PEC2_RMM

Amelia Martínez Sequera

Junio 2020

Ejercicio 1.

(a)Estudia la posible multicolinealidad.

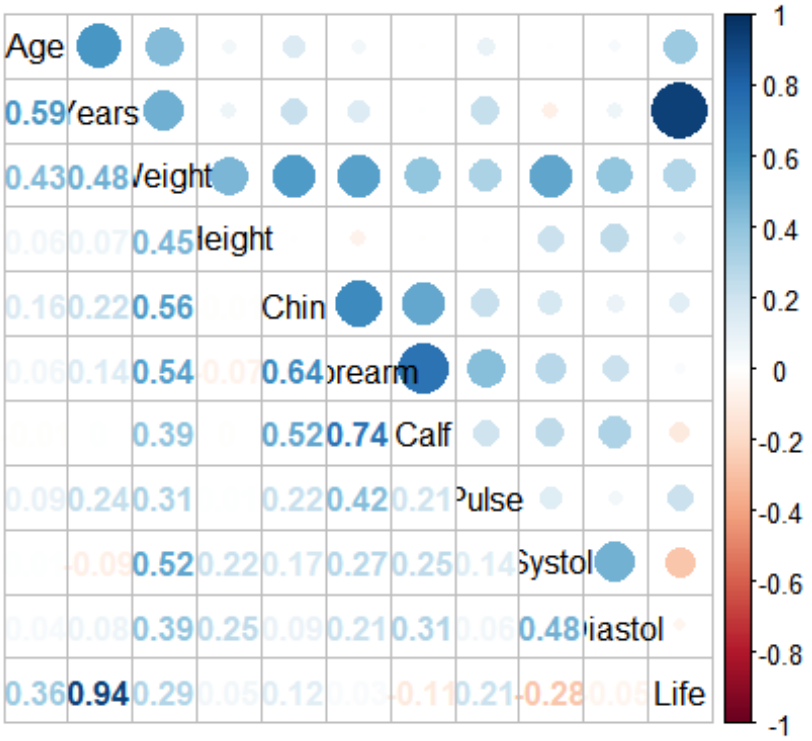
```
peru2<- dplyr::mutate(peru, Life=Years/Age)
lmod<- lm(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
data = peru2)
summary(lmod)
```

```
##
## Call:
## lm(formula = Systol ~ Age + Years + Life + Weight + Height +
##     Chin + Forearm + Calf + Pulse, data = peru2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.3442  -6.3972   0.0507   5.7292  14.5257
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  146.81907   48.97096   2.998 0.005526 **
## Age           -1.12144    0.32741  -3.425 0.001855 **
## Years          2.45538    0.81458   3.014 0.005306 **
## Life        -115.29395   30.16900  -3.822 0.000648 ***
## Weight         1.41393    0.43097   3.281 0.002697 **
## Height        -0.03464    0.03686  -0.940 0.355194
## Chin          -0.94369    0.74097  -1.274 0.212923
## Forearm       -1.17085    1.19329  -0.981 0.334612
## Calf          -0.15867    0.53716  -0.295 0.769810
## Pulse          0.11455    0.17043   0.672 0.506818
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.655 on 29 degrees of freedom
## Multiple R-squared:  0.6674, Adjusted R-squared:  0.5641
## F-statistic: 6.465 on 9 and 29 DF,  p-value: 5.241e-05
```

Los predictores más significativos son Age, Years Life y Wight. Hay bastante diferencia entre el R2 y el R2 ajustado

Correlaciones:

```
corrplot.mixed(cor(peru2), tl.col="black")
```



Hay algunas correlaciones altas.

Descomposición en eigen valores, sin incluir el término de intercepción:

```
X<- model.matrix(lmod)[,-1]
e<-  eigen(t(X)%*%X)
e$value

## [1] 9.774650e+07 6.059674e+03 3.264507e+03 1.358492e+03 1.140880e+03
## [6] 3.974050e+02 1.410325e+02 4.889583e+01 8.313858e-02

sqrt(e$values[1]/e$values)

## [1]     1.0000   127.0065   173.0381   268.2391   292.7053   495.9455
832.5129
## [8]  1413.8870 34288.5648
```

Hay un amplio rango de eigen valores, y los números de condición son muy altos, mucho mayores que 30. Es evidente que hay más de una combinación lineal.

VIFs (factores de inflación de la varianza):

```
summary(lm(X[,1] ~ X[,-1]))$r.squared

## [1] 0.6888004

1 / (1-0.6888)
```

```
## [1] 3.213368

vif(lmod)

##       Age     Years      Life    Weight    Height      Chin   Forearm
Calf
##  3.213372 34.289194 24.387468  4.747711  1.913991  2.063866  3.802313
2.414602
##     Pulse
##  1.329233
```

El VIF para predictores ortogonales es 1(<< 3.2). Existe mucha inflación de la varianza. Hay 3 vifs superiores a 4. Existe gran multicolinealidad.

(b)Eliminar una única observación de la muestra para que el modelo mejore notablemente.

Podría haber observaciones inusuales. Vamos a estudiar las que tienen un alto leverage:

```
k<-9
n<- length(peru2$Systol)
hat<- hatvalues(lmod)
which(hat>2*(k+1)/n)

##  5  8 38 39
##  5  8 38 39

head(sort(hat,decreasing = T))

##        39        38         8         5         1         4
## 0.6220632 0.5951000 0.5517533 0.5178689 0.4388073 0.3671580
```

Las observaciones 38 y 39 son las que tienen el leverage más alto. Recalculamos el modelo sin la observación 39:

```
lmod2<-lm(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
data = peru2[-39,])
X<- model.matrix(lmod2)[,-1]
E<-eigen(t(X)%*%X)
E$values

## [1] 9.535011e+07 4.097361e+03 3.120505e+03 1.211009e+03 1.138203e+03
## [6] 3.457315e+02 1.409287e+02 4.217127e+01 7.967207e-02

sqrt(E$values[1]/E$values)

## [1]     1.0000   152.5487   174.8027   280.5995   289.4348   525.1594
822.5474
## [8]  1503.6690 34594.5388

summary(lm(X[,1] ~ X[,-1]))$r.squared
```

```
## [1] 0.6409072
```

```
1 / (1-0.641)
```

```
## [1] 2.785515
```

```
vif(lmod2)
```

```
##       Age     Years      Life    Weight    Height      Chin   Forearm
Calf
##  2.784795 30.140790 24.145238  3.778450  2.186369  1.876244  3.299083
2.858594
##     Pulse
##  1.172357
```

No se soluciona el problema de multicolinealidad, aunque ha mejorado un poco. Solo Years y Life tienen vifs superiores a 4, lo que no sorprende ya que sabemos que uno es una combinación lineal de otro.

También podemos calcular los outliers y los puntos influyentes.

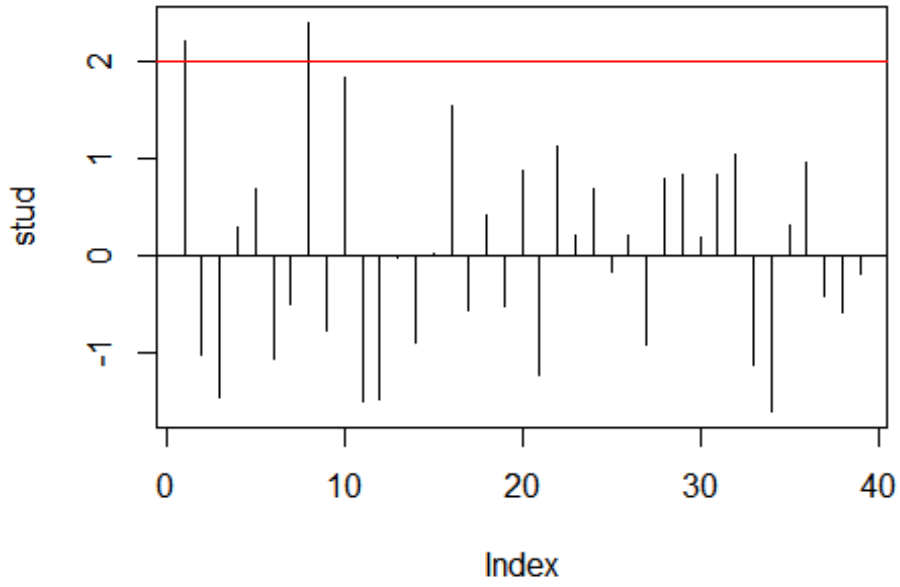
```
stud <- rstudent(lmod)
head(sort(stud, decreasing=TRUE)) #valor studentizado de los residuos
```

```
##        8        1       10       16       22       32
## 2.395715 2.214292 1.839835 1.532232 1.122302 1.027278
```

```
#Aplicando la corrección de Bonferroni para comparaciones múltiples el
valor crítico será:
gll <- length(lmod$fitted.values)-(length(lmod$coefficients))-1
abs(qt(0.05/(length(lmod$fitted.values)*2),gll))
```

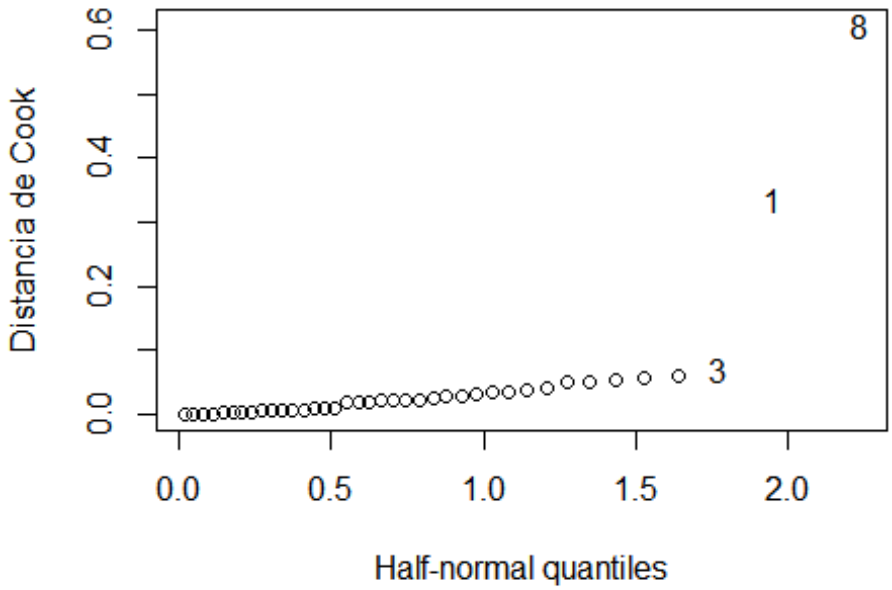
```
## [1] 3.579253
```

```
plot(stud, type="h")
abline(h=-2, col="red"); abline(h=0); abline(h=2, col="red")
```



Observaciones influyentes:

```
cook <- cooks.distance(lmod)
head(sort(cook,decreasing=TRUE))

##          8          1          3         32          6         38
## 0.60723694 0.33790354 0.07105152 0.05869655 0.05771889 0.05230309

halfnorm(cook,nlab=3, ylab="Distancia de Cook")
```




```
outlierTest(lmod)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##   rstudent unadjusted p-value Bonferroni p
## 8 2.395715           0.023515      0.91709
```

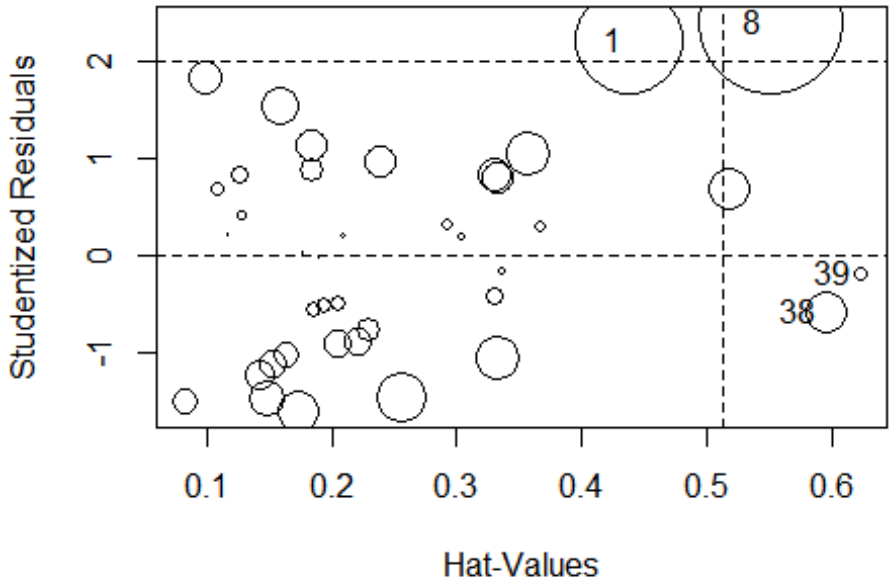
```
4/(length(lmod$fitted.values)-length(coefficients(lmod))-2)
```

```
## [1] 0.1481481
```

```
length(cook[cook>(4/(length(lmod$fitted.values)-
length(coefficients(lmod))-2))])
```

```
## [1] 2
```

```
influencePlot(lmod)
```




```
##       StudRes       Hat       CookD
## 1   2.2142915 0.4388073 0.337903541
## 8   2.3957154 0.5517533 0.607236944
## 38 -0.5897986 0.5951000 0.052303089
## 39 -0.1958988 0.6220632 0.006533174
```

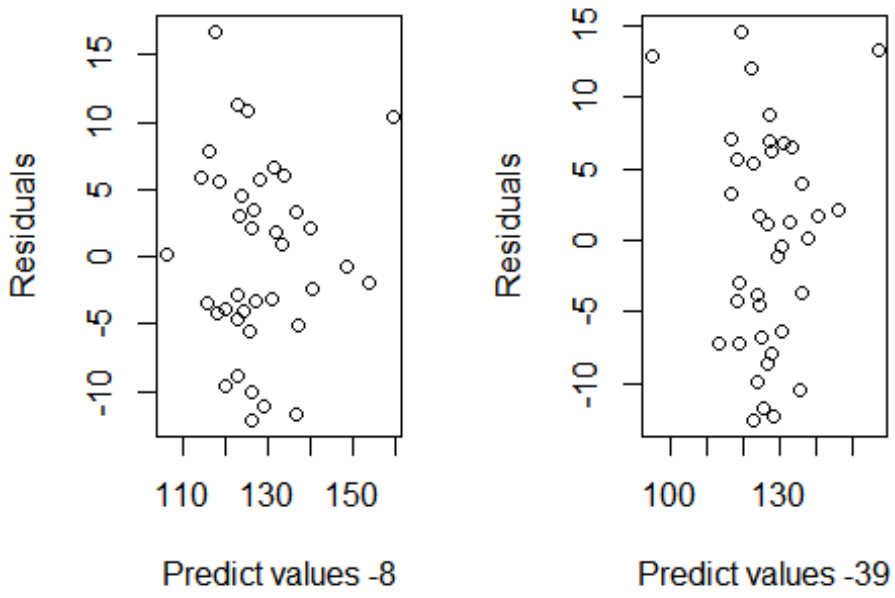
Recalculamos el modelo sin la observación 8:

```
lmod22<-lm(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
data = peru2[-8,])
X<- model.matrix(lmod22)[,-1]
E<-eigen(t(X)%*%X)
E$values

## [1] 9.527768e+07 6.045489e+03 3.227182e+03 1.200416e+03 9.063131e+02
## [6] 3.959810e+02 1.342871e+02 4.181297e+01 7.026280e-02

sqrt(E$values[1]/E$values)

## [1]     1.0000   125.5394   171.8240   281.7278   324.2324   490.5219
842.3225
## [8]  1509.5243 36824.1629

summary(lm(X[,1] ~ X[,-1]))$r.squared

## [1] 0.6808899

1 / (1-0.68)
```

```
## [1] 3.125

vif(lmod22)

##       Age     Years      Life    Weight    Height      Chin   Forearm
Calf
##  3.133715 36.685699 25.504326  4.773409  1.983682  2.029157  4.583513
2.588792
##     Pulse
##  1.309554
```

Si los comparamos con los resultados anteriores (omitir la observación 39) se observa que, aunque los VIFs no son mejores, el valor de R2 ajustado es mucho mayor. Se decide prescindir de la observación 8.

```
sum<- summary(lmod)
sum$adj.r.squared

## [1] 0.5641335

summary(lmod2)$adj.r.squared

## [1] 0.5149638

summary(lmod22)$adj.r.squared

## [1] 0.6122649

par(mfrow=c(1,2))
plot(fitted(lmod22),residuals(lmod22),xlab="Predict values -
8",ylab="Residuals")
sumary(lm(sqrt(abs(residuals(lmod22)))~fitted(lmod22)))

##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.2799809  1.6330650  2.0085  0.05214
## fitted(lmod22) -0.0080878  0.0127219 -0.6357  0.52896
##
## n = 38, p = 2, Residual SE = 0.83827, R-Squared = 0.01

plot(fitted(lmod2),residuals(lmod2),xlab="Predict values -
39",ylab="Residuals")
```




```
sumary(lm(sqrt(abs(residuals(lmod2)))~fitted(lmod2)))

##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.139500   1.805844  2.8460 0.007263
## fitted(lmod2) -0.021777   0.014202 -1.5333 0.133944
##
## n = 38, p = 2, Residual SE = 0.86870, R-Squared = 0.06
```

(c)Con los 38 datos restantes, hallar el "mejor" modelo consensuado por dos métodos diferentes de selección de variables como, por ejemplo, R2adj y Cp de Mallows. Identifica el mejor modelo de cada tamaño, entendiendo por mejor modelo aquel que tiene menor RSS.

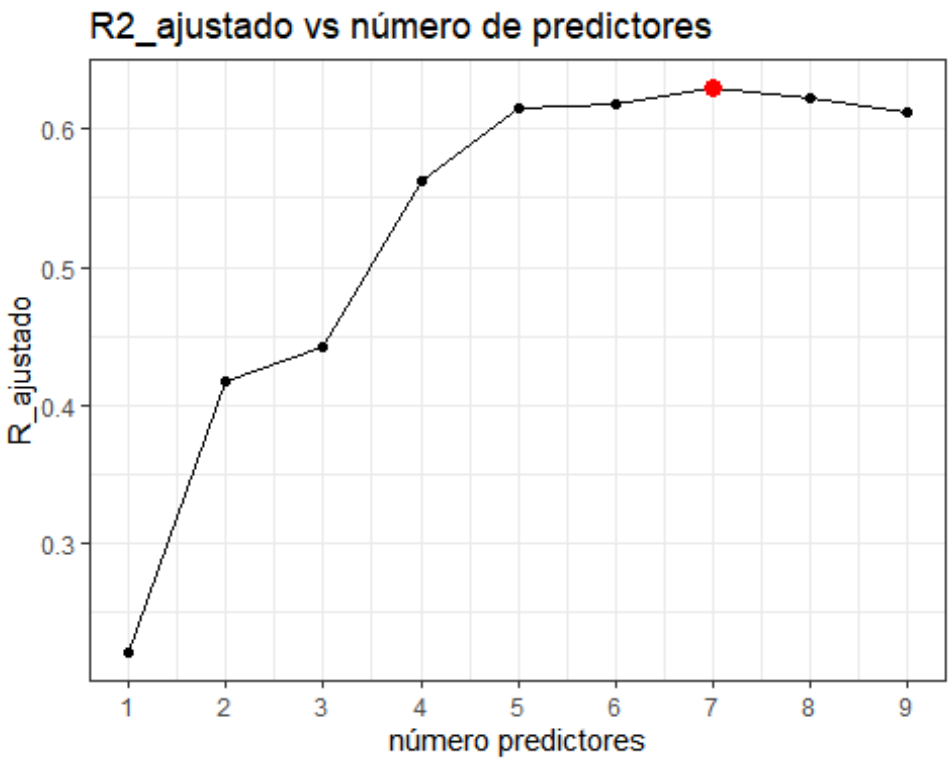
$R^2_{ajustado}$

Tenemos que ver si el número de predictores se aproxima o supera al de observaciones, en tal caso la regresión por mínimos cuadrados no sería adecuada.

```
mejores_modelos <-
regsubsets(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
data = peru2[-8,], nvmax = 9)
rs<- summary(mejores_modelos)
#se identifica que modelo tiene el valor máximo de R ajustado
which.max(rs$adjr2)

## [1] 7
```

El modelo que más se ajusta tiene 7 predictores. Se puede ver gráficamente la evolución de la precisión del modelo en función del tamaño y si la mejora es sustancial:

```
p <- ggplot(data = data.frame(n_predictores = 1:9,
                              R_ajustado =
summary(mejores_modelos)$adjr2),
            aes(x = n_predictores, y = R_ajustado)) +
    geom_line() +
    geom_point()

#Se identifica en rojo el máximo
p <- p + geom_point(aes(
                    x =
n_predictores[which.max(summary(mejores_modelos)$adjr2)],
                    y =
R_ajustado[which.max(summary(mejores_modelos)$adjr2)]),
                    colour = "red", size = 3)
p <- p +  scale_x_continuous(breaks = c(0:9)) +
          theme_bw() +
          labs(title = 'R2_ajustado vs número de predictores',
               x =  'número predictores')
p
```



Para conocer cuáles son los coeficientes del mejor modelo y su estimación:

```
coef(object = mejores_modelos, id = 7)
```

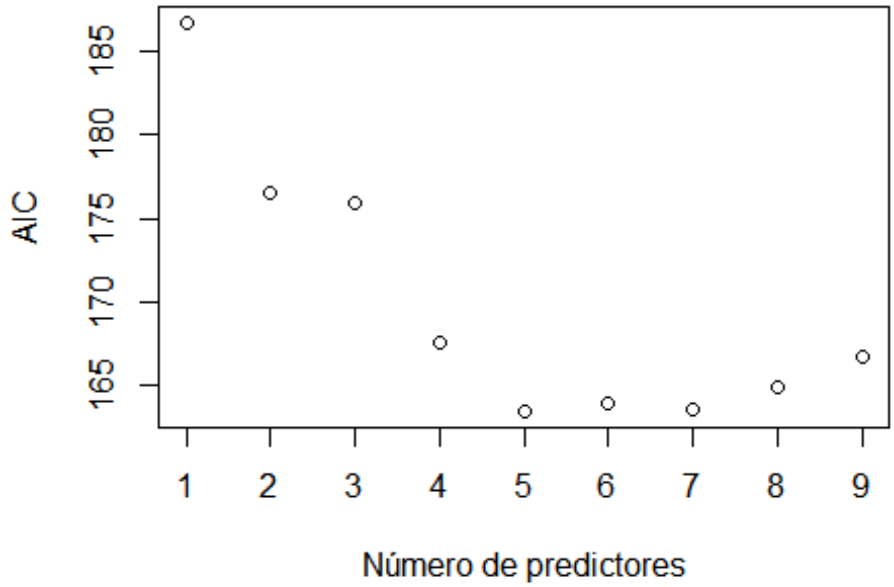
```
##   (Intercept)           Age         Years          Life        Weight
##  167.48929050   -1.20029466    3.02337985 -144.56745503    1.62410963
##        Height         Chin       Forearm
##   -0.04688751   -0.92559381   -1.72542557
```

Si bien el modelo con mayor $R^2_{ajustado}$ es el formado por 7 predictores, la diferencia con la conseguida a partir de 5 o 6 predictores es mínima:

```
rs$adjr2[6] #6

## [1] 0.61816

rs$adjr2[5] #5

## [1] 0.6150372

rs$adjr2[7] #7

## [1] 0.6295699
```

AIC

```
rs$outmat

##          Age Years Life Weight Height Chin Forearm Calf Pulse
## 1  ( 1 ) " " " "   " "  "*"    " "    " "  " "     " "  " "
## 2  ( 1 ) " " " "   "*"  "*"    " "    " "  " "     " "  " "
## 3  ( 1 ) "*" "*"   "*"  " "    " "    " "  " "     " "  " "
## 4  ( 1 ) "*" "*"   "*"  "*"    " "    " "  " "     " "  " "
## 5  ( 1 ) "*" "*"   "*"  "*"    " "    " "  "*"     " "  " "
## 6  ( 1 ) "*" "*"   "*"  "*"    "*"    " "  "*"     " "  " "
## 7  ( 1 ) "*" "*"   "*"  "*"    "*"    "*"  "*"     " "  " "
## 8  ( 1 ) "*" "*"   "*"  "*"    "*"    "*"  "*"     "*"  " "
## 9  ( 1 ) "*" "*"   "*"  "*"    "*"    "*"  "*"     "*"  "*"

n <- 38
k <- length(rs$rss)  # Número de variables predictoras
p <- k + 1           # Número de parámetros (incluye la intercepción)
(AIC <- n*log(rs$rss/n) + (2:p)*2)

## [1] 186.7110 176.5760 175.9015 167.5574 163.4672 163.9512 163.5524
164.9293
## [9] 166.6657

plot(1:k, AIC, ylab="AIC", xlab="Número de predictores", axes=F)
box(); axis(1,at=1:k); axis(2)
```



Los mejores valores de AIC se alcanzan para 5 predictores:

```
rs$outmat[5,]
```

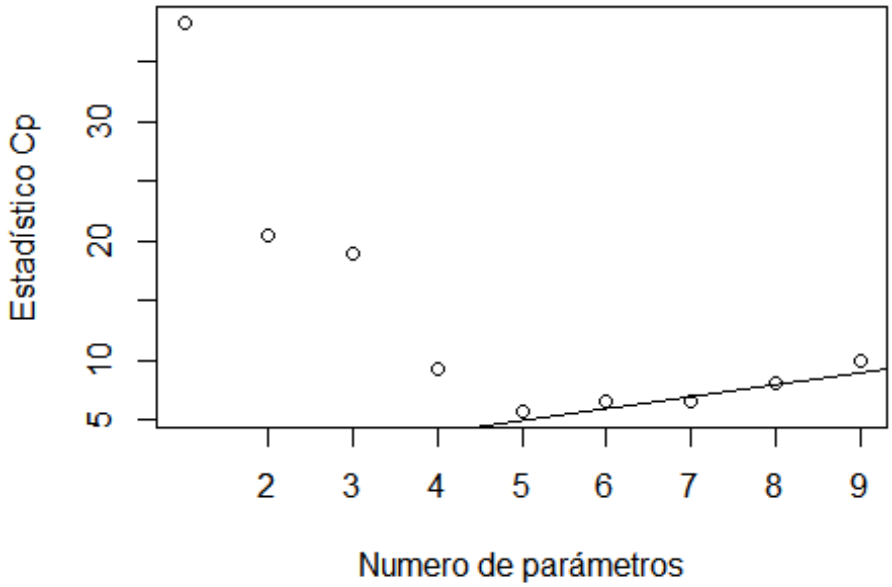
```
##     Age   Years    Life  Weight  Height    Chin Forearm    Calf
Pulse
##     "*"     "*"     "*"     "*"     " "     " "     "*"     " "     "
"
```

Mallows Cp

```
rs$cp
```

```
## [1] 38.295709 20.532028 18.961941  9.293879  5.771204  6.528675
6.661067
## [8]  8.194940 10.000000
```

```
plot(rs$cp, xlab="Numero de parámetros", ylab="Estadístico Cp", axes = F)
box(); axis(1,at=2:10); axis(2)
abline(a=0,b=1)
```



El mínimo Cp se alcanza con 5 predictores(6 parámetros).

Forward and Backward Stepwise Selection.

```
backward <-
regsubsets(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
data = peru2[-8,], nvmax = 9, method = "backward")
# Se identifica el valor máximo de R ajustado
which.max(summary(backward)$adjr2)

## [1] 7

forward<-
regsubsets(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
data = peru2[-8,], nvmax = 9,method = "forward")
which.max(summary(forward)$adjr2)

## [1] 7

coef(object =backward, 7)

##   (Intercept)           Age         Years          Life        Weight
##  167.48929050   -1.20029466    3.02337985 -144.56745503    1.62410963
##        Height          Chin       Forearm
##   -0.04688751   -0.92559381   -1.72542557

coef(object =forward, 7)

##   (Intercept)           Age         Years          Life        Weight
##  167.48929050   -1.20029466    3.02337985 -144.56745503    1.62410963
```

```
##        Height          Chin       Forearm
##   -0.04688751   -0.92559381   -1.72542557
```

Ambos métodos (backward y forward) identifican como mejor modelo el formado por los mismos 7 predictores.

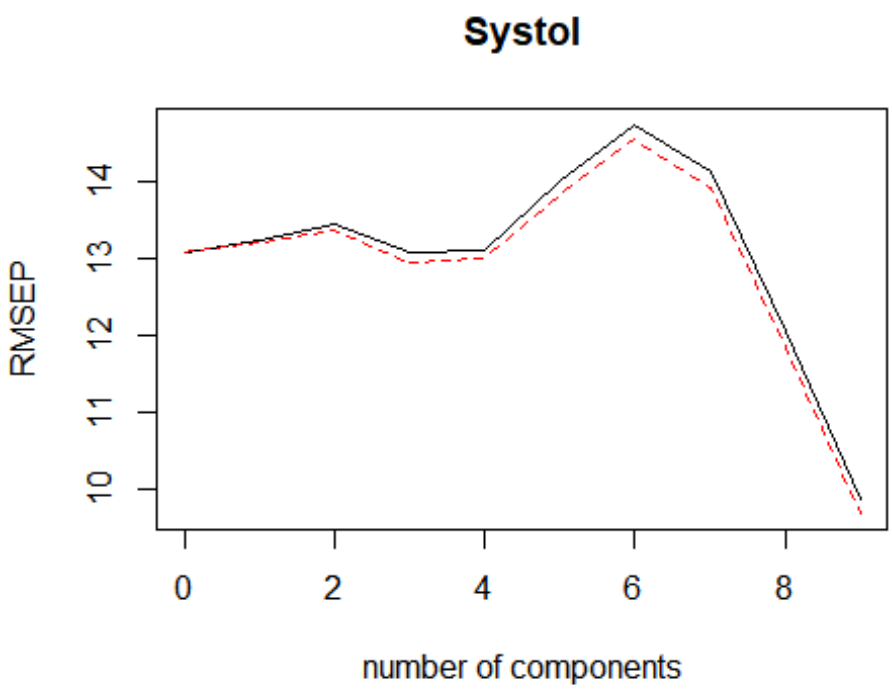
Principal Component regression PCR.

```
peru3<- peru2[-8,]
# Estandarizamos las variables indicándolo con el argumento scale
# Indicando validation = CV, se emplea 10-fold-cross-validation para
# identificar el número óptimo de componentes.
modelo_pcr <-
pcr(Systol~Age+Years+Life+Weight+Height+Chin+Forearm+Calf+Pulse,
                                  data =peru3, scale = TRUE, validation =
"CV")
summary(modelo_pcr)
```

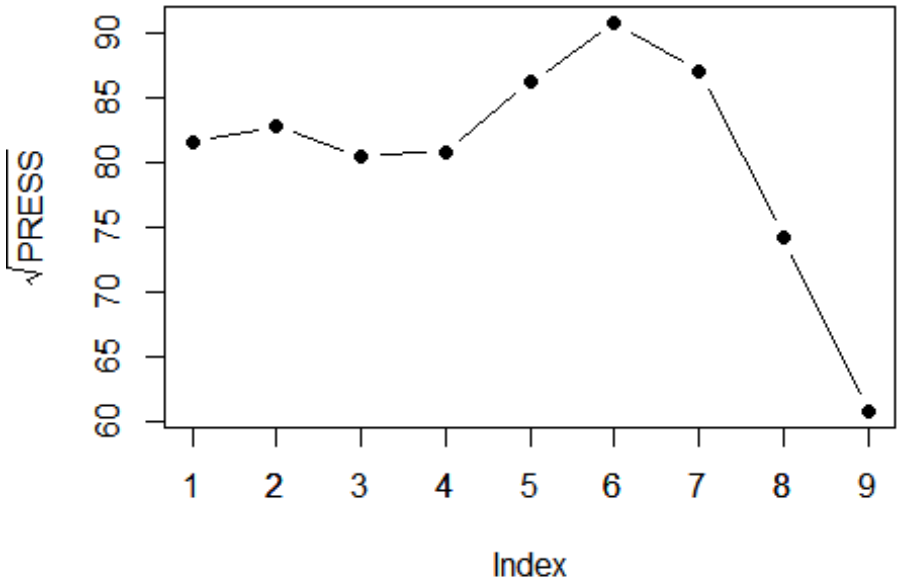
```
## Data:    X dimension: 38 9
##  Y dimension: 38 1
## Fit method: svdpc
## Number of components considered: 9
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6
comps
## CV           13.06    13.24    13.43    13.06     13.1    13.98
14.72
## adjCV        13.06    13.19    13.36    12.95     13.0    13.82
14.53
##        7 comps  8 comps  9 comps
## CV       14.11    12.04    9.869
## adjCV    13.91    11.85    9.691
##
## TRAINING: % variance explained
##         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
8 comps
## X        38.471    62.11    75.01    83.81    90.50    95.71    98.49
99.83
## Systol    4.591    10.86    22.36    23.05    25.51    25.52    37.70
52.92
##         9 comps
## X        100.00
## Systol    70.66
```

El summary del modelo pcr devuelve la estimación del RMSEP (raíz cuadrada del MSE) para cada posible número de componentes introducidas en el modelo. También se muestra el % de varianza explicada acumulada por cada número de componentes. Si se incluye hasta la componente 6, se explica un 95.71% de la varianza observada.

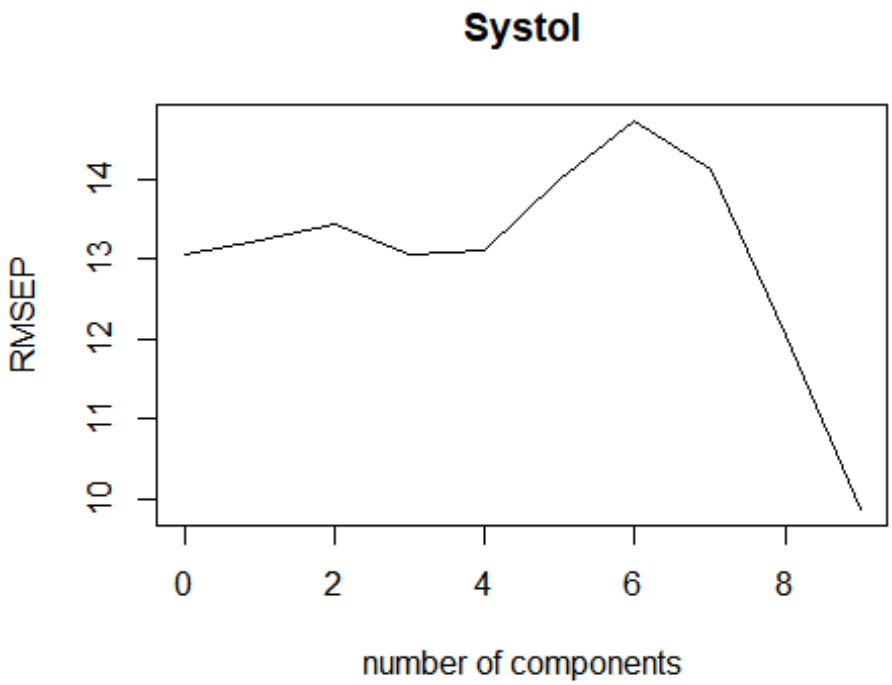
```r
validationplot(modelo_pcr, val.type = "RMSEP")
```




```r
# Vemos con más detalle a partir de la componente 1
# PRESS es el Predicted Sum of Squares
plot(as.numeric(sqrt(modelo_pcr$validation$PRESS)), type = "b", pch = 19,
     ylab = expression(sqrt("PRESS")))
axis(side = 1, at = 1:9)
```




```
#Para conocer el número de componentes con el que se minimiza el error
which.min(x = modelo_pcr$validation$PRESS)

## [1] 9

pcrCV <- RMSEP(modelo_pcr,estimate="CV")
plot(pcrCV)
```




```
which.min(pcrCV$val)
```

```
## [1] 10
```

Utilizando los 9 parámetros obtenemos el mínimo RMSEP. PCR intenta encontrar combinaciones lineales de los predictores que explican la mayor parte de la variación. El propósito es la reducción de dimensiones. Debido a que los componentes principales pueden ser combinaciones lineales de todos los predictores, el número de variables utilizadas no siempre se reduce.

Los componentes principales se seleccionan usando solo la matriz X y no la respuesta, por lo que no hay garantía definitiva de que la PCR predecirá la respuesta particularmente bien, aunque esto sucede a menudo. Por lo tanto, la PCR está más orientada a la explicación que a la predicción (al contrario que PLS).

Stepwise.

Se basa en el método Akaike(AIC), que tiende a ser más restrictivo e introducir menos predictores que el $R^2_{ajustado}$.

```
step(object = lmod22, direction = "both", trace = 1)
```

```
## Start:  AIC=166.67
## Systol ~ Age + Years + Life + Weight + Height + Chin + Forearm +
##     Calf + Pulse
##
##           Df Sum of Sq    RSS    AIC
```

```
## - Pulse    1     12.55 1815.6 164.93
## - Calf     1     31.67 1834.7 165.33
## <none>                 1803.0 166.67
## - Chin     1    122.78 1925.8 167.17
## - Height   1    135.58 1938.6 167.42
## - Forearm  1    247.06 2050.1 169.55
## - Years    1    941.33 2744.3 180.63
## - Age      1    980.58 2783.6 181.17
## - Weight   1   1040.09 2843.1 181.97
## - Life     1   1448.38 3251.4 187.07
##
## Step:  AIC=164.93
## Systol ~ Age + Years + Life + Weight + Height + Chin + Forearm +
##     Calf
##
##           Df Sum of Sq    RSS    AIC
## - Calf     1     30.02 1845.6 163.55
## <none>                 1815.6 164.93
## - Chin     1    133.59 1949.2 165.63
## - Height   1    140.56 1956.1 165.76
## + Pulse    1     12.55 1803.0 166.67
## - Forearm  1    234.62 2050.2 167.55
## - Years    1    933.31 2748.9 178.69
## - Age      1    974.74 2790.3 179.26
## - Weight   1   1085.84 2901.4 180.74
## - Life     1   1437.00 3252.6 185.09
##
## Step:  AIC=163.55
## Systol ~ Age + Years + Life + Weight + Height + Chin + Forearm
##
##           Df Sum of Sq    RSS    AIC
## <none>                 1845.6 163.55
## - Chin     1    120.26 1965.8 163.95
## - Height   1    121.47 1967.0 163.97
## + Calf     1     30.02 1815.6 164.93
## + Pulse    1     10.89 1834.7 165.33
## - Forearm  1    240.10 2085.7 166.20
## - Years    1    940.08 2785.7 177.20
## - Age      1   1003.94 2849.5 178.06
## - Weight   1   1056.96 2902.5 178.76
## - Life     1   1435.20 3280.8 183.41

##
## Call:
## lm(formula = Systol ~ Age + Years + Life + Weight + Height +
##     Chin + Forearm, data = peru2[-8, ])
##
## Coefficients:
## (Intercept)          Age        Years         Life       Weight
Height
```

```
##   167.48929     -1.20029      3.02338   -144.56746      1.62411     -
0.04689
##        Chin      Forearm
##    -0.92559     -1.72543
```

(i) ¿Cuales son las variables seleccionadas? Según los resultados del apartado anterior, 7 variables nos proporcionan un modelo satisfactorio:

```
lmod3<-lm(formula = Systol ~ Age + Years + Life + Weight + Height+Chin+
Forearm,data = peru3)
X3<- model.matrix(lmod3)[,-1]
E3<-eigen(t(X3)%*%X3)
E3$values
```

```
## [1] 9.508866e+07 5.537780e+03 1.184652e+03 9.118585e+02 1.733120e+02
## [6] 7.802195e+01 7.102143e-02
```

```
sqrt(E3$values[1]/E3$values)
```

```
## [1]     1.0000   131.0378   283.3146   322.9242   740.7130  1103.9666
36590.6094
```

```
summary(lm(X3[,1] ~ X3[,-1]))$r.squared
```

```
## [1] 0.6790037
```

```
1 / (1-0.68)
```

```
## [1] 3.125
```

```
vif(lmod3)
```

```
##       Age     Years      Life    Weight    Height      Chin   Forearm
##  3.115301 36.594026 25.396977  4.514642  1.907550  1.972452  2.473013
```

El problema persiste.

(ii)¿Cual es el coeficiente de determinación ajustado de este modelo? Compararlo con el del modelo completo.

```
summary(lmod22)$adj.r.squared
```

```
## [1] 0.6122649
```

```
summary(lmod3)$adj.r.squared
```

```
## [1] 0.6295699
```

El modelo ha mejorado, aunque muy poco.

(iii)¿Se gana en eficiencia con el modelo reducido? Comparar los intervalos de confianza de la estimación del coeficiente de la variable Age.

```
confint(lmod3)[2,]
```

```
##     2.5 %    97.5 %
## -1.807103 -0.593486

confint(lmod22)[2,]

##      2.5 %     97.5 %
## -1.8142681 -0.5652217
```

El intervalo se amplía.

(d)Los investigadores sugieren adoptar el modelo reducido que contenga únicamente las variables significativas(alpha = 0.1) con el test t en sustitución del modelo completo con las 9 variables explicativas.¿Es ese un buen criterio de selección? Realizar un test adecuado que resuelva su sugerencia. Discutir el resultado en consonancia con los resultados obtenidos en el apartado anterior.

```
summary(lmod22)

##
## Call:
## lm(formula = Systol ~ Age + Years + Life + Weight + Height +
##     Chin + Forearm + Calf + Pulse, data = peru2[-8, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.1497  -4.1489  -0.2525   5.2688  16.7433
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  166.35727   46.12801   3.606 0.001194 **
## Age           -1.18974    0.30488  -3.902 0.000546 ***
## Years          3.02918    0.79227   3.823 0.000673 ***
## Life        -145.53620   30.68660  -4.743  5.6e-05 ***
## Weight         1.65662    0.41220   4.019 0.000399 ***
## Height        -0.05052    0.03481  -1.451 0.157876
## Chin          -0.94857    0.68696  -1.381 0.178257
## Forearm       -2.38282    1.21649  -1.959 0.060168 .
## Calf           0.38367    0.54705   0.701 0.488874
## Pulse          0.07024    0.15909   0.442 0.662228
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.025 on 28 degrees of freedom
## Multiple R-squared:  0.7066, Adjusted R-squared:  0.6123
## F-statistic: 7.492 on 9 and 28 DF,  p-value: 1.696e-05

lmod4<- lm(Systol ~ Age+Years + Life + Weight + Forearm, data = peru2[-
8,])
summary(lmod4)
```

```
##
## Call:
## lm(formula = Systol ~ Age + Years + Life + Weight + Forearm,
##     data = peru2[-8, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.162  -5.498   0.333   5.539  15.894
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  116.6799    20.0133   5.830 1.78e-06 ***
## Age           -1.2035     0.3024  -3.980 0.000371 ***
## Years          3.2951     0.7724   4.266 0.000165 ***
## Life        -153.2570    30.1142  -5.089 1.53e-05 ***
## Weight         1.1624     0.2817   4.126 0.000245 ***
## Forearm       -1.7233     0.7307  -2.358 0.024623 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.996 on 32 degrees of freedom
## Multiple R-squared:  0.6671, Adjusted R-squared:  0.615
## F-statistic: 12.82 on 5 and 32 DF,  p-value: 6.991e-07
```

El valor de $R^2_{ajustado}$ no mejora, y el RSE practicamente tampoco, pero se reduce la distancia entre el R^2 y el $R^2_{ajustado}$.

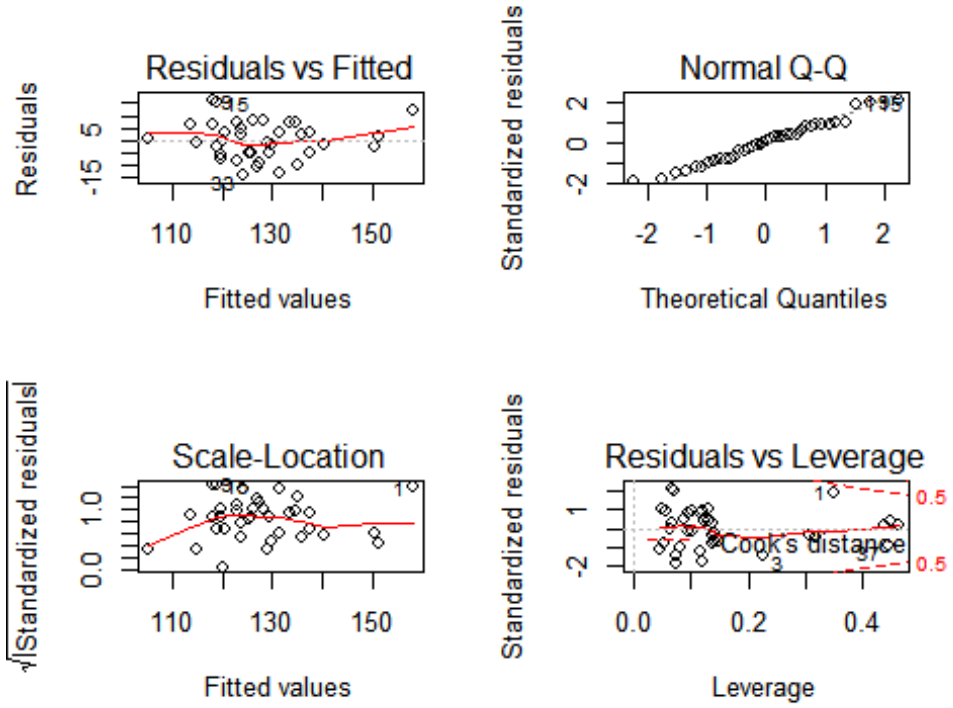
```
anova(lmod2, lmod4)
```

```
## Analysis of Variance Table
##
## Model 1: Systol ~ Age + Years + Life + Weight + Height + Chin +
Forearm +
##     Calf + Pulse
## Model 2: Systol ~ Age + Years + Life + Weight + Forearm
##   Res.Df    RSS Df Sum of Sq F Pr(>F)
## 1     28 2169.6
## 2     32 2045.8 -4    123.77
```

Se aceptaría el modelo reducido.

En cuanto a los residuos:

```
par(mfrow=c(2,2))
plot(lmod4)
```




```
summary(lm(sqrt(abs(residuals(lmod4)))~fitted(lmod4)))

##
## Call:
## lm(formula = sqrt(abs(residuals(lmod4))) ~ fitted(lmod4))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.16104 -0.70855  0.09574  0.55179  1.73378
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1386942  1.9119685   1.119    0.271
## fitted(lmod4) 0.0009678  0.0148975   0.065    0.949
##
## Residual standard error: 0.9538 on 36 degrees of freedom
## Multiple R-squared:  0.0001172,  Adjusted R-squared:  -0.02766
## F-statistic: 0.00422 on 1 and 36 DF,  p-value: 0.9486
```

Parece que no hay relación entre los valores ajustados del modelo como predictores de los residuos,con lo que asumimos que la varianza de los errores de la regresión es constante.

(e)Comprobar si hemos solucionado el problema de multicolinealidad en el modelo reducido del apartado anterior.

```
X4<- model.matrix(lmod4)[,-1]
V4<-  eigen(t(X4)%*%X4)$values
sqrt(max(V4)/V4)

## [1]    1.000000    8.240151   14.033314   39.023583 1319.374174

summary(lm(X4[,1] ~ X4[,-1]))$r.squared

## [1] 0.6779389

1 / (1-0.678)

## [1] 3.10559

vif(lmod4)

##       Age     Years      Life    Weight   Forearm
##  3.105001 35.118905 24.738540  2.245658  1.665539
```

Sigue existiendo multicolinealidad, lo que no es de estrañar sabiendo que Life es una combinación lineal de Age y Years.

PLS.

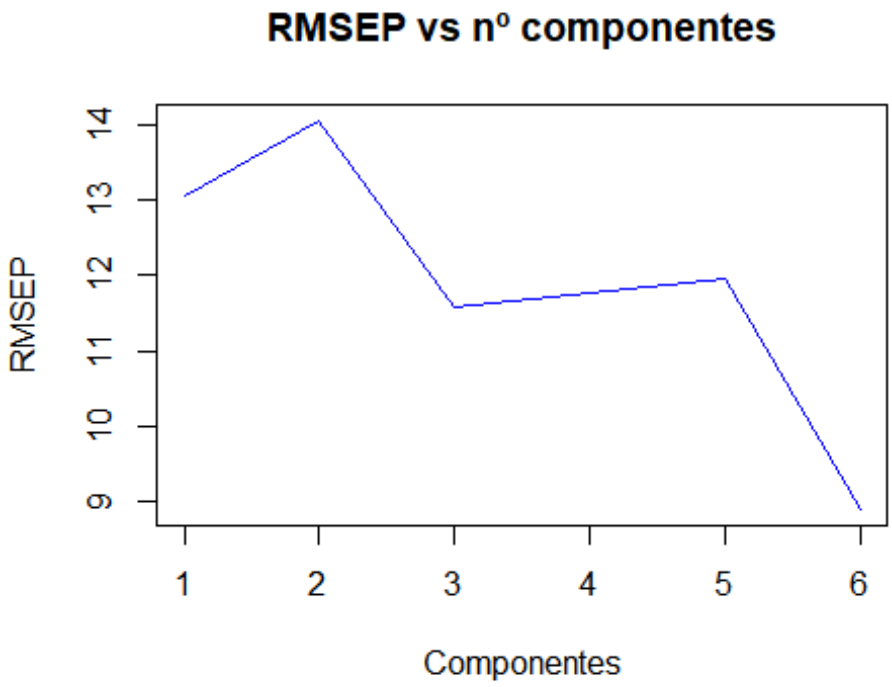
Como los investigadores no quieren prescindir de más variables, se plantea una regresión Partial Least Squares (PLS). ¿Cuantas componentes se necesitan para minimizar el RMSEP?

Escalamos las variables ya que hay diferentes magnitudes.

```
modelo_pls <- plsr(Systol~Age+Years + Life + Weight + Forearm, data =
peru3, scale. = TRUE, validation = "CV")
modelo_pls_CV <- RMSEP(modelo_pls, estimate = "CV")
which.min(modelo_pls_CV$val)

## [1] 6
```

Para minimizar el RMSEP se necesitan 6 componentes, las 5 variables predictoras.

```
plot(modelo_pls_CV$val, main = "RMSEP vs nº componentes", type = "l",
     ylab = "RMSEP", col = "blue", xlab = "Componentes")
```



Calcular los coeficientes de las variables originales, también para beta0, que proporciona este método con el número de componentes necesario.¿Es adecuado este método de regresión con estas variables? ¿Es útil? PLS encuentra combinaciones lineales de los predictores que mejor explican la respuesta, pero usualmente no supone una reducción de la dimensión, ya que cada predictor contribuye a esas combinaciones lineales. Es mas eficaz cuando hay un gran número de variables a considerar, que no es nuestro caso. Si tiene éxito, la variabilidad de la predicción se reduce sustancialmente. En el caso de querer utilizar este método con fines predictivos, es necesario verificar que se cumplen las condiciones necesarias para regresión por mínimos cuadrados

```
coef(modelo_pls)
```

```
## , , 5 comps
##
##              Systol
## Age       -1.203490
## Years      3.295147
## Life    -153.256987
## Weight     1.162357
## Forearm   -1.723303
```

(f)Siguiendo con el modelo reducido, otra posibilidad es utilizar la Ridge Regression. ¿Cuales son los coeficientes obtenidos? Explicar brevemente las ventajas e inconvenientes de este método frente a la selección de variables.

Ridge regression.

Ridge regression supone que los coeficientes de regresión (después de la normalización) no deben ser muy grandes. Esto resulta útil cuando tienes una gran cantidad de predictores y se cree que muchos de ellos tienen algún efecto en la respuesta. Es particularmente efectivo cuando la matriz del modelo es colineal y las estimaciones de mínimos cuadrados de beta parecen ser inestables.

La principal ventaja del ajuste por ridge regression frente al ajuste por mínimos cuadrados es la reducción de varianza. Por lo general, en situaciones en las que la relación entre la variable respuesta y los predictores es aproximadamente lineal, las estimaciones por mínimos cuadrados tienen poco bias pero aún pueden sufrir alta varianza (pequeños cambios en los datos training tienen mucho impacto en el modelo resultante). Este problema se acentúa conforme el número de predictores introducido en el modelo se aproxima al número de observaciones de training, llegando al punto en que, si p>n, no es posible ajustar por mínimos cuadrados. Empleando un valor adecuado de λ, identificado mediante cross-validation, el método de ridge regression es capaz de reducir varianza, consiguiendo así un menor error total. El inconveniente es que la reducción de la varianza va a aumentar el bias de los coeficientes.

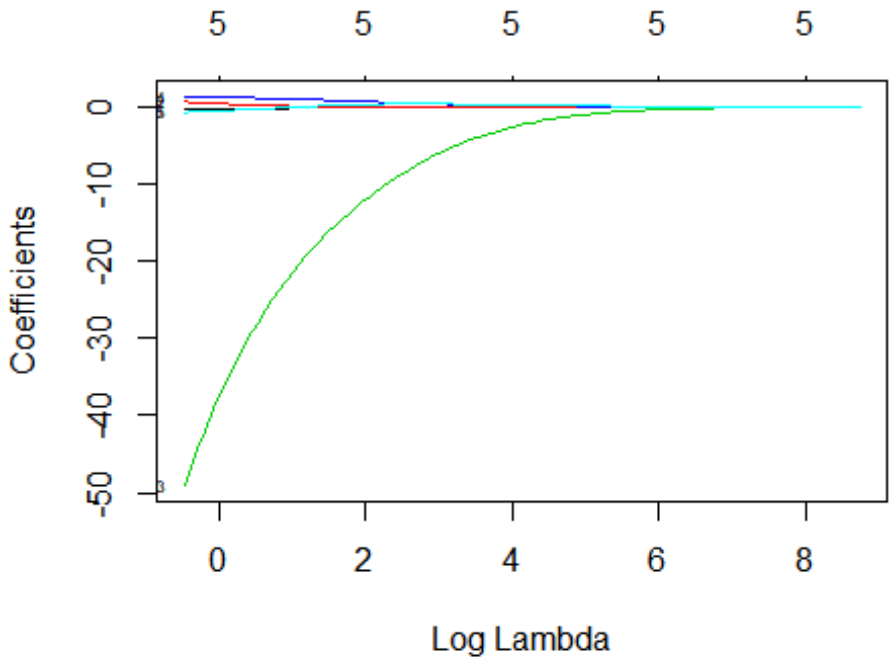
La limitación del método de ajuste por ridge regression en comparación a los métodos de subset selection es que el modelo final va a incluir todos los predictores. Esto es así porque, si bien la penalización empleada fuerza a que los coeficientes tiendan a cero, nunca llegan a ser exactamente cero (solo si λ=∞). Este método consigue minimizar la influencia sobre el modelo de los predictores menos relacionados con la variable respuesta, pero en el modelo final van a seguir apareciendo. Aunque esto no supone un problema para la precisión del modelo, sí lo es para su interpretación.

La función glmnet() estandariza por defecto las variables antes de realizar el ajuste del modelo.

```
x <- model.matrix(lmod4, data = peru3)[, -1]
y<- peru3$Systol
# Para obtener un ajuste mediante ridge regression se indica argumento
alpha=0.
modelos_ridge <- glmnet(x = x, y = y, alpha = 0)
```

glmnet() almacena en una matriz el valor de los coeficientes de regresión de los predictores para cada valor de λ.

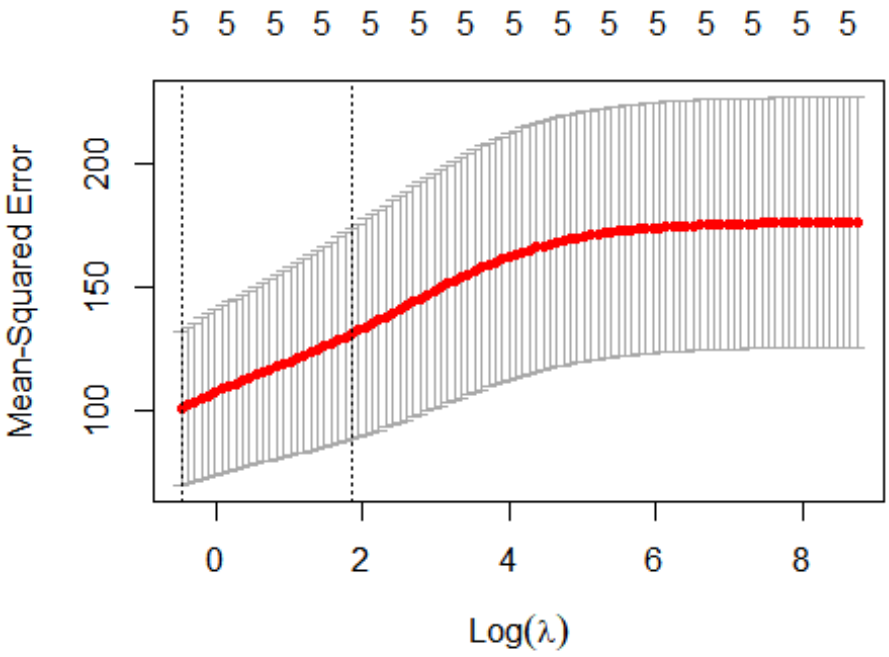
```
plot(modelos_ridge, xvar = "lambda", label = TRUE)
```



Como es de esperar, los coeficientes se van haciendo más pequeños a medida que se incrementa el valor de λ. Cuando $\lambda=0$ la penalización es nula y los resultados son equivalentes a los obtenidos por mínimos cuadrados, cuando $\lambda=\infty$ todos los coeficientes son cero, lo que equivale al modelo sin ningún predictor (modelo nulo).

Con el fin de identificar el valor de λ que da lugar al mejor modelo, se puede recurrir a Cross-Validation. La función cv.glmnet() calcula el cv-test-error, utilizando por defecto k=10. El gráfico muestra el cv-test-error (Mean Square Error) para cada valor de λ junto con la barra de error correspondiente.

```
set.seed(2020)
cv_error_ridge <- cv.glmnet(x = x, y = y, alpha = 0, nfolds = 10,
                            type.measure = "mse")
plot(cv_error_ridge)
```




```
# Valor lambda con el que se consigue el mínimo cvtest-error
cv_error_ridge$lambda.min
```

```
## [1] 0.6260626
```

```
# Valor lambda óptimo: mayor valor de lambda con el que el test-error no
se
# aleja más de 1 sd del mínimo cvtest-error posible.
cv_error_ridge$lambda.1se
```

```
## [1] 6.407945
```

Acorde al principio de parsimonia y la norma de *one standard error rule*, el mejor modelo es el que se obtiene con λ = 6.4. lambda.1se siempre es mayor que lambda.min.

```
# Se muestra el valor de los coeficientes para el valor de lambda óptimo
modelo_final_ridge <- glmnet(x = x, y = y, alpha = 0, lambda = 6.4)
coef(modelo_final_ridge)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept)  89.49580075
## Age          -0.10065853
## Years        -0.01631233
## Life        -13.20029078
## Weight        0.72530600
## Forearm       0.25205293
```

Calcular el RMSE de la regresión OLS, PLS (con 5, 4, 3 y 2 componentes) y Ridge (con lampda óptima por GCV) para el modelo reducido.¿Cual es la valoración con todo lo que sabemos hasta ahora? ### Ordinary least square (regresión por mínimos cuadrados)

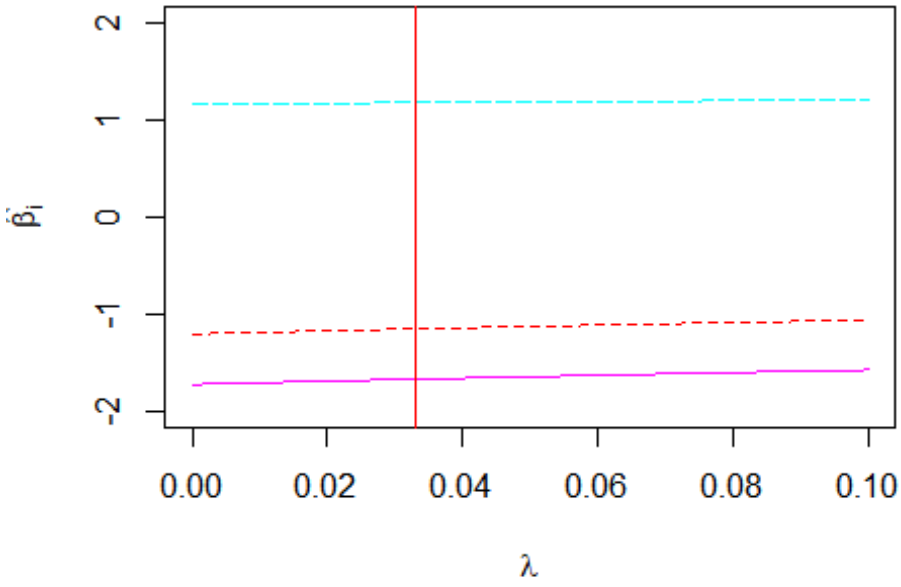
```
set.seed(1)
indices_entrenamiento <- sample(x = 1:nrow(peru3),size = 30)
indices_test <- (1:nrow(peru3))[-indices_entrenamiento]
Peru_1 <- peru3[indices_entrenamiento,]
Peru_2 <- peru3[indices_test,]
modelo_OLS   <- lm(formula = Systol~Age+Years + Life + Weight + Forearm,
data = Peru_1)
(test_RMSE_OLS <- sqrt(mean((predict(modelo_OLS, Peru_2) -
Peru_2$Systol)^2)))

## [1] 9.223916
```

PLS

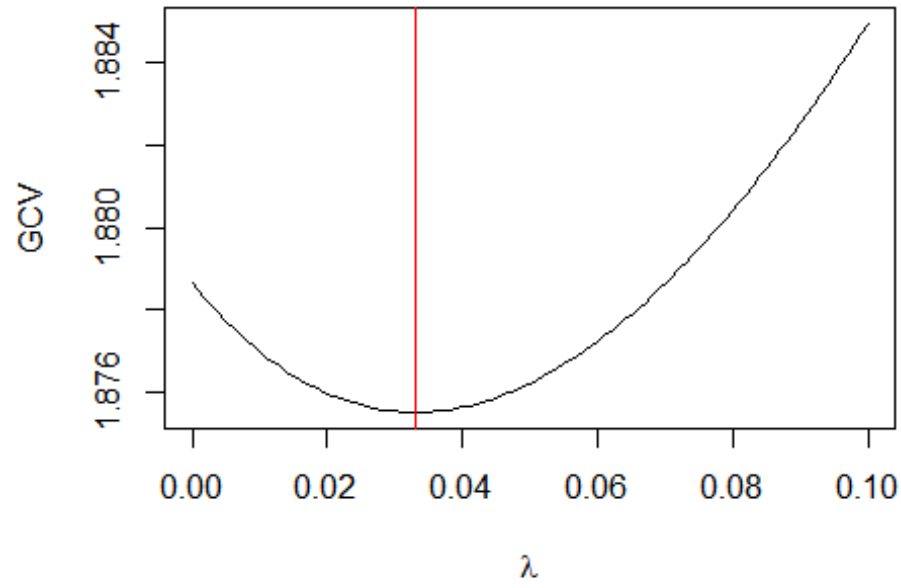
```
predicciones5 <- predict(modelo_pls, newdata = Peru_2, ncomp = 5)
predicciones4 <- predict(modelo_pls, newdata = Peru_2, ncomp = 4)
predicciones3 <- predict(modelo_pls, newdata = Peru_2, ncomp = 3)
predicciones2 <- predict(modelo_pls, newdata = Peru_2, ncomp = 2)

(test_rmse_pls5 <- sqrt(mean((predicciones5 - Peru_2$Systol)^2)))

## [1] 7.627487

(test_rmse_pls4 <- sqrt(mean((predicciones4 - Peru_2$Systol)^2)))

## [1] 12.44332

(test_rmse_pls3 <- sqrt(mean((predicciones3 - Peru_2$Systol)^2)))

## [1] 12.32702

(test_rmse_pls2 <- sqrt(mean((predicciones2 - Peru_2$Systol)^2)))

## [1] 13.00289
```

Ridge Regression.

```
ridge<- lm.ridge(Systol~Age+Years + Life + Weight + Forearm, data =
peru3, scale. = TRUE, lambda=(seq(0,0.1,0.001)))
(nGCV <- which.min(ridge$GCV))

## 0.033
##    34

lGCV <- ridge$lambda[nGCV]
matplot(ridge$lambda,coef(ridge),type="l", ylim=c(-2,2),
xlab=expression(lambda),ylab=expression(hat(beta[i])))
abline(v=lGCV,col=2)
```



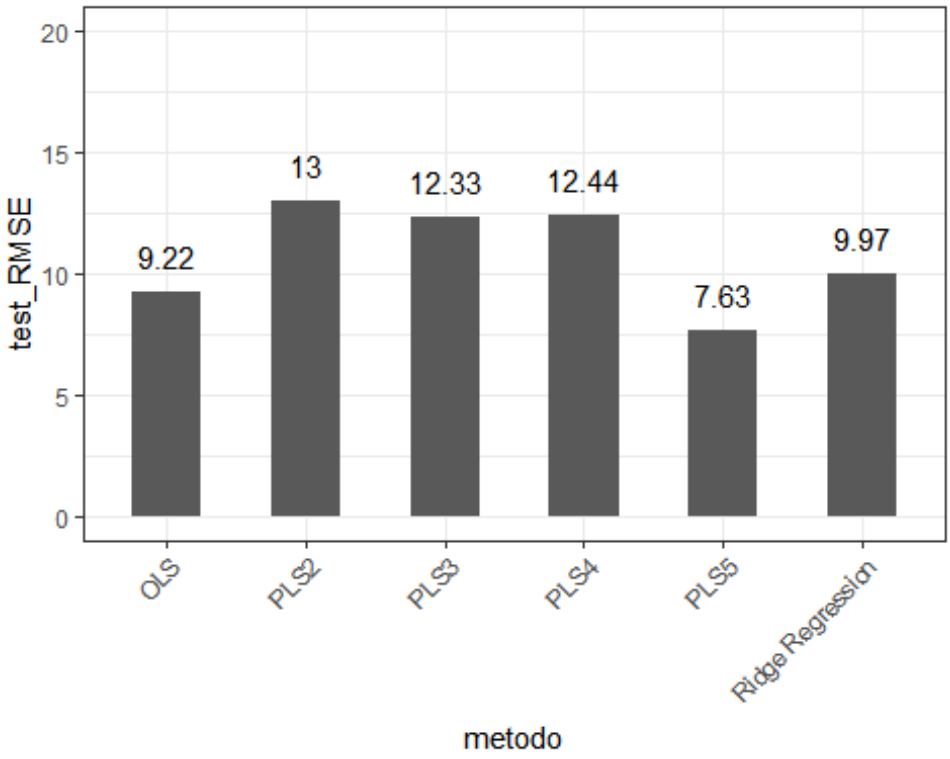

```r
plot(ridge$lambda,ridge$GCV,type="l",xlab=expression(lambda),ylab="GCV")
abline(v=lGCV,col=2)
```



```r
Rigde_model<- lm.ridge(Systol~Age+Years + Life + Weight + Forearm, data =
peru3, scale. = TRUE, lambda=lGCV)

x_Peru_1 <- model.matrix(Systol~Age+Years + Life + Weight + Forearm, data
= Peru_1)[, -1]
y_Peru_1 <- Peru_1$Systol

x_Peru_2 <- model.matrix(Systol~Age+Years + Life + Weight + Forearm, data
= Peru_2)[, -1]
y_Peru_2 <- Peru_2$Systol

modelo_ridge <- glmnet(x = x_Peru_1, y = y_Peru_1, alpha = 0,
                       lambda = lGCV)
predicciones <- predict(object = modelo_ridge, newx = x_Peru_2,
                        s = lGCV, exact = TRUE)
(test_RMSE_ridge <- sqrt(mean((predicciones - Peru_2$Systol)^2)))
```

```
## [1] 9.968587
```

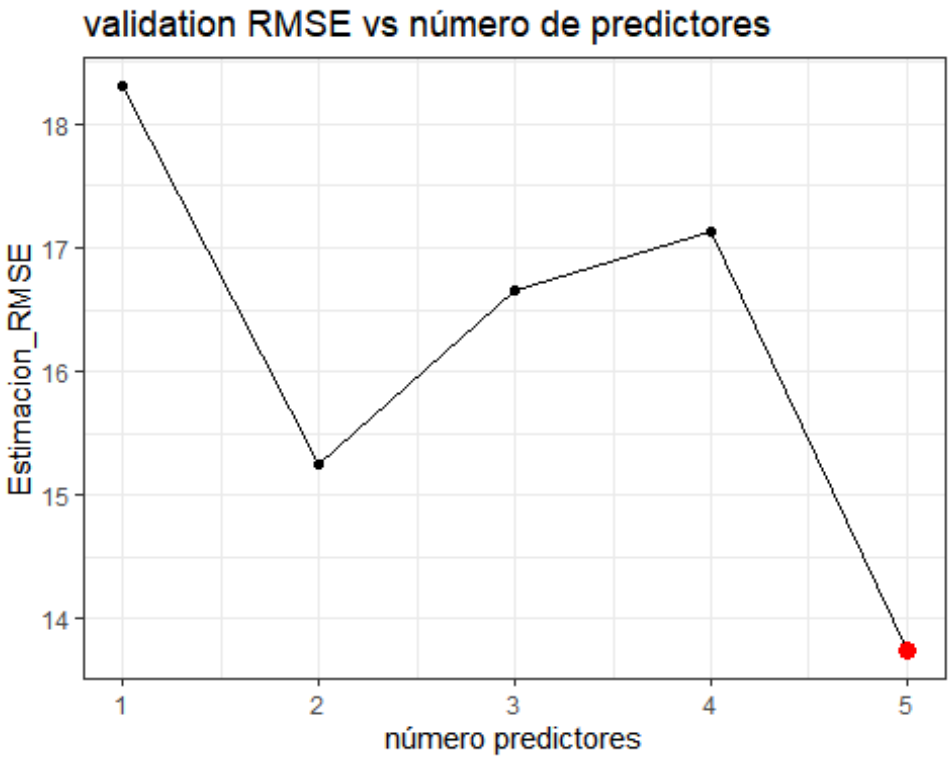
```r
valores_testRMSE <- data.frame(metodo = c(
"PLS5","PLS4","PLS3","PLS2","OLS",
                                          "Ridge Regression"),
                               test_RMSE =
c(test_rmse_pls5,test_rmse_pls4,
                                           test_rmse_pls3,test_rmse_pls2,

test_RMSE_OLS,test_RMSE_ridge))
ggplot(data = valores_testRMSE, aes(x = metodo, y = test_RMSE)) +
  geom_col(width = 0.5) +
  lims(y = c(0, 20)) +
  geom_text(aes(label = round(test_RMSE, 2)), vjust = -1) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



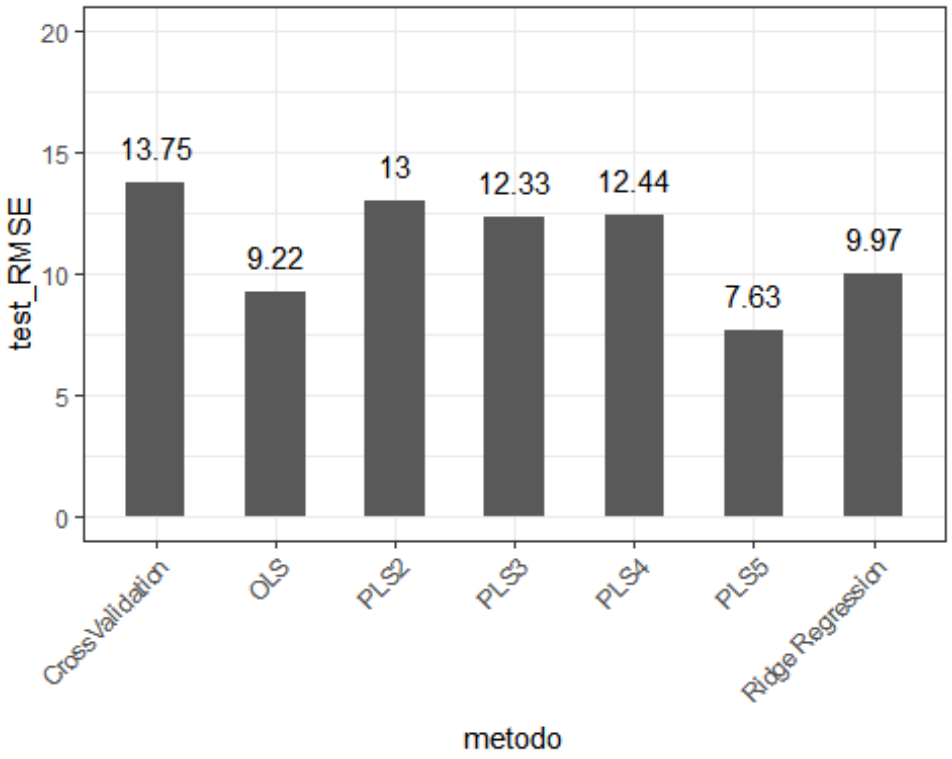
(g)Sabemos que el RMSE calculado en un modelo para todos los datos observados es muy optimista. Es mejor un cálculo por validación cruzada. Con el modelo reducido de los apartados anteriores y para comparar los métodos estudiados (OLS,PLS (con 4 componentes) y Ridge (con lambda óptimo por GCV) haremos lo siguiente:1. Dividiremos los datos aleatoriamente en dos grupos, uno de 8 observaciones (grupo test) y otro del resto (grupo train). Recordemos que el número total de observaciones es ahora de 38.2. Ajustaremos cada modelo con el grupo train y calcularemos el RMSE con el grupo test.3. Repetiremos los pasos 1 y 2 mil veces.4. Finalmente compararemos los resultados para cada modelo con algún estadístico y también gráficamente con las densidades de los RMSE.¿Qué podemos decir?

```
library(leaps)
# En este caso se emplea forward stepwise selection
set.seed(11)
train <- sample(x = 1:38, size = 30, replace = FALSE)
mejores_modelos <- regsubsets(Systol~Age + Years + Life + Weight
+Forearm, data = peru3[train,], nvmax = 5, method = "forward")
validation_error <- rep(NA, 5)
test_matrix <- model.matrix(lmod2, data = peru3[-train, ])
# Para cada uno de los modelos almacenados en la variable mejores modelos
for (i in 1:5){
  # Se extraen los coeficientes del modelo
  coeficientes <- coef(object = mejores_modelos, id = i)
  # Se identifican los predictores que forman el modelo y se extraen de
la
```

```r
  # matriz modelo
  predictores <- test_matrix[, names(coeficientes)]
  # Se obtienen las predicciones mediante el producto matricial de los
  # predictores extraídos y los coeficientes del modelo
  predicciones <- predictores %*% coeficientes
  # Finalmente se calcula la estimación del test RMSE
  validation_error[i] <- sqrt(mean((peru3$Systol[-train] -
predicciones)^2))
  }
which.min(validation_error)

## [1] 5

# Gráfico
p <- ggplot(data = data.frame(n_predictores = 1:5,
                              Estimacion_RMSE = validation_error),
            aes(x = n_predictores, y = Estimacion_RMSE)) +
    geom_line() +
    geom_point()

# Se identifica en rojo el mínimo
p <- p + geom_point(aes(x = n_predictores[which.min(validation_error)],
                        y =
validation_error[which.min(validation_error)]),
                        colour = "red", size = 3)

p <- p +  scale_x_continuous(breaks = c(0:5)) +
          theme_bw() +
          labs(title = 'validation RMSE vs número de predictores',
               x =  'número predictores')
p
```




```
valores_testRMSE <- data.frame(metodo = c(
"PLS5","PLS4","PLS3","PLS2","OLS",
                                          "Ridge Regression",
"CrossValidation"),
                               test_RMSE =
c(test_rmse_pls5,test_rmse_pls4,
                                           test_rmse_pls3,test_rmse_pls2,
                                           test_RMSE_OLS,test_RMSE_ridge,
                                           validation_error[5]))
ggplot(data = valores_testRMSE, aes(x = metodo, y = test_RMSE)) +
  geom_col(width = 0.5) +
  lims(y = c(0, 20)) +
  geom_text(aes(label = round(test_RMSE, 2)), vjust = -1) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



(h)Calcular los grados de libertad de la Ridge regression para el λ óptimo del apartado (e).

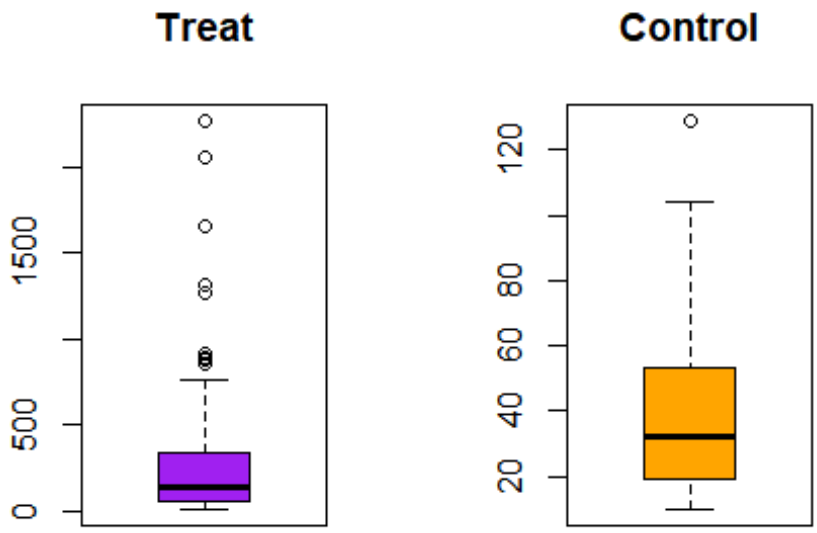
```
modelo_final_ridge$df

## [1] 5
```

Ejercicio 2.

```
T33 <- read_table2("T33.1", col_names = FALSE)
T33<- T33[,-c(1:4,7,8)]
colnames(T33)<- c("sex","age","C","D")
cancer<- c(rep("stomach",13), rep("bronq",17), rep("colon",17),
rep("other",53))
cancer<- as.factor(cancer)
T34<- dplyr::mutate(T33, cancer)
T34$sex<- factor(T34$sex, levels=c("F","M"))
View(T34)
```

(a)Estudiar la transformación que mejora la distribución de los datos C y los datos D (100 observaciones en cada caso). Se puede utilizar el método de Box-Cox.

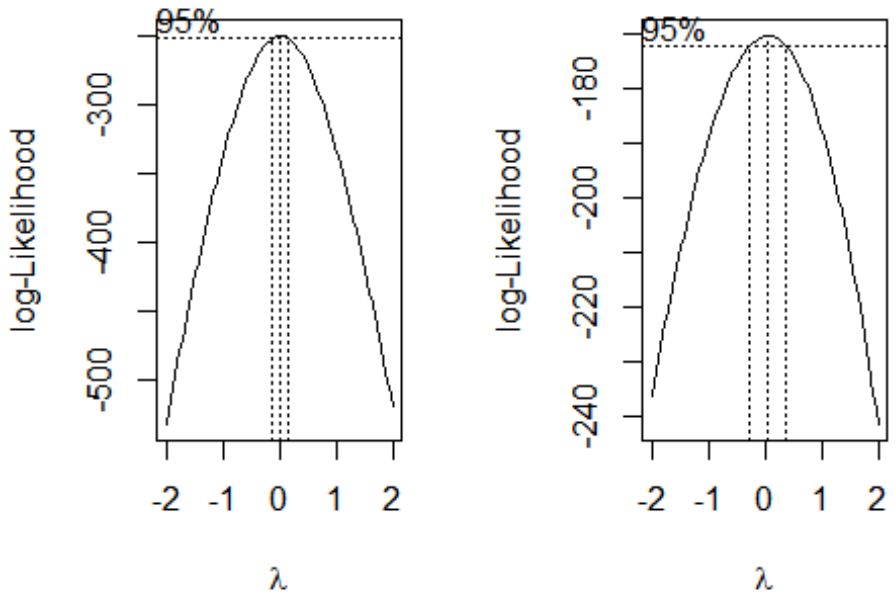
```
par(mfrow=c(1,2))
boxplot(T34$C, main="Treat", col="purple")
boxplot(T34$D, main="Control", col="orange")
```



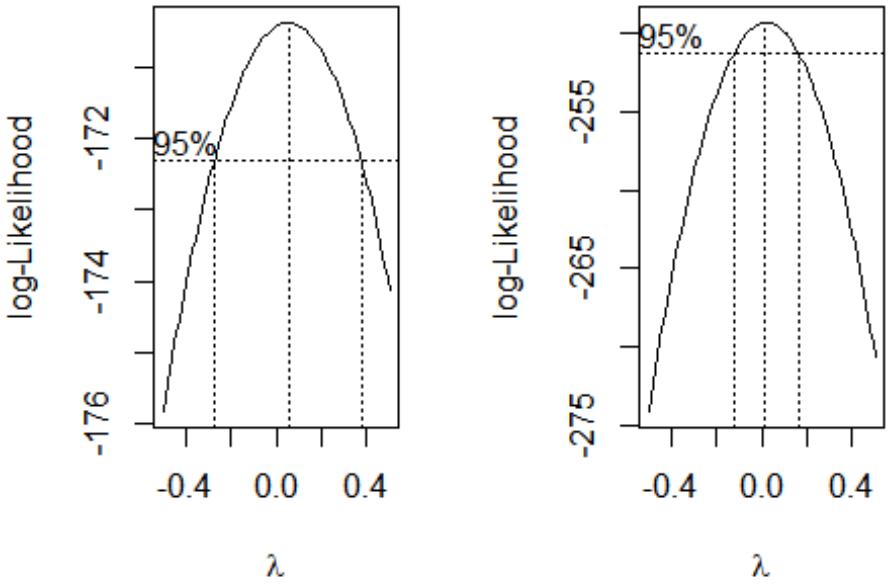
Los datos son muy asimétricos.

```
BoxCoxTrans(T34$C)

## Box-Cox Transformation
##
## 100 data points used to estimate Lambda
##
## Input data summary:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     8.0    57.5   136.5   293.3   338.2  2270.0
##
## Largest/Smallest: 284
## Sample Skewness: 2.78
##
## Estimated Lambda: 0
## With fudge factor, Lambda = 0 will be used for transformations

BoxCoxTrans(T34$D)

## Box-Cox Transformation
##
## 100 data points used to estimate Lambda
##
## Input data summary:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.00   19.75   32.00   37.79   52.75  129.00
##
```

```
## Largest/Smallest: 12.9
## Sample Skewness: 1.15
##
## Estimated Lambda: 0
## With fudge factor, Lambda = 0 will be used for transformations

lm34C<- lm(C~age+sex+cancer, data = T34)
lm34D<- lm(D~age+sex+cancer, data = T34)
par(mfrow=c(1,2))
boxcox(lm34C,plotit=T)
boxcox(lm34D, plotit=T)
```



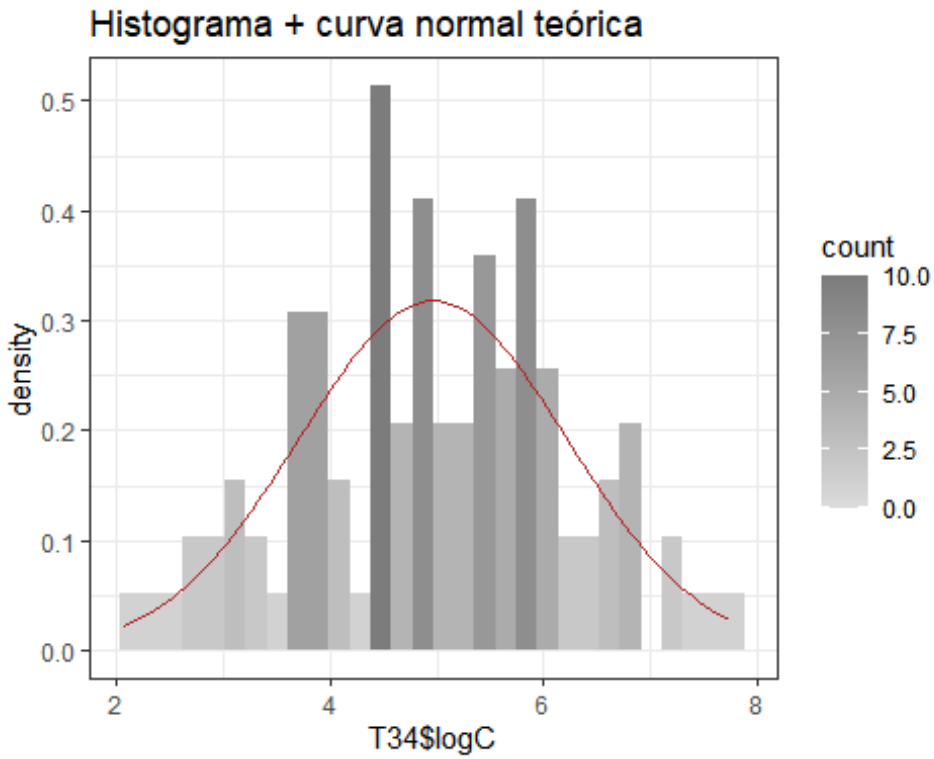

```
par(mfrow=c(1,2))
boxcox(lm34D, plotit = TRUE, lambda = seq(-0.5, 0.5, by = 0.1))
boxcox(lm34C, plotit = TRUE, lambda = seq(-0.5, 0.5, by = 0.1))
```



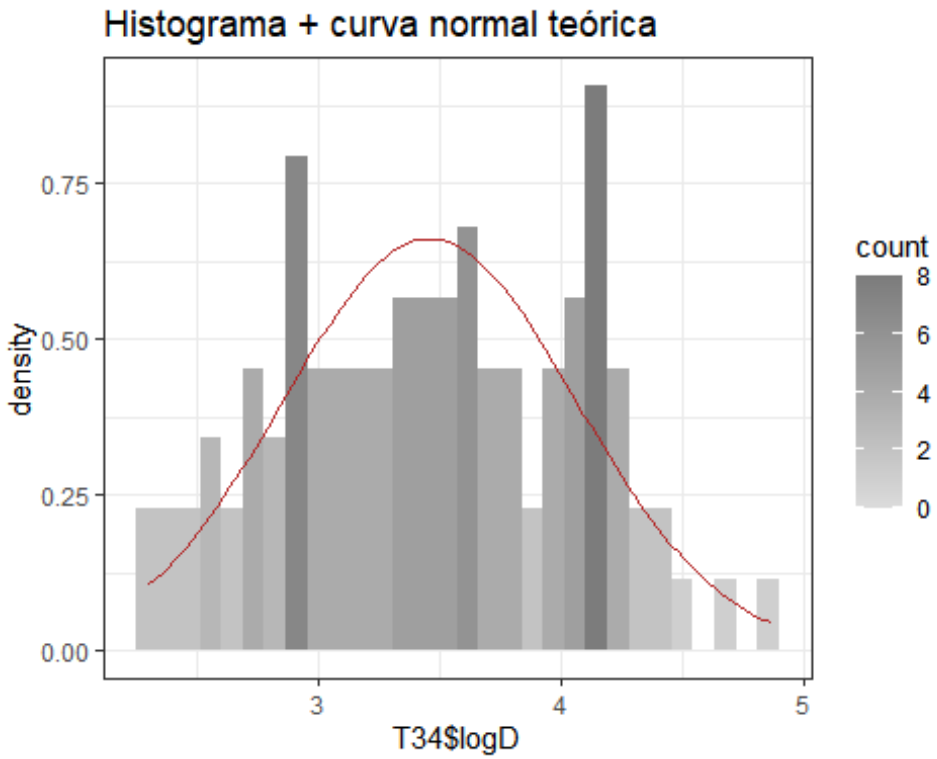
Un valor de lambda tan próximo a 0 sugiere aplicar log.

Una vez transformados, comparar si el tiempo de supervivencia C es superior al de los controles D con todas las observaciones.

```
T34$logC<-log(T34$C)
T34$logD<-log(T34$D)
ggplot(data = T34, aes(x = T34$logC)) +
  geom_histogram(aes(y = ..density.., fill = ..count..)) +
  scale_fill_gradient(low = "#DCDCDC", high = "#7C7C7C") +
  stat_function(fun = dnorm, colour = "firebrick",
                args = list(mean = mean(T34$logC), sd = sd(T34$logC))) +
  ggtitle("Histograma + curva normal teórica") +
  theme_bw()
```

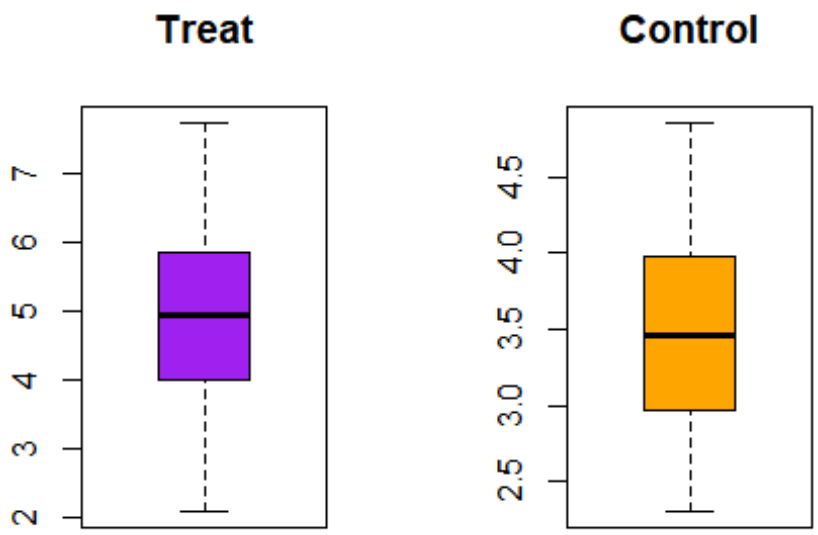



```
ggplot(data = T34, aes(x = T34$logD)) +
  geom_histogram(aes(y = ..density.., fill = ..count..)) +
  scale_fill_gradient(low = "#DCDCDC", high = "#7C7C7C") +
  stat_function(fun = dnorm, colour = "firebrick",
                args = list(mean = mean(T34$logD), sd = sd(T34$logD))) +
  ggtitle("Histograma + curva normal teórica") +
  theme_bw()
```



Como no hay evidencias de que los datos ahora sean no-normales, comparamos las medias de controles y tratados con un t test unilateral (sólo nos interesa la mejora):

```
shapiro.test(T34$logC)

##
##  Shapiro-Wilk normality test
##
## data:  T34$logC
## W = 0.99167, p-value = 0.7971

shapiro.test(T34$logD)

##
##  Shapiro-Wilk normality test
##
## data:  T34$logD
## W = 0.97747, p-value = 0.08423

par(mfrow=c(1,2))
boxplot(T34$logC, main="Treat", col="purple")
boxplot(T34$logD, main="Control", col="orange")
```




```
t.test(T34$logD, T34$logC, alternative = "less",
       mu = 0, paired = F, conf.level = 0.95, var.equal = T)

##
##  Two Sample t-test
##
## data:  T34$logD and T34$logC
## t = -10.822, df = 198, p-value < 2.2e-16
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##       -Inf -1.275626
## sample estimates:
## mean of x mean of y
##  3.455923  4.961451
```

Rechazamos la hipótesis nula. El log del tiempo de supervivencia es mayor que el log del tiempo de los controles.

(b)Ahora estamos interesados en comparar la mejora en función del tipo de cáncer. Nos centraremos exclusivamente en los tres tipos de cáncer de la tabla 1 de más arriba y no tendremos en cuenta el sexo.

```
logCD<-round(log(T34$C/T34$D),2)
T34<- dplyr::mutate(T34,logCD)
View(T34)
lmb<- lm(logCD~0+cancer, data=T34[1:47,])
```

```
X<- model.matrix(lmb)
Y<- T34$logCD[1:47]
```

Calcular los elementos de dicha tabla con la matriz de diseño X de este modelo y resolver con ellos el contraste H0 : μ1 = μ2 = μ3 cuando la variable respuesta Y es el logaritmo de la razón entre la supervivencia de los tratados y la supervivencia de sus controles. ¿Cual es la conclusión? Nota: Habrá que tener en cuenta que en la tabla 2 se supone que el número de réplicas r es el mismo para todos los niveles, cosa que no pasa en este caso.

```
#uncorrected SS
SSuc<- t(Y)%*%Y
paste("uncorrected SS:",round(SSuc,2))
```

```
## [1] "uncorrected SS: 185.89"
```

```
#SS(model)
xtxi<- solve(t(X)%*%X)
b<- xtxi%*%t(X)%*%Y
SSmod<- t(b) %*%t(X)%*%Y
paste("model SS:",round(SSmod,2))
```

```
## [1] "model SS: 118.85"
```

```
#Media: SS(μ)
#SSmu<- t(b)*(t(unos)%*%Y)
n<- 47
unos<- c(rep(1,47))
mediaY<- (1/n)%*%t(unos)%*%Y
SSmu<-n*mediaY^2
paste("mediaSS:",round(SSmu,2))
```

```
## [1] "mediaSS: 113.01"
```

```
#SS(Regr) = SS(Model) – SS(μ)
SSreg<- SSmod-SSmu
paste("regressionSS:" , round(SSreg,2))
```

```
## [1] "regressionSS: 5.84"
```

```
#SS(res)
SSres<- SSuc-SSmod
paste("residualSS:",round(SSres,2))
```

```
## [1] "residualSS: 67.03"
```

Calculamos el estadístico F:

```
RSS<- t(Y)%*%Y - t(Y)%*%X%*%b
lmb2<- lm(logCD~1, data=T34[1:47,])
Xo<-model.matrix(lmb2)
#μ1 = μ2 = μ3=0
```

```r
RSSH<- t(Y)%*%Y - t(Y)%*%Xo%*%solve(t(Xo)%*%Xo)%*%t(Xo)%*%Y
q<- 2 #grados de libertad 3-1=2
qr(X)$rank
```

```
## [1] 3
```

```r
qr(Xo)$rank
```

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

```r
nF<- (RSSH-RSS)/q  #numerador test F
dF<- RSS/(47-3) #denominador test F
Ftest<- nF/dF; Ftest
```

```
##          [,1]
## [1,] 1.917412
```

```r
1-pf(Ftest,3,43)
```

```
##           [,1]
## [1,] 0.1410381
```

```r
#Suponemos igualdad de varianzas.
tStu<- sqrt(Ftest); tStu
```

```
##          [,1]
## [1,] 1.384706
```

```r
anova(lmb2,lmb)
```

```
## Analysis of Variance Table
##
## Model 1: logCD ~ 1
## Model 2: logCD ~ 0 + cancer
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     46 72.875
## 2     44 67.033  2    5.8423 1.9174 0.1591
```

No podemos rechazar la hipótesis nula, no existen diferencias entre los 3 tipos de cáncer.

(c)La edad de los pacientes presenta una cierta variabilidad y puede influir en su supervivencia. Añadir a la matriz X del apartado anterior el vector columna con las edades centradas. Utilizar las sumas de cuadrados de los residuos de este modelo y del anterior para contrastar la importancia de ajustar con la edad.¿Se puede utilizar un test t de Student? Como comparamos un factor con 3 niveles (3 medias), se utiliza un testa anova.

```r
T34$agecen<-T34$age-mean(T34$age)
lmc<- lm(logCD~0+scale(age)+cancer, data=T34[1:47,])
X2<- model.matrix(lmc)
Y<- T34$logCD[1:47]
#uncorrected SS
```

```
SSuc<- t(Y)%*%Y
SSuc
```

```
##          [,1]
## [1,] 185.8856
```

```
#SS(model)
xtxi2<- solve(t(X2)%*%X2)
b2<- xtxi2%*%t(X2)%*%Y
SSmod2<- t(b2) %*%t(X2)%*%Y
SSmod2
```

```
##          [,1]
## [1,] 119.1554
```

```
#Media: SS(μ)
#SSmu<- t(b)*(t(unos)%*%Y)
n<- 47
unos<- c(rep(1,47))
mediaY<- (1/n)%*%t(unos)%*%Y
SSmu<-n*mediaY^2
SSmu
```

```
##          [,1]
## [1,] 113.0105
```

```
#SS(Regr) = SS(Model) – SS(μ)
SSreg2<- SSmod2-SSmu
SSreg2
```

```
##          [,1]
## [1,] 6.144881
```

```
#SS(res)
SSres2<- SSuc-SSmod2
SSres2; SSres
```

```
##         [,1]
## [1,] 66.7302
```

```
##          [,1]
## [1,] 67.03283
```

El SSres no varía practicamente.

(d)Aunque la regresión de la edad en el modelo anterior pudiera no ser importante, se decidió que cada grupo debería tener su propia regresión sobre la edad para verificar si la edad no es importante en ningunode los grupos. Modificar adecuadamente la matriz de diseño para acomodar esta nueva situación y completar el test para la hipótesis nula de que la regresión sobre la edad es la misma en los tres grupos de cáncer. ¿Cual es la conclusión?

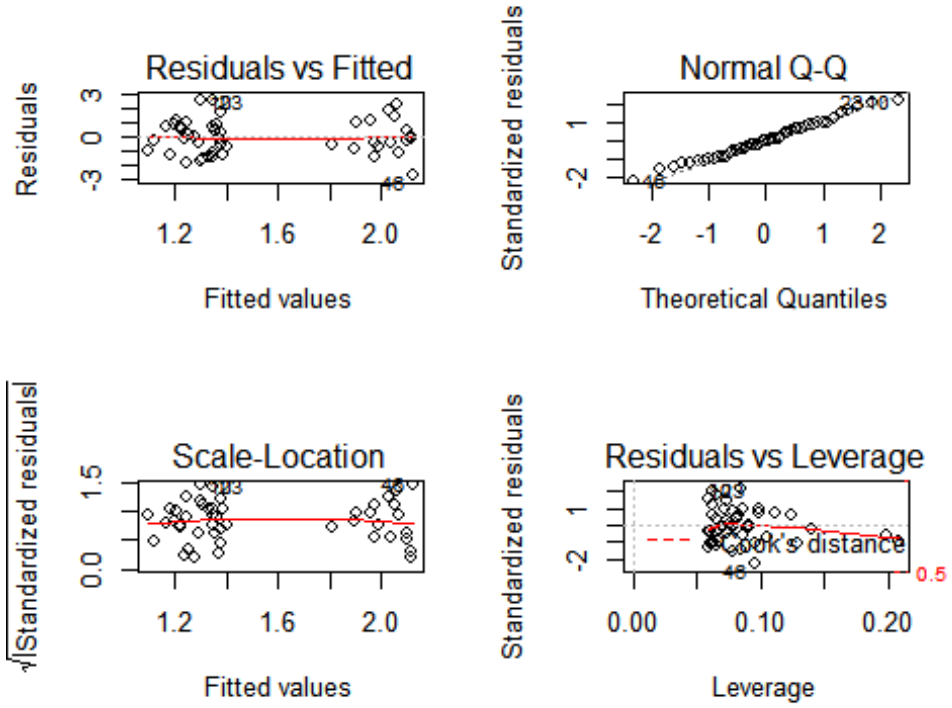
```
RSS2<- t(Y)%*%Y - t(Y)%*%X2%*%b2
lmb0<- lm(logCD~0+scale(age), data=T34[1:47,])
Xo2<-model.matrix(lmb0) #μ1 = μ2 = μ3=0
RSSH2<- t(Y)%*%Y - t(Y)%*%Xo2%*%solve(t(Xo2)%*%Xo2)%*%t(Xo2)%*%Y
(q2<-qr(X2)$rank-qr(Xo2)$rank)

## [1] 3

qr(X2)$rank

## [1] 4

nF2<- (RSSH2-RSS2)/q2  #numerador test F
dF2<- RSS2/(47-4) #denominador test F
(Ftest2<- nF2/dF2)

##          [,1]
## [1,] 25.54851

#Suponemos igualdad de varianzas.
(tStu2<- sqrt(Ftest2))

##          [,1]
## [1,] 5.054554

1-pf(Ftest2,2,44)

##              [,1]
## [1,] 4.327991e-08
```

El estadístico F aumenta. Valores muy altos de F nos hacen rechazar la hipótesis nula.

```
anova(lmb0, lmc)

## Analysis of Variance Table
##
## Model 1: logCD ~ 0 + scale(age)
## Model 2: logCD ~ 0 + scale(age) + cancer
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     46 185.67
## 2     43  66.73  3    118.94 25.549 1.201e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Rechazar la hipótesis nula, los resultados difieren según el tipo de cáncer. Los resultados difieren en función de si incluimos o no la variable age. En cuanto a los residuos:

```
par(mfrow=c(2,2))
plot(lmc)
```




```
shapiro.test(residuals(lmc))

##
##  Shapiro-Wilk normality test
##
## data:  residuals(lmc)
## W = 0.98421, p-value = 0.7691
```

Apreciamos una cierta falta de normalidad de los residuos, aunque un test no llega a encontrarla significativa.

Ejercicio 3.

```
diabet <- read_csv("diabetes.txt")

## Parsed with column specification:
## cols(
##   pregnant = col_double(),
##   glucose = col_double(),
##   pressure = col_double(),
##   triceps = col_double(),
##   insulin = col_double(),
##   mass = col_double(),
##   pedigree = col_double(),
##   age = col_double(),
##   diabetes = col_character()
## )
```

```r
diabet$diab_status<- factor(diabet$diabetes, levels = c("neg","pos"),
labels = c(0,1))
View(diabet)
```

(a)

Ajustar un modelo de regresión logística para predecir la diabetes utilizando todas las otras variables como predictoras. Dar la ecuación del modelo obtenido y clasificar las variables según sean factores protectores o de riesgo para la diabetes.

```r
modelo <-
glm(formula=diab_status~pregnant+glucose+pressure+triceps+insulin+mass+pe
digree+age,data=diabet,family=binomial())
summary(modelo)
```

```
##
## Call:
## glm(formula = diab_status ~ pregnant + glucose + pressure + triceps +
##     insulin + mass + pedigree + age, family = binomial(), data =
diabet)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7823  -0.6603  -0.3642   0.6409   2.5612
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.004e+01  1.218e+00  -8.246  < 2e-16 ***
## pregnant     8.216e-02  5.543e-02   1.482  0.13825
## glucose      3.827e-02  5.768e-03   6.635 3.24e-11 ***
## pressure    -1.420e-03  1.183e-02  -0.120  0.90446
## triceps      1.122e-02  1.708e-02   0.657  0.51128
## insulin     -8.253e-04  1.306e-03  -0.632  0.52757
## mass         7.054e-02  2.734e-02   2.580  0.00989 **
## pedigree     1.141e+00  4.274e-01   2.669  0.00760 **
## age          3.395e-02  1.838e-02   1.847  0.06474 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 344.02  on 383  degrees of freedom
## AIC: 362.02
##
## Number of Fisher Scoring iterations: 5
```

Los predictores más significativos son *glucose, mass, pedigree,* por el valor de p-value.

```r
round(modelo$coefficients,2)
```

```
## (Intercept)    pregnant     glucose    pressure     triceps
insulin
##      -10.04        0.08        0.04        0.00        0.01
0.00
##        mass    pedigree         age
##        0.07        1.14        0.03
```

logit(diabetes) Y´=log(P/(1-P))= -10.04+0.08xpregnant + 0.04xglucose + 0.01xtriceps + 0.07xmass + 1.14xpedigree + 0.03xage

El odd es P/(1-P)= e^(-10.04+0.08xpregnant + 0.04xglucose + 0.01xtriceps + 0.07xmass + 1.14xpedigree + 0.03xage)

P(diabetes)= (e^(-10.04+0.08xpregnant + 0.04xglucose + 0.01xtriceps + 0.07xmass + 1.14xpedigree + 0.03xage))/ (1+ e^(-10.04+0.08xpregnant + 0.04xglucose + 0.01xtriceps + 0.07xmass + 1.14xpedigree + 0.03xage))

Intervalo de confianza y odd ratio para cada predictor:

```
confint.default(modelo)
```

```
##                      2.5 %      97.5 %
## (Intercept) -12.427337020 -7.65414134
## pregnant     -0.026472652  0.19079150
## glucose       0.026965021  0.04957402
## pressure     -0.024613319  0.02177274
## triceps      -0.022262065  0.04470484
## insulin      -0.003385886  0.00173526
## mass          0.016947976  0.12412719
## pedigree      0.303153921  1.97866332
## age          -0.002075888  0.06997913
```

```
round(exp(coef(modelo)),3)
```

```
## (Intercept)    pregnant     glucose    pressure     triceps
insulin
##       0.000       1.086       1.039       0.999       1.011
0.999
##        mass    pedigree         age
##       1.073       3.130       1.035
```

Un aumento en una unidad en todos los factores menos pedigree, multiplica por aproximadamente 1 la oportunidad (odds) de padecer diabetes frente a no padecerla, por lo que serían factores neutros, o de riesgo leve. Este odd se multiplica por 3.13 para cada unidad de aumento en el pedigree, lo que lo convierte en un factor de riesgo importante. En general, un odd ratio de 1 nos indica que no hay interacción respuesta/predictor, ya que tendríamos que la p(exito)/p(no exito)=1. Si esta relación es mayor que uno, la probabilidad de éxito aumentan multiplicativamente por e^i, por cada unidad de aumento en ese coeficiente. Si la razón de probabilidades es menor que 1, entonces las probabilidades de éxito son menores para niveles más altos de un

predictor continuo (o para el nivel indicado de un factor). Los valores más alejados de 1 representan grados más fuertes de asociación

(b)

Calcular el odds ratio de la variable pedigree, así como su intervalo de confianza.

```
exp(coef(modelo)["pedigree"])
```

```
## pedigree
## 3.129611
```

```
exp(confint(modelo,parm="pedigree"))
```

```
## Waiting for profiling to be done...
```

```
##    2.5 %   97.5 %
## 1.378380 7.368273
```

(c)

Calcular el odds ratio y la probabilidad de tener diabetes para el individuo de la observación 9.

```
diabet[9,]
```

```
## # A tibble: 1 x 10
##   pregnant glucose pressure triceps insulin  mass pedigree   age
diabetes
##      <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>    <dbl> <dbl> <chr>
## 1        1     115       70      30      96  34.6    0.529    32 pos
## # ... with 1 more variable: diab_status <fct>
```

```
P9<-(exp(-
10.04+0.08*1+0.04*115+0.01*30+0.07*34.6+1.14*0.529+0.03*32))/(1+(exp(-
10.04+0.08*1+0.04*115+0.01*30+0.07*34.6+1.14*0.529+0.03*32)))
P9
```

```
## [1] 0.2544648
```

```
odd9<-exp((-10.04)+0.08*1+0.04*115+0.01*30+0.07*34.6+1.14*0.529+0.03*32)
odd9
```

```
## [1] 0.3413182
```

(d)

¿Como valoras la bondad de ajuste del modelo? Realizar los contrastes o cálculos que se consideren necesarios. Test de verosimilitud.

```
modelonull<- glm(diab_status~1, family = binomial(), data=diabet)
anova(modelonull, modelo, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: diab_status ~ 1
## Model 2: diab_status ~ pregnant + glucose + pressure + triceps +
insulin +
##     mass + pedigree + age
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       391     498.10
## 2       383     344.02  8   154.08 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
modelonull$deviance-modelo$deviance
```

```
## [1] 154.0766
```

El valor del estadístico es:

```
498.1-344.02
```

```
## [1] 154.08
```

```
anova(modelo, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: diab_status
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                       391     498.10
## pregnant  1   25.063       390     473.03 5.549e-07 ***
## glucose   1   97.533       389     375.50 < 2.2e-16 ***
## pressure  1    1.982       388     373.52 0.1592005
## triceps   1   11.017       387     362.50 0.0009027 ***
## insulin   1    0.000       386     362.50 0.9940331
## mass      1    6.209       385     356.29 0.0127093 *
## pedigree  1    8.744       384     347.55 0.0031058 **
## age       1    3.529       383     344.02 0.0603217 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Test de bondad de ajuste Hosmer-Lemeshow:

```
hoslem.test(diabet$diab_status, fitted(modelo))
```

```
## Warning in Ops.factor(1, y): '-' not meaningful for factors
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  diabet$diab_status, fitted(modelo)
## X-squared = 392, df = 8, p-value < 2.2e-16
```

El valor tan pequeño de p indica falta de ajuste del modelo.

```
pR2(modelo)

## fitting null model for pseudo-r2

##          llh      llhNull           G2     McFadden         r2ML
r2CU
## -172.0106159 -249.0489027  154.0765735    0.3093300    0.3250067
0.4518042

R2 <- 1-modelo$deviance/modelonull$deviance
R2

## [1] 0.30933
```

(e)

Considerar ahora el modelo reducido con las variables pregnant, glucose, mass, pedigree y age.¿Es significativa la variable pregnant? Comparar los dos modelos.

```
modelo2 <-
glm(formula=diab_status~pregnant+glucose+mass+pedigree+age,data=diabet,fa
mily=binomial())
summary(modelo2)

##
## Call:
## glm(formula = diab_status ~ pregnant + glucose + mass + pedigree +
##     age, family = binomial(), data = diabet)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8827  -0.6535  -0.3694   0.6521   2.5814
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -9.992080   1.086866  -9.193  < 2e-16 ***
## pregnant     0.083953   0.055031   1.526 0.127117
## glucose      0.036458   0.004978   7.324 2.41e-13 ***
## mass         0.078139   0.020605   3.792 0.000149 ***
## pedigree     1.150913   0.424242   2.713 0.006670 **
## age          0.034360   0.017810   1.929 0.053692 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 344.89  on 386  degrees of freedom
## AIC: 356.89
##
## Number of Fisher Scoring iterations: 5
```

La variable pregnant no es significativa.

ara evaluar el modelo,se puede comparar el valor real con el predicho.

```
predicciones <- ifelse(test = modelo2$fitted.values > 0.5, yes = 1, no =
0)
matriz_confusion <- table(modelo2$model$diab_status, predicciones,
                          dnn = c("observaciones", "predicciones"))
matriz_confusion
```

```
##              predicciones
## observaciones   0   1
##             0 232  30
##             1  55  75
```

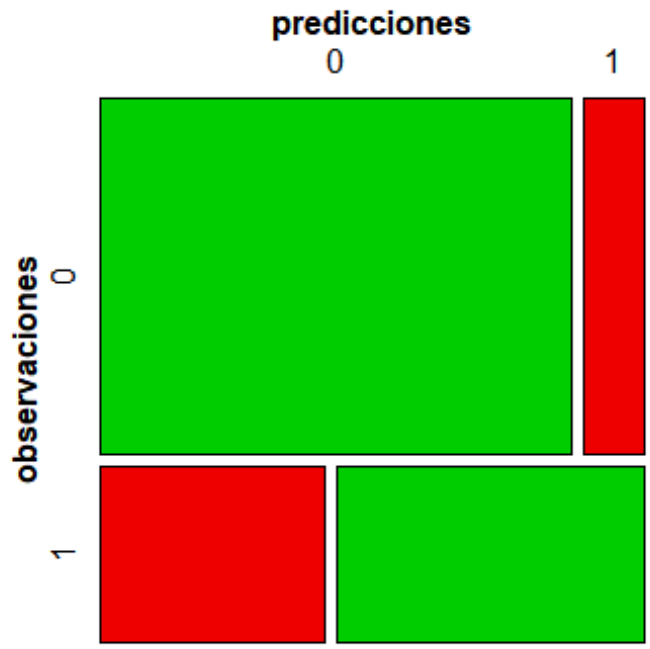
El modelo es capaz de clasificar correctamente :

```
(232+75)/(232+75+30+55)
```

```
## [1] 0.7831633
```

el 78.3% de las observaciones.

```
mosaic(matriz_confusion, shade = T, colorize = T,
       gp = gpar(fill = matrix(c("green3", "red2", "red2", "green3"), 2,
2)))
```



El porcentaje de falsos negativos es bastante alto. Seleccionar otro threshold puede mejorar la exactitud del modelo.

```
predicciones <- ifelse(test = modelo2$fitted.values > 0.4, yes = 1, no =
0)
matriz_confusion <- table(modelo2$model$diab_status, predicciones,
                          dnn = c("observaciones", "predicciones"))
matriz_confusion
```

```
##              predicciones
## observaciones   0   1
##             0 224  38
##             1  40  90
```

Si comparamos los dos modelos:

```
anova(modelo, modelo2, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: diab_status ~ pregnant + glucose + pressure + triceps +
insulin +
##     mass + pedigree + age
## Model 2: diab_status ~ pregnant + glucose + mass + pedigree + age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       383     344.02
## 2       386     344.89 -3  -0.8639   0.8341
```

El modelo2 es aceptable, podemos prescindir de las otras variables.