Tipología y ciclo de vida de los datos

PRACTICA 2

Autor: Antonio Arencibia

Descripción del dataset.

Para la realización del ejercicio se ha bajado un dataset de la página https://www.kaggle.com/danielpanizzo/wine-quality.
En éste dataset los datos hacen referencia a las características de la variante blanco del vino portugues “Vinho Verde”.

El mundo de la enología me parece muy interesante, conocer que es lo que hace a un vino bueno, mediocre o excelente a partir de sus características quimicas me resulta un método más objetivo y realista que el clasificarlos por los años de contacto del vino con el barril, normalmente de roble, para clasificarlos, como ocurre en españa. Viviendo en uno de los paises con mejores vinos del mundo, es además es un negocio rentable.

La pregunta a la que intentamos responder es ¿Existen grandes diferencias entre las caracteristicas de los niveles de calidad en los datos de los vinos de la muestra?

Cambiamos el directorio de trabajo

In [1]:
```
setwd("E:/R/tipologia_ciclo_de_vida_datos")
```

Cargamos los paquetes que necesitamos para el ejercicio

In [113]:
```
library(readr)
library(RcmdrMisc)
library(stats)
library(car)
library("IRdisplay")
```

Importamos el dataset wineQualityWhites.csv

In [9]:
```
whitewine_clean <- read.table("E:/R/tipologia_ciclo_de_vida_datos/wineQualityWhites.csv",sep=",",header=TRUE)
```

Ejecutamos la función attach() para no tener que escribir el dataset con cada variable.

In [4]:
```
attach(whitewine_clean)
```

Vamos a pedir las dimensiones del dataset y los tipos de variables importadas

In [5]:
```
str(whitewine_clean)
```

```
'data.frame':   4898 obs. of  13 variables:
 $ X                   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ fixed.acidity       : num  7 6.3 8.1 7.2 7.2 8.1 6.2 7 6.3 8.1 ...
 $ volatile.acidity    : num  0.27 0.3 0.28 0.23 0.23 0.28 0.32 0.27 0.3 0.22 ...
 $ citric.acid         : num  0.36 0.34 0.4 0.32 0.32 0.4 0.16 0.36 0.34 0.43 ...
 $ residual.sugar      : num  20.7 1.6 6.9 8.5 8.5 6.9 7 20.7 1.6 1.5 ...
 $ chlorides           : num  0.045 0.049 0.05 0.058 0.058 0.05 0.045 0.045 0.049 0.044 ...
 $ free.sulfur.dioxide : num  45 14 30 47 47 30 30 45 14 28 ...
 $ total.sulfur.dioxide: num  170 132 97 186 186 97 136 170 132 129 ...
 $ density             : num  1.001 0.994 0.995 0.996 0.996 ...
 $ pH                  : num  3 3.3 3.26 3.19 3.19 3.26 3.18 3 3.3 3.22 ...
 $ sulphates           : num  0.45 0.49 0.44 0.4 0.4 0.44 0.47 0.45 0.49 0.45 ...
 $ alcohol             : num  8.8 9.5 10.1 9.9 9.9 10.1 9.6 8.8 9.5 11 ...
 $ quality             : int  6 6 6 6 6 6 6 6 6 6 ...
```

Tansformamos la variable quality a tipo factor.

In [61]:
```
whitewine_clean$quality = as.factor(whitewine_clean$quality)
```

In [62]:
```
str(whitewine_clean)
```

```
'data.frame':   4898 obs. of  13 variables:
 $ X                   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ fixed.acidity       : num  7 6.3 8.1 7.2 7.2 8.1 6.2 7 6.3 8.1 ...
 $ volatile.acidity    : num  0.27 0.3 0.28 0.23 0.23 0.28 0.32 0.27 0.3 0.22 ...
 $ citric.acid         : num  0.36 0.34 0.4 0.32 0.32 0.4 0.16 0.36 0.34 0.43 ...
 $ residual.sugar      : num  20.7 1.6 6.9 8.5 8.5 6.9 7 20.7 1.6 1.5 ...
 $ chlorides           : num  0.045 0.049 0.05 0.058 0.058 0.05 0.045 0.045 0.049 0.044 ...
 $ free.sulfur.dioxide : num  45 14 30 47 47 30 30 45 14 28 ...
 $ total.sulfur.dioxide: num  170 132 97 186 186 97 136 170 132 129 ...
 $ density             : num  1.001 0.994 0.995 0.996 0.996 ...
 $ pH                  : num  3 3.3 3.26 3.19 3.19 3.26 3.18 3 3.3 3.22 ...
 $ sulphates           : num  0.45 0.49 0.44 0.4 0.4 0.44 0.47 0.45 0.49 0.45 ...
 $ alcohol             : num  8.8 9.5 10.1 9.9 9.9 10.1 9.6 8.8 9.5 11 ...
 $ quality             : num  6 6 6 6 6 6 6 6 6 6 ...
```

Vemos que el dataset se compone de 4898 observaciones y 13 variables, todas ellas cuantitativas. La variable x es solo un indice de los datos por lo que no la tendremos en cuenta en los estudios estadisticos que hagamos.

Vemos tambien los tipos de variables que tenemos.

Al tranformar la variable quality vemos como existen 7 niveles de calidad en las observaciones.

Visualizamos los seis primeros registros del dataset

In [11]:
```
head(whitewine_clean)
```

X	fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
1	7.0	0.27	0.36	20.7	0.045	45	170	1.0010	3.00	0.45	8.8	6
2	6.3	0.30	0.34	1.6	0.049	14	132	0.9940	3.30	0.49	9.5	6
3	8.1	0.28	0.40	6.9	0.050	30	97	0.9951	3.26	0.44	10.1	6
4	7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.40	9.9	6
5	7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.40	9.9	6
6	8.1	0.28	0.40	6.9	0.050	30	97	0.9951	3.26	0.44	10.1	6

Buscamos si existen valores omitidos o perdidos en el dataset

In [121]:
```
anyNA(whitewine_clean)
```

FALSE

No existen valores omiitdos o perdidos en el dataset.

Se realiza un breve análisis descriptivo de los datos previo mediante la función summary()

In [12]:
```
summary(whitewine_clean)
```

```
       X        fixed.acidity    volatile.acidity  citric.acid
 Min.   :   1   Min.   : 3.800   Min.   :0.0800   Min.   :0.0000
 1st Qu.:1225   1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700
 Median :2450   Median : 6.800   Median :0.2600   Median :0.3200
 Mean   :2450   Mean   : 6.855   Mean   :0.2782   Mean   :0.3342
 3rd Qu.:3674   3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3900
 Max.   :4898   Max.   :14.200   Max.   :1.1000   Max.   :1.6600
 residual.sugar     chlorides       free.sulfur.dioxide total.sulfur.dioxide
 Min.   : 0.600   Min.   :0.00900   Min.   :  2.00      Min.   :  9.0
 1st Qu.: 1.700   1st Qu.:0.03600   1st Qu.: 23.00      1st Qu.:108.0
 Median : 5.200   Median :0.04300   Median : 34.00      Median :134.0
 Mean   : 6.391   Mean   :0.04577   Mean   : 35.31      Mean   :138.4
 3rd Qu.: 9.900   3rd Qu.:0.05000   3rd Qu.: 46.00      3rd Qu.:167.0
 Max.   :65.800   Max.   :0.34600   Max.   :289.00      Max.   :440.0
    density             pH          sulphates         alcohol
 Min.   :0.9871   Min.   :2.720   Min.   :0.2200   Min.   : 8.00
 1st Qu.:0.9917   1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50
 Median :0.9937   Median :3.180   Median :0.4700   Median :10.40
 Mean   :0.9940   Mean   :3.188   Mean   :0.4898   Mean   :10.51
 3rd Qu.:0.9961   3rd Qu.:3.280   3rd Qu.:0.5500   3rd Qu.:11.40
 Max.   :1.0390   Max.   :3.820   Max.   :1.0800   Max.   :14.20
    quality
 Min.   :3.000
 1st Qu.:5.000
 Median :6.000
 Mean   :5.878
 3rd Qu.:6.000
 Max.   :9.000
```

****

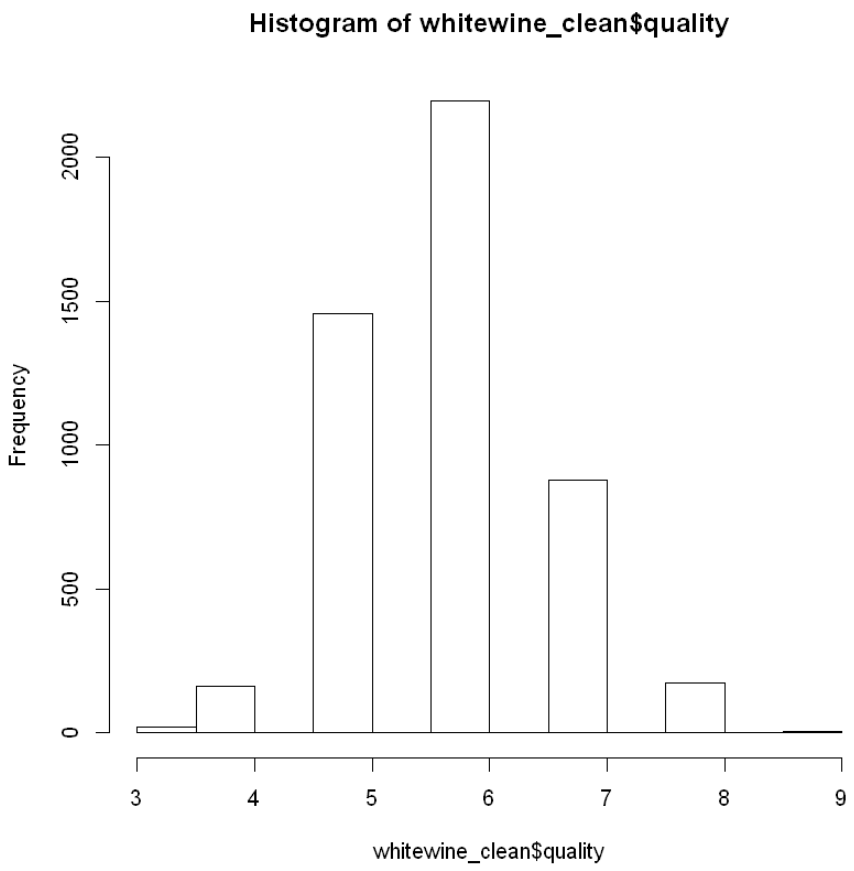
Comprobación de la normalidad

Las pruebas de normalidad comparan la distribucion de la muestra con una distribucion normal teórica. Si la forma de la distribución de la muestra no es muy diferente de la distribucion normal teórica no se rechaza la hipótesis nula:
Ho: La muestra proviene de una poblacion con distribución normal.

Si las dos distribuciones son muy diferentes rechazamos la hipótesis nula.

Dibujamos la distribución de la muestra de la variable "quality"

In [63]:
```
hist(whitewine_clean$quality)
```



Nivel de significacion de las pruebas al 5 %, que es el valor por defecto de las pruebas

Prueba Shapiro test

In [64]:
```
shapiro.test(whitewine_clean$quality)$p.value
```

1.34011123514322e-50

Rechazamos la hipótesis nula dado que el valor de p-valor es superior al 5% o 0,05 de nivel de significación.

PRUEBAS DE NORMALIDAD MULTIVARIADA

Prueba de normalidad multivariada de Mardia

In [65]:
```
library(MVN)
mardiaTest(whitewine_clean)
```

```
   Mardia's Multivariate Normality Test
---------------------------------------
   data : whitewine_clean

   g1p            : 297.4586
   chi.skew       : 242825.4
   p.value.skew   : 0

   g2p            : 915.9055
   z.kurtosis     : 1277.396
   p.value.kurt   : 0

   chi.small.skew : 242995.4
   p.value.small  : 0

   Result          : Data are not multivariate normal.
---------------------------------------
```

Prueba de normalidad multivariada de Henze-Zirkler

In [29]:
```
hzTest(whitewine_clean)
```

```
  Henze-Zirkler's Multivariate Normality Test
---------------------------------------------
  data : whitewine_clean

  HZ      : 2.920475
  p-value : 0

  Result  : Data are not multivariate normal.
---------------------------------------------
```

Prueba de normalidad multivariada de Shapiro

In [34]:
```
library(mvnormtest)
mshapiro.test(t(whitewine_clean))
```

```
        Shapiro-Wilk normality test

data:  Z
W = 0.63806, p-value < 2.2e-16
```

En todas la pruebas multivariada los resultados nos dicen que los datos del dataset no están dsipuestos en una distribución normal.

****

Homegeneidad de la varianza

Se ha ajustado un modelo de regresion lineal para explicar la varible calidad.
Las variables predictoras que son estadisticamente significativas son: Alcohol, density,fixed.acidity,free.sulfur.dioxide,pH,residual.sugar,sulphates y volatile.acidity.

In [66]:
```
RegModel.4 <- lm(quality~alcohol+density+fixed.acidity+free.sulfur.dioxide+pH+residual.sugar+sulphates+volatile.acidity,
data=whitewine_clean)
summary(RegModel.4)
```

```
Call:
lm(formula = quality ~ alcohol + density + fixed.acidity + free.sulfur.dioxide +
    pH + residual.sugar + sulphates + volatile.acidity, data = whitewine_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8246 -0.4938 -0.0396  0.4660  3.1208

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.541e+02  1.810e+01   8.514  < 2e-16 ***
alcohol              1.932e-01  2.408e-02   8.021 1.31e-15 ***
density             -1.543e+02  1.834e+01  -8.411  < 2e-16 ***
fixed.acidity        6.810e-02  2.043e-02   3.333 0.000864 ***
free.sulfur.dioxide  3.349e-03  6.766e-04   4.950 7.67e-07 ***
pH                   6.942e-01  1.034e-01   6.717 2.07e-11 ***
residual.sugar       8.285e-02  7.287e-03  11.370  < 2e-16 ***
sulphates            6.285e-01  9.997e-02   6.287 3.52e-10 ***
volatile.acidity    -1.888e+00  1.095e-01 -17.242  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7512 on 4889 degrees of freedom
Multiple R-squared:  0.2818,    Adjusted R-squared:  0.2806
F-statistic: 239.7 on 8 and 4889 DF,  p-value: < 2.2e-16
```

Ejecutamos el test para comprobar si se da la homogeneidad de varianzas mediante la funcion ncvTest()

In [67]:
```
ncvTest(RegModel.4)
```

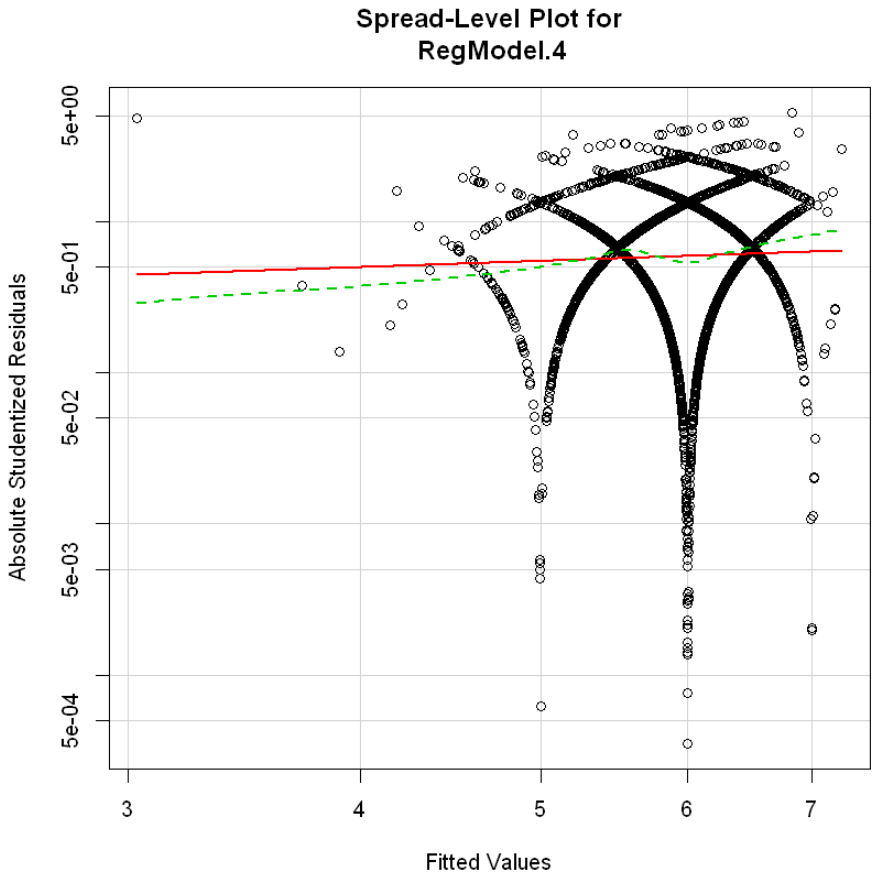
```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 16.73906    Df = 1     p = 4.288887e-05
```

Con este test comprobamos que no se cumple la homogeneidad de varianzas.

Representación gráfica de los residuos absolutos studentizados VS valores ajustados por el modelo.

In [68]:
```
spreadLevelPlot(RegModel.4)
```

```
Suggested power transformation:  0.576763
```



Indentificamos los outliers

In [69]:
```
outlierTest(RegModel.4)
```

```
      rstudent unadjusted p-value Bonferonni p
4746 -5.246380         1.6167e-07   0.00079185
2782  4.844262         1.3101e-06   0.00641690
3308 -4.601124         4.3085e-06   0.02110300
254  -4.522712         6.2497e-06   0.03061100
446  -4.477447         7.7261e-06   0.03784200
```

Podemos ver los registros de los outliers significativos de la siguiente manera para su análisis.

In [73]:
```
whitewine_clean[c(4746,2782,3308,254,446),]
```

	X	fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
4746	4746	6.1	0.260	0.25	2.9	0.047	289	440	0.99314	3.44	0.64	10.5	3
2782	2782	7.8	0.965	0.60	65.8	0.074	8	160	1.03898	3.39	0.69	11.7	6
3308	3308	9.4	0.240	0.29	8.5	0.037	124	208	0.99395	2.90	0.38	11.0	3
254	254	5.8	0.240	0.44	3.5	0.029	5	109	0.99130	3.53	0.43	11.7	3
446	446	7.1	0.320	0.32	11.0	0.038	16	66	0.99370	3.24	0.40	11.5	3

Podemos ahora calcular las Distancias .cook para ver los casos influyentes en el modelo de regresión.

In [74]:
```
dcook = cooks.distance(RegModel.4)
```

In [75]:
```
table(dcook>1)
```

```
FALSE  TRUE
 4897     1
```

In [76]:
```
which(dcook>1)
```

2782: 2782

In [77]:
```
whitewine_clean[2782,]
```

	X	fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
2782	2782	7.8	0.965	0.6	65.8	0.074	8	160	1.03898	3.39	0.69	11.7	6

Ahora vamos a ajustar un modelo de regresion robusto debido a que no se cumplian las condiciones de normalidad, homogeneidad de varianzas y por otra parte, teniamos varios casos outliers y un punto influyente.

In [80]:
```
library(MASS)
Robusto_RegModel.4 <- rlm(quality~alcohol+density+fixed.acidity+free.sulfur.dioxide+pH+residual.sugar+sulphates+volatile.acidity,
data=whitewine_clean)
summary(Robusto_RegModel.4)
```

```
Call: rlm(formula = quality ~ alcohol + density + fixed.acidity + free.sulfur.dioxide +
    pH + residual.sugar + sulphates + volatile.acidity, data = whitewine_clean)
Residuals:
     Min       1Q   Median       3Q      Max
-4.04360 -0.48617 -0.02932  0.47318  3.96592

Coefficients:
                    Value     Std. Error t value
(Intercept)          184.4088   17.1446    10.7561
alcohol                0.1565    0.0228     6.8619
density             -185.0706   17.3756   -10.6512
fixed.acidity          0.0941    0.0194     4.8618
free.sulfur.dioxide    0.0042    0.0006     6.4867
pH                     0.8079    0.0979     8.2530
residual.sugar         0.0886    0.0069    12.8382
sulphates              0.7039    0.0947     7.4337
volatile.acidity      -1.8070    0.1037   -17.4209

Residual standard error: 0.716 on 4889 degrees of freedom
```

****

Implementación un modelo de arboles de decisión.

Una vez hecho un análisis preliminar vamos a implementar sobre nuestro dataset un modelo de árboles de decisión. El objetivo es que a través de las caracteristicas se pueda preveer la puntuación de calidad. Nuestra variable clasificatoria será “quality” que tiene un rango de clasificación de 3 puntos a 9 puntos.

Creamos dos muestras. La primera muestra para la creación del arbol y la segunda para su testeo La primera muestra tendrá un 75% de los registros del dataset y la segunda muestra un 25%. Calculamos el 75% de los registros del dataset. El valor lo guardamos en la variable porcentaje_75. Con la función sample extraemos esa cantidad de valores aleatoriamente del dataset para formar la muestra con la que crearemos el arbol de decisión.

Antes de nada cargamos las librerias rpart() y rpart.plot()

In [36]:
```
library(rpart)
library(rpart.plot)
```

```
Warning message:
"package 'rpart.plot' was built under R version 3.4.3"
```

In [37]:
```
porcentaje_75<- nrow(whitewine_clean)*0.75
mw_1 <- whitewine_clean[sample(nrow(whitewine_clean), porcentaje_75),]
```

La muestra mw_1 contendrá las siguientes observaciones

In [38]:
```
nrow(mw_1)
```

3673

La distribción de los valores de la muestra mw_1 son:

In [39]:
```
summary(mw_1)
```

```
       X        fixed.acidity    volatile.acidity  citric.acid
 Min.   :   1   Min.   : 3.800   Min.   :0.0800   Min.   :0.0000
 1st Qu.:1215   1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700
 Median :2427   Median : 6.800   Median :0.2600   Median :0.3100
 Mean   :2441   Mean   : 6.856   Mean   :0.2787   Mean   :0.3326
 3rd Qu.:3660   3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3800
 Max.   :4898   Max.   :14.200   Max.   :1.1000   Max.   :1.0000
 residual.sugar     chlorides       free.sulfur.dioxide total.sulfur.dioxide
 Min.   : 0.600   Min.   :0.00900   Min.   :  2.00      Min.   :  9
 1st Qu.: 1.700   1st Qu.:0.03600   1st Qu.: 23.00      1st Qu.:108
 Median : 5.100   Median :0.04300   Median : 34.00      Median :134
 Mean   : 6.329   Mean   :0.04556   Mean   : 35.16      Mean   :138
 3rd Qu.: 9.700   3rd Qu.:0.05000   3rd Qu.: 46.00      3rd Qu.:167
 Max.   :65.800   Max.   :0.34600   Max.   :289.00      Max.   :440
    density             pH          sulphates         alcohol
 Min.   :0.9872   Min.   :2.720   Min.   :0.2200   Min.   : 8.00
 1st Qu.:0.9917   1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50
 Median :0.9937   Median :3.180   Median :0.4700   Median :10.40
 Mean   :0.9940   Mean   :3.188   Mean   :0.4895   Mean   :10.52
 3rd Qu.:0.9960   3rd Qu.:3.280   3rd Qu.:0.5500   3rd Qu.:11.40
 Max.   :1.0390   Max.   :3.820   Max.   :1.0800   Max.   :14.20
    quality
 Min.   :3.000
 1st Qu.:5.000
 Median :6.000
 Mean   :5.875
 3rd Qu.:6.000
 Max.   :9.000
```

Hacemos la misma operación para obtener la segunda muestra, con un 25% de los registros.

In [40]:
```
porcentaje_25 <- nrow(whitewine_clean)*0.25
mw_2 <- whitewine_clean[sample(nrow(whitewine_clean), porcentaje_25),]
```

La muestra mw_2 contendrá las siguientes observaciones

In [41]:
```
nrow(mw_2)
```

1224

La distribución de los valores de la muestra mw_2 son:

In [42]:
```
summary(mw_2)
```

```
       X        fixed.acidity    volatile.acidity  citric.acid
 Min.   :   1   Min.   : 4.700   Min.   :0.0800   Min.   :0.0000
 1st Qu.:1174   1st Qu.: 6.300   1st Qu.:0.2200   1st Qu.:0.2700
 Median :2342   Median : 6.800   Median :0.2700   Median :0.3200
 Mean   :2405   Mean   : 6.906   Mean   :0.2809   Mean   :0.3373
 3rd Qu.:3640   3rd Qu.: 7.400   3rd Qu.:0.3200   3rd Qu.:0.3800
 Max.   :4898   Max.   :14.200   Max.   :0.9650   Max.   :1.0000
 residual.sugar     chlorides       free.sulfur.dioxide total.sulfur.dioxide
 Min.   : 0.600   Min.   :0.01300   Min.   :  3.00      Min.   : 10
 1st Qu.: 1.800   1st Qu.:0.03500   1st Qu.: 23.00      1st Qu.:107
 Median : 6.000   Median :0.04300   Median : 34.00      Median :134
 Mean   : 6.756   Mean   :0.04614   Mean   : 35.55      Mean   :138
 3rd Qu.:10.400   3rd Qu.:0.05025   3rd Qu.: 47.00      3rd Qu.:168
 Max.   :65.800   Max.   :0.29000   Max.   :108.00      Max.   :282
    density             pH          sulphates         alcohol
 Min.   :0.9875   Min.   :2.720   Min.   :0.2200   Min.   : 8.40
 1st Qu.:0.9919   1st Qu.:3.080   1st Qu.:0.4100   1st Qu.: 9.40
 Median :0.9940   Median :3.180   Median :0.4700   Median :10.40
 Mean   :0.9942   Mean   :3.187   Mean   :0.4865   Mean   :10.48
 3rd Qu.:0.9964   3rd Qu.:3.270   3rd Qu.:0.5400   3rd Qu.:11.30
 Max.   :1.0390   Max.   :3.820   Max.   :1.0800   Max.   :14.00
    quality
 Min.   :3.000
 1st Qu.:5.000
 Median :6.000
 Mean   :5.863
 3rd Qu.:6.000
 Max.   :9.000
```

Realizamos la construcción del modelo de árbol de decisión mediante el método CART con la funcion de R rpart(), validando posteriormente los resultados.

```
Con la muestra_1 vamos a crear el árbol de decisión utilizando la funcion rpart().
La variable quality es la que clasifica y con el resto de variables construiremos el modelo.
```

Construimos el modelo

In [43]:
```
mw_1_tree <- rpart(quality ~ fixed.acidity+volatile.acidity+citric.acid+residual.sugar+chlorides+free.sulfur.dioxide+total.sulfur.dioxide+density+pH+sulpha
```

Mostramos el resultado de rpart

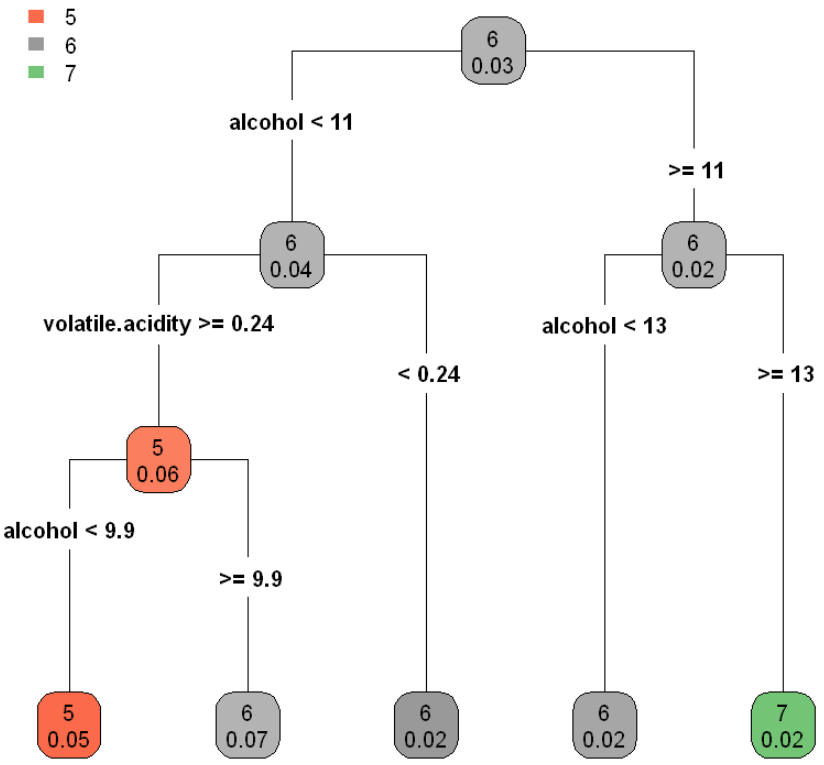
In [44]:
```
print(mw_1_tree)
```

```
n= 3673

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 3673 2001 6 (0.0041 0.035 0.29 0.46 0.18 0.033 0.0011)
  2) alcohol< 10.625 2127 1197 6 (0.0042 0.043 0.42 0.44 0.081 0.011 0.00047)
    4) volatile.acidity>=0.2375 1344  633 5 (0.0045 0.057 0.53 0.37 0.039 0.0015 0.00074)
      8) alcohol< 9.85 907  365 5 (0.0055 0.05 0.6 0.32 0.028 0.0011 0) *
      9) alcohol>=9.85 437  231 6 (0.0023 0.071 0.39 0.47 0.064 0.0023 0.0023) *
    5) volatile.acidity< 0.2375 783  348 6 (0.0038 0.019 0.24 0.56 0.15 0.027 0) *
  3) alcohol>=10.625 1546  804 6 (0.0039 0.023 0.11 0.48 0.31 0.064 0.0019)
    6) alcohol< 12.55 1282  629 6 (0.0039 0.025 0.13 0.51 0.28 0.051 0.00078) *
    7) alcohol>=12.55 264  135 7 (0.0038 0.015 0.023 0.34 0.49 0.12 0.0076) *
```

Vemos el arbol en forma gráfica con rpart.plot()

In [45]:
```
rpart.plot(mw_1_tree,type =4,extra=6)
```

```
Warning message:
"extra=6 but the response has 7 levels (only the 2nd level is displayed)"
```

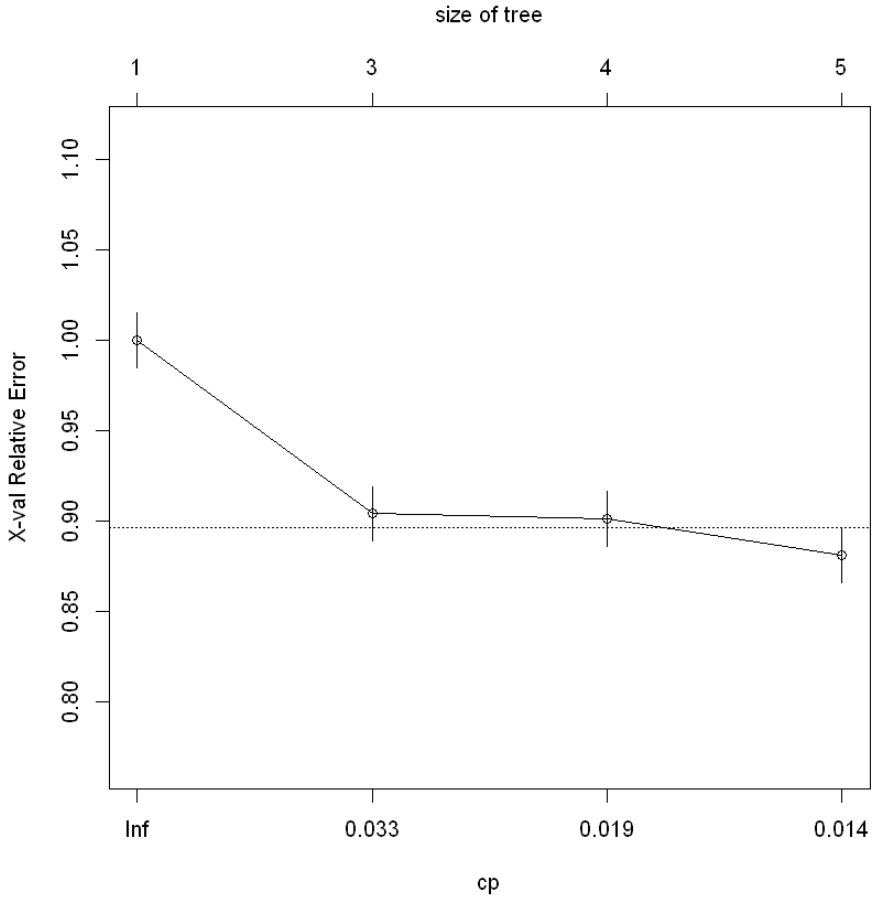


In [46]:
```
printcp(mw_1_tree)
```

```
Classification tree:
rpart(formula = quality ~ fixed.acidity + volatile.acidity +
    citric.acid + residual.sugar + chlorides + free.sulfur.dioxide +
    total.sulfur.dioxide + density + pH + sulphates + alcohol,
    data = mw_1, method = "class")

Variables actually used in tree construction:
[1] alcohol          volatile.acidity

Root node error: 2001/3673 = 0.54479

n= 3673

        CP nsplit rel error  xerror     xstd
1 0.053973      0   1.00000 1.00000 0.015083
2 0.019990      2   0.89205 0.90405 0.015142
3 0.018491      3   0.87206 0.90105 0.015141
4 0.010000      4   0.85357 0.88106 0.015132
```

Se puede ver graficamente con plotcp() la evolución del error promedio mientras aumenta el nsplit.

In [47]:
```
plotcp(mw_1_tree)
```



Utilizamos el valor de cp para el corte donde deja de disminuir el error promedio para podar el arbol. En este caso no es necesario podar el arbol ya que el error promedio no deja de disminuir. Si quisieramos podar el arbol en un punto, por ejemplo en el punto donde el valor de cp= 0.029 Utilizariamos la función prune() de la libreria rpart() para hacer la poda en ese punto.

In [ ]:
```
mw_1_tree_poda<- prune(mw_1_tree, cp=0.029)
```

```
Validamos la capacidad de predicción del árbol con nuestra muestra de validación mw_2.

mw_2 tenia un 25 % de observaciones del dataset whitewine_clean.
Para la validación utilizamos la funcion predict y mostramos los resultados de la previsión junto a los valores reales en una matriz de confusión.
```

In [48]:
```
tablaw <- table(predict(mw_1_tree, newdata = mw_2, type = "class"), mw_2$quality)
tablaw
```

```
   3  4  5  6  7  8  9
3  0  0  0  0  0  0  0
4  0  0  0  0  0  0  0
5  2 18 197 103  6  0  0
6  4 20 186 404 163 34  3
7  0  1  1 28 43 10  1
8  0  0  0  0  0  0  0
9  0  0  0  0  0  0  0
```

```
Podemos calcular la tasa de aciertos de la siguiente manera.
Sumamos los valores de la diagonal de la tabla, que son los valores verdaderos y los dividimos por el total de
valores de la tabla.
```

In [49]:
```
(sum(diag(tablaw)) / sum(tablaw))*100
```

52.6143790849673

```
La tasa de acierto es de un 52.61%. Es un mal resultado,lo que indica es la mala calidad predictiva del modelo construido.
```

```
**********************************************************************************************************************************
```

Implementación del modelo de agregación por el método K-means

```
Utilizaremos todas los datos de las varibles cuantitativas del dataframe menos la variable x.
```

Normalizamos los datos:

In [87]:
```
vinos_normalizados <- as.data.frame(scale(whitewine_clean[,2:13]))
```

Visualizamos las seis primera filas del dataset con los datos normalizados

In [88]:
```
head(vinos_normalizados)
```

fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
0.1720794	-0.08176155	0.21325843	2.8210611	-0.03535139	0.5698734	0.7444890	2.331273996	-1.24679399	-0.34914861	-1.3930102	0.1378561
-0.6574340	0.21587359	0.04799622	-0.9446688	0.14773200	-1.2528907	-0.1496693	-0.009153237	0.73995309	0.00134171	-0.8241915	0.1378561
1.4756004	0.01745016	0.54378284	0.1002720	0.19350284	-0.3121093	-0.9732363	0.358628185	0.47505348	-0.43677119	-0.3366326	0.1378561
0.4090832	-0.47860841	-0.11726599	0.4157258	0.55966962	0.6874711	1.1209768	0.525801559	0.01147916	-0.78726151	-0.4991523	0.1378561
0.4090832	-0.47860841	-0.11726599	0.4157258	0.55966962	0.6874711	1.1209768	0.525801559	0.01147916	-0.78726151	-0.4991523	0.1378561
1.4756004	0.01745016	0.54378284	0.1002720	0.19350284	-0.3121093	-0.9732363	0.358628185	0.47505348	-0.43677119	-0.3366326	0.1378561

Comenzamos con el proceso de agrupamiento o clustering.

Fijamos la semilla que utilizará el algoritmo kmeans.

In [89]:
```
set.seed(30)
```

Ejecutamos el método kmeans y le expecificamos el valor de k o el número de centros que queremos, lo que equivale al número de cluster que quermos crear.

In [90]:
```
vinos_cluster <- kmeans(vinos_normalizados,centers=5)
```

Vemos los atributos del objeto vinos_cluster

In [91]:
```
names(vinos_cluster)
```

'cluster' 'centers' 'totss' 'withinss' 'tot.withinss' 'betweenss' 'size' 'iter' 'ifault'

Con "size" vemos el número de observaciones de cada cluster

In [92]:
```
vinos_cluster$size
```

107 1130 1134 1494 1033

Podemos ver la distribución de las caracteristicas de los vinos en los diferentes cluster, diferencias por la variable "quality"

In [93]:
```
table(whitewine_clean$quality, vinos_cluster$cluster)
```

	1	2	3	4	5
3	1	3	0	8	8
4	4	30	3	33	93
5	50	296	24	671	416
6	48	614	437	647	452
7	2	172	526	118	62
8	2	15	140	17	1
9	0	0	4	0	1

Con "cluster" del objeto seguros-cluster vemos la asignación de las observaciones a cada uno de los cluster.

In [94]:
```
vinos_cluster$cluster
```

```
5 5 4 2 5 2 4 5 5 5 2 5 3 5 4 5 5 5 1 5 5 2 4 4 4 4 2 3 2 2 3 4 5 4 4 5 2 4 4 4 4 4 5 4 4 3 5 5 5 5 2 3 1 1
4 4 3 5 2 2 4 2 4 4 4 4 2 4 4 2 5 4 2 4 2 5 4 3 2 2 2 4 5 4 5 2 2 4 2 4 5 2 5 2 1 3 2 5 5 2 2 2 4 3 2 2 3 4
1 2 5 4 3 5 4 2 2 4 5 5 4 3 4 4 1 4 5 4 5 2 4 2 4 2 2 4 4 4 5 4 2 5 3 2 4 4 4 2 2 3 4 2 4 4 5 4 4 2 2 2 5 2
4 5 3 4 3 3 5 3 5 2 5 4 2 5 5 3 4 2 4 5 4 2 4 4 5 4 4 4 2 5 3 5 3 3 3 5 3 3 3 5 3 2 4 2 4 2 2 4 4 3 4 3 2 1
1 3 4 1 2 3 4 2 3 4 4 4 4 2 2 3 2 5 2 4 5 5 5 4 2 4 2 4 2 1 4 5 3 5 2 2 4 4 5 4 5 5 4 4 4 4 4 4 4 2 4 5 4 4 5
4 4 4 4 4 4 4 4 4 2 3 4 5 4 5 4 4 2 3 3 4 4 5 5 5 3 3 4 5 4 4 4 4 2 5 5 1 1 5 2 4 4 4 5 4 2 4 4 2 5 4 4 4 4 4
4 5 4 4 2 4 4 5 3 2 4 4 5 2 4 5 5 5 5 4 4 4 5 4 2 2 5 2 5 4 4 4 2 5 5 4 4 4 4 4 4 4 5 4 4 3 5 4 2 5 4 4 4 4
4 4 2 2 2 5 2 4 4 2 2 3 5 2 2 4 5 5 3 5 3 4 2 2 4 2 1 1 1 4 5 4 5 5 2 4 5 5 2 5 2 5 4 2 2 5 2 5 4 4 4 2 5 4
4 5 5 5 4 2 4 2 4 2 5 2 2 5 5 5 3 3 2 2 2 2 2 2 2 2 5 2 5 5 5 2 2 5 5 5 2 4 4 2 4 4 4 5 4 4 2 4 3 4 2 2 5
4 4 4 4 4 4 4 4 2 4 4 5 5 3 5 4 3 4 4 3 4 4 4 2 2 2 4 4 5 2 4 5 4 2 4 4 4 4 2 4 4 2 4 4 3 3 4 5 5 4 4 5 5 2 5
2 3 5 5 4 5 5 4 4 4 4 4 5 4 4 4 5 2 4 5 4 5 4 2 2 5 1 5 2 4 3 5 5 4 2 5 3 5 4 4 2 2 4 5 5 4 5 3 5 3 2 2 3 2

5 4 5 5 2 3 5 4 5 4 5 4 4 4 4 2 4 2 5 2 2 4 4 4 2 4 4 4 4 2 4 5 4 2 5 4 4 2 4 4 2 2 5 1 4 1 4 2 5 4 2 4 5 2
2 4 2 4 2 2 2 4 4 2 4 1 2 2 4 4 4 4 1 1 4 3 2 5 2 3 3 4 2 4 3 3 3 2 4 4 5 3 2 4 5 5 4 3 5 3 3 4 3 4 4 3 5 2
2 2 4 5 5 5 5 5 4 5 4 3 4 4 4 4 2 2 2 2 3 2 2 4 4 5 5 1 4 4 2 2 2 3 2 5 4 1 5 5 2 4 2 5 4 4 4 2 2 1 5 5 2 3
2 2 4 2 2 2 3 3 3 4 2 2 5 3 3 2 2 2 4 2 4 3 5 2 5 5 4 5 5 4 3 2 4 2 2 4 4 2 2 5 2 4 5 4 3 4 2 2 2 2 2 4 2 4
4 4 4 4 4 5 4 5 4 2 1 2 4 2 4 2 2 2 2 2 2 4 4 2 2 4 4 5 5 4 4 4 5 5 2 2 4 4 2 2 2 5 3 5 4 3 5 1 3 4 4 2 4 4 4
4 4 4 5 5 4 2 2 4 4 4 2 5 4 4 2 4 4 5 5 4 4 4 3 4 4 4 4 2 5 3 3 2 4 3 4 4 3 1 5 4 1 3 2 2 4 2 5 5 4 3 5 4 2
```

Con "totss" obtenemos la suma total de los cuadrados, es decir, la distancia cuadrática entre las observaciones.

In [95]:
```
vinos_cluster$totss
```

58764

Con "betweenss" obtenemos la resta de la suma de cuadrados total menos la suma de cuadrados de los cluster.Esta medida cuanto mayor mejor ya que nos referimos las diferencias entre grupos.

In [96]:
```
vinos_cluster$betweenss
```

21003.9420737984

Con "withinss" obtenemos la suma de cuadrados dentro de los cluster. Cuando menores sean está cantidad mejor ya que indicará el nivel de semejanza entre las observaciones de cada cluster.

In [97]:
```
vinos_cluster$withinss
```

1422.32264481323 8613.80123544871 7459.07261131049 12102.8996214487 8161.96181318049

Con "tot.withinss" obtenemos la suma de todas las sumas de los cuadrados dentro de los cluster. Esta cantidad nos interesa también que sea lo más pequeña posible

In [98]:
```
vinos_cluster$tot.withinss
```

37760.0579262016

Centros de los grupos o clusters

In [99]:
```
vinos_cluster$centers
```

	fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
1	-0.1911788	0.38787147	0.88589106	-0.3482179	5.35020899	0.28549581	0.134455745	0.1221169	-0.6080641	-0.22221870	-0.77027147	-0.44253723
2	-0.5672991	-0.28058007	-0.36179555	-0.4711828	-0.04527516	-0.03767996	0.002685331	-0.1646896	0.8779119	0.42293814	-0.14879063	-0.02501826
3	-0.4481224	0.16915485	-0.14641276	-0.5301228	-0.52595513	-0.30332054	-0.657421484	-1.0836564	0.1421186	-0.13588730	1.30423281	0.92247112
4	0.1638303	0.04301677	0.25827966	1.1233062	0.13921506	0.71200995	0.863102190	1.1187725	-0.2911873	0.02351128	-0.86459481	-0.32241129
5	0.8953655	0.01884278	0.09119157	-0.4911562	-0.12862112	-0.68513754	-0.543446173	-0.2609361	-0.6322423	-0.32446500	0.06123629	-0.47316328

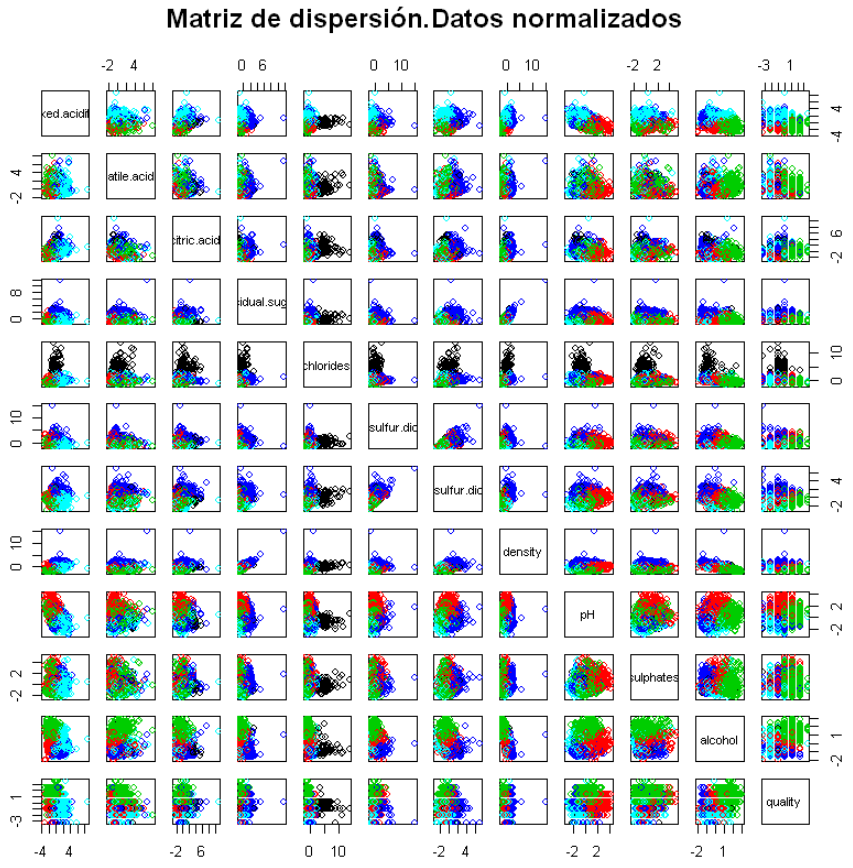
Observamos que en el cluster 3 están los vinos con más calidad y los de menos calidad en el cluster 5.

Vemos tambien que el nivel de alcohol alto y niveles de densidad bajos es característico de los vinos de calidad.

Podemos decucir que si que existen diferencias notables entre los distintos niveles de calidad de los vinos atendiendo a los resultados obtenidos en los diferentes estudios.

Visualizamos un plot general entre los pares de variables del dataframe con los datos ya estandarizados.

In [108]:
```
plot(vinos_normalizados, col =vinos_cluster$cluster,main="Matriz de dispersión.Datos normalizados")
```

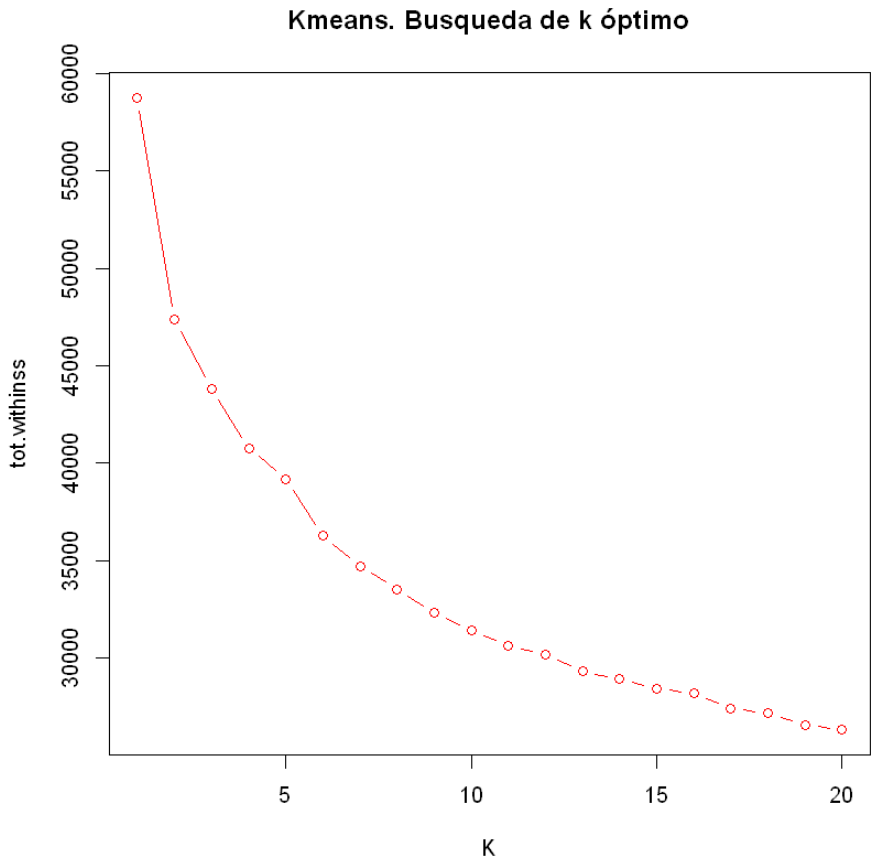


Determinar un número optimo de clusters.

Para hacer esto ejecutamos kmeans para que cree un sólo cluster y el resultado lo almacenamos en un vector. Realizamos un bucle con el número de iteraciones que queramos probar, que serán el número de cluster creados que queramos testear.
Nos apoyamos en las medidas de el atributo "tot.withinss", este valor nos interesa que sea lo menor posible, es decir, la suma de todas las sumas de los cuadrados dentro de los cluster. Por lo que si deja de decrecer esta medida, a medida que aumenta k en el gráfico, significará que ese punto k es un buen candidato para realizar de nuevo el método kmeans con ese valor de k.

In [25]:
```
opt_cluster<- kmeans(vinos_normalizados,centers=1)$tot.withinss

for(i in 2:20) opt_cluster[i]<- kmeans(vinos_normalizados,centers=i)$tot.withinss

plot(1:20,opt_cluster,type="b",xlab="K",ylab="tot.withinss",col="red", main="Kmeans. Busqueda de k óptimo")
```

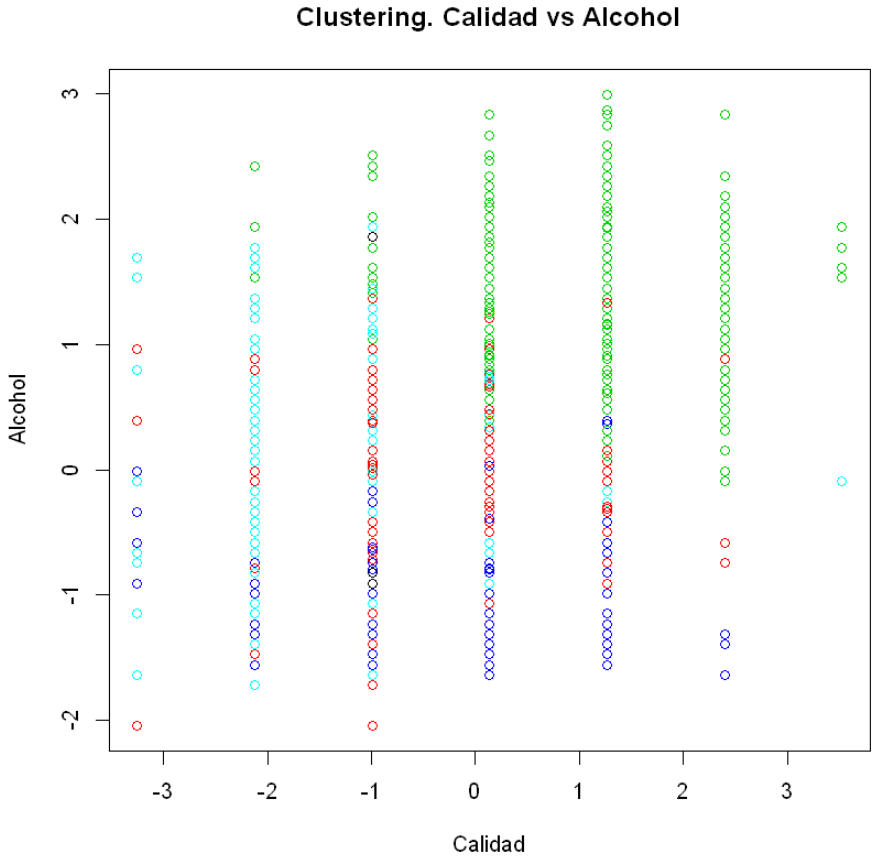
```
Warning message:
"did not converge in 10 iterations"
```



Vemos en este ejemplo que tot.withinss no deja de decrecer.

Podemos inspeccionar los resultados por pares de variables con más detalle que en la matrix de dispersion.

In [110]:
```
plot(vinos_normalizados$quality, vinos_normalizados$alcohol, col=vinos_cluster$cluster, xlab="Calidad", ylab="Alcohol",
main="Clustering. Calidad vs Alcohol")
```




```
Vemos como en la gráfica los diferentes cluster visualmente diferenciados y alineados por los valores normalizados de los niveles de calidad. Vemos
también como existe una relación clara entre las variables alcohol y calidad. Con valores altos de alcohol se observan los mayores niveles de calidad.
```

Podemos ver los datos agrupados por clusters y con sus medias, esto nos dará mas información del conjutno de los datos.

In [104]:
```
medias <- aggregate(whitewine_clean[,2:13],by=list(vinos_cluster$cluster),mean)
medias
```

Group.1	fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
1	6.693458	0.3173364	0.4414019	4.625234	0.16266355	40.16355	144.0748	0.9943926	3.096449	0.4644860	9.566355	5.485981
2	6.376062	0.2499602	0.2904071	4.001549	0.04478319	34.66726	138.4748	0.9935348	3.320832	0.5381150	10.331162	5.855752
3	6.476631	0.2952910	0.3164727	3.702601	0.03428131	30.14947	110.4215	0.9907863	3.209727	0.4743386	12.119283	6.694885
4	6.993039	0.2825770	0.3654485	12.088889	0.04881392	47.41734	175.0408	0.9973735	3.144297	0.4925301	9.450279	5.592369
5	7.610358	0.2801404	0.3452275	3.900242	0.04296225	23.65586	115.2652	0.9932469	3.092798	0.4528170	10.589626	5.458858

```
Con estos datos podemos deducir algunas cosas, por ejemplo, que los vinos blancos de calidad media de 6,69 tienen una media de alcohol de 12,11.
```

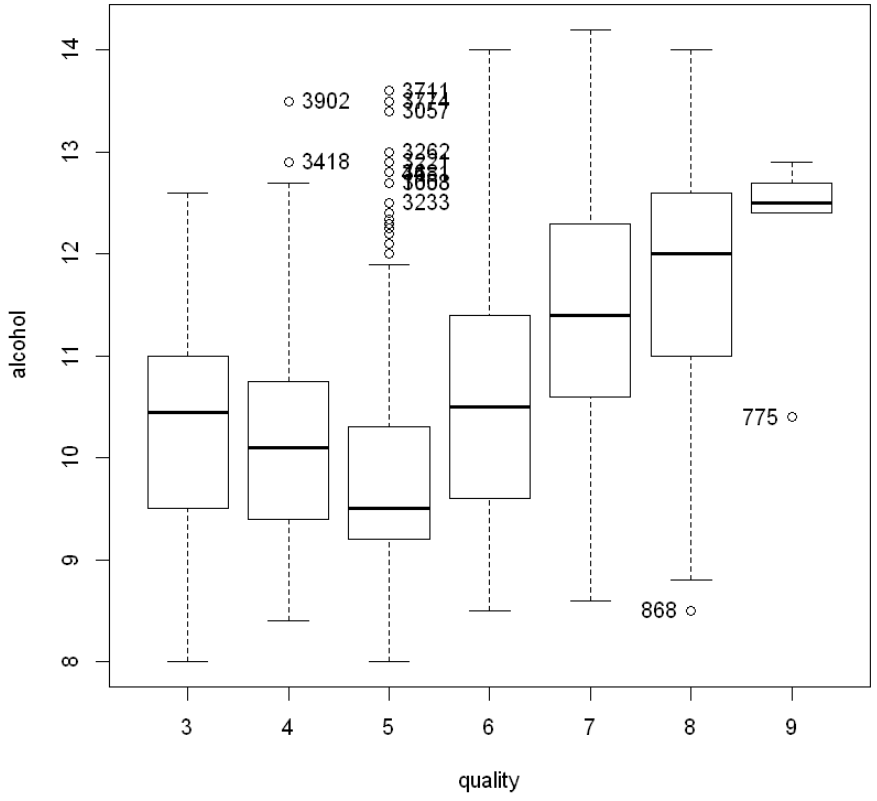
```
**********************************************************************************************************************************
```

```
Representamos gráficamente los niveles medios de alcohol por calidades
```

In [ ]:
```
whitewine_clean$quality = as.factor(whitewine_clean$quality)
```

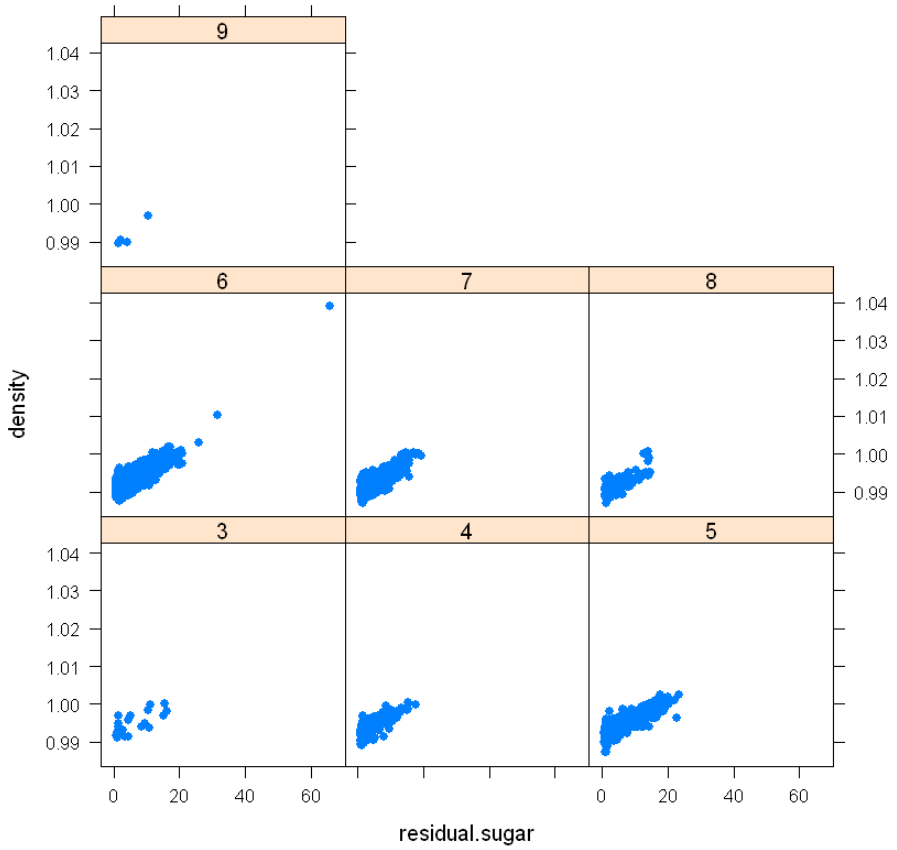
In [122]:
```
Boxplot(alcohol~quality, data=whitewine_clean, id.method="y")
```

'3418' '3902' '3711' '3774' '3057' '3262' '3221' '36' '4481' '1603' '3058' '3233' '868' '775'



Vamos a representar la relación entre los residuos de azucar y la densidad por niveles de calidad.

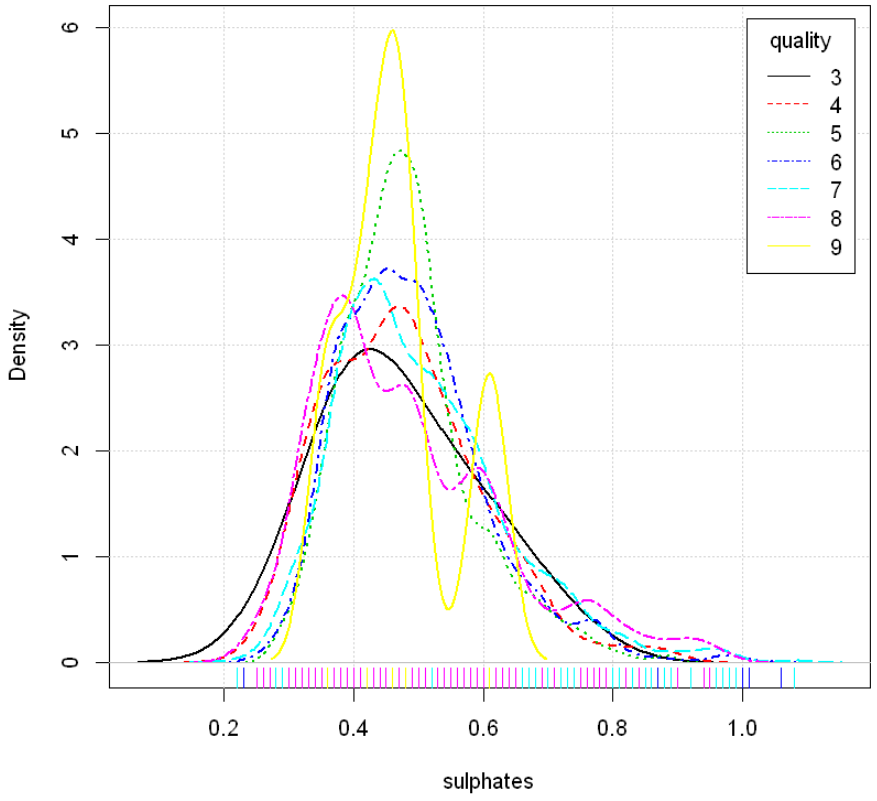
In [121]:
```
library(lattice)
xyplot(density ~ residual.sugar | quality, type="p", pch=16,
  auto.key=list(border=TRUE), par.settings=simpleTheme(pch=16),
  scales=list(x=list(relation='same'), y=list(relation='same')),
  data=whitewine_clean)
```



Vemos como existe una correlación fuerte entre los niveles de densidad y los residuos de azucar en todos los niveles de calidad.

Por último representamos la densidad de sulfatos presentes en los diferentes niveles de calidad.

In [120]:
```
densityPlot(sulphates~quality, data=whitewine_clean, bw="SJ", adjust=1, kernel="gaussian")
```



Comprobamos que también existen diferencias observables entre el contenido de sulfatos en los diferentes niveles de calidad.

In [ ]:
```
**********************************************************************************************************************************************
```

In [123]:
```
write.table(whitewine_clean, "E:/R/tipologia_ciclo_de_vida_datos/wineQualityWhites_2.csv", sep=";", col.names=TRUE, row.names=FALSE, quote=TRUE, na="NA")
```

In [124]:
```
write.table(vinos_normalizados, "E:/R/tipologia_ciclo_de_vida_datos/vinos_normalizados.csv", sep=";", col.names=TRUE, row.names=FALSE, quote=TRUE, na="NA")
```