# 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

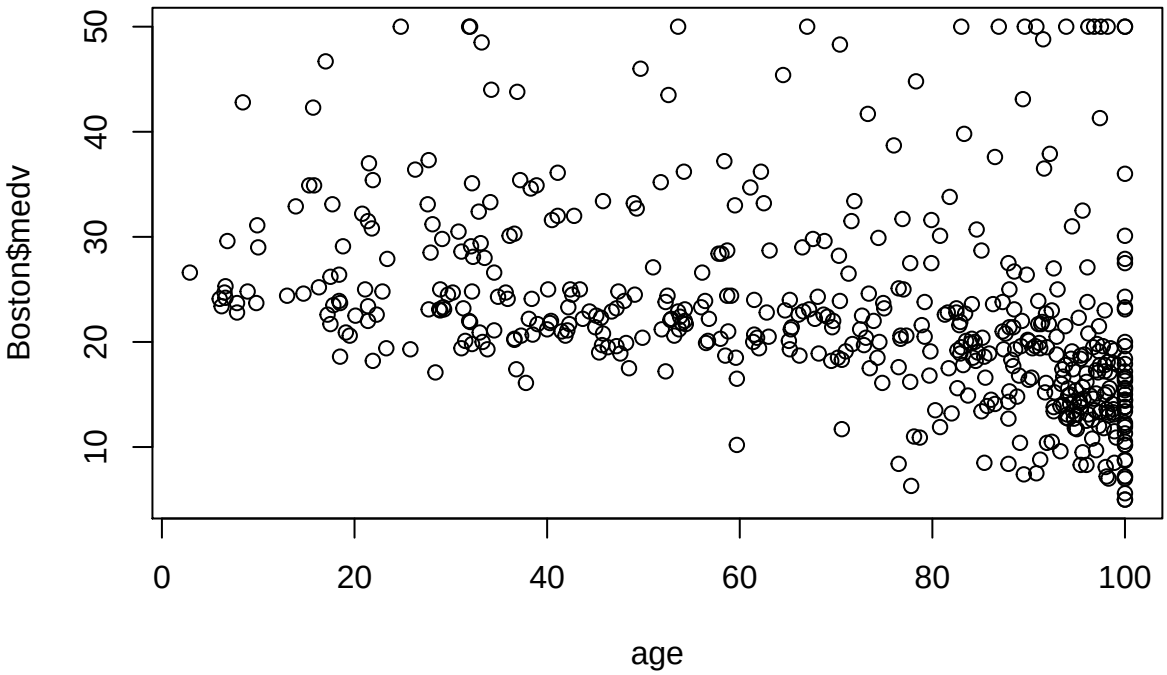
Se aconseja ejecutar el comando attach para acceder directamente a los campos.

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

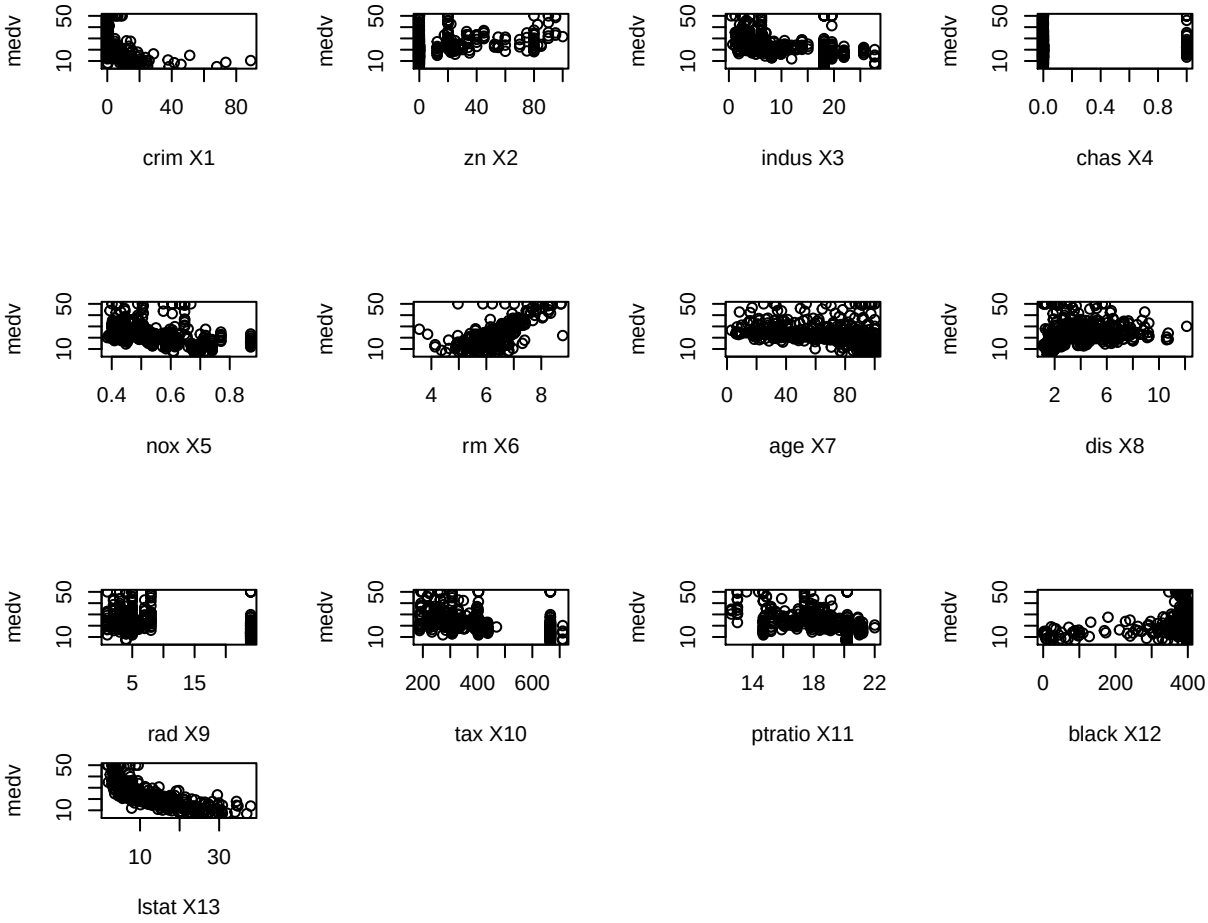
```
## [1] 4.98 9.14 4.03 2.94 5.33 5.21
```

### Visualización de datos

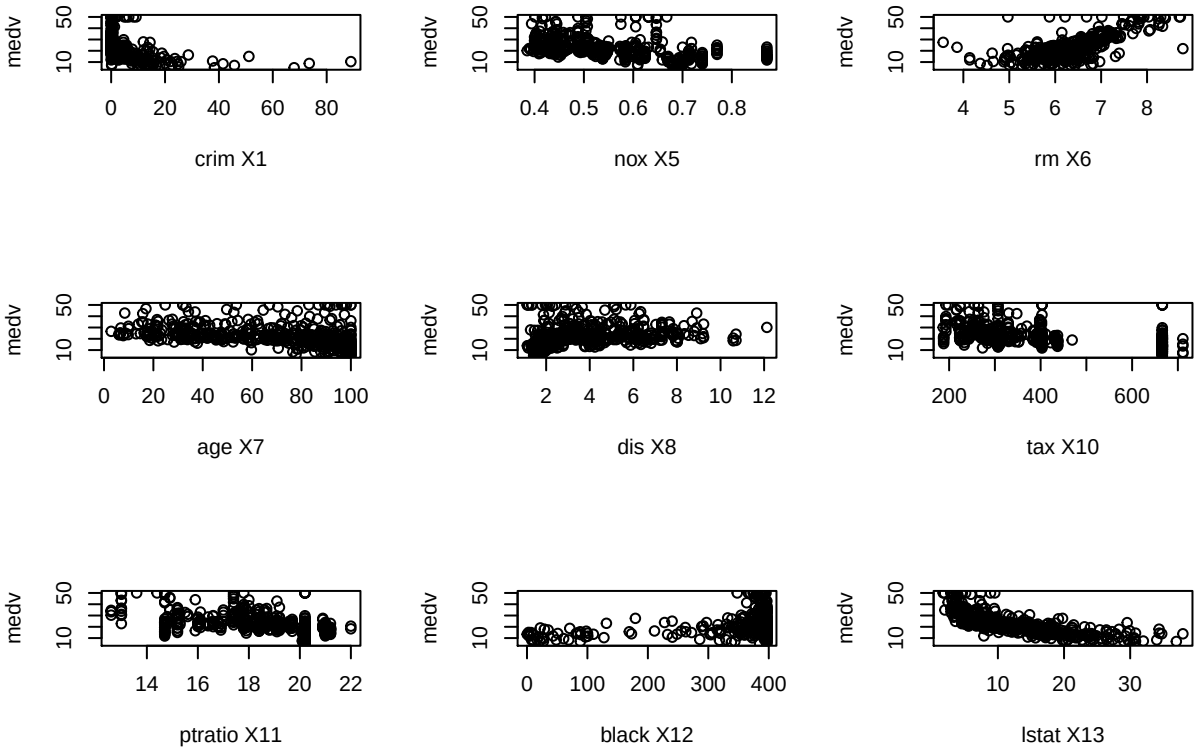
Para comprobar que datos intervienen en el valor final es necesario visualizar los datos con respecto de la salida. Esto lo podemos conseguir mirando cada una de las variables con respecto de la salida




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"
```

Para calcular la raíz del error cuadrático mínimo utilizamos el siguiente  comando

```
```r
sqrt(sum(fit1$residuals^2)/length(fit1$residuals))
```
```

```
## [1] 6.203464
```

Este error cuadrático mínimo no coincide con el que nos devolvía y eso es porque se utiliza el n-p para

```r
sqrt(sum(fit1$residuals^2)/(length(fit1$residuals)-2))
```

```
## [1] 6.21576
```

En este caso al ajustar el denominador con -2  si se obtiene el coeficiente que nos devolvió el summary

### Predicción sobre nuevos datos.

El fin último de la regresión es la estimación del valor del parámetro de salida a partir del modelo. Para predecir con el modelo que hemos creado anteriormente utilizaremos el siguiente comando:

```
  predict(fit1, data.frame(lstat=c(5,10,15)))
```

```
##        1        2        3
## 29.80359 25.05335 20.30310
```

El cálculo de la raíz del error cuadrático mínimo para el conjunto de test lo hacemos ejecutando la siguiente sentencia:

```
   yprime=predict(fit1,data.frame(lstat=Boston$lstat))
   sqrt(sum(abs(Boston$medv-yprime)^2)/length(yprime))
```

```
## [1] 6.203464
```

## Modelo de regresión lineal múltiple

Dado que el modelo lineal con una sola variable no es lo suficientemente bueno podemos añadirles variabl

```
  fit3=lm(medv~lstat+age,data=Boston)
  summary(fit3)
```

```
##
## Call:
## lm(formula = medv ~ lstat + age, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.981  -3.978  -1.283   1.968  23.158
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 33.22276    0.73085  45.458  < 2e-16 ***
## lstat       -1.03207    0.04819 -21.416  < 2e-16 ***
## age          0.03454    0.01223   2.826  0.00491 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.173 on 503 degrees of freedom
## Multiple R-squared:  0.5513, Adjusted R-squared:  0.5495
## F-statistic:   309 on 2 and 503 DF,  p-value: < 2.2e-16
```

En este primer modelo lineal compuesto vemos tres variables, la combinación de lstat y age, así como cada una de las variables por separado. Ante el test de hipótesis de vemos que la variable age está marcada por dos **, esto no es buena señal puesto que indica que aporta menos que las demás a la salida en el modelo.

Sabiendo esto cambiamos la variable age por rm y comprobamos que es lo que obtenemos

```
fit4=lm(medv~lstat+rm,data=Boston)
summary(fit4)
```

```
##
## Call:
## lm(formula = medv ~ lstat + rm, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.076  -3.516  -1.010   1.909  28.131
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.35827    3.17283  -0.428    0.669
## lstat       -0.64236    0.04373 -14.689   <2e-16 ***
## rm           5.09479    0.44447  11.463   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.54 on 503 degrees of freedom
## Multiple R-squared:  0.6386, Adjusted R-squared:  0.6371
## F-statistic: 444.3 on 2 and 503 DF,  p-value: < 2.2e-16
```

Ahora la variable de nuestro modelo que no interviene es la combinación de ambas, por lo que seguiremos modificando el modelo pero esta vez no lo haremos al tuntún sino que probaremos con el conjunto de todas las variables.

```
fit5=lm(medv~.,data=Boston)
summary(fit5)
```

```
##
## Call:
## lm(formula = medv ~ ., data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas         2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

Todas aquellas variables con un alto Pr son candidatas a ser eliminadas. Por ello vamos a eliminar de nuestro modelo a las variables age e indus

```
fit6=lm(medv~.-age-indus,data=Boston)
summary(fit6)
```

```
##
## Call:
## lm(formula = medv ~ . - age - indus, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5984  -2.7386  -0.5046   1.7273  26.2373
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.341145   5.067492   7.171 2.73e-12 ***
## crim         -0.108413   0.032779  -3.307 0.001010 **
## zn            0.045845   0.013523   3.390 0.000754 ***
## chas          2.718716   0.854240   3.183 0.001551 **
## nox         -17.376023   3.535243  -4.915 1.21e-06 ***
## rm            3.801579   0.406316   9.356  < 2e-16 ***
## dis          -1.492711   0.185731  -8.037 6.84e-15 ***
## rad           0.299608   0.063402   4.726 3.00e-06 ***
## tax          -0.011778   0.003372  -3.493 0.000521 ***
## ptratio      -0.946525   0.129066  -7.334 9.24e-13 ***
## black         0.009291   0.002674   3.475 0.000557 ***
## lstat        -0.522553   0.047424 -11.019  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.736 on 494 degrees of freedom
```

```
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7348
## F-statistic: 128.2 on 11 and 494 DF,  p-value: < 2.2e-16
```

Comparando las dos R cuadradas parece que hemos mejorado levemente el modelo. Probemos a eliminar aquellas variables marcadas únuicamente con **

```
fit7=lm(medv~.-age-indus-chas-crim,data=Boston)
summary(fit7)
```

```
##
## Call:
## lm(formula = medv ~ . - age - indus - chas - crim, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.8917  -2.7329  -0.4988   1.8547  26.6433
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.459724   5.158054   6.875 1.87e-11 ***
## zn            0.041396   0.013737   3.013 0.002715 **
## nox         -15.502932   3.583879  -4.326 1.84e-05 ***
## rm            3.879580   0.414180   9.367  < 2e-16 ***
## dis          -1.451648   0.187926  -7.725 6.26e-14 ***
## rad           0.252412   0.061778   4.086 5.12e-05 ***
## tax          -0.012360   0.003427  -3.606 0.000342 ***
## ptratio      -0.968703   0.131248  -7.381 6.69e-13 ***
## black         0.010842   0.002705   4.008 7.06e-05 ***
## lstat        -0.555124   0.047699 -11.638  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.832 on 496 degrees of freedom
## Multiple R-squared:  0.7289, Adjusted R-squared:  0.724
## F-statistic: 148.2 on 9 and 496 DF,  p-value: < 2.2e-16
```

En este caso vemos que empeora con respecto al caso anterior en casi un 1%

### Iteracciones y no linealidad

# Interacción

Probemos esta vez con interrelaciones entre variables.

```
fit8=lm(medv~lstat*rm,Boston)
summary(fit8)
```

```
##
## Call:
## lm(formula = medv ~ lstat * rm, data = Boston)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23.2349  -2.6897  -0.6158   1.9663  31.6141
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -29.12452    3.34250  -8.713   <2e-16 ***
## lstat         2.19398    0.20570  10.666   <2e-16 ***
## rm            9.70126    0.50023  19.393   <2e-16 ***
## lstat:rm     -0.48494    0.03459 -14.018   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.701 on 502 degrees of freedom
## Multiple R-squared:  0.7402, Adjusted R-squared:  0.7387
## F-statistic: 476.9 on 3 and 502 DF,  p-value: < 2.2e-16
```

Esta combinación nos da el mejor modelo hasta la fecha.

## No linealidad

Si asumimos que el modelo no es lineal porque la intervención de alguna variable no lo sea, lo podemos probar haciendo uso de la función l

```
fit9=lm(medv~I(lstat^2),Boston)
summary(fit9)
```

```
##
## Call:
## lm(formula = medv ~ I(lstat^2), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.242  -4.488  -2.202   2.540  24.554
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 27.647392   0.429925   64.31   <2e-16 ***
## I(lstat^2)  -0.024240   0.001359  -17.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.207 on 504 degrees of freedom
## Multiple R-squared:  0.3871, Adjusted R-squared:  0.3859
## F-statistic: 318.3 on 1 and 504 DF,  p-value: < 2.2e-16
```

Lo que hay que tener en cuenta que hay que poner un término por cada grado hasta llegar al grado que deseamos.

```
  fit9=lm(medv~lstat+I(lstat^2),Boston)
summary(fit9)
```

```
##
## Call:
## lm(formula = medv ~ lstat + I(lstat^2), data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2834  -3.8313  -0.5295   2.3095  25.4148
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.862007   0.872084   49.15   <2e-16 ***
## lstat       -2.332821   0.123803  -18.84   <2e-16 ***
## I(lstat^2)   0.043547   0.003745   11.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.524 on 503 degrees of freedom
## Multiple R-squared:  0.6407, Adjusted R-squared:  0.6393
## F-statistic: 448.5 on 2 and 503 DF,  p-value: < 2.2e-16
```

En el caso de que el grado sea muy grande podemos hacer uso de la función poly para que nos lo haga R.

```
  fit10=lm(medv~poly(lstat,18))
summary(fit10)
```

```
##
## Call:
## lm(formula = medv ~ poly(lstat, 18))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.2819  -3.0406  -0.6712   1.9576  26.6949
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         22.5328     0.2294  98.207  < 2e-16 ***
## poly(lstat, 18)1  -152.4595     5.1611 -29.540  < 2e-16 ***
## poly(lstat, 18)2    64.2272     5.1611  12.444  < 2e-16 ***
## poly(lstat, 18)3   -27.0511     5.1611  -5.241 2.38e-07 ***
## poly(lstat, 18)4    25.4517     5.1611   4.931 1.12e-06 ***
## poly(lstat, 18)5   -19.2524     5.1611  -3.730 0.000214 ***
## poly(lstat, 18)6     6.5088     5.1611   1.261 0.207875
## poly(lstat, 18)7     1.9416     5.1611   0.376 0.706930
## poly(lstat, 18)8    -6.7299     5.1611  -1.304 0.192867
## poly(lstat, 18)9     8.4168     5.1611   1.631 0.103578
## poly(lstat, 18)10   -7.3351     5.1611  -1.421 0.155892
## poly(lstat, 18)11    9.4344     5.1611   1.828 0.068167 .
## poly(lstat, 18)12    4.1255     5.1611   0.799 0.424480
## poly(lstat, 18)13    3.8898     5.1611   0.754 0.451408
## poly(lstat, 18)14   11.6301     5.1611   2.253 0.024679 *
## poly(lstat, 18)15   -8.7030     5.1611  -1.686 0.092387 .
## poly(lstat, 18)16    7.6770     5.1611   1.487 0.137538
## poly(lstat, 18)17   -2.9429     5.1611  -0.570 0.568810
```

```
## poly(lstat, 18)18   -2.9605     5.1611  -0.574 0.566489
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.161 on 487 degrees of freedom
## Multiple R-squared:  0.6963, Adjusted R-squared:  0.6851
## F-statistic: 62.03 on 18 and 487 DF,  p-value: < 2.2e-16
```

En este resultado podemos ver como a partir del grado 5, las variables de grado superior dejan de ser importantes para el modelo.