Introducción a analitíca

Modulo 2- Tarea 1

Nota: 5.0 muy bien

Por:

Daniela Pico Arredondo

Juan Sebastián Falcón Granada

Jhonatan mse Muñoz García

Entregado a:

Mauricio Mazo Lopera García Muñoz

Universidad Nacional de Colombia

Sede Medellín

Facultad de ciencias

1) Descargue de Yahoo Finance la base de datos de los precios de cierre diarios de la acción que se le asignó a su grupo en el periodo que va del 1 de enero de 2016 hasta el 31 de diciembre de 2020.

A continuación se muestra el encabezado de la base de datos

```
## [1] 1258    7
```

Date	Open	High	Low	Close	Adj.Close	Volume
2016-01-04	36.01	36.01	35.36	35.75	32.58296	18784400
2016-01-05	35.86	36.12	35.49	35.64	32.48270	25340700
2016-01-06	35.50	36.14	35.36	35.82	32.64677	18165700
2016-01-07	35.25	35.68	34.88	35.04	31.93587	22591400
2016-01-08	35.13	35.28	34.61	34.65	31.58041	21962200
2016-01-11	34.75	35.12	34.52	34.94	31.84471	18726600

De la base datos ORCL se observa que esta compuesta por 1258 observaciones y 7 variables

Contexto

Oracle Corporation ofrece productos y servicios que abordan los entornos de tecnología de la información empresarial en todo el mundo. Su oferta de software como servicio en la nube Oracle incluye varias aplicaciones de software en la nube, incluida la planificación de recursos empresariales (ERP) en la nube Oracle Fusion, la gestión del rendimiento empresarial en la nube Oracle Fusion, la gestión de fabricación y la cadena de suministro en la nube Oracle Fusion, la gestión del capital humano en la nube Oracle Fusion, Oracle Fusion publicidad en la nube y experiencia del cliente, y suite de aplicaciones NetSuite. La compañía también ofrece soluciones industriales basadas en la nube para diversas industrias; Licencias de aplicaciones de Oracle; y servicios de soporte de licencias de Oracle. Además, proporciona tecnologías de infraestructura de negocios de licencia y en la nube, como Oracle Database, una base de datos empresarial; Java, un lenguaje de desarrollo de software; y middleware, incluidas herramientas de desarrollo y otros, donde:

- Date: Fecha del registro
- Open: precio de la accion en el mercado financiero
- High: precio más alto
- Low: precio más bajo
- Close: precio de cierre
- Adj. Close: precio de cierre despues de los ajustes para todas las divisiones y distribuciones de dividendos aplicables
- Volume: Volumen de la acción

construya una base de datos con la misma estructura de los datos Smarket

En clase se estudio que los retornos del dia t, es decir la variable "Today" es hallada mediante la siguiente formula:

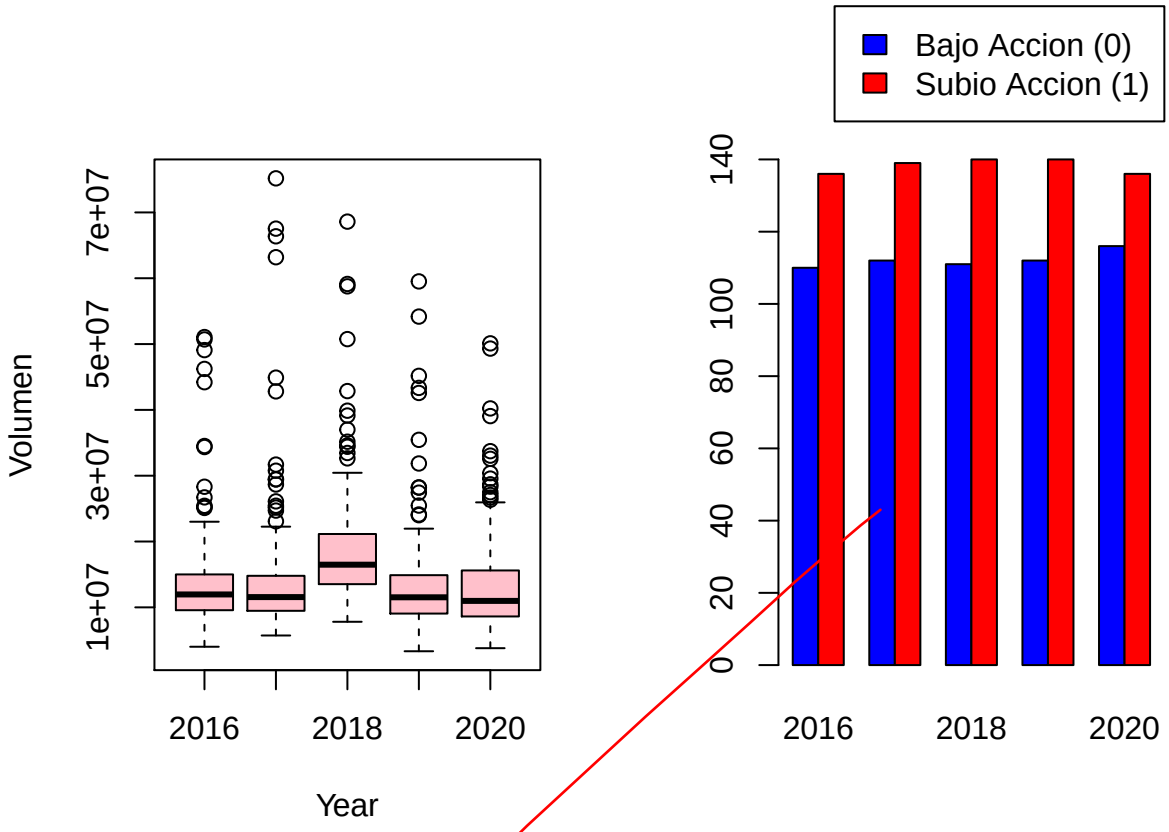
$$r_t = ln(\frac{close_i}{close_{t-1}})x100$$

```
close=data$Close
date=data$Date
Today=c()
for(i in 1:nrow(data)){
  Today[i]=log(close[i]/(close[i-1]))*100
}
Direction= as.factor(ifelse(Today<0,"down","up"))
Year=as.numeric(substr(date,1,4))
df= data.frame(Year,"Volume"=data$Volume,Today,Direction)

#rezagos
df$lag1= Lag(df$Today, 1)
df$lag2= Lag(df$Today, 2)
df$lag3= Lag(df$Today, 3)
df$lag4= Lag(df$Today, 4)
df$lag5= Lag(df$Today, 5)

df=na.omit(df)
kable(head(df))%>%
  kable_styling(position = "center")%>%
  kable_styling(latex_options = "HOLD_position")
```

	Year	Volume	Today	Direction	lag1	lag2	lag3	lag4	lag5
7	2016	18457300	1.223170	up	0.8334493	-1.1192511	-2.2016102	0.5037822	-0.3081695
8	2016	28322900	-3.715321	down	1.2231699	0.8334493	-1.1192511	-2.2016102	0.5037822
9	2016	19523600	2.061926	up	-3.7153214	1.2231699	0.8334493	-1.1192511	-2.2016102
10	2016	25425000	-1.944632	down	2.0619257	-3.7153214	1.2231699	0.8334493	-1.1192511
11	2016	21423100	1.252383	up	-1.9446325	2.0619257	-3.7153214	1.2231699	0.8334493
12	2016	25278700	-1.781329	down	1.2523828	-1.9446325	2.0619257	-3.7153214	1.2231699

Analisis Descriptivo

```
par(mfrow=c(1,2))
boxplot(df$Volume~df$Year,ylab="Volumen", xlab="Year", col="pink")
barplot(table(df$Direction,df$Year),beside=T,col=c("blue","red"))
legend("topright",c("Bajo Accion (0)","Subio Accion (1)"),fill = c("blue",
"red"),inset = c(0,-0.3), xpd = TRUE)
```



En el boxplot del volumen por año se observa que la mediana del Volumen de la acción estubo en valores similares excepto en el año 2018, donde fue mas alta, se aprecia la influencia de datos atipicos en todos los años, siendo el mayor en 2017 con un valor aproximado de 80000000 USD.

En el diagrama de barras se observa un patron similar en cada año, a principio de cada año el precio de la acción baja pero luego aumenta significativamente.

Utilizando validación cruzada, encuentre el K (el número de vecinos), del modelo KNN, que mejores resultados arroje en términos del error de prueba estimado para predecir si el precio de la acción sube (o se mantiene igual) o baja en función de los 5 "lags" pasados y el volumen

```
library(dplyr)
library(class)
library(tidyverse)

train=df %>% filter(Year<2019)
test=df %>% filter(Year>=2019)



set.seed(1234)
sp_ctrl=trainControl(method="cv", number=5)
sp_train=train(Direction  ~ ., data=train, method="knn", tuneLength=20, trControl=sp_ctrl, preProcess=c
sp_train
```

```
## k-Nearest Neighbors
##
```

```
## 748 samples
##   8 predictor
##   2 classes: 'down', 'up'
##
## Pre-processing: centered (8), scaled (8)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 598, 599, 598, 598, 599
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.8102282  0.6137863
##    7  0.8222371  0.6379837
##    9  0.8302192  0.6547270
##   11  0.8302282  0.6537068
##   13  0.8435705  0.6806960
##   15  0.8382192  0.6688384
##   17  0.8341745  0.6603124
##   19  0.8395615  0.6704464
##   21  0.8462371  0.6842197
##   23  0.8569485  0.7063896
##   25  0.8556331  0.7044218
##   27  0.8596152  0.7124761
##   29  0.8462371  0.6845139
##   31  0.8502461  0.6927090
##   33  0.8489038  0.6894340
##   35  0.8475705  0.6866119
##   37  0.8556063  0.7032598
##   39  0.8582908  0.7091567
##   41  0.8596421  0.7117153
##   43  0.8596600  0.7117269
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 43.
```

Como se puede observar usando cross validation se tiene un k optimo igual a 43.

Con los datos de entrenamiento, ajuste un modelo logístico, un KNN con K encontrado en el item (b), un LDA y un QDA Para cada modelo obtenga la matriz de confusión y el estimador del error de prueba.

Modelo knn

```
train2<-select(train,-Year)
test2<-select(test,-Year)
XTrain = train2 %>% dplyr::select(-Direction)
XTest = test2 %>% dplyr::select(-Direction)
modeloknn=knn(train = XTrain, test = XTest, cl=train2$Direction , k = 43)
data1<-data.frame(modeloknn,test2$Direction)#KNN
```

Modelo logistico

```
modglm=glm(train2$Direction~lag1 + lag2 + lag3 + lag4 + lag5 + Volume, data=train, family=binomial)
pred=predict(modglm,newdata=XTest,type = "response")
```

```
predicted=ifelse(pred > 0.5, 1,0)
data2<-data.frame(test2$Direction,pred= predicted)
```

Modelo LDA

```
modlda=lda(train2$Direction ~ lag1 + lag2 + lag3 + lag4 + lag5 + Volume, data=train)
predicted1 <- predict(modlda,newdata=XTest,type = "response") #LDA
data3=data.frame(predicted1$class,test2$Direction)
```

Modelo QDA

```
modqda=qda(train2$Direction ~ lag1 + lag2 + lag3 + lag4 + lag5 + Volume, data=train)
predicted2 <- predict(modqda,newdata=XTest,type = "response") #QDA
data4=data.frame(predicted2$class,test2$Direction)
```

Matrices de confusión para los modelos

```
library(kableExtra)

c1=table(data1)
c2=table(data2)
c3=table(data3)
c4=table(data4)
kable(c1) #MATRIZ DE CONFUSION KNN
```

	down	up
down	35	39
up	193	237

```
kable(c2)#MATRIZ DE CONFUSION LOGISTIC
```

	0	1
down	15	213
up	18	258

```
kable(c3)#MATRIZ DE CONFUSION LDA
```

	down	up
down	15	18
up	213	258

```
kable(c4)#MATRIZ DE CONFUSION QDA
```

	down	up
down	57	51
up	171	225

Estimador del error de prueba

A partir de las matrices de confusión se obtiene el MSE para los datos de prueba

```
sum(diag(c1)/sum(c1))#KNN
```

```
## [1] 0.5396825
```

```
sum(diag(c2)/sum(c2))#LOGISTIC
```

```
## [1] 0.5416667
```

```
sum(diag(c3)/sum(c3)) #LDA
```

```
## [1] 0.5416667
```

```
sum(diag(c4)/sum(c4)) #QDA
```

```
## [1] 0.5595238
```

Se busca el modelo que minimice el error cuadratico medio, es decir el que tenga menor valor, en este caso se obtuvo en el modelo con K vecinos más cercanos.

Saque conclusiones de los resultados obtenidos en el item (c)

Del punto anterior se observa que el MSE de todos los modelos fue similar, incluso el modelo logistico y el modelo de analisis discriminante lineal obtuvo el mismo valor, aunque fue levemente menor en el modelo knn lo que significa que para esta situación especifica el mejor clasificador para determinar si sube o si baja la acción de la empresa resulta al utilizar el modelo con k vecinos más cercanos, siendo k=43 vecinos.

3) Pruebe que si en el modelo de regresión lineal múltiple se tiene que p > n (el número de covariables es mayor que el tamaño muestral) entonces los estimadores de los coeficientes $\widehat{\beta}$, no son únicos. ¿Cuáles son las alternativas para "resolver" este problema?

Tenemos el modelo de regresion lineal multiple $yi = \beta 0 + \beta_1 xi1 + \beta 2xi2 + \beta 3xi3 + ... + \beta kxik + \varepsilon i$, el cual podemos denotar $Y = X\beta + \epsilon$ donde las componentes son matrices $Ynx1, X_{nxp}, \beta_{px1}, \epsilon_{nx1}$ con n observaciones y p parametros.

Se tiene la funcion de minimos cuadrados $S(\beta) = (Y - X\beta)^T(Y - X\beta)$

Con el fin de estimar β se deriva $S(\beta)$ y se iguala al vector nulo, se tiene que

$(X^TX)\widehat{\beta} = X^TY$ Con $(X^TX)_{pxp}$

Sabiendo que p>n se va probar que $Ran(X^TX) \leq p$.

Se tiene que $Ran(X^TX) \leq min\{Ran(X^T), Ran(X)\}$, ya que $Ran(X^T) = Ran(X) \leq min\{n, p\}$

Luego, bajo el supuesto de p>n, $Ran(X^TX) \leq min\{n, p\}$

Se concluye que $Ran(X^TX) \leq n < p$, por lo que $(X^TX)\widehat{\beta} = X^TY$ tiene infinitas soluciones y $\widehat{\beta}$ no es unica

Las alternativas para solucionar este problema son la implementacion de regresion mediante el metodo Ridge y metodo Lasso

4) Se pide simular un modelo lineal de la forma $Y = X\beta + \epsilon$ dadas las siguientes condiciones:

a) T es una vble numerica independiente

b) X1,X2,X3, variables numericas independientes

c) X4 una categorisca con 3 caegorias

d) $(\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5)^\top = (1, 0.3, 0.6, -1, 1.5, -2)^\top$

e) $\epsilon \sim N\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right)$, donde $\sigma = 2$

f) n = 500

Ahora, con este modelo, compare modelo de todas las variables vs variables individuales dados los metodos de validacion cruzada vistos en clase:

Y = b0 +x$b1$+$x2$b2+x3$b3$+$x4$b4+e

Este es el modelo que se pide ajustar.

Para ello, se generaran aleatoreamente valores de distribuciones para x1, x2, x3 (numericos) y x4 (categoricos)

```
# Valores para la simulacion pedida
set.seed(1234)
n = 500
x1 = runif(n=n, min = 0, max=1)
x2 = rpois(n=n, lambda = 3)
x3 = rgamma(n=n, shape = 3,scale =2 ,)
x4 = c("x1", "x2", "x3n")
x4 = sample(x4, n, replace = T)
```

```
# Matriz del modelo

x = model.matrix(~ x1+x2+x3+x4)

# Vector de parametros
beta = c(1,0.3,0.6,-1,1.5,-2)
error = rt(n, 9)
```

```
# Variable respuesta
library(dplyr)
y = x %*% beta + error
```

```
# Base de datos con la simulacion realizada

data = data.frame(y,x[,2:6]) # No se toma intercepto pues interes matriz de parametros
```

Ahora, se utiliza el metodo de bootstrap para obtener un IC al 95% de σ, R^2, R^2_{adj}

```r
library(boot)
```

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

```
## The following object is masked from 'package:survival':
##
##     aml
```

```
## The following object is masked from 'package:lattice':
##
##     melanoma
```

```r
sigma = function(data,i){
  modelo = lm(y~x, data=data, subset = i)
  summary(modelo)$sigma
}

x2 = boot(data, sigma, R=500)

boot.ci(x2, conf = 0.95, type = "all")
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 500 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = x2, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.990,  1.286 )   ( 0.973,  1.264 )
##
## Level     Percentile            BCa
## 95%   ( 0.999,  1.291 )   ( 1.015,  1.327 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

Se toma por convencion el IC normal, donde finalmente se tiene que la estimacion de un IC para sigma es (0.990, 1.286)

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 500 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = z, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.9043,  0.9472 )   ( 0.9077,  0.9515 )
##
## Level     Percentile            BCa
```

```
## 95%   ( 0.9001,  0.9439 )   ( 0.8918,  0.9412 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

Ahora, el IC para el parametro de R^2 dado los datos de la simulacion con una confianza del 95% se encuentra entre 0.9043 y 0.9472

```
# Ic para R2 ajustado
r_2_ajustado = function(data,i){
  modelo = lm(y~x, data=data, subset = i)
  summary(modelo)$adj.r.squared
}
w = boot(data,r_2_ajustado, R=500)

boot.ci(w, conf = 0.95, type = "all")
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 500 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = w, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.9032,  0.9472 )   ( 0.9072,  0.9526 )
##
## Level     Percentile            BCa
## 95%   ( 0.8974,  0.9428 )   ( 0.8949,  0.9413 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

Finalmente, un IC al 95% de confianza para el R^2_{adj} es (0.9032, 0.9472)

Ahora, se pide errores con distibucion t student con 9 grados de libertad.

```
v = 9
var_esp = v/(v-2)
var_esp
```

```
## [1] 1.285714
```

Está será la varianza esperada para los errores.

Simulando el modelo anterior, se tiene que:

$\beta_{1,3} = 2$

```
set.seed(1039705595)
s = 2

# Valores para la simulacion pedida
n = 500
x1 = runif(n=n, min = 0, max=1)
```

```
x2 = rpois(n=n, lambda = 3)
x3 = rgamma(n=n, shape = 3,scale =2 ,)
x4 = c("A", "B", "C")
x4 = sample(x4, n, replace = T)

x = model.matrix(~x1+I(x1^3)+x2+x3+x4) # Asi lo pide el ejercicio
```

```
beta = c(1,0.3,2,0.6,-1,1.5,-2)
error = rnorm(n,0,s)

y = x %*% beta + error
```

```
# Base de datos simulacion 2
data2 = data.frame(y, x[,2:7])
```

Utilizando esta base de datos se separará datos de prueba y de entrenamiento

```
prop = 0.7
ni = nrow(data2)
sample1 = sample(1:n, size = prop*ni)
```

```
Train = data2[sample1,] # Datos de entrenamiento

Test = data2[-sample1,] # Datps de prueba
```

```
y_train = Train$y
y_test = Test$y
```

```
# Modelo con x1
mod1 = lm(y~., data = Train)
mse1 = mean((y_test-predict(mod1,Test))^2)

# Modelo con x1^2
mod2 = lm(y~x1+I(x1^2)+x2+x3+x4B+x4C, data = Train)
mse2 = mean((y_test-predict(mod2,Test))^2)

# Modelo con x1^3
mod3 = lm(y~x1+I(x1^3)+x2+x3+x4B+x4C, data = Train)
mse3 = mean((y_test-predict(mod3,Test))^2)

# Modelo con x1^4
mod4 = lm(y~x1+I(x1^4)+x2+x3+x4B+x4C, data = Train)
mse4 = mean((y_test-predict(mod4,Test))^2)

# Modelo con x1^5
mod5 = lm(y~x1+I(x1^5)+x2+x3+x4B+x4C, data = Train)
mse5 = mean((y_test-predict(mod5,Test))^2)

# Modelo con x1^6
mod6 = lm(y~x1+I(x1^6)+x2+x3+x4B+x4C, data = Train)
mse6 = mean((y_test-predict(mod6,Test))^2)
```

```
# Modelo con x1^7
mod7 = lm(y~x1+I(x1^7)+x2+x3+x4B+x4C, data = Train)
mse7 = mean((y_test-predict(mod7,Test))^2)

# Modelo con x1^8
mod8 = lm(y~x1+I(x1^8)+x2+x3+x4B+x4C, data = Train)
mse8 = mean((y_test-predict(mod8,Test))^2)

# Modelo con x1^9
mod9 = lm(y~x1+I(x1^9)+x2+x3+x4B+x4C, data = Train)
mse9 = mean((y_test-predict(mod9,Test))^2)

# Modelo con x1^10
mod10 = lm(y~x1+I(x1^10)+x2+x3+x4B+x4C, data = Train)
mse10 = mean((y_test-predict(mod10,Test))^2)
```

```
tabla = cbind(c(mse1,mse2,mse3,mse4,mse5,mse6,mse7,mse8,mse9,mse10))
colnames(tabla) = c("MSE")
rownames(tabla) = c("Grado 1", "Grado 2", "Grado 3", "Grado 4", "Grado 5", "Grado 6", "Grado 7", "Grado
```

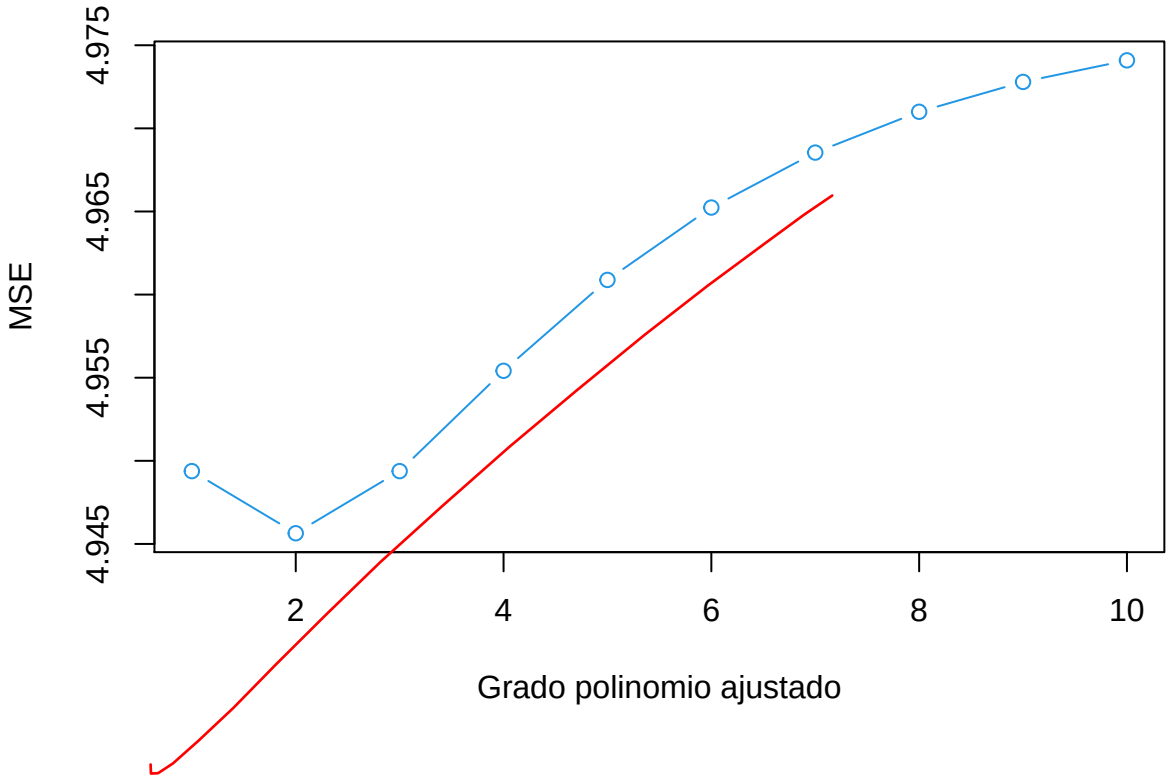
```
tabla
```

```
##                MSE
## Grado 1   4.949381
## Grado 2   4.945648
## Grado 3   4.949381
## Grado 4   4.955415
## Grado 5   4.960883
## Grado 6   4.965237
## Grado 7   4.968543
## Grado 8   4.970998
## Grado 9   4.972795
## Grado 10  4.974091
```

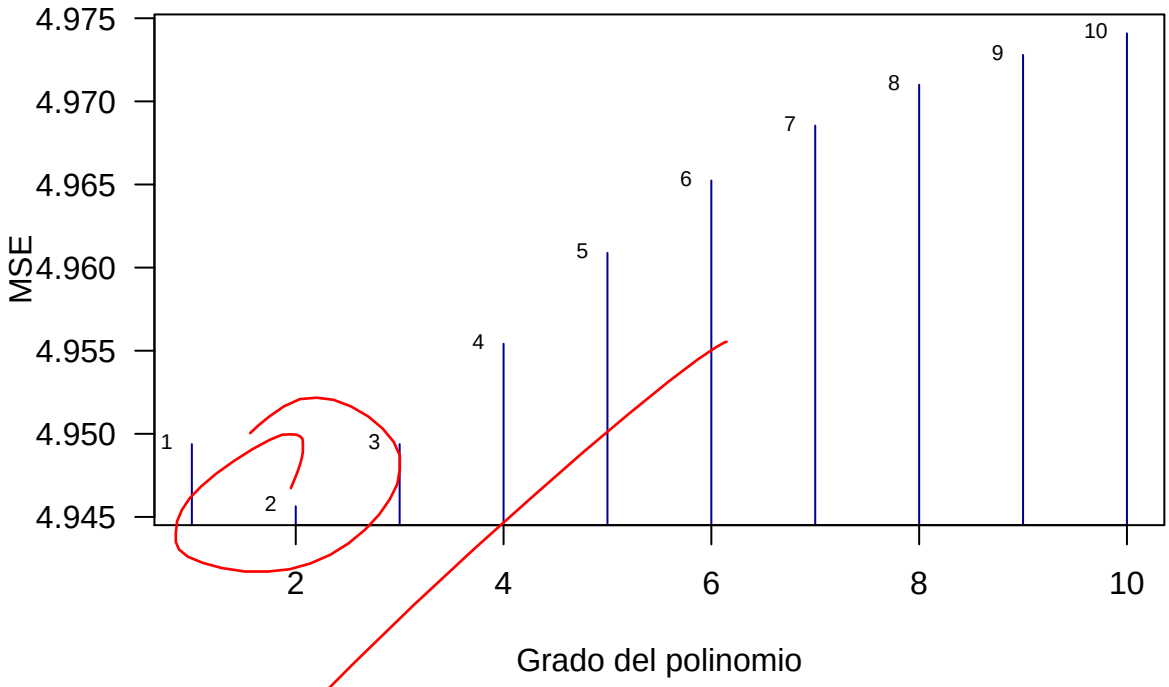
De la anterior tabla,se obtienen los valores de los respectivos MSE para cada polinomio ajustado segun el grado. Se desea el modelo de menor grado posible a menor MSE; teniendo en cuenta siempre el principio de parsimonia.

ASi, se selecciona el modelo de grado 2, pues su medida de MSE es la menor y es el modelo con el grado mas bajo.

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




```
plot (c(mse1,mse2,mse3,mse4,mse5, mse6,mse7,mse8,mse9,mse10), xlab = "Grado del polinomio", ylab = "MSE
labels = c("1","2","3","4","5","6","7","8","9","10")
text(c(mse1,mse2,mse3,mse4,mse5, mse6,mse7,mse8,mse9,mse10), labels,cex = 0.7, pos=2)
```



De forma grafica se observa igualmente que el mejor polinomio es el de grado 2.

5) La base de datos a trabajar (SURGICAL) contiene informacion acerca de la supervivencia de pacientes con intervenciones quirurgicas hepaticas Las variables presentadas son:

- liver_test : Puntuacion de la funcion de prueba hepatica
- enzyme_test : Resultado prueba de enzimas.
- pindex : Indice de pronostico.
- bcs : Puntuacion de coagulacion sanguinea.
- age : edad en años del paciente en cuestion
- gender : Genero del paciente. Variable indicadora que toma el valor de 1 para femenino y 0 para masculino
- alc_mod: Consumo de alcohol. Variable indicadora que toma el valor para 0 como no consume licor, 1 consumo moderado.
- alc_heavy : Variable indicadora del historial del consumo de alcohol. 0 para no consumo, 1 para consumo fuerte.
- y : Tiempo de supervivencia.

Se busca explicar el tiempo de supervivencia a traves de las variables disponibles. Es decir, ¿cuales de todas estas variables son las mejores para explicar el tiempo de supervivencia de un paciente sometido a una intervencion quirurgica hepatica?

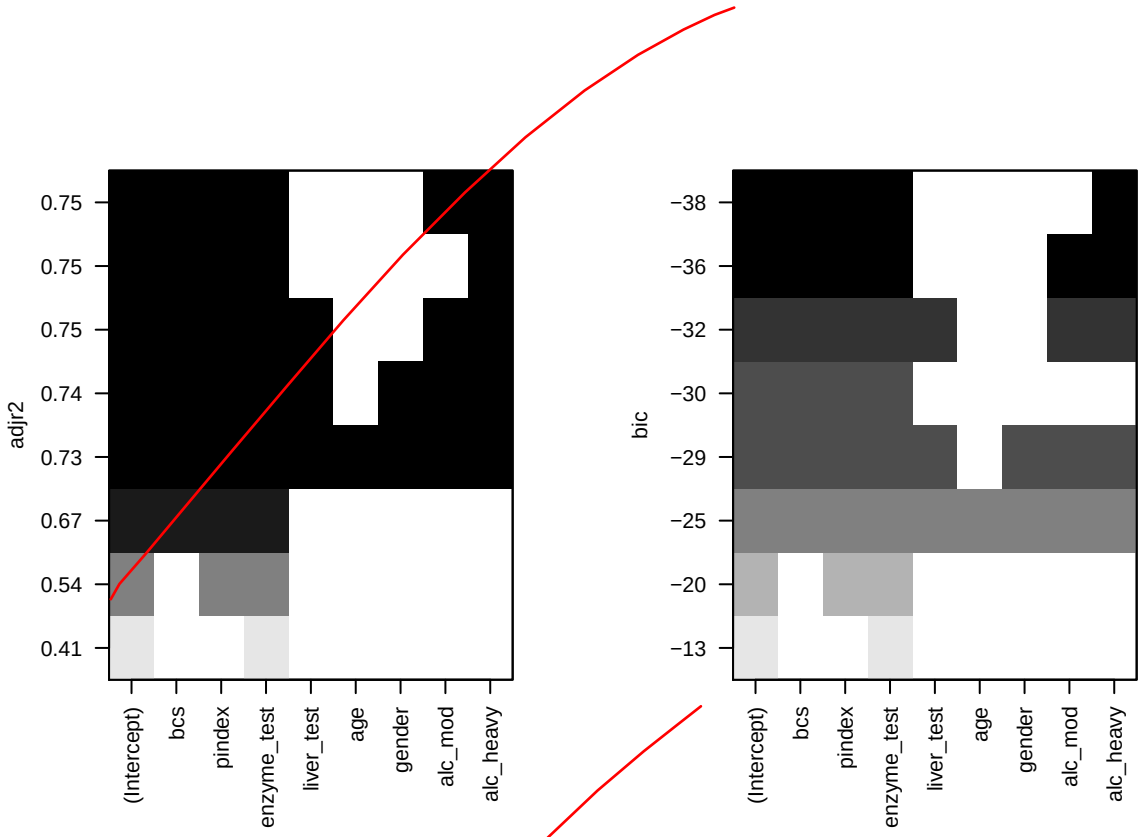
Procediendo con validacion cruzada, se selecciona el 70% de la base de datos para entrenamiento y el 30% como datos de validacion o prueba (valores tomados empiricamente).

```
set.seed(1998)
porcentaje = 0.7 # Proporcion seleccioar base de datos
t_1 = sample(length(surgical$y), size = (length(surgical$y)*porcentaje))
train = surgical[t_1,]
test = surgical[-t_1,]
```

Una vez que la base de datos ha sido seleccionada, se utiliza la funcion *regsubset* para la seleccion de variables "hacia adelante".

```
library("leaps")
reg_fitted = regsubsets(y~., data= train, nvmax = 8, method = "forward")
```

Con este metodo, se obtiene el siguiente grafico para la toma de decisiones segun diversos criterios.

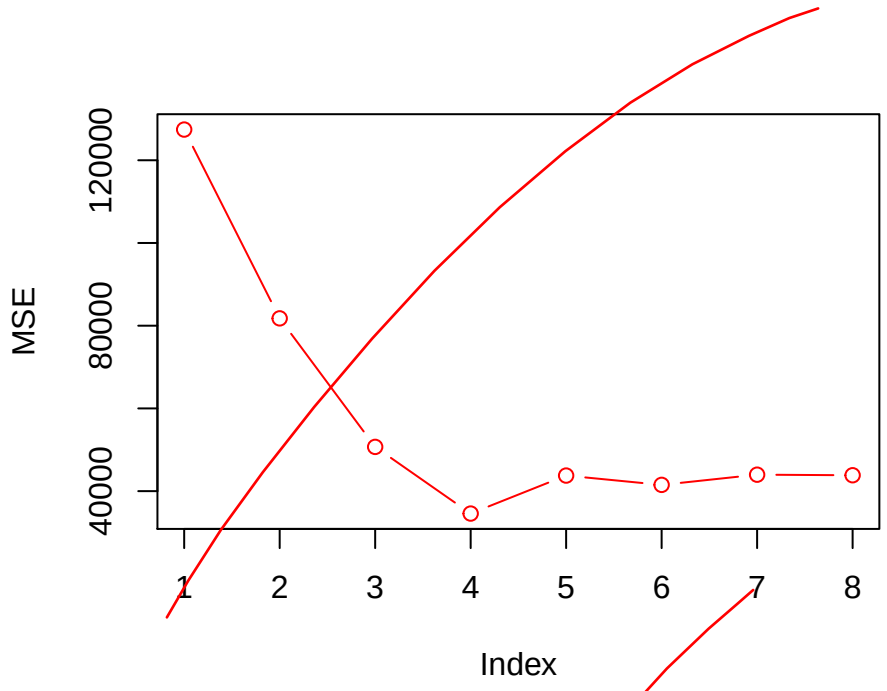


Con este grafico, se busca seleccionar via el criterio de R^2_{adj} y bic los modelos más "consistentes" entre si, puesto que se busca seleccionar el valor mas alto en el primero teniendo siempre en cuenta el principio de parsimonia y con bic el valor mas pequeño.

- Segun R^2_{adj} el mejor modelo es el que tiene 6 variables; intercepto, bcs, pindex, enzym_test, alc_mod y alc_heavy
- Segun criterio bic el mejor modelo contiene 5 variables; intercept,bcs, pindex, enzyme_test y alc_heaby

Ahora, ¿Que modelo es mejor? Para responder a esta pregunta, se procede a analizar el MSE de cada modelo.

```
predict.regsubsets =function (object,newdata,y){
  form<-as.formula(object$call[[2]])
  mat<-model.matrix(form ,newdata)
  val.errors =rep(NA, (ncol(mat)-1))
  for(i in 1:length(val.errors)){
    coefi<-coef(object ,id=i)
    xvars<-names (coefi)
    pred<-mat[,xvars]%*%coefi
    val.errors [i]= mean((y-pred)^2)
  }
  val.errors
}
```



Con el grafico anterior, se observa los valores del MSE. El modelo con menor MSE es el de indice 4.

```
mse <- which.min(MSE)
regfit.for <- regsubsets(y~.,data=surgical ,nvmax =8, method = "forward")
```

```
coef(regfit.for,mse)
```

```
## (Intercept)      pindex enzyme_test  liver_test   alc_heavy
## -789.012038    7.876493    7.547724  125.473761  359.874634
```

Segun esto, las variables que mejor explica la variable respuesta están dadas por las anteriores.

El modelo es, finalmente:

$Y = -789.912038 + 7876493 * pindex + 7.547724 * enzyme.test + 125.473761 * liver.test + 359.87464 * alc.heavy$

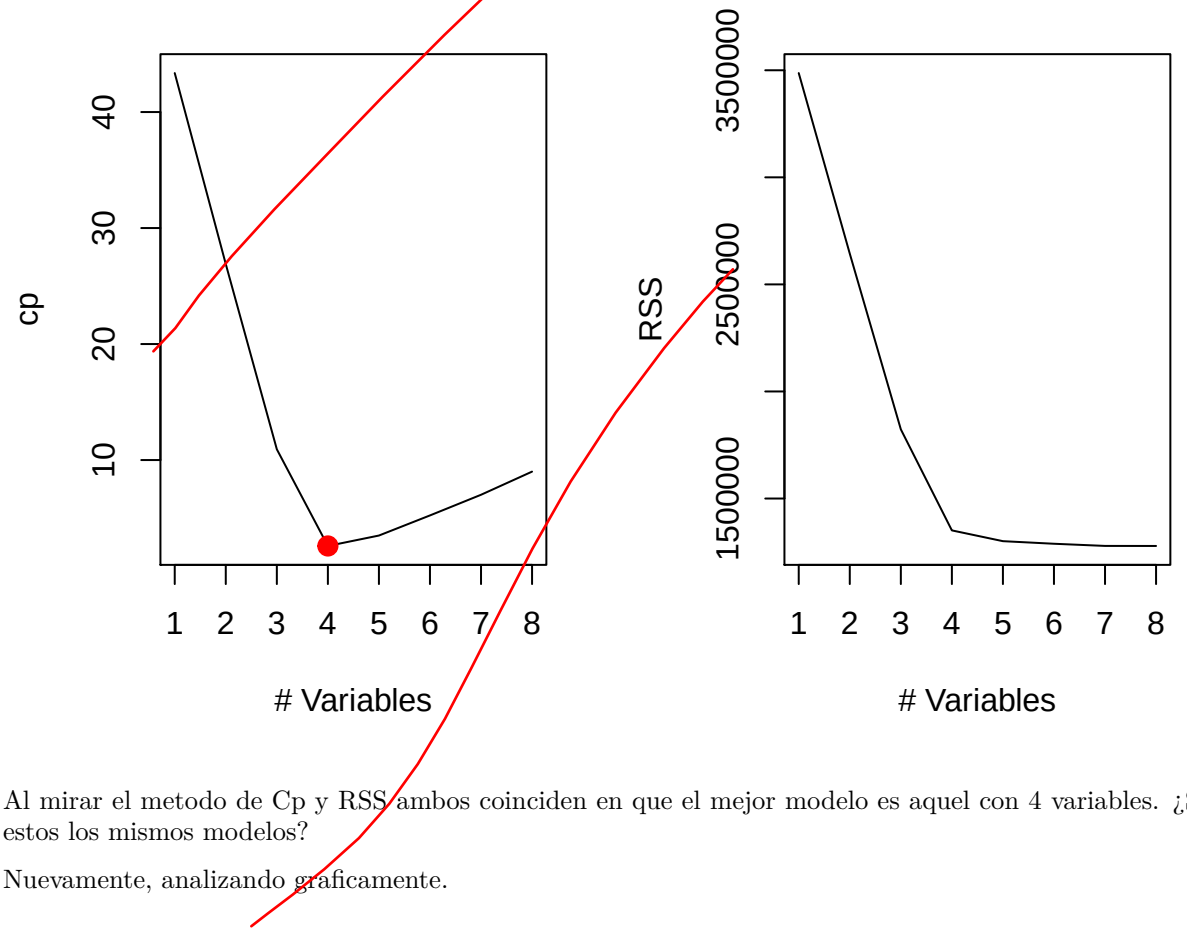
Sin embargo, ¿esto cambiará radicalmente si se utiliza otro metodo de seleccion? Lo recomendable seria verificar varios metodos de seleccion de variables y observar segun el problema, cual seria el modelo más optimo.

Suponga que hay un experto que está a cargo de la investigacion. Esta persona podría ayudar a seleccionar cuales de todos los modelos resultantes de todos los posibles metodos de seleccion de variables (llamese cp, backward, step-wise, por nombrar algunos) serian mas utiles acorde a su conocimiento en el campo.

¿Por qué? Sin importar que metodo se utilice, se espera cierta consistencia entre metodos. Puedan diferir quizas en algunas variables pero de manera general, deberian seleccionar mas o menos las mismas. Un ejemplo para ilustrar esto con otros criterios de seleccion en forward.

```
reg_fitted_cp = regsubsets(y~., data= train, nvmax = 8)
reg.summary = summary(reg_fitted_cp)
```

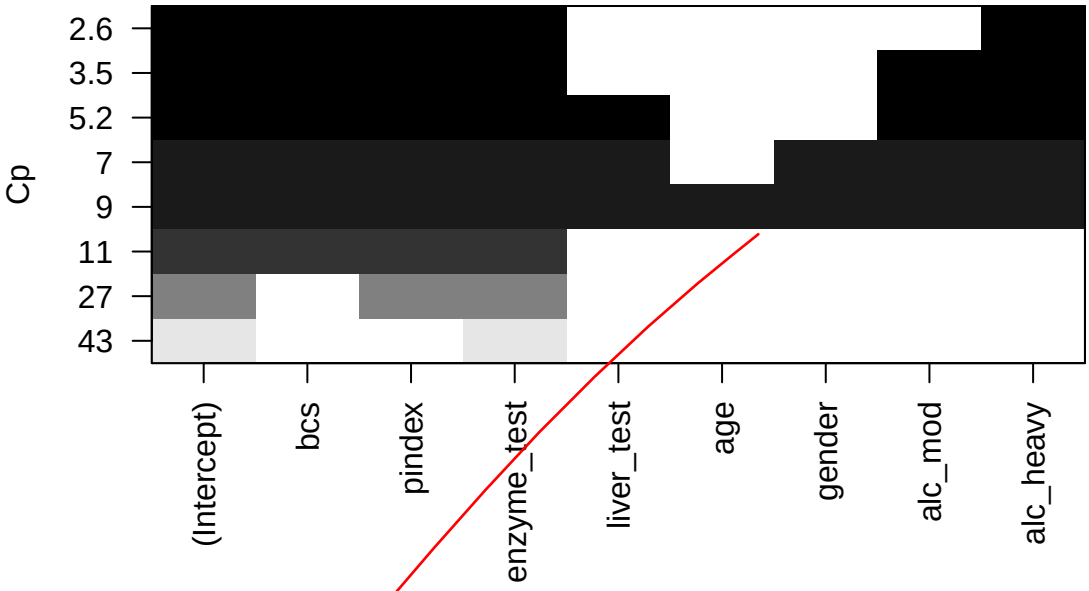
```
par(mfrow = c(1,2))
plot(reg.summary$cp, xlab = "# Variables", ylab = "cp", type = "l")
a1 = which.min(reg.summary$cp)
points(a1, reg.summary$cp [a1], col = "red", cex = 2, pch = 20)
plot(reg.summary$rss, xlab = "# Variables", ylab = "RSS", type = "l")
```



Al mirar el metodo de Cp y RSS ambos coinciden en que el mejor modelo es aquel con 4 variables. ¿Serán estos los mismos modelos?

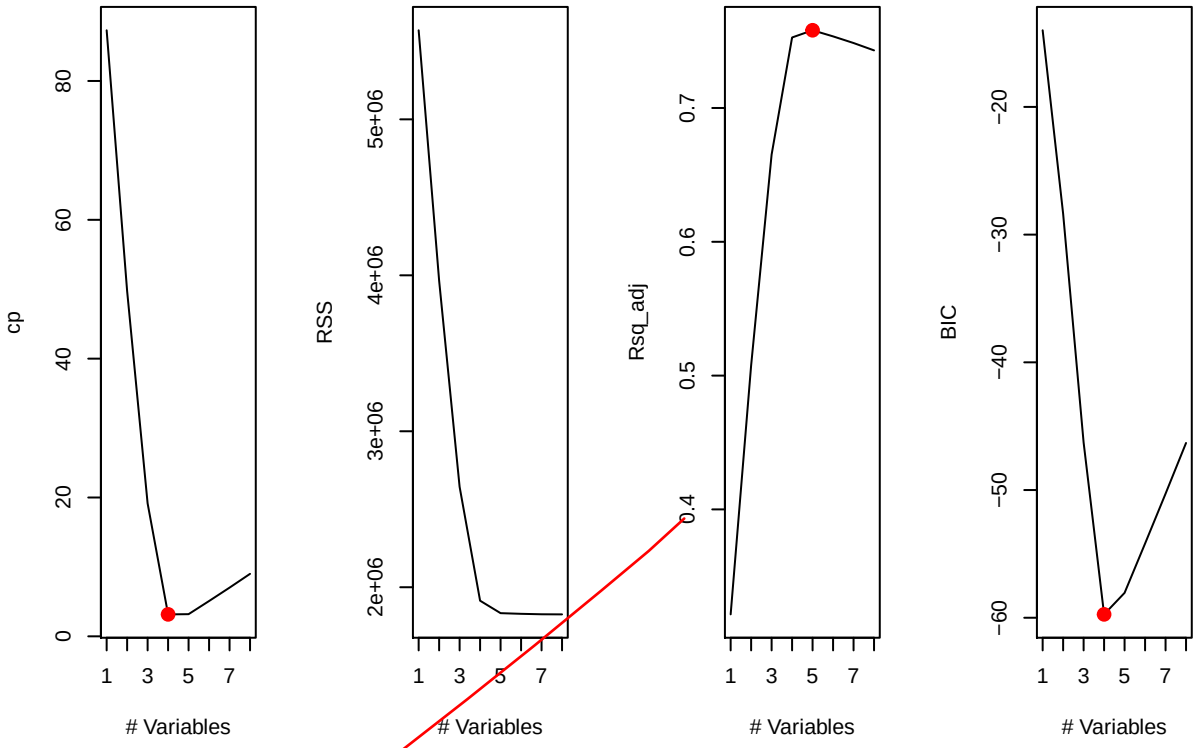
Nuevamente, analizando graficamente.

```
plot(reg_fitted_cp, scale = "Cp")
```

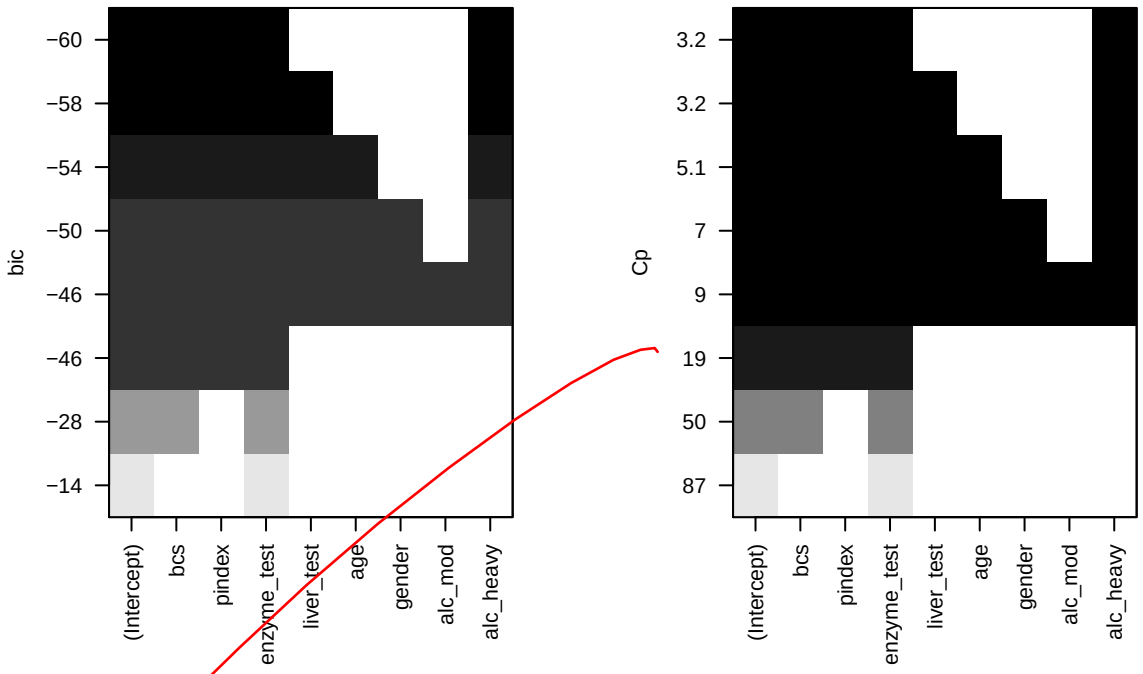


Se muestra solo un metodo mas seleccionado y se observa que, dada una seleccion forward por el criterio de Cp de Mallows, los metodos son muy parecidos entre si pues selecciona practicamente las mismas variables pero, ¿Y si se parte de una seleccion hacia atras? ¿Es igual la seleccion?

```
regfit.back = regsubsets(y~.,data=surgical ,nvmax =8, method = "backward")
reg.back.summary = summary(regfit.back)
par(mfrow = c(1,4))
plot(reg.back.summary$cp, xlab = "# Variables", ylab = "cp", type = "l")
a1 = which.min(reg.back.summary$cp)
points(a1, reg.back.summary$cp [a1], col = "red", cex = 2, pch = 20)
plot(reg.back.summary$rss, xlab = "# Variables", ylab = "RSS", type = "l")
plot(reg.back.summary$adjr2, xlab = "# Variables", ylab = "Rsq_adj", type = "l")
a4 = which.max(reg.back.summary$adjr2)
points(a4, reg.back.summary$adjr2 [a4], col = "red", cex = 2, pch = 20)
plot(reg.back.summary$bic, xlab = "# Variables", ylab = "BIC", cex = 2, pch = 20, type = "l")
a5 = which.min(reg.back.summary$bic)
points(a5, reg.back.summary$bic [a5], col = "red", cex = 2, pch = 20)
```



Al observar rapidamente los metodos de seleccion, todos coinciden en que el modelo mas optimo es de 4 vvariables, por cualquier criterio de los mostrados dado los valores. ¿Será nuevamente los modelos con las mismas variables? Se ilustran solo 2 de los criterios, a modo de ejemplo.



Finalmente, se observa que al aplicar metodo de selccion forward y backward, dado los criterios probados como bic, R^2_{adj}, cp y rss, se observa que los modelos seleccionados son, en terminos generales, el mismo; exeptuando cambios muy puntuales pero que en ultimas, no son cambios significativos.

De esta manera, se concluye que el modelo final será el primero hallado, teniendo en cuenta los comentarios anteriores y una posible verificacion del experto ya que estos modelos casi que convergen al mismo.

6) Utilice las técnicas ridge y lasso para regularizar las bases de datos BASE_DATOS_1 y BA-SE_DATOS_2. Según estas técnicas, ¿cuáles variables aparentemente muestran no ser relevantes para explicar la variable aleatoria Y ?

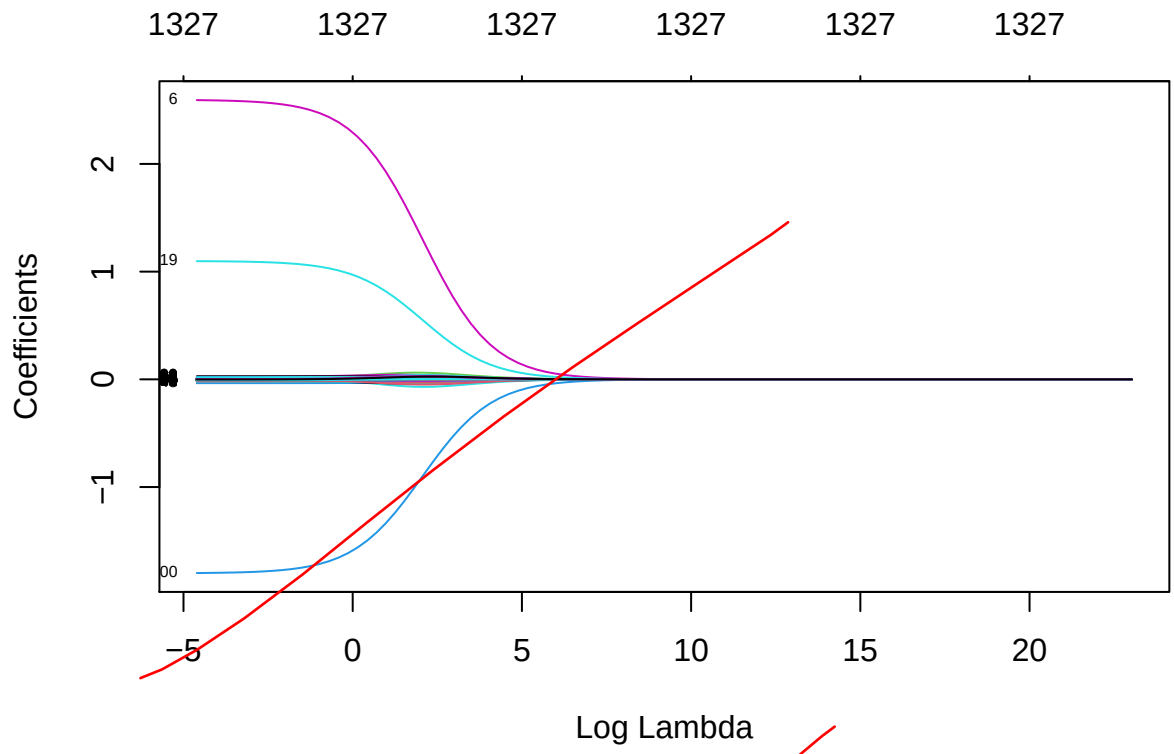
Base de datos 1

```
base1=read.table(file.choose(),header=T,sep=" ")
```

```
require(ISLR)
library(glmnet)
library(Rcpp)
##TRABAJANDO CON RIDGE REGRESSION
x<-model.matrix(Y~.,base1)[,-1]
y<-base1$Y
gridz<-10^seq(-2,10, length=100)
ridge.mod<-glmnet(x,y,alpha=0, lambda=gridz)
dim(coef(ridge.mod))
```

```
## [1] 1328  100
```

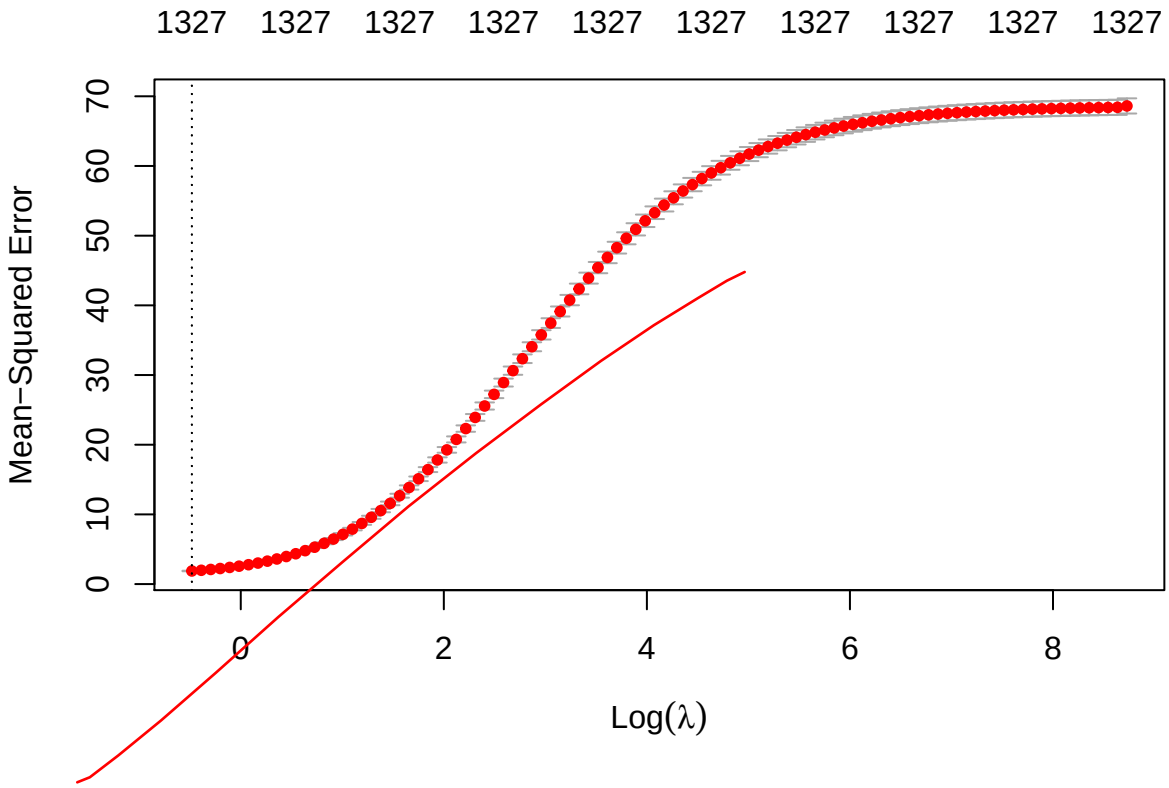
```
plot(ridge.mod, xvar="lambda", label=TRUE)
```



Se observa que el modelo que se ajusto tiene 1328 variables y 100 distintos valores de lamda, es decir, que para cada valor de lamda se estiman 1328 parametros. En el plot del modelo se observa que las covariables 0, 119 y 6 son las más significativas, siendo la covariable 6 la que mas explica a Y

A continuación se seleccionara el mejor valor de lamda pero usando validación cruzada

```
cedula<-1234
set.seed(cedula)
train<-sample(1: nrow(x), nrow(x)/2)
test<- -train
y.test<-y[test]
cv.out<-cv.glmnet(x[train,],y[train],alpha=0)
plot(cv.out)
```



A continuación se muestra el MSE para cada valor de lamda, se busca el valor de lamda en donde se obtenga el menor error cuadratico medio, graficamente se observa que esta entre e^{λ} con λ entre [-0.5, 0].

```
bestlam<-cv.out$lambda.min
bestlam
```

```
## [1] 0.6176261
```

Con el conjunto de datos de entreamiento el lambda que minimiza la suma de cuadrados medios fue de 0.676261, con este valor se aplica Ridge reggresion de la siguiente manera

```
ridge.pred<-predict(ridge.mod, s=bestlam,newx=x[test,])
mean((ridge.pred-y.test)^2)
```

```
## [1] 1.249494
```

el MSE de prueba es 1.249994

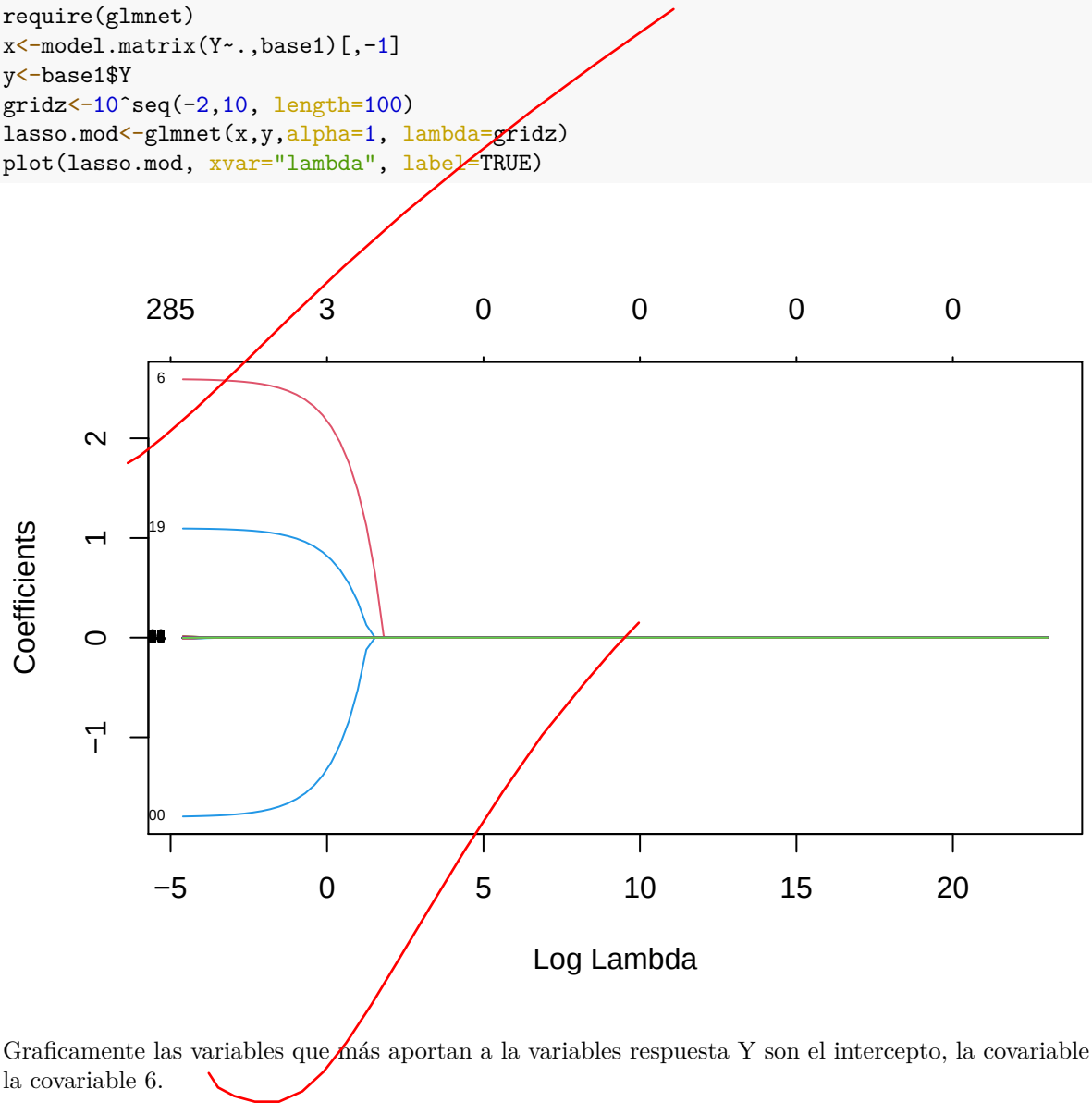
```
out<-glmnet (x,y,alpha=0)
predict(out,type="coefficients",s=bestlam)[1:20,]
```

```
##   (Intercept)            x1            x2            x3            x4
##  5.3386520044  0.0053204869 -0.0007282800 -0.0022703303  0.0017456111
##            x5            x6            x7            x8            x9
```

```
##  0.0042121390  2.3990289490 -0.0050922008  0.0151311077 -0.0026113980
##           x10           x11           x12           x13           x14
## -0.0016748673  0.0006581419 -0.0040353854  0.0033875239 -0.0080172056
##           x15           x16           x17           x18           x19
## -0.0032646788 -0.0019154261  0.0040006092 -0.0025157290  0.0009127641
```

las candidatas a descartar del modelo son las que en valor absoluto sean cercanos a cero, en este caso serían X11, X2, X19, aunque casi todas toman valores cercanos a 0.

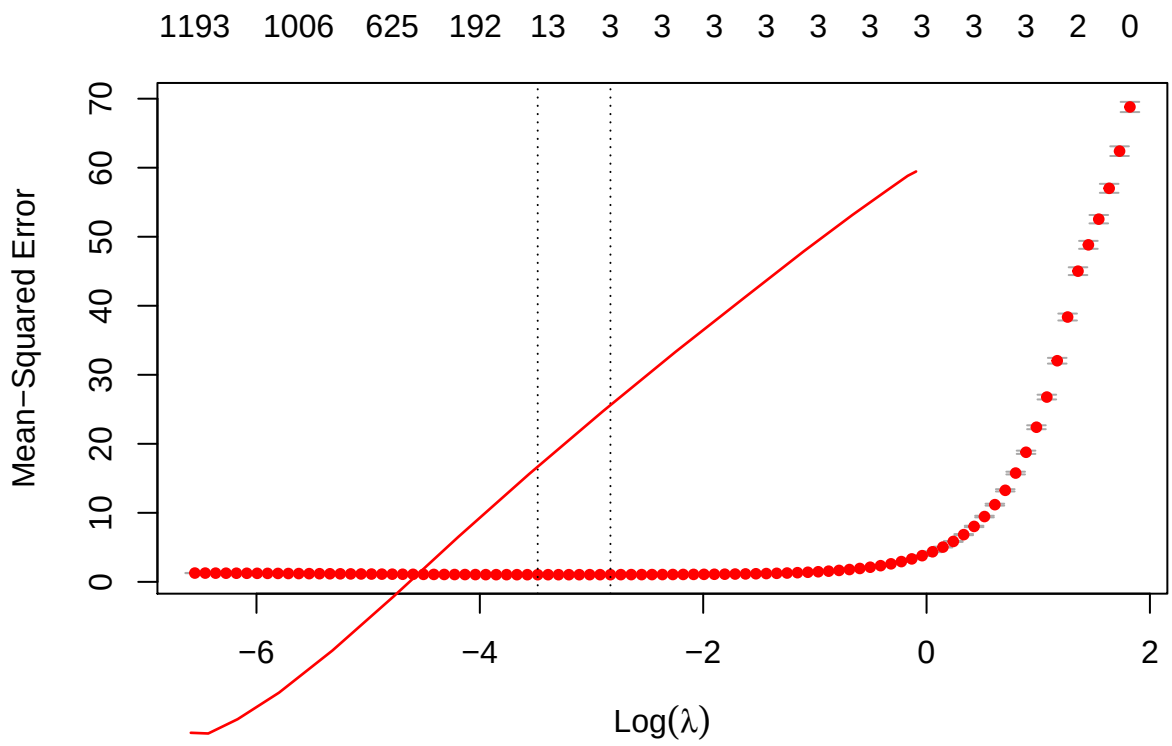
```
#TRABAJANDO CON LASSO REGRESSION
require(ISLR)
require(glmnet)
x<-model.matrix(Y~.,base1)[,-1]
y<-base1$Y
gridz<-10^seq(-2,10, length=100)
lasso.mod<-glmnet(x,y,alpha=1, lambda=gridz)
plot(lasso.mod, xvar="lambda", label=TRUE)
```



Graficamente las variables que más aportan a la variables respuesta Y son el intercepto, la covariable 119 y la covariable 6.

```
cedula<-1
set.seed(cedula)
train<-sample(1: nrow(x), nrow(x)/2)
test<- -train
y.test<-y[test]
```

```
cv.out<-cv.glmnet(x[train,],y[train],alpha=1)
plot(cv.out)
```




```
bestlam<-cv.out$lambda.min
bestlam
```

```
## [1] 0.03075528
```

De la grafica se observa que el lambda que minimiza el MSE es 0.0307

```
lasso.pred<-predict(lasso.mod, s=bestlam,newx=x[test,])
mean((lasso.pred-y.test)^2)
```

```
## [1] 1.012285
```

Usando los datos de prueba el MSE fue de 1.012285

```
out<-glmnet (x,y,alpha=1)
lasso.coef<-predict(out,type="coefficients",s=bestlam)[1:20,]
lasso.coef
```

```
## (Intercept)          x1          x2          x3          x4          x5
##    3.129152    0.000000    0.000000    0.000000    0.000000    0.000000
```

```
##          x6          x7          x8          x9         x10         x11
##    2.583821    0.000000    0.000000    0.000000    0.000000    0.000000
##         x12         x13         x14         x15         x16         x17
##    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
##         x18         x19
##    0.000000    0.000000
```

Finalmente se observa que todas la covariables se descartan, excepto el intercepto y la covariable X6.

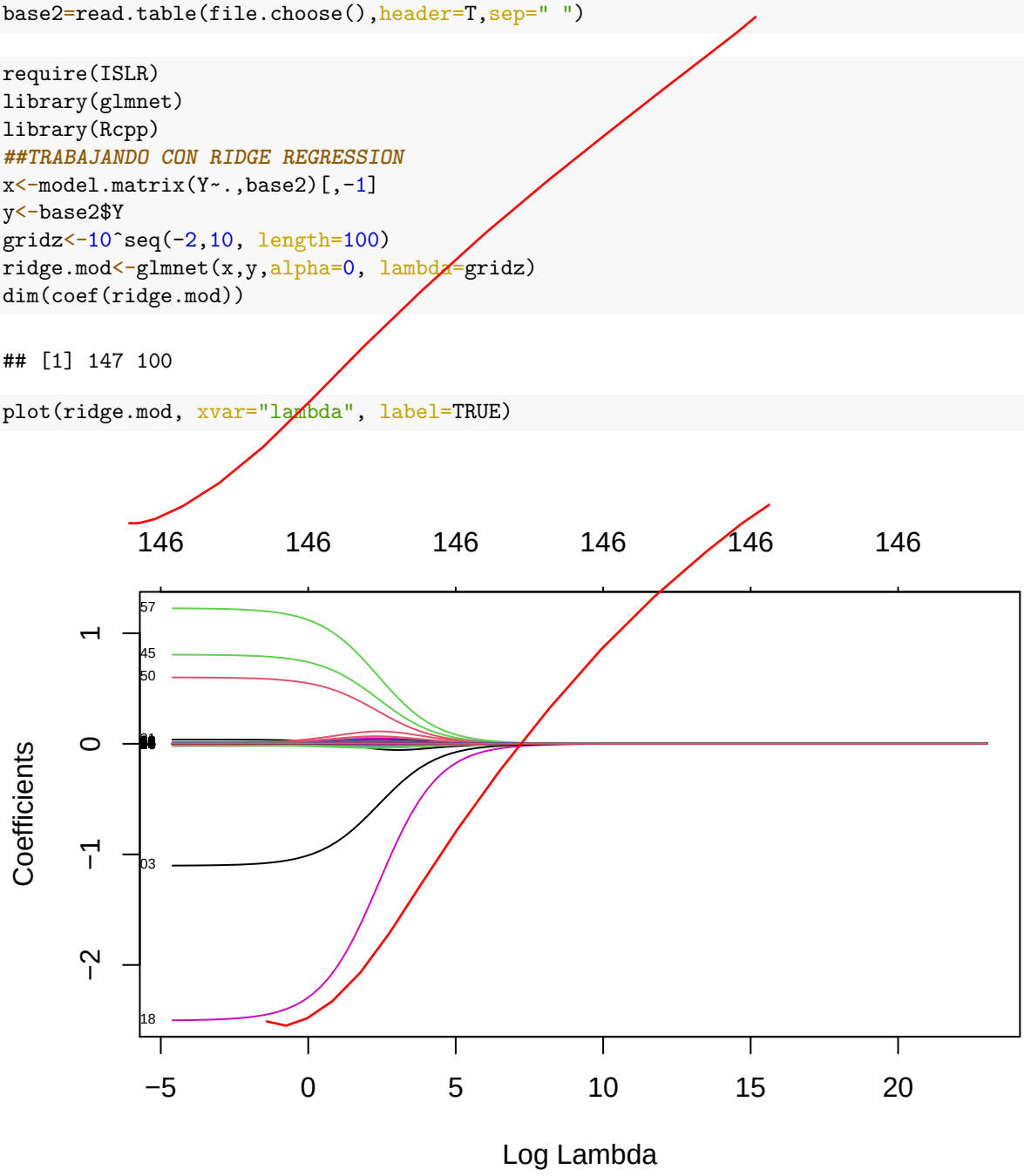
Base de datos 2

```
base2=read.table(file.choose(),header=T,sep=" ")
```

```
require(ISLR)
library(glmnet)
library(Rcpp)
##TRABAJANDO CON RIDGE REGRESSION
x<-model.matrix(Y~.,base2)[,-1]
y<-base2$Y
gridz<-10^seq(-2,10, length=100)
ridge.mod<-glmnet(x,y,alpha=0, lambda=gridz)
dim(coef(ridge.mod))
```

```
## [1] 147 100
```

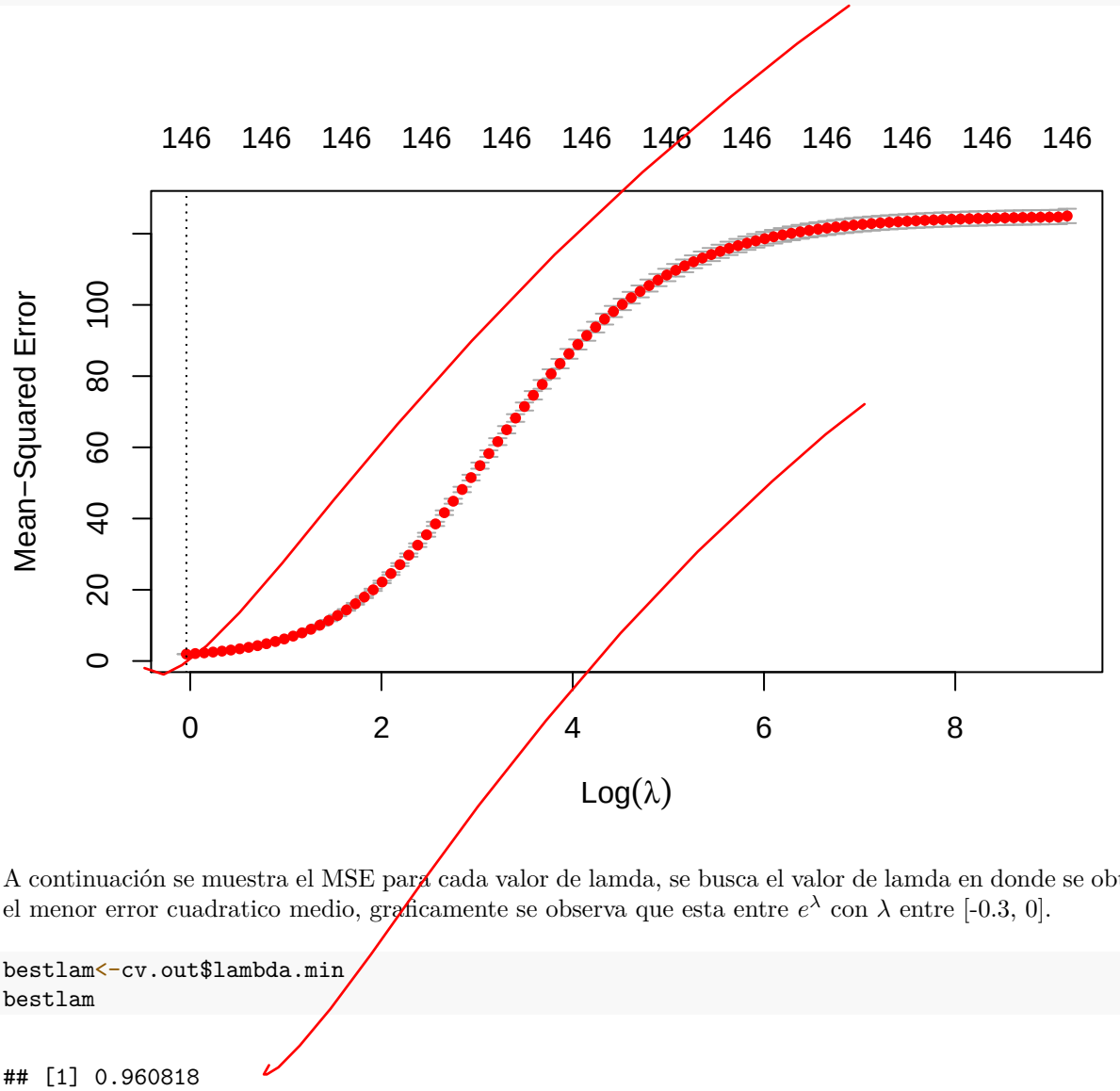
```
plot(ridge.mod, xvar="lambda", label=TRUE)
```



Se observa que el modelo que se ajusto tiene 147 variables y 100 distintos valores de lamda, es decir, que para cada valor de lamda se estiman 147 parametros. En el plot del modelo se observa que las covariables 18, 103, 50, 45 y 57 son las más significarivas, siendo la covariable 18 la que mas explica a Y

A continuación se seleccionara el mejor valor de lamda pero usando validación cruzada

```
cedula<-1234
set.seed(cedula)
train<-sample(1: nrow(x), nrow(x)/2)
test<- -train
y.test<-y[test]
cv.out<-cv.glmnet(x[train,],y[train],alpha=0)
plot(cv.out)
```



A continuación se muestra el MSE para cada valor de lamda, se busca el valor de lamda en donde se obtenga el menor error cuadratico medio, graficamente se observa que esta entre e^{λ} con λ entre [-0.3, 0].

```
bestlam<-cv.out$lambda.min
bestlam
```

```
## [1] 0.960818
```

Con el conjunto de datos de entreamiento el lambda que minimiza la suma de cuadrados medios fue de 0.960818, con este valor se aplica Ridge reggresion de la siguiente manera

```
ridge.pred<-predict(ridge.mod, s=bestlam,newx=x[test,])
mean((ridge.pred-y.test)^2)
```

```
## [1] 1.803808
```

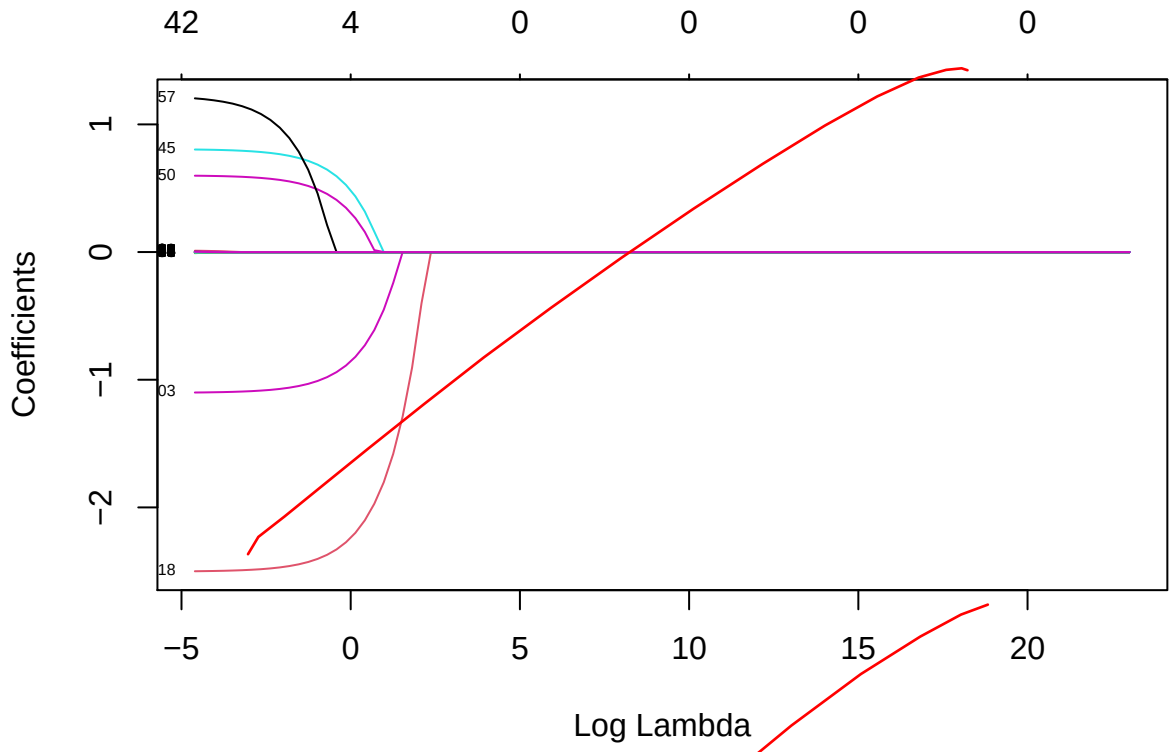
el MSE de prueba es 1.803808

```
out<-glmnet (x,y,alpha=0)
predict(out,type="coefficients",s=bestlam)[1:20,]
```

```
##   (Intercept)            x1            x2            x3            x4
## -4.3433994514 -0.0005212099 -0.0012035822 -0.0058478714  0.0063764780
##            x5            x6            x7            x8            x9
##  0.0073943533  0.0004357416 -0.0046706056  0.0053723093 -0.0035054460
##           x10           x11           x12           x13           x14
##  0.0027940534  0.0009632078 -0.0079142698  0.0061356276 -0.0018141062
##           x15           x16           x17           x18           x19
##  0.0051744656  0.0028044068 -0.0135383176 -2.3000810602 -0.0005414276
```

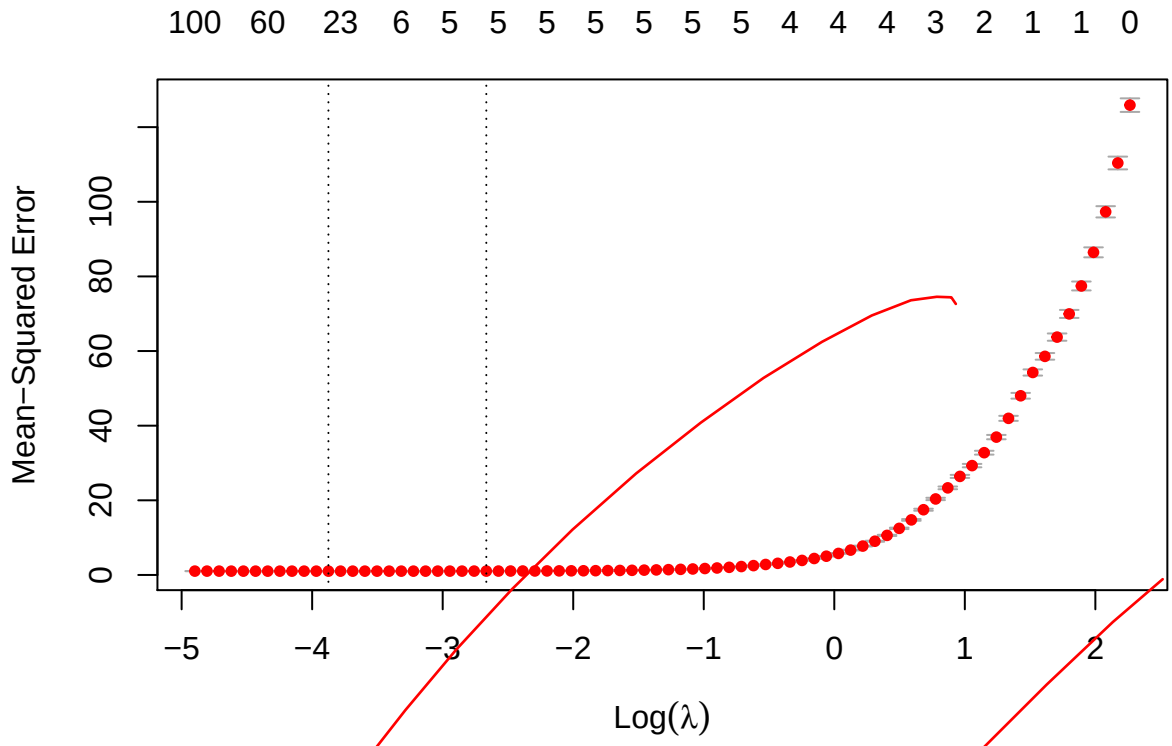
las candidatas a descartar del modelo son las que en valor absoluto sean cercanos a cero, en este caso serían X1, X6, X11,x19, aunque casi todas toman valores cercanos a 0.

```
#TRABAJANDO CON LASSO REGRESSION
require(ISLR)
require(glmnet)
x<-model.matrix(Y~.,base2)[,-1]
y<-base2$Y
gridz<-10^seq(-2,10, length=100)
lasso.mod<-glmnet(x,y,alpha=1, lambda=gridz)
plot(lasso.mod, xvar="lambda", label=TRUE)
```



Graficamente las variables que más aportan a la variables respuesta Y son x18, x103,x50,x45,x57

```
cedula<-1
set.seed(cedula)
train<-sample(1: nrow(x), nrow(x)/2)
test<- -train
y.test<-y[test]
cv.out<-cv.glmnet(x[train,],y[train],alpha=1)
plot(cv.out)
```




```
bestlam<-cv.out$lambda.min
bestlam
```

```
## [1] 0.0207485
```

De la grafica se observa que el lambda que minimiza el MSE es 0.0207485

```
lasso.pred<-predict(lasso.mod, s=bestlam,newx=x[test,])
mean((lasso.pred-y.test)^2)
```

```
## [1] 0.9920548
```

Usando los datos de prueba el MSE fue de 0.9920548

```
out<-glmnet (x,y,alpha=1)
lasso.coef<-predict(out,type="coefficients",s=bestlam)[1:20,]
lasso.coef
```

```
## (Intercept)          x1          x2          x3          x4          x5
##   -3.956996    0.000000    0.000000    0.000000    0.000000    0.000000
##          x6          x7          x8          x9         x10         x11
##    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
##         x12         x13         x14         x15         x16         x17
##    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
##         x18         x19
##   -2.494539    0.000000
```

Finalmente se observa que todas la covariables se descartan, excepto el intercepto y la covariable X18.