Bikes

Jose Javier Martíb García

23/10/2019

El siguiente estudio corresponde a una base de datos llamada "Bikes" la cual está compuesta por 16 variables repartidas en 731 registros. Las variables que forman dicha base de datos son las siguientes:

- Dteday: que corresponde con la fecha del día de la observación.
- Season: es una variable categórica que corresponde con la estación del año. Ha sido convertida a variable dummie.
- Yr: corresponde con el año y se ha convertido a variable dummie siendo el valor 0 para el año 2011 y el valor 1 para el año 2012.
- Mnth: corresponde al mes del año toma valores de 1 a 12 (Enero - Diciembre).
- Holiday: es una variable categórica que hace referencia a a los días de vacaciones toma valores de 0 y de 1.
- Weekday: es una variable categórica que corresponde a los días de la semana, toma valores de 0 a 6. Siendo 0 el domingo y el 6 el sábado.
- Workingday: hace referencia a los días laborales y también es una variable categórica.
- Weathersit: es una variable categórica que toma valores entre 1 y 4, es explicada de la siguiente manera: 1: Despejado, pocas nubes, parcialmente nublado, parcialmente nublado. 2: Niebla + Nublado, Niebla + Nubes rotas, Niebla + Pocas nubes, Niebla. 3: Nieve ligera, lluvia ligera + tormenta eléctrica + nubes dispersas, lluvia ligera + nubes dispersas. 4: Lluvia intensa + paletas de hielo + tormenta eléctrica + niebla, nieve + niebla.
- Temp: es la temperatura en grados Celsius.
- Atemp: es la sensación termica en grados Celsius.
- Hum: se corresponde con la humedad sus valores se han dividido entre 100.
- Windspeed: corresponde a la velocidad del viento, los valores han sido divididos entre 67 como valor máximo.
- Casual: número de usuarios que hacen uso del servicio de bicicletas esporadicamente.
- Registered: número de usuarios registrados en el sistema.
- Cnt: total de usuarios (Casual + Registered)

```
library(ISLR)
library(boot)
library(gam)
```

```
## Loading required package: splines
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.16.1
```

```r
library(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
```

```r
bikes <- read.csv("C:/Users/Equipo/Desktop/CUNEF/Prediccion/Datos/day.csv")
```

Lo primero que he hecho ha sido eliminar las columnas que no voy a usar, la razon de no usar "Casual" y "Registered" es porque "Count" es la suma de ambas. Finalmente decidí que solo iba a usar variables que personalmente he considerado relevantes aunque no he eliminado el resto, esas variables con las que me he quedado han sido: "Temp", "Windspeef" y "Hum" ya que en mi opinión son las más relevantes a la hora de usar un servicio de bicicletas.

```r
names(bikes)
```

```
##  [1] "instant"    "dteday"     "season"     "yr"         "mnth"
##  [6] "holiday"    "weekday"    "workingday" "weathersit" "temp"
## [11] "atemp"      "hum"        "windspeed"  "casual"     "registered"
## [16] "cnt"
```

```r
bikes <- select(bikes, -dteday, -instant, -casual, -registered)
```

He fijado una semilla de 88 para realizar los calculos y que el lector pueda reproducir el codigo y poder obtener los mismos resultados que yo.

```r
set.seed(88)
```

Y calculo los coeficientes de error y posteriormente hago la raiz cuadrada de los coeficientes de error ya que dicho valor aparece al cuadrado. Se ha usado al relacion de variables Count y Temperatura en el primer caso.

```r
cv.errors <- data.frame(degree = seq(1,5,1),
                        error = rep(NA, 5))

for (i in 1:5) {
  glm.fit <- glm(cnt~poly(temp, i), data = bikes)
  cv.errors$error[i] <- cv.glm(bikes, glm.fit, K = 10)$delta[1]

}

sqrt(cv.errors$error)
```

```
## [1] 1512.469 1436.844 1425.023 1425.496 1427.192
```

En este caso me quedo con el polinomio de grado tres ya que es el que menor termino de error me ha dado.

Ahora calculo los coeficientes de error para la relacion de variables Count y Humedad.

```
cv.errors1 <- data.frame(degree = seq(1,5,1),
                         error = rep(NA, 5))

for (i in 1:5) {
  glm.fit <- glm(cnt~poly(hum, i), data = bikes)
  cv.errors1$error[i] <- cv.glm(bikes, glm.fit, K = 10)$delta[1]

}

sqrt(cv.errors1$error)
```

```
## [1] 1931.739 1867.423 1869.171 1900.157 1864.498
```

En este caso me vuelvo a quedar con el polinomio de grado tres ya que es el que menor termino de error me da.

Y finalmente calculo los coeficientes de error para el par de variables Count y Velocidad del viento.

```
cv.errors2 <- data.frame(degree = seq(1,5,1),
                         error = rep(NA, 5))

for (i in 1:5) {
  glm.fit <- glm(cnt~poly(windspeed, i), data = bikes)
  cv.errors2$error[i] <- cv.glm(bikes, glm.fit, K = 10)$delta[1]

}

sqrt(cv.errors2$error)
```

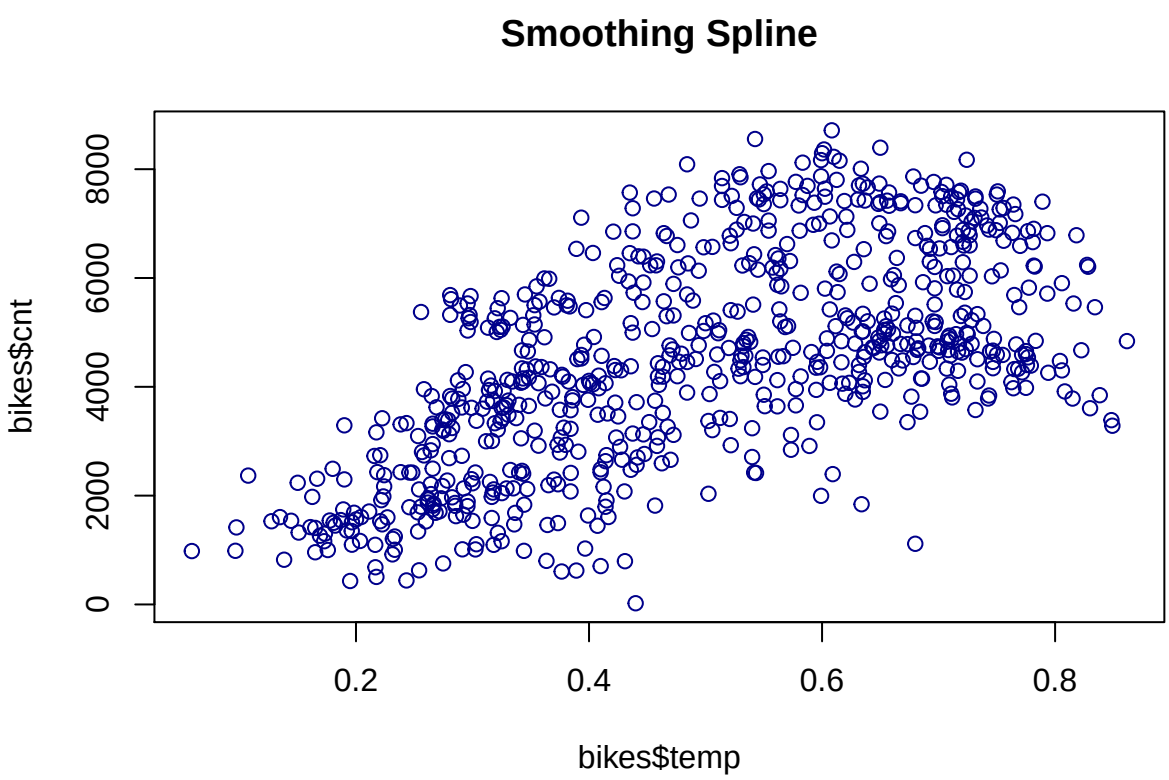
```
## [1] 1884.601 1883.396 1887.387 1885.711 1932.129
```

En este caso me quedo de nuevo con el polinomio de tercer grado porque es el que menor error me da.

Lo siguiente que he hecho ha sido calcular los grados de libertad optimos para el modelo, usando los mismos pares de variables siendo Cnt la variable dependiente y Temp, Windspeed y Hum las explicativas.

```
tempLims <- range(bikes$temp)

plot(bikes$temp, bikes$cnt, xlim = tempLims, col = "darkblue")
title("Smoothing Spline")
```




```
fit1 <- smooth.spline(bikes$temp, bikes$cnt, df = 16)
fit2 <- smooth.spline(bikes$temp, bikes$cnt, cv = TRUE)
```

```
## Warning in smooth.spline(bikes$temp, bikes$cnt, cv = TRUE): cross-
## validation with non-unique 'x' values seems doubtful
```

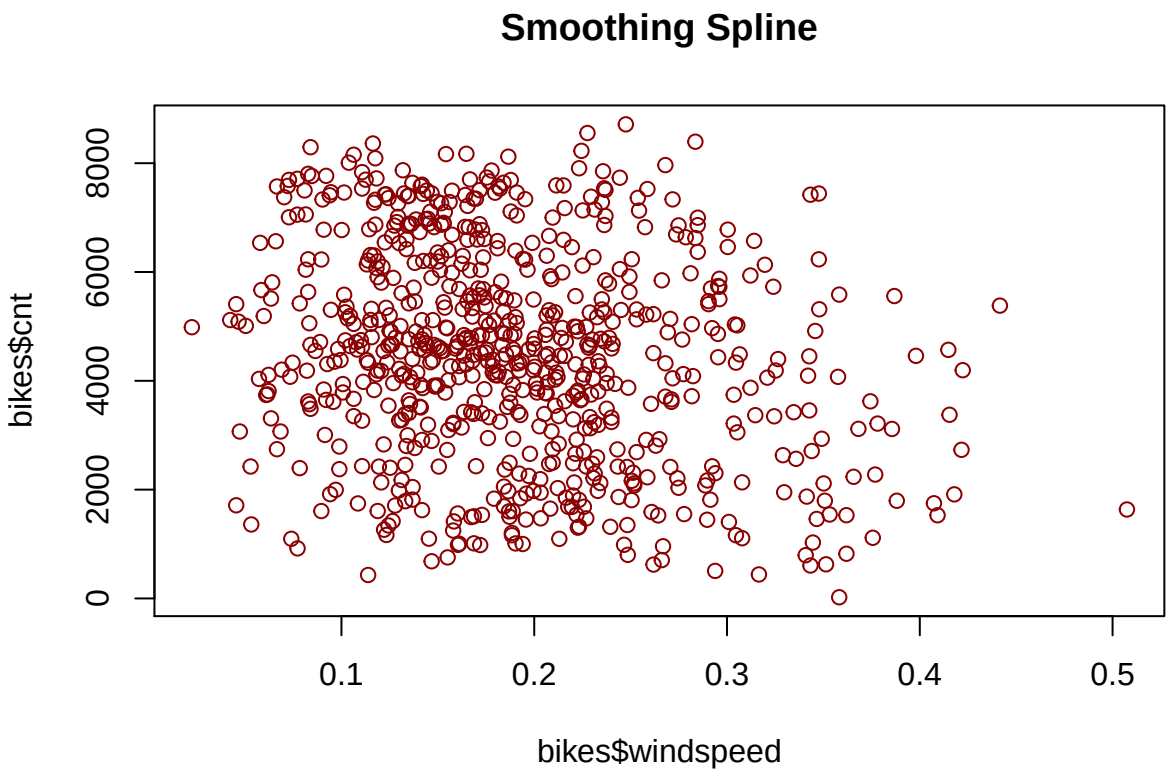
```
fit2$df
```

```
## [1] 9.103704
```

Mediante Cross Validation ha escogido 9.103704 grados de libertad optimos para la variable Temperatura como explicativa de Cnt.

```
windspeedLims <- range(bikes$windspeed) #Calculo los limites de la variable

plot(bikes$windspeed, bikes$cnt, xlim = windspeedLims, col = "darkred")
title("Smoothing Spline")
```




```
fit3 <- smooth.spline(bikes$windspeed, bikes$cnt, df = 16)
fit4 <- smooth.spline(bikes$windspeed, bikes$cnt, cv = TRUE)
```

```
## Warning in smooth.spline(bikes$windspeed, bikes$cnt, cv = TRUE): cross-
## validation with non-unique 'x' values seems doubtful
```

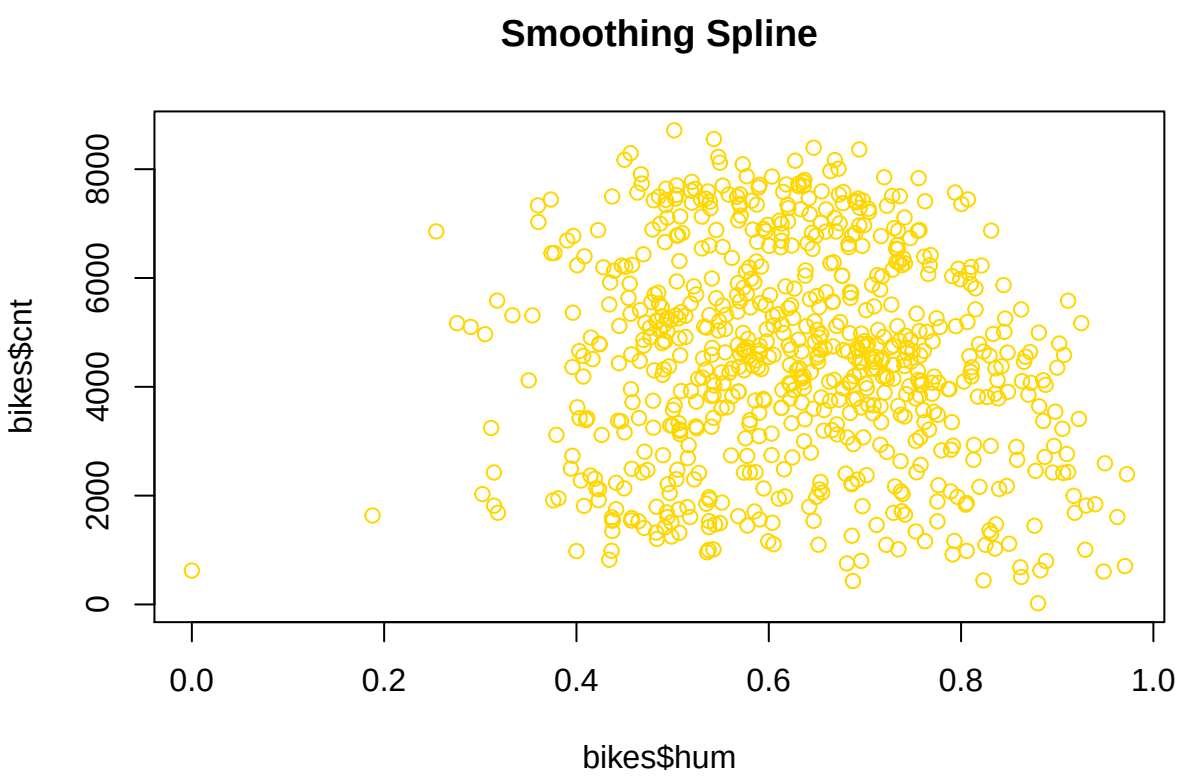
```
fit4$df
```

```
## [1] 6.007664
```

Mediante Cross Validation ha escogido 6.007664 grados de libertad optimos para la variable Windspeed como explicativa de Cnt.

```
HumLims <- range(bikes$hum) #Calculo los limites de la variable

plot(bikes$hum, bikes$cnt, xlim = HumLims, col = "gold1")
title("Smoothing Spline")
```




```
fit5 <- smooth.spline(bikes$hum, bikes$cnt, df = 16)
fit6 <- smooth.spline(bikes$hum, bikes$cnt, cv = TRUE)
```

```
## Warning in smooth.spline(bikes$hum, bikes$cnt, cv = TRUE): cross-validation
## with non-unique 'x' values seems doubtful
```

```
fit6$df
```

```
## [1] 4.548876
```

Mediante Cross Validation ha escogido 4.548876 grados de libertad optimos para la variable humedad como explicativa de cnt.

Una vez hecho lo anterior he creado un modelo aditivo generalizado con las variables explicativas que mas me convencieron y sus correspondientes grados de libertad optimos, que fueron calculados en pasos anteriores.

```
gam1 <- gam(cnt~s(temp, 9.103704) + s(windspeed, 6.007664) + s(hum, 4.548876), data = bikes)
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
gam1
```

```
## Call:
## gam(formula = cnt ~ s(temp, 9.103704) + s(windspeed, 6.007664) +
##     s(hum, 4.548876), data = bikes)
##
## Degrees of Freedom: 730 total; 710.3397 Residual
## Residual Deviance: 1061668352
```

Ahora lo que haré es el modelo spline para polinomios.

```
glm.fit <- glm(cnt ~ poly(temp, 3) + poly(windspeed, 3) + poly(hum, 3) , data = bikes)
glm.fit
```

```
##
## Call:  glm(formula = cnt ~ poly(temp, 3) + poly(windspeed, 3) + poly(hum,
##     3), data = bikes)
##
## Coefficients:
##         (Intercept)       poly(temp, 3)1       poly(temp, 3)2
##              4504.3              32091.6             -18256.1
##      poly(temp, 3)3  poly(windspeed, 3)1  poly(windspeed, 3)2
##             -6934.0             -10691.2               -263.2
## poly(windspeed, 3)3        poly(hum, 3)1        poly(hum, 3)2
##              -116.2             -16850.5              -9239.5
##       poly(hum, 3)3
##              1173.7
##
## Degrees of Freedom: 730 Total (i.e. Null);  721 Residual
## Null Deviance:       2.74e+09
## Residual Deviance: 1.075e+09     AIC: 12480
```

Y finalmente he hecho Cross Validation para validad el modelo al cual lo he dividido en 10 capas.

```
cross_val <- cv.glm(bikes, gam1, K = 10)
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
sqrt(cross_val$delta)
```

```
## [1] 1237.448 1235.999
```

Con la formula anterior he podido comprobar los valores de delta que corresponden con el error estimado de la prediccion (1240.548) y el valor ajustado del estimador por cross validation (1238.970).