Exploración de los datos y Preprocesamiento

Roberto Saborit Roig

22/3/2021

Contents

Dataset

Lo primero es ver que dataset vamos a utilizar, disponemos de dos conjuntos de datos para realizar el estudio, uno es una cohorte de datos longitudinal, mientras que el otro que tenemos en uno seccional. Estos conjuntos pertenecen al proyecto OASIS, que es un proyecto que pretende poner a libre disposición datos de estudios realizados en MRI, donde tenemos una sere de pacientes clasificados como que tienen demencia o no, que utilizaremos como variable respuesta, y variables neuropsicológicas, demográficas y de neuroimagen, que nos servirá como variables predictoras. Iremos realizando el preprocesamiento en ambos conjuntos de manera paralela. Y dispondremos de los dos conjuntos para generar el modelo, probablemente utilizaremos uno como conjunto de de entrenamiento y otro para el test. Pero eso lo valoraremos más adelante en función de los resultados que obtengamos en la exploración de los datos y el preprocesamiento.

```
#Lo primero será cargar los datos y guardarlos en ls dataframes oasis_longitudinal y oasis_cross_seccio
oasis_cross_sectional <- read.csv("oasis_cross-sectional.csv")
oasis_longitudinal <- readxl::read_excel("oasis_longitudinal_demographics.xlsx")
```

Exploración de los datos

Antes de comezar a generar el modelo, incluso antes del preprocesamiento, vamos a realizar una exploración de los datos con los siguientes objetivos:

- Ver si existen valores ausentes en el conjunto de datos y ver su distribución entre las distintas variables.
- Explorar los tipos de variable y ver si necesitamos cambiar la clase de alguna variable.
- Ver la distribución de las variables, tanto de la respuesta como de las variables descriptivas.

Tipos de variables

La primera comprobación que haremos será ver los tipos de variables que hay y si todas tienen el tipo de valor que le corresponde:

```
#Exploramos las variables, sus tipos y vemos si es necesario cambiar algún tipo para posteriores anális
str(oasis_cross_sectional)
```

```
## 'data.frame':    436 obs. of  12 variables:
##  $ ID   : chr  "OAS1_0001_MR1" "OAS1_0002_MR1" "OAS1_0003_MR1" "OAS1_0004_MR1" ...
##  $ M.F  : chr  "F" "F" "F" "M" ...
##  $ Hand : chr  "R" "R" "R" "R" ...
##  $ Age  : int  74 55 73 28 18 24 21 20 74 52 ...
##  $ Educ : int  2 4 4 NA NA NA NA NA 5 3 ...
##  $ SES  : int  3 1 3 NA NA NA NA NA 2 2 ...
##  $ MMSE : int  29 29 27 NA NA NA NA NA 30 30 ...
##  $ CDR  : num  0 0 0.5 NA NA NA NA NA 0 0 ...
##  $ eTIV : int  1344 1147 1454 1588 1737 1131 1516 1505 1636 1321 ...
##  $ nWBV : num  0.743 0.81 0.708 0.803 0.848 0.862 0.83 0.843 0.689 0.827 ...
##  $ ASF  : num  1.31 1.53 1.21 1.1 1.01 ...
##  $ Delay: chr  "N/A" "N/A" "N/A" "N/A" ...
```

```
str(oasis_longitudinal)
```

```
## tibble [373 x 15] (S3: tbl_df/tbl/data.frame)
##  $ Subject ID: chr [1:373] "OAS2_0001" "OAS2_0001" "OAS2_0002" "OAS2_0002" ...
##  $ MRI ID    : chr [1:373] "OAS2_0001_MR1" "OAS2_0001_MR2" "OAS2_0002_MR1" "OAS2_0002_MR2" ...
##  $ Group     : chr [1:373] "Nondemented" "Nondemented" "Demented" "Demented" ...
##  $ Visit     : chr [1:373] "1" "2" "1" "2" ...
##  $ MR Delay  : num [1:373] 0 457 0 560 1895 ...
##  $ M/F       : chr [1:373] "M" "M" "M" "M" ...
##  $ Hand      : chr [1:373] "R" "R" "R" "R" ...
##  $ Age       : num [1:373] 87 88 75 76 80 88 90 80 83 85 ...
##  $ EDUC      : num [1:373] 14 14 12 12 12 18 18 12 12 12 ...
##  $ SES       : num [1:373] 2 2 NA NA NA 3 3 4 4 4 ...
##  $ MMSE      : num [1:373] 27 30 23 28 22 28 27 28 29 30 ...
##  $ CDR       : num [1:373] 0 0 0.5 0.5 0.5 0 0 0 0.5 0 ...
##  $ eTIV      : num [1:373] 1987 2004 1678 1738 1698 ...
##  $ nWBV      : num [1:373] 0.696 0.681 0.736 0.713 0.701 ...
##  $ ASF       : num [1:373] 0.883 0.876 1.046 1.01 1.034 ...
```

Las variables que encontramos son las siguientes:

- *ID o subject ID*: La identificación del paciente.
- *MRI ID*: La identificación del paciente y el número de visita de ese paciente, a la resonancia magnética (MRI).
- *Visit*: Número de visita.
- *MR Delay*: Días que transcurrieron desde la última visita al MRI.
- *M/F*: Sexo.
- *Age*: Edad.
- *EDUC*: Nivel de estudios.
- *SES*: Estatus socioeconómico.
- *MMSE*: Test minimental state. Prueba que indica el nivel de deterioro cognitivo del paciente.
- *CDR*: Clinical dementia ratio. Índice de demencia (leve, muy leve, moderada...)
- *eTIV*: Volumen intracraneal.
- *nWBV*: Volumen intracraneal normalizado.
- *ASF*: Atlas Scaling Factor.

Lo primero que tenemos que tener en cuenta en ambos conjuntos es que el conjunto longitudinal esta etiquetado por grupos en la variable `Group`, estos son "demented", "non demented" y "converted".Hacen referencia a sujetos con demencia, sin demencia y que desarrollaron demencia a lo largo del experimento, repectivamente. En cambio el estudio seccional no tiene estas etiquetas, pero realmente la variable `CDR` (Clasificación Clínica de Demencia), hace referencia a lo mismo, es decir, al nivel de demencia que tienen los pacientes según este test, que va desde 0 (no tiene demencia), 0.5 (demencia muy leve), 1 (demencia leve) o 2 (demencia moderada). Podemos abordar el problema simplemente creando la variable `Group` y etiqutando como demented aquellos que tienen un nivel de demencia mayor de 0.

Además vemos que tenemos 2 variables más en el estudio longitudinal que son `visit`que hace referencia al número de visita de esa observación y `MR delay`, que es el tiempo que ha pasado entre visita y visita.

Por otro lado la variable Hand no aporta ninguna información ya que todos los sujetos son diestros, por tanto, la eliminaremos del conjunto, más adelante en el preprocesamiento cuando decidamos las variables que utilizaremos como predictores:

```
#La función levels muestra los niveles de la variable $Hand
levels(as.factor(oasis_cross_sectional$Hand))
```

```
## [1] "R"
```

```r
levels(as.factor(oasis_longitudinal$Hand))
```

```
## [1] "R"
```

Como vemos solo existe un nivel en el factor que es "R" (Right/Diestro), por lo que no nos aportará ningún valor al modelo, pero si hay que tener en cuenta que el modelo lo habremos realizado solo en personas diestras y si esta variable tuviera importancia en la aparición de demencia los resultados podrían no ser tan precisos en personas zurdas.

Datos y valores ausentes

Vamos a comprobar ahora el número de observaciones y variables que tenemos, comprobando el número de filas y columnas que tienen nuestros datos y la cantidad de valores ausentes que hay, porque muchos modelos de machine learning no aceptan NAs, y en que variables están estos NA, ya que esto será importante para saber como tratar estos valores, si podemos eliminar simplemente las observaciones o los tratamos de otra manera:

```r
#El número total de filas nos indica la cantidad de observaciones de cada uno de los datasets
nrow(oasis_cross_sectional); ncol(oasis_cross_sectional)
```

```
## [1] 436
```

```
## [1] 12
```

```r
nrow(oasis_longitudinal); ncol(oasis_longitudinal)
```

```
## [1] 373
```

```
## [1] 15
```

```r
any(is.na(oasis_cross_sectional)); any(is.na(oasis_longitudinal))
```

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

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

```r
#Tenemos datos ausentes en ambos conjuntos
#Podemos comprobar que variables tienen mayor cantidad de NA
apply(is.na(oasis_cross_sectional), 2, sum)
```

```
##    ID   M.F  Hand   Age  Educ   SES  MMSE   CDR  eTIV  nWBV   ASF Delay
##     0     0     0     0   201   220   201   201     0     0     0     0
```

```r
apply(is.na(oasis_longitudinal), 2, sum)
```

```
## Subject ID     MRI ID      Group      Visit   MR Delay        M/F       Hand
##          0          0          0          0          0          0          0
##        Age       EDUC        SES       MMSE        CDR       eTIV       nWBV
##          0          0         19          2          0          0          0
##        ASF
##          0
```

En el caso del estudio seccional tenemos una gran cantidad de datos ausentes en las variables Educ, SES, MMSE, CDR, que suponen casi un 50% de los datos, en esas variables, lo que puede suponer un problema si durante el preprocesamiento optamos por eliminar las observaciones con datos ausentes, ya que prederíamos una gran cantidad de información. El problema es que tenemos los valores ausentes en la variable CDR que será la variable respuesta en este conjunto de datos, por tanto, habrá que eliminar las observaciones con datos ausentes, al menos que sean ausentes en la variable CDR.

En cambio en el conjunto de datos longitudinal no tenemos apenas datos ausentes, solo unos pocos en la variable SES, que es el estatus socioeconómico, y solo 2 en MMSE (test de deterioro cognitivo). Seguramente también eliminemos las observaciones con NAs, ya que son pocas y no perderemos demasiada información haciendo esto.

Distribución de la variable respuesta

Otra información importante que debemos conocer antes de comenzar con el modelo es la distribución de la variable respuesta, es decir la cantidad de observaciones que hay según los grupos que tenemos en la variable respuesta, en nuestro caso nos interesa conocer como está distribuida la variable `Group` en el estudio longitudinal y la variable `CDR` en el estudio longitudinal. Para ello vamos a generar una tabla con los datos y vamos a verlo también gráficamente:

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.4
```

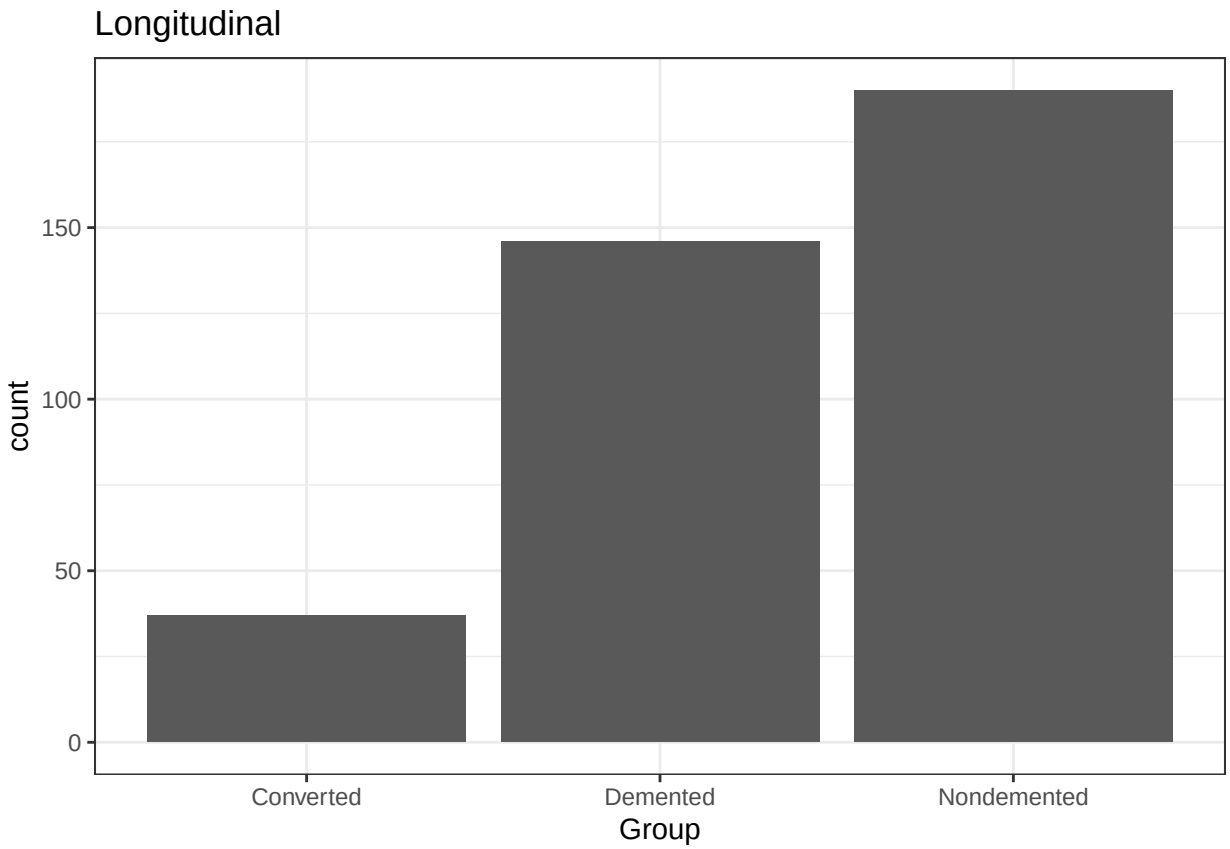
```
par(mfrow=c(1,2))

#Generamos las tablas, con datos absolutos y relativos
table(oasis_longitudinal$Group)
```

```
##
##   Converted    Demented Nondemented
##          37         146         190
```

```
round(prop.table(table(oasis_longitudinal$Group)), 2)
```

```
##
##   Converted    Demented Nondemented
##        0.10        0.39        0.51
```

```
#Y generamos el gráfico
ggplot(data = oasis_longitudinal, aes(x = Group, y = ..count.., )) +
  geom_bar() +
  labs(title = "Longitudinal") +
  theme_bw() +
  theme(legend.position = "bottom")
```




```
#Hacemos lo mismo con el estudio seccional
table(oasis_cross_sectional$CDR)
```

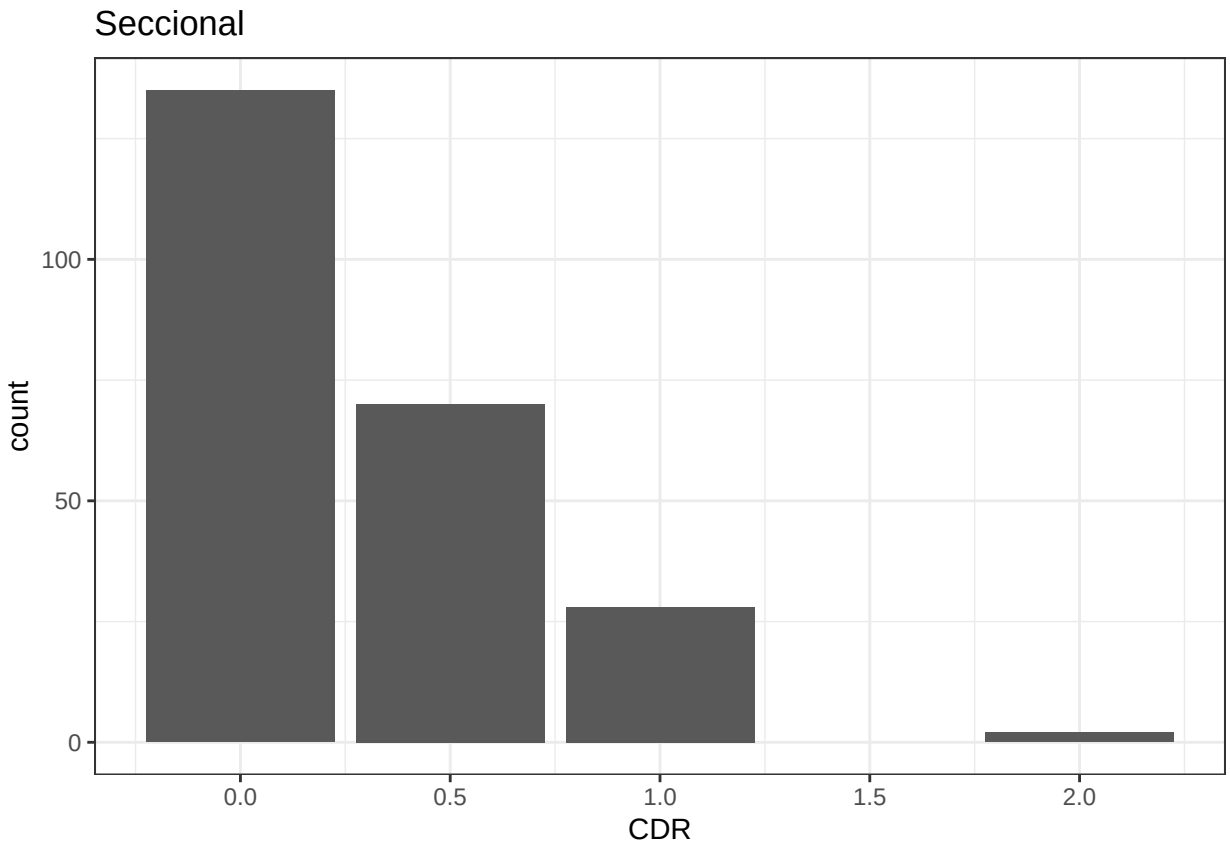
```
##
##   0 0.5   1   2
## 135  70  28   2
```

```
round(prop.table(table(oasis_cross_sectional$CDR)), 2)
```

```
##
##    0  0.5    1    2
## 0.57 0.30 0.12 0.01
```

```
ggplot(data = oasis_cross_sectional, aes(x = CDR, y = ..count.., ),) +
  geom_bar() +
  labs(title = "Seccional") +
  theme_bw() +
  theme(legend.position = "bottom")
```

```
## Warning: Removed 201 rows containing non-finite values (stat_count).
```



Como vemos el porcentaje de converted que son aquellos que al principio del experimento no tenían demencia y en las sucesivas medidas la desarrollaron, es del 10% de los datos, estos es importante conocerlo, si desamos crear un modelo efctivo es importante que acierte más del 10% de converted, que podría acertarse si simplemente calsificamos todos los sujetos como converted.

En el caso del estudio seccional lo hemos dividido en grupos según la variable CDR que muestra si se no se tiene demencia, y también vemos una distribución desigual, donde los pacientes con demencia son menos que los que no tienen demencia, sobre todo aquellos que tienen demencia leve y pacientes con demencia moderada apenas hay.

Antes de continuar vamos a eliminar los valores de CDR=2, ya que nos interesa detectar la demencia en estado leve y muy leve por tanto queremos solo los valores de CDR<2, así clasificaremos personas sin demencia por un lado y con demencia en estado muy leve y leve. De esta forma también nos quitaremos los valores ausentes que haya en nuestra variable respuesta, que no ibamos a poder imputarlos.

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
oasis_cross_sectional_2=filter(oasis_cross_sectional, CDR<2)
```

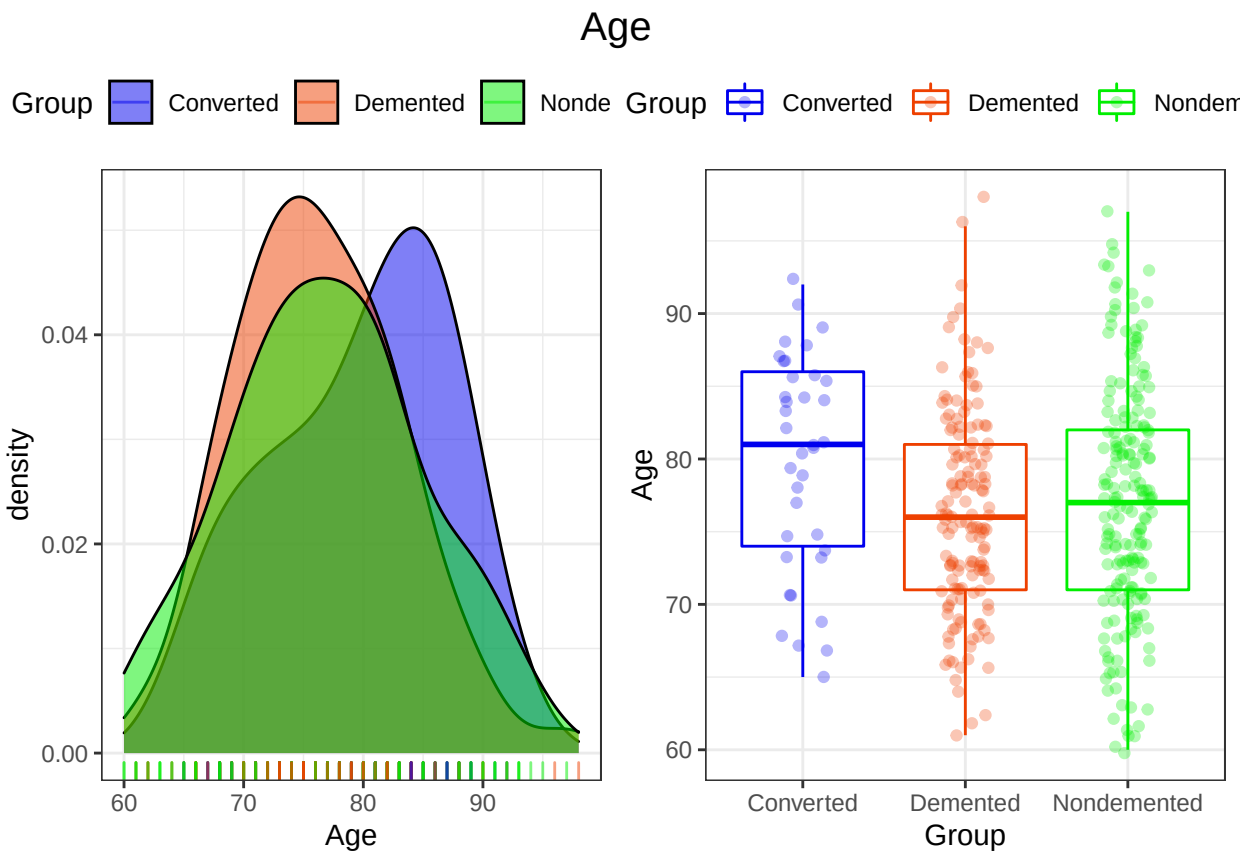
Distribución de las variables predictoras

Tras ver la distribución de la variable respuesta, nos interesa conocer la de las variables predictoras

```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.0.3
```

```
#Creamos un gráfico para ver la distribución de la variable Age
p1 <- ggplot(data = oasis_longitudinal, aes(x = Age, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("blue2", "orangered2", "green2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
p2 <- ggplot(data = oasis_longitudinal, aes(x = Group, y = Age, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("Age", size = 15))
final_plot
```




```
#Calculamos la media y la mediana
tapply(oasis_longitudinal$Age, oasis_longitudinal$Group, mean)
```

```
##    Converted    Demented Nondemented
##     79.75676    76.26027    77.05789
```

```
tapply(oasis_longitudinal$Age, oasis_longitudinal$Group, median)
```

```
##    Converted    Demented Nondemented
##           81          76          77
```

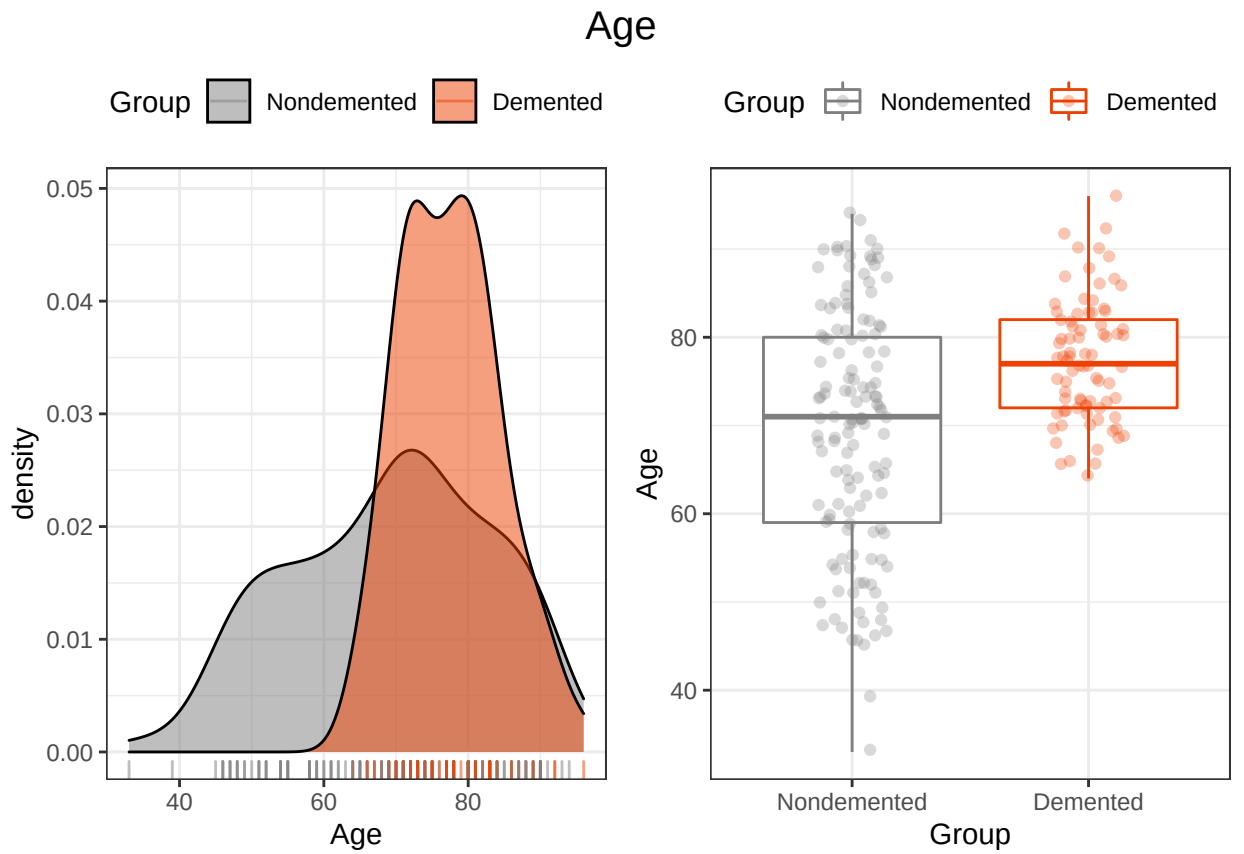
```
levels(oasis_cross_sectional$CDR)
```

```
## NULL
```

```
oasis_cross_sectional_2$Group=factor(oasis_cross_sectional_2$CDR,levels=c(0,0.5,1),
                          labels=c("Nondemented", "Demented", "Demented"))

oasis_cross_sectional_narm=na.omit(oasis_cross_sectional_2)
nrow(oasis_cross_sectional_narm)
```

```
## [1] 214
```

```
p1 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Age, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("gray50", "orangered2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
p2 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Group, y = Age, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("Age", size = 15))
final_plot
```




```
#Calculamos la media y la mediana
tapply(oasis_cross_sectional_narm$Age, oasis_cross_sectional_narm$Group, mean)
```

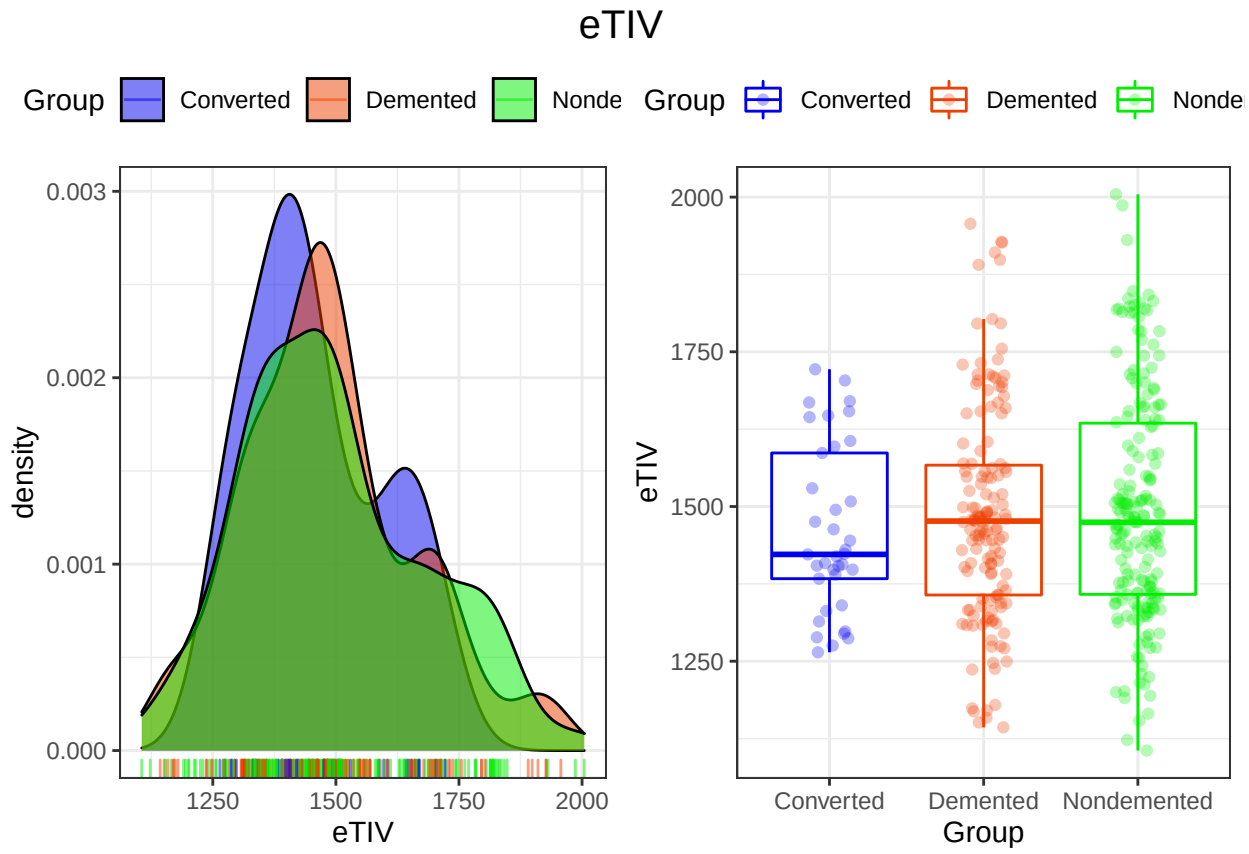
```
## Nondemented    Demented
##    69.23308    77.48148
```

```
tapply(oasis_cross_sectional_narm$Age, oasis_cross_sectional_narm$Group, median)
```

```
## Nondemented    Demented
##          71          77
```

En el estudio longitudinal el grupo converted tiene una edad media significativamente más baja que las otras dos variables, al igual que pasa con la mediana. Mientras que en el estudio seccional la media y mediana son mayores en el grupo con demencia.

```
#Volumen intracraneal
p1 <- ggplot(data = oasis_longitudinal, aes(x = eTIV, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("blue2", "orangered2", "green2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
p2 <- ggplot(data = oasis_longitudinal, aes(x = Group, y = eTIV, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("eTIV", size = 15))
final_plot
```



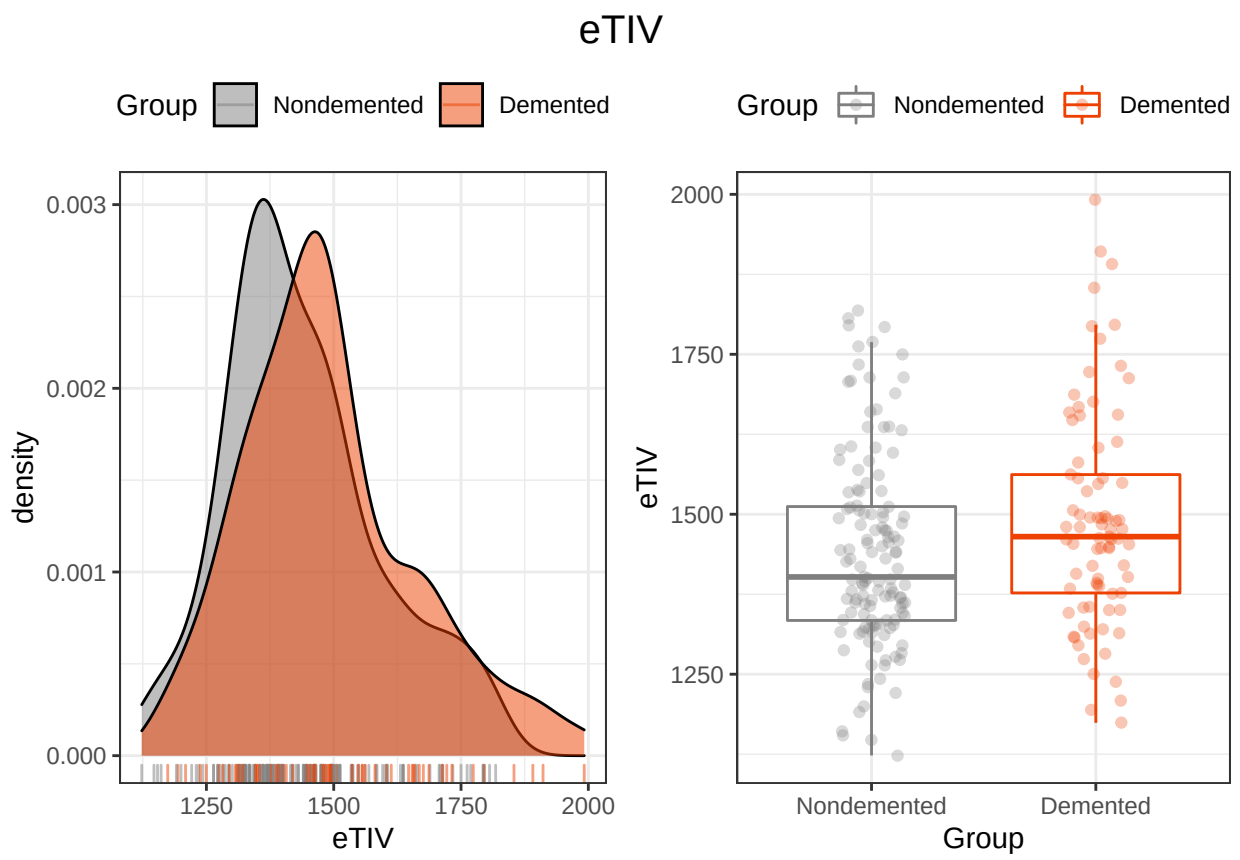

```
tapply(oasis_longitudinal$eTIV, oasis_longitudinal$Group, mean)
```

```
##    Converted     Demented Nondemented
##     1459.347     1485.848     1495.472
```

```r
tapply(oasis_longitudinal$eTIV, oasis_longitudinal$Group, median)
```

```
##   Converted    Demented Nondemented
##    1422.623    1476.460    1474.505
```

```r
p1 <- ggplot(data = oasis_cross_sectional_narm, aes(x = eTIV, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("gray50", "orangered2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
p2 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Group, y = eTIV, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("eTIV", size = 15))
final_plot
```




```r
#Calculamos la media y la mediana
tapply(oasis_cross_sectional_narm$eTIV, oasis_cross_sectional_narm$Group, mean)
```

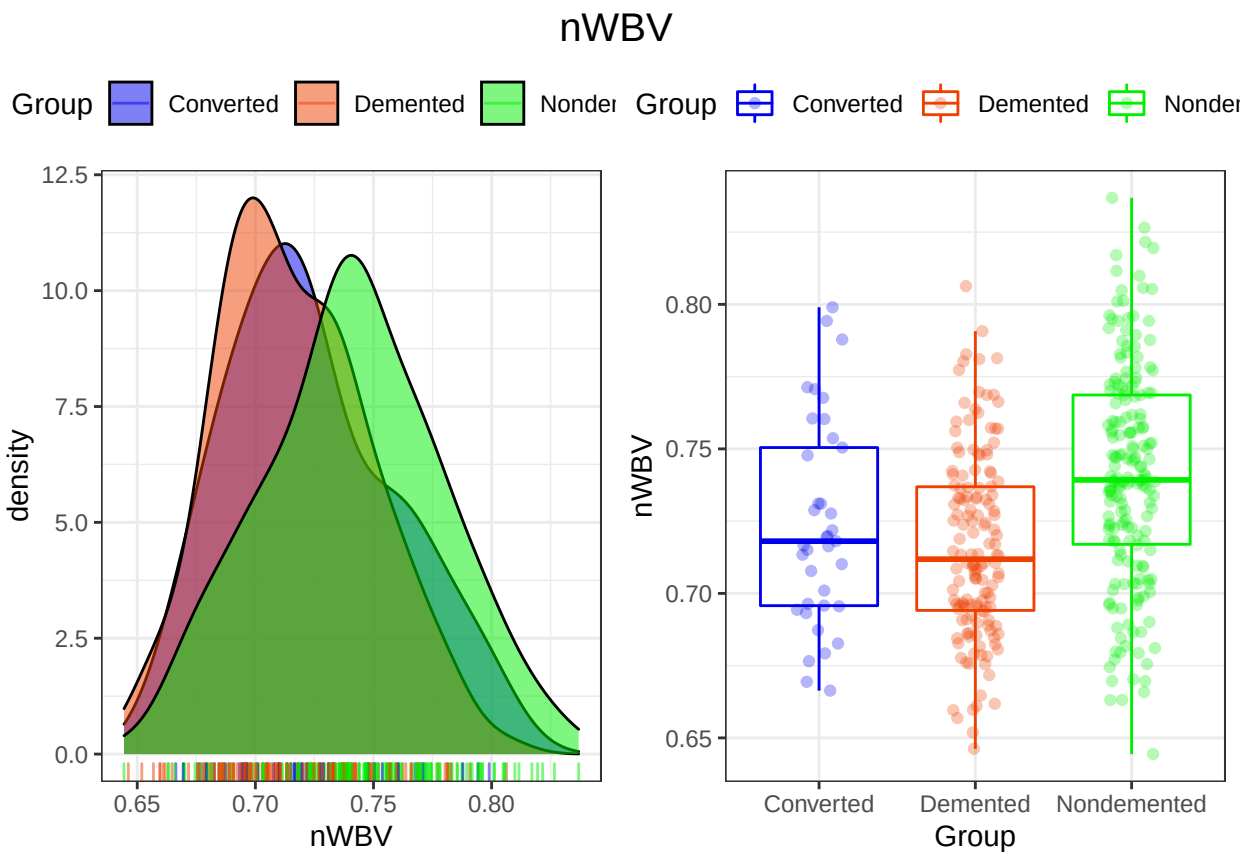
```
## Nondemented    Demented
##    1439.444    1490.210
```

```
tapply(oasis_cross_sectional_narm$eTIV, oasis_cross_sectional_narm$Group, median)
```

```
## Nondemented    Demented
##        1402        1465
```

En esta variable el grupo converted tiene una mediana significativamente más baja que las otras dos, en la media en cambio no hay tanta diferencia. La variable Nondemented tiene una media por encima de Demented, en cambio una mediana por debajo. En los datos seccionales no hay demasiada diferencia entre grupos.

```
#Volumen total normalizado
p1 <- ggplot(data = oasis_longitudinal, aes(x = nWBV, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("blue2", "orangered2", "green2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
p2 <- ggplot(data = oasis_longitudinal, aes(x = Group, y = nWBV, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("nWBV", size = 15))
final_plot
```



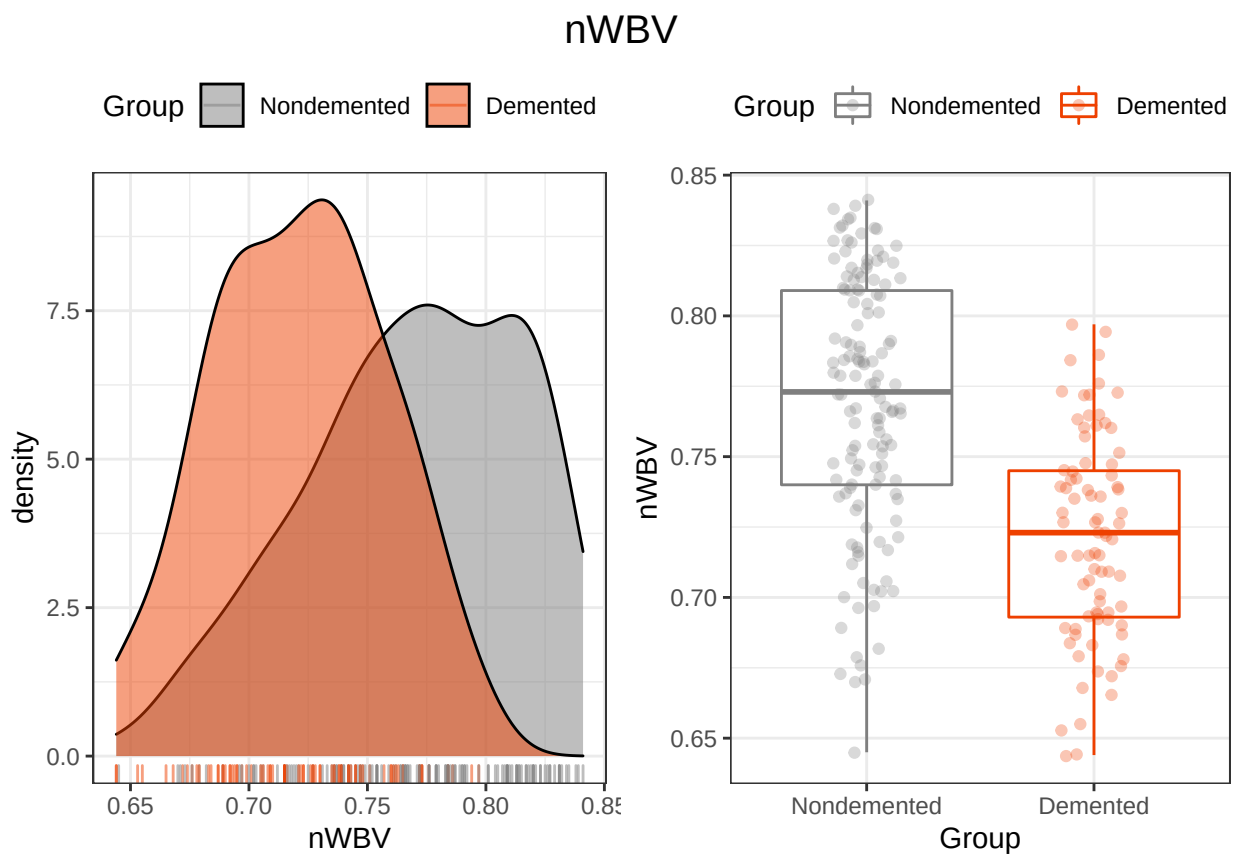
```
tapply(oasis_longitudinal$nWBV, oasis_longitudinal$Group, mean)
```

```
##    Converted    Demented Nondemented
##    0.7237336   0.7163034   0.7408726
```

```
tapply(oasis_longitudinal$nWBV, oasis_longitudinal$Group, median)
```

```
##    Converted    Demented Nondemented
##    0.7180650   0.7118355   0.7392630
```

```
p1 <- ggplot(data = oasis_cross_sectional_narm, aes(x = nWBV, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("gray50", "orangered2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
p2 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Group, y = nWBV, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("nWBV", size = 15))
final_plot
```



```r
#Calculamos la media y la mediana
tapply(oasis_cross_sectional_narm$nWBV, oasis_cross_sectional_narm$Group, mean)
```

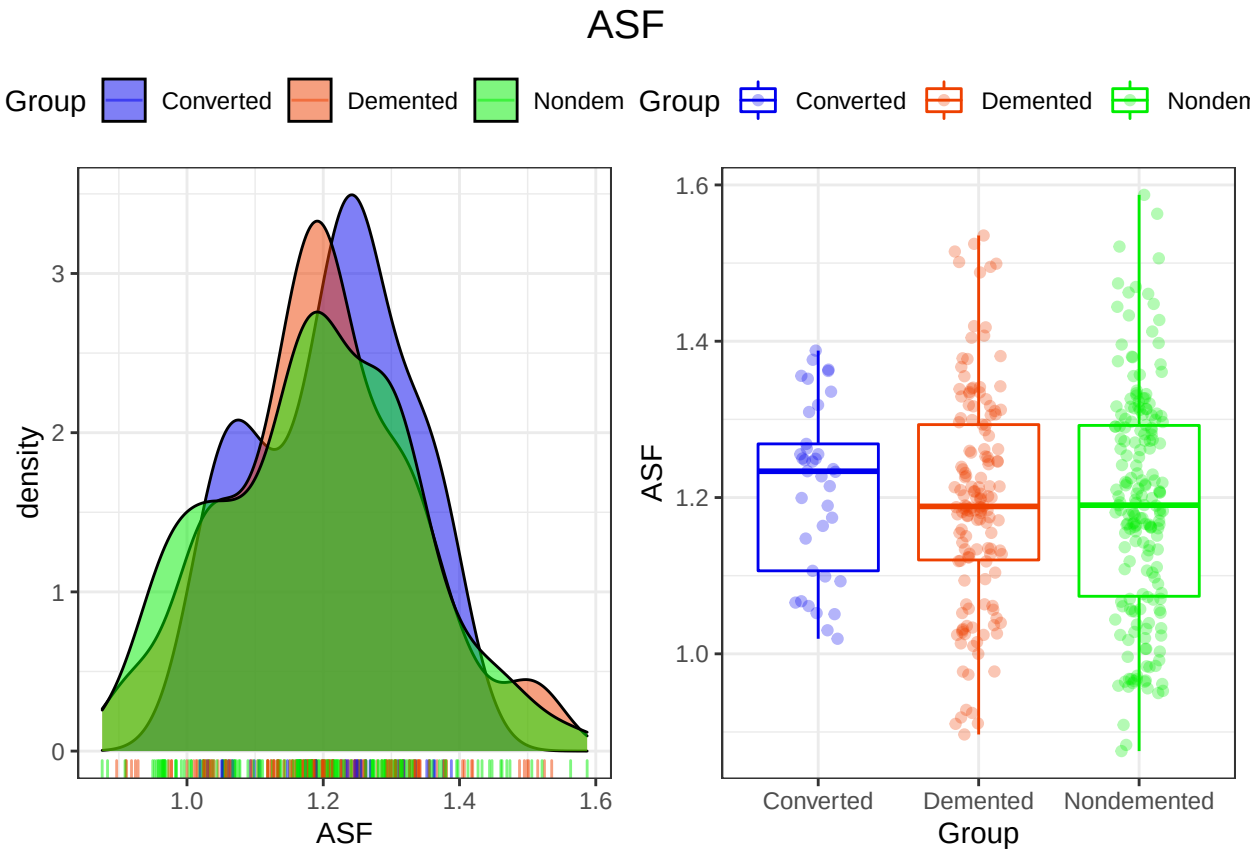
```
## Nondemented    Demented
##   0.7691880   0.7214568
```

```r
tapply(oasis_cross_sectional_narm$nWBV, oasis_cross_sectional_narm$Group, median)
```

```
## Nondemented    Demented
##       0.773       0.723
```

En esta variabe sí se ven diferencias entre los grupos, los no dementes tienen valores en promedio más altos, que los dementes y los converted. En este caso la diferencia entre grupos es más significativa, en los datos seccionales.

```r
#Atlas Scaling Factor
p1 <- ggplot(data = oasis_longitudinal, aes(x = ASF, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("blue2", "orangered2", "green2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
p2 <- ggplot(data = oasis_longitudinal, aes(x = Group, y = ASF, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("ASF", size = 15))
final_plot
```



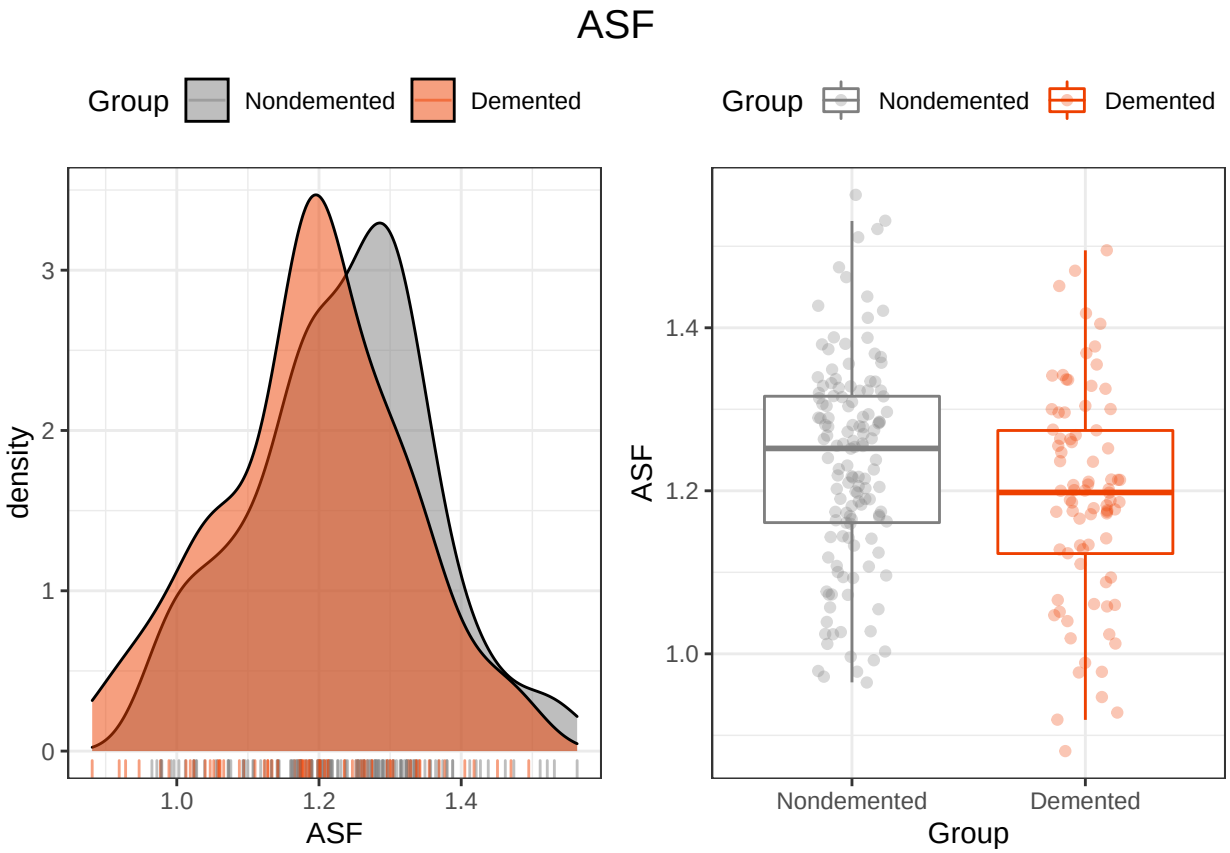

```
tapply(oasis_longitudinal$ASF, oasis_longitudinal$Group, mean)
```

```
##   Converted    Demented Nondemented
##    1.212422    1.196880    1.191066
```

```
tapply(oasis_longitudinal$ASF, oasis_longitudinal$Group, median)
```

```
##   Converted    Demented Nondemented
##    1.233637    1.188655    1.190225
```

```
p1 <- ggplot(data = oasis_cross_sectional_narm, aes(x = ASF, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("gray50", "orangered2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
p2 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Group, y = ASF, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("ASF", size = 15))
final_plot
```




```
#Calculamos la media y la mediana
tapply(oasis_cross_sectional_narm$ASF, oasis_cross_sectional_narm$Group, mean)
```

```
## Nondemented    Demented
##    1.232474    1.192309
```

```
tapply(oasis_cross_sectional_narm$ASF, oasis_cross_sectional_narm$Group, median)
```

```
## Nondemented    Demented
##       1.252       1.198
```

En este caso no parece haber diferencias en la distribución.

```
#MMSE (Test de deterioro cognitivo)

p1 <- ggplot(data = oasis_longitudinal, aes(x = MMSE, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("blue2", "orangered2", "green2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
p2 <- ggplot(data = oasis_longitudinal, aes(x = Group, y = MMSE, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
```

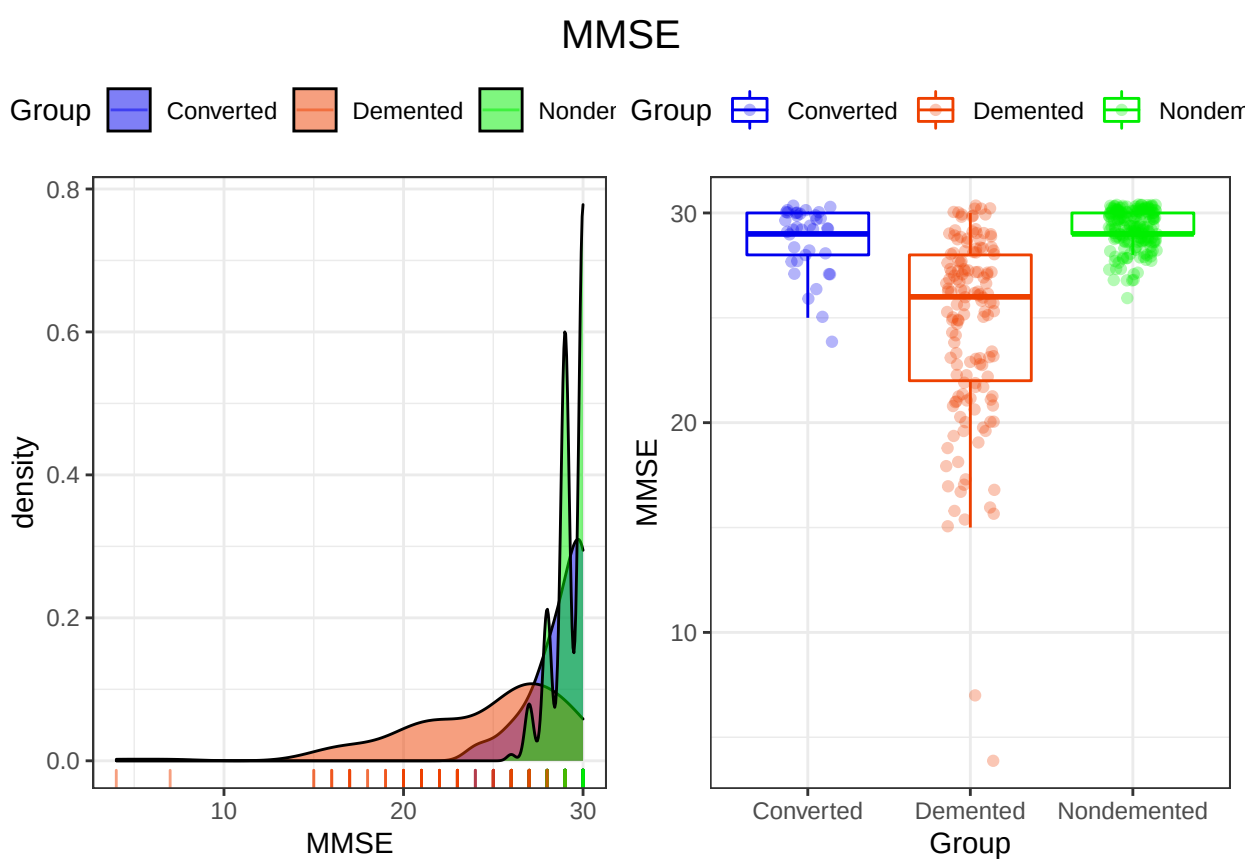
```
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
```

```
## Warning: Removed 2 rows containing non-finite values (stat_density).
```

```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```

```
final_plot <- annotate_figure(final_plot, top = text_grob("MMSE", size = 15))
final_plot
```



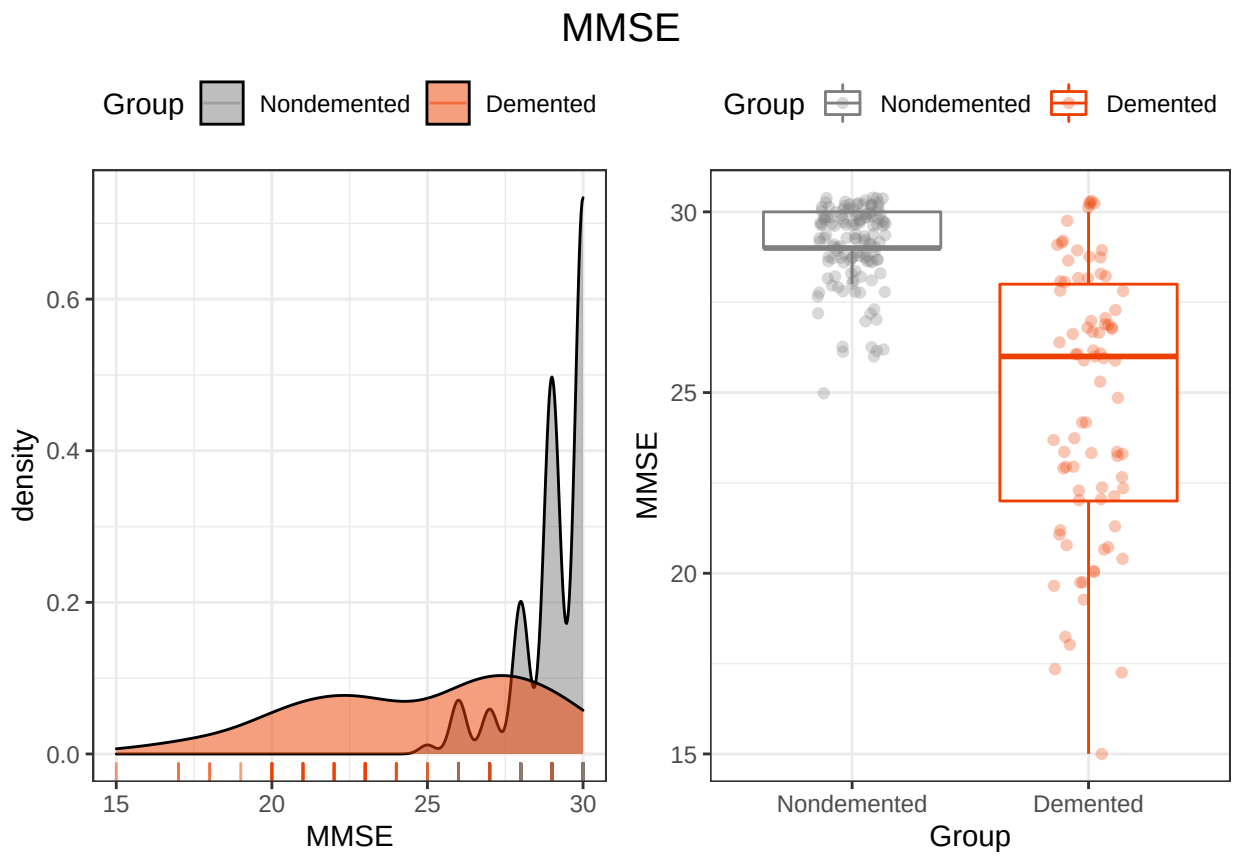

```
tapply(oasis_longitudinal$MMSE, oasis_longitudinal$Group, na.rm= TRUE, mean)
```

```
##    Converted     Demented Nondemented
##     28.67568     24.51389     29.22632
```

```
tapply(oasis_longitudinal$MMSE, oasis_longitudinal$Group, na.rm= TRUE, median)
```

```
##    Converted     Demented Nondemented
##           29           26           29
```

```
p1 <- ggplot(data = oasis_cross_sectional_narm, aes(x = MMSE, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("gray50", "orangered2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
p2 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Group, y = MMSE, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("MMSE", size = 15))
final_plot
```




```
#Calculamos la media y la mediana
tapply(oasis_cross_sectional_narm$MMSE, oasis_cross_sectional_narm$Group, mean)
```

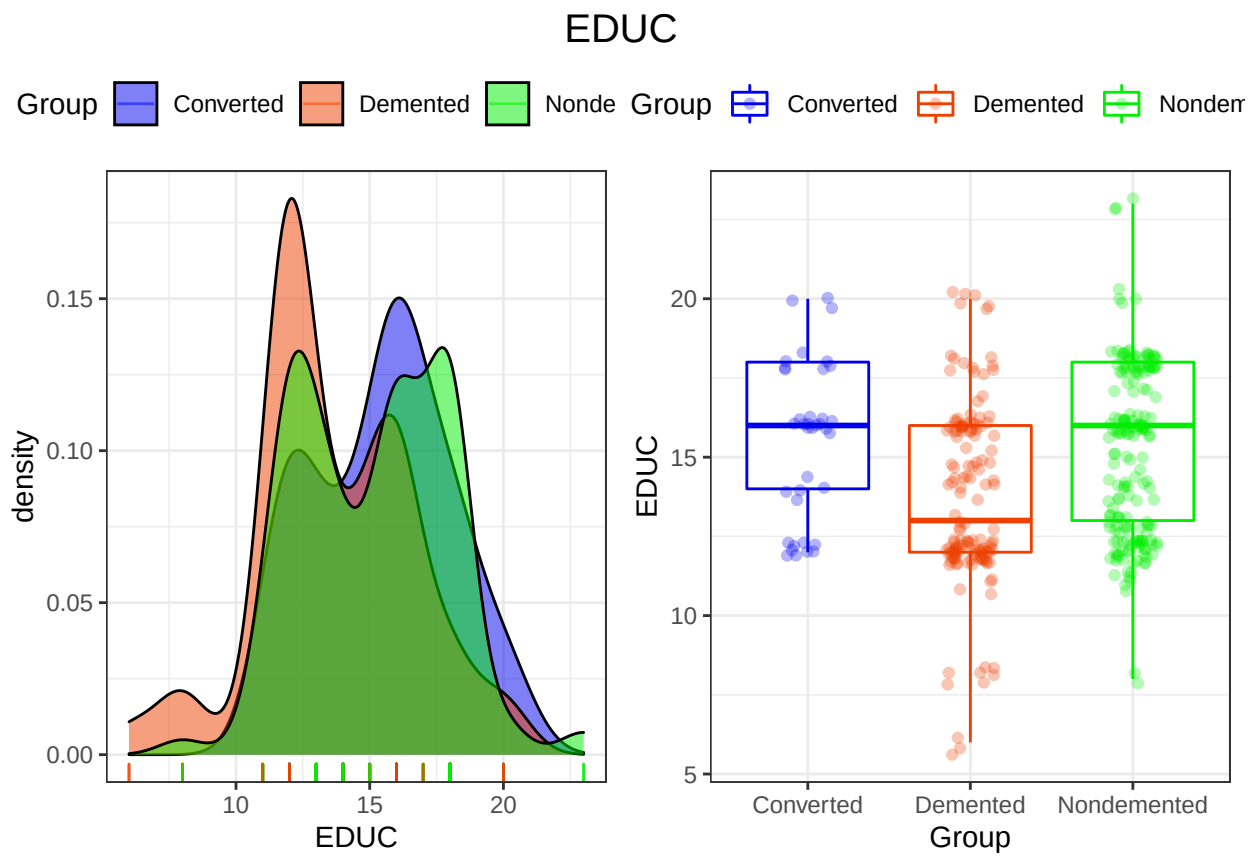
```
## Nondemented    Demented
##    29.09774    24.71605
```

```
tapply(oasis_cross_sectional_narm$MMSE, oasis_cross_sectional_narm$Group, median)
```

```
## Nondemented    Demented
##          29          26
```

En este caso se ven diferencias entre los que tienen demencia y los otros dos grupos, pero no entre converted y nondemented. En el caso de los datos seccionales hay una diferencia significativa entre los grupos.

```
#Nivel de estudios
p1 <- ggplot(data = oasis_longitudinal, aes(x = EDUC, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("blue2", "orangered2", "green2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
p2 <- ggplot(data = oasis_longitudinal, aes(x = Group, y = EDUC, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("blue2", "orangered2", "green2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("EDUC", size = 15))
final_plot
```



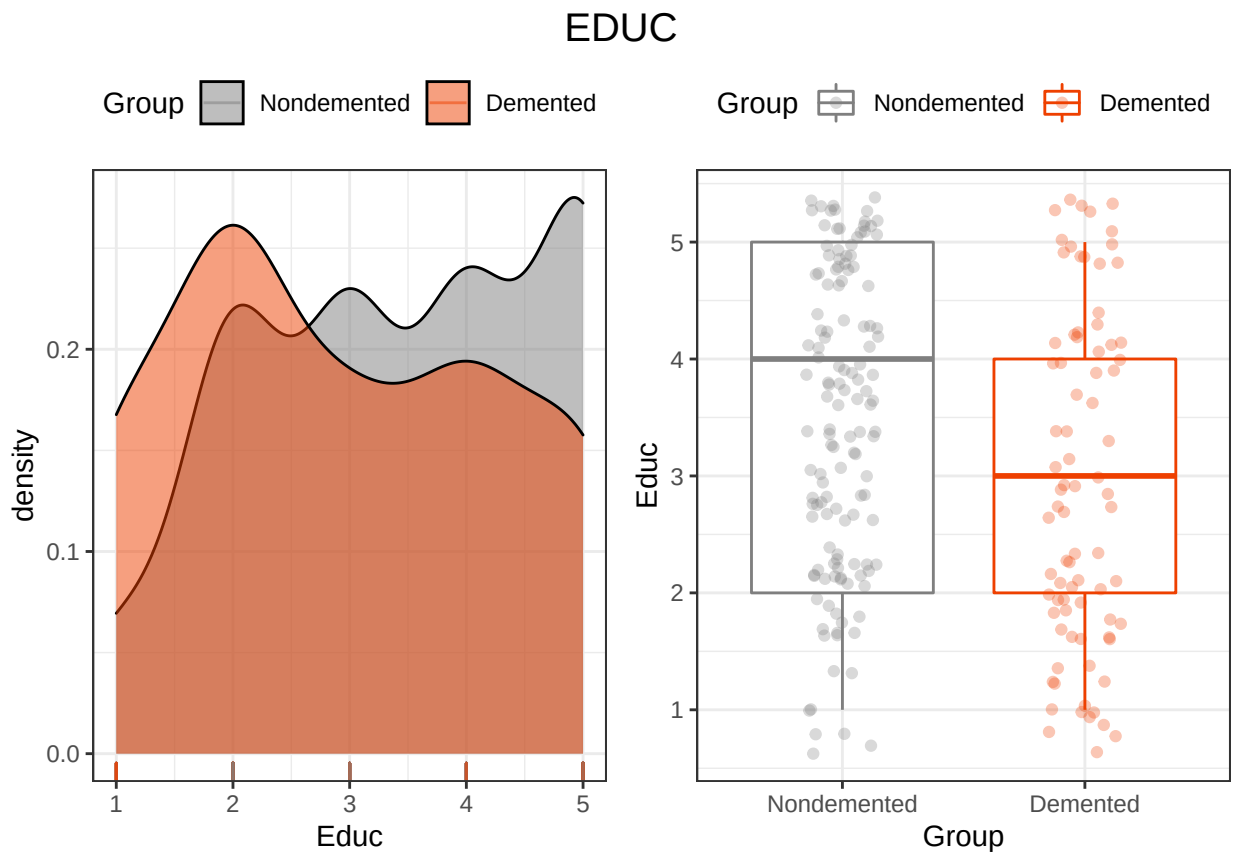

```
tapply(oasis_longitudinal$EDUC, oasis_longitudinal$Group, na.rm= TRUE, mean)
```

```
##    Converted    Demented Nondemented
##     15.45946    13.67123    15.14211
```

```
tapply(oasis_longitudinal$EDUC, oasis_longitudinal$Group, na.rm= TRUE, median)
```

```
##    Converted    Demented Nondemented
##           16          13          16
```

```
p1 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Educ, fill = Group)) +
      geom_density(alpha = 0.5) +
      scale_fill_manual(values = c("gray50", "orangered2")) +
      geom_rug(aes(color = Group), alpha = 0.5) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
p2 <- ggplot(data = oasis_cross_sectional_narm, aes(x = Group, y = Educ, color = Group)) +
      geom_boxplot(outlier.shape = NA) +
      geom_jitter(alpha = 0.3, width = 0.15) +
      scale_color_manual(values = c("gray50", "orangered2")) +
      theme_bw()
final_plot <- ggarrange(p1, p2, legend = "top")
final_plot <- annotate_figure(final_plot, top = text_grob("EDUC", size = 15))
final_plot
```




```
#Calculamos la media y la mediana
tapply(oasis_cross_sectional_narm$Educ, oasis_cross_sectional_narm$Group, mean)
```

```
## Nondemented    Demented
##    3.443609    2.913580
```

```
tapply(oasis_cross_sectional_narm$Educ, oasis_cross_sectional_narm$Group, median)
```

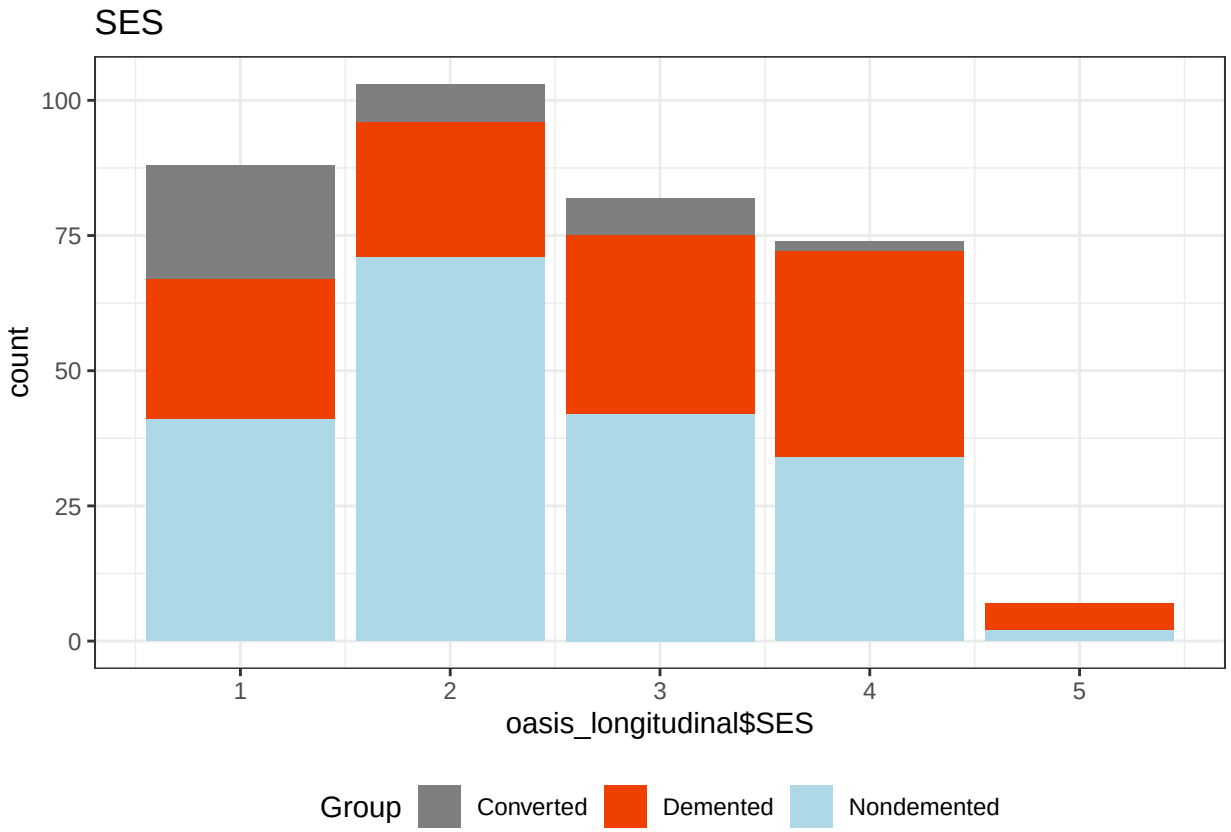
```
## Nondemented    Demented
##           4           3
```

En esta variable también se ven diferencias entre los sujetos con demencia y los otros dos grupos. En el caso de los datos seccionales también se ven diferencias.

```
#Status socioeconómico
ggplot(data = oasis_longitudinal, aes(x = oasis_longitudinal$SES, y = ..count.., fill = Group)) +
  geom_bar() +
  labs(title = "SES") +
  scale_fill_manual(values = c("gray50", "orangered2", "lightblue")) +
  theme_bw() +
  theme(legend.position = "bottom")
```

```
## Warning: Use of ‘oasis_longitudinal$SES‘ is discouraged. Use ‘SES‘ instead.
```

```
## Warning: Removed 19 rows containing non-finite values (stat_count).
```

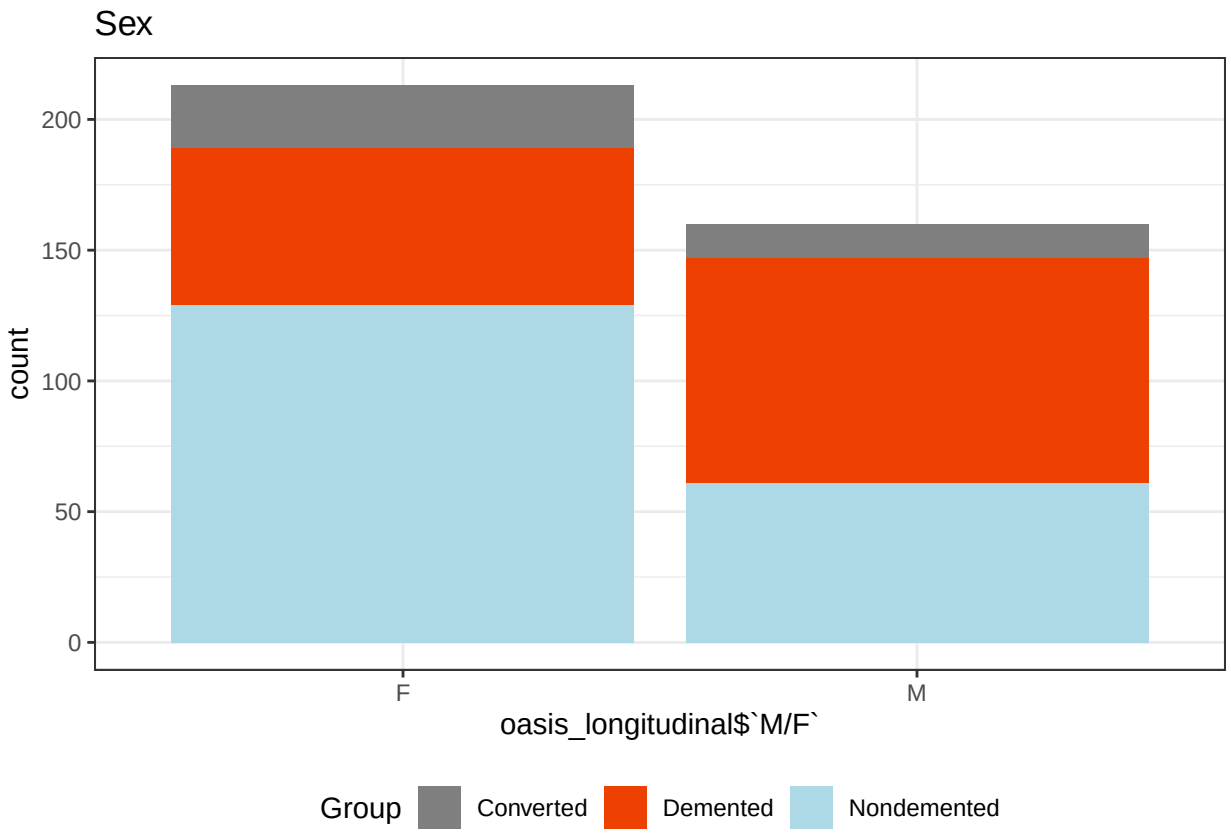



```
round(prop.table(table(oasis_longitudinal$SES, oasis_longitudinal$Group)),2)
```

```
##
##      Converted Demented Nondemented
##   1       0.06     0.07        0.12
##   2       0.02     0.07        0.20
##   3       0.02     0.09        0.12
##   4       0.01     0.11        0.10
##   5       0.00     0.01        0.01
```

```
#Sexo
ggplot(data = oasis_longitudinal, aes(x = oasis_longitudinal$`M/F`, y = ..count.., fill = Group)) +
  geom_bar() +
  labs(title = "Sex") +
  scale_fill_manual(values = c("gray50", "orangered2", "lightblue")) +
  theme_bw() +
  theme(legend.position = "bottom")
```

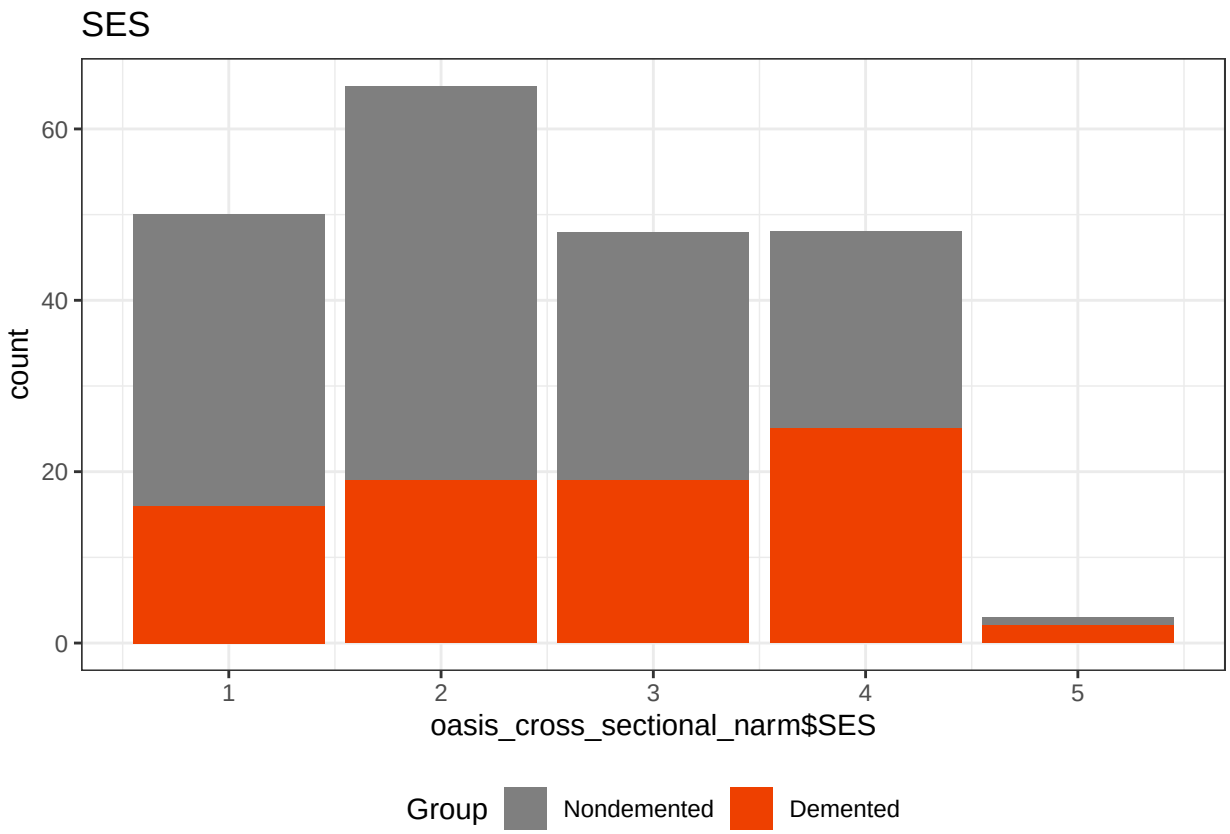
```
## Warning: Use of `oasis_longitudinal$`M/F`` is discouraged. Use `M/F` instead.
```




```
round(prop.table(table(oasis_longitudinal$`M/F`, oasis_longitudinal$Group)),2)
```

```
##
##      Converted Demented Nondemented
##   F       0.06     0.16        0.35
##   M       0.03     0.23        0.16
```

```r
#Status socioeconómico
ggplot(data = oasis_cross_sectional_narm, aes(x = oasis_cross_sectional_narm$SES, y = ..count.., fill =
  geom_bar() +
  labs(title = "SES") +
  scale_fill_manual(values = c("gray50", "orangered2")) +
  theme_bw() +
  theme(legend.position = "bottom")
```

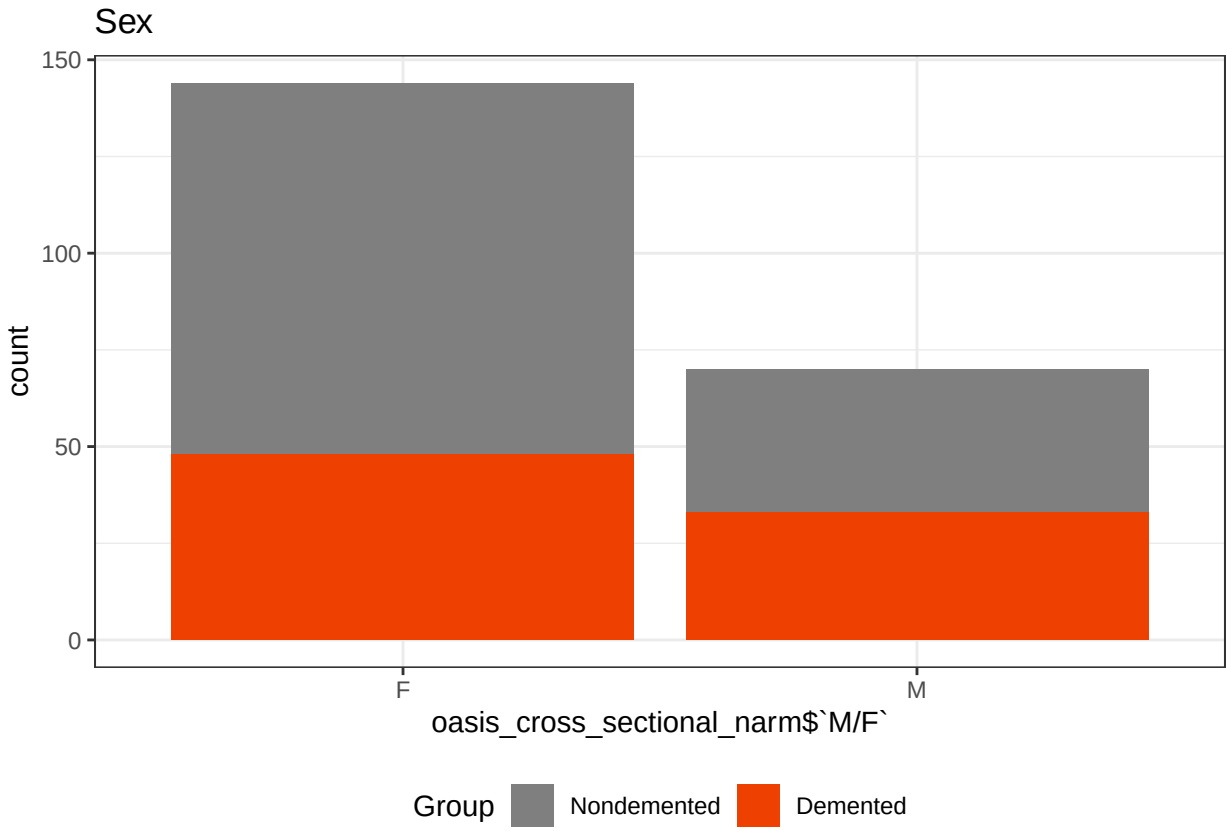



```r
round(prop.table(table(oasis_cross_sectional_narm$SES, oasis_cross_sectional_narm$Group)),2)
```

```
##
##     Nondemented Demented
##   1        0.16     0.07
##   2        0.21     0.09
##   3        0.14     0.09
##   4        0.11     0.12
##   5        0.00     0.01
```

```r
#Sexo
oasis_cross_sectional_narm$`M/F`=oasis_cross_sectional_narm$M.F
ggplot(data = oasis_cross_sectional_narm, aes(x = oasis_cross_sectional_narm$`M/F`, y = ..count.., fill
  geom_bar() +
  labs(title = "Sex") +
  scale_fill_manual(values = c("gray50", "orangered2", "lightblue")) +
```

```
  theme_bw() +
  theme(legend.position = "bottom")
```




```
round(prop.table(table(oasis_cross_sectional_narm$`M/F`, oasis_cross_sectional_narm$Group)),2)
```

```
##
##     Nondemented Demented
##   F        0.45     0.22
##   M        0.17     0.15
```

En este caso en los datos longitudinales parece haber diferencias en la variable M/F pero no en la SES, mientras que en los seccionales parece haber diferencias en ambas.

Random forest

Para terminar el apartado del análisis exploratorio, y complementar el último punto, vamos a realizar un análisis random forest con el que descubriremos que variables predicen mejor la variable respuesta:

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.4
```

```
## randomForest 4.6-14
```

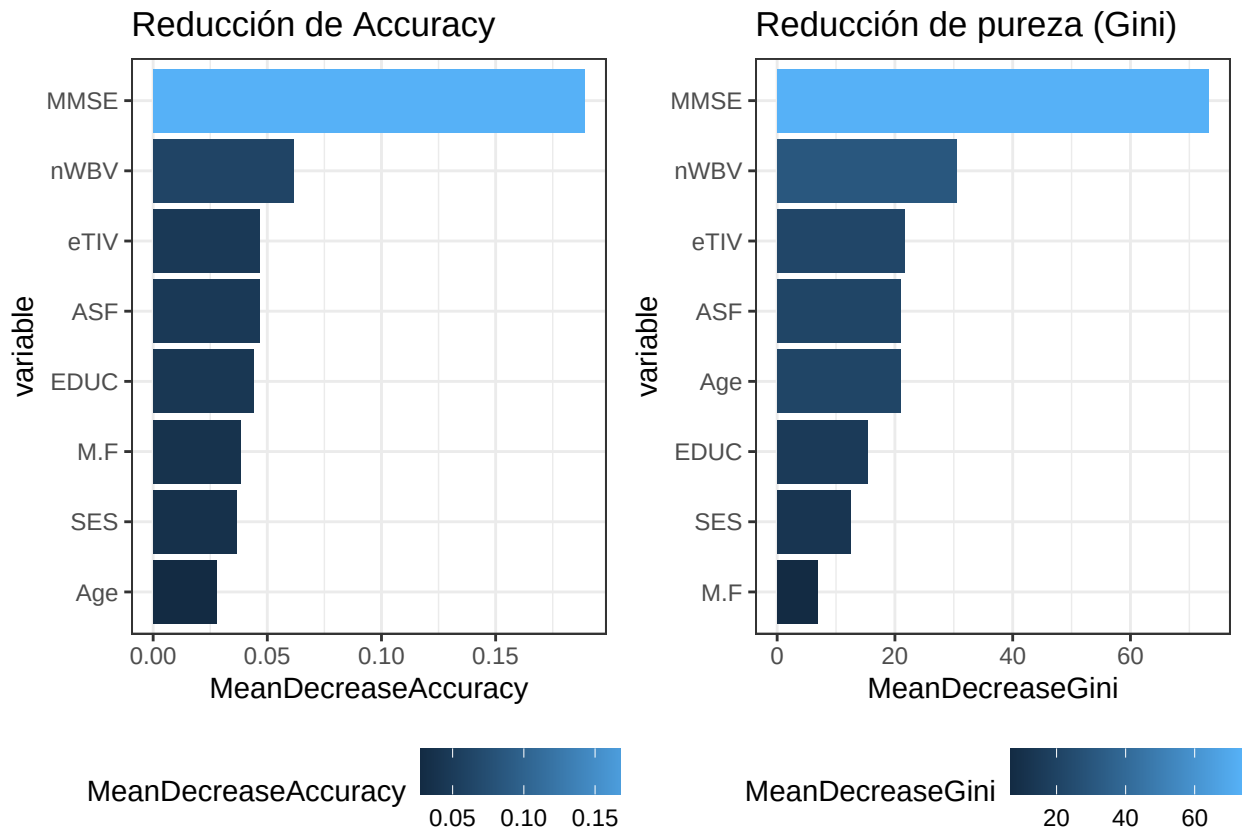
```
## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

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

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.3

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v tibble  3.0.3     v purrr   0.3.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## Warning: package 'tidyr' was built under R version 4.0.3

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x randomForest::combine() masks dplyr::combine()
## x dplyr::filter()         masks stats::filter()
## x dplyr::lag()            masks stats::lag()
## x randomForest::margin()  masks ggplot2::margin()
```

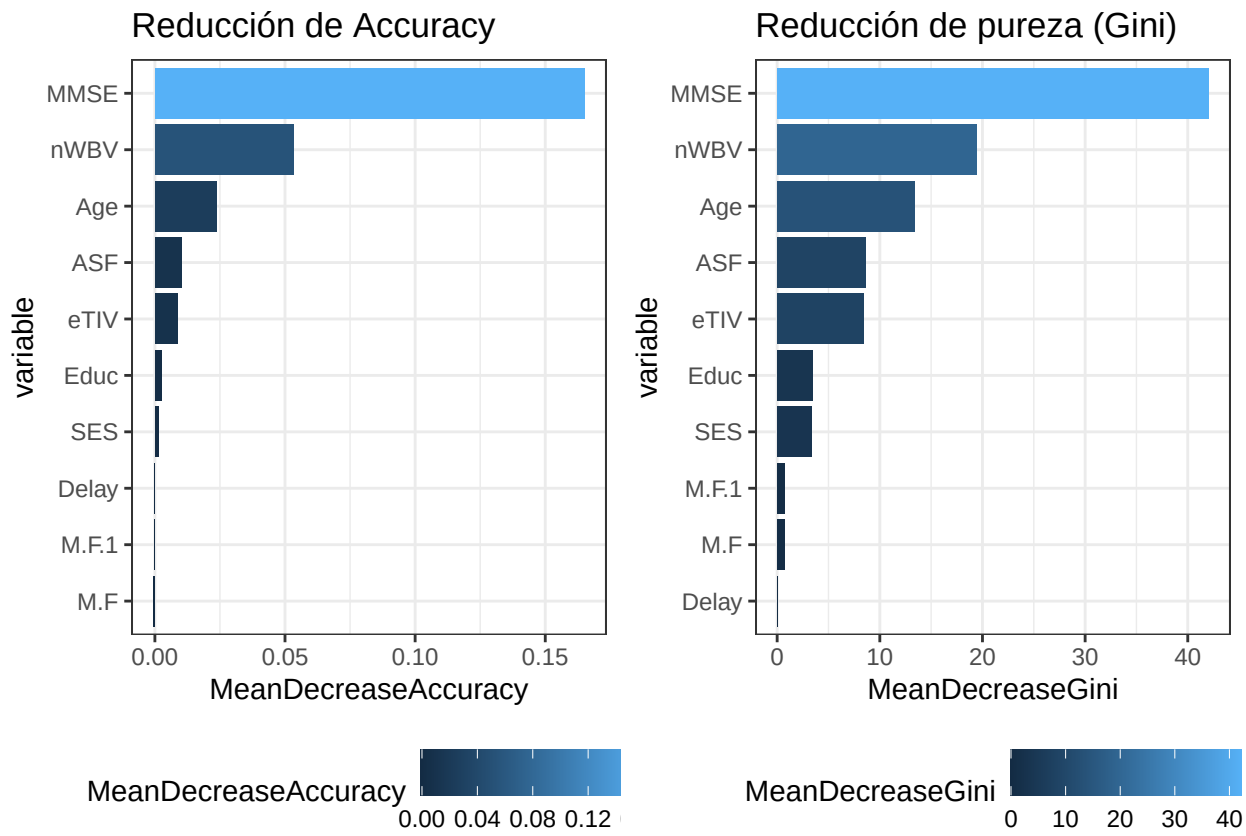
```
#Seleccionamos las variables y generamos el análisis random forest
datos_rf <- oasis_longitudinal %>%
            select(-`Subject ID`, -`MRI ID`, -`MR Delay`, -Visit, -Hand, -CDR) %>% na.omit()
datos_rf <- map_if(.x = datos_rf, .p = is.character, .f = as.factor) %>% as.data.frame()
modelo_randforest <- randomForest(formula = Group ~ . ,
                                  data = na.omit(datos_rf),
                                  mtry = 5,
                                  importance = TRUE,
                                  ntree = 1000)
importancia <- as.data.frame(modelo_randforest$importance)
importancia <- rownames_to_column(importancia,var = "variable")

p1 <- ggplot(data = importancia, aes(x = reorder(variable, MeanDecreaseAccuracy),
                                  y = MeanDecreaseAccuracy,
                                  fill = MeanDecreaseAccuracy)) +
      labs(x = "variable", title = "Reducción de Accuracy") +
      geom_col() +
      coord_flip() +
      theme_bw() +
      theme(legend.position = "bottom")
```

```
p2 <- ggplot(data = importancia, aes(x = reorder(variable, MeanDecreaseGini),
                                     y = MeanDecreaseGini,
                                     fill = MeanDecreaseGini)) +
      labs(x = "variable", title = "Reducción de pureza (Gini)") +
      geom_col() +
      coord_flip() +
      theme_bw() +
      theme(legend.position = "bottom")
ggarrange(p1, p2)
```




```
datos_rf <- oasis_cross_sectional_narm %>%
            select(-ID, -Hand, -CDR) %>% na.omit()
datos_rf <- map_if(.x = datos_rf, .p = is.character, .f = as.factor) %>% as.data.frame()
modelo_randforest <- randomForest(formula = Group ~ . ,
                                  data = na.omit(datos_rf),
                                  mtry = 5,
                                  importance = TRUE,
                                  ntree = 1000)
importancia <- as.data.frame(modelo_randforest$importance)
importancia <- rownames_to_column(importancia,var = "variable")

p1 <- ggplot(data = importancia, aes(x = reorder(variable, MeanDecreaseAccuracy),
                                     y = MeanDecreaseAccuracy,
                                     fill = MeanDecreaseAccuracy)) +
      labs(x = "variable", title = "Reducción de Accuracy") +
```

```
      geom_col() +
      coord_flip() +
      theme_bw() +
      theme(legend.position = "bottom")

p2 <- ggplot(data = importancia, aes(x = reorder(variable, MeanDecreaseGini),
                                     y = MeanDecreaseGini,
                                     fill = MeanDecreaseGini)) +
      labs(x = "variable", title = "Reducción de pureza (Gini)") +
      geom_col() +
      coord_flip() +
      theme_bw() +
      theme(legend.position = "bottom")
ggarrange(p1, p2)
```



Este análisis apunta a que las mejores variables para predecir la demencia son MMSE y nWBV.

Preprocesamiento

Tratamiento de los valores ausentes

Como vimos en la exploración de los datos hay valores ausentes, rincipalmente concentrados en las variables SES, EDUC y MMSE. Antes de seguir habría que eliminarlos ya que hay algoritmos de ML que no admiten estos valores. Tenemos dos opciones, eliminar las variables con valores ausentes o eliminar las observaciones

con valores ausentes. El del conjunto seccional ya lo habíamos eliminado para visualizar la distribución de las variables así que solo lo haremos con el longitudinal.

También vamos a eliminar del conjunto longitudinal las observaciones "converted", ya que queremos solo demencia y no demencia para hacer una clasificación y nos quedaremos solo con una observación por individuo.

```
# Eliminamos los NAs
oasis_longitudinal_narm=na.omit(oasis_longitudinal)
# Eliminamos las observaciones "Converted" y escogemos solo la primera visita de cada paciente
library(dplyr)
oasis_longitudinal_narm_2=filter(oasis_longitudinal_narm, Group!="Converted", Visit==1)
table(oasis_longitudinal_narm_2$Group)
```

```
##
##    Demented Nondemented
##          56          72
```

```
table(oasis_cross_sectional_narm$Group)
```

```
##
## Nondemented    Demented
##         133          81
```

```
nrow(na.omit(oasis_cross_sectional_narm))
```

```
## [1] 214
```

```
nrow(na.omit(oasis_longitudinal_narm_2))
```

```
## [1] 128
```

Tenemos 214 observaciones en el seccional y 128 en el longitudinal. Ya que tenemos el doble en el seccional lo utilizaremos como datos para el entrenamiento y el longitudinal como datos para el test. Como vemos hay más datos Nondemented que demented.

Variables con varianza cercana 0

Otra parte importante del preprocesado será eliminar variables que no aporten nada, como vimos la variable Hand no se utilizará ya que no tiene diferentes niveles, pero además hay una forma de ser si las variables pueden no aortar información y es viendo si su varianza es igual o cercana a 0. Con la función `nearZeroVars`, podemos averiguar si alguna función tiene varianza cercana a 0.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.4
```

```
## Loading required package: lattice
```

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

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
oasis_longitudinal %>% select(Age, `M/F`, EDUC, SES, MMSE, CDR, eTIV, nWBV, ASF) %>% nearZeroVar(saveMe
```

```
##      freqRatio percentUnique zeroVar   nzv
## Age   1.181818     10.455764   FALSE FALSE
## M/F   1.331250      0.536193   FALSE FALSE
## EDUC  1.271605      3.217158   FALSE FALSE
## SES   1.170455      1.340483   FALSE FALSE
## MMSE  1.252747      4.825737   FALSE FALSE
## CDR   1.674797      1.072386   FALSE FALSE
## eTIV  1.000000     99.463807   FALSE FALSE
## nWBV  1.000000    100.000000   FALSE FALSE
## ASF   1.000000     99.463807   FALSE FALSE
```

Entre los predictores incluidos en el modelo, no se detecta ninguno con varianza cero o próxima a cero. Por lo que no vamos a eliminar por este motivo ninguna variable.

Normalización

La normalización es un paso importante para ajustar el modelo, y que la escala de las variables no de mayor peso a aquellas con números más grandes.

```
#Normalización estándar
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
  }

oasis_longitudinal_n <- as.data.frame(lapply(oasis_longitudinal_narm_2[8:15], normalize))

oasis_cross_sectional_n=as.data.frame(lapply(oasis_cross_sectional_narm[4:11], normalize))
```

Preparación de los datos de entrenamiento y test

Finalmente antes de aplicar el modelo vamos a preparar los datos, para tener dos conjuntos de entrenamiento y test, con las mismas variables, por lo que tendremos que ajustar ambos conjuntos para que tengan exactamente las mismas variables, y nombres de variables. También tendremos que cambiar aquellas variables que seán cualitativas, por numéricas ya que algunos modelos no aceptan variables cualitativas.

```
#Cambiamos el nombre de la variable educ para que esté igual en ambos conjuntos de datos
require(reshape)
```

```
## Loading required package: reshape
```

```
## Warning: package 'reshape' was built under R version 4.0.3
```

```
##
## Attaching package: 'reshape'

## The following objects are masked from 'package:tidyr':
##
##     expand, smiths

## The following object is masked from 'package:dplyr':
##
##     rename
```

```
oasis_longitudinal_n = rename(oasis_longitudinal_n, c(EDUC="Educ"))

#Tras normalizarlo añadimos la variable sexo al conjunto
oasis_longitudinal_n2=cbind(oasis_longitudinal_narm_2$`M/F`, oasis_longitudinal_n)
oasis_cross_sectional_n2=cbind(oasis_cross_sectional_narm$M.F, oasis_cross_sectional_n)

#Camiamos el nombre de esta variable a Sex
oasis_longitudinal_n3 = rename(oasis_longitudinal_n2, c("oasis_longitudinal_narm_2$`M/F`"="Sex"))
names(oasis_longitudinal_n3)
```

```
## [1] "Sex"  "Age"  "Educ" "SES"  "MMSE" "CDR"  "eTIV" "nWBV" "ASF"
```

```
oasis_cross_sectional_n3 = rename(oasis_cross_sectional_n2, c(`oasis_cross_sectional_narm$M.F`="Sex"))
names(oasis_cross_sectional_n3)
```

```
## [1] "Sex"  "Age"  "Educ" "SES"  "MMSE" "CDR"  "eTIV" "nWBV" "ASF"
```

```
#Y borramos la variable CDR ya que no aporta información
oasis_longitudinal_n4=select(oasis_longitudinal_n3, -CDR)
oasis_cross_sectional_n4=select(oasis_cross_sectional_n3, -CDR)

#Convertimos la variable Sex a números f=0 y m=1
oasis_longitudinal_n4$Sex=factor(oasis_longitudinal_n4$Sex,levels=c("F", "M"),
                           labels=c(0,1))

oasis_cross_sectional_n4$Sex=factor(oasis_cross_sectional_n4$Sex,levels=c("F", "M"),
                           labels=c(0,1))


test=oasis_longitudinal_n4 #Los datos longitudinales serán los del test
train=oasis_cross_sectional_n4 #Los seccionales el entrenamiento
#Guardamos los labels en otro vector
test_labels=oasis_longitudinal_narm_2$Group
train_labels=oasis_cross_sectional_narm$Group
#Lo convertimos a factor
test_labels=as.factor(test_labels)
```

Entrenamiento de distintos modelos

En este apartado vamos a utilizar distintos modelos, con distintos tipos de algoritmos de machine learning. Para después compararlos y comprobar cuales funcionan mejor y tienen y una mejor precisión a la hora de

predecir la demencia. Como objetivo nos hemos propuesto que nuestro modelo cumpla con una precisión mayor del 0.8, un kappa mayor de 0.6 y un AUC de 0.7. Para empezar aplicaremos distintos modelos y veremos cuales complen con la precisión y el kappa mínimo, tras lo cual veremos de estos que cumplen cual tiene el mejor AUC.

Algoritmo k-NN

Vamos a empezar utilizando el algoritmo k-NN, que es una algoritmo que utiliza la distancia entre los datos para agruparlos en los distintos grupos, podemos cambiar el valor k, que es la cantidad de observaciones que utiliza el algoritmo para las comparaciones, es decir, si k = 1 el algoritmo utilizará soo una, la observación más parecida para agrupar otra en ese mismo grupo. Vamos a realizar el algoritmo con k=[1:12].

```
library(class)
```

```
##
## Attaching package: 'class'

## The following object is masked from 'package:reshape':
##
##     condense
```

```
library(gmodels)
```

```
## Warning: package 'gmodels' was built under R version 4.0.3
```

```
library(caret)

#Obtenemos el modelo
knn_1 <- knn(train = train, test = test, cl = train_labels, k = 1)
#Lo evaluamos
CrossTable(x = test_labels, y = knn_1, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_1
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          14 |          42 |          56 |
##              |       0.250 |       0.750 |       0.438 |
```

```
##              |       0.189 |       0.778 |             |
##              |       0.109 |       0.328 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          60 |          12 |          72 |
##              |       0.833 |       0.167 |       0.562 |
##              |       0.811 |       0.222 |             |
##              |       0.469 |       0.094 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          74 |          54 |         128 |
##              |       0.578 |       0.422 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_1, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_1, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.

## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          60       12
##   Demented             14       42
##
##                Accuracy : 0.7969
##                  95% CI : (0.7167, 0.8628)
##     No Information Rate : 0.5781
##     P-Value [Acc > NIR] : 1.438e-07
##
##                   Kappa : 0.5857
##
##  Mcnemar's Test P-Value : 0.8445
##
##             Sensitivity : 0.7778
##             Specificity : 0.8108
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.8333
##              Prevalence : 0.4219
##          Detection Rate : 0.3281
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7943
##
##        'Positive' Class : Demented
##
```

```
knn_2 <- knn(train = train, test = test, cl = train_labels, k = 2)

CrossTable(x = test_labels, y = knn_2, prop.chisq= FALSE)
```

```
##
```

```
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_2
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          17 |          39 |          56 |
##              |       0.304 |       0.696 |       0.438 |
##              |       0.221 |       0.765 |             |
##              |       0.133 |       0.305 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          60 |          12 |          72 |
##              |       0.833 |       0.167 |       0.562 |
##              |       0.779 |       0.235 |             |
##              |       0.469 |       0.094 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          77 |          51 |         128 |
##              |       0.602 |       0.398 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_2, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_2, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          60       12
##   Demented             17       39
##
##                Accuracy : 0.7734
##                  95% CI : (0.6911, 0.8427)
##     No Information Rate : 0.6016
##     P-Value [Acc > NIR] : 2.943e-05
##
##                   Kappa : 0.5351
##
##  Mcnemar's Test P-Value : 0.4576
```

```
##
##             Sensitivity : 0.7647
##             Specificity : 0.7792
##          Pos Pred Value : 0.6964
##          Neg Pred Value : 0.8333
##              Prevalence : 0.3984
##          Detection Rate : 0.3047
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7720
##
##        'Positive' Class : Demented
##
```

```
knn_3 <- knn(train = train, test = test, cl = train_labels, k = 3)

CrossTable(x = test_labels, y = knn_3, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_3
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          15 |          41 |          56 |
##              |       0.268 |       0.732 |       0.438 |
##              |       0.205 |       0.745 |             |
##              |       0.117 |       0.320 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          58 |          14 |          72 |
##              |       0.806 |       0.194 |       0.562 |
##              |       0.795 |       0.255 |             |
##              |       0.453 |       0.109 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          73 |          55 |         128 |
##              |       0.570 |       0.430 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_3, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_3, positive = "Demented"):
```

```
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          58       14
##   Demented             15       41
##
##                Accuracy : 0.7734
##                  95% CI : (0.6911, 0.8427)
##     No Information Rate : 0.5703
##     P-Value [Acc > NIR] : 1.232e-06
##
##                   Kappa : 0.5388
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.7455
##             Specificity : 0.7945
##          Pos Pred Value : 0.7321
##          Neg Pred Value : 0.8056
##              Prevalence : 0.4297
##          Detection Rate : 0.3203
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7700
##
##        'Positive' Class : Demented
##
```

```
knn_4 <- knn(train = train, test = test, cl = train_labels, k = 5)

CrossTable(x = test_labels, y = knn_4, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_4
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          17 |          39 |          56 |
```

```
##              |       0.304 |       0.696 |       0.438 |
##              |       0.215 |       0.796 |             |
##              |       0.133 |       0.305 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          62 |          10 |          72 |
##              |       0.861 |       0.139 |       0.562 |
##              |       0.785 |       0.204 |             |
##              |       0.484 |       0.078 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          79 |          49 |         128 |
##              |       0.617 |       0.383 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_4, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_4, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          62       10
##   Demented             17       39
##
##                Accuracy : 0.7891
##                  95% CI : (0.7081, 0.8562)
##     No Information Rate : 0.6172
##     P-Value [Acc > NIR] : 2.396e-05
##
##                   Kappa : 0.5654
##
##  Mcnemar's Test P-Value : 0.2482
##
##             Sensitivity : 0.7959
##             Specificity : 0.7848
##          Pos Pred Value : 0.6964
##          Neg Pred Value : 0.8611
##              Prevalence : 0.3828
##          Detection Rate : 0.3047
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7904
##
##        'Positive' Class : Demented
##
```

```
knn_5 <- knn(train = train, test = test, cl = train_labels, k = 7)

CrossTable(x = test_labels, y = knn_5, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_5
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          19 |          37 |          56 |
##              |       0.339 |       0.661 |       0.438 |
##              |       0.232 |       0.804 |             |
##              |       0.148 |       0.289 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          63 |           9 |          72 |
##              |       0.875 |       0.125 |       0.562 |
##              |       0.768 |       0.196 |             |
##              |       0.492 |       0.070 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          82 |          46 |         128 |
##              |       0.641 |       0.359 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_5, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_5, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          63        9
##   Demented             19       37
##
##                Accuracy : 0.7812
##                  95% CI : (0.6996, 0.8495)
##     No Information Rate : 0.6406
##     P-Value [Acc > NIR] : 0.0004246
##
##                   Kappa : 0.5466
##
```

```
##  Mcnemar's Test P-Value : 0.0889730
##
##             Sensitivity : 0.8043
##             Specificity : 0.7683
##          Pos Pred Value : 0.6607
##          Neg Pred Value : 0.8750
##              Prevalence : 0.3594
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7863
##
##        'Positive' Class : Demented
##
```

```
knn_6 <- knn(train = train, test = test, cl = train_labels, k = 8)

CrossTable(x = test_labels, y = knn_6, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_6
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          19 |          37 |          56 |
##              |       0.339 |       0.661 |       0.438 |
##              |       0.237 |       0.771 |             |
##              |       0.148 |       0.289 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          61 |          11 |          72 |
##              |       0.847 |       0.153 |       0.562 |
##              |       0.762 |       0.229 |             |
##              |       0.477 |       0.086 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          80 |          48 |         128 |
##              |       0.625 |       0.375 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_6, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_6, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          61       11
##   Demented             19       37
##
##                Accuracy : 0.7656
##                  95% CI : (0.6826, 0.8359)
##     No Information Rate : 0.625
##     P-Value [Acc > NIR] : 0.0004948
##
##                   Kappa : 0.5161
##
##  Mcnemar's Test P-Value : 0.2012426
##
##             Sensitivity : 0.7708
##             Specificity : 0.7625
##          Pos Pred Value : 0.6607
##          Neg Pred Value : 0.8472
##              Prevalence : 0.3750
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7667
##
##        'Positive' Class : Demented
##
```

```
knn_7 <- knn(train = train, test = test, cl = train_labels, k = 9)

CrossTable(x = test_labels, y = knn_7, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_7
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
```

```
##     Demented |          17 |          39 |          56 |
##              |       0.304 |       0.696 |       0.438 |
##              |       0.202 |       0.886 |             |
##              |       0.133 |       0.305 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          67 |           5 |          72 |
##              |       0.931 |       0.069 |       0.562 |
##              |       0.798 |       0.114 |             |
##              |       0.523 |       0.039 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          84 |          44 |         128 |
##              |       0.656 |       0.344 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_7, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_7, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          67        5
##   Demented             17       39
##
##                Accuracy : 0.8281
##                  95% CI : (0.7514, 0.889)
##     No Information Rate : 0.6562
##     P-Value [Acc > NIR] : 1.232e-05
##
##                   Kappa : 0.6423
##
##  Mcnemar's Test P-Value : 0.01902
##
##             Sensitivity : 0.8864
##             Specificity : 0.7976
##          Pos Pred Value : 0.6964
##          Neg Pred Value : 0.9306
##              Prevalence : 0.3438
##          Detection Rate : 0.3047
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8420
##
##        'Positive' Class : Demented
##
```

```
knn_8 <- knn(train = train, test = test, cl = train_labels, k = 12)

CrossTable(x = test_labels, y = knn_8, prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | knn_8
##  test_labels | Nondemented |    Demented |   Row Total |
## -------------|-------------|-------------|-------------|
##     Demented |          21 |          35 |          56 |
##              |       0.375 |       0.625 |       0.438 |
##              |       0.253 |       0.778 |             |
##              |       0.164 |       0.273 |             |
## -------------|-------------|-------------|-------------|
##  Nondemented |          62 |          10 |          72 |
##              |       0.861 |       0.139 |       0.562 |
##              |       0.747 |       0.222 |             |
##              |       0.484 |       0.078 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          83 |          45 |         128 |
##              |       0.648 |       0.352 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, knn_8, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_8, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          62       10
##   Demented             21       35
##
##                Accuracy : 0.7578
##                  95% CI : (0.6742, 0.8291)
##     No Information Rate : 0.6484
##     P-Value [Acc > NIR] : 0.005223
##
##                   Kappa : 0.497
##
```

```
##  Mcnemar's Test P-Value : 0.072486
##
##             Sensitivity : 0.7778
##             Specificity : 0.7470
##          Pos Pred Value : 0.6250
##          Neg Pred Value : 0.8611
##              Prevalence : 0.3516
##          Detection Rate : 0.2734
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7624
##
##        'Positive' Class : Demented
##
```

El modelo con k=9, es el único que pasa nuestro mínimo con una precisión y un kappa mayor a 0.8 y 0.6 respectivamente. Vamos a obtener sus probabilidades para sacar posteriormente el AUC.

```
knn_7 <- knn(train = train, test = test, cl = train_labels, k = 9, prob=TRUE)
prob=attr(knn_7,"prob")
prob_knn <- ifelse(knn_7 =="Demented", 1-prob, prob)
```

Algoritmo Bayesiano

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.0.4
```

```
library(gmodels)

bayes_model <- naiveBayes(train, train_labels)
pred_bayes <- predict(bayes_model, test)

CrossTable(pred_bayes, test_labels, prop.chisq = FALSE, prop.t = FALSE,
dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  128
##
##
##              | actual
##    predicted |    Demented | Nondemented |   Row Total |
```

```
## -------------|-------------|-------------|-------------|
##  Nondemented |          18 |          67 |          85 |
##              |       0.212 |       0.788 |       0.664 |
##              |       0.321 |       0.931 |             |
## -------------|-------------|-------------|-------------|
##     Demented |          38 |           5 |          43 |
##              |       0.884 |       0.116 |       0.336 |
##              |       0.679 |       0.069 |             |
## -------------|-------------|-------------|-------------|
## Column Total |          56 |          72 |         128 |
##              |       0.438 |       0.562 |             |
## -------------|-------------|-------------|-------------|
##
##
```

```
confusionMatrix(test_labels, pred_bayes, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, pred_bayes, positive =
## "Demented"): Levels are not in the same order for reference and data.
## Refactoring data to match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          67        5
##   Demented             18       38
##
##                Accuracy : 0.8203
##                  95% CI : (0.7427, 0.8826)
##     No Information Rate : 0.6641
##     P-Value [Acc > NIR] : 6.422e-05
##
##                   Kappa : 0.6253
##
##  Mcnemar's Test P-Value : 0.01234
##
##             Sensitivity : 0.8837
##             Specificity : 0.7882
##          Pos Pred Value : 0.6786
##          Neg Pred Value : 0.9306
##              Prevalence : 0.3359
##          Detection Rate : 0.2969
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8360
##
##        'Positive' Class : Demented
##
```

Este modelo también pasaría nuestro mínimo.

Árboles de decisión (C.50)

```
library(C50)
```

```
## Warning: package 'C50' was built under R version 4.0.4
```

```
c50_model <- C5.0(train, train_labels)
prc50=predict(c50_model, test)
confusionMatrix(test_labels, prc50, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, prc50, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          71        1
##   Demented             19       37
##
##                Accuracy : 0.8438
##                  95% CI : (0.7691, 0.9019)
##     No Information Rate : 0.7031
##     P-Value [Acc > NIR] : 0.0001753
##
##                   Kappa : 0.6708
##
##  Mcnemar's Test P-Value : 0.0001439
##
##             Sensitivity : 0.9737
##             Specificity : 0.7889
##          Pos Pred Value : 0.6607
##          Neg Pred Value : 0.9861
##              Prevalence : 0.2969
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8813
##
##        'Positive' Class : Demented
##
```

Support Vector Machine

```
datalabels=cbind(train_labels, train)

tlb=cbind(test_labels, test)
svm_model <- svm(train_labels ~ ., data = datalabels, probability=TRUE)
```

```
pred <- predict(svm_model, tlb, probability=TRUE)

confusionMatrix(test_labels, pred, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, pred, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          61       11
##   Demented             15       41
##
##                Accuracy : 0.7969
##                  95% CI : (0.7167, 0.8628)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : 8.459e-07
##
##                   Kappa : 0.584
##
##  Mcnemar's Test P-Value : 0.5563
##
##             Sensitivity : 0.7885
##             Specificity : 0.8026
##          Pos Pred Value : 0.7321
##          Neg Pred Value : 0.8472
##              Prevalence : 0.4062
##          Detection Rate : 0.3203
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.7955
##
##        'Positive' Class : Demented
##
```

```
prob_SVM=attr(pred, "probabilities")
```

Curvas ROC y valor AUC

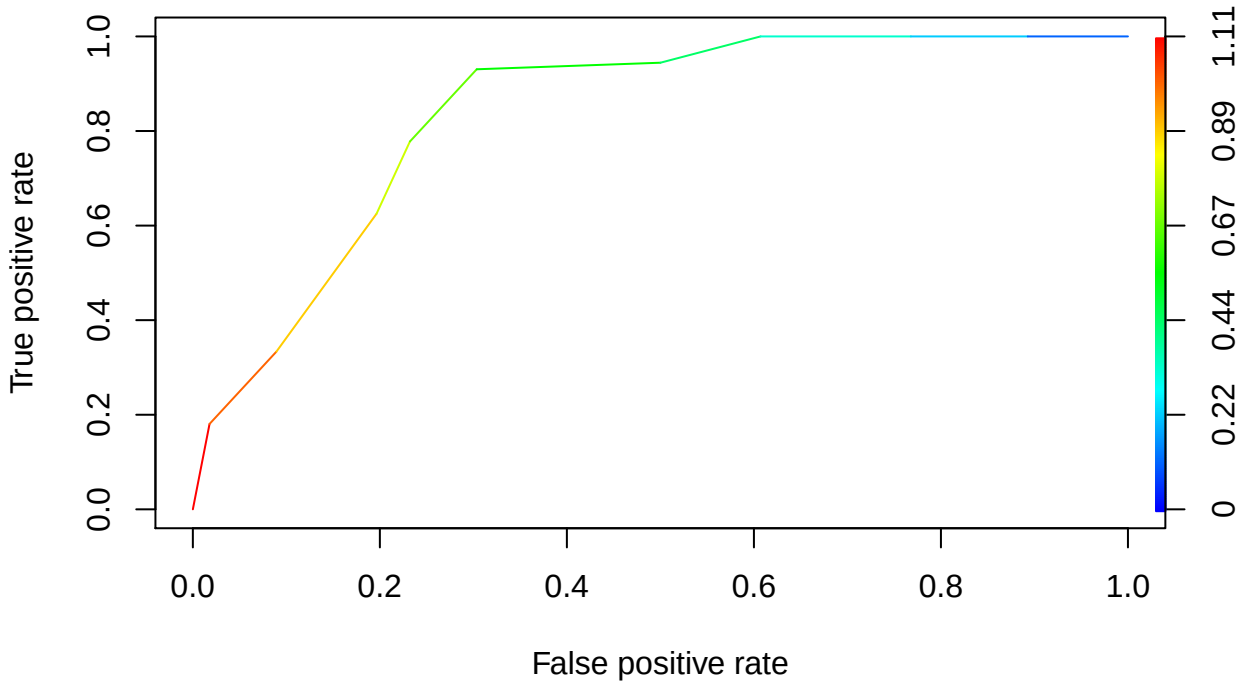
Los modelos que han obtenido el mínimo para ser considerados buenos, por los valores kappa y de precisión que nos fijamos, son los generados con árboles de decisión, algortimo bayesiano y SVM, así que con ellos vamos a generar una curva ROC y obtener el valor AUC para ver el más óptimo.

```
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.0.4
```

```
#knn
predicciones=prediction(prob_knn, test_labels)
```

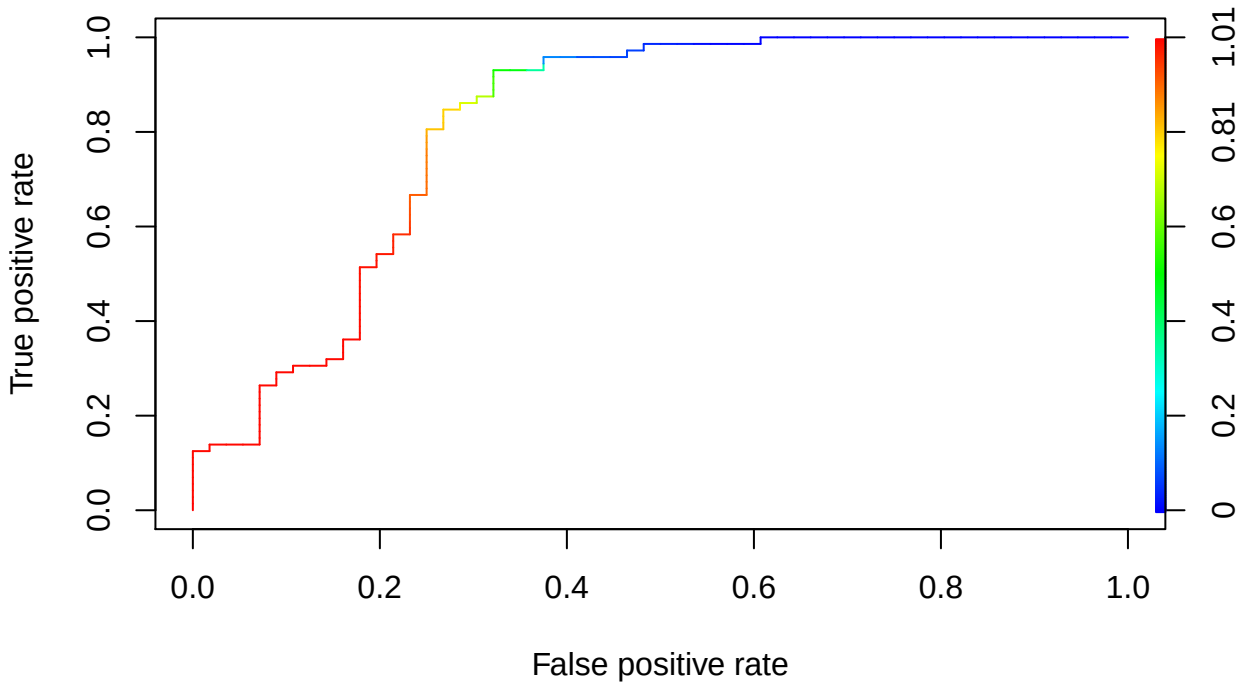
```r
perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```r
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

```
## [1] 0.8385417
```

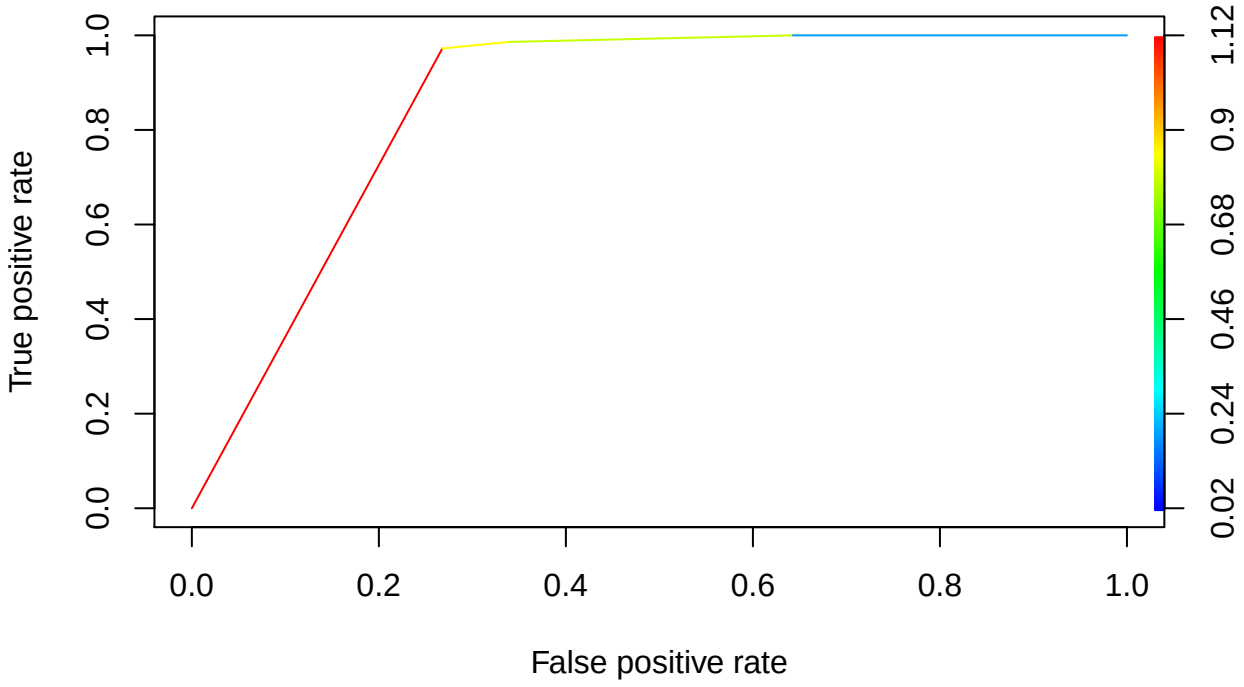
```r
#bayes
pred <- predict(bayes_model, test, type="raw")
predicciones=prediction(pred[,1], test_labels)

perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

```
## [1] 0.813244
```

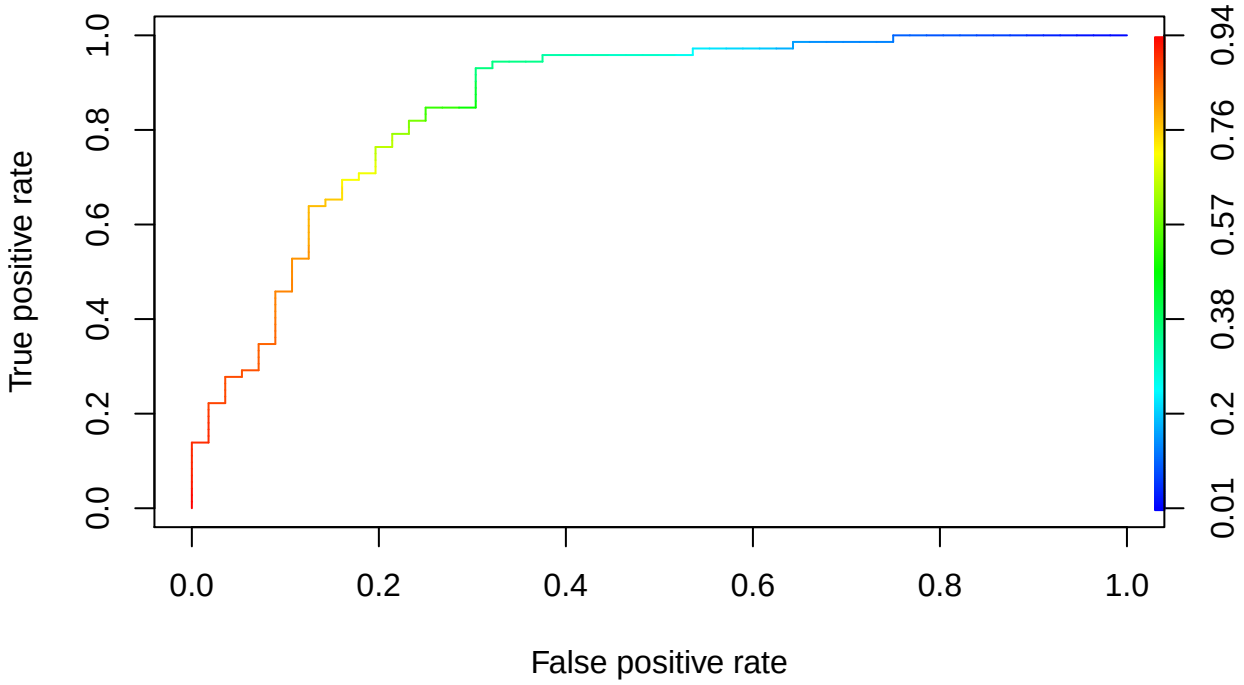
```
#Árboles de decisión
pred <- predict(c50_model, test, type = "prob")
predicciones=prediction(pred[,1], test_labels)

perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

```
## [1] 0.858755
```

```
#SVM
predicciones=prediction(prob_SVM[,1], test_labels)

perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

```
## [1] 0.8573909
```

Reajustes de los modelos

Para mejorar los modelos vamos a realizar varios cambios a ver si obtenemos mejores resultados, aunque estos modelos ya serían válidos para los mínimos que nos fijamos (a excepción del SVM que no supera la precisión y el kappa mínimo).

Cambio del tipo de normalización

```
#Normalización z score
oasis_longitudinal_z <- as.data.frame(scale(oasis_longitudinal_narm_2[8:15]))

oasis_seccional_z <- as.data.frame(scale(oasis_cross_sectional_narm[4:11]))


oasis_longitudinal_z = rename(oasis_longitudinal_z, c(EDUC="Educ"))
```

```r
#Tras normalizarlo añadimos la variable sexo al conjunto
oasis_longitudinal_z2=cbind(oasis_longitudinal_narm_2$`M/F`, oasis_longitudinal_z)

#Camiamos el nombre de esta variable a Sex
oasis_longitudinal_z3 = rename(oasis_longitudinal_z2, c("oasis_longitudinal_narm_2$`M/F`"="Sex"))



#Tras normalizarlo añadimos la variable sexo al conjunto
oasis_seccional_z2=cbind(oasis_cross_sectional_narm$M.F, oasis_seccional_z)

#Camiamos el nombre de esta variable a Sex
oasis_seccional_z3 = rename(oasis_seccional_z2, c("oasis_cross_sectional_narm$M.F"="Sex"))

names(oasis_longitudinal_z3)
```

```
## [1] "Sex"  "Age"  "Educ" "SES"  "MMSE" "CDR"  "eTIV" "nWBV" "ASF"
```

```r
names(oasis_seccional_z3)
```

```
## [1] "Sex"  "Age"  "Educ" "SES"  "MMSE" "CDR"  "eTIV" "nWBV" "ASF"
```

```r
#Borramos la variable CDR
oasis_longitudinal_z4=select(oasis_longitudinal_z3, -CDR)
oasis_seccional_z4=select(oasis_seccional_z3, -CDR)


#Convertimos por último la variable Sex a números f=0 y m=1
oasis_longitudinal_z4$Sex=factor(oasis_longitudinal_z4$Sex,levels=c("F", "M"),
                           labels=c(0,1))

oasis_seccional_z4$Sex=factor(oasis_seccional_z4$Sex,levels=c("F", "M"),
                           labels=c(0,1))



test_z=oasis_longitudinal_z4
train_z=oasis_seccional_z4
test_labels=oasis_longitudinal_narm_2$Group
train_labels=oasis_cross_sectional_narm$Group
test_labels=as.factor(test_labels)
```

Una vez normalizados por este nuevo método vamos a aplicar los mismos modelos y ver si funcionan mejor:

```r
#knn
knn_9=knn(train = train_z, test = test_z, train_labels, k=9, prob=TRUE)
confusionMatrix(test_labels, knn_9, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, knn_9, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          67        5
##   Demented             23       33
##
##                Accuracy : 0.7812
##                  95% CI : (0.6996, 0.8495)
##     No Information Rate : 0.7031
##     P-Value [Acc > NIR] : 0.030398
##
##                   Kappa : 0.5391
##
##  Mcnemar's Test P-Value : 0.001315
##
##             Sensitivity : 0.8684
##             Specificity : 0.7444
##          Pos Pred Value : 0.5893
##          Neg Pred Value : 0.9306
##              Prevalence : 0.2969
##          Detection Rate : 0.2578
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8064
##
##        'Positive' Class : Demented
##
```

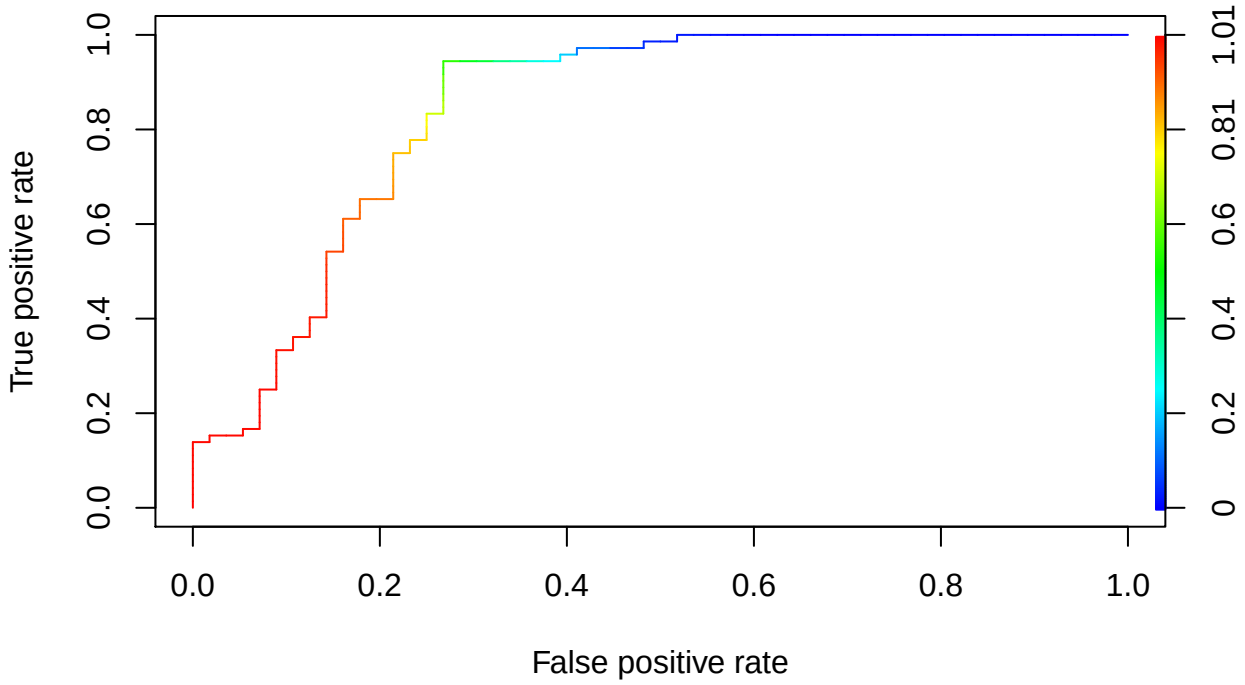
```
#Bayes
bayes_model <- naiveBayes(train_z, train_labels)
pred_bayes <- predict(bayes_model, test_z)
confusionMatrix(test_labels, pred_bayes, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, pred_bayes, positive =
## "Demented"): Levels are not in the same order for reference and data.
## Refactoring data to match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          68        4
##   Demented             15       41
##
##                Accuracy : 0.8516
##                  95% CI : (0.7779, 0.9082)
##     No Information Rate : 0.6484
##     P-Value [Acc > NIR] : 2.347e-07
##
##                   Kappa : 0.6917
##
##  Mcnemar's Test P-Value : 0.02178
##
```

```
##             Sensitivity : 0.9111
##             Specificity : 0.8193
##          Pos Pred Value : 0.7321
##          Neg Pred Value : 0.9444
##              Prevalence : 0.3516
##          Detection Rate : 0.3203
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8652
##
##        'Positive' Class : Demented
##
```

```
pred_bayes <- predict(bayes_model, test_z, type="raw")
predicciones=prediction(pred_bayes[,1], test_labels)

perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

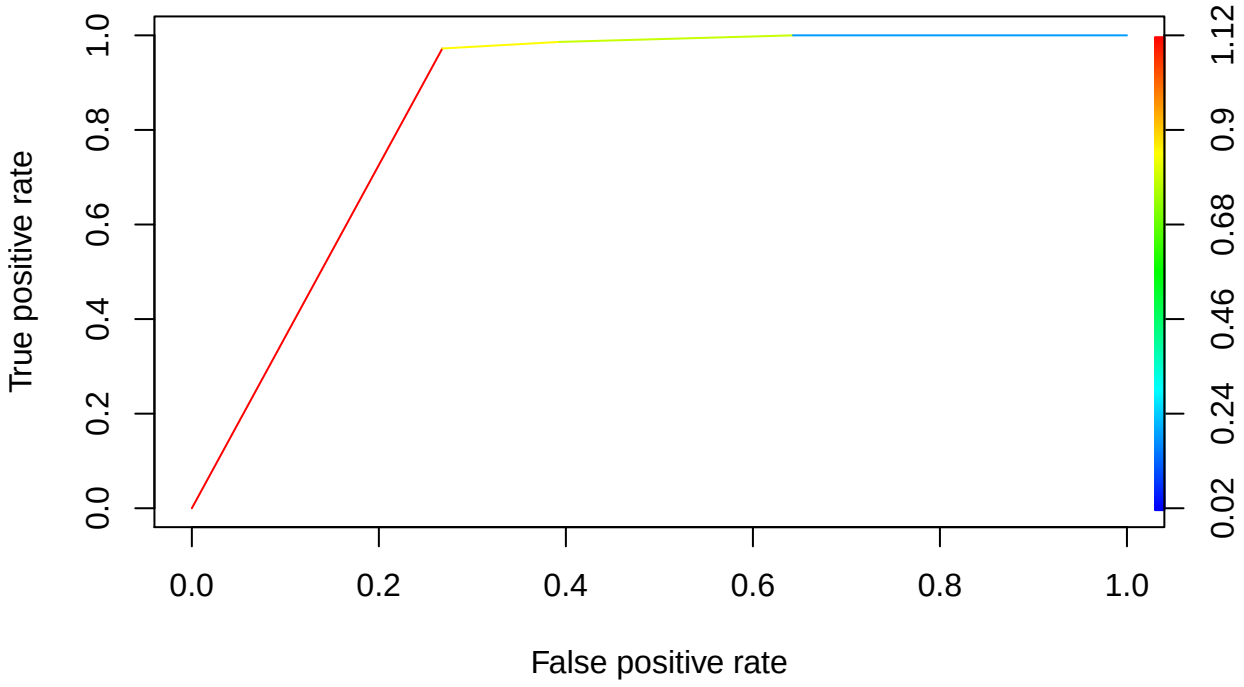
```
## [1] 0.843006
```

```r
#ÁRboles
c50_model <- C5.0(train_z, train_labels)
confusionMatrix(test_labels, prc50, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, prc50, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
##   Nondemented          71        1
##   Demented             19       37
##
##                Accuracy : 0.8438
##                  95% CI : (0.7691, 0.9019)
##     No Information Rate : 0.7031
##     P-Value [Acc > NIR] : 0.0001753
##
##                   Kappa : 0.6708
##
##  Mcnemar's Test P-Value : 0.0001439
##
##             Sensitivity : 0.9737
##             Specificity : 0.7889
##          Pos Pred Value : 0.6607
##          Neg Pred Value : 0.9861
##              Prevalence : 0.2969
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8813
##
##        'Positive' Class : Demented
##
```

```r
pred <- predict(c50_model, test_z, type = "prob")
predicciones=prediction(pred[,1], test_labels)

perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

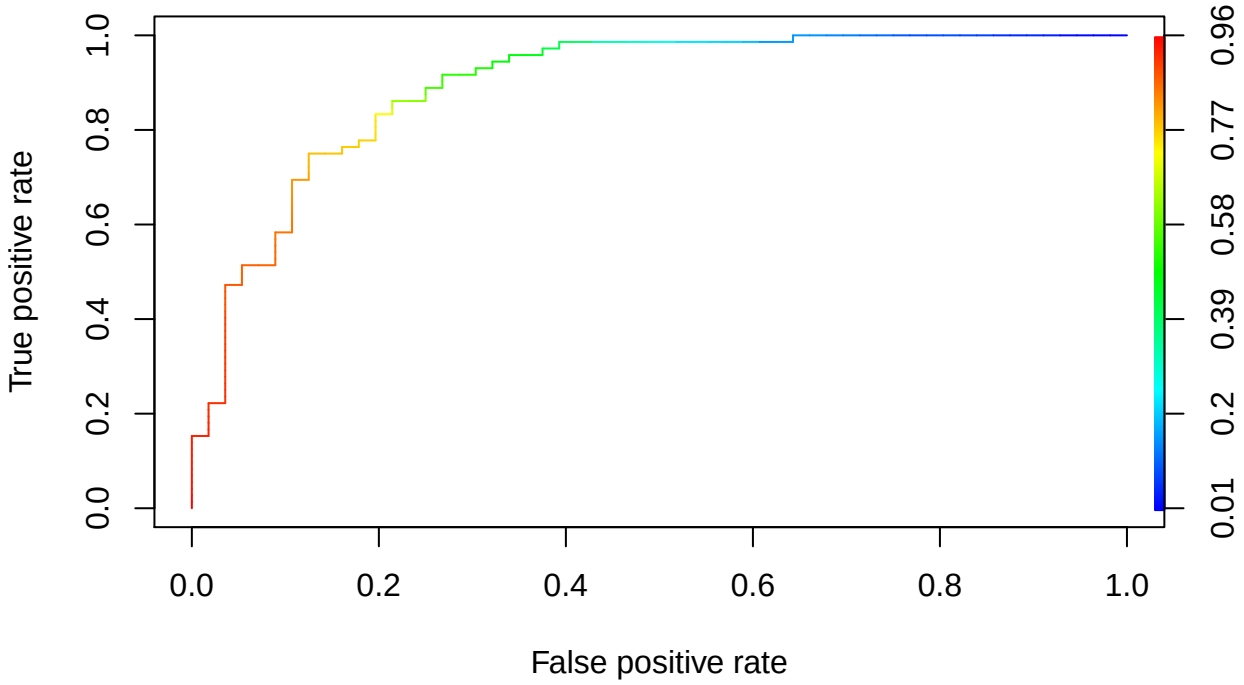
```
## [1] 0.8580109
```

```
#SVM
datalabels_z=cbind(train_labels, train_z)

tlb_z=cbind(test_labels, test_z)
model <- svm(train_labels ~ ., data = datalabels_z, probability=TRUE)

pred <- predict(model, tlb_z, probability=TRUE)

confusionMatrix(test_labels, pred, positive = "Demented")
```

```
## Warning in confusionMatrix.default(test_labels, pred, positive = "Demented"):
## Levels are not in the same order for reference and data. Refactoring data to
## match.

## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Nondemented Demented
```

```
##   Nondemented          67        5
##   Demented             17       39
##
##                Accuracy : 0.8281
##                  95% CI : (0.7514, 0.889)
##     No Information Rate : 0.6562
##     P-Value [Acc > NIR] : 1.232e-05
##
##                   Kappa : 0.6423
##
##  Mcnemar's Test P-Value : 0.01902
##
##             Sensitivity : 0.8864
##             Specificity : 0.7976
##          Pos Pred Value : 0.6964
##          Neg Pred Value : 0.9306
##              Prevalence : 0.3438
##          Detection Rate : 0.3047
##    Detection Prevalence : 0.4375
##       Balanced Accuracy : 0.8420
##
##        'Positive' Class : Demented
##
```

```
prob_SVM=attr(pred, "probabilities")


predicciones=prediction(prob_SVM[,1], test_labels)

perf=performance(predicciones, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
perf.auc <- performance(predicciones, measure = "auc")
AUC=unlist(perf.auc@y.values)
AUC
```

```
## [1] 0.8936012
```

Automatic tuning

```
#Con normalización max/min
set.seed(123)
knn=train(train_labels ~ ., data = datalabels, method = "knn")
knn
```

```
## k-Nearest Neighbors
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
```

```
##
##   k  Accuracy   Kappa
##   5  0.7406930  0.4357329
##   7  0.7551139  0.4624777
##   9  0.7584501  0.4627087
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
arbol <- train(train_labels ~ ., data = datalabels, method = "C5.0")
arbol
```

```
## C5.0
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
##
##   model  winnow  trials  Accuracy   Kappa
##   rules  FALSE    1      0.8025091  0.5749996
##   rules  FALSE   10      0.8287478  0.6257659
##   rules  FALSE   20      0.8314367  0.6320133
##   rules   TRUE    1      0.8109305  0.5882802
##   rules   TRUE   10      0.8201923  0.6054489
##   rules   TRUE   20      0.8248962  0.6169025
##   tree   FALSE    1      0.7951641  0.5588480
##   tree   FALSE   10      0.8292029  0.6269106
##   tree   FALSE   20      0.8217522  0.6096168
##   tree    TRUE    1      0.8029467  0.5750924
##   tree    TRUE   10      0.8180770  0.6018856
##   tree    TRUE   20      0.8191349  0.6023755
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were trials = 20, model = rules and
##  winnow = FALSE.
```

```
bayes <- train(train_labels ~ ., data = datalabels, method = "nb")
bayes
```

```
## Naive Bayes
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
```

```
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.8223682  0.6231698
##    TRUE      0.8094231  0.5835302
##
## Tuning parameter 'fL' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 0, usekernel = FALSE and adjust
##  = 1.
```

```
svmmodel <- train(train_labels ~ ., data = datalabels, method = "svmRadial")
svmmodel
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.25  0.7757655  0.4916837
##   0.50  0.7829656  0.5151112
##   1.00  0.7812342  0.5190828
##
## Tuning parameter 'sigma' was held constant at a value of 0.1264807
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.1264807 and C = 0.5.
```

```
#Con normalización z score
set.seed(123)
knn_z=train(train_labels ~ ., data = datalabels_z, method = "knn")
knn_z
```

```
## k-Nearest Neighbors
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
```

```
##   5  0.7657508  0.4931871
##   7  0.7623757  0.4792319
##   9  0.7652998  0.4826126
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
arbol_z <- train(train_labels ~ ., data = datalabels_z, method = "C5.0")
arbol_z
```

```
## C5.0
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
##
##   model  winnow  trials  Accuracy   Kappa
##   rules  FALSE    1      0.8030029  0.5759738
##   rules  FALSE   10      0.8269953  0.6215156
##   rules  FALSE   20      0.8292086  0.6273447
##   rules   TRUE    1      0.8109305  0.5882802
##   rules   TRUE   10      0.8207727  0.6062828
##   rules   TRUE   20      0.8254335  0.6170386
##   tree   FALSE    1      0.7962059  0.5611270
##   tree   FALSE   10      0.8277284  0.6241817
##   tree   FALSE   20      0.8233814  0.6127662
##   tree    TRUE    1      0.8029467  0.5750924
##   tree    TRUE   10      0.8165846  0.5988199
##   tree    TRUE   20      0.8169620  0.5969468
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were trials = 20, model = rules and
##  winnow = FALSE.
```

```
bayes_z <- train(train_labels ~ ., data = datalabels_z, method = "nb")
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 70
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 72
```

```
bayes_z
```

```
## Naive Bayes
##
```

```
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.8223682  0.6231698
##    TRUE      0.8099360  0.5845402
##
## Tuning parameter 'fL' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 0, usekernel = FALSE and adjust
##  = 1.
```

```
svmmodel_z <- train(train_labels ~ ., data = datalabels_z, method = "svmRadial")
svmmodel_z
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 214 samples
##   8 predictor
##   2 classes: 'Nondemented', 'Demented'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 214, 214, 214, 214, 214, 214, ...
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.25  0.7757655  0.4916837
##   0.50  0.7829656  0.5151112
##   1.00  0.7812342  0.5190828
##
## Tuning parameter 'sigma' was held constant at a value of 0.1264807
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.1264807 and C = 0.5.
```

Los resultados son muy parecidos con ambas normalizaciones. Los mejores los obtenemos con el algoritmo bayesiano con el parámetro kernel = FALSE, y con el los árboles de decisión ajustando los parámetros del siguiente modo: model=rules, winnow=FALSE y trials=20.