Regresión logística binomial

Contents

1 Regresión logística binaria y regresión logística binomial

```
d = read.csv("datos/MichelinNY.csv")
```

Los datos que analizamos con el modelo logit pueden estar codificados en dos maneras diferentes:

- Con ceros y unos. Este es el caso de los datos del archivo *MichelinNY.csv*, donde la variable respuesta tiene un 0 si el restaurante no está en la Guía Michelin y un 1 si está dentro de la Guía.

```
head(d)
```

```
##   InMichelin Restaurant.Name Food Decor Service Price
## 1          0  14 Wall Street   19    20      19    50
## 2          0             212   17    17      16    43
## 3          0        26 Seats   23    17      21    35
## 4          1              44   19    23      16    52
## 5          0               A   23    12      19    24
## 6          0          A.O.C.   18    17      17    36
```

Como la variable analizada, InMichelin, está codificada como 0 - 1, el modelo se denomina **regresión logística binaria**. Es el modelo que se ha estimado en las secciones precedentes.

- Con datos agrupados. En lugar de ceros y unos podemos indicar el numero de restaurantes que pertenecen a la Guía Michelin y que tienen un valor de la variable Food determinado. Por ejemplo:

```
(d1 = table(d$Food, d$InMichelin))
```

```
##
##       0  1
##   15  1  0
##   16  1  0
##   17  8  0
##   18 13  2
##   19 13  5
```

```
##    20 25  8
##    21 11 15
##    22  8  4
##    23  6 12
##    24  1  6
##    25  1 11
##    26  1  1
##    27  1  6
##    28  0  4
```

Es decir, para Food = 20 tenemos 25 + 8 = 33 restaurantes con esa puntuación, de los cuales 8 están en la Guía Michelín.

La probabilidad de que 8 de 33 restaurantes estén en la Guía se calcula de la siguiente manera. Definimos la variable aleatoria y_i : "número de restaurantes incluidos en la Guía para $x_i = 20$". Por tanto

$$P(y_i = 8) = \binom{33}{8} \pi_i^8 (1 - \pi_i)^{33-8}$$

donde π_i es la probabilidad de que un restaurante con puntuación 20 esté en la Guía. De manera general, si Y_i : "número de restaurantes incluidos en la Guía para un x_i dado" se tiene que:

$$P(Y_i = y_i) = \binom{m_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{m_i - y_i}$$

donde m_i es el número total de restaurantes para x_i. De la variable Y_i se dice que tiene distribución binomial (de ahí que se utilice la familia binomial en glm). Algunas propiedades de la distribución binomial son:

- Los valores que puede tomar la variable son $Y_i = 0, 1, \cdots, m_i$.
- La esperanza es: $E[Y_i] = m_i \cdot p_i$.
- El caso binario es un caso particular del caso binomial: $Y_i = 0, 1$, $m_i = 1$, $E[Y_i] = 1 \cdot p_i = p_i$.

2 Modelo

Los datos disponibles son

$$\begin{array}{ccc} y_1 & x_1 & m_1 \\ y_2 & x_2 & m_2 \\ \cdots & \cdots & \cdots \\ y_n & x_n & m_n \end{array}$$

Al igual que el modelo binario se trabaja con probabilidades:

$$P(Y_i = y_i) = \binom{m_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{m_i - y_i}, \quad y_i = 0, 1, \cdots, m_i$$

donde se adopta que:

$$\pi_i = \frac{exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)}$$

Ambas ecuaciones forman el modelo que vamos a utilizar para analizar los datos incluidos en la variable d1.

Otra forma de ver el modelo es:

$$Y_i = f(x_i) + u_i, \quad E[u_i] = 0,$$

Por tanto:

$$E[Y_i] = f(x_i)$$

Com Y_i tiene distribución binomial, su esperanza es

$$E[Y_i] = m_i \cdot \pi_i$$

lo que implica que

$$Y_i = \frac{m_i \cdot exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)} + u_i, \quad E[u_i] = 0$$

3 Estimación de los parámetros

3.1 La función de verosimilitud

Dada la muestra $\{Y_1 = y_1, Y_2 = y_2, \cdots, Y_n = y_n\}$, la probabilidad de obtener dicha muestra es:

$$P(Y_1 = y_1, Y_2 = y_2, \cdots, Y_n = y_n) = \prod_{i=1}^{n} P(Y_i = y_i) = \prod_{i=1}^{n} \binom{m_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{m_i - y_i}$$

Se denomina función de verosimilitud a la probabilidad de obtener la muestra:

$$L(\beta) = \prod_{i=1}^{n} \binom{m_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{m_i - y_i}$$

donde $\beta = [\beta_0 \quad \beta_1]^T$. El logaritmo de la verosimilitud es:

$$logL(\beta) = log \prod_{i=1}^{n} \binom{m_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{m_i - y_i} = \sum_{i=1}^{n} \left(log \binom{m_i}{y_i} + y_i log(\pi_i) + (m_i - y_i) log(1 - \pi_i) \right)$$

$$= \sum_{i=1}^{n} \left(y_i log \left(\frac{exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)} \right) + (m_i - y_i) log \left(1 - \frac{exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)} \right) + log \binom{m_i}{y_i} \right)$$

$$= \sum_{i=1}^{n} \left(y_i log \left(\frac{exp(x_i^T \beta)}{1 + exp(x_i^T \beta)} \right) + (m_i - y_i) log \left(\frac{1}{1 + exp(x_i^T \beta)} \right) + log \binom{m_i}{y_i} \right)$$

$$= \sum_{i=1}^{n} \left(y_i log(exp(\beta_0 + \beta_1 x_i) - y_i log(1 + exp(\beta_0 + \beta_1 x_i)) - (m_i - y_i) log(1 + exp(\beta_0 + \beta_1 x_i)) + log \binom{m_i}{y_i} \right)$$

$$= \sum_{i=1}^{n} \left(y_i (\beta_0 + \beta_1 x_i) - m_i log(1 + exp(\beta_0 + \beta_1 x_i)) + log \binom{m_i}{y_i} \right)$$

En R:

```
logLb = function(beta,y,x,m){
  # beta = [beta0 beta1]
  n = length(y)
  suma = 0
  for (i in 1:n){
    suma = suma + y[i]*(beta[1] + beta[2]*x[i]) -
      m[i]*log(1 + exp(beta[1] + beta[2]*x[i])) +
      log(choose(m[i],y[i]))
  }
  return(suma)
}
```

Por ejemplo, para $\beta_0 = -12$ y $\beta_1 = 1$, la función de verosimilitud vale:

```
y = d1[,2]
x = as.integer(row.names(d1))
m = d1[,1] + d1[,2]
```

```
beta = c(-12,1)
logLb(beta,y,x,m)
```

```
##        15
## -646.0697
```

3.2 El máximo de la función de verosimilitud

Tenemos que derivar e igualar a cero:

$$\frac{\partial logL(\beta)}{\partial \beta_0} = \sum_{i=1}^{n} \left(y_i - \frac{m_i exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)} \right) = \sum_{i=1}^{n} (y_i - m_i \pi_i)$$

$$\frac{\partial logL(\beta)}{\partial \beta_1} = \sum_{i=1}^{n} \left(y_i x_i - \frac{m_i x_i exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)} \right) = \sum_{i=1}^{n} x_i (y_i - m_i \pi_i)$$

En forma matricial tenemos el vector gradiente:

$$\frac{\partial logL(\beta)}{\partial \beta} = \begin{bmatrix} \frac{\partial logL(\beta)}{\partial \beta_0} \\ \frac{\partial logL(\beta)}{\partial \beta_1} \end{bmatrix} = \sum_{i=1}^{n} \begin{bmatrix} 1 \\ x_{1i} \end{bmatrix} (y_i - m_i \pi_i) = X^T (y - \pi) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

donde X es la matriz de regresores:

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \dots & \dots \\ 1 & x_n \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}, \quad \pi = \begin{bmatrix} m_1 \pi_1 \\ m_2 \pi_2 \\ \dots \\ m_n \pi_n \end{bmatrix}$$

De igual manera se obtiene la matriz hessiana:

$$\frac{\partial logL(\beta)}{\partial \beta \partial \beta^T} = \begin{bmatrix} \frac{\partial^2 logL(\beta)}{\partial \beta_0^2} & \frac{\partial^2 logL(\beta)}{\partial \beta_0 \partial \beta_1} \\ \frac{\partial^2 logL(\beta)}{\partial \beta_0 \partial \beta_1} & \frac{\partial^2 logL(\beta)}{\partial \beta_1^2} \end{bmatrix} = -\sum_{i=1}^{n} \begin{bmatrix} 1 \\ x_i \end{bmatrix} m_i \pi_i (1 - \pi_i) \begin{bmatrix} 1 & x_i \end{bmatrix} = -X^T W X$$

donde W es una matriz diagonal con

$$W_{ii} = m_i \pi_i (1 - \pi_i)$$

En R:

```
grad_logLb = function(beta,y,x,m){
  n = length(y)
  X = cbind(rep(1,n),x)
  y = matrix(y, nrow = n, ncol = 1)
  pi = matrix(0, nrow = n, ncol = 1)
  for (i in 1:n){
    pi[i,1] = m[i]*exp(beta[1] + beta[2]*x[i])/(1 + exp(beta[1] + beta[2]*x[i]))
  }
  grad = t(X) %*% (y - pi)
  return(grad)
}
```

Comprobacion:

```
beta = c(-12,1)
grad_logLb(beta, y, x, m)
```

```
##           [,1]
##      -89.81236
## x -1791.80199
```

```
hess_logLb = function(beta,x,m){
  n = length(x)
  X = cbind(rep(1,n),x)
  W = matrix(0, nrow = n, ncol = n)
  for (i in 1:n){
    pi = exp(beta[1] + beta[2]*x[i])/(1 + exp(beta[1] + beta[2]*x[i]))
    W[i,i] = m[i]*pi*(1-pi)
  }
  hess = -t(X) %*% W %*% X
  return(hess)
}
```

```
beta = c(-12,1)
hess_logLb(beta, x, m)
```

```
##                        x
##   -0.1845959  -3.150987
## x -3.1509868 -54.265816
```

```
# fdHess calcula el gradiente y el hessiano numéricamente,
# mediante diferencias finitas (para comprobar)
nlme::fdHess(beta,logLb, y, x , m)
```

```
## $mean
## [1] -646.0697
##
## $gradient
## [1]   -89.81236 -1791.80199
##
## $Hessian
##            [,1]       [,2]
## [1,] -0.1846241  -3.150774
```

```
## [2,] -3.1507741 -54.263073
```

4 El máximo de la función de verosimilitud

Lo vamos a calcular con optim():

```
logLb_optim = function(beta,y,x,m){
  logL = logLb(beta,y,x,m)
  return(-logL)
}
```

```
m1 = lm(y/m ~ x)
beta_i = coef(m1)
mle = optim(par = beta_i, fn = logLb_optim, y, x, m, gr = NULL, method = "BFGS", hessian = TRUE, contro
```

```
## initial  value 43.461359
## iter   2 value 38.182346
## iter   3 value 22.941678
## iter   4 value 19.215416
## iter   5 value 18.773934
## iter   6 value 18.745943
## iter   7 value 18.745823
## iter   8 value 18.745691
## iter   9 value 18.745560
## iter   9 value 18.745560
## iter   9 value 18.745560
## final  value 18.745560
## converged
```

```
mle$par
```

```
## (Intercept)           x
## -10.8416674   0.5012424
```

4.1 Estimacion con R

```
y = cbind(d1[,2], d1[,1]) # la primera columna tiene que ser la de 1
x = as.integer(row.names(d1))
m2 = glm(y ~ x, family = binomial)
summary(m2)
```

```
##
## Call:
## glm(formula = y ~ x, family = binomial)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.84154    1.86236  -5.821 5.84e-09 ***
## x             0.50124    0.08768   5.717 1.08e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 61.427  on 13  degrees of freedom
```

```
## Residual deviance: 11.368  on 12  degrees of freedom
## AIC: 41.491
##
## Number of Fisher Scoring iterations: 4
```

5 Interpretación de π_i

En el caso de datos agrupados, π_i es el valor de la probabilidad de que un restaurante esté en la Guía Michelin dada su puntuación. Podemos representar los valores obtenidos de los datos junto a los valores estimados por el modelo para estas probabilidades:

```
prob_observada = d1[,2]/m
prob_estimada = exp(coef(m2)[1] + coef(m2)[2]*x)/(1+exp(coef(m2)[1] + coef(m2)[2]*x))
plot(x,prob_observada)
lines(x,prob_estimada, col = "red", lty = 2)
```