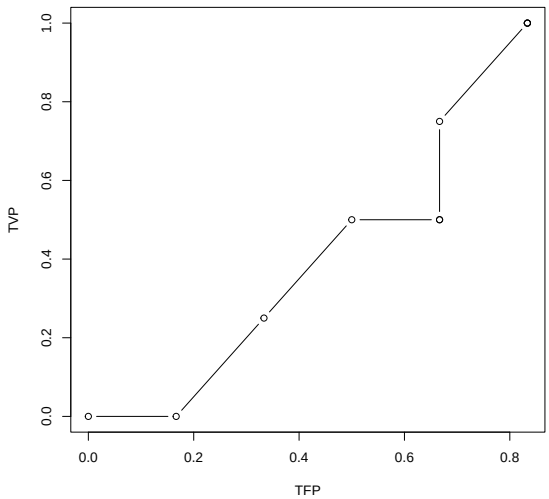
Exercice 3.

```
# les données
t <- data.frame(
t1=c(0.27,0.18,0.56,0.61,0.09,0.27,0.96,0.34,0.69,0.13)
)
t$t2=1-t$t1 # probabilités d'appartenir à la classe 2

compare <- data.frame(
verite=c(2,2,2,2,1,1,1,1,1,1)
)
compare$decision=compare$verite

TFP <- TVP <- NULL
for (ii in 0:10)
{
compare$decision[t$t2>ii/10]=2
compare$decision[t$t2<=ii/10]=1
verite <- factor(compare$verite,levels=c(1:2))
decision <- factor(compare$decision,levels=c(1:2))
confusion <- table(verite,decision)
TFP[ii]=confusion[1,2]/(confusion[1,1]+confusion[1,2])
TVP[ii]=confusion[2,2]/(confusion[2,1]+confusion[2,2])
cat('-------','\n')
cat('seuil pour la classe 2 :',ii/10,'\n')
print(confusion)
cat('TFP :',TFP[ii],'\n')
cat('TVP :',TVP[ii],'\n')
}

## -------
## seuil pour la classe 2 : 0
##       decision
## verite 1 2
```

```
##      1 0 6
##      2 0 4
## TFP :
## TVP :
## -------
## seuil pour la classe 2 : 0.1
##       decision
## verite 1 2
##      1 1 5
##      2 0 4
## TFP : 0.8333333
## TVP : 1
## -------
## seuil pour la classe 2 : 0.2
##       decision
## verite 1 2
##      1 1 5
##      2 0 4
## TFP : 0.8333333
## TVP : 1
## -------
## seuil pour la classe 2 : 0.3
##       decision
## verite 1 2
##      1 1 5
##      2 0 4
## TFP : 0.8333333
## TVP : 1
## -------
## seuil pour la classe 2 : 0.4
##       decision
## verite 1 2
##      1 2 4
##      2 1 3
## TFP : 0.6666667
## TVP : 0.75
## -------
## seuil pour la classe 2 : 0.5
##       decision
## verite 1 2
##      1 2 4
##      2 2 2
## TFP : 0.6666667
## TVP : 0.5
## -------
## seuil pour la classe 2 : 0.6
##       decision
## verite 1 2
##      1 2 4
##      2 2 2
## TFP : 0.6666667
## TVP : 0.5
## -------
## seuil pour la classe 2 : 0.7
##       decision
## verite 1 2
##      1 3 3
##      2 2 2
## TFP : 0.5
## TVP : 0.5
## -------
```

```
## seuil pour la classe 2 : 0.8
##       decision
## verite 1 2
##      1 4 2
##      2 3 1
## TFP : 0.3333333
## TVP : 0.25
## -------
## seuil pour la classe 2 : 0.9
##       decision
## verite 1 2
##      1 5 1
##      2 4 0
## TFP : 0.1666667
## TVP : 0
## -------
## seuil pour la classe 2 : 1
##       decision
## verite 1 2
##      1 6 0
##      2 4 0
## TFP : 0
## TVP : 0
```

```
plot(TFP,TVP,'b',xlab='TFP',ylab='TVP')
```




```
library(zoo)
id <- order(TFP)
AUC <- sum(diff(TFP[id])*rollmean(TVP[id],2)) # area under ROC curve
print(AUC)
```

```
## [1] 0.3125
```

Exercice 4.

Modèle gaussien hétéroscédastique

```
# BIC du modèle gaussien hétéroscédastique avec Rmixmod
library(Rmixmod)
model=mixmodGaussianModel(listModels=c("Gaussian_pk_Lk_Ck")) # modèle gaussien hétéroscédasti
out <- mixmodLearn(flux,knownLabels=as.factor(solvable),models=model,criterion=c('BIC'))
bic=-out[9][[1]][3]/2
cat('BIC :',bic,'\n')
```

```
## BIC : -29.77396

# BIC du modèle gaussien hétéroscédastique à la main
n=nrow(mydata)
n1=sum(solvable=='O') # nbre de solvables
pi1=n1/n # proportion de solvables
m1=mean(flux[solvable=='O']) # moyenne du flux chez les solvables
v1=var(flux[solvable=='O'])*(n1-1)/n1 # variance du flux chez les solvables
n2=sum(solvable=='N') # nbre de non solvables
pi2=n2/n # proportion de non solvables
m2=mean(flux[solvable=='N']) # moyenne du flux chez les non solvables
v2=var(flux[solvable=='N'])*(n2-1)/n2 # variance du flux chez les non solvables
loglik = 0
for (i in 1:n){loglik=loglik+
(solvable[i]=='O')*(log(pi1)+log(dnorm(flux[i],m1,sqrt(v1))))+
(solvable[i]=='N')*(log(pi2)+log(dnorm(flux[i],m2,sqrt(v2))))
}
eta=1+2+2
bic=loglik-eta/2*log(n)
cat('BIC :',bic,'\n')

## BIC : -29.77396
```

Modèle gaussien homoscédastique

```
# BIC du modèle gaussien homoscédastique avec Rmixmod
library(Rmixmod)
model=mixmodGaussianModel(listModels=c("Gaussian_pk_L_C")) # modèle gaussien homoscédastique
out <- mixmodLearn(flux,knownLabels=as.factor(solvable),models=model,criterion=c('BIC'))
bic=-out[9][[1]][3]/2
cat('BIC :',bic,'\n')

## BIC : -28.66772

# BIC du modèle gaussien homoscédastique à la main
n=nrow(mydata)
m=pi1*m1+pi2*m2 # moyenne du flux
v=pi1*v1+pi2*v2 # estimateur MV de la variance du flux
loglik = 0
for (i in 1:n){loglik=loglik+
(solvable[i]=='O')*(log(pi1)+log(dnorm(flux[i],m1,sqrt(v))))+
(solvable[i]=='N')*(log(pi2)+log(dnorm(flux[i],m2,sqrt(v))))
}
eta=1+2+1
bic=loglik-eta/2*log(n)
cat('BIC :',bic,'\n')

## BIC : -28.66772
```

D'après BIC, le modèle Gaussien homoscédastique est meilleur que le modèle Gaussien homoscédastique.