Tarea Machine Learning en el Projecto MONICA de la OMS

2019-09-13

Profesor: Javier Portela

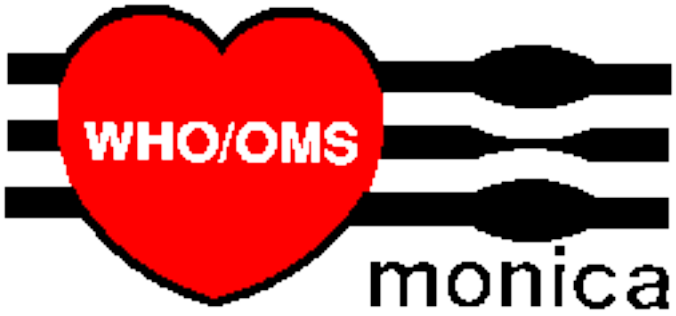
Alumno: Beatriz González García

1.- INTRODUCCIÓN

El proyecto MONICA (Multinational **MONI**toring of Trends and Determinants in **CA**rdiovascular Disease) de la OMS se diseñó en los años 1980 para estudiar las tendencias y determinantes en las enfermedades cardiovasculares. Se llevó a cabo en 32 centros de 21 países, recogiendo datos de 10 millones de individuos de ambos sexos con edades comprendidas entre 25 y 64 años.

2.- DATOS UTILIZADOS

Este análisis está basado en el dataset monica. El objetivo es predecir la variable **Outcome**, binaria que mide el resultado de mortalidad como un factor con dos niveles live o dead. Para la realización de esta tarea se ha usado el paquete R.

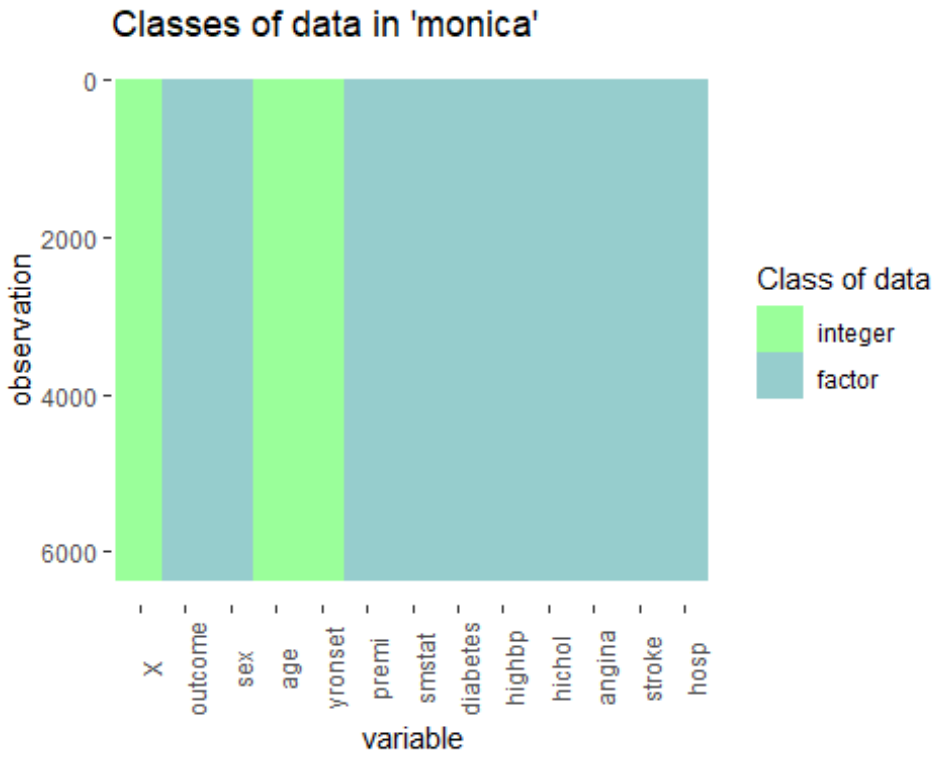


3.- ESTUDIO DEL DATASET

Es un archivo en formato csv.

```
monica <- read.csv("monica.csv",header=TRUE,sep = ",")
```

Hacemos un primer análisis exploratorio



Cambiamos el nombre de las variables.

```
names(monica)

##  [1] "X"         "outcome"   "sex"       "age"       "yronset"   "premi"
##  [7] "smstat"    "diabetes" "highbp"    "hichol"    "angina"    "stroke"
## [13] "hosp"

names (monica) = c("X", "Mortalidad", "Sexo", "Edad", "Year",
                   "Infarto","Fumador", "Diabetes",
                   "Tension","Colesterol","Angina",
                   "Accidente","Hospitalizacion")
```

El conjunto de datos está formado por 6.367 observaciones y 13 variables.

Dentro de estas variables dos son cuantitativas y el resto cualitativas.

De las 13 variables **X** corresponde al número secuencial que identifica la observación por lo que en realidad son 12 variables efectivas.

Son las siguientes:

- *Sexo* - Sexo
- *Edad* - Edad
- *Year* - Año en que se tomó el dato

- *Infarto* - Infarto de miocardio previo, un factor con los niveles y(yes), n(not) y nk (not known)
- *Fumador* - Status de fumador, un factor con los niveles c(Fumador), x(Exfumador), n(Nofumador) y nk (not known)
- *Diabetes* - Un factor con los niveles y(yes), n(not) y nk (not known)
- *Tension* - Presión arterial alta, un factor con los niveles y(yes), n(not) y nk (not known)
- *Colesterol* - Colesterol alto, un factor con los niveles y(yes), n(not) y nk (not known)
- *Angina* - Angina de pecho previa, un factor con los niveles y(yes), n(not) y nk (not known)
- *Accidente* - Accidente cerebrovascular, un factor con los niveles y(yes), n(not) y nk (not known)
- *Hospitalizacion* - Hospitalización después de sufrir el infarto, un factor con los niveles y(yes) y n(not)

La variable objetivo es Mortalidad que toma dos posibles valores

- *live* - Supervivencia a un infarto
- *dead* - Muerte después de un infarto

Recodificamos algunas variables para que sea más sencillo el estudio de los datos. Para ello utilizamos la función recode de la librería car.

```
            monica$Sexo <- car::recode(monica$Sexo,
                                       "c('f')='Mujer';
                                        c('m')='Hombre'")
         monica$Infarto <- car::recode(monica$Infarto,
                                       "c('y')='Si';
                                        c('n')='No'")
         monica$Fumador <- car::recode(monica$Fumador,
                                       "c('c')='Habitual';
                                        c('x')='Exfumador';
                                        c('n')='Nofumador'")
       monica$Diabetes <- car::recode(monica$Diabetes,
                                       "c('y')='Si';
                                        c('n')='No'")
         monica$Tension <- car::recode(monica$Tension,
                                       "c('y')='Si';
                                        c('n')='No'")
     monica$Colesterol <- car::recode(monica$Colesterol,
                                       "c('y')='Si';
                                        c('n')='No'")
          monica$Angina <- car::recode(monica$Angina,
                                       "c('y')='Si';
                                        c('n')='No'")
       monica$Accidente <- car::recode(monica$Accidente,
                                       "c('y')='Si';
```

```
                                          c('n')='No'")
  monica$Hospitalizacion <- car::recode(monica$Hospitalizacion,
                                        "c('y')='Si';
                                         c('n')='No'")
```

Vemos las primeras observaciones

```
head(monica)
```

```
##   X Mortalidad   Sexo Edad Year Infarto   Fumador Diabetes Tension
## 1 1       live  Mujer   63   85      No Exfumador       No      Si
## 2 2       live Hombre   59   85      Si Exfumador       No      Si
## 3 3       live Hombre   68   85      No Nofumador       No      Si
## 4 4       live Hombre   46   85      No  Habitual       No      No
## 5 5       dead Hombre   48   85      No Nofumador       Si      No
## 6 6       live  Mujer   55   85      No  Habitual       No      Si
##   Colesterol Angina Accidente Hospitalizacion
## 1         Si     No        No              Si
## 2         No     No        No              Si
## 3         No     No        No              Si
## 4         No     No        No              Si
## 5         No     Si        No              Si
## 6         Si     No        No              Si
```

Un resumen de los datos

```
summary(monica)
```

```
##        X        Mortalidad      Sexo           Edad            Year
##  Min.   :   1   dead:2842   Hombre:4605   Min.   :35.00   Min.
:85.00
##  1st Qu.:1592   live:3525   Mujer :1762   1st Qu.:55.00   1st
Qu.:87.00
##  Median :3184                             Median :61.00   Median
:89.00
##  Mean   :3184                             Mean   :59.42   Mean
:88.75
##  3rd Qu.:4776                             3rd Qu.:66.00   3rd
Qu.:91.00
##  Max.   :6367                             Max.   :69.00   Max.
:93.00
##  Infarto        Fumador     Diabetes  Tension   Colesterol Angina
##  nk: 734   Exfumador:1938   nk: 885   nk: 948   nk:1233    nk: 975
##  No:4122   Habitual :2051   No:4664   No:2542   No:3294    No:3473
##  Si:1511   nk       : 918   Si: 818   Si:2877   Si:1840    Si:1919
##            Nofumador:1460
##
##
##  Accidente Hospitalizacion
##  nk: 926   No:1925
##  No:4881   Si:4442
```

```
##  Si: 560
##
##
##
```

Y la tipología de las variables

```
str(monica)
```

```
## 'data.frame':    6367 obs. of  13 variables:
##  $ X              : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Mortalidad     : Factor w/ 2 levels "dead","live": 2 2 2 2 1 2 2 2
2 2 ...
##  $ Sexo           : Factor w/ 2 levels "Hombre","Mujer": 2 1 1 1 1 2 1
2 1 2 ...
##  $ Edad           : int  63 59 68 46 48 55 56 68 69 64 ...
##  $ Year           : int  85 85 85 85 85 85 85 85 85 85 ...
##  $ Infarto        : Factor w/ 3 levels "nk","No","Si": 2 3 2 2 2 2 2 3
2 2 ...
##  $ Fumador        : Factor w/ 4 levels "Exfumador","Habitual",..: 1 1
4 2 4 2 1 3 4 1 ...
##  $ Diabetes       : Factor w/ 3 levels "nk","No","Si": 2 2 2 2 3 2 2 1
2 2 ...
##  $ Tension        : Factor w/ 3 levels "nk","No","Si": 3 3 3 2 2 3 3 3
3 3 ...
##  $ Colesterol     : Factor w/ 3 levels "nk","No","Si": 3 2 2 2 2 3 2 1
3 2 ...
##  $ Angina         : Factor w/ 3 levels "nk","No","Si": 2 2 2 2 3 2 2 3
2 3 ...
##  $ Accidente      : Factor w/ 3 levels "nk","No","Si": 2 2 2 2 2 2 2 2
2 2 ...
##  $ Hospitalizacion: Factor w/ 2 levels "No","Si": 2 2 2 2 2 2 2 2 2 2
...
```

Contamos para las variables numéricas el número de valores diferentes:

```
sapply(Filter(is.numeric, monica),function(x) length(unique(x)))
```

```
##    X Edad Year
## 6367   35    9
```

La variable Year se puede convertir en factor porque sólo tiene nueve categorías

```
# convertimos en factor la variable Year que sólo tiene nueve categorías

monica$Year <- factor(monica$Year)
```

3.1.- Tratamiento Missings

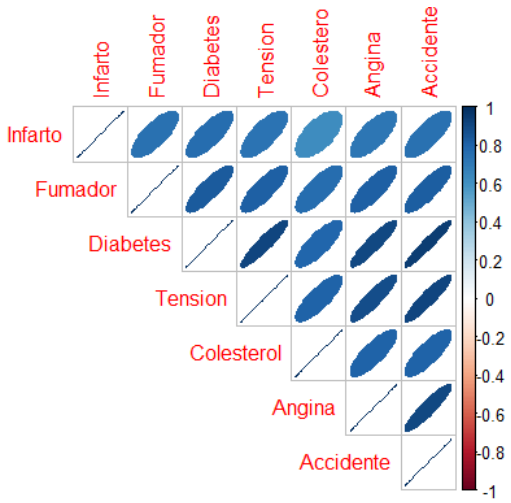
En este archivo hay muchas variables que presentan el valor nk(not know) estas variables son missings y por eso las recodificamos como NA.

Utilizamos la función recode del paquete questionr

```
monica$Infarto <- questionr::recode.na(monica$Infarto,"nk")
monica$Fumador <- questionr::recode.na(monica$Fumador,"nk")
monica$Diabetes <- questionr::recode.na(monica$Diabetes,"nk")
monica$Tension <- questionr::recode.na(monica$Tension,"nk")
monica$Colesterol <- questionr::recode.na(monica$Colesterol,"nk")
monica$Angina <- questionr::recode.na(monica$Angina,"nk")
monica$Accidente <- questionr::recode.na(monica$Accidente,"nk")
```

Vemos si hay algún patrón en la proporción de missings

```
corrplot(cor(is.na(monica[colnames(monica)[colSums(is.na(monica))>0]])),method = "ellipse",type = "upper")
```



Vemos que hay observaciones en las que la mayoría de los valores son missings

Incorporamos en cada fila su proporción de missings para posteriormente poder seleccionar las observaciones con menor número de missings:

```
monica$prop_missings<-apply(is.na(monica),1,mean)
```

Por otro lado también vemos la proporción de missings por variable

```
# Vemos la proporcion de missings por variable
(prop_missingsVars<-apply(is.na(monica),2,mean))
```

```
##                X      Mortalidad            Sexo            Edad
##       0.0000000       0.0000000       0.0000000       0.0000000
##            Year         Infarto         Fumador        Diabetes
##       0.0000000       0.1152819       0.1441809       0.1389980
##         Tension      Colesterol          Angina       Accidente
##       0.1488927       0.1936548       0.1531333       0.1454374
## Hospitalizacion   prop_missings
##       0.0000000       0.0000000
```

No hay variables con más del 50% de missings.

Seleccionamos del conjunto de datos aquellas observaciones con menos del 50 % de missings:

```
input <- subset(monica, prop_missings< 0.5)#Se han eliminado 621
observaciones
```

Utilizamos las funciones de mineria de datos para imputar los NA por un valor aleatorio:

```
# Aplicamos a todas las variables factor la transformación de los NA a un
valor aleatorio
input[,as.vector(which(sapply(input, class) == "factor"))]<-
sapply(Filter(is.factor, input),function(x)
ImputacionCuali(x,"aleatorio"))

# como esta funciona cambia los factores a character los volvemos a
cambiar a factor
input[,as.vector(which(sapply(input, class) == "character"))] <-
lapply(input[,as.vector(which(sapply(input, class)=="character"))] ,
factor)

# Comprobamos que no quedan observaciones con valores missings
summary(input)
```

```
##        X         Mortalidad      Sexo           Edad             Year
##  Min.   :   1   dead:2221   Hombre:4163   Min.   :35.00   87     : 734
##  1st Qu.:1569   live:3525   Mujer :1583   1st Qu.:55.00   85     : 706
##  Median :3104                             Median :61.00   86     : 691
##  Mean   :3156                             Mean   :59.35   90     : 641
##  3rd Qu.:4749                             3rd Qu.:66.00   89     : 640
##  Max.   :6367                             Max.   :69.00   88     : 639
##                                                           (Other):1695
##  Infarto        Fumador     Diabetes  Tension   Colesterol Angina
##  No:4211   Exfumador:2036   No:4892   No:2697   No:3652    No:3699
##  Si:1535   Habitual :2168   Si: 854   Si:3049   Si:2094    Si:2047
##            Nofumador:1542
##
##
##
##
##  Accidente Hospitalizacion prop_missings
##  No:5158   No:1413         Min.   :0.00000
##  Si: 588   Si:4333         1st Qu.:0.00000
##                            Median :0.00000
##                            Mean   :0.03042
##                            3rd Qu.:0.00000
##                            Max.   :0.46154
##
```

Nos quedamos con los datos depurados

```
# Quitamos el identificador(X) y la proporcion de missings(14) y
guardamos los datos depurados
Monica_Dep <- as.data.frame(input[,-c(1,14)])
```

3.2.- Tratamiento Outliers

En este dataset sólo hay una variable cuantitativa. Y comprobamos que no presenta valores fuera de rango

```
psych::describe(Filter(is.numeric, monica))
```

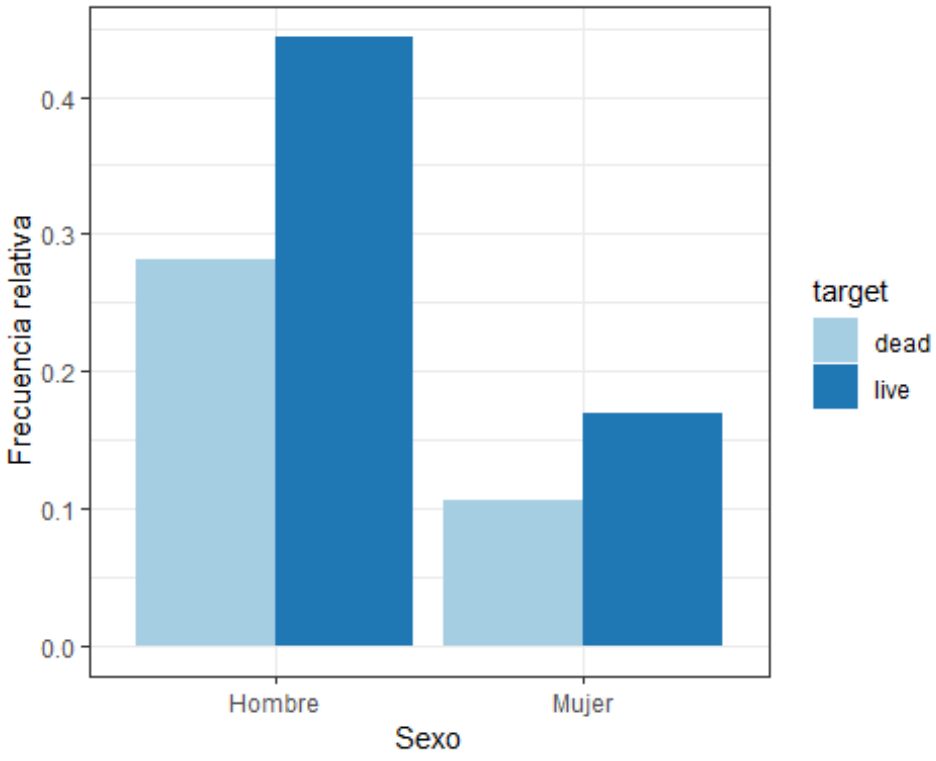
```
##    vars    n  mean   sd median trimmed  mad min max range  skew
kurtosis
## X1    1 5746 59.35 7.89     61   60.29 7.41  35  69    34 -0.93
0.17
##     se
## X1 0.1
```

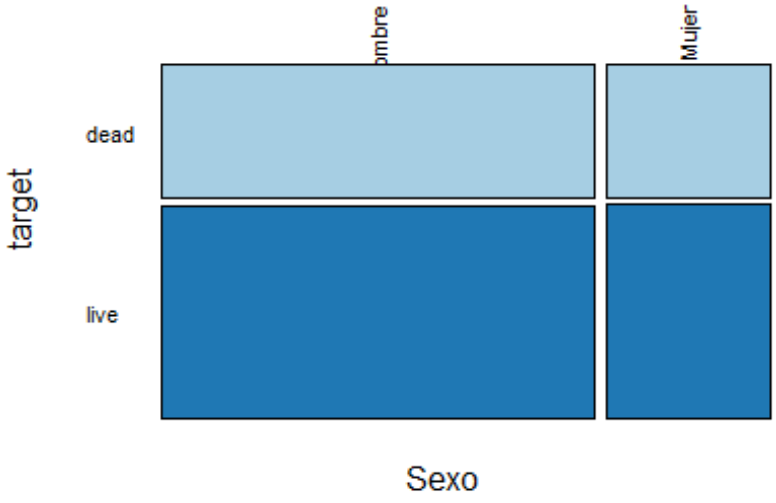
3.3.- Relación con la variable objetivo

Vemos la frecuencia y la relación con la variable objetivo de las distintas variables input:

- **Sexo**: Los hombres sufren más infartos que las mujeres

```
##           n    % val%
## Hombre 4163 72.5 72.5
## Mujer  1583 27.5 27.5
```

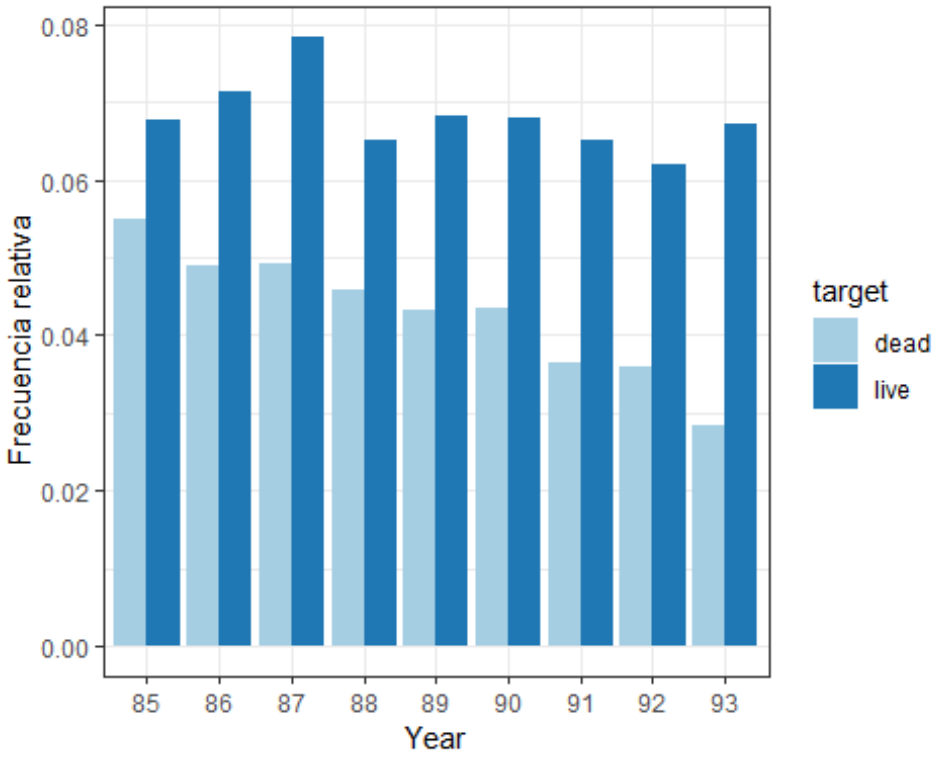


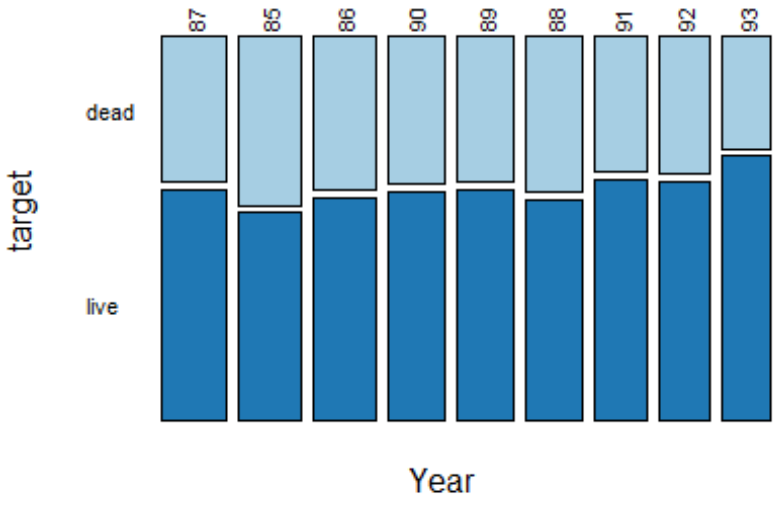


- **Year**: A partir del año 89 han disminuido los infartos y la supervivencia es mayor

```
##      n    % val%
## 85 706 12.3 12.3
## 86 691 12.0 12.0
```

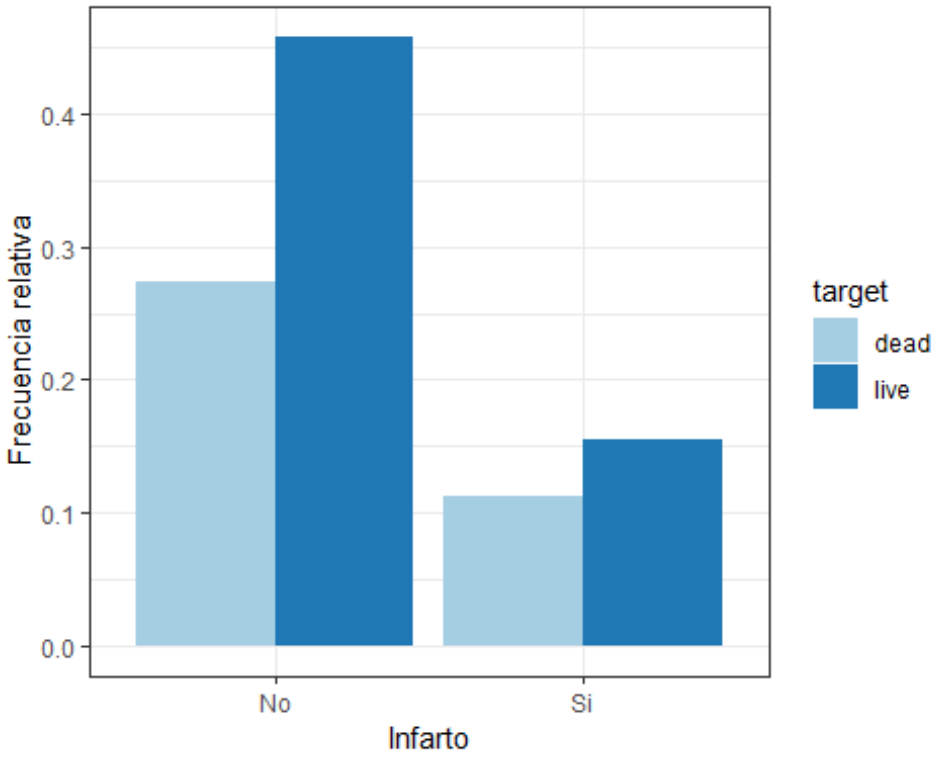
```
## 87 734 12.8 12.8
## 88 639 11.1 11.1
## 89 640 11.1 11.1
## 90 641 11.2 11.2
## 91 584 10.2 10.2
## 92 562  9.8  9.8
## 93 549  9.6  9.6
```

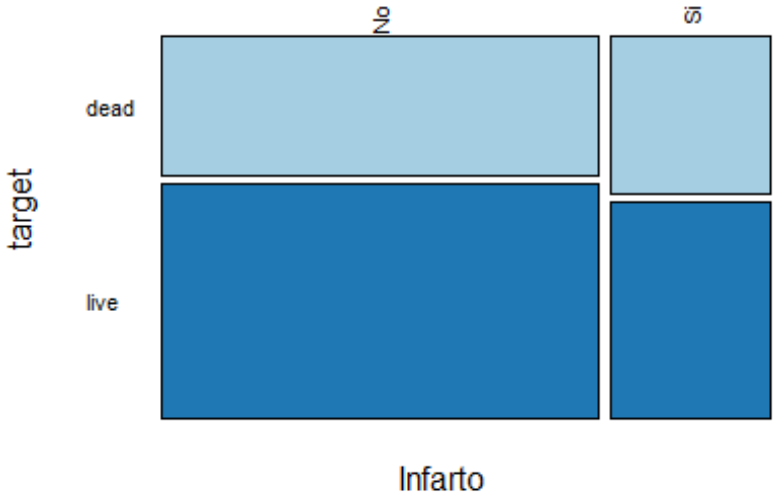




- **Infarto Previo**: En la mayoría de los casos era el primer infarto

```
##       n    % val%
## No 4204 73.2 73.2
## Si 1542 26.8 26.8
```

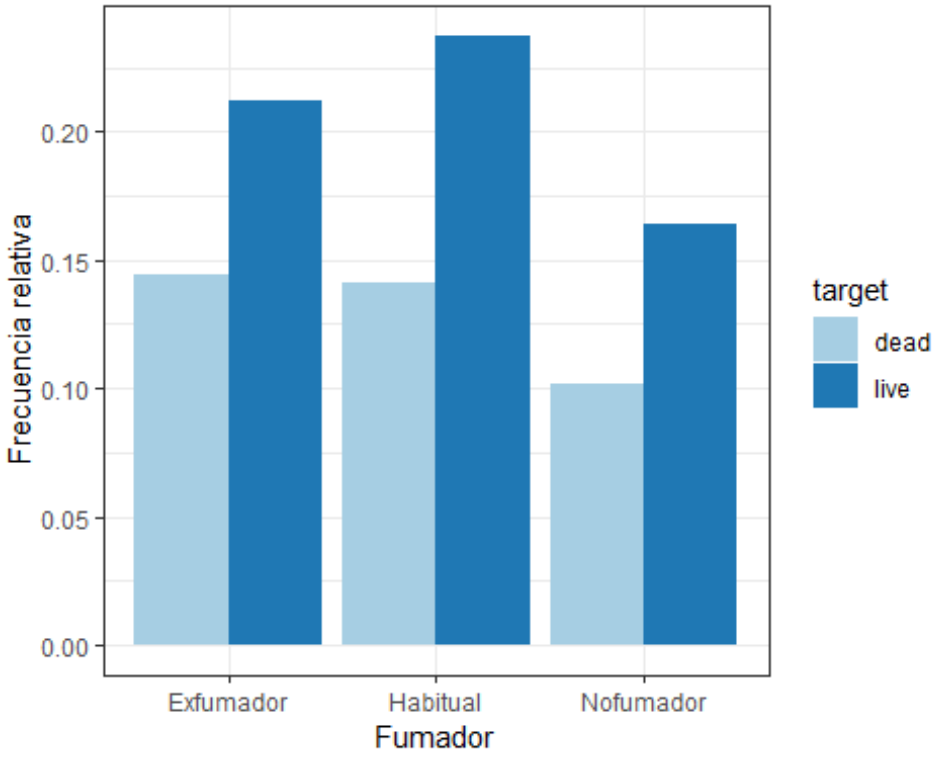


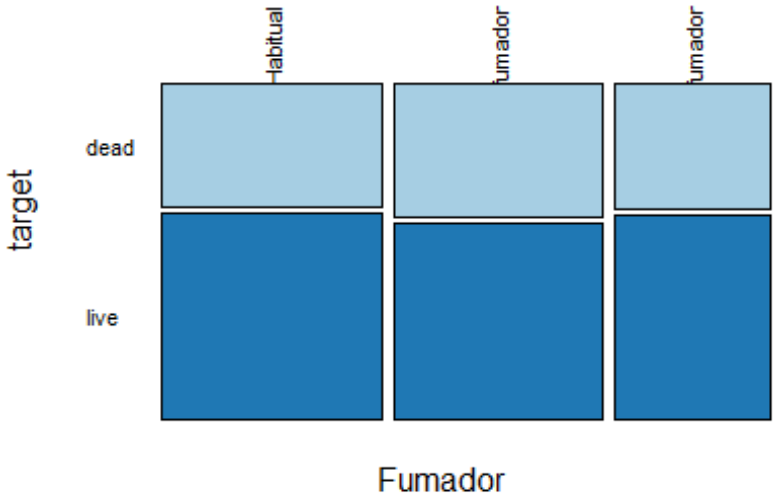


- **Status de fumador**: Los infartos son más frecuentes en fumadores y Exfumadores

```
##                n    % val%
## Exfumador 2048 35.6 35.6
```

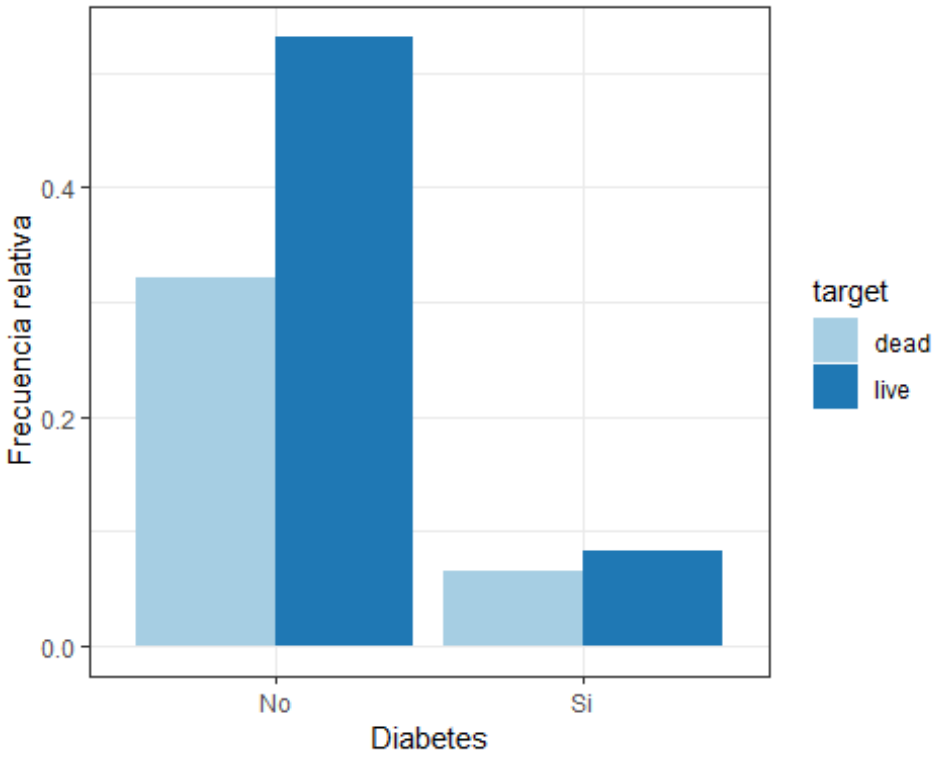
```
## Habitual  2173 37.8 37.8
## Nofumador 1525 26.5 26.5
```

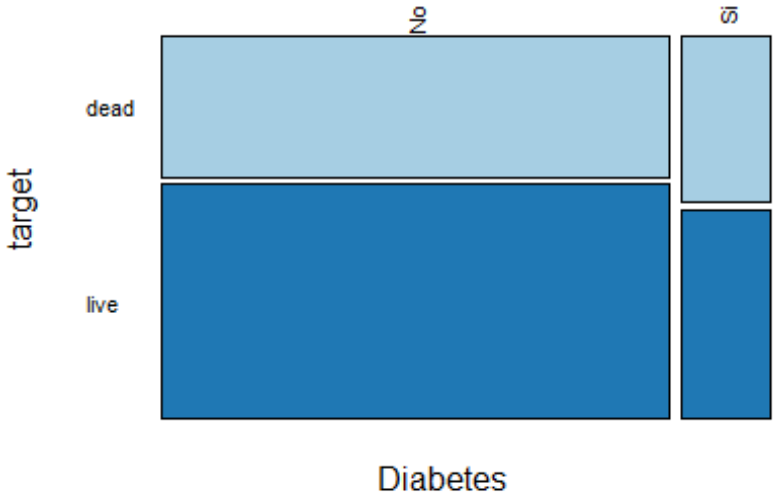




- **Diabetes**: La mayoría de las observaciones corresponden a personas no diabéticas

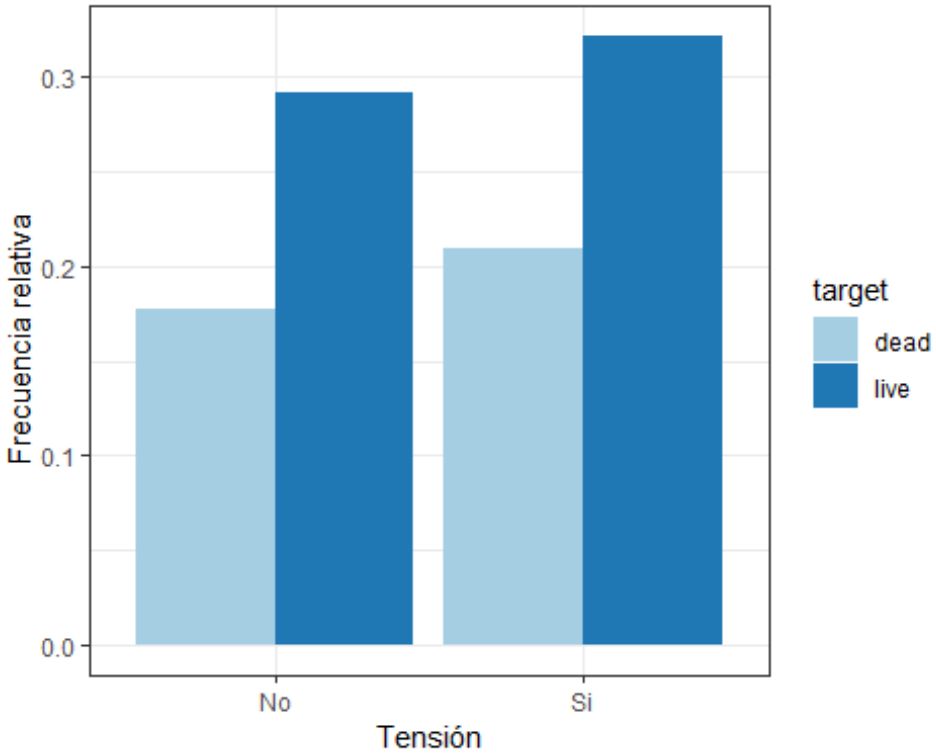
```
##       n    % val%
## No 4895 85.2 85.2
## Si  851 14.8 14.8
```

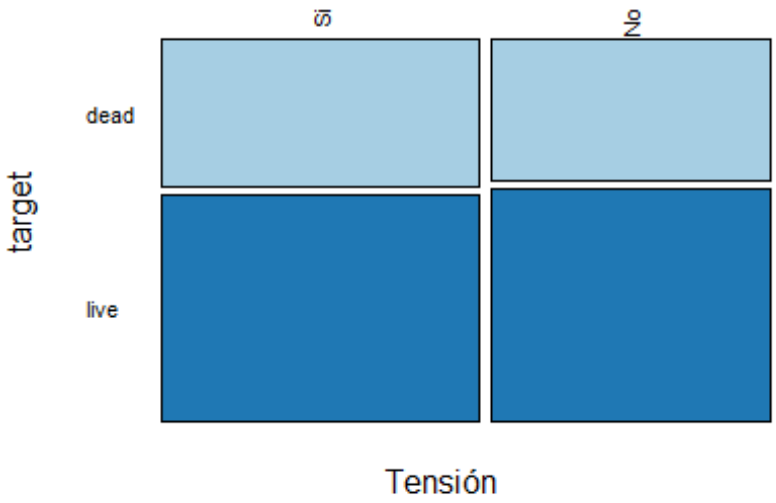




- **Tensión Alta**: La tensión alta es algo común entre las personas que sufren un infarto

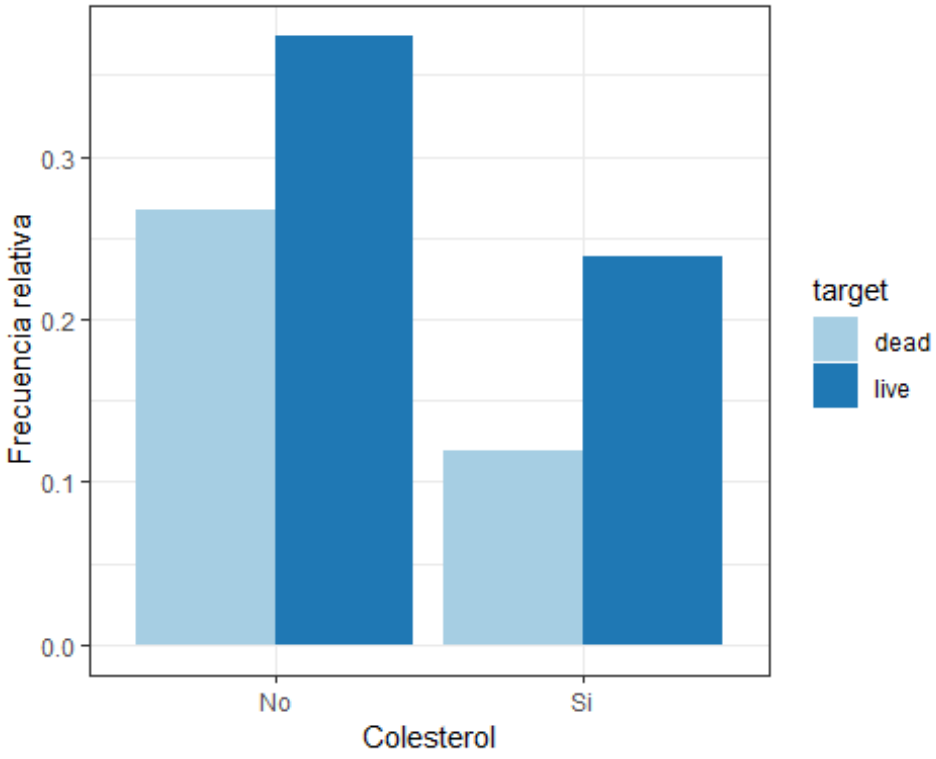
```
##       n    % val%
## No 2694 46.9 46.9
## Si 3052 53.1 53.1
```

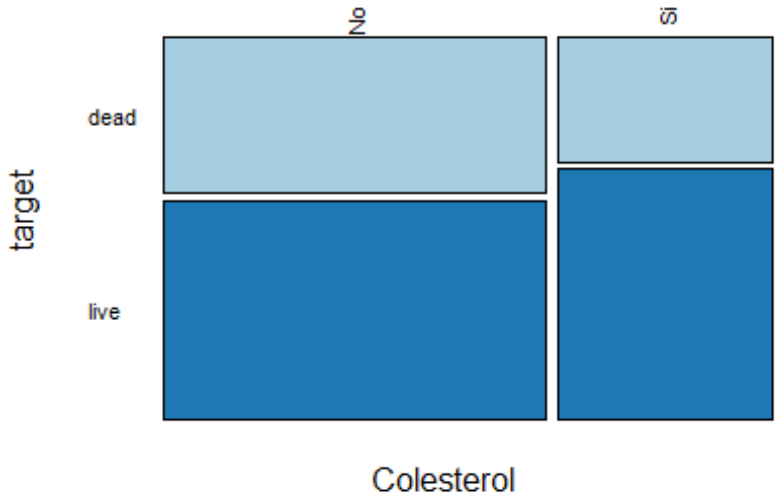




- **Colesterol Alto**: La mayoría de los pacientes no tenían colesterol alto

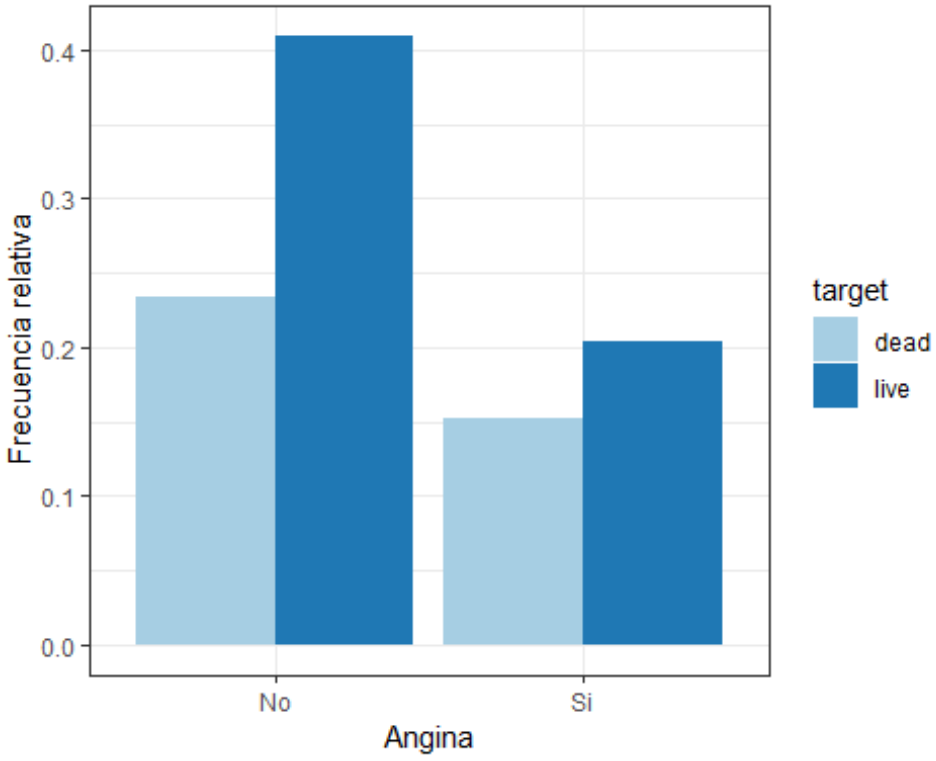
```
##       n    % val%
## No 3688 64.2 64.2
## Si 2058 35.8 35.8
```

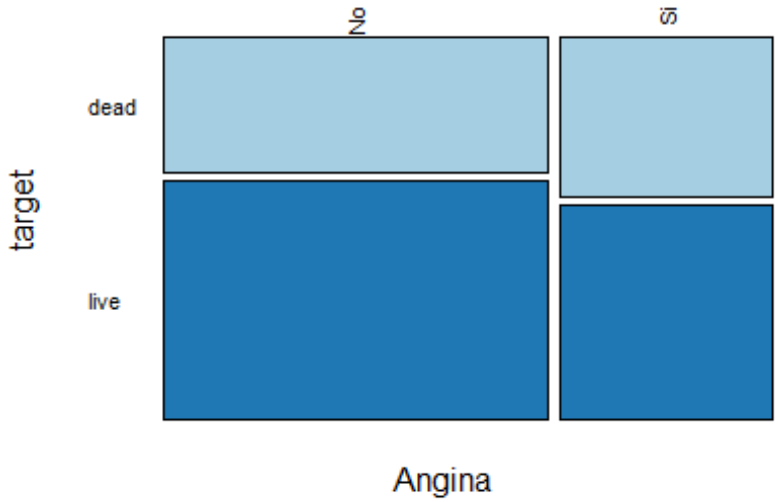




- **Angina de pecho**: Ni habían sufrido una angina de pecho

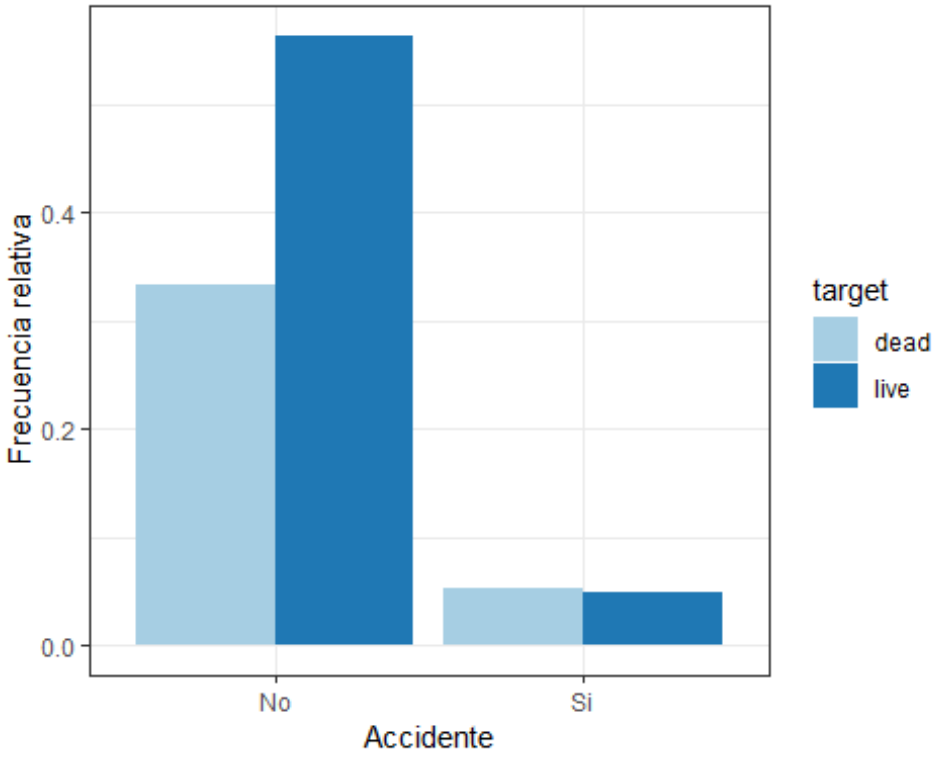
```
##       n    % val%
## No 3698 64.4 64.4
## Si 2048 35.6 35.6
```

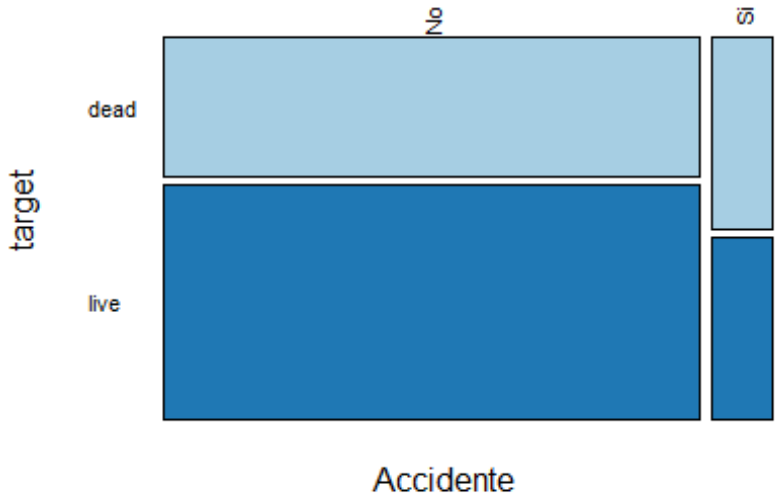




- **Accidente Cerebral**: Ni accidentes cerebrales

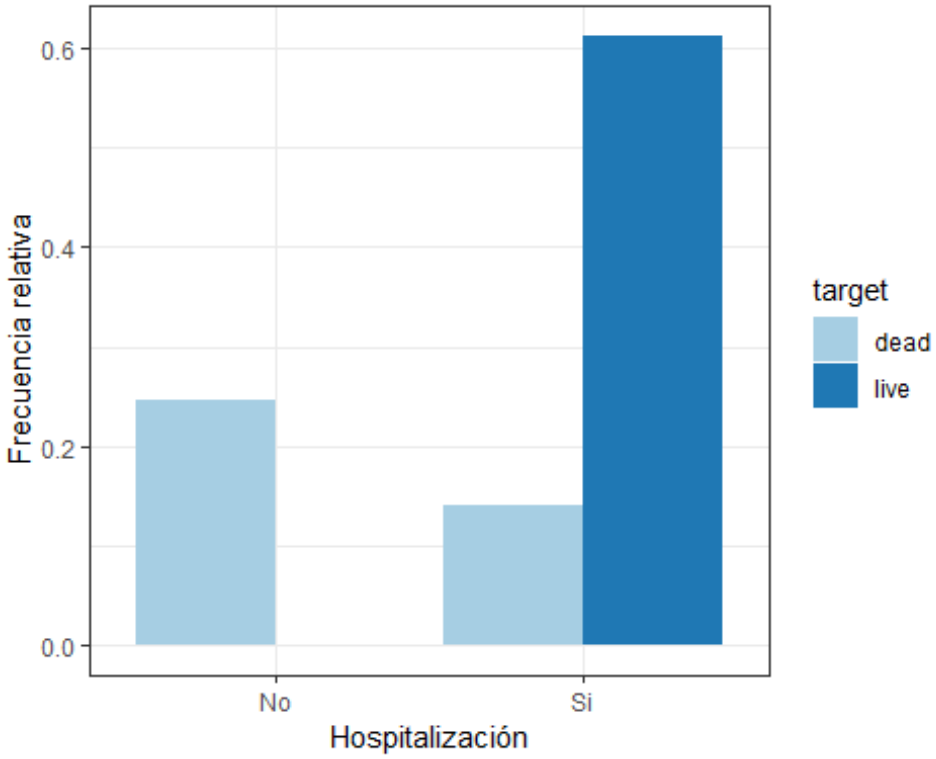
```
##       n    % val%
## No 5161 89.8 89.8
## Si  585 10.2 10.2
```

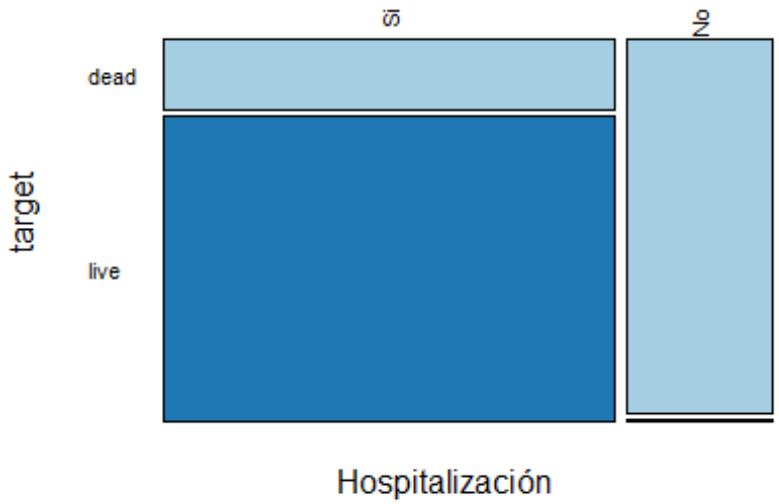




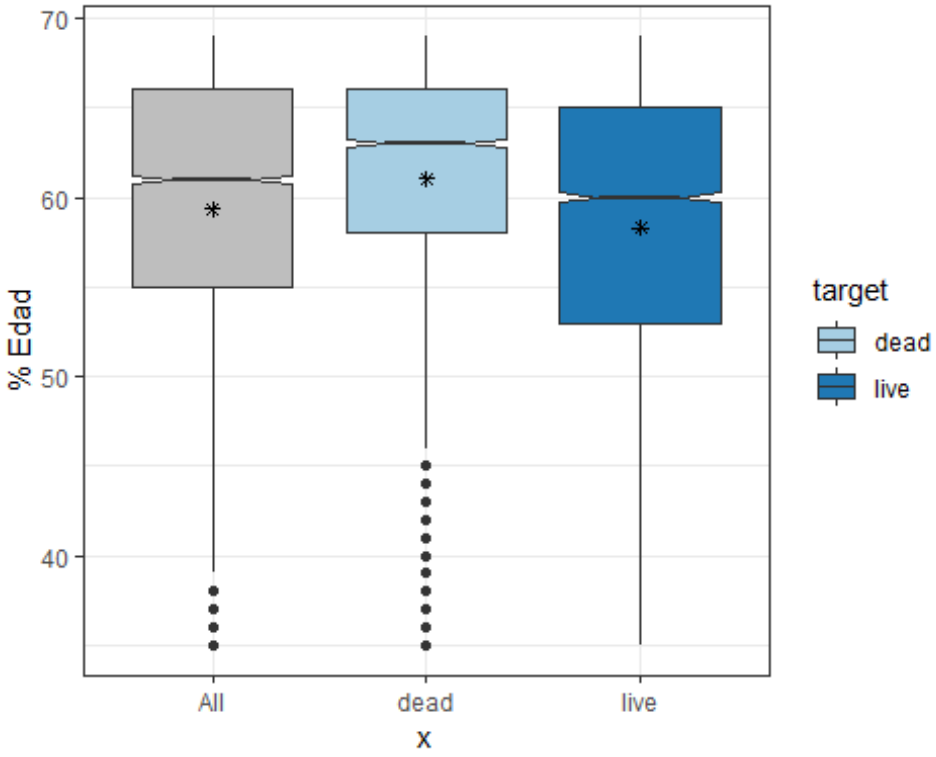
- **Hospitalización**: No se puede superar un infarto sin tratamiento hospitalario

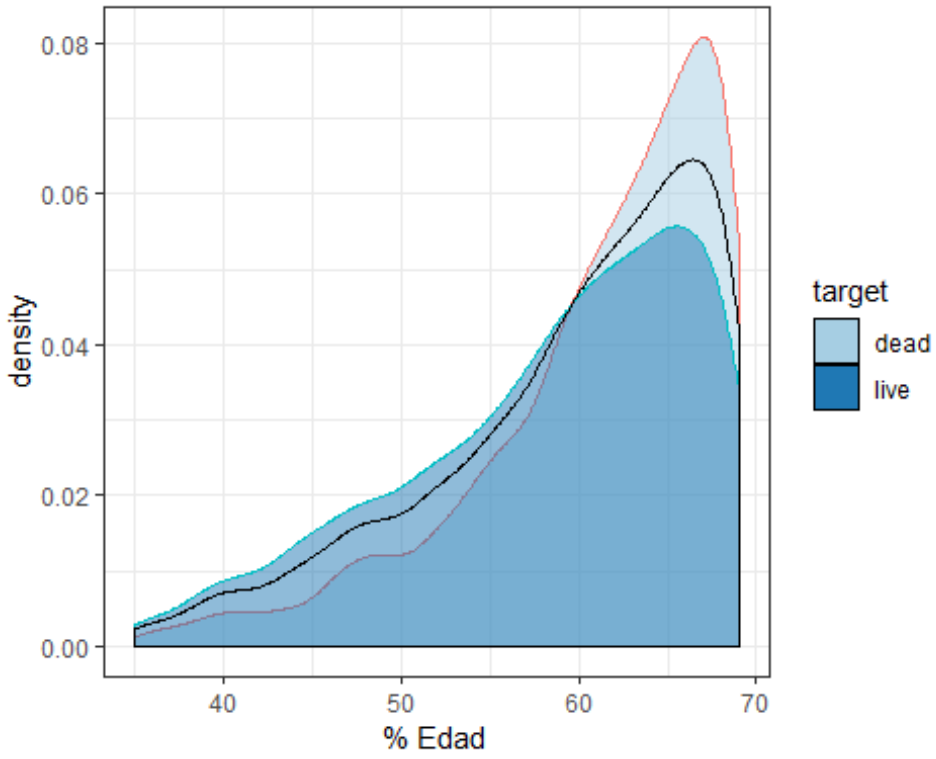
```
##       n    % val%
## No 1413 24.6 24.6
## Si 4333 75.4 75.4
```





- **Edad**: La edad tiene una efecto negativo en la supervivencia a un infarto





3.4.- Selección de variables

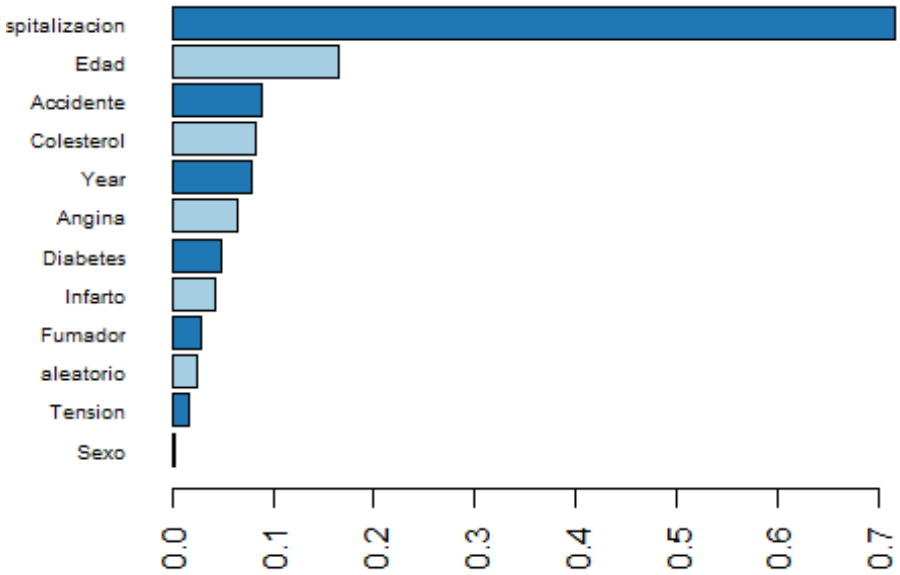
En este apartado hemos usado dos métodos de selección de variables:

- **V de Cramer**: se basa en el estadístico X^2. Lo primero que se hace es crear una variable aleatoria que nos permite tener una referencia de la utilidad de las variables.

```
# Creación de la variable aleatoria
input$aleatorio <- runif(nrow(input))
```

El gráfico que se obtiene es:

```
# Para representar la V de Cramer para todas las variables input, usamos
la función:
graficoVcramer(input,varObjBin)
```



Según este gráfico las variables que mejor explican la variable objetivo son: Hospitalización, Edad, Accidente, Colesterol, Year y Angina. Observamos que hay variables que tienen menos poder explicativo que la propia variable aleatoria, como Sexo.

- **Stepwise o paso a paso**: En cada paso se evaluan las diferentes variables a introducir y a eliminar y se selecciona cual de estas acciones mejora más el modelo. Para calcular la mejora del modelo en este proceso iterativo utilizaremos el criterio de información de Akaike (AIC) y el criterio de información Bayesiana (BIC).

Utilizaremos la funcion steprepetido binaria. Para utlizar esta función previamente hay que estandarizar las variables continuas y las categóricas pasarlas a dummies. También modificamos la variable objetivo para que esté codificada como Yes, en el caso de dead, y No para el caso de live.

```
monica$Edad <- as.numeric(monica$Edad)

listconti <- c("Edad")

listclass <- c("Sexo", "Year", "Infarto", "Fumador",
               "Diabetes", "Tension", "Colesterol",
               "Angina", "Accidente","Hospitalizacion")

vardep <- "Mortalidad"

# Transformamos las categóricas a dummies
library(dummies)
monicabis <- dummy.data.frame(monica, listclass, sep = ".")

# Estandarizar la variable continua
monicabis$Edad <- (monicabis$Edad - mean(monicabis$Edad)) /
sd(monicabis$Edad)
# Convertir la variable objetivo en Yes/No para que no fallen algoritmos
de Machine Learning
monicabis$Mortalidad<-ifelse(monicabis$Mortalidad=="dead","Yes","No")
```

Ahora utilizamos la funcion steprepetidobinaria para el criterio BIC

```
source("funcion steprepetido binaria.R")
listconti<-c("Sexo.Hombre", "Sexo.Mujer", "Edad", "Year.85",
             "Year.86", "Year.87", "Year.88", "Year.89", "Year.90",
"Year.91",
             "Year.92", "Year.93", "Infarto.No", "Infarto.Si",
             "Fumador.Exfumador", "Fumador.Habitual",
"Fumador.Nofumador",
             "Diabetes.No", "Diabetes.Si", "Tension.No", "Tension.Si",
             "Colesterol.No", "Colesterol.Si", "Angina.No", "Angina.Si",
             "Accidente.No", "Accidente.Si", "Hospitalizacion.No",
             "Hospitalizacion.Si")
vardep<-c("Mortalidad")

data<-monicabis

lista<-steprepetidobinaria(data=data,
 vardep=vardep,listconti=listconti,sinicio=12345,
 sfinal=12355,porcen=0.8,criterio="BIC")

tabla<-lista[[1]]
variables_BIC <- dput(lista[[2]][[1]])
```

```
## c("Hospitalizacion.No", "Edad", "Angina.No", "Accidente.No",
## "Year.85", "Diabetes.No")
```

Y para el cirterio AIC

```
lista<-steprepetidobinaria(data=data,
 vardep=vardep,listconti=listconti,sinicio=12345,
 sfinal=12355,porcen=0.8,criterio="AIC")

tabla<-lista[[1]]
variables_AIC <- dput(lista[[2]][[1]])

## c("Hospitalizacion.Si", "Edad", "Angina.No", "Accidente.No",
## "Year.93", "Diabetes.No", "Colesterol.No", "Year.85", "Infarto.Si",
## "Sexo.Hombre", "Fumador.Nofumador", "Year.91")
```

El criterio AIC selecciona más variables que el criterio BIC. Ahora con estos grupos de variables seleccionados por stepwise con AIC y stepwise con BIC aplicamos logística para ver con que grupos de variables se obtienen mejores resultados. También incluiremos una prueba con todas las variables.

```
variables_all <- c("Sexo.Hombre", "Sexo.Mujer", "Edad", "Year.85",
                   "Year.86", "Year.87", "Year.88", "Year.89", "Year.90",
"Year.91",
                    "Year.92", "Year.93", "Infarto.No", "Infarto.Si",
                    "Fumador.Exfumador", "Fumador.Habitual",
"Fumador.Nofumador",
                    "Diabetes.No", "Diabetes.Si", "Tension.No",
"Tension.Si",
                    "Colesterol.No", "Colesterol.Si", "Angina.No",
"Angina.Si",
                    "Accidente.No", "Accidente.Si", "Hospitalizacion.No",
                    "Hospitalizacion.Si")
```

Para ello utilizaremos la función cruzadalogistica:

```
source("cruzadas avnnet y log binaria.R") #Función para obtener
cruzadalogística

# Realizamos regresión logística para el conjunto BIC

mediasl1<-cruzadalogistica(data=data,
                           vardep=vardep,listconti=variables_BIC,
                           listclass=c(""),grupos=4,sinicio=1234,repe=20)

mediasl1$modelo="logística_BIC"

# Realizamos regresión logística para el conjunto AIC

mediasl2<-cruzadalogistica(data=data,
                           vardep=vardep,listconti=variables_AIC,
```

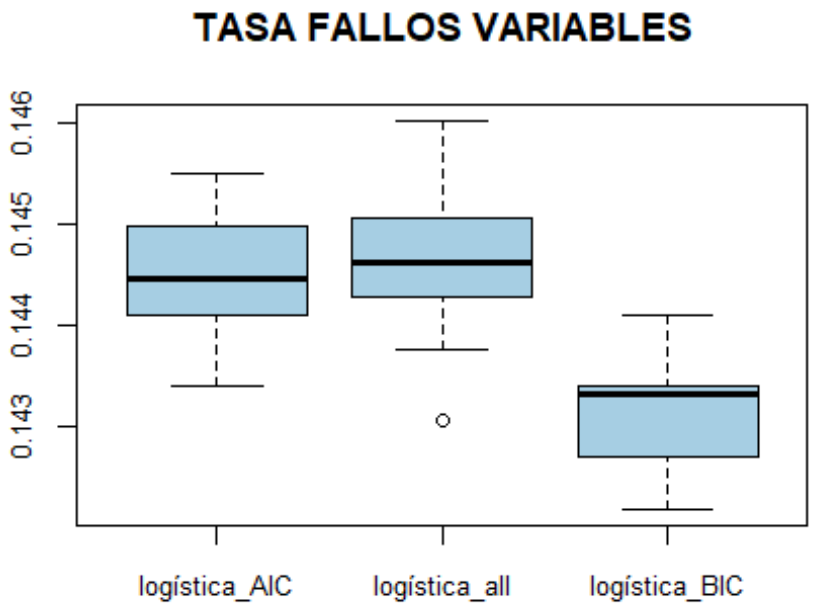
```
                              listclass=c(""),grupos=4,sinicio=1234,repe=20)

mediasl2$modelo="logística_AIC"

# Realizamos regresión logística con todas las variables

mediasl3<-cruzadalogistica(data=data,
                              vardep=vardep,listconti=variables_all,
                              listclass=c(""),grupos=4,sinicio=1234,repe=20)

mediasl3$modelo="logística_all"

union1 <- rbind(mediasl1, mediasl2,mediasl3)
```

Siendo los resultados

Modelo	Tasa media	AUC media
logistica_BIC	0.1431344	0.8795537
logistica_AIC	**0.1444831**	**0.8828342**
logistica_all	0.1446659	0.8816957

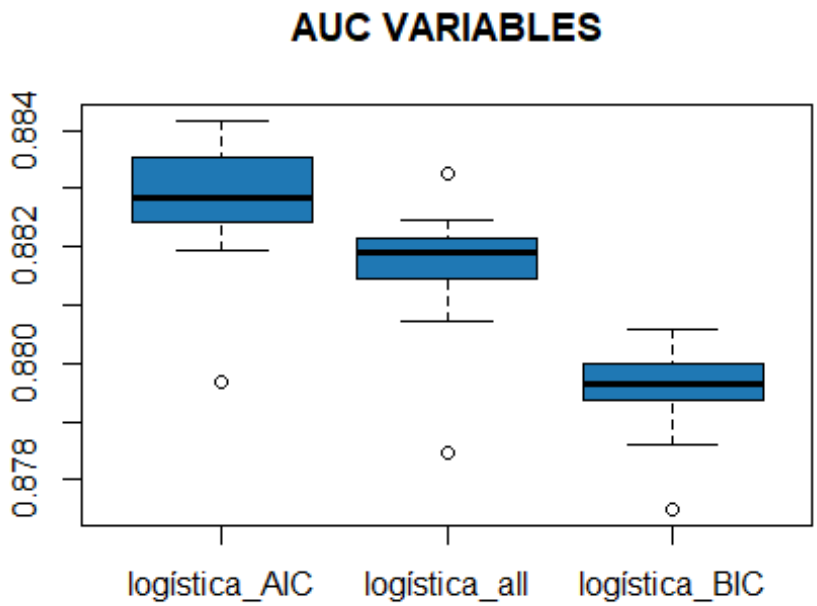
Y gráficamente

```
par(cex.axis = 0.8)
boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
VARIABLES",col=c("#A6CEE3"))
```



El modelo con menor tasa de fallos es el de las variables BIC.

```
boxplot(data = union1, auc ~ modelo, main = "AUC
VARIABLES",col=c("#1F78B4"))
```



Y respecto al AUC el modelo que obtiene mejores resultados es el de las variables AIC.

El conjunto formado por todas las variables presenta una situación intermedia entre AIC y BIC.

Nos quedaremos con el conjunto de variables de AIC que incluye variables que también resultaron importantes en la V de Cramer.

Con este conjunto de variables ejecutaremos los diferentes algoritmos de Machine Learning.

4.- ALGORITMOS DE APRENDIZAJE AUTOMATICO

Para obtener el mejor modelo de predicción de la variable Mortalidad se van a utilizar los siguientes algoritmos:

- Arboles
- Redes Neuronales
- Bagging
- Random Forest
- Gradient Boosting

- Support Vector Machines (SVM)

Utilizaremos las diferentes funciones proporcionadas por el profesor que utilizan validación cruzada repetida. Se obtiene para los diferentes modelos de cada algoritmo la tasa de fallos y el valor del AUC.

Se grafican en un boxplot y de este modo se selecciona el mejor modelo para cada uno de los algoritmos. Una vez obtenido el mejor modelo se comparará con el mejor modelo de regresión logística. Así podemos ver si alguno de estos algoritmos supera a la logística.

En las siguientes secciones se hace un resumen de las características de los modelos utilizados. No sé ha considerado importante mostrar el código que se podrá consultar en el ANEXO.

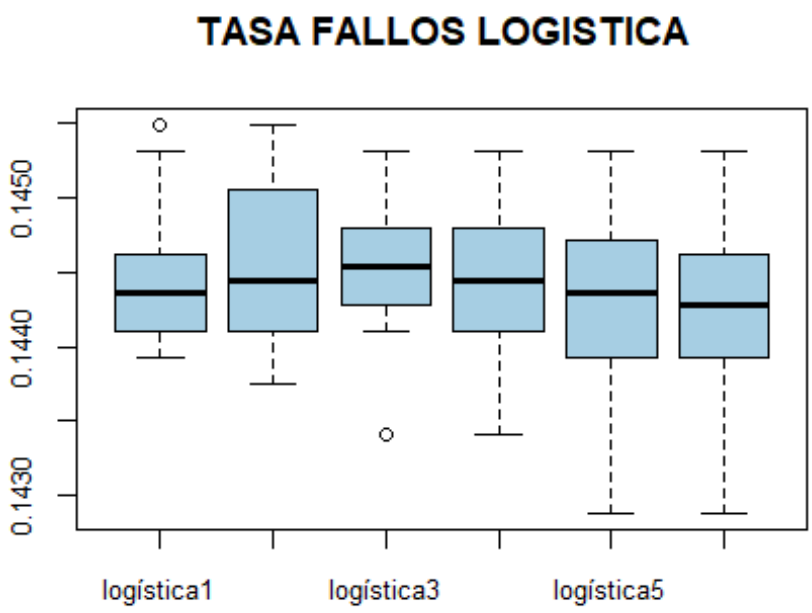
4.1.- Regresión logística

En primer lugar buscamos el mejor modelo de regresión logística para después poder compararlo con los modelos obtenidos con los distintos algoritmos.

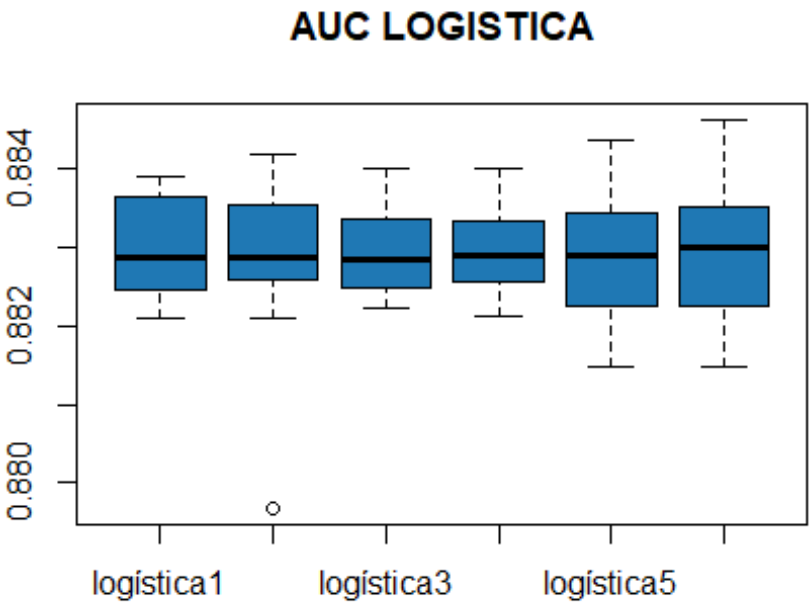
Se indican a continuación las características de los modelos utilizados.

Modelo	Tasa media	AUC media	Semilla	Repeticiones
logistica1	0.1444831	0.8829462	1234	10
logistica2	0.1445643	0.8828273	1234	15
logistica3	0.1445179	0.8829745	5678	10
logistica4	0.1444483	0.8829803	5678	15
logistica5	0.1442656	0.8828368	34567	20
logistica6	**0.1442743**	**0.8829448**	**34567**	**30**

Gráficamente vemos que el modelo logistica6 presenta menor tasa de fallos



Y también es el modelo con mayor AUC por tanto será el seleccionado para la comparación con el resto de algoritmos



4.2.- Árboles

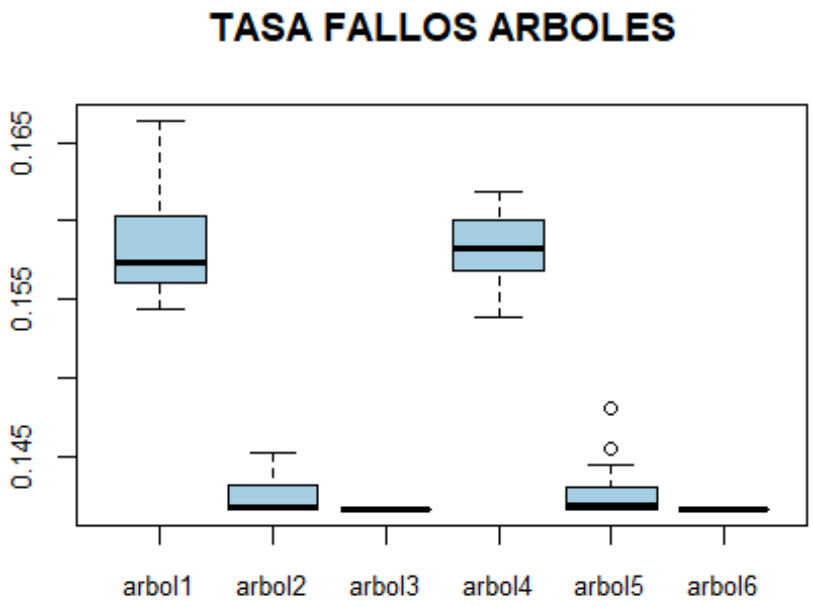
Los parámetros a tunear de los árboles son:

- minbucket: número de observaciones máximas en el nodo final
- cp: parámetro de corte de penalización por complejidad (cp = 0, máxima complejidad)

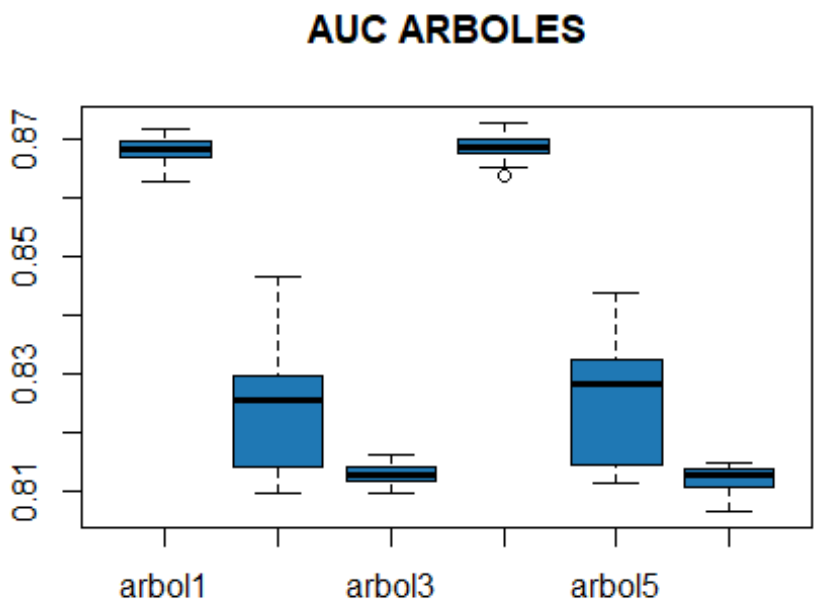
Los modelos que se han probado han sido los siguientes:

Modelo	Tasa media	AUC media	Semilla	Repeticiones	cp	minbucket
arbol1	0.1584813	0.8681750	1234	30	0	20
arbol2	0.1423889	0.8235285	1234	30	0.001	20
arbol3	0.1416638	0.8129940	1234	30	0.005	30
arbol4	**0.1582579**	**0.8687869**	**5678**	**20**	**0**	**20**
arbol5	0.1427341	0.8266223	5678	20	0.001	20
arbol6	0.1416638	0.8122747	5678	20	0.005	30

Gráficamente vemos que los modelos arbol1 y arbol4 presenta tasa de fallos parecidas



Y también el AUC es similar.



Elegimos el modelo arbol4 por tener ser un poco mayor AUC.
Se observa que estos modelos tienen peores resultados que la regresión logística.

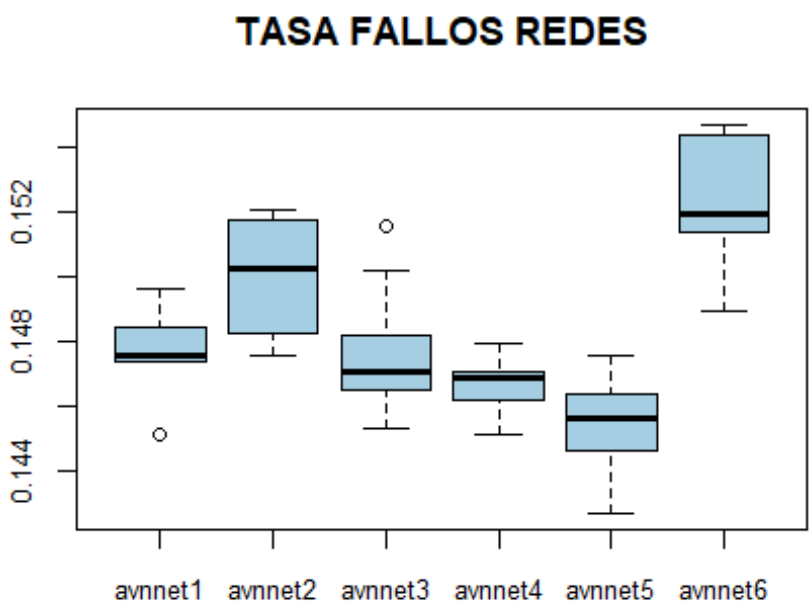
4.3.- Redes Neuronales

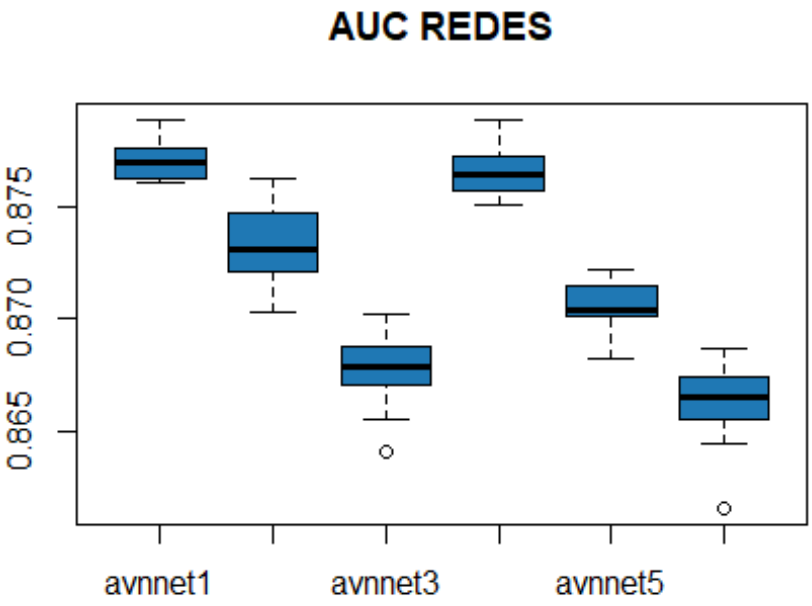
Algunos de los parámetros a tunear en las redes son:

- Size: número de nodos en la capa oculta
- Decay: velocidad de decrecimiento de los pesos

Los modelos que se han probado han sido los siguientes:

Modelo	Tasa media	AUC media	Semilla	Nodos	Decay	Repe	Iteraciones
avnnet1	0.1476505	0.8771107	1234	10	0.01	5	100
avnnet2	0.1500348	0.8732446	1234	15	0.01	10	100
avnnet3	0.1475200	0.8678961	1234	20	0.001	20	120
avnnet4	**0.1466411**	**0.8766292**	**5678**	**10**	**0.01**	**5**	**100**
avnnet5	0.1453881	0.8705316	5678	15	0.001	10	150
avnnet6	0.1523495	0.8681750	5678	20	0.001	10	200





El modelo avnnet4 es el elegido en este caso debido a que tiene alto AUC y su tasa de fallos es baja.

Se observa que estos modelos tienen mejores resultados que los árboles pero siguen siendo peores resultados que la regresión logística.

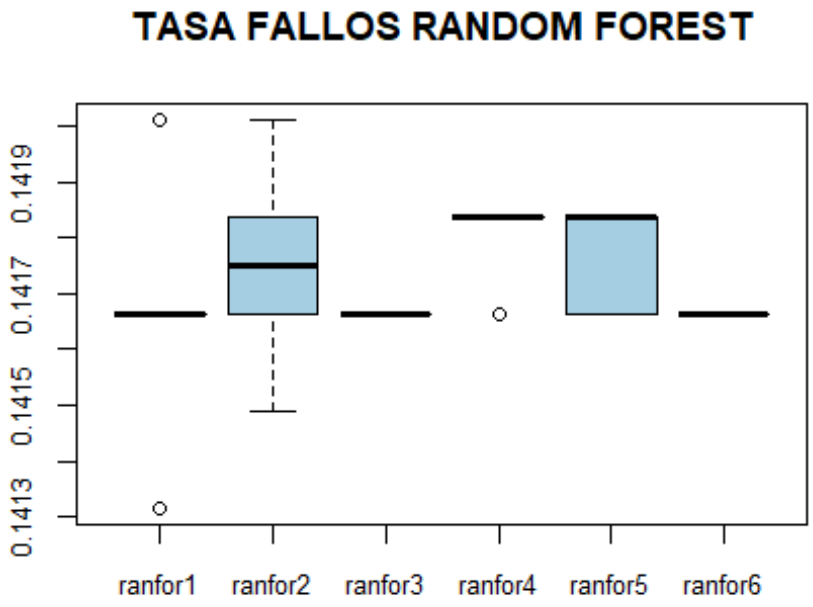
4.4.- Random Forest

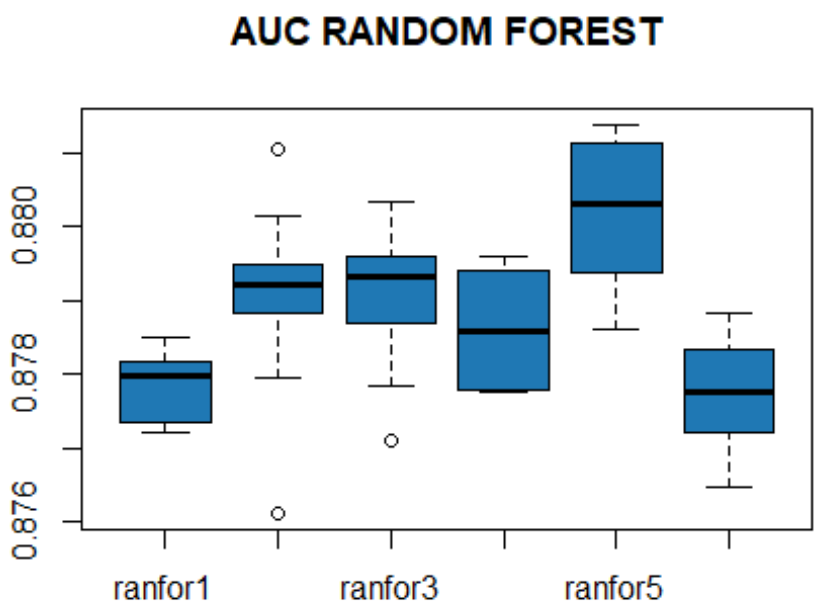
Algunos de los parámetros a tunear son:

- mtry: es número de variables
- nodesize: tamaño máximo de nodos totales (mide la complejidad)
- ntree: número de árboles o iteraciones
- sampsize: el tamaño de cada muestra
- replace: indica si hay reemplazamiento (TRUE) o no (FALSE)

Los modelos que se han probado han sido los siguientes:

Modelo	Tasa media	AUC media	Semilla	mtry	Nodos	Arboles	Repe	Sampsize
ranfor1	0.1416638	0.8778336	1234	8	10	300	5	50
ranfor2	0.1417508	0.8790164	1234	10	15	500	10	100
ranfor3	0.1416638	0.8790931	1234	10	20	1000	15	150
ranfor4	0.1418030	0.8786248	5678	8	10	1500	5	50
ranfor5	**0.1417682**	**0.8801358**	**5678**	**10**	**15**	**1500**	**10**	**150**
ranfor6	0.1416638	0.8777418	5678	10	20	3500	20	100





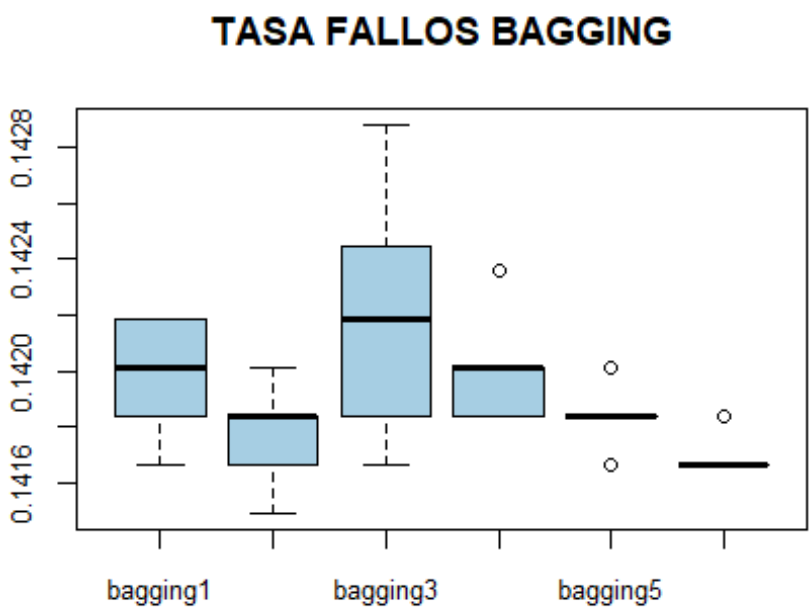
Elegimos el modelo ranfor5 que es el de mayor AUC.

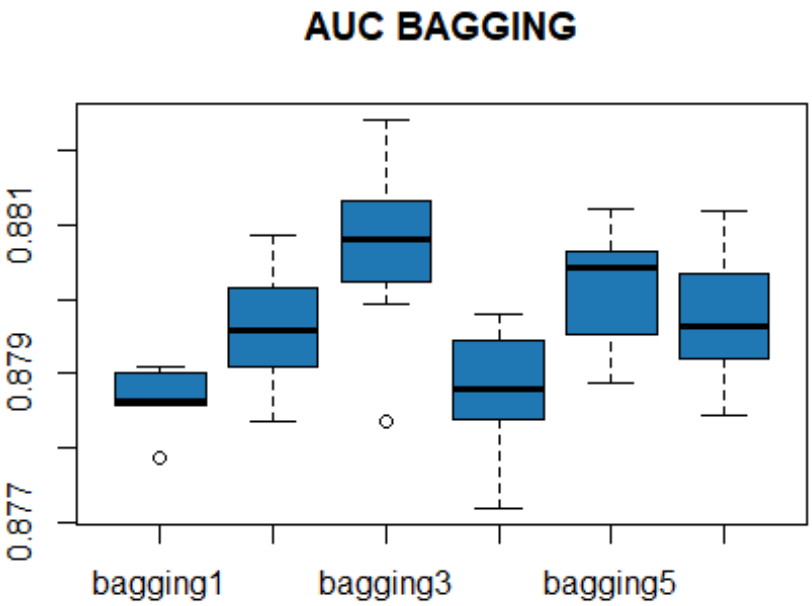
4.5.- Bagging

Bagging es un caso particular de Random Forest. Los parámetros a tunear serán los mismos. La diferencia con Random Forest es que mtry debe coincidir con el número de variables input, 12 en nuestro caso.

Los modelos que se han probado han sido los siguientes:

Modelo	Tasa media	AUC media	Semilla	Nodos	Arboles	Repe	Sampsize
bagging1	0.1419770	0.8786311	1234	10	2000	5	50
bagging2	0.1417682	0.8795765	1234	15	2500	10	100
bagging3	0.1421946	0.8807482	1234	10	2500	20	150
bagging4	0.1420118	0.8787219	5678	10	3000	5	50
bagging5	**0.1418552**	**0.8802490**	**5678**	**15**	**4000**	**10**	**150**
bagging6	0.1416812	0.8797063	5678	20	3500	20	200





Elegimos el modelo bagging5 que si bien no es el de mayor AUC tampoco es el que menos y además tiene baja tasa de fallos.

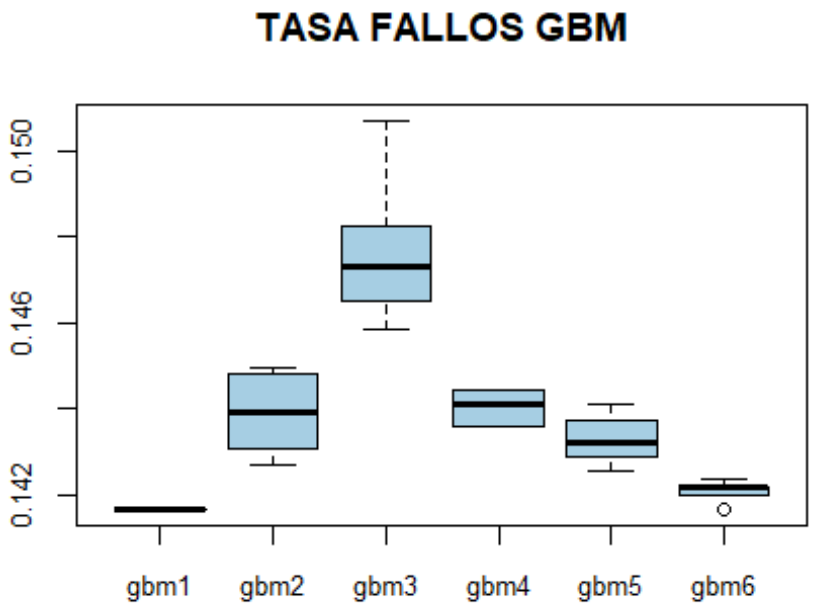
4.6.- Gradient Boosting

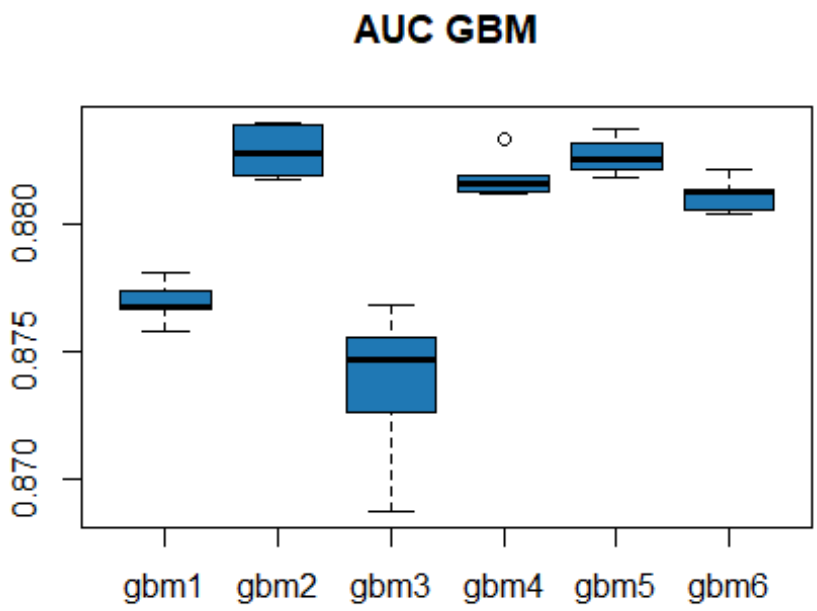
Algunos de los parámetros a tunear son:

- shrinkage: parámetro v de regularización (entre 0.001 y 0.2)
- n.minobsnode: tamaño máximo de nodos finales
- n.trees: el número de iteraciones (árboles)
- interaction.depth (2 para árboles binarios)

Los modelos que se han probado han sido los siguientes:

Modelo	Tasa media	AUC media	Semilla	Nodos	Arboles	Repe	Shrinkage
gbm1	0.1416638	0.8769332	1234	10	2000	5	0.001
gbm2	0.1438914	0.8828756	1234	10	1500	5	0.01
gbm3	0.1475635	0.8739846	1234	15	2000	10	0.1
gbm4	0.1440306	0.8818624	5678	10	2000	5	0.01
gbm5	**0.1432649**	**0.8826606**	**5678**	**15**	**1500**	**10**	**0.005**
gbm6	0.1420814	0.8811630	5678	10	1000	5	0.005





Elegimos el modelo gbm5 que es el de mayor AUC y uno de los de menor tasa de fallos.

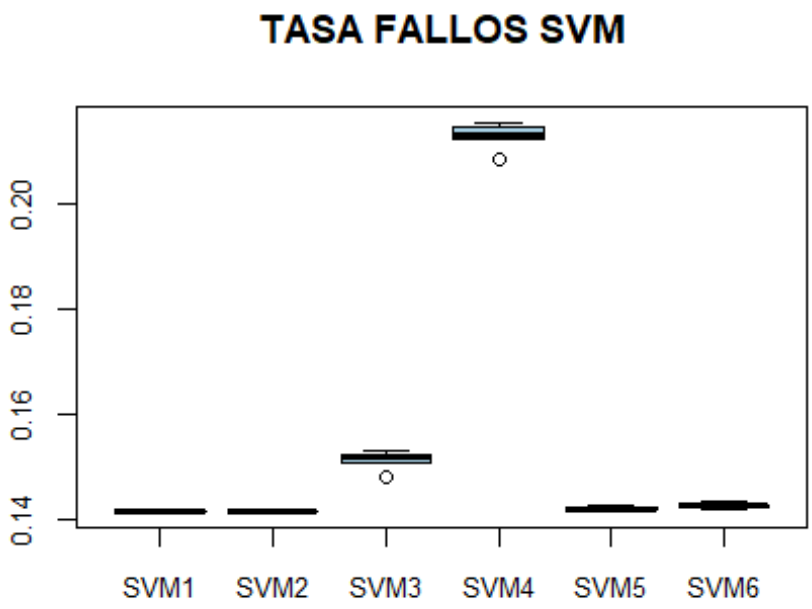
4.7.- Support Vector Machines (SVM)

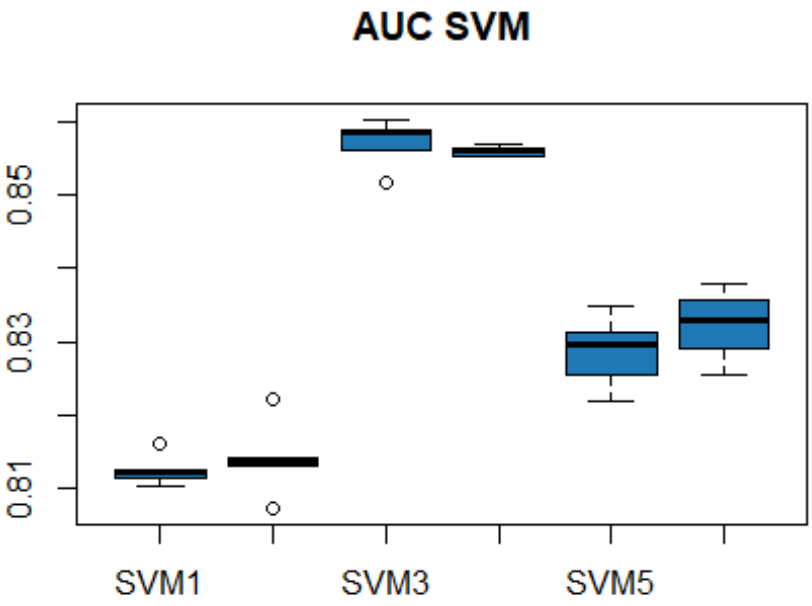
Para crear los modelos de este algoritmo he utilizado tres funciones Kernel diferentes:

- Lineal
- Polinomial
- RBF

Los modelos que se han probado han sido los siguientes:

Modelo	Tasa media	AUC media	Semilla	Funcion	Repe	c
SVM1	0.1416638	0.8125783	1234	Lineal	5	0.05
SVM2	0.1416638	0.8140171	1234	Lineal	5	0.10
SVM3	**0.1513401**	**0.8571159**	**1234**	**RBF**	**5**	**0.10**
SVM4	0.2128089	0.8559973	1234	RBF	5	0.001
SVM5	0.1422555	0.8286087	1234	Polinomial	10	0.01
SVM6	0.1427341	0.8322083	1234	Polinomial	20	0.001





Elegimos el modelo SVM3 que es el de mayor AUC y menor tasa de fallos.

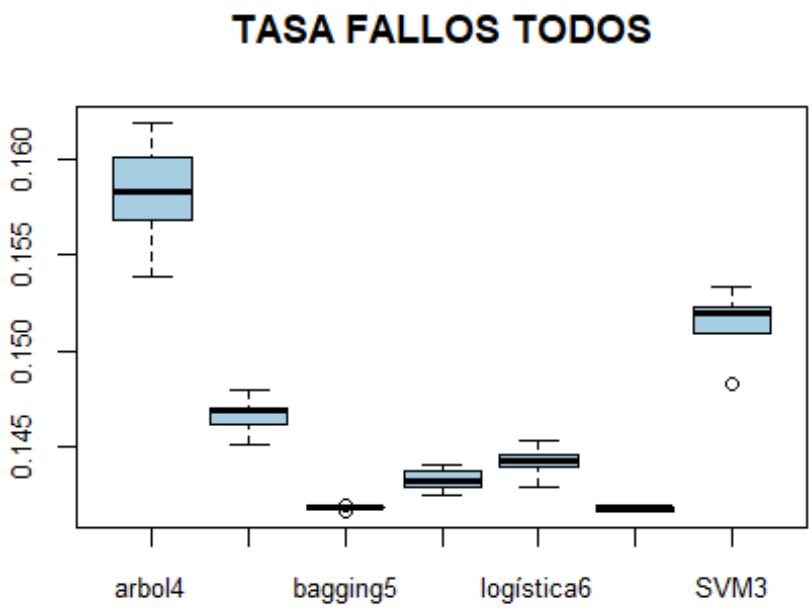
Se han obtenido un total de 42 modelos.

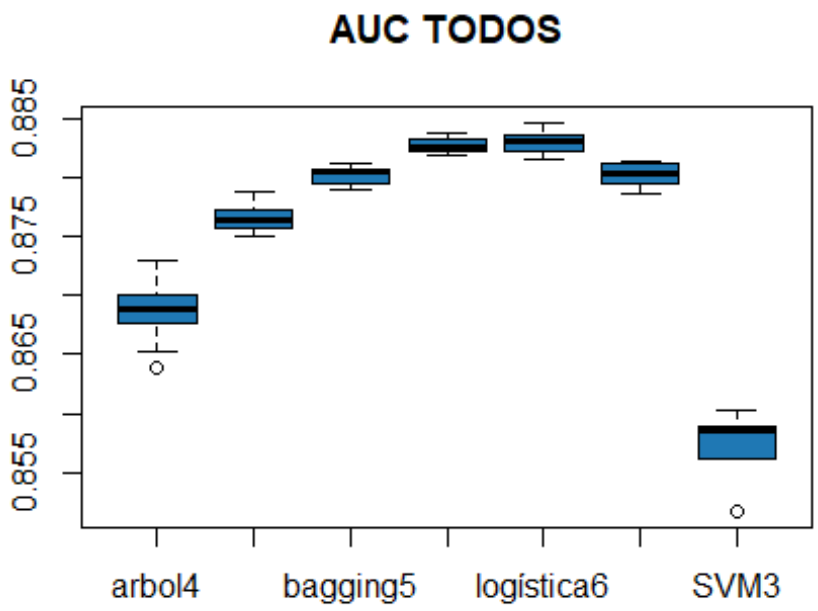
5.- COMPARACIÓN MEJORES MODELOS FRENTE A LOGÍSTICA

En este apartado compararemos gráficamente el mejor modelo de logística(logistica6) con cada uno de los mejores modelos resultantes de los diferentes algoritmos.

La tasa de fallos media y el AUC de los modelos ganadores de cada algoritmo es:

Modelo	Tasa media	AUC media
logistica6	0.1442743	0.8829448
arbol4	0.1582579	0.8687869
avnnet4	0.1466411	0.8766292
ranfor5	0.1417682	0.8801358
bagging5	0.1418552	0.8802490
gbm5	0.1432649	0.8826606
SVM3	0.1513401	0.8571159





Observando los gráficos obtenidos gbm5 presenta valores parecidos a logistica6 tanto en AUC como en tasa de fallos.

Por lo que no parece que logistica6 sea el claro ganador.

6.- ENSEMBLE DE MODELOS

Con la finalidad de mejorar la precisión de los modelos anteriores vamos a utilizar los métodos Ensemble construyendo predicciones combinando varios modelos(stacking).

Utilizaremos la función "cruzadas ensamblado binaria fuente.R" proporcionada por el profesor. Dentro de esta función no se tratan los árboles de decisión.

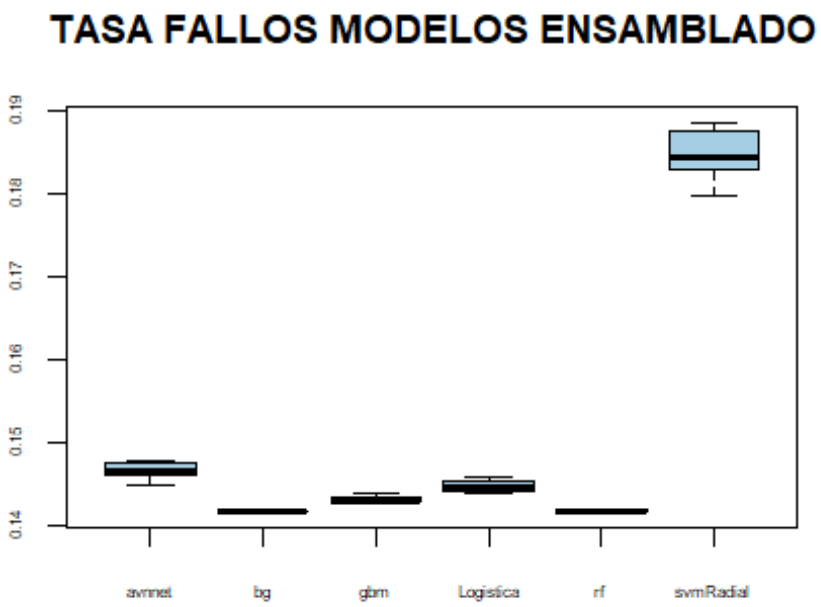
Para comparar los diferentes modelos lo primero que hacemos es fijar los parámetros comunes a todos:

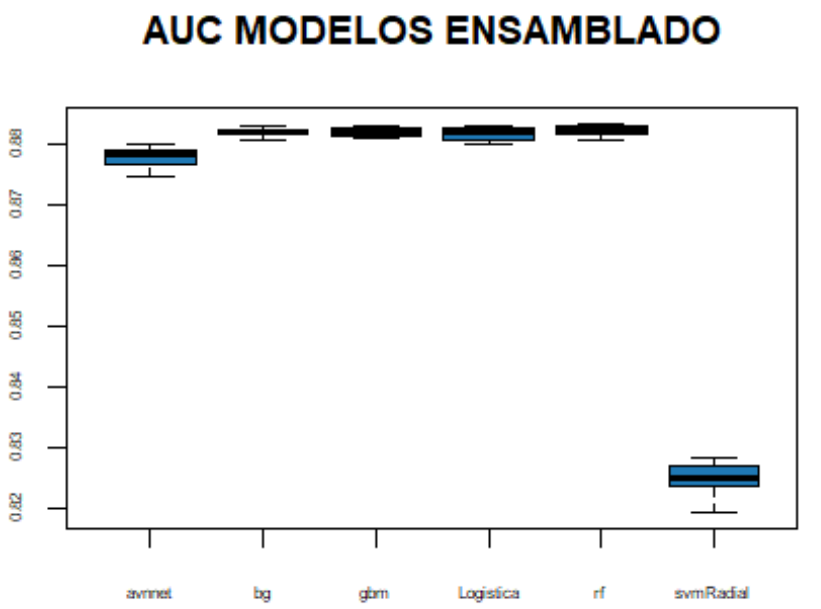
- grupos : 4
- semilla de inicio : 5678
- 10 repeticiones

También se indica cual es el dataset utilizado, la variable objetivo y el resto de variables.

Ahora para los diferentes modelos ganadores del apartado anterior (menos árboles) hacemos validación cruzada repetida y obtenemos un vector con las predicciones de cada algoritmo.

Si comparamos los modelos generados para el ensamblado:



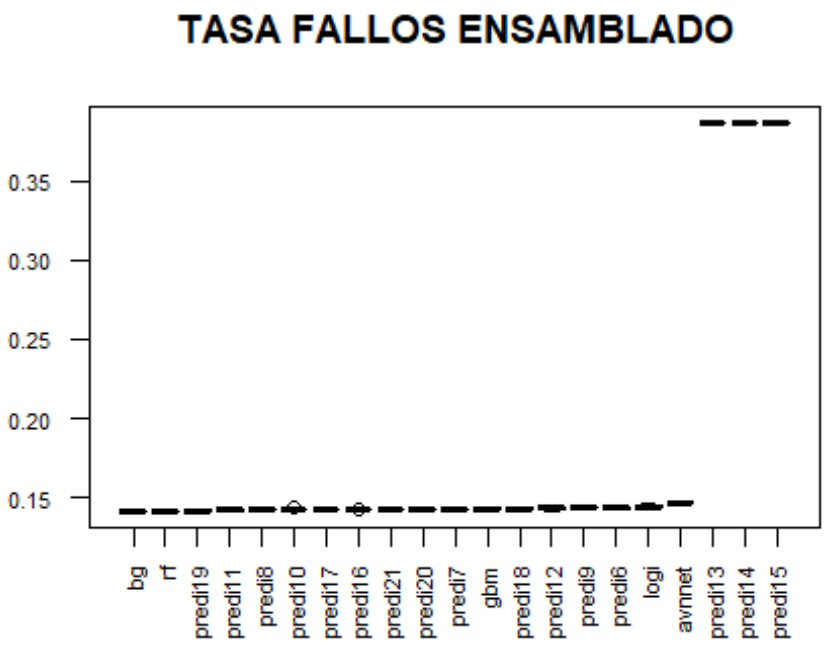


Eliminamos svm para el ensamblado porque no tiene buenos resultados.

Indicamos en la siguiente tabla los ensamblados y el porcentaje hay de cada modelo:

Ensamblado	Logistica(%)	Redes(%)	Random Forest(%)	Bagging(%)	GBM(%)
1	100	0	0	0	0
2	0	100	0	0	0
3	**0**	**0**	**100**	**0**	**0**
4	**0**	**0**	**0**	**100**	**0**
5	0	0	0	0	100
6	50	50	0	0	0
7	50	0	50	0	0
8	50	0	0	50	0
9	50	0	0	0	50
10	0	50	50	0	0
11	0	50	0	50	0
12	0	50	0	0	50
13	0	0	50	50	0
14	0	0	50	0	50
15	0	0	0	50	50
16	33	33	33	0	0
17	33	33	0	33	0
18	33	33	0	0	33
19	**33**	**0**	**33**	**33**	**0**
20	33	0	33	0	33
21	33	0	0	33	33

Y en el gráfico observamos que los ensamblados que presentan menor tasa de fallos son Random Forest, Bagging y la predicción 19:



7.- ANEXO

Se indica en este apartado el código en R utilizado para la realización de esta tarea

```
#************************************************************
# MACHINE LEARNING
# Proyecto MONICA
# Autor: Beatriz González García
#************************************************************

# Librerías y funciones a utilizar

source("FuncionesML.R")
library(readxl)
library(questionr)
library(psych)
library(car)
library(corrplot)
library(dplyr)
library(frequency)
library(e1071)
library(caret)
library(ggplot2)
library(plotly)
library(lmSupport)
library(pivottabler)
```

```r
library(pROC)
library("RColorBrewer")
library(dfexplore)
library(dummies)


#-----------------------------------------------------------*
#              DEPURACION DE DATOS
#
#      Recodificación de variables
#      Tratamiento de missings
#      Outliers(valores fuera de rango)
#-----------------------------------------------------------*

# Leemos el fichero csv
  monica <- read.csv("monica.csv",header=TRUE,sep = ",")

# Analisis exploratorio
  dfplot(monica)

# Cambio el nombre de las variables
  names(monica)
  names (monica) = c("X", "Mortalidad", "Sexo", "Edad", "Year",
                     "Infarto","Fumador", "Diabetes",
                     "Tension","Colesterol","Angina",
                     "Accidente","Hospitalizacion")
  names (monica)

# Un resumen de los datos
  summary(monica)

# La estrutura del archivo con las clases de las variables
#   6367 obs. of  13 variables:
  str(monica)

# Cuento el número de valores diferentes para las numéricas
  sapply(Filter(is.numeric, monica),function(x) length(unique(x)))

# Convertimos en factor la variable Year que sólo tiene nueve categorías
  monica$Year <- factor(monica$Year)
  str(monica)

# Recodifico los valores de algunas variables para que sean mas claros

          monica$Sexo <-
car::recode(monica$Sexo,"c('f')='Mujer';c('m')='Hombre'")
        monica$Infarto <-
car::recode(monica$Infarto,"c('y')='Si';c('n')='No'")
        monica$Fumador <-
```

```r
car::recode(monica$Fumador,"c('c')='Habitual';c('x')='Exfumador';c('n')='
Nofumador'")
       monica$Diabetes <-
car::recode(monica$Diabetes,"c('y')='Si';c('n')='No'")
        monica$Tension <-
car::recode(monica$Tension,"c('y')='Si';c('n')='No'")
     monica$Colesterol <-
car::recode(monica$Colesterol,"c('y')='Si';c('n')='No'")
         monica$Angina <-
car::recode(monica$Angina,"c('y')='Si';c('n')='No'")
      monica$Accidente <-
car::recode(monica$Accidente,"c('y')='Si';c('n')='No'")
monica$Hospitalizacion <-
car::recode(monica$Hospitalizacion,"c('y')='Si';c('n')='No'")


# MISSINGS
# --------

# Recodifico los valores nk (not know) como NA con la funcion recode.na
del paquete questionr
   monica$Infarto <- questionr::recode.na(monica$Infarto,"nk")
   monica$Fumador <- questionr::recode.na(monica$Fumador,"nk")
  monica$Diabetes <- questionr::recode.na(monica$Diabetes,"nk")
   monica$Tension <- questionr::recode.na(monica$Tension,"nk")
monica$Colesterol <- questionr::recode.na(monica$Colesterol,"nk")
    monica$Angina <- questionr::recode.na(monica$Angina,"nk")
 monica$Accidente <- questionr::recode.na(monica$Accidente,"nk")


# Busqueda de la existencia de algún patrón en los missings, que pueda
ayudar a entenderlos

corrplot(cor(is.na(monica[colnames(monica)[colSums(is.na(monica))>0]])),m
ethod = "ellipse",type = "upper")

# Incluimos para cada fila su proporción de missings
  monica$prop_missings <- apply(is.na(monica),1,mean)

# Vemos la proporcion de missings por variable
  (prop_missingsVars <- apply(is.na(monica),2,mean))# no hay variable con
mas del 50 % de missings


# Seleccionamos las observaciones que tienen menos del 50 % de missings
  input <- subset(monica, prop_missings< 0.5)#Se han eliminado 621
observaciones
  summary(input)
  row.names(monica) <- monica$X
```

```r
# Aplicamos a todas las variables factor la transformación de los NA a un valor aleatorio
  input[,as.vector(which(sapply(input, class) == "factor"))] <- sapply(Filter(is.factor, input),function(x) ImputacionCuali(x,"aleatorio"))

# como esta funciona cambia los factores a character los volvemos a cambiar a factor
  input[,as.vector(which(sapply(input, class) == "character"))] <- lapply(input[,as.vector(which(sapply(input, class)=="character"))] , factor)

# Comprobamos que no quedan observaciones con valores missings
  summary(input)


# Quitamos el identificador(X) y la proporcíon de missings(14) y guardamos los datos depurados
  Monica_Dep <- as.data.frame(input[,-c(1,14)])

# 'data.frame': 5746 obs. of  12 variables:
#   $ Mortalidad        : Factor w/ 2 levels "dead","live": 2 2 2 2 1 2 2 2 2 2 ...
# $ Sexo              : Factor w/ 2 levels "Hombre","Mujer": 2 1 1 1 1 2 1 2 1 2 ...
# $ Edad              : int  63 59 68 46 48 55 56 68 69 64 ...
# $ Year              : Factor w/ 9 levels "85","86","87",..: 1 1 1 1 1 1 1 1 1 1 ...
# $ Infarto           : Factor w/ 2 levels "No","Si": 1 2 1 1 1 1 1 2 1 1 ...
# $ Fumador           : Factor w/ 3 levels "Exfumador","Habitual",..: 1 1 3 2 3 2 1 2 3 1 ...
# $ Diabetes          : Factor w/ 2 levels "No","Si": 1 1 1 1 2 1 1 1 1 1 ...
# $ Tension           : Factor w/ 2 levels "No","Si": 2 2 2 1 1 2 2 2 2 2 ...
# $ Colesterol        : Factor w/ 2 levels "No","Si": 2 1 1 1 1 2 1 1 2 1 ...
# $ Angina            : Factor w/ 2 levels "No","Si": 1 1 1 1 2 1 1 2 1 2 ...
# $ Accidente         : Factor w/ 2 levels "No","Si": 1 1 1 1 1 1 1 1 1 1 ...
# $ Hospitalizacion   : Factor w/ 2 levels "No","Si": 2 2 2 2 2 2 2 2 2 2 ...

# Salvamos el fichero con los datos depurados
  saveRDS(cbind(Monica_Dep),"Monica_Dep")
  monica <- readRDS("Monica_Dep")
```

```r
# Remove objetos de Global Enviroment
  rm(input)
  rm(Monica_Dep)

#-----------------------------------------------------------*
#      ANALISIS EXPLORATORIO DE LOS DATOS
#
#      Frecuencia de variables
#      Relación de variables con variable objetivo
#-----------------------------------------------------------*


  attach(monica)

  # Grafica de alguna de las variables y su frecuencia

# SEXO
  questionr::freq(Sexo)
  barras_targetbinaria(Sexo,Mortalidad,"Sexo")
  mosaico_targetbinaria(Sexo,Mortalidad,"Sexo")

# AÑO en que se tomó el dato
  questionr::freq(Year)
  barras_targetbinaria(Year,Mortalidad,"Year")
  mosaico_targetbinaria(Year,Mortalidad,"Year")

# INFARTO PREVIO
  questionr::freq(Infarto)
  barras_targetbinaria(Infarto,Mortalidad,"Infarto")
  mosaico_targetbinaria(Infarto,Mortalidad,"Infarto")

# STATUS FUMADOR
  questionr::freq(Fumador)
  barras_targetbinaria(Fumador,Mortalidad,"Fumador")
  mosaico_targetbinaria(Fumador,Mortalidad,"Fumador")

# DIABETES
  questionr::freq(Diabetes)
  barras_targetbinaria(Diabetes,Mortalidad,"Diabetes")
  mosaico_targetbinaria(Diabetes,Mortalidad,"Diabetes")

# TENSION ALTA
  questionr::freq(Tension)
  barras_targetbinaria(Tension,Mortalidad,"Tensión")
  mosaico_targetbinaria(Tension,Mortalidad,"Tensión")

# COLESTEROL ALTO
  questionr::freq(Colesterol)
```

```r
  barras_targetbinaria(Colesterol,Mortalidad,"Colesterol")
  mosaico_targetbinaria(Colesterol,Mortalidad,"Colesterol")

# ANGINA DE PECHO
  questionr::freq(Angina)
  barras_targetbinaria(Angina,Mortalidad,"Angina")
  mosaico_targetbinaria(Angina,Mortalidad,"Angina")

# ACCIDENTE CEREBRAL
  questionr::freq(Accidente)
  barras_targetbinaria(Accidente,Mortalidad,"Accidente")
  mosaico_targetbinaria(Accidente,Mortalidad,"Accidente")


# HOSPITALIZACION
  questionr::freq(Hospitalizacion)
  barras_targetbinaria(Hospitalizacion,Mortalidad,"Hospitalización")
  mosaico_targetbinaria(Hospitalizacion,Mortalidad,"Hospitalización")


# EDAD
  psych::describe(Filter(is.numeric, monica))
  boxplot_targetbinaria(Edad,Mortalidad,"% Edad")
  hist_targetbinaria(Edad,Mortalidad,"% Edad")




#-------------------------------------------------------------*
#      SELECCIÓN DE VARIABLES
#
#      V de Cramer
#      Con remuestreo (función steprepetido binaria)
#-------------------------------------------------------------*


#--------------- V DE CRAMER ----------------#

  varObjBin <- monica$Mortalidad
  input <- as.data.frame(monica[,-c(1)])

# Creación de la variable aleatoria
  input$aleatorio <- runif(nrow(input))

# Para representar la V de Cramer para todas las variables input, usamos
la función:
  graficoVcramer(input,varObjBin)

# Parece que la variable aleatoria tiene más importancia que Sexo
```

```r
  dput(names(monica))

  todas <- c("Mortalidad", "Sexo", "Edad", "Year", "Infarto", "Fumador",
             "Diabetes", "Tension", "Colesterol", "Angina", "Accidente",
             "Hospitalizacion")

# Se transforma a numérica porque como int da error la estandarización
  monica$Edad <- as.numeric(monica$Edad)
  listconti <- c("Edad")

# Variables categóricas
  listclass <- c("Sexo", "Year", "Infarto", "Fumador",
                 "Diabetes", "Tension", "Colesterol",
                 "Angina", "Accidente","Hospitalizacion")

# Variables objetivo
  vardep <- "Mortalidad"

# Transformamos las categóricas a dummies

  monicabis <- dummy.data.frame(monica, listclass, sep = ".")

# Estandarizar la variable continua

# Calculo medias y desviación tipica de datos y estandarizo

  monicabis$Edad <- (monicabis$Edad - mean(monicabis$Edad)) /
sd(monicabis$Edad)

# Se modifica la variable objetivo para que incluya las categorías Yes y
No, necesario
# porque algunos algoritmos de machine learning fallan sin no se hace
este paso

    monicabis$Mortalidad <-
ifelse(monicabis$Mortalidad=="dead","Yes","No")

# Salvamos el dataset
  save(monicabis,file = "monicabis.Rda")
  load("monicabis.Rda")

#-------------- STEPWISE REPETIDO --------#

  source("funcion steprepetido binaria.R")

# Para saber los nombres de las variables de nuestro dataset
  dput(names(monicabis))
```

```r
# Lista de todas las variables
  listconti <- c("Sexo.Hombre", "Sexo.Mujer", "Edad", "Year.85",
                 "Year.86", "Year.87", "Year.88", "Year.89",
                 "Year.90", "Year.91", "Year.92", "Year.93",
                 "Infarto.No", "Infarto.Si", "Fumador.Exfumador",
                 "Fumador.Habitual", "Fumador.Nofumador",
                 "Diabetes.No", "Diabetes.Si", "Tension.No",
                 "Tension.Si","Colesterol.No", "Colesterol.Si",
                 "Angina.No", "Angina.Si","Accidente.No",
                 "Accidente.Si", "Hospitalizacion.No",
                 "Hospitalizacion.Si")

  vardep <- c("Mortalidad")

  data <- monicabis

# Criterio BIC
# ------------

  lista <- steprepetidobinaria(data=data,

vardep=vardep,listconti=listconti,sinicio=12345,
                               sfinal=12355,porcen=0.8,criterio="BIC")

  tabla <- lista[[1]]
  variables_BIC <- dput(lista[[2]][[1]])


# Criterio AIC
# ------------

  lista <- steprepetidobinaria(data=data,

vardep=vardep,listconti=listconti,sinicio=12345,
                               sfinal=12355,porcen=0.8,criterio="AIC")

  tabla <- lista[[1]]
  variables_AIC <- dput(lista[[2]][[1]])


#-------------REGRESIÓN LOGISTICA PARA SELECCIONAR EL MEJOR CONJUNTO DE
VARIABLES

  variables_all <- c("Sexo.Hombre", "Sexo.Mujer", "Edad", "Year.85",
                     "Year.86", "Year.87", "Year.88", "Year.89",
                     "Year.90", "Year.91", "Year.92", "Year.93",
                     "Infarto.No", "Infarto.Si", "Fumador.Exfumador",
                     "Fumador.Habitual", "Fumador.Nofumador",
```

```
                    "Diabetes.No", "Diabetes.Si", "Tension.No",
                    "Tension.Si","Colesterol.No", "Colesterol.Si",
                    "Angina.No", "Angina.Si","Accidente.No",
                    "Accidente.Si", "Hospitalizacion.No",
                    "Hospitalizacion.Si")

  source("cruzadas avnnet y log binaria.R") #Función para obtener
cruzadalogística

# Realizamos regresión logística para el conjunto BIC

  mediasl1 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_BIC,

listclass=c(""),grupos=4,sinicio=1234,repe=20)

  mediasl1$modelo ="logística_BIC"

# Realizamos regresión logística para el conjunto AIC

  mediasl2 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,

listclass=c(""),grupos=4,sinicio=1234,repe=20)


  mediasl2$modelo = "logística_AIC"

# Realizamos regresión logística con todas las variables

  mediasl3 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_all,

listclass=c(""),grupos=4,sinicio=1234,repe=20)

  mediasl3$modelo="logística_all"

  union1 <- rbind(mediasl1, mediasl2,mediasl3)


  par(cex.axis = 0.8)
  boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
VARIABLES",col=c("#A6CEE3"))
  boxplot(data = union1, auc ~ modelo, main = "AUC
VARIABLES",col=c("#1F78B4"))

# Elegimos como conjunto de variables las seleccionadas por el criterio
AIC
```

```r
#-------------------------------------------------------*
#
#      ALGORITMOS APRENDIZAJE SUPERVISADO
#
#-------------------------------------------------------*


  load("monicabis.Rda")

  data <- monicabis

  variables_AIC <- c("Hospitalizacion.Si", "Edad", "Angina.No",
"Accidente.No",
                     "Year.93", "Diabetes.No", "Colesterol.No",
"Year.85",
                     "Infarto.Si", "Sexo.Hombre", "Fumador.Nofumador",
"Year.91")

  vardep <- c("Mortalidad")

#--------------   LOGISTICA ---------------------------------

  source("cruzadas avnnet y log binaria.R")

# Este modelo no tiene tuneado

# Modelo logística 1 : semilla 1234 y 10 repeticiones
  mediasl1 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,

listclass=c(""),grupos=4,sinicio=1234,repe=10)

   mediasl1$modelo = "logística1"
   save(mediasl1,file = "mediasl1.Rda")

# Modelo logística 2 : semilla 1234 y  15 repeticiones

  mediasl2 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,

listclass=c(""),grupos=4,sinicio=1234,repe=15)

   mediasl2$modelo = "logística2"
   save(mediasl2,file = "mediasl2.Rda")

# Modelo logística 3 : semilla 5678 y  10 repeticiones

  mediasl3 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,
```

```
listclass=c(""),grupos=4,sinicio=5678,repe=10)

   mediasl3$modelo = "logística3"
   save(mediasl3,file = "mediasl3.Rda")

# Modelo logística 4 : semilla 5678 y  15 repeticiones

  mediasl4 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,

listclass=c(""),grupos=4,sinicio=5678,repe=15)

   mediasl4$modelo = "logística4"
   save(mediasl4,file = "mediasl4.Rda")

# Modelo logística 5 : semilla 34567 y  20 repeticiones

  mediasl5 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,

listclass=c(""),grupos=4,sinicio=34567,repe=20)

   mediasl5$modelo = "logística5"
   save(mediasl5,file = "mediasl5.Rda")

# Modelo logística 6 : semilla 34567 y  20 repeticiones

  mediasl6 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=variables_AIC,

listclass=c(""),grupos=4,sinicio=34567,repe=30)

   mediasl6$modelo = "logística6"
   save(mediasl6,file = "mediasl6.Rda")

   load("mediasl1.Rda")
   load("mediasl2.Rda")
   load("mediasl3.Rda")
   load("mediasl4.Rda")
   load("mediasl5.Rda")
   load("mediasl6.Rda")

   union1 <- rbind(mediasl1, mediasl2,mediasl3,mediasl4,
mediasl5,mediasl6)


   par(cex.axis = 0.8)
   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
```

```
LOGISTICA",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
LOGISTICA",col=c("#1F78B4"))

# El modelo logistica 6 es el que presenta menor tasa de fallos y mayor
AUC


#---------------  ARBOLES --------------------------------

  source("cruzada arbolbin.R")

#   Parámetros a tunear:
#     - minbucket: número de observaciones máximas en el nodo final
#     - cp: parámetro de corte de penalización por complejidad (cp = 0,
máxima complejidad)

# Modelo Arboles 1 : semilla 1234, 30 repeticiones, cp=0 y minbucket = 20

  mediasa1 <- cruzadaarbolbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),

grupos=4,sinicio=1234,repe=30,cp=c(0),minbucket =20)


   mediasa1$modelo="arbol1"
   save(mediasa1,file = "mediasa1.Rda")


# Modelo Arboles 2 : semilla 1234, 30 repeticiones, cp=0.001 y minbucket
= 20

  mediasa2 <- cruzadaarbolbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),

grupos=4,sinicio=1234,repe=30,cp=c(0.001),minbucket =20)


   mediasa2$modelo="arbol2"
   save(mediasa2,file = "mediasa2.Rda")

# Modelo Arboles 3 : semilla 1234, 30 repeticiones, cp=0.005 y minbucket
= 30

  mediasa3 <- cruzadaarbolbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),
```

```
grupos=4,sinicio=1234,repe=30,cp=c(0.005),minbucket =30)


   mediasa3$modelo="arbol3"
   save(mediasa3,file = "mediasa3.Rda")

# Modelo Arboles 4 : semilla 5678, 20 repeticiones, cp=0 y minbucket = 20

  mediasa4 <- cruzadaarbolbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),

grupos=4,sinicio=5678,repe=20,cp=c(0),minbucket =20)


   mediasa4$modelo="arbol4"
   save(mediasa4,file = "mediasa4.Rda")

# Modelo Arboles 5 : semilla 5678, 20 repeticiones, cp=0.001 y minbucket
= 20

  mediasa5 <- cruzadaarbolbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),

grupos=4,sinicio=5678,repe=20,cp=c(0.001),minbucket =20)


   mediasa5$modelo="arbol5"
   save(mediasa5,file = "mediasa5.Rda")

# Modelo Arboles 6 : semilla 5678, 20 repeticiones, cp=0.005 y minbucket
= 30

  mediasa6 <- cruzadaarbolbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),

grupos=4,sinicio=5678,repe=20,cp=c(0.005),minbucket =30)

   mediasa6$modelo="arbol6"
   save(mediasa6,file = "mediasa6.Rda")

   load("mediasa1.Rda")
   load("mediasa2.Rda")
   load("mediasa3.Rda")
   load("mediasa4.Rda")
   load("mediasa5.Rda")
```

```r
   load("mediasa6.Rda")

   union1 <- rbind(mediasa1, mediasa2,mediasa3,mediasa4,
mediasa5,mediasa6)



   par(cex.axis = 0.8)
   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
ARBOLES",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
ARBOLES",col=c("#1F78B4"))

# Los modelos arbol1 y arbol4 son bastante parecidos. Elegimos el arbol4
que tiene un poco más AUC

#---------------  REDES ---------------------------------

  source("cruzadas avnnet y log binaria.R")

#   Parámetros a tunear:
#     - Size: número de nodos en la capa oculta
#     - Decay: velocidad de decrecimiento de los pesos

# Modelo Red 1 : semilla 1234, 20 repeticiones, repe = 5, size = 10,
decay = 0.01 y 100 iteraciones

  mediasr1 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=1234,repe=5,size=c(10),
                               decay=c(0.01),repeticiones=20,itera=100)


   mediasr1$modelo="avnnet1"
   save(mediasr1,file = "mediasr1.Rda")


# Modelo Red 2 : semilla 1234, 20 repeticiones, repe = 10, size = 15,
decay = 0.01 y 100 iteraciones

  mediasr2 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=1234,repe=10,size=c(15),
                               decay=c(0.01),repeticiones=20,itera=100)

   mediasr2$modelo="avnnet2"
   save(mediasr2,file = "mediasr2.Rda")
```

```r
# Modelo Red 3 : semilla 1234, 20 repeticiones, repe = 20, size = 20,
decay = 0.001 y 120 iteraciones

  mediasr3 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=1234,repe=20,size=c(20),
                               decay=c(0.001),repeticiones=20,itera=120)

   mediasr3$modelo="avnnet3"
   save(mediasr3,file = "mediasr3.Rda")

# Modelo Red 4 : semilla 5678, 20 repeticiones, repe = 5, size = 10,
decay = 0.01 y 100 iteraciones

  mediasr4 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=5678,repe=5,size=c(10),
                               decay=c(0.01),repeticiones=20,itera=100)

   mediasr4$modelo="avnnet4"
   save(mediasr4,file = "mediasr4.Rda")

# Modelo Red 5 : semilla 5678, 20 repeticiones, repe = 10, size = 15,
decay = 0.001 y 150 iteraciones

  mediasr5 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=5678,repe=10,size=c(15),
                               decay=c(0.001),repeticiones=20,itera=150)

   mediasr5$modelo="avnnet5"
   save(mediasr5,file = "mediasr5.Rda")

# Modelo Red 6 : semilla 5678, 20 repeticiones, repe = 10, size = 20,
decay = 0.001 y 200 iteraciones

  mediasr6 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=5678,repe=10,size=c(20),
                               decay=c(0.001),repeticiones=20,itera=200)

   mediasr6$modelo="avnnet6"
   save(mediasr6,file = "mediasr6.Rda")

   load("mediasr1.Rda")
```

```r
   load("mediasr2.Rda")
   load("mediasr3.Rda")
   load("mediasr4.Rda")
   load("mediasr5.Rda")
   load("mediasr6.Rda")

   union1 <- rbind(mediasr1, mediasr2,mediasr3,mediasr4,
mediasr5,mediasr6)



   par(cex.axis = 0.8)
   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
REDES",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
REDES",col=c("#1F78B4"))

# El modelo avnnet4 es el elegido en este caso debido a que tiene alto
AUC y su tasa de fallos no es demasiado baja

#---------------  RANDOM FOREST ---------------------------------

  source("cruzada rf binaria.R")

#   Parámetros a tunear:
#   -------------------
#     - mtry: es número de variables
#     - nodesize: tamaño máximo de nodos totales (mide la complejidad)
#     - ntree: número de árboles o iteraciones
#     - sampsize: el tamaño de cada muestra
#     - replace: indica si hay reemplazamiento (TRUE) o no (FALSE)
#



# Modelo Random Forest 1 : semilla 1234,mtry= 8, repe = 5, nodesize = 10,
ntree = 300 y sampsize 50

  mediasrf1 <- cruzadarfbin(data=data,
                            vardep=vardep,listconti=variables_AIC,
                            listclass=c(""),
                            grupos=4,sinicio=1234,repe=5,nodesize=10,
                            mtry=8,ntree=300,replace=TRUE,sampsize=50)

   mediasrf1$modelo="ranfor1"
   save(mediasrf1,file = "mediasrf1.Rda")

# Modelo Random Forest 2 : semilla 1234, mtry= 10,repe = 10, nodesize =
15, ntree = 500 y sampsize 100

  mediasrf2 <- cruzadarfbin(data=data,
```

```
                            vardep=vardep,listconti=variables_AIC,
                            listclass=c(""),
                            grupos=4,sinicio=1234,repe=10,nodesize=15,
                            mtry=10,ntree=500,replace=TRUE,sampsize=100)

   mediasrf2$modelo="ranfor2"
   save(mediasrf2,file = "mediasrf2.Rda")

# Modelo Random Forest 3 : semilla 1234, mtry= 10,repe = 15, nodesize =
20, ntree = 1000 y sampsize 150

  mediasrf3 <- cruzadarfbin(data=data,
                          vardep=vardep,listconti=variables_AIC,
                          listclass=c(""),
                          grupos=4,sinicio=1234,repe=15,nodesize=20,
                          mtry=10,ntree=1000,replace=TRUE,sampsize=150)

   mediasrf3$modelo="ranfor3"
   save(mediasrf3,file = "mediasrf3.Rda")

# Modelo Random Forest 4 : semilla 5678, mtry= 8,repe = 5, nodesize = 10,
ntree = 1500 y sampsize 50

  mediasrf4 <- cruzadarfbin(data=data,
                          vardep=vardep,listconti=variables_AIC,
                          listclass=c(""),
                          grupos=4,sinicio=5678,repe=5,nodesize=10,
                          mtry=8,ntree=1500,replace=TRUE,sampsize=50)

   mediasrf4$modelo="ranfor4"
   save(mediasrf4,file = "mediasrf4.Rda")

# Modelo Random Forest 5 : semilla 5678, mtry= 10,repe = 10, nodesize =
15, ntree = 1500 y sampsize 150

  mediasrf5 <- cruzadarfbin(data=data,
                            vardep=vardep,listconti=variables_AIC,
                            listclass=c(""),
                            grupos=4,sinicio=5678,repe=10,nodesize=15,
                            mtry=10,ntree=1500,replace=TRUE,sampsize=150)

   mediasrf5$modelo="ranfor5"
   save(mediasrf5,file = "mediasrf5.Rda")

# Modelo Random Forest 6 : semilla 5678, mtry= 10,repe = 20, nodesize =
20, ntree = 3500 y sampsize 100

  mediasrf6 <- cruzadarfbin(data=data,
                            vardep=vardep,listconti=variables_AIC,
```

```
                            listclass=c(""),
                            grupos=4,sinicio=5678,repe=20,nodesize=20,
                            mtry=10,ntree=3500,replace=TRUE,sampsize=100)

  mediasrf6$modelo="ranfor6"
  save(mediasrf6,file = "mediasrf6.Rda")



  load("mediasrf1.Rda")
  load("mediasrf2.Rda")
  load("mediasrf3.Rda")
  load("mediasrf4.Rda")
  load("mediasrf5.Rda")
  load("mediasrf6.Rda")

  union1 <- rbind(mediasrf1, mediasrf2,mediasrf3,mediasrf4,
mediasrf5,mediasrf6)



  par(cex.axis = 0.8)
  boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS RANDOM
FOREST",col=c("#A6CEE3"))
  boxplot(data = union1, auc ~ modelo, main = "AUC RANDOM
FOREST",col=c("#1F78B4"))

# Elegimos el modelo Random forest 5.

#---------------  BAGGING ----------------------------------

  source("cruzada rf binaria.R")

#   Parámetros a tunear:
#   -------------------
#     Bagging es un caso particular de Random Forest, se tuneando los
mismos
#     parámetros de Random Forest a excepción de mtry que debe coincidir
con
#     el número de variables input, 12 en nuestro caso
#

# Modelo Bagging 1 : semilla 1234, repe = 5, nodesize = 10, ntree = 2000
y sampsize 50

  mediasb1 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=variables_AIC,
                           listclass=c(""),
                           grupos=4,sinicio=1234,repe=5,nodesize=10,
                           mtry=12,ntree=2000,replace=TRUE,sampsize=50)
```

```
  mediasb1$modelo="bagging1"
  save(mediasb1,file = "mediasb1.Rda")

# Modelo Bagging 2 : semilla 1234, repe = 10, nodesize = 15, ntree = 2500
y sampsize 100

  mediasb2 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=variables_AIC,
                           listclass=c(""),
                           grupos=4,sinicio=1234,repe=10,nodesize=15,
                           mtry=12,ntree=2500,replace=TRUE,sampsize=100)

   mediasb2$modelo="bagging2"
   save(mediasb2,file = "mediasb2.Rda")

# Modelo Bagging 3 : semilla 1234, repe = 20, nodesize = 10, ntree = 2500
y sampsize 150

  mediasb3 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=variables_AIC,
                           listclass=c(""),
                           grupos=4,sinicio=1234,repe=20,nodesize=10,
                           mtry=12,ntree=2500,replace=TRUE,sampsize=150)

   mediasb3$modelo="bagging3"
   save(mediasb3,file = "mediasb3.Rda")

# Modelo Bagging 4 : semilla 5678, repe = 5, nodesize = 10, ntree = 3000
y sampsize 50

  mediasb4 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=variables_AIC,
                           listclass=c(""),
                           grupos=4,sinicio=5678,repe=5,nodesize=10,
                           mtry=12,ntree=3000,replace=TRUE,sampsize=50)

   mediasb4$modelo="bagging4"
   save(mediasb4,file = "mediasb4.Rda")

# Modelo Bagging 5 : semilla 5678, repe = 10, nodesize = 15, ntree = 4000
y sampsize 150

  mediasb5 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=variables_AIC,
                           listclass=c(""),
                           grupos=4,sinicio=5678,repe=10,nodesize=15,
                           mtry=12,ntree=4000,replace=TRUE,sampsize=150)
```

```
   mediasb5$modelo="bagging5"
   save(mediasb5,file = "mediasb5.Rda")

# Modelo Bagging 6 : semilla 5678, repe = 20, nodesize = 20, ntree = 3500
y sampsize 200

  mediasb6 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=variables_AIC,
                           listclass=c(""),
                           grupos=4,sinicio=5678,repe=20,nodesize=20,
                           mtry=12,ntree=3500,replace=TRUE,sampsize=200)

   mediasb6$modelo="bagging6"
   save(mediasb6,file = "mediasb6.Rda")


  load("mediasb1.Rda")
  load("mediasb2.Rda")
  load("mediasb3.Rda")
  load("mediasb4.Rda")
  load("mediasb5.Rda")
  load("mediasb6.Rda")

  union1 <- rbind(mediasb1, mediasb2,mediasb3,mediasb4,
mediasb5,mediasb6)


  par(cex.axis = 0.8)
  boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
BAGGING",col=c("#A6CEE3"))
  boxplot(data = union1, auc ~ modelo, main = "AUC
BAGGING",col=c("#1F78B4"))

# Elegimos el modelo bagging5 que si bien no es el de mayor AUC tampoco
es el que menos y además tiene baja tasa de fallos.



#---------------  GRADIENT BOOSTING ---------------------------------

  source ("cruzada gbm binaria.R")

#   Parámetros a tunear:
#   -------------------
#     - shrinkage: parámetro v de regularización (entre 0.001 y 0.2)
#     - n.minobsnode: tamaño máximo de nodos finales
#     - n.trees=el número de iteraciones (árboles)
#     - interaction.depth (2 para árboles binarios)
#
```

```r
# Modelo GBM 1 : semilla 1234, repe = 5, minobsinnode = 10, ntree = 2000
y shrinkage = 0.001

  mediasgbm1 <- cruzadagbmbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),
                              grupos=4,sinicio=1234,repe=5,

n.minobsinnode=10,shrinkage=0.001,n.trees=2000,
                              interaction.depth=2)
   mediasgbm1$modelo="gbm1"
   save(mediasgbm1,file = "mediasgbm1.Rda")

# Modelo GBM 2 : semilla 1234, repe = 5, minobsinnode = 10, ntree = 1500
y shrinkage = 0.01

  mediasgbm2 <- cruzadagbmbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),
                              grupos=4,sinicio=1234,repe=5,

n.minobsinnode=10,shrinkage=0.01,n.trees=1500,
                              interaction.depth=2)
   mediasgbm2$modelo="gbm2"
   save(mediasgbm2,file = "mediasgbm2.Rda")

# Modelo GBM 3 : semilla 1234, repe = 10, minobsinnode = 15, ntree = 2000
y shrinkage = 0.1

   mediasgbm3 <- cruzadagbmbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=1234,repe=10,

n.minobsinnode=15,shrinkage=0.1,n.trees=2000,
                               interaction.depth=2)
   mediasgbm3$modelo="gbm3"
   save(mediasgbm3,file = "mediasgbm3.Rda")

# Modelo GBM 4 : semilla 5678, repe = 5, minobsinnode = 10, ntree = 2000
y shrinkage = 0.01

   mediasgbm4 <- cruzadagbmbin(data=data,
                               vardep=vardep,listconti=variables_AIC,
                               listclass=c(""),
                               grupos=4,sinicio=5678,repe=5,

n.minobsinnode=10,shrinkage=0.01,n.trees=2000,
```

```
                                interaction.depth=2)
   mediasgbm4$modelo="gbm4"
   save(mediasgbm4,file = "mediasgbm4.Rda")


# Modelo GBM 5 : semilla 5678, repe = 10, minobsinnode = 15, ntree = 1500
y shrinkage = 0.005

   mediasgbm5 <- cruzadagbmbin(data=data,
                                vardep=vardep,listconti=variables_AIC,
                                listclass=c(""),
                                grupos=4,sinicio=5678,repe=10,

n.minobsinnode=15,shrinkage=0.005,n.trees=1500,
                                interaction.depth=2)
   mediasgbm5$modelo="gbm5"
   save(mediasgbm5,file = "mediasgbm5.Rda")

# Modelo GBM 6 : semilla 5678, repe = 5, minobsinnode = 10, ntree = 1000
y shrinkage = 0.005

   mediasgbm6 <- cruzadagbmbin(data=data,
                                vardep=vardep,listconti=variables_AIC,
                                listclass=c(""),
                                grupos=4,sinicio=5678,repe=5,

n.minobsinnode=10,shrinkage=0.005,n.trees=1000,
                                interaction.depth=2)
   mediasgbm6$modelo="gbm6"
   save(mediasgbm6,file = "mediasgbm6.Rda")

   load("mediasgbm1.Rda")
   load("mediasgbm2.Rda")
   load("mediasgbm3.Rda")
   load("mediasgbm4.Rda")
   load("mediasgbm5.Rda")
   load("mediasgbm6.Rda")

   union1 <- rbind(mediasgbm1, mediasgbm2,mediasgbm3,mediasgbm4,
mediasgbm5,mediasgbm6)


   par(cex.axis = 0.8)
   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
GBM",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
GBM",col=c("#1F78B4"))

# Elegimos el modelo gbm5 que es el de mayor AUC y uno de los de menor
```

```
tasa de fallos.


#---------------  SVM ---------------------------------

  source ("cruzada SVM binaria lineal.R")
  source ("cruzada SVM binaria polinomial.R")
  source ("cruzada SVM binaria RBF.R")

#   Para utilizar este algoritmo hemos utilizado tres funciones kernel
diferentes:
#       - Lineal
#       - RBF
#       - Polinomial

# Modelo SVM 1 Kernel de funcion lineal: semilla 1234, repe = 5 y c= 0.05

  mediassvm1 <- cruzadaSVMbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),
                              grupos=4,sinicio=1234,repe=5,C=0.05)

   mediassvm1$modelo="SVM1"
   save(mediassvm1,file = "mediassvm1.Rda")


# Modelo SVM 2 Kernel de funcion lineal: semilla 1234, repe = 5 y c= 0.10

  mediassvm2 <- cruzadaSVMbin(data=data,
                              vardep=vardep,listconti=variables_AIC,
                              listclass=c(""),
                              grupos=4,sinicio=1234,repe=5,C=0.10)

   mediassvm2$modelo="SVM2"
   save(mediassvm2,file = "mediassvm2.Rda")

# Modelo SVM 3 Kernel de funcion RBF: semilla 1234, repe = 5, c= 0.10 y
sigma=0.2

  mediassvm3 <- cruzadaSVMbinRBF(data=data,
                                 vardep=vardep,listconti=variables_AIC,
                                 listclass=c(""),

grupos=4,sinicio=1234,repe=5,C=0.10,sigma=0.2)

   mediassvm3$modelo="SVM3"
   save(mediassvm3,file = "mediassvm3.Rda")

# Modelo SVM 4 Kernel de funcion RBF: semilla 1234, repe = 5 y c= 0.001,
```

```
sigma=0.4

  mediassvm4 <- cruzadaSVMbinRBF(data=data,
                                 vardep=vardep,listconti=variables_AIC,
                                 listclass=c(""),

grupos=4,sinicio=1234,repe=5,C=0.001,sigma=0.4)

  mediassvm4$modelo="SVM4"
  save(mediassvm4,file = "mediassvm4.Rda")

# Modelo SVM 5 Kernel de funcion Polinomial : semilla 1234, repe = 10 y
c= 0.01, degree=5 y scale = 0.05

  mediassvm5 <- cruzadaSVMbinPoly(data=data,
                                 vardep=vardep,listconti=variables_AIC,
                                 listclass=c(""),

grupos=4,sinicio=1234,repe=10,C=0.01,degree=5,scale=0.05)

   mediassvm5$modelo="SVM5"
   save(mediassvm5,file = "mediassvm5.Rda")

# Modelo SVM 6 Kernel de funcion Polinomial : semilla 1234, repe = 20 y
c= 0.001, degree=5 y scale = 0.1

  mediassvm6 <- cruzadaSVMbinPoly(data=data,
                                  vardep=vardep,listconti=variables_AIC,
                                  listclass=c(""),

grupos=4,sinicio=1234,repe=20,C=0.001,degree=5,scale=0.1)


   mediassvm6$modelo="SVM6"
   save(mediassvm6,file = "mediassvm6.Rda")


   load("mediassvm1.Rda")
   load("mediassvm2.Rda")
   load("mediassvm3.Rda")
   load("mediassvm4.Rda")
   load("mediassvm5.Rda")
   load("mediassvm6.Rda")

   union1 <- rbind(mediassvm1, mediassvm2,mediassvm3,mediassvm4,
mediassvm5,mediassvm6)


   par(cex.axis = 0.8)
```

```r
   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
SVM",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
SVM",col=c("#1F78B4"))

# Elegimos el modelo SVM 3


#-------------------------------------------------------------*
#
#      COMPARACION CON LOGISTICA
#
#-------------------------------------------------------------*


   load("mediasl6.Rda")
   load("mediasa4.Rda")
   load("mediasr4.Rda")
   load("mediasrf5.Rda")
   load("mediasb5.Rda")
   load("mediasgbm5.Rda")
   load("mediassvm3.Rda")

   union1 <- rbind(mediasl6, mediasa4,mediasr4,mediasrf5,
mediasb5,mediasgbm5, mediassvm3)

   par(cex.axis = 0.8)
   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
TODOS",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
TODOS",col=c("#1F78B4"))


#-------------------------------------------------------------*
#
#      TECNICAS DE ENSAMBLADO
#
#-------------------------------------------------------------*

   source("cruzadas ensamblado binaria fuente.R")

   listconti <- c("Sexo.Hombre", "Sexo.Mujer", "Edad", "Year.85",
                  "Year.86", "Year.87", "Year.88", "Year.89",
                  "Year.90", "Year.91", "Year.92", "Year.93",
                  "Infarto.No", "Infarto.Si", "Fumador.Exfumador",
                  "Fumador.Habitual", "Fumador.Nofumador",
                  "Diabetes.No", "Diabetes.Si", "Tension.No",
                  "Tension.Si","Colesterol.No", "Colesterol.Si",
                  "Angina.No", "Angina.Si","Accidente.No",
```

```
                  "Accidente.Si", "Hospitalizacion.No",
                  "Hospitalizacion.Si")

   listclass<-c("")

   vardep <- c("Mortalidad")

   data <- monicabis


   grupos <- 4
   sinicio <- 5678
   repe <- 10



# APLICACIÓN CRUZADAS PARA ENSAMBLAR

#  Logística ensamblado
   medias1 <- cruzadalogistica(data=data,
                               vardep=vardep,listconti=listconti,

listclass=listclass,grupos=grupos,sinicio=sinicio,repe=repe)

   medias1bis <- as.data.frame(medias1[1])
   medias1bis$modelo <- "Logistica"
   predi1 <- as.data.frame(medias1[2])
   predi1$logi < -predi1$Yes

   save(medias1,file = "medias1.Rda")

#  Redes ensamblado
   medias2 <- cruzadaavnnetbin(data=data,
                               vardep=vardep,listconti=listconti,

listclass=listclass,grupos=grupos,sinicio=sinicio,repe=repe,

size=c(5),decay=c(0.1),repeticiones=20,itera=200)

   medias2bis <- as.data.frame(medias2[1])
   medias2bis$modelo <- "avnnet"
   predi2 <- as.data.frame(medias2[2])
   predi2$avnnet <- predi2$Yes

   save(medias2,file = "medias2.Rda")

#  Random Forest ensamblado
   medias3 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=listconti,
```

```
listclass=listclass,grupos=grupos,sinicio=sinicio,repe=repe,

mtry=10,ntree=1500,nodesize=10,replace=TRUE,sampsize=150)


   medias3bis <- as.data.frame(medias3[1])
   medias3bis$modelo <- "rf"
   predi3 <- as.data.frame(medias3[2])
   predi3$rf <- predi3$Yes

   save(medias3,file = "medias3.Rda")

#  Bagging ensamblado
   medias4 <- cruzadarfbin(data=data,
                           vardep=vardep,listconti=listconti,

listclass=listclass,grupos=grupos,sinicio=sinicio,repe=repe,

mtry=12,ntree=4000,nodesize=15,replace=TRUE,sampsize=150)


   medias4bis <- as.data.frame(medias4[1])
   medias4bis$modelo <- "bg"
   predi4 <- as.data.frame(medias4[2])
   predi4$bg <- predi4$Yes

   save(medias4,file = "medias4.Rda")

#  GBM ensamblado
   medias5 <- cruzadagbmbin(data=data,
                            vardep=vardep,listconti=listconti,

listclass=listclass,grupos=grupos,sinicio=sinicio,repe=repe,

n.minobsinnode=15,shrinkage=0.005,n.trees=1500,interaction.depth=2)

   medias5bis<-as.data.frame(medias5[1])
   medias5bis$modelo<-"gbm"
   predi5<-as.data.frame(medias5[2])
   predi5$gbm<-predi5$Yes

   save(medias5,file = "medias5.Rda")

#  SVM RBF ensamblado
   medias6 <- cruzadaSVMbinRBF(data=data,
                               vardep=vardep,listconti=listconti,
                               listclass=listclass,grupos=grupos,
                               sinicio=sinicio,repe=repe,
```

```
                                C=5,sigma=0.2)

   medias6bis<-as.data.frame(medias6[1])
   medias6bis$modelo<-"svmRadial"
   predi6<-as.data.frame(medias6[2])
   predi6$svmRadial<-predi6$Yes

   save(medias6,file = "medias6.Rda")


   union1<-rbind(medias1bis,medias2bis,
                 medias3bis,medias4bis,medias5bis,medias6bis)

   save(union1,file = "union1.Rda")
   load("union1.Rda")

   par(cex.axis=0.5)

   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
ENSAMBLADO",col=c("#A6CEE3"))
   boxplot(data = union1, auc ~ modelo, main = "AUC
ENSAMBLADO",col=c("#1F78B4"))

# CONSTRUCCIÓN DE TODOS LOS ENSAMBLADOS
# SE UTILIZARÁN LOS ARCHIVOS SURGIDOS DE LAS FUNCIONES LLAMADOS
predi1,...

#  quitamos predi6 de svm porque no tiene buenos resultados

   unipredi <- cbind(predi1,predi2,predi3,predi4,predi5)

# Esto es para eliminar columnas duplicadas
   unipredi <- unipredi[, !duplicated(colnames(unipredi))]

# Construccion de ensamblados, cambiar al gusto

    unipredi$predi6 <- (0.5*unipredi$logi + 0.5*unipredi$avnnet)
    unipredi$predi7 <- (0.5*unipredi$logi + 0.5*unipredi$rf)
    unipredi$predi8 <- (0.5*unipredi$logi + 0.5*unipredi$bg)
    unipredi$predi9 <- (0.5*unipredi$logi + 0.5*unipredi$gbm)

   unipredi$predi10 <- (0.5*unipredi$avnnet + 0.5*unipredi$rf)
   unipredi$predi11 <- (0.5*unipredi$avnnet + 0.5*unipredi$bg)
   unipredi$predi12 <- (0.5*unipredi$avnnet + 0.5*unipredi$gbm)

   unipredi$predi13 <- (0.5*unipredi$rf + 0.5*unipredi$bg)/2
   unipredi$predi14 <- (0.5*unipredi$rf + 0.5*unipredi$gbm)/2

   unipredi$predi15 <- (0.5*unipredi$bg + 0.5*unipredi$gbm)/2
```

```r
   unipredi$predi16 <- (0.33*unipredi$logi + 0.33*unipredi$avnnet +
0.33*unipredi$rf)
   unipredi$predi17 <- (0.33*unipredi$logi + 0.33*unipredi$avnnet +
0.33*unipredi$bg)
   unipredi$predi18 <- (0.33*unipredi$logi + 0.33*unipredi$avnnet +
0.33*unipredi$gbm)

   unipredi$predi19 <- (0.33*unipredi$logi + 0.33*unipredi$rf +
0.33*unipredi$bg)
   unipredi$predi20 <- (0.33*unipredi$logi + 0.33*unipredi$rf +
0.33*unipredi$gbm)

   unipredi$predi21 <- (0.33*unipredi$logi + 0.33*unipredi$bg +
0.33*unipredi$gbm)

   save(unipredi,file = "unipredi.Rda")

# Listado de modelos a considerar

   dput(names(unipredi))

   listado<-c("logi", "avnnet",  "rf","bg",  "gbm",
              "predi6", "predi7", "predi8",
              "predi9", "predi10", "predi11", "predi12",
              "predi13", "predi14", "predi15", "predi16", "predi17",
"predi18",
              "predi19", "predi20", "predi21")

   # Cambiar a Yes, No, todas las predicciones

   for (prediccion in listado)
   {
     unipredi[,prediccion]<-ifelse(unipredi[,prediccion]>0.5,"Yes","No")
   }

# Defino funcion tasafallos

   tasafallos<-function(x,y) {
     confu<-confusionMatrix(x,y)
     tasa<-confu[[3]][1]
     return(tasa)
   }

   # Se obtiene el numero de repeticiones CV y se calculan las medias por
repe en
   # el data frame medias0

   repeticiones<-nlevels(factor(unipredi$Rep))
```

```
   unipredi$Rep<-as.factor(unipredi$Rep)
   unipredi$Rep<-as.numeric(unipredi$Rep)

   medias0<-data.frame(c())
   for (prediccion in listado)
   {
     for (repe in 1:repeticiones)
     {
       paso <- unipredi[(unipredi$Rep==repe),]
       pre<-factor(paso[,prediccion])
       obs<-paso[,c("obs")]
       tasa=1-tasafallos(pre,obs)
       t<-as.data.frame(tasa)
       t$modelo<-prediccion
       medias0<-rbind(medias0,t)
     }
   }

   save(medias0,file = "medias0.Rda")

# Finalmente boxplot (solo  tasa fallos)

   par(cex.axis=0.5,las=2)

   boxplot(data = union1, tasa ~ modelo, main = "TASA FALLOS
PREDICCIONES",col=c("#A6CEE3"))

# Presentación Tabla Medias

   tablamedias<-medias0 %>%
     group_by(modelo) %>%
     summarize(tasa=mean(tasa))

   tablamedias<-tablamedias[order(tablamedias$tasa),]

# ORDENACIÓN DEL FACTOR MODELO POR LAS MEDIANA EN TASA
# PARA EL GRAFICO

   medias0$modelo <- with(medias0,
                          reorder(modelo,tasa, median))
   par(cex.axis=0.7,las=2)
   boxplot(data=medias0,tasa~modelo,main = "TASA FALLOS
ORDENADO",col="#A6CEE3")
```