ProyectoR.R

```
#Proyecto R
#Autor: Araceli Macía Barrado

# Funciones

CalcularMedia <- function (dt, fc){
  #Funcion para calcular la media de un dato en relacion a un dato de tip
o factor
  #si lo que entra no es un dato numerico, no devuelve nada.
  sal <- vector()
  if (is.numeric(dt)){
    sal <- tapply (dt, fc, mean)
  }
  dts <- data.frame(sal)
  #return(sal)
  return(dts)
}
#Funcion para dados dos dataframe, devolver un dataframe con los datos de
A que no
#estan en B
setdiffDF <- function(A, B){
  f <- function(A, B)
    A[!duplicated(rbind(B, A))[nrow(B) + 1:nrow(A)], ]
  df1 <- f(A, B)
  df2 <- f(B, A)
  rbind(df1, df2)
}



#**************************************************************************
#**************************************************************************
#Para el conjunto de datos de diabetes cargar los datos en R.
# El fichero tiene cabecera, lo cargo en mi DataSet de Datos.
# Eliminar los missing values que estan codificados como -9999.00
# Los valores -9999 se convierten en valores NA.
#Con lo que no habrá que tenerlos en cuenta para
# las operaciones.
#**************************************************************************
#**************************************************************************
setwd("~/Documents/3.CIFF/5.ENTORNOS DATA/R/PROYECTO R")
Datos <-  read.table("diabetes.data", header = T, na.strings=c("-", "-999
9.0"))
#Elimino los datos con valores NA, asi no tengo que estar en cada funcion
#añadiendo el parametro de na.rm=True
Datos <- na.omit( Datos )
```

```r
#*****************************************************************************
#*****************************************************************************
#Ver el tipo de cada una de las variables.
#*****************************************************************************
#*****************************************************************************

sapply(Datos,class)
```

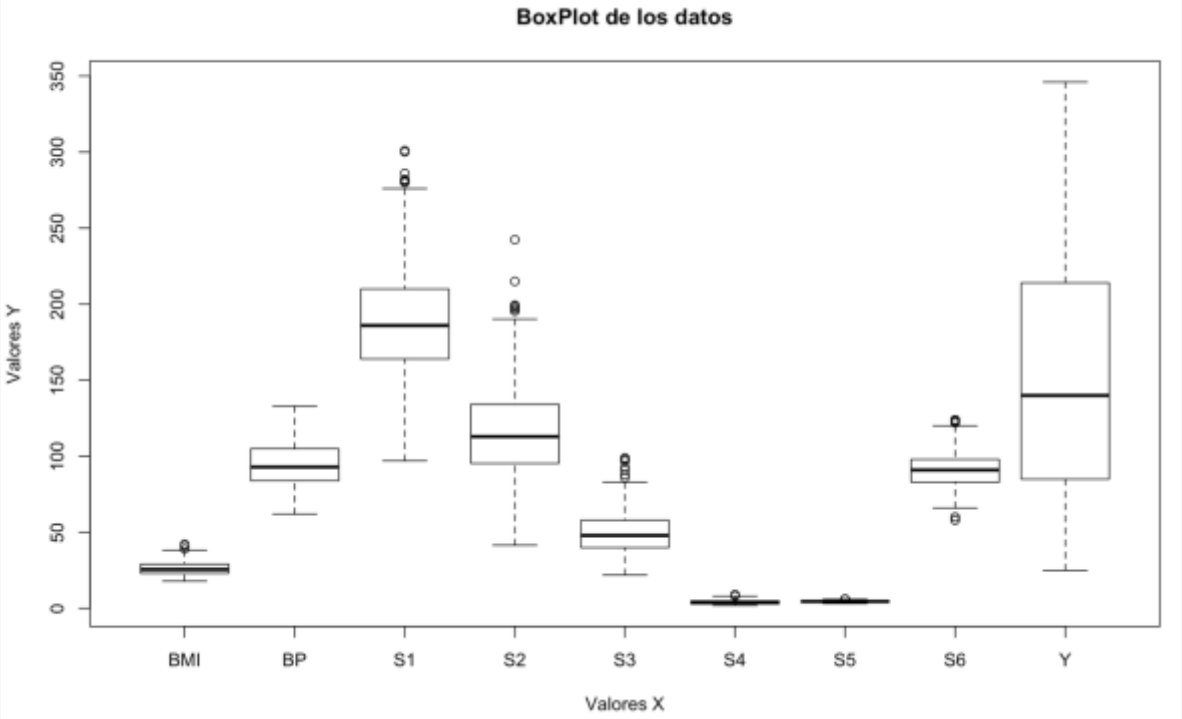
```
##       AGE       SEX       BMI        BP        S1        S2        S3
## "integer"  "factor" "numeric" "numeric" "integer" "numeric" "numeric"
##        S4        S5        S6         Y
## "numeric" "numeric" "integer" "integer"
```

```r
#*****************************************************************************
#*****************************************************************************
#Realizar un analisis estadistico de las variables, calcular la media, va
rianza..
#¿Tienen las distintas variables rangos muy diferentes?.
#*****************************************************************************
#*****************************************************************************

#Con la funcion summary veo el resumen de los estadisticos de las columna
s para poder
#hacer un estudio
summary(Datos)
```

```
##       AGE        SEX          BMI              BP               S1
##  Min.   :19.00   F:203   Min.   :18.00   Min.   : 62.00   Min.   : 97.
0
##  1st Qu.:38.00   M:230   1st Qu.:23.10   1st Qu.: 84.00   1st Qu.:164.
0
##  Median :50.00           Median :25.70   Median : 93.00   Median :186.
0
##  Mean   :48.48           Mean   :26.35   Mean   : 94.65   Mean   :189.
3
##  3rd Qu.:59.00           3rd Qu.:29.20   3rd Qu.:105.00   3rd Qu.:210.
0
##  Max.   :79.00           Max.   :42.20   Max.   :133.00   Max.   :301.
0
##        S2              S3              S4              S5
##  Min.   : 41.6   Min.   :22.00   Min.   :2.000   Min.   :3.258
##  1st Qu.: 95.4   1st Qu.:40.00   1st Qu.:3.000   1st Qu.:4.277
##  Median :113.0   Median :48.00   Median :4.000   Median :4.635
##  Mean   :115.4   Mean   :49.86   Mean   :4.071   Mean   :4.645
##  3rd Qu.:134.2   3rd Qu.:58.00   3rd Qu.:5.000   3rd Qu.:4.997
##  Max.   :242.4   Max.   :99.00   Max.   :9.090   Max.   :6.107
##        S6               Y
##  Min.   : 58.00   Min.   : 25.0
```

```
##  1st Qu.: 83.00   1st Qu.: 85.0
##  Median : 91.00   Median :140.0
##  Mean   : 91.25   Mean   :152.2
##  3rd Qu.: 98.00   3rd Qu.:214.0
##  Max.   :124.00   Max.   :346.0

#visualizo los rangos de las columnas numericas, de BMI a Y.
rangoVal <- sapply(Datos[,3:11], range)
print (rangoVal)

##       BMI  BP  S1    S2 S3   S4     S5  S6   Y
## [1,] 18.0  62  97  41.6 22 2.00 3.2581  58  25
## [2,] 42.2 133 301 242.4 99 9.09 6.1070 124 346

#Los rangos son diferentes. Los unicos que se acercan mas son S4 Y S5 y B
P y S6.
```

```r
#*************************************************************************
#*************************************************************************
#Hacer un grafico de cajas ( Boxplot) donde se pueda ver la informacion a
nterior en
#forma grafica.
#*************************************************************************
#*************************************************************************

#Solo voy a mostrar los datos numericos, que no son la edad ni el Sexo.

boxplot(Datos[,3:11],data=Datos, main="BoxPlot de los datos",
        xlab="Valores X", ylab="Valores Y")
```



#En el grafico se puede ver que los datos que mas se parecen es S4 y S5 Y BP y S6.

```r
#*************************************************************************
#*************************************************************************
#Calcular la media para las filas que tienen SEX=M y la media para las fi
las
#que tienen SEX=F, utilizando la funcíon tapply.
#dd <- tapply(Datos$AGE,Datos$SEX,mean )

sex <- c("F","M")
dtValoresXSexo <- data.frame(sex)

for (i in 1:ncol(Datos)) {
  if (is.numeric(Datos[[i]])){
     sal <- tapply(Datos[[i]],Datos$SEX,mean )
     saldt <- data.frame(sal)
     names(saldt)[1] <- names(Datos)[i]
     dtValoresXSexo <- cbind(dtValoresXSexo,saldt[1])
  }
}



print (t(dtValoresXSexo)) #lo traspongo para verlo mejor al imprimirlo.
```

```
##     F          M
## sex "F"        "M"
## AGE "50.86700" "46.36522"
## BMI "26.80099" "25.95609"
## BP  "98.17562" "91.53474"
## S1  "190.6552" "188.0304"
## S2  "120.1103" "111.1726"
## S3  "44.54187" "54.56304"
## S4  "4.537882" "3.658217"
## S5  "4.731533" "4.569392"
## S6  "93.86207" "88.94348"
## Y   "155.8079" "149.0391"
```

```r
#*************************************************************************
#*************************************************************************
#Calcular la correlación de todas las variables numericas con la variable
#Y.
#*************************************************************************
#*************************************************************************


#Primero genero un DataSet que tenga solamente los valores numericos
columnasTipo <- sapply(Datos, is.numeric)
print (columnasTipo)
```

```
##   AGE   SEX   BMI    BP    S1    S2    S3    S4    S5    S6     Y
##  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```r
#Vuelco al nuevo DataSet solo las columnas de tipo Numerico
DatosNumericos <- Datos[columnasTipo==TRUE]
head(DatosNumericos)
```

```
##   AGE  BMI  BP  S1    S2 S3 S4     S5 S6   Y
## 1  59 32.1 101 157  93.2 38  4 4.8598 87 151
## 2  48 21.6  87 183 103.2 70  3 3.8918 69  75
## 3  72 30.5  93 156  93.6 41  4 4.6728 85 141
## 4  24 25.3  84 198 131.4 40  5 4.8903 89 206
## 5  50 23.0 101 192 125.4 52  4 4.2905 80 135
## 6  23 22.6  89 139  64.8 61  2 4.1897 68  97
```

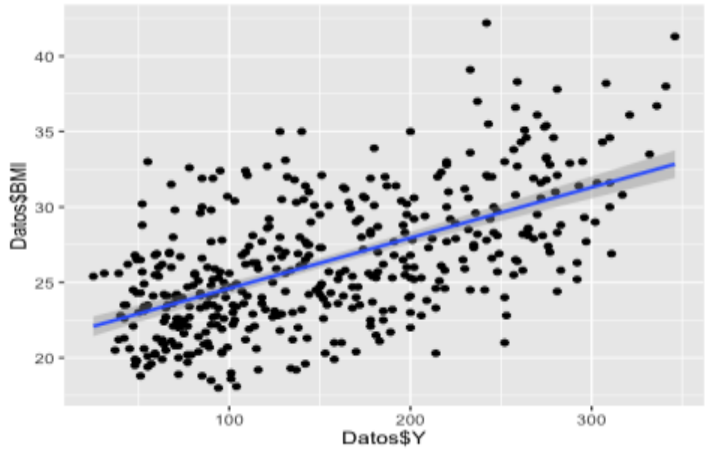
```r
correlacion<- cor(DatosNumericos,DatosNumericos$Y)
print (correlacion)
```

```
##           [,1]
## AGE  0.1889540
## BMI  0.5863673
## BP   0.4398515
## S1   0.2133325
## S2   0.1747189
## S3  -0.3963076
## S4   0.4325640
## S5   0.5703164
## S6   0.3892246
## Y    1.0000000
```

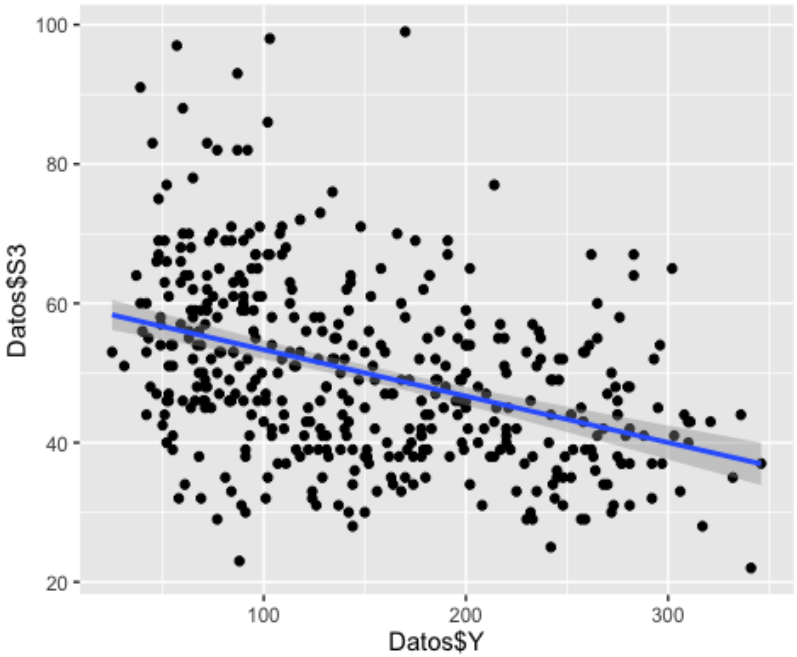
```r
#=======>>>>>>>> El valor maximo es el de BMI y el minimo es S3


#*************************************************************************
#*************************************************************************
#Realizar un grafico de dispersionn para las variables que tienen mas y
#menos correlacion con Y y comentar los resultados.
```

```r
#¿Como seria  el grafico de dispersion entre dos cosas con correlacion 1?
.
#max y min valor
#**********************************************************************
#**********************************************************************

#=======>>>>>>>> El valor maximo es el de BMI y el minimo es S3

library(ggplot2)
```

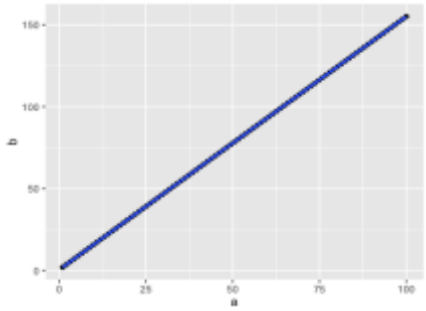
```r
par(mfcol=c(1,2))

p <- ggplot(Datos, aes(x=Datos$Y, y=Datos$BMI))
p + geom_point() + geom_smooth(method = 'lm', formula=y~x, aes(group=1))
```




```r
#====> En este caso, el valor de correlacion es 0.5863673,  cuanto mayor es el dato de Y mayor es BMI.

y <- ggplot(Datos, aes(x=Datos$Y, y=Datos$S3))
y + geom_point() + geom_smooth(method = 'lm', formula=y~x, aes(group=1))
```




```
#=====>En este caso, el valor de correlacion es -0,3963,  cuanto mayor es el dato de Y menor es BMI.

#**************************
#¿Como seria  el grafico de dispersion entre dos cosas con correlacion 1?
.
a <- seq(1,100,1.1)
b <- seq(2,156,1.7)
c <- cor(a, b) #Dos variables con correlacion = 1
d <-data.frame(a,b)

z <- ggplot(d, aes(a, b))
z + geom_point() + geom_smooth(method = 'lm', formula=y~x, aes(group=1))
```



```
# =====>>>>  No habria dispersion, seria una recta con una inclinacion de 45 grados.
```

```r
#*************************************************************************
#*************************************************************************
#Transformar la variable SEX, que es un factor, en una variable numerica
#utilizando, por ejemplo, la codificacion M=1 y F=2.
#*************************************************************************
#*************************************************************************

Datos$SEX <- as.numeric(Datos$SEX)
table(Datos$SEX) #Ha quedado como F:1 y M:2, lo cambiamos

##
##   1   2
## 203 230

library(car)
Datos$SEX <- recode(Datos$SEX, "1=4;2=5;5=1;4=2")
Datos$SEX <- recode(Datos$SEX, "5=1;4=2")

table(Datos$SEX) #Ha quedado como M:1 y F:2

##
##   1   2
## 230 203

sapply(Datos,class) #Ya tengo todas las columnas numericas

##       AGE       SEX       BMI        BP        S1        S2        S3
## "integer" "numeric" "numeric" "numeric" "integer" "numeric" "numeric"
##        S4        S5        S6         Y
## "numeric" "numeric" "integer" "integer"

#*************************************************************************
#*************************************************************************
#Definimos los outliers como los elementos (filas) de los datos para
#los que cualquiera de las variables esta  por encima o por debajo de
#la mediana mas/menos 3 veces el MAD (Median Absolute Deviation)
#Identificar estos outliers y quitarlos
#*************************************************************************
#*************************************************************************

#Utilizar Funcion mad(x, center, constant = 1.4826, na.rm = FALSE)
cat ("DataSet inicial tiene :", nrow(Datos), " filas\n")

## DataSet inicial tiene : 433  filas

for (i in 3:ncol(Datos)) {  #las columnas de edad y el Sexo no lo conside
ro.
  indBajo <- mean(Datos[[i]])-(3*mad(Datos[[i]]))
  indSuperior <- mean(Datos[[i]])+(3*mad(Datos[[i]]))
  Outlier <- Datos[ (Datos[[i]]<indBajo) | (Datos[[i]]> indSuperior) ,]
```

```
  cat ("Columna :" , names(Datos)[i]," IndBajo:", indBajo, "   IndSup:",
indSuperior, "\n")
  cat ("Outlier, numero de filas encontradas :" , nrow(Outlier), "\n")
      #Asigno valor NA a los datos que son OutLier y luego los elimino de
l DataSet.
  Datos[ (Datos[[i]]<indBajo) | (Datos[[i]]> indSuperior) ,] <- NA
  Datos <- na.omit( Datos )

}
```

```
## Columna : BMI  IndBajo: 13.45357    IndSup: 39.25081
## Outlier, numero de filas encontradas : 2
## Columna : BP  IndBajo: 50.1917    IndSup: 139.1477
## Outlier, numero de filas encontradas : 0
## Columna : S1  IndBajo: 86.96046    IndSup: 291.5593
## Outlier, numero de filas encontradas : 2
## Columna : S2  IndBajo: 29.41553    IndSup: 200.211
## Outlier, numero de filas encontradas : 0
## Columna : S3  IndBajo: 9.917352    IndSup: 89.97775
## Outlier, numero de filas encontradas : 5
## Columna : S4  IndBajo: -0.3759368    IndSup: 8.519663
## Outlier, numero de filas encontradas : 0
## Columna : S5  IndBajo: 3.011581    IndSup: 6.27849
## Outlier, numero de filas encontradas : 0
## Columna : S6  IndBajo: 60.0588    IndSup: 122.328
## Outlier, numero de filas encontradas : 8
## Columna : Y  IndBajo: -115.6613    IndSup: 418.0747
## Outlier, numero de filas encontradas : 0
```

```
cat ("DataSet final tiene :", nrow(Datos), " filas\n")
```

```
## DataSet final tiene : 416  filas
```

```
#**************************************************************************
#**************************************************************************
#Separar el conjunto de datos en dos, el primero (entrenamiento) contenie
ndo un 70%
#de los datos y el segundo (test) un 30%, de forma aleatoria.
#**************************************************************************
#**************************************************************************
#Calculo el numero de filas que son el 70% de los datos.
numFilasEntrenamiento <- round((nrow(Datos)*70)/100,0)
#Genero un dataset cogiendo los valores aleatoriamente
DatosEntrenamiento <-Datos[sample(1:nrow(Datos), numFilasEntrenamiento, r
eplace=FALSE),]
#Utilizo funcion para que me devuelva el resultado de quitar al DT princi
pal el de entrenamiento
DatosTest <-setdiffDF(Datos,DatosEntrenamiento)

cat ("Se ha genarado DataSet de Entrenamiento con", nrow(DatosEntrenamien
to), "filas \n")
```

```r
## Se ha genarado DataSet de Entrenamiento con 291 filas

cat ("Se ha genarado DataSet de Test con", nrow(DatosTest), " filas \n")

## Se ha genarado DataSet de Test con 125  filas

#**************************************************************************
#**************************************************************************
#Escalar los datos para que tengan media 0 y varianza 1, es decir,
#restar a cada variable numerica su media y dividir por la desviacion tip
ica
#Calcular la media y desviacion en el conjunto de train,
#y utilizar esa misma media y desviacion para escalar el conjunto de test
#**************************************************************************
#**************************************************************************

vectorMedia <- sapply(DatosEntrenamiento[,3:11],mean)
vectorSd <- sapply(DatosEntrenamiento[,3:11],sd)
print (vectorMedia)

##        BMI         BP         S1         S2         S3         S4
##  26.184536  94.660928 188.714777 114.752577  49.505155   4.060481
##         S5         S6          Y
##   4.655654  91.096220 152.615120

print (vectorSd)

##        BMI         BP         S1         S2         S3         S4
##  4.1592276 13.9428098 33.8829286 29.2441108 12.1253144  1.2746661
##         S5         S6          Y
##  0.5340362 10.4978332 77.2652595

for (i in 1:nrow(DatosTest))
   DatosTest[i,3:11] <- (DatosTest[i,3:11]-vectorMedia)/vectorSd

vectorMedia <- sapply(DatosTest[,3:11],mean)
vectorSd <- sapply(DatosTest[,3:11],sd)
print (round(vectorMedia))

## BMI  BP  S1  S2  S3  S4  S5  S6   Y
##   0   0   0   0   0   0   0   0   0

print (round(vectorSd))

## BMI  BP  S1  S2  S3  S4  S5  S6   Y
##   1   1   1   1   1   1   1   1   1
```

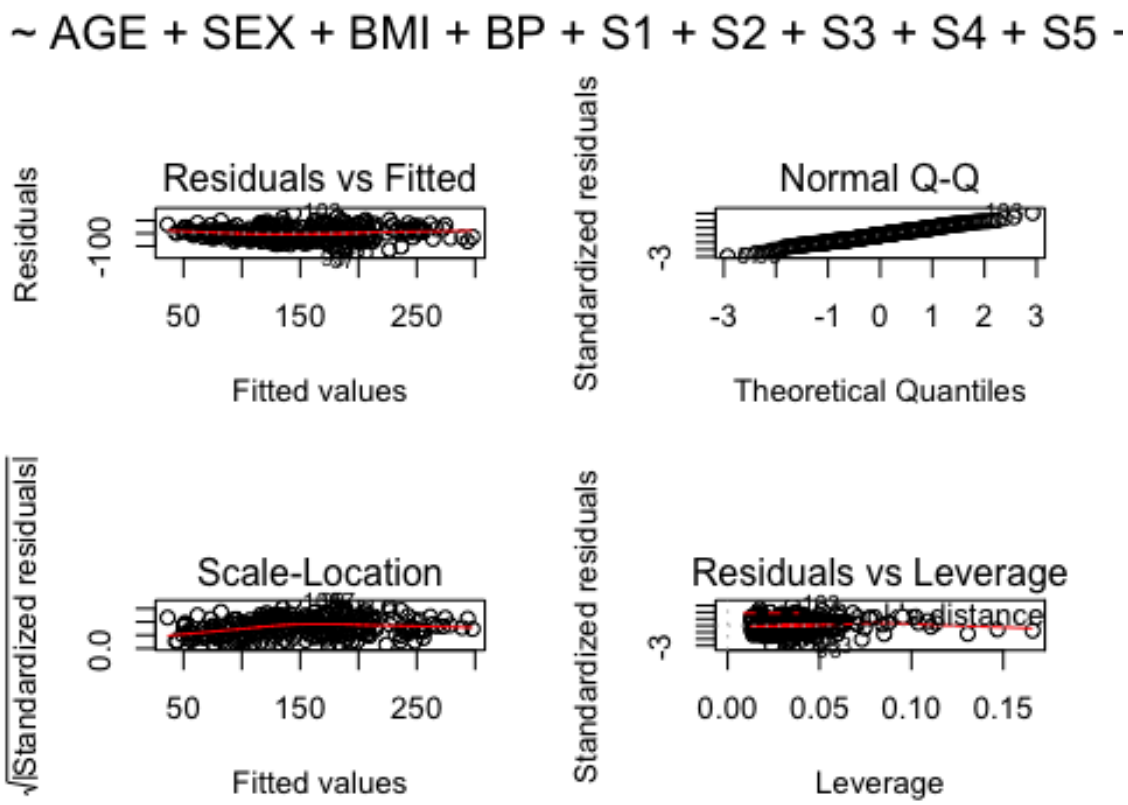
```r
#*************************************************************************
#*************************************************************************
#(Opcional) Realizar un modelo de regresion lineal de la variable de resp
uesta sobre el
#resto y ajustarlo por minimos cuadrados usando
#unicamente los datos del conjunto de entrenamiento.
#*************************************************************************
#*************************************************************************
head(DatosEntrenamiento)
```

```
##     AGE SEX  BMI     BP  S1    S2 S3   S4     S5  S6   Y
## 228  67   2 23.6 111.33 189 105.4 70 2.70 4.2195  93 108
## 376  61   2 26.1 126.00 215 129.8 57 4.00 4.9488  96 217
## 286  52   1 24.5  90.00 198 129.0 29 7.00 5.2983  86 233
## 361  53   1 28.3 101.00 179 107.0 48 4.00 4.7875 101 281
## 409  66   1 21.7 126.00 212 127.8 45 4.71 5.2781 101 189
## 272  59   2 27.2 107.00 158 102.0 39 4.00 4.4427  93 127
```

```r
regresion <- lm(Y ~ AGE + SEX + BMI + BP + S1 + S2 + S3 + S4+ S5 + S6, da
ta=DatosEntrenamiento)
summary (regresion)
```

```
##
## Call:
## lm(formula = Y ~ AGE + SEX + BMI + BP + S1 + S2 + S3 + S4 + S5 +
##     S6, data = DatosEntrenamiento)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -153.834  -36.846    1.079   35.222  157.728
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -374.32721   83.31861  -4.493 1.03e-05 ***
## AGE            0.05766    0.26267   0.219 0.826422
## SEX          -22.25181    6.91041  -3.220 0.001433 **
## BMI            5.42299    0.93316   5.811 1.68e-08 ***
## BP             1.40725    0.26339   5.343 1.90e-07 ***
## S1            -1.48568    0.66172  -2.245 0.025538 *
## S2             1.11675    0.60166   1.856 0.064487 .
## S3             0.68126    0.97446   0.699 0.485060
## S4             8.93262    7.28612   1.226 0.221237
## S5            73.73669   18.84997   3.912 0.000115 ***
## S6             0.22648    0.35881   0.631 0.528433
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.79 on 280 degrees of freedom
## Multiple R-squared:  0.5493, Adjusted R-squared:  0.5332
## F-statistic: 34.12 on 10 and 280 DF,  p-value: < 2.2e-16
```

```r
oldpar <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(regresion)
```




```r
par(oldpar)
```

```r
#**************************************************************************
#**************************************************************************
#(Opcional) Calcular el error cuadratico medio de los datos del conjunto
de entrenamiento y
# de los datos del conjunto del test.
#**************************************************************************
#**************************************************************************
yPredict=predict(regresion)
ecm=(mean(DatosEntrenamiento$Y-yPredict))^2
cat ("El error cuadratico para los datos de entrenamiento :", ecm , "\n")

## El error cuadratico para los datos de entrenamiento : 8.569833e-25

regresionTest <- lm(Y ~ AGE + SEX + BMI + BP + S1 + S2 + S3 + S4+ S5 + S6
, data=DatosTest)
yPredictT=predict(regresionTest)
ecmtest=(mean(DatosTest$Y-yPredictT))^2
cat ("El error cuadratico para los datos de test :", ecmtest , "\n")

## El error cuadratico para los datos de test : 2.568342e-30
```