Practica_3

Cedric Prieels

29/11/2016

```
rm(list=ls())
library(car)
library(boot)
```

```
##
## Attaching package: 'boot'

## The following object is masked from 'package:car':
##
##     logit
```

```
library(ISLR)
```

Primera serie de datos

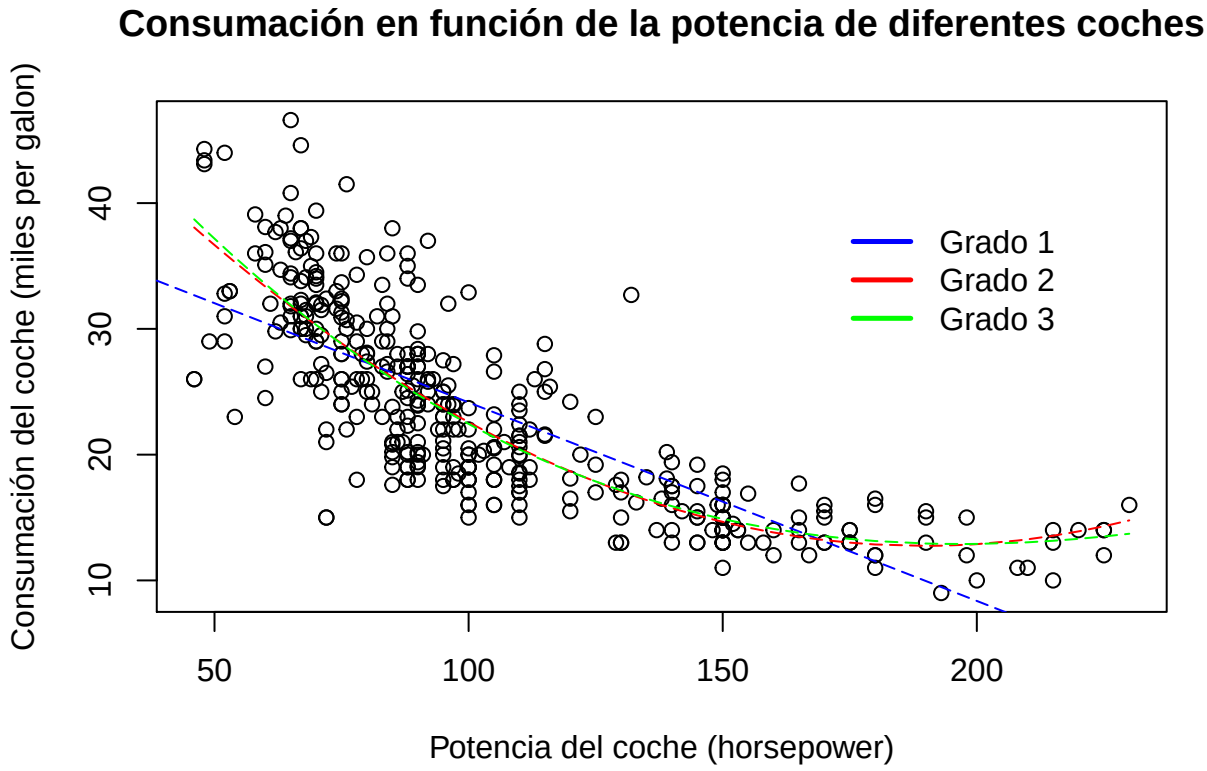
Primero, abrimos el fichero de datos llamado *Auto.rda* que hemos recibido. Podemos ver que tenemos 397 datos, repartidos en columnas que representan variables diferentes (como la consumación del coche y su potencia, las dos variables que vamos a estudiar en esta primera parte).

```
load('Auto.rda')
attach(Auto)
```

Empezamos ahora el analisis de la relación (que suponemos de tipo polinomial) que existe entre la consumación del coche y su potencia, usando diferentes métodos (holdout, and k-fold con diferentes valores de k). Pero primero, pintamos nuestros datos y unas lineas que corresponden a los polinomios de regressión de diferentes grados.

```
par(mfrow=c(1,1))
plot(horsepower, mpg,
     main="Consumación en función de la potencia de diferentes coches",
     xlab="Potencia del coche (horsepower)", ylab="Consumación del coche (miles per galon)")

#Grado 1
Reg.train <- lm(mpg~horsepower, data=Auto)
abline(a=Reg.train$coefficients[1], b=Reg.train$coefficients[2],
       col="blue", lty = 5)
#Grado 2
Reg.train <- lm(mpg~poly(horsepower, 2), data=Auto)
lines(sort(horsepower), predict(Reg.train, data.frame(horsepower=sort(horsepower))),
      col="red", lty = 5)
#Grado 3
Reg.train <- lm(mpg~poly(horsepower, 3), data=Auto)
lines(sort(horsepower), predict(Reg.train, data.frame(horsepower=sort(horsepower))),
      col="green", lty = 5)
```

```
legend(170, 40 , legend = c("Grado 1", "Grado 2", "Grado 3"),
       lty=c(1,1, 1), lwd=c(2.5,2.5, 2.5), col=c("blue","red", "green"), bty = "n")
```



Método holdout

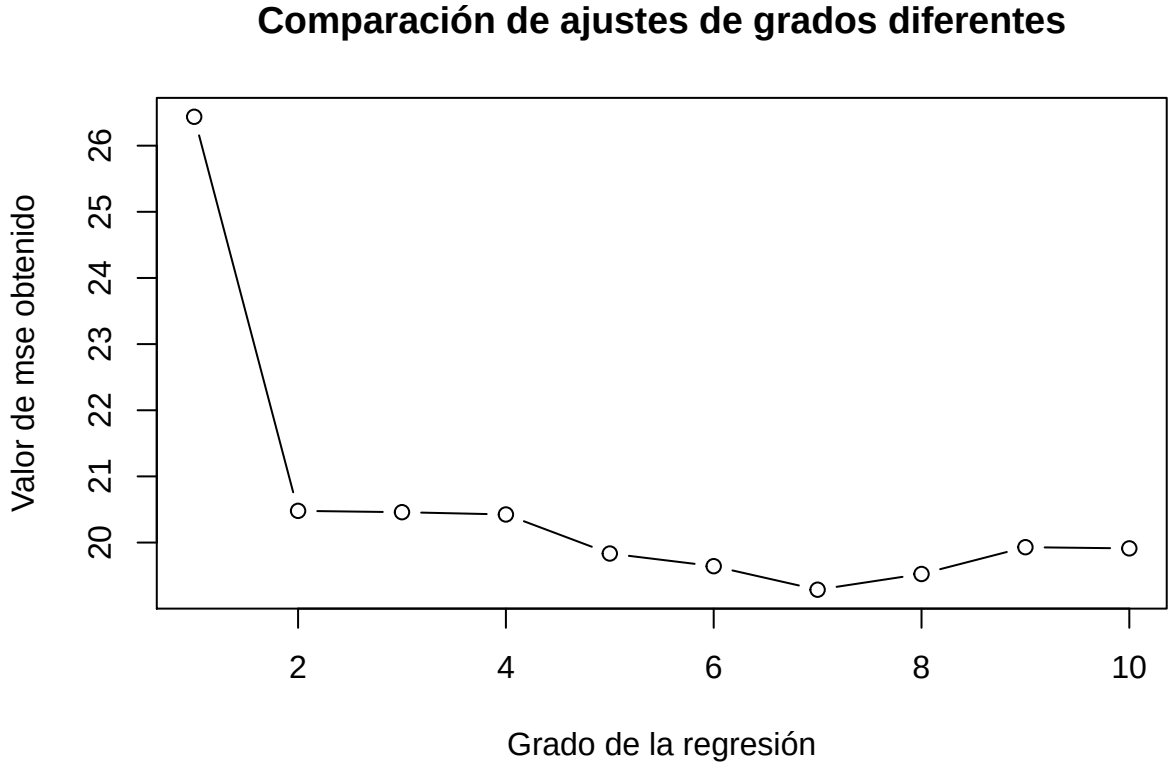
Ahora intentamos modelizar nuestros datos usando primero el método holdout, para determinar cuál es el polinomio que aproxima lo mejor nuestros datos (vamos a calcular y comparar los valores de *mse* para cada polinomio de grado diferente). Definimos primero por lo tanto una función que nos devuelve el valor de mse, dadas unas observaciones y unas estimaciones.

```
#Definimos la función mse, que nos devuelve
#la media de la diferencia entre los valores del modelo y los valores medidos al cuadrado
mse <-function(obs,est){
  return(mean((obs-est)^2))
}
```

El método holdout consiste en guardar la mitad de los datos como muestra de train, y la otra mitad como muestra de test para verificar nuestro modelo (por supuesto, hay que elegir la mitad de los datos de manera aleatoria, para evitar todo sesgo que podría aparecer si elegimos la primera mitad de los datos como train, por ejemplo).

```
n <- nrow(Auto)

#Eligimos la mitad de los datos como train de manera aleatoria
train <- sample(n,ceiling(n/2))
```

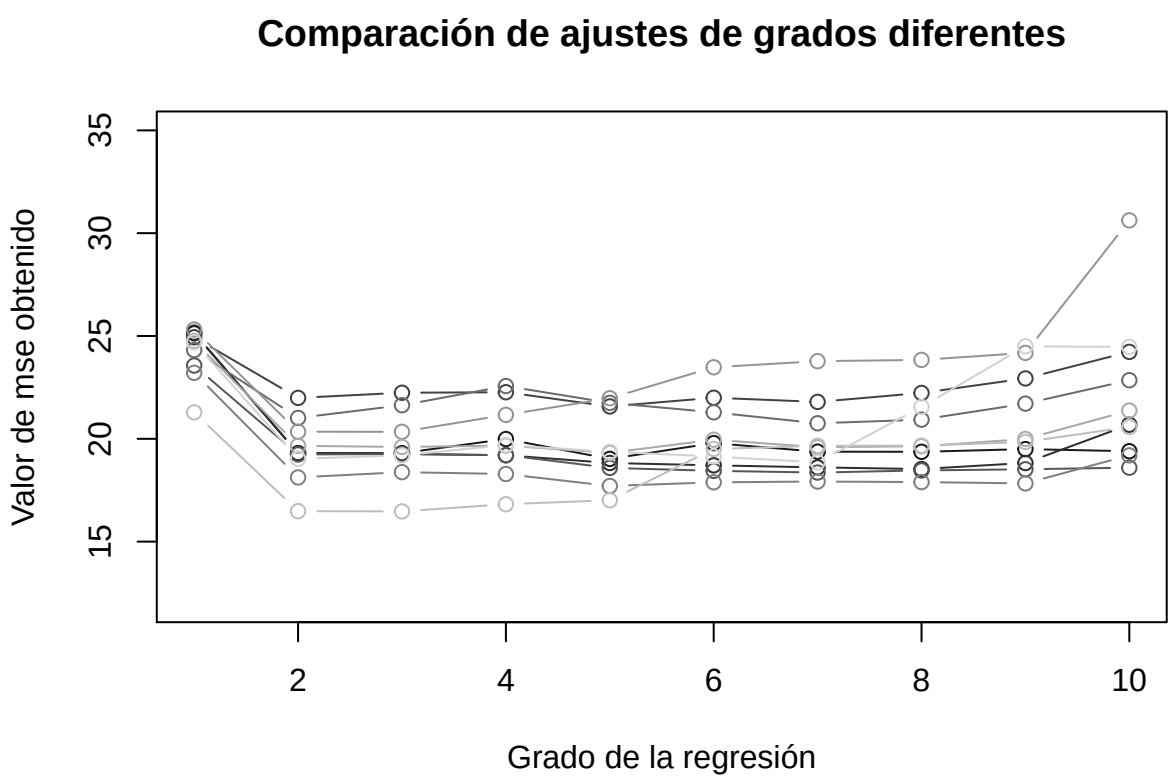
```r
mse.train <- rep(NA, 10)
mse.test <- rep(NA, 10)

#Hacemos un bucle sobre i de 1 a 10, i siendo el grado del polinomio de la regresión
for(i in 1:10) {
  Reg.train <- lm(mpg~poly(horsepower, i), data=Auto, subset=train)

  mse.train[i] <- mse(mpg[train], fitted(Reg.train))
  mse.test[i] <- mse(mpg[-train], predict(Reg.train, data.frame(horsepower=horsepower[-train])))
}

plot(seq(1:10), mse.test, type = "b", xlab="Grado de la regresión", ylab="Valor de mse obtenido", main =
```



Vemos con este último plot el valor de mse obtenido para una regresión polinomial de diferentes grados. El valor más bajo del error mse se obtiene para el grado 2, mientras que el valor más alto se obtiene para el grado 1. Ahora, queremos repetir este mismo cálculo muchas veces (en este caso, 10 veces) para ver si el valor de mse obtenido cambia si usamos muestras de train diferentes.

```r
#Creamos un plot vacio
plot(1, type = "n",
     xlab="Grado de la regresión", ylab="Valor de mse obtenido",
     main = "Comparación de ajustes de grados diferentes", xlim=c(1,10), ylim=c(12,35))

#Esta variable se usa para pintar cada linea del plot
#de un color gris un poco distinto para distinguirlas
counter <- 10

#Repetimos el cálculo 10 veces
for(j in 1:10) {
```

```r
  #Cada vez se usa una muestra de train diferente
  train <- sample(n,ceiling(n/2))
  mse.train <- rep(NA, 10)
  mse.test <- rep(NA, 10)

  #Consideramos polinomios hasta un grado 10
  for(i in 1:10) {
    Reg.train <- lm(mpg~poly(horsepower, i), data=Auto, subset=train)

    mse.train[i] <- mse(mpg[train], fitted(Reg.train))
    mse.test[i] <- mse(mpg[-train], predict(Reg.train, data.frame(horsepower=horsepower[-train])))
  }

  #Pintamos cada linea una vez el valor de mse calculado
  myColor = paste("grey", counter, sep="")
  lines(seq(1:10), mse.test, type = "b", col=myColor)
  counter <- counter + 8

}
```



En este último plot, se ha repetido el cálculo del mse para diferentes grados de regresión 10 veces, usando cada vez muestras de train diferentes. Se ve claramente que el valor del error obtenido para cada grado cambia bastante en función de los datos de train usados, lo que está mal. Vamos ahora a intentar usar otros métodos para corregir este problema.

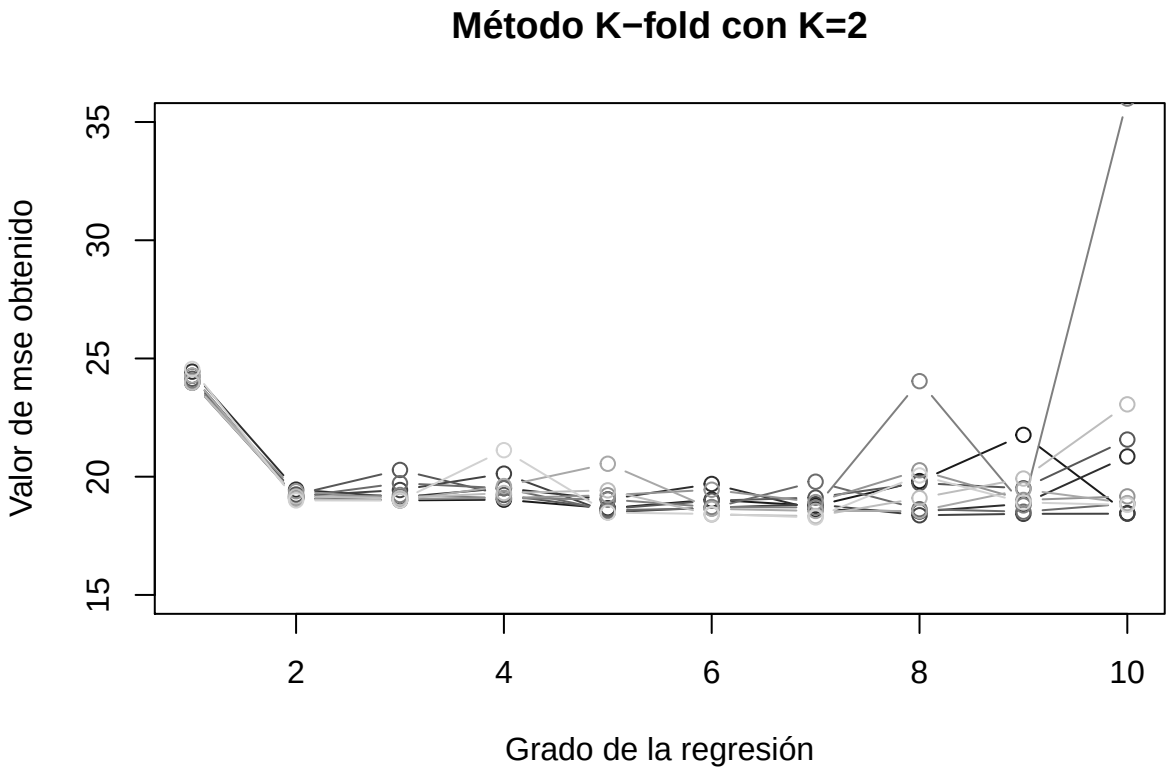
Podríamos estudiar el resultado obtenido usando el método Leave-one-out (usar todos los datos menos uno como train, para mejorar el proceso de regresión) pero este método cuesta mucho computacionalmente y la ejecución del programa es entonces mucho más lenta, como tenemos un número de datos bastante grande. Por lo tanto, solamente vamos a comparar los resultados precedientes con los que salen del uso del método

K-fold con K=2 y con K=10 (tiene que ser más preciso, pero cuesta más computacionalmente que el método holdout también).

Método K-fold

Con k = 2

```
resultsWithoutCorrection <- rep(NA, 10)
resultsWithCorrection <- rep(NA, 10)

plot(1, type = "n",
     xlab="Grado de la regresión", ylab="Valor de mse obtenido",
     main = "Método K-fold con K=2", xlim=c(1,10), ylim=c(15,35))

cv.err <- rep(NA, 10)
counter <- 10

#Repetimos el cálculo de la mse 10 veces
for(j in 1:10) {

  #Consideramos polinomios hasta un grado 10
  for(i in 1:10) {
    Reg.K2 <- glm(mpg~poly(horsepower, i), data=Auto)
    cv.err <- cv.glm(Auto, Reg.K2, K=2)
    resultsWithoutCorrection[i] <- cv.err$delta[1]
    resultsWithCorrection[i] <- cv.err$delta[2]
  }

  myColor = paste("grey", counter, sep="")
  lines(seq(1:10), resultsWithCorrection, type = "b", col=myColor)
  counter <- counter + 8

}
```

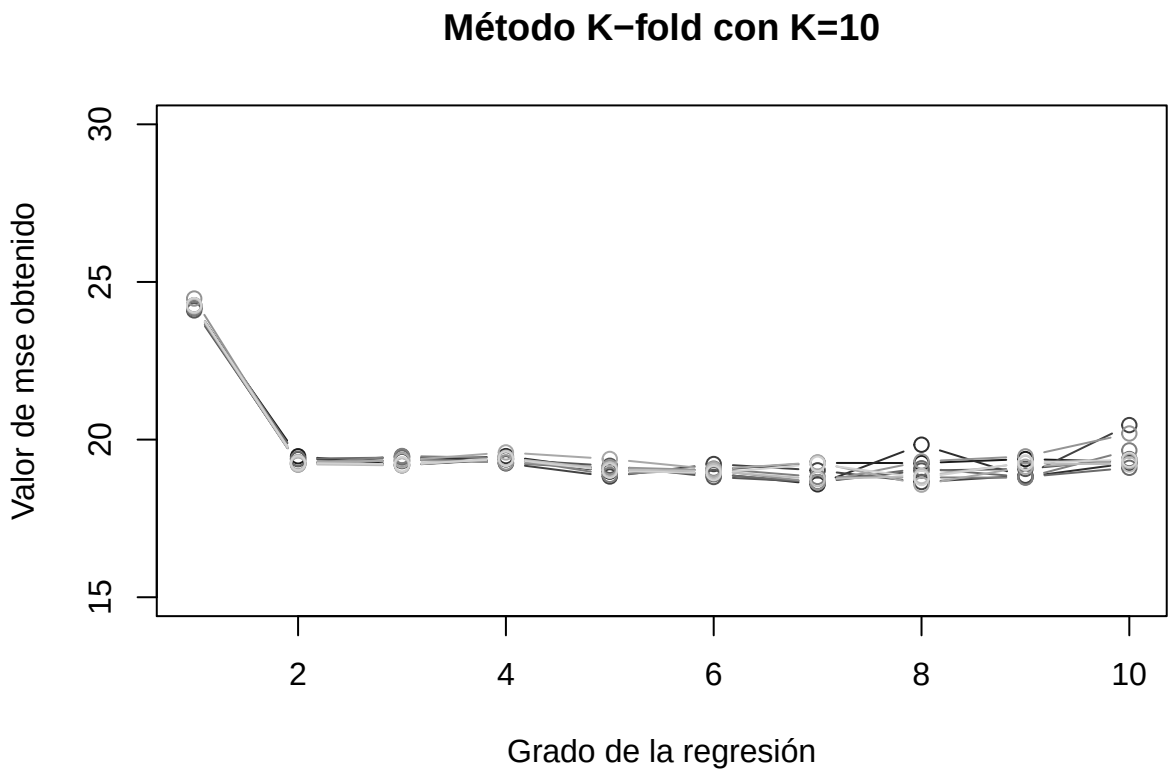


Vemos claramente en este caso que el valor de mse obtenido para cada grado es más o menos constante (excepto algunas excepciones) cuando repetimos el proceso de cálculo con muestras de entranamiento diferentes. Este modelo ya mejora los resultados obtenidos por el método holdout. Ahora volvemos a repetir lo mismo, usando K=10.

Con k = 10

```
resultsWithoutCorrection <- rep(NA, 10)
resultsWithCorrection <- rep(NA, 10)

plot(1, type = "n",
     xlab="Grado de la regresión", ylab="Valor de mse obtenido",
     main = "Método K-fold con K=10", xlim=c(1,10), ylim=c(15,30))

cv.err <- rep(NA, 10)
counter <- 10

for(j in 1:10) {

  for(i in 1:10) {
    Reg.K10 <- glm(mpg~poly(horsepower, i), data=Auto)
    cv.err <- cv.glm(Auto, Reg.K10, K=10)
    resultsWithoutCorrection[i] <- cv.err$delta[1]
    resultsWithCorrection[i] <- cv.err$delta[2]
  }

  myColor = paste("grey", counter, sep="")
  lines(seq(1:10), resultsWithCorrection, type = "b", col=myColor)
```

```
  counter <- counter + 8

}
```



Vemos que usar un valor de K igual a 10 da resultados todavía mucho mejor, pero la ejecución del programa tarda más.

En conclusión, con este método se representan cada vez los valores de mse obtenidos para diferentes grados de polinomios, para las 10 muestras de train diferentes. Vemos que el error varia bastante depiendo del train si K=2 (aunque menos que con el método precediente) pero no tanto si K=10 (y esto confirma lo que esperabamos).

Segunda serie de datos

Ahora se puede repetir lo mismo de antes, pero con otro fichero de datos. Desafortunadamente, no he econtrado en mi area de trabajo (física de partículas) unos datos acesibles con los cuales trabajar así que he decidio usar para esta parte unos datos que provienen de la base de datos de una página web que estoy desarollando. En resumen, esta página permite a los usuarios añadir a una base de datos sus vuelos (número de vuelo, hora de salida y de llegada,...) cada vez que viajan en avión. He decidido intentar comparar la relación que existe entre el tiempo de vuelo y la distancia entre dos aeropuertos, tomando como datos los 1500 primeros vuelos de mi base de datos.

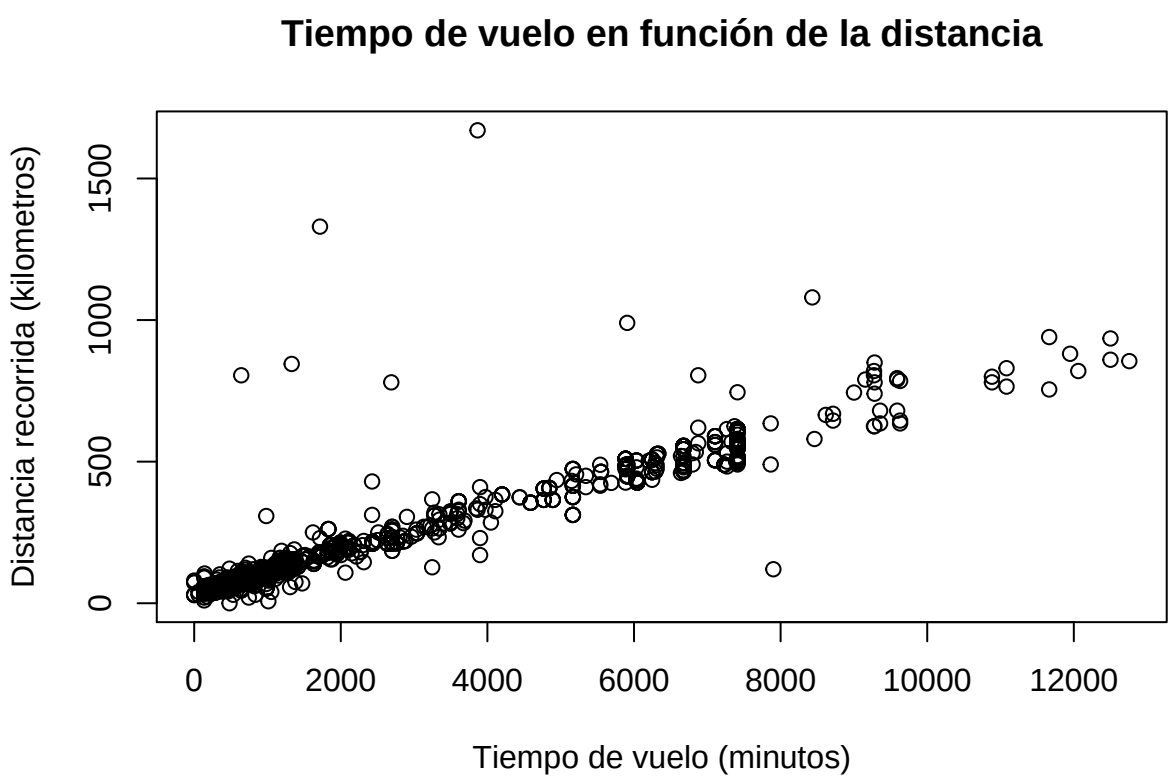
```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Modelizacion/')
Vuelos <- read.table("datos1500Vuelos.txt", header=F)
head(Vuelos)
```

```
##    V1       V2   V3
## 1 393 00:01:15  532
## 2 394 00:01:20  532
## 3 395 00:01:55 1012
## 4 396 00:01:45 1012
## 5 397 00:00:58  532
## 6 398 00:01:20  633
```

```
tiempoVuelo <- Vuelos[,2]
distanciaVuelo <- Vuelos[,3]
```

Lo primero que hay que hacer con estos datos es transformar el tiempo de vuelo del formato días:horas:minutos a un tiempo de vuelo en minutos. Se crea entonces un nuevo dataframe que contiene estos datos en el nuevo formato.

```
tiempoVueloEnMinutos <- 24*60*as.numeric(substring(tiempoVuelo, 0, 2)) + 60*as.numeric(substring(tiempo
plot(distanciaVuelo, tiempoVueloEnMinutos, main="Tiempo de vuelo en función de la distancia", xlab = "T
```

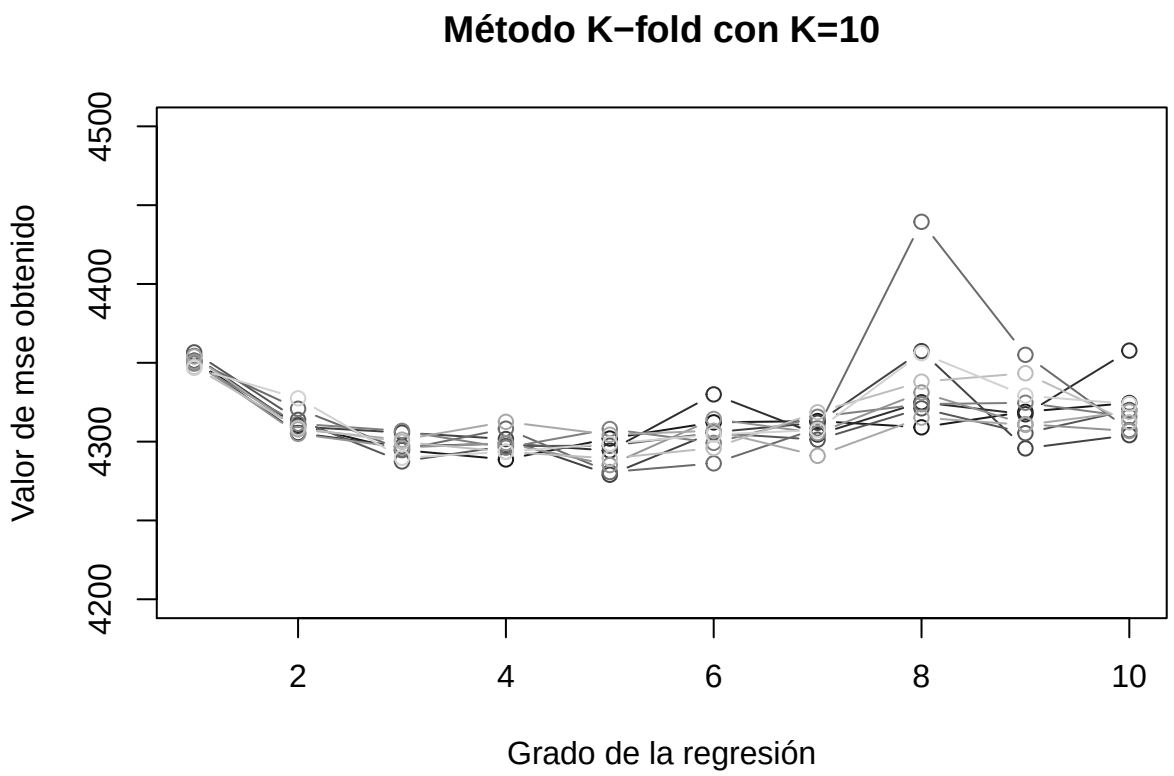



```
VuelosEnMinutos <- data.frame(distanciaVuelo, tiempoVueloEnMinutos)
head(VuelosEnMinutos)
```

```
##   distanciaVuelo tiempoVueloEnMinutos
## 1            532                   75
## 2            532                   80
## 3           1012                  115
## 4           1012                  105
## 5            532                   58
## 6            633                   80
```

Ahora volvemos a repetir lo mismo de antes (método K-fold con K=10) para verificar el grado de la relación que existe entre el tiempo de vuelo en minutos y la distancia entre aeropuertos (si hacemos la hipotesis que la relación entre estas dos variables es de tipo polinomial).

```
#Método k-fold con K=10
resultsWithoutCorrection <- rep(NA, 10)
resultsWithCorrection <- rep(NA, 10)

plot(1, type = "n", xlab="Grado de la regresión", ylab="Valor de mse obtenido", main = "Método K-fold c

cv.err <- rep(NA, 10)
counter <- 10

for(j in 1:10) {

  for(i in 1:10) {
    Reg.K10 <- glm(tiempoVueloEnMinutos~poly(distanciaVuelo, i), data=VuelosEnMinutos)
    cv.err <- cv.glm(VuelosEnMinutos, Reg.K10, K=10)
    resultsWithoutCorrection[i] <- cv.err$delta[1]
    resultsWithCorrection[i] <- cv.err$delta[2]
  }

  myColor = paste("grey", counter, sep="")
  lines(seq(1:10), resultsWithCorrection, type = "b", col=myColor)
  counter <- counter + 8

}
```



Con esta segunda serie de datos, se puede observar que el valor de mse más alto se obtiene por un grado de regresión 1, como en el caso precediente. Es más dificil sacar una conclusión de esta serie de datos, porque

vemos que el valor de mse no cambia tanto cuando cambia el grado de la relación polinomial. Se podría hacer un estudio más completo usando un método leave-one-out o bien usando un valor de K más alto para obtener resultados más precisos, pero entonces el código tendría que correr durante mucho más tiempo.

Bibliografia

STACKOVERFLOW. 2014. *Plot polynomial regression curve in R* [sitio web]. [Consulta: 24 noviembre 2016]. Disponible en: http://stackoverflow.com/questions/23334360/plot-polynomial-regression-curve-in-r

STACKEXCHANGE. 2012. *How to add second order terms into the model in R?* [sitio web]. [Consulta: 26 noviembre 2016]. Disponible en: http://stats.stackexchange.com/questions/25975/how-to-add-second-order-terms-into-the-model-in-r

STACKOVERFLOW. 2013. *R legend not working* [sitio web]. [Consulta: 26 noviembre 2016]. Disponible en: http://stackoverflow.com/questions/18870356/r-legend-not-working

STACKOVERFLOW. 2011. *R - How to draw an empty plot?* [sitio web]. [Consulta: 30 noviembre 2016]. Disponible en: http://stackoverflow.com/questions/4785657/r-how-to-draw-an-empty-plot

COLUMBIA UNIVERSITY. *Colors in R* [sitio web]. [Consulta: 6 diciembre 2016]. Disponible en: http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf

LOGMYFLIGHT. *Logmyflight - Personal and interactive flight statistics* [sitio web]. [Consulta: 6 diciembre 2016]. Disponible en: http://www.logmyflight.net