Journée de formation : Statistique en Grande dimension

Le but de ce TP n'est pas de comprendre les instructions proposées dans le logiciel statistique R mais d'observer les différents résultats obtenus par l'exécution de ces instructions.
Comme application, nous nous intéressons au probème de prédiction de l'efflorescence algale.

1. Présentation générale du problème (source : wikipedia.org)

Une efflorescence algale est une augmentation relativement rapide de la concentration d'une (ou de quelques) espèce(s) de phytoplancton dans un système aquatique. Cette augmentation de concentration se traduit généralement par une coloration de l'eau (rouge, brun-jaune ou vert). Ce phénomène peut concerner des eaux douces ou marines. Le phénomène peut être naturel ou favorisé par des pollutions terrigènes. Dans ces derniers cas, des proliférations intenses et longues peuvent conduire à des zones mortes, en raison d'une consommation de la totalité de l'oxygène dissout dans l'eau la nuit et/ou d'émissions de toxines par certaines espèces de plancton. Les efflorescences algales peuvent alors localement déséquilibrer la chaîne alimentaire, voire entraîner des déséquilibres écologiques plus durables (émissions de gaz à effet de serre, mortalité de poissons et crustacés), sur de vastes zones (la plus grande a atteint 22 000 km^2 en 2007, au large de l'estuaire du Mississippi).

2. Données

On dispose d'un base de données constituée de 184 observations. Chaque observation est le fruit de l'agrégation de plusieurs mesures dans différentes eaux d'une même rivière sur une période de 3 mois durant une même saison. Chaque observation contient des informations sur 11 variables. Parmi celles-ci 3 sont nominales et décrivent la saison de l'année durant laquelle l'échantillon a été collecté, la taille et la vitesse de la rivière en question. Les 8 autres variables contiennent les valeurs de paramètres chimiques mesurés dans les eaux prélevées et sont :

- La valeur maximal du pH,
- La teneur minimal en O_2 (oxygène),
- La teneur moyenne en Cl (chloride),
- La teneur moyenne en NO_3^- (nitrate),
- La teneur moyenne en NH_4^+ (ammoniac),
- La teneur moyenne en PO_4^{3-} (orthophosphate),
- La teneur moyenne en PO_4 (phosphate),
- La teneur moyenne en Chlorophylle.

Associé à chacun de ces paramètres, nous avons la fréquence d'apparition d'une algue trouvée dans les différentes eaux prélevées. Aucune information sur le nom de cette algue ne permet de l'identifier.

3. Charger, résumer et visualiser les données

Suivre les instructions suivantes :

(1) Aller à la page web
http ://perso-math.univ-mlv.fr/users/hebiri.mohamed/ à la rubrique 'Enseignements' et enregistrer les données nommées 'algue.RData' sur votre espace de travail.

(2) charger ces données et en faire un affichage succinct :

```
load('algue.RData')
head(algue)
```

(3) Afficher les statistiques basiques des différentes variables :

```
summary(algue)
```

(4) Pour avoir un petit apperçu visuel des données, taper :

```
library(car)
op = par(mfrow=c(1,2))
hist(algue$mxPH, prob=T, xlab="",
main="Histogram of maximum pH value",ylim=0 :1)
lines(density(algue$mxPH,na.rm=T))
rug(jitter(algue$mxPH))
qqnorm(algue$mxPH,main="Normal QQ plot of maximum pH")
par(op)
```

(5) Pour tracer des boxplot conditionnelles où le conditionnement est fait par rapport à une variable catégorielle (ici size) :

```
library(lattice)
bwplot(size ~ freq, data=algue, ylab="Taille des Rivieres",xlab="Algue")
```

4. Méthode des Moindres Carrés

Pour nous assurer que les estimateurs construits ne sont pas sensibles à l'échantillon utilisé pour construire les estimateurs, nous définissons une bases d'apprentissage et une base de test :

(1) Construire une base d'apprentissage et une base de test :

```
ind.test=4*c(1 :46)
algue.app=algue[-ind.test,]
algue.test=algue[c(ind.test),]
```

(2) Calcul des coefficients de la régression :

```
mod1 = lm(freq ∼ ., data = algue.app[, 1 :12])
summary(mod1)
```

(3) Calcul de l'erreur de prédiction sur l'échantillon d'apprentissage :

```
res1=residuals(mod1)
mean(res1**2)
```

(4) Calcul de l'erreur de prédiction sur l'échantillon test :

```
pred1.test=predict(mod1, newdata=algue.test)
res1.test=pred1.test-algue.test$freq
mean(res1.test**2)
```

5. Méthodes de sélection de variables

La dimension du problème (11 variables) n'est pas très grande. Toutefois, au vue des résultats ci-dessus, on peut se demander si on ne pourrait pas obtenir de meilleurs résultats en ne considérant qu'un sous-ensemble des variables.

5.1. **Méthode "stepwise".**

(1) Pour choisir les variables pertinentes :

```
anova(mod1)
```

(2) Pour déterminer un sous-modèle qui ne contient pas de variable inutile :

```
Step.mod1 = step(mod1)
```

(3) Calcul de l'erreur de prédiction sur l'échantillon d'apprentissage pour le meilleur sous modèle :

```
Step.res1=residuals(Step.mod1)
mean(Step.res1**2)
```

(4) Calcul de l'erreur de prédiction sur l'échantillon test pour le meilleur sous modèle :

```
Step.pred1.test=predict(Step.mod1, newdata=algue.test)
Step.res1.test=Step.pred1.test-algue.test$freq
mean(Step.res1.test**2)
```

5.2. **Méthode LASSO.**

(1) Une alternative plus populaire est l'estimateur LASSO :

```
library(lasso2)
Lasso1 = l1ce(freq ∼ ., algue.app,bound=(1 :100)/100,absolute.t=FALSE)
coefficients1=coef(Lasso1)
penalite.relative=c(1 :100)/100
matplot(penalite.relative,coefficients1[,-1], lty=1 :3,type="l",col=1 :10)
legend("topleft",legend=colnames(coefficients1[,-1]),col=1 :10,lty=1 :3)
```

(2) Sélection la solution optimale :

```
vc1=gcv(Lasso1)
crit.vc1=vc1[,"gcv"]
bound.opt1=vc1[which.min(crit.vc1),"rel.bound"]
Lasso.opt1 = l1ce(freq ∼ ., algue.app,
bound=bound.opt1, absolute.t=FALSE)
coef1=coef(Lasso.opt1)
```

(3) Calcul de l'erreur de prédiction sur l'échantillon d'apprentissage :

```
fitLasso1=fitted(Lasso.opt1)
mean((fitLasso1-algue.app[,"freq"])**2)
```

(4) Calcul de l'erreur de prédiction sur l'échantillon test :

```
predictionLasso1 = predict(Lasso.opt1,newdata=algue.test)
mean((predictionLasso1-algue.test[,"freq"])**2)
```

6. Données en Grande Dimension

On se propose à présent de transformer les données "algue" pour en faire des données en grande dimension. On les analysera par la suite comme précédemment :

(1) Aggrandir les données en ajoutant des variables "bruits" :

```
AlgueGrand = cbind(algue,matrix(rnorm(139*184),nrow=184))
```

(2) Construire les échantillons d'apprentissage et de test :

```
AlgueGrand.app=AlgueGrand[-ind.test,]
AlgueGrand.test=AlgueGrand[c(ind.test),]
```

(3) Construire et analyser le modèle linéaire comme dans la section précédente :

```
mod2 = lm(freq ∼ ., data = AlgueGrand.app[, 1 :151])
summary(mod2)
# erreur d'apprentissage
res2 = residuals(mod2)
mean(res2**2)
# erreur de test
pred2.test = predict(mod2, newdata=AlgueGrand.test)
res2.test = pred2.test-AlgueGrand.test$freq
mean(res2.test**2)
```

(4) Utiliser la méthode stepwise :

```
anova(mod2)
Step.mod2 = step(mod2)
# erreur d'apprentissage
Step.res2=residuals(Step.mod2)
mean(Step.res2**2)
# erreur de test
Step.pred2.test=predict(Step.mod2, newdata=AlgueGrand.test)
Step.res2.test=Step.pred2.test-AlgueGrand.test$freq
mean(Step.res2.test**2)
```

(5) Construire l'estimateur LASSO :

```
Lasso2 = l1ce(freq ∼ ., AlgueGrand.app,bound=(1 :100)/100,absolute.t=FALSE)
coefficients2=coef(Lasso2)
# selection de l'estimateur optimal
vc2 = gcv(Lasso2)
crit.vc2 = vc2[,"gcv"]
bound.opt2 = vc2[which.min(crit.vc2),"rel.bound"]
Lasso.opt2 = l1ce(freq ∼ ., AlgueGrand.app,
bound = bound.opt2, absolute.t=FALSE)
coef2=coef(Lasso.opt2)
# erreur d'apprentissage
fitLasso2=fitted(Lasso.opt2)
mean((fitLasso2-AlgueGrand.app[,"freq"])**2)
# erreur de test
predictionLasso2 = predict(Lasso.opt2,newdata=AlgueGrand.test)
mean((predictionLasso2-AlgueGrand.test[,"freq"])**2)
```