# Realizaremos una actividad para encontrar el orden o grado

# del polinomio que permita el mejor ajuste entre millas

# por galón y caballos de fuerza del conjunto de datos

# Autos

# PASO 1: #

library (ISLR) # Librería del texto guía

plot(Auto$horsepower, Auto$mpg, xlab="Horsepower", ylab="mpg", pch=19, col="blue")

grid()

# PASO 2: #

############################################

# MÉTODO DEL CONJUNTO DE VALIDACIÓN CRUZADA

############################################

# Seleccionemos el subconjunto de datos dividiendo 50-50:

cedula<-333 # Cambie este número por su cédula

set.seed (cedula)

porc <- 0.5 # Porcentaje de datos de entrenamiento

train<-sample (392 ,(392\*porc)) # índice del conjunto de entrenamiento con 196 datos

#Ilustramos los puntos de entrenamiento:

plot(Auto$horsepower, Auto$mpg, xlab="Horsepower", ylab="mpg", pch=19, col="blue")

grid()

points(Auto$horsepower[-train], Auto$mpg[-train], col="red",pch=19)

# PASO 3: #

# Ajustemos el modelo de grado 1 con el conjunto de entrenamiento

lm.fit<-lm(mpg~horsepower, data=Auto, subset =train)

# Calculemos el MSE:

attach (Auto)

mean((mpg-predict(lm.fit, Auto))[-train]^2)

# Ajustemos el modelo de grado 2 con el conjunto de entrenamiento

lm.fit2<-lm(mpg~poly(horsepower,2),data=Auto,subset =train )

# Calculemos el MSE:

mean((mpg-predict(lm.fit2,Auto))[-train]^2)

# Ajustemos el modelo de grado 3 con el conjunto de entrenamiento

lm.fit3<-lm(mpg~poly(horsepower,3),data=Auto,subset =train )

# Calculemos el MSE:

mean((mpg-predict(lm.fit3,Auto))[-train]^2)

##### ¿CUÁL DIO UN MENOR MSE? #######

# PASO 4: #

Train1<-Auto[train,] # 196 datos de entrenamiento

Test1<-Auto[-train,] # 196 datos de prueba

MSE<-vector()

for (i in 1:10){

Modelo1<-lm(mpg~poly(horsepower,i), data=Train1)

Pred1<-predict(Modelo1,Test1)

MSE[i]<-mean((Pred1-Test1$mpg)^2)

}

plot(1:10,MSE, xlab="Grado", ylab="MSE",type="b",col=4)

# PASO 5: #

###############

# MÉTODO LOOCV

###############

require(boot) # Librería con LOOCV incorporado

# Ajustemos el modelo de grado 1 con el conjunto de entrenamiento

glm.fit1<-glm(mpg~horsepower, data=Auto)

cv.err1<-cv.glm(Auto ,glm.fit1)

names(cv.err1)

cv.err1$delta

# Ajustemos el modelo de grado 2 con el conjunto de entrenamiento

glm.fit2<-glm(mpg~poly(horsepower,2), data=Auto)

cv.err2<-cv.glm(Auto ,glm.fit2)

names(cv.err2)

cv.err2$delta

# Ajustemos el modelo de grado 3 con el conjunto de entrenamiento

glm.fit3<-glm(mpg~poly(horsepower,3), data=Auto)

cv.err3<-cv.glm(Auto ,glm.fit3)

names(cv.err3)

cv.err3$delta

# PASO 6: #

#############################

# MÉTODO k-pliegues (k-fold)

#############################

# Ajustemos el modelo de grado 1 con el conjunto de entrenamiento

glm.fit1<-glm(mpg~horsepower, data=Auto)

cv.err.k1<-cv.glm(Auto ,glm.fit1, K=10)

names(cv.err.k1)

cv.err.k1$delta

# Ajustemos el modelo de grado 2 con el conjunto de entrenamiento

glm.fit.k2<-glm(mpg~poly(horsepower,2), data=Auto)

cv.err.k2<-cv.glm(Auto ,glm.fit2, K=10)

names(cv.err.k2)

cv.err.k2$delta

# Ajustemos el modelo de grado 3 con el conjunto de entrenamiento

glm.fit3<-glm(mpg~poly(horsepower,3), data=Auto)

cv.err.k3<-cv.glm(Auto ,glm.fit3, K=10)

names(cv.err.k3)

cv.err.k3$delta

# PASO 7: #

###############

# Bootstrap

###############

# En este caso usamos el Bootstrap para estimar

# el error estándar de las estimaciones beta\_0 y beta\_1

boot.fn<-function (data,index){

Modelo<-lm(mpg~horsepower,data=data, subset =index)

return (coef(Modelo))

}

cedula<-19282

set.seed(cedula)

muestra<-sample (392,392,replace =T)

boot.fn(Auto, muestra)

# Usemos la función boot

boot(Auto,boot.fn,1000)

#Comparando con el modelo

Mod.completo<-lm(mpg~horsepower,data=Auto)

summary(Mod.completo)