Análisis caso ABB

Jordi López Sintas

3 de septiembre de 2014

Lectura de datos

La base de datos completa con los datos de las valoraciones, elecciones de las empresas así como datos descriptivos de la empresa se encuentra en el fichero "abb-r.txt". Para leerlo utilizamos la función `read.table` con los parámetros adecuados como vemos en el código que se muestra. También cargamos los paquetes `ggplot` y `dplyr`. El primero si queremos visualizar los datos (es opcional, pues podemos utilizar las funciones gráficas del paquete base) y el segundo para realizar tablas de datos.

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(dplyr)
```

```
## Loading required package: dplyr
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:stats':
##
##     filter, lag
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#leer el fichero de datos abb-R.txt, el cual contiene los datos de la elección de las empresas eléctricas y la descripción de su volumen de compras así cómo su distrito.
abb<-read.table("abb-r.txt", header=T)

#También podemos utilizar la opción file.chose() como argumento de la función read.table. #Este argumentos nos permite escoger un fichero de datos guardado en el ordenador local)
#abb<-read.table(file.choose(), header=T)

#la función head() nos permite visualizar las primer seis líneas de un objeto de datos.
#He traspuesto el resupado con la función t() con el objeto de facilitar la lectura. Así las líneas representan las variables y las columnas los
```

```r
valors para las 6 primeras observaciones.
t(head(abb))
```

```
##                 1     2     3              4        5     6
## id              "1"   "1"   "1"            "1"      "2"   "2"
## Alternatives    "ABB" "GE"  "Westinghouse" "Edison" "ABB" "GE"
## choice          "0"   "1"   "0"            "0"      "0"   "0"
## price           "6"   "6"   "6"            "5"      "3"   "3"
## energy_loss     "6"   "6"   "5"            "5"      "4"   "4"
## maintenance     "7"   "6"   "7"            "6"      "5"   "5"
## warranty        "6"   "7"   "5"            "7"      "4"   "4"
## spare_parts     "6"   "9"   "3"            "8"      "4"   "7"
## ease_install    "5"   "9"   "4"            "2"      "5"   "3"
## problem_solving "7"   "7"   "7"            "6"      "6"   "5"
## quality         "5"   "5"   "6"            "5"      "4"   "5"
## DA              "1"   "0"   "0"            "0"      "1"   "0"
## DB              "0"   "1"   "0"            "0"      "0"   "1"
## DC              "0"   "0"   "1"            "0"      "0"   "0"
## DD              "0"   "0"   "0"            "1"      "0"   "0"
## volume          "761" "761" "761"          "761"    "627" "627"
## district        "1"   "1"   "1"            "1"      "1"   "1"
```

```r
#la funcion names() muestra los nombres de las variables
names(abb)
```

```
##  [1] "id"              "Alternatives"    "choice"
##  [4] "price"           "energy_loss"     "maintenance"
##  [7] "warranty"        "spare_parts"     "ease_install"
## [10] "problem_solving" "quality"         "DA"
## [13] "DB"              "DC"              "DD"
## [16] "volume"          "district"
```

```r
#La función str() nos proporciona una descripción de la base de datos
str(abb)
```

```
## 'data.frame':    352 obs. of  17 variables:
##  $ id             : int  1 1 1 1 2 2 2 2 3 3 ...
##  $ Alternatives   : Factor w/ 4 levels "ABB","Edison",..: 1 3 4 2 1 3
4 2 1 3 ...
##  $ choice         : num  0 1 0 0 0 0 0 1 1 0 ...
##  $ price          : num  6 6 6 5 3 3 4 4 6 5 ...
##  $ energy_loss    : num  6 6 5 5 4 4 5 5 6 6 ...
##  $ maintenance    : num  7 6 7 6 5 5 5 6 7 7 ...
##  $ warranty       : num  6 7 5 7 4 4 5 5 7 7 ...
##  $ spare_parts    : num  6 9 3 8 4 7 5 4 6 5 ...
##  $ ease_install   : num  5 9 4 2 5 3 7 5 7 6 ...
##  $ problem_solving: num  7 7 7 6 6 5 6 5 7 8 ...
##  $ quality        : num  5 5 6 5 4 5 4 6 6 6 ...
##  $ DA             : num  1 0 0 0 1 0 0 0 1 0 ...
##  $ DB             : num  0 1 0 0 0 1 0 0 0 1 ...
##  $ DC             : num  0 0 1 0 0 0 1 0 0 0 ...
```

```
##  $ DD             : int  0 0 0 1 0 0 0 1 0 0 ...
##  $ volume         : int  761 761 761 761 627 627 627 627 643 643 ...
##  $ district       : int  1 1 1 1 1 1 1 1 2 2 ...
```

```
#cambiar la clase de las variables según sea apropiado.
#las variables choice y district deberían se factores.
abb$district <- as.factor(abb$district)
abb$choice <- as.factor(abb$choice)
```

Exploración de los datos

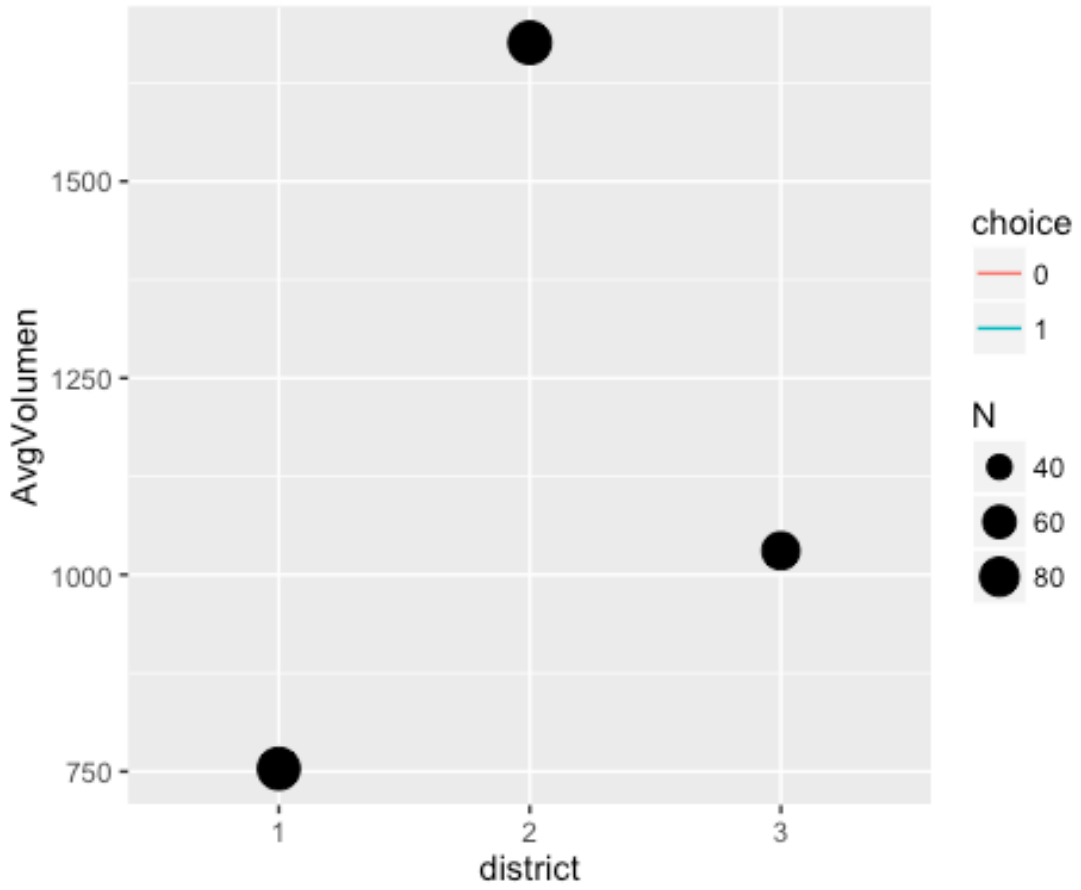
Con el pauete `dplyr` podemos rápidamente realizar informes con la base de datos.

```
#Ahora con la ayuda de la función select() del paquete dplyr
#y del operador tubería (pipeline) %>% calculamos el valor medio del
precio y volumen
A= select(abb, choice, volume, district, price) %>%
  group_by(district, choice) %>%
  summarize(AvgPrice = mean(price), AvgVolumen = mean(volume), N =
length(price))
A
```

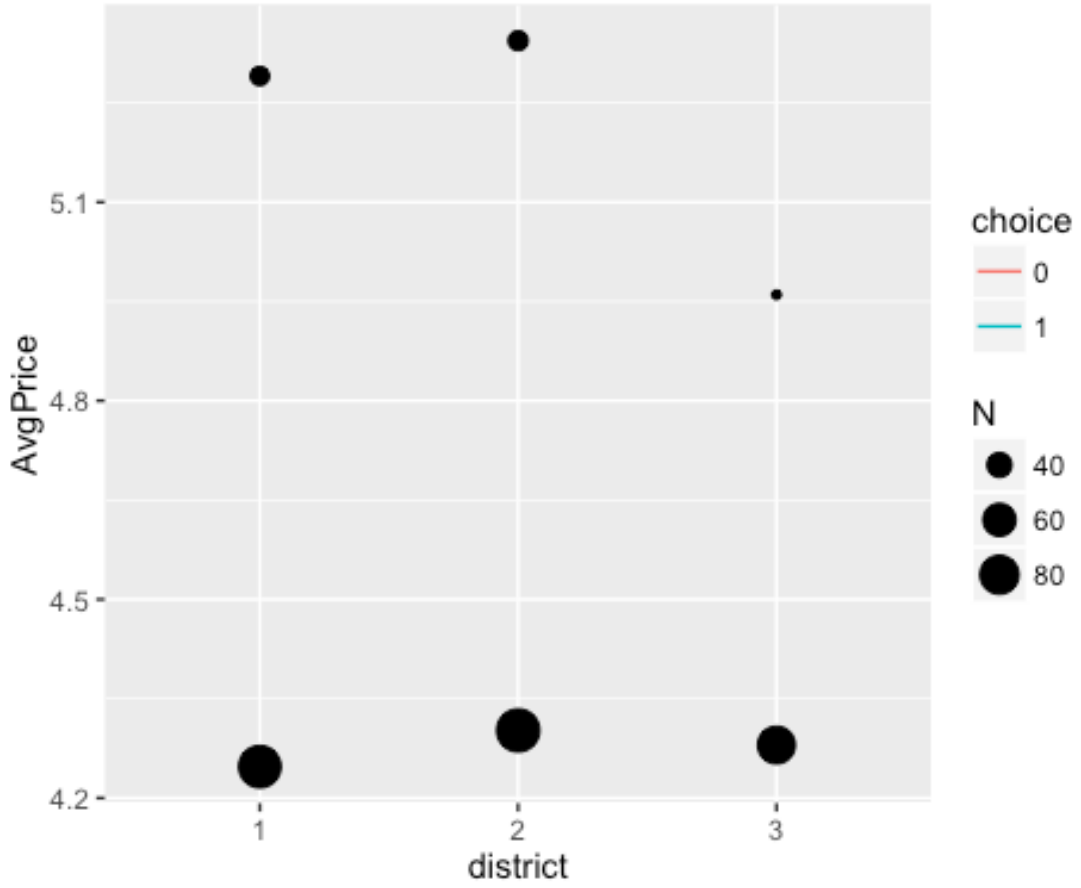
```
## Source: local data frame [6 x 5]
## Groups: district [?]
##
##   district choice AvgPrice AvgVolumen     N
##     (fctr) (fctr)    (dbl)      (dbl) (int)
## 1        1      0 4.247312   753.5161    93
## 2        1      1 5.290323   753.5161    31
## 3        2      0 4.302083  1676.2188    96
## 4        2      1 5.343750  1676.2188    32
## 5        3      0 4.280000  1030.4400    75
## 6        3      1 4.960000  1030.4400    25
```

You can also embed plots, for example:

```
## geom_path: Each group consists of only one observation. Do you need to
## adjust the group aesthetic?
```




```
## geom_path: Each group consists of only one observation. Do you need to
## adjust the group aesthetic?
```



Análisis

Ahora cargamos el paquete `survival` para poder utilizar la función `clogit` para estimar los parámetros del modelo de elección discreta.

```
## Call:
## coxph(formula = Surv(rep(1, 352L), choice) ~ price + energy_loss +
##     maintenance + warranty + spare_parts + ease_install +
problem_solving +
##     quality + DA + DB + DC + strata(id), data = abb, method = "exact")
##
##   n= 352, number of events= 88
##
##                    coef exp(coef) se(coef)      z Pr(>|z|)
## price            2.1806    8.8515   0.5866  3.717 0.000201 ***
## energy_loss      2.6556   14.2337   0.6737  3.942 8.09e-05 ***
## maintenance      0.5937    1.8107   0.4370  1.358 0.174313
## warranty         1.1407    3.1290   0.3310  3.446 0.000568 ***
## spare_parts     -0.1326    0.8758   0.2176 -0.610 0.542158
## ease_install     0.5200    1.6821   0.1729  3.008 0.002629 **
## problem_solving  2.0322    7.6307   0.5497  3.697 0.000218 ***
```

```
## quality          2.6394   14.0050   0.6877  3.838 0.000124 ***
## DA              -0.1238    0.8836   0.6785 -0.182 0.855241
## DB              -0.6712    0.5111   0.7194 -0.933 0.350814
## DC              -0.6872    0.5030   0.7150 -0.961 0.336499
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                 exp(coef) exp(-coef) lower .95 upper .95
## price              8.8515    0.11298    2.8036    27.945
## energy_loss       14.2337    0.07026    3.8006    53.306
## maintenance        1.8107    0.55228    0.7688     4.264
## warranty           3.1290    0.31959    1.6355     5.986
## spare_parts        0.8758    1.14182    0.5718     1.342
## ease_install       1.6821    0.59451    1.1986     2.360
## problem_solving    7.6307    0.13105    2.5982    22.410
## quality           14.0050    0.07140    3.6381    53.913
## DA                 0.8836    1.13178    0.2337     3.341
## DB                 0.5111    1.95662    0.1248     2.093
## DC                 0.5030    1.98821    0.1238     2.043
##
## Rsquare= 0.411   (max possible= 0.5 )
## Likelihood ratio test= 186.4  on 11 df,   p=0
## Wald test            = 23.67  on 11 df,   p=0.01419
## Score (logrank) test = 103.3  on 11 df,   p=0
```

También podemos elaborar unas tablas de resultados más profesionales con la función stargazer()

```
##
## Please cite as:
##
##  Hlavac, Marek (2015). stargazer: Well-Formatted Regression and
Summary Statistics Tables.
##  R package version 5.2. http://CRAN.R-project.org/package=stargazer

##
## Regression
## ================================================
##                           Dependent variable:
##                       ---------------------------
##                                  choice
## ------------------------------------------------
## price                            2.181***
##                                   (0.587)
## energy_loss                      2.656***
##                                   (0.674)
## maintenance                       0.594
##                                   (0.437)
## warranty                         1.141***
##                                   (0.331)
## spare_parts                       -0.133
```

```
##                                  (0.218)
## ease_install                    0.520***
##                                  (0.173)
## problem_solving                 2.032***
##                                  (0.550)
## quality                         2.639***
##                                  (0.688)
## DA                               -0.124
##                                  (0.679)
## DB                               -0.671
##                                  (0.719)
## DC                               -0.687
##                                  (0.715)
## ------------------------------------------------
## Observations                       352
## R2                                0.411
## Max. Possible R2                  0.500
## Log Likelihood                   -28.774
## Wald Test                  23.670** (df = 11)
## LR Test                   186.439*** (df = 11)
## Score (Logrank) Test      103.266*** (df = 11)
## ================================================
## Note:                  *p<0.1; **p<0.05; ***p<0.01
```

Ahora calculamos la predicción de la utilidad de cada elección según el modelo estimado

```
u <- predict(abb.clogit)
head(u)
```

```
##          1          2          3          4          5          6
##  2.0458929  3.7277069  0.2033874 -5.9769871 -3.7209496 -5.0990503
```

Después obtenemos exp(u) y lo asignamos al objeto *eu*, y sumamos exp(u) para cada individuo

```
eu <- exp(u)
sumaeu <- by(eu, abb$id, sum)
head(sumaeu)
```

```
## abb$id
##          1          2          3          4          5          6
##   50.54779  516.23324  248.18063  164.16144 2069.30050  153.61078
```

Ahora calculamos la probabilidad de elección de cada marca. Para ello definimos una función que llamaremos prob()

```
prob<-function(suma, eutil, indiv){
#suma, eutil, inviv son los argumentos de la función
n<-0
#Crea un vector con tantos elementos como el producto entre
#lis individuos y las marcas
```

```
p<-1:indiv*4
#Para cada individuo
for (i in 1:indiv) {
#para cada marca
for (j in 1:4) {
#construye un índice
n<-n+1
#calcula la probabilidad de que el individuo i compre la #marca j
p[n]<-eutil[n]/suma[i]
}
}
#Devuelve el vector de probabilidades
return(p)
}
```

Y después la utilizamos con los datos calculados previamente

```
pchoice <- prob(sumaeu, eu, 88)
head(pchoice)
```

```
## [1] 1.530445e-01 8.226600e-01 2.424532e-02 5.017938e-05 4.689928e-05
## [6] 1.182128e-05
```

```
abb$pchoice <- pchoice
t(head(abb))
```

```
##                 1             2             3
## id              "1"           "1"           "1"
## Alternatives    "ABB"         "GE"          "Westinghouse"
## choice          "1"           "2"           "1"
## price           "6"           "6"           "6"
## energy_loss     "6"           "6"           "5"
## maintenance     "7"           "6"           "7"
## warranty        "6"           "7"           "5"
## spare_parts     "6"           "9"           "3"
## ease_install    "5"           "9"           "4"
## problem_solving "7"           "7"           "7"
## quality         "5"           "5"           "6"
## DA              "1"           "0"           "0"
## DB              "0"           "1"           "0"
## DC              "0"           "0"           "1"
## DD              "0"           "0"           "0"
## volume          "761"         "761"         "761"
## district        "1"           "1"           "1"
## pchoice         "1.530445e-01" "8.226600e-01" "2.424532e-02"
##                 4             5             6
## id              "1"           "2"           "2"
## Alternatives    "Edison"      "ABB"         "GE"
## choice          "1"           "1"           "1"
## price           "5"           "3"           "3"
## energy_loss     "5"           "4"           "4"
```

```
## maintenance     "6"             "5"             "5"
## warranty        "7"             "4"             "4"
## spare_parts     "8"             "4"             "7"
## ease_install    "2"             "5"             "3"
## problem_solving "6"             "6"             "5"
## quality         "5"             "4"             "5"
## DA              "0"             "1"             "0"
## DB              "0"             "0"             "1"
## DC              "0"             "0"             "0"
## DD              "1"             "0"             "0"
## volume          "761"           "627"           "627"
## district        "1"             "1"             "1"
## pchoice         "5.017938e-05" "4.689928e-05" "1.182128e-05"
```

Ahora creamos una función para clasificar a los clientes en función de su probabilidad de compra

```
msegment<-function(p, indiv){
# p es el vector de probabilidades
# in es el número de individuos
s<-1:indiv*4
j<-0
for (i in 1:indiv) {
#para cada individuo
j=j+4
#Leales
if (p[j-3]>0.8) {s[j-3]<-"L"; s[j-2]<-"L"; s[j-1]<-"L"; s[j]<-"L"}
#Competitivos
if (p[j-3]<=0.8 & p[j-3]>0.5) {s[j-3]<-"C"; s[j-2]<-"C"; s[j-1]<-"C";
s[j]<-"C"}
#Apropiables
if (p[j-3]<=0.5 & p[j-3]>0.15) {s[j-3]<-"A"; s[j-2]<-"A"; s[j-1]<-"A";
s[j]<-"A"}
#Perdidos
if (p[j-3]<=0.15) {s[j-3]<-"P"; s[j-2]<-"P"; s[j-1]<-"P"; s[j]<-"P"}
}
#Devuelve el resultado de la función
return(s)
}
```

Ahora utilizamos la nueva función para clasificar la base de datos

```
seg <- msegment(pchoice, 88)
abb$seg <- seg

abb.select.ord <- select(abb, volume, pchoice, seg) %>%
  arrange(-volume)
options(digits=7)
head(abb.select.ord)
```

```
##   volume      pchoice seg
## 1  14798 4.989266e-04   P
## 2  14798 6.259664e-08   P
## 3  14798 8.011871e-07   P
## 4  14798 9.995002e-01   P
## 5  12514 7.866831e-03   P
## 6  12514 3.195145e-04   P

###con las funciones básicas
#o<-order(abb$VOLUME, decreasing=TRUE)
#abbor<-cbind(abb$VOLUME[o], abb$pchoice[o], abb$seg[o])
#abbor
```

primero seleccionamos las variables que queremos ordenar, después