Funciones avanzadas de regresión en R

F. Tusell*

Curso 2.011-2.012

1. Introducción

La función `lsfit` y asociadas proporcionan toda la funcionalidad necesaria para hacer regresión lineal. La función `lm` proporciona la misma funcionalidad más cómodamente. Además, la sintaxis es homogénea y similar a la de otras funciones de regresión generalizada como `glm` y `gam`.

Puedes consultar el modo de empleo en las notas de clase, en la ayuda on-line del sistema, y en Chambers & Hastie (1992). Ni la función `lm` ni las `glm` y `gam` están documentadas en Becker et al. (1988). Puedes también servirte de Fox (2002), Venables & Ripley (1999*a*) (junto con sus complementos on-line, Venables & Ripley (1999*b*)), Ugarte et al. (2008) y de Kuhnert & Venables (2005).

Este documento contiene sólo la transcripción de una pequeña sesión, sobre un conjunto de datos que conoces, junto con comentarios que remiten a los números de línea del listado en el Apéndice A. Puedes leer lo que sigue junto con el Cap. 7 de las notas de clase, más detalladas.

2. Comentarios

Linea 21. Se ha empleado un `read.table` para leer los datos en el fichero `presis.dat`. Como dicho fichero tiene una primera línea con los nombres de las variables y `read.table` se ha invocado con la opción `header=TRUE`, las variables han quedado nombradas sin nuestra intervención (pero, a efectos didácticos, se renombran más abajo).

Variables con valores alfanuméricos, como "REP" y "DEM", son leidas por `read.table` como una variable cualitativa (un factor, en la terminología de R), en este caso con dos niveles o valores.

*Actualización del día 10 de noviembre de 2011. La última versión de este documento, posiblemente más reciente, puede obtenerse de `http://www.et.bs.ehu.es/~etptupaf` o de Moodle.

Obsérvese que `read.table` devolvería *dataframes* incluso aunque los datos fueran totalmente numéricos[1].

Lineas 29 La función `dimnames` aplicada a una matriz (o *dataframe*) permite obtener los nombres de filas y columnas o asignarlos. Ello es interesante si se quiere obtener una salida inteligible y autodocumentada.

Los nombres de filas y columnas de una matriz (o *dataframe*) se almacenan en una lista de dos componentes: el primero[2] nombra las filas, y el segundo las columnas. En esta línea se han nombrado las columnas. Las filas han resultado nombradas automáticamente por la función `read.table`; cuando se detecta una columna "sobrante", se interpreta que es la primera y que da nombre a las filas.

Linea 30. Si se quieren conocer los niveles de un factor (o variable cualitativa), la función `levels` permite obtenerlos. Si se deseara reconvertirlos a valores numéricos, podría hacerse fácilmente con la función `codes`.

Linea 32. Se define una fórmula. La variable a la izquierda de la ~ es el regresando o respuesta. Las variables a la derecha, los regresores, o funciones de ellos. Las variables se evalúan en la *dataframe* que se suministra como argumento `data=`. Si no lo hubiera, las variables se buscarían en el entorno en el espacio de trabajo.

Linea 33. Se estima la regresión cuya fórmula funcional viene dada por `form1` sobre los datos en la *dataframe* `presis`. No es estrictamente necesario definir previamente la fórmula; puede hacerse directamente en `lm` invocando la función así:

```
lm(Edad ~ Partido + Eleccion,data=presis)
```

El resultado de la regresión se asigna a `ajuste1`. Es un objeto compuesto por múltiples componentes. Si tecleamos su nombre podemos ver como valor alguna información sobre la regresión estimada. Obtendríamos una información mas completa tecleando

```
anova(ajuste1)
summary(ajuste1)
```

Esto último se ha hecho en la línea 59 con `ajuste2`. Si se desea ver "todo" lo que contiene un objeto compuesto, es útil la función `str`.

[1] Si por algun motivo nos interesara convertir la *dataframe* en una matriz, podríamos hacerlo mediante la función `as.matrix`.

[2] Que se referencia mediante `[[1]]`; recuerda que los índices de listas se escriben entre dobles corchetes, a diferencia de los de vectores.

Linea 43. Una regresión puede re-estimarse mediante la función `update`, entregando como argumentos el ajuste precedente y la nueva fórmula o regresión que se desea estimar. En la especificación de ésta última, el punto, `.` , designa "lo que ya había". En esta línea se ha añadido a los regresores presentes en `ajuste1` el término `Eleccion^2`.

Linea 53. Componentes seleccionados del resultado de `lm` se obtienen mediante funciones especializadas que retornan sus valores.

Linea 79. La instrucción

```
postscript(file="fichero")
```

redirige las salidas gráficas a un fichero de nombre "fichero". Una simple instrucción `plot` hace un gráfico mostrando el ajuste obtenido. No es preciso especificar qué tipo de gráfico queremos, rótulos de los ejes de ordenadas y abscisas, etc.: toda esta información existe ya en el objeto `ajuste2` que damos como argumento.

El gráfico queda almacenado en el directorio de trabajo, en un fichero Postscript del nombre que hayamos dado, cuando se cierra el dispositivo mediante una instrucción `dev.off()`. Podemos incluir dicho gráfico en un documento.

No es preciso hacer nada especial si sólo se desea el gráfico en pantalla. Alternativamente, puede obtenerse, en lugar de en Postscript, en otros muchos formatos, como PDF (con la instrucción `pdf`, JPG, PNG, etc. Puedes escoger el formato de salida que más te convenga para insertar en tu documento.

> Si empleas el procedimiento de trabajo sugerido en clase (y documentado en Tusell (2005)), no tienes que preocuparte de nada: los gráficos se insertan automáticamente en tu documento LaTeX.

Lineas 81. Como argumento `data=` de `lm` hemos de dar la *dataframe* de la que se sacan las variables empleadas. Para evitar hacer esto de modo repetitivo, podemos vincular una *dataframe* mediante `attach`: hasta el final de la sesión (o un nuevo `attach`), los nombres de variables que tecleemos se buscarán entre las columnas de dicha *dataframe* (además de entre los objetos en el espacio de trabajo).

El uso de `attach` es peligroso no obstante: si por descuido se define —o existía ya en el espacio de trabajo— una variable con el mismo nombre que una columna de la dataframe "attacheada", la columna quedará inaccesible, y la variable en el espacio de trabajo será la utilizada. Esto puede dar lugar a errores muy difíciles de trazar. Mejor teclea un poco más y señala `data=` cada vez.

Linea 91. Podemos introducir funciones de los regresores sin tener que crear explícitamente las columnas correspondientes en la *dataframe*. Hay funciones auxiliares como `poly` que facilitan aún mas las cosas, permitiendo crear polinomios del orden deseado en una variable[3]. Se eliminan automáticamente las columnas que darían lugar a dependencia lineal exacta con la columna de "unos", si la hay. Esto ocurre en particular con las columnas de unos y ceros representando factores[4].

Linea 92. Un término como `Partido*Eleccion` significa el producto (interacción) de `Partido` y `Eleccion` *y los términos de orden inferior* (en este caso, los efectos simples `Partido` y `Eleccion`). Si quisiéramos sólo la interacción, especificaríamos `Partido:Eleccion`. (Si tuviéramos dos variables numéricas, `V1` y `V2`, y realmente deseáramos un regresor que fuera el producto ordinario de ellas, habríamos de escribirlo como `I(V1*V2)`.)

Factores y factores ordenados. En el ejemplo que se comenta hay un factor (`Partido`) que podemos ver como una variable nominal: hay dos estados, sin orden natural entre ellos. En el extremo opuesto tendríamos variables medibles en una escala continua y bien definida: temperatura, peso, edad, longitud, etc. Entre los dos extremos, tenemos variables que admiten un número finito de estados entre los cuáles existe un orden natural: son las variables ordinales.

Por ejemplo, en las respuestas a una encuesta podemos tener cinco posibilidades: "Muy de acuerdo", "De acuerdo", "Indiferente", "En desacuerdo" y "Muy en desacuerdo". Hay un orden natural: "Muy de acuerdo" es "más" de acuerdo que simplemente "De acuerdo", pero no podriamos decir si es el doble o tres veces más de acuerdo. Sólo el orden está definido, no la magnitud relativa.

R soporta el concepto de factores ordenados para este tipo de variables. Por omisión, un factor ordinario (correspondiente a una variable nominal) se trata creando el número preciso de columnas de ceros y unos en la matriz de diseño. Un factor ordenado se trata asignando valores numéricos a los estados que reflejan su orden, e incluyendo como columnas en la matriz de diseño polinomios ortogonales evaluados para dichos valores.

En todos los casos, si no se especifica otra cosa, se elimina uno de los niveles (=una de las columnas de ceros y unos) para evitar la multicolinealidad exacta que de otro modo tendríamos. Véanse más detalles en Chambers & Hastie (1992) y en las notas de clase, Sec. 7.1.

[3]Se emplean polinomios ortogonales, para mejorar las estabilidad numérica de los cálculos. Mira por ejemplo `http://es.wikipedia.org/wiki/Polinomios_ortogonales` o Seber (1977), Sec. 8.2.

[4]El modo exacto en que se estiman los parámetros correspondientes a factores, es configurable: véase la opcion `contrasts=` en la ayuda *on line* o en Chambers & Hastie (1992).

3. Versiones e incompatibilidades

El listado en la Sección A utiliza la versión 2.14.0 (para Linux) de R. El mismo código funcionaría sin problemas en R bajo cualquier otro sistema operativo.

Referencias

Becker, R. A., Chambers, J. M. & Wilks, A. R. (1988), *The New S Language. A Programming Environment for Data Analysis and Graphics*, Wadsworth & Brooks/Cole, Pacific Grove, California.

Chambers, J. M. & Hastie, T. J. (1992), *Statistical Models in S*, Wadsworth & Brooks/Cole, Pacific Grove, Ca.

Dalgaard, P. (2002), *Introductory Statistics with R*, Statistics and Computing, Springer-Verlag. Signatura: 519.682 DAL.

Faraway, J. J. (2005), *Linear Models with R*, Chapman & Hall/CRC. Signatura: 519.233 FAR.

Fox, J. (2002), *An R and S-Plus Companion to Applied Regression*, Sage Pub.

Kuhnert, P. & Venables, W. (2005), *An Introduction to R: Software for Statistical Modelling and Computing*, CSIRO Mathematical and Information Sciences, Cleveland, Australia.

Seber, G. A. F. (1977), *Linear Regression Analysis*, Wiley, New York.

Tusell, F. (2005), *Formas de Utilización de R y Uso de Emacs + ESS*.

Ugarte, M., Militino, A. & Arnholt, A. (2008), *Probability and Statistics with R*, CRC Press.

Venables, W. & Ripley, B. (1999*a*), *Modern Applied Statistics with S-Plus*, third edn, Springer-Verlag, New York.

Venables, W. & Ripley, B. D. (1999*b*), 'R complements to *Modern Applied Statistics with S-Plus.*', En `http://www.stats.ox.ac.uk/pub/MASS3`.

A. Transcripción de una sesión

```
1
2 R version 2.14.0 (2011-10-31)
3 Copyright (C) 2011 The R Foundation for Statistical Computing
4 ISBN 3-900051-07-0
5 Platform: i686-pc-linux-gnu (32-bit)
6
7 R es un software libre y viene sin GARANTIA ALGUNA.
8 Usted puede redistribuirlo bajo ciertas circunstancias.
9 Escriba 'license()' o 'licence()' para detalles de distribucion.
10
11 R es un proyecto colaborativo con muchos contribuyentes.
12 Escriba 'contributors()' para obtener más información y
13 'citation()' para saber cómo citar R o paquetes de R en publicaciones.
14
15 Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
16 o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
17 Escriba 'q()' para salir de R.
18
19 [Previously saved workspace restored]
20
21 > presis <- read.table("presis.dat",header=TRUE)
22 > presis[1:5,]
23                    Partido FechaElec Edad
24 Franklin Pierce          DEM      1853   48
25 James Buchanan           DEM      1857   65
26 Abraham Lincoln-I        REP      1861   52
27 Abraham Lincoln-II       REP      1865   56
28 Andrew Johnson           REP      1865   56
29 > dimnames(presis)[[2]] <- c("Partido","Eleccion","Edad")
30 > levels(presis[,"Partido"])
31 [1] "DEM" "REP"
32 > form1 <- Edad ~ Partido + Eleccion
33 > ajuste1 <- lm(form1,data=presis)
34 > ajuste1
35
36 Call:
37 lm(formula = form1, data = presis)
38
39 Coefficients:
40 (Intercept)   PartidoREP      Eleccion
41   -34.17183       1.86698       0.04565
42
43 > ajuste2 <- update(ajuste1, . ~ . + I(Eleccion^2))
44 > ajuste2
45
46 Call:
47 lm(formula = Edad ~ Partido + Eleccion + I(Eleccion^2), data = presis)
```

```
48
49 Coefficients:
50   (Intercept)      PartidoREP        Eleccion   I(Eleccion^2)
51    -1.395e+03       1.952e+00       1.457e+00      -3.657e-04
52
53 > coef(ajuste2)
54   (Intercept)     PartidoREP       Eleccion I(Eleccion^2)
55 -1.395318e+03  1.952426e+00  1.456977e+00 -3.656543e-04
56 > form2 <- formula(ajuste2)
57 > form2
58 Edad ~ Partido + Eleccion + I(Eleccion^2)
59 > summary(ajuste2)
60
61 Call:
62 lm(formula = Edad ~ Partido + Eleccion + I(Eleccion^2), data = presis)
63
64 Residuals:
65      Min       1Q   Median       3Q      Max
66 -12.9464  -4.1549   0.0566   3.9258  15.6514
67
68 Coefficients:
69                  Estimate Std. Error t value Pr(>|t|)
70 (Intercept)    -1.395e+03  1.963e+03  -0.711    0.481
71 PartidoREP      1.952e+00  1.985e+00   0.984    0.331
72 Eleccion        1.457e+00  2.035e+00   0.716    0.478
73 I(Eleccion^2) -3.657e-04  5.273e-04  -0.693    0.492
74
75 Residual standard error: 6.575 on 43 degrees of freedom
76 Multiple R-squared: 0.1102,     Adjusted R-squared: 0.04808
77 F-statistic: 1.774 on 3 and 43 DF,  p-value: 0.1663
78
79 > postscript(file="fichero.ps")
80 > plot(ajuste2)
81 > attach(presis)
82 > Partido
83  [1] DEM DEM REP REP REP REP REP REP REP REP DEM REP DEM REP REP REP REP REP DEM
84 [20] DEM REP REP REP REP DEM DEM DEM DEM DEM DEM REP REP DEM DEM DEM REP REP REP
85 [39] DEM REP REP REP DEM DEM REP REP DEM
86 Levels: DEM REP
87 > ajuste3 <- lm(Edad ~ Partido + poly(Eleccion,2))
88 > mframe <- lm(Edad ~ Partido + poly(Eleccion,2),
89 +                       method="model.frame")
90 > ajuste4 <- lm(Edad ~ Partido + Eleccion - 1)
91 > ajuste5 <- lm(Edad ~ Partido + poly(Eleccion,3))
92 > ajuste6 <- lm(Edad ~ Partido*Eleccion)
93 > drop1(ajuste5,keep=T)
94 Single term deletions
95
96 Model:
```

```
Edad ~ Partido + poly(Eleccion, 3)
                   Df Sum of Sq    RSS    AIC
<none>                          1521.5 173.43
Partido             1    100.77 1622.3 174.45
poly(Eleccion, 3)   3    551.62 2073.1 181.97
> vacio <- lm(Edad ~ 1)
> add1(vacio, . ~ Partido + Eleccion + I(Eleccion^2))
Single term additions

Model:
Edad ~ 1
                 Df Sum of Sq    RSS    AIC
<none>                       2089.0 180.33
Partido           1    15.873 2073.1 181.97
Eleccion          1   170.928 1918.0 178.32
I(Eleccion^2)     1   169.769 1919.2 178.35
> q()
> proc.time()
   user  system elapsed
  0.924   0.052   0.940
```