TFG: La Inteligencia Artificial aplicada a la Inteligencia Emocional

Jorge de Andrés

8 de Junio de 2019

Borramos el environment lo primero para siempre tenerlo limpio en la ejecución:

```
rm(list=ls())
```

Importamos todas las librerías que vamos a ir necesitando:

```
#install.packages("ggplot2")
#install.packages("caret")
#install.packages("plyr")
#install.packages("wordcloud")
#install.packages("hexbin")
#install.packages("RColorBrewer")
#install.packages("corrplot")
#install.packages("FactoMineR")
#devtools::install_github("kassambara/factoextra")
#install.packages("factoextra")
#install.packages("nnet")
#install.packages("plotly")
#install.packages("class")
#install.packages("gmodels")
#install.packages("randomForest")
#install.packages("e1071")
#install.packages("ape")
#install.packages("cluster")
#install.packages("fpc")
#install.packages("devtools")
#devtools::install_github("vqv/ggbiplot")
#install.packages("party")
#install.packages("pROC")

library("ggplot2")
library("caret")

## Loading required package: lattice

library("plyr")
library("wordcloud")

## Loading required package: RColorBrewer
```

```
library("hexbin")
library("RColorBrewer")
library("corrplot")

## corrplot 0.84 loaded

library("FactoMineR")
library("factoextra")

## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at
https://goo.gl/13EFCZ

library("nnet")
library("plotly")

##
## Attaching package: 'plotly'

## The following objects are masked from 'package:plyr':
##
##     arrange, mutate, rename, summarise

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout

library("class")
library("gmodels")
library("randomForest")

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

library("e1071")
library("ape")
library("cluster")
library("fpc")
```

```r
library("devtools")
library("ggbiplot")
```

```
## Loading required package: scales
```

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

```r
library("party")
```

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

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'modeltools'
```

```
## The following object is masked from 'package:plyr':
##
##     empty
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
##
## Attaching package: 'party'
```

```
## The following object is masked from 'package:ape':
##
##     where
```

```r
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following object is masked from 'package:gmodels':
##
##     ci
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

library("ade4")

##
## Attaching package: 'ade4'

## The following object is masked from 'package:FactoMineR':
##
##     reconst
```

Lo primero que tengo que hacer es importar el dataset que he creado:

```
dataset <- read.csv("Datos/datos.txt", header = TRUE)
```

Ahora lo que hago es pasarlo a una matriz, quitando tanto el nombre (que no me interesa) como la etiqueta (que no la necesito por ahora):

```
matriz.pacientes.etiquetas <- dataset[, -1]
matriz.pacientes.datos <- matriz.pacientes.etiquetas[, -25]
```

Análisis Exploratorio

Primero compruebo que todos los datos tienen un tipo correcto.

```
sapply(matriz.pacientes.datos, class)

##              edad               sex rel_ctxo_rel_mala
rel_ctxo_trauma
##         "integer"         "integer"         "integer"
"integer"
##    rel_ctxo_buena           ed_perm           ed_norm
ed_estr
##         "integer"         "integer"         "integer"
"integer"
##          resil_ba          resil_me          resil_al
pen_dic
##         "integer"         "integer"         "integer"
"integer"
##            gen_ex              etiq           fil_men
max_min
##         "integer"         "integer"         "integer"
"integer"
##          conc_arb          pseu_res               deb
raz_emo
##         "integer"         "integer"         "integer"
"integer"
##             inhib             asert             agres
impuls
```

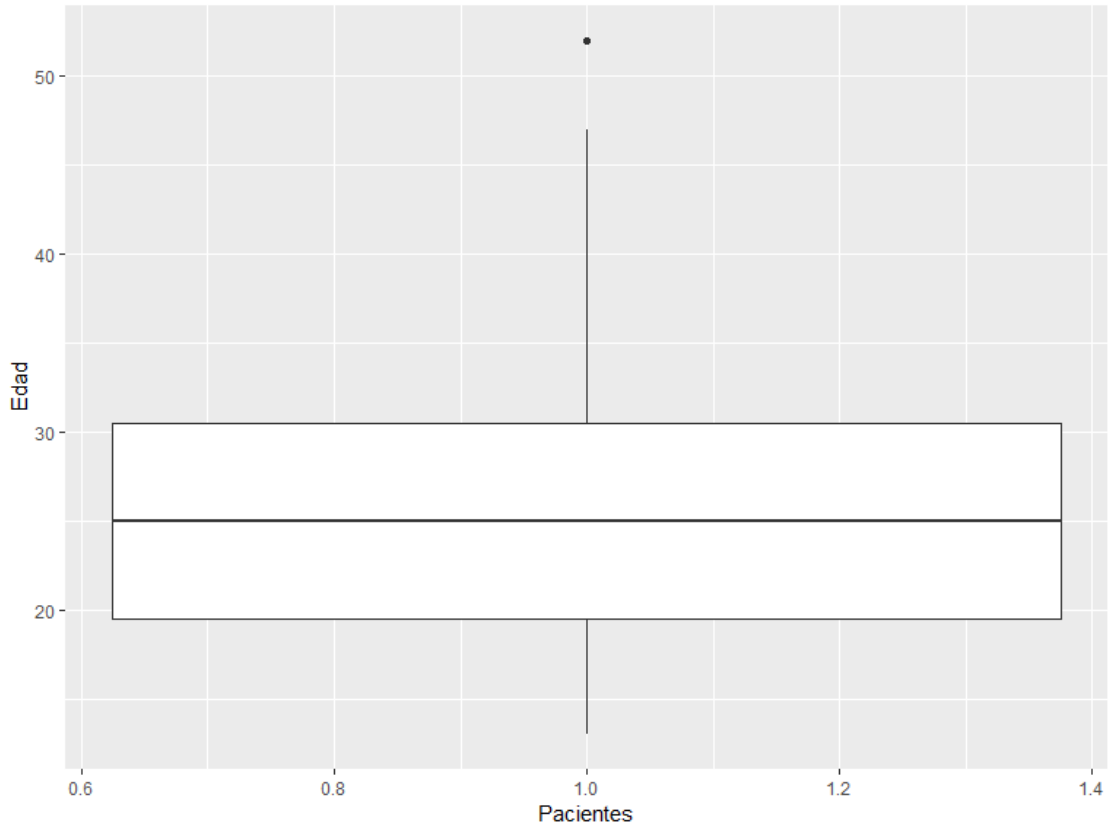
```
##         "integer"         "integer"         "integer"
"integer"
```

Veo la media de la edad de los pacientes y el rango en el que se mueve

```
mean(matriz.pacientes.datos[, 1])

## [1] 26.46269

range(matriz.pacientes.datos[, 1])

## [1] 13 52
```

Voy a ver estos datos gráficamente:

```
qplot(1, matriz.pacientes.datos[, 1], xlab = "Pacientes", ylab = "Edad",
geom="boxplot")
```



Pasamos el qplot a PDF:

```
pdf("Imágenes Obtenidas/boxplotEdadPacientes.pdf")

qplot(1, matriz.pacientes.datos[, 1], xlab = "Pacientes", ylab = "Edad",
geom="boxplot")

dev.off
```

```
## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Finalmente, veo un resúmen de cada columna

```
summary(matriz.pacientes.datos)
```

```
##       edad            sex        rel_ctxo_rel_mala rel_ctxo_trauma
##  Min.   :13.00   Min.   :0.000   Min.   :0.0000    Min.   :0.0000
##  1st Qu.:19.50   1st Qu.:0.000   1st Qu.:0.0000    1st Qu.:0.0000
##  Median :25.00   Median :0.000   Median :0.0000    Median :0.0000
##  Mean   :26.46   Mean   :0.209   Mean   :0.1343    Mean   :0.3582
##  3rd Qu.:30.50   3rd Qu.:0.000   3rd Qu.:0.0000    3rd Qu.:1.0000
##  Max.   :52.00   Max.   :1.000   Max.   :1.0000    Max.   :1.0000
##  rel_ctxo_buena     ed_perm          ed_norm          ed_estr
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.5075   Mean   :0.2836   Mean   :0.4925   Mean   :0.2239
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##     resil_ba         resil_me         resil_al          pen_dic
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.0000
##  Median :1.0000   Median :0.0000   Median :0.00000   Median :1.0000
##  Mean   :0.5672   Mean   :0.4179   Mean   :0.01493   Mean   :0.8955
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##      gen_ex            etiq            fil_men          max_min
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:1.0000   1st Qu.:0.5000   1st Qu.:1.000   1st Qu.:1.0000
##  Median :1.0000   Median :1.0000   Median :1.000   Median :1.0000
##  Mean   :0.9552   Mean   :0.7463   Mean   :0.791   Mean   :0.9701
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.000   Max.   :1.0000
##     conc_arb         pseu_res           deb            raz_emo
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:1.000
##  Median :1.0000   Median :1.0000   Median :1.0000   Median :1.000
##  Mean   :0.9851   Mean   :0.5075   Mean   :0.9403   Mean   :0.791
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000
##      inhib            asert            agres            impuls
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
```

```
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :0.0000   Median :0.0000   Median :1.0000
##  Mean   :0.6567   Mean   :0.1343   Mean   :0.2239   Mean   :0.6119
##  3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

Como se puede ver, los datos de los pacientes están muy distanciados, y además su media es muy alta. Así, la media de la edad difiere enormemente del resto de valores de la matriz. Debido a ello, debemos de hacer un preprocesado de los datos del problema.

Preparación de los datos

Como he comentado antes, Lo que voy a hacer ahora es un centrado y escalado de los datos de la matriz. De esta manera, la red neuronal no tendrá ningún valor que destaque especialmente y con ello no dará de inicio más peso a unos valores que a otros, ya que no lo buscamos.

Ahora hacemos un centrado y escalado de los datos, ya que la edad no sigue el rango del resto de valores, y distorsionaría la predicción

```
preObjeto <- preProcess(matriz.pacientes.datos, method=c("center",
"scale"))  # Quiero hacer un centrado y escalado
matriz.pacientes.datos.centscal <- predict(preObjeto,
matriz.pacientes.datos) # Obtengo los valores en la matriz centscal
```

Después del preprocesado, aunque con los datos no preprocesados, voy a hacer la visualización de algunas relaciones entre variables, de tal manera que podamos ver gráficamente algunos aspectos interesantes:

Visualización de Datos

Para empezar voy a sacar una nube de palabras para mostrar los nombres más comúnes en los datos facilitados:

```
# Lo primero que tengo que hacer es contar la frecuencia de los nombres

dataNombres <- ddply(dataset,.(nom),nrow)
dataNombres <- dataNombres[order(dataNombres$V1, decreasing = TRUE), ]
```

Una vez que tengo los nombres contados y ordenados, es el momento de crear la WordCloud

```
set.seed(9999) # Para el mantenimiento del mismo patrón

wordcloud(words = dataNombres$nom, freq = dataNombres$V1, min.freq = 1,
random.order=FALSE, rot.per=0.5, colors=c("Orange","Purple","Pink",
"Red", "Yellow", "Green", "Blue", "Black"))
```

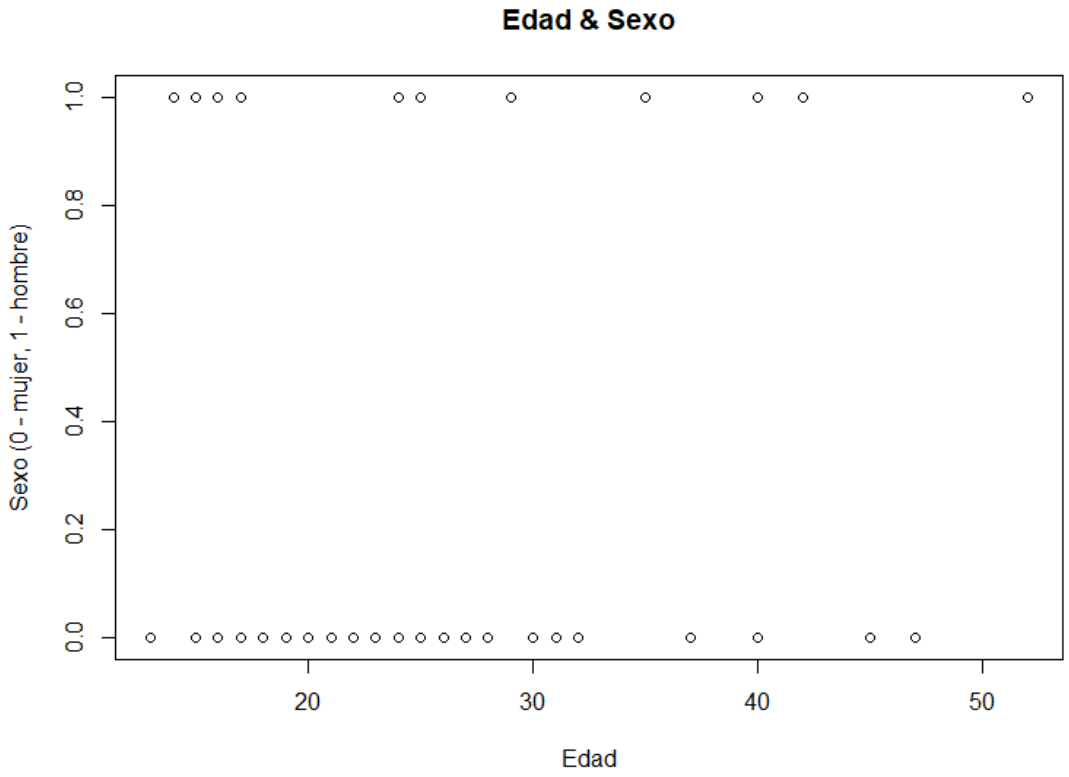
Lo pasamos a PDF:

```
set.seed(9999)

pdf("Imágenes Obtenidas/wordcloudNombresPacientes.pdf")

wordcloud(words = dataNombres$nom, freq = dataNombres$V1, min.freq = 1,
random.order=FALSE, rot.per=0.5, colors=c("Orange","Purple","Pink",
"Red", "Yellow", "Green", "Blue", "Black"))

dev.off
```

```
## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Ahora voy a sacar un plot para ver la relación entre la edad y el sexo de las personas que están en consulta

```r
plot(matriz.pacientes.datos[,1], matriz.pacientes.datos[,2], xlab="Edad",
ylab="Sexo (0 - mujer, 1 - hombre)", main="Edad & Sexo")
```



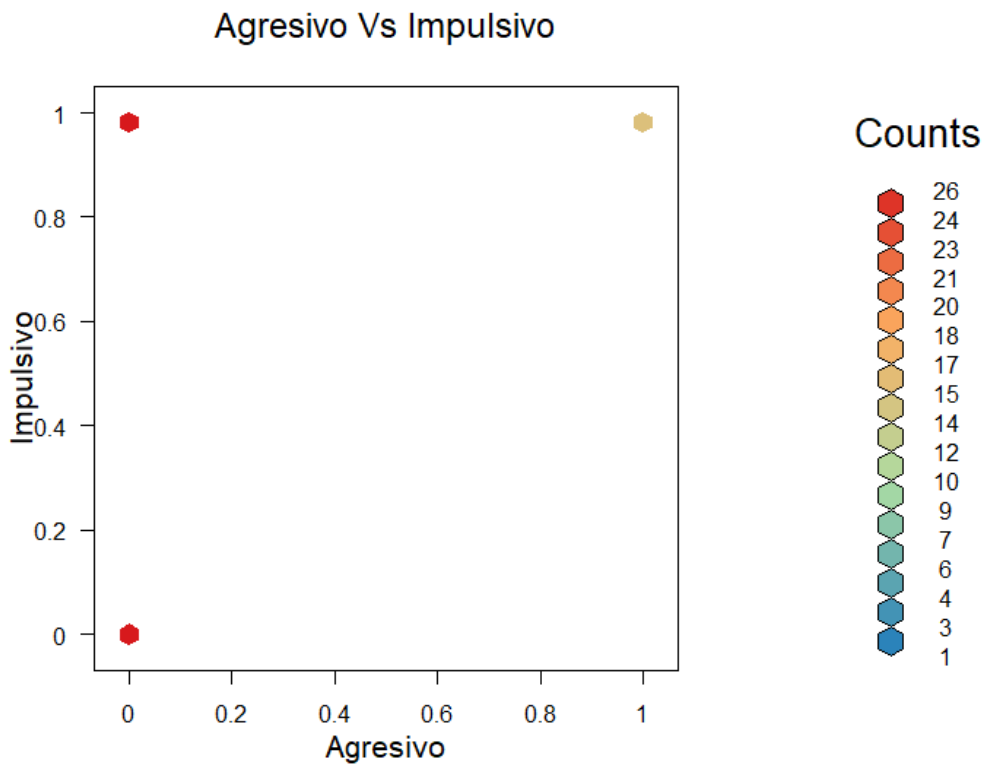
Lo pasamos a PDF:

```r
pdf("Imágenes Obtenidas/GráficoEdad-Sexo.pdf")

plot(matriz.pacientes.datos[,1], matriz.pacientes.datos[,2], xlab="Edad",
ylab="Sexo (0 - mujer, 1 - hombre)", main="Edad & Sexo")

dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Otro plot para ver la correlación entre ser agresivo y ser impulsivo

```r
rf <- colorRampPalette(rev(brewer.pal(4,'Spectral')))
df <- data.frame(matriz.pacientes.datos[, 23], matriz.pacientes.datos[,
```

```
24])
h <- hexbin(df)

plot(h, colramp=rf, xlab="Agresivo", ylab="Impulsivo", main="Agresivo Vs
Impulsivo")
```

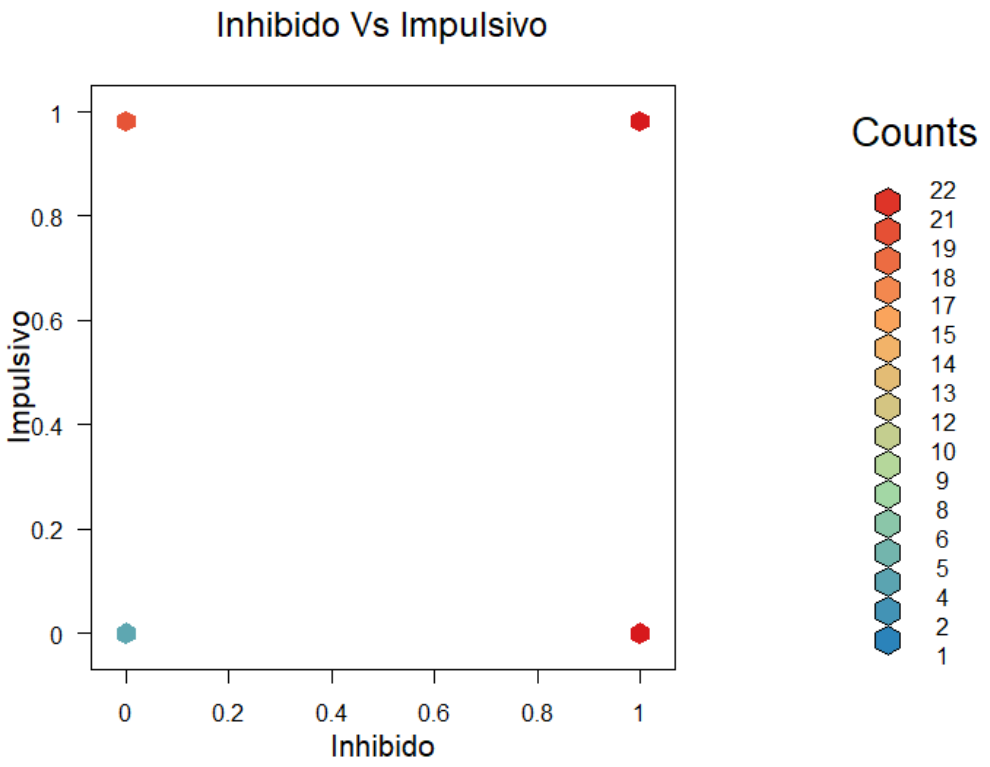


Lo pasamos a PDF:

```
pdf("Imágenes Obtenidas/GraficoAgresivoVsImpulsivo.pdf")

plot(h, colramp=rf, xlab="Agresivo", ylab="Impulsivo", main="Agresivo Vs
Impulsivo")

dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Otro plot similar para ver la relación de ser inhibido e impulsivo

```
df <- data.frame(matriz.pacientes.datos[, 21], matriz.pacientes.datos[,
24])
h <- hexbin(df)

plot(h, colramp=rf, xlab="Inhibido", ylab="Impulsivo", main="Inhibido Vs
Impulsivo")
```



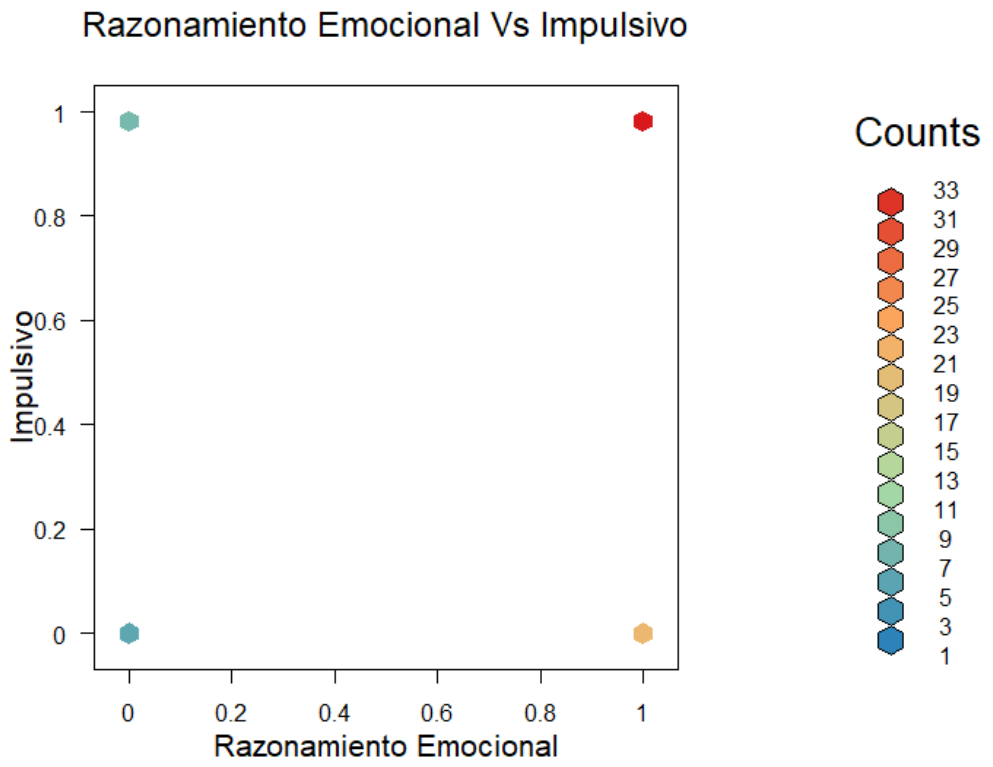
Lo guardo en PDF:

```
pdf("Imágenes Obtenidas/GraficoInhibidoVsImpulsivo.pdf")

plot(h, colramp=rf, xlab="Inhibido", ylab="Impulsivo", main="Inhibido Vs
Impulsivo")

dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
```

```
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Voy a ver la relación entre el razonamiento emocional (actuar según tus sentimientos) y la impulsividad

```
df <- data.frame(matriz.pacientes.datos[, 20], matriz.pacientes.datos[,
24])
h <- hexbin(df)

plot(h, colramp=rf, xlab="Razonamiento Emocional", ylab="Impulsivo",
main="Razonamiento Emocional Vs Impulsivo")
```



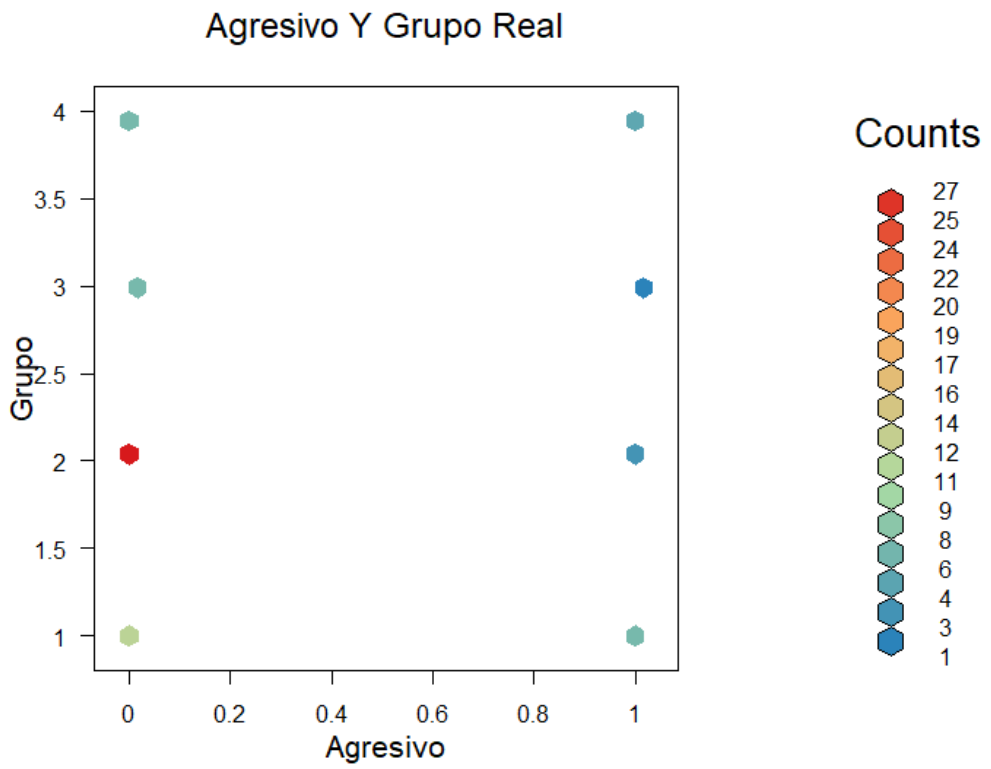
Lo guardo en PDF:

```
pdf("Imágenes Obtenidas/GraficoRazonamientoEmocionalVsImpulsivo.pdf")

plot(h, colramp=rf, xlab="Razonamiento Emocional", ylab="Impulsivo",
main="Razonamiento Emocional Vs Impulsivo")

dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
```

```
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Ahora quiero sacar una relación entre ser agresivo y ver el grupo en el que están

```
rf <- colorRampPalette(rev(brewer.pal(4,'Spectral')))
df <- data.frame(matriz.pacientes.datos[, 23],
matriz.pacientes.etiquetas[, 25])
h <- hexbin(df)

plot(h, colramp=rf, xlab="Agresivo", ylab="Grupo", main="Agresivo Y Grupo
Real")
```



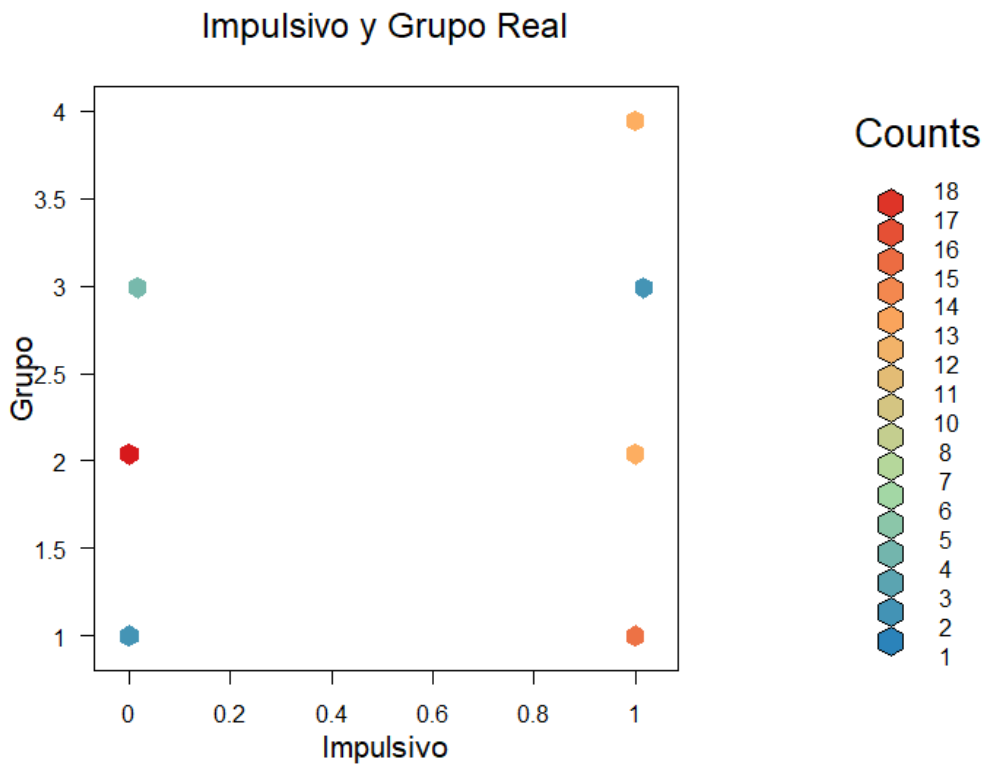
Lo guardo en PDF:

```
pdf("Imágenes Obtenidas/GraficoAgresivoVsGrupo.pdf")

plot(h, colramp=rf, xlab="Agresivo", ylab="Grupo", main="Agresivo Y Grupo
Real")

dev.off
```

```
## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Voy a hacer lo mismo con la impulsividad

```
rf <- colorRampPalette(rev(brewer.pal(4,'Spectral')))
df <- data.frame(matriz.pacientes.datos[, 24],
matriz.pacientes.etiquetas[, 25])
h <- hexbin(df)

plot(h, colramp=rf, xlab="Impulsivo", ylab="Grupo", main="Impulsivo y
Grupo Real")
```



Lo guardo en PDF:

```
pdf("Imágenes Obtenidas/GraficoImpulsivoVsGrupo.pdf")

plot(h, colramp=rf, xlab="Impulsivo", ylab="Grupo", main="Impulsivo y
Grupo Real")
```

```
dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

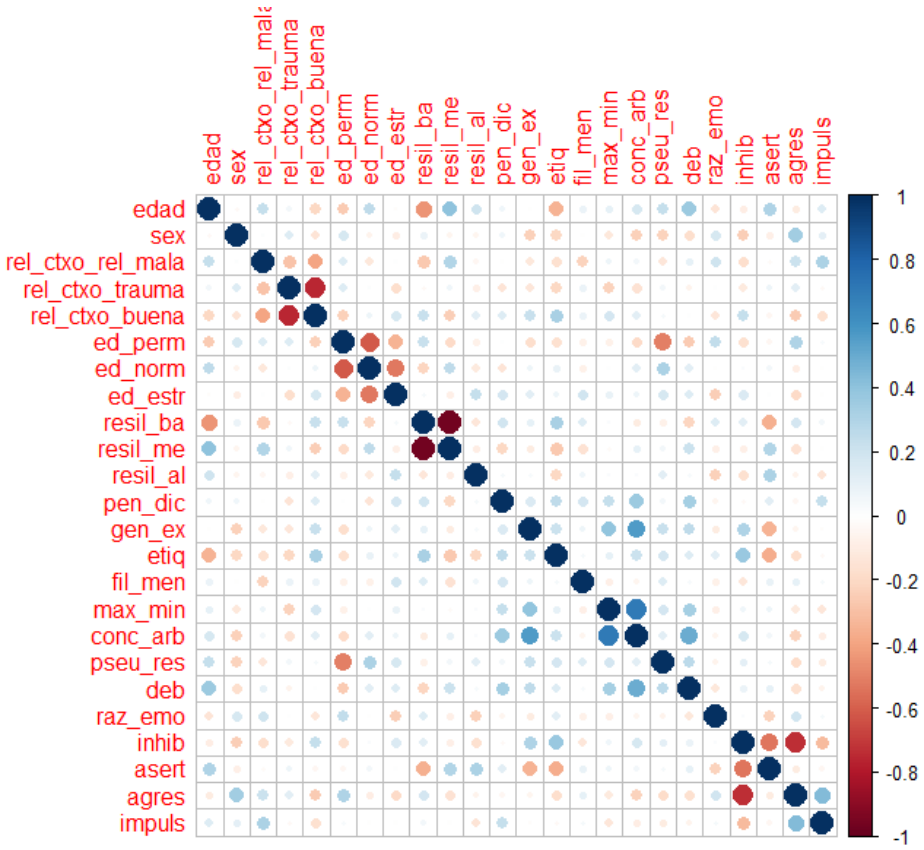
De estas gráficas estamos obteniendo información realmente interesante antes de la predicción de los datos. He preferido hacer gráficas en 2D porque las gráficas en 3D son mucho más difíciles de interpretar que estas bonitas gráficas en 2D

Vamos a ver la correlación que tienen mis variables

```
res <- cor(matriz.pacientes.datos[, 1:24], method = "spearman") # Por mi tipo de datos, hacemos la correlación por spearman
options(width = 100)
res.round <- round(res, 2)
```

Como saca una tabla enorme, lo que voy a hacer es usar una librería que me da una función para sacar de una forma bonita las correlaciones entre las variables.

```
corrplot(res.round, method="circle")
```



Guardamos la matriz de correlación en PDF para tener mejor visualización:

```
pdf("Imágenes Obtenidas/Corrplot.pdf")

corrplot(res.round, method="circle")

dev.off
```

```
## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Como podemos ver, por ejemplo, resiliencia baja y media tienen una correlación de -1, ya que si hay una no hay la otra y viceversa. Esto pasa igual con las relaciones entre contexto, ya que buena - trauma, trauma - mala, mala - buena tienen que ser inversas.

Ahora voy a sacar un PCA para ver la importancia de las variables:

Para los cálculos, uso la matriz con el centrado y escalado ya hechos

```r
resultado.pca <- PCA(matriz.pacientes.datos.centscal, graph = FALSE)

#Con la siguiente línea podemos ver que podemos hacer con esto calculado
print(resultado.pca)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 67 individuals, described by 24
variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

Nos interesa ver los eigenvalues, que son los que presentarán la cantidad de varianza que aportan las variables:

```r
eigenvalues.PCA <- resultado.pca$eig
eigenvalues.PCA
```

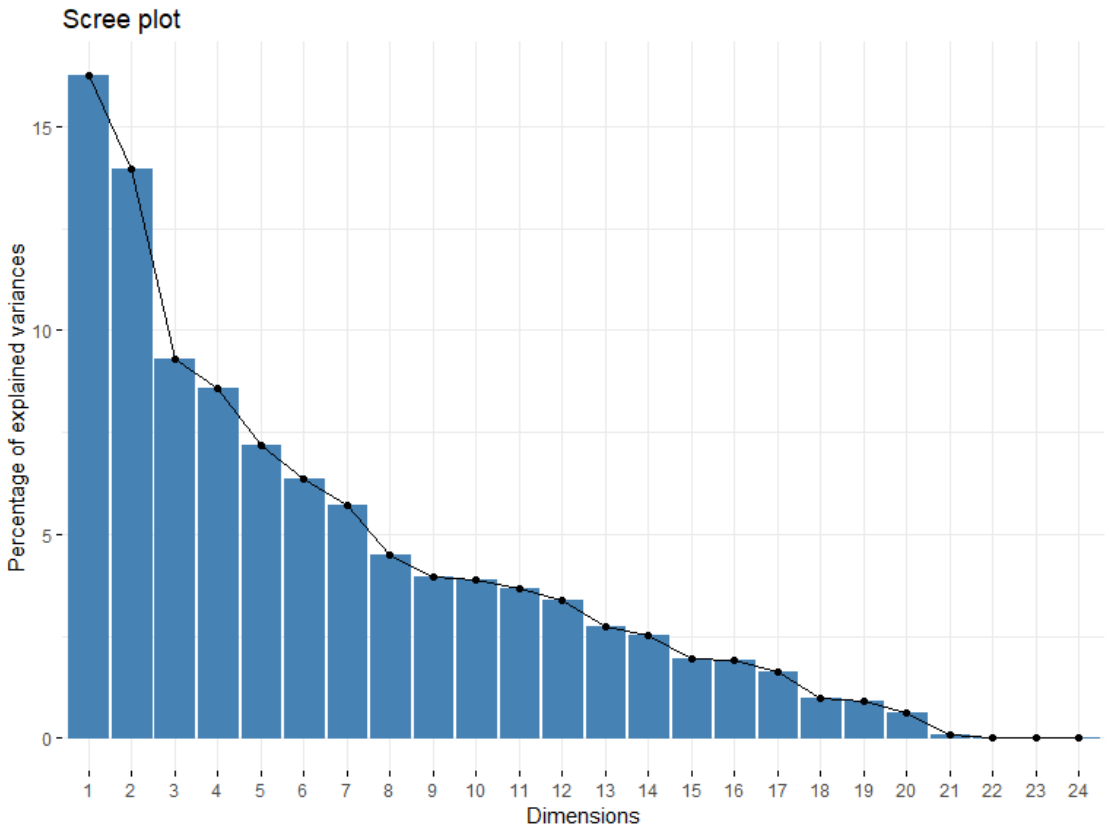
```
##           eigenvalue percentage of variance cumulative percentage of
variance
## comp 1  3.901309e+00           1.625546e+01
16.25546
## comp 2  3.351564e+00           1.396485e+01
30.22031
## comp 3  2.227420e+00           9.280918e+00
39.50122
## comp 4  2.057473e+00           8.572804e+00
48.07403
## comp 5  1.723129e+00           7.179706e+00
55.25373
## comp 6  1.523874e+00           6.349473e+00
61.60321
## comp 7  1.367911e+00           5.699627e+00
67.30283
## comp 8  1.080649e+00           4.502703e+00
71.80554
```

```
## comp 9  9.451881e-01           3.938284e+00
75.74382
## comp 10 9.303616e-01           3.876507e+00
79.62033
## comp 11 8.797962e-01           3.665817e+00
83.28615
## comp 12 8.090124e-01           3.370885e+00
86.65703
## comp 13 6.586878e-01           2.744533e+00
89.40156
## comp 14 6.005323e-01           2.502218e+00
91.90378
## comp 15 4.700232e-01           1.958430e+00
93.86221
## comp 16 4.612585e-01           1.921910e+00
95.78412
## comp 17 3.888962e-01           1.620401e+00
97.40452
## comp 18 2.333650e-01           9.723542e-01
98.37688
## comp 19 2.209107e-01           9.204610e-01
99.29734
## comp 20 1.481544e-01           6.173101e-01
99.91465
## comp 21 2.048464e-02           8.535265e-02
100.00000
## comp 22 1.760166e-31           7.334027e-31
100.00000
## comp 23 7.908215e-32           3.295090e-31
100.00000
## comp 24 7.613437e-33           3.172266e-32
100.00000
```

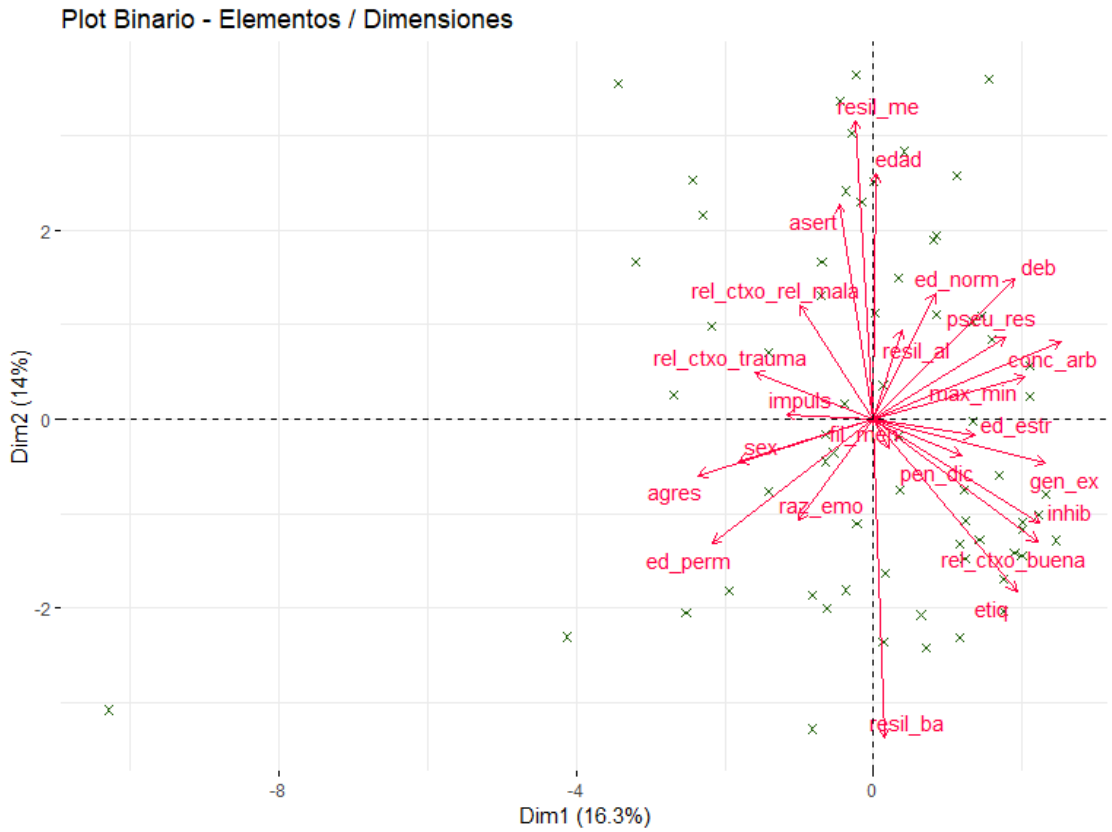
Como se puede comprobar, de las 24 variables (componentes) que tenemos, la mitad de la varianza la conseguimos con aproximadamente 5 variables. También se puede ver que a parti de las 17 variables prácticamente no hay un aumento de la varianza. En el caso de un problema grande, sería interesante la eliminación de algunas de las variables, para dejar un dataset más pequeño con el que poder trabajar. En nuestro caso, nuestro problema es pequeño, y además las variables están escogidas a mano, por lo que no haré una reducción del dataset.

Ahora, para completar este apartado de PCA, lo que voy a hacer es sacar la gráfica de la varianza acumulada con los valores anteriores:

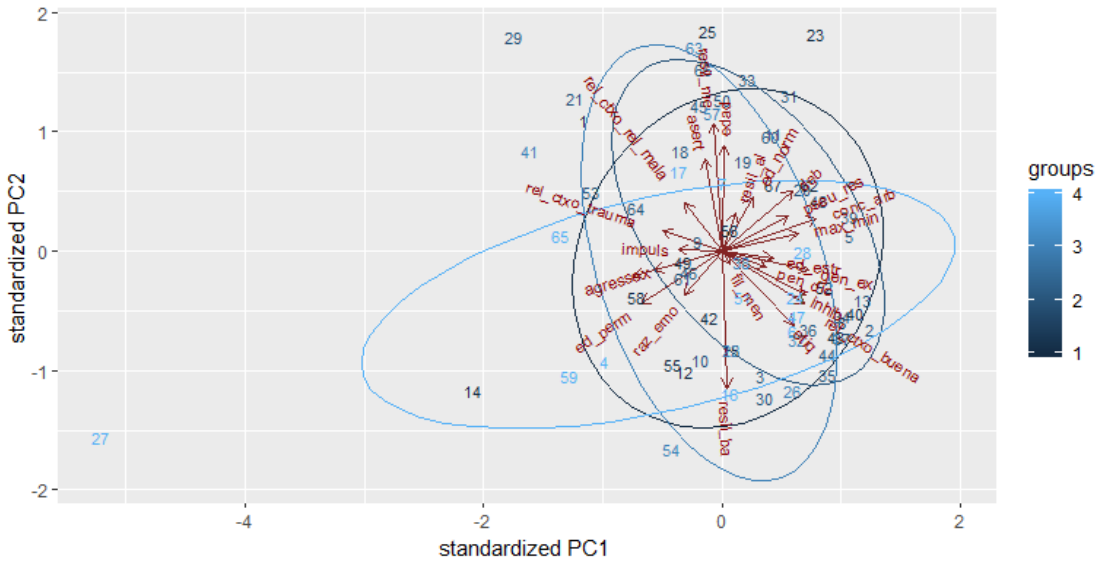
```
plotPCA <- fviz_screeplot(resultado.pca, ncp=24)
plot(plotPCA)
```




```r
fviz_pca_biplot(resultado.pca, repel = TRUE,
                col.var = "#FF0040", # Color de las variables
(vectores)
                col.ind = "#21610B", # Color de cada individuo
(puntos)
                label = "var",
                title = "Plot Binario - Elementos / Dimensiones",
                geom.ind = "point",
                pointshape = 4)
```




```
ggbiplot(resultado.pca, ellipse=TRUE, labels=rownames(dataset),
groups=dataset$grupo)
```



Con esto puedo sacar conclusiones al igual que con el gran gráfico de correlaciones de variables, solo que esta representación está intencionada para más de 2 dimensiones.

Puedo ver algunas de las conclusiones fáciles que saqué anteriormente, como que resiliencia media es contraria a baja, o que la relación con el contexto de trauma y mala son contrarias a buena.

Otras relaciones también puedo ver, como que los deberías y el razonamiento emocional parecen ser ciertamente contrarios, o que el filtro mental no depende de prácticamente nada ya que está en todo el centro.

También es importante ver como, mediante dos componentes principales (dos dimensiones), solo estoy explicando un del 30,2% del total, lo que es muy poco. Por unirlo con los gráficos anteriores, estas dos componentes que se han elegido como x e y son las dos variables que más varianza (y por lo tanto, explicación) tenían en el gráfico de barras anterior.

Obtengo estos gráficos en PDF para tener una mejor visualización:

```
pdf("Imágenes Obtenidas/GraficoEigenvalues.pdf")

plot(plotPCA)

fviz_pca_biplot(resultado.pca, repel = TRUE,
                col.var = "#FF0040", # Color de las variables
```

```
(vectores)
                     col.ind = "#21610B", # Color de cada individuo
(puntos)
                     label = "var",
                     title = "Plot Binario - Elementos / Dimensiones",
                     geom.ind = "point",
                     pointshape = 4)

ggbiplot(resultado.pca, ellipse=TRUE, labels=rownames(dataset),
groups=dataset$grupo)

dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Ahora voy a sacar un "Factor Map" de las variables. Esto lo puedo hacer gracias a las coordenadas que me da una de las variables tras hacer el PCA. Así, voy primero a ver la tabla y luego voy a sacar el mapa:

```
resultado.pca$var$coord

##                          Dim.1       Dim.2      Dim.3       Dim.4
Dim.5
## edad                0.009449477  0.66220660  0.2012166 -0.01688598
0.034353130
## sex                -0.461817396 -0.11469866  0.1142057  0.04821226
0.138691683
## rel_ctxo_rel_mala -0.253565155  0.30698270  0.2523704  0.54165055 -
0.331171378
## rel_ctxo_trauma   -0.406566127  0.12583417 -0.2367909 -0.10989250
0.353071033
## rel_ctxo_buena     0.562872310 -0.33007031  0.0549572 -0.26405906 -
0.112727268
## ed_perm           -0.553298015 -0.33359344  0.2204075  0.17649166 -
0.382130027
## ed_norm            0.210631011  0.33868151 -0.4992287  0.16199773
0.581029158
## ed_estr            0.345664380 -0.04548281  0.3604255 -0.38513556 -
0.283661513
## resil_ba           0.036989747 -0.86064157  0.1925883 -0.06725075
0.209126794
## resil_me          -0.060865419  0.80506164 -0.2729037  0.19505694 -
0.175488418
```

```
## resil_al           0.096429585  0.24207008  0.3231128 -0.51861611 -
0.140713832
## pen_dic            0.301692791 -0.09722286  0.6037969  0.09908508
0.162269610
## gen_ex             0.590510842 -0.11826460  0.1414008  0.27533229
0.139625847
## etiq               0.490832373 -0.46558987 -0.1291992  0.20553979
0.163608829
## fil_men            0.054242913 -0.07841106  0.2818291 -0.38746594
0.409914887
## max_min            0.519764910  0.11369217  0.3655504  0.32864289 -
0.062767279
## conc_arb           0.641299587  0.20703282  0.2929369  0.43016356
0.005040268
## pseu_res           0.452663235  0.21985405 -0.1133463 -0.14273197
0.451008261
## deb                0.481970404  0.37956964  0.2376543  0.24593023
0.104409060
## raz_emo           -0.256501489 -0.27198444 -0.1469739  0.45651306
0.026547345
## inhib              0.569860339 -0.27821097 -0.4926674  0.08127591 -
0.330837919
## asert             -0.118639889  0.57971021  0.2835803 -0.39517909
0.016646777
## agres             -0.600724015 -0.15246890  0.3900381  0.22510662
0.384194829
## impuls            -0.299148404  0.01187884  0.3756371  0.29777080
0.283406252
```

Como se puede ver, me está poniendo mis 24 variables en 5 dimensiones, con unas coordenadas concretas. Ahora, lo que voy a hacer, es representarlo. Con esta representación podré sacar algunas conclusiones:

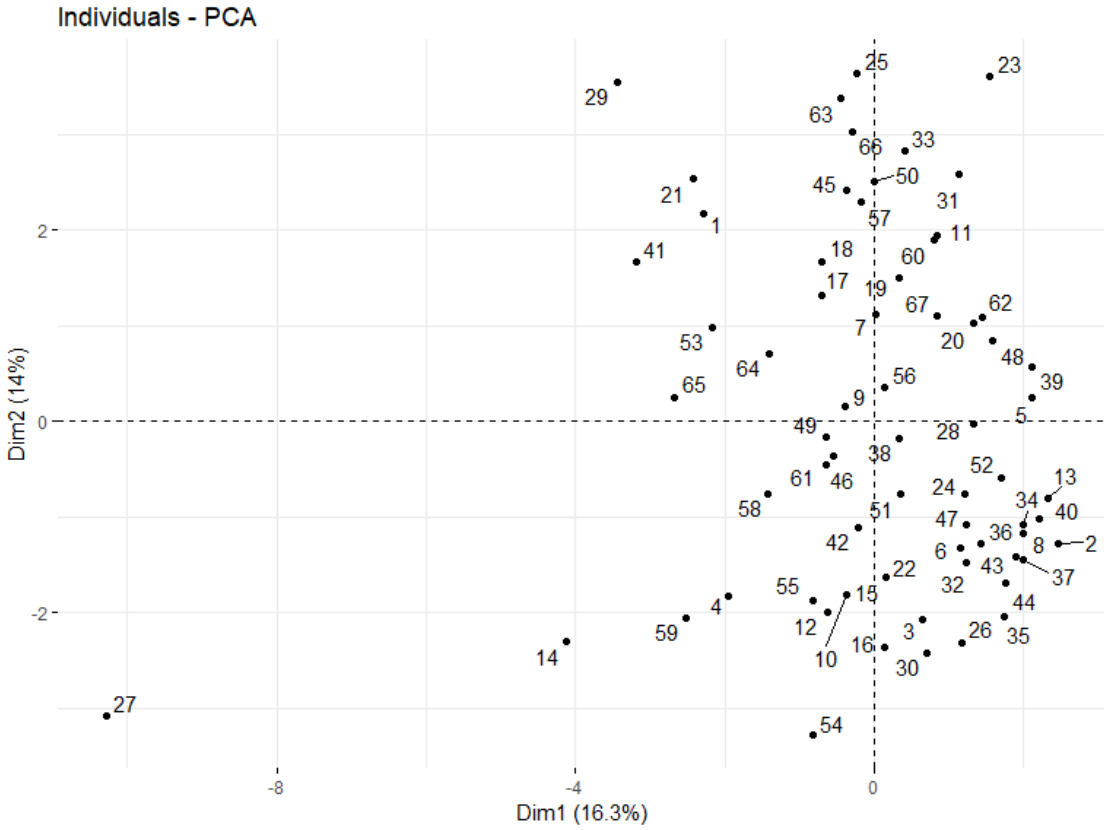
Ahora mi siguiente paso es sacar un gráfico de los individuos, para ver donde están colocados en este sistema:

```
head(resultado.pca$ind$coord) # Solo saco los primeros para no ocupar
demasiado espacio
```

```
##        Dim.1      Dim.2       Dim.3      Dim.4      Dim.5
## 1 -2.2940308  2.1634024 -0.75784032  2.6256358 -0.8316371
## 2  2.4550347 -1.2849670  0.05998261 -1.1988926 -1.4255692
## 3  0.6380076 -2.0755240 -0.22249577  0.6948728 -2.1789437
## 4 -1.9455134 -1.8228537  1.57190908  0.9311446  0.9776417
## 5  2.1050031  0.2444206 -0.28941341 -0.7153514 -1.1887786
## 6  1.1614700 -1.3278368 -0.71761135  1.0750939 -0.2075874
```

Ahora, tras ver que todos mis individuos tienen unas ciertas coordenadas, vamos a representarlos gráficamente:

```r
# Saco este gráfico para ver los individuos de una forma más clara

fviz_pca_ind(resultado.pca, repel = T)
```



Exporto a PDF las coordenadas de los individuos:

```r
pdf("Imágenes Obtenidas/GraficoIndividuosPCA.pdf")

fviz_pca_ind(resultado.pca, repel = T)

dev.off
```

```
## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Se puede ver que la mayoría de los pacientes están en torno al centro, mientras que tenemos un outlayer, que es el número 27.

Modelos de Inteligencia Artificial supervisados

Ahora lo que hago es coger un conjunto muy grande de los datos para hacer el entrenamiento

```
conjuntoEntrenamiento <- sample(1:67, 55)
```

1 NEURONA

Lo que voy a hacer ahora es entrenar la red neuronal con diferente cantidad de neuronas,y voy a ir comparando el resultado...

SIN SOFTMAX

```
set.seed(1)

dataframe.resultados.1neu <- data.frame(Ent_1Neu = numeric(),
                                        Test_1Neu = numeric())

conf.1neu <- vector(mode = "list", length = 50)

for(i in 1:50)
{
  pacientes.1neu <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                         class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                         size=1,
                         trace = F)

  #Una vez que lo tengo entrenado, lo que voy a hacer es calcular el
error tanto en el entrenamiento como en el test de cada uno

  pacientes.prediccion.1neu <- predict( pacientes.1neu,

matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                        type="raw" )
  head(pacientes.prediccion.1neu) # Vemos las probabilidades de
pertenencia de cada valor

  # Ahora que los tengo todos entrenados, Determinamos cual es la máxima,
es decir, la clase a la que hay que asignar los objetos
```

```r
  pacientes.prediccion.1neu.class <- apply( pacientes.prediccion.1neu,
MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.1neu.class

  # Lo visualizo en forma de tabla para ir viendo el error

  table( pacientes.prediccion.1neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  #Calculo el acierto

  acierto.ent.teorico.1neu <- sum( diag( table(
pacientes.prediccion.1neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  #TEST

  pacientes.prediccion.test.1neu <- predict( pacientes.1neu,

matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24],
                                          type="raw" )
  pacientes.prediccion.test.1neu

  pacientes.prediccion.test.1neu.class <- apply(
pacientes.prediccion.test.1neu, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.1neu.class

  conf.1neu[[i]] <- table( pacientes.prediccion.test.1neu.class ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.test.teorico.1neu <- sum( diag( table(
pacientes.prediccion.test.1neu.class, matriz.pacientes.etiquetas[-
conjuntoEntrenamiento, 25] ) ) )/12

  dataframe.pasada <- data.frame(Ent_1Neu = acierto.ent.teorico.1neu,
                                 Test_1Neu = acierto.test.teorico.1neu)

  dataframe.resultados.1neu <- rbind(dataframe.resultados.1neu,
dataframe.pasada)




}

head(dataframe.resultados.1neu[order(dataframe.resultados.1neu$Test_1Neu,
decreasing = T), ])
```

```
##     Ent_1Neu Test_1Neu
## 4  0.6181818 0.6666667
## 7  0.6181818 0.6666667
## 22 0.6363636 0.6666667
## 45 0.6545455 0.6666667
## 1  0.7272727 0.5833333
## 8  0.6727273 0.5833333
```

```
# Como vemos, el mejor resultado lo obtenemos en el entrenamiento #48
# Lo automatizamos y sacamos la matriz de confusión:

conf.1neu[[which.max(dataframe.resultados.1neu$Test_1Neu)]]
```

```
##
## pacientes.prediccion.test.1neu.class 1 2 4
##                                      1 1 1 2
##                                      2 0 7 1
```

Lo voy a entrenar también con el SOFTMAX = true. Esto optimiza la verosimilitud, no el error cuadrático medio…

CON SOFTMAX

```
set.seed(1)

dataframe.resultados.1neu.soft <- data.frame(Ent_1Neu_soft = numeric(),
                                             Test_1Neu_soft = numeric())
conf.1neu.s <- vector(mode = "list", length = 50)

for(i in 1:50)
{
  pacientes.1neu.softmax <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                 class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                                 size=1,
                                 softmax = T,
                                 trace = F)

  #Una vez que lo tengo entrenado, lo que voy a hacer es calcular el
error tanto en el entrenamiento como en el test de cada uno

  pacientes.prediccion.1neu.softmax <- predict( pacientes.1neu.softmax,

matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                                type="raw" )
  head(pacientes.prediccion.1neu.softmax) # Vemos las probabilidades de
pertenencia de cada valor

  # Ahora que los tengo todos entrenados, Determinamos cual es la máxima,
```

```
es decir, la clase a la que hay que asignar los objetos

  pacientes.prediccion.1neu.class.softmax <- apply(
pacientes.prediccion.1neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.1neu.class.softmax

  # Lo visualizo en forma de tabla para ir viendo el error

  table( pacientes.prediccion.1neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  #Calculo el acierto

  acierto.ent.teorico.1neu.soft <- sum( diag( table(
pacientes.prediccion.1neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  #TEST

  pacientes.prediccion.test.1neu.softmax <- predict(
pacientes.1neu.softmax,

matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24],
                                                    type="raw" )
  pacientes.prediccion.test.1neu.softmax

  pacientes.prediccion.test.1neu.class.softmax <- apply(
pacientes.prediccion.test.1neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.1neu.class.softmax

  conf.1neu.s[[i]] <- table( pacientes.prediccion.test.1neu.class.softmax
, matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.test.teorico.1neu.soft <-
sum(diag(table(pacientes.prediccion.test.1neu.class.softmax,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25])))/12

  dataframe.pasada <- data.frame(Ent_1Neu_soft =
acierto.ent.teorico.1neu.soft,
                                 Test_1Neu_soft =
acierto.test.teorico.1neu.soft)

  dataframe.resultados.1neu.soft <- rbind(dataframe.resultados.1neu.soft
,dataframe.pasada)

}
```

```
head(dataframe.resultados.1neu.soft[order(dataframe.resultados.1neu.soft$
Test_1Neu_soft, decreasing = T), ])
```

```
##    Ent_1Neu_soft Test_1Neu_soft
## 9      0.5272727      0.6666667
## 15     0.6000000      0.6666667
## 40     0.6000000      0.6666667
## 43     0.6181818      0.6666667
## 5      0.6363636      0.5833333
## 13     0.6181818      0.5833333
```

```
# El mejor resultado ha sido en el entrenamiento #27

conf.1neu.s[[27]]
```

```
##
## pacientes.prediccion.test.1neu.class.softmax 1 2 4
##                                              1 0 5 0
##                                              2 1 3 3
```

2 NEURONAS

A partir de ahora voy a hacer exactamente lo mismo, por lo que haré chunks más grandes para evitar una sobrecarga de chunks, y reduciré la cantidad de comentarios, ya que serán redundantes

SIN SOFTMAX

```
set.seed(1)

dataframe.resultados.2neu <- data.frame(Ent_2Neu = numeric(),
                                        Test_2Neu = numeric())
conf.2neu <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.2neu <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                         class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                         size=2,
                         trace = F )

  pacientes.prediccion.2neu <- predict( pacientes.2neu,

matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
```

```
                                        type="raw" )
  head(pacientes.prediccion.2neu) # Vemos las probabilidades de
pertenencia de cada valor

  pacientes.prediccion.2neu.class <- apply( pacientes.prediccion.2neu,
MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.2neu.class


  table( pacientes.prediccion.2neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  acierto.teorico.entrenamiento.2neu <- sum( diag( table(
pacientes.prediccion.2neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  # TEST

  pacientes.prediccion.test.2neu <- predict( pacientes.2neu,

matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24],
                                           type="raw" )
  pacientes.prediccion.test.2neu

  pacientes.prediccion.test.2neu.class <- apply(
pacientes.prediccion.test.2neu, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.2neu.class

  conf.2neu[[i]] <- table( pacientes.prediccion.test.2neu.class ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.2neu <- sum( diag( table(
pacientes.prediccion.test.2neu.class, matriz.pacientes.etiquetas[-
conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_2Neu =
acierto.teorico.entrenamiento.2neu,
                                 Test_2neu = acierto.teorico.test.2neu)

  dataframe.resultados.2neu <- rbind(dataframe.resultados.2neu,
dataframe.pasada)


}

head(dataframe.resultados.2neu[order(dataframe.resultados.2neu$Test_2neu,
decreasing = T), ])
```

```
##     Ent_2Neu Test_2neu
## 2  0.7636364 0.6666667
## 23 0.5818182 0.6666667
## 29 0.8000000 0.6666667
## 49 0.6363636 0.6666667
## 10 0.6363636 0.5833333
## 14 0.8181818 0.5833333
```

```
# El mejor entrenamiento ha sido en la pasada #9, 18 y 38

conf.2neu[[9]]
```

```
##
## pacientes.prediccion.test.2neu.class 1 2 4
##                                      2 0 7 1
##                                      4 1 1 2
```

```
conf.2neu[[18]]
```

```
##
## pacientes.prediccion.test.2neu.class 1 2 4
##                                      1 0 5 1
##                                      2 0 3 1
##                                      3 1 0 1
```

```
conf.2neu[[38]]
```

```
##
## pacientes.prediccion.test.2neu.class 1 2 4
##                                      2 1 8 3
```

CON SOFTMAX

```
set.seed(1)

dataframe.resultados.2neu.soft <- data.frame(Ent_2Neu_soft = numeric(),
                                             Test_2Neu_soft = numeric())

conf.2neu.s <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.2neu.softmax <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                 class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                                 size=2,
                                 softmax = T,
                                 trace = F )
```

```
  pacientes.prediccion.ent.2neu.softmax <- predict(
pacientes.2neu.softmax,

matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                                    type="raw" )
  head(pacientes.prediccion.ent.2neu.softmax)

  pacientes.prediccion.ent.2neu.class.softmax <- apply(
pacientes.prediccion.ent.2neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.ent.2neu.class.softmax

  table( pacientes.prediccion.ent.2neu.class.softmax ,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )
  acierto.teorico.ent.2neu.softmax <- sum( diag( table(
pacientes.prediccion.ent.2neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55

  # TEST

  pacientes.prediccion.test.2neu.softmax <- predict(
pacientes.2neu.softmax,

matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24],
                                                     type="raw" )
  pacientes.prediccion.test.2neu.softmax

  pacientes.prediccion.test.2neu.class.softmax <- apply(
pacientes.prediccion.test.2neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.2neu.class.softmax

  conf.2neu.s[[i]] <- table( pacientes.prediccion.test.2neu.class.softmax
, matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.2neu.softmax <-
sum(diag(table(pacientes.prediccion.test.2neu.class.softmax,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_2Neu_soft =
acierto.teorico.ent.2neu.softmax,
                                 Test_2neu_soft =
acierto.teorico.test.2neu.softmax)

  dataframe.resultados.2neu.soft <- rbind(dataframe.resultados.2neu.soft,
dataframe.pasada)
}

head(dataframe.resultados.2neu.soft[order(dataframe.resultados.2neu.soft$
Test_2neu_soft, decreasing = T), ])
```

```
##    Ent_2Neu_soft Test_2neu_soft
## 14     0.7636364      0.7500000
## 8      0.7636364      0.6666667
## 16     0.8000000      0.6666667
## 20     0.8545455      0.6666667
## 34     0.8363636      0.6666667
## 43     0.8727273      0.6666667

# El mejor entrenamiento ha sido en la pasada 9, 10...

conf.2neu.s[[9]]

##
## pacientes.prediccion.test.2neu.class.softmax 1 2 4
##                                              1 0 2 0
##                                              2 1 2 3
##                                              3 0 4 0

conf.2neu.s[[10]]

##
## pacientes.prediccion.test.2neu.class.softmax 1 2 4
##                                              1 0 5 0
##                                              2 0 1 1
##                                              3 0 1 0
##                                              4 1 1 2
```

3 NEURONAS

SIN SOFTMAX

```
set.seed(1)

dataframe.resultados.3neu <- data.frame(Ent_3Neu = numeric(),
                                        Test_3Neu = numeric())

conf.3neu <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.3neu <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                         class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                         size=3,
                         trace = F)
```

```
  pacientes.prediccion.3neu <- predict( pacientes.3neu,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.3neu) # Vemos las probabilidades de
pertenencia de cada valor


  pacientes.prediccion.3neu.class <- apply( pacientes.prediccion.3neu,
MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.3neu.class


  table( pacientes.prediccion.3neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.


  acierto.teorico.entrenamiento.3neu <- sum( diag( table(
pacientes.prediccion.3neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  # TEST

  pacientes.prediccion.test.3neu <- predict( pacientes.3neu,
matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24], type="raw"
)
  pacientes.prediccion.test.3neu

  pacientes.prediccion.test.3neu.class <- apply(
pacientes.prediccion.test.3neu, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.3neu.class

  conf.3neu[[i]] <- table( pacientes.prediccion.test.3neu.class ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.3neu <- sum( diag( table(
pacientes.prediccion.test.3neu.class, matriz.pacientes.etiquetas[-
conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_3Neu =
acierto.teorico.entrenamiento.3neu,
                                 Test_3neu = acierto.teorico.test.3neu)

  dataframe.resultados.3neu <- rbind(dataframe.resultados.3neu,
dataframe.pasada)
}
```

```r
head(dataframe.resultados.3neu[order(dataframe.resultados.3neu$Test_3neu,
decreasing = T), ])
```

```
##     Ent_3Neu Test_3neu
## 14 0.5636364 0.7500000
## 2  0.6363636 0.6666667
## 12 0.6545455 0.6666667
## 15 0.6363636 0.6666667
## 3  0.8363636 0.5833333
## 13 0.8181818 0.5833333
```

```r
# El mejor entrenamiento ha sido en la pasada 44

conf.3neu[[44]]
```

```
##
## pacientes.prediccion.test.3neu.class 1 2 4
##                                    1 0 5 1
##                                    2 0 3 1
##                                    3 1 0 1
```

CON SOFTMAX

```r
set.seed(1)

dataframe.resultados.3neu.soft <- data.frame(Ent_3Neu_soft = numeric(),
                                             Test_3Neu_soft = numeric())

conf.3neu.s <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.3neu.softmax <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                 class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                                 size=3,
                                 softmax = T,
                                 trace = F)

  pacientes.prediccion.3neu.softmax <- predict( pacientes.3neu.softmax,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.3neu.softmax) # Vemos las probabilidades de
pertenencia de cada valor


  pacientes.prediccion.3neu.class.softmax <-
apply(pacientes.prediccion.3neu.softmax, MARGIN=1, FUN='which.is.max')
```

```
  pacientes.prediccion.3neu.class.softmax


  table( pacientes.prediccion.3neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.


  acierto.teorico.ent.3neu.softmax <- sum( diag( table(
pacientes.prediccion.3neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  #TEST

  pacientes.prediccion.test.3neu.softmax <- predict(
pacientes.3neu.softmax, matriz.pacientes.datos.centscal[-
conjuntoEntrenamiento, 1:24], type="raw" )
  pacientes.prediccion.test.3neu.softmax

  pacientes.prediccion.test.3neu.class.softmax <- apply(
pacientes.prediccion.test.3neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.3neu.class.softmax

  conf.3neu.s[[i]] <- table( pacientes.prediccion.test.3neu.class.softmax
, matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.3neu.softmax <- sum( diag( table(
pacientes.prediccion.test.3neu.class.softmax,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_3Neu_soft =
acierto.teorico.ent.3neu.softmax,
                                 Test_3neu_soft =
acierto.teorico.test.3neu.softmax)

  dataframe.resultados.3neu.soft <- rbind(dataframe.resultados.3neu.soft,
dataframe.pasada)
}

head(dataframe.resultados.3neu.soft[order(dataframe.resultados.3neu.soft$
Test_3neu_soft, decreasing = T), ])

##    Ent_3Neu_soft Test_3neu_soft
## 35     0.8363636      0.7500000
## 8      0.8000000      0.6666667
## 11     0.8181818      0.6666667
## 18     0.8545455      0.6666667
```

```
## 30     0.8909091      0.6666667
## 36     0.8181818      0.6666667

# El mejor entrenamieno ha sido en la pasada 13, 38, 39, 48

conf.3neu.s[[13]]

##
## pacientes.prediccion.test.3neu.class.softmax 1 2 4
##                                              1 1 4 1
##                                              2 0 3 1
##                                              4 0 1 1

conf.3neu.s[[38]]

##
## pacientes.prediccion.test.3neu.class.softmax 1 2 4
##                                              1 1 1 0
##                                              2 0 6 3
##                                              4 0 1 0

conf.3neu.s[[39]]

##
## pacientes.prediccion.test.3neu.class.softmax 1 2 4
##                                              1 0 0 1
##                                              2 1 7 2
##                                              4 0 1 0

conf.3neu.s[[48]]

##
## pacientes.prediccion.test.3neu.class.softmax 1 2 4
##                                              1 0 3 1
##                                              2 0 4 0
##                                              4 1 1 2
```

3 NEURONAS

Con Decay

SIN SOFTMAX

```
set.seed(1)

dataframe.resultados.3neu.decay <- data.frame(Ent_3Neu_decay = numeric(),
                                              Test_3Neu_decay =
numeric())
```

```r
conf.3neu.d <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.3neu.decay <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                               class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                               size=3,
                               decay = 0.2,
                               trace = F)

  pacientes.prediccion.3neu.decay <- predict( pacientes.3neu.decay,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.3neu.decay) # Vemos las probabilidades de
pertenencia de cada valor


  pacientes.prediccion.3neu.class.decay <- apply(
pacientes.prediccion.3neu.decay, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.3neu.class.decay


  table( pacientes.prediccion.3neu.class.decay,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.


  acierto.teorico.ent.3neu.decay <- sum( diag( table(
pacientes.prediccion.3neu.class.decay,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  #TEST

  pacientes.prediccion.test.3neu.decay <- predict( pacientes.3neu.decay,
matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24], type="raw"
)
  pacientes.prediccion.test.3neu.decay

  pacientes.prediccion.test.3neu.class.decay <- apply(
pacientes.prediccion.test.3neu.decay, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.3neu.class.decay

  conf.3neu.d[[i]] <- table( pacientes.prediccion.test.3neu.class.decay ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
```

```
  acierto.teorico.test.3neu.decay <- sum( diag( table(
pacientes.prediccion.test.3neu.class.decay, matriz.pacientes.etiquetas[-
conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_3Neu_decay =
acierto.teorico.ent.3neu.decay,
                                 Test_3neu_decay =
acierto.teorico.test.3neu.decay)

  dataframe.resultados.3neu.decay <-
rbind(dataframe.resultados.3neu.decay, dataframe.pasada)

}

head(dataframe.resultados.3neu.decay[order(dataframe.resultados.3neu.deca
y$Test_3neu_decay, decreasing = T), ])
```

```
##    Ent_3Neu_decay Test_3neu_decay
## 4       0.6727273            0.75
## 15      0.6545455            0.75
## 17      0.6545455            0.75
## 18      0.6545455            0.75
## 19      0.6909091            0.75
## 26      0.6909091            0.75
```

```
# El mejor entenamiento ha sido en la pasada 7, 29, 36

conf.3neu.d[[7]]
```

```
##
## pacientes.prediccion.test.3neu.class.decay 1 2 4
##                                          1 0 2 1
##                                          2 0 6 1
##                                          4 1 0 1
```

```
conf.3neu.d[[29]]
```

```
##
## pacientes.prediccion.test.3neu.class.decay 1 2 4
##                                          1 1 2 2
##                                          2 0 6 1
```

```
conf.3neu.d[[36]]
```

```
##
## pacientes.prediccion.test.3neu.class.decay 1 2 4
##                                          1 1 1 1
##                                          2 0 7 1
##                                          3 0 0 1
```

CON SOFTMAX

```
set.seed(1)

dataframe.resultados.3neu.decay.softmax <- data.frame(Ent_3Neu_decay_sf =
numeric(),
                                                      Test_3Neu_decay_sf
= numeric())
conf.3neu.d.s <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.3neu.decay.softmax <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                       class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                                       size=3,
                                       softmax = T,
                                       decay = 0.03,
                                       trace = F)

  pacientes.prediccion.3neu.decay.softmax <- predict(
pacientes.3neu.decay.softmax,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.3neu.decay.softmax) # Vemos las
probabilidades de pertenencia de cada valor


  pacientes.prediccion.3neu.class.decay.softmax <- apply(
pacientes.prediccion.3neu.decay.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.3neu.class.decay.softmax


  table( pacientes.prediccion.3neu.class.decay.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.


  acierto.teorico.ent.3neu.decay.sf <- sum( diag( table(
pacientes.prediccion.3neu.class.decay.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  # TEST

  pacientes.prediccion.test.3neu.decay.softmax <- predict(
pacientes.3neu.decay.softmax, matriz.pacientes.datos.centscal[-
```

```
conjuntoEntrenamiento, 1:24], type="raw" )
  pacientes.prediccion.test.3neu.decay.softmax

  pacientes.prediccion.test.3neu.class.decay.softmax <- apply(
pacientes.prediccion.test.3neu.decay.softmax, MARGIN=1,
FUN='which.is.max')
  pacientes.prediccion.test.3neu.class.decay.softmax

  conf.3neu.d.s[[i]] <- table(
pacientes.prediccion.test.3neu.class.decay.softmax ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )

  acierto.teorico.test.3neu.decay.sf <- sum( diag( table(
pacientes.prediccion.test.3neu.class.decay.softmax,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] ) ) )/12



  dataframe.pasada <- data.frame(Ent_3Neu_decay_sf =
acierto.teorico.ent.3neu.decay.sf,
                                 Test_3neu_decay_sf =
acierto.teorico.test.3neu.decay.sf)

  dataframe.resultados.3neu.decay.softmax <-
rbind(dataframe.resultados.3neu.decay.softmax, dataframe.pasada)
}

head(dataframe.resultados.3neu.decay.softmax[order(dataframe.resultados.3
neu.decay.softmax$Test_3neu_decay_sf, decreasing = T), ])
```

```
##    Ent_3Neu_decay_sf Test_3neu_decay_sf
## 25         0.8727273          0.6666667
## 9          0.8727273          0.5833333
## 10         0.8545455          0.5833333
## 13         0.8545455          0.5833333
## 17         0.8909091          0.5833333
## 21         0.8909091          0.5833333
```

```
# El mejor entrenamiento ha sido en la pasada 33

conf.3neu.d.s[[33]]
```

```
##
## pacientes.prediccion.test.3neu.class.decay.softmax 1 2 4
##                                                    1 0 2 0
##                                                    2 0 4 1
##                                                    3 0 1 2
##                                                    4 1 1 0
```

5 NEURONAS

SIN SOFTMAX

```
set.seed(1)

dataframe.resultados.5neu <- data.frame(Ent_5Neu = numeric(),
                                        Test_5Neu = numeric())

conf.5neu <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.5neu <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                         class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                         size=5,
                         trace = F )

  pacientes.prediccion.5neu <- predict( pacientes.5neu,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.5neu) # Vemos las probabilidades de
pertenencia de cada valor

  pacientes.prediccion.5neu.class <- apply( pacientes.prediccion.5neu,
MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.5neu.class

  table( pacientes.prediccion.5neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  acierto.teorico.entrenamiento.5neu <- sum( diag( table(
pacientes.prediccion.5neu.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  #TEST

  pacientes.prediccion.test.5neu <- predict( pacientes.5neu,
matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24], type="raw"
)
  pacientes.prediccion.test.5neu
```

```
  pacientes.prediccion.test.5neu.class <- apply(
pacientes.prediccion.test.5neu, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.5neu.class

  conf.5neu[[i]] <- table( pacientes.prediccion.test.5neu.class ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.5neu <- sum( diag( table(
pacientes.prediccion.test.5neu.class, matriz.pacientes.etiquetas[-
conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_5Neu =
acierto.teorico.entrenamiento.5neu,
                                 Test_5neu = acierto.teorico.test.5neu)

  dataframe.resultados.5neu <- rbind(dataframe.resultados.5neu,
dataframe.pasada)

}

head(dataframe.resultados.5neu[order(dataframe.resultados.5neu$Test_5neu,
decreasing = T), ])
```

```
##     Ent_5Neu Test_5neu
## 20 0.8727273 0.7500000
## 26 0.9090909 0.6666667
## 27 0.9090909 0.6666667
## 1  0.6727273 0.5833333
## 3  0.5272727 0.5833333
## 36 0.4363636 0.5833333
```

```
# El mejor entrenamiento ha sido en la pasada 4 y 35

conf.5neu[[4]]
```

```
##
## pacientes.prediccion.test.5neu.class 1 2 4
##                                    1 0 4 2
##                                    2 1 3 1
##                                    4 0 1 0
```

```
conf.5neu[[35]]
```

```
##
## pacientes.prediccion.test.5neu.class 1 2 4
##                                    1 0 6 0
##                                    2 0 2 1
##                                    3 0 0 2
##                                    4 1 0 0
```

CON SOFTMAX

```r
set.seed(1)

dataframe.resultados.5neu.soft <- data.frame(Ent_5Neu_soft = numeric(),
                                             Test_5Neu_soft = numeric())

conf.5neu.s <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.5neu.softmax <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                                 class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                                 size=5,
                                 softmax = T,
                                 trace = F )

  pacientes.prediccion.5neu.softmax <- predict( pacientes.5neu.softmax,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.5neu.softmax) # Vemos las probabilidades de
pertenencia de cada valor

  pacientes.prediccion.5neu.class.softmax <- apply(
pacientes.prediccion.5neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.5neu.class.softmax

  table( pacientes.prediccion.5neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  acierto.teorico.ent.5neu.softmax <- sum( diag( table(
pacientes.prediccion.5neu.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  # TEST

  pacientes.prediccion.test.5neu.softmax <- predict(
pacientes.5neu.softmax, matriz.pacientes.datos.centscal[-
conjuntoEntrenamiento, 1:24], type="raw" )
  pacientes.prediccion.test.5neu.softmax

  pacientes.prediccion.test.5neu.class.softmax <- apply(
pacientes.prediccion.test.5neu.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.5neu.class.softmax

  conf.5neu.s[[i]] <-table( pacientes.prediccion.test.5neu.class.softmax,
```

```
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.5neu.softmax <- sum( diag( table(
pacientes.prediccion.test.5neu.class.softmax,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_5Neu_soft =
acierto.teorico.ent.5neu.softmax,
                                 Test_5neu_soft =
acierto.teorico.test.5neu.softmax)

  dataframe.resultados.5neu.soft <- rbind(dataframe.resultados.5neu.soft,
dataframe.pasada)
}

head(dataframe.resultados.5neu.soft[order(dataframe.resultados.5neu.soft$
Test_5neu_soft, decreasing = T), ])

##    Ent_5Neu_soft Test_5neu_soft
## 14     0.9636364      0.6666667
## 27     0.8909091      0.6666667
## 12     0.9454545      0.5833333
## 31     0.9272727      0.5833333
## 37     0.9272727      0.5833333
## 5      0.9090909      0.5000000

# El mejor entrenamiento ha sido en la pasada 39

conf.5neu.s[[39]]

##
## pacientes.prediccion.test.5neu.class.softmax 1 2 4
##                                              1 0 5 0
##                                              2 1 3 2
##                                              3 0 0 1
```

5 NEURONAS

CON DECAY

SIN SOFTMAX

```
set.seed(1)

dataframe.resultados.5neu.decay <- data.frame(Ent_5Neu_decay = numeric(),
                                              Test_5Neu_decay =
numeric())
```

```r
conf.5neu.d <- vector(mode = "list", length = 50)

for(i in 1:50)
{

  pacientes.5neu.decay <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
                               class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ),
                               size=5,
                               decay=0.1,
                               trace = F)

  pacientes.prediccion.5neu.decay <- predict( pacientes.5neu.decay,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.5neu.decay) # Vemos las probabilidades de
pertenencia de cada valor

  pacientes.prediccion.5neu.decay.class <- apply(
pacientes.prediccion.5neu.decay, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.5neu.decay.class

  table( pacientes.prediccion.5neu.decay.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  acierto.teorico.ent.5neu.decay <- sum( diag( table(
pacientes.prediccion.5neu.decay.class,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  # TEST

  pacientes.prediccion.test.decay.5neu <- predict( pacientes.5neu.decay,
matriz.pacientes.datos.centscal[-conjuntoEntrenamiento, 1:24], type="raw"
)
  pacientes.prediccion.test.decay.5neu

  pacientes.prediccion.test.decay.5neu.class <- apply(
pacientes.prediccion.test.decay.5neu, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.test.decay.5neu.class

  conf.5neu.d[[i]] <- table( pacientes.prediccion.test.decay.5neu.class ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.5neu.decay <- sum( diag( table(
pacientes.prediccion.test.decay.5neu.class, matriz.pacientes.etiquetas[-
conjuntoEntrenamiento, 25] ) ) )/12
```

```
  dataframe.pasada <- data.frame(Ent_5Neu_decay =
acierto.teorico.ent.5neu.decay,
                                 Test_5neu_decay =
acierto.teorico.test.5neu.decay)

  dataframe.resultados.5neu.decay <-
rbind(dataframe.resultados.5neu.decay, dataframe.pasada)

}

head(dataframe.resultados.5neu.decay[order(dataframe.resultados.5neu.deca
y$Test_5neu_decay, decreasing = T), ])
```

```
##    Ent_5Neu_decay Test_5neu_decay
## 16      0.8727273       0.7500000
## 20      0.9090909       0.6666667
## 21      0.9090909       0.6666667
## 23      0.9090909       0.6666667
## 27      0.8909091       0.6666667
## 28      0.9090909       0.6666667
```

```
# El mejor entrenamiento ha sido en la pasada # 3

conf.5neu.d[[3]]
```

```
##
## pacientes.prediccion.test.decay.5neu.class 1 2 4
##                                            1 0 4 0
##                                            2 1 4 1
##                                            4 0 0 2
```

CON SOFTMAX

```
set.seed(1)

dataframe.resultados.5neu.decay.softmax <- data.frame(Ent_5Neu_decay_sf =
numeric(),
                                                      Test_5Neu_decay_sf
= numeric())

conf.5neu.d.s <- vector(mode = "list", length = 50)

for(i in 1:50)
{


  pacientes.5neu.decay.softmax <- nnet(
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24],
```

```
                                        class.ind(
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ,
                                        size=5,
                                        softmax = T,
                                        decay = 0.05,
                                        trace = F)

  pacientes.prediccion.5neu.decay.softmax <- predict(
pacientes.5neu.decay.softmax,
matriz.pacientes.datos.centscal[conjuntoEntrenamiento, 1:24], type="raw"
)
  head(pacientes.prediccion.5neu.decay.softmax) # Vemos las
probabilidades de pertenencia de cada valor

  pacientes.prediccion.5neu.decay.class.softmax <- apply(
pacientes.prediccion.5neu.decay.softmax, MARGIN=1, FUN='which.is.max')
  pacientes.prediccion.5neu.decay.class.softmax

  table( pacientes.prediccion.5neu.decay.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] )  # Lo vemos en
forma de tabla.

  acierto.teorico.ent.5neu.decay.sf <- sum( diag( table(
pacientes.prediccion.5neu.decay.class.softmax,
matriz.pacientes.etiquetas[conjuntoEntrenamiento, 25] ) ) )/55 # Esta
cuenta nos da el índice de acierto

  # TEST

  pacientes.prediccion.test.decay.5neu.softmax <- predict(
pacientes.5neu.decay.softmax, matriz.pacientes.datos.centscal[-
conjuntoEntrenamiento, 1:24], type="raw" )
  pacientes.prediccion.test.decay.5neu.softmax

  pacientes.prediccion.test.decay.5neu.class.softmax <- apply(
pacientes.prediccion.test.decay.5neu.softmax, MARGIN=1,
FUN='which.is.max')
  pacientes.prediccion.test.decay.5neu.class.softmax

  conf.5neu.d.s[[i]] <- table(
pacientes.prediccion.test.decay.5neu.class.softmax ,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] )
  acierto.teorico.test.5neu.decay.sf <- sum( diag( table(
pacientes.prediccion.test.decay.5neu.class.softmax,
matriz.pacientes.etiquetas[-conjuntoEntrenamiento, 25] ) ) )/12


  dataframe.pasada <- data.frame(Ent_5Neu_decay_sf =
acierto.teorico.ent.5neu.decay.sf,
```

```
                                    Test_5neu_decay_sf =
acierto.teorico.test.5neu.decay.sf)

  dataframe.resultados.5neu.decay.softmax <-
rbind(dataframe.resultados.5neu.decay.softmax, dataframe.pasada)

}

head(dataframe.resultados.5neu.decay.softmax[order(dataframe.resultados.5
neu.decay.softmax$Test_5neu_decay_sf, decreasing = T), ])
```

```
##    Ent_5Neu_decay_sf Test_5neu_decay_sf
## 13         0.9454545          0.5000000
## 24         0.9818182          0.5000000
## 47         0.9818182          0.5000000
## 2          0.9818182          0.4166667
## 6          0.9636364          0.4166667
## 12         0.9818182          0.4166667
```

```
# El mejor entrenamiento ha sido en la pasada 12, 25, 27

conf.5neu.d.s[[12]]
```

```
##
## pacientes.prediccion.test.decay.5neu.class.softmax 1 2 4
##                                                    1 0 5 0
##                                                    2 0 3 1
##                                                    4 1 0 2
```

```
conf.5neu.d.s[[25]]
```

```
##
## pacientes.prediccion.test.decay.5neu.class.softmax 1 2 4
##                                                    1 0 3 0
##                                                    2 1 4 1
##                                                    3 0 1 0
##                                                    4 0 0 2
```

```
conf.5neu.d.s[[27]]
```

```
##
## pacientes.prediccion.test.decay.5neu.class.softmax 1 2 4
##                                                    1 0 4 0
##                                                    2 1 3 1
##                                                    3 0 1 0
##                                                    4 0 0 2
```

Ahora lo que hay que hacer es unir todos los resultados:

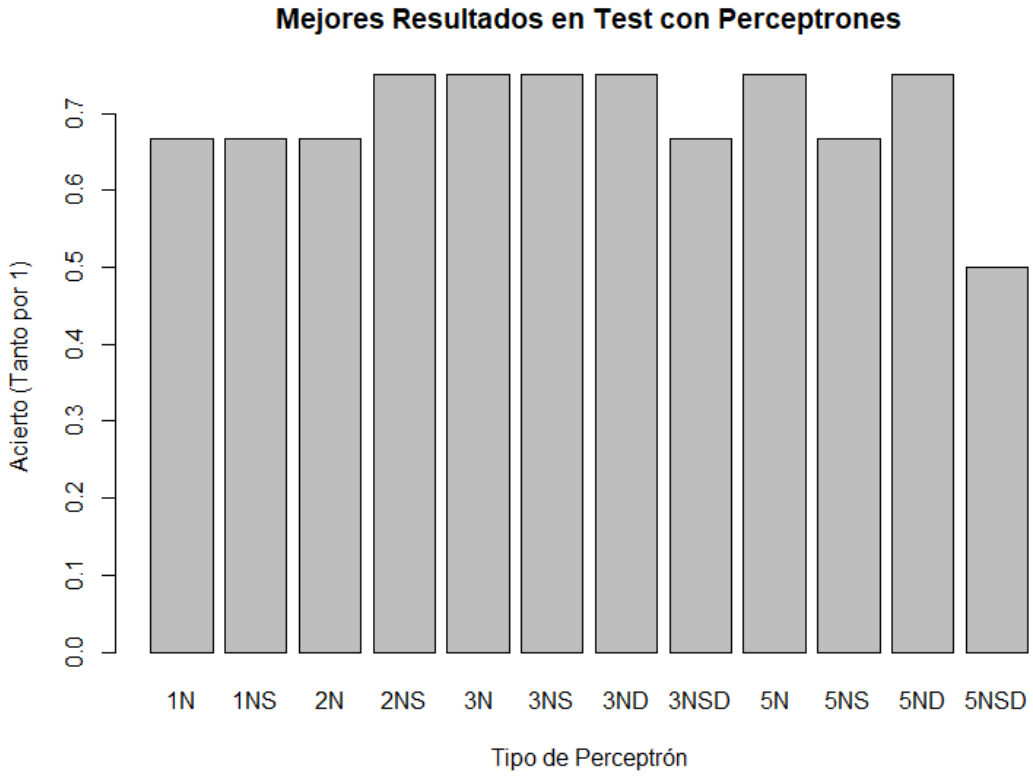
```
dataframe.resultados.perceptron <- cbind(dataframe.resultados.1neu,
                                         dataframe.resultados.1neu.soft,
                                         dataframe.resultados.2neu,
```

```
                                          dataframe.resultados.2neu.soft,
                                          dataframe.resultados.3neu,
                                          dataframe.resultados.3neu.soft,
                                          dataframe.resultados.3neu.decay,

dataframe.resultados.3neu.decay.softmax,
                                          dataframe.resultados.5neu,
                                          dataframe.resultados.5neu.soft,
                                          dataframe.resultados.5neu.decay,

dataframe.resultados.5neu.decay.softmax)

remove(dataframe.resultados.1neu)
remove(dataframe.resultados.1neu.soft)
remove(dataframe.resultados.2neu)
remove(dataframe.resultados.2neu.soft)
remove(dataframe.resultados.3neu)
remove(dataframe.resultados.3neu.soft)
remove(dataframe.resultados.3neu.decay)
remove(dataframe.resultados.3neu.decay.softmax)
remove(dataframe.resultados.5neu)
remove(dataframe.resultados.5neu.soft)
remove(dataframe.resultados.5neu.decay)
remove(dataframe.resultados.5neu.decay.softmax)
```

Ahora visualizamos los mejores resultados de cada entrenamiento:

```
# Obtenemos los máximos de cada columna de test y guardamos:

max.1neu <- max(dataframe.resultados.perceptron[, 2])
max.1neu.s <- max(dataframe.resultados.perceptron[, 4])
max.2neu <- max(dataframe.resultados.perceptron[, 6])
max.2neu.s <- max(dataframe.resultados.perceptron[, 8])
max.3neu <- max(dataframe.resultados.perceptron[, 10])
max.3neu.s <- max(dataframe.resultados.perceptron[, 12])
max.3neu.d <- max(dataframe.resultados.perceptron[, 14])
max.3neu.d.s <- max(dataframe.resultados.perceptron[, 16])
max.5neu <- max(dataframe.resultados.perceptron[, 18])
max.5neu.s <- max(dataframe.resultados.perceptron[, 20])
max.5neu.d <- max(dataframe.resultados.perceptron[, 22])
max.5neu.d.s <- max(dataframe.resultados.perceptron[, 24])

array.maximos.perceptron <- c(max.1neu,
                              max.1neu.s,
                              max.2neu,
                              max.2neu.s,
                              max.3neu,
                              max.3neu.s,
                              max.3neu.d,
                              max.3neu.d.s,
```

```
                                max.5neu,
                                max.5neu.s,
                                max.5neu.d,
                                max.5neu.d.s)

barplot(array.maximos.perceptron,
        main = "Mejores Resultados en Test con Perceptrones",
        xlab = "Tipo de Perceptrón",
        ylab = "Acierto (Tanto por 1)",
        names.arg = c("1N", "1NS",
                      "2N", "2NS",
                      "3N", "3NS", "3ND", "3NSD",
                      "5N", "5NS", "5ND", "5NSD")
      )
```



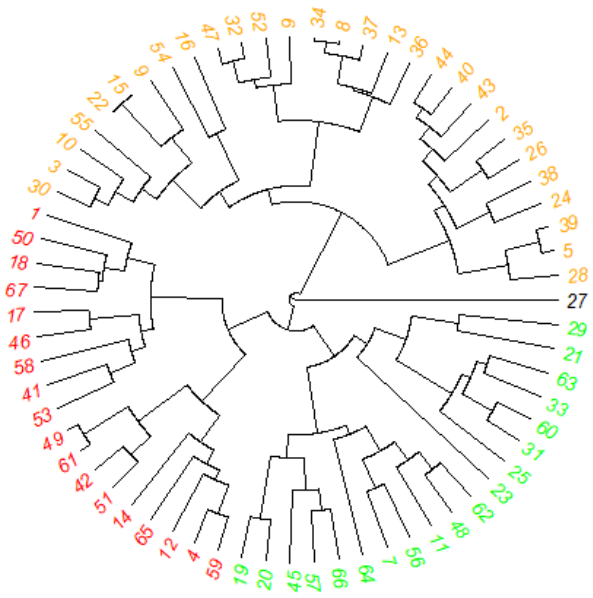
Exporto a PDF este barplot:

```
pdf("Imágenes Obtenidas/BarplotResultadosPerceptron.pdf")

barplot(array.maximos.perceptron,
        main = "Mejores Resultados en Test con Perceptrones",
        xlab = "Tipo de Perceptrón",
        ylab = "Acierto (Tanto por 1)",
        names.arg = c("1 Neu", "1 Neu Soft",
                      "2 Neu", "2 Neu Soft",
```

```
                      "3 Neu", "3 Neu Soft", "3 Neu Decay", "3 Neu Soft
Decay",
                      "5 Neu", "5 Neu Soft", "5 Neu Decay", "5 Neu Soft
Decay")
      )

dev.off

## function (which = dev.cur())
## {
##     if (which == 1)
##         stop("cannot shut down device 1 (the null device)")
##     .External(C_devoff, as.integer(which))
##     dev.cur()
## }
## <bytecode: 0x0000000016406b18>
## <environment: namespace:grDevices>
```

Ahora voy a hacer el mismo dendrograma pero con el DataSet de centrado y escalado:

```
set.seed(6)

dd <- dist(scale(matriz.pacientes.datos.centscal), method = "euclidean")
hier.clust <- hclust(dd, method = "ward.D2")
colores.dendrograma <- c("red", "orange", "green", "black")
cluster.4 <- cutree(hier.clust, 4)
plot(as.phylo(hier.clust), type = "fan", tip.color =
colores.dendrograma[cluster.4], label.offset = 0.3, cex = 0.8)
```



Lo exportamos a PDF tras la creación:

```
pdf("Imágenes Obtenidas/dendrograma_pacientes_centscal.pdf")

plot(as.phylo(hier.clust), type = "fan", tip.color =
colores.dendrograma[cluster.4], label.offset = 0.3, cex = 0.8)

dev.off()
```