capítulo 5

2024-10-14

```
library(ISLR2)
library(boot)
```

Creamos los datos

Generamos una base de datos en RStudio con dos variables numéricas relacionadas por un polinomio de grado 3, siguiendo los siguientes pasos.

- Establecer la semilla para reproducibilidad (así tenemos todos los mismos datos, este paso no es del todo necesario)

```
set.seed(123)
```

- Definir el número de observaciones

```
n <- 1000
```

- Generar la variable independiente X

```
X <- seq(-20, 20, length.out = n)
```

- Generar la variable dependiente y como un polinomio de grado 3 $y = ax^3 + bx^2 + cx + d + ruido$,

```
a <- 1
b <- -2
c <- 3
d <- 5
ruido <- rnorm(n, mean = 0, sd = 10)  # Ruido aleatorio
```

```
Y <- a * X^3 + b * X^2 + c * X + d + ruido
```

- Creamos el data frame.

```
datos <- data.frame(X,  Y)
attach(datos)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     X, Y
```

- Exploramos los datos.

```
lmfit<-lm(Y ~ poly(X, 3))
```

```
lmfit
```

```
##
## Call:
## lm(formula = Y ~ poly(X, 3))
##
## Coefficients:
## (Intercept)  poly(X, 3)1  poly(X, 3)2  poly(X, 3)3
```

```
##        -262        88995        -7557        38360
```

```
Y[1]
```

```
## [1] -8860.605
```

```
predict(lmfit)[1]
```

```
##         1
## -8854.652
```

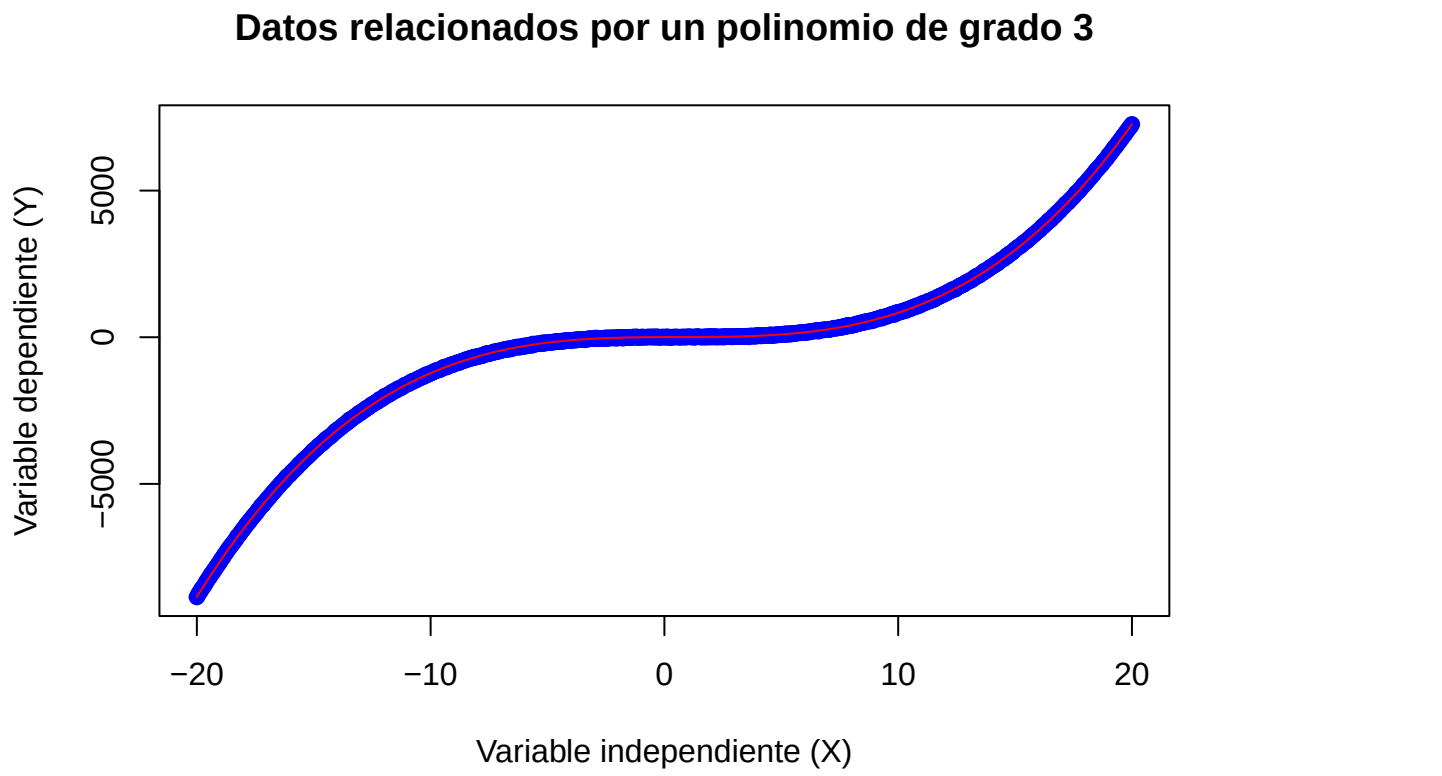
```
# Guardar el data frame en un archivo CSV
write.csv(datos, file = "datos.csv", row.names = FALSE)
```

```
head(datos)
```

```
##           X         Y
## 1 -20.00000 -8860.605
## 2 -19.95996 -8806.030
## 3 -19.91992 -8737.067
## 4 -19.87988 -8701.074
## 5 -19.83984 -8649.808
## 6 -19.79980 -8583.469
```

- Graficar los datos

```
plot( X,  Y, main = "Datos relacionados por un polinomio de grado 3",
     xlab = "Variable independiente (X)", ylab = "Variable dependiente (Y)",
     pch = 19, col = "blue")
lines(sort(datos$X), predict(lm(Y ~ poly(X, 3)), newdata = data.frame(x = sort(datos$x))), col = "red")
```



Leave-One-Out Cross-Validation (K=n)

Estudiamos que modelo polinomial se adapta mejor a los datos con LOOCV, polinomios desde grado 1 hasta 10. Veamos primero la orden a utilizar.

```
aver <- glm(Y ~ X, data = datos)
cv.err <- cv.glm(datos, aver)
names(cv.err)
```

```
## [1] "call"  "K"     "delta" "seed"
```

```
cv.err$delta
```

```
## [1] 1536508 1536504
```

```
cv.err$K
```

```
## [1] 1000
```

```
cv.error <- rep(0, 10)
for (i in 1:10) {
  glm.fit <- glm(Y ~ poly(X, i), data = datos)
  cv.error[i] <- cv.glm(datos, glm.fit)$delta[1]
}
cv.error
```

```
##  [1] 1.536508e+06 1.484635e+06 9.902116e+01 9.915569e+01 9.903423e+01
##  [6] 9.921710e+01 9.938485e+01 9.955239e+01 9.972490e+01 9.991395e+01
```

10-Fold Cross-Validation

Ahora con validación cruzada con 10 iteraciones y polinomios desde grado 1 hasta 10.

```
set.seed(17)
cv.error.10 <- rep(0, 10)
for (i in 1:10) {
  glm.fit <- glm(Y ~ poly(X, i), data = datos)
  cv.error.10[i] <- cv.glm(datos, glm.fit, K = 10)$delta[1]
}
cv.error.10
```

```
##  [1] 1.533187e+06 1.491312e+06 9.910213e+01 9.914910e+01 9.893925e+01
##  [6] 9.979209e+01 9.987284e+01 9.967586e+01 1.002647e+02 1.003185e+02
```

Bootstrapping

Aplicando Bootstrapping, evaluamos la variabilidad de las estimaciones para $\beta_0, \beta_1, \beta_2$ y β_3 del modelo polinomial $f(X) = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3$.

```
bootfuncion <- function(data, index)
  coef(lm(formula = Y ~ poly(X, 3), data = datos, subset = index))
bootfuncion(datos, 1:1000)
```

```
## (Intercept) poly(X, 3)1 poly(X, 3)2 poly(X, 3)3
##   -262.0393  88995.4271  -7556.7017  38360.4400
```

```
set.seed(1)
bootfuncion(datos, sample(1000, 1000, replace = T))
```

```
## (Intercept) poly(X, 3)1 poly(X, 3)2 poly(X, 3)3
##   -261.6257  88992.1234  -7549.5770  38347.2741
```

```r
boot(datos, bootfuncion, 3000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = datos, statistic = bootfuncion, R = 3000)
##
##
## Bootstrap Statistics :
##       original       bias    std. error
## t1*  -262.0393 -0.004207485   0.3204138
## t2* 88995.4271 -0.152489997   9.9650641
## t3* -7556.7017 -0.220842636   9.6145434
## t4* 38360.4400  0.257337132   9.6471565
```

```r
summary(lmfit)
```

```
##
## Call:
## lm(formula = Y ~ poly(X, 3))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.266  -6.459  -0.003   6.498  32.274
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -262.0393     0.3141  -834.3   <2e-16 ***
## poly(X, 3)1 88995.4271     9.9317  8960.7   <2e-16 ***
## poly(X, 3)2 -7556.7017     9.9317  -760.9   <2e-16 ***
## poly(X, 3)3 38360.4400     9.9317  3862.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.932 on 996 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.193e+07 on 3 and 996 DF,  p-value: < 2.2e-16
```