Tp1 mlg

Trabajo práctico 1

Ejercicio 1

Primer paso: seteamos el directorio de trabajo y luego levantamos los datos guardándolos en un objeto nuevo

```
datos<-read.table("prostata.txt", header=TRUE)
```

a) Realizamos el ajuste lineal pedido, siendo lpsa la variable dependiente y todo el resto de las variables como covariables. Luego escribimos como resulta el modelo ajustado basado en los datos.

```
reg<-lm(lpsa~., data=datos)
summary(reg)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.76644 -0.35510 -0.00328  0.38087  1.55770
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.181561   1.320568   0.137  0.89096
## lcavol       0.564341   0.087833   6.425 6.55e-09 ***
## lweight      0.622020   0.200897   3.096  0.00263 **
## age         -0.021248   0.011084  -1.917  0.05848 .
## lbph         0.096713   0.057913   1.670  0.09848 .
## svi          0.761673   0.241176   3.158  0.00218 **
## lcp         -0.106051   0.089868  -1.180  0.24115
## gleason      0.049228   0.155341   0.317  0.75207
## pgg45        0.004458   0.004365   1.021  0.31000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6995 on 88 degrees of freedom
## Multiple R-squared:  0.6634, Adjusted R-squared:  0.6328
## F-statistic: 21.68 on 8 and 88 DF,  p-value: < 2.2e-16
```

i. El modelo propuesto será:

$$y = \beta_0 + \beta_1 lcavol + \beta_2 lweight + \beta_3 Age + \beta_4 lbph + \beta_5 svi + \beta_6 lcp + \beta_7 gleason + \beta_8 pgg45 + \epsilon$$

ii. El modelo ajustado resulta:

$$\hat{y} = 0.181 + 0.56 lcavol + 0.622 lweight - 0.02 Age + 0.096 lbph + 0.76 svi - 0.1 lcp + 0.05 gleason + 0.004 pgg45$$

b) Los coeficientes estimados en función de la muestra son:

```
#Intercept
summary(reg)$coeff[1,1]
```

```
## [1] 0.1815608
```

```
#Age
summary(reg)$coeff[4,1]
```

```
## [1] -0.02124819
```

c) Para estimar la varianza, podemos realizar el cálculo utilizando los datos y la siguiente fórmula:

$S^2 = ||y - \hat{y}||^2/(n-p) = \sum r^2/(n-p)$

O podemos encontrar el resultado en la salida del lm.

```
#haciendo la cuenta
s2<-t(reg$residuals)%*%reg$residuals/(length(reg$residuals)-ncol(model.matrix(reg)))
s<-sqrt(s2)
s
```

```
##         [,1]
## [1,] 0.6995
```

```
#usando la salida de lm
S<-sigma(reg)
S
```

```
## [1] 0.6995
```

d) Observando la salida del summary, podemos ver que el p-valor para el test para el coeficiente de la variable age es 0.06 por lo tanto no hay evidencia con un nivel de significación de 0.01 de que el coeficiente sea distinto de cero. (resto del ítem a resolver en el pizarrón)

e) Para hallar un intervalo de confianza de nivel 0.95 para β_0 podemos utilizar los resultados de la salida:

$IC = [\hat{\beta}_0 - t_{(87,0.975)} * Sb, \hat{\beta}_0 + t_{(87,0.975)} * Sb]$

La estimación por intervalos resulta:

```
IC<-c(reg$coefficients[1]-qt(0.975,87)*summary(reg)$coef[1,2],
      reg$coefficients[1]+qt(0.975,87)*summary(reg)$coef[1,2])
names(IC)<-c("LI","LS")
IC
```

```
##        LI        LS
## -2.443211  2.806333
```

Como contiene al cero, concluyo nuevamente que no tengo evidencias de que el coeficiente correspondiente al intercept sea distinto de cero, con un nivel de significación de 0.05.

f) El valor de R^2 lo podemos observar en la salida, dando un valor de 0.6634. Esto se interpreta como que el 66.34% de la variabilidad de lpsa es explicada por el modelo utilizando todas las covariables.

g) A resolver en el pizarrón

h) Mirando la salida, la estimación del coeficiente asociado a la variable lcavol da un valor de

```
summary(reg)$coeff[2,1]
```

```
## [1] 0.5643413
```

con un p-valor (para las hipótesis bilaterales) de

```
summary(reg)$coeff[2,4]
```

```
## [1] 6.548216e-09
```

Para el caso unilateral, el p-valor será la mitad.

i) Resolución similar al ítem e)

j) Observando la salida, las variables más relevantes parecen ser lcavol, lweight y svi.

Ejercicio 2

Guardamos en el objeto "cargamento" los datos del archivo .csv

```
cargamento<-read.csv("glakes.csv")
```

En este ejercicio debemos usar el método "leave-one-out cross validation", que consiste en realizar el ajuste por cuadrados mínimos utilizando todas las observaciones menos una, y luego midiendo el error de predicción sobre la observación que no se utilizó para armar el modelo ajustado.

Sigamos los pasos para el primer modelo propuesto:

```
#modelo sin intercept
w<-c()
for( i in 1:nrow(cargamento))
{
  datos<-cargamento[-i,]
  reg<-lm(Time~Tonnage-1,data=datos)
  predicho<-reg$coefficients[1]*cargamento[i,2]
  w[i]<-(cargamento[i,3]-predicho)^2
}
W1<-sum(w)
```

Usamos el mismo método para los otros 3 modelos, y calculamos W2, W3 y W4 para luego compararlos y elegir el modelo con menor W

```
#modelo con intercept

w<-c()
for( i in 1:nrow(cargamento))
{
  datos<-cargamento[-i,]
  reg<-lm(Time~Tonnage,data=datos)
  predicho<-reg$coefficients[1]+reg$coefficients[2]*cargamento[i,2]
  w[i]<-(cargamento[i,3]-predicho)^2
}
W2<-sum(w)

#modelo 3

w<-c()
for( i in 1:nrow(cargamento))
{
  datos<-cargamento[-i,]
  y<-log(datos[,3])
  x<-datos[,2]^(0.25)
  reg<-lm(y~x)
  predicho<-exp(reg$coefficients[1]+
                  reg$coefficients[2]*(cargamento[i,2])^(0.25))
  w[i]<-(cargamento[i,3]-predicho)^2
}
W3<-sum(w)
```

```r
#modelo 4

w<-c()
for( i in 1:nrow(cargamento))
{
  datos<-cargamento[-i,]
  y<-log(datos[,3])
  x1<-datos[,2]^(0.5)
  x2<-datos[,2]^(0.25)
  reg<-lm(y~x1+x2)
  predicho<-exp(reg$coefficients[1]+
                  reg$coefficients[2]*(cargamento[i,2])^(0.5)+
                  reg$coefficients[3]*(cargamento[i,2])^(0.25))
  w[i]<-(cargamento[i,3]-predicho)^2
}
W4<-sum(w)

W1;W2;W3;W4
```

```
## [1] 6583.148
```

```
## [1] 4375.183
```

```
## [1] 4105.534
```

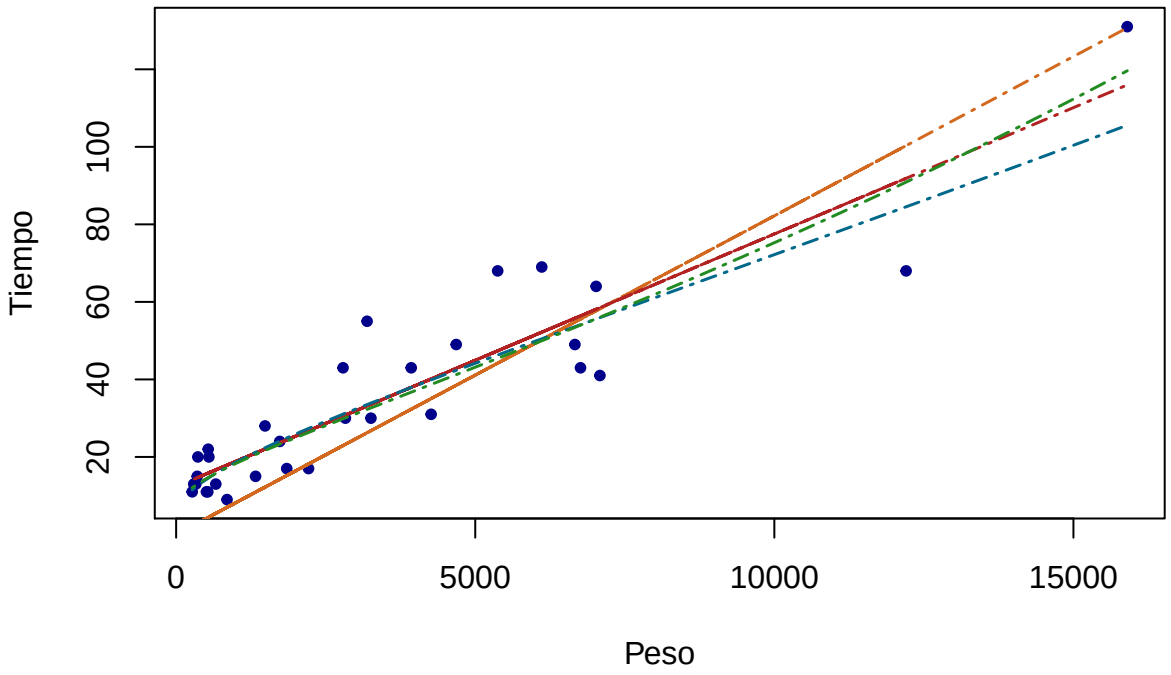
```
## [1] 4559.155
```

Comparando los valores de W, el mejor modelo es el 3. Graficamos los datos y los 4 ajustes para poder ver lo que pasa:

```r
reg1<-lm(Time~Tonnage-1,data=cargamento)
reg2<-lm(Time~Tonnage,data=cargamento)
y<-log(cargamento[,3])
x1<-cargamento[,2]^(0.5)
x2<-cargamento[,2]^(0.25)
reg3<-lm(y~x2)
x<-seq(min(cargamento$Tonnage),max(cargamento$Tonnage),length=100)
predicho3<-exp(reg3$coefficients[1]+
                reg3$coefficients[2]*x^(0.25))
reg4<-lm(y~x1+x2)
predicho4<-exp(reg4$coefficients[1]+reg4$coefficients[2]*x^(0.5)+
                reg4$coefficients[3]*x^(0.25))

plot(cargamento$Tonnage,cargamento$Time,pch=20,col="darkblue",
     xlab= "Peso", ylab="Tiempo")

lines(cargamento$Tonnage,reg1$fitted.values,col="chocolate",lty=6,lwd=1.5)
lines(cargamento$Tonnage,reg2$fitted.values,col="firebrick",lty=6,lwd=1.5)
lines(x,predicho3,col="deepskyblue4",lty=6,lwd=1.5)
lines(x,predicho4,col="forestgreen",lty=6,lwd=1.5)
```

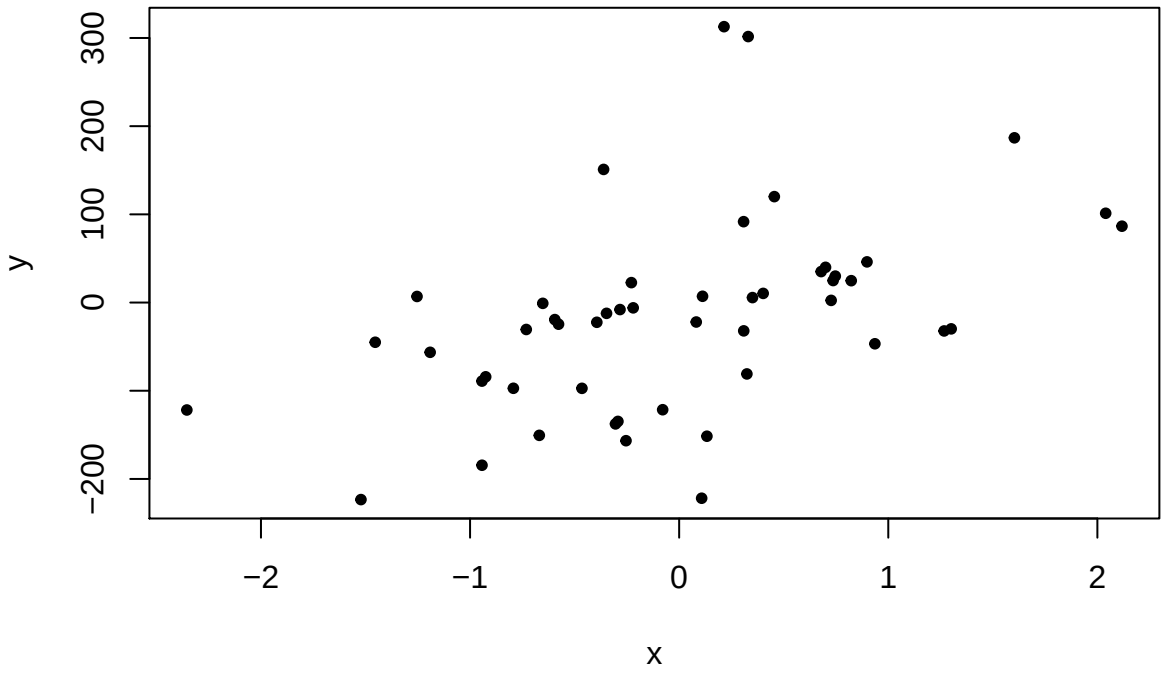


Ejercicio 3

Errores heterocedásticos

a) Generamos una muestra de variables y de manera tal que $y_i = 2i + \epsilon_i$ con ϵ_i una variable con distribución normal de media 0 y varianza $9i^2$

```
n<-50
eps<-c()
x<-rnorm(n)
for(i in 1:n)
{
  eps[i]<-rnorm(1,0,3*i)
}
y<-1+50*x+eps
plot(x,y,pch=20)
```



Ahora ajustamos los dos modelos, M1 como si no nos diéramos cuenta del modelo heterocedástico, y M2 corrigiéndo el problema usando pesos w

```
tt<-1:n
w<-1/tt^2
M1<-lm(y~x)
M2<-lm(y~x,weights = w)
```

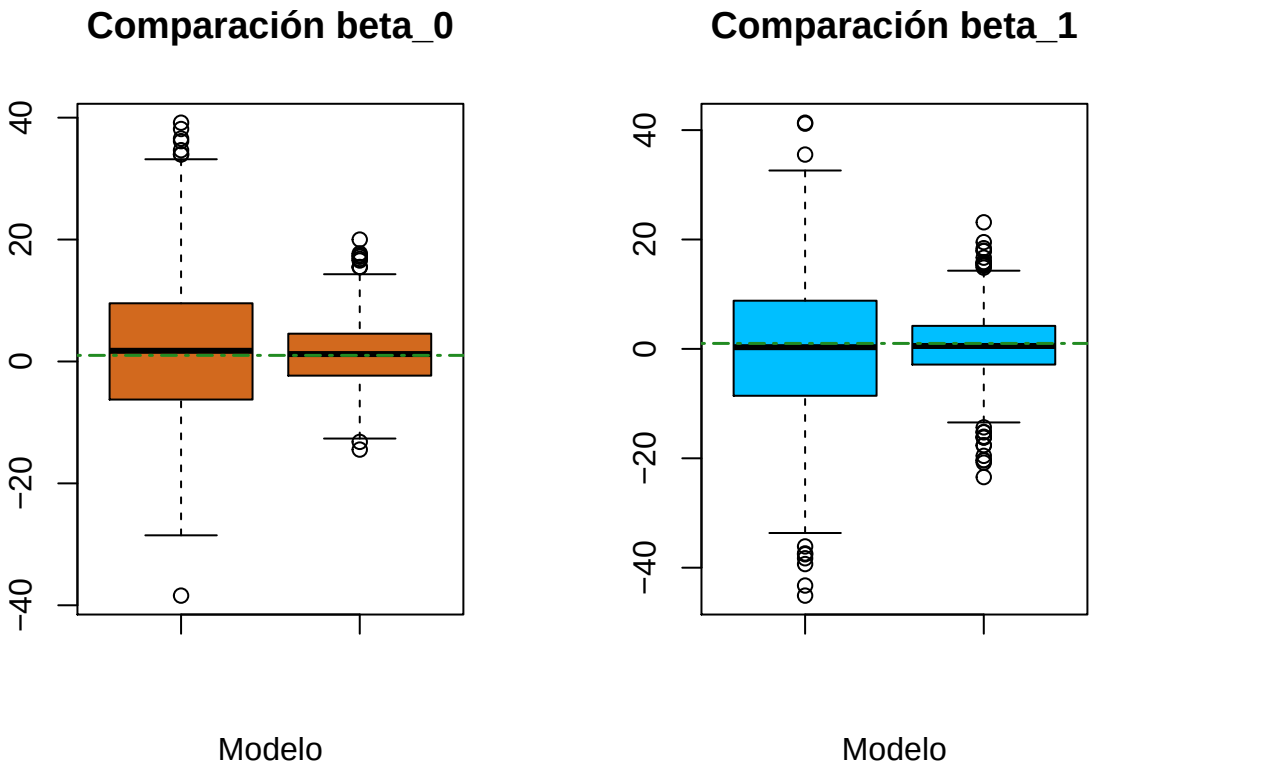
Por último debemos repetir las estimaciones 1000 veces más para 1000 simulaciones diferentes de la muestra, y luego realizar box-plot para las estimaciones de los dos modelos y comparar con el valor real.

```
Nrep<-1000
beta01<-c()
beta02<-c()
beta11<-c()
beta12<-c()

for(i in 1:Nrep)
{
  n<-50
  eps<-c()
  x<-rnorm(n)
  for(j in 1:n)
  {
    eps[j]<-rnorm(1,0,3*j)
  }
  y<-1+x+eps
  tt<-1:n
  w<-1/tt
  M1<-lm(y~x)
  M2<-lm(y~x,weights = w)
  beta01[i]<-M1$coefficients[1]
  beta02[i]<-M2$coefficients[1]
  beta11[i]<-M1$coefficients[2]
```

```
  beta12[i]<-M2$coefficients[2]
}

par(mfrow=c(1,2))
boxplot(beta01,beta02, main="Comparación beta_0", xlab="Modelo",col="chocolate")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
boxplot(beta11,beta12, main="Comparación beta_1", xlab="Modelo",col="deepskyblue")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
```




```
par(mfrow=c(1,1))
```

Sesgo por variables omitidas

a) Generamos los datos de acuerdo con el enunciado:

```
n<-50
x1<-rnorm(n)
x2<-rnorm(n)
eps<-rnorm(n,0,0.25)
y<-1+x1+10*x2+eps
```

Ahora realizamos los dos ajustes pedidos

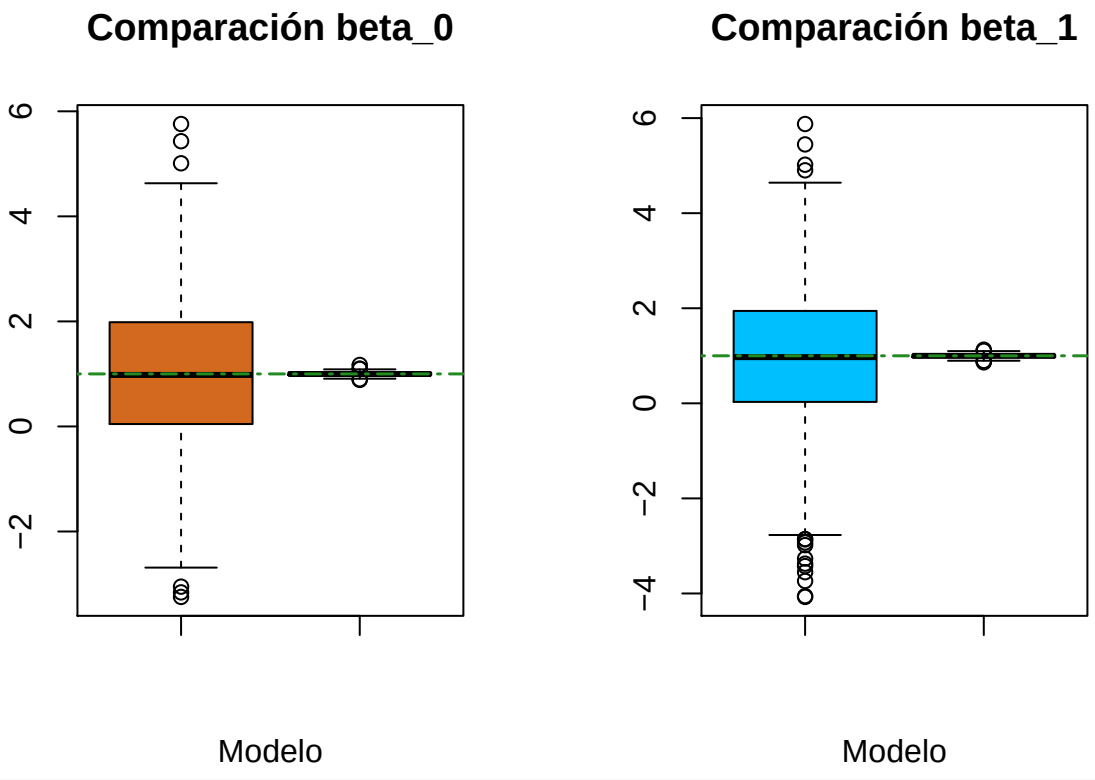
```
M1<-lm(y~x1)
M2<-lm(y~x1+x2)
```

De esta forma conseguimos estimaciones para β_0 y β_1 en ambos modelos. Debemos repetir las estimaciones 999 veces más para 1000 simulaciones diferentes de la muestra, y luego realizar box-plot para las estimaciones de los dos modelos y comparar con el valor real.

```r
Nrep<-1000
beta01<-c()
beta02<-c()
beta11<-c()
beta12<-c()

for(i in 1:Nrep)
{
  n<-50
  x1<-rnorm(n)
  x2<-rnorm(n)
  eps<-rnorm(n,0,0.25)
  y<-1+x1+10*x2+eps
  M1<-lm(y~x1)
  M2<-lm(y~x1+x2)
  beta01[i]<-M1$coefficients[1]
  beta02[i]<-M2$coefficients[1]
  beta11[i]<-M1$coefficients[2]
  beta12[i]<-M2$coefficients[2]
}

par(mfrow=c(1,2))
boxplot(beta01,beta02, main="Comparación beta_0", xlab="Modelo",col="chocolate")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
boxplot(beta11,beta12, main="Comparación beta_1", xlab="Modelo",col="deepskyblue")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
```




```r
par(mfrow=c(1,1))
```

b) Generamos nuevos datos cambiando unicamente el valor de β_2:

```
n<-50
x1<-rnorm(n)
x2<-rnorm(n)
eps<-rnorm(n,0,0.25)
y<-1+x1+0.1*x2+eps
```

Ahora realizamos los dos ajustes pedidos

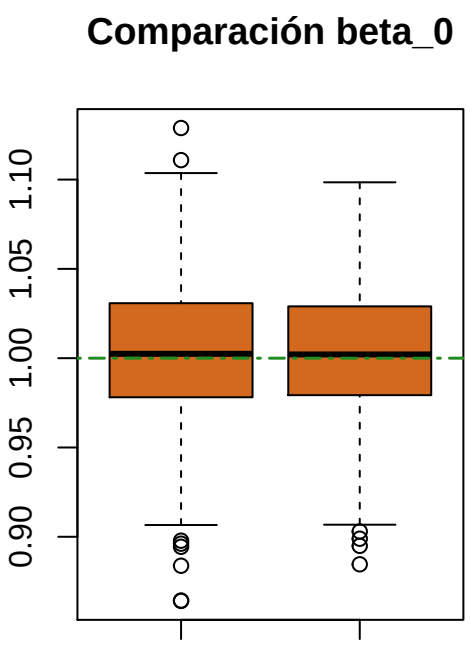
```
M1<-lm(y~x1)
M2<-lm(y~x1+x2)
```

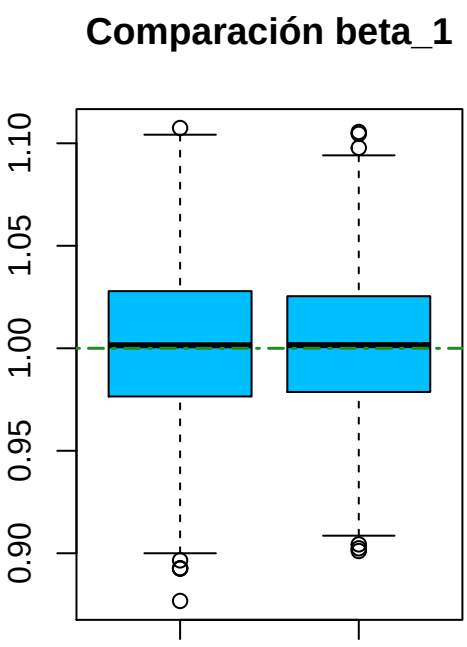
De esta forma conseguimos estimaciones para β_0 y β_1 en ambos modelos. Debemos repetir las estimaciones 1000 veces más para 1000 simulaciones diferentes de la muestra, y luego realizar box-plot para las estimaciones de los dos modelos y comparar con el valor real.

```
Nrep<-1000
beta01<-c()
beta02<-c()
beta11<-c()
beta12<-c()

for(i in 1:Nrep)
{
  n<-50
  x1<-rnorm(n)
  x2<-rnorm(n)
  eps<-rnorm(n,0,0.25)
  y<-1+x1+0.1*x2+eps
  M1<-lm(y~x1)
  M2<-lm(y~x1+x2)
  beta01[i]<-M1$coefficients[1]
  beta02[i]<-M2$coefficients[1]
  beta11[i]<-M1$coefficients[2]
  beta12[i]<-M2$coefficients[2]
}

par(mfrow=c(1,2))
boxplot(beta01,beta02, main="Comparación beta_0", xlab="Modelo",col="chocolate")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
boxplot(beta11,beta12, main="Comparación beta_1", xlab="Modelo",col="deepskyblue")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
```






```
par(mfrow=c(1,1))
```

c) Ahora el cambio es que las covariables están correlacionadas, x2 es función de x1.

Para el primer modelo

```
n<-50
x1<-rnorm(n)
u<-rnorm(n,0,0.5)
x2<-0.5*x1+u
eps<-rnorm(n,0,0.25)
y<-1+x1+10*x2+eps
```

Realizamos los dos ajustes pedidos

```
M1<-lm(y~x1)
M2<-lm(y~x1+x2)
```

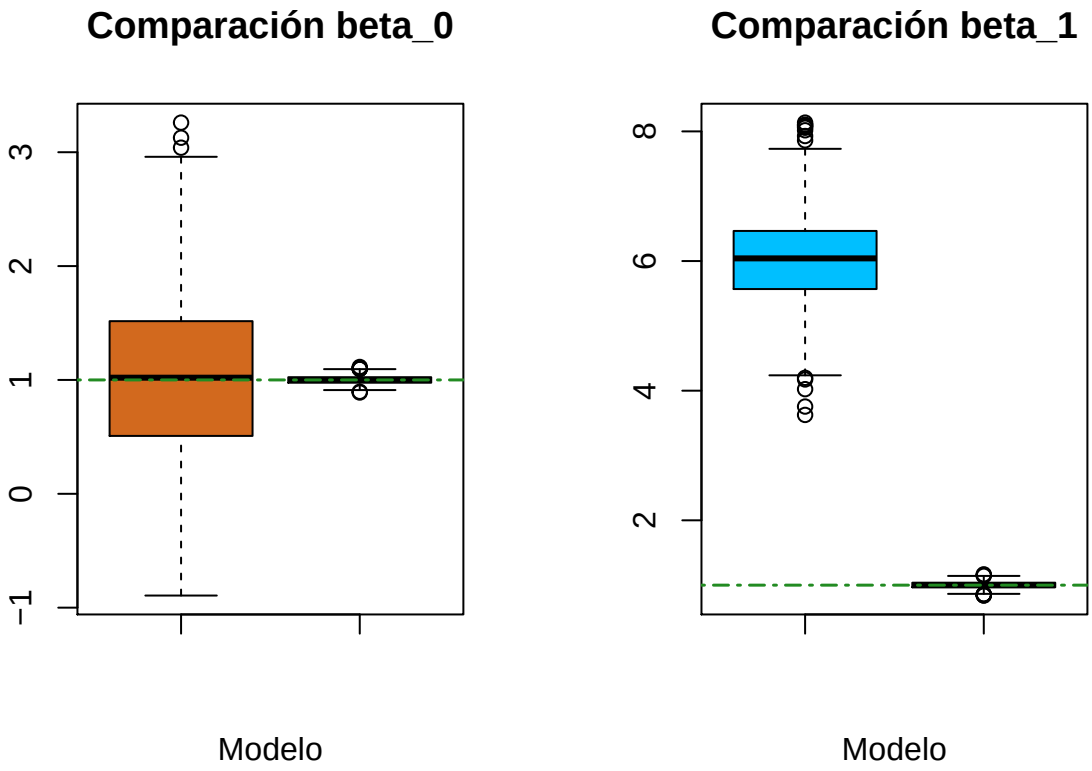
Simulamos 1000 muestras

```
Nrep<-1000
beta01<-c()
beta02<-c()
beta11<-c()
beta12<-c()

for(i in 1:Nrep)
{
  n<-50
  x1<-rnorm(n)
```

```
  u<-rnorm(n,0,0.5)
  x2<-0.5*x1+u
  eps<-rnorm(n,0,0.25)
  y<-1+x1+10*x2+eps
  M1<-lm(y~x1)
  M2<-lm(y~x1+x2)
  beta01[i]<-M1$coefficients[1]
  beta02[i]<-M2$coefficients[1]
  beta11[i]<-M1$coefficients[2]
  beta12[i]<-M2$coefficients[2]
}

par(mfrow=c(1,2))
boxplot(beta01,beta02, main="Comparación beta_0", xlab="Modelo",col="chocolate")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
boxplot(beta11,beta12, main="Comparación beta_1", xlab="Modelo",col="deepskyblue")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
```




```
par(mfrow=c(1,1))
```

Para el segundo modelo

```
n<-50
x1<-rnorm(n)
u<-rnorm(n,0,0.5)
x2<-0.5*x1+u
eps<-rnorm(n,0,0.25)
y<-1+x1+0.1*x2+eps
```

Realizamos los dos ajustes pedidos

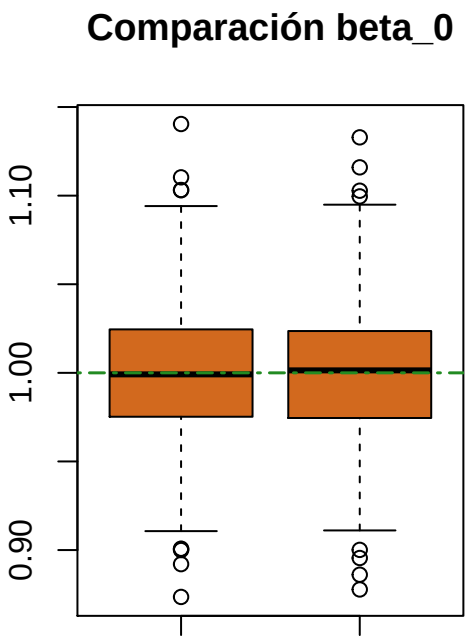
```
M1<-lm(y~x1)
M2<-lm(y~x1+x2)
```

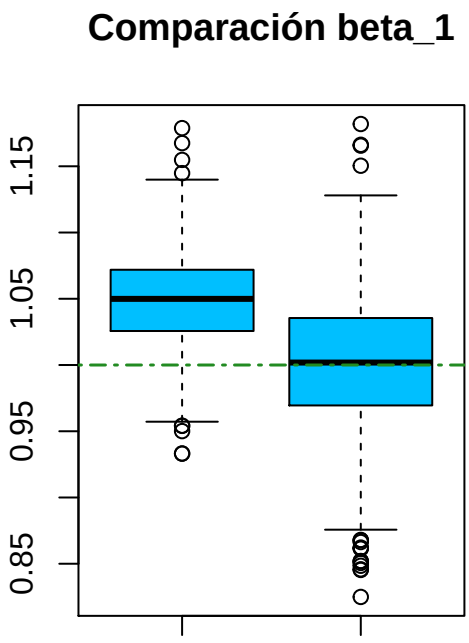
Simulamos 1000 muestras

```
Nrep<-1000
beta01<-c()
beta02<-c()
beta11<-c()
beta12<-c()

for(i in 1:Nrep)
{
  n<-50
  x1<-rnorm(n)
  u<-rnorm(n,0,0.5)
  x2<-0.5*x1+u
  eps<-rnorm(n,0,0.25)
  y<-1+x1+0.1*x2+eps
  M1<-lm(y~x1)
  M2<-lm(y~x1+x2)
  beta01[i]<-M1$coefficients[1]
  beta02[i]<-M2$coefficients[1]
  beta11[i]<-M1$coefficients[2]
  beta12[i]<-M2$coefficients[2]
}

par(mfrow=c(1,2))
boxplot(beta01,beta02, main="Comparación beta_0", xlab="Modelo",col="chocolate")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
boxplot(beta11,beta12, main="Comparación beta_1", xlab="Modelo",col="deepskyblue")
abline(h=1,col="forestgreen",lwd=1.5,lty=6)
```






```
par(mfrow=c(1,1))
```

Ejercicio 4

Vamos a realizar el paso 1, esto es: separar la muestra de 97 observaciones en dos subconjuntos elegidos al azar: Entrenamiento y testeo. El subconjunto de entrenamiento estará formado por 2/3 de la muestra total, seleccionada al azar. El resto formará parte del subconjunto de testeo. Para poder separar al azar, usaremos la función **Sample**

```
set.seed(27)
datos<-read.table("prostata.txt", header=TRUE)
train<-sample(1:97,64,replace = FALSE)
entrenamiento<-datos[train,]
testeo<-datos[-train,]
k<-nrow(testeo)
n<-nrow(datos)
```

Para el paso 2, como debemos hallar el valor de W para 8 modelos diferentes, vamos a crear una función.

```
W<-function(datos,train)#los datos deben tener la variable y en la primer col
{
  x<-as.matrix(datos[train,2:ncol(datos)])
  y<-datos[train,1]
  reg<-lm(y~x)
  x_test<-datos[-train,2:ncol(datos)]
  y_test<-datos[-train,1]
  unos<-rep(1,length(y_test))
  X<-as.matrix(cbind(unos,x_test))
  y_sombrero<-reg$coefficients%*%t(X)
  w<-sum((y_test-y_sombrero)^2)
```

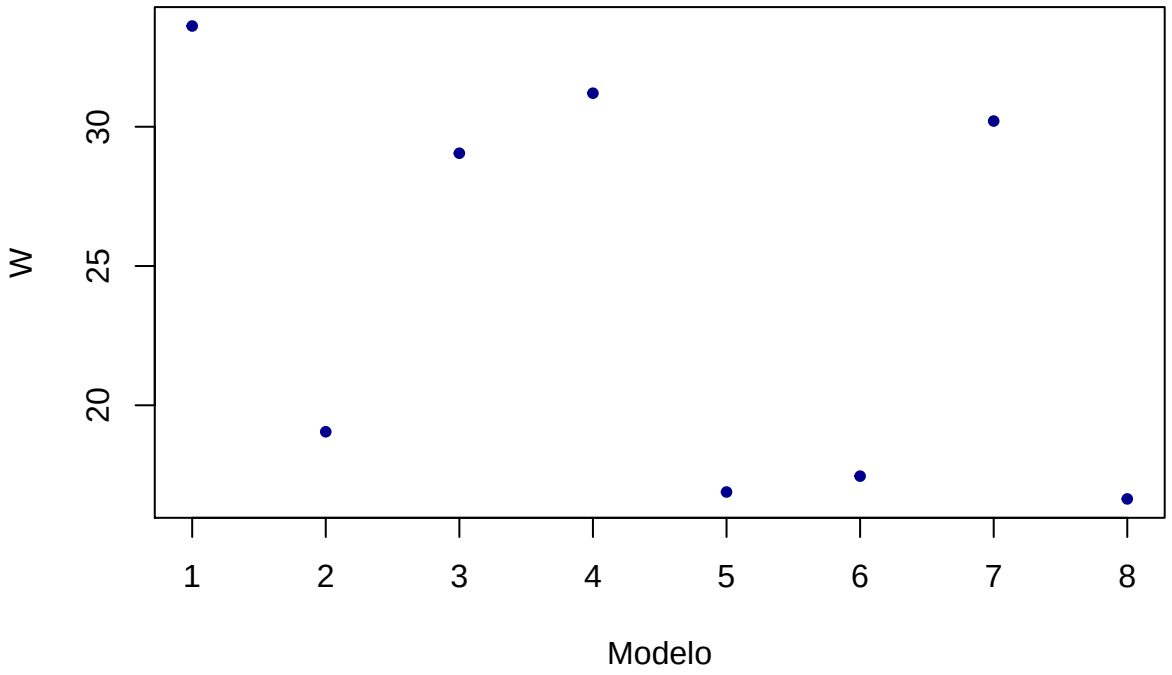
```
  return(w)
}
```

Y ahora lo calculamos para los 8 modelos

```
datos2<-datos[,c(1,2,4,9)]
names(datos2)<-c("x1","x2","x3","y")

WW<-c(sum((datos2[-train,4]-mean(datos2[train,4]))^2),
      W(cbind(datos2[,4],datos2[,1]),train),
      W(cbind(datos2[,4],datos2[,2]),train),
      W(cbind(datos2[,4],datos2[,3]),train),
      W(cbind(datos2[,4],datos2[,1:2]),train),
      W(cbind(datos2[,4],datos2[,c(1,3)]),train),
      W(cbind(datos2[,4],datos2[,2:3]),train),
      W(cbind(datos2[,4],datos2[,1:3]),train))

plot(1:8,WW,pch=20, col="darkblue", xlab = "Modelo", ylab = "W")
```




```
which.min(WW)
```

```
## [1] 8
```