Université de Montpellier Année 2019-2020
HMMA304 - Modèles linéaires généralisés

Examen de contrôle continu
Durée 1h15 - Sans document - 8 octobre 2019

Exercice 1 (5 pts)

On considère le modèle de régression suivant :

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_3 x_{i,3} + \epsilon_i$$

pour $1 \geq i \geq n$, les vecteurs des x_i sont déterministes et le vecteur des ϵ_i est un vecteur gaussien centré de matrice de variance covariance $\sigma^2 I_n$. On pose $y = (y_1, \ldots, y_n)$ et

$$X = \begin{bmatrix} 1 & x_{1,1} & \ldots & x_{1,3} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n,1} & \ldots & x_{n,3} \end{bmatrix}$$

Nous avons

$$X^T X = \begin{bmatrix} 50 & 0 & 0 & 0 \\ 0 & 20 & 15 & 4 \\ 0 & 15 & 30 & 10 \\ 0 & 4 & 10 & 40 \end{bmatrix}, \quad X^T y = (100, 50, 40, 80), \quad y^T y = 640.$$

On admet que

$$\begin{bmatrix} 20 & 15 & 4 \\ 15 & 30 & 10 \\ 4 & 10 & 40 \end{bmatrix}^{-1} = \frac{1}{13720} \begin{bmatrix} 1100 & -560 & 30 \\ -560 & 784 & -140 \\ 30 & -140 & 375 \end{bmatrix}.$$

1 (1 pt) Donner la valeur de n.

2 (1 pt) Interpréter les 0 de la matrice $X^T X$.

3 (1 pt) Donner l'estimateur des moindres carrés de $(\beta_0, \beta_1, \beta_2, \beta_3)$, peut-on calculer l'estimation associée avec les données fournies ?

4 (1 pt) Donner un intervalle de confiance de niveau de confiance à 95% pour β_3.

5 (1 pt) Construire un test de niveau 5% de l'hypothèse $H_0 : \beta_3 = 0$ contre $H_1 : \beta_3 \neq 0$.

Exercice 2 (3 pts)

On considère n variables aléatoires indépendantes $y_1, \ldots, y_n$ telles que y_i est distribuée suivant une loi de Poisson de paramètre λ_i.

1 (0.5 pt) Montrer que la loi de Poisson appartient à la famille exponentielle.

2 (0.5 pt) Calculer $\mathbb{E}(y_i)$ et $\mathbb{V}(y_i)$.

Pour tout $i = 1, \ldots, n$, on suppose que $\log(\sqrt{\lambda_i}) = \beta_1 + \beta_2 x_i$.

3 (1 pt) Montrer qu'il s'agit d'un modèle linéaire généralisé. La fonction de lien canonique a-t-elle été utilisée ?

Exercice 3 (6 pts)
Expliquer en détails les résultats produits par le code Python ci-dessous

```
import numpy as np
import matplotlib.pyplot as plt

from sklearn import datasets
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV

diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target

print(diabetes.data.shape)
# (442, 10)

print(diabetes.target.shape)
# (442,)

lasso = Lasso()
alphas = np.logspace(-4, -0.5, 30)
alphas[0]
# 0.0001
alphas[29]
# 0.31622776601683794
10**(-0.5)
# 0.31622776601683794

tuned_parameters = [{'alpha': alphas}]
n_folds = 5

clf = GridSearchCV(lasso, tuned_parameters, cv=n_folds)
clf.fit(X, y)
scores = clf.cv_results_['mean_test_score']
scores_std = clf.cv_results_['std_test_score']
```

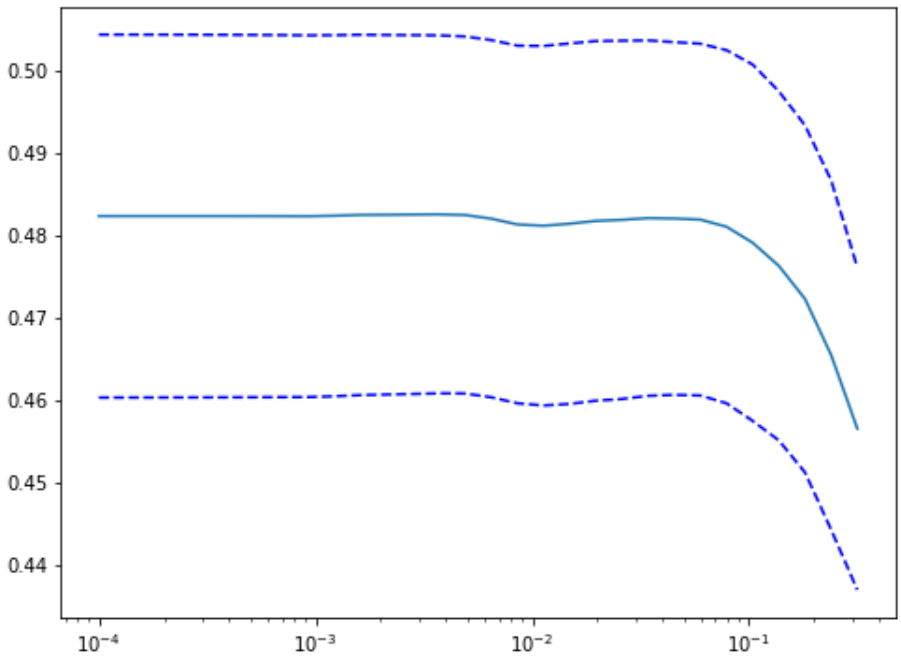
```
print(scores)
#[0.48229145 0.48229179 0.48229211 0.48229232 0.48229222 0.48229143
# 0.48228924 0.48228437 0.48227446 0.48233369 0.48242998 0.48244909
# 0.48247616 0.48249863 0.48242247 0.48197794 0.48127519 0.48112164
# 0.48136664 0.48170547 0.48183326 0.48204223 0.48199684 0.4818627
# 0.48099981 0.47905578 0.47625075 0.47224129 0.46542727 0.45651226]

print(scores_std)
# [0.04923312 0.04922588 0.04921635 0.04920385 0.04918746 0.04916604
#  0.04913812 0.0491019  0.04905525 0.04897275 0.04887265 0.04876697
#  0.04865917 0.0485564  0.04842364 0.04844148 0.04849849 0.04875972
#  0.04888454 0.048766   0.04859029 0.04819974 0.0478335  0.04771292
#  0.04788694 0.04833115 0.04734514 0.04707518 0.04738262 0.04372668]

plt.figure().set_size_inches(8, 6)
plt.semilogx(alphas, scores)

std_error = scores_std / np.sqrt(n_folds)

plt.semilogx(alphas, scores + std_error, 'b--')
plt.semilogx(alphas, scores - std_error, 'b--')
```



Exercice 4 (6 pts)
Expliquer en détails les résultats produits par le code R ci-dessous

```
n <- 100
x1 <- rnorm(n)
x2 <- rnorm(n)
x3 <- rnorm(n)
u <- rnorm(n)
y <- 1+1.5*x1+2*x2-0.7*x3+u

d <- 80
z <- matrix(runif(n*d),n)

app <- data.frame(y=y,x1=x1,x2=x2,x3=x3,z=z)
names(app)[-(1:4)] <- paste("z",1:80,sep="")

model1 <- lm(y~1,data=app)
model2 <- lm(y~.,data=app)
model3 <- lm(y~x1+x2+x3,data=app)

library(glmnet)
library(caret)
x <- as.matrix(app[,-1])
model4 <- glmnet(x,y,family="gaussian",nlambda=50,alpha=1)
model5 <- train(x,y,method="glmnet",metric="RMSE",
                trControl=trainControl(method="repeatedcv",
                                       number=5,repeats=100),
                tuneGrid=data.frame(alpha=1,lambda=model4$lambda))
ntest <- 1000
x1test <- rnorm(n)
x2test <- rnorm(n)
x3test <- rnorm(n)
utest <- rnorm(n)
ytest <- 1+1.5*x1test+2*x2test-0.7*x3test+utest
ztest <- matrix(runif(n*d),n)

test <- data.frame(y=ytest,x1=x1test,x2=x2test,x3=x3test,z=ztest)
names(test)[-(1:4)] <- paste("z",1:80,sep="")

mean((predict(model3,test)-ytest)^2)
# 1.000278
mean((predict(model1,test)-ytest)^2)
# 10.37723
mean((predict(model2,test)-ytest)^2)
# 6.840056
mean((predict(model5,test)-ytest)^2)
# 1.217017
```

Correction examen ch-contrôle-continu
8 octobre 2019
HMMA 304

Exercice 1

1) $n = 50$

2) Les variables explicatives x_1, x_2 et x_3 sont centrées.

3) $\hat{\beta} = (X^T X)^{-1} X^T y$

$\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3)$

Nous pouvons calculer l'estimation ordinaire avec les données du problème

$$(X^T X)^{-1} = \begin{bmatrix} 1/50 & 0 & 0 & 0 \\ 0 & \begin{bmatrix} 20 & 15 & 4 \\ 15 & 30 & 10 \\ 4 & 10 & 40 \end{bmatrix}^{-1} \\ 0 & \\ 0 & \end{bmatrix}$$

4] Nous savons que

$$\hat{\beta}_3 - \beta_3 \sim N_1\left(0, \sigma^2 (X^T X)^{-1}_{(4,4)}\right)$$

Par ailleurs, on peut estimer σ^2 par $\hat{\sigma}^2 = \frac{(y-\hat{y})^T (y-\hat{y})}{n-4}$

$(y-\hat{y})^T (y-\hat{y})$ est la somme des carrés résiduelle

$$\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y$$

$$(y-\hat{y}) = \left[I_n - X(X^T X)^{-1} X^T\right] y$$

$$(y-\hat{y})^T (y-\hat{y}) = y\left[I_n - X(X^T X)^{-1} X^T\right] y$$

$$(y-\hat{y})^T (y-\hat{y}) = y^T y - y^T X (X^T X)^{-1} X^T y$$

Avec les valeurs du problème, on peut calculer $(y-\hat{y})^T (y-\hat{y})$.

Άμεση,

$$\frac{(\hat{\beta}_3 - \beta_3) / \sqrt{\sigma^2 (X^T X)^{-1}_{(4,4)}}}{\sqrt{\frac{(y-\hat{y})^T (y-\hat{y})}{\sigma^2 (n-4)}}} \sim T(n-4)$$

$$\frac{(\hat{\beta}_3 - \beta_3)}{\sqrt{\hat{\sigma}^2 (X^T X)^{-1}_{(4,4)}}} \sim T(n-4)$$

Άρα, αν $q = F^{-1}_{T(n-4)}(0,975)$

$$P\left(-q \leq \frac{(\hat{\beta}_3 - \beta_3)}{\sqrt{\hat{\sigma}^2 (X^T X)^{-1}_{(4,4)}}} \leq q\right) = 0.95$$

$$\text{ΔΕ}_{95\%}(\beta_3) = \left[\hat{\beta}_3 \pm q \sqrt{\hat{\sigma}^2 (X^T X)^{-1}_{(4,4)}}\right]$$

5] On rejette $H_0: \beta_3 = 0$ (4)

si $0 \notin IC_{95\%}(\beta_3)$.

Exercice 2

1] $P_\lambda(y) = \frac{e^{-\lambda} \lambda^y}{y!} \mathbb{1}_{\mathbb{N}}(y)$ $\Gamma(y+1)$

on $\Gamma(\cdot)$ est la même chose comptons ça.

$P_\lambda(y) = \exp\{-\lambda + y \log(\lambda)\} \nu(y)$

où $\nu(y) = \frac{1}{y!} \mathbb{1}_{\mathbb{N}}(y) \Gamma(y+1)$

La loi de Poisson appartient bien à la famille exponentielle canonique.

$\theta = \log(\lambda) \Rightarrow \lambda = e^\theta$

$b(\theta) = e^\theta$

2] $E(y) = b'(\theta) = e^\theta = \lambda$

$V(y) = b''(\theta) = e^\theta = \lambda$

3| mon nom

$$\log(\sqrt{n}) = \beta_1 + \beta_2 x$$

$$\Rightarrow \quad n = \exp(2\beta_1 + 2\beta_2 x)$$

$$\Rightarrow E(y) = f(\beta_1 + \beta_2 x)$$

IC n'existe pas car un modèle linéaire généralisé.

Par ailleurs, $0 \neq \beta_1 + \beta_2 x$ donc ce n'est pas le bon lien canonique qui a été utilisé.

Cependant, $0 = 2\beta_1 + 2\beta_2 x$ et donc à une reparamétrisation près, il s'agit bien du lien canonique.

Exercice 3

Le jeu de données utilisé est simulé. Il contient une variable quantitative à prédire et 10 variables quantitatives prédictives. Les observations sont réparties aléatoirement dans 3 blocs.

Une régression linéaire est mise en œuvre, la pénalisation de régularisation α est calibrée par validation croisée à 5 ensembles. 30 valeurs sont testées de 10^{-4} à $10^{-1/2}$. Plus α est fort, plus on régularise. La valeur de α choisie par le critère du R^2 est très petite, il n'y a pas de différence significative pour les petites valeurs. La régularisation semble ici inutile.

Exercice 4

On simule 100 données suivant le modèle

$$y = 1 + 1{,}5\,x_1 + 2\,x_2 - 0{,}7\,x_3 + u$$

avec $u \sim \mathcal{N}(0,1)$, les x_i sont également des gaussiennes centrées réduites.

On teste 4 modèles :

- modèle 1 : contient uniquement la constante
- modèle 2 : contient 83 variables x_1, x_2 et x_3 et 80 variables de bruit
- modèle 3 : modèle vrai, contient les 3 variables x_1, x_2 et x_3
- modèle 5 : régression lasso sur les 83 variables avec paramètre de régularisation choisi par validation croisée à 5 ensembles répétée 100 fois

Sur un jeu de données de test de taille 1 000, on observe que la meilleure forêt donne d'excellents résultats. Son erreur quadratique moyenne est très proche de celle de la meilleure forêt.