EST 662 - Examen final - Primavera 2014

Christian I. Ponce

21 de abril de 2014

1. Creación de la muestra

1.1. Matriz X

```
randx = function(n, j, mu, sigma) {
    # n=número de filas j=número de columnas
    X = matrix(ncol = j, nrow = n)
    uno = rep(1, n)
    for (i in 1:n) {
        for (j in 1:j) {
            X[i, j] = rnorm(1, mean = (mu - j), sd = sigma)
        }  #cierra for(j in 1:j){
    }  #cierra for(i in 1:n){
    return(X = cbind(uno, X))
}
```

Generar una matriz de X, $dim(X) = 25 \times 10$

```
X = randx(n = 25, j = 10, mu = 11, sigma = 1)
```

1.2. Vector de medias μ

```
beta = c(5, seq(1, 10, 1))/10
```

```
mu = X %*% beta
```

1.3. Muestra y $Y_i \sim \log -N(\mu_i, 1)$

```
yi = rlnorm(25, meanlog = mu, sdlog = 1)
```

1.4. Asignación de datos censurados a la muestra

```
c = sort(yi)[20]
cen = yi > c  # qué datos son sensurados
scen = yi <= c  # qué datos no son cesurados
yi[cen] = exp(25)
```

2. Estimación de parámetros

2.1. Algoritmo EM

Este algoritmo es una adaptación de la función `em.regcen.completa()` adaptado para la regresión lineal multivariada $\log(Y_i) \sim N(\mu_i, \sigma^2)$:

```
em.censurada = function(y, x, c, n) {
    # y vector de datos censurados ~\sim N(mu_i,sigma) x matriz de datos [1,x*]
    # c valor en el que están truncadas las observaciones
    cen = y >= c
    no.cen = y < c
    ly = lm(y[no.cen] ~ X[no.cen, -1])
    b = coef(ly)
    sigma = sqrt(deviance(ly)/(sum(no.cen)))
    for (i in 1:n) {
        media = c(X %*% b)
        z = (y - media)/sigma
        sesgo = sigma * dnorm(z)/(1 - pnorm(z))
        y[cen] = c(media + sesgo)[cen]
        ly = lm(y ~ X[, -1])
        b = coef(ly)
        suma = sum((z * sesgo * sigma + sigma^2 - sesgo^2)[cen])
        sigma = sqrt((deviance(ly) + suma)/nrow(X))
        if (i == n)
            {
                return(ly)
            }  #cierra if(i==n){
    }  #for(i in 1:n){
}  #cierra function(y,x,c,n)
```

2.2. Transformación de los datos

Si $Y_i \sim \log\text{–}N(\mu_i, 1)$, entonces $\log(Y_i) \sim N(\mu_i^*, \sigma^2)$

```
y = log(yi)
```

```
xtable(matrix(y, nrow = 5), caption = "Muestra aleatoria $\\log(y)$, los datos $\\log(y)>25$ fue-
ron truncados en 25")
```

	1	2	3	4	5
1	25.00	22.17	24.73	22.92	25.00
2	20.17	20.95	25.00	21.52	21.43
3	21.20	24.25	22.44	24.45	20.58
4	19.30	25.20	20.70	23.78	20.31
5	25.00	21.47	24.90	21.20	25.00

Cuadro 1: Muestra aleatoria $\log(y)$, los datos $\log(y) > 25$ fueron truncados en 25

2.3. Estimación de los parámetros

Utilizando el algoritmo modificado, se estimó los parámetros junto con las observaciones faltantes:

```
m = 2
yc = em.censurada(y, X, 25, n = m)  #n=1 el sesgo es muy peqeño
```

En el Cuadro 2 se muestran los coeficientes estimados hasta la m-ésima iteración.

```
beta = matrix(yc$coefficients, nrow = 1)
colnames(beta) = c("b0", "b1", "b2", "b3", "b4", "b5", "b6", "b7", "b8", "b9",
    "b10")
```

```
xtable(beta, caption = "Coeficientes estimados tras $m$ iteraciones", label = "tab:coef",
    digits = 4)
```

	b0	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10
1	-1.2580	-0.0476	0.1556	1.0520	-0.2198	0.9771	0.6019	0.6948	0.7638	0.4836	1.4305

Cuadro 2: Coeficientes estimados tras m iteraciones

El algoritmo como tal, no converge, por ejemplo, tras 2 iteraciones, los valores de $\hat{y}_i$ son:

```
yc$fit[cen]
```

1 5 12 21 25 27.97 26.93 25.74 25.90 26.30

Sin embargo, si el número de iteraciones aumenta, los valores $\hat{y}_i$ crecen sin una cota superior:

```
m = 10
yc = em.censurada(y, X, 25, n = m)  #n=1 el sesgo es muy peqeño
```

```
xtable(matrix(yc$fit[cen], ncol = 5), caption = "Estimación de las observaciones $y_i$ truncadas")
```

	1	2	3	4	5
1	31.03	29.89	29.22	29.17	28.35

Cuadro 3: Estimación de las observaciones y_i truncadas

En el Cuadro 4, se observa que los coeficientes no son iguales a los del Cuadro 2, a pesar de utilizar la misma matriz **X** y el mismo vector de observaciones **y**.

```
beta = matrix(yc$coefficients, nrow = 1)
colnames(beta) = c("b0", "b1", "b2", "b3", "b4", "b5", "b6", "b7", "b8", "b9",
    "b10")
xtable(beta, caption = "Coeficientes estimados tras $m$ iteraciones", label = "tab:coef",
    digits = 4)
```

	b0	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10
1	-9.5649	0.2763	0.1718	1.6050	-0.7097	1.6630	0.4900	0.5786	1.0978	0.3591	2.1807

Cuadro 4: Coeficientes estimados tras m iteraciones

3. Conclusiones

Por algún motivo el sesgo decrece en las primeras 2 iteraciones, pero después de este punto comienza a crecer lentamente y después rápidamente (estos datos no se muestran).

Faltaría analizar más a fondo el algoritmo para identificar si esto es un error dentro de la función programada o se debe a las características propias de la distribución.

Se descarta que esto sea un efecto ocasionado por una mala muestra porque durante el desarrollo de este trabajo, su utilizó varias muestra distintas, tanto para la matriz **X** como para el vector de observaciones **y**, obteniendo siempre resultados similares en la convergencia.