ProcesamientoEvaluacion.R

moises

Sun Mar 5 21:31:26 2017

```
#PREPROCESAMIENTO

library(foreign)

#Ejercicio1

##APARTADO A

#Pienso que estos deberían de ser los tipos asociados a cada columna

# V1 : numeric (ya que hay valores reales)
# V2 : Factor w/ 2 levels "0","1" ó logical (depende de si se mapean los 2 valores (p.e. "MASCULINO","F
# V3 : numeric (ya que hay valores reales)
# V4 : Factor w/ 2 levels "0","1" ó logical (depende de si se mapean los 2 valores (p.e. "MASCULINO","F
# V5 : numeric (ya que hay valores reales)
# V6 : numeric (ya que hay valores reales)
# V7 : numeric (ya que hay valores reales)
# V8 : numeric (ya que hay valores reales)
# V9 : numeric (ya que hay valores reales)
# V10: numeric (ya que hay valores reales)
# V11: Factor w/ 1 levels "name" (parece que se sustituye el nombre real por 'name' para mantener el an
# V12: Factor w/ 2 levels "1","2" ó logical (depende de si se mapean los 2 valores (p.e. "MASCULINO","F
# V13: Factor w/ 2 levels "0","1" ó logical (depende de si se mapean los 2 valores (p.e. "MASCULINO","F

##APARTADO B

#Sí, y están representados por signo final de interrogación '?' y 'name'

##APARTADO C

#Faltan los nombres de las columnas ya que el fichero no lleva los encabezados,
#y si sólo contemplamos el fichero 'echocardiogram.data'
#nos faltan datos para saber que son esos 'name', luego en otro fichero se indica que se
#sustituye el nombre real del paciente por 'name' para mantener su anonimato. Es decir,
#tenemos otro fichero aparte que nos da información sobre como están estructurados los datos.
#Dicho fichero se denomina 'echocardiogram.names.txt' y podemos encontrar la estructura
#de los datos en la sección 7 llamada 'Attribute Information'


#Ejercicio 2

##APARTADO A

echocardiogram.data <- read.csv("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Práct
str(echocardiogram.data)
```

```
## 'data.frame':    132 obs. of  13 variables:
```

```
##  $ V1 : Factor w/ 57 levels "?","0.25","03",..: 9 16 14 53 16 26 12 49 16 24 ...
##  $ V2 : Factor w/ 3 levels "?","0","1": 2 2 2 2 3 2 2 2 2 2 ...
##  $ V3 : Factor w/ 39 levels "?","35","46",..: 28 29 11 16 13 25 18 16 3 10 ...
##  $ V4 : Factor w/ 3 levels "?","0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ V5 : Factor w/ 73 levels "?","0.010","0.030",..: 49 64 49 46 25 49 41 58 59 20 ...
##  $ V6 : Factor w/ 92 levels "?","0","10","10.2",..: 82 68 56 12 45 61 55 78 2 18 ...
##  $ V7 : Factor w/ 106 levels "?","2.32","3.100",..: 53 24 5 54 92 37 85 76 70 46 ...
##  $ V8 : Factor w/ 47 levels "?","10","10.5",..: 15 15 15 20 26 7 35 15 20 18 ...
##  $ V9 : Factor w/ 66 levels "?","1","1.04",..: 2 47 2 36 59 2 51 2 10 15 ...
##  $ V10: Factor w/ 31 levels "?","0.140","0.28",..: 29 13 29 24 12 26 26 29 30 28 ...
##  $ V11: Factor w/ 1 level "name": 1 1 1 1 1 1 1 1 1 1 ...
##  $ V12: Factor w/ 3 levels "?","1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ V13: Factor w/ 3 levels "?","0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

```
# V1 : Factor w/ 57 levels "?","0.25","03",..: 9 16 14 53 16 26 12 49 16 24 ...
# V2 : Factor w/ 3 levels "?","0","1": 2 2 2 2 3 2 2 2 2 2 ...
# V3 : Factor w/ 39 levels "?","35","46",..: 28 29 11 16 13 25 18 16 3 10 ...
# V4 : Factor w/ 3 levels "?","0","1": 2 2 2 2 2 2 2 2 2 2 ...
# V5 : Factor w/ 73 levels "?","0.010","0.030",..: 49 64 49 46 25 49 41 58 59 20 ...
# V6 : Factor w/ 92 levels "?","0","10","10.2",..: 82 68 56 12 45 61 55 78 2 18 ...
# V7 : Factor w/ 106 levels "?","2.32","3.100",..: 53 24 5 54 92 37 85 76 70 46 ...
# V8 : Factor w/ 47 levels "?","10","10.5",..: 15 15 15 20 26 7 35 15 20 18 ...
# V9 : Factor w/ 66 levels "?","1","1.04",..: 2 47 2 36 59 2 51 2 10 15 ...
# V10: Factor w/ 31 levels "?","0.140","0.28",..: 29 13 29 24 12 26 26 29 30 28 ...
# V11: Factor w/ 1 level "name": 1 1 1 1 1 1 1 1 1 1 ...
# V12: Factor w/ 3 levels "?","1","2": 2 2 2 2 2 2 2 2 2 2 ...
# V13: Factor w/ 3 levels "?","0","1": 2 2 2 2 2 2 2 2 2 2 ...

#La principal anomalía que encontramos es que los tipos de las columnas no son los esperados
#porque está tomando los valores ausentes '?' como parte del conjunto de datos cuando en
#realidad no debería de ser así ya que es un NA (Not Available) y no debería ser tenido en
#cuenta. Otra anomalía, que seguramente venga provocada por la anterior, es que los números
#son cogidos como palabras y no como numeric (reales) como se esperaba. Otra anomalía es que
#para los que sí que se esperaba una columna de tipo Factor w/ de 2 niveles, aparecen 3 niveles
#y esto es debido a que coge '?' como parte del conjunto de valores cuando debería omitirlo.

##APARTADO B

complete.cases(echocardiogram.data)
```

```
##   [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [15] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [29] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [43] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [71] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [85] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [99] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [113] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [127] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
#No es el resultado esperado porque nos dice que todas las filas están completas cuando no
#es así porque tienen valores ausentes (NA) pero no se lo hemos indicado al importar los datos.
```

```
##APARTADO C
echocardiogram.data <- read.csv("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Práct
str(echocardiogram.data)
```

```
## 'data.frame':    132 obs. of  13 variables:
##  $ V1 : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ V2 : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ V3 : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ V4 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ V5 : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ V6 : num  9 6 4 12.1 22 ...
##  $ V7 : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ V8 : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ V9 : num  1 1.7 1 1.45 2.25 ...
##  $ V10: num  1 0.588 1 0.788 0.571 ...
##  $ V11: Factor w/ 1 level "name": 1 1 1 1 1 1 1 1 1 1 ...
##  $ V12: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ V13: int  0 0 0 0 0 0 0 0 0 0 ...
```

```
# V1 : num  11 19 16 57 19 26 13 50 19 25 ...
# V2 : int  0 0 0 0 1 0 0 0 0 0 ...
# V3 : num  71 72 55 60 57 68 62 60 46 54 ...
# V4 : int  0 0 0 0 0 0 0 0 0 0 ...
# V5 : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
# V6 : num  9 6 4 12.1 22 ...
# V7 : num  4.6 4.1 3.42 4.6 5.75 ...
# V8 : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
# V9 : num  1 1.7 1 1.45 2.25 ...
# V10: num  1 0.588 1 0.788 0.571 ...
# V11: Factor w/ 1 level "name": 1 1 1 1 1 1 1 1 1 1 ...
# V12: int  1 1 1 1 1 1 1 1 1 1 ...
# V13: int  0 0 0 0 0 0 0 0 0 0 ...

#Sí. Sigue habiendo una anomalía que puede apreciarse ya que las columnas que esperábamos que
#fueran logical (booleanas) -o conjunto de palabras de dos niveles en su caso- en realidad las
#está cogiendo como si fuesen de tipo entero. Por lo que tenemos que tratar dichas columnas.

##APARTADO D

complete.cases(echocardiogram.data)
```

```
##   [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [12]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
##  [34] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE
##  [45] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
##  [56]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE  TRUE
##  [67] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE
##  [78] FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE
##  [89]  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE
## [100]  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
## [111] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [122] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```r
#Al ejecutar la instrucción anterior todas las filas que muestren un FALSE indican que contienen
#valores desconocidos. Si queremos ser más específicos podemos guardar el resultado en un vector
#y mostrar el índice cuyo contenido sea FALSE.

imprimirFilasIncompletas <- function(datos){
  vectorFilas <- complete.cases(datos)
  for(i in 1:length(vectorFilas)) {
    if(vectorFilas[i] == FALSE) {
      print(i)
    }
  }
}

imprimirFilasIncompletas(echocardiogram.data) #Estas serían las filas con valores desconocidos.
```

```
## [1] 25
## [1] 26
## [1] 27
## [1] 28
## [1] 29
## [1] 31
## [1] 32
## [1] 33
## [1] 34
## [1] 35
## [1] 37
## [1] 38
## [1] 39
## [1] 40
## [1] 44
## [1] 45
## [1] 46
## [1] 47
## [1] 49
## [1] 50
## [1] 51
## [1] 52
## [1] 53
## [1] 60
## [1] 62
## [1] 64
## [1] 65
## [1] 67
## [1] 75
## [1] 77
## [1] 78
## [1] 80
## [1] 81
## [1] 83
## [1] 85
## [1] 86
## [1] 87
## [1] 88
```

```
## [1] 90
## [1] 91
## [1] 92
## [1] 94
## [1] 95
## [1] 96
## [1] 98
## [1] 101
## [1] 102
## [1] 104
## [1] 108
## [1] 111
## [1] 112
## [1] 113
## [1] 114
## [1] 115
## [1] 116
## [1] 117
## [1] 118
## [1] 119
## [1] 120
## [1] 121
## [1] 122
## [1] 123
## [1] 124
## [1] 125
## [1] 126
## [1] 127
## [1] 128
## [1] 129
## [1] 130
## [1] 131
## [1] 132
```

```
#Ejercicio 3

##APARTADO A

echocardiogram.data$V2 <- as.logical(echocardiogram.data$V2)
echocardiogram.data$V4 <- as.logical(echocardiogram.data$V4)
echocardiogram.data$V13 <- as.logical(echocardiogram.data$V13)
str(echocardiogram.data)
```

```
## 'data.frame':    132 obs. of  13 variables:
##  $ V1 : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ V2 : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ V3 : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ V4 : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ V5 : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ V6 : num  9 6 4 12.1 22 ...
##  $ V7 : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ V8 : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ V9 : num  1 1.7 1 1.45 2.25 ...
##  $ V10: num  1 0.588 1 0.788 0.571 ...
```

```
##  $ V11: Factor w/ 1 level "name": 1 1 1 1 1 1 1 1 1 1 ...
##  $ V12: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ V13: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#Se cambian estas columnas a binario (logical TRUE, FALSE) según la documentación del fichero.
#Las demás columnas no se tocan o bien porque son valores reales que ya está como numeric, o
#bien porque son columnas que la documentación nos dice expresamente que ignoremos

##APARTADO B

atributos <- c("Survival", "StillAlive", "AgeAttack", "PericardEffu", "FracShort",
"EPSS", "LVDD", "WMS", "WMI", "Mult", "Name", "Group", "AliveAt1")
colnames(echocardiogram.data) <- atributos
str(echocardiogram.data)
```

```
## 'data.frame':    132 obs. of  13 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ Mult        : num  1 0.588 1 0.788 0.571 ...
##  $ Name        : Factor w/ 1 level "name": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Group       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
##APARTADO C

#Según la documentación la columna AliveAt1 viene como resultado de las dos primeras
#columnas en las que el paciente debe de cumplir dos condiciones:
# 1) Estar vivo
# 2) Haber sufrido el ataque hace 12 meses o más (tiempo de supervivencia)
#Los valores disponibles aportados están mal ya que están del revés:
#20,1,59,0,0.030,21.300,6.290,17,1.310,0.928,name,2,0 (se espera 1 no 0)
#0.25,1,63,1,?,?,?,23,2.300,0.714,name,2,1 (se espera 0 no 1)
#Habría que recalcular todos los valores y no sólo los NA ya que puede que la
#persona que los introdujo los metiera mal. Debido a esto se va a hacer una
#bifurcación con dos datasets uno dejándolo como está y otro corrigiendo valores
#de las primeras dos columnas para que sean coherentes con el resultado
#de la columna de clasificación (la última)

#Copiamos entonces el dataset original en otro dataset
echocardiogram.datos_coherentes <- echocardiogram.data

calcularAliveAt1 <- function(meses, vivo) {
  vivo & (meses >= 12)
}

echocardiogram.data$AliveAt1
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [12] FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE    NA    NA    NA    NA    NA FALSE    NA    NA    NA
##  [34]    NA FALSE FALSE FALSE  TRUE FALSE    NA FALSE FALSE FALSE FALSE
##  [45]    NA    NA FALSE FALSE  TRUE    NA  TRUE  TRUE    NA FALSE  TRUE
##  [56] FALSE FALSE FALSE FALSE    NA FALSE    NA FALSE    NA    NA FALSE
##  [67]    NA FALSE FALSE FALSE  TRUE FALSE FALSE FALSE    NA FALSE    NA
##  [78]  TRUE FALSE    NA    NA FALSE    NA  TRUE  TRUE    NA    NA    NA
##  [89] FALSE    NA    NA    NA FALSE    NA  TRUE    NA FALSE    NA  TRUE
## [100]  TRUE    NA FALSE FALSE    NA  TRUE FALSE FALSE    NA  TRUE  TRUE
## [111]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [122]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
```

```
for(i in 1:dim(echocardiogram.data)[1]){
  if(is.na(echocardiogram.data$AliveAt1[i])){
    echocardiogram.data$AliveAt1[i] <- calcularAliveAt1(echocardiogram.data$Survival[i],echocardiogram.
  }
}

echocardiogram.data$AliveAt1
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [12] FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [45] FALSE FALSE FALSE FALSE  TRUE    NA  TRUE  TRUE FALSE FALSE  TRUE
##  [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [67] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [78]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
##  [89] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
## [100]  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE
## [111] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [122] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
#¡Ojo! En la fila 50 tenemos un NA como resultado porque en las dos primeras columnas tenemos
#también dos NA, y como ya sabemos NA AND NA da como resultado NA, por lo que más tarde,
#después de hacer las imputaciones hay que calcular este NA de la columna de clasificación
#a partir de los nuevos datos imputados.


##APARTADO D

#Sería conveniente eliminar las columnas Mult y Group ya que el propio experto recomienda
#que las ignoremos y además también podríamos quitar la columna Name ya que al cambiar el
#experto los nombres reales por 'name' para mantener el anonimato de los pacientes, el valor
```

name' por sí solo no nos aporta ninguna información y se repite en las 132 instancias.

```
#Por tanto procedemos a su eliminación del dataframe
echocardiogram.data <- echocardiogram.data[,!colnames(echocardiogram.data)=="Mult"]
echocardiogram.data <- echocardiogram.data[,!colnames(echocardiogram.data)=="Group"]
echocardiogram.data <- echocardiogram.data[,!colnames(echocardiogram.data)=="Name"]
str(echocardiogram.data)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#Ejercicio 4

##APARTADO A

library(VIM)
```

```
## Loading required package: colorspace
```

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

```
## Loading required package: data.table
```

```
## VIM is ready to use.
##  Since version 4.0.0 the GUI is in its own package VIMGUI.
##
##           Please use the package to use the new (and old) GUI.
```

```
## Suggestions and bug-reports can be submitted at: https://github.com/alexkowa/VIM/issues
```

```
##
## Attaching package: 'VIM'
```

```
## The following object is masked from 'package:datasets':
##
##     sleep
```

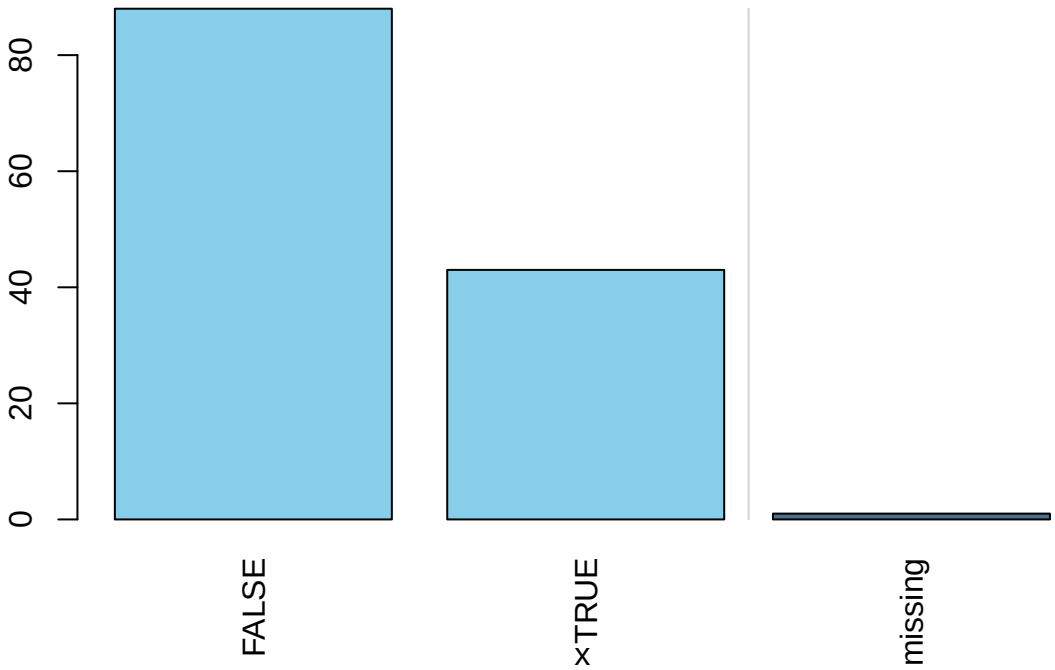
```
proporcionNAPorColumna <- function(columna) {
  x <- countNA(columna)
  y <- dim(columna)[1]
  x/y
}

calcularVectorProporcionNA <- function(datos) {
  i <- 1
  vectorCalculado <- c()
  while(i <= dim(datos)[2]){
    vectorCalculado = append(vectorCalculado, proporcionNAPorColumna(datos[i]))
    i <- i + 1
  }
```

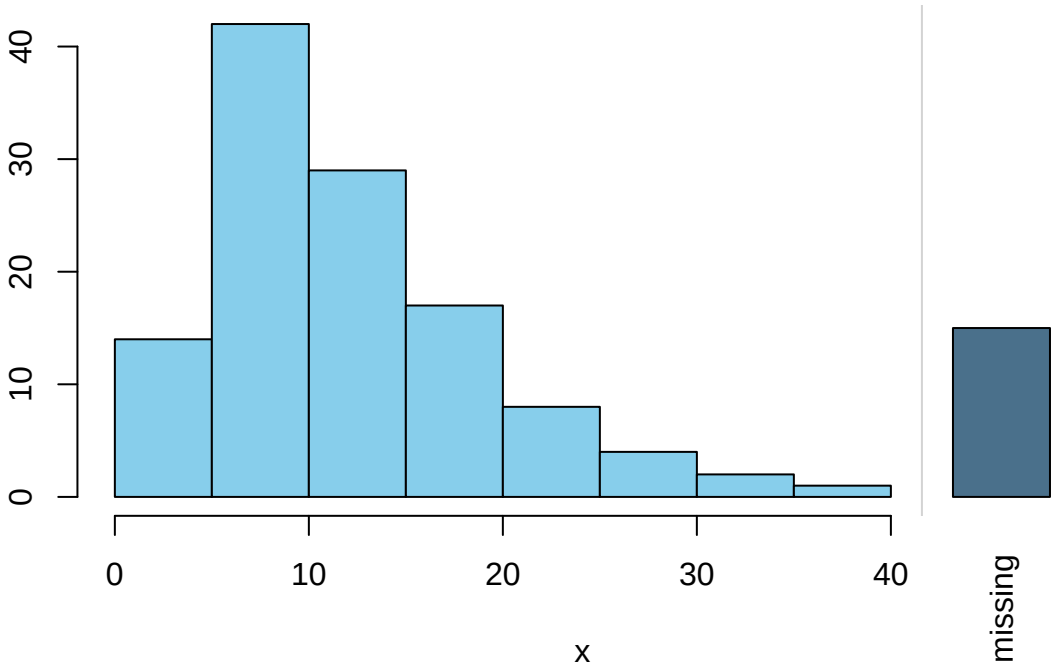
```
  vectorCalculado
}
vectorProporcionNA <- calcularVectorProporcionNA(echocardiogram.data)
vectorProporcionNA
```

```
##  [1] 0.015151515 0.007575758 0.037878788 0.007575758 0.060606061
##  [6] 0.113636364 0.083333333 0.030303030 0.007575758 0.007575758
```

```
barMiss(echocardiogram.data$StillAlive)
```




```
barMiss(echocardiogram.data$EPSS) #Con esta función de VIM podemos ver gráficamente como
```




```
#la columna EPSS tiene un gran porcentaje de valores perdidos que como hemos calculado
#antes a mano es del 11.36% de los elementos de dicha columna. Si comparamos esta gráfica de
#EPSS con la de StillAlive podemos ver gráficamente la diferencia de que ésta no tiene
#apenas nada en la barra de missing, ya que como previamente habíamos calculado la proporción
#para la columna StillAlive es del 0.76%

##APARTADO B

library(Hmisc)
```

```
## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units
```

```
#Primero lo hacemos con Median (la mediana)
echocardiogram.data.imp_median <- echocardiogram.data
str(echocardiogram.data.imp_median)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
echocardiogram.data.imp_median$Survival <- impute(echocardiogram.data.imp_median$Survival,fun=median)
echocardiogram.data.imp_median$StillAlive <- impute(echocardiogram.data.imp_median$StillAlive,fun=median
echocardiogram.data.imp_median$AgeAttack <- impute(echocardiogram.data.imp_median$AgeAttack,fun=median)
echocardiogram.data.imp_median$PericardEffu <- impute(echocardiogram.data.imp_median$PericardEffu,fun=m
echocardiogram.data.imp_median$FracShort <- impute(echocardiogram.data.imp_median$FracShort,fun=median)
echocardiogram.data.imp_median$EPSS <- impute(echocardiogram.data.imp_median$EPSS,fun=median)
echocardiogram.data.imp_median$LVDD <- impute(echocardiogram.data.imp_median$LVDD,fun=median)
echocardiogram.data.imp_median$WMS <- impute(echocardiogram.data.imp_median$WMS,fun=median)
echocardiogram.data.imp_median$WMI <- impute(echocardiogram.data.imp_median$WMI,fun=median)
#Imputamos todos los datos de todas las columnas excepto de la última cuyos NA los podemos
#calcular a partir de los datos imputados en las dos primeras columnas
echocardiogram.data.imp_median$AliveAt1[50] <- calcularAliveAt1(echocardiogram.data.imp_median$Survival
echocardiogram.data.imp_median$AliveAt1
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [12] FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [45] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE  TRUE
##  [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [67] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [78]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
##  [89] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
## [100]  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE
## [111] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [122] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
#Efectivamente ahora la fila 50 de la columna de clasificación ya no presenta un NA
complete.cases(echocardiogram.data.imp_median)
```

```
##   [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [15] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [29] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [43] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [71] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [85] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [99] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [113] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [127] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
#Con el complete.cases nos aseguramos que todas las columnas ya no contienen NA
str(echocardiogram.data.imp_median)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    :Class 'impute'  atomic [1:132] 11 19 16 57 19 26 13 50 19 25 ...
##   .. ..- attr(*, "imputed")= int [1:2] 50 95
##  $ StillAlive  :Class 'impute'  atomic [1:132] FALSE FALSE FALSE FALSE TRUE FALSE ...
##   .. ..- attr(*, "imputed")= int 50
##  $ AgeAttack   :Class 'impute'  atomic [1:132] 71 72 55 60 57 68 62 60 46 54 ...
##   .. ..- attr(*, "imputed")= int [1:5] 33 78 117 120 126
##  $ PericardEffu:Class 'impute'  atomic [1:132] FALSE FALSE FALSE FALSE FALSE FALSE ...
##   .. ..- attr(*, "imputed")= int 50
##  $ FracShort   :Class 'impute'  atomic [1:132] 0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ..
##   .. ..- attr(*, "imputed")= int [1:8] 28 29 33 35 37 49 50 85
##  $ EPSS        :Class 'impute'  atomic [1:132] 9 6 4 12.1 22 ...
##   .. ..- attr(*, "imputed")= int [1:15] 29 31 37 39 44 46 47 49 50 52 ...
##  $ LVDD        :Class 'impute'  atomic [1:132] 4.6 4.1 3.42 4.6 5.75 ...
##   .. ..- attr(*, "imputed")= int [1:11] 28 29 33 35 38 46 47 49 50 52 ...
##  $ WMS         :Class 'impute'  atomic [1:132] 14 14 14 16 18 12 22.5 14 16 15.5 ...
##   .. ..- attr(*, "imputed")= int [1:4] 29 40 50 51
##  $ WMI         :Class 'impute'  atomic [1:132] 1 1.7 1 1.45 2.25 ...
##   .. ..- attr(*, "imputed")= int 51
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#Todos los tipos de las columnas han cambiado a Class 'impute', excepto la columna de clasificación.
#Ojo con esto porque hay que cambiar el tipo más tarde a su tipo original.
echocardiogram.data.imp_median$Survival
```

```
##   [1] 11.00  19.00  16.00  57.00  19.00  26.00  13.00  50.00  19.00  25.00
##  [11] 10.00  52.00  52.00  44.00   0.50  24.00   0.50   0.50  22.00   1.00
##  [21]  0.75   0.75   0.50   5.00  48.00  29.00  29.00  29.00   0.25  36.00
##  [31]  1.00   1.00   3.00  27.00  35.00  26.00  16.00   1.00  19.00  31.00
##  [41] 32.00  16.00  40.00  46.00   2.00  37.00  19.50  20.00   0.25  23.50*
##  [51]  2.00   7.00  10.00  12.00   1.00  10.00  45.00  22.00  53.00  38.00
##  [61] 26.00   9.00  26.00   0.50  12.00  49.00   0.75  49.00  47.00  41.00
##  [71]  0.25  33.00  29.00  41.00  26.00  15.00   0.25   0.03  12.00  32.00
##  [81] 32.00  27.00  23.00   0.75   0.75  34.00   1.00  21.00  55.00  15.00
##  [91]  0.50  35.00  53.00  33.00  23.50* 33.00  40.00  33.00   5.00   4.00
## [101] 31.00  33.00  22.00  25.00   1.25  24.00  25.00  24.00   0.75   3.00
## [111] 27.00  13.00  36.00  25.00  27.00  34.00  37.00  34.00  28.00  28.00
## [121] 17.00  38.00  31.00  12.00  36.00  17.00  21.00   7.50  41.00  36.00
## [131] 22.00  20.00
```

```
echocardiogram.data.imp_median$StillAlive
```

```
##   [1] FALSE  FALSE  FALSE  FALSE   TRUE  FALSE  FALSE  FALSE  FALSE  FALSE
##  [11]  TRUE  FALSE  FALSE  FALSE   TRUE  FALSE   TRUE   TRUE   TRUE   TRUE
##  [21]  TRUE   TRUE   TRUE   TRUE  FALSE  FALSE  FALSE  FALSE   TRUE  FALSE
##  [31]  TRUE   TRUE   TRUE  FALSE  FALSE  FALSE  FALSE   TRUE  FALSE  FALSE
##  [41] FALSE  FALSE  FALSE  FALSE   TRUE  FALSE   TRUE   TRUE   TRUE  FALSE*
##  [51]  TRUE   TRUE  FALSE  FALSE   TRUE  FALSE  FALSE  FALSE  FALSE  FALSE
##  [61] FALSE  FALSE  FALSE   TRUE  FALSE  FALSE   TRUE  FALSE  FALSE  FALSE
```

```
##  [71]  TRUE  FALSE  FALSE  FALSE  FALSE  FALSE   TRUE   TRUE  FALSE  FALSE
##  [81] FALSE  FALSE  FALSE   TRUE   TRUE  FALSE   TRUE   TRUE  FALSE   TRUE
##  [91]  TRUE  FALSE  FALSE  FALSE   TRUE  FALSE   TRUE  FALSE   TRUE   TRUE
## [101] FALSE  FALSE  FALSE  FALSE   TRUE  FALSE  FALSE  FALSE   TRUE   TRUE
## [111] FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE   TRUE  FALSE
## [121] FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE   TRUE  FALSE  FALSE
## [131] FALSE  FALSE
```

```
#En las dos anteriores instruccione podemos observar los datos imputados en la fila 50 mediante
#un asterisco ('*') después del valor imputado.

#Cambiamos los tipos a los originales para evitar problemas a la hora de realizar cálculos
#posteriores.
echocardiogram.data.imp_median$Survival <- as.numeric(echocardiogram.data.imp_median$Survival)
echocardiogram.data.imp_median$StillAlive <- as.logical(echocardiogram.data.imp_median$StillAlive)
echocardiogram.data.imp_median$AgeAttack <- as.numeric(echocardiogram.data.imp_median$AgeAttack)
echocardiogram.data.imp_median$PericardEffu <- as.logical(echocardiogram.data.imp_median$PericardEffu)
echocardiogram.data.imp_median$FracShort <- as.numeric(echocardiogram.data.imp_median$FracShort)
echocardiogram.data.imp_median$EPSS <- as.numeric(echocardiogram.data.imp_median$EPSS)
echocardiogram.data.imp_median$LVDD <- as.numeric(echocardiogram.data.imp_median$LVDD)
echocardiogram.data.imp_median$WMS <- as.numeric(echocardiogram.data.imp_median$WMS)
echocardiogram.data.imp_median$WMI <- as.numeric(echocardiogram.data.imp_median$WMI)
str(echocardiogram.data.imp_median)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#Repetimos todo el proceso anterior pero con Mean (la media)
echocardiogram.data.imp_mean <- echocardiogram.data
str(echocardiogram.data.imp_mean)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
echocardiogram.data.imp_mean$Survival <- impute(echocardiogram.data.imp_mean$Survival,fun=mean)
echocardiogram.data.imp_mean$StillAlive <- impute(echocardiogram.data.imp_mean$StillAlive,fun=mean)
echocardiogram.data.imp_mean$AgeAttack <- impute(echocardiogram.data.imp_mean$AgeAttack,fun=mean)
echocardiogram.data.imp_mean$PericardEffu <- impute(echocardiogram.data.imp_mean$PericardEffu,fun=mean)
echocardiogram.data.imp_mean$FracShort <- impute(echocardiogram.data.imp_mean$FracShort,fun=mean)
echocardiogram.data.imp_mean$EPSS <- impute(echocardiogram.data.imp_mean$EPSS,fun=mean)
echocardiogram.data.imp_mean$LVDD <- impute(echocardiogram.data.imp_mean$LVDD,fun=mean)
echocardiogram.data.imp_mean$WMS <- impute(echocardiogram.data.imp_mean$WMS,fun=mean)
echocardiogram.data.imp_mean$WMI <- impute(echocardiogram.data.imp_mean$WMI,fun=mean)
#Imputamos todos los datos de todas las columnas excepto de la última cuyos NA los podemos
#calcular a partir de los datos imputados en las dos primeras columnas
echocardiogram.data.imp_mean$AliveAt1[50] <- calcularAliveAt1(echocardiogram.data.imp_mean$Survival[50]
echocardiogram.data.imp_mean$AliveAt1
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [12] FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [45] FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
##  [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [67] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [78]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
##  [89] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
## [100]  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE
## [111] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [122] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
#Efectivamente ahora la fila 50 de la columna de clasificación ya no presenta un NA
complete.cases(echocardiogram.data.imp_mean)
```

```
##   [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [15] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [29] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [43] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [71] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [85] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [99] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [113] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [127] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
#Con el complete.cases nos aseguramos que todas las columnas ya no contienen NA
str(echocardiogram.data.imp_mean)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    :Class 'impute'  atomic [1:132] 11 19 16 57 19 26 13 50 19 25 ...
##   .. ..- attr(*, "imputed")= int [1:2] 50 95
##  $ StillAlive  :Class 'impute'  atomic [1:132] 0 0 0 0 1 0 0 0 0 0 ...
##   .. ..- attr(*, "imputed")= int 50
##  $ AgeAttack   :Class 'impute'  atomic [1:132] 71 72 55 60 57 68 62 60 46 54 ...
##   .. ..- attr(*, "imputed")= int [1:5] 33 78 117 120 126
##  $ PericardEffu:Class 'impute'  atomic [1:132] 0 0 0 0 0 0 0 0 0 0 ...
```

```
##   .. ..- attr(*, "imputed")= int 50
##  $ FracShort   :Class 'impute'  atomic [1:132] 0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ..
##   .. ..- attr(*, "imputed")= int [1:8] 28 29 33 35 37 49 50 85
##  $ EPSS        :Class 'impute'  atomic [1:132] 9 6 4 12.1 22 ...
##   .. ..- attr(*, "imputed")= int [1:15] 29 31 37 39 44 46 47 49 50 52 ...
##  $ LVDD        :Class 'impute'  atomic [1:132] 4.6 4.1 3.42 4.6 5.75 ...
##   .. ..- attr(*, "imputed")= int [1:11] 28 29 33 35 38 46 47 49 50 52 ...
##  $ WMS         :Class 'impute'  atomic [1:132] 14 14 14 16 18 12 22.5 14 16 15.5 ...
##   .. ..- attr(*, "imputed")= int [1:4] 29 40 50 51
##  $ WMI         :Class 'impute'  atomic [1:132] 1 1.7 1 1.45 2.25 ...
##   .. ..- attr(*, "imputed")= int 51
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#Todos los tipos de las columnas han cambiado a Class 'impute', excepto la columna de clasificación.
#Ojo con esto porque hay que cambiar el tipo más tarde a su tipo original.
echocardiogram.data.imp_mean$Survival
```

```
##   [1] 11.00000  19.00000  16.00000  57.00000  19.00000  26.00000  13.00000
##   [8] 50.00000  19.00000  25.00000  10.00000  52.00000  52.00000  44.00000
##  [15]  0.50000  24.00000   0.50000   0.50000  22.00000   1.00000   0.75000
##  [22]  0.75000   0.50000   5.00000  48.00000  29.00000  29.00000  29.00000
##  [29]  0.25000  36.00000   1.00000   1.00000   3.00000  27.00000  35.00000
##  [36] 26.00000  16.00000   1.00000  19.00000  31.00000  32.00000  16.00000
##  [43] 40.00000  46.00000   2.00000  37.00000  19.50000  20.00000   0.25000
##  [50] 22.18292*  2.00000   7.00000  10.00000  12.00000   1.00000  10.00000
##  [57] 45.00000  22.00000  53.00000  38.00000  26.00000   9.00000  26.00000
##  [64]  0.50000  12.00000  49.00000   0.75000  49.00000  47.00000  41.00000
##  [71]  0.25000  33.00000  29.00000  41.00000  26.00000  15.00000   0.25000
##  [78]  0.03000  12.00000  32.00000  32.00000  27.00000  23.00000   0.75000
##  [85]  0.75000  34.00000   1.00000  21.00000  55.00000  15.00000   0.50000
##  [92] 35.00000  53.00000  33.00000  22.18292* 33.00000  40.00000  33.00000
##  [99]  5.00000   4.00000  31.00000  33.00000  22.00000  25.00000   1.25000
## [106] 24.00000  25.00000  24.00000   0.75000   3.00000  27.00000  13.00000
## [113] 36.00000  25.00000  27.00000  34.00000  37.00000  34.00000  28.00000
## [120] 28.00000  17.00000  38.00000  31.00000  12.00000  36.00000  17.00000
## [127] 21.00000   7.50000  41.00000  36.00000  22.00000  20.00000
```

```
echocardiogram.data.imp_mean$StillAlive
```

```
##   [1] 0.0000000  0.0000000  0.0000000  0.0000000  1.0000000  0.0000000
##   [7] 0.0000000  0.0000000  0.0000000  0.0000000  1.0000000  0.0000000
##  [13] 0.0000000  0.0000000  1.0000000  0.0000000  1.0000000  1.0000000
##  [19] 1.0000000  1.0000000  1.0000000  1.0000000  1.0000000  1.0000000
##  [25] 0.0000000  0.0000000  0.0000000  0.0000000  1.0000000  0.0000000
##  [31] 1.0000000  1.0000000  1.0000000  0.0000000  0.0000000  0.0000000
##  [37] 0.0000000  1.0000000  0.0000000  0.0000000  0.0000000  0.0000000
##  [43] 0.0000000  0.0000000  1.0000000  0.0000000  1.0000000  1.0000000
##  [49] 1.0000000  0.3282443* 1.0000000  1.0000000  0.0000000  0.0000000
##  [55] 1.0000000  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000
##  [61] 0.0000000  0.0000000  0.0000000  1.0000000  0.0000000  0.0000000
##  [67] 1.0000000  0.0000000  0.0000000  0.0000000  1.0000000  0.0000000
##  [73] 0.0000000  0.0000000  0.0000000  0.0000000  1.0000000  1.0000000
##  [79] 0.0000000  0.0000000  0.0000000  0.0000000  0.0000000  1.0000000
```

```
##  [85] 1.0000000  0.0000000  1.0000000  1.0000000  0.0000000  1.0000000
##  [91] 1.0000000  0.0000000  0.0000000  0.0000000  1.0000000  0.0000000
##  [97] 1.0000000  0.0000000  1.0000000  1.0000000  0.0000000  0.0000000
## [103] 0.0000000  0.0000000  1.0000000  0.0000000  0.0000000  0.0000000
## [109] 1.0000000  1.0000000  0.0000000  0.0000000  0.0000000  0.0000000
## [115] 0.0000000  0.0000000  0.0000000  0.0000000  1.0000000  0.0000000
## [121] 0.0000000  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000
## [127] 0.0000000  1.0000000  0.0000000  0.0000000  0.0000000  0.0000000
```

```
#En las dos anteriores instruccione podemos observar los datos imputados en la fila 50 mediante
#un asterisco ('*') después del valor imputado.

#Cambiamos los tipos a los originales para evitar problemas a la hora de realizar cálculos
#posteriores.
echocardiogram.data.imp_mean$Survival <- as.numeric(echocardiogram.data.imp_mean$Survival)
echocardiogram.data.imp_mean$StillAlive <- as.logical(echocardiogram.data.imp_mean$StillAlive)
echocardiogram.data.imp_mean$AgeAttack <- as.numeric(echocardiogram.data.imp_mean$AgeAttack)
echocardiogram.data.imp_mean$PericardEffu <- as.logical(echocardiogram.data.imp_mean$PericardEffu)
echocardiogram.data.imp_mean$FracShort <- as.numeric(echocardiogram.data.imp_mean$FracShort)
echocardiogram.data.imp_mean$EPSS <- as.numeric(echocardiogram.data.imp_mean$EPSS)
echocardiogram.data.imp_mean$LVDD <- as.numeric(echocardiogram.data.imp_mean$LVDD)
echocardiogram.data.imp_mean$WMS <- as.numeric(echocardiogram.data.imp_mean$WMS)
echocardiogram.data.imp_mean$WMI <- as.numeric(echocardiogram.data.imp_mean$WMI)
str(echocardiogram.data.imp_mean)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#Como dato a tener en cuenta podemos observar una diferencia de resultado en la fila 50 de la columna
#de clasificación, ya que con Median da FALSE y con Mean da TRUE. Ojo también con esto.

##APARTADO C

library(mice)
```

```
## Loading required package: Rcpp
```

```
## mice 2.25 2015-11-09
```

```
methods(mice)
```

```
## Warning in .S3methods(generic.function, class, parent.frame()): function
## 'mice' appears not to be S3 generic; found functions that look like S3
## methods
```

```
##  [1] mice.impute.2l.norm      mice.impute.2lonly.mean
##  [3] mice.impute.2lonly.norm  mice.impute.2lonly.pmm
##  [5] mice.impute.2l.pan       mice.impute.cart
##  [7] mice.impute.fastpmm      mice.impute.lda
##  [9] mice.impute.logreg       mice.impute.logreg.boot
## [11] mice.impute.mean         mice.impute.norm
## [13] mice.impute.norm.boot    mice.impute.norm.nob
## [15] mice.impute.norm.predict mice.impute.passive
## [17] mice.impute.pmm          mice.impute.polr
## [19] mice.impute.polyreg      mice.impute.quadratic
## [21] mice.impute.rf           mice.impute.ri
## [23] mice.impute.sample       mice.mids
## [25] mice.theme
## see '?methods' for accessing help and source code
```

```
tempImp.mice <- mice(echocardiogram.data, m=5, method = "pmm", seed = 500)
```

```
##
##  iter imp variable
##   1   1  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   1   2  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   1   3  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   1   4  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   1   5  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   2   1  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   2   2  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   2   3  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   2   4  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   2   5  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   3   1  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   3   2  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   3   3  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   3   4  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   3   5  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   4   1  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   4   2  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   4   3  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   4   4  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   4   5  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   5   1  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   5   2  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   5   3  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   5   4  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
##   5   5  Survival  StillAlive  AgeAttack  PericardEffu  FracShort  EPSS  LVDD  WMS  WMI  AliveAt1
```

```
summary(tempImp.mice)
```

```
## Multiply imputed data set
## Call:
## mice(data = echocardiogram.data, m = 5, method = "pmm", seed = 500)
## Number of multiple imputations:  5
## Missing cells per column:
##     Survival   StillAlive    AgeAttack PericardEffu    FracShort
```

```
##            2            1            5            1            8
##         EPSS         LVDD          WMS          WMI     AliveAt1
##           15           11            4            1            1
## Imputation methods:
##     Survival   StillAlive    AgeAttack PericardEffu    FracShort
##        "pmm"        "pmm"        "pmm"        "pmm"        "pmm"
##         EPSS         LVDD          WMS          WMI     AliveAt1
##        "pmm"        "pmm"        "pmm"        "pmm"        "pmm"
## VisitSequence:
##     Survival   StillAlive    AgeAttack PericardEffu    FracShort
##            1            2            3            4            5
##         EPSS         LVDD          WMS          WMI     AliveAt1
##            6            7            8            9           10
## PredictorMatrix:
##              Survival StillAlive AgeAttack PericardEffu FracShort EPSS
## Survival            0          1         1            1         1    1
## StillAlive          1          0         1            1         1    1
## AgeAttack           1          1         0            1         1    1
## PericardEffu        1          1         1            0         1    1
## FracShort           1          1         1            1         0    1
## EPSS                1          1         1            1         1    0
## LVDD                1          1         1            1         1    1
## WMS                 1          1         1            1         1    1
## WMI                 1          1         1            1         1    1
## AliveAt1            1          1         1            1         1    1
##              LVDD WMS WMI AliveAt1
## Survival        1   1   1        1
## StillAlive      1   1   1        1
## AgeAttack       1   1   1        1
## PericardEffu    1   1   1        1
## FracShort       1   1   1        1
## EPSS            1   1   1        1
## LVDD            0   1   1        1
## WMS             1   0   1        1
## WMI             1   1   0        1
## AliveAt1        1   1   1        0
## Random generator seed value:  500
```

```
tempImp.mice$imp$Survival
```

```
##        1  2    3    4  5
## 50 41.00 12 19.5 21.0  1
## 95  0.75  1  1.0  0.5 10
```

```
echocardiogram.data.imp_mice <- complete(tempImp.mice, 1)
complete.cases(echocardiogram.data.imp_mice)
```

```
##   [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [15] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [29] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [43] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [71] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
##  [85] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [99] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [113] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [127] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
str(echocardiogram.data.imp_mice)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
#La función mice() con el método pmm hace las imputaciones sobre todo el dataset ofreciendo un par de
#ventajas pero un gran inconveniente. Las dos ventajas es que lo hace todo de una tacada, y la segunda
#es que mantiene los tipos. El principal inconveniente es que en la columna de clasificación nos ha
#imputado un valor en el NA que teníamos en la fila 50 que se derivaba de los valores de las dos primer
#columnas, lo que puede dar como resultado a una incoherencia. Para asegurarnos de que no hay incoheren
#en la columna de clasificación podríamos recalcular el valor para la fila 50 a partir de los valores
#imputados en las dos primeras columnas.
echocardiogram.data.imp_mice$AliveAt1[50]
```

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

```
#De modo que recalculamos para asegurarnos de guardar la coherencia del experto.
echocardiogram.data.imp_mice$AliveAt1[50] <- calcularAliveAt1(echocardiogram.data.imp_mice$Survival[50]
echocardiogram.data.imp_mice$AliveAt1[50]
```

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

```
#Por suerte el valor se mantiene a FALSE, pero de este modo nos aseguramos la coherencia

#Función kNN
echocardiogram.data.imp_kNN <- kNN(echocardiogram.data, imp_var = FALSE)
complete.cases(echocardiogram.data.imp_mice)
```

```
##   [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [15] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [29] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [43] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [71] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [85] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [99] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [113] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [127] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
str(echocardiogram.data.imp_kNN)
```

```
## 'data.frame':    132 obs. of  10 variables:
##  $ Survival    : num  11 19 16 57 19 26 13 50 19 25 ...
##  $ StillAlive  : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ AgeAttack   : num  71 72 55 60 57 68 62 60 46 54 ...
##  $ PericardEffu: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ FracShort   : num  0.26 0.38 0.26 0.253 0.16 0.26 0.23 0.33 0.34 0.14 ...
##  $ EPSS        : num  9 6 4 12.1 22 ...
##  $ LVDD        : num  4.6 4.1 3.42 4.6 5.75 ...
##  $ WMS         : num  14 14 14 16 18 12 22.5 14 16 15.5 ...
##  $ WMI         : num  1 1.7 1 1.45 2.25 ...
##  $ AliveAt1    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```
echocardiogram.data.imp_kNN$Survival
```

```
##   [1] 11.00 19.00 16.00 57.00 19.00 26.00 13.00 50.00 19.00 25.00 10.00
##  [12] 52.00 52.00 44.00  0.50 24.00  0.50  0.50 22.00  1.00  0.75  0.75
##  [23]  0.50  5.00 48.00 29.00 29.00 29.00  0.25 36.00  1.00  1.00  3.00
##  [34] 27.00 35.00 26.00 16.00  1.00 19.00 31.00 32.00 16.00 40.00 46.00
##  [45]  2.00 37.00 19.50 20.00  0.25  0.75  2.00  7.00 10.00 12.00  1.00
##  [56] 10.00 45.00 22.00 53.00 38.00 26.00  9.00 26.00  0.50 12.00 49.00
##  [67]  0.75 49.00 47.00 41.00  0.25 33.00 29.00 41.00 26.00 15.00  0.25
##  [78]  0.03 12.00 32.00 32.00 27.00 23.00  0.75  0.75 34.00  1.00 21.00
##  [89] 55.00 15.00  0.50 35.00 53.00 33.00 33.00 33.00 40.00 33.00  5.00
## [100]  4.00 31.00 33.00 22.00 25.00  1.25 24.00 25.00 24.00  0.75  3.00
## [111] 27.00 13.00 36.00 25.00 27.00 34.00 37.00 34.00 28.00 28.00 17.00
## [122] 38.00 31.00 12.00 36.00 17.00 21.00  7.50 41.00 36.00 22.00 20.00
```

```
echocardiogram.data.imp_kNN$Survival_imp
```

```
## NULL
```

```
echocardiogram.data.imp_kNN$StillAlive
```

```
##   [1] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [12] FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE
##  [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [45]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
##  [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
##  [67]  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [78]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE  TRUE
##  [89] FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE
## [100]  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE
## [111] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [122] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
```

```
echocardiogram.data.imp_kNN$StillAlive_imp
```

```
## NULL
```

```
echocardiogram.data.imp_kNN$AliveAt1
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
##  [12] FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [45] FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
##  [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [67] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [78]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
##  [89] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
## [100]  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE
## [111] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [122] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
echocardiogram.data.imp_kNN$AliveAt1_imp
```

```
## NULL
```

```
#Al igual que con la función mice(), kNN hace las imputaciones sobre todo el dataset manteniendo los
#tipos originales de las columnas con las ventajas que eso conlleva. No obstante, esto acarrea
#un par de cosas negativas: 1) Por defecto nos genera por cada columna otra para indicarnos donde están
#las imputaciones. Por lo que duplica el número de columnas, esto se puede evitar mediante el argumento
#imp_var = FALSE, pero entonces no sabríamos qué valores han sido los imputados. Finalmente se he optad
#por no crear estas columnas de imputación para no duplicar. 2) Según las indicaciones del
#experto existe una incoherencia en la fila 50 de la columna de clasificación, ya que el valor de esta
#columna ha sido imputado también. Por lo que lo correcto sería recalcularlo como el experto nos indica
echocardiogram.data.imp_kNN$AliveAt1[50]
```

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

```
#De modo que recalculamos para asegurarnos de guardar la coherencia del experto.
echocardiogram.data.imp_kNN$AliveAt1[50] <- calcularAliveAt1(echocardiogram.data.imp_kNN$Survival[50],e
echocardiogram.data.imp_kNN$AliveAt1[50]
```

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

```
#En este caso la función kNN había imputado un TRUE cuando según el experto debería de ser un FALSE.
#Con esto queda corregida la incoherencia.

##APARTADO D

write.csv(echocardiogram.data.imp_median, file = "~/Dropbox/Universidad de Murcia/Máster Big Data/Miner
write.csv(echocardiogram.data.imp_mean, file = "~/Dropbox/Universidad de Murcia/Máster Big Data/Minería
write.csv(echocardiogram.data.imp_mice, file = "~/Dropbox/Universidad de Murcia/Máster Big Data/Minería
write.csv(echocardiogram.data.imp_kNN, file = "~/Dropbox/Universidad de Murcia/Máster Big Data/Minería


#COMPARACIÓN DE MODELOS
```

```r
#Ejercicio 5

#Cuando tenemos que comparar dos clasificadores en un dominio, el primer paso a seguir es comprobar que
#cumplen con las condiciones para que se pueda realizar un test t de Student con medidas pareadas.
#Para ello deben de cumplir dos condiciones: Normalidad y Homocedasticidad (homogeneidad de las varianz

#Primero antes de nada leemos el fichero de resultados de precisión
ejercicio1.dat <- read.csv("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Prácticas/
ejercicio1.dat
```

```
##          SVM       LDA
## 1  0.9333333 0.9333333
## 2  0.8750000 0.8750000
## 3  0.8125000 0.8750000
## 4  0.8125000 0.8750000
## 5  0.9333333 0.9333333
## 6  0.8000000 0.8000000
## 7  0.6000000 0.5333333
## 8  0.9375000 0.9375000
## 9  0.7500000 0.9375000
## 10 0.8000000 0.7333333
```

```r
#Comprobamos su normalidad con Shapiro ya que sólo disponemos de 10 muestras (menos de 50 muestras)
shapiro.test(ejercicio1.dat$SVM-ejercicio1.dat$LDA)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ejercicio1.dat$SVM - ejercicio1.dat$LDA
## W = 0.83634, p-value = 0.03989
```

```r
#Shapiro-Wilk normality test

#data:  ejercicio1.dat$SVM - ejercicio1.dat$LDA
#W = 0.83634, p-value = 0.03989

#Como podemos observar, podemos decir con un 95% de confianza que la muestra NO sigue una distribución
#normal ya que al hacer la diferencia entre los dos clasificadores su p-valor es 0,03989; es decir,
#menor que alfa = 0.05.

#Aunque no es necesario, porque nuestra muestra no es de 50 o más elementos, podemos también hacer
#el test de Lilliefors para contrastar
library(nortest)
lillie.test(ejercicio1.dat$SVM-ejercicio1.dat$LDA)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  ejercicio1.dat$SVM - ejercicio1.dat$LDA
## D = 0.29627, p-value = 0.01304
```

```r
#Lilliefors (Kolmogorov-Smirnov) normality test

#data:  ejercicio1.dat$SVM - ejercicio1.dat$LDA
#D = 0.29627, p-value = 0.01304

#El resultado es que su p-valor es 0.01304, o sea aún menor que el test anterior. Por lo que nuevamente
#este test nos indica que la muestra NO sigue una distribución normal. Y que por lo tanto ya no cumple
#el primer requisito para poder realizar el test de Student para medias pareadas. Por lo que, para este
#caso los tests más interesantes a aplicar los tests no paramétricos como el test de McNemar o el test
#de la suma de rangos de Wilcoxon. Pero antes de realizar los test necesitamos primero calcular la
#matriz de confusión.Para ello vamos hacer uso de los objetos SVMFit y LDAFit, y la base de datos hepat

#Los cargamos en primer lugar
SVMFit <- readRDS("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Prácticas/Ejercicios
SVMFit
```

```
## Support Vector Machines with Linear Kernel
##
## 155 samples
##  14 predictors
##   2 classes: 'FALLECE', 'VIVE'
##
## Pre-processing: centered (14), scaled (14)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 140, 139, 139, 139, 140, 140, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8441667  0.4581748
##
## Tuning parameter 'C' was held constant at a value of 1
##
```

```r
LDAFit <- readRDS("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Prácticas/Ejercicios
LDAFit
```

```
## Linear Discriminant Analysis
##
## 155 samples
##  14 predictors
##   2 classes: 'FALLECE', 'VIVE'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 140, 139, 139, 139, 140, 140, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8433333  0.4584597
##
##
```

```
hepatitis <- read.csv("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Prácticas/Ejerci
hepatitis
```

```
##     ALBUMINA PROTIME FOSFATOalc ESTEROIDES SEXO ANTIVIRALES HIGgrande
## 1        4.0      72         78          0    1           0         0
## 2        3.5      41         19          0    0           0         0
## 3        4.0      74         84          1    0           0         1
## 4        4.0      80         51          1    0           1         1
## 5        4.0      75         78          1    0           0         1
## 6        4.0      75         83          1    0           0         1
## 7        4.0      63         78          0    0           0         1
## 8        4.0      72         78          1    0           0         1
## 9        4.4      74         78          1    0           0         1
## 10       3.9      72         78          1    0           0         1
## 11       4.4      85         73          0    0           1         0
## 12       3.7      54         60          1    0           1         1
## 13       3.9      52         75          1    0           1         1
## 14       4.9      78         58          1    0           0         1
## 15       4.3      66         78          0    0           1         1
## 16       2.9      46         69          0    0           0         1
## 17       4.3      50          3          1    0           0         1
## 18       4.0      63         62          0    0           0         1
## 19       4.1      85         55          1    0           0         1
## 20       4.2      62         67          0    0           1         0
## 21       4.2      64         52          1    1           1         1
## 22       4.1      39         18          1    0           0         0
## 23       4.0     100         78          1    0           0         1
## 24       4.7      47         61          1    0           0         1
## 25       4.3      70         50          0    1           1         1
## 26       3.8     100         83          0    0           0         1
## 27       3.7      67         78          0    0           1         1
## 28       2.7      36         33          1    1           0         1
## 29       3.8     100         73          0    0           0         0
## 30       4.6      52         73          0    0           1         1
## 31       3.8      40         44          0    0           1         1
## 32       4.0      72         78          0    0           0         1
## 33       5.0      74         75          1    1           1         1
## 34       3.8      60         19          0    1           0         1
## 35       4.3      73         59          1    0           0         1
## 36       4.2      76         65          1    0           0         1
## 37       4.1      90         38          1    0           0         1
## 38       3.9      40         24          0    1           0         0
## 39       4.0     100         78          0    0           0         1
## 40       2.9      74         35          1    0           0         1
## 41       4.0      21         71          0    0           1         1
## 42       4.6      64         57          1    0           0         1
## 43       4.4      60         51          1    0           0         1
## 44       4.4     100         68          0    0           0         1
## 45       4.0      75         78          1    0           0         1
## 46       4.4      74         70          1    0           0         1
## 47       3.8      76         74          0    0           0         1
## 48       3.3      46         37          1    1           0         1
## 49       4.2      85         78          0    0           0         1
```

```
## 50       4.2      77         14          1     0           1          1
## 51       4.0      73         78          0     0           0          1
## 52       4.0      85         78          1     0           0          1
## 53       4.4      70          7          1     0           0          1
## 54       4.0     100         78          1     0           1          1
## 55       4.2      74         53          1     0           0          1
## 56       4.2     100         82          0     0           0          1
## 57       4.0      75         78          0     0           0          1
## 58       3.9      52         54          1     0           1          1
## 59       4.3      74         74          1     0           1          0
## 60       4.0      85         78          0     1           0          1
## 61       4.5     100         43          1     0           0          1
## 62       4.0      90          3          1     0           0          1
## 63       3.4      29         39          1     0           0          1
## 64       3.1      41         28          0     0           0          0
## 65       3.5      66          4          1     0           0          1
## 66       4.2     100         76          1     0           0          1
## 67       3.0      62         62          0     0           0          1
## 68       2.6      73         35          1     0           0          1
## 69       4.0      54         47          1     0           0          1
## 70       3.7      46         66          0     0           0          1
## 71       4.3     100         76          1     0           0          1
## 72       3.5      57         16          0     0           0          0
## 73       4.4      52         72          1     0           0          1
## 74       4.0      57         51          0     0           0          1
## 75       5.3      41          2          1     0           0          1
## 76       4.1      56         56          0     0           1          1
## 77       3.3      58         31          1     0           0          0
## 78       4.0      76         46          0     1           1          1
## 79       4.4      57         69          0     0           0          0
## 80       4.9      74         78          1     0           0          1
## 81       4.8     100         76          1     0           0          1
## 82       4.2     100         62          1     0           0          1
## 83       4.0     100          2          0     0           0          1
## 84       2.9      38         34          0     1           0          1
## 85       3.9      58         22          1     1           0          1
## 86       4.0      46         49          0     0           0          1
## 87       4.3      41         19          0     0           0          1
## 88       2.8      78         29          1     0           0          1
## 89       2.8      56         10          0     0           0          1
## 90       4.4      84         72          0     0           0          0
## 91       3.4      41         40          0     1           0          0
## 92       3.3      47         64          0     0           0          1
## 93       4.0      72         78          1     0           0          1
## 94       2.9      36         34          0     0           0          1
## 95       3.6      38          5          0     0           0          1
## 96       4.2      67         48          0     0           1          0
## 97       3.9     100         23          0     0           0          1
## 98       3.5      43          8          0     0           0          0
## 99       4.2      66         77          1     0           0          1
## 100      4.3      74         78          0     0           0          0
## 101      2.7      31         13          0     0           0          1
## 102      4.4      64         78          1     0           0          1
## 103      3.0      66         32          0     0           0          1
```

```
## 104      3.9      51         39          0    0           0         1
## 105      2.1      46         79          1    0           0         1
## 106      6.4      70         74          1    0           0         1
## 107      2.6      41         20          0    0           0         1
## 108      3.9      66         78          0    0           0         1
## 109      3.6      67         25          0    0           1         0
## 110      3.0      31         63          0    0           0         1
## 111      4.2      80         42          1    0           0         1
## 112      3.8      29         62          0    0           1         1
## 113      4.0      52         78          0    0           0         1
## 114      3.0      66         75          0    0           0         1
## 115      3.3      57         22          0    0           0         1
## 116      3.8      50         11          1    1           0         1
## 117      4.1      72         78          0    0           0         1
## 118      3.9      62         21          1    0           0         1
## 119      4.0      74         78          0    0           0         1
## 120      3.8      48         78          1    0           0         0
## 121      4.0      66         81          0    0           0         0
## 122      2.9      23         27          0    0           0         1
## 123      2.9      47         78          1    0           0         1
## 124      4.3      73         72          0    0           0         1
## 125      4.0      72         78          1    0           0         1
## 126      4.1     100         67          1    1           0         0
## 127      4.0      72         78          1    0           0         1
## 128      2.4      32         25          1    0           0         1
## 129      3.1      66         78          0    0           0         1
## 130      3.3      30         76          0    0           0         1
## 131      4.5       0         78          1    0           0         1
## 132      2.2      54         29          0    0           0         1
## 133      3.8      58         26          1    0           0         1
## 134      3.4      50          9          1    0           1         1
## 135      3.8      90         41          0    0           0         1
## 136      4.5      57         36          1    0           0         0
## 137      4.5      63         76          1    0           0         0
## 138      3.5      56         17          1    0           0         1
## 139      2.6      31         30          1    0           0         1
## 140      4.2      85         78          1    0           1         1
## 141      2.7      57         45          0    0           0         0
## 142      3.5      43         12          0    0           0         1
## 143      3.0      63         47          1    0           0         1
## 144      3.5      35         78          0    0           0         1
## 145      2.4      40         45          1    0           0         1
## 146      4.2      54         71          0    0           0         1
## 147      3.4      46         64          1    0           0         1
## 148      2.8      35          6          0    0           0         1
## 149      4.0      64         80          0    0           0         1
## 150      4.0      67         12          1    0           0         1
## 151      3.3      50         17          1    0           0         1
## 152      4.3      47         15          1    0           0         1
## 153      4.1      85         71          0    0           0         0
## 154      4.1      48         75          0    1           0         1
## 155      3.1      42          2          1    0           0         1
##     HIGfirme ANOREXIA SGOT BAZOpalpa VARICES EDAD FATIGA PRONOSTICO
## 1          0        0   18         0       0   30      0       VIVE
```

```
## 2          0         0   42         0         0   50       1       VIVE
## 3          0         0   32         0         0   78       1       VIVE
## 4          0         0   52         0         0   31       0       VIVE
## 5          0         0  200         0         0   34       0       VIVE
## 6          0         0   28         0         0   34       0       VIVE
## 7          0         1   60         1         0   51       1    FALLECE
## 8          0         0   60         0         0   23       0       VIVE
## 9          1         0   48         0         0   39       1       VIVE
## 10         0         0  120         0         0   30       0       VIVE
## 11         1         0   30         0         0   39       0       VIVE
## 12         1         0  249         0         0   32       1       VIVE
## 13         1         0   60         0         0   41       1       VIVE
## 14         1         0  144         0         0   30       1       VIVE
## 15         0         0   60         0         0   47       0       VIVE
## 16         0         1   89         0         0   38       1       VIVE
## 17         0         0   53         0         0   66       1       VIVE
## 18         1         0  166         0         0   40       1       VIVE
## 19         0         0   42         0         0   38       0       VIVE
## 20         1         0   28         0         0   38       0       VIVE
## 21         0         0   20         0         0   22       1       VIVE
## 22         1         1   98         1         0   27       1       VIVE
## 23         0         0   20         0         0   31       0       VIVE
## 24         0         0   63         0         0   42       0       VIVE
## 25         0         0   18         0         0   25       0       VIVE
## 26         0         0   46         0         0   27       1       VIVE
## 27         1         1   48         0         0   49       1       VIVE
## 28         1         0   55         0         0   58       1       VIVE
## 29         1         0   25         0         0   61       1       VIVE
## 30         0         0   58         0         0   51       1       VIVE
## 31         1         0   98         0         0   39       1    FALLECE
## 32         0         0   60         0         0   62       1    FALLECE
## 33         0         1   53         0         0   41       1       VIVE
## 34         1         0   29         0         0   26       0       VIVE
## 35         0         0   92         0         0   35       1       VIVE
## 36         0         0   28         0         0   37       1    FALLECE
## 37         0         1  150         1         0   23       1       VIVE
## 38         1         1   68         1         0   20       1       VIVE
## 39         0         0   14         0         0   42       0       VIVE
## 40         1         0   53         1         0   65       1       VIVE
## 41         0         0   55         0         0   52       0       VIVE
## 42         0         0   16         0         0   23       0       VIVE
## 43         0         0   90         0         0   33       0       VIVE
## 44         0         0   18         0         0   56       1       VIVE
## 45         0         0   86         0         0   34       0       VIVE
## 46         0         0  110         0         0   28       1       VIVE
## 47         1         0   80         0         0   37       0       VIVE
## 48         1         0  420         0         0   28       1       VIVE
## 49         0         0   44         1         0   36       0       VIVE
## 50         0         1   65         0         0   38       1       VIVE
## 51         0         0   60         0         0   39       0       VIVE
## 52         0         0   20         0         0   39       0       VIVE
## 53         0         0  145         0         0   44       0       VIVE
## 54         1         0   31         1         0   40       1       VIVE
## 55         0         0   78         0         0   30       1       VIVE
```

```
## 56          0         1   59          0         0   37       1       VIVE
## 57          0         0   86          0         0   34       1       VIVE
## 58          0         0   38          0         0   30       0       VIVE
## 59          1         0   38          0         0   64       1       VIVE
## 60          0         0   75          1         0   45       1       VIVE
## 61          0         0   58          0         0   37       0       VIVE
## 62          0         0   64          0         0   32       0       VIVE
## 63          0         1   54          0         1   32       1       VIVE
## 64          1         0   44          1         0   36       0       VIVE
## 65          0         0   43          0         0   49       1       VIVE
## 66          0         0   38          0         0   27       0       VIVE
## 67          0         0   33          0         0   56       0       VIVE
## 68          0         1   48          0         0   57       1    FALLECE
## 69          0         0   15          0         0   39       1       VIVE
## 70          0         0   68          0         0   44       1       VIVE
## 71          0         0   39          0         0   24       0       VIVE
## 72          1         0  182          0         0   34       1    FALLECE
## 73          0         1  271          0         0   51       1       VIVE
## 74          1         1   45          0         0   36       1       VIVE
## 75          0         0  100          0         0   50       0       VIVE
## 76          0         0   45          0         0   32       1       VIVE
## 77          1         0  242          1         0   58       1    FALLECE
## 78          1         0   24          0         0   34       0       VIVE
## 79          1         0   46          0         0   34       1       VIVE
## 80          0         0   31          0         0   28       0       VIVE
## 81          0         1   14          0         0   23       1       VIVE
## 82          0         0  224          0         0   36       0       VIVE
## 83          0         0   31          0         0   30       0       VIVE
## 84          0         0   69          0         0   67       1       VIVE
## 85          1         0  156          0         0   62       1       VIVE
## 86          1         1  123          0         0   28       1       VIVE
## 87          0         0   55          1         1   44       1    FALLECE
## 88          1         1   64          0         1   30       1    FALLECE
## 89          1         1   16          0         1   38       1    FALLECE
## 90          1         1   18          0         0   38       1       VIVE
## 91          1         0  117          1         0   50       1       VIVE
## 92          0         1   55          1         1   42       1    FALLECE
## 93          0         0   60          0         0   33       0       VIVE
## 94          0         0   69          0         0   52       0       VIVE
## 95          1         0  157          1         0   59       1    FALLECE
## 96          1         1   69          0         0   40       1       VIVE
## 97          1         0  128          0         0   30       1       VIVE
## 98          1         0   65          0         0   44       1       VIVE
## 99          0         0   23          0         1   47       0    FALLECE
## 100         1         0   40          1         0   60       1       VIVE
## 101         1         0  157          0         1   48       1    FALLECE
## 102         0         0   24          0         0   22       0       VIVE
## 103         1         0  227          0         0   27       1       VIVE
## 104         1         1  269          1         1   51       1       VIVE
## 105         1         0   20          0         1   47       1    FALLECE
## 106         0         0   34          0         0   25       0       VIVE
## 107         0         0   58          1         0   35       1    FALLECE
## 108         0         1  648          0         0   45       1       VIVE
## 109         1         0  225          0         0   54       0       VIVE
```

```
## 110        0        0   80         0        0   33       1    FALLECE
## 111        1        0   25         1        0    7       0       VIVE
## 112        0        0   68         0        0   42       1    FALLECE
## 113        0        0   30         0        0   52       1       VIVE
## 114        1        0   65         1        0   45       1       VIVE
## 115        0        0   75         0        0   36       0       VIVE
## 116        0        0  136         0        0   69       1       VIVE
## 117        0        0   34         0        0   24       1       VIVE
## 118        0        0   81         0        0   50       0       VIVE
## 119        0        0   60         0        0   61       1    FALLECE
## 120        1        0   28         0        0   54       1       VIVE
## 121        1        1  153         0        0   56       1    FALLECE
## 122        0        1  118         0        0   20       1       VIVE
## 123        0        0   40         1        0   42       0       VIVE
## 124        0        0  231         0        0   37       1       VIVE
## 125        1        0   75         1        0   50       0       VIVE
## 126        1        1   24         0        0   34       1       VIVE
## 127        0        1   20         0        0   28       1       VIVE
## 128        1        0   75         1        1   50       1    FALLECE
## 129        0        0   92         0        0   54       1       VIVE
## 130        0        0   55         0        0   57       1    FALLECE
## 131        0        0   30         0        0   54       0       VIVE
## 132        0        1  101         1        0   31       1    FALLECE
## 133        1        1  278         0        0   48       1       VIVE
## 134        1        0   52         0        0   72       1       VIVE
## 135        1        0   49         0        0   38       0    FALLECE
## 136        1        0  181         1        1   25       1       VIVE
## 137        1        0   33         0        0   51       0       VIVE
## 138        1        0  140         0        1   38       0       VIVE
## 139        1        0   30         0        1   47       1    FALLECE
## 140        0        0   44         0        0   45       0       VIVE
## 141        1        1   60         0        1   36       1       VIVE
## 142        1        0   28         1        0   54       1    FALLECE
## 143        1        0   20         1        1   51       1       VIVE
## 144        0        0   70         1        0   49       1    FALLECE
## 145        0        1  114         0        0   45       1    FALLECE
## 146        0        0  173         0        0   31       1       VIVE
## 147        1        0  120         1        1   41       1    FALLECE
## 148        1        1  528         0        0   70       1    FALLECE
## 149        0        0  152         0        0   20       0       VIVE
## 150        0        0   30         0        0   36       0       VIVE
## 151        0        1  242         0        1   46       1    FALLECE
## 152        1        0  142         0        0   44       1       VIVE
## 153        1        0   20         0        0   61       1       VIVE
## 154        0        0   19         1        1   53       1       VIVE
## 155        0        0   19         1        0   43       1    FALLECE
```

```
#Cargamos la librería caret para hacer las predicciones de ambos clasificadores y así poder extraer
#la matriz de confusión
library(caret)
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:survival':
##
##     cluster
```

```
prediccion1 <- predict(LDAFit, hepatitis[,c(1:dim(hepatitis)[2]-1)])
```

```
## Loading required package: MASS
```

```
prediccion2 <- predict(SVMFit, hepatitis[,c(1:dim(hepatitis)[2]-1)])
```

```
## Loading required package: kernlab
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
mc <- confusionMatrix(prediccion1, prediccion2)
matrizConfusion <- as.matrix(mc)
matrizConfusion
```

```
##         FALLECE VIVE
## FALLECE      22    6
## VIVE          0  127
```

```
#Por último aplicamos el test de McNemar que es el más fiable para este caso.
mcnemar.test(matrizConfusion)
```

```
##
##  McNemar's Chi-squared test with continuity correction
##
## data:  matrizConfusion
## McNemar's chi-squared = 4.1667, df = 1, p-value = 0.04123
```

```
# McNemar's Chi-squared test with continuity correction
#
# data:  matrizConfusion
# McNemar's chi-squared = 4.1667, df = 1, p-value = 0.04123

#Como conclusión de este ejercicio 5, podemos observar que el test de McNemar nos indica, con un 95% de
#confianza, de que NO hay diferencias significativas entre ambos modelos ya que X²=4.1667 y el p-value
#de 0.04123 es menor que alfa que es 0.05.


#Ejercicio 6

#Cuando tenemos multiples clasificadores debemos de aplicar el test de ANOVA para medidas pareadas o
#bien el test de Friedman (que es su equivalente no paramétrico). Pero antes de aplicar el test de
```

```r
#ANOVA tenemos que comprobar que cumple dos condiciones: Normalidad y esfericidad (que es el
#equivalente a la homocedasticidad del test de Student pero para el test de ANOVA de medidas pareadas)

ejercicio2.dat <- read.csv("~/Dropbox/Universidad de Murcia/Máster Big Data/Minería de Datos/Prácticas/
ejercicio2.dat
```

```
##      RegA    RegB    RegC    RegD
## 1 5190.18 5048.76 5545.57 5864.49
## 2 5361.07 5331.02 5107.87 5420.78
## 3 5457.76 4839.14 5142.34 5421.95
## 4 5246.77 5107.47 5308.94 4989.79
## 5 5182.62 5015.22 5548.14 5229.75
## 6 5046.50 5232.16 5151.10 5347.82
## 7 5127.59 5104.78 5534.55 5786.63
## 8 6873.05 4575.82 6193.76 5993.44
## 9 4879.28 4820.10 5344.75 5591.12
```

```r
#A continuación, con el propósito de facilitar los cálculos, vamos a generar una tabla que tenga tres
#columnas: una para la precisión, otra para los clasificadores y otra para los conjuntos de datos.

df.stack <- stack(ejercicio2.dat)
df.stack$DataSet <- as.factor(rep(row.names(ejercicio2.dat), times = 4))
names(df.stack) <- c("MSE","Regressor", "DataSet")
df.stack
```

```
##        MSE Regressor DataSet
## 1  5190.18      RegA       1
## 2  5361.07      RegA       2
## 3  5457.76      RegA       3
## 4  5246.77      RegA       4
## 5  5182.62      RegA       5
## 6  5046.50      RegA       6
## 7  5127.59      RegA       7
## 8  6873.05      RegA       8
## 9  4879.28      RegA       9
## 10 5048.76      RegB       1
## 11 5331.02      RegB       2
## 12 4839.14      RegB       3
## 13 5107.47      RegB       4
## 14 5015.22      RegB       5
## 15 5232.16      RegB       6
## 16 5104.78      RegB       7
## 17 4575.82      RegB       8
## 18 4820.10      RegB       9
## 19 5545.57      RegC       1
## 20 5107.87      RegC       2
## 21 5142.34      RegC       3
## 22 5308.94      RegC       4
## 23 5548.14      RegC       5
## 24 5151.10      RegC       6
## 25 5534.55      RegC       7
## 26 6193.76      RegC       8
## 27 5344.75      RegC       9
```

```
## 28 5864.49      RegD       1
## 29 5420.78      RegD       2
## 30 5421.95      RegD       3
## 31 4989.79      RegD       4
## 32 5229.75      RegD       5
## 33 5347.82      RegD       6
## 34 5786.63      RegD       7
## 35 5993.44      RegD       8
## 36 5591.12      RegD       9
```

```
#Ahora comprobamos la Normalidad. Como las muestras son sólo de 9 elementos, por lo que es inferior a 5
#aplicamos el test de Shapiro-Wilk
test <- tapply(df.stack$MSE, df.stack$Regressor, shapiro.test)
test
```

```
## $RegA
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.67109, p-value = 0.0006518
##
##
## $RegB
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.96076, p-value = 0.8062
##
##
## $RegC
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.83463, p-value = 0.05032
##
##
## $RegD
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.97161, p-value = 0.9081
```

```
# $RegA
#
# Shapiro-Wilk normality test
#
# data:  X[[i]]
# W = 0.67109, p-value = 0.0006518
#
```

```
#
# $RegB
#
# Shapiro-Wilk normality test
#
# data:  X[[i]]
# W = 0.96076, p-value = 0.8062
#
#
# $RegC
#
# Shapiro-Wilk normality test
#
# data:  X[[i]]
# W = 0.83463, p-value = 0.05032
#
#
# $RegD
#
# Shapiro-Wilk normality test
#
# data:  X[[i]]
# W = 0.97161, p-value = 0.9081

#Como se puede apreciar la regresión RegA tiene p-value = 0.0006518 que es inferior a alfa = 0.05.
#Por lo que concluimos con un 95% de confianza que NO hay una distribución normal, y por tanto,
#no cumple el primer requisito para poderle realizar el test de ANOVA. Tendremos que usar el test
#de Friedman para este caso no paramétrico.
test.friedman <- friedman.test(MSE ~ Regressor | DataSet, data = df.stack )
test.friedman
```

```
##
##  Friedman rank sum test
##
## data:  MSE and Regressor and DataSet
## Friedman chi-squared = 9.2667, df = 3, p-value = 0.02595
```

```
# Friedman rank sum test
#
# data:  MSE and Regressor and DataSet
# Friedman chi-squared = 9.2667, df = 3, p-value = 0.02595

#Como se puede apreciar, podemos afirmar con el 95% de confianza de que existen diferencias
#significativas en los Errores Cuadráticos Medios obtenidos por los cuatro regresores en los
#conjuntos de datos utilizados (F = 9,2667, p-value = 0,02595).

#Una vez confirmada la existencia de estas diferencias entre los regresores, tenemos que aplicar el
#test posthoc de Nemenyi para localizar dónde se encuentran dichas diferencias.

library(PMCMR)
nemenyi.test <- posthoc.friedman.nemenyi.test(MSE ~ Regressor | DataSet, data = df.stack)
nemenyi.test
```

```
##
```

```
##  Pairwise comparisons using Nemenyi multiple comparison test
##              with q approximation for unreplicated blocked data
##
## data:  MSE and Regressor and DataSet
##
##      RegA  RegB  RegC
## RegB 0.261 -     -
## RegC 0.983 0.126 -
## RegD 0.692 0.018 0.885
##
## P value adjustment method: none
```

```
# Pairwise comparisons using Nemenyi multiple comparison test
# with q approximation for unreplicated blocked data
#
# data:  MSE and Regressor and DataSet
#
#      RegA  RegB  RegC
# RegB 0.261 -     -
# RegC 0.983 0.126 -
# RegD 0.692 0.018 0.885
#
# P value adjustment method: none

#Observando los resultados, podemos comprobar, con una confianza del 95%, que no existen diferencias
#entre RegA y el resto de regresores, tampoco entre RegB y RegC, ni entre RegC y Reg D; pero sí que
#existen diferencias signitficativas entre el regresor RegB y el RegD ya que su p-value = 0.018 es
#menor al de alfa = 0.05.
```