Tipologia y ciclo de vida de los datos - PRA2

Autor: Eduardo Diaz Villanueva e Ignasi Domingo González

Enero 2021

Contents

Descripción del dataset.

```
library(stringr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
# Limpiamos la aplicacion de datos anteriores y cargo el fichero.
rm(list = ls())
datos <- read.csv("winequality-red.csv", sep=",")
datos_originales <- datos
#shape(datos)
```

```
#describe(datos)
head(datos,5)
```

```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.4             0.70        0.00            1.9     0.076
## 2           7.8             0.88        0.00            2.6     0.098
## 3           7.8             0.76        0.04            2.3     0.092
## 4          11.2             0.28        0.56            1.9     0.075
## 5           7.4             0.70        0.00            1.9     0.076
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  11                   34  0.9978 3.51      0.56     9.4
## 2                  25                   67  0.9968 3.20      0.68     9.8
## 3                  15                   54  0.9970 3.26      0.65     9.8
## 4                  17                   60  0.9980 3.16      0.58     9.8
## 5                  11                   34  0.9978 3.51      0.56     9.4
##   quality
## 1       5
## 2       5
## 3       5
## 4       6
## 5       5
```

El dataset seleccionado contiene 11 variables que describen las propiedades químicas de un vino, como puede ser la acidez, ph nivel de azúcar, etc... estas variables tendrán influencia en la calidad final del vino.

Con este ejercicio queremos estudiar que variables son mas representativas y encontrar modelos que puedan predecir la calidad del vino.

Si pensamos por ejemplo en una industria, podríamos reducir el tiempo y coste reduciendo el numero de pruebas de calidad a las variables mas significativas. Incluso mejorar la calidad del producto final, focalizando esfuerzos y recursos a reducir la variabilidad de las variables que mas contribuyan a la calidad final.

Integracion y seleccion de los datos de interes

Realizaremos un primer análisis estadístico para familiarizarnos con las variables y sus tipos de datos.

```
summary(datos)
```

```
##  fixed.acidity   volatile.acidity  citric.acid    residual.sugar
##  Min.   : 4.60   Min.   :0.1200   Min.   :0.000   Min.   : 0.900
##  1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900
##  Median : 7.90   Median :0.5200   Median :0.260   Median : 2.200
##  Mean   : 8.32   Mean   :0.5278   Mean   :0.271   Mean   : 2.539
##  3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600
##  Max.   :15.90   Max.   :1.5800   Max.   :1.000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 22.00       1st Qu.:0.9956
##  Median :0.07900   Median :14.00       Median : 38.00       Median :0.9968
##  Mean   :0.08747   Mean   :15.87       Mean   : 46.47       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 62.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :72.00       Max.   :289.00       Max.   :1.0037
##        pH          sulphates         alcohol         quality
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.40   Min.   :3.000
##  1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
```

```
##  Median :3.310   Median :0.6200   Median :10.20   Median :6.000
##  Mean   :3.311   Mean   :0.6581   Mean   :10.42   Mean   :5.636
##  3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

```
str(datos)
```

```
## 'data.frame':    1599 obs. of  12 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...
##  $ residual.sugar      : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...
##  $ chlorides           : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...
##  $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 17 ...
##  $ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 102 ...
##  $ density             : num  0.998 0.997 0.997 0.998 0.998 ...
##  $ pH                  : num  3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36 3.35 ...
##  $ sulphates           : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...
##  $ alcohol             : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...
##  $ quality             : int  5 5 5 6 5 5 5 7 7 5 ...
```

```
#Tipo de dato asignado a cada campo
sapply(datos, function(x) class(x))
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##            "numeric"            "numeric"            "numeric"
##       residual.sugar           chlorides  free.sulfur.dioxide
##            "numeric"            "numeric"            "numeric"
## total.sulfur.dioxide              density                   pH
##            "numeric"            "numeric"            "numeric"
##            sulphates              alcohol              quality
##            "numeric"            "numeric"            "integer"
```

Observamos que los tipos de datos asignados a las variables corresponden con las variables que representan.
Limpieza de los datos

Elementos vacios

Analizamos los valores de las variables para detectar falta o ausencia de datos

```
# Analizamos la existencia de datos NA
colSums(is.na(datos))
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##                    0                    0                    0
##       residual.sugar           chlorides  free.sulfur.dioxide
##                    0                    0                    0
## total.sulfur.dioxide              density                   pH
##                    0                    0                    0
##            sulphates              alcohol              quality
##                    0                    0                    0
```

```
# Analizamos la existencia de datos vacios
colSums(datos=="")
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##                    0                    0                    0
##       residual.sugar           chlorides  free.sulfur.dioxide
```

```
##                    0                    0                    0
## total.sulfur.dioxide              density                   pH
##                    0                    0                    0
##            sulphates              alcohol              quality
##                    0                    0                    0
```

```
# Analizamos la existencia de datos con valor 0
colSums(datos==0)
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##                    0                    0                  132
##       residual.sugar            chlorides  free.sulfur.dioxide
##                    0                    0                    0
## total.sulfur.dioxide              density                   pH
##                    0                    0                    0
##            sulphates              alcohol              quality
##                    0                    0                    0
```

Observamos la variable "Citric.acid" con una gran cantidad de valores 0. En la uva, el ácido cítrico es un componente que presente pero menor, con muy baja presencia, lo que si no se añade posteriormente puede presentar valores de 0. Consideramos que dicha variable tiene valores correctos.

Elementos duplicados

Dado que no hay presentes registros con elementos vacios, vamos a verificar si hay registros duplicados.

```
# Analizamos la existencia de registros duplicados.
sum(duplicated(datos))
```

```
## [1] 240
```

Algunas filas están duplicadas. A modo de ejemplo, la fila 1 y la fila 5 son la misma.

```
# Ejemplo de registro duplicado
a <- datos %>% filter(row_number() == 1)
b <- datos %>% filter(row_number() == 5)
a == b
```

```
##      fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## [1,]          TRUE             TRUE        TRUE           TRUE      TRUE
##      free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## [1,]                TRUE                 TRUE    TRUE TRUE      TRUE    TRUE
##      quality
## [1,]    TRUE
```

Eliminamos las filas duplicadas

```
# Eliminamos los registros duplicados
datos <- datos[!duplicated(datos), ]
```

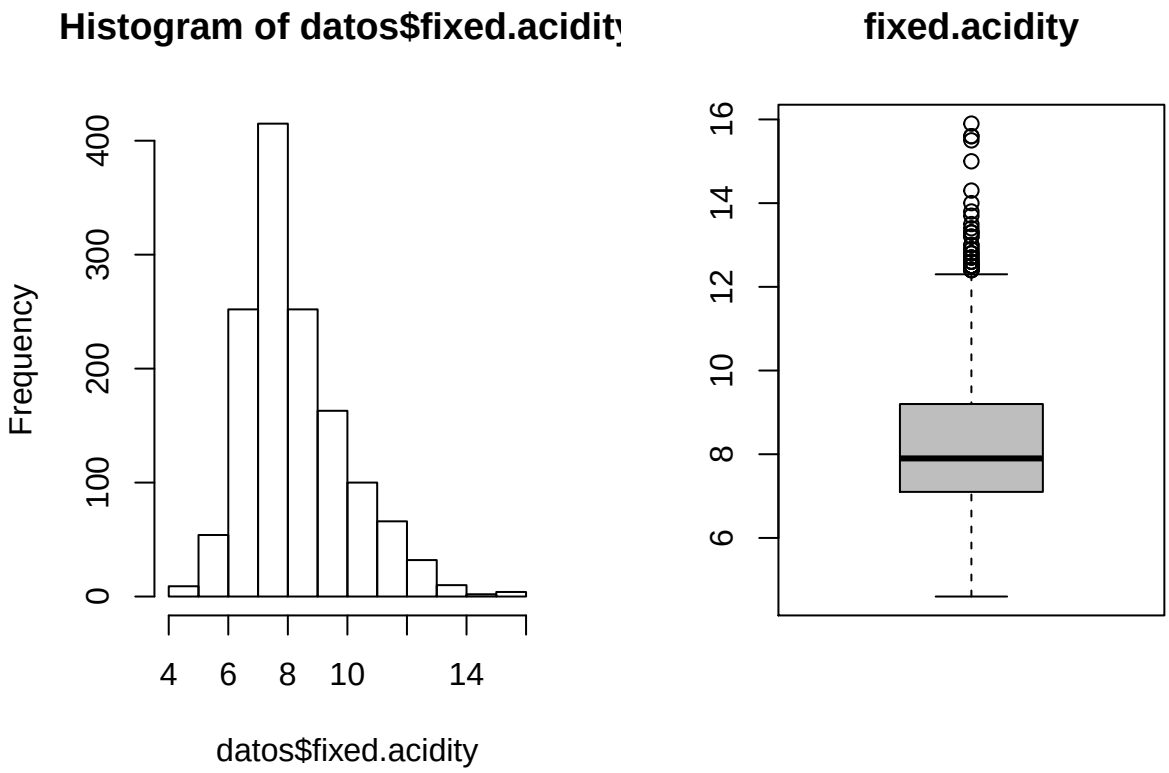
```
# Revisamos el tamaño de nuestro nuevo dataset
dim(datos)
```

```
## [1] 1359   12
```

Valores extremos

Analizaremos individualmente cada una de las variables focalizándonos en la distribución de los datos y sus valores extremos.

```
par(mfrow=c(1,2))
hist(datos$fixed.acidity)
boxplot(datos$fixed.acidity,main="fixed.acidity", col="gray")
```

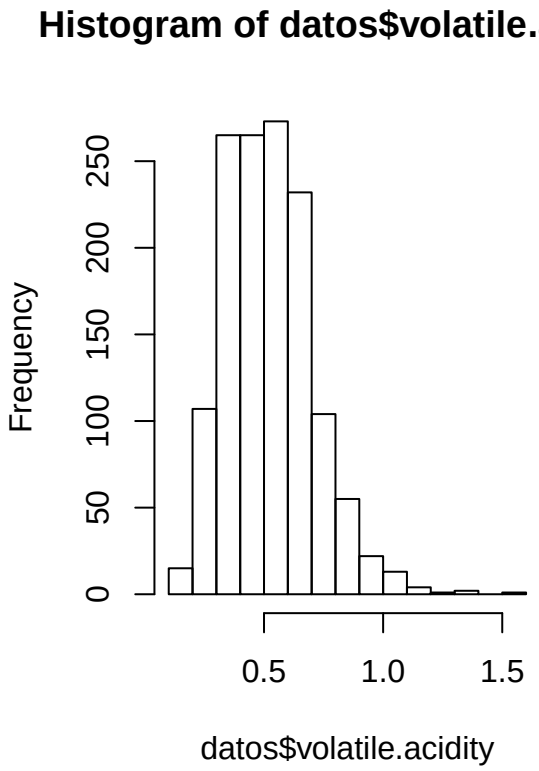


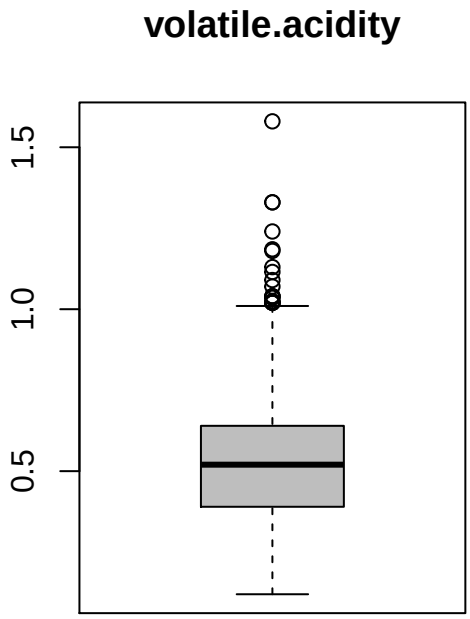

```
boxplot.stats(datos$fixed.acidity)$out
```

```
##  [1] 12.8 15.0 12.5 13.3 13.4 12.4 12.5 13.8 13.5 12.6 12.5 12.8 14.0 13.7 12.7
## [16] 12.5 12.8 12.6 15.6 12.5 13.0 12.5 13.3 12.4 12.5 12.9 14.3 12.4 15.5 15.6
## [31] 13.0 12.7 12.4 12.7 13.2 13.2 15.9 13.3 12.9 12.6 12.6
```

Observamos como el atributo “fixed.acidity” tiene 41 valores extremos, distribuidos entre 12.4 y 16

```
par(mfrow=c(1,2))
hist(datos$volatile.acidity)
boxplot(datos$volatile.acidity,main="volatile.acidity", col="gray")
```



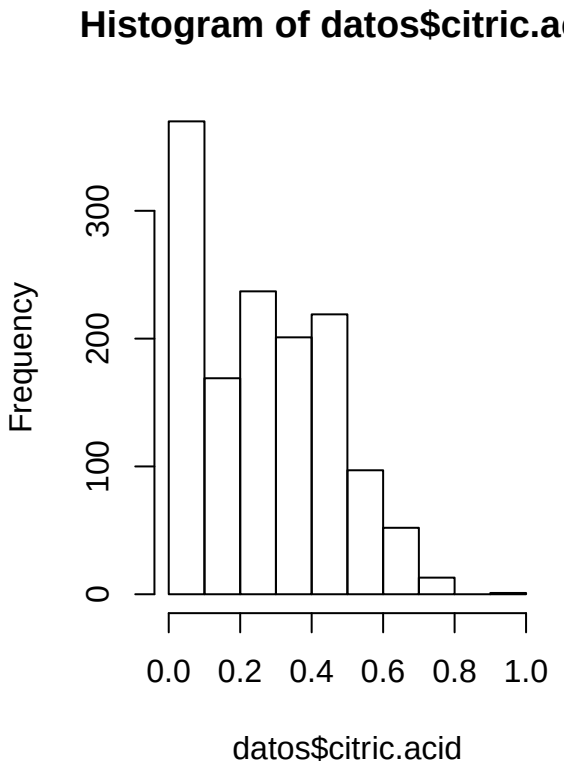


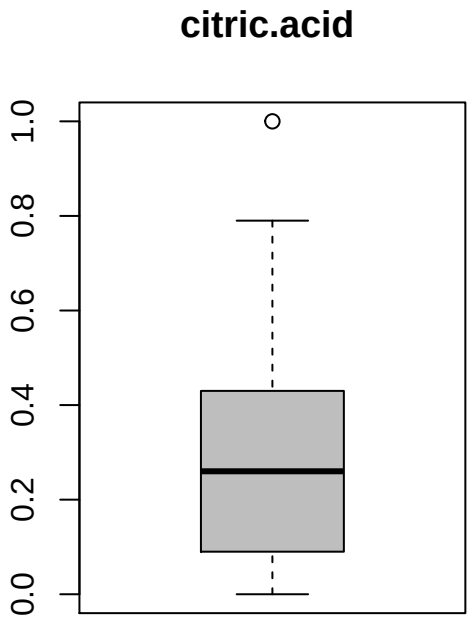

```
boxplot.stats(datos$volatile.acidity)$out
```

```
##  [1] 1.130 1.020 1.070 1.330 1.330 1.040 1.090 1.040 1.240 1.185 1.020 1.035
## [13] 1.025 1.115 1.020 1.020 1.580 1.180 1.040
```

Observamos como el atributo “volatile.acidity” tiene 19 valores extremos, distribuidos entre 1 y 1.6

```
par(mfrow=c(1,2))
hist(datos$citric.acid )
boxplot(datos$citric.acid ,main="citric.acid ", col="gray")
```



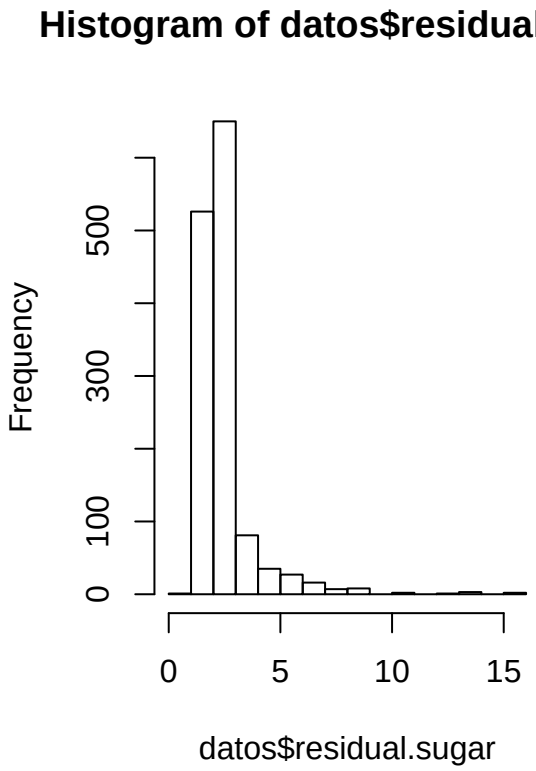


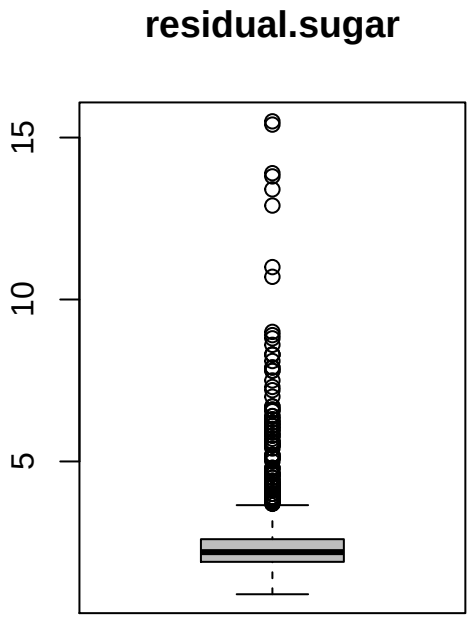

```
boxplot.stats(datos$citric.acid )$out
```

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

Observamos como el atributo “citric.acid” tiene un valor extremo de valor 1.

```
par(mfrow=c(1,2))
hist(datos$residual.sugar)
boxplot(datos$residual.sugar,main="residual.sugar", col="gray")
```



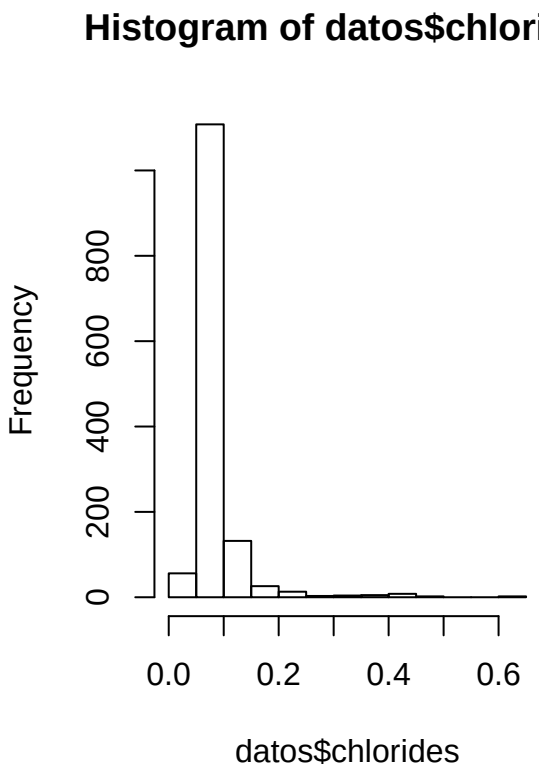


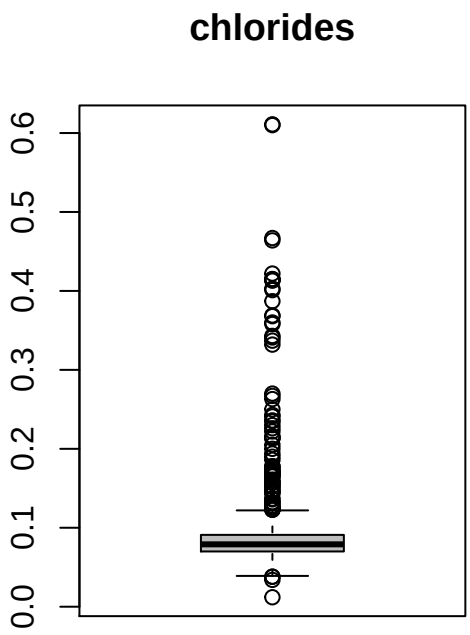

```
boxplot.stats(datos$residual.sugar)$out
```

```
##   [1]  6.10  3.80  3.90  4.40 10.70  5.50  5.90  3.80  5.10  4.65  5.50  5.50
##  [13]  7.30  7.20  3.80  5.60  4.00  4.00  4.00  7.00  6.40  5.60 11.00  4.50
##  [25]  4.80  5.80  3.80  4.40  6.20  4.20  7.90  3.70  4.50  6.70  6.60  3.70
##  [37]  5.20 15.50  4.10  8.30  6.55  4.60  6.10  4.30  5.80  5.15  6.30  4.20
##  [49]  4.60  4.20  4.30  7.90  4.60  5.10  5.60  6.00  8.60  7.50  4.40  4.25
##  [61]  6.00  3.90  4.20  4.00  4.00  6.60  6.00  3.80  9.00  4.60  8.80  5.00
##  [73]  3.80  4.10  5.90  4.10  6.20  8.90  4.00  3.90  8.10  6.40  8.30  8.30
##  [85]  4.70  5.50  4.30  5.50  3.70  6.20  5.60  7.80  4.60  5.80  4.10 12.90
##  [97]  4.30 13.40  4.80  6.30  4.50  4.30  3.90  3.80  5.40  3.80  6.10  3.90
## [109]  5.10  3.90 15.40  4.80  5.20  5.20  3.75 13.80  5.70  4.30  4.10  4.10
## [121]  4.40  3.70  6.70 13.90  5.10  7.80
```

Observamos como el atributo “residual.sugar” tiene 126 valores extremos, distribuidos entre 4 y 16

```
par(mfrow=c(1,2))
hist(datos$chlorides)
boxplot(datos$chlorides,main="chlorides", col="gray")
```



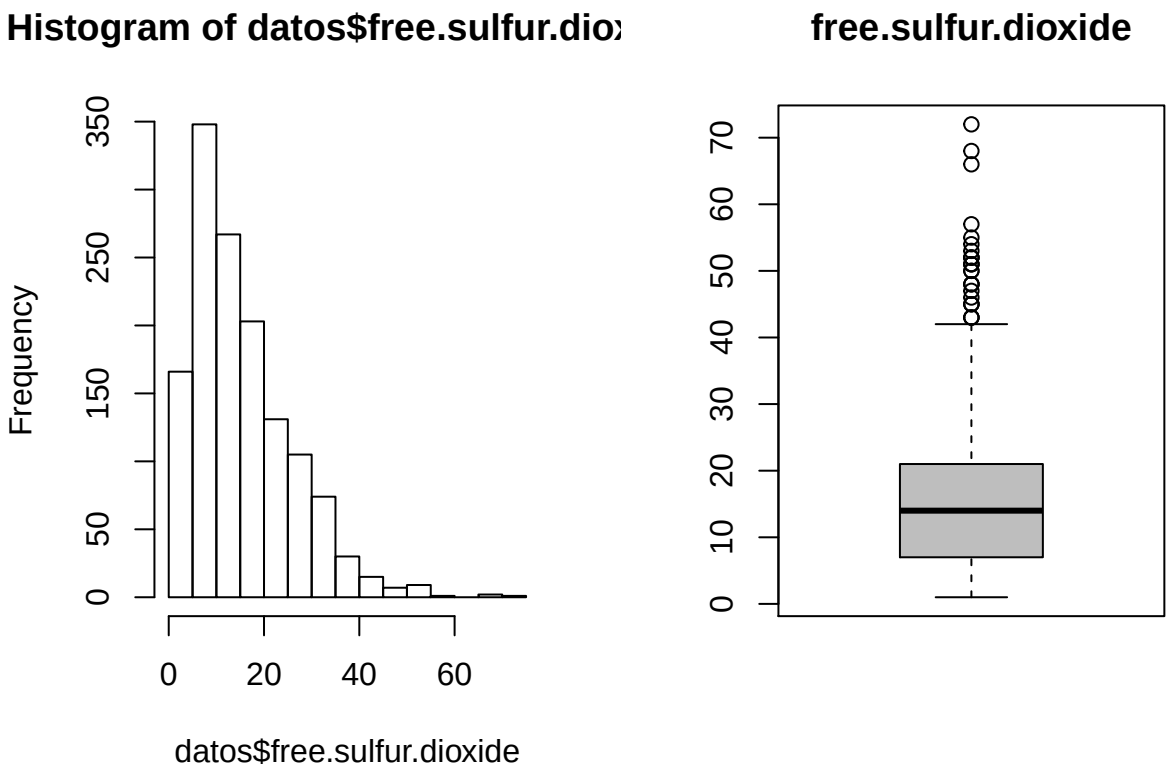



```
boxplot.stats(datos$chlorides)$out
```

```
##  [1] 0.176 0.170 0.368 0.341 0.172 0.332 0.464 0.401 0.467 0.178 0.146 0.236
## [13] 0.610 0.360 0.270 0.337 0.263 0.611 0.358 0.343 0.186 0.213 0.214 0.128
## [25] 0.159 0.124 0.174 0.127 0.413 0.152 0.152 0.125 0.200 0.171 0.226 0.250
## [37] 0.148 0.124 0.143 0.222 0.157 0.422 0.034 0.387 0.415 0.157 0.157 0.243
## [49] 0.241 0.190 0.132 0.126 0.038 0.165 0.145 0.147 0.012 0.194 0.132 0.161
## [61] 0.123 0.414 0.216 0.171 0.178 0.369 0.166 0.166 0.136 0.132 0.123 0.123
## [73] 0.403 0.137 0.414 0.166 0.168 0.415 0.153 0.267 0.123 0.214 0.169 0.205
## [85] 0.235 0.230 0.038
```

Observamos como el atributo “chlorides” tiene 87 valores extremos, distribuidos entre 0 y 0.05 por la parte inferior y entre 0.12 y 0.6 por la parte superior.

```
par(mfrow=c(1,2))
hist(datos$free.sulfur.dioxide)
boxplot(datos$free.sulfur.dioxide,main="free.sulfur.dioxide", col="gray")
```

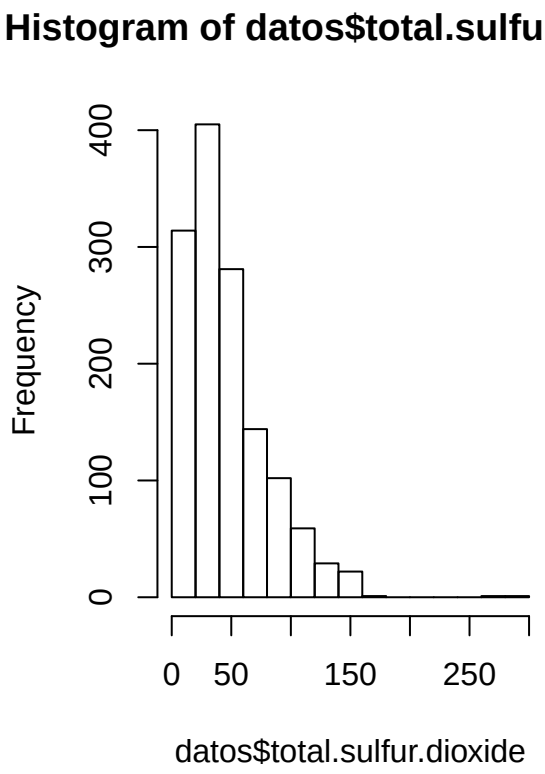


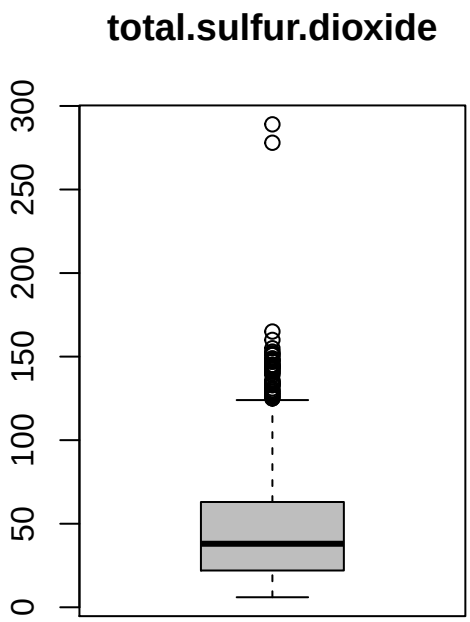

```
boxplot.stats(datos$free.sulfur.dioxide)$out
```

```
##  [1] 52 51 50 68 43 47 54 46 45 53 52 51 45 57 50 45 48 43 48 72 43 51 52 55 48
## [26] 66
```

Observamos como el atributo “free.sulfur.dioxide” tiene 26 valores extremos, distribuidos entre el 43 y el 60

```
par(mfrow=c(1,2))
hist(datos$total.sulfur.dioxide)
boxplot(datos$total.sulfur.dioxide,main="total.sulfur.dioxide", col="gray")
```



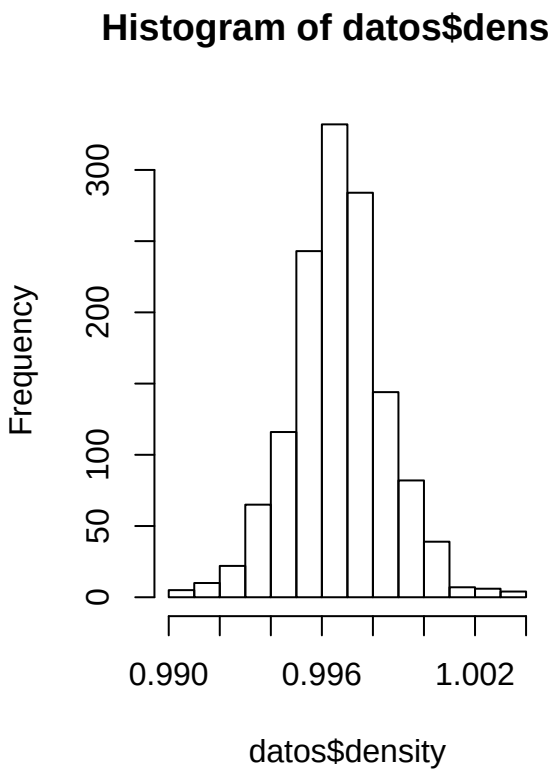


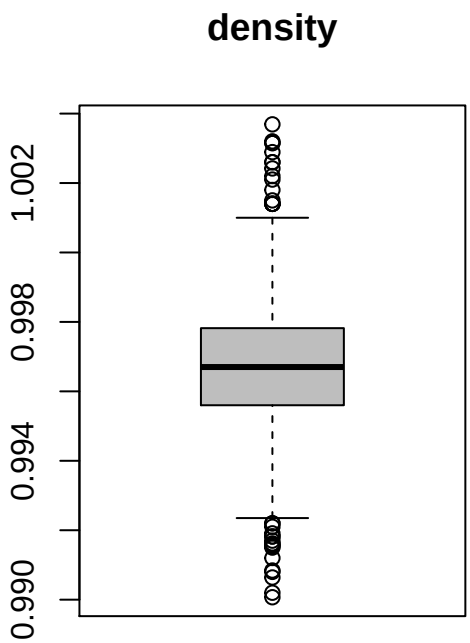

```
boxplot.stats(datos$total.sulfur.dioxide)$out
```

```
##  [1] 145 148 136 125 140 133 153 134 141 129 128 143 144 127 126 145 144 135 165
## [20] 134 129 151 133 142 149 147 145 148 155 151 152 125 127 139 143 144 130 278
## [39] 289 135 160 141 133 147 131
```

Observamos como el atributo "total.sulfur.dioxide" tiene 45 valores extremos, distribuidos entre el 120 y el 300.

```
par(mfrow=c(1,2))
hist(datos$density)
boxplot(datos$density,main="density", col="gray")
```



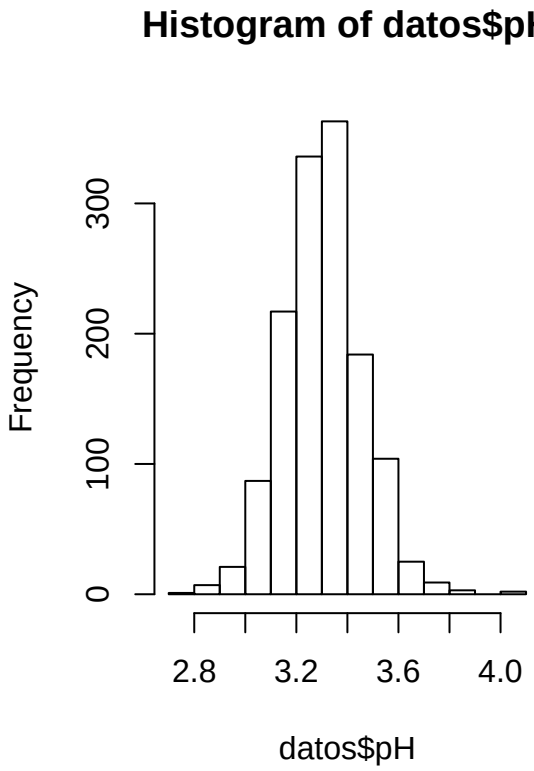


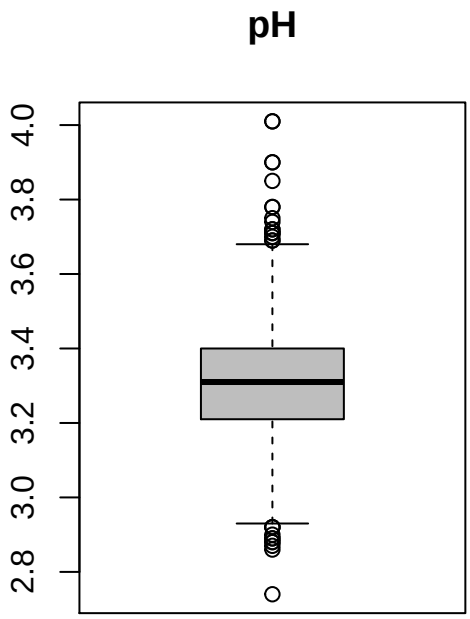

```
boxplot.stats(datos$density)$out
```

```
##  [1] 0.99160 1.00140 1.00150 1.00180 0.99120 1.00220 1.00140 1.00140 1.00140
## [10] 1.00320 1.00260 1.00140 1.00315 1.00315 1.00210 0.99170 0.99220 1.00260
## [19] 0.99210 0.99154 0.99064 1.00289 0.99162 0.99007 0.99020 0.99220 0.99150
## [28] 0.99157 0.99080 0.99084 0.99191 1.00369 1.00242 0.99182 0.99182
```

Observamos como el atributo “density” tiene 35 valores extremos, distribuidos entre 0.990 y 0.992 por la parte inferior y entre 1.001 y 1.004 por la parte superior.

```
par(mfrow=c(1,2))
hist(datos$pH)
boxplot(datos$pH,main="pH", col="gray")
```



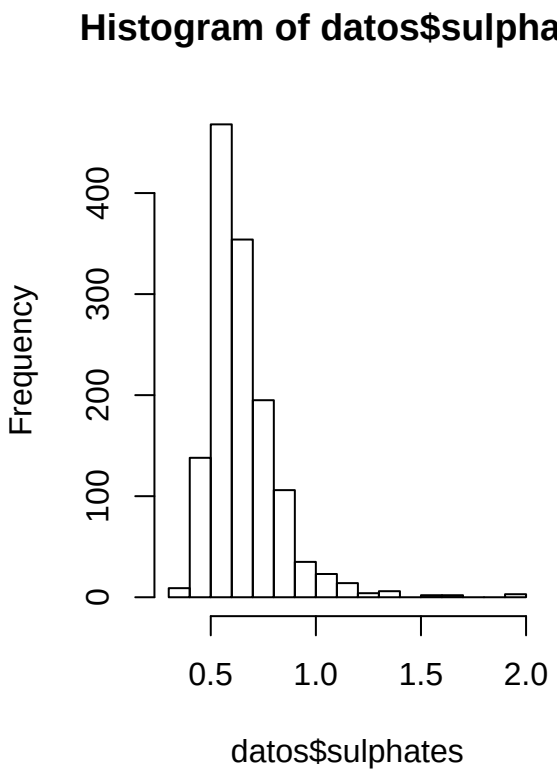


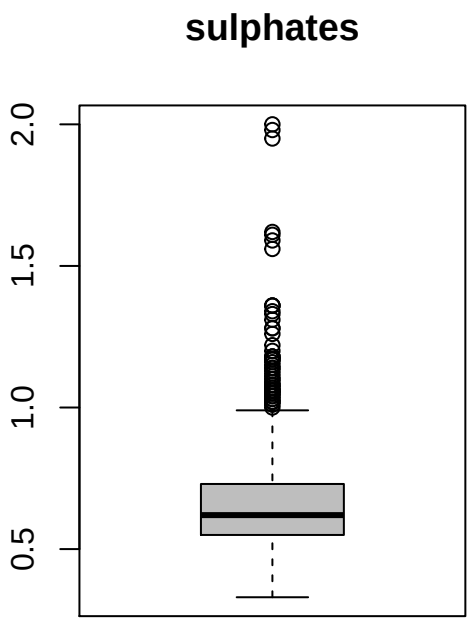

```
boxplot.stats(datos$pH)$out
```

```
##  [1] 3.90 3.75 3.85 2.74 3.69 2.88 2.86 3.74 2.92 2.92 3.72 2.87 2.89 2.92 3.90
## [16] 3.71 3.69 3.71 2.89 3.78 3.70 3.78 4.01 2.90 4.01 3.71 2.88 3.72
```

Observamos como el atributo “pH” tiene 28 valores extremos, distribuidos entre 2.7 y 2.9 por la parte inferior y entre 3.7 y 4 por la parte superior.

```
par(mfrow=c(1,2))
hist(datos$sulphates)
boxplot(datos$sulphates,main="sulphates", col="gray")
```



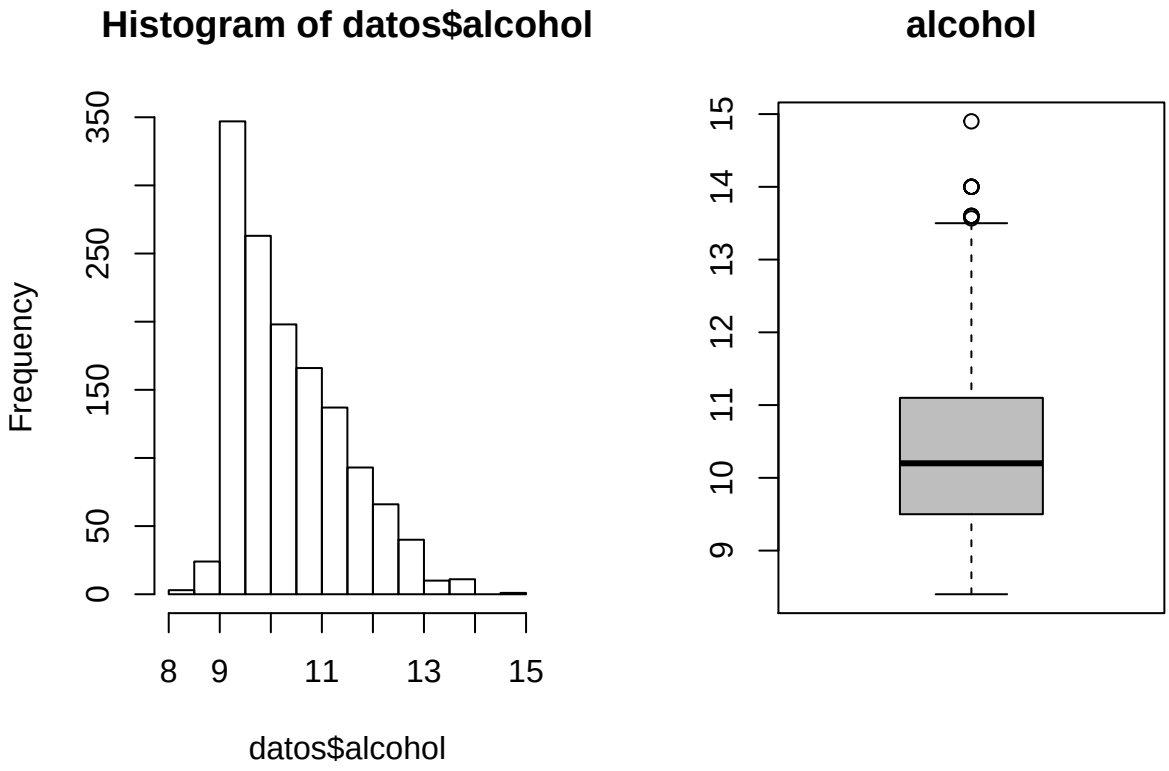



```
boxplot.stats(datos$sulphates)$out
```

```
##  [1] 1.56 1.28 1.08 1.20 1.12 1.28 1.14 1.95 1.22 1.98 1.31 2.00 1.08 1.59 1.02
## [16] 1.03 1.61 1.09 1.26 1.08 1.00 1.36 1.18 1.13 1.04 1.11 1.07 1.06 1.06 1.05
## [31] 1.06 1.04 1.05 1.02 1.14 1.02 1.36 1.36 1.05 1.17 1.62 1.06 1.18 1.07 1.34
## [46] 1.16 1.10 1.15 1.17 1.33 1.18 1.17 1.03 1.10 1.01
```

Observamos como el atributo “sulphates” tiene 55 valores extremos, distribuidos entre 1 y 2.

```
par(mfrow=c(1,2))
hist(datos$alcohol)
boxplot(datos$alcohol,main="alcohol", col="gray")
```

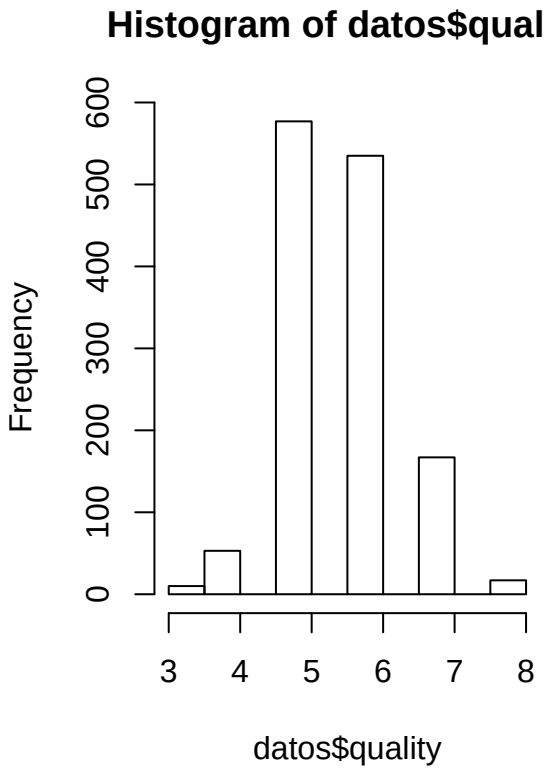


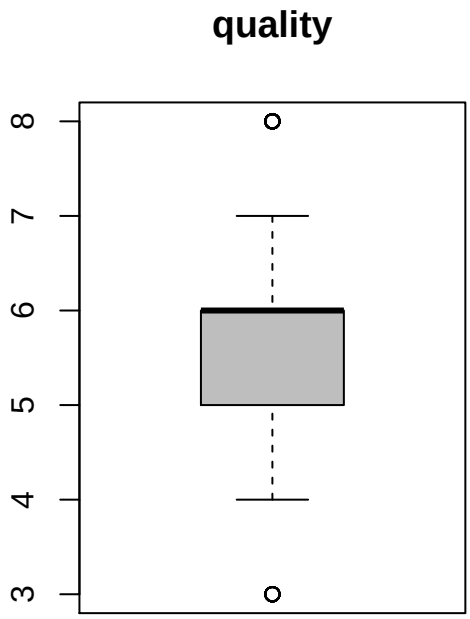

```
boxplot.stats(datos$alcohol)$out
```

```
##  [1] 14.00000 14.00000 14.00000 14.90000 14.00000 13.60000 13.60000 13.60000
##  [9] 14.00000 14.00000 13.56667 13.60000
```

Observamos como el atributo “alcohol” tiene 12 valores extremos, distribuidos entre el 13.5 y el 14.

```
par(mfrow=c(1,2))
hist(datos$quality)
boxplot(datos$quality,main="quality", col="gray")
```






```
boxplot.stats(datos$quality)$out
```

```
##  [1] 8 8 8 8 8 3 8 8 3 8 3 8 3 3 8 8 8 8 8 3 3 8 8 3 3 3 8
```

Observamos como el atributo "quality" tiene 27 valores extremos, distribuidos entre el 3 por la parte inferior, y el 8 en la parte superior. Este valor es una valoración del vino, por lo que estos valores no se pueden considerar extremos.

En conclusión, con el análisis realizado para cada variable, el número de valores extremos es muy dispar, siendo bajo en algunas variables y relativamente alto en otras.

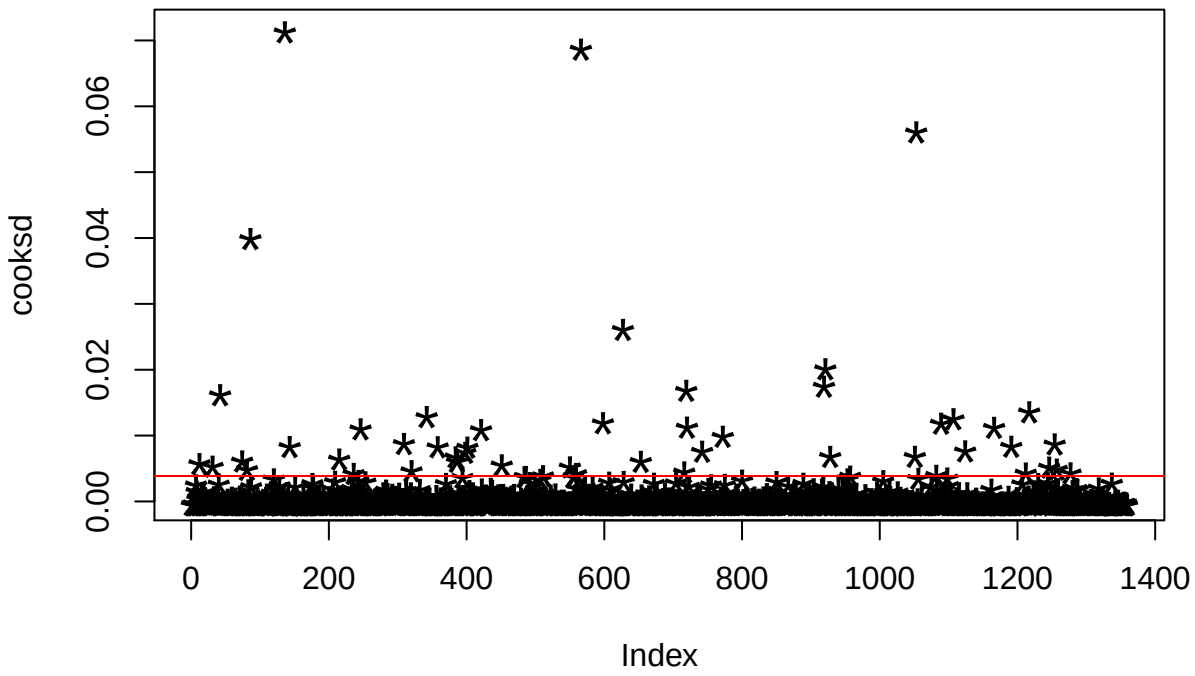
Como la eliminación de todos los valores extremos detectados afectaría sensiblemente a la muestra, vamos a buscar cuales de dichos valores son realmente extremos a partir de la distancia de Cook, estimando el grado de influencia de cada uno de los valores al realizar un análisis de regresión por mínimos cuadrados.

Para realizarlo, tendremos en cuenta todos los atributos a excepción de "quality".

```
# Cálculo y visualización de resultados de aplicar la distancia de Cook a nuestros datos.
outliers = c()
for ( i in 1:11 ) {
  stats = boxplot.stats(datos[[i]])$stats
  bottom_outlier_rows = which(datos[[i]] < stats[1])
  top_outlier_rows = which(datos[[i]] > stats[5])
  outliers = c(outliers , top_outlier_rows[ !top_outlier_rows %in% outliers ] )
  outliers = c(outliers , bottom_outlier_rows[ !bottom_outlier_rows %in% outliers ] )
}

mod = lm(quality ~ ., data = datos)
cooksd = cooks.distance(mod)
```

```
plot(cooksd, pch = "*", cex = 2, main = "Observaciones relevantes en función de la distancia de Cook")
abline(h = 4*mean(cooksd, na.rm = T), col = "red")
```




```
# Obtenemos el listado de los valores extremos que afectan sensiblemente a los datos.
head(datos[cooksd > 4 * mean(cooksd, na.rm=T), ])
```

```
##    fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 14           7.8            0.610        0.29            1.6     0.114
## 34           6.9            0.605        0.12           10.7     0.073
## 46           4.6            0.520        0.15            2.1     0.054
## 80           8.3            0.625        0.20            1.5     0.080
## 87           8.6            0.490        0.28            1.9     0.110
## 93           8.6            0.490        0.29            2.0     0.110
##    free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 14                   9                   29  0.9974 3.26      1.56     9.1
## 34                  40                   83  0.9993 3.45      0.52     9.4
## 46                   8                   65  0.9934 3.90      0.56    13.1
## 80                  27                  119  0.9972 3.16      1.12     9.1
## 87                  20                  136  0.9972 2.93      1.95     9.9
## 93                  19                  133  0.9972 2.93      1.98     9.8
##    quality
## 14       5
## 34       6
## 46       4
## 80       4
## 87       6
```

```
## 93       5
```

Si visualizamos las primeras entradas, todos ellos tienen valores atípicos en una o más variables.

El registro 14 tiene los "chlorides" y los "sulphates" muy altos. El registro 34 tiene el "residual.sugar" muy alto. El registro 46 tiene el "pH" muy alto. El registro 80 tiene los "sulphates" altos. El registro 87 y 93 tienen los "chlorides", los "sulphates" y el "total.sulfur.dioxide" altos.

Vamos a eliminar los valores extremos detectados.

```
# Identificamos los registros a eliminar
coutliers = as.numeric(rownames(datos[cooksd > 4 * mean(cooksd, na.rm=T), ]))
```

```
# Eliminamos los elementos detectados como extremos.
datos = datos[-c(coutliers), ]

# Visualizamos el tamaño y los valores básicos de nuestro nuevo conjunto de datos.
dim(datos)
```

```
## [1] 1308   12
```

```
summary(datos)
```

```
##  fixed.acidity    volatile.acidity  citric.acid    residual.sugar
##  Min.   : 4.600   Min.   :0.1200   Min.   :0.000   Min.   : 0.900
##  1st Qu.: 7.100   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900
##  Median : 7.900   Median :0.5200   Median :0.260   Median : 2.200
##  Mean   : 8.301   Mean   :0.5288   Mean   :0.271   Mean   : 2.531
##  3rd Qu.: 9.200   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600
##  Max.   :15.900   Max.   :1.5800   Max.   :1.000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 22.00       1st Qu.:0.9956
##  Median :0.07900   Median :14.00       Median : 38.00       Median :0.9967
##  Mean   :0.08797   Mean   :15.92       Mean   : 46.82       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:22.00       3rd Qu.: 63.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :72.00       Max.   :289.00       Max.   :1.0037
##        pH          sulphates         alcohol         quality
##  Min.   :2.74   Min.   :0.3700   Min.   : 8.40   Min.   :3.000
##  1st Qu.:3.21   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.31   Median :0.6200   Median :10.20   Median :6.000
##  Mean   :3.31   Mean   :0.6588   Mean   :10.44   Mean   :5.625
##  3rd Qu.:3.40   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
##  Max.   :4.01   Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

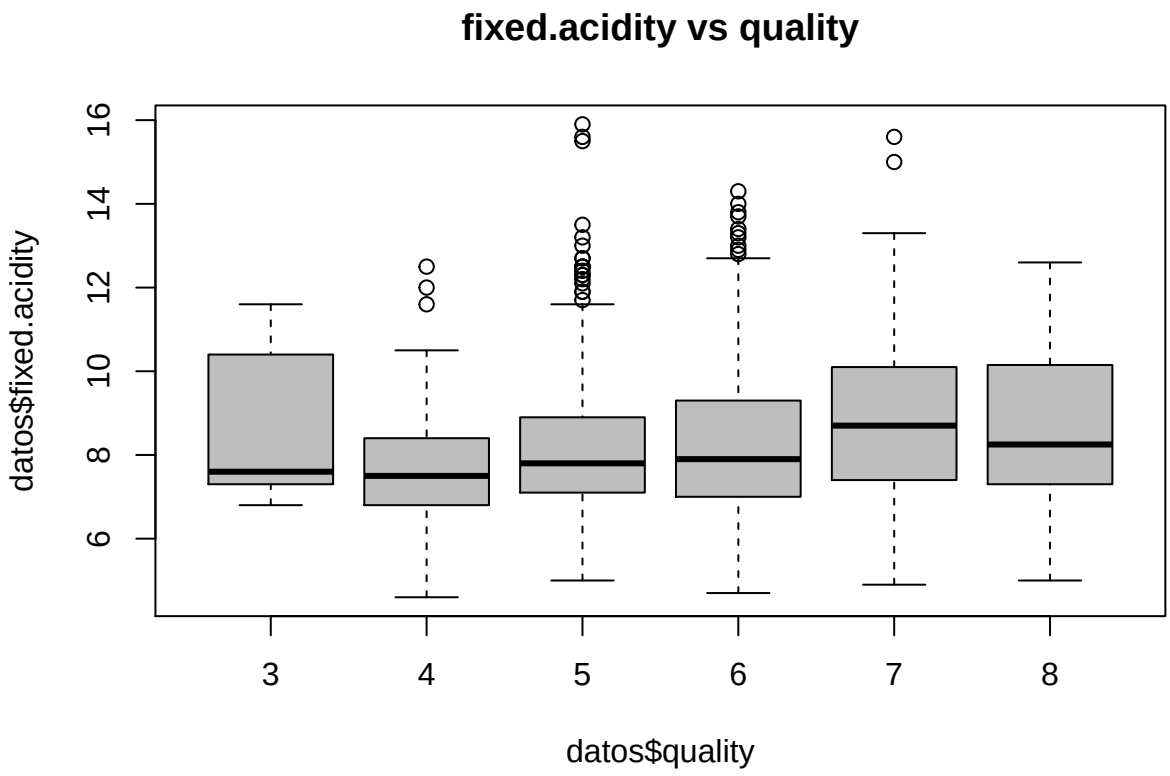
Análisis

Antes de comenzar con el análisis guardaremos una copia de los datos después del proceso de limpieza

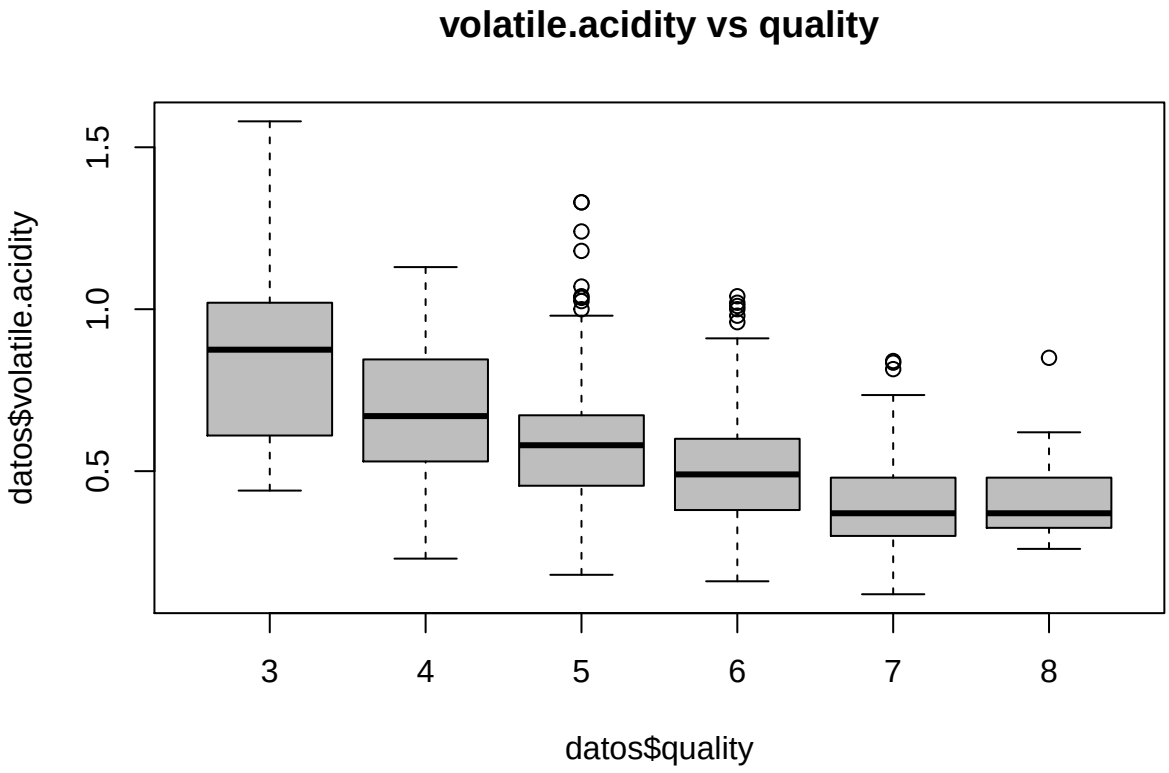
```
# Exportación de los datos limpios en .csv
#write.csv(datos, "RedWinQuality_clean.csv")
write.csv(datos, "winequality-red-clean.csv")
```

Analizaremos las variables frente a la calidad para decidir cuales utilizar en el resto del analisis

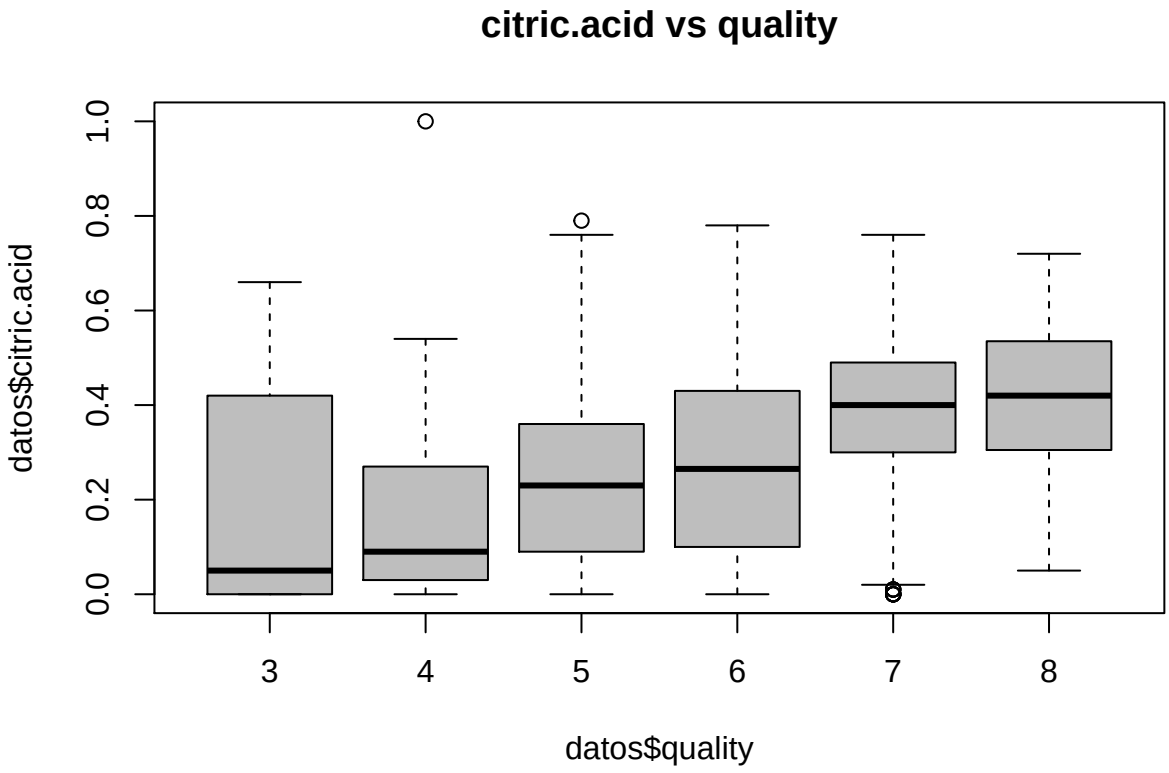
```
#boxplot(datos$pH,main="quality", col="gray")
boxplot(formula = datos$fixed.acidity ~ datos$quality, main="fixed.acidity vs quality", col="gray")
```



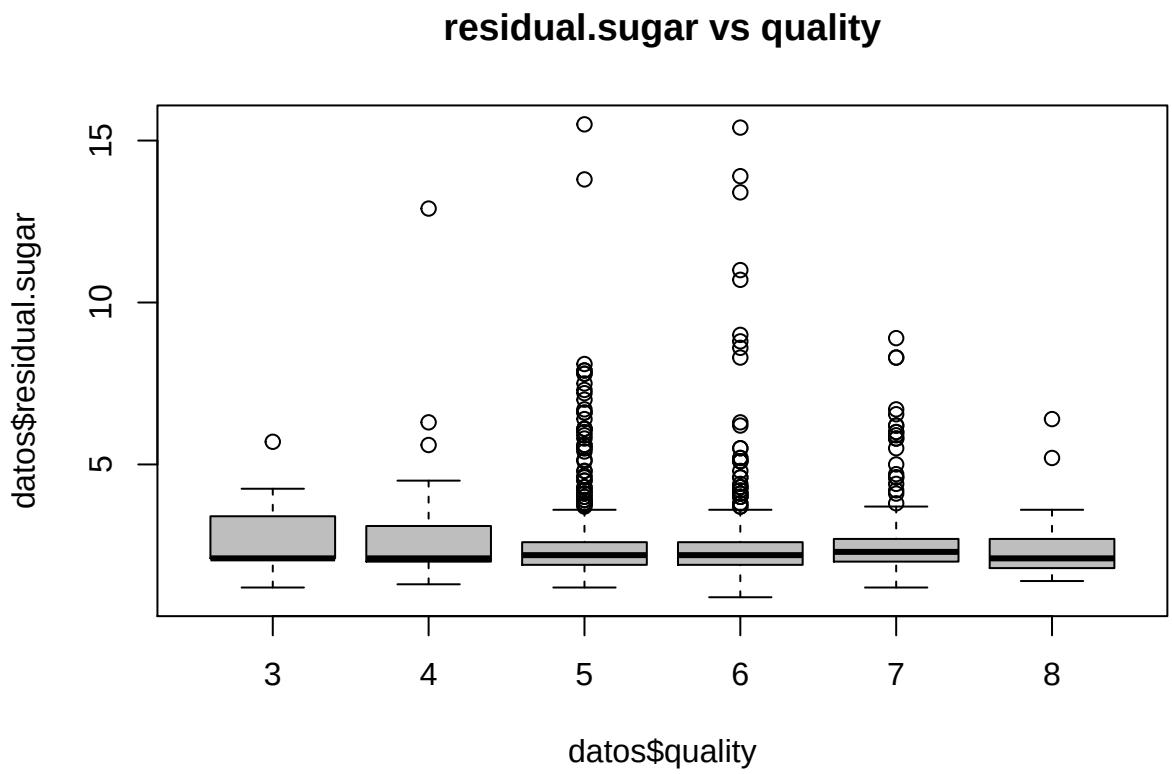

```
boxplot(formula = datos$volatile.acidity ~ datos$quality, main="volatile.acidity vs quality", col="gray
```



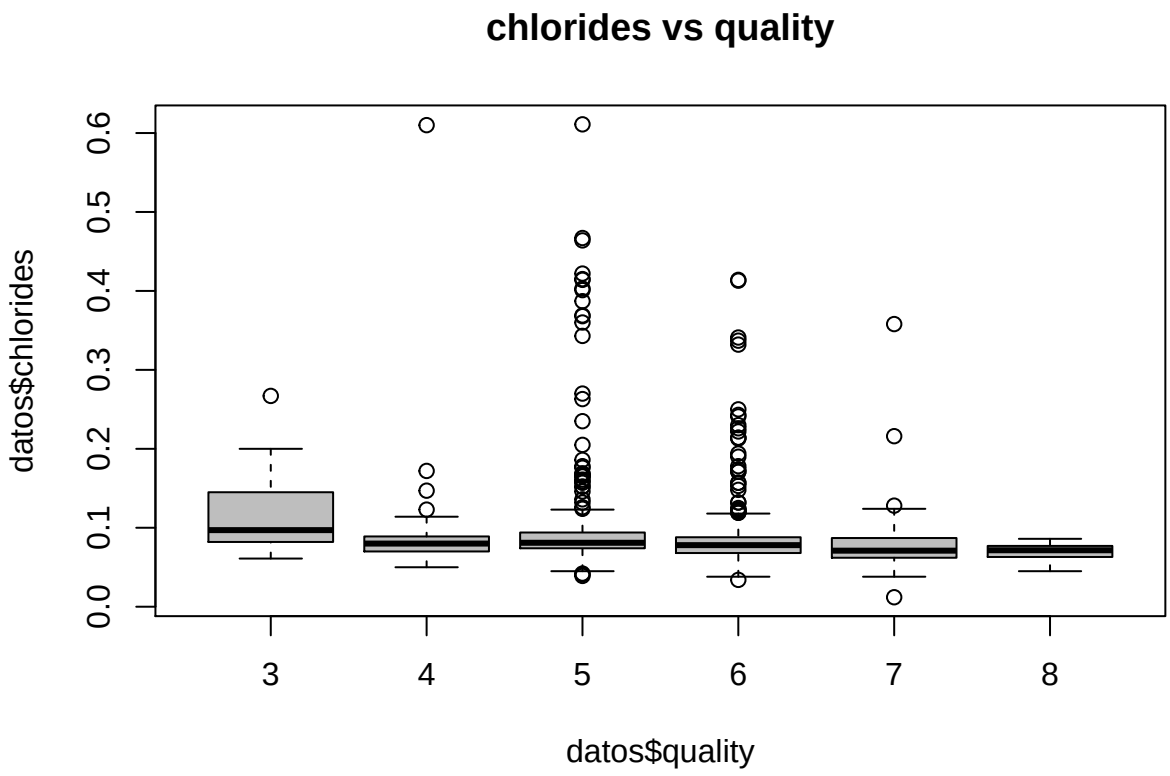

```
boxplot(formula = datos$citric.acid ~ datos$quality, main="citric.acid vs quality", col="gray")
```



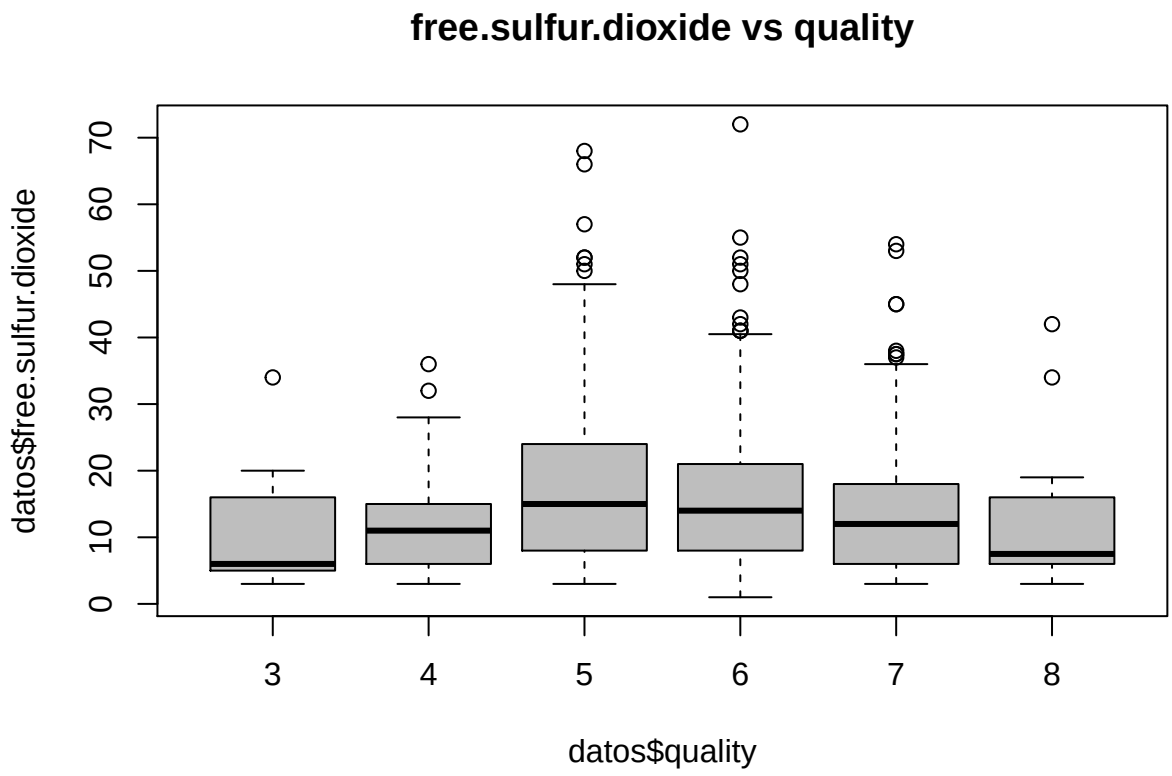

```
boxplot(formula = datos$residual.sugar ~ datos$quality, main="residual.sugar vs quality", col="gray")
```



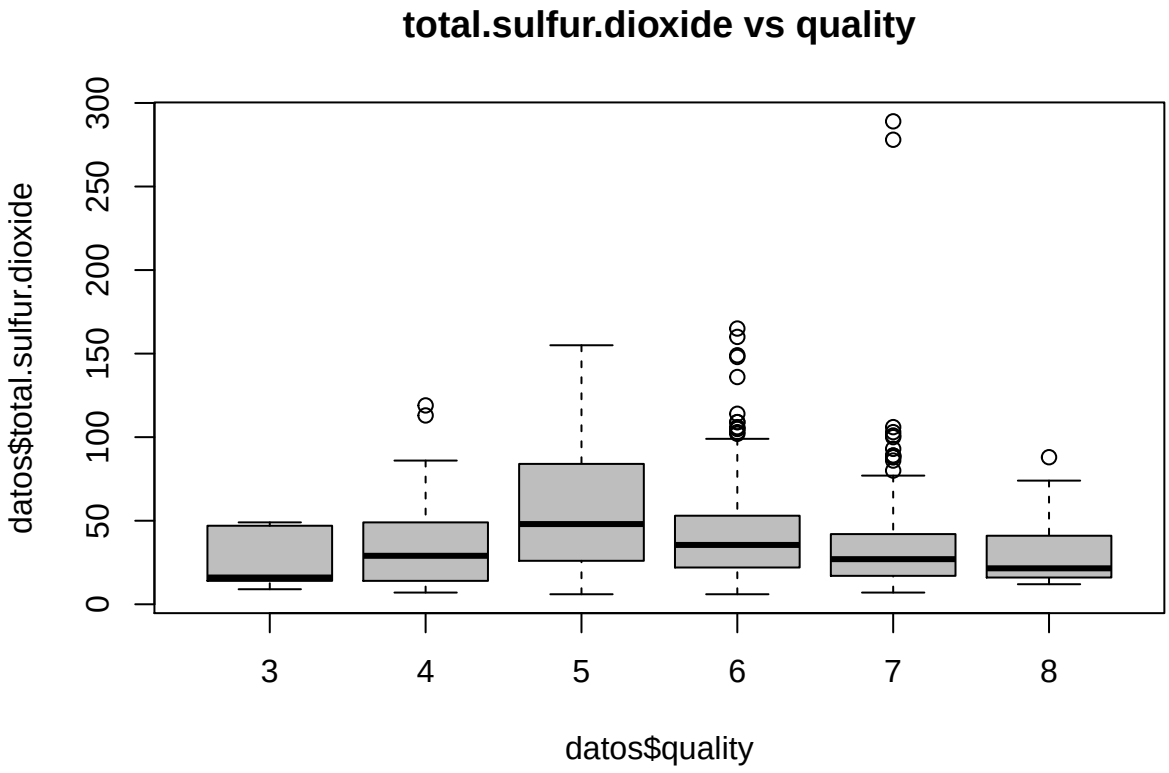

```
boxplot(formula = datos$chlorides ~ datos$quality, main="chlorides vs quality", col="gray")
```



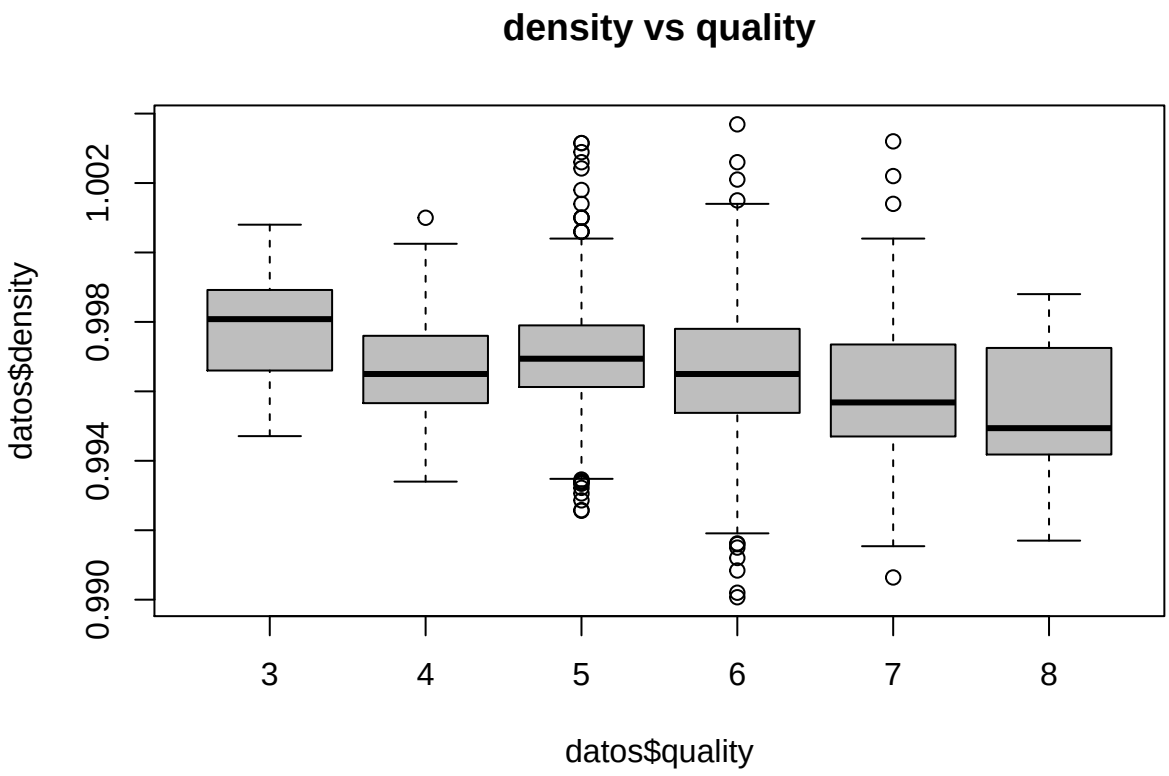

```
boxplot(formula = datos$free.sulfur.dioxide ~ datos$quality, main="free.sulfur.dioxide vs quality", col=
```



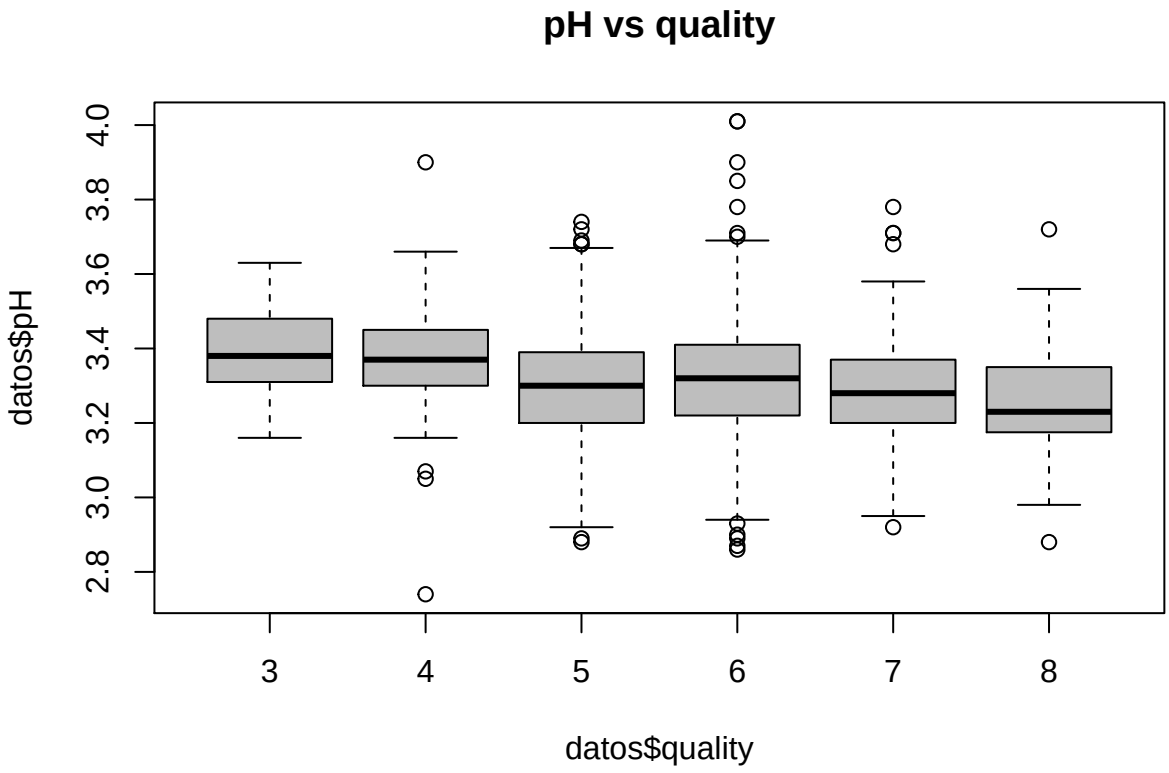

```
boxplot(formula = datos$total.sulfur.dioxide ~ datos$quality, main="total.sulfur.dioxide vs quality", c
```



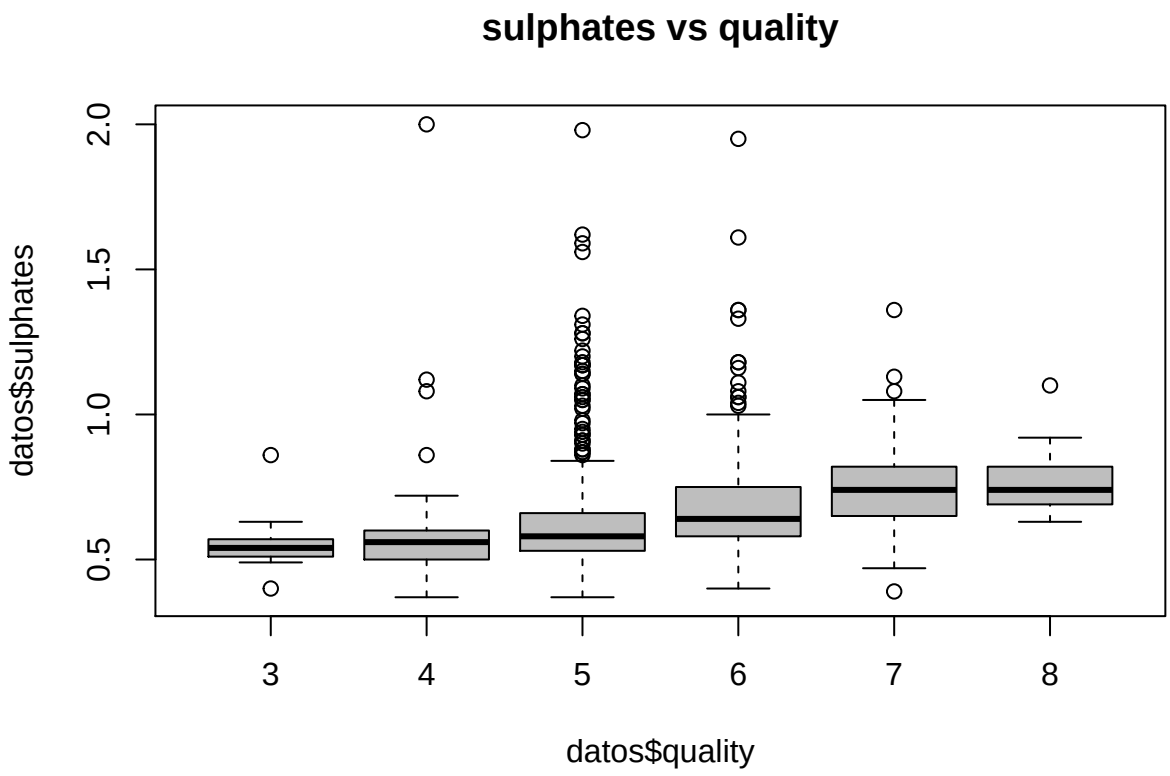

```
boxplot(formula = datos$density ~ datos$quality, main="density vs quality", col="gray")
```



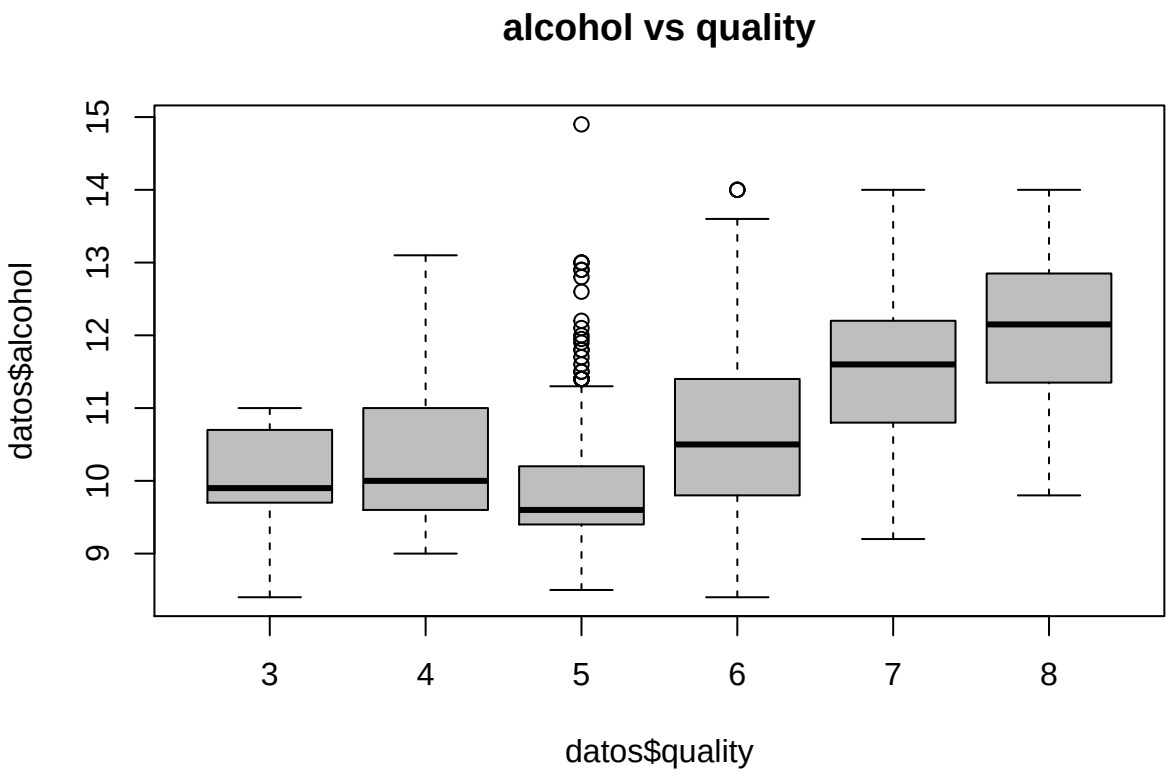

```
boxplot(formula = datos$pH ~ datos$quality, main="pH vs quality", col="gray")
```




```
boxplot(formula = datos$sulphates ~ datos$quality, main="sulphates vs quality", col="gray")
```




```
boxplot(formula = datos$alcohol ~ datos$quality, main="alcohol vs quality", col="gray")
```



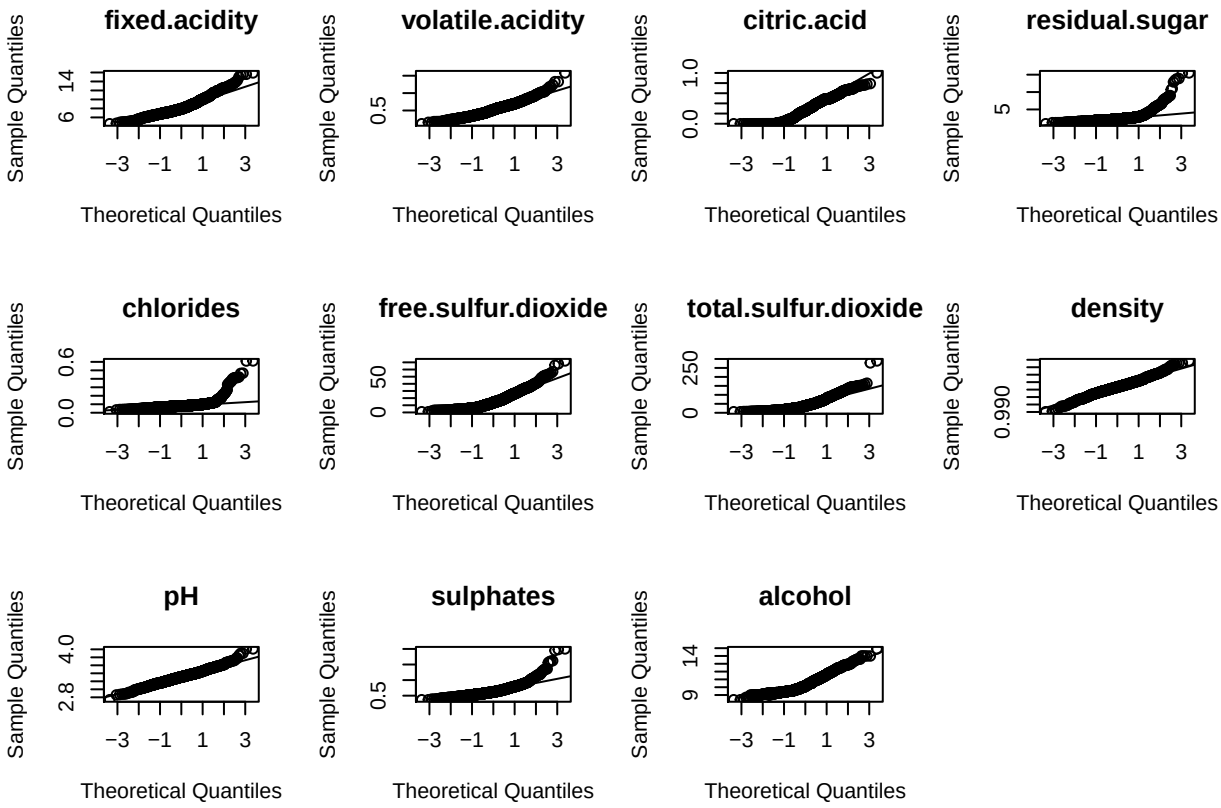
Seleccion grupo de datos De la observación del grupo de datos nos interesa seleccionar los que pudieran tener una mayor relación con el resultado de calidad. Por ello vamos a seleccionar las que se intuye una cierta relación lineal para poder aplicar modelos de predicción. Las variables "fixed acidity", "citrix acid" , "alcohol" y "sulphates", conforme aumentan, aumenta el valor de la calidad. Por el contrario para que aumente el valor de la calidad es necesario que disminuyan "volatile acidity", "density" y "pH". Crearemos un subconjunto de datos con estas cinco variables

```
subdatos <- select(datos, fixed.acidity, volatile.acidity, citric.acid, sulphates, alcohol)
```

Comprobación de la normalidad y homogeneidad de la varianza.

Comprobaremos la normalidad de los datos y ejecutaremos el test. Para ello nos ayudaremos de las graficas quantilie-quantile

```
par(mfrow=c(3,4))
for (i in 1:(ncol(datos)-1)) {
  qqnorm(datos[,i], main = colnames(datos[i]))
  qqline(datos[,i])
}
```



Los resultados graficos nos indican que las variables pueden ser candidatas a la normalizacion si fuera necesario. Aun asi y dado que los grupos tienen menos de 50 eventos emplearemos el test de Shapiro-Wilk para confirmar la hipotesis.

```
# Test de Shapiro
#shapiro.test(datos$fixed.acidity)
# Creamos la matriz para almacenar los datos
matrixsha <- matrix(nc = 3, nr = 0)
colnames(matrixsha) <- c("Variable","w","p-value")

# Recorremos el dataset ejecutando el test
for (i in 1:(ncol(datos)-1)) {

  sha <- shapiro.test(datos[,i])
  # Añadimos los datos a la matriz
  test = matrix(ncol = 3, nrow = 1)
  test[1][1] = colnames(datos[i])
  test[2][1] = sha[1]
  test[3][1] = sha[2]
  matrixsha <- rbind(matrixsha, test)
}

# Mostramos la tabla
```

Podemos observar de los valores obtenidos que el valor de W en ningun caso es demasiado pequeño con lo que no podemos rechazar la hipotesis nula. Siendo esta que la poblacion de los datos esta distribuida normalmente. Una vez comprobada la normalidad de los datos, realizaremos un análisis de la varianza.

!!! COMPLETAR CON ANOVA?? !!! ***

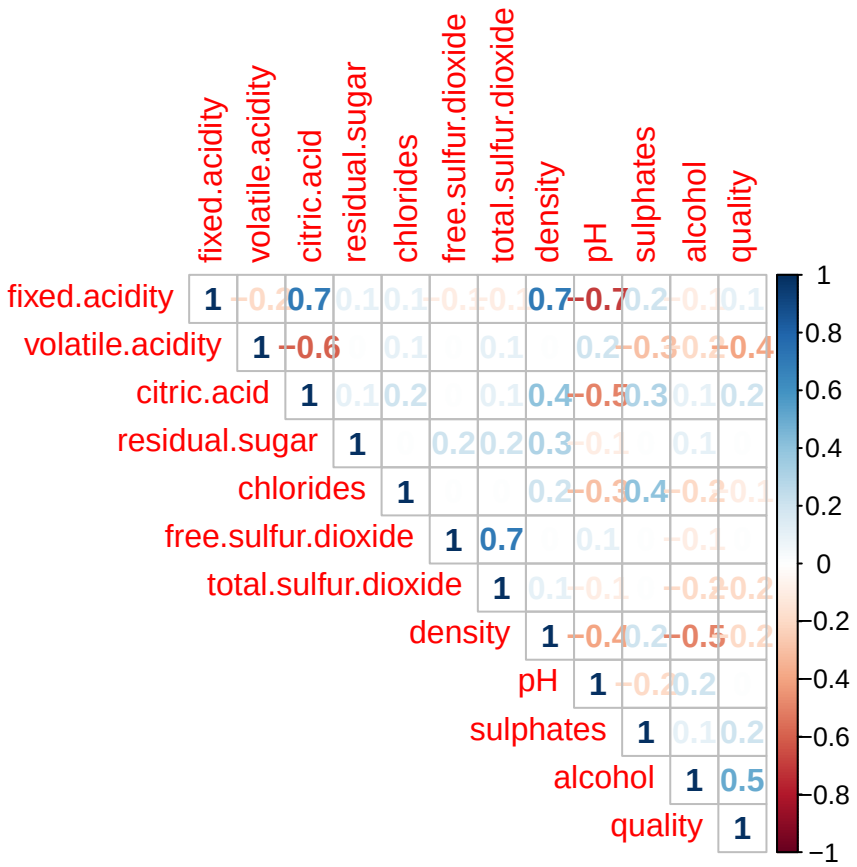
```
#anova <- aov(datos$bateo ~ datos$posicion)
#summary(anova)
##                 Df Sum Sq  Mean Sq F value Pr(>F)
## datos$posicion   3 0.0076 0.002519   1.994  0.115
## Residuals      323 0.4080 0.001263
#plot(anova)
```

Aplicación de pruebas estadísticas para comparar los grupos de datos

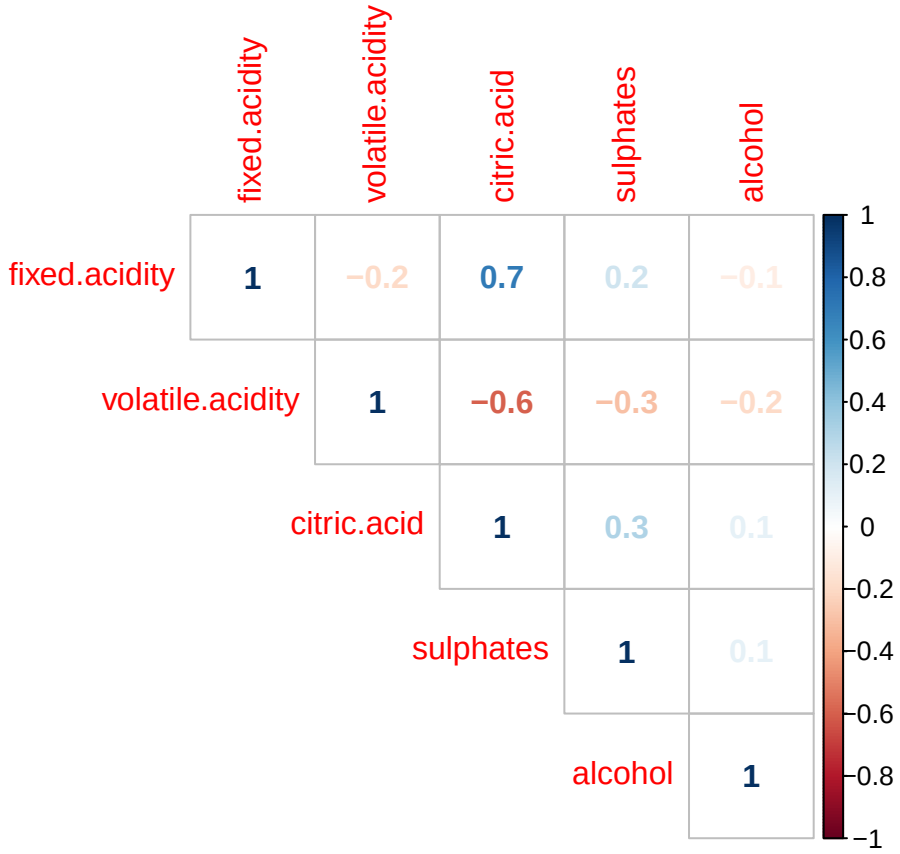
Aplicaremos diversas pruebas estadisticas para analizar la relacion de los datos y poder crear el modelo de prediccion de la calidad. Comenzaremos por analizar los valores de correlación de las variables con la variable "quality"

Correlacion

```
# Visualizaremos la matriz de correlacion de variables
correlacion<-round(cor(datos), 1)
corrplot(correlacion, method="number", type="upper")
```




```
# Visualizaremos la matriz de correlacion de las variables seleccionadas anteriormente
correlacion<-round(cor(subdatos), 1)
corrplot(correlacion, method="number", type="upper")
```



Guardaremos los datos de correlación en una matriz ordenada para decir que variables utilizar en siguientes estudios.

```
# Creamos la matriz para almacenar los datos
matrixcor <- matrix(nc = 2, nr = 0)
colnames(matrixcor) <- c("Variable","correlacion")
# Recorremos el dataset ejecutando el test
for (i in 1:(ncol(datos)-1)) {

  coef <- cor(x=datos$quality, y = datos[,i], method="spearman")
  # Añadimos los datos a la matriz
  pair = matrix(ncol = 2, nrow = 1)
  pair[1][1] = colnames(datos[i])
  pair[2][1] = coef
  matrixcor <- rbind(matrixcor, pair)
}
# Ordenamos por el valor de correlacion
matrixcor[order(matrixcor[,"correlacion"]), ]
```

```
##       Variable              correlacion
##  [1,] "pH"                  "-0.0315415647397534"
##  [2,] "free.sulfur.dioxide" "-0.0554375867293084"
##  [3,] "density"             "-0.190626869848596"
##  [4,] "total.sulfur.dioxide" "-0.198585035207389"
##  [5,] "chlorides"           "-0.213873297333666"
##  [6,] "volatile.acidity"    "-0.386472726595641"
##  [7,] "residual.sugar"      "0.0218226020265835"
```

```
##  [8,] "fixed.acidity"        "0.100763905616064"
##  [9,] "citric.acid"          "0.212705834869355"
## [10,] "sulphates"            "0.378086884588351"
## [11,] "alcohol"              "0.488947641940172"
```

Correlacion lineal

Con este grupo de datos y las relaciones observadas tanto en las gráficas de caja como los datos de correlación estimaremos por mínimos cuadrados ordinarios un modelo lineal que explique la variable "quality".

```
# Creamos el modelo
modelo <- (lm(formula = quality ~ fixed.acidity + citric.acid + alcohol + sulphates + volatile.acidity +
summary(modelo)
```

```
##
## Call:
## lm(formula = quality ~ fixed.acidity + citric.acid + alcohol +
##     sulphates + volatile.acidity + density + pH, data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.73806 -0.37234 -0.05244  0.45667  2.11417
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       24.70255   19.36814   1.275  0.20239
## fixed.acidity      0.06214    0.02470   2.516  0.01198 *
## citric.acid       -0.48746    0.15466  -3.152  0.00166 **
## alcohol            0.30601    0.02573  11.891  < 2e-16 ***
## sulphates          0.71135    0.11688   6.086 1.52e-09 ***
## volatile.acidity  -1.34983    0.12616 -10.699  < 2e-16 ***
## density          -22.15730   19.72166  -1.124  0.26143
## pH                -0.09859    0.19579  -0.504  0.61464
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.663 on 1300 degrees of freedom
## Multiple R-squared:  0.3482, Adjusted R-squared:  0.3447
## F-statistic:  99.2 on 7 and 1300 DF,  p-value: < 2.2e-16
```

Podemos observar que la densidad y el pH son variales poco significativas para el modelo. Con un valor de R cuadraddo ajustado del 34.47%.

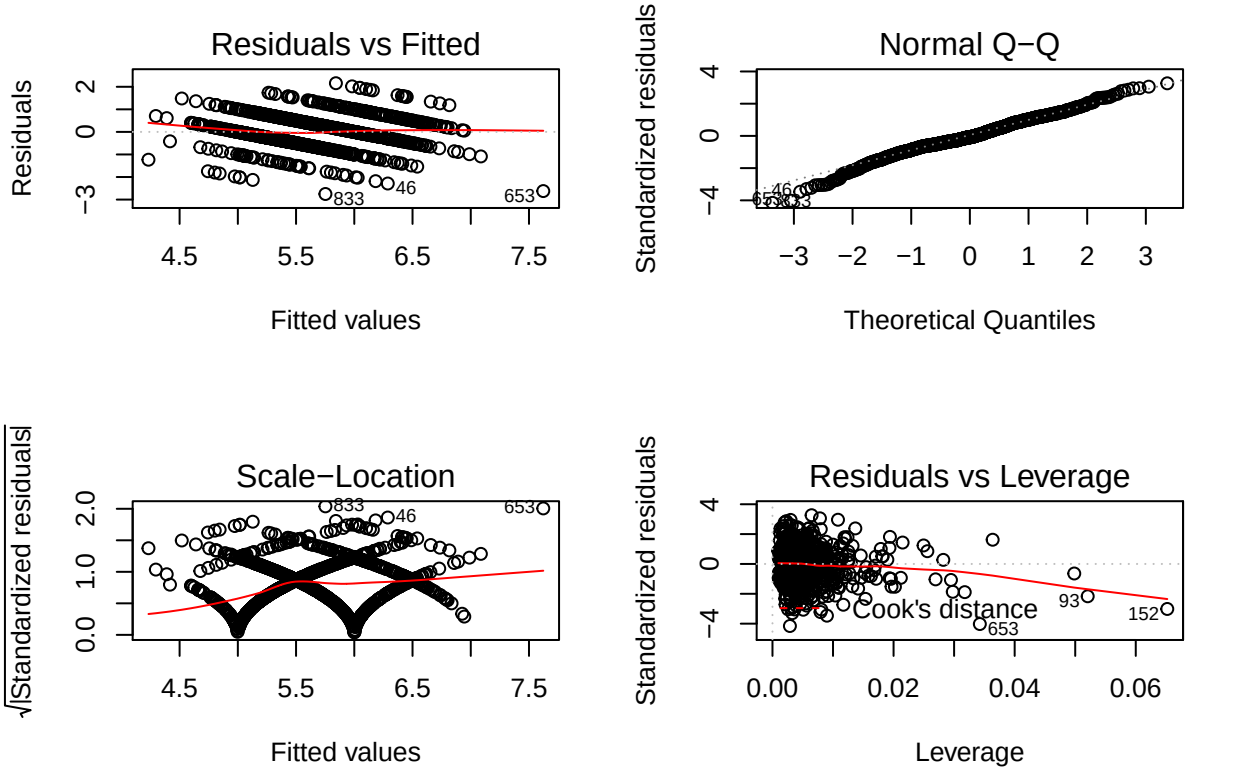
```
# Creamos un modelo reducido
modelo_reducido <- (lm(formula = quality ~ fixed.acidity + citric.acid + alcohol + sulphates + volatile
summary(modelo_reducido)
```

```
##
## Call:
## lm(formula = quality ~ fixed.acidity + citric.acid + alcohol +
##     sulphates + volatile.acidity, data = datos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7521 -0.3822 -0.0561  0.4575  2.1579
##
```

```
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        2.25994    0.24701   9.149  < 2e-16 ***
## fixed.acidity      0.05151    0.01461   3.526 0.000436 ***
## citric.acid       -0.47772    0.15204  -3.142 0.001715 **
## alcohol            0.32006    0.01750  18.285  < 2e-16 ***
## sulphates          0.69415    0.11362   6.109 1.32e-09 ***
## volatile.acidity -1.38297    0.12380 -11.171  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6632 on 1302 degrees of freedom
## Multiple R-squared:  0.3469, Adjusted R-squared:  0.3444
## F-statistic: 138.3 on 5 and 1302 DF,  p-value: < 2.2e-16
```

Analizamos estadisticamente el modelo.

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



La gráfica de residuos frente a Fitted muestra si los residuos tienen patrones no lineales. Los residuos alrededor de una línea horizontal sin patrones distintos indican que tenemos relaciones lineales.

La gráfica QQ plot normal muestra los residuos que se ajustan a la línea normal distribuidos.

La grafica Scale-Location muestra si los residuos se distribuyen por igual a lo largo de los rangos de predictores de forma que podemos verificar el supuesto de varianza igual (homocedasticidad). Podemos observar una linea horizontal con puntos de dispersión iguales.

El gráfico de residuos vs apalancamiento tiene un aspecto típico cuando hay algún caso influyente. Apenas puede ver las líneas de distancia de Cook (una línea punteada roja) porque todos los casos están dentro de la distancia de Cook.

Correlacion logistica

Con el objetivo de tener un modelo que nos permitiera tener un control de calidad excluyente que nos permitiera decidir si el producto final es bueno o malo. Vamos a crear un modelo de regresion logistica y compararemos resultados.

```
# Creamo una variale calidad de tipo factor
datos$quality_factor <- as.factor(datos$quality)

# Creamos un variable calidad binomial
datos$category[datos$quality <= 5] <- 0
datos$category[datos$quality > 5] <- 1
datos$category <- as.factor(datos$category)
```

```
# Generamos el modelo
#modelo_log <- glm(quality ~ fixed.acidity + citric.acid + alcohol + sulphates + volatile.acidity, data
modelo_log <- glm(category ~ fixed.acidity + citric.acid + alcohol + sulphates + volatile.acidity  + de
summary(modelo_log)
```

```
##
## Call:
## glm(formula = category ~ fixed.acidity + citric.acid + alcohol +
##     sulphates + volatile.acidity + density + pH, family = "binomial",
##     data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6131  -0.8537   0.2997   0.8361   2.4520
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       47.01232   71.06913   0.662 0.508291
## fixed.acidity      0.26492    0.08996   2.945 0.003233 **
## citric.acid       -2.18520    0.57581  -3.795 0.000148 ***
## alcohol            0.98322    0.10212   9.628  < 2e-16 ***
## sulphates          2.15759    0.42605   5.064 4.10e-07 ***
## volatile.acidity  -4.02377    0.50816  -7.918 2.41e-15 ***
## density          -60.99130   72.35714  -0.843 0.399273
## pH                 0.85662    0.71471   1.199 0.230705
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1809.1  on 1307  degrees of freedom
## Residual deviance: 1377.0  on 1300  degrees of freedom
## AIC: 1393
##
## Number of Fisher Scoring iterations: 4
```

```
confint(object = modelo_log, level = 0.95 )
```

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

```
##                           2.5 %     97.5 %
## (Intercept)       -92.07475172 186.837138
## fixed.acidity       0.08973946   0.442792
## citric.acid        -3.32838478  -1.069480
## alcohol             0.78605590   1.186668
## sulphates           1.33335447   3.006328
## volatile.acidity   -5.04294797  -3.049484
## density          -203.40099670  80.565805
## pH                 -0.53906155   2.264811
```

Creamos la tala de confusión correspondiente al modelo.

```
library(vcd)
```

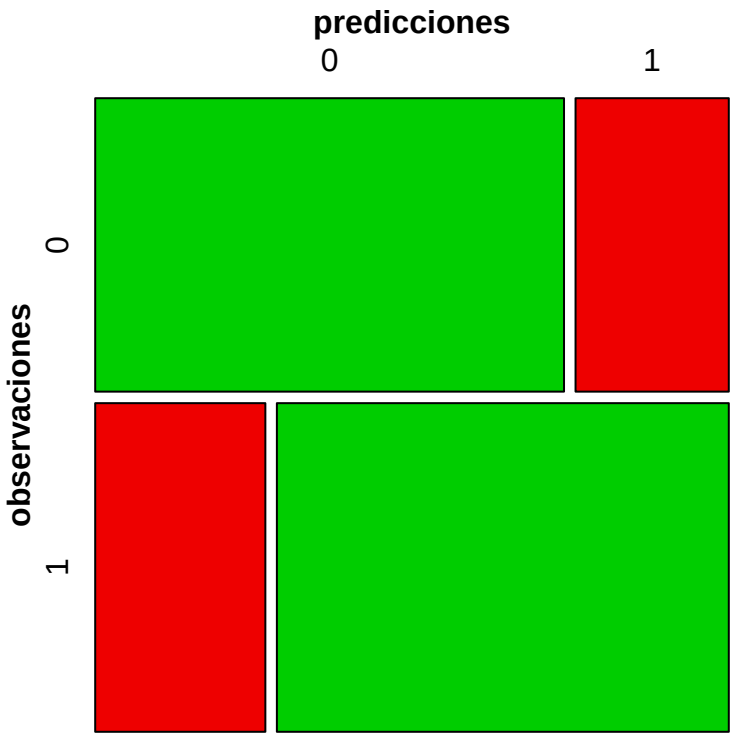
```
## Loading required package: grid
```

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

```
##              predicciones
## observaciones   0   1
##             0 465 152
##             1 189 502
```

Visualizamos los resultados.

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



!!! COMPLETAR CON UNA CONCLUSION DEL MODELO !!! ***

```
# La interpretación del modelo seria

#y <- 23.89874 + (0.06451 * fixed.acidity) - (0.48194 * citric.acid) + (0.30768 * alcohol) + (0.71648 *
```

!!! PARTE DE CLASIFICACION EN PRUEBAS VALORAR INCLUIRLA Y COMO !!! ***

Modelos de clasificacion

Tambien nos seria util tener algun modelo de clasificacion que pudieramos determinar la calidad del vino en funcion de sus caracteristicas. Podria agrupar los productos o produccion en varios productos de venta, etc...

Crearemos un modelo supervisado.

```
#install.packages("rminer")
library(rminer)

h<-holdout(datos$quality,ratio=2/3,mode="stratified")
data_train<-datos[h$tr,]
data_test<-datos[h$ts,]
print(table(data_train$quality))
```

```
##
##   3   4   5   6   7   8
##   5  31 374 336 114  12
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
train_control1<- trainControl(method="LOOCV")
train_control2<- trainControl(method="cv", number=10)
train_control3<- trainControl(method="repeatedcv", number=4, repeats=10)
```

```
train_control<- trainControl(method="cv", number=4)
mod<-train(quality~., data=data_train, method="rf", trControl = train_control)
pred <- predict(mod, newdata=data_test)
####confusionMatrix(pred,data_test$quality_factor,positive="7")
```

Crearemos un modelo no supervisado

```
datos.cl<-datos
datos.cl$quality<-NULL
kmeans.res<-kmeans(datos.cl,9)
table(datos$quality,kmeans.res$cluster)
```

```
##
##       1   2   3   4   5   6   7   8   9
##   3   0   5   1   0   0   2   0   0   1
##   4   2  17  12   0   2   4   1   6   5
##   5  58 101  95  21  63  46  25  70  80
##   6   5 113 128   4  29  65  21  52  97
##   7   1  59  42   2   8  15   7   1  26
##   8   0   8   3   0   1   2   1   0   1
```

```
library(cluster)
kmedoids.res1<-pam(datos.cl,9)
table(datos$quality,kmedoids.res1$cluster)
```

```
##
##       1   2   3   4   5   6   7   8   9
##   3   1   0   0   1   5   0   0   2   0
##   4   4   3   5  11  17   2   0   5   2
##   5  72  39  58  84 100  53  35  51  67
##   6  79  47  34 118 108   8   5  82  33
##   7  20  10   1  40  56   2   2  22   8
##   8   1   1   0   3   8   0   0   2   1
```

```
#library(fpc)
#kmedoids.res2<-pamk(datos.cl)
#table(datos$quality,kmedoids.res2$pamobject$clustering)
```

Representación de los resultados

Resolución del problema