Statistical Learning

Prova d'esame

14 Luglio 2022

Tempo a disposizione: 180 minuti

Problema 1

Si considerino i seguenti dati:

Player	n_i	s_i	pi_i
Baines	90	26	0.289
Barfield	90	22	0.256
Bell	90	23	0.265
Biggio	90	25	0.287
Bonds	90	27	0.297

di $p = 5$ giocatori di baseball, dove n_i e s_i indicano rispettivamente il numero di volte a battuta e il numero di battute valide, mentre π_i indica la vera media battuta (calcolata su tutta la carriera di ciascun giocatore).

Sia Z_i la variabile aleatoria Binomiale$(n_i, \pi_i)/n_i$, e si supponga che $Z_1, \ldots, Z_p$ siano indipendenti.

Si consideri valida la seguente approssimazione

$$Z_i \approx N(\pi_i, \sigma_0^2)$$

dove $\sigma_0^2 = \frac{\bar{p}(1-\bar{p})}{90}$ con $\bar{p}$ pari alla media dei valori (s_i/n_i).

a) Sia $\hat{\pi}^{\text{MLE}}$ la stima di massima verosimiglianza per $\pi = (\pi_1, \ldots, \pi_p)$. Riportare il valore della stima per Bell.

```
tab[row_i,3]/tab[row_i,2]
```

```
## [1] 0.2555556
```

b) Sia $\hat{\pi}^{\text{JS}}$ la stima secondo James-Stein per π. Riportare il valore della stima per Bell.

```
p_bar <- mean(tab[,3]/tab[,2])
sigma2_0 <- p_bar*(1-p_bar)/90
x_i = (tab[,3]/tab[,2])/sqrt(sigma2_0)
x_bar <- mean(x_i)
S <- sum((x_i-x_bar)^2)
mu_i_js = x_bar + (1 - ((p-3)/S)) * (x_i - x_bar)
pi_i_js = mu_i_js * sqrt(sigma2_0)
b <- pi_i_js[row_i]
b
```

```
## [1] 0.2925085
```

c) Calcolare

$$\frac{\sum_{i=1}^{p}(\hat{\pi}_i^{\text{JS}} - \pi_i)^2}{\sum_{i=1}^{p}(\hat{\pi}_i^{\text{MLE}} - \pi_i)^2}$$

```
sum( (pi_i_js - tab[,4])^2 ) / sum( (tab[,3]/tab[,2] - tab[,4])^2 )
```

```
## [1] 22.88772
```

Problema 2

Si consideri il dataset `cement` presente nella libreria `MASS`. La variabile risposta è `y`, i predittori sono le rimanenti variabili.

a. Sia X la matrice del disegno. Calcolare il *condition number* di $(X^\mathsf{T} X + \lambda I_p)$ per il modello di regressione *ridge* con $\lambda = 10$.

```
rm(list=ls())
library(MASS)
fit <- lm(y ~ ., cement)
X <- model.matrix(fit)
d <- svd(crossprod(X))$d
lambda <- 10
(max(d)+lambda)/(min(d)+lambda)
```

```
## [1] 4468.076
```

b. Calcolare la stima dell'errore di previsione tramite *leave-one-out cross-validation*

$$\text{LOO}_\lambda = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i^t \hat{\beta}_\lambda^{(-i)})^2$$

dove $\hat{\beta}_\lambda^{(-i)}$ è la stima *ridge* calcolata su $(n-1)$ osservazioni escludendo l'osservazione (x_i, y_i) per λ pari a $0, 1, 2, \ldots, 99$. Riportare il valore minimo di LOO_λ.

```
y <- cement$y
lambdas = 0:99
SVD_X <- svd(X)
d <- SVD_X$d
U <- SVD_X$u
LOO <- vector()
for (l in lambdas){
  R_l <- U %*% diag((d^2)/(d^2 + l)) %*% t(U)
  y_hat <- R_l %*% y
  LOO[l+1]<-mean( ( (y - y_hat) / ( 1 - diag(R_l) ) )^2 )
}
min(LOO)
```

```
## [1] 7.569966
```

c. Calcolare la stima dell'errore di previsione tramite *generalized cross-validation* con il valore di λ che ha minimizzato LOO_λ nel punto precedente.

```
l <- lambdas[which.min(LOO)]
l
```

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

```r
R_l <- U %*% diag((d^2)/(d^2 + l)) %*% t(U)
y_hat <- R_l %*% y
GCV <- mean( ( (y - y_hat) / ( 1 - mean(diag(R_l)) ) )^2 )
GCV
```

```
## [1] 8.427681
```

Soluzione alternativa al punto a. basata sulla matrice dei dati standardizzati

```r
rm(list=ls())
library(MASS)
X <- model.matrix(lm(y~.,cement))[,-1]
n <- nrow(X)
X_mean <- colMeans(X)
X <- X - rep(1,n) %*% t(X_mean)
X_scale <- sqrt( diag( (1/n) * crossprod(X) ) )
X <- X %*% diag( 1 / X_scale )
d <- svd(crossprod(X))$d
lambda <- 10
(max(d)+lambda)/(min(d)+lambda)
```

```
## [1] 3.898187
```

Soluzione alternativa al punto b. basata sulla funzione `glmnet`.

```r
rm(list=ls())
library(MASS)
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-4
```

```r
# glmnet
y <- cement$y
X <- model.matrix(lm(y~.,cement))[,-1]
fit_glmnet <- cv.glmnet(x=X,y=y,alpha=0, lambda=0:99, nfolds = nrow(X), grouped=FALSE)
fit_glmnet
```

```
##
## Call:  cv.glmnet(x = X, y = y, lambda = 0:99, nfolds = nrow(X), grouped = FALSE,      alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min      1    99   7.530 2.144       4
## 1se      2    98   8.768 2.464       4
```

```r
l = fit_glmnet$lambda.min
fit_glmnet$cvm[fit_glmnet$lambda==l]
```

```
##      s98
## 7.530415
```

Problema 3

Si consegni il file .R che produce le risposte alle domande richieste. Il codice deve essere **riproducibile** e, se eseguito, deve stampare in output **solo** il risultati richiesti dalle domande a) e b).

Si consideri il dataset `Boston` presente nella libreria `MASS`. La variabile risposta è `medv`, i predittori sono le rimanenti variabili.

Si consideri come *training set* le prime 505 osservazioni (righe) del dataset `Boston`, e come *test point* l'ultima osservazione (riga 506).

Si utilizzi l'algoritmo *split conformal* considerando come *Learning set* le osservazioni del training set con indici pari, i.e. $L = \{2, 4, \ldots, 504\}$ e come *Inference set* le osservazioni del training set con indici dispari, i.e. $I = \{1, 3, \ldots, 505\}$.

Si costruisca l'intervallo di previsione per il test point a livello $1-\alpha$ con $\alpha = 25/254$. L'algoritmo da utilizzare è la regressione *forward*, impostata in modo da selezionare 6 variabili (per la regressione *forward* è obbligatorio utilizzare la funzione `step` presente nella libreria `stats`).

Riportare

a) gli estremi dell'intervallo di previsione;
b) il valore `TRUE` se il test point si trova all'interno dell'intervallo di previsione, `FALSE` altrimenti.

```
rm(list=ls())
library(MASS)
train = Boston[-506,]
test = Boston[506,]
x = train[,-14]
y = train[,14]
n = nrow(x)
p = ncol(x)
x_new = test[,-14]
y_true = as.numeric(test[,14])
L = seq(2,505, by=2)
I = seq(1,505, by=2)
m = length(I)
alpha = 25/254
fit_null <- lm(medv ~ 1, data = train, subset=L)
fit_full <- lm(medv ~., data = train, subset=L)
fit_L = step(fit_null,
scope=list(upper=fit_full),
direction = "forward", k=0, trace = 0, steps = 6)
y_hat = predict(fit_L, newdata=train[I,])
res = abs(y[I] - y_hat)
o = order(res)
c = ceiling((1-alpha)*(m+1))
r = res[o][c]
y_new = predict(fit_L, newdata=x_new)
lo = y_new - r
up = y_new + r
c(lo,up)
```

```
##      506      506
## 15.35411 30.50379
```

```
y_true
```

```
## [1] 11.9
```

```
(lo <= y_true & y_true <= up)
```

```
##   506
## FALSE
```

Problema 4

Si consideri il seguente esperimento (ipotetico) con tecnologia *microarray*. Ci sono $n = 400$ partecipanti che entrano nello studio uno per giorno, e ricevono il Trattamento o il Placebo a giorni alterni (il giorno 1 il soggetto 1 riceve il Trattamento, il giorno 2 il soggetto 2 riceve il Placebo, il giorno 3 il soggetto 3 riceve Trattamento, etc.). A ciascun soggetto viene misurata la risposta di $p = 200$ geni. La matrice dei dati X è di dimensione 400×200, e ciascun elemento è indipendente con distribuzione gaussiana, ovvero

$$X_{ij} \overset{\text{ind}}{\sim} \mathcal{N}(\mu_{ij}, 1)$$

Si supponga che per $j = 30, 48, 57, 65, 84, 92, 113, 128, 143, 195$

$$\mu_{ij} = 2 \quad i = 1, 3, \ldots, 317, 319 \text{ (Treatment)} \quad \mu_{ij} = -2 \quad i = 2, 4, \ldots, 318, 320, \text{ (Placebo)}$$

e per tutto il resto $\mu_{ij} = 0$. Si consideri l'utilizzo della Foresta Casuale (*Random Forest*)

1. con suddivisione in training set di 320 soggetti e test set di 80 soggetti in maniera casuale (RF-I);
2. con suddivisione in training set con i primi 320 soggetti e test set con gli ultimi 80 soggetti (RF-II);

Si risponda alle seguenti domande (motivare la risposta per ciascun quesito.)

a) Ti aspetti che l'errore di previsione valutato sul training set con RF-I sia minore/uguale/maggiore a quello con RF-II?

b) Ti aspetti che l'errore di previsione valutato sul test set con RF-II sia minore/uguale/maggiore al 50%?

c) Quante variabili (geni) ti aspetti di trovare con elevato punteggio di *Variable Importance* (calcolato sul training set) se utilizzi RF-II?