

# PRA2 - Tipología y ciclo de vida de los datos

Álvaro Rodríguez Pardo, Óscar Rojo Martín

01 junio, 2021

## Índice

<b>1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?</b>	<b>2</b>
<b>2. Integración y selección de los datos de interés a analizar.</b>	<b>5</b>
<b>3. Limpieza de datos</b>	<b>9</b>
3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos? . . . . .	9
3.2. Estudio de valores atípicos (outliers) . . . . .	10
<b>4. Análisis de datos</b>	<b>22</b>
4.0. EDA . . . . .	22
4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar). . . . .	24
4.2. Comprobación de la normalidad y homogeneidad de la varianza. . . . .	39
4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes. . . . .	40
<b>5. Representación de los resultados a partir de tablas y gráficas.</b>	<b>63</b>
5.1 Tabla del dataset . . . . .	63
5.2 Representaciones gráficas . . . . .	63
5.3 Algoritmo Naive Bayes . . . . .	64
5.4 Relación Enfermedad vs IMC y Presión Arterial . . . . .	67
5.5 Intervalo de densidades de los atributos del dataset . . . . .	70
<b>6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?</b>	<b>74</b>
<b>7. Código: Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y representación de los datos. Si lo preferís, también podéis trabajar en Python.</b>	<b>75</b>
<b>8. Dónde consultar el proyecto</b>	<b>76</b>
<b>9. Contribuciones al trabajo</b>	<b>77</b>

---

**1. DESCRIPCIÓN DEL DATASET. ¿POR QUÉ ES IMPORTANTE Y QUÉ PREGUNTA/PROBLEMA PRETENDE RESPONDER?**

---

## **1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?**

El conjunto de datos que se utilizará para la práctica se llama Cardiovascular Disease Dataset y se puede encontrar en [kaggle](#). Este contiene 70000 registros que hacen referencia a pacientes, a los cuales se les hace un estudio utilizando 13 variables distintas. Una de estas variables es la variable objetivo, la cual es binaria y explica si el paciente en cuestión padece de una enfermedad cardiovascular o no. En este sentido, este conjunto de datos es importante porque con él podemos estudiar cuáles son los atributos más importantes que tienen una influencia directa sobre las enfermedades cardiovasculares, siendo esta la pregunta principal que se tratará de responder mediante el estudio.

Antes de continuar, hagamos un repaso sobre cada una de las variables del dataset:

1. ID: Número de identificación del paciente.
2. Age: Edad del paciente en días.
3. Height: Altura del paciente en centímetros.
4. Weight: Peso del paciente en kilogramos.
5. Gender: Género del paciente.
6. Systolic blood pressure: Presión arterial sistólica del paciente.
7. Diastolic blood pressure: Presión arterial diastólica del paciente.
8. Cholesterol: Niveles de colesterol del paciente.
9. Glucose: Niveles de glucosa del paciente.
10. Smoking: Si el paciente es fumador o no.
11. Alcohol intake: Si el paciente ingiere alcohol o no.
12. Physical activity: Si el paciente realiza actividad física o no.
13. Presence or absence of cardiovascular disease (Variable objetivo): Si el paciente presenta una enfermedad cardiovascular o no.



***1. DESCRIPCIÓN DEL DATASET. ¿POR QUÉ ES IMPORTANTE Y QUÉ PREGUNTA/PROBLEMA PRETENDE RESPONDER?***

---

**Preparado workspace**

```
# Limpiamos el workspace, por si hubiera algun dataset o informacion cargada  
rm(list = ls())  
  
# Limpiamos la consola  
cat("\014")
```



***1. DESCRIPCIÓN DEL DATASET. ¿POR QUÉ ES IMPORTANTE Y QUÉ PREGUNTA/PROBLEMA PRETENDE RESPONDER?***

---

**Carga de librerías**

```
# source("loadPackages.R")

packages <- c("ggplot2", "ggpubr", "readr", "plotly", "tidyverse", "lubridate",
           "magrittr", "funModeling", "skimr", "dplyr", "nortest", "caret",
           "rpart", "rpart.plot", "pROC", "ROCR", "performance", "see",
           "plotrix", "e1071", "interplot", "bayestestR", "rstanarm", "fBasics",
           "glmnet", "rminer") 

new <- packages[!(packages %in% installed.packages() [, "Package"])]
if(length(new)) install.packages(new)
a= lapply(packages, require, character.only=TRUE)
```



## 2. Integración y selección de los datos de interés a analizar.

Procedemos a cargar el conjunto de datos

```
cardio <- read.csv('data/cardio_train.csv', sep = ';', header = FALSE,
                    skip = 1)

# Le damos nombre a las variables para una mejor comprensión que las que vienen
# por defecto
colnames(cardio) <- c('id', 'age', 'gender', 'height', 'weight',
                      'systolic_blood_pressure', 'diastolic_blood_pressure',
                      'cholesterol', 'glucose', 'smoking', 'alcohol_intake',
                      'physical_activity', 'cardiovascular_disease')
```

Echamos un vistazo a sus dimensiones, a los tipos de variables que contiene, a un par de observaciones iniciales y al resumen de las estadísticas principales de dichas variables

```
# Dimensiones del conjunto de datos
dim(cardio)

## [1] 70000      13

# Clases de las variables y valores de las primeras observaciones
str(cardio)

## 'data.frame':    70000 obs. of  13 variables:
##   $ id          : int  0 1 2 3 4 8 9 12 13 14 ...
##   $ age         : int  18393 20228 18857 17623 17474 21914 22113 22584 17668 19834 ...
##   $ gender      : int  2 1 1 2 1 1 1 2 1 1 ...
##   $ height      : int  168 156 165 169 156 151 157 178 158 164 ...
##   $ weight       : num  62 85 64 82 56 67 93 95 71 68 ...
##   $ systolic_blood_pressure : int  110 140 130 150 100 120 130 130 110 110 ...
##   $ diastolic_blood_pressure: int  80 90 70 100 60 80 80 90 70 60 ...
##   $ cholesterol  : int  1 3 3 1 1 2 3 3 1 1 ...
##   $ glucose      : int  1 1 1 1 1 2 1 3 1 1 ...
##   $ smoking      : int  0 0 0 0 0 0 0 0 0 0 ...
##   $ alcohol_intake: int  0 0 0 0 0 0 0 0 0 0 ...
##   $ physical_activity: int  1 1 0 1 0 0 1 1 1 0 ...
##   $ cardiovascular_disease: int  0 1 1 1 0 0 0 1 0 0 ...

# Resumen de las estadísticas principales de las variables
summary(cardio)

##      id           age        gender      height
## Min.   : 0   Min.   :10798   Min.   :1.00   Min.   : 55.0
## 1st Qu.:25007 1st Qu.:17664   1st Qu.:1.00   1st Qu.:159.0
## Median :50002 Median :19703   Median :1.00   Median :165.0
## Mean   :49972 Mean   :19469   Mean   :1.35   Mean   :164.4
## 3rd Qu.:74889 3rd Qu.:21327   3rd Qu.:2.00   3rd Qu.:170.0
## Max.   :99999  Max.   :23713   Max.   :2.00   Max.   :250.0
## 
##      weight      systolic_blood_pressure diastolic_blood_pressure
## Min.   :10.00   Min.   :-150.00          Min.   :-70.00
## 1st Qu.:65.00   1st Qu.: 120.00          1st Qu.: 80.00
## Median :72.00   Median : 120.00          Median : 80.00
## Mean   :74.21   Mean   : 128.8          Mean   : 96.63
## 3rd Qu.:82.00   3rd Qu.: 140.0          3rd Qu.: 90.00
## Max.   :200.00  Max.   :16020.0         Max.   :11000.00
```



## 2. INTEGRACIÓN Y SELECCIÓN DE LOS DATOS DE INTERÉS A ANALIZAR.

```

##   cholesterol      glucose      smoking      alcohol_intake
##   Min.   :1.000   Min.   :1.000   Min.   :0.00000   Min.   :0.00000
## 1st Qu.:1.000  1st Qu.:1.000  1st Qu.:0.00000  1st Qu.:0.00000
## Median :1.000  Median :1.000  Median :0.00000  Median :0.00000
## Mean    :1.367  Mean    :1.226  Mean    :0.08813  Mean    :0.05377
## 3rd Qu.:2.000  3rd Qu.:1.000  3rd Qu.:0.00000  3rd Qu.:0.00000
## Max.    :3.000  Max.    :3.000  Max.    :1.00000  Max.    :1.00000
## physical_activity cardiovascular_disease
## Min.   :0.0000   Min.   :0.0000
## 1st Qu.:1.0000  1st Qu.:0.0000
## Median :1.0000  Median :0.0000
## Mean    :0.8037  Mean    :0.4997
## 3rd Qu.:1.0000  3rd Qu.:1.0000
## Max.    :1.0000  Max.    :1.0000

# Resumen general
skimr::skim(cardio)

```

Table 1: Data summary

Name	cardio
Number of rows	70000
Number of columns	13
Column type frequency:	
numeric	13
Group variables	None

### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	q90
id	0	1	49972.42	28851.30	0	25006.75	50001.5	74889.25	90000.00
age	0	1	19468.87	2467.25	10798	17664.00	19703.0	21327.00	22000.00
gender	0	1	1.35	0.48	1	1.00	1.0	2.00	2.00
height	0	1	164.36	8.21	55	159.00	165.0	170.00	175.00
weight	0	1	74.21	14.40	10	65.00	72.0	82.00	90.00
systolic_blood_pressure	0	1	128.82	154.01	-150	120.00	120.0	140.00	150.00
diastolic_blood_pressure	0	1	96.63	188.47	-70	80.00	80.0	90.00	100.00
cholesterol	0	1	1.37	0.68	1	1.00	1.0	2.00	2.00
glucose	0	1	1.23	0.57	1	1.00	1.0	1.00	1.00
smoking	0	1	0.09	0.28	0	0.00	0.0	0.00	0.00
alcohol_intake	0	1	0.05	0.23	0	0.00	0.0	0.00	0.00
physical_activity	0	1	0.80	0.40	0	1.00	1.0	1.00	1.00
cardiovascular_disease	0	1	0.50	0.50	0	0.00	0.0	1.00	1.00

```

# Ver zeros, NA, dtype y unique
funModeling::status(cardio)

```

```

##                                     variable q_zeros      p_zeros q_na
##                                     id       1 1.428571e-05     0
##                                     age      0 0.000000e+00     0

```



## 2. INTEGRACIÓN Y SELECCIÓN DE LOS DATOS DE INTERÉS A ANALIZAR.

---

```

## gender           gender      0 0.000000e+00 0
## height          height      0 0.000000e+00 0
## weight          weight      0 0.000000e+00 0
## systolic_blood_pressure systolic_blood_pressure 0 0.000000e+00 0
## diastolic_blood_pressure diastolic_blood_pressure 21 3.000000e-04 0
## cholesterol     cholesterol 0 0.000000e+00 0
## glucose         glucose     0 0.000000e+00 0
## smoking         smoking    63831 9.118714e-01 0
## alcohol_intake alcohol_intake 66236 9.462286e-01 0
## physical_activity physical_activity 13739 1.962714e-01 0
## cardiovascular_disease cardiovascular_disease 35021 5.003000e-01 0
## p_na q_inf p_inf type unique
## id              0 0 0 integer 70000
## age             0 0 0 integer 8076
## gender          0 0 0 integer 2
## height          0 0 0 integer 109
## weight          0 0 0 numeric 287
## systolic_blood_pressure 0 0 0 integer 153
## diastolic_blood_pressure 0 0 0 integer 157
## cholesterol     0 0 0 integer 3
## glucose         0 0 0 integer 3
## smoking         0 0 0 integer 2
## alcohol_intake 0 0 0 integer 2
## physical_activity 0 0 0 integer 2
## cardiovascular_disease 0 0 0 integer 2

```

Observamos que el dataset contiene 70000 observaciones (filas) y 13 variables (columnas), de las cuales todas son del tipo entero excepto la variable “weight”, la cual es del tipo numérico. Si nos fijamos en los valores mínimos y máximos que toman ciertas variables, veremos como se trata de variables categóricas, por lo que procederemos a transformarlas en tipo factor

```

cardio$gender <- ifelse(cardio$gender == 1, 'Mujer', 'Hombre')
cardio$gender <- as.factor(cardio$gender)

cardio$smoking <- ifelse(cardio$smoking == 0, 'No', 'Sí')
cardio$smoking <- as.factor(cardio$smoking)

cardio$alcohol_intake <- ifelse(cardio$alcohol_intake == 0, 'No', 'Sí')
cardio$alcohol_intake <- as.factor(cardio$alcohol_intake)

cardio$physical_activity <- ifelse(cardio$physical_activity == 0, 'No', 'Sí')
cardio$physical_activity <- as.factor(cardio$physical_activity)

cardio$cardiovascular_disease <- ifelse(cardio$cardiovascular_disease == 0,
                                         'No', 'Sí')
cardio$cardiovascular_disease <- as.factor(cardio$cardiovascular_disease)

# library(dplyr)
cardio$cholesterol <- as.factor(cardio$cholesterol)
cardio$cholesterol <- recode_factor(cardio$cholesterol, '1' = "Normal",
                                      '2' = "Por_encima_de_lo_normal",
                                      '3' = "Muy_por_encima_de_lo_normal")

cardio$glucose <- as.factor(cardio$glucose)
cardio$glucose <- recode_factor(cardio$glucose, '1' = "Normal",

```



## 2. INTEGRACIÓN Y SELECCIÓN DE LOS DATOS DE INTERÉS A ANALIZAR.

```
'2' = "Por_encima_de_lo_normal",
'3' = "Muy_por_encima_de_lo_normal")
```

```
# Cambiamos la edad de los pacientes de días a años
cardio$age <- trunc(cardio$age/365)
```

Además, la variable “id” hace referencia a la identificación del paciente y esto no es interesante, por lo que la eliminaremos del dataframe

```
cardio <- cardio[, -1]
```

Todas las demás variables pueden aportar bastante información valiosa para el estudio, así que estas será las variables de interés a analizar.



### 3. Limpieza de datos

#### 3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

Veamos si existen valores perdidos (missing values) en el dataset y los ceros que puedan tener las variables numéricas representando quizás valores perdidos

```
cat("Contamos el número de valores na por atributo: \n")

## Contamos el número de valores na por atributo:
colSums(is.na(cardio))

##          age           gender         height
##            0             0             0
##      weight systolic_blood_pressure diastolic_blood_pressure
##            0                 0                 0
##      cholesterol       glucose       smoking
##            0                 0                 0
##      alcohol_intake physical_activity cardiovascular_disease
##            0                     0                     0

cat("\nContamos el número de valores vacíos por atributo: \n")

##
## Contamos el número de valores vacíos por atributo:
colSums(cardio=="")

##          age           gender         height
##            0             0             0
##      weight systolic_blood_pressure diastolic_blood_pressure
##            0                 0                 0
##      cholesterol       glucose       smoking
##            0                 0                 0
##      alcohol_intake physical_activity cardiovascular_disease
##            0                     0                     0

cat("\nContamos el número de ceros por atributo: \n")

##
## Contamos el número de ceros por atributo:
colSums(cardio == 0)

##          age           gender         height
##            0             0             0
##      weight systolic_blood_pressure diastolic_blood_pressure
##            0                 0                 21
##      cholesterol       glucose       smoking
##            0                 0                 0
##      alcohol_intake physical_activity cardiovascular_disease
##            0                     0                     0
```

Como podemos comprobar, no existen valores perdidos. En el caso de haberlos habido, estos podrían haber sido tratados mediante su reemplazo por la etiqueta “Desconocido”, por ejemplo, en el caso de que la variable en cuestión tuviese campos de tipo *string*. Si, por el contrario, se tratase de variables numéricas, podríamos optar por reemplazar los registros perdidos por una misma medida de tendencia central, es decir, por la



media o la mediana de ese atributo, dependiendo de la distribución de los datos; o podríamos implementar métodos probabilistas para imputar los valores perdidos mediante el uso de métodos de regresión, inferencias basadas en modelos bayesianos o árboles de decisión.

En cuanto a variables numéricas con valores iguales a 0, vemos que “diastolic\_blood\_pressure” contiene 21. Estos valores podrían bien ser valores atípicos más que perdidos, por lo que los trataremos en el siguiente apartado.

### 3.2. Estudio de valores atípicos (outliers)

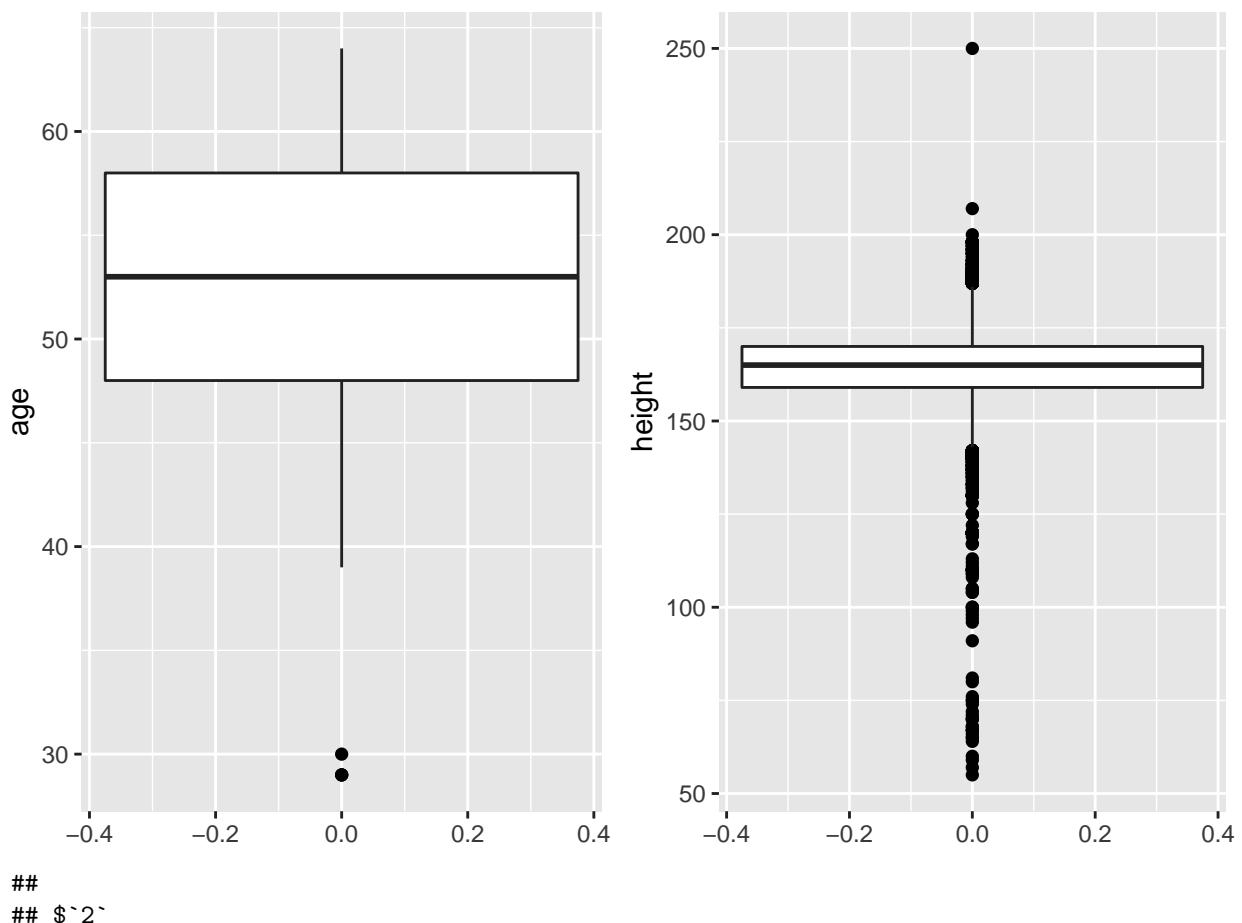
Procederemos a representar gráficamente las variables numéricas a través de sus cuartiles mediante el uso de gráficos de caja (boxplots) para estudiar si existen valores atípicos.

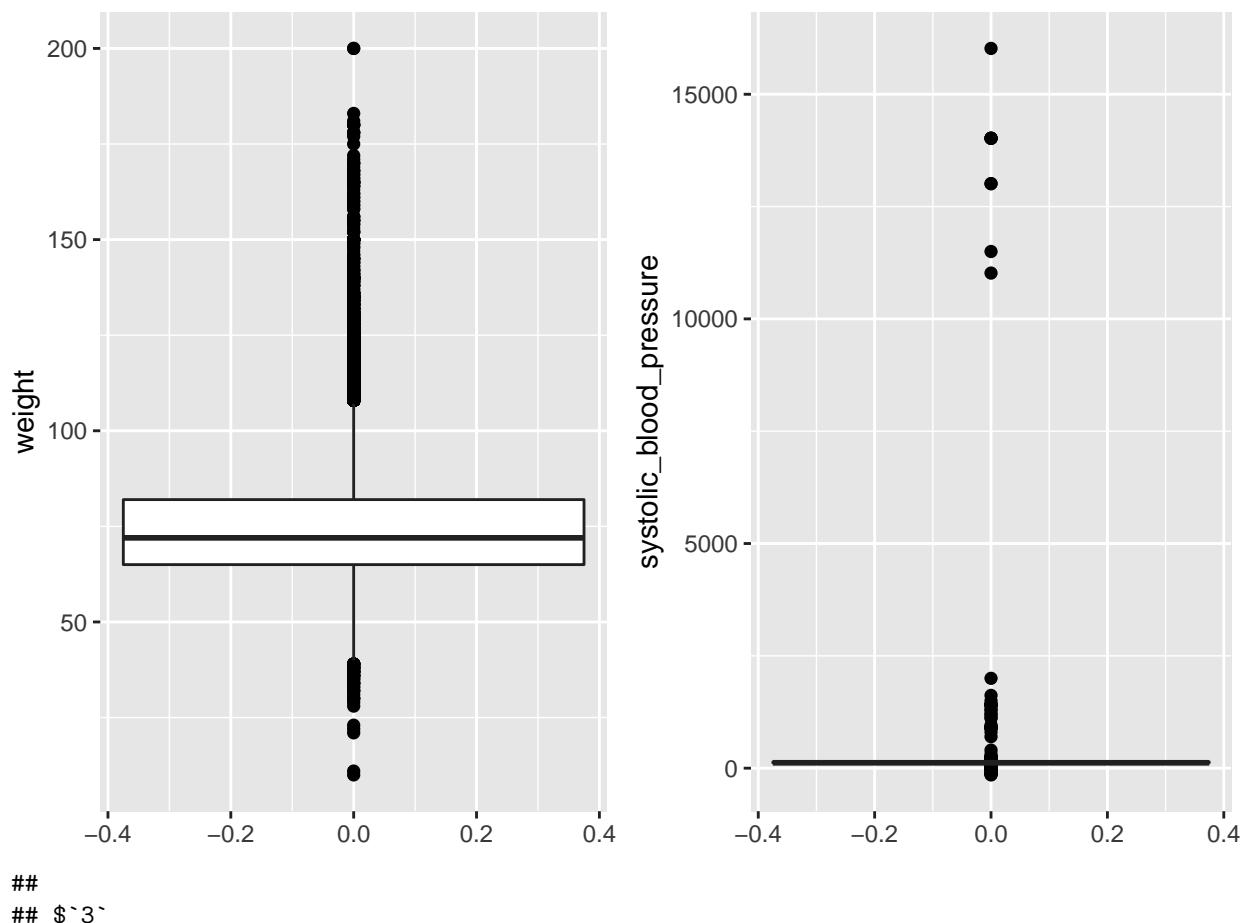
```
# Guardamos los boxplots en variables
a <- ggplot(cardio, aes(y=age)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                notch=FALSE)
h <- ggplot(cardio, aes(y=height)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                notch=FALSE)
w <- ggplot(cardio, aes(y=weight)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                notch=FALSE)
sbp <- ggplot(cardio, aes(y=systolic_blood_pressure)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                notch=FALSE)
dbp <- ggplot(cardio, aes(y=diastolic_blood_pressure)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                notch=FALSE)

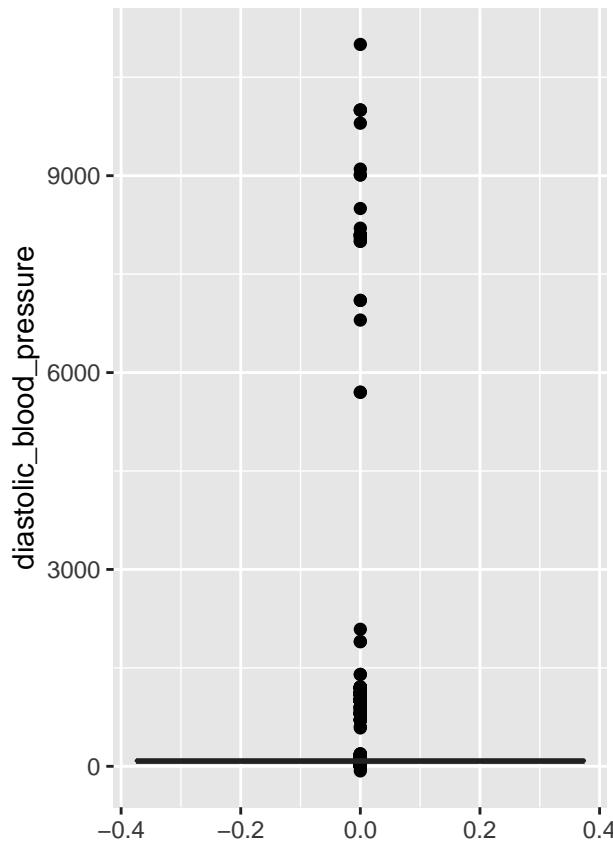
# Graficamos los boxplots en un mismo layout
ggarrange(a,h,w,sbp,dbp, ncol = 2)
```

```
## $`1`
```







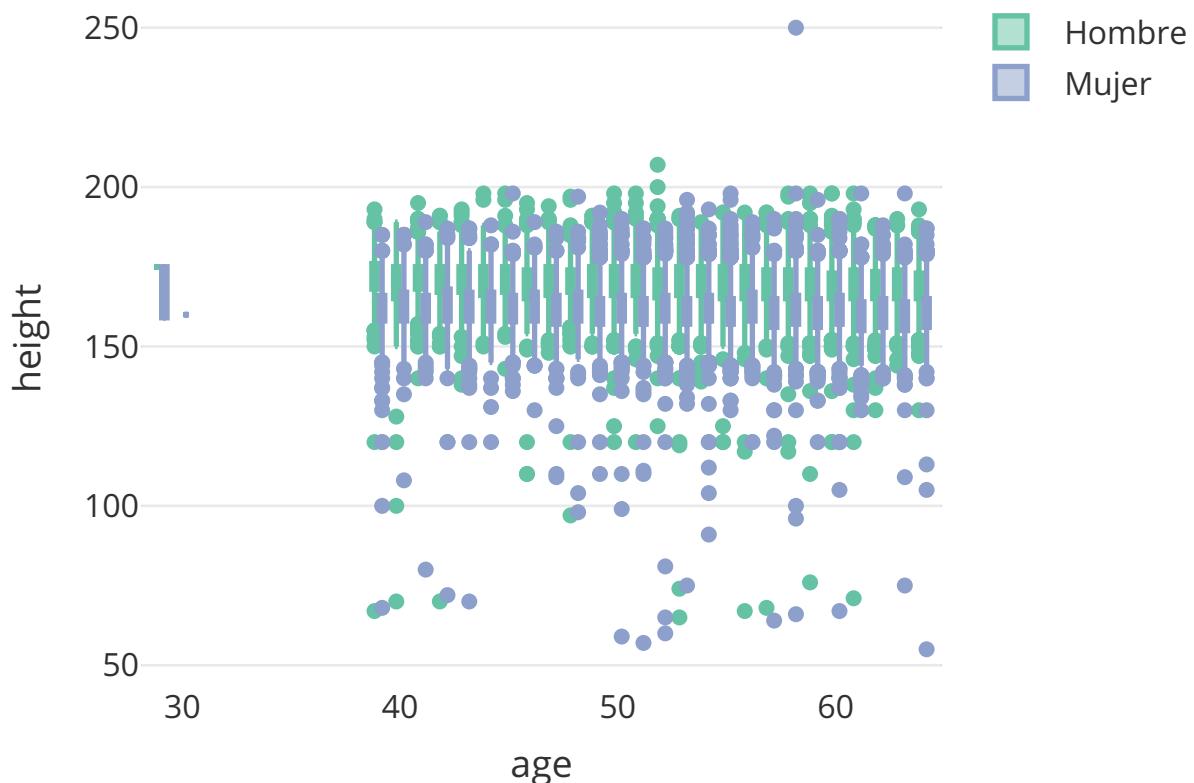


```
##  
## attr(,"class")  
## [1] "list"      "ggarrange"
```

También podemos representar una serie de gráficos que comparan dos variables y ver como se distribuyen las observaciones, diferenciando a su vez por las categorías de una tercera variable

```
fig <- plot_ly(cardio, x = ~age, y = ~height, color = ~gender, type = "box")  
fig <- fig %>% layout(boxmode = "group")  
  
fig
```

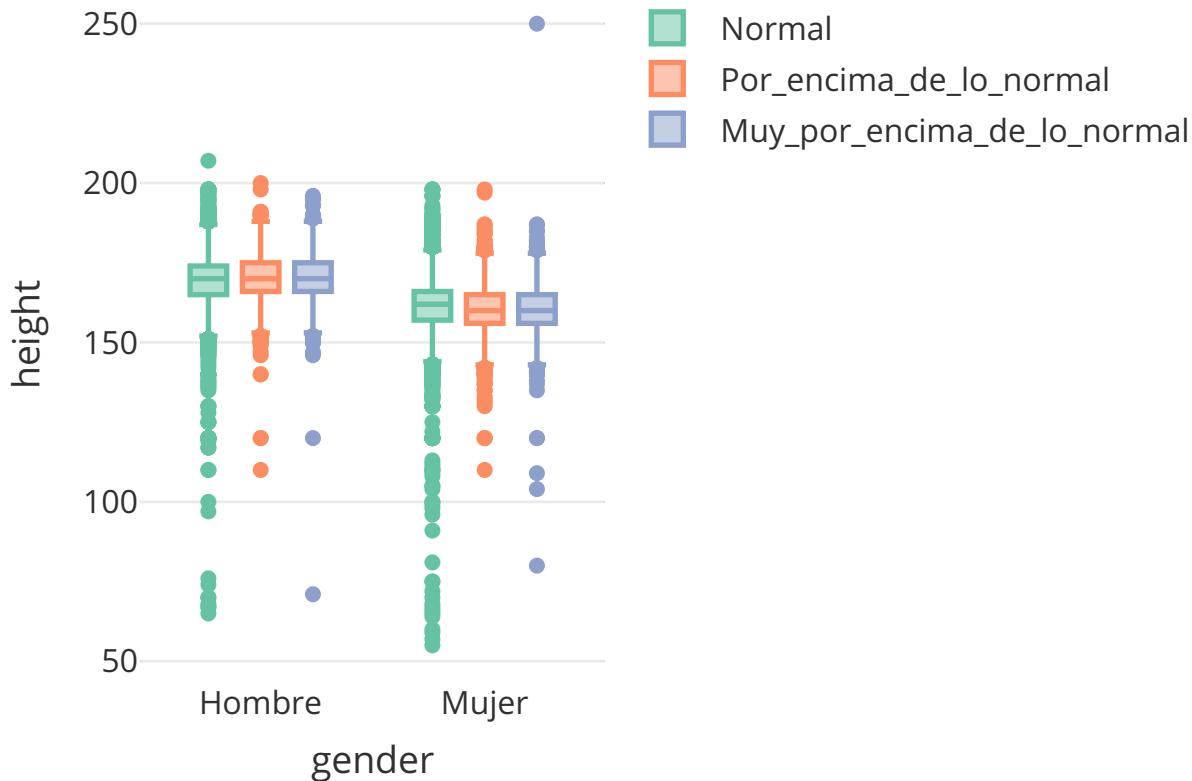




```
fig <- plot_ly(cardio, x = ~gender, y = ~height, color = ~cholesterol, type = "box")
fig <- fig %>% layout(boxmode = "group")

fig
```





Hemos podido observar que todas las variables presentan valores atípicos, los cuales trataremos a medida que vayamos estudiando cada una de las variables. En este sentido, optaremos por eliminar aquellas observaciones extremas que no nos interesen, pues como nuestro conjunto de datos es muy grande podemos eliminar observaciones sin que influya en los resultados finales, pues hay información de sobra.

Comenzaremos creando la variable IMC. El índice de masa corporal (IMC) es una razón matemática que asocia la masa y la talla de un individuo, por lo que las variables "height" y "weight" nos serán de utilidad. En base a [menudiet](#), podemos hablar de delgadez extrema cuando el IMC del paciente está por debajo de 18. Por lo tanto, daremos un margen establecido a criterio personal y eliminaremos del dataset aquellas personas cuyo IMC sea menor que 15, pues se corresponderán con valores muy extremos.

Lo primero será crear la nueva variable referente al IMC. El cálculo de este es  $IMC = \frac{Peso(kg)}{Altura^2(m)}$

```
# Creamos la variable
cardio$imc <- (cardio$weight)/((cardio$height)/100)^2

# Creamos un nuevo dataset en el que borramos las observaciones donde el IMC
# sea menor que 18.
cardio_clean <- cardio[!(cardio$imc < 15),]
```

Ahora estudiaremos la altura. Veamos el valor máximo y mínimo que puede tomar la variable

```
cat("El valor mínimo de 'height' es:", min(cardio_clean$height), "\n")
```

```
## El valor mínimo de 'height' es: 55
```

```
cat("El valor máximo de 'height' es:", max(cardio_clean$height), "\n")
```

```
## El valor máximo de 'height' es: 207
```

Eliminaremos las observaciones de las personas con una estatura extremadamente baja, las cuales estarían debajo de los 145 centímetros en el caso de los hombres y 136 en el caso de las mujeres, basándonos en la siguiente [web](#). En el caso de personas altas, no es tan extraño ver a alguien con algo más de dos metros, por



lo que estas observaciones las dejaremos

```
cardio_clean <- cardio_clean[!(cardio_clean[, "height"] < 145
                           & cardio_clean[, "gender"] == 'Hombre'), ]
cardio_clean <- cardio_clean[!(cardio_clean[, "height"] < 136
                           & cardio_clean[, "gender"] == 'Mujer'), ]
```

Las variables presión arterial sistólica (PS) y diastólica (PD) las podemos utilizar para crear la variable referente a la presión arterial media (PAM). Gracias a [MediCalc](#) sabemos que la fórmula para calcular la PAM es  $PAM = \frac{PS+2PD}{3}$ . Si recordamos los gráficos de caja para las variables “systolic\_blood\_pressure” y “diastolic\_blood\_pressure”, estas tenían algunos valores demasiado pequeños y grandes, lo que hace pensar que hayan sido introducidos por error. Si nos fijamos en sus mínimos y máximos

```
cat("El valor mínimo de 'systolic_blood_pressure' es :",min(cardio$systolic_blood_pressure),"\n")
## El valor mínimo de 'systolic_blood_pressure' es : -150
cat("El valor mínimo de 'diastolic_blood_pressure' es :",min(cardio$diastolic_blood_pressure),"\n")

## El valor mínimo de 'diastolic_blood_pressure' es : -70
cat("El valor máximo de 'systolic_blood_pressure' es :",max(cardio$systolic_blood_pressure),"\n")
## El valor máximo de 'systolic_blood_pressure' es : 16020
cat("El valor máximo de 'diastolic_blood_pressure' es :",max(cardio$diastolic_blood_pressure),"\n")

## El valor máximo de 'diastolic_blood_pressure' es : 11000
```

no tiene sentido que una persona tenga valores negativos de presión arterial sistólica o diastólica porque, según se explica en [Mayo Clinic](#), si la primera es menor que 90 mmHg y la segunda es menor que 60 mmHg, la presión es más baja de lo normal. Asimismo, tampoco es lógico que alguien tenga estos valores tan sumamente altos ya que, en base a [soyvida](#), unos niveles de presión arterial sistólica y diastólica mayores que 180 y 120, respectivamente, son indicadores de hipertensión. Por lo tanto, a modo de criterio personal elegiremos un margen de 30 mmHg para los valores de presión baja y de 40 mmHg para los de presión alta. Además, utilizando la presión arterial media, eliminaremos también las observaciones cuyos valores de esta sean menores que 40 o mayores que 250. El primer límite lo elegimos de nuevo en base a criterio personal, pues si calculamos la PAM para PS = 90 y PD = 60 obtenemos un resultado de 70 mmHg, así que escogeremos un margen de 30; y el segundo límite lo pondremos en referencia a [MD.SAÚDE](#), en el cual se explica que algunos pacientes llegan a tener 240 o 250 mmHg de presión máxima durante el pico hipertensivo.

```
# Creamos la variable PAM
cardio_clean$blood_pressure <- (cardio_clean$systolic_blood_pressure +
                                 2*cardio_clean$diastolic_blood_pressure)/3

# Eliminamos las observaciones que no nos interesan
cardio_clean <- cardio_clean[!(
    cardio_clean[,"systolic_blood_pressure"] < 60), ]
cardio_clean <- cardio_clean[!(
    cardio_clean[,"systolic_blood_pressure"] > 220), ]
cardio_clean <- cardio_clean[!(
    cardio_clean[,"diastolic_blood_pressure"] < 30), ]
cardio_clean <- cardio_clean[!(
    cardio_clean[,"diastolic_blood_pressure"] > 160), ]
cardio_clean <- cardio_clean[!(cardio_clean[, "blood_pressure"] > 250), ]
cardio_clean <- cardio_clean[!(cardio_clean[, "blood_pressure"] < 40), ]
cardio_clean <- cardio_clean[!(cardio_clean[, "blood_pressure"] > 250), ]
```

Vemos cómo quedan los boxplots de las variables numéricas una vez nos hemos desecho de las observaciones

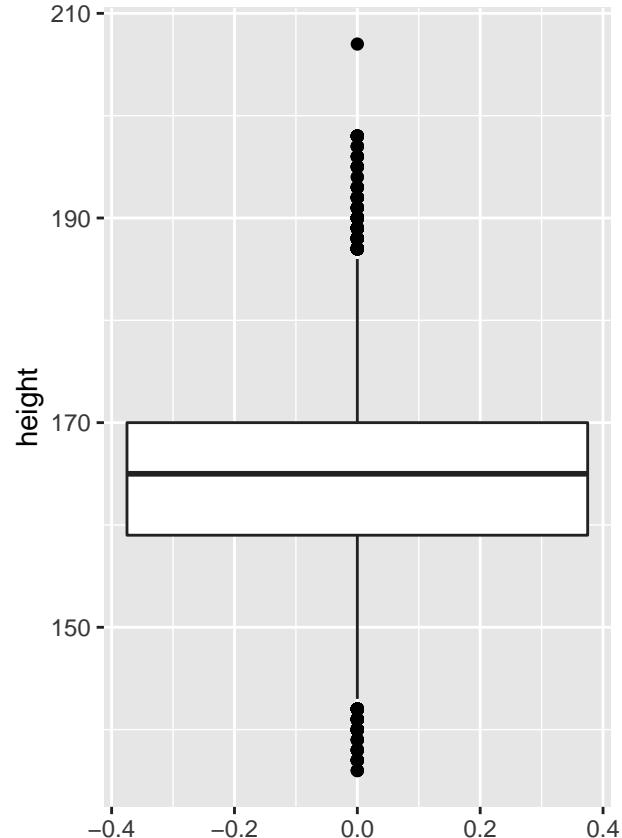
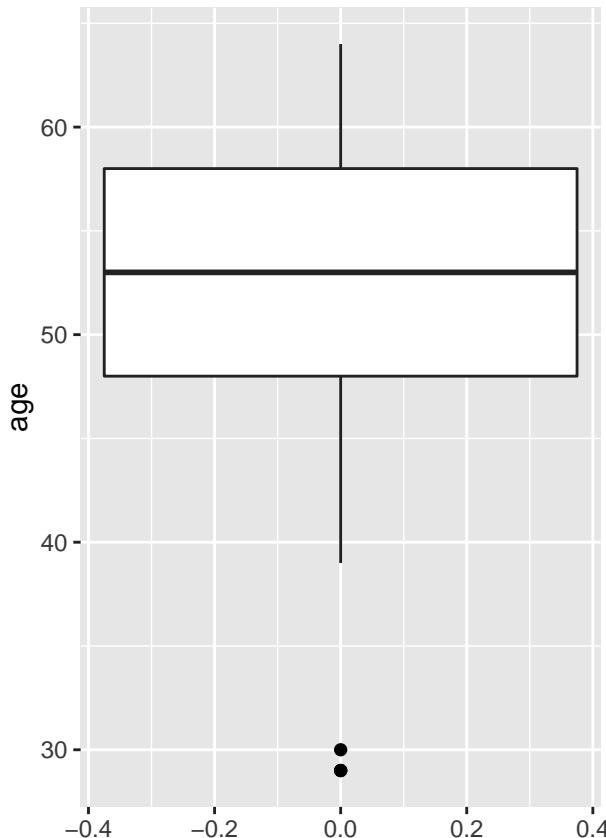


que no necesitábamos

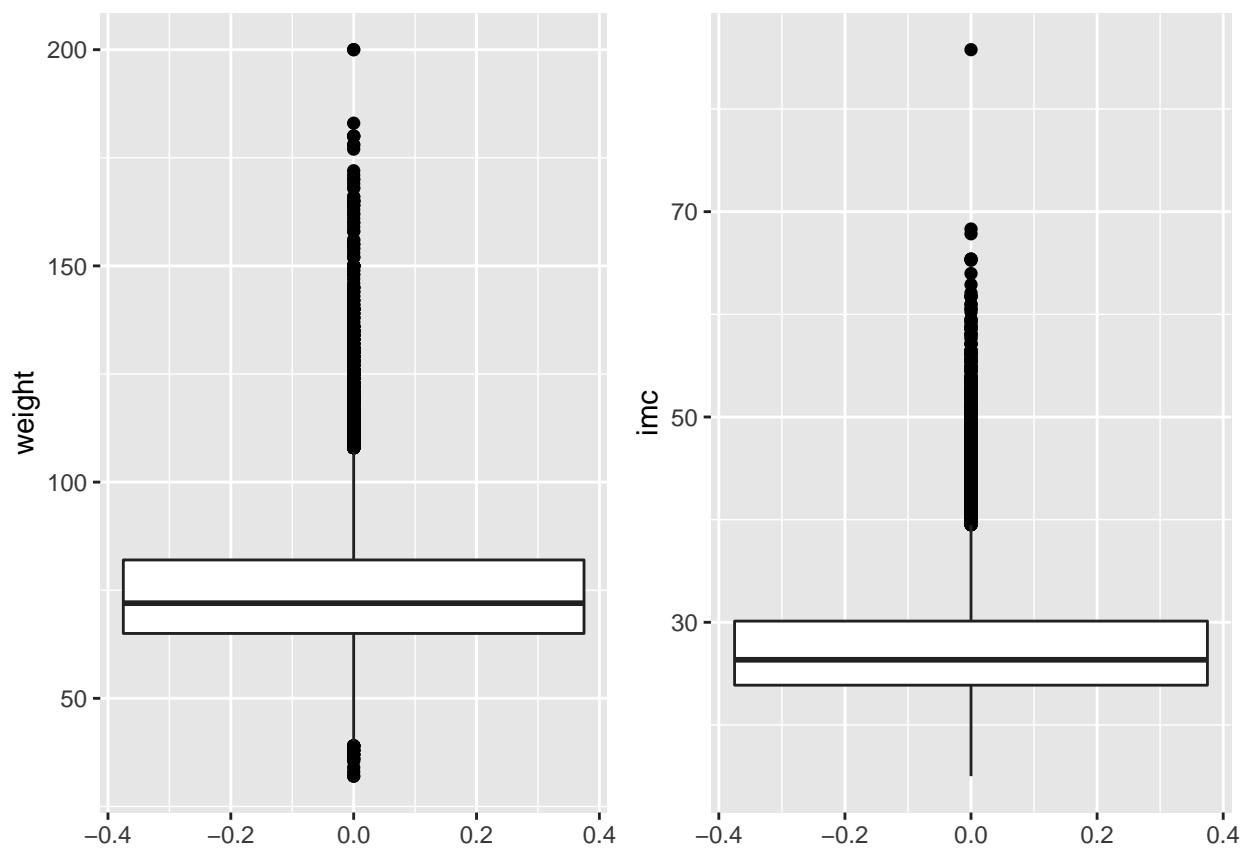
```
a <- ggplot(cardio_clean, aes(y=age)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)
h <- ggplot(cardio_clean, aes(y=height)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)
w <- ggplot(cardio_clean, aes(y=weight)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)
sbp <- ggplot(cardio_clean, aes(y=systolic_blood_pressure)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)
dbp <- ggplot(cardio_clean, aes(y=diastolic_blood_pressure)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)
i <- ggplot(cardio_clean, aes(y=imc)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)
bp <- ggplot(cardio_clean, aes(y=blood_pressure)) +
    geom_boxplot(outlier.colour="black",outlier.shape=16,outlier.size=2,
                 notch=FALSE)

ggarrange(a,h,w,i,sbp,dbp,bp, ncol = 2)
```

```
## $`1`
```

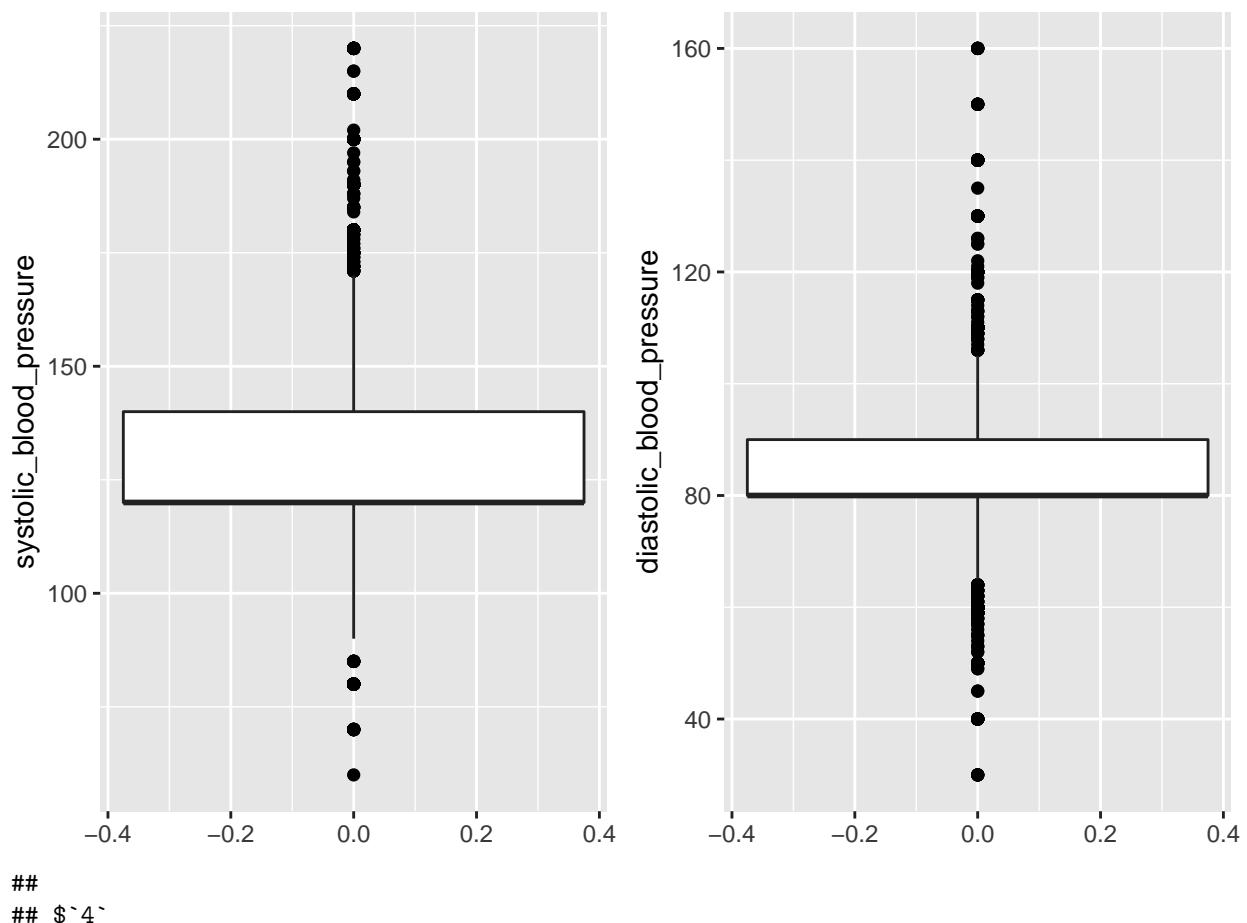


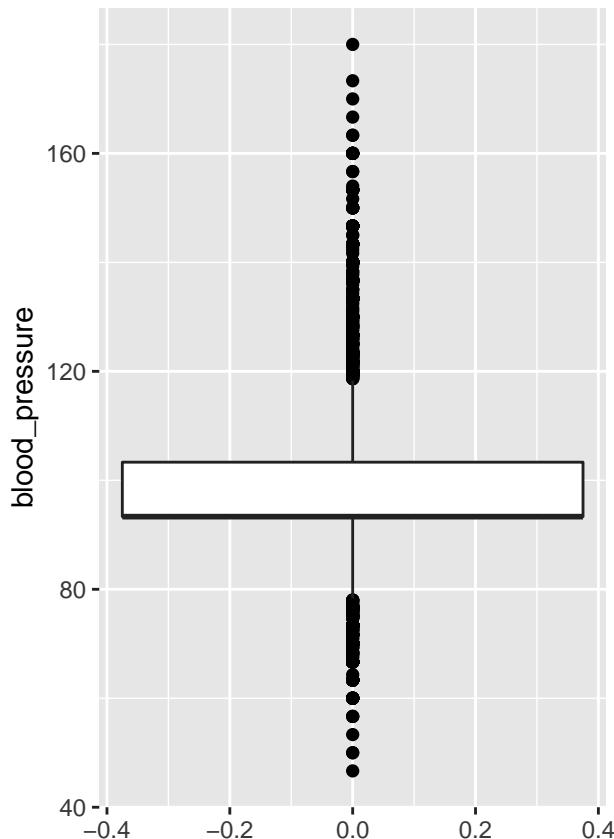
```
##  
## $^2`
```



```
##  
## $^3`
```







```
##  
## attr(,"class")  
## [1] "list"      "ggarrange"
```

Como vemos aún se aprecian valores atípicos, pero estos tienen sentido y pueden aportar información valiosa sobre los pacientes del dataset, por lo que tomaremos la decisión de no tratarlos y seguir el estudio con ellos.

Si recordamos el apartado anterior, la variable “diastolic\_blood\_pressure” contenía una serie de valores iguales a 0. Por lo tanto, vamos a comprobar la situación actual del conjunto de datos tras haber tratado los valores atípicos

```
cat("\nContamos el número de ceros por atributo: \n")
```

```
##  
## Contamos el número de ceros por atributo:  
colSums(cardio == 0)
```

	age	gender	height
##	0	0	0
##	weight	systolic_blood_pressure	diastolic_blood_pressure
##	0	0	21
##	cholesterol	glucose	smoking
##	0	0	0
##	alcohol_intake	physical_activity	cardiovascular_disease
##	0	0	0
##	imc		
##	0		

Podemos ver que ahora ninguna variable numérica contiene valores iguales a 0, por lo que hemos corregido



las incidencias.



## 4. Análisis de datos

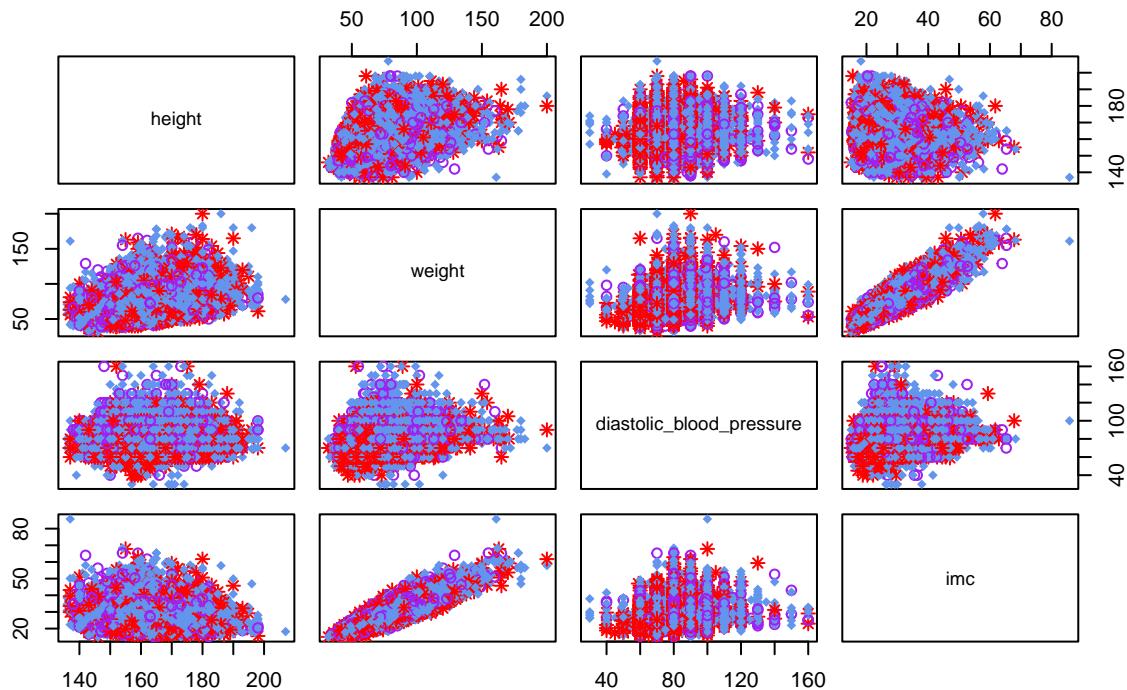
### 4.0. EDA

#### 4.0.1 Matriz con gráficos de dispersión de las variables cuantitativas

```
group <- NA
group[cardio_clean$age < 45] <- 1
group[cardio_clean$age >= 45 & cardio_clean$age <= 60] <- 2
group[cardio_clean$age > 60] <- 3

pairs(~ height+weight+diastolic_blood_pressure+diastolic_blood_pressure+imc,
      data = cardio_clean, col = c("red", "cornflowerblue", "purple")[group],
      pch = c(8, 18, 1)[group], main = "Pairs Cardio", rowlattop = TRUE,
      gap = 1, cex.labels = NULL, font.labels = 1)
```

**Pairs Cardio**



#### 4.0.2 Cálculo de los coeficientes de correlación entre variables

Estudiamos la correlación solo para variables numéricas

```
cor(cardio_clean[sapply(cardio_clean, is.numeric)])
```

```
##                               age      height      weight
## age                         1.0000000 -0.08680899 0.05533615
## height                      -0.08680899  1.00000000 0.31200237
## weight                       0.05533615  0.31200237 1.00000000
## systolic_blood_pressure     0.20872746  0.01800591 0.26898541
## diastolic_blood_pressure    0.15331978  0.03566157 0.24991179
## imc                          0.10382484 -0.19249427 0.86761588
## blood_pressure                0.19411692  0.02974268 0.28047232
##                               systolic_blood_pressure diastolic_blood_pressure
```



```

## age                      0.20872746   0.15331978
## height                   0.01800591   0.03566157
## weight                   0.26898541   0.24991179
## systolic_blood_pressure 1.00000000   0.70130219
## diastolic_blood_pressure 0.70130219   1.00000000
## imc                      0.26845516   0.23975643
## blood_pressure            0.91074364   0.93309956
##                               imc blood_pressure
## age                      0.1038248    0.19411692
## height                   -0.1924943   0.02974268
## weight                   0.8676159    0.28047232
## systolic_blood_pressure  0.2684552    0.91074364
## diastolic_blood_pressure 0.2397564    0.93309956
## imc                      1.0000000    0.27432169
## blood_pressure            0.2743217    1.00000000

```

A continuación, calculamos la correlación entre todas las variables, convirtiendo las variables categóricas en numéricas.

```

# Para variables numéricas y no numéricas. Será necesario transformar
# todas las variables en tipo numérico antes
cardio_numeric <- data.frame(lapply(cardio_clean, function(x) as.numeric(x)))
cor(cardio_numeric)

```

```

##                               age      gender      height      weight
## age                      1.00000000  0.023382018 -0.086808987  0.05533615
## gender                   0.02338202  1.000000000 -0.525197962 -0.15732513
## height                   -0.08680899 -0.525197962  1.000000000  0.31200237
## weight                   0.05533615 -0.157325134  0.312002366  1.00000000
## systolic_blood_pressure  0.20872746 -0.060929956  0.018005909  0.26898541
## diastolic_blood_pressure 0.15331978 -0.067584616  0.035661566  0.24991179
## cholesterol              0.15536464  0.036387677 -0.056284502  0.14165509
## glucose                  0.09923476  0.021112654 -0.021262462  0.10738146
## smoking                  -0.04815771 -0.338986008  0.196987552  0.06695342
## alcohol_intake           -0.02914764 -0.171547413  0.098850878  0.06796151
## physical_activity         -0.01080125 -0.005894587 -0.009803114 -0.01792861
## cardiovascular_disease   0.23945019 -0.007220868 -0.012641901  0.17999456
## imc                      0.10382484  0.112026507 -0.192494274  0.86761588
## blood_pressure             0.19411692 -0.069890017  0.029742676  0.28047232
##                               systolic_blood_pressure diastolic_blood_pressure
## age                      0.208727462   0.1533197765
## gender                   -0.060929956   -0.0675846159
## height                   0.018005909   0.0356615657
## weight                   0.268985408   0.2499117892
## systolic_blood_pressure  1.000000000   0.7013021873
## diastolic_blood_pressure 0.701302187   1.0000000000
## cholesterol              0.194459563   0.1600171004
## glucose                  0.093184371   0.0761917306
## smoking                  0.026862634   0.0252890740
## alcohol_intake           0.032362043   0.0419815012
## physical_activity         -0.001465257   0.0001972699
## cardiovascular_disease   0.425979687   0.3364235460
## imc                      0.268455160   0.2397564287
## blood_pressure             0.910743636   0.9330995571
##                               cholesterol      glucose      smoking alcohol_intake

```



```

## age                      0.155364637  0.099234756 -0.048157710 -0.029147643
## gender                   0.036387677  0.021112654 -0.338986008 -0.171547413
## height                  -0.056284502 -0.021262462  0.196987552  0.098850878
## weight                   0.141655091  0.107381463  0.066953417  0.067961511
## systolic_blood_pressure  0.194459563  0.093184371  0.026862634  0.032362043
## diastolic_blood_pressure 0.160017100  0.076191731  0.025289074  0.041981501
## cholesterol              1.000000000  0.451336699  0.010094195  0.035587487
## glucose                  0.451336699  1.000000000 -0.005614405  0.010811345
## smoking                  0.010094195 -0.005614405  1.000000000  0.340182605
## alcohol_intake           0.035587487  0.010811345  0.340182605  1.000000000
## physical_activity         0.009069461 -0.007564442  0.025178741  0.024927383
## cardiovascular_disease   0.221412943  0.089736042 -0.016272507 -0.008247764
## imc                      0.174152716  0.120790977 -0.033628186  0.017826145
## blood_pressure             0.190799058  0.091147606  0.028201694  0.040646135
##                                         physical_activity cardiovascular_disease      imc
## age                      -0.0108012528          0.239450185  0.10382484
## gender                   -0.0058945868          -0.007220868  0.11202651
## height                  -0.0098031138          -0.012641901 -0.19249427
## weight                   -0.0179286140          0.179994558  0.86761588
## systolic_blood_pressure -0.0014652567          0.425979687  0.26845516
## diastolic_blood_pressure 0.0001972699          0.336423546  0.23975643
## cholesterol              0.0090694613          0.221412943  0.17415272
## glucose                  -0.0075644424          0.089736042  0.12079098
## smoking                  0.0251787409          -0.016272507 -0.03362819
## alcohol_intake           0.0249273831          -0.008247764  0.01782614
## physical_activity         1.0000000000          -0.037439868 -0.01403546
## cardiovascular_disease   -0.0374398678          1.000000000  0.19177333
## imc                      -0.0140354615          0.191773334  1.00000000
## blood_pressure             -0.0006248960          0.409788615  0.27432169
##                                         blood_pressure
## age                      0.194116923
## gender                   -0.069890017
## height                  0.029742676
## weight                   0.280472324
## systolic_blood_pressure  0.910743636
## diastolic_blood_pressure 0.933099557
## cholesterol              0.190799058
## glucose                  0.091147606
## smoking                  0.028201694
## alcohol_intake           0.040646135
## physical_activity        -0.000624896
## cardiovascular_disease  0.409788615
## imc                      0.274321688
## blood_pressure            1.000000000

```

#### 4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).

Vamos a realizar la discretización de una serie de variables y su posterior visualización, las cuales nos permitirán realizar un análisis más a fondo.

Comenzaremos mediante la variable “edad”, la cual dividiremos en varias categorías en función de la web [inreality](#). Como en la página web hay varios grupos de edades, lo primero que haremos será ver el valor mínimo y máximo que tiene la variable “edad”. Así, podremos ver los grupos de edades mediante los que discretizar



```
cat("La edad minima es :", min(cardio_clean$age), "\n")
```

```
## La edad minima es : 29
```

```
cat("La edad maxima es :", max(cardio_clean$age), "\n")
```

```
## La edad maxima es : 64
```

Como la edad de las personas del dataset comienza en 29, no utilizaremos el primer grupo de edad (Youth (<18)). Crearemos los grupos Adulto Joven (18-35 años), Adulto (36-55 años) y Senior (56 años y mayores)

```
cardio_clean$group_age <- cut(cardio_clean$age,  
                                breaks = c(18,35,55,Inf),  
                                labels = c("Adulto_Joven", "Adulto", "Senior"))
```

Discretizaremos ahora la altura (height). Como dependiendo del país la altura de las personas varía mucho, no hay un criterio específico para ver si una persona es alta o baja, por lo que cada grupo creado será un intervalo que varíe en 15 centímetros

```
cardio_clean$group_height <- cut(cardio_clean$height,  
                                    breaks = c(-Inf,150,165,180,195,Inf),  
                                    labels = c("(-Inf, 150)", "(151, 165)",  
                                              "(166, 180)", "(181, 195)",  
                                              "(196, Inf)"))
```

Haremos lo mismo con la variable “weight”, la cual dividiremos en intervalos de 30 kilogramos cada uno

```
# Vemos los mínimo y máximo de la variable  
cat("El peso minimo es :", min(cardio_clean$weight),"\n")
```

```
## El peso minimo es : 32
```

```
cat("El peso máximo es :", max(cardio_clean$weight),"\n")
```

```
## El peso máximo es : 200
```

Discretizamos la variable

```
cardio_clean$group_weight <- cut(cardio_clean$weight,  
                                    breaks = c(-Inf,80,120,160,Inf),  
                                    labels = c("(-Inf, 80)", "(81, 120)",  
                                              "(121, 160)", "(161, Inf)"))
```

Podemos discretizar también la variable que creamos referente al IMC para dividir las personas en función del peso. Así, podremos saber si una persona tiene un peso normal o no ya que esta variable esta construida en base al peso y la altura del paciente. En este sentido, nos guiamos por la web [Center For Disease Control and Prevention](#)

```
cardio_clean$group_imc <- cut(cardio_clean$imc,  
                                breaks = c(-Inf,18.5,24.9,29.9,Inf),  
                                labels = c("Peso_inferior_al_normal",  
                                          "Peso_normal", "Sobrepeso",  
                                          "Obesidad"))
```

En cuanto a la otra variable que creamos, “blood\_pressure”, podemos discretizarla gracias a los valores dados por [Seguros Bilbao](#), en donde se explica que los niveles normales van desde los 90 mmHg para la presión arterial sistólica y los 60mmHg para la diastólica, expresado como 90/60mmHg, hasta los 120/80mmHg. Valores por debajo de los 90/60mmHg significan que la tensión está baja y si está por encima de los 140/90 mmHg existe hipertensión. Además, valores entre los 120/80 mmHg y los 139/89mmHg se consideran normales-altos.



```
# Discretizamos la variable que acabamos de crear
cardio_clean$hypertension <- cut(cardio_clean$diastolic_blood_pressure,
                                    breaks = c(-Inf,((90+2*60)/3),
                                              ((120+2*80)/3),
                                              ((139+2*89)/3), Inf),
                                    labels = c("Tensión_baja",
                                              "Presión_arterial_normal",
                                              "Presión_arterial_normal_alta",
                                              "Hipertensión"))
```

Representamos ahora la distribución de las variables discretas que hemos creado junto con el resto de variables de este tipo del dataset mediante gráficos de barras

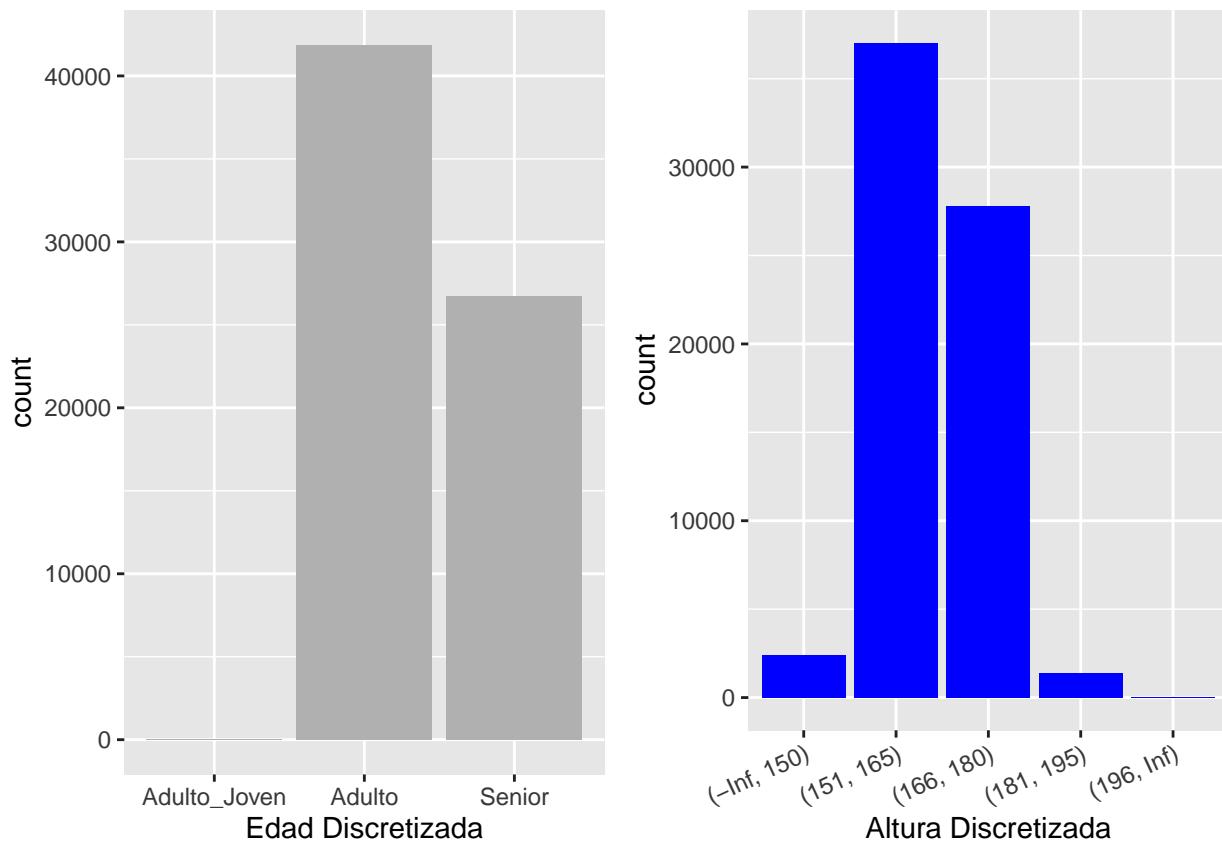
```
ga <- ggplot(cardio_clean, aes(group_age)) + geom_bar(fill='grey') +
  xlab("Edad Discretizada")
gh <- ggplot(cardio_clean, aes(group_height)) + geom_bar(fill='blue') +
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +
  xlab("Altura Discretizada")
gw <- ggplot(cardio_clean, aes(group_weight)) + geom_bar(fill='pink') +
  xlab("Peso Discretizada")
gi <- ggplot(cardio_clean, aes(group_imc)) + geom_bar(fill='lightblue') +
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +
  xlab("IMC Discretizada")
ghy <- ggplot(cardio_clean, aes(hypertension)) + geom_bar(fill='green') +
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +
  xlab("Hipertensión Discretizada")
ge <- ggplot(cardio_clean, aes(gender)) + geom_bar(fill='cadetblue3') +
  xlab("Género")
ch <- ggplot(cardio_clean, aes(cholesterol)) + geom_bar(fill='red') +
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +
  xlab("Colesterol")
gl <- ggplot(cardio_clean, aes(glucose)) + geom_bar(fill='purple') +
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +
  xlab("Glucosa")
sm <- ggplot(cardio_clean, aes(smoking)) + geom_bar(fill='yellow') +
  xlab("Fumador")
ai <- ggplot(cardio_clean, aes(alcohol_intake)) + geom_bar(fill='brown') +
  xlab("Ingesta de alcohol")
pa <- ggplot(cardio_clean, aes(physical_activity)) + geom_bar(fill='orange') +
  xlab("Actividad física")

# Variable objetivo
cd <- ggplot(cardio_clean, aes(cardiovascular_disease)) +
  geom_bar(fill='aquamarine4') +
  xlab("Enfermedad Cardiovascular")

ggarrange(ga,gh,gw,gi,ghy,ge,ch,gl,sm,ai,pa,cd, ncol = 2)

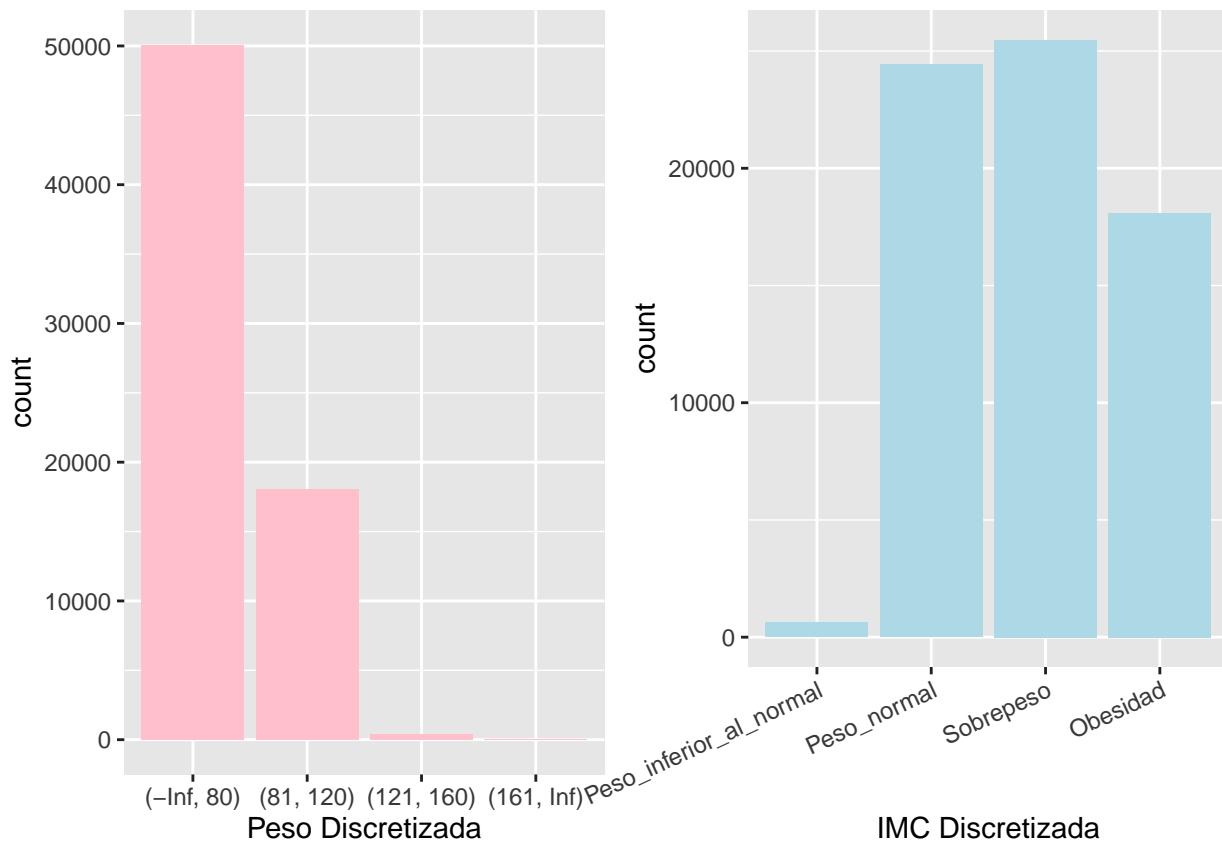
## $`1`
```





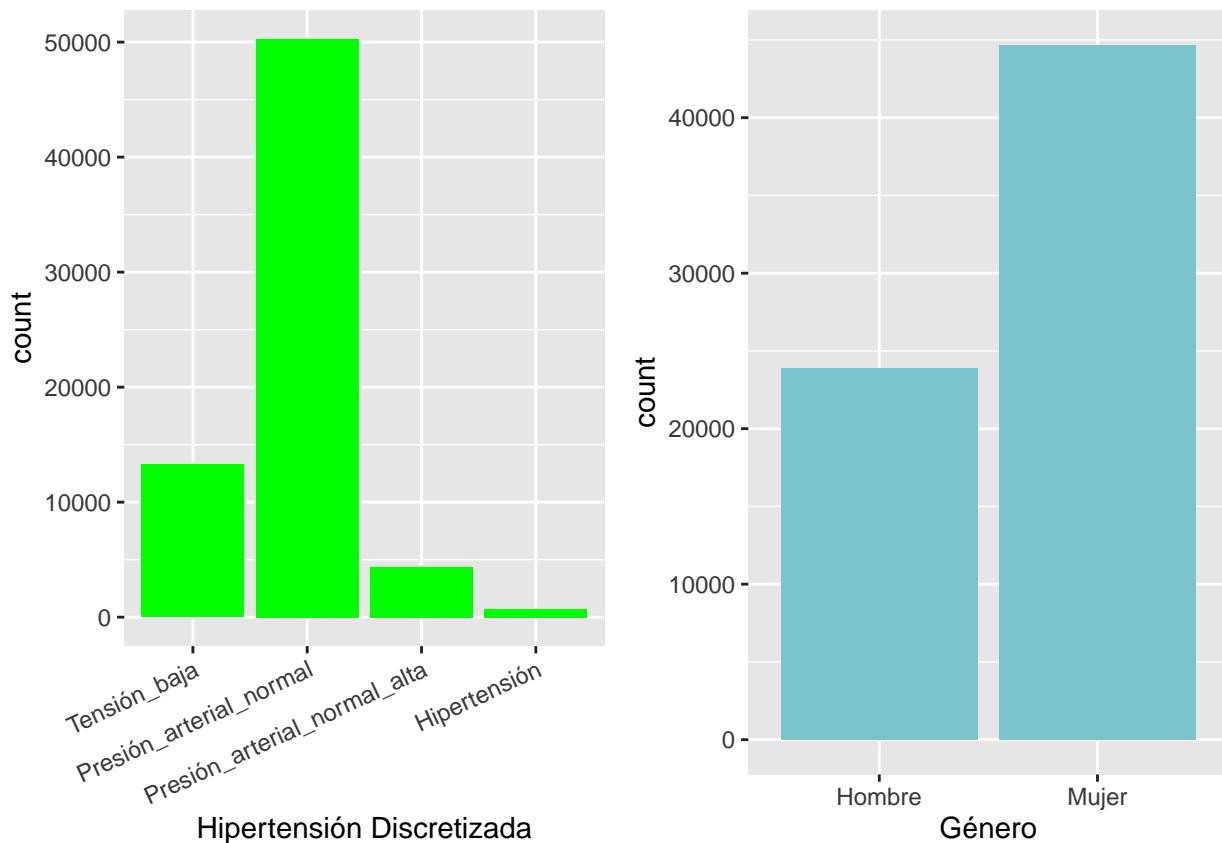
```
##  
## $`2`
```





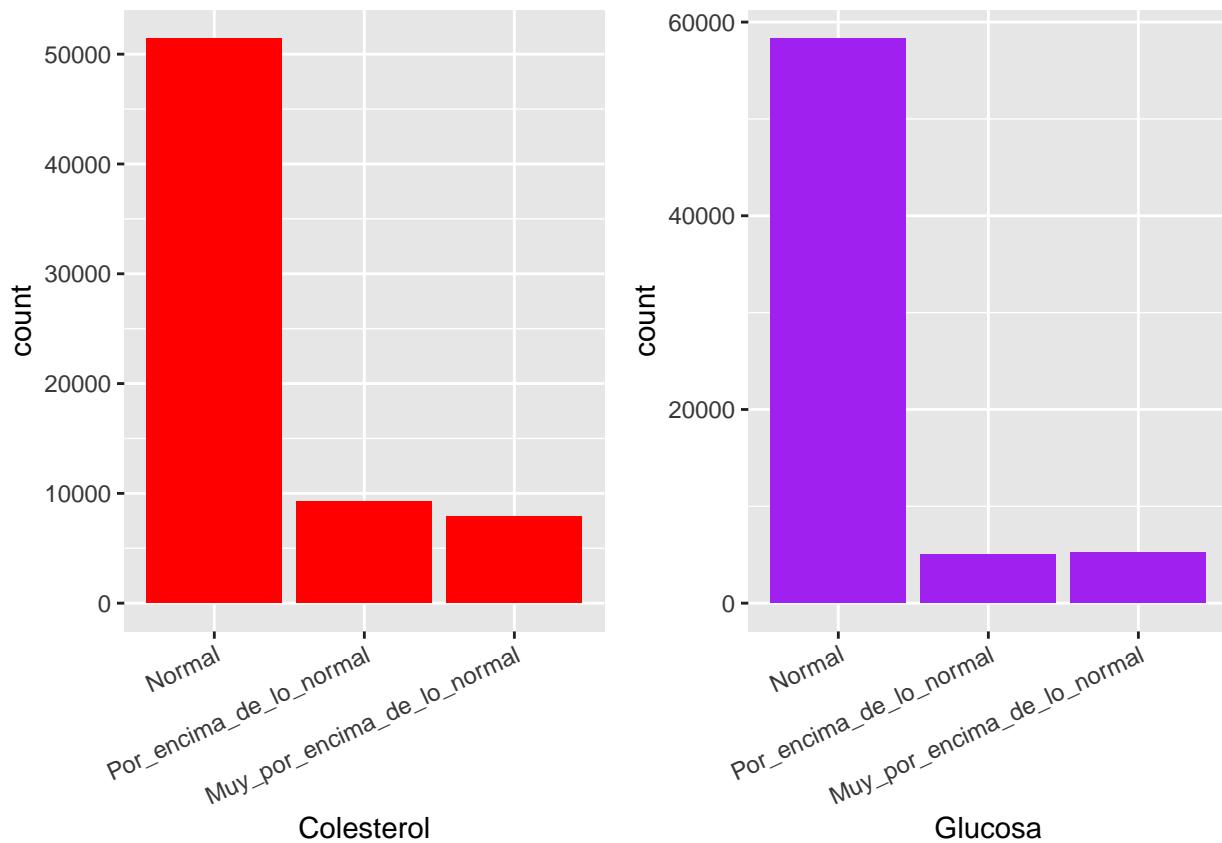
```
##  
## $`3`
```





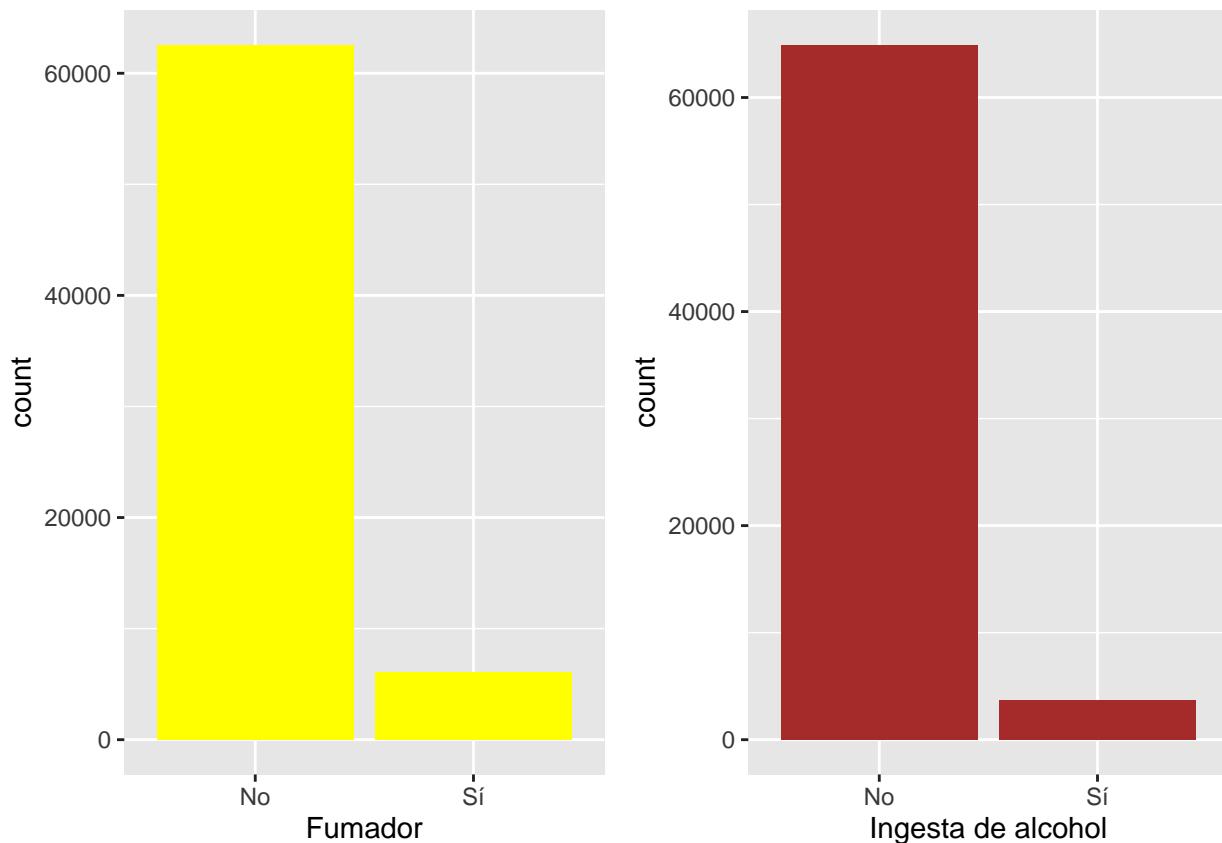
```
##  
## $`4`
```





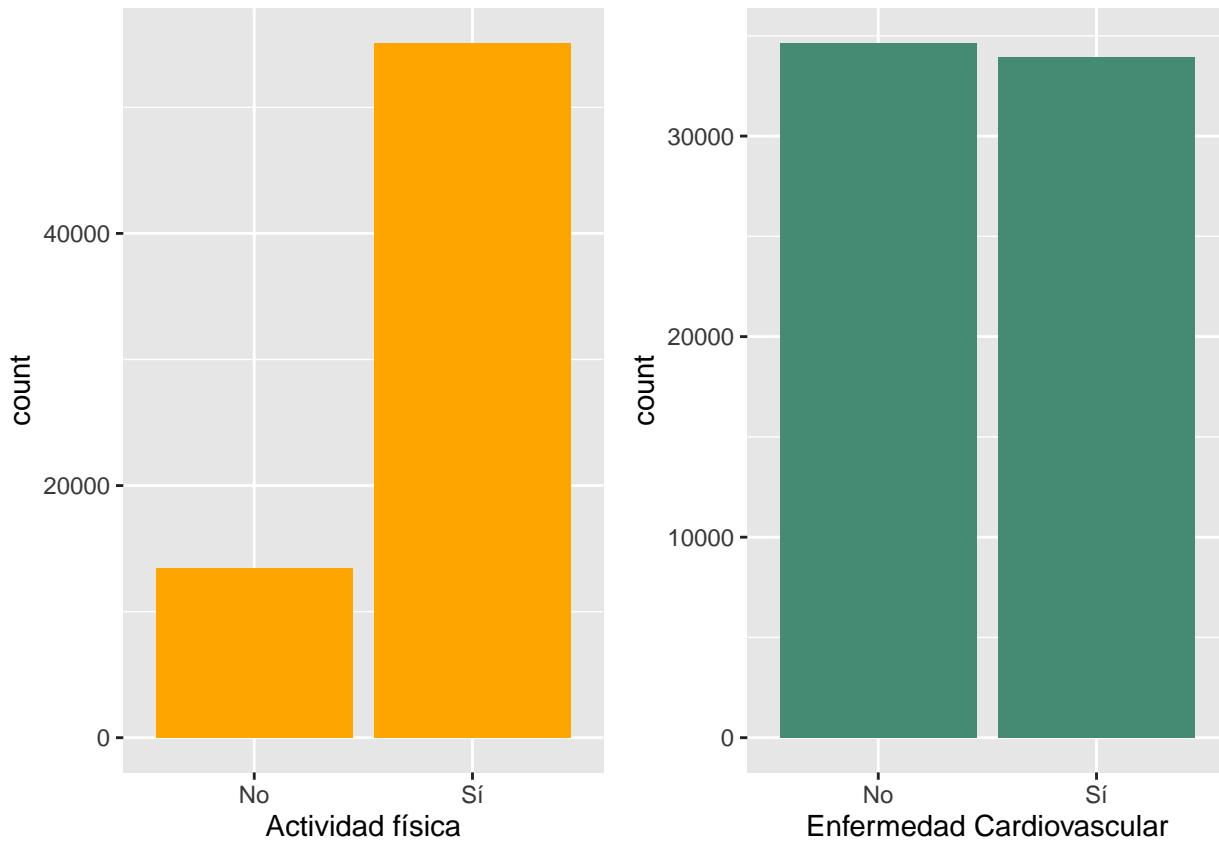
```
##  
## $`5`
```





```
##  
## $`6`
```





```
##  
## attr(),"class")  
## [1] "list"      "ggarrange"
```

Podemos ver que las personas que más abundan en el conjunto de datos son aquellas en edad adulta, con una altura entre 1.51 y 1.65 metros, con un peso menor a 80 kilogramos siendo este normal en función del IMC, con una presión arterial normal, mujeres, con un nivel de colesterol y glucosa normales, no son fumadores ni ingieren alcohol, hacen deporte y no tienen una enfermedad cardiovascular, estando este último grupo bastante igualado con las personas que sí la tienen.

También sería interesante representar la probabilidad de padecer o no una enfermedad cardiovascular para cada una de las variables categóricas

```
ga <- ggplot(cardio_clean, aes(group_age, fill=cardiovascular_disease)) +  
  geom_bar(position = 'fill') + xlab("Edad Discretizada")  
gh <- ggplot(cardio_clean, aes(group_height, fill=cardiovascular_disease)) +  
  geom_bar(position = 'fill') + xlab("Altura Discretizada")  
gw <- ggplot(cardio_clean, aes(group_weight, fill=cardiovascular_disease)) +  
  geom_bar(position = 'fill') + xlab("Peso Discretizada")  
gi <- ggplot(cardio_clean, aes(group_imc, fill=cardiovascular_disease)) +  
  geom_bar(position = 'fill') +  
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +  
  xlab("IMC Discretizada")  
ghy <- ggplot(cardio_clean, aes(hypertension, fill=cardiovascular_disease)) +  
  geom_bar(position = 'fill') +  
  theme(axis.text.x = element_text(angle = 25, hjust=1)) +  
  xlab("Hipertensión Discretizada")  
ge <- ggplot(cardio_clean, aes(gender, fill=cardiovascular_disease)) +
```

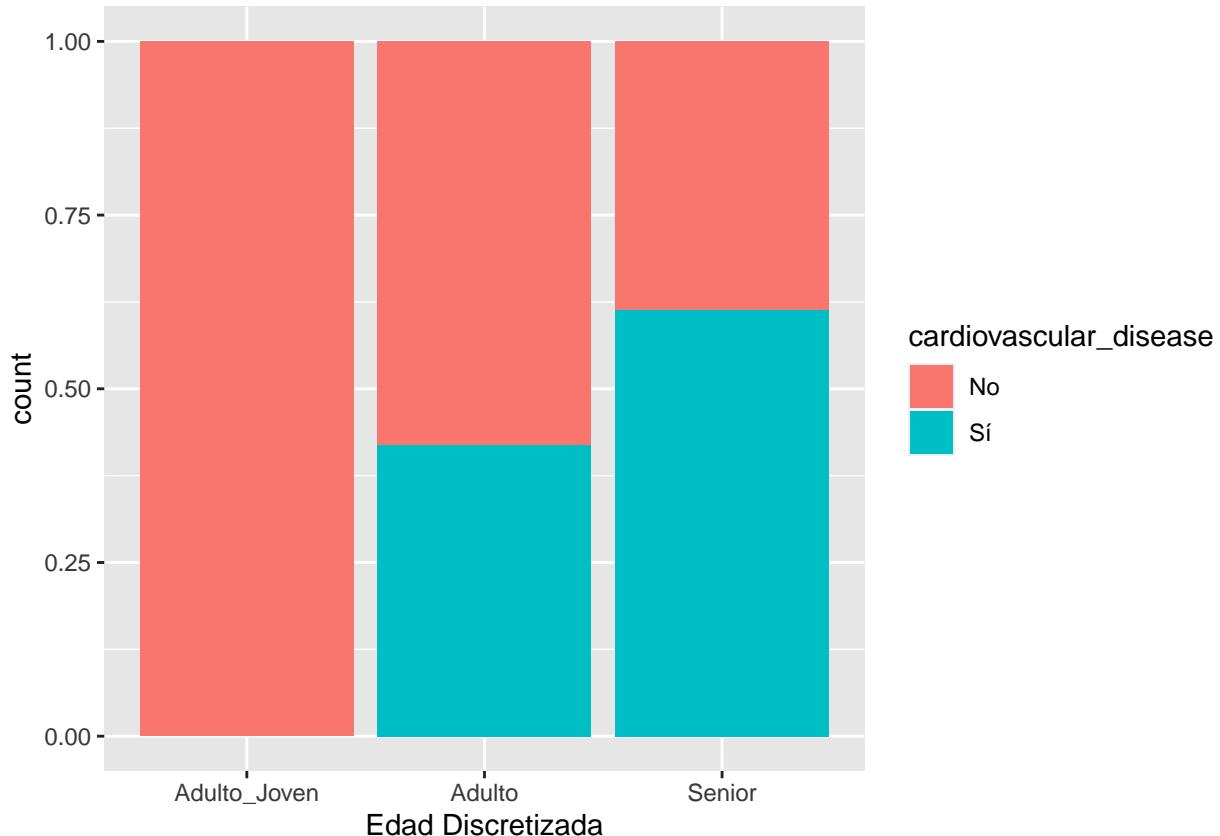


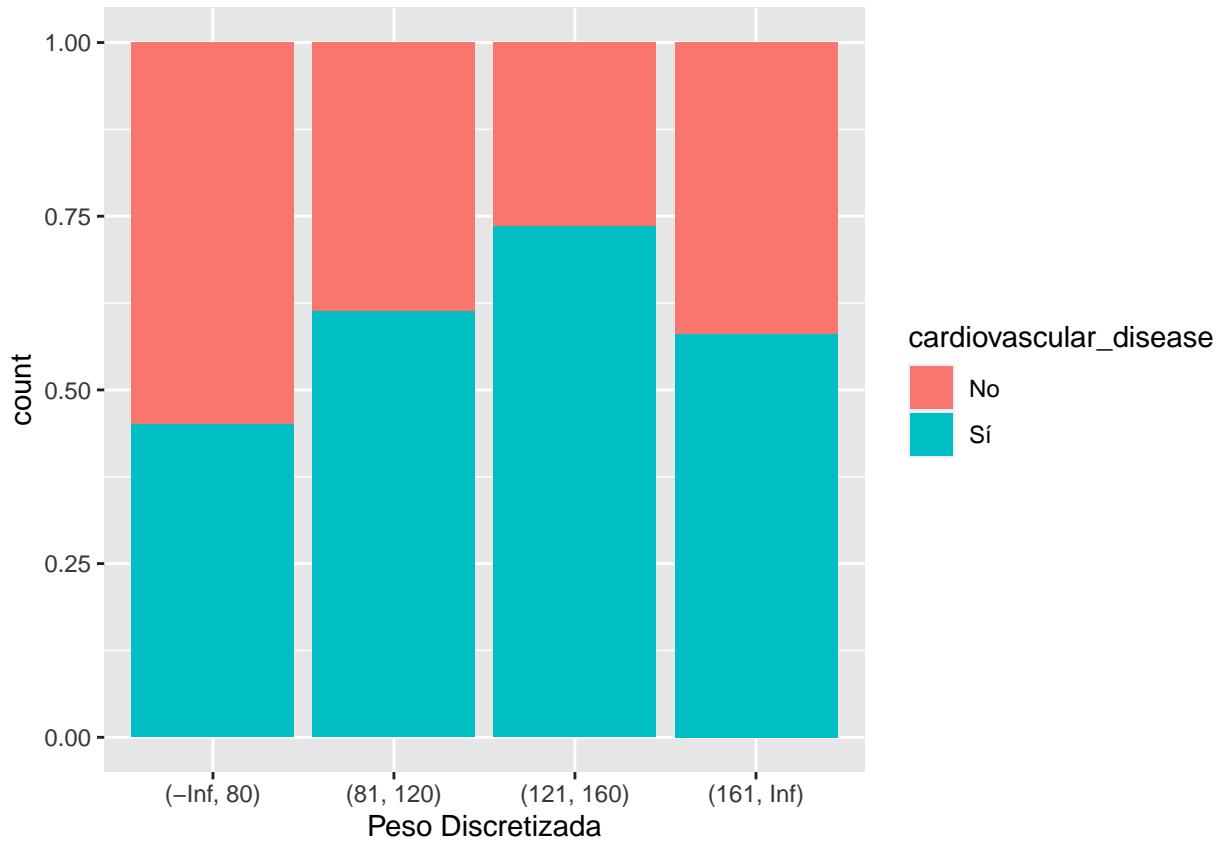
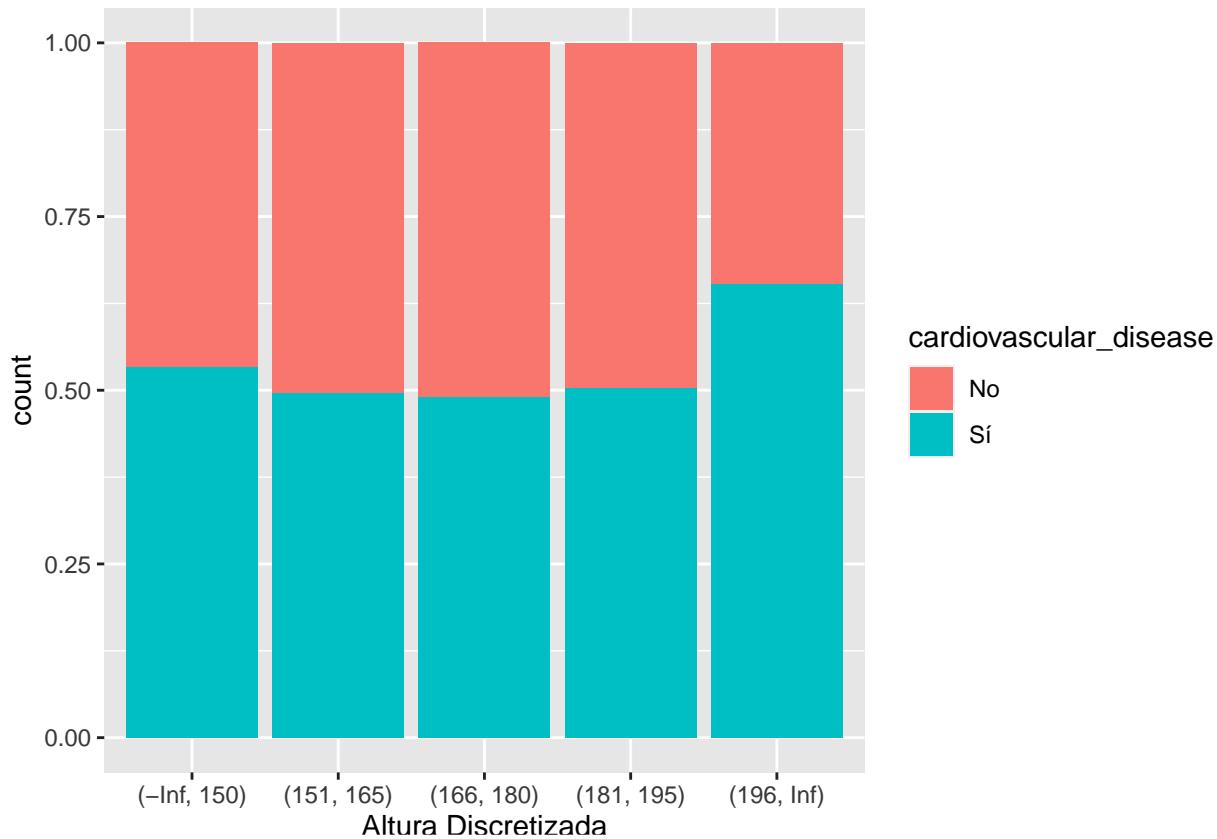
```

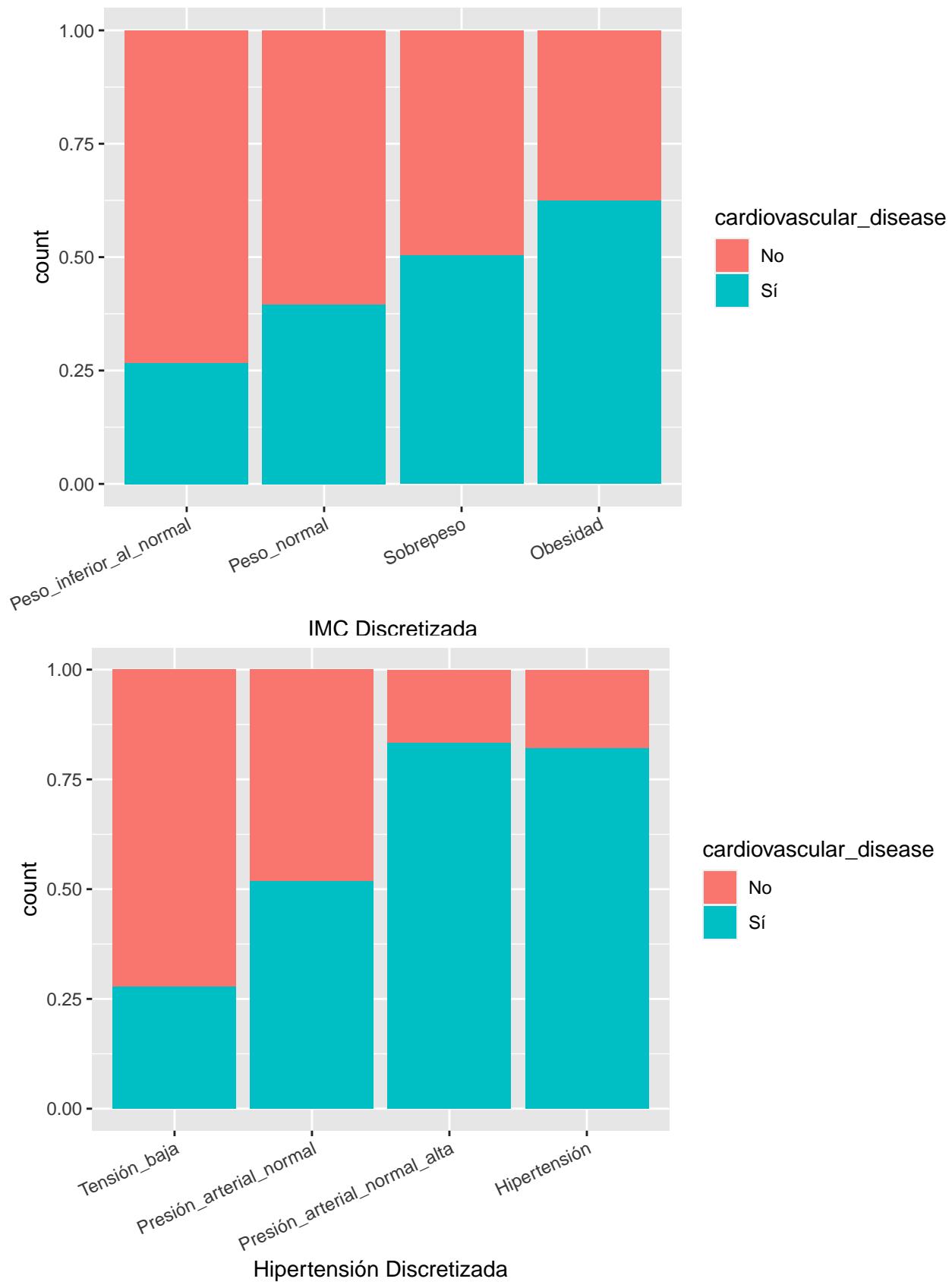
    geom_bar(position = 'fill') + xlab("Género")
ch <- ggplot(cardio_clean, aes(cholesterol, fill=cardiovascular_disease)) +
    geom_bar(position = 'fill') +
    theme(axis.text.x = element_text(angle = 25, hjust=1)) +
    xlab("Colesterol")
gl <- ggplot(cardio_clean, aes(glucose, fill=cardiovascular_disease)) +
    geom_bar(position = 'fill') +
    theme(axis.text.x = element_text(angle = 25, hjust=1)) +
    xlab("Glucosa")
sm <- ggplot(cardio_clean, aes(smoking, fill=cardiovascular_disease)) +
    geom_bar(position = 'fill') + xlab("Fumador")
ai <- ggplot(cardio_clean, aes(alcohol_intake, fill=cardiovascular_disease)) +
    geom_bar(position = 'fill') + xlab("Ingesta de alcohol")
pa <- ggplot(cardio_clean, aes(physical_activity,
                                fill=cardiovascular_disease)) +
    geom_bar(position = 'fill') + xlab("Actividad física")

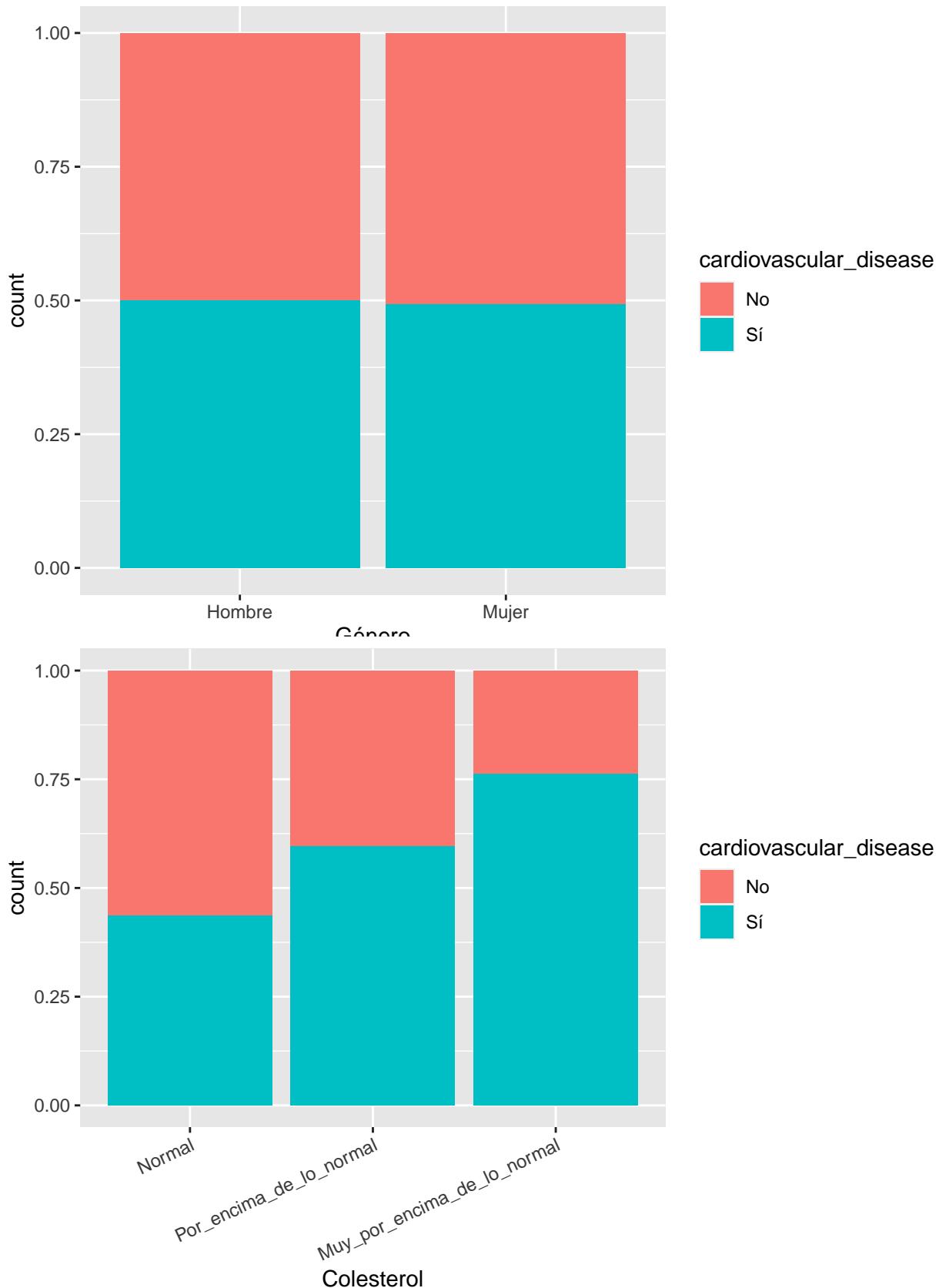
# Ploteamos
ga;gh;gw;gi;ghy;ge;ch;gl;sm;ai;pa

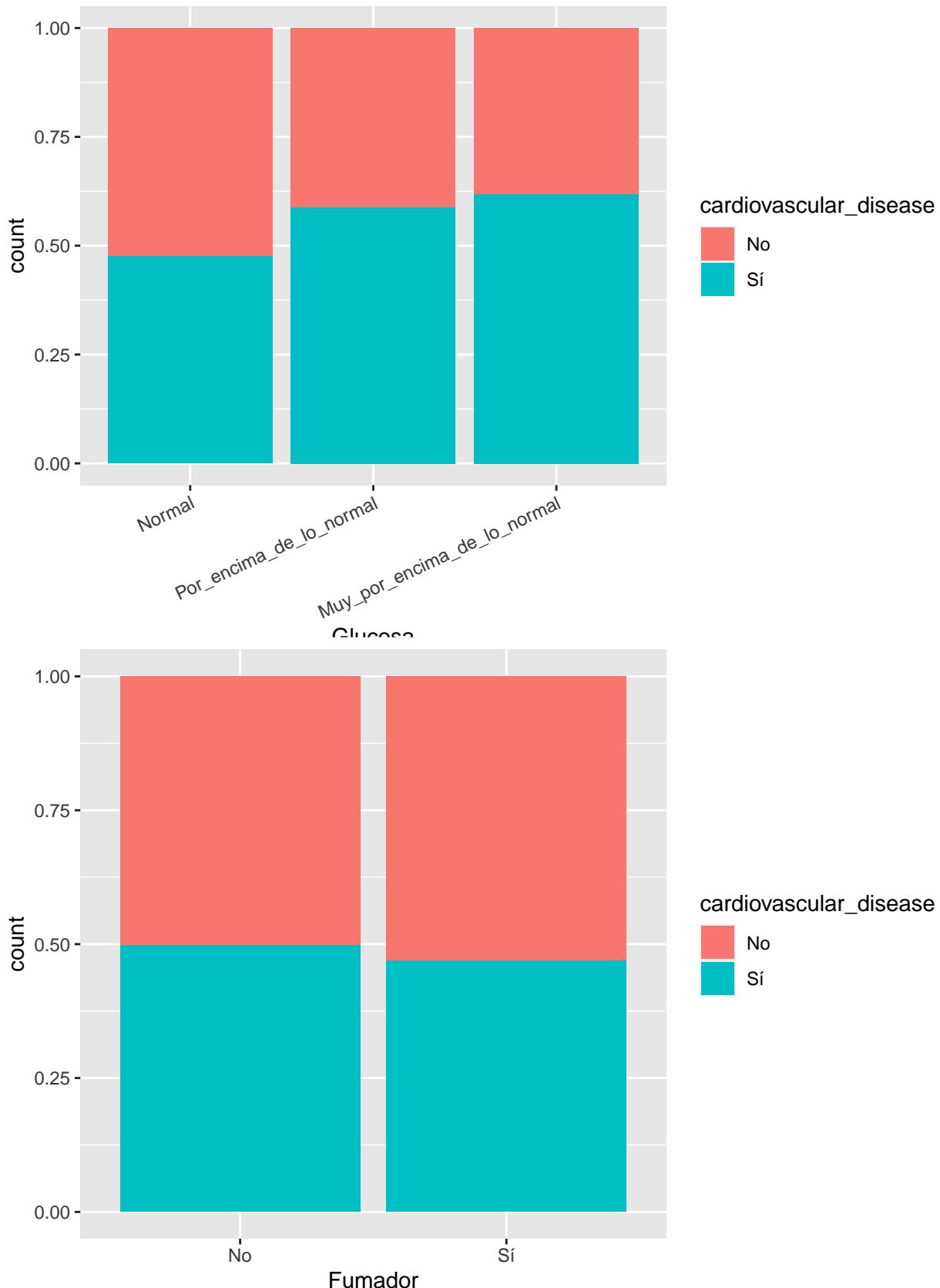
```

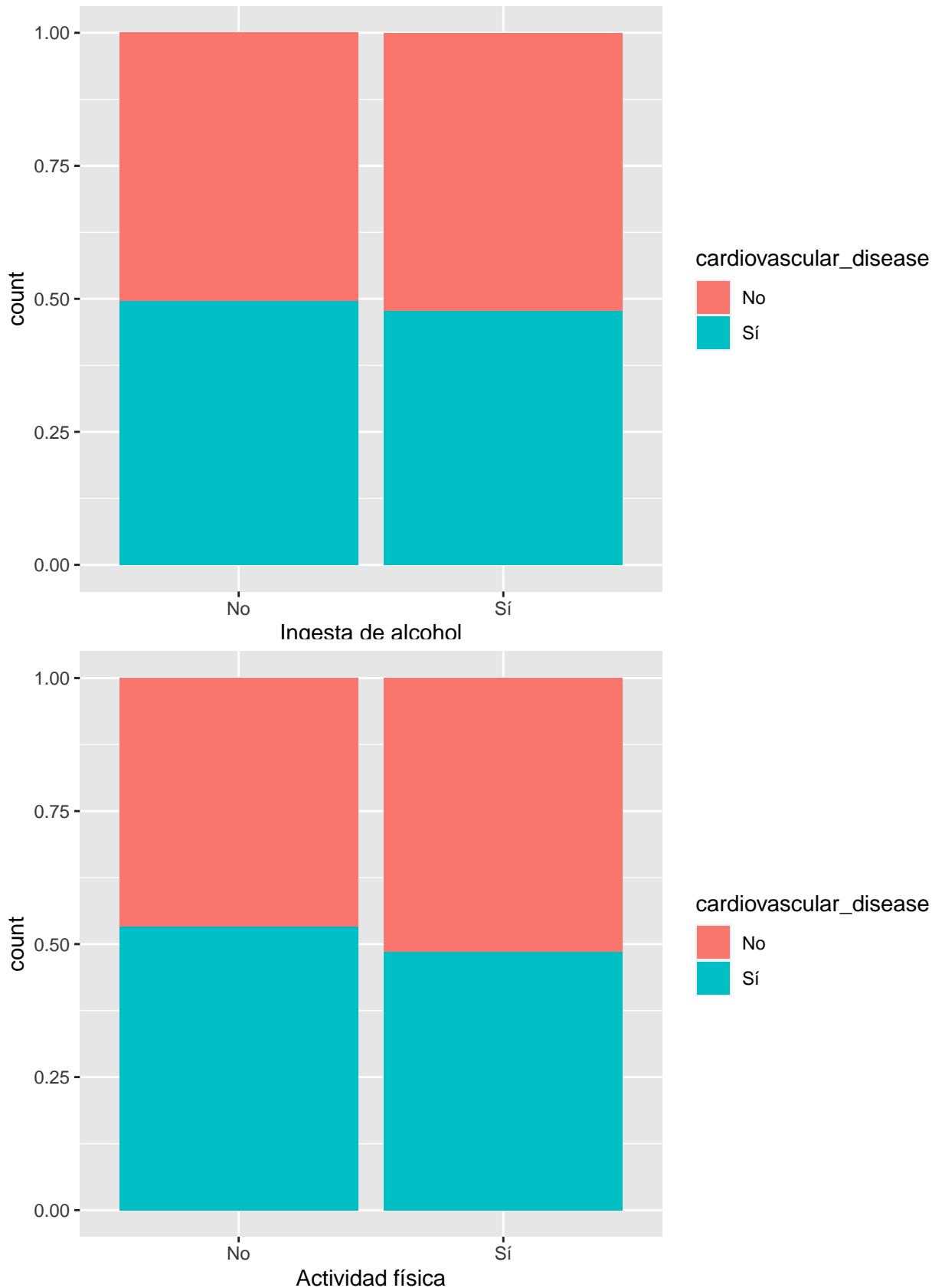












Centrándonos en aquellos grupos que poseen una mayor probabilidad de tener una enfermedad cardiovascular tenemos a las personas con edad Senior, que miden más de 1.96 metros, pesan entre 121 y 160 kilogramos, padecen obesidad, tienen la presión arterial normal-alta (seguida muy de cerca de las personas con hipertensión), son hombres (aunque la probabilidad es prácticamente igual entre hombres y mujeres), tienen los niveles de colesterol y glucosa muy por encima de lo normal, no son fumadores ni ingieren alcohol (siendo muy similar a la probabilidad que tienen los fumadores y los que sí ingieren alcohol) y no realizan actividad física.

Antes de pasar al siguiente apartado, como ya tenemos el dataset preparado para trabajar con él (valores atípicos tratados y variables continuas discretizadas), lo guardaremos con el nombre ***cardio\_clean.csv***.

```
write.csv(cardio_clean,'data/cardio_clean.csv')
```

## 4.2. Comprobación de la normalidad y homogeneidad de la varianza.

Vamos a estudiar la normalidad de las variables de la muestra. La hipótesis en este caso es

$$\begin{aligned} H_0 &: X \sim N(\mu, \sigma^2) \\ H_1 &: X \not\sim N(\mu, \sigma^2) \end{aligned}$$

```
# Establecemos el valor por defecto de alpha
alpha = 0.05
col.names = colnames(cardio_clean)
for (i in 1:ncol(cardio_clean)) {
  if (i == 1) cat("Variables que no siguen una distribución normal:\n")
  if (is.integer(cardio_clean[,i]) | is.numeric(cardio_clean[,i])) {
    # Como nuestro conjunto de datos es grande, utilizaremos la prueba de
    # Kolmogorov-Smirnov
    p_val = lillie.test(cardio_clean[,i])$p.value
    if (p_val < alpha) {
      cat(col.names[i])
      # Establecemos cómo queremos ver la salida que muestra el bucle
      if (i < ncol(cardio_clean) - 1) cat(", ")
      if (i %% 3 == 0) cat("\n")
    }
  }
}

## Variables que no siguen una distribución normal:
## age, height,
## weight, systolic_blood_pressure, diastolic_blood_pressure,
## imc, blood_pressure,
```

Vemos como ninguna variable numérica sigue una distribución normal ya que el p-valor ha sido menor que el valor de aplha y, por lo tanto, no se puede aceptar la hipótesis nula de normalidad.

Debido a que no se cumple la hipótesis de normalidad, para el estudio de la homogeneidad de la varianza utilizaremos el test no paramétrico de [Fligner-Killeen](#), el cual compara las varianzas basándose en la mediana. La hipótesis en este caso es

$$\begin{aligned} H_0 &: \sigma_x^2 = \sigma_y^2 \\ H_1 &: \sigma_x^2 \neq \sigma_y^2 \end{aligned}$$

Lo que haremos será estudiar la homocedasticidad de la varianza de cada variable numérica del dataset en cuanto a los grupos conformados por las personas que sufren una enfermedad cardiovascular frente a las que no la padecen



```

alpha = 0.05
col.names = colnames(cardio_clean)
for (i in 1:ncol(cardio_clean)) {
  if (i == 1) cat("Variables cuya varianza no es homogénea:\n")
  if (is.integer(cardio_clean[,i]) | is.numeric(cardio_clean[,i])) {
    # Como no se cumple la hipótesis de normalidad, usamos el test de
    # Fligner-Killeen
    p_val = fligner.test(cardio_clean[,i] ~ cardiovascular_disease,
                          cardio_clean)$p.value
    if (p_val < alpha) {
      cat(col.names[i])
    }
    # Establecemos cómo queremos ver la salida que muestra el bucle
    if (i < ncol(cardio_clean) - 1) cat(", ")
    if (i %% 3 == 0) cat("\n")
  }
}
## Variables cuya varianza no es homogénea:
## age, height,
## weight, systolic_blood_pressure, diastolic_blood_pressure,
## imc, blood_pressure,

```

Como el p-valor es menor que 0.05 para cada una de las variables, no se puede aceptar la hipótesis nula, por lo que las varianzas de las muestras son heterocedásticas (no homogéneas).

### 4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

Empezaremos planteando la siguiente hipótesis: Nos preguntamos si el Índice de Masa Corporal es igual en hombres y en mujeres. Para esto, realizaremos un contraste de hipótesis de dos muestras sobre la media con varianzas desconocidas.

Planteamos la hipótesis nula y alternativa

$$H_0 : \mu_{hombres} = \mu_{mujeres}$$

$$H_1 : \mu_{hombres} \neq \mu_{mujeres}$$

Debido a que los datos no cumplen las hipótesis de normalidad y homocedasticidad no podremos utilizar el estadístico t de Student, sino que deberemos usar el test no paramétrico equivalente: el test de suma de rangos de Wilcoxon, el cual es equivalente al test U de Mann-Whitney y por eso también se lo conoce por este otro nombre. Pero antes, estudiaremos si estamos ante varianzas desconocidas iguales o diferentes. En este sentido, queremos aplicar el siguiente contraste

$$H_0 : \sigma_1^2 = \sigma_2^2$$

$$H_1 : \sigma_1^2 \neq \sigma_2^2$$

Aplicamos el test

```

varianceTest(cardio_clean$imc[cardio_clean$gender=="Mujer"],
              cardio_clean$imc[cardio_clean$gender=="Hombre"], method = 'fligner')

```

```

##
## Title:
## Fligner-Killeen Test for Homogeneity of Variances

```



```
##  
## Test Results:  
##    STATISTIC:  
##        FK:med chi-squared: 1252.2268  
##    P VALUE:  
##        < 2.2e-16  
##  
## Description:  
## Tue Jun 1 20:47:07 2021
```

Como el p-valor es menor que 0.05 (valor por defecto de *alpha*) no podemos aceptar la hipótesis nula, por lo que descartamos igualdad de varianzas en las dos poblaciones.

En consecuencia, aplicaremos un test bilateral de dos muestras independientes sobre la media con varianza desconocida y diferente

```
wilcox.test(cardio_clean$imc[cardio_clean$gender=="Mujer"],  
            cardio_clean$imc[cardio_clean$gender=="Hombre"],  
            alternative="two.sided", var.equal=FALSE, conf.level = 0.95)
```

```
##  
## Wilcoxon rank sum test with continuity correction  
##  
## data: cardio_clean$imc[cardio_clean$gender == "Mujer"] and cardio_clean$imc[cardio_clean$gender ==  
## W = 591291136, p-value < 2.2e-16  
## alternative hypothesis: true location shift is not equal to 0
```

Observamos que el p-valor es menor que 0.05, por lo que no se puede aceptar la hipótesis nula. Entonces, el Índice de Masa Corporal es diferente para los hombres y las mujeres de la muestra con un 95% de nivel de confianza.

Podemos ahondar un poco más y plantear la hipótesis de si el IMC de los hombres es superior al de las mujeres. El contraste sería el siguiente

$$H_0 : \mu_{hombres} = \mu_{mujeres}$$

$$H_1 : \mu_{hombres} > \mu_{mujeres}$$

En este caso, como ya sabemos que las varianzas son diferentes, el test será unilateral de cola derecha para dos muestras independientes sobre la media con varianza desconocida y diferente

```
wilcox.test(cardio_clean$imc[cardio_clean$gender=="Hombre"],  
            cardio_clean$imc[cardio_clean$gender=="Mujer"],  
            alternative="greater", var.equal=FALSE, conf.level = 0.95)
```

```
##  
## Wilcoxon rank sum test with continuity correction  
##  
## data: cardio_clean$imc[cardio_clean$gender == "Hombre"] and cardio_clean$imc[cardio_clean$gender ==  
## W = 477371408, p-value = 1  
## alternative hypothesis: true location shift is greater than 0
```

Como obtenemos un p-valor de 1, no podemos rechazar la hipótesis nula, por lo que los hombres de la muestra no tienen un IMC mayor que el de las mujeres con un 95% de nivel de confianza.

Para finalizar este apartado de hipótesis, veamos otro contraste muy interesante. Nos preguntamos si el nivel de presión arterial es mayor en las personas con alguna enfermedad cardiovascular que en las personas sin este tipo de enfermedad. El contraste en este caso es

$$H_0 : \mu_{cardiovascular\ disease} = \mu_{no\ cardiovascular\ disease}$$

$$H_1 : \mu_{cardiovascular\ disease} > \mu_{no\ cardiovascular\ disease}$$



Empezaremos estudiando si estamos ante varianzas iguales o diferentes

```
varianceTest(cardio_clean$blood_pressure[
    cardio_clean$cardiovascular_disease=="Sí"],
    cardio_clean$blood_pressure[
        cardio_clean$cardiovascular_disease=="No"],
    method = 'fligner')

## 
## Title:
## Fligner-Killeen Test for Homogeneity of Variances
##
## Test Results:
##   STATISTIC:
##     FK:med chi-squared: 3631.839
##   P VALUE:
##     < 2.2e-16
##
## Description:
## Tue Jun 1 20:47:07 2021
```

Como el p-valor es prácticamente 0 no podemos aceptar la hipótesis nula de igualdad de varianzas, por lo que aplicaremos un test de cola derecha para dos muestras independientes sobre la media con varianza desconocida y diferente

```
wilcox.test(
    cardio_clean$blood_pressure[cardio_clean$cardiovascular_disease=="Sí"],
    cardio_clean$blood_pressure[cardio_clean$cardiovascular_disease=="No"],
    alternative="greater", var.equal=FALSE, conf.level = 0.95)

## 
## Wilcoxon rank sum test with continuity correction
##
## data: cardio_clean$blood_pressure[cardio_clean$cardiovascular_disease == "Sí"] and cardio_clean$blood_
## W = 875462463, p-value < 2.2e-16
## alternative hypothesis: true location shift is greater than 0
```

Obtenemos un p-valor menor que 0.05, por lo que, con un 95% de confianza, no podemos aceptar la hipótesis nula, es decir, la presión arterial media es mayor en personas con enfermedades cardiovasculares que en personas sanas, resultado que era de esperarse.

#### 4.3.0 Cross Validation

Para esta metodología, los datos de entrenamiento se dividen en datos de entrenamiento (70%) y datos de validación (30%)

```
set.seed(42)
partition <- createDataPartition(y = cardio_clean$cardiovascular_disease,
                                 p = 0.7, list = F)
trainingdata <- cardio_clean[partition, ]
test <- cardio_clean[-partition, ]
```

El conjunto de entrenamiento contiene una salida conocida y el modelo aprende estos datos para poder generalizarlos a otros datos en el proceso. De este modo, el modelo predecirá valores para el 30% de los datos (validación). El cálculo del RMSE permite determinar la precisión de la predicción del modelo

```
set.seed(42)
partitiontraining <- createDataPartition(
```



4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS

```
y = trainingdata$cardiovascular_disease, p = 0.8, list = F)
training <- trainingdata[partitiontraining, ]
validation <- trainingdata[-partitiontraining, ]
```

Tras dividir el modelo en datos de entrenamiento y validación, es necesario evaluar las variables con mayor poder predictivo. El mejor método es revisar los valores p - en el modelo de regresión, sin embargo hay muchas variables en el modelo por lo que tomará un tiempo considerable hacerlo manualmente. En este caso, se implementa el algoritmo de regularización del lazo para identificar las variables que serán significativas en el modelo.

```
x <- model.matrix(~.,trainingdata[, -12])
y <- as.numeric(trainingdata$cardiovascular_disease)
cv.out <- cv.glmnet(x,y,alpha=1,type.measure = "mse")
lambda_min <- cv.out$lambda.min
lambda_1se <- cv.out$lambda.1se
coef <- as.matrix(coef(cv.out,s=lambda_1se))
coef2 <- as.data.frame(as.table(coef))
coef2 <- coef2 %>%
  dplyr::select(-Var2)%>%
  dplyr::filter(Freq != 0)%>%
  rename(Variable = Var1, Coeficients = Freq)
```

coef2

	Variable	Coeficients
## 1	(Intercept)	-0.6450322553
## 2	age	0.0097391522
## 3	weight	0.0009678785
## 4	systolic_blood_pressure	0.0075485086
## 5	cholesterolPor_encima_de_lo_normal	0.0640600356
## 6	cholesterolMuy_por_encima_de_lo_normal	0.1757940726
## 7	glucoseMuy_por_encima_de_lo_normal	-0.0253723170
## 8	smokingSí	-0.0145248000
## 9	alcohol_intakeSí	-0.0299172781
## 10	physical_activitySí	-0.0398141726
## 11	imc	0.0019538545
## 12	blood_pressure	0.0058465421
## 13	group_weight(81, 120)	0.0075333078
## 14	group_weight(161, Inf)	-0.0306541438
## 15	group_imcPeso_normal	-0.0153192233
## 16	hypertensionPresión_arterial_normal	-0.0054606125
## 17	hypertensionPresión_arterial_normal_alta	-0.0395177814
## 18	hypertensionHipertensión	-0.1502275237

El resultado muestra que sólo las variables que se han determinado como significativas sobre la base de los valores p tienen coeficientes distintos de cero. Los coeficientes de todas las demás variables han sido puestos a cero por el algoritmo.

Con las variables significativas del modelo, se aplica el algoritmo de regresión lineal y se utiliza este modelo para predecir el precio de venta en el conjunto de datos de validación.

```
test$cardiovascular_disease <- as.numeric(test$cardiovascular_disease)
validation$cardiovascular_disease <- as.numeric(
  validation$cardiovascular_disease)
training$cardiovascular_disease <- as.numeric(training$cardiovascular_disease)
model <- lm(cardiovascular_disease ~ cholesterol + age +
```



4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS

```
systolic_blood_pressure + group_weight + blood_pressure +  
imc + weight + smoking + alcohol_intake, data=training)
```

Se calcula el RMSE, que evalúa la precisión del modelo. A continuación, se aplica el modelo a los datos de prueba para obtener el RMSE.

```
p <- predict(model, validation)  
error <- (p - validation$cardiovascular_disease)  
RMSE_Model <- sqrt(mean(error^2))  
  
ptest <- predict(model, test)  
error1 <- (ptest - test$cardiovascular_disease)  
RMSE_NewData <- sqrt(mean(error1^2))
```

```
#library(kableExtra)  
Method <- c("Train/Test Split")  
ModelRMSE <- c(RMSE_Model)  
RMSENewData <- c(RMSE_NewData)  
  
table1 <- data.frame(Method, ModelRMSE, RMSENewData)  
  
table1
```

```
##           Method ModelRMSE RMSENewData  
## 1 Train/Test Split 0.4403899 0.4398323  
# kable(table1) %>% kable_styling(c("striped", "bordered")) %>% column_spec(2:3, border_left = T)
```

Estos RMSE se compararán con el resultado obtenido mediante la metodología de validación cruzada.

**K-fold Cross Validation** En la validación cruzada K-fold, el conjunto de datos se divide en k partes separadas. El proceso de entrenamiento se repite k veces. Cada vez, una parte se utiliza como datos de validación y el resto se utiliza para entrenar un modelo. A continuación, se calcula la media del error para evaluar un modelo. Hay que tener en cuenta que la validación cruzada k-fold aumenta los requisitos computacionales para el entrenamiento de nuestro modelo en un factor de k.

En este enfoque, todo el conjunto de entrenamiento se utiliza para predecir el precio de venta en el conjunto de prueba. El número de k seleccionado para el enfoque de validación cruzada es 10.

```
trainingdata$cardiovascular_disease <- as.numeric(  
    trainingdata$cardiovascular_disease)  
#classProbs = TRUE  
modelcv <- train(cardiovascular_disease ~ cholesterol + age +  
                    systolic_blood_pressure + group_weight +  
                    blood_pressure + imc + weight + smoking +  
                    alcohol_intake, data=trainingdata, method = "lm",  
                    trControl = trainControl(method = "cv", number = 10 ))
```

```
RMSE_Modelcv <- modelcv$results$RMSE  
  
pcv <- predict(modelcv, test)  
#test$cardiovascular_disease <- as.factor(test$cardiovascular_disease)  
errorcv <- (pcv - test$cardiovascular_disease)  
RMSE_NewDatacv <- sqrt(mean(errorcv^2))
```

```
Method <- c("CrossValidation")  
ModelRMSE <- c(RMSE_Modelcv)
```



```
RMSENewData <- c(RMSE_NewDatacv)
table2 <- data.frame(Method, ModelRMSE, RMSENewData)

table2

##           Method ModelRMSE RMSENewData
## 1 CrossValidation 0.4397274   0.4397853
#kable(table2) %>% kable_styling(c("striped", "bordered")) %>% column_spec(2:3, border_left = T)
```

A continuación se muestran los resultados de los dos enfoques en términos de RMSE.

```
table12 <- rbind(table1,table2)
table12

##           Method ModelRMSE RMSENewData
## 1 Train/Test Split 0.4403899   0.4398323
## 2 CrossValidation 0.4397274   0.4397853
#kable(table12) %>% kable_styling(c("striped", "bordered")) %>% column_spec(2:3, border_left = T)
```

En conclusión, la estrategia de validación cruzada tiene un RMSE menor en los nuevos datos en comparación con el RMSE medio del modelo. Esto indica que el modelo tiene un menor valor predictivo cuando se prueba con los nuevos datos en comparación con el enfoque de división entrenamiento/prueba, en el que el RMSE del conjunto de validación es mucho mayor que el del conjunto de prueba (nuevos datos).

### 4.3.1 Regresión Lineal

Ahora, vamos a ver con resúmenes tabulares y gráficos que comparan los índices de rendimiento de los modelos. Para ello realizaremos una serie de regresiones lineales en las cuales intentaremos explicar la variable referente a padecer una enfermedad cardiovascular en base a una serie de variables. Empezaremos por una regresión lineal simple e iremos añadiendo en las siguientes una variable más

```
# Elegimos 2/3 para el conjunto de entrenamiento
smp_size <- floor(2/3 * nrow(cardio_clean))

# Establecemos la semilla para que el ejemplo sea reproducible
set.seed(222)
train_ind <- sample(seq_len(nrow(cardio_clean)), size = smp_size)

# Establecemos los conjuntos de entrenamiento y prueba
cardio_train <- cardio_clean[train_ind, ]
cardio_test <- cardio_clean[-train_ind, ]

# Convertimos la variable a tipo numérica debido a que es de tipo factor
cardio_train$cardio_num <- as.numeric(cardio_train$cardiovascular_disease)

# Realizamos las regresiones lineales
lm1 <- lm(cardio_num~blood_pressure, data=cardio_train)
lm2 <- lm(cardio_num~blood_pressure+imc, data=cardio_train)
lm3 <- lm(cardio_num~blood_pressure+imc+systolic_blood_pressure,
           data=cardio_train)
lm4 <- lm(cardio_num~blood_pressure+imc+systolic_blood_pressure+
           diastolic_blood_pressure, data=cardio_train)
lm5 <- lm(cardio_num~blood_pressure+imc+systolic_blood_pressure+
           diastolic_blood_pressure+glucose, data=cardio_train)
```



4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS

```
# Vemos las estadísticas de la última regresión hecha, que es la que más
# variables posee
summary(lm5)

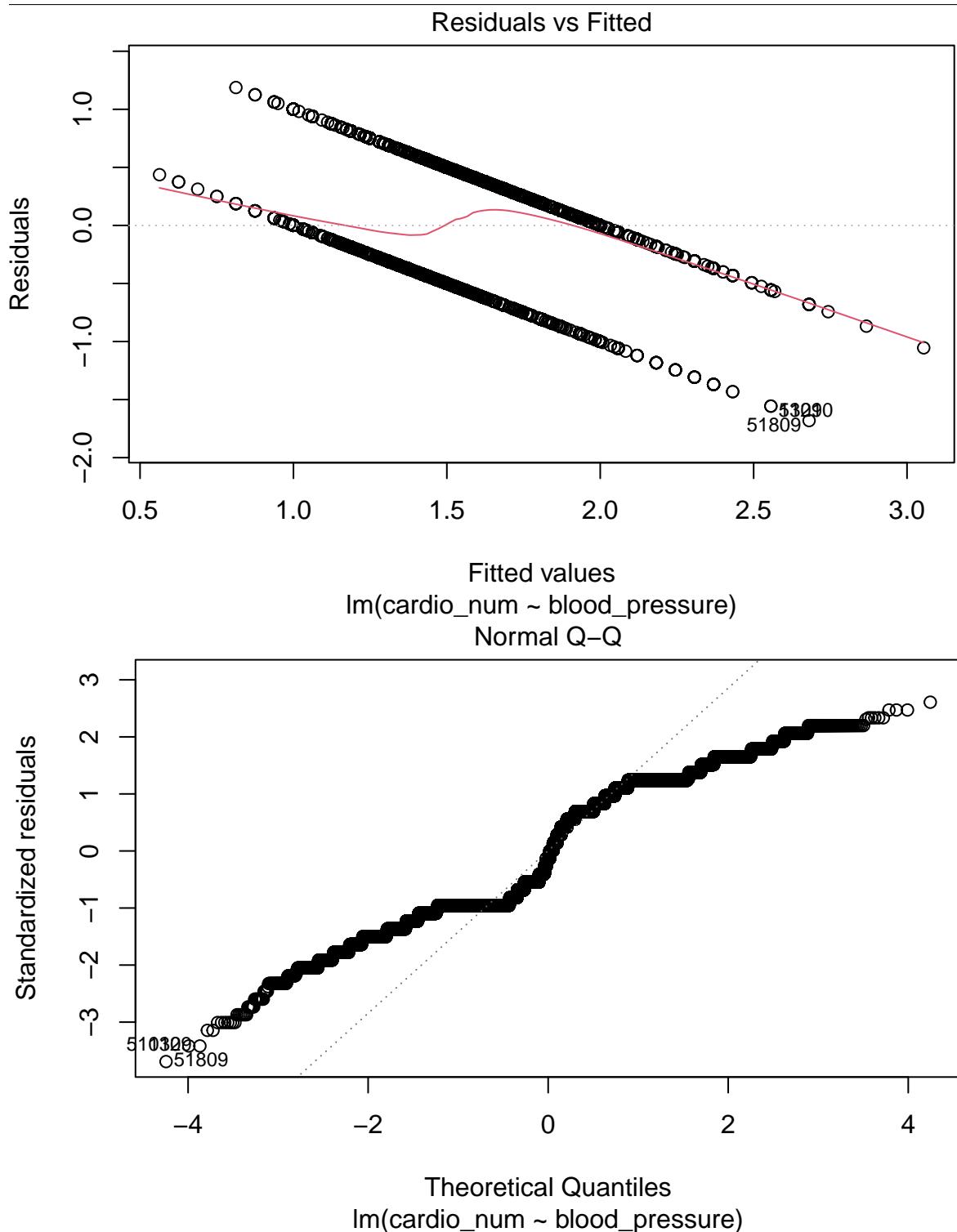
##
## Call:
## lm(formula = cardio_num ~ blood_pressure + imc + systolic_blood_pressure +
##     diastolic_blood_pressure + glucose, data = cardio_train)
##
## Residuals:
##      Min        1Q    Median        3Q       Max
## -1.79590 -0.39121 -0.09995  0.41598  1.15966
##
## Coefficients: (1 not defined because of singularities)
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -0.3489270  0.0196916 -17.720 < 2e-16 ***
## blood_pressure                0.0054003  0.0004627  11.671 < 2e-16 ***
## imc                          0.0070994  0.0004238  16.750 < 2e-16 ***
## systolic_blood_pressure      0.0088336  0.0003052  28.948 < 2e-16 ***
## diastolic_blood_pressure          NA         NA        NA       NA
## glucosePor_encima_de_lo_normal   0.0345973  0.0081500   4.245 2.19e-05 ***
## glucoseMuy_por_encima_de_lo_normal 0.0775780  0.0080058   9.690 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.449 on 45727 degrees of freedom
## Multiple R-squared:  0.1937, Adjusted R-squared:  0.1936
## F-statistic: 2197 on 5 and 45727 DF,  p-value: < 2.2e-16
```

La regresión tiene una bondad del ajuste ( $R^2$  cuadrado) del 19.37%, lo que significa que las variables exógenas explican en ese porcentaje a la variable endógena (cardio\_num). Este resultado nos indica que el modelo es capaz de explicar solo el 19.37% de la variabilidad observada en si tiene enfermedad o no. El valor de  $R^2$ -ajustado es muy cercano a  $R^2$  (Adjusted R-squared: 0.1936) lo que indica que el modelo no tiene predictores útiles. El test F muestra un p-value de  $2.2 \times 10^{-16}$  por lo que el modelo en conjunto es significativo. Esto se corrobora con el p-value de cada predictor, pues en ambos casos es significativo.

Representamos gráficamente los residuos de la primera y última regresión lineal

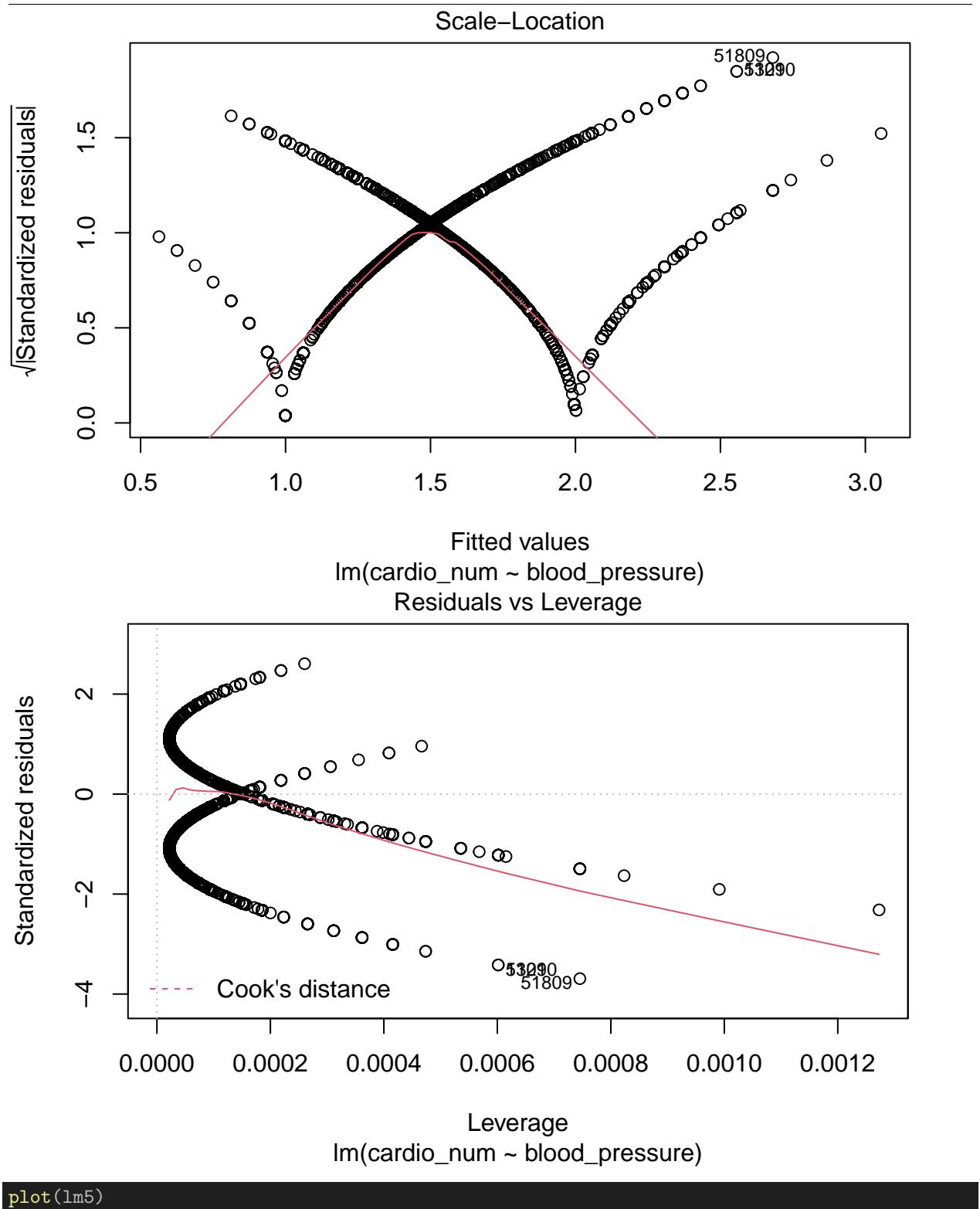
```
plot(lm1)
```

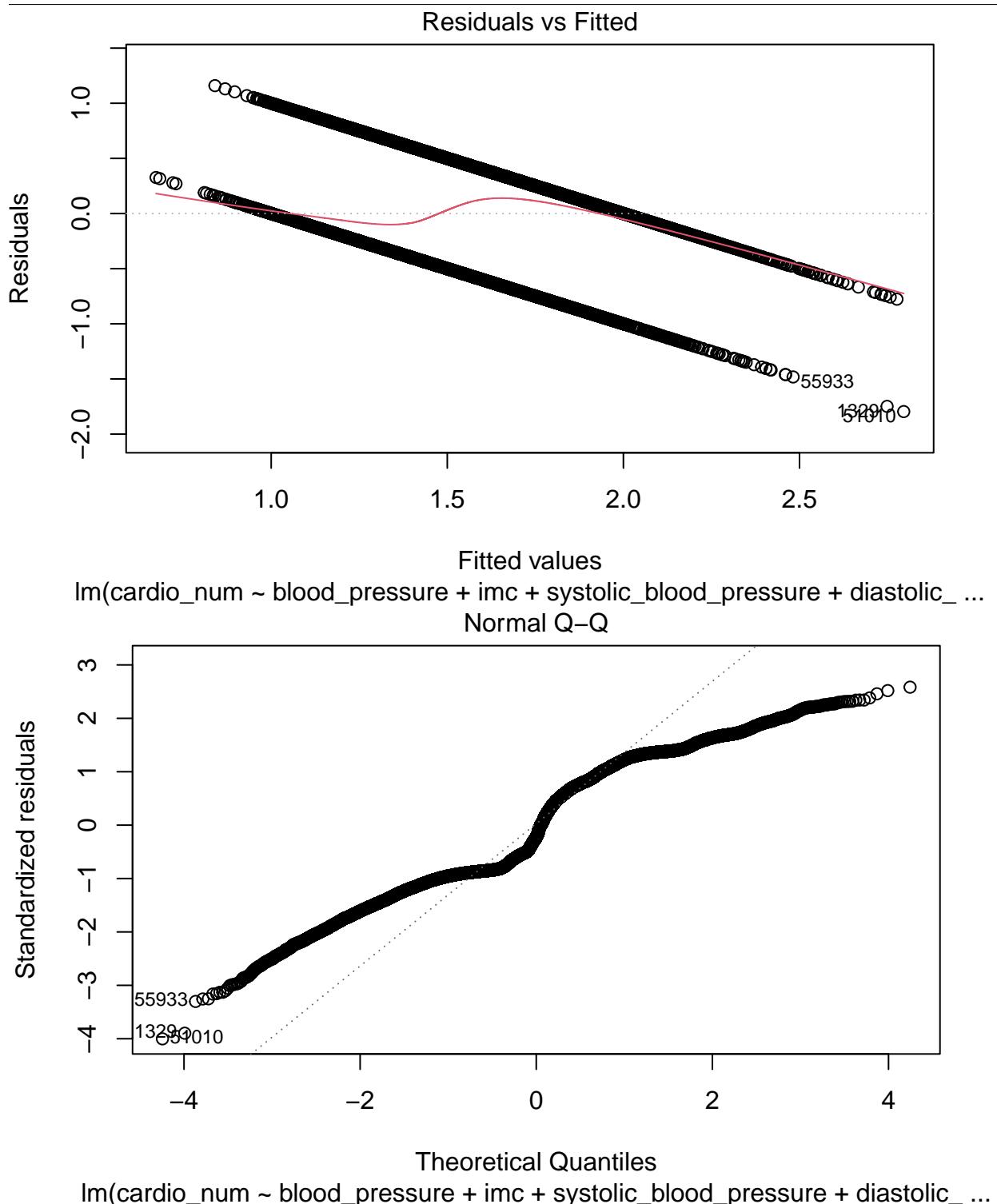




4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

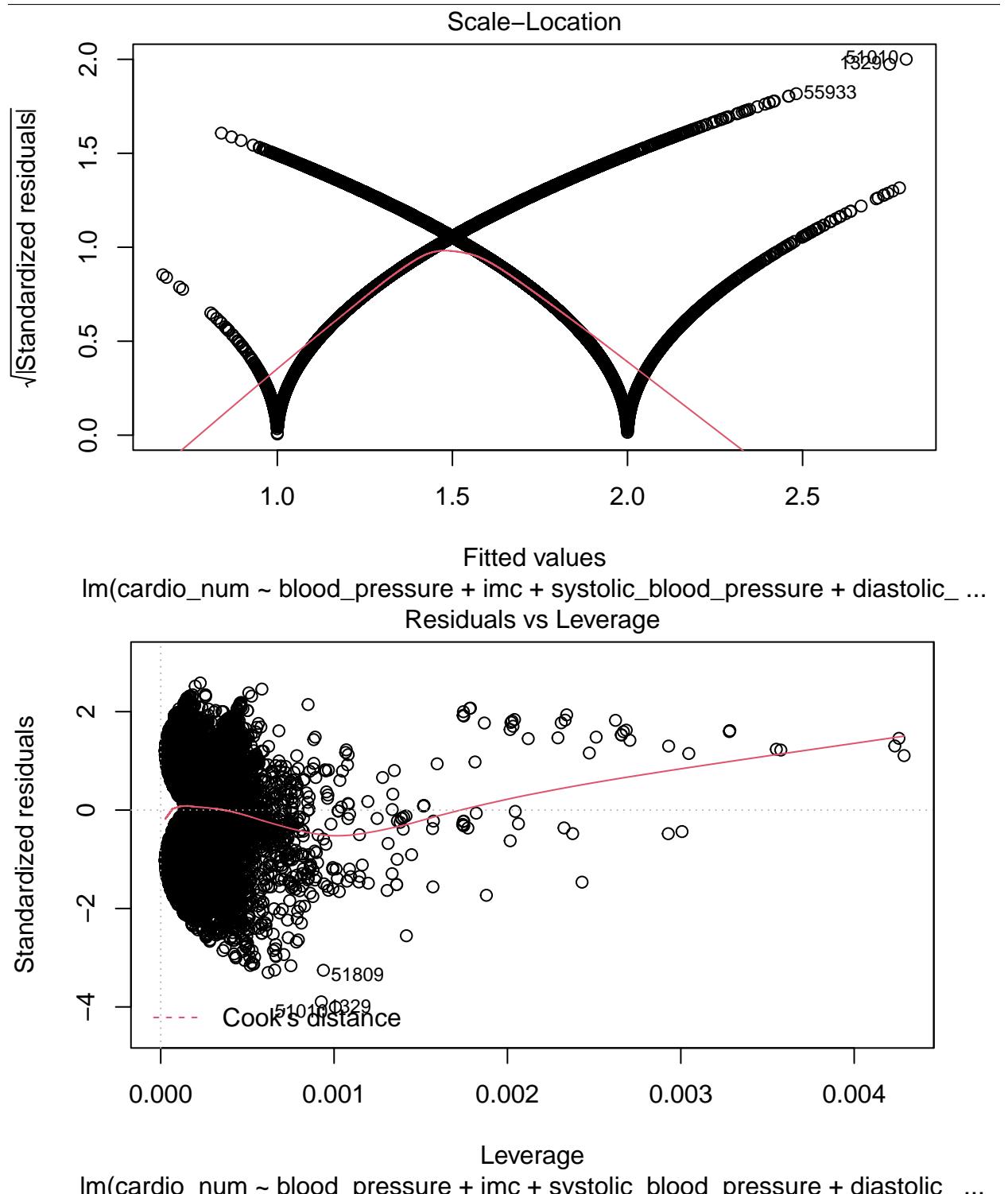
#### 4. ANÁLISIS DE DATOS





4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS



Vemos en el modelo lm1 y lm5 que los modelos no están distribuidos normalmente, puesto que los puntos se desvian mucho de la linea diagonal.

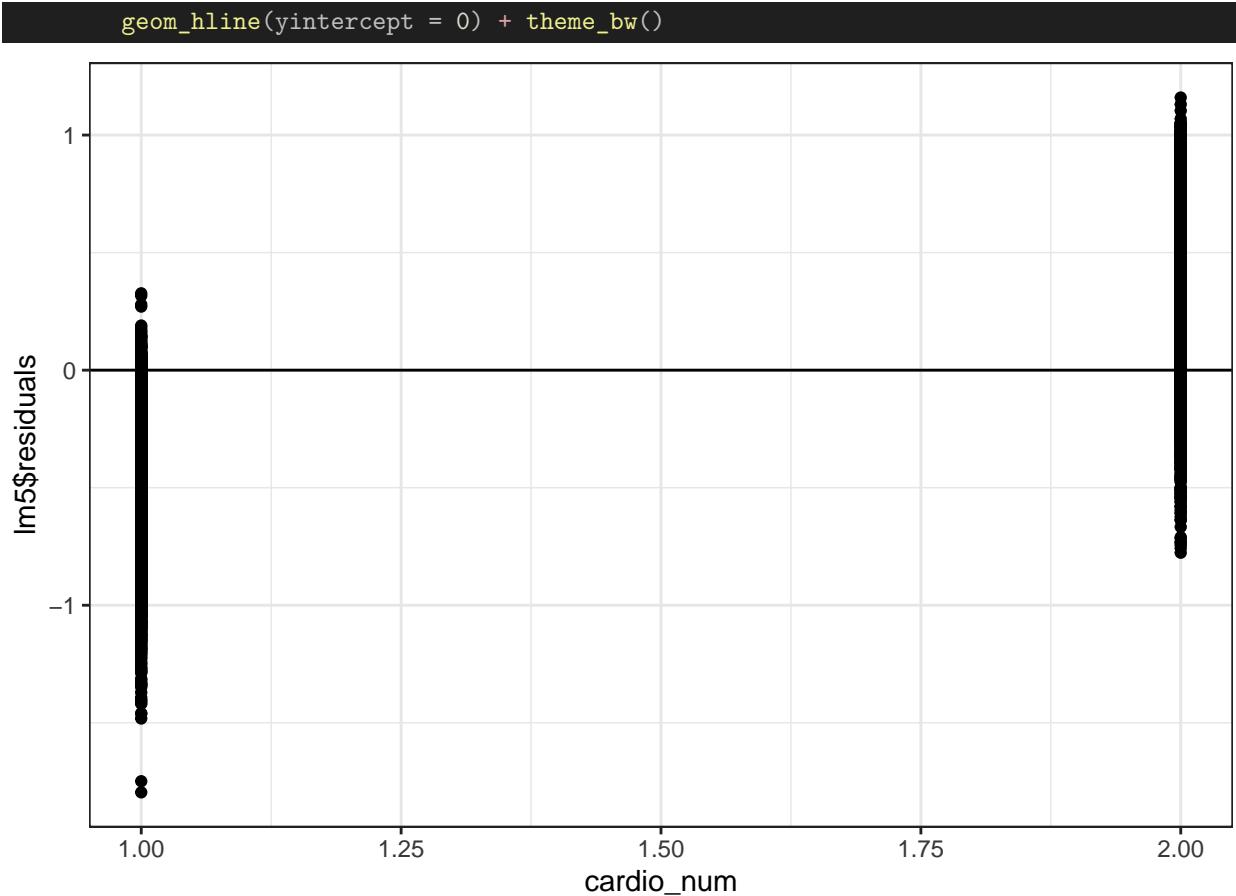
Relación lineal entre los predictores numéricos y la variable dependiente:

```
ggplot(data = cardio_train, aes(x = cardio_num, y = lm5$residuals)) +
  geom_point() + geom_smooth(color = "firebrick") +
```



4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS



No se satisface la condición de linealidad y se aprecia un posible dato atípico.

```
shapiro.test(lm5$residuals[0:5000])
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: lm5$residuals[0:5000]  
## W = 0.92418, p-value < 2.2e-16
```

Utilizando el test de Shapiro obtenemos un p-value menor a 0.05, por lo que no podemos aceptar la hipótesis nula de normalidad en la distribución de los residuos, como es en este caso.

Comparamos ahora las diferentes regresiones lineales obtenidas anteriormente

```
comp <- compare_performance(lm1, lm2, lm3, lm4, lm5)  
comp
```

```
## # Comparison of Model Performance Indices  
##  
## Name | Model | AIC | BIC | R2 | R2 (adj.) | RMSE | Sigma  
## -----  
## lm1 | lm | 57846.325 | 57872.517 | 0.170 | 0.170 | 0.455 | 0.455  
## lm2 | lm | 57483.644 | 57518.567 | 0.177 | 0.177 | 0.454 | 0.454  
## lm3 | lm | 56646.889 | 56690.542 | 0.192 | 0.192 | 0.449 | 0.449  
## lm4 | lm | 56646.889 | 56690.542 | 0.192 | 0.192 | 0.449 | 0.449  
## lm5 | lm | 56545.752 | 56606.866 | 0.194 | 0.194 | 0.449 | 0.449
```



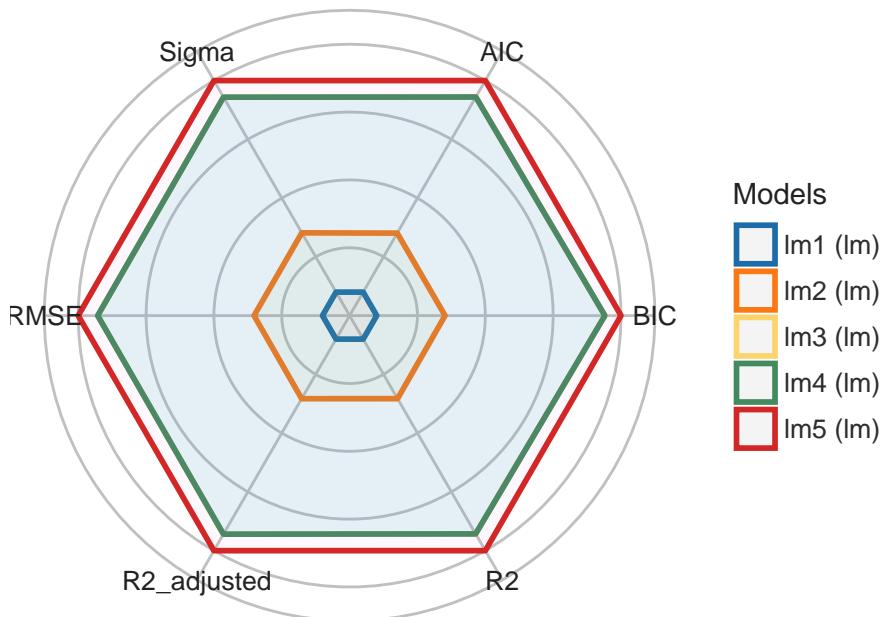
4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS

Podemos observar que las diferencias no son muy significativas. Vemos el gráfico

```
plot(comp)
```

Comparison of Model Indices



Teniendo en cuenta todos estos datos vemos que ninguno de los modelos lineales son significativos, por lo que podemos rechazar este modelo de predicción.

#### 4.3.2 Regresión Logistica

Realizaremos a continuación un algoritmo de clasificación, la regresión logística, cuya variable dependiente o endógena será “cardiovascular\_disease”. Elegiremos una serie de variables independientes o exógenas y no incluiremos aquellas que puedan presentar problemas de multicolinealidad

```
# Clasificamos los conjuntos de entrenamiento y prueba mediante el método
# de validación cruzada
set.seed(123)
h <- holdout(cardio_clean$cardiovascular_disease, ratio=2/3, mode="stratified")
data_train <- cardio_clean[h$tr,]
data_test <- cardio_clean[h$ts,]
train_control <- trainControl(method="cv", number=4)

# Realizamos la regresión
logit_model <- glm(formula=cardiovascular_disease~age+gender+smoking+
                     physical_activity+group_imc+glucose+alcohol_intake+
                     hypertension+cholesterol, data=data_train,
                     family=binomial)

# Estadísticas del modelo
summary(logit_model)

##
```



4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS

```

## physical_activity + group_imc + glucose + alcohol_intake +
## hypertension + cholesterol, family = binomial, data = data_train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.6494 -1.0283 -0.4934  1.0669  2.2931
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -4.537874  0.149473 -30.359 < 2e-16
## age                         0.061196  0.001571  38.949 < 2e-16
## genderMujer                 -0.102980  0.022877 -4.501 6.75e-06
## smokingSí                   -0.131317  0.040144 -3.271 0.00107
## physical_activitySí        -0.195837  0.025486 -7.684 1.54e-14
## group_imcPeso_normal        0.344127  0.122436  2.811 0.00494
## group_imcSobrepeso          0.616233  0.122385  5.035 4.77e-07
## group_imcObesidad           0.923747  0.123063  7.506 6.08e-14
## glucosePor_encima_de_lo_normal 0.061967  0.041702  1.486 0.13729
## glucoseMuy_por_encima_de_lo_normal -0.331375  0.046163 -7.178 7.05e-13
## alcohol_intakeSí            -0.137844  0.048769 -2.826 0.00471
## hypertensionPresión_arterial_normal 0.814274  0.027464 29.649 < 2e-16
## hypertensionPresión_arterial_normal_alta 2.191626  0.057229 38.296 < 2e-16
## hypertensionHipertensión        2.215421  0.131286 16.875 < 2e-16
## cholesterolPor_encima_de_lo_normal 0.502213  0.031156 16.119 < 2e-16
## cholesterolMuy_por_encima_de_lo_normal 1.242392  0.041756 29.754 < 2e-16
##
## (Intercept) ***
## age ***
## genderMujer ***
## smokingSí **
## physical_activitySí ***
## group_imcPeso_normal **
## group_imcSobrepeso ***
## group_imcObesidad ***
## glucosePor_encima_de_lo_normal ***
## glucoseMuy_por_encima_de_lo_normal ***
## alcohol_intakeSí **
## hypertensionPresión_arterial_normal ***
## hypertensionPresión_arterial_normal_alta ***
## hypertensionHipertensión ***
## cholesterolPor_encima_de_lo_normal ***
## cholesterolMuy_por_encima_de_lo_normal ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 63395  on 45732  degrees of freedom
## Residual deviance: 55781  on 45717  degrees of freedom
## AIC: 55813
##
## Number of Fisher Scoring iterations: 4

```

Si nos fijamos en la significancia de las variables, solo hay una categoría no significativa (glucose: Por encima



4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

#### 4. ANÁLISIS DE DATOS

de lo normal). En cuanto a las demás, todas son significativas al 0.1% excepto tres categoría que son significativas al 1% (smoking: Sí, group\_imc: Peso\_normal, alcohol\_intake: Sí).

Para ver el incremento o decremento en términos de probabilidad de cada variable exógena sobre la variable endógena debemos estudiar los Odds Ratio (OR)

```
exp(coefficients(logit_model))
```

```
##                               (Intercept)
##                               0.01069613
##                               age
##                               1.06310739
##                               genderMujer
##                               0.90214546
##                               smokingSí
##                               0.87693965
##                               physical_activitySí
##                               0.82214645
##                               group_imcPeso_normal
##                               1.41075732
##                               group_imcSobrepeso
##                               1.85193887
##                               group_imcObesidad
##                               2.51870946
##                               glucosePor_encima_de_lo_normal
##                               1.06392742
##                               glucoseMuy_por_encima_de_lo_normal
##                               0.71793561
##                               alcohol_intakeSí
##                               0.87123418
##                               hypertensionPresión_arterial_normal
##                               2.25753583
## hypertensionPresión_arterial_normal_alta
##                               8.94975205
##                               hypertensionHipertensión
##                               9.16526303
##                               cholesterolPor_encima_de_lo_normal
##                               1.65237407
##                               cholesterolMuy_por_encima_de_lo_normal
##                               3.46389056
```

Estudiando los OR que tienen un mayor impacto, vemos cómo las personas con hipertensión tienen una probabilidad de padecer una enfermedad cardiovascular 9.17 veces mayor en comparación con aquellas que tienen la tensión baja (categoría de referencia) y aquellas que tienen la presión arterial normal-alta tienen dicha probabilidad 8.95 veces mayor; las personas que tienen el colesterol muy por encima de lo normal tienen una probabilidad 3.46 veces mayor de padecer este tipo de enfermedades en comparación con aquellas que tienen unos niveles de colesterol normales (categoría de referencia); y las personas obesas o con sobrepeso poseen 2.52 y 1.85 veces, respectivamente, más probabilidades de tener una enfermedad cardiovascular en comparación con aquellas que tienen un peso inferior al normal (categoría de referencia). Por lo tanto, podemos concluir que en nuestro modelo las tres variables que más influencia tienen a la hora de padecer una enfermedad cardiovascular son la hipertensión, el colesterol y el Índice de Masa Corporal, lo cual, de primeras, se asemeja bastante a la realidad.

Una vez hecha la regresión y estudiado los OR, calculamos la matriz de confusión para ver la precisión del modelo



```

mod <- train(cardiovascular_disease~age+gender+smoking+
              physical_activity+group_imc+glucose+alcohol_intake+
              hypertension+cholesterol, data=data_train, method="glm",
              trControl = train_control)

pred <- predict(mod, newdata=data_test)

confusionMatrix(pred,data_test$cardiovascular_disease,positive="Sí")

## Confusion Matrix and Statistics
##
##             Reference
## Prediction   No    Sí
##       No 8082 4141
##       Sí 3468 7176
##
##               Accuracy : 0.6672
##                   95% CI : (0.6611, 0.6734)
##       No Information Rate : 0.5051
##       P-Value [Acc > NIR] : < 2.2e-16
##
##               Kappa : 0.334
##
## McNemar's Test P-Value : 1.321e-14
##
##               Sensitivity : 0.6341
##               Specificity : 0.6997
##       Pos Pred Value : 0.6742
##       Neg Pred Value : 0.6612
##               Prevalence : 0.4949
##       Detection Rate : 0.3138
## Detection Prevalence : 0.4655
##       Balanced Accuracy : 0.6669
##
##       'Positive' Class : Sí
##

```

El modelo ha predicho 3468 falsos positivos y 4141 falsos negativos. Su precisión es del 66.72%, siendo la sensibilidad (predicción de verdaderos positivos) del 63.41% y la especificidad (predicción de verdaderos negativos) del 69.97%. Por lo tanto, podemos decir que la calidad es normal, ya que no es ni muy alta ni muy baja.

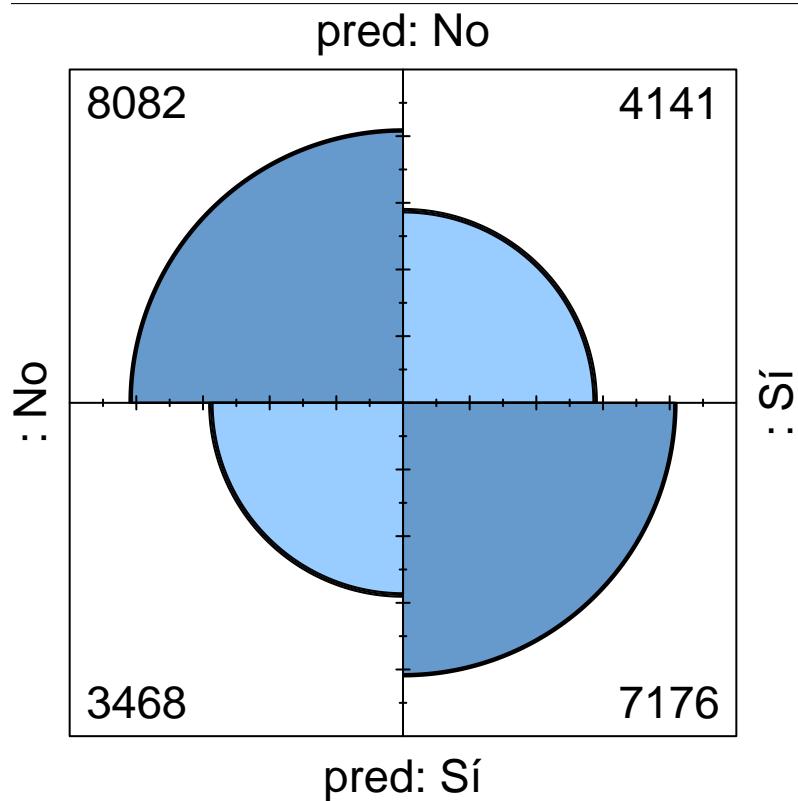
Representemos de forma gráfica la matriz de confusión

```

cfmtx <- confusionMatrix(table(pred,
                                 data_test[["cardiovascular_disease"]]),
                           positive = "Sí")
fourfoldplot(cfmtx$table)

```

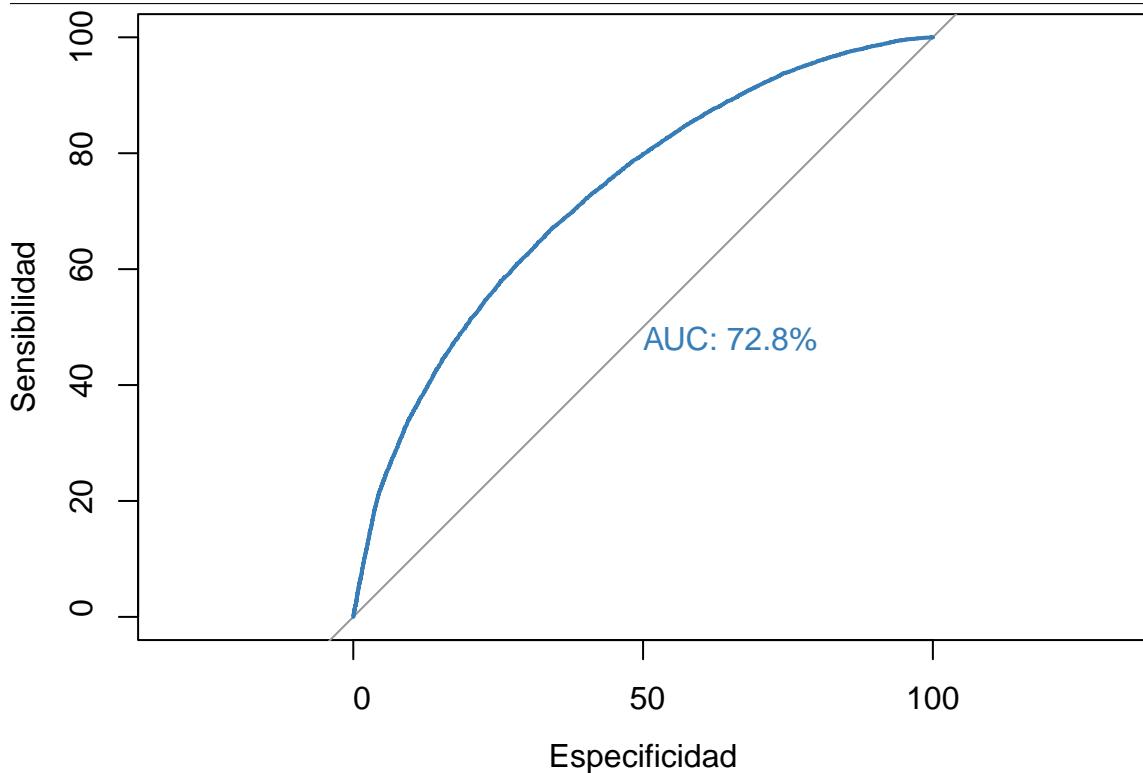




Seguidamente, graficamos la curva de ROC para tener una representación gráfica de la sensibilidad frente a la especificidad

```
roc(data_train$cardiovascular_disease ~ logit_model$fitted.values,
  plot = TRUE, legacy.axes = TRUE, percent = TRUE,
  xlab = "Especificidad", ylab = "Sensibilidad", col = "#377eb8",
  lwd = 2, print.auc = TRUE)
```





```
##
```

```
## Call:
```

```
## roc.formula(formula = data_train$cardiovascular_disease ~ logit_model$fitted.values,      plot = TRUE)
## 
## Data: logit_model$fitted.values in 23100 controls (data_train$cardiovascular_disease No) < 22633 cases (Yes)
## Area under the curve: 72.75%
```

Podemos apreciar cómo obtenemos un AUC del 72.8%, lo que nos indica que este modelo tiene una probabilidad del 72.8% de clasificar a los pacientes enfermos como enfermos (sensibilidad) y a los exentos de enfermedad como sanos (especificidad).

#### 4.3.1 Árbol de decisión

Finalizaremos este apartado realizando otro método de clasificación: el algoritmo de aprendizaje supervisado de árbol de decisión. Así, podremos ver también como clasifica el modelo a las personas del dataset y veremos la precisión que tiene, comparándola con el obtenida mediante el método de regresión logística. Utilizaremos la técnica conocida como CART: Classification And Regression Trees. La implementación particular de CART que usaremos es la Recursive Partitioning and Regression Trees o RPART. De forma general, lo que hace este algoritmo es encontrar la variable independiente que mejor separa nuestros datos en grupos, que se corresponden con las categorías de la variable objetivo. Esta mejor separación es expresada con una regla y a cada regla le corresponde un nodo.

Comenzamos cambiando los tipos de todas las variables que tenemos a factor, pues la función que utilizaremos, rpart(), ejecutará un árbol de regresión si la variable de respuesta es numérica, y un árbol de clasificación si es un factor.

```
# Creamos un nuevo dataset para no sobrescribir el que ya tenemos
cardio_clean_factor <- cardio_clean

# Vemos las variables numéricas que tenemos
nums <- unlist(lapply(cardio_clean_factor, is.numeric))
```



```
# Las convertimos en tipo factor
cardio_clean_factor[,nums] <- lapply(cardio_clean_factor[,nums], factor)

# Nos quedamos con las variables que utilizaremos, que serán solamente las
# discretas. Además, excluiremos todas aquellas que puedan generar problemas
# de multicolinealidad
cardio_clean_factor <- subset(cardio_clean_factor,select=-c(
    age,height,weight,systolic_blood_pressure,
    diastolic_blood_pressure,imc,blood_pressure,group_height,
    group_weight))
```

Ahora que tenemos el dataset preparado crearemos los sets de entrenamiento y prueba, pero antes desordenaremos las observaciones, lo que nos interesa debido a que las tenemos ordenadas

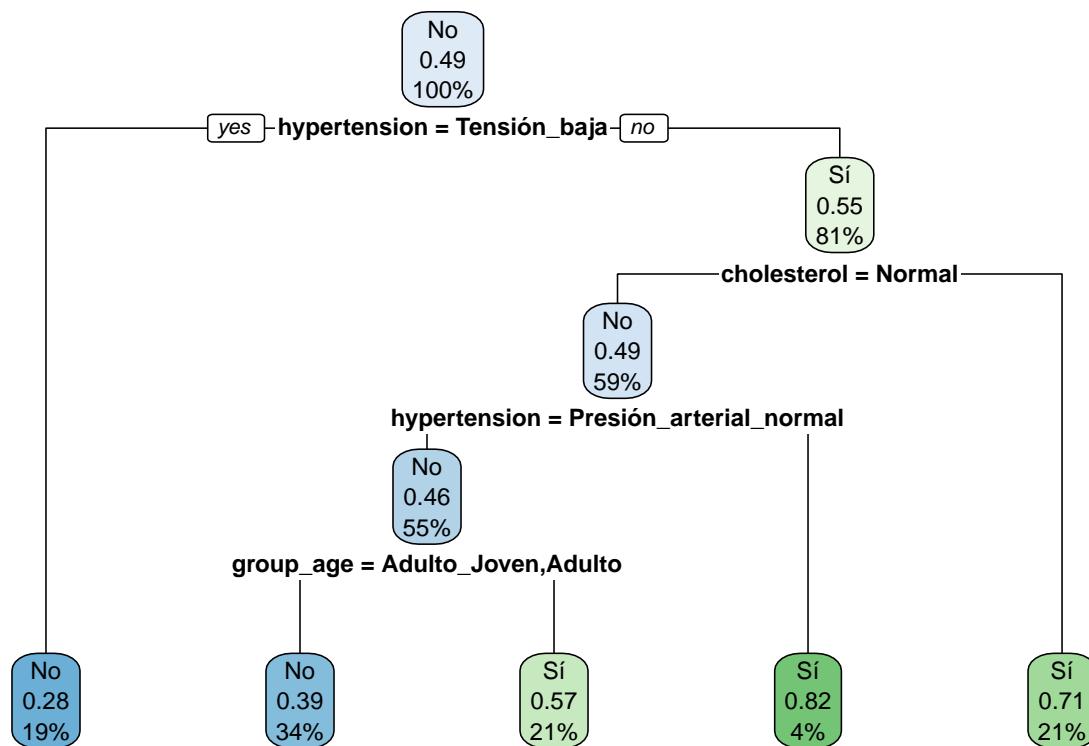
```
set.seed(666)
cardio_clean_factor <- cardio_clean_factor[sample(nrow(cardio_clean_factor)),]
```

```
# Creamos los conjuntos mediante el método de validación cruzada
set.seed(789)
h <- holdout(cardio_clean_factor$cardiovascular_disease, ratio=2/3,
              mode="stratified")
data_train <- cardio_clean_factor[h$tr,]
data_test <- cardio_clean_factor[h$ts,]
train_control <- trainControl(method="cv", number=4)
```

Procedemos a entrenar el modelo

```
rpart_tree <- rpart(formula = cardiovascular_disease~., data = data_train)

# Visualizamos el árbol de decisión
rpart.plot(rpart_tree)
```



Calculamos la matriz de confusión para ver la calidad del modelo

```
mod <- train(cardiovascular_disease~., data=data_train, method="rpart",
              trControl = train_control)

pred <- predict(mod, newdata=data_test)

confusionMatrix(pred,data_test$cardiovascular_disease,positive="Sí")

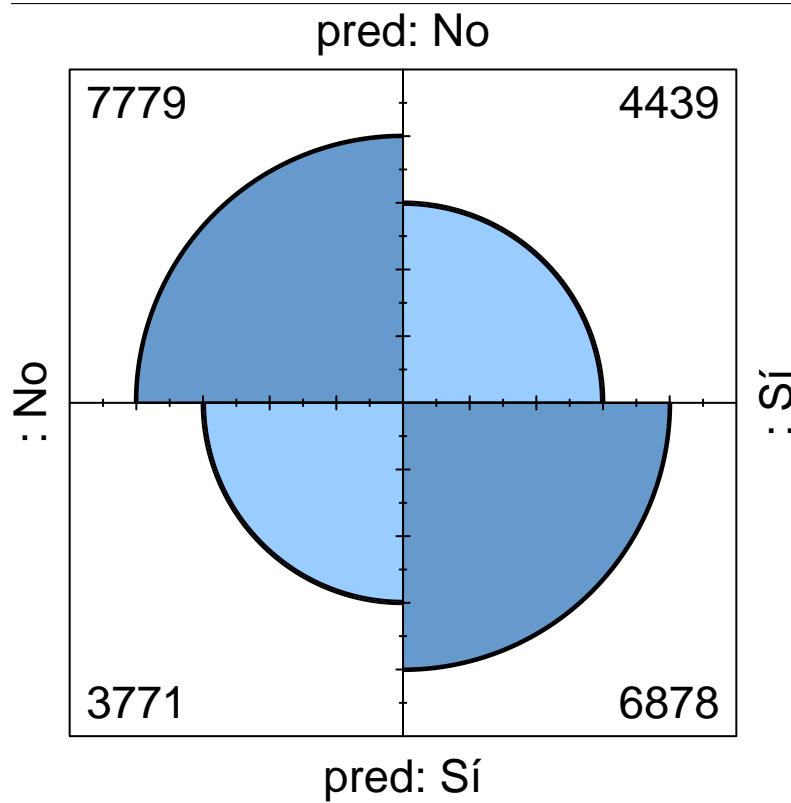
## Confusion Matrix and Statistics
##
##             Reference
## Prediction    No     Sí
##           No 7779  4439
##           Sí 3771  6878
##
##           Accuracy : 0.641
##             95% CI : (0.6347, 0.6472)
##   No Information Rate : 0.5051
##   P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.2814
##
## McNemar's Test P-Value : 1.821e-13
##
##           Sensitivity : 0.6078
##           Specificity  : 0.6735
##   Pos Pred Value : 0.6459
##   Neg Pred Value : 0.6367
##   Prevalence    : 0.4949
##   Detection Rate : 0.3008
## Detection Prevalence : 0.4657
##   Balanced Accuracy : 0.6406
##
## 'Positive' Class : Sí
##
```

Se puede apreciar que la precisión del modelo ha disminuido un poco con respecto al modelo de regresión, pues antes era de un 66.72% y ahora es de un 64.1%. Por lo tanto, su calidad es peor. En cuanto a su predicción, este ha predicho 3771 falsos positivos y 4439 falsos negativos, siendo la sensibilidad del 60.78% y la especificidad del 67.35%.

Vemos de forma gráfica la matriz de confusión

```
cfmtx_2 <- confusionMatrix(table(pred,
                                    data_test[["cardiovascular_disease"]]),
                            positive = "Sí")
fourfoldplot(cfmtx_2$table)
```



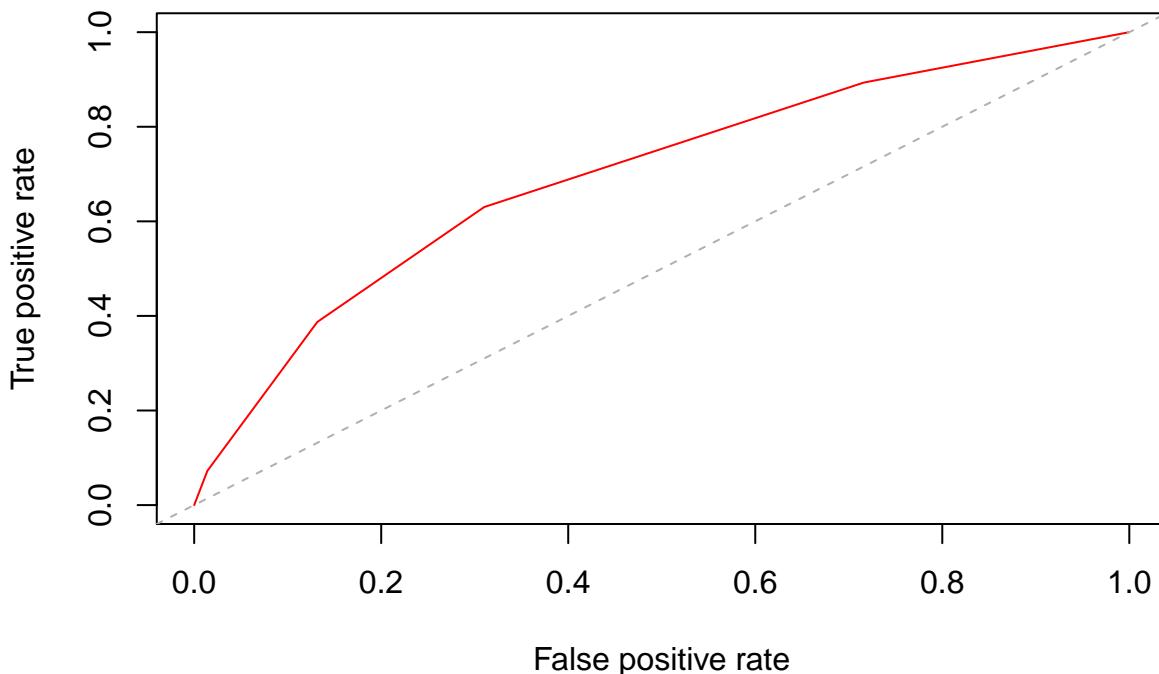


Graficamos ahora la curva de ROC y vemos el AUC

```
probs <- predict(rpart_tree, data_test, type = "prob")[,2]
pred <- prediction(probs, data_test$cardiovascular_disease)
perf <- ROCR::performance(pred, "tpr" , "fpr")
plot(perf,col="red", main="Curva ROC")
abline(0,1, lty = 8, col = "grey")
```



### Curva ROC



```
# Calculamos el área bajo la curva de ROC (AUC)
perf_2 <- ROCR::performance(pred, "auc")
cat("El área bajo la curva de ROC (AUC) es: ", perf_2@y.values[[1]])
```

## El área bajo la curva de ROC (AUC) es: 0.6963832

Fijándonos en el valor del AUC, podemos afirmar que el modelo tiene una probabilidad del 69.64% de clasificar a los pacientes enfermos como enfermos y a los no enfermos como sanos.

Pasamos a estudiar la importancia de las variables exógenas en el árbol de decisión

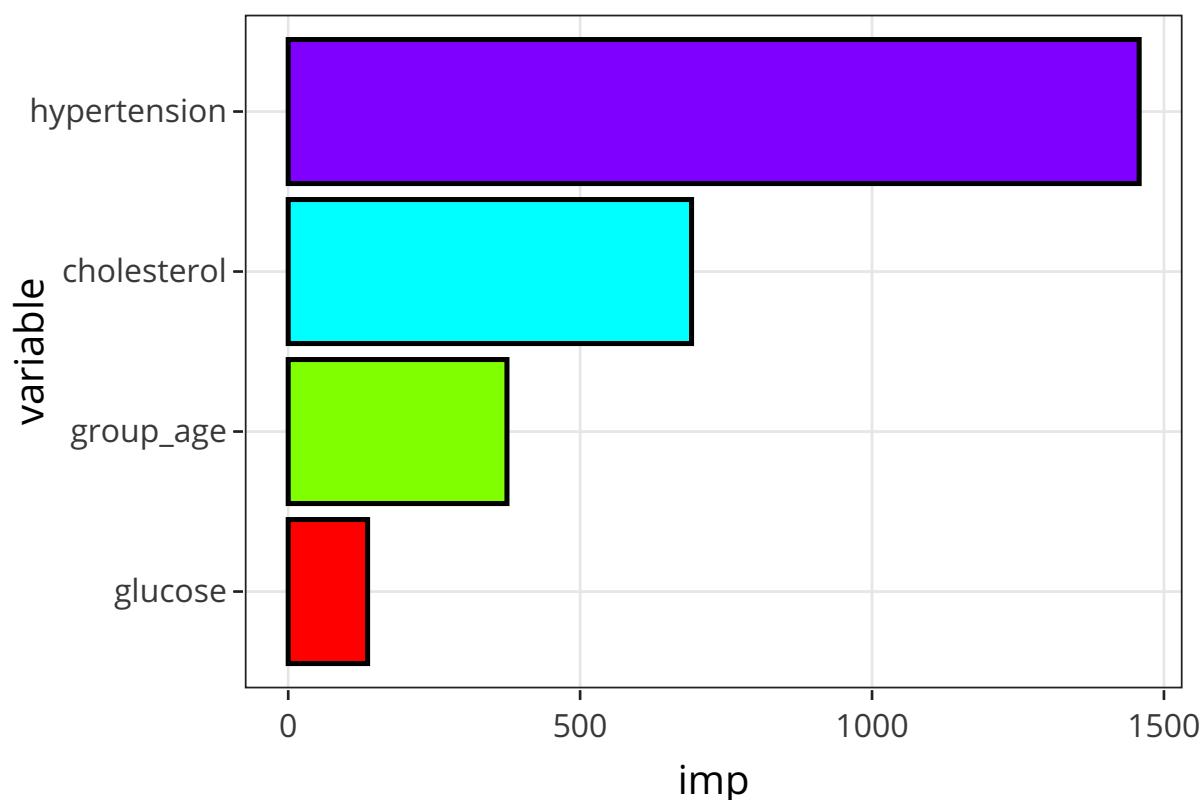
```
rpart_tree$variable.importance
```

```
## hypertension cholesterol group_age glucose
## 1457.5629    690.8900   374.6870   135.9509
```

Vemos como, al igual que ocurría en la regresión logística, las dos variables más importantes y que más influyen a la hora de padecer o no enfermedades cardiovasculares son la hipertensión y el colesterol. La diferencia más notable es que mediante el modelo de regresión la tercera variable más influyente era el IMC y en el árbol de decisión es la edad de la persona, seguida del nivel de glucosa en sangre. Esto lo podemos ver también de forma gráfica

```
df <- data.frame(imp = rpart_tree$variable.importance)
df2 <- df %>%
  tibble::rownames_to_column() %>%
  dplyr::rename("variable" = rowname) %>%
  dplyr::arrange(imp) %>%
  dplyr::mutate(variable =forcats::fct_inorder(variable))
ggplotly(ggplot2::ggplot(df2) +
  geom_col(aes(x = variable, y = imp),
  col = "black", fill=rainbow(n=length(df$imp)),
  show.legend = F) + coord_flip() + scale_fill_grey() +
```





Vemos que no todas las variables que se han utilizado para clasificar la variable objetivo son importantes. Solo las cuatro que vemos en el gráfico son aquellas que el árbol ha tenido en cuenta para dicha clasificación, mientras que el resto han sido desechadas.



## 5. Representación de los resultados a partir de tablas y gráficas.

Dado que hemos desarrollado este punto en el punto 4, en este apartado vamos a aportar información visual adicional y aplicaremos el algoritmo Naive Bayes.

### 5.1 Tabla del dataset

Los datos del dataset utilizado para la aplicación de los algoritmos y tras la correspondiente ETL es:

```
# Desactivado para generar el PDF
library(DT)
datatable(cardio_clean, filter = 'bottom', options = list(pageLength = 5))
```

Search: [ ] entries																		
age	gender	height	weight	systolic_blood_pressure	diastolic_blood_pressure	cholesterol	glucose	smoking	alcohol_intake	physical_activity	cardiovascular_disease	inc	blood_pressure	group_age	group_height	group_weight	group_inc	hypertension
1	50	Hombre	168	62	110	80	Normal	No	No	Si	No	21.9671201814059	90	Adulto	(166, 180)	(3,6f,80)	Peso normal	Presión arterial normal
2	55	Mujer	156	85	140	90	Muy por encima de lo normal	Normal	No	Si	Si	34.927691504484	106.66666666666667	Adulto	(151, 165)	(8f, 120)	Obesidad	Presión arterial normal
3	51	Mujer	165	64	120	70	Muy por encima de lo normal	Normal	No	No	Si	23.507805220871	90	Adulto	(151, 165)	(3,6f,80)	Peso normal	Tensión baja
4	48	Hombre	169	82	150	100	Normal	Normal	No	Si	Si	28.710479249536	116.66666666666667	Adulto	(166, 180)	(8f, 120)	Sobrepeso	Presión arterial normal alta
5	47	Mujer	156	56	100	60	Normal	Normal	No	No	No	23.011768572307	73.32323232323232	Adulto	(151, 165)	(3,6f,80)	Peso normal	Tensión baja

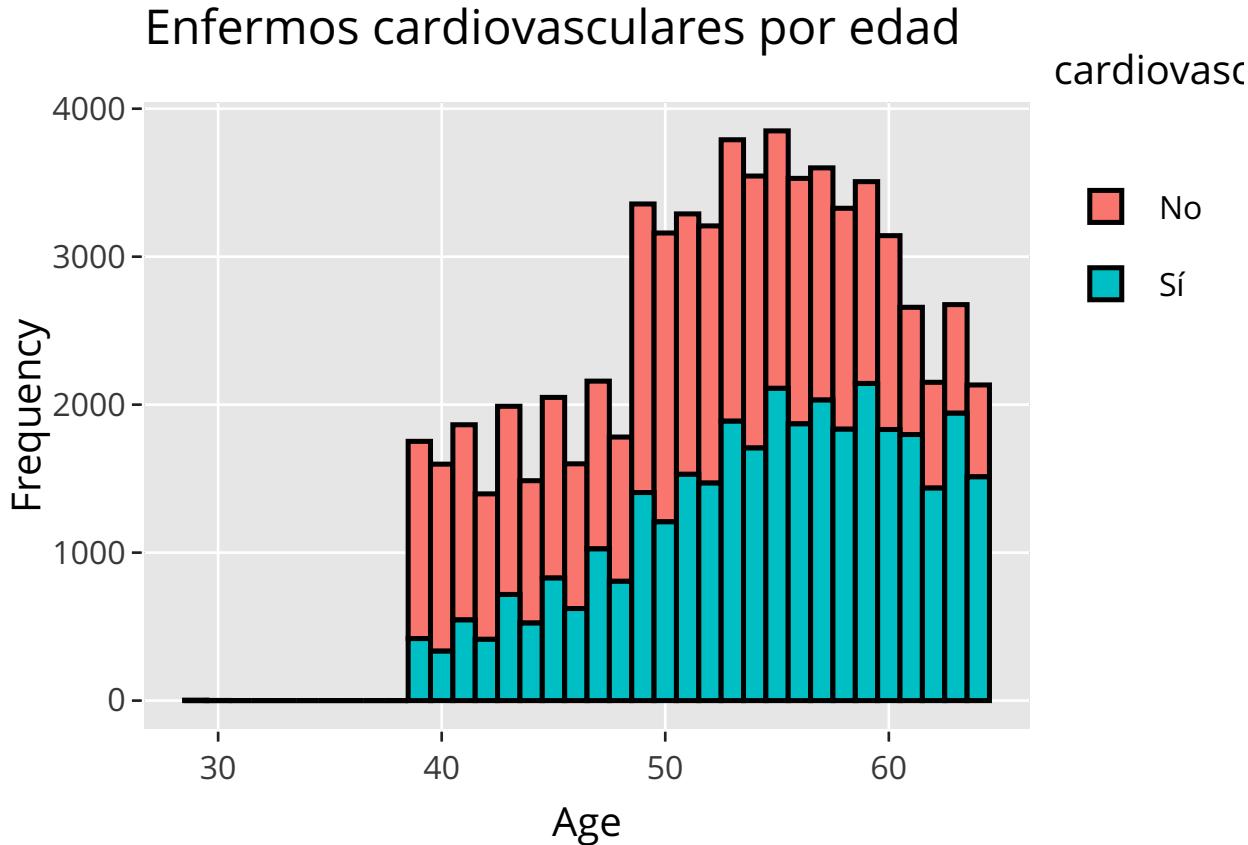
Showing 1 to 5 of 68,600 entries

Previous 1 2 3 4 5 ... 13720 Next

### 5.2 Representaciones gráficas

Representamos mediante un histograma la frecuencia de Enfermos Cardiovasculares relacionados con la Edad

```
ggplotly(ggplot(cardio_clean,aes(x=age,fill=cardiovascular_disease)) +
  geom_histogram(binwidth = 1,color="black") +
  labs(x = "Age",y = "Frequency",
       title = "Enfermos cardiovasculares por edad"))
```



Se puede apreciar cómo a medida que aumenta la edad se incrementan los pacientes con enfermedades cardiovasculares.

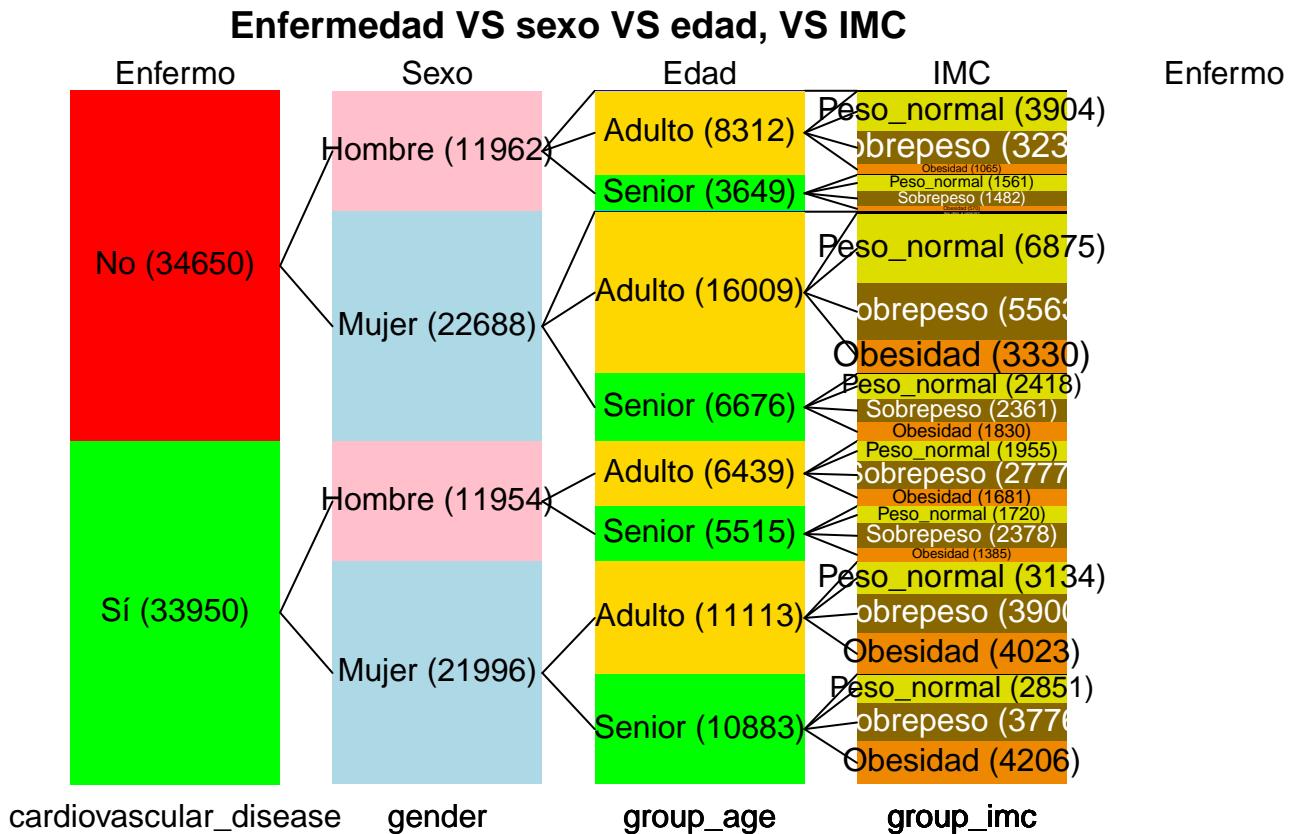


Ahora, veamos como están repartidos los enfermos a través del sexo, edad e IMC

```
#library(plotrix)
shecol<-list( c("red","green"),c("pink","lightblue"),c("blue","gold","green"),
              c("#000000","#dddd00","#886600","#ee8800"))

data<- cardio_clean %>%
  dplyr::select(cardiovascular_disease, gender, group_age, group_imc)

sizetree(data,main="Enfermedad VS sexo VS edad, VS IMC", col=shecol,
         toplab=c("Enfermo" , "Sexo", "Edad" , "IMC"))
```



### 5.3 Algoritmo Naive Bayes

Naive Bayes es un modelo de predicción basado en la probabilidad Bayesiana. El modelo es muy simple, pero poderoso, en cuanto a que es resultado directo de los datos y su tratamiento se basa en simple estadística bayesiana de la probabilidad condicionada. Hay que tener en cuenta que se asume, por simplificación que las variables son todas sucesos independientes.

El modelo bayesiano de probabilidad condicionada se representa como:

$$(P(A|B) = P(A \cap B)/P(B))$$

Es decir, la probabilidad de que se dé el caso ( $A$ ) dado ( $B$ ) es igual a la probabilidad de la intersección de  $A$  con  $B$  ( $A \cap B$ ) partido de la probabilidad de ( $B$ )

Estirando esta formulación llegaríamos al teorema de Bayes, cuya expresión más típica es la siguiente:



$$(P(A|B) = PP(B|A) * P(A)/P(B))$$

```
# Creamos los conjuntos mediante el método de validación cruzada
set.seed(159)
h <- holdout(cardio_clean$cardiovascular_disease, ratio=2/3,
              mode="stratified")
data_train <- cardio_clean[h$tr,]
data_test <- cardio_clean[h$ts,]
train_control <- trainControl(method="cv", number=4)

# library(e1071) contains the naivebayes model, for that reason we have to first call this library.
#library(e1071)
model_nb <- naiveBayes(cardiovascular_disease~age+gender+smoking+
                        physical_activity+group_imc+glucose+alcohol_intake+
                        hypertension+cholesterol, data = data_train)

model_nb

##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##      No      Sí
## 0.5051057 0.4948943
##
## Conditional probabilities:
## age
## Y      [,1]    [,2]
## No 51.24758 6.80479
## Sí 54.46344 6.34699
##
## gender
## Y      Hombre     Mujer
## No 0.3455844 0.6544156
## Sí 0.3509477 0.6490523
##
## smoking
## Y          No      Sí
## No 0.90722944 0.09277056
## Sí 0.91485883 0.08514117
##
## physical_activity
## Y          No      Sí
## No 0.1811255 0.8188745
## Sí 0.2135819 0.7864181
##
## group_imc
## Y  Peso_inferior_al_normal Peso_normal   Sobrepeso   Obesidad
## No           0.013506494 0.428268398 0.363722944 0.194502165
## Sí           0.004904343 0.285291389 0.376088013 0.333716255
```



```
##
##      glucose
## Y      Normal Por_encima_de_lo_normal Muy_por_encima_de_lo_normal
##  No   0.88367965          0.05948052          0.05683983
##  Sí   0.81557902          0.08845491          0.09596607
##
##      alcohol_intake
## Y      No      Sí
##  No   0.94385281 0.05614719
##  Sí   0.94812884 0.05187116
##
##      hypertension
## Y      Tensión_baja Presión_arterial_normal Presión_arterial_normal_alta
##  No   0.278398268          0.696709957          0.020909091
##  Sí   0.109795431          0.766137940          0.107011885
##      hypertension
## Y      Hipertensión
##  No   0.003982684
##  Sí   0.017054743
##
##      cholesterol
## Y      Normal Por_encima_de_lo_normal Muy_por_encima_de_lo_normal
##  No   0.83839827          0.10874459          0.05285714
##  Sí   0.65819821          0.16361066          0.17819114
```

Comprobamos el modelo con los datos de entrenamiento y creamos su matriz de confusión para conocer la precisión del modelo.

```
mod <- train(cardiovascular_disease~age+gender+smoking+
               physical_activity+group_imc+glucose+alcohol_intake+
               hypertension+cholesterol, data=data_train, method="nb",
               trControl = train_control)

pred <- predict(mod, newdata=data_train)

confusionMatrix(pred,data_train$cardiovascular_disease,positive="Sí")
```

```
## Confusion Matrix and Statistics
##
##      Reference
## Prediction  No      Sí
##  No   19888 14208
##  Sí   3212  8425
##
##      Accuracy : 0.6191
##      95% CI : (0.6146, 0.6235)
##      No Information Rate : 0.5051
##      P-Value [Acc > NIR] : < 2.2e-16
##
##      Kappa : 0.2343
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##      Sensitivity : 0.3722
##      Specificity : 0.8610
```



```
##           Pos Pred Value : 0.7240
##           Neg Pred Value : 0.5833
##           Prevalence : 0.4949
##           Detection Rate : 0.1842
##   Detection Prevalence : 0.2545
##           Balanced Accuracy : 0.6166
##
##           'Positive' Class : Sí
##
```

Podemos decir que el algoritmo de Bayes tiene una precisión del 61.91% para los datos de entrenamiento.

Ahora validaremos el modelo con los datos de prueba del mismo modo que como lo acabamos de hacer, mediante la predicción y la creación de la matriz de confusión

```
pred <- predict(mod, newdata=data_test)
confusionMatrix(pred,data_test$cardiovascular_disease,positive="Sí")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No    Sí
##       No 9906 7203
##       Sí 1644 4114
##
##           Accuracy : 0.6131
##             95% CI : (0.6068, 0.6194)
##   No Information Rate : 0.5051
##   P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.2223
##
##   Mcnemar's Test P-Value : < 2.2e-16
##
##           Sensitivity : 0.3635
##           Specificity : 0.8577
##           Pos Pred Value : 0.7145
##           Neg Pred Value : 0.5790
##           Prevalence : 0.4949
##           Detection Rate : 0.1799
##   Detection Prevalence : 0.2518
##           Balanced Accuracy : 0.6106
##
##           'Positive' Class : Sí
##
```

Podemos concluir afirmando que el algoritmo de Bayes tiene una precisión del 61.31% o, lo que es lo mismo, una tasa de clasificación errónea del 38.69%, prediciendo 1644 falsos positivos y 7203 falsos negativos. La sensibilidad del modelo es del 36.35% (bastante baja) y la especificidad es del 85.77%..

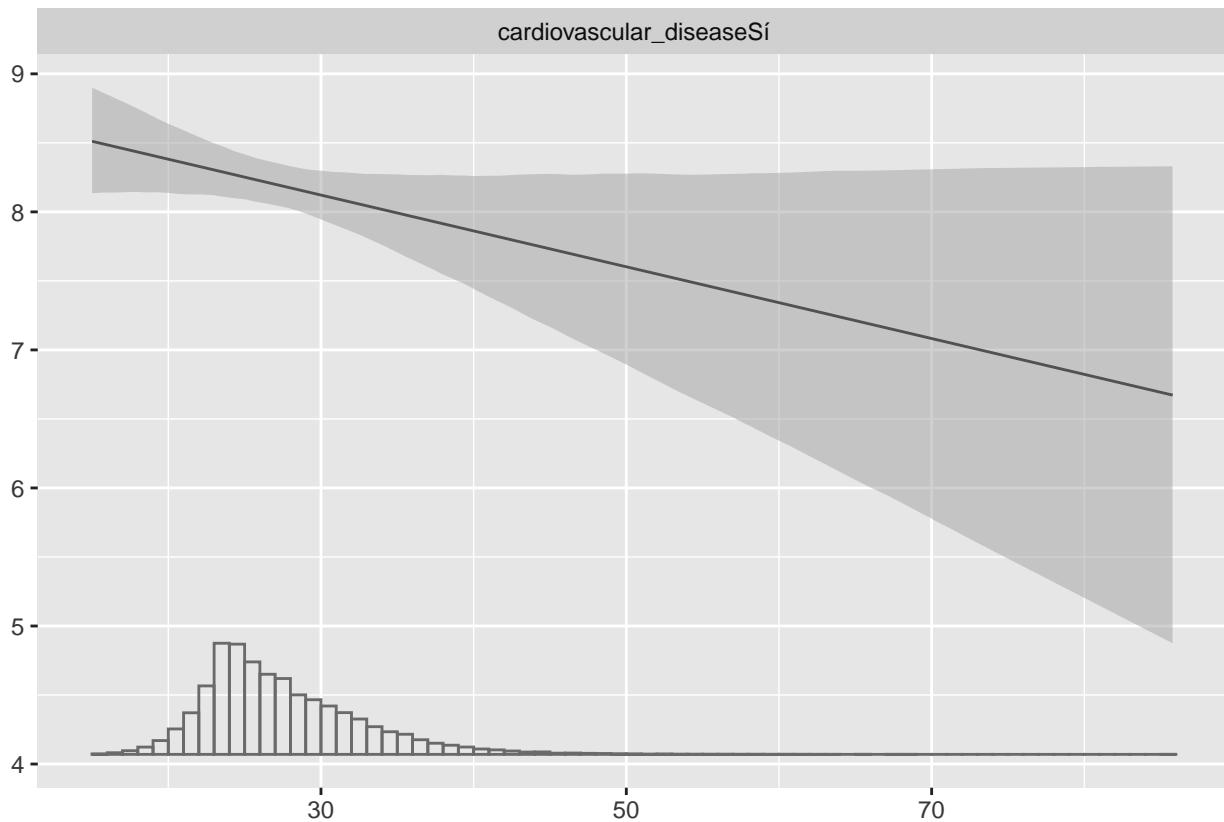
## 5.4 Relación Enfermedad vs IMC y Presión Arterial

Supongamos que nos interesa saber cómo afecta la enfermedad a la relación entre el IMC y la presión arterial, y cómo afecta el número de cilindros a la relación entre la enfermedad y la presión arterial. Estos efectos condicionales se modelan utilizando un término de interacción multiplicativo de dos vías.



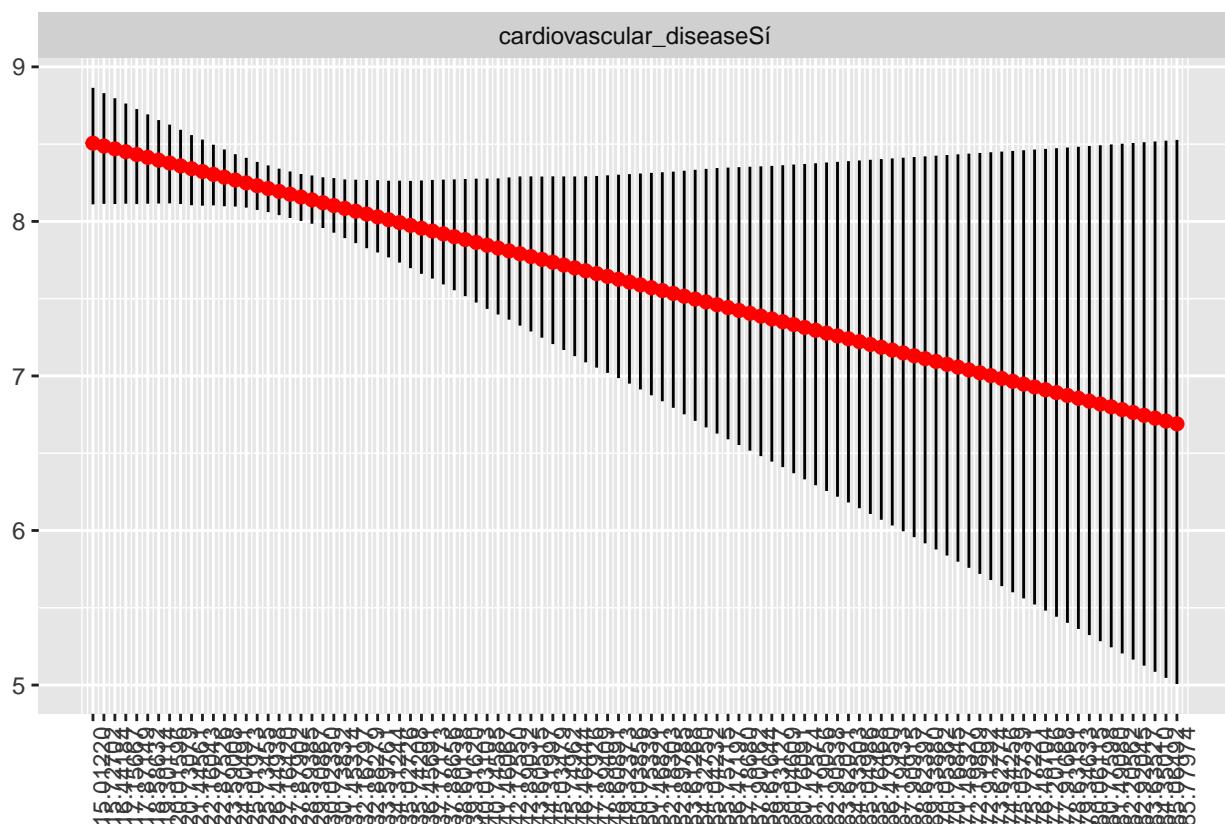
```
library(interplot)
lm_age <- lm(blood_pressure ~ cardiovascular_disease*imc, data=cardio_clean)

interplot(m=lm_age, var1 = "cardiovascular_disease", var2 = "imc", hist = TRUE)
```



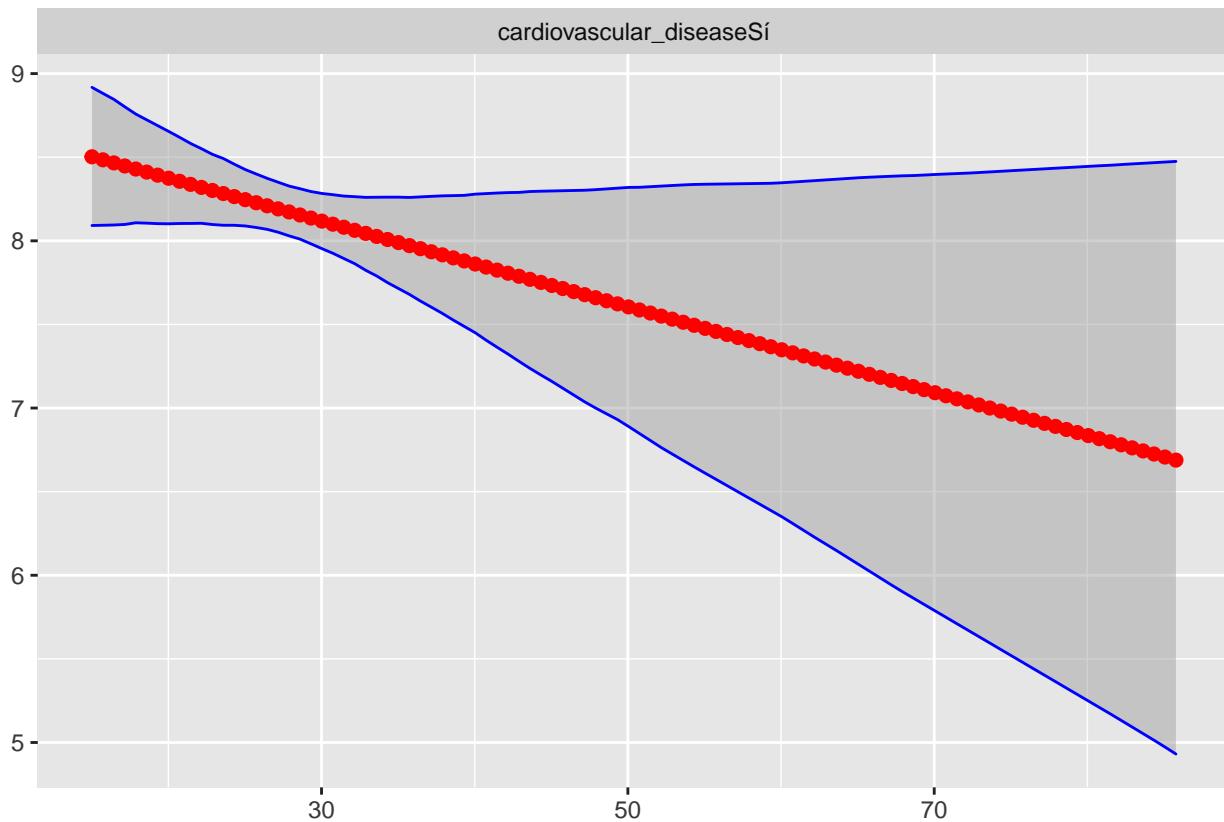
```
interplot(m = lm_age, var1 = "cardiovascular_disease", var2 = "imc", point = T) +
  # changing the angle of x labels for a clearer vision
  theme(axis.text.x = element_text(angle=90)) +
  geom_point(size = 2, color = "red")
```





```
interplot(m = lm_age, var1 = "cardiovascular_disease", var2 = "imc", ercolor = "blue", esize = 1.5) +  
  geom_point(size = 2, color = "red")
```





## 5.5 Intervalo de densidades de los atributos del dataset

```
#library(bayestestR)
#library(see)
library(rstanarm)

model <- stan_glm(blood_pressure ~ cardiovascular_disease*imc*age*gender,
                   data=cardio_clean)

## 
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 3.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.38 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration: 1 / 2000 [  0%] (Warmup)
## Chain 1: Iteration: 200 / 2000 [ 10%] (Warmup)
## Chain 1: Iteration: 400 / 2000 [ 20%] (Warmup)
## Chain 1: Iteration: 600 / 2000 [ 30%] (Warmup)
## Chain 1: Iteration: 800 / 2000 [ 40%] (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%] (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%] (Sampling)
```

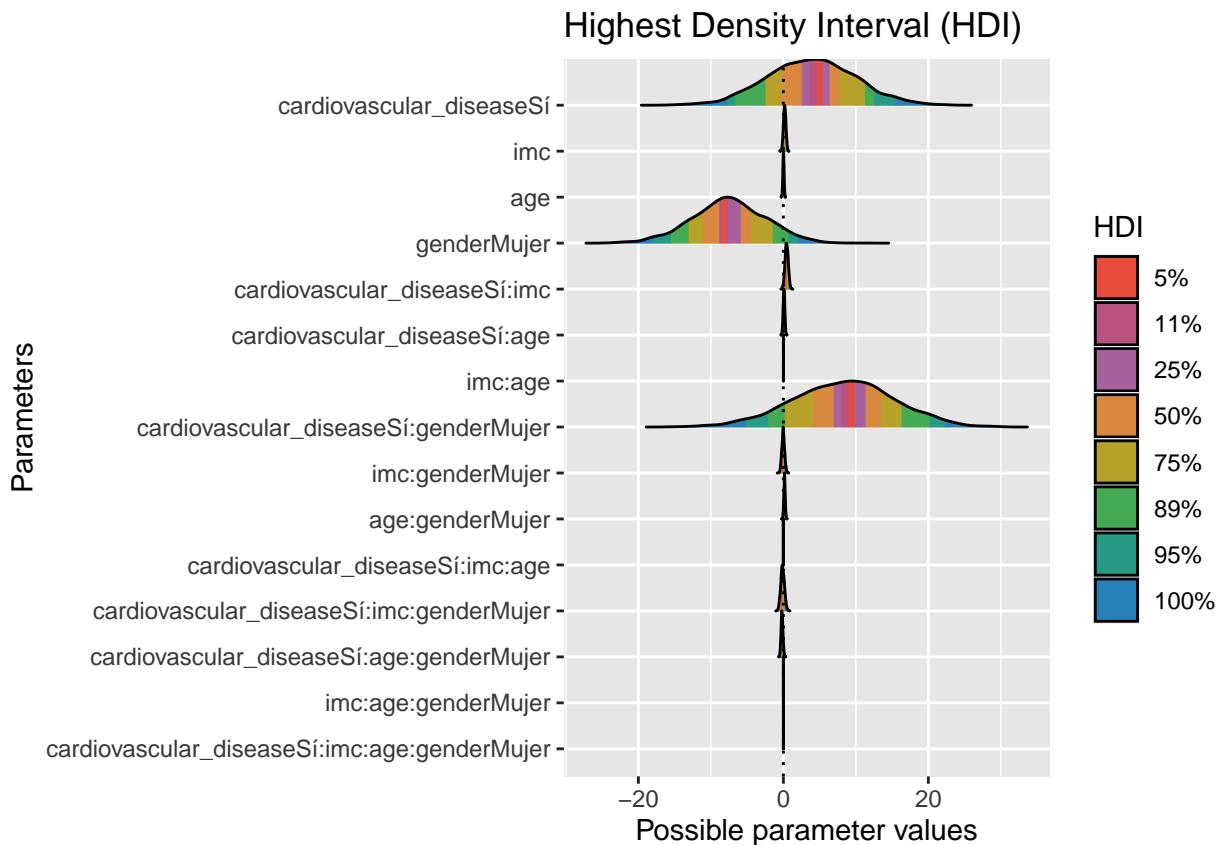


```
## Chain 1: Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%] (Sampling)
## Chain 1:
## Chain 1: Elapsed Time: 14.0599 seconds (Warm-up)
## Chain 1:           17.3024 seconds (Sampling)
## Chain 1:           31.3623 seconds (Total)
## Chain 1:
## 
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.2e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.22 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration: 1 / 2000 [  0%] (Warmup)
## Chain 2: Iteration: 200 / 2000 [ 10%] (Warmup)
## Chain 2: Iteration: 400 / 2000 [ 20%] (Warmup)
## Chain 2: Iteration: 600 / 2000 [ 30%] (Warmup)
## Chain 2: Iteration: 800 / 2000 [ 40%] (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%] (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%] (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%] (Sampling)
## Chain 2:
## Chain 2: Elapsed Time: 7.80057 seconds (Warm-up)
## Chain 2:           19.6731 seconds (Sampling)
## Chain 2:           27.4737 seconds (Total)
## Chain 2:
## 
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.3e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.23 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration: 1 / 2000 [  0%] (Warmup)
## Chain 3: Iteration: 200 / 2000 [ 10%] (Warmup)
## Chain 3: Iteration: 400 / 2000 [ 20%] (Warmup)
## Chain 3: Iteration: 600 / 2000 [ 30%] (Warmup)
## Chain 3: Iteration: 800 / 2000 [ 40%] (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%] (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%] (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%] (Sampling)
## Chain 3:
## Chain 3: Elapsed Time: 9.36758 seconds (Warm-up)
```



```
## Chain 3:           15.4938 seconds (Sampling)
## Chain 3:           24.8613 seconds (Total)
## Chain 3:
## 
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.9e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration: 1 / 2000 [  0%] (Warmup)
## Chain 4: Iteration: 200 / 2000 [ 10%] (Warmup)
## Chain 4: Iteration: 400 / 2000 [ 20%] (Warmup)
## Chain 4: Iteration: 600 / 2000 [ 30%] (Warmup)
## Chain 4: Iteration: 800 / 2000 [ 40%] (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%] (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%] (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%] (Sampling)
## Chain 4:
## Chain 4: Elapsed Time: 13.8237 seconds (Warm-up)
## Chain 4:           14.384 seconds (Sampling)
## Chain 4:           28.2077 seconds (Total)
## Chain 4:
plot(hdi(model, ci = c(0.05, 0.11, 0.25, 0.5, 0.75, 0.89, 0.95)))
```





El intervalo de mayor densidad (posterior) para una muestra de valores es el intervalo más estrecho que contiene el x% de los datos. El HPDI del 90% sería el intervalo más pequeño que contiene el 90% de los datos.



---

**6. RESOLUCIÓN DEL PROBLEMA. A PARTIR DE LOS RESULTADOS OBTENIDOS, ¿CUÁLES SON LAS CONCLUSIONES? ¿LOS RESULTADOS PERMITEN RESPONDER AL PROBLEMA?**

---

## **6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?**

En base al estudio realizado, se han obtenido unos resultados bastante interesantes.

En primer lugar, mediante los contrastes de hipótesis realizados hemos visto que, con un 95% de confianza, el Índice de Masa corporal es diferente entre hombres y mujeres, y la presión arterial media es mayor en personas con enfermedades cardiovasculares que en personas sanas.

En segundo lugar, gracias al modelo de regresión logística vimos que las variables categóricas que más influyen a la hora de padecer una enfermedad cardiovascular son la hipertensión, el colesterol y el peso de la persona, siendo la precisión de dicho modelo del 66.72%.

En tercer lugar, una vez aplicado otro algoritmo de clasificación como el árbol de decisión se observó que, con una calidad del 64.1% (algo menor que la del modelo anterior), de nuevo, las dos variables que más importancia tienen a la hora de padecer o no una enfermedad cardiovascular fueron la hipertensión y el colesterol, mientras que la tercera variable, a diferencia del modelo de regresión, fue el peso de la persona, seguida del nivel de glucosa en sangre.

Finalmente, el algoritmo clasificador probabilístico fundamentado en el teorema de Bayes, conocido como Naive Bayes, obtuvo una precisión del 61.31%, siendo esta calidad la menor de las obtenidas en los modelos de clasificación que hemos empleado en este estudio.

Podemos concluir en que se ha conseguido responder a la pregunta que se formuló al inicio del estudio, ya que ahora podemos afirmar, en base a los resultados obtenidos, que unos niveles de presión en sangre altos (hipertensión) y unos niveles de colesterol muy por encima de lo normal (en este orden) tienen una influencia directa a la hora de aumentar la probabilidad de padecer una enfermedad cardiovascular. Además, un Índice de Masa corporal alto (persona obesa) (en base la regresión logística) o el aumento de la edad (en base al árbol de decisión) son también factores que tienen un peso importante sobre este tipo de enfermedades.



*7. CÓDIGO: HAY QUE ADJUNTAR EL CÓDIGO, PREFERIBLEMENTE EN R, CON EL QUE SE HA REALIZADO LA LIMPIEZA, ANÁLISIS Y REPRESENTACIÓN DE LOS DATOS. SI LO PREFERÍS, TAMBIÉN PODÉIS TRABAJAR EN PYTHON.*

**7. Código:** Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y representación de los datos. Si lo preferís, también podéis trabajar en Python.

El código realizado en lenguaje R se puede ver en el repositorio de GitHub, concretamente en la carpeta *code*.



## 8. Dónde consultar el proyecto

- **GITHUB**
- **DATA**
- **CODE**
- **PDF**
- **README**



## **9. Contribuciones al trabajo**

Contribuciones	Firma
Investigación Previa	ORM, ARP
Redacción de las respuestas	ORM, ARP
Desarrollo código	ORM, ARP

