Boston

Laura del Pino Díaz

24/11/2016

Boston

Boston es un conjunto de datos en el que están almacenados los distintos parámetros de las casas de la ciudad de Boston. El objetivo es intentar predecir el valor de las casas sabiendo el valor de un conjunto de ellas.

Los parámetros

- crim - proporción de crimen per cápita en el barrio.
- zn - proporción de zona residencial por cada 25 000 pies cuadrados
- indus - proporción de acres de negocios industriales.
- chas - indica si la casa da al río Charles o no.
- nox - concentración de óxido de nitrógenso en partes por 10 millones.
- rm - número de habitaciones
- age - proporción de viviendas anteriores a 1940.
- dis - media ponderada de distancias a los cinco centros de empleo de Boston.
- rad - índice de accesibilidad a los pasos altos.
- tax - impuestos
- ptratio - proporción de profesores.
- black - 100(Bk -0.63)^2 donde Bk es la proporción de black en el pueblo
- lstat - porcentaje de la población de estatus bajo
- medv - valor de la casa en 1000 dólares

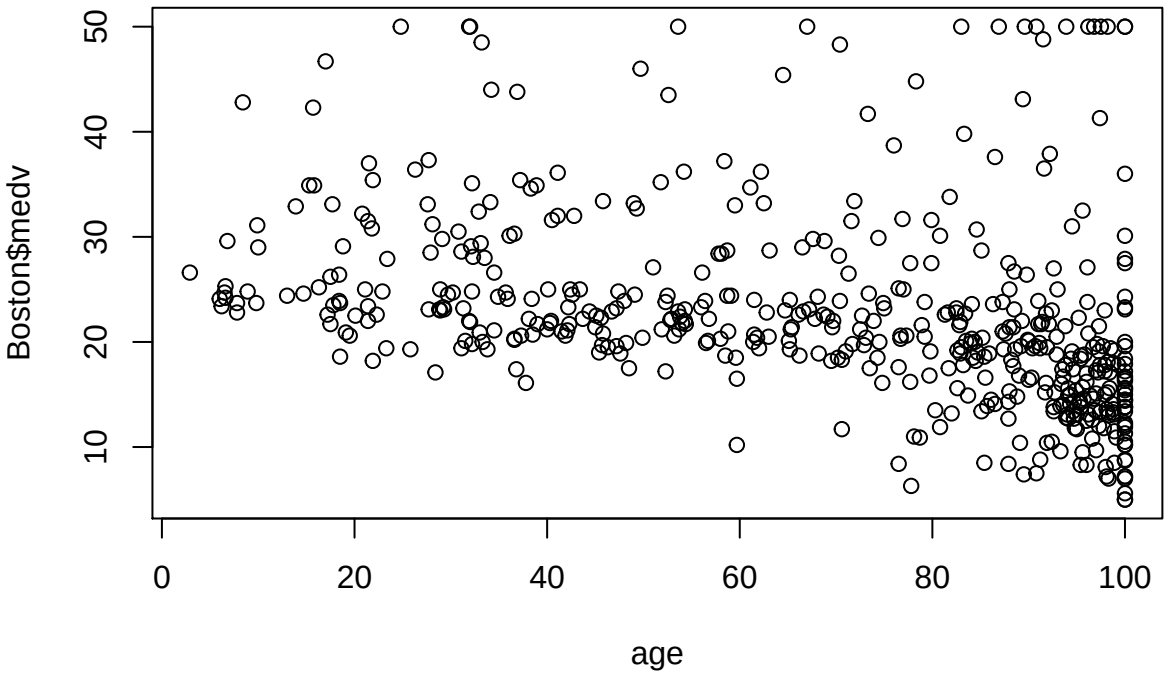
Se aconseja ejecutar el comando attach para acceder directamente a los campos.

```
attach(Boston)
head(lstat)
```

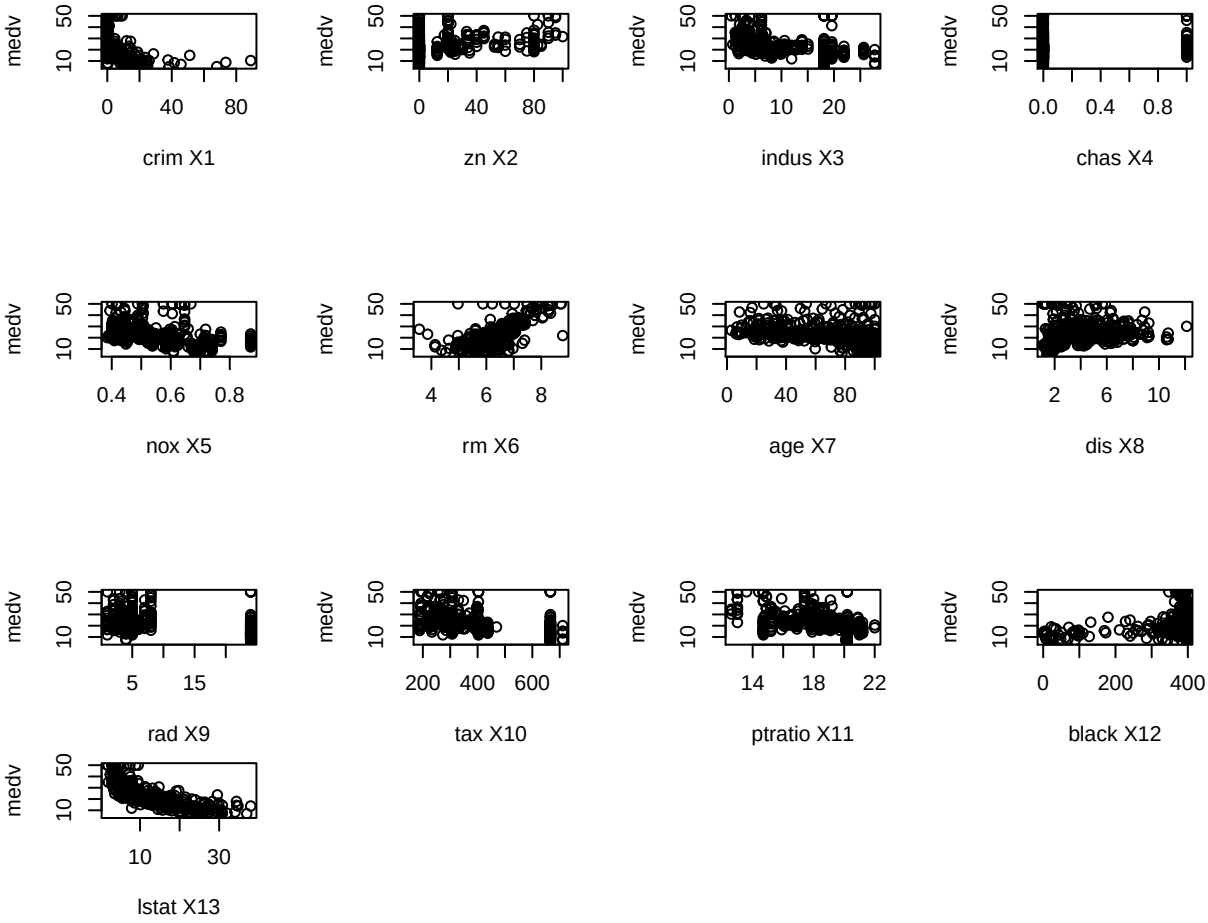
```
## [1] 4.98 9.14 4.03 2.94 5.33 5.21
```

Visualización de datos

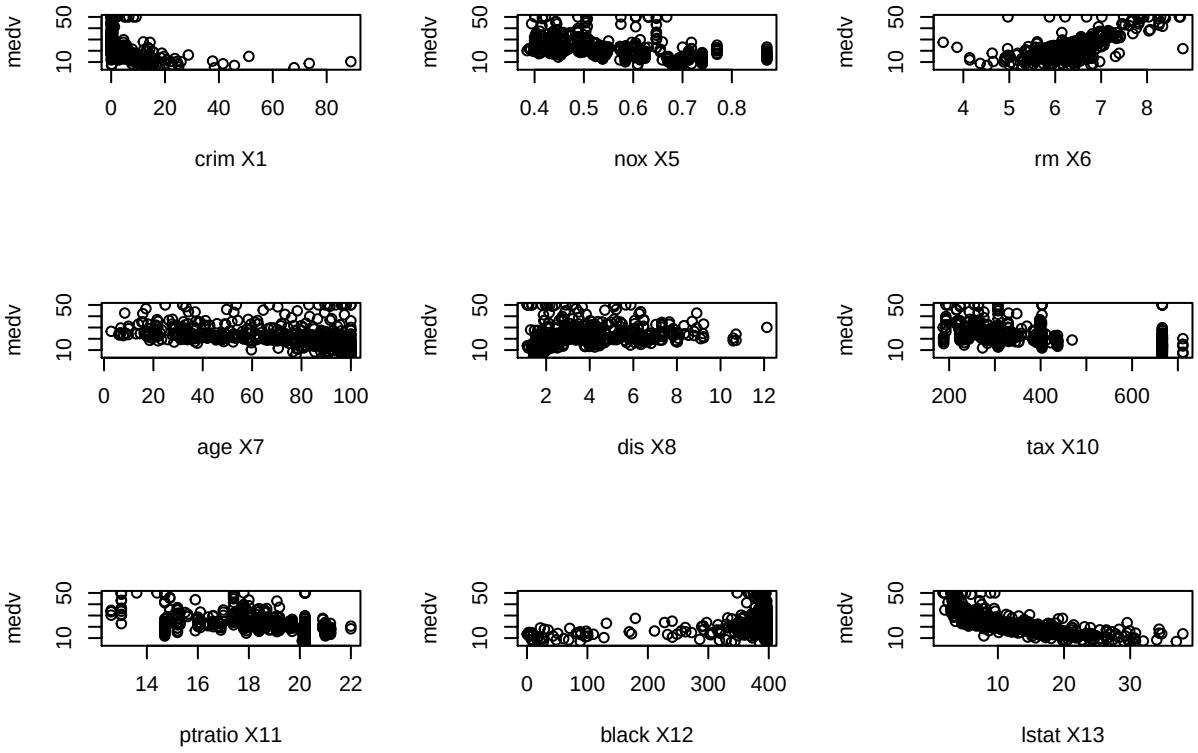
Para comprobar que datos intervienen en el valor final es necesario visualizar los datos con respecto de la salida. Esto lo podemos conseguir mirando cada una de las variables con respecto de la salida



O directamente mirando todas las variables entre sí o con respecto a la salida.



Y una vez las comparamos todas elegimos las que nos parecen más relevantes.



En este caso nos parece que tienen un ajustes lineal las variables rm y lstat.

Obtención de un modelo lineal

Para obtener el modelo lineal sobre una variable utilizamos la función lm, que tiene como parámetros la

```
fit1 = lm(medv ~lstat, data = Boston)
fit2 = lm(medv~rm,data=Boston)
```

Ahora que tenemos el modelo lineal, vamos a comprobar la bondad del mismo para ello ejecutamos un summa

```
summary(fit1)
```

```
##
## Call:
## lm(formula = medv ~ lstat, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.168  -3.990  -1.318   2.034  24.500
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.55384    0.56263   61.41   <2e-16 ***
## lstat       -0.95005    0.03873  -24.53   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.216 on 504 degrees of freedom
## Multiple R-squared:  0.5441, Adjusted R-squared:  0.5432
## F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
summary(fit2)
```

```
##
## Call:
## lm(formula = medv ~ rm, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.346  -2.547   0.090   2.986  39.433
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -34.671      2.650  -13.08   <2e-16 ***
## rm             9.102      0.419   21.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.616 on 504 degrees of freedom
## Multiple R-squared:  0.4835, Adjusted R-squared:  0.4825
## F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16
```

Para saber cuan bueno es el modelo lineal nos fijaremos en el parámetro de salida de la llamada al summa
Este parámetro cuanto más cercano a 1 mejor, por lo tanto en estos modelos que hemos generado donde R-s

Accediendo a la información del modelo

Para saber que información tiene nuestro modelo podemos consultar los campor realizando una llamada a na

```
names(fit1)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```