Pràctica 2: Neteja i anàlisi de les dades - Anàlisi de vins

Rigau, Pol. Tienda, Arnau

13/05/2020

Descripció del dataset

En aquesta practica hem escollit el dataset "Red Wine Quality" disponible a la pagina web Kaggle i al repositori UCI. Es pot trobar en els següents enllaços:

https://archive.ics.uci.edu/ml/datasets/wine+quality

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

Aquest dataset conté informació sobre diversos paràmetres químics resultants de l'analisis de vins blancs i negres de la regió portugesa 'Vinho verde' i una classificació segons la seva qualitat.

Amb aquestes dades es poden crear algoritmes per a la classificació de vins. Ens permet agrupar vins per les seves semblances i també determinar quins son els factors que afecten més a la seva qualitat. Així doncs, la variable classe és una nota del 0 al 10 que indica la qualitat del vi i sera la nostre indicador objectiu.

Integració i selecció de les dades

En primer lloc, carreguem llibreries que s'usen per la pràctica.

```
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(fpc)
library(ggplot2)
library(C50)
library(partykit)
```

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

```
## Loading required package: libcoin
```

```
## Loading required package: mvtnorm
```

```
library(cluster)
library(class)
```

Es comennça l'anàlisi carregant les dades:

```
wine <- read.csv(paste0(getwd(), '/winequality-red.csv'), header = TRUE)
```

Es mostren quins son els atributs:

```
colnames(wine)
```

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

Tenim 11 descriptors i una classe, que és la **qualitat**.

A continuació realitzem un primer anàlisi dels atributs que tenim:

```
str(wine)
```

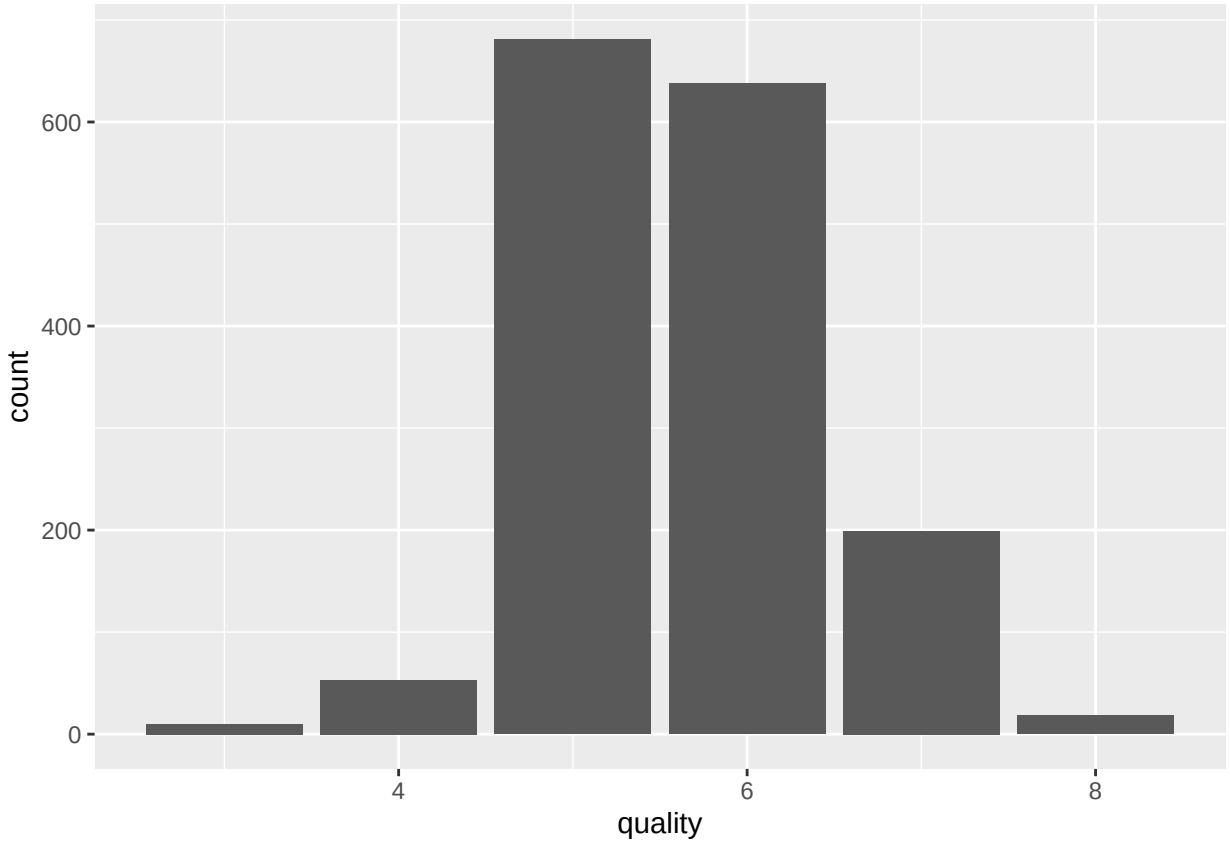
```
## 'data.frame':    1599 obs. of  12 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...
##  $ residual.sugar      : 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 ...
```

A partir de la observació del Dataset determinem que tots els atributs poden ser descriptors potencials de la classe **quality**. De moment no descartarem cap dels disponibles.

Observem que totes les dades que tenim, excepte les de l'atribut objectiu **quality**, són números. No és necessari canviar els tipus, sinó que podem treballar amb els d'origen.

Si gràfiquem **quality** mitjançant un diagràma de barres per a veure el rang de valors disponibles, observem que només disposem de valors entre 3 i 8, en les proporcions seguents:

```
ggplot(data = wine, aes(x = quality)) + geom_bar()
```



Per tant, no tenim valors de vins excelents o molt dolents entre la mostra.

Neteja de les dades

Zeros o elements buits

Per a la neteja del dataset, ens fixarem en cel · les buides (valor "") i cel · les amb valor NA.

```
colSums(is.na(wine), wine == "")
```

```
##        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
```

Les dades no contenen valors desconeguts. Pel que fa a valors igual a 0, no se'n realitza cap tractament especial, ja que són valors possibles que poden prendre alguns dels atributs. Per exemple, podem veure que per la variable àcid cítric tenim registres que tenen valor 0, però aquest resultat pot correspondre a una mesura real.

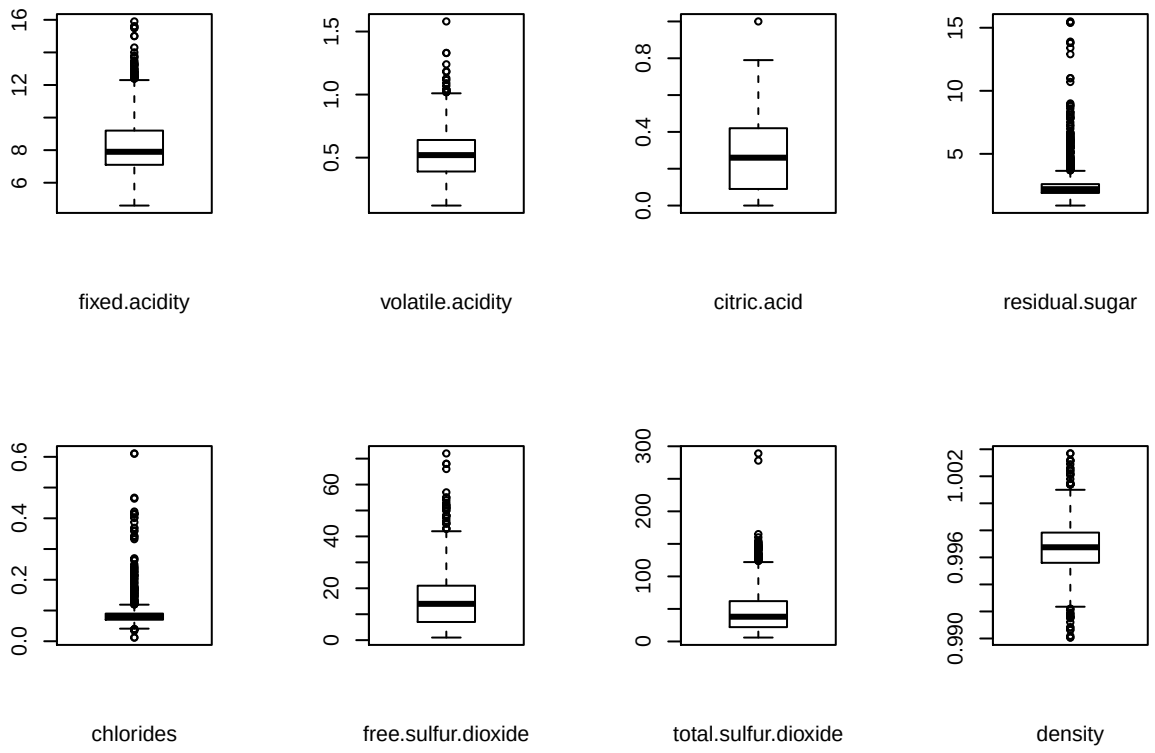
Si s'haguéssin detectats valors perduts, com estem tractant amb valors numèrics continus, es podrien haver realitzat diferents operacions. Les més interessants, segons la nostra opinió, serían les dues següents:

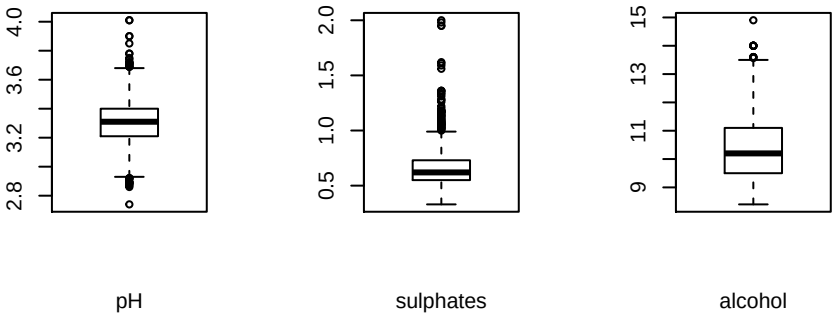
- Canviar els valors perduts pel valor de la mitjana de l'atribut. D'aquesta manera s'evitaria variar massa la distribució de probabilitat de la variable.
- Utilitzar un mètode d'imputació com kNN, en el qual s'utilitzarien els valors dels k veïns més propers per a triar a quin grup pertany, i assignar-li un valor. Es poden utilitzar diferents mètriques de distància.

Outliers

A continuació realitzarem una exploració dels outliers, pel que graficarem els bloxpot dels atributs números per a tenir una representació visual.

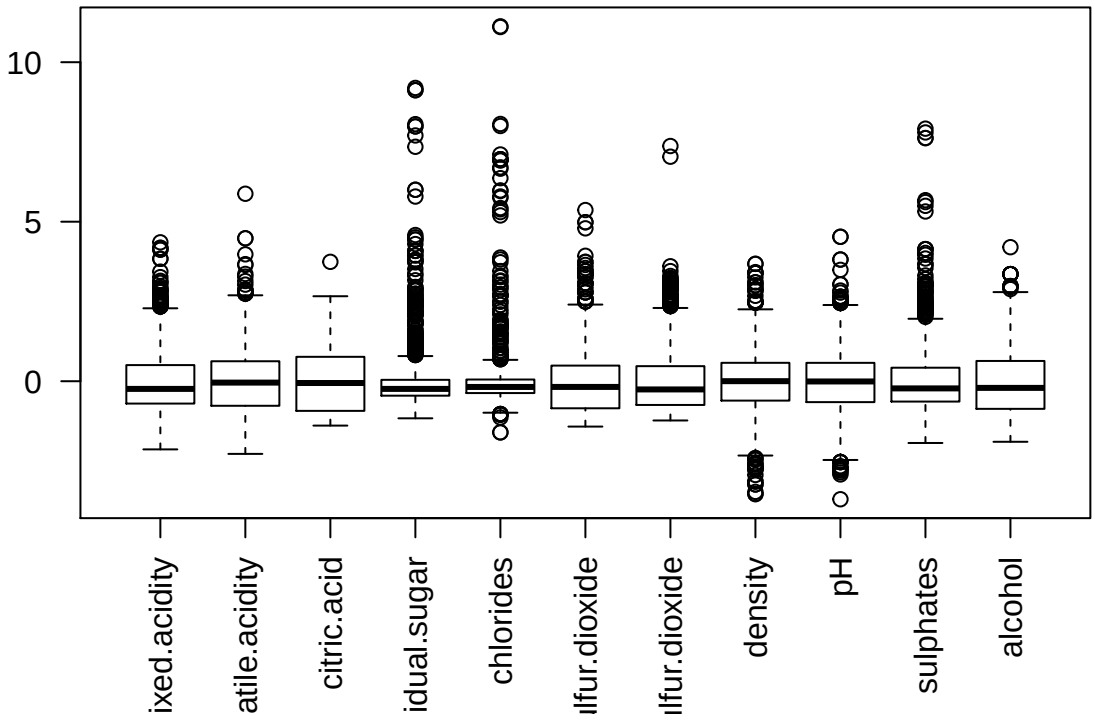
```
# Seleccionem les variables contínues
wine_cont <- select(wine,-'quality')
graph_wine = par(mfrow = c(2,4))

for (i in colnames(wine_cont)){
    boxplot(wine[[i]], xlab = i)
}
```





Si ens volem fer una idea del tamany dels quartils comparant les diferents variables, podem normalitzar-les i mostrar tots els boxplots en una mateixa figura.

```
wine_norm<- scale(wine_cont,center=T,scale=T)
boxplot(wine_norm,las = 2)
```

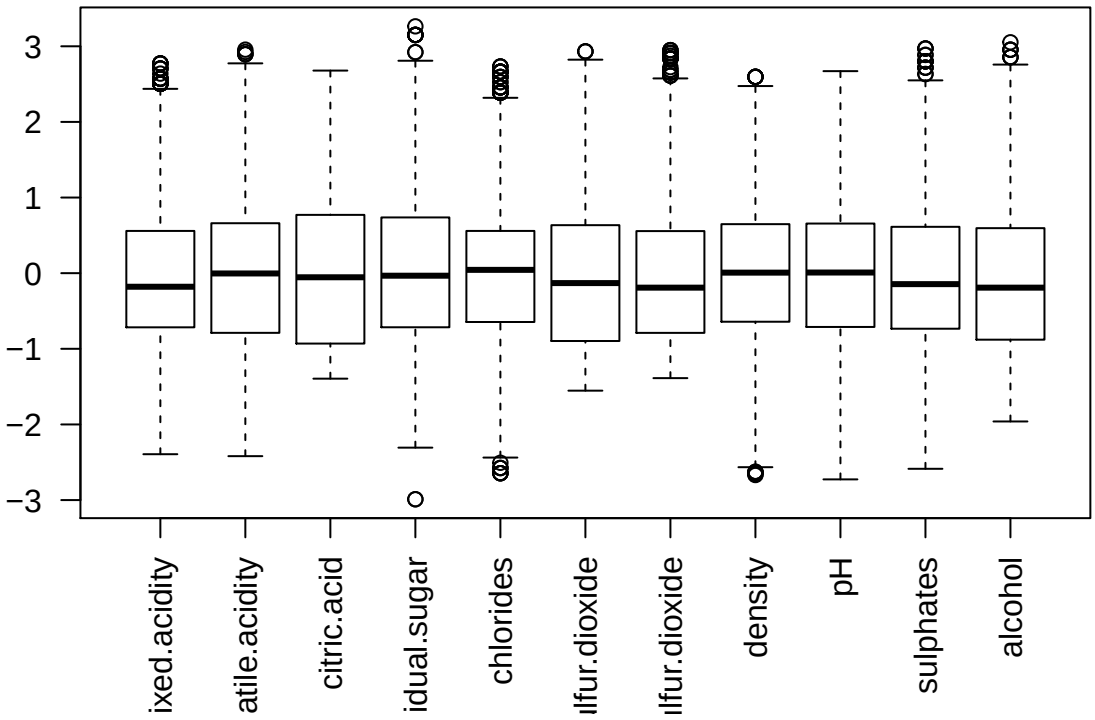


Observem que en la majoria de casos tenim valors extrems en la part de valors més elevada. Es decideix substituir els outliers pel valor de la mitjana de l'atribut.

```
# Trobem els índex dels outliers de cada una de les variables contínues i el
# substituim per la mitjana de l'atribut

remove_outliers <- function(x){
  outliers <- boxplot.stats(x)$out
  index <- x %in% outliers
  x[index] <- mean(x)
  return(x)
}

wine_cont <- sapply(wine_cont, remove_outliers)
```

Tornem a mostrar les dades normalitzades sense els outliers que acabem de substituir per les mitjanes.

```
wine_norm<- scale(wine_cont,center=T,scale=T)
boxplot(wine_norm,las = 2)
```



Anàlisi de dades

Selecció de variables

Per a seleccionar les variables que ens permeten definir la qualitat del vi, crearem un model de regressió lineal múltiple per observar quines són els atributs que guarden més relació amb la **qualitat**.

```
# Calculem el model de regressió
model <- lm(quality ~ fixed.acidity + volatile.acidity + citric.acid + residual.sugar +
              chlorides + free.sulfur.dioxide + total.sulfur.dioxide + density + pH +
              sulphates + alcohol, data=wine)
summary(model)
```

```
##
## Call:
## lm(formula = quality ~ fixed.acidity + volatile.acidity + citric.acid +
##     residual.sugar + chlorides + free.sulfur.dioxide + total.sulfur.dioxide +
##     density + pH + sulphates + alcohol, data = wine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.68911 -0.36652 -0.04699  0.45202  2.02498
##
## Coefficients:
```

```
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           2.197e+01  2.119e+01   1.036   0.3002
## fixed.acidity         2.499e-02  2.595e-02   0.963   0.3357
## volatile.acidity     -1.084e+00  1.211e-01  -8.948  < 2e-16 ***
## citric.acid          -1.826e-01  1.472e-01  -1.240   0.2150
## residual.sugar        1.633e-02  1.500e-02   1.089   0.2765
## chlorides            -1.874e+00  4.193e-01  -4.470 8.37e-06 ***
## free.sulfur.dioxide   4.361e-03  2.171e-03   2.009   0.0447 *
## total.sulfur.dioxide -3.265e-03  7.287e-04  -4.480 8.00e-06 ***
## density              -1.788e+01  2.163e+01  -0.827   0.4086
## pH                   -4.137e-01  1.916e-01  -2.159   0.0310 *
## sulphates             9.163e-01  1.143e-01   8.014 2.13e-15 ***
## alcohol               2.762e-01  2.648e-02  10.429  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.648 on 1587 degrees of freedom
## Multiple R-squared:  0.3606, Adjusted R-squared:  0.3561
## F-statistic: 81.35 on 11 and 1587 DF,  p-value: < 2.2e-16
```

A partir dels valors Pr(>|t|), podem veure la importància de cada variable per a la definició de la variable qualitat (relació entre variable dependent i variables explicatives). Observem que únicament hi ha 5 valors amb valors p inferiors a 0.01. Aquests **els considerarem atributs significatius per a la definició de la qualitat del vi**. Descartem la resta d'atributs.

```
wine_signif = select(wine,c("volatile.acidity","chlorides","total.sulfur.dioxide",
                            "sulphates","alcohol","quality"))

model <- lm(quality ~ volatile.acidity + chlorides +  total.sulfur.dioxide
            + sulphates + alcohol, data=wine_signif)
summary(model)
```

```
##
## Call:
## lm(formula = quality ~ volatile.acidity + chlorides + total.sulfur.dioxide +
##     sulphates + alcohol, data = wine_signif)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.67443 -0.38254 -0.06368  0.44893  2.07310
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           3.0048920  0.2037663  14.747  < 2e-16 ***
## volatile.acidity     -1.1419024  0.0969400 -11.779  < 2e-16 ***
## chlorides            -1.7047871  0.3916886  -4.352 1.43e-05 ***
## total.sulfur.dioxide -0.0023096  0.0005082  -4.544 5.92e-06 ***
## sulphates             0.9148320  0.1102702   8.296 2.26e-16 ***
## alcohol               0.2770979  0.0164836  16.811  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6514 on 1593 degrees of freedom
```

```
## Multiple R-squared:  0.3515, Adjusted R-squared:  0.3495
## F-statistic: 172.7 on 5 and 1593 DF,  p-value: < 2.2e-16
```

Comprovació de la normalitat i homogeneïtat de la variància.

A continuacio es farà un anàlisis de les dades disponibles.

```
summary(wine_signif)
```

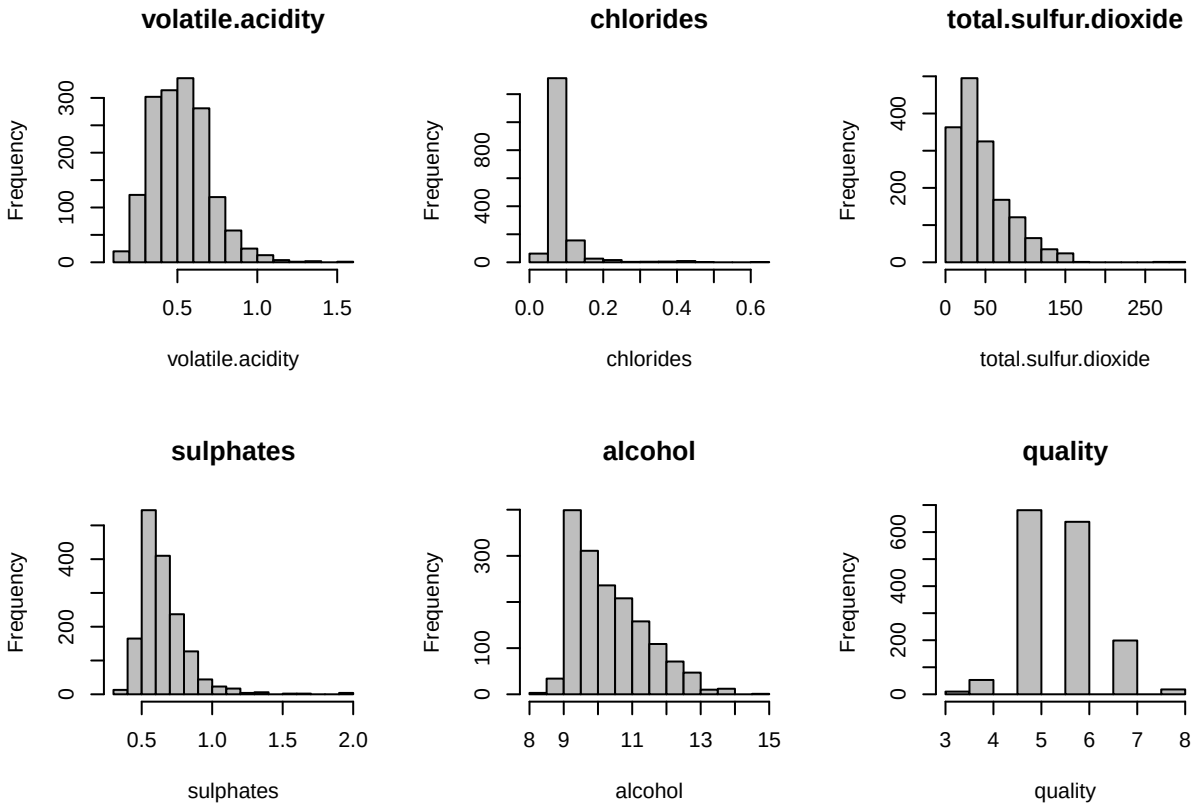
```
##  volatile.acidity   chlorides       total.sulfur.dioxide   sulphates
##  Min.   :0.1200   Min.   :0.01200   Min.   :  6.00       Min.   :0.3300
##  1st Qu.:0.3900   1st Qu.:0.07000   1st Qu.: 22.00       1st Qu.:0.5500
##  Median :0.5200   Median :0.07900   Median : 38.00       Median :0.6200
##  Mean   :0.5278   Mean   :0.08747   Mean   : 46.47       Mean   :0.6581
##  3rd Qu.:0.6400   3rd Qu.:0.09000   3rd Qu.: 62.00       3rd Qu.:0.7300
##  Max.   :1.5800   Max.   :0.61100   Max.   :289.00       Max.   :2.0000
##     alcohol         quality
##  Min.   : 8.40   Min.   :3.000
##  1st Qu.: 9.50   1st Qu.:5.000
##  Median :10.20   Median :6.000
##  Mean   :10.42   Mean   :5.636
##  3rd Qu.:11.10   3rd Qu.:6.000
##  Max.   :14.90   Max.   :8.000
```

Es realitza un altre tipus de gràfic, en aquest cas un histograma, per a observar la distribució de les dades. Aquest analisi, també ens ajudara a comprobar si segueixen una distribució normal.

```
graph_wine = par(mfrow = c(2,3))

for (i in colnames(wine_signif)){
  hist(as.numeric(wine_signif[[i]]), xlab = i, main= i, col = 'grey')
  if (!(i == 'quality')){
    results_test = shapiro.test(wine[[i]])
    print(paste('Variable: ', names(wine[i])))
    print(results_test)
  }
}
```

```
## [1] "Variable:  volatile.acidity"
##
##  Shapiro-Wilk normality test
##
## data:  wine[[i]]
## W = 0.97434, p-value = 2.693e-16

## [1] "Variable:  chlorides"
##
##  Shapiro-Wilk normality test
##
## data:  wine[[i]]
## W = 0.48425, p-value < 2.2e-16
```

```
## [1] "Variable:  total.sulfur.dioxide"
##
##  Shapiro-Wilk normality test
##
## data:  wine[[i]]
## W = 0.87322, p-value < 2.2e-16

## [1] "Variable:  sulphates"
##
##  Shapiro-Wilk normality test
##
## data:  wine[[i]]
## W = 0.83304, p-value < 2.2e-16

## [1] "Variable:  alcohol"
##
##  Shapiro-Wilk normality test
##
## data:  wine[[i]]
## W = 0.92884, p-value < 2.2e-16
```



S'observa, com ja s'havia vist en el bloxplot, les variables no presenten una distribució normal. Cap d'elles cumpleix la hipotesi nula del test de Shapiro-Wilk, pel que podem determinar que no compleixen la normalitat. El p-value es inferior al nivell de significació 0.05.

Es realitza a continuació el test per a comprobar la homogeneïtat de les dades. Com que aquestes no compleixen la normalitat, es realitza el test de Fligner-Killeen.

```
fligner.test(wine_signif)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  wine_signif
## Fligner-Killeen:med chi-squared = 6643.9, df = 5, p-value < 2.2e-16
```

Els resultats obtinguts són contraris a la hipotesi, obtenint un p-value inferior al nivell de significació 0.05, pel que podem certificar que les variables no compleixen homogeneïtat.

Aplicació de proves estadístiques per a comparar els grups de dades

Mètode 1 - Correlació entre variables

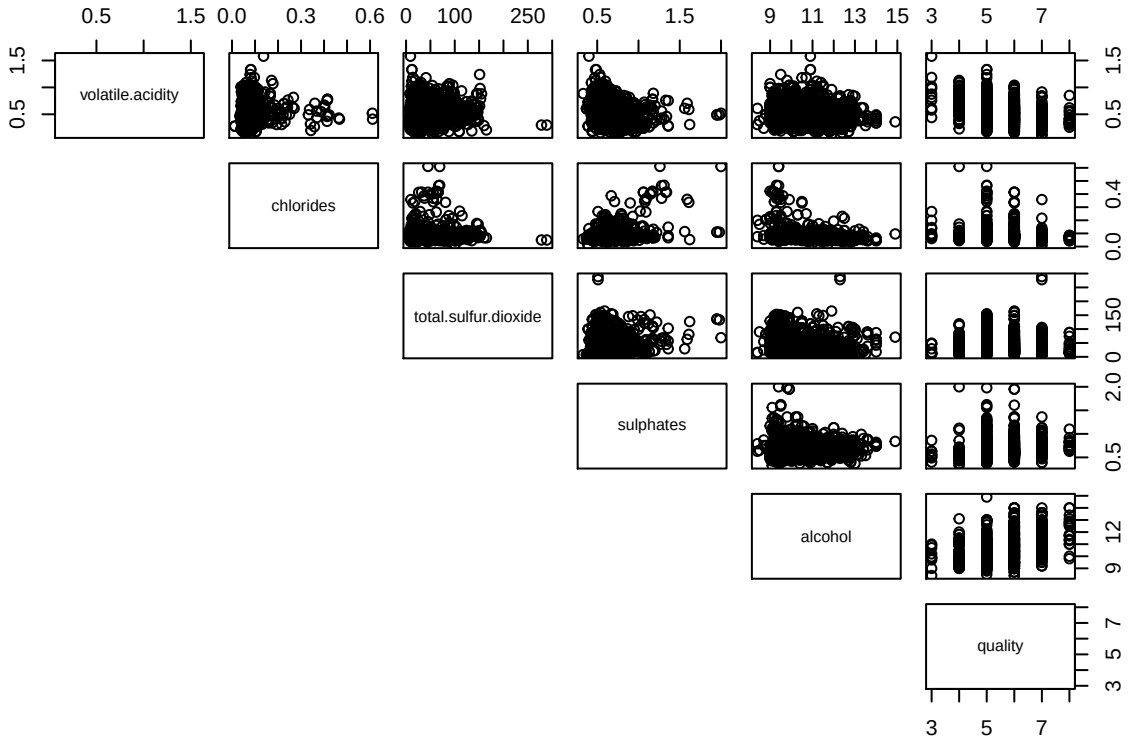
A continuació s'estudiara la correlació de les dades, és a dir, la relació que tenen entre cada una d'elles. Per començar es farà de manera numèrica.

```
cor(wine_signif)
```

```
##                       volatile.acidity   chlorides total.sulfur.dioxide
## volatile.acidity           1.00000000  0.06129777           0.07647000
## chlorides                  0.06129777  1.00000000           0.04740047
## total.sulfur.dioxide       0.07647000  0.04740047           1.00000000
## sulphates                 -0.26098669  0.37126048           0.04294684
## alcohol                   -0.20228803 -0.22114054          -0.20565394
## quality                   -0.39055778 -0.12890656          -0.18510029
##                         sulphates     alcohol    quality
## volatile.acidity     -0.26098669 -0.20228803 -0.3905578
## chlorides             0.37126048 -0.22114054 -0.1289066
## total.sulfur.dioxide  0.04294684 -0.20565394 -0.1851003
## sulphates             1.00000000  0.09359475  0.2513971
## alcohol               0.09359475  1.00000000  0.4761663
## quality               0.25139708  0.47616632  1.0000000
```

Seguidament, visualitzarem diagrames de dispersió per parelles de variables, per analitzar de manera visual aquestes correlacions.

```
pairs(wine_signif,lower.panel = NULL)
```



No observem correlacions molt altes entre variables, de manera que podem seguir utilitzant-les totes sense tenir problemes de multicolinealitat.

Sobre el model de regressió lineal múltiple que hem creat, tenia un valor de $R^2 = 0.35$, de manera que únicament explicava el 35% de la variància de la variable qualitat.

Mètode 2 - Kmeans

S'utilitzarà un mètode no supervisat per a poder determinar si podem agrupar els vins en diferents grups segons les seves propietats. Per a realitzar aquest 'clustering' hem escollit el mètode kmeans.

Per a realitzar l'anàlisi amb aquest mètode, en primer lloc necesitem tenir un set de dades no supervisat.

```
wine_signif_ns <- wine_signif
wine_signif_ns$quality <- NULL
```

Un cop tenim el data set preparat podem aplicar l'algoritme. Per a fer-ho, bé podem determinar el nombre de grups k que ens interesa o bé aplicar la funció pamk() que optimitza el nombre de grups més adient pel nostre dataset.

Aplicarem aquesta segona opció:

```
kmeans_wine<-pamk(wine_signif_ns)
```

Comprovem l'error que obtenim:

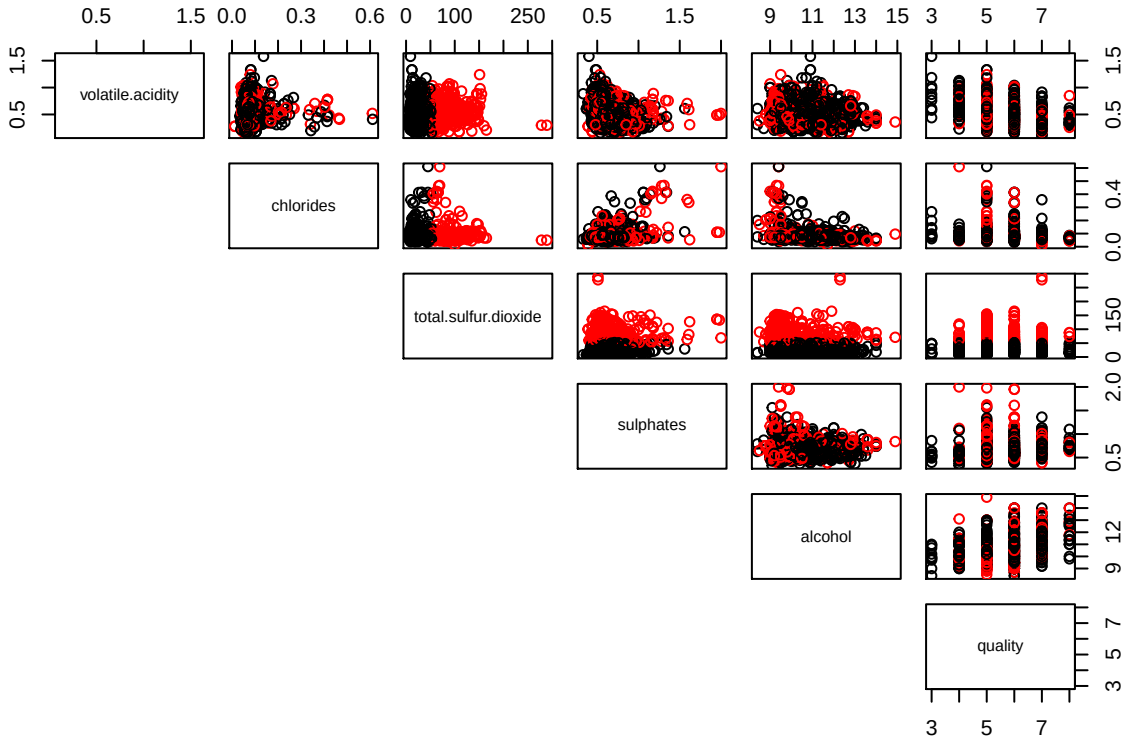
```
d  <- daisy(wine_signif_ns)
sil <- silhouette(kmeans_wine$pamobject, d)
mean(sil[,3])
```

```
## [1] 0.6235003
```

S'obté un precisió de 62%.

Per a veure graficament els resultats del algoritme tornarem a observar les nostres variables significatives amb el gràfic *pairs()*, però aquest cop colorejant segons a quin grup dels obtingut pertany.

```
wine_signif_ns$group <- as.factor(kmeans_wine$pamobject$clustering)
pairs(wine_signif, lower.panel = NULL, col=wine_signif_ns$group)
```

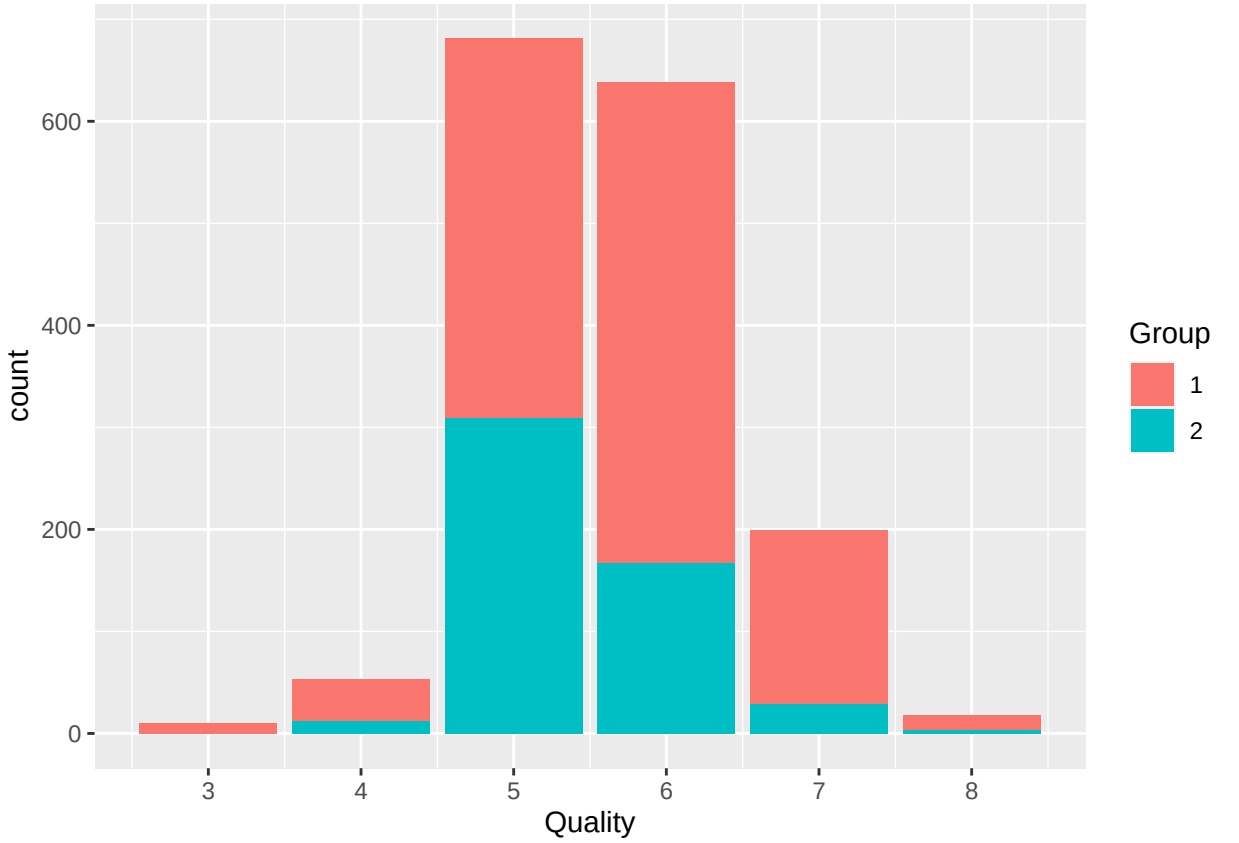


Observem que la variable que té mes influencia en la classificació dels grups es **total.sulfur.dioxide**. Els nostres grups semblen dividir-se segons la quantitat de diòxid de sulfur present al vi. En cambi, no segueixen cap patró respecte les altres variables.

Per a comprobar visualment de forma més precisa si aquesta agrupació és representativa per a determinar la qualitat dels vins, realitzarem un histograma amb només aquest atribut.

```
graph <- ggplot(wine_signif_ns[1:nrow(wine_signif),],
                aes(x=wine_signif$quality, fill=wine_signif_ns$group)) + geom_bar()
graph + scale_fill_discrete(name = "Group") + labs(x = 'Quality') +
        scale_x_continuous(breaks = c(seq(3,8)))
```

```
## Warning: Use of ‘wine_signif_ns$group‘ is discouraged. Use ‘group‘ instead.
```



Observem que si bé hi ha una lleugera tendencia a augmentar la proporció del grup 1 com major és la qualitat, aquesta agrupació no ens serviria per determinar-ho amb seguretat. Per a obtenir un algoritme que pugui determinar quines són les variables esencials i els seus valors per a conseguir una bona qualitat farem servir un mètode supervisat basat en un arbre de decisió. El primer pas per a realitzar aquest algoritme és separar les dades en les variables i en l'objectiu.

Mètode 3 - Arbres de decisió

```
y <- as.factor(wine_signif$quality)
X <- wine_signif[-6]
```

També es divideix el conjunt de dades en dos grups, el de entrenament i el de test. Hem seleccionat que el d'entrenament representi 2/3 parts del total de dades i el de test la resta.

```
set.seed(150)
indexes = sample(1:nrow(wine_signif), size=floor((2/3)*nrow(wine_signif)))
trainX <- X[indexes,]
trainy <- y[indexes]
testX  <- X[-indexes,]
testy  <- y[-indexes]
```

Com que hem realitzat una partició aleatoria, comprobem que les dades que tenim no siguin esbiaixades fent un petit anàlisis.

```r
summary(trainX)
```

```
##  volatile.acidity   chlorides       total.sulfur.dioxide   sulphates
##  Min.   :0.1200   Min.   :0.01200   Min.   :  6.00       Min.   :0.3300
##  1st Qu.:0.3900   1st Qu.:0.07100   1st Qu.: 22.00       1st Qu.:0.5500
##  Median :0.5200   Median :0.08000   Median : 38.00       Median :0.6200
##  Mean   :0.5295   Mean   :0.08759   Mean   : 47.11       Mean   :0.6547
##  3rd Qu.:0.6400   3rd Qu.:0.09000   3rd Qu.: 63.00       3rd Qu.:0.7300
##  Max.   :1.5800   Max.   :0.46400   Max.   :289.00       Max.   :1.9500
##     alcohol
##  Min.   : 8.40
##  1st Qu.: 9.50
##  Median :10.10
##  Mean   :10.41
##  3rd Qu.:11.10
##  Max.   :14.00
```

```r
levels(trainy)
```

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

D'aquesta manera podem determinar que la mitja per a cada variable és semblant a la que teniem abans del fraccionament i que disposem de totes les clases de qualitat presents a les dades originals.

Utilitzarem l'algoritme d'arbre de decisió C5.0 de la llibreria C50, carregada previament.

Abans de realitzar-ho, especificarem uns paràmetres de control *pre-prunning*, per a evitar que sigui massa especific.

```r
ctrl = C5.0Control(CF = 0.9, minCases = 6)
model <- C50::C5.0(trainX, trainy, rules=TRUE, control = ctrl)
summary(model)
```

```
##
## Call:
## C5.0.default(x = trainX, y = trainy, rules = TRUE, control = ctrl)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sun Jun  7 20:36:38 2020
## -------------------------------
##
## Class specified by attribute 'outcome'
##
## Read 1066 cases (6 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (11/1, lift 2.0)
##  total.sulfur.dioxide > 86
##  sulphates > 0.67
##  alcohol > 10.3
##  alcohol <= 11.4
```

```
##  ->  class 5  [0.846]
##
## Rule 2: (208/45, lift 1.8)
##  volatile.acidity > 0.34
##  total.sulfur.dioxide > 46
##  alcohol <= 9.8
##  ->  class 5  [0.781]
##
## Rule 3: (36/8, lift 1.8)
##  volatile.acidity > 0.64
##  alcohol > 9.95
##  alcohol <= 10.3
##  ->  class 5  [0.763]
##
## Rule 4: (25/6, lift 1.7)
##  chlorides > 0.094
##  sulphates > 0.67
##  sulphates <= 0.75
##  alcohol <= 11.2
##  ->  class 5  [0.741]
##
## Rule 5: (170/44, lift 1.7)
##  volatile.acidity > 0.49
##  sulphates <= 0.54
##  ->  class 5  [0.738]
##
## Rule 6: (371/98, lift 1.7)
##  volatile.acidity > 0.34
##  alcohol > 9.05
##  alcohol <= 9.8
##  ->  class 5  [0.735]
##
## Rule 7: (515/196, lift 1.4)
##  volatile.acidity <= 0.92
##  sulphates <= 0.64
##  alcohol <= 11.4
##  ->  class 5  [0.619]
##
## Rule 8: (25/1, lift 2.3)
##  chlorides > 0.071
##  total.sulfur.dioxide > 25
##  total.sulfur.dioxide <= 36
##  sulphates <= 0.67
##  alcohol > 10.3
##  ->  class 6  [0.926]
##
## Rule 9: (7, lift 2.2)
##  total.sulfur.dioxide <= 11
##  sulphates <= 0.54
##  alcohol > 11.4
##  ->  class 6  [0.889]
##
## Rule 10: (21/2, lift 2.2)
##  chlorides > 0.067
```

```
##  total.sulfur.dioxide > 15
##  sulphates <= 0.63
##  alcohol > 11.4
##  ->  class 6  [0.870]
##
## Rule 11: (11/1, lift 2.1)
##  volatile.acidity <= 0.605
##  chlorides > 0.081
##  total.sulfur.dioxide <= 46
##  sulphates > 0.53
##  alcohol > 9.05
##  alcohol <= 9.3
##  ->  class 6  [0.846]
##
## Rule 12: (13/2, lift 2.0)
##  total.sulfur.dioxide <= 46
##  sulphates > 0.53
##  alcohol <= 9.05
##  ->  class 6  [0.800]
##
## Rule 13: (68/13, lift 2.0)
##  chlorides > 0.071
##  total.sulfur.dioxide > 25
##  sulphates > 0.52
##  sulphates <= 0.67
##  alcohol > 10.3
##  ->  class 6  [0.800]
##
## Rule 14: (12/2, lift 2.0)
##  chlorides > 0.071
##  sulphates <= 0.67
##  alcohol > 11.2
##  alcohol <= 11.4
##  ->  class 6  [0.786]
##
## Rule 15: (30/7, lift 1.9)
##  volatile.acidity <= 0.49
##  total.sulfur.dioxide > 15
##  sulphates <= 0.63
##  alcohol > 11.4
##  ->  class 6  [0.750]
##
## Rule 16: (33/11, lift 1.7)
##  volatile.acidity <= 0.605
##  total.sulfur.dioxide <= 46
##  sulphates > 0.53
##  alcohol <= 9.3
##  ->  class 6  [0.657]
##
## Rule 17: (59/22, lift 1.6)
##  volatile.acidity <= 0.64
##  sulphates > 0.61
##  alcohol > 9.8
##  alcohol <= 10.3
```

```
##  ->  class 6  [0.623]
##
## Rule 18: (50/21, lift 1.5)
##  volatile.acidity <= 0.34
##  alcohol <= 10.3
##  ->  class 6  [0.577]
##
## Rule 19: (537/256, lift 1.3)
##  sulphates > 0.53
##  alcohol > 9.8
##  ->  class 6  [0.523]
##
## Rule 20: (13/3, lift 6.0)
##  chlorides <= 0.084
##  total.sulfur.dioxide > 34
##  sulphates > 0.63
##  alcohol > 11.9
##  alcohol <= 12.5
##  ->  class 7  [0.733]
##
## Rule 21: (10/3, lift 5.5)
##  chlorides > 0.085
##  chlorides <= 0.094
##  total.sulfur.dioxide <= 86
##  sulphates > 0.67
##  alcohol > 10.3
##  alcohol <= 11.4
##  ->  class 7  [0.667]
##
## Rule 22: (43/14, lift 5.5)
##  total.sulfur.dioxide <= 34
##  sulphates > 0.63
##  alcohol > 11.4
##  alcohol <= 12.5
##  ->  class 7  [0.667]
##
## Rule 23: (14/5, lift 5.1)
##  volatile.acidity <= 0.5
##  sulphates > 0.68
##  alcohol > 12.5
##  ->  class 7  [0.625]
##
## Rule 24: (46/19, lift 4.8)
##  total.sulfur.dioxide <= 15
##  alcohol > 11.4
##  ->  class 7  [0.583]
##
## Rule 25: (6/2, lift 47.6)
##  volatile.acidity > 0.5
##  sulphates > 0.68
##  alcohol > 12.5
##  ->  class 8  [0.625]
##
## Default class: 5
```

```
##
##
## Evaluation on training data (1066 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##      25  308(28.9%)   <<
##
##
##     (a)   (b)   (c)   (d)   (e)   (f)    <-classified as
##    ----  ----  ----  ----  ----  ----
##                   4     1                (a): class 3
##                  26     7     2          (b): class 4
##                 415    44           1    (c): class 5
##                 126   272    23     1    (d): class 6
##                  10    53    67          (e): class 7
##                         5     5     4    (f): class 8
##
##
##  Attribute usage:
##
##   98.78% alcohol
##   90.53% sulphates
##   72.23% volatile.acidity
##   42.50% total.sulfur.dioxide
##   14.26% chlorides
##
##
## Time: 0.0 secs
```

De les 1066 dades que tenia el set d'entrenament, l'arbre de decisó ha classificat correctament 728 i incorrectament 308, el que representa un error del 28,9%. Ha trobat 25 regles de classificació en que principalment intervenen **alcohol** i **sulphates** i en menor mesura **volatile.acidity** i **total.sulfur.dioxide**. La variable **chloride** gairebé no té afectació.

A continuació comprovarem la qualitat del model amb les dades de test que teniem reservades.

```
predicted_model <- predict(model, testX, type="class")
table(testy,Predicted=predicted_model)
```

```
##      Predicted
## testy   3   4   5   6   7   8
##     3   0   0   4   1   0   0
##     4   0   0  13   5   0   0
##     5   0   0 168  49   4   0
##     6   0   0  84 112  18   2
##     7   0   0   9  35  22   3
##     8   0   0   0   2   2   0
```

```
sprintf("La precisió de l'arbre és: %.4f %%",100*sum(predicted_model == testy)
        / length(predicted_model))
```

```
## [1] "La precisió de l'arbre és: 56.6604 %"
```

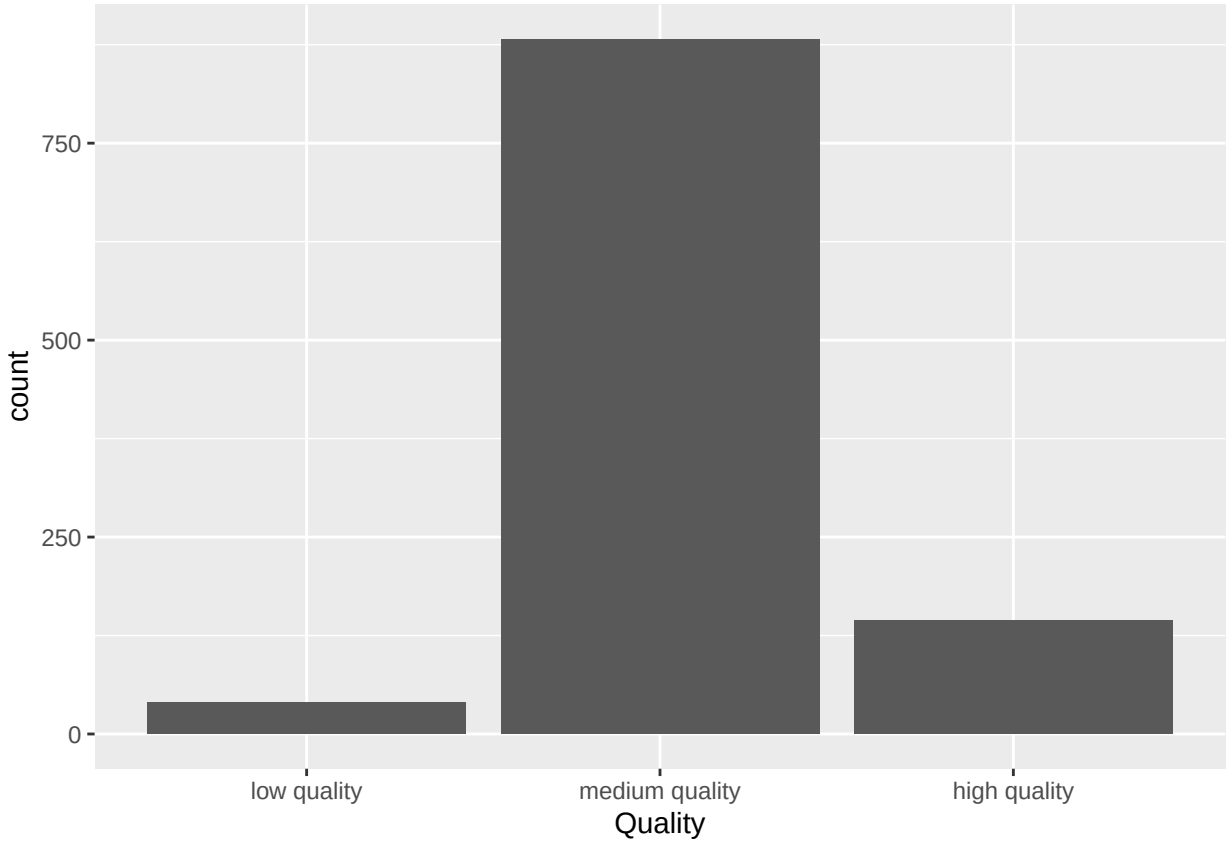
Obtenim la matriu de confusió i la precisió de l'arbre. Veiem que aquesta precisió és molt baixa ja que només encerta aproximadament la meitat de les prediccions que realitza.

Valorem la opció de perdre informació de les dades per a millorar la seva precisió. Per tant, agruparem els nivells de qualitat que tenim en tres grups, que pasaran a ser **low quality** per aquells que eren de qualitat 3 o 4, **medium quality** pels de 5 o 6 i **high quality** pels de 7 o 8.

```
new_quality <- c('low quality','low quality','medium quality', 'medium quality',
                    'high quality', 'high quality'  )
levels(trainy) <- new_quality
levels(testy) <- new_quality
```

Els nous valors es visualitzen de la seguent manera.

```
ggplot(data.frame(trainy), aes(x=trainy)) +
  geom_bar() + xlab('Quality')
```



És evident que tenim una gran quantitat de dades que pertanyen al grup de 'medium quality' (qualitat intermitja del vi) i poques a qualitat del vi bó o dolent. Ens sembla una bona representació de la realitat ja que és habitual trobar vins blancs acceptables, però més díficil trobar molt dolent o molt bons.

```
model <- C50::C5.0(trainX, trainy, rules=TRUE, control = ctrl)
model
```

```
##
## Call:
## C5.0.default(x = trainX, y = trainy, rules = TRUE, control = ctrl)
##
## Rule-Based Model
## Number of samples: 1066
## Number of predictors: 5
##
## Number of Rules: 8
##
## Non-standard options: attempt to group attributes, confidence level:
##  0.9, minimum number of cases: 6
```

```
summary(model)
```

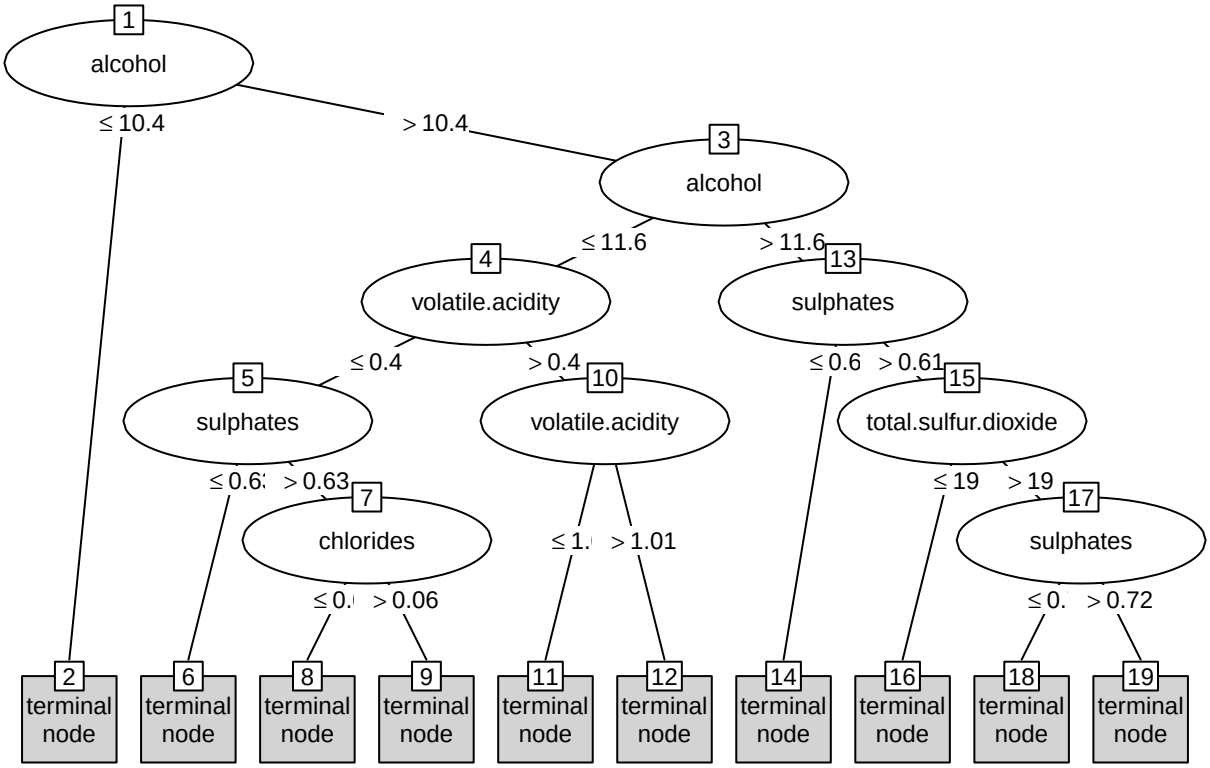
```
##
## Call:
## C5.0.default(x = trainX, y = trainy, rules = TRUE, control = ctrl)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sun Jun  7 20:36:38 2020
## -------------------------------
##
## Class specified by attribute 'outcome'
##
## Read 1066 cases (6 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (613/48, lift 1.1)
##  alcohol <= 10.4
##  ->  class medium quality  [0.920]
##
## Rule 2: (621/55, lift 1.1)
##  total.sulfur.dioxide > 19
##  sulphates <= 0.72
##  ->  class medium quality  [0.910]
##
## Rule 3: (692/62, lift 1.1)
##  volatile.acidity > 0.4
##  alcohol <= 11.6
##  ->  class medium quality  [0.909]
##
## Rule 4: (524/48, lift 1.1)
##  sulphates <= 0.61
##  ->  class medium quality  [0.907]
##
## Rule 5: (841/90, lift 1.1)
##  chlorides > 0.06
##  alcohol <= 11.6
##  ->  class medium quality  [0.892]
##
## Rule 6: (31/5, lift 6.1)
##  total.sulfur.dioxide <= 19
```

```
##  sulphates > 0.61
##  alcohol > 11.6
##  ->  class high quality  [0.818]
##
## Rule 7: (16/4, lift 5.3)
##  volatile.acidity <= 0.4
##  chlorides <= 0.06
##  sulphates > 0.63
##  alcohol > 10.4
##  alcohol <= 11.6
##  ->  class high quality  [0.722]
##
## Rule 8: (57/18, lift 5.0)
##  sulphates > 0.72
##  alcohol > 11.6
##  ->  class high quality  [0.678]
##
## Default class: medium quality
##
##
## Evaluation on training data (1066 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##       8  139(13.0%)   <<
##
##
##     (a)   (b)   (c)    <-classified as
##    ----  ----  ----
##            40          (a): class low quality
##           860    22    (b): class medium quality
##            77    67    (c): class high quality
##
##
##  Attribute usage:
##
##   92.03% alcohol
##   80.39% chlorides
##   76.55% sulphates
##   66.42% volatile.acidity
##   61.16% total.sulfur.dioxide
##
##
## Time: 0.0 secs
```

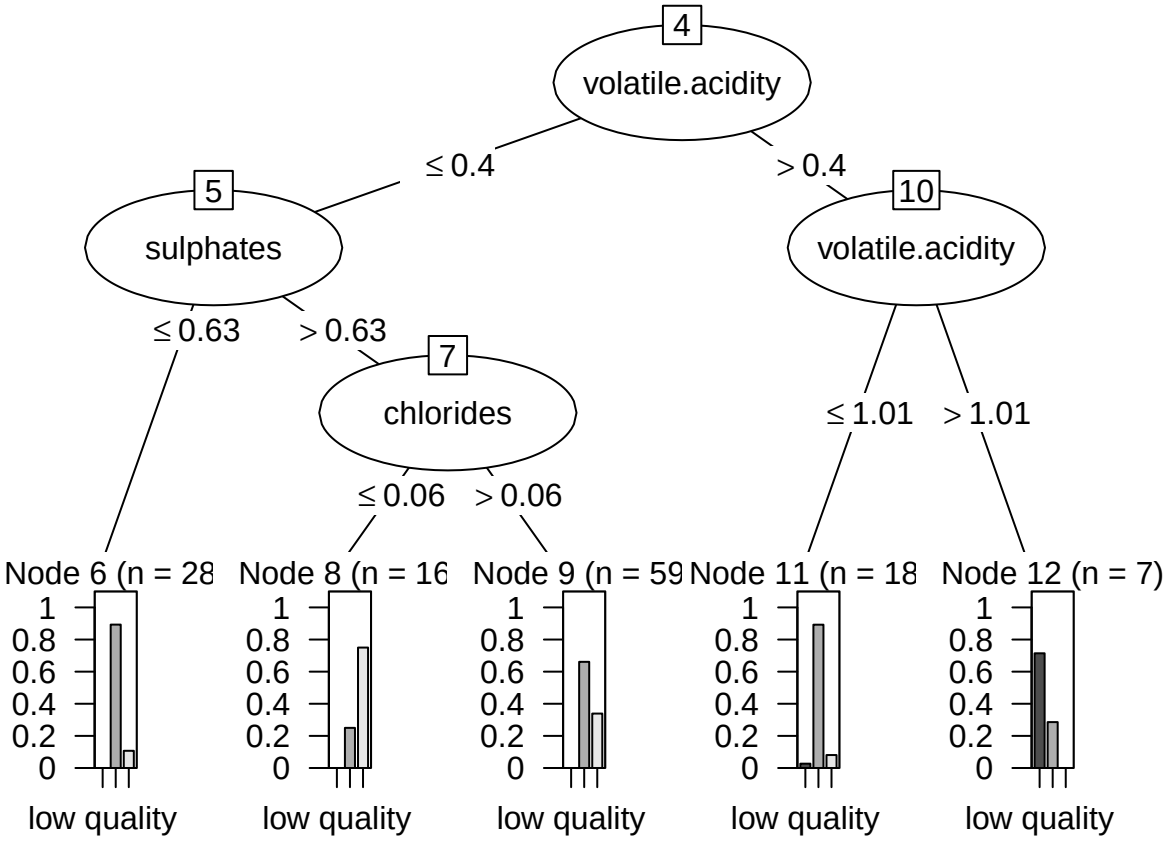
En aquest cas hem trobat una percentatge d'errors molt més baix que en el primer arbre. De 1066 casos, 139 eren incorrectes, el que representa un 13% d'error. Hem trobat 8 regles, el que també representa una simplificació respecte a les 25 anteriors.

Ara que tenim unes dades més manipulables, graficarem l'algoritme per a fer un analisis visual.

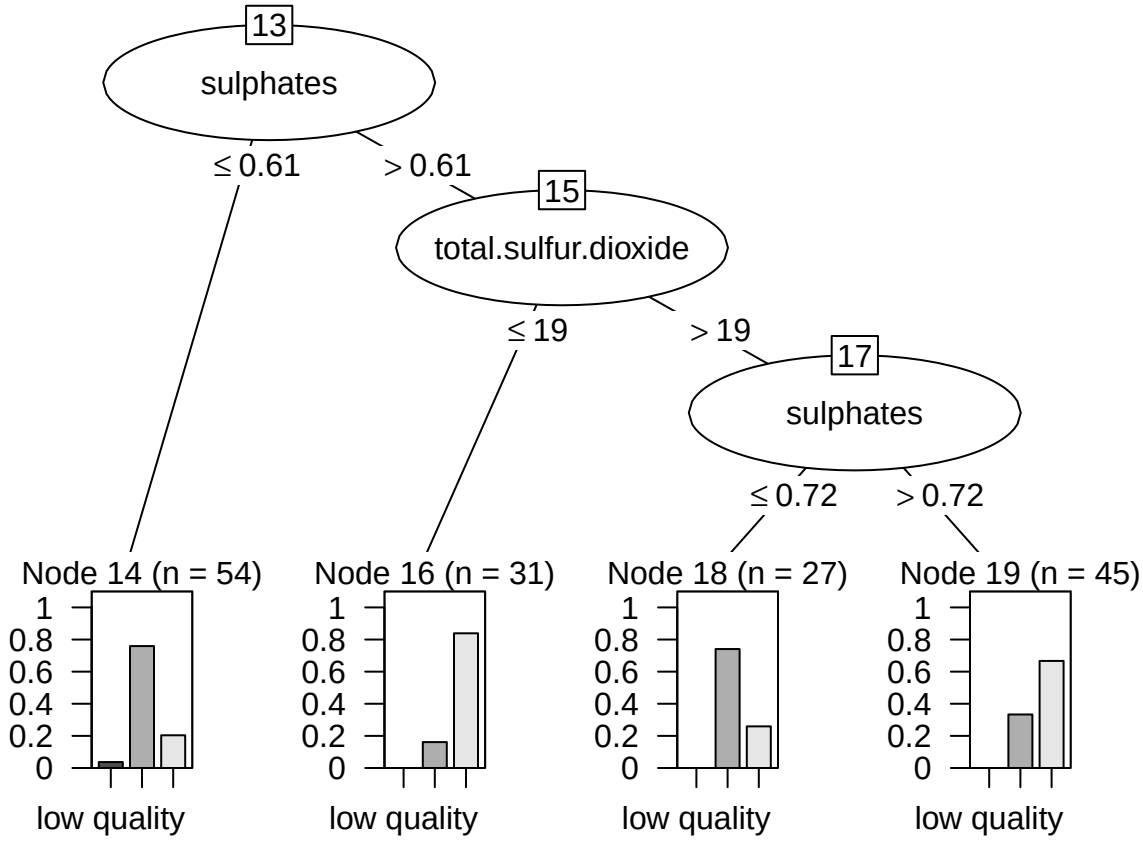
```
model <- C50::C5.0(trainX, trainy)

#Es gràfica el model complet
plot(model, gp = gpar(fontsize = 9), terminal_panel = node_terminal)
```




```
#S'amplien les dues branques principals per a tenir una millor comprensió

model %>% plot(subtree = 4)
```




```
model %>% plot(subtree = 13)
```



En primer lloc, es veu que l'algoritme que tenim no classifica cap vi com a 'low quality'. Tots els que eren d'aquesta classe s'engloben dintre de **medium quality**. Per a obtenir un vi d'alta qualitat, el més probable és que presenti una quantitat d'alcohol i de sulfats alts i clorurs i acidesa baixos.

Per a finalitzar, realitzem la comprobació amb les dades de test

```
predicted_model <- predict(model, testX, type="class")
table(testy,Predicted=predicted_model)
```

```
##                 Predicted
## testy            low quality medium quality high quality
##   low quality              3             20            0
##   medium quality           2            420           15
##   high quality             0             48           25
```

```
sprintf("La precisió de l'arbre és: %.4f %%", 100*sum(predicted_model == testy)
        / length(predicted_model))
```

```
## [1] "La precisió de l'arbre és: 84.0525 %"
```

Amb aquest nou arbre de decisió obtenim uan precisió de 84,05%, molt més elevada que amb 6 nivells.

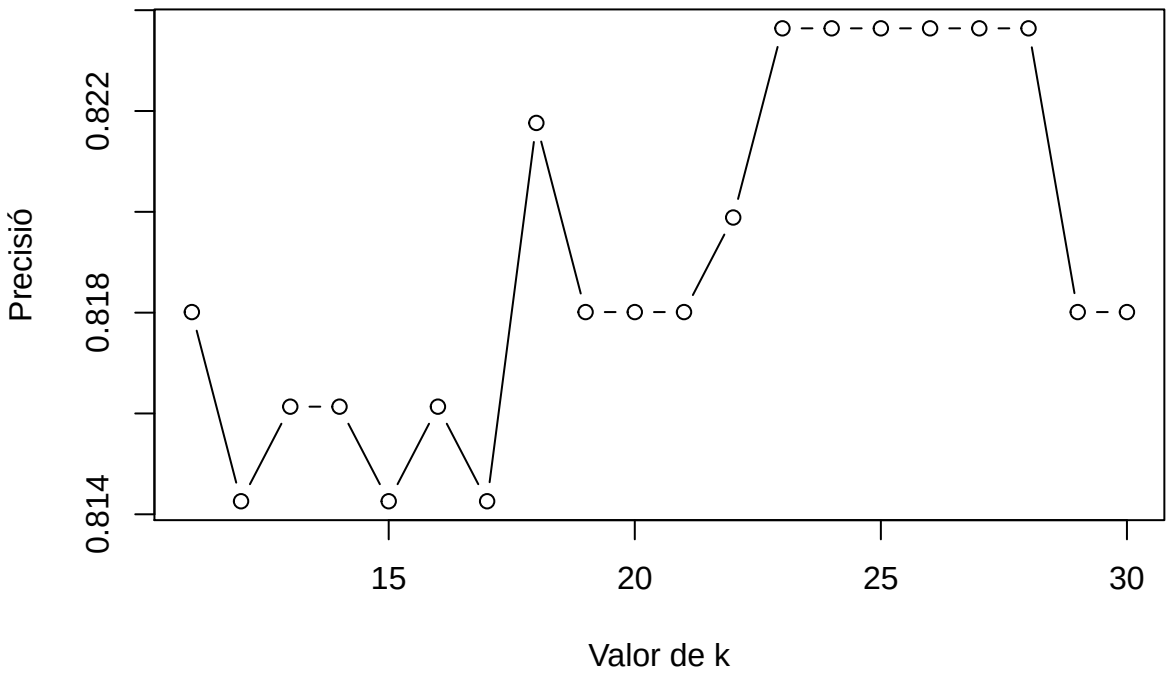
Mètode 4 - knn

A continuació treballarem amb l'algorisme supervisat knn. Seguirem treballant amb les dades discretitzades, és a dir, amb tres possibles classes. Veurem si és possible superar el 84% de precisió obtingut amb els arbres de decisió.

```r
# Realitzem l'execució de l'algorisme per a diferents valors de k. S'escull el millor
precisio = 1
for (i in 10:30){
  pr <- knn(trainX,testX,cl=trainy,k=i)
  ## Creem matriu de confusió
  perc <- table(pr,testy)
  precisio[i-10] = (perc[1,1]+ perc[2,2] + perc[3,3]) / sum(perc)
  print(sprintf("La precisió de l'algorisme knn és %.2f %% amb k = %s",
                100*precisio[i-10],i))
}
```

```
## character(0)
## [1] "La precisió de l'algorisme knn és 81.80 % amb k = 11"
## [1] "La precisió de l'algorisme knn és 81.43 % amb k = 12"
## [1] "La precisió de l'algorisme knn és 81.61 % amb k = 13"
## [1] "La precisió de l'algorisme knn és 81.61 % amb k = 14"
## [1] "La precisió de l'algorisme knn és 81.43 % amb k = 15"
## [1] "La precisió de l'algorisme knn és 81.61 % amb k = 16"
## [1] "La precisió de l'algorisme knn és 81.43 % amb k = 17"
## [1] "La precisió de l'algorisme knn és 82.18 % amb k = 18"
## [1] "La precisió de l'algorisme knn és 81.80 % amb k = 19"
## [1] "La precisió de l'algorisme knn és 81.80 % amb k = 20"
## [1] "La precisió de l'algorisme knn és 81.80 % amb k = 21"
## [1] "La precisió de l'algorisme knn és 81.99 % amb k = 22"
## [1] "La precisió de l'algorisme knn és 82.36 % amb k = 23"
## [1] "La precisió de l'algorisme knn és 82.36 % amb k = 24"
## [1] "La precisió de l'algorisme knn és 82.36 % amb k = 25"
## [1] "La precisió de l'algorisme knn és 82.36 % amb k = 26"
## [1] "La precisió de l'algorisme knn és 82.36 % amb k = 27"
## [1] "La precisió de l'algorisme knn és 82.36 % amb k = 28"
## [1] "La precisió de l'algorisme knn és 81.80 % amb k = 29"
## [1] "La precisió de l'algorisme knn és 81.80 % amb k = 30"
```

```r
# Mostrem l'evolució de la precisió en funció del valor de k
plot(c(11:30), precisio, type="b", xlab="Valor de k", ylab="Precisió")
```



A partir de valor de k = 24 aproximadament, ja no obtenim millora en els resultats. Veiem que els resultats són lleugerament pitjors que amb arbres de decisió, tot i que es considera que segueixen sent prou bons (arriben al 82% de precisió). Mostrem la matriu de confusió per al valor de k=24.

```
pr <- knn(trainX,testX,cl=trainy,k=24)

## Creem matriu de confusió
table(pr,testy)
```

```
##                 testy
## pr               low quality medium quality high quality
##   low quality              0              0            0
##   medium quality          23            435           69
##   high quality             0              2            4
```

Podem veure que l'algorisme té molts problemes per classificar vins de baixa i alta qualitat. Això és degut a que les dades inicials disposen de molts pocs valors d'aquestes qualitats, comparats amb el nombre de casos que són de mitja qualitat. Per a millors resultats, necessitaríem una mostra més diversa en el que tinguéssim aproximadament el mateix nombre de registres per a les 3 qualitats.

Representació dels resultats a partir de taules i gràfiques

A l'apartat anterior s'han utilitzat els següents mètodes d'anàlisi:

- Correlació entre variables
- K-means
- Arbres de decisió
- Knn

Per tal d'analitzar quin dels models ens serveix per classificar de la millor manera les dades que tenim, i per a cada pas que s'ha realitzat a l'apartat anterior, s'han anat realitzant les visualitzacions i taules necessàries durant l'apartat 4.3. Així doncs, la resposta d'aquest apartat número 5 es troba en l'apartat anterior. S'ha realitzat d'aquesta manera per tal de poder seguir el fil de les explicacions quan donem detalls de les decisions que es prenen en cada pas de la modelització.

Conclusions

A partir de l'anàlisi realitzat, s'han pogut extreure les següents conclusions.

- Disposàvem d'onze atributs al començar l'anàlisi, dels quals s'ha pogut detectar que **únicament 5 d'ells** tenien rellavància a l'hora de definir la qualitat del vi. Aquestes 5 variables **expliquen un 35% de la variància de la variable qualitat**. Les variables són **volatile.acidiy, chlorides, total.sulfur.dioxide", sulphates** i **alcohol**.
- No s'han detectat valors buits ni "NA", però sí outliers. **L'estratègia ha estat substituir-los per la mitja de l'atribut.**
- No s'ha detectat cap correlació entre les variables utilitzades.
- Utilitzant un algorisme no supervisat, en aquest cas **K-means**, s'ha obtingut una **mala classificació**. Té sentit tenint en compte que ha de classificar en 6 classes diferents, i que les dades d'entrenament són molt poc heterogènies (hi ha molts registres d'una classe determinada, i pocs de les altres)
- Utilitzant **arbres de decisió**, s'aconsegueix una **precisió propera al 50%**. No ens és útil per a realitzar prediccions.
- Tenint en compte el punt anterior, s'ha decidit **discretitzar les 6 categoríes** (notes de qualitat del 3 al 8) **en 3** (low quality, medium quality, high quality). Treballant amb només 3 classes, s'obté una bona **precisió del 84%.**
- Finalment, amb l'algorisme **Knn**, s'ha buscat el valor òptim de k, **k=24**, i s'ha obtingut **una precisió de 82%**.
- De manera resumida, s'ha buscat aquelles variables que descriuen la majoria de la variància de la classe qualitat, i s'han provat algorismes supervisats i no supervisats per a predir noves mostres d'un set de test reservat per a fer les proves. Pel que fa als algorismes supervisats, primer s'ha discretitzat la variable de classe per reduir de 6 a 3 possibles valors. Amb aquest canvi, s'ha arribat a precisions de predicció del 84% amb arbres de decisió.
- S'ha determinat, a traves de l'arbre de decisió que els parametres per a obtenir un vi de bona qualitat es tenir uns nivells de sulfats i una graduació alcoholica alts i acidesa i concentració de clorurs baixos.

Bibliografía.

Red Wine Quality. https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009/undefined. 2018

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.