Modello di regressione logistica

Domenico De Stefano

Obiettivo delle seguenti lezioni del laboratorio di TSC è l'utilizzo di R per l'analisi di modelli logit. In questa lezione vedremo il più semplice di questi, ovvero il modello di regressione logistica. A tale scopo utilizzeremo il alcuni dataset (tra cui il dataset 'Studenti.csv' già impiegato nella precedente lezione). Ricordiamo che esso contiene una serie di informazioni sui 356 immatricolati presso la facoltà di Economia dell'Università di Trieste nell'A.A. 2009/2010.
Ripercorreremo in R le fasi della costruzione del modello viste durante il corso di TSC.
Ricordiamo, in breve, i desiderata di un modello statistico in generale:

- Inclusione delle variabili statisticamente rilevanti
  - Importanti sulla base del contesto (problema/disciplina)
  - controllo di possibili fattori di "confondimento"
  - evitare rischio di "overfitting"
- Modelli parsimoniosi
  - risultati più stabili
  - minor dipendenza dagli specifici dati osservati
  - maggiore generalizzazione
- buon adattamento del modello ai dati

Questi desiderata si traducono in alcuni passi operativi da seguire:

1. Analisi dati univariata
2. Selezione delle variabili da includere nel modello
3. Test statistici per la bontà di adattamento del modello

1 Alcuni esempi di regressione logistica con R

1.1 Analisi dei dati Challenger.DAT

Nel campione in `challenger.dat` si hanno le registrazioni dei guasti occorsi alle guarnizioni ad anelli in 23 lanci dello *space shuttle* e le temperature atmosferiche al momento del lancio.
In particolare, la variabile `failures` vale 0 se non si è verificato alcun guasto, vale 1 se si è verificato almeno un guasto, `temp` è la temperatura.

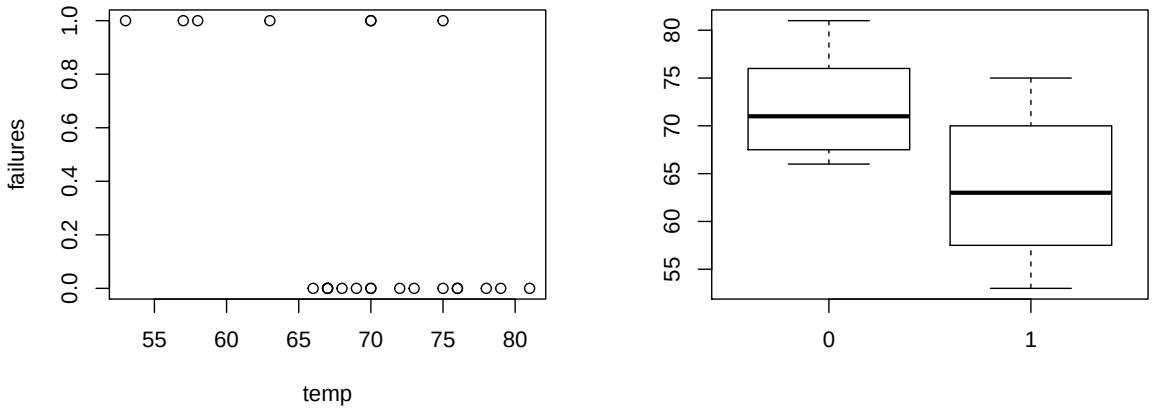
```
chl = read.table("challenger.dat", header = TRUE)
head(chl)

##   failures temp
## 1        0   66
## 2        1   70
## 3        0   69
## 4        0   68
## 5        0   67
## 6        0   72

attach(chl)
```

L'obiettivo dell'analisi è stabilire se la probabilità di un guasto dipenda dalla temperatura.
Possiamo rappresentare i dati mediante un diagramma di dispersione o mediante box-plot.

```
par(mfrow = c(1, 2))
plot(temp, failures)
plot(factor(failures), temp)
```



Dall'esame dei grafici appare plausibile che la probabilità di guasto dipenda dalla temperatura. Per quantificare l'effetto si decide di specificare un modello di regressione logistica, detta Y_i la variabile `failures` e x_i la variabile `temp` si assume

$\pi_i = P(Y_i = 1)$

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_1 + \beta_2 x_i$$

La stima di massima verosimiglianza del modello si ottiene con il comando

```
chl.fit = glm(failures ~ temp, family = binomial(link = logit))
summary(chl.fit)

##
## Call:
## glm(formula = failures ~ temp, family = binomial(link = logit))
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.061  -0.761  -0.378   0.452   2.217
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   15.043      7.379    2.04    0.041 *
## temp          -0.232      0.108   -2.14    0.032 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 28.267  on 22  degrees of freedom
## Residual deviance: 20.315  on 21  degrees of freedom
## AIC: 24.32
##
## Number of Fisher Scoring iterations: 5
```

dove `family=binomial(link=logit)` specifica il modello binomiale con legame logistico. La precisazione sul legame può essere omessa in questo caso poichè il legame logit è quello predefinito. L'errore standard per il coefficiente β_r è la radice dell'elemento diagonale r-esimo della matrice di informazione attesa (che coincide con l'osservata), $[\sqrt{(X^T\hat{V}X)^{-1}}]_{rr}$, il valore z è

$$z_r = \frac{\hat{\beta}_r}{[\sqrt{(X^T\hat{V}X)^{-1}}]_{rr}}$$

ad esempio per β_2 si ha

```
beta2.hat = coef(chl.fit)[2]
beta2.se = summary(chl.fit)$coef[2, 2]
z2 = beta2.hat/beta2.se
z2

##   temp
## -2.145
```

e il valore p si ottiene come

$$2(1 - \Phi(|z_r|))$$

ad esempio per β_2 si ha

```
2 * (1 - pnorm(abs(z2)))

##    temp
## 0.03196
```

In base a questo, si può concludere che la probabilità di un guasto dipende dalla temperatura.
Si può poi calcolare la probabilità di un guasto a una fissata temperatura, ad esempio $x_0 = 55$ gradi calcolando prima $\hat{\beta}_1 + \hat{\beta}_2 x_0$ (che è anche chiamato predittore lineare)

```
predlin0.hat = coef(chl.fit)[1] + coef(chl.fit)[2] * 55
predlin0.hat

## (Intercept)
##       2.274
```

e poi calcolando l'inverso del logit,

$$\hat{\pi}_0 = \frac{e^{\hat{\beta}_1 + \hat{\beta}_2 x_0}}{1 + e^{\hat{\beta}_1 + \hat{\beta}_2 x_0}}$$

si ha quindi

```
pi0.hat = exp(predlin0.hat)/(1 + exp(predlin0.hat))
pi0.hat

## (Intercept)
##      0.9067
```

Col comando seguente si calcola il predittore lineare in corrispondenza a tutti i valori osservati di temperatura

```
predlin.hat = coef(chl.fit)[1] + coef(chl.fit)[2] * temp
predlin.hat

##  [1] -0.2798 -1.2085 -0.9763 -0.7442 -0.5120 -1.6728 -1.9050 -1.2085
##  [9]  1.8096  0.4166 -1.2085 -3.0658 -0.5120  2.7383 -0.5120 -2.3693
## [17] -1.2085 -3.7623 -2.6015 -3.2980 -2.3693 -2.6015  1.5775
```

e si ottiene poi la probabilità con

```
pi.hat = exp(predlin.hat)/(1 + exp(predlin.hat))
pi.hat
```

```
##  [1] 0.43049 0.22997 0.27362 0.32209 0.37472 0.15805 0.12955 0.22997
##  [9] 0.85932 0.60268 0.22997 0.04454 0.37472 0.93925 0.37472 0.08554
## [17] 0.22997 0.02270 0.06904 0.03564 0.08554 0.06904 0.82884
```

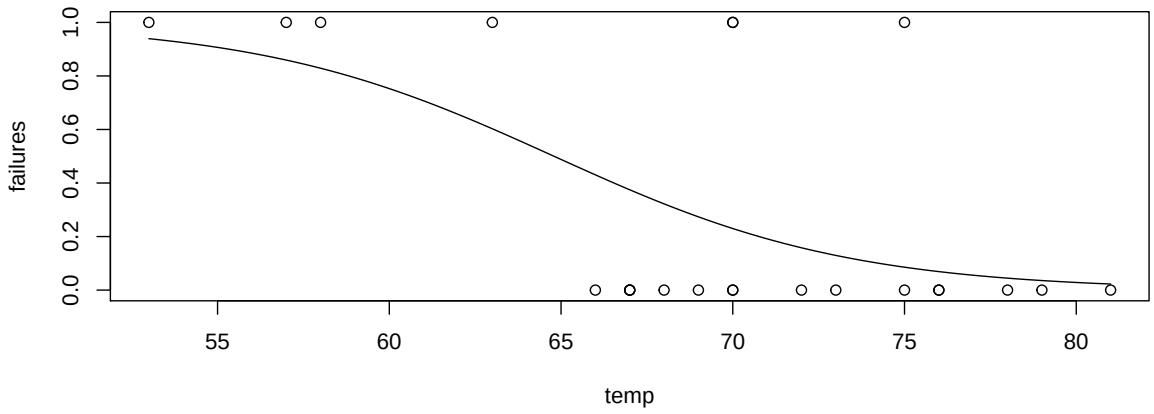
Si noti che `pi.hat` si può ottenere anche usando la funzione `predict` specificando l'argomento `type=response`

```
predict(chl.fit, type = "response")
```

```
##       1       2       3       4       5       6       7       8       9
## 0.43049 0.22997 0.27362 0.32209 0.37472 0.15805 0.12955 0.22997 0.85932
##      10      11      12      13      14      15      16      17      18
## 0.60268 0.22997 0.04454 0.37472 0.93925 0.37472 0.08554 0.22997 0.02270
##      19      20      21      22      23
## 0.06904 0.03564 0.08554 0.06904 0.82884
```

mentre si ottiene `predlin.hat` con l'argomento `type=link`.
Si può aggiungere al grafico la probabilità stimata con

```
plot(temp, failures)
curve(predict(chl.fit, newdata = data.frame(temp = x), type = "response"), add = TRUE)
```




```
detach(chl)
```

1.2 Dati sulla sopravvivenza di pazienti in reparti di cura intensiva

Per 200 pazienti ammessi in un reparto di cura intensiva sono state registrate alcune caratteristiche rilevanti per il loro stato di salute, oltre all'esito finale, cioè se siano sopravvissuti o meno.
In particolare si sono osservate le variabili

- `stato`: stato del paziente (0 = vivo, 1 = deceduto)
- `eta`: età in anni del paziente
- `causa`: causa dell'ammissione (1= programmata, 2 = emergenza)
- `coscienza`: Livello di coscienza all'ammissione (1 = no coma o stupor, 2= stupor o coma)

Gli obiettivi che ci si pone sono da una parte stabilire quali delle caratteristiche rilevate abbiano un'influenza sulla probabilità di sopravvivenza dall'altra predire la probabilità di sopravvivenza date le sopra citata caratteristiche.
Si carica il campione con

```
ICU = read.table("ICUdata.dat", header = TRUE)
head(ICU)

##   stato eta causa coscienza
## 1     0  27     2         1
## 2     0  59     2         1
## 3     0  77     1         1
## 4     0  54     2         1
## 5     0  87     2         1
## 6     0  69     2         1

attach(ICU)
```

Ricodifichiamo i fattori in maniera opportuna, assegnando nomi interpretabili per i livelli.

```
ICU$causa = factor(ICU$causa)
levels(ICU$causa) = c("P", "E")
ICU$coscienza = factor(ICU$coscienza)
levels(ICU$coscienza) = c("SI", "NO")
```

Per investigare l'effetto delle variabili esplicative, è naturale specificare un modello di regressione logistica in cui, sia Y_i la variabile che vale 1 se il paziente decede, x_2 l'età, x_3 la causa di ammissione al reparto (1 se l'ammissione è non programmata (emergenza)) e x_4 è la variabile che vale 1 se il paziente entra incosciente, si assume
$\pi_i = P(Y_i = 1)$

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_1 + \beta_2 x_{i2} + \beta_2 x_{i3} + \beta_2 x_{i4}$$

La stima di massima verosimiglianza del modello si ottiene con

```
ICU.fit = glm(stato ~ ., family = binomial, data = ICU)
summary(ICU.fit)
```

```
##
## Call:
## glm(formula = stato ~ ., family = binomial, data = ICU)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.480  -0.655  -0.349  -0.198   2.562
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.7715     1.1462   -5.04  4.8e-07 ***
## eta           0.0337     0.0121    2.79   0.0053 **
## causaE        2.3447     0.8068    2.91   0.0037 **
## coscienzaNO   3.4562     0.8293    4.17  3.1e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 200.16  on 199  degrees of freedom
## Residual deviance: 145.31  on 196  degrees of freedom
## AIC: 153.3
##
## Number of Fisher Scoring iterations: 6
```

I test sulla nullità dei singoli coefficienti rifiutano tutti l'ipotesi nulla, consideriamo comunque anche il test per la significatività del modello nel suo complesso, cioè verifichiamo l'ipotesi:
$H_0 : \beta_2 = \beta_3 = \beta_4 = 0$
confrontando, mediante il test del rapporto di verosimiglianza, il modello con la sola intercetta e il modello completo, appena stimato,
$W = 2(\ell(\hat{\beta}) - \ell(\tilde{\beta}))$
La stima del modello con la sola intercetta si ottiene come

```
ICU.fit0 = glm(stato ~ 1, family = binomial, data = ICU)
summary(ICU.fit0)
```

```
##
## Call:
## glm(formula = stato ~ 1, family = binomial, data = ICU)
##
## Deviance Residuals:
```

```
##    Min      1Q  Median      3Q     Max
## -0.668  -0.668  -0.668  -0.668   1.794
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.386      0.177   -7.84  4.4e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 200.16  on 199  degrees of freedom
## Residual deviance: 200.16  on 199  degrees of freedom
## AIC: 202.2
##
## Number of Fisher Scoring iterations: 4
```

Vale la pena notare che la probabilità stimata π_0, la stessa per tutti, è

```
p0 = exp(coef(ICU.fit0))/(1 + exp(coef(ICU.fit0)))
p0

## (Intercept)
##         0.2
```

che è pari alla frequenza osservata di decessi nel campione, infatti

```
table(ICU$stato)

##
##   0   1
## 160  40
```

La log verosimiglianza nel modello ridotto è pertanto
$\ell(\tilde{\beta}) = (\sum_{i=1}^{n} y_i)\log(\pi_0) + (n - \sum_{i=1}^{n} y_i)\log(\pi_0)$
si ha infatti

```
n = nrow(ICU)
s = sum(ICU$stato)
s * log(p0) + (n - s) * log(1 - p0)

## (Intercept)
##      -100.1
```

questo stesso risultato si può ottenere con

```
logLik(ICU.fit0)

## 'log Lik.' -100.1 (df=1)
```

Il test del rapporto di verosimiglianza è dunque

```
W = 2 * (logLik(ICU.fit) - logLik(ICU.fit0))
W

## 'log Lik.' 54.85 (df=4)
```

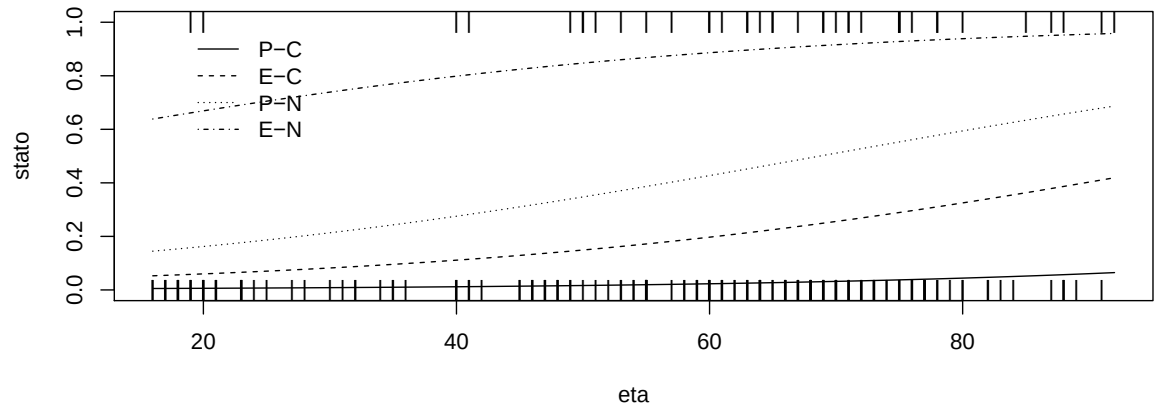
e quindi il valore p è

```
1 - pchisq(W, 3)

## 'log Lik.' 7.39e-12 (df=4)
```

Possiamo aggiungere al grafico le curve stimate:

```
plot(eta, stato, pch = "|", ylim = c(0, 1))
curve(predict(ICU.fit, newdata = data.frame(eta = x, causa = "P", coscienza = "SI"),
    type = "response"), add = TRUE, lty = 1)
curve(predict(ICU.fit, newdata = data.frame(eta = x, causa = "E", coscienza = "SI"),
    type = "response"), add = TRUE, lty = 2)
curve(predict(ICU.fit, newdata = data.frame(eta = x, causa = "P", coscienza = "NO"),
    type = "response"), add = TRUE, lty = 3)
curve(predict(ICU.fit, newdata = data.frame(eta = x, causa = "E", coscienza = "NO"),
    type = "response"), add = TRUE, lty = 4)
legend(18, 1, lty = c(1, 2, 3, 4), legend = c("P-C", "E-C", "P-N", "E-N"), bty = "n")
```



```
detach(ICU)
```

2 Dati sui neonati

```
## Loading required package:  MASS
```

Consideriamo i dati sulle nascite di neonati sottopeso (es. tratto dal volume di Hosmer & lameshow, usato durante il corso di TIS). Le nascite sottopeso vengono messe in relazione al peso della madre e al fatto che la madre sia o meno fumatrice.
Carichiamo il campione con:

```
neonati = read.table("neonati.dat", header = TRUE)
head(neonati)

##    Sottopeso Madre_Fumatrice Peso_della_Madre
## 85         0               0              182
## 86         0               0              155
## 87         0               1              105
## 88         0               1              108
## 89         0               1              107
## 91         0               0              124

attach(neonati)
```

e per essi consideriamo il modello di regressione logistica $\pi_i = P(Y_i = 1)$

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_1 + \beta_2 x_{i2} + \beta_2 x_{i3} + \beta_2 x_{i4}$$

dove Y_i la variabile che vale 1 se il neonato nasce sottopeso, x_2 la variabile che vale 1 se la madre è fumatrice e x_3 è il peso della madre.
Si ha allora

```
neonati.fit = glm(Sottopeso ~ ., data = neonati, family = binomial)
summary(neonati.fit)

##
## Call:
## glm(formula = Sottopeso ~ ., family = binomial, data = neonati)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
```

```
## -1.249  -0.846  -0.737   1.249   2.081
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.62200    0.79592    0.78    0.435
## Madre_Fumatrice   0.67667    0.32470    2.08    0.037 *
## Peso_della_Madre -0.01332    0.00609   -2.19    0.029 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 234.67  on 188  degrees of freedom
## Residual deviance: 224.34  on 186  degrees of freedom
## AIC: 230.3
##
## Number of Fisher Scoring iterations: 4
```

Si ottiene poi la tabella che confronta previsioni ottenute dal modello e valori osservati con

```
table((predict(neonati.fit, type = "response") > 0.5), neonati$Sottopeso)
```

```
##
##           0   1
##   FALSE 124  54
##   TRUE    6   5
```

Infine, otteniamo il grafico con le curve stimate mediante

```
plot(birthwt$lwt, birthwt$low, pch = "|", xlab = "Peso della madre", ylab = "Neonato sottopes
    yaxt = "n")
axis(2, at = c(0, 1), las = 2)
curve(predict(neonati.fit, newdata = data.frame(Madre_Fumatrice = 0, Peso_della_Madre = x),
    type = "response"), add = TRUE, lwd = 2)
curve(predict(neonati.fit, newdata = data.frame(Madre_Fumatrice = 1, Peso_della_Madre = x),
    type = "response"), add = TRUE, lty = 2, lwd = 2)
```