Tipología y Ciclo de Vida de los Datos - aula 1

Práctica 2- Limpieza y análisis de datos

Alumnos: Enrique J.Villalobos Torregrosa y F.Javier Albarrán González - Profesor: Diego Pérez

Mayo 2022

Contents

Descripción de la práctica

El objetivo de esta actividad será el tratamiento de un dataset, que puede ser el creado en la práctica 1 o bien cualquier dataset libre disponible en Kaggle (https://www.kaggle.com).

Algunos ejemplos de dataset con los que podéis trabajar son:

- Red Wine Quality (https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009)
- Titanic: Machine Learning from Disaster (https://www.kaggle.com/c/titanic)

El último ejemplo corresponde a una competición activa de Kaggle de manera que, opcionalmente, podéis aprovechar el trabajo realizado durante la práctica para entrar en esta competición.

Importante: si se elige un dataset diferente de los propuestos es importante que este contenga una amplia variedad de datos numéricos y categóricos para poder realizar un análisis más rico y poder responder a las diferentes preguntas planteadas en el enunciado de la práctica.

Siguiendo las principales etapas de un proyecto analítico, las diferentes tareas a realizar (y **justificar**) son las siguientes:

1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?
2. Integración y selección de los datos de interés a analizar. Puede ser el resultado de adicionar diferentes datasets o una subselección útil de los datos originales, en base al objetivo que se quiera conseguir.

3. Limpieza de los datos. 3.1. ¿Los datos contienen ceros o elementos vacíos? Gestiona cada uno de estos casos. 3.2. Identifica y gestiona los valores extremos.
4. Análisis de los datos.

- 4.1. Selección de los grupos de datos que se quieren analizar/comparar (p. e., si se van a comparar grupos de datos, ¿cuáles son estos grupos y qué tipo de análisis se van a aplicar?)
- 4.2. Comprobación de la normalidad y homogeneidad de la varianza.
- 4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

5. Representación de los resultados a partir de tablas y gráficas. Este apartado se puede responder a lo largo de la práctica, sin necesidad de concentrar todas las representaciones en este punto de la práctica.
6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?
7. Código: Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y representación de los datos. Si lo preferís, también podéis trabajar en Python.

Resolución

Descripción del dataset

El conjunto de datos elegido es el correspondiente a la calidad del vino del link del kaggle propuesto. Se accede a él en la siguiente url:

https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009

Contiene las caracterísiticas fisicoquímicas y sensoriales de 1599 vinos portugueses. Las características o atributos son los siguientes:

- **Características físicoquímicas**:

1. *fixed.acidity* - la mayoría de los ácidos involucrados con el vino o fijos o no volátiles (no se evaporan fácilmente).
2. *volatile.acidity* - la cantidad de ácido acético en el vino, que en niveles demasiado altos puede provocar un sabor desagradable a vinagre.
3. *citric.acid* - presente en pequeñas cantidades, el ácido cítrico puede añadir 'frescura' y sabor a los vinos.
4. *residual sugar* - la cantidad de azúcar que queda después de que se detiene la fermentación, es raro encontrar vinos con menos de 1 gramo/litro y vinos con más de 45 gramos/litro se consideran dulces.
5. *chlorides* - la cantidad de sal en el vino.
6. *free.sulfur.dioxide* - la forma libre de SO2 existe en equilibrio entre el SO2 molecular (como gas disuelto) y el ion bisulfito; previene el crecimiento microbiano y la oxidación del vino.
7. *total.sulfur.dioxide* - cantidad de formas libres y ligadas de S02; en bajas concentraciones, el SO2 es mayormente indetectable en el vino, pero en concentraciones de SO2 libres superiores a 50 ppm, el SO2 se vuelve evidente en la nariz y el sabor del vino.
8. *density* - la densidad del agua es cercana a la del agua dependiendo del porcentaje de contenido de alcohol y azúcar.
9. *pH* - describe qué tan ácido o básico es un vino en una escala de 0 (muy ácido) a 14 (muy base); la mayoría de los vinos están entre 3 y 4 en la escala de pH.
10. *sulphates* - un aditivo del vino que puede contribuir a los niveles de dióxido de azufre (S02), que actúa como antimicrobiano y antioxidante.
11. *alcohol* - el porcentaje de contenido de alcohol del vino.

- **Características sensoriales:**

12. *quality* - variable de salida/objetivo, calidad, que ofrece una puntuación a cada instancia de vino, basada en datos sensoriales. Puntuación entre 0 y 10.

Importancia del dataset

Estamos ante un conjunto de datos de amplia difusión en el mundo de *Ciencia de Datos*, comparable al conjunto de datos de la *flor iris*, utilizado en múltiples ocasiones a modo de ejemplo por la versatilidad de sus datos y el ámplio abanico de cuestiones que se pueden plantear sobre su contenido.

El dataset "Red Wine Quality" está ámpliamente difundido y es conocido por quienes se dedican al estudio de los datos o quieren dedicarse a ello. Permite plantear problemas tanto de clasificación como de regresión, que pueden tener relevancia en el sector vinícola.

Pero más allá del uso de estos datos en el mundo académico de la *Ciencia de Datos*, si entramos en el significado de los datos que proporciona, podemos extraer conocimiento sobre los compuestos de los vinos para tratar de mejorar su composición y su calidad. De ahí que pueda aportar valor a bodegas y otros productores de vino similar al del dataset.

Puede servir, también, de referencia para validar las puntuaciones de catadores y puede servir a éstos a entrenar su olfato y paladar para detectar los mejores vinos.

En definitiva, que nos podría ofrecer la posibilidad de "fusionar", con modelos basados en datos objetivos, el mundo físico del vino y sus características con el mundo sensorial y subjetivo de los profesionales del sector.

Preguntas a responder

La problemática que nos planteamos resolver es, por una parte, clasificar un tipo de vino dado en función de sus atributos físicos, y por otro, poder anticiparnos al estudio sensorial y poder ofrecer una estimación sobre la calidad.

Integración y selección de datos

Para nuestro estudio vamos a trabajar con todos los datos que se facilitan en el dataset. Sin conocer como pueden afectar los distintos compuestos químicos a la calidad, o incluso, cómo pueden interactuar entre ellos, nos parece arriesgado eliminar características a priori.

Del mismo modo trabajaremos con todas las instancias del dataset, salvo que se tenga que eliminar algún registro, ya sea porque no contenga información relevante o porque tenga un elevado número de nulos.

En este sentido, a lo largo de los puntos siguientes, procederemos a la limpieza del conjunto de datos, al análisis de valores atípicos (*outliers*) y su tratamiento si procediera, para continuar con el estudio univariante de los atributos a través de la visualización de sus distribuciones/histogramas, estudio bivariante de algún datos con respecto al atributo *quality*, y finalmente, estudiar las correlaciones entre los datos.

Limpieza de datos

En primer lugar procedemos con la carga de librerías y la lectura del conjunto de datos.

```
library(corrplot)
library(VIM)
library(ggplot2)
library(scales)
library(ggridges)
library (caret)
library(dplyr)
library(randomForest)
library(gmodels)
library(ResourceSelection)
library(pROC)
```

```
df <- read.csv("winequality-red.csv")
print(dim(df))
```

```
## [1] 1599   12
```

```
str(df)
```

```
## '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 ...
```

Resumimos a continuación los datos para tener una mejor aproximación a ellos a través de sus datos estadísticos básicos, rangos, distribución o valores nulos.

```
summary(df)
```

```
##  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
```

Todas las variables son cuantitativas y continuas a excepción de la variable *quality* que es discreta, con valores según un convenio/criterio, y que debemos considerar como cualitativa para los procesos siguientes.

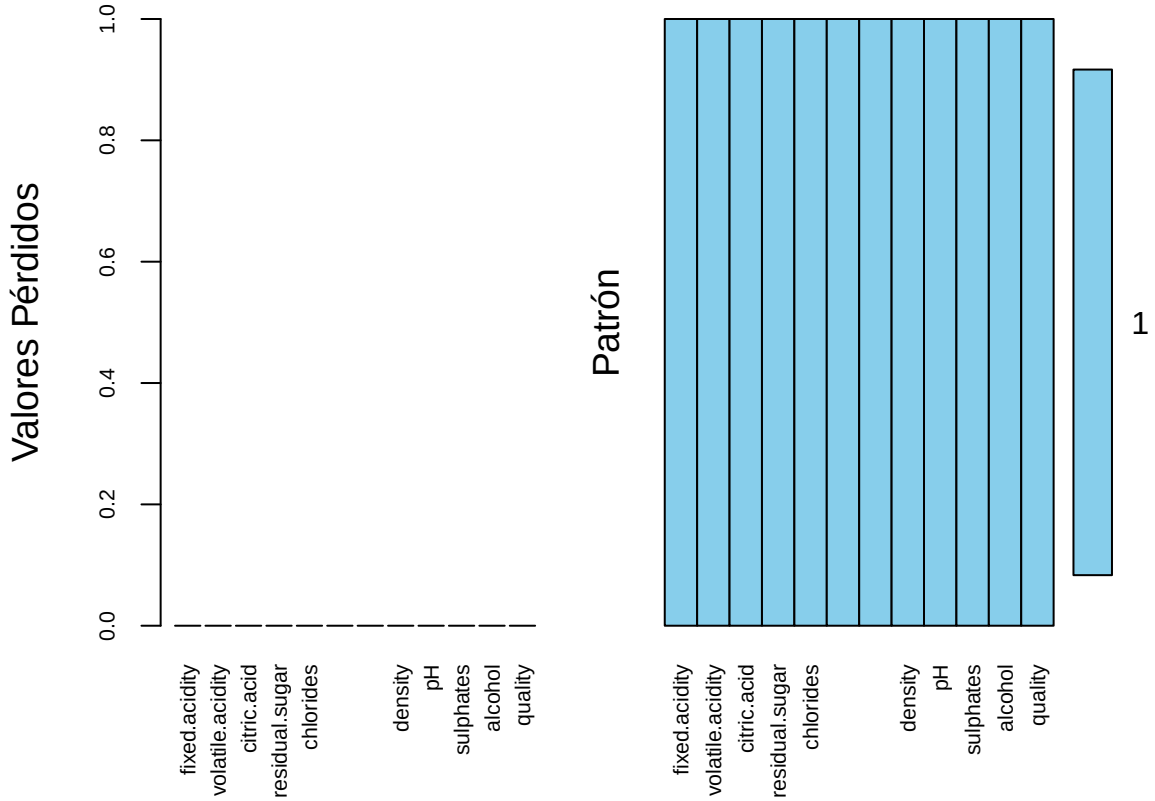
Una vez observada la naturaleza de los datos veamos si existen elementos vacíos que puedan afectar a los resultados del análisis posterior.

```
NaNs <- apply(X = is.na(df), MARGIN = 2, FUN = sum)
NaNs
```

```
##        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
```

```
aggr(df, numbers=TRUE, sortVars=TRUE, labels=names(df),
cex.axis=.7, gap=3, ylab=c("Valores Pérdidos","Patrón"))
```




```
##
##  Variables sorted by number of missings:
##               Variable Count
##          fixed.acidity     0
##       volatile.acidity     0
##            citric.acid     0
##         residual.sugar     0
##              chlorides     0
##    free.sulfur.dioxide     0
##   total.sulfur.dioxide     0
##                density     0
##                     pH     0
##              sulphates     0
##                alcohol     0
##                quality     0
```

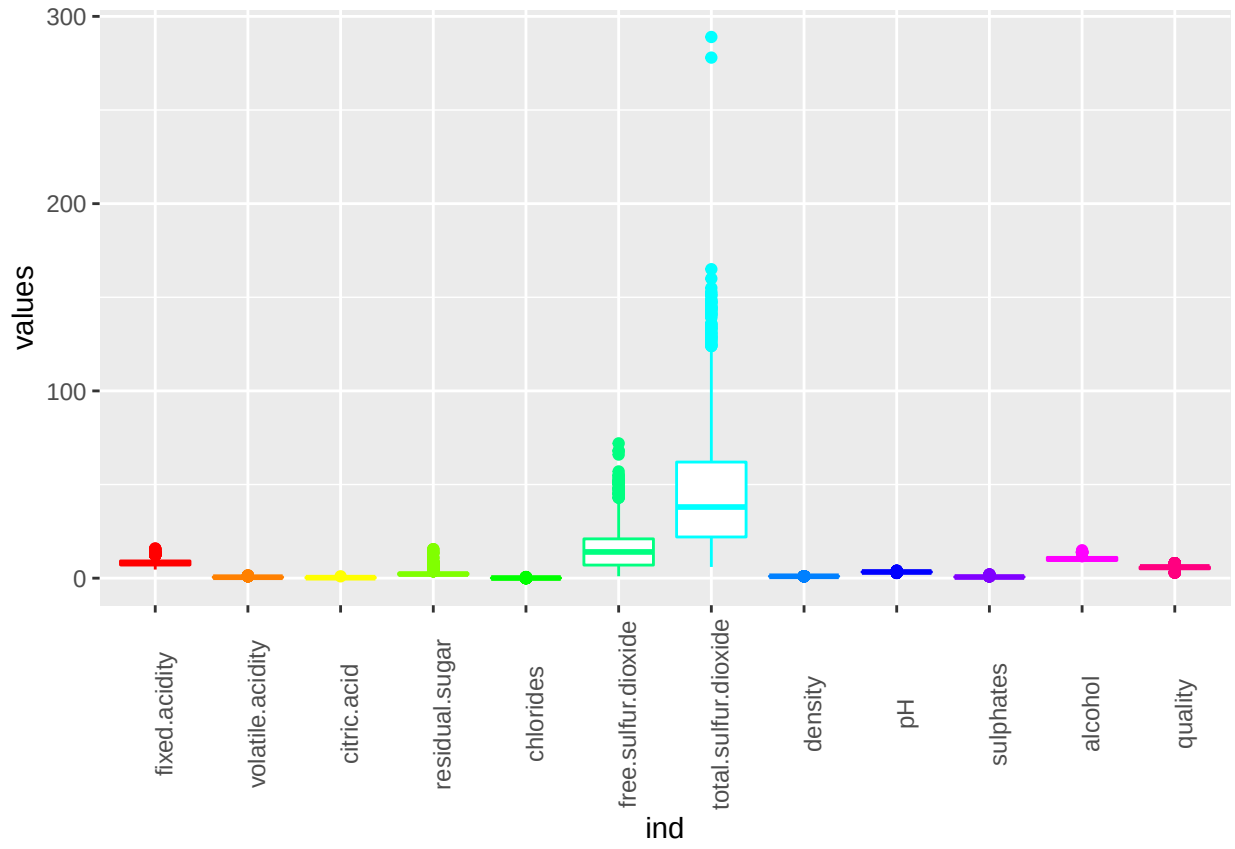
Podemos apreciar que ninguna variable presenta valores faltantes (*NAs*). En cuanto a las variables con valores 0, observamos que la variable **citric.acid** es la única que tiene valores de 0. Tras las investigaciones efectuadas en uvadoc.uva.es, se ha acordado que el valor 0 esta dentro del dominio del atributo.

```
df_0 <- Filter(function(x) sum(x == 0) > 0, df)
head(sort(df_0$citric.acid, decreasing = F, na.last = TRUE),132)
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [75] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [112] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Veamos ahora si existen valores extremos (*outliers*), para ello usaremos un boxplot.

```
ggplot(stack(df), aes(x = ind, y = values)) +
  geom_boxplot(col=rainbow(12))+
  theme(axis.text.x = element_text(angle = 90))
```



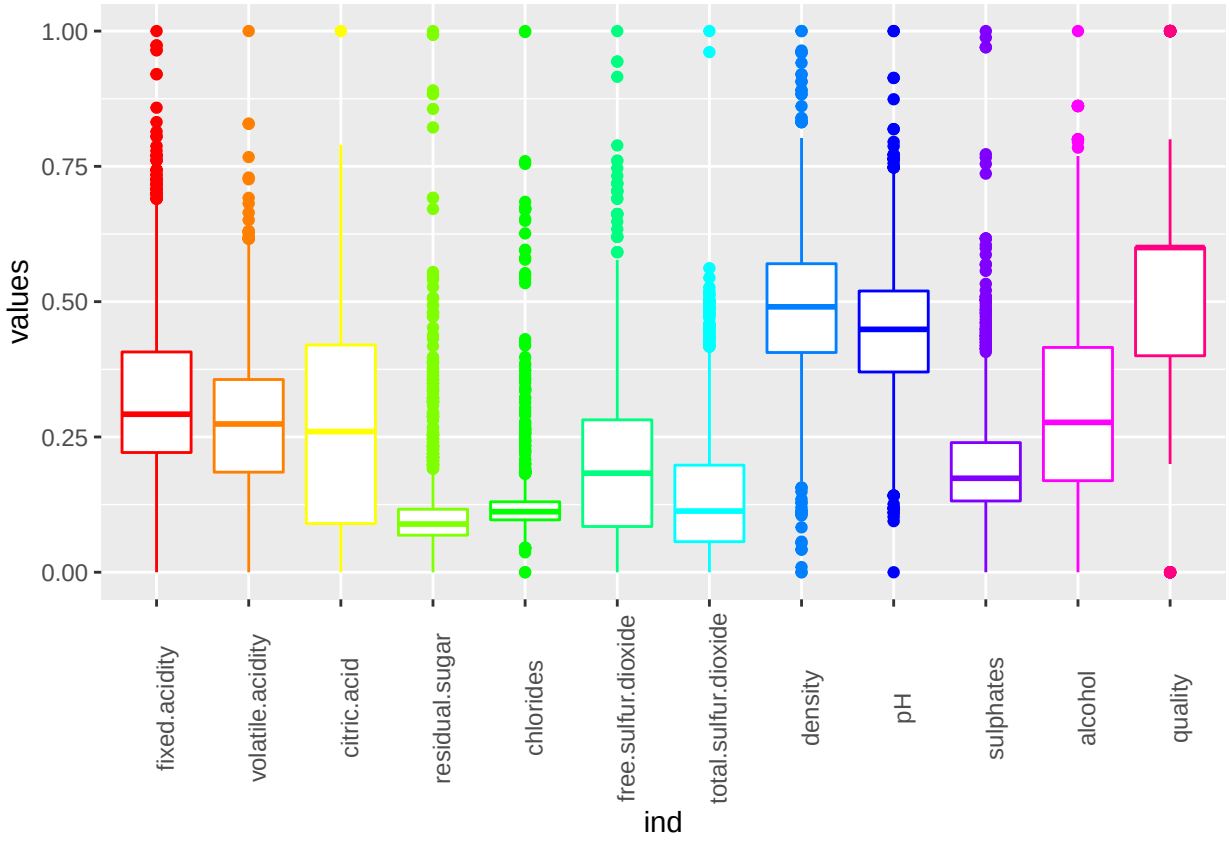
Observamos que las variables han sido medidas en diferentes escalas lo que influye en la gráfica. Para corregir el efecto de la escala vamos a normalizar el conjunto de datos, escalando entre 0 y 1 todos los valores, pero respetando las distribuciones, y volver a graficarlo.

```
minMax <- function(x) {
  (x - min(x)) / (max(x) - min(x))
}

df_norm <- as.data.frame(lapply(df, minMax))

ggplot(stack(df_norm),  aes(x = ind, y = values)) +
```

```
  geom_boxplot(col=rainbow(12))+
  theme(axis.text.x = element_text(angle = 90))
```



Apreciamos la existencia de un buen número de valores extremos, que no necesariamente deben considerarse como atípicos o anómalos. Lo que sí parece cierto es que cada vino posee valores muy diferentes de los atributos en estudio y que, en principio, todos ellos contribuyen a determinar la calidad final del producto.

Pongamos un ejemplo. El *pH*.

```
summary(df$pH)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.740   3.210   3.310   3.311   3.400   4.010
```

Investigando el dato, los valores aceptable para una calidad óptima, varían entre 3.1 y 3.9 valtea.es. Sin embargo, entre los vinos en estudio, tenemos vinos con pésima calidad, por lo que es muy probable que su pH se encuentre también fuera de rango. No podemos considerar estos valores como anómalos por un proceso de captura de datos erroneo y eliminarlos sin perder información que puede ser útil a nivel de conjunto para identificar vinos de inferior calidad.

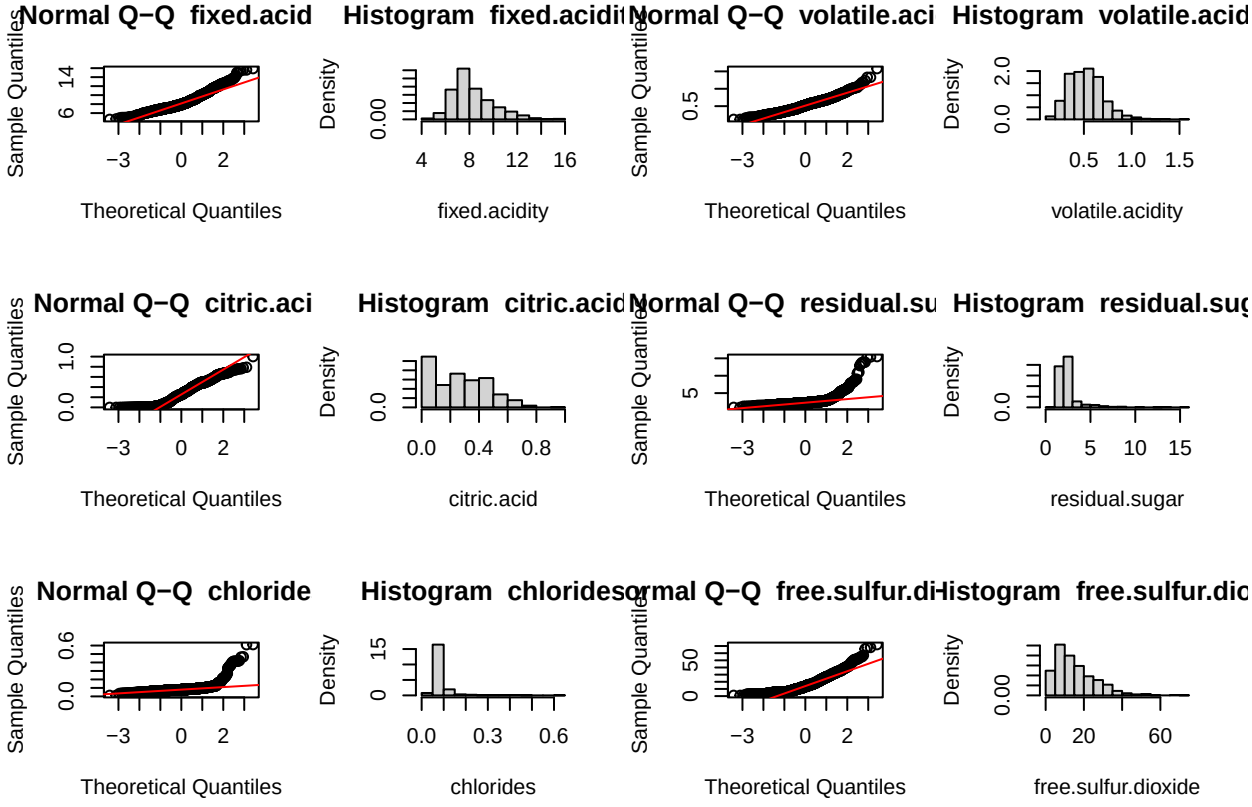
El mismo razonamiento aplicamos al resto de atributos.

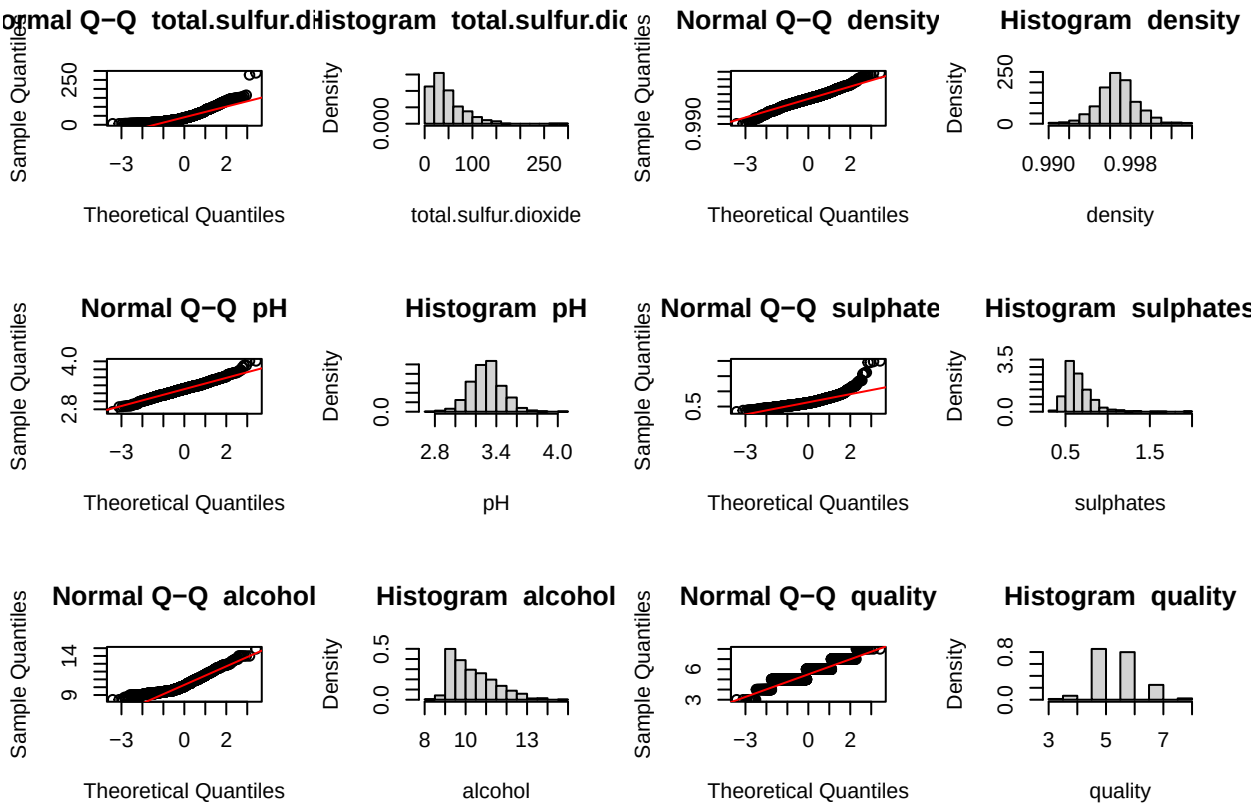
Es de recibo mencionar que lo mismo sucede con la variable target *quality*. Su distribución se centra en los valores medios, pero existen vinos excepcionales, con las puntuaciones más altas, que en ningún caso pueden ser considerados atípicos o *outliers* al uso. Por lo tanto, mantenemos en el dataset todos estos valores sin hacer tratamientos o borrados de los mismos, dada su posible relevancia en el resultado final.

Normalidad y homocedasticidad de los datos

De manera visual vamos a proceder a examinar si los datos se acomodan a una distribución normal para poder elegir los estimadores oportunos en caso de necesitarlos para hacer **hipótesis de contraste**.

```
par(mfrow=c(3,4))
for(i in 1:ncol(df)) {
  if (is.numeric(df[,i])){
    qqnorm(df[,i],main = paste("Normal Q-Q ",colnames(df)[i]))
    qqline(df[,i],col="red")
    hist(df[,i],main=paste("Histogram ", colnames(df)[i]),xlab=colnames(df)[i],
         freq =FALSE)
  }
}
```





En general parece que los datos siguen la normal, pero no tenemos una certeza plena si lo trasladamos a la población. Aplicaremos el text de normalidad de *Shapiro Wilk* para comprobarlo. Este test trabaja con la **hipótesis nula** de normalidad poblacional de los datos. En caso de que el p-value sea mayor a 0.05 (nivel de significancia estándar), se acepta dicha hipótesis.

```
col.names = colnames(df)
for(i in 1:ncol(df)) {
  a <- shapiro.test(df[,i])
  if (a[2] < 0.05){
    print(paste("atributo ", col.names[i], "p-value", a[2]))
    }
  }
```

```
## [1] "atributo  fixed.acidity p-value 1.52501179295091e-24"
## [1] "atributo  volatile.acidity p-value 2.69293489456032e-16"
## [1] "atributo  citric.acid p-value 1.02193162131975e-21"
## [1] "atributo  residual.sugar p-value 1.02016171149076e-52"
## [1] "atributo  chlorides p-value 1.17905575371677e-55"
## [1] "atributo  free.sulfur.dioxide p-value 7.69459692029225e-31"
## [1] "atributo  total.sulfur.dioxide p-value 3.57345139578549e-34"
## [1] "atributo  density p-value 1.93605282884883e-08"
## [1] "atributo  pH p-value 1.71223728301906e-06"
## [1] "atributo  sulphates p-value 5.82314039765996e-38"
## [1] "atributo  alcohol p-value 6.64405672007326e-27"
## [1] "atributo  quality p-value 9.51508538364454e-36"
```

El resultado es demoledor. La población no sigue una distribución normal en ninguno de sus atributos. Si

embargo, al tratarse de una muestra lo suficientemente grande, se puede asumir, por el **teorema del límite central** que siguen una distribución normal y trabajar bajo esta hipótesis.

Puesto que los datos no siguen una Normal, tendremos que aplicar ahora, para comprobar la **homocedasticidad** de los datos, el text de *Fligner-Killeen*. Este test asume como hipótesis nula la igualdad de las varianzas.

Lo haremos entre los datos sulphates y cantidad de alcohol (ambas continuas).

```
fligner.test(alcohol ~ sulphates , data = df)
```

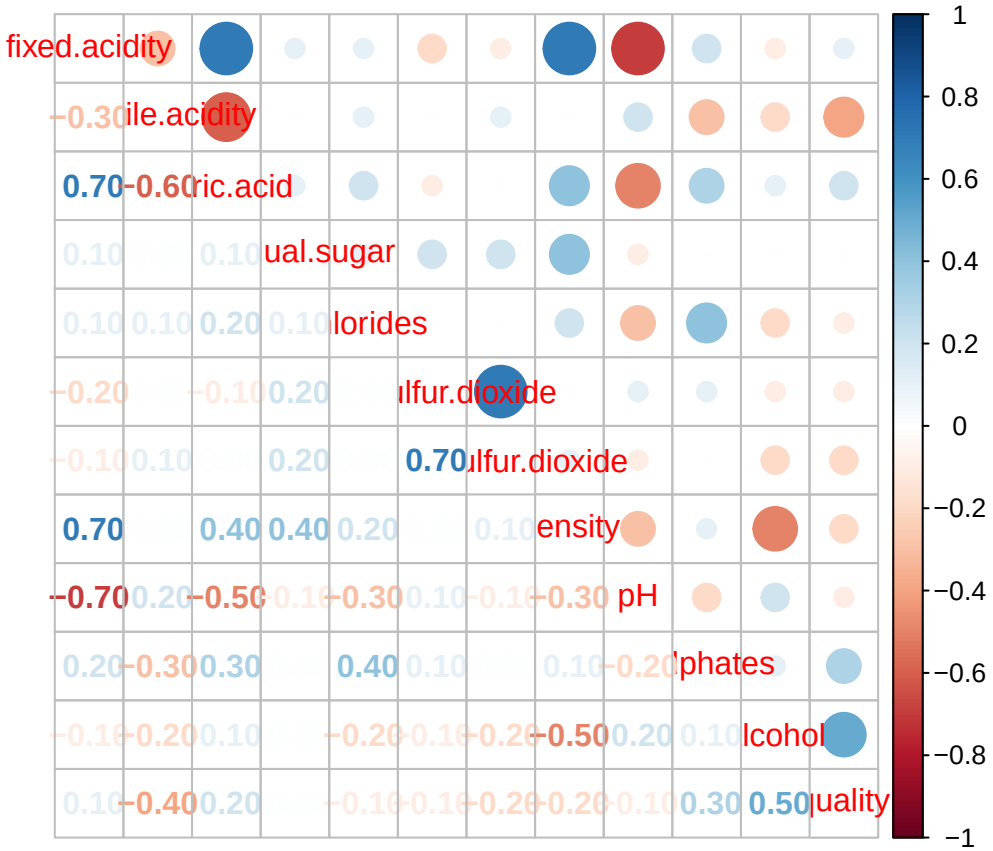
```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  alcohol by sulphates
## Fligner-Killeen:med chi-squared = 161.19, df = 95, p-value = 2.655e-05
```

Puesto que el valor del p-value es inferior al nivel de confianza con el que se trabaja (0.05), tenemos que rechazar la hipótesis nula, asumiendo que tienen varianzas diferentes estos dos atributos.

Análisis de los datos

A pesar de los resultados anteriores, asumiendo la normalidad de la muestra, por el **teorema central del límite**, dado que cuenta con el número suficiente de instancias para poder asumirlo, vamos a comprobar el grado de correlación existente entre los datos para ver si podemos sacar alguna conclusión o simplificar la información del dataset que finalmente acabemos pasando a los algoritmos de minería que vayamos a utilizar.

```
matriz <- round(cor(df, use = "complete.obs"),1)
corrplot.mixed(matriz)
```



```
#knitr::kable(matriz)
```

Apreciamos fuertes correlaciones de *fixed.acidity* con *density* (0.7), *citric.acid* (0.7) y con *ph* (-0.7), en este último caso, negativa. Podemos proceder a eliminar "fixed acidity" ya que comparte información con las otras tres. Es decir, explica sustancialmente la variabilidad de cada una de las otras y podría ser contraproducente mantener todas. Podríamos optar por eliminar las otras tres en lugar de *fixed.acidity*, pero consideramos que perderíamos mucha información, además de existir una correlación negativa con *ph* (a más acidez fija, menor ph - recordemos que valores menores de 7 indican acidez) que pensamos es importante mantener por su interacción con otros datos y afectación en los modelos.

Igual de fuerte es la correlación entre *free.sulfur.dioxide* con *total.sulfur.dioxide* (0.7). Entendemos que la primera es un subconjunto particular del segundo, lo cual explicaría esta correlación, y por lo tanto, vamos a eliminarla también del dataset para simplificar el modelo.

Existen otras correlaciones relativamente considerables, como *volatile.acidity* con *citric.acid* (-0.6) o la llamativa entre *alcohol* y *quality* (0.5) que nos indica que a más alcohol, más calidad. Por un principio de cautela, vamos a mantener estas variables.

```
eliminar <- c(1,6)
df2 <- df[, -eliminar]
colnames(df2)
```

```
##  [1] "volatile.acidity"     "citric.acid"          "residual.sugar"
##  [4] "chlorides"            "total.sulfur.dioxide" "density"
##  [7] "pH"                   "sulphates"            "alcohol"
## [10] "quality"
```

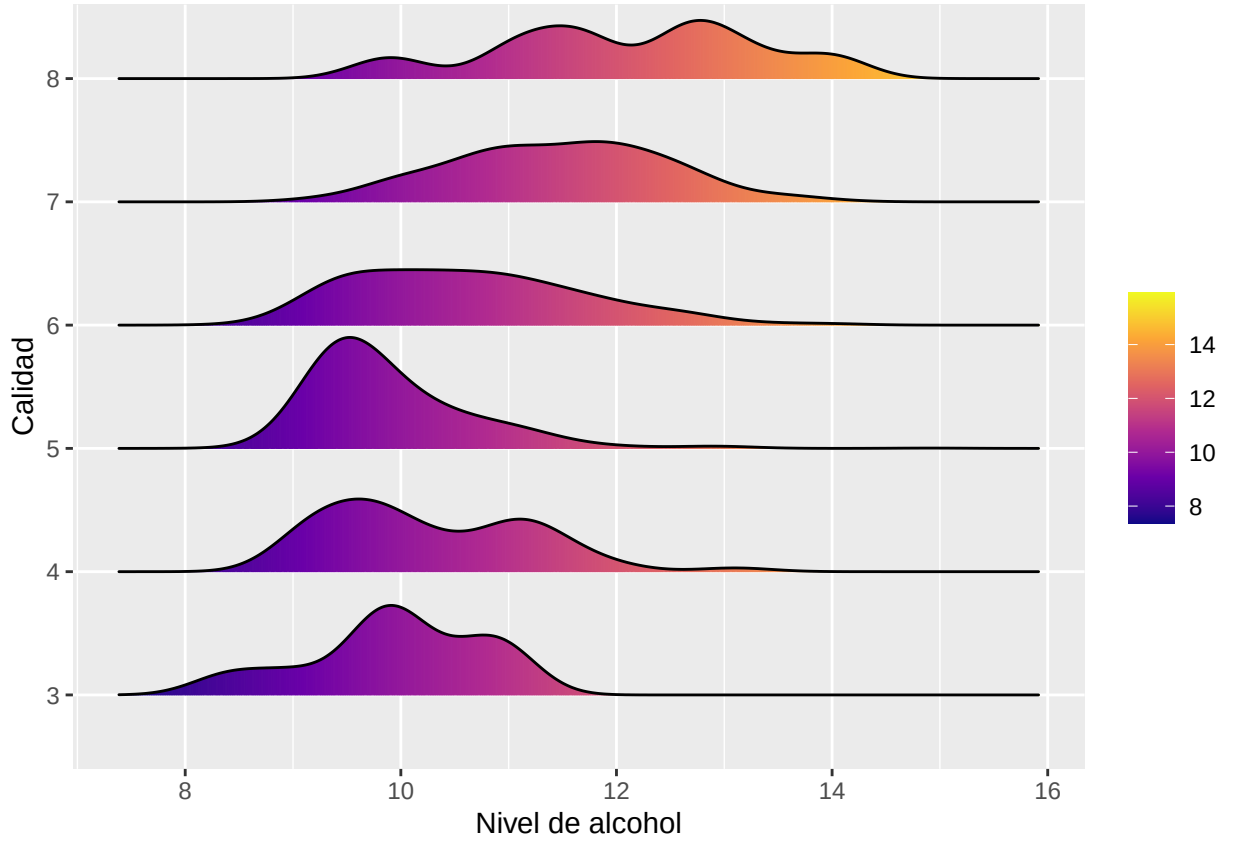
Análisis bivariante (frente a variable objetivo) de algún atributo

Puesto que nos ha resultado llamativa la relación entre alcohol y calidad vamos a ver como se reparten los distintos caldos entre estos dos campos.

```
df2$quality <-as.factor(df2$quality)

ggplot(df2, aes(x = alcohol, y = quality, fill = ..x..)) +
  geom_density_ridges_gradient(scale = .9, gradient_lwd = .5, color = "black") +
  scale_fill_viridis_c(option = "plasma", name = "") +
  labs(x = "Nivel de alcohol", y = "Calidad") #+
```

```
## Picking joint bandwidth of 0.338
```




```
#theme_ridges(font_family = "Roboto Condensed", grid = FALSE)
```

Apreciamos como los vinos con calidad inferior se "amontonan" en torno al 8-9% de alcohol, y cómo los de calidad superior están por encima de 11%.

Nota: No se realizan más análisis bivariantes exploratorios por las limitaciones de espacio de la práctica.

Selección de atributos

Ya hemos sacado algunas informaciones útiles a través del análisis exploratorio visual (EDA) y eliminado algún atributo. Pero ahora, para poder responder a la pregunta inicial, vamos a crear 3 modelos de los cuales esperamos poder obtener algún tipo de conclusión que nos ayude a tal fin.

De la exploración de los datos, correlaciones y gráficas, y del contenido de muchas etiquetas de vino, nos parecen interesantes los siguientes atributos:

- alcohol
- volatile.acidity
- total.sulfur.dioxide
- sulphates
- density

Estas características serán las que a priori utilizaremos en nuestro primer modelo y que, a tenor de los resultados, iremos enriqueciendo mediante la adición de nuevos atributos, de forma que si los resultados no son óptimos, podremos incluir todos los atributos ya que de alguna manera, seguramente influyen en la calidad del vino.

Creación de modelos

Antes de aplicar modelos hay que preparar los dataset a introducir al modelo, que inicialmente incluirán los atributos antes señalados.

También crearemos un conjunto de datos de entrenamiento con el que entrenar el modelo, y otro de test con el que validaremos su bondad. No debe haber elementos compartidos entre ambos subconjuntos. Vamos a proceder diviendo el dataset pero de forma que respetemos la proporción de cada valor de la variable objetivo, *quality*, de manera estratificada. Así los datos de entrenamiento tendrán la misma proporción de datos según la calidad de los caldos.

```
#creamos los conjuntos train y test estratificados por quality y dimensiones 2/3
#y 1/3 respectivamente
train <- createDataPartition(y = df2$quality, p = 0.66, list = FALSE, times = 1)
datos_train <- df2[train, ]
datos_test  <- df2[-train, ]
```

Sacamos la variable objetivo de los subconjuntos, y ya estaremos listos para aplicarlos a los modelos que vamos a utilizar.

```
target_train <- datos_train$quality
target_test <- datos_test$quality

datos_train <- select(datos_train, -quality)
datos_test <- select(datos_test, -quality)

levels(target_train)
```

```
## [1] "3" "4" "5" "6" "7" "8"
```

```
levels(target_test)
```

```
## [1] "3" "4" "5" "6" "7" "8"
```

```
dim(datos_train)
```

```
## [1] 1058    9
```

```
dim(datos_test)
```

```
## [1] 541   9
```

Árbol de decisión

Nuestro primer intento va a ser tratar de clasificar los datos con un árbol de decisión para tratar de discernir si hay algunas reglas ocultas en ello. Las ventajas de este modelo son su sencillez, la facilidad de interpretación de las reglas creadas y que no necesitan ningún tipo de transformación entre los datos de entrada (tan solo que la variable objetivo debe ser de tipo **factor**).

Para generar un modelo sencillo, vamos a reducir el conjunto de atributos de la manera anteriormente expuesta. Los enumeramos de nuevo:

- alcohol
- volatile.acidity
- total.sulfur.dioxide
- sulphates
- density

```
#convertimos la variable objetivo a tipo factor
target_train <- as.factor(target_train)
target_test <- as.factor(target_test)
```

```r
#reducimos el conjunto de datos a los atributos citados
datos_train_reducido <- select(datos_train, c(9, 1, 5, 8, 6))
datos_test_reducido <- select(datos_test, c(9, 1, 5, 8, 6))
```

```r
#construimos el modelo
modelo.tree <- C50::C5.0(datos_train_reducido, target_train, rules = TRUE )
modelo.tree
```

```
##
## Call:
## C5.0.default(x = datos_train_reducido, y = target_train, rules = TRUE)
##
## Rule-Based Model
## Number of samples: 1058
## Number of predictors: 5
##
## Number of Rules: 54
##
## Non-standard options: attempt to group attributes
```

```r
#validamos el modelo
data_predicted <- predict(modelo.tree, datos_test_reducido)
confusionMatrix(data = data_predicted, reference = target_test, positive = "yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   5   0   0
##          5   3  13 155  39   5   0
##          6   0   4  74 135  31   6
##          7   0   1   2  36  30   0
##          8   0   0   0   1   1   0
##
## Overall Statistics
##
##                Accuracy : 0.5915
##                  95% CI : (0.5487, 0.6332)
##     No Information Rate : 0.427
##     P-Value [Acc > NIR] : 1.095e-14
##
##                   Kappa : 0.3512
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000 0.000000   0.6710   0.6250  0.44776 0.000000
## Specificity          1.000000 0.990440   0.8065   0.6462  0.91772 0.996262
## Pos Pred Value            NaN 0.000000   0.7209   0.5400  0.43478 0.000000
## Neg Pred Value       0.994455 0.966418   0.7669   0.7216  0.92161 0.988868
## Prevalence           0.005545 0.033272   0.4270   0.3993  0.12384 0.011091
```

```
## Detection Rate       0.000000 0.000000   0.2865   0.2495  0.05545 0.000000
## Detection Prevalence 0.000000 0.009242   0.3974   0.4621  0.12754 0.003697
## Balanced Accuracy    0.500000 0.495220   0.7387   0.6356  0.68274 0.498131
```

A través del comando en R *summary (modelo.tree)* podemos ver las reglas que se han creado y que se podrían implementar para tratar de predecir la calidad de algún vino.

La precisión, contrastada contra el conjunto de pruebas, están en el entorno del 60%. Reseñar también el valor de Kappa, métrica que mide el porcentaje de aciertos más allá del que se podría conseguir haciendo predicciones meramente al azar. El modelo llega al 34%, lo cual deja bastante que desear.

Por otro lado, de la matriz de confusión vemos que, en general, clasifica bastante bien, acumulando los errores en las categorías 5 y 6 que deben tener atributos muy parecidos.

Podríamos continuar el proceso de mejora del árbol de decisión con otros atributos o con otros modelos basados en árboles de decisión, como RandomForest, pero con éstos no podríamos traducir el resultado a reglas sencillas ya que éste proviene del resultado de numerosos árboles. En cualquier caso, siempre podríamos tratar de predecir casos nuevos con el comando *predict* de R, y si la precisión fuera aceptable, podría ser de utilidad para los fines que se persiguen.

Random Forest

Vamos a probar con Random Forest con todos los datos de entrenamiento, es decir, con todos los atributos.

```
modeloRF <- randomForest(target_train ~ ., data=datos_train,
                         importance=TRUE, proximity=TRUE)
```

```
predicted_model <- predict( modeloRF, datos_test, type="class" )
print(sprintf("La precisión del árbol es: %.4f %%",
              100*sum(predicted_model == target_test) / length(predicted_model)))
```

```
## [1] "La precisión del árbol es: 68.0222 %"
```

```
#data_predicted <- predict(modeloRF, datos_test)
confusionMatrix(data = predicted_model, reference = target_test, positive = "yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   1   0   0
##          5   2  13 173  31   4   0
##          6   1   4  54 164  32   5
##          7   0   1   4  20  31   1
##          8   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.6802
##                  95% CI : (0.6391, 0.7194)
##     No Information Rate : 0.427
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4834
##
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000 0.000000   0.7489   0.7593   0.4627  0.00000
## Specificity          1.000000 0.998088   0.8387   0.7046   0.9451  1.00000
## Pos Pred Value            NaN 0.000000   0.7758   0.6308   0.5439      NaN
## Neg Pred Value       0.994455 0.966667   0.8176   0.8149   0.9256  0.98891
## Prevalence           0.005545 0.033272   0.4270   0.3993   0.1238  0.01109
## Detection Rate       0.000000 0.000000   0.3198   0.3031   0.0573  0.00000
## Detection Prevalence 0.000000 0.001848   0.4122   0.4806   0.1054  0.00000
## Balanced Accuracy    0.500000 0.499044   0.7938   0.7319   0.7039  0.50000
```

Notamos una sensible mejoría. Alcanzamos una precisión del entorno del 70% y kappa rondando el 51%. Debemos indicar que kappa es un indicador menos optimista que la precisión.

Regresión logística

Por último trataremos de explicar con un modelo de regresión logística la variabilidad del atributo objetivo, *quality*, que actuará como variable dependiente. Utilizaremos los 5 campos del árbol de decisión como variables explicativas y analizaremos los resultados. No podemos aplicar un modelo lineal múltiple ya que, aunque la variable dependiente es numérica, no se trata de un valor continuo, sino de un convenio o un orden de calidad, es decir, actúa como una variable cualitativa más que cuantitativa. Por ellos utilizaremos el modelo de regresión logística, diseñado para estos casos.

Tenemos que hacer, previamente, una transformación en la variable dependiente porque esta debe tomar dos valores. En este sentido crearemos una nueva variable, *bueno*, que tomará el valor "malo" cuando quality es menor o igual que 7 y "bueno", para vinos con calidad superior.

```
df2$bueno <- cut(df$quality, breaks = c(0, 7, 10),
    labels = c("malo", "bueno"))
df2$bueno <- as.factor(df2$bueno)
```

```
#generación del modelo de regresión logística
modeloREG=glm(formula=bueno ~ alcohol + volatile.acidity +
              total.sulfur.dioxide + sulphates + density, data = df2,
              family=binomial(link=logit))
summary(modeloREG)
```

```
##
## Call:
## glm(formula = bueno ~ alcohol + volatile.acidity + total.sulfur.dioxide +
##     sulphates + density, family = binomial(link = logit), data = df2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8724  -0.1349  -0.0687  -0.0379   3.3101
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           72.78153  140.99481   0.516  0.60571
## alcohol                1.05304    0.24817   4.243  2.2e-05 ***
## volatile.acidity      -1.27459    1.72294  -0.740  0.45944
## total.sulfur.dioxide  -0.02053    0.01183  -1.736  0.08259 .
## sulphates              3.88657    1.37119   2.834  0.00459 **
## density              -90.89024  140.30577  -0.648  0.51711
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 197.32  on 1598  degrees of freedom
## Residual deviance: 152.04  on 1593  degrees of freedom
## AIC: 164.04
##
## Number of Fisher Scoring iterations: 9
```

Por los valores del p-value que arroja el modelo podemos concluir que *alcohol* y *sulphates* son variables significativas ($\Pr(>|z|) < 0.05$), que aportan información relevante, mientras que los demás atributos no contribuyen sifnificativa a mejorar la calidad de los resultados. Podemos verlo a través de los *odd-ratios* que nos indican el grado de asociación entre las variables:

```
#cálculo de los OR
exp(modeloREG$coefficients)
```

```
##          (Intercept)              alcohol     volatile.acidity
##         4.060857e+31         2.866365e+00         2.795469e-01
## total.sulfur.dioxide            sulphates              density
##         9.796764e-01         4.874356e+01         3.364106e-40
```

Alcohol y *sulphates* tienes valores mayores que la unidad, lo que significa que actúan como factores de riesgos. Es decir, es más probable un aumento en probabilidad de que el vino sea "bueno" cuando aumentan los valores de estas variables.

De todos modos, necesitamos saber si el modelo se ajusta bien a la variabilidad de los datos. Para ello utilizaremos el test de **Hosman-Lemenshow** que asume como hipótesis nula la inexistencia de diferencias entre los valores observados con los esperados.

```
hoslem.test(df2$bueno,fitted(modeloREG))
```

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

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  df2$bueno, fitted(modeloREG)
## X-squared = 1599, df = 8, p-value < 2.2e-16
```

Dado que el valor de p-value es menor que 0.05 podemos rechazar la hipótesis nula, lo que nos apunta a que hay diferencias sensibles entre lo que espera el modelo y los datos observados. Por tanto habrá que seguir afinando los parámetros del modelo.

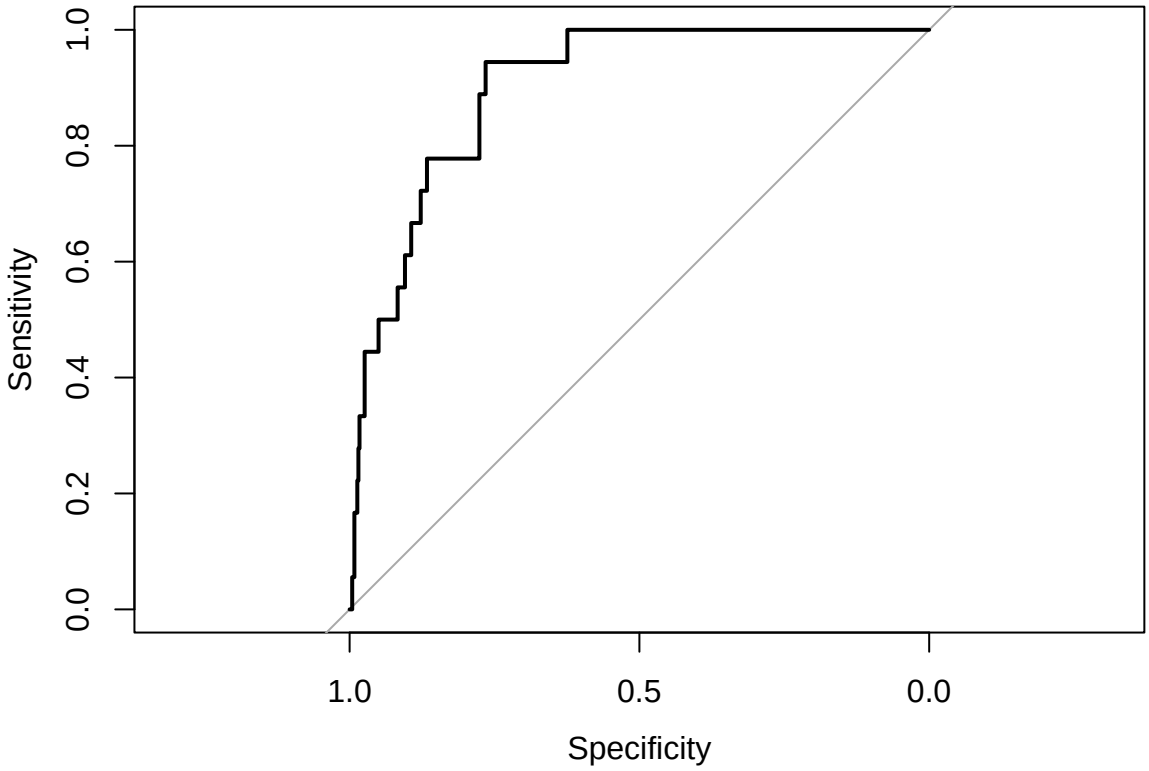
Otra manera de comprobar el resultado es a traves de la curva ROC que relaciona sensibilidad y 1-especificidad, de modo que, a mayor área bajo la gráfica, mejor precisión.

```
prob=predict(modeloREG, df2, type="response")
r=roc(df2$bueno,prob, data=df2)
```

```
## Setting levels: control = malo, case = bueno
```

```
## Setting direction: controls < cases
```

```
plot (r)
```




```
#area bajo la curva
auc(r)
```

```
## Area under the curve: 0.9018
```

El modelo es bastante bueno ya que el área bajo la curva ROC es superior al 80%.

Conclusiones

Tras el estudio de los datos hemos comprobado que existe una relación clara entre la cantidad de alcohol y los vinos recogidos en el dataset. Es la variable que más información aporta para explicar la variabilidad de los datos y aparece en todos los modelos con gran relevancia.

En cuanto a los modelos utilizados, hemos conseguido afinar un modelo logístico que, si bien no es capaz de predecir la calidad de los vinos en la escala de 0 a 10 inicial, sí lo hace, con un buen nivel de precisión, para diferenciar vinos buenos de los de calidad mediana o inferior. Esto podría ser de utilidad y ofrecer una ventaja competitiva a los productores a la hora de "diseñar" sus vinos... otro estudio sería cómo conseguir la condiciones óptimas, cosa que ya queda en manos de biólogos y enólogos.

Contribuciones	Firma
Investigación previa	EJVT, FJAG
Redacción de las respuestas	EJVT, FJAG
Desarrollo código	EJVT, FJAG