Tipología y ciclo de vida de los datos: PRACT2 - Limpieza de datos

Sergio Sánchez Romero y Lucia Blanc Velázquez

Diciembre 2023

ÍNDICE

1 Introducción

1.1 Presentación

En esta práctica se elabora un caso práctico orientado a aprender a identificar los datos relevantes para un proyecto analítico y usar las herramientas de integración, limpieza, validación y análisis de los mismos.

1.2 Objetivos

● Aprender a aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios o multidisciplinares.

● Saber identificar los datos relevantes y los tratamientos necesarios (integración, limpieza y validación) para llevar a cabo un proyecto analítico.

● Aprender a analizar los datos adecuadamente para abordar la información contenida en los datos.

● Identificar la mejor representación de los resultados para aportar conclusiones sobre el problema planteado en el proceso analítico.

● Actuar con los principios éticos y legales relacionados con la manipulación de datos en función del ámbito de aplicación.

● Desarrollar las habilidades de aprendizaje que les permitan continuar estudiando de un modo que tendrá que ser en gran medida autodirigido o autónomo.

● Desarrollar la capacidad de búsqueda, gestión y uso de información y recursos en el ámbito de la ciencia de datos.

2 Descripción del conjunto de datos

El conjunto de datos escogido para esta práctica, se titula: **"Heart Attack Analysis & Prediction dataset"**, el cual tiene como objetivo detectar aquellos factores que pueden actuar como potenciales precursores de las enfermedades cardiovasculares, y así ayudar a la detección y gestión temprana mediante la creación de un modelo.

En el conjunto de datos se contemplan un total de 14 características importantes, tanto numéricas como categóricas que pueden ayudar a la predicción de desarrollar o no una enfermedad cardíaca. Según la información encontrada, se sabe que las enfermedades cardiovasculares ocupan un porcentaje del 31% en relación a las muertes que se producen en el mundo cada año, por lo que supone una de las causas principales de muerte. La variable 'output', consiste en valores de 0 o 1 que indican si una persona tiene más probabilidad de sufrir un infarto (1) o menor probabilidad (0).

Es importante tener en cuenta cuales son los atributos que pueden conllevar a un mayor riesgo cardiovascular o al desarrollo de enfermedad cardiovascular, por lo que, el objetivo es predecir que variables influyen más en este desarrollo.

A continuación, realizamos la descripción de las variables que hay en el dataset "Heart Attack Analysis & Prediction dataset", usando la información encontrada en la web [Kaggle datasets] (https://www.kaggle.com/datasets (https://www.kaggle.com/datasets)), concretamente en el siguiente enlace: https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset (https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset)

** Características de las variables incluidas:**

- **age**: Edad del paciente
- **sex** : Sexo del paciente (F=0; M=1)
- **cp** : Tipo dolor torácico
    - **Value 1** : Angina típica (TA)
    - **Value 2** : Angina atípica (ATA)
    - **Value 3** : Dolor no-anginal (NAP)
    - **Value 4** : Asintomático (ASY)
- **trtbps** : Presión arterial en reposo (in mm Hg)
- **chol** : Colesterol en mg/dl obtenido a través del sensor de IMC
- **fbs** : (Glucemia en ayunas > 120 mg/dl) (1 = true; 0 = false)
- **restecg** : Resultados del electrocardiograma en reposo
    - **Value 0** : Normal
    - **Value 1** : Presentar anomalías de la onda ST-T (inversión de la onda T y/o elevación o depresión del ST de > 0,05 mV)
    - **Value 2** : Hipertrofia ventricular izquierda probable o definida según los criterios de Estes
- **thalachh** : Frecuencia cardiaca máxima alcanzada
- **exng** : Angina inducida por esfuerzo (1 = yes; 0 = no)
- **oldpeak** : Pico previo
- **slp** : Pendiente del segmento ST máximo del ejercicio
- **caa** : Número de buques principales (0-4)
- **thall** : Tasa de mortalidad
- **output** : 0= menor probabilidad de infarto 1= mayor probabilidad de infarto

Nota: Aunque en Kaggle la descripción del dataset habla de la variable **ca** como el número de vasos principales (0-3) coloreados por fluoroscopia vemos como en el fichero csv la columna se llama **caa** y los valores que existen van de 0 a 4. Este detalle también parece que esta mal de donde probablemente se han extraído los datos (https://archive.ics.uci.edu/dataset/45/heart+disease (https://archive.ics.uci.edu/dataset/45/heart+disease)). Seguiremos llamando a nuestra variable caa pero contemplando el valor 4.

3 Integración y selección de los datos de interés a analizar

Puede ser el resultado de adicionar diferentes datasets o una subselección útil de los datos originales, en base al objetivo que se quiera conseguir.

3.1 Carga de Librerías

Primero de todo, cargamos las librerías que vamos a usar durante la práctica.

```
# Lista de paquetes
packages <- c('csv', 'dplyr', 'ggplot2', 'reshape', 'plotly', 'plyr', 'Stat2Data',
              'Matrix', 'patchwork', 'ggcorrplot', 'corrplot', 'DataExplorer', 'psych',
              'highcharter', 'tidyverse', 'GGally', 'htmltools', 'RColorBrewer',
              'dendextend', 'cluster', 'factoextra', 'caTools', 'gridExtra', 'car',
              'randomForest', 'class', 'pROC')

# Instalar y cargar paquetes
for (package in packages) {
  if (!require(package, character.only = TRUE)) {
    install.packages(package)
    library(package, character.only = TRUE)
  }
}
# Set working directory
dir <- dirname(rstudioapi::getSourceEditorContext()$path)
setwd(dir)
```

3.2 Carga del dataset

Cargamos los datos de la base de datos “heart” y tipificamos las variables que tiene el conjunto de datos como corresponde

```
heart <- read.csv("heart.csv", header=T,sep=",")
```

3.3 Observación del dataset

Mostramos los primeros registros del conjunto de datos, con el fin de ver una aproximación de cómo es el conjunto y su estructura.

```
head(heart, max(10))
```

```
##    age sex cp trtbps chol fbs restecg thalachh exng oldpeak slp caa thall
## 1   63   1  3    145  233   1       0      150    0     2.3   0   0     1
## 2   37   1  2    130  250   0       1      187    0     3.5   0   0     2
## 3   41   0  1    130  204   0       0      172    0     1.4   2   0     2
## 4   56   1  1    120  236   0       1      178    0     0.8   2   0     2
## 5   57   0  0    120  354   0       1      163    1     0.6   2   0     2
## 6   57   1  0    140  192   0       1      148    0     0.4   1   0     1
## 7   56   0  1    140  294   0       0      153    0     1.3   1   0     2
## 8   44   1  1    120  263   0       1      173    0     0.0   2   0     3
## 9   52   1  2    172  199   1       1      162    0     0.5   2   0     3
## 10  57   1  2    150  168   0       1      174    0     1.6   2   0     2
##    output
## 1       1
## 2       1
## 3       1
## 4       1
## 5       1
## 6       1
## 7       1
## 8       1
## 9       1
## 10      1
```

```
str(heart)
```

```
## 'data.frame':    303 obs. of  14 variables:
##  $ age     : int  63 37 41 56 57 57 56 44 52 57 ...
##  $ sex     : int  1 1 0 1 0 1 0 1 1 1 ...
##  $ cp      : int  3 2 1 1 0 0 1 1 2 2 ...
##  $ trtbps  : int  145 130 130 120 120 140 140 120 172 150 ...
##  $ chol    : int  233 250 204 236 354 192 294 263 199 168 ...
##  $ fbs     : int  1 0 0 0 0 0 0 0 1 0 ...
##  $ restecg : int  0 1 0 1 1 1 0 1 1 1 ...
##  $ thalachh: int  150 187 172 178 163 148 153 173 162 174 ...
##  $ exng    : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ oldpeak : num  2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
##  $ slp     : int  0 0 2 2 2 1 1 2 2 2 ...
##  $ caa     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ thall   : int  1 2 2 2 2 1 2 3 3 2 ...
##  $ output  : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
# Obtener la cantidad de filas y columnas
dimensiones <- dim(heart)
num_filas <- dimensiones[1]
num_columnas <- dimensiones[2]

# Imprimir el mensaje
mensaje <- paste("El conjunto tiene", num_filas, "filas y", num_columnas, "columnas.")
print(mensaje)
```

```
## [1] "El conjunto tiene 303 filas y 14 columnas."
```

Obtenemos un primer vistazo estadístico de cada atributo.

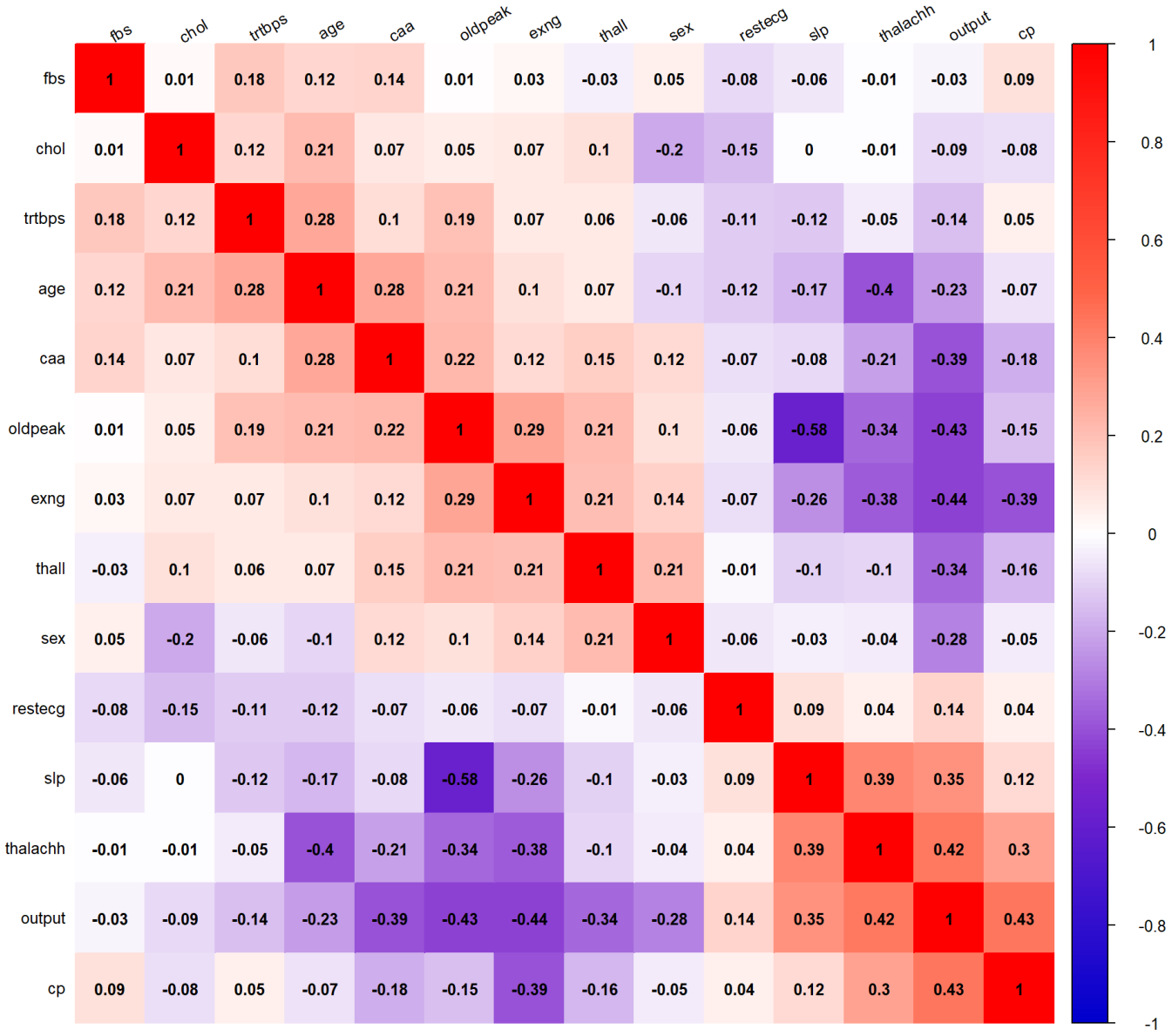
```
describeBy(heart)
```

```
##          vars   n   mean    sd median trimmed   mad min   max range  skew
## age         1 303  54.37  9.08   55.0   54.54 10.38  29  77.0  48.0 -0.20
## sex         2 303   0.68  0.47    1.0    0.73  0.00   0   1.0   1.0 -0.78
## cp          3 303   0.97  1.03    1.0    0.86  1.48   0   3.0   3.0  0.48
## trtbps      4 303 131.62 17.54  130.0  130.44 14.83  94 200.0 106.0  0.71
## chol        5 303 246.26 51.83  240.0  243.49 47.44 126 564.0 438.0  1.13
## fbs         6 303   0.15  0.36    0.0    0.06  0.00   0   1.0   1.0  1.97
## restecg     7 303   0.53  0.53    1.0    0.52  0.00   0   2.0   2.0  0.16
## thalachh    8 303 149.65 22.91  153.0  150.98 22.24  71 202.0 131.0 -0.53
## exng        9 303   0.33  0.47    0.0    0.28  0.00   0   1.0   1.0  0.74
## oldpeak    10 303   1.04  1.16    0.8    0.86  1.19   0   6.2   6.2  1.26
## slp        11 303   1.40  0.62    1.0    1.46  1.48   0   2.0   2.0 -0.50
## caa        12 303   0.73  1.02    0.0    0.54  0.00   0   4.0   4.0  1.30
## thall      13 303   2.31  0.61    2.0    2.36  0.00   0   3.0   3.0 -0.47
## output     14 303   0.54  0.50    1.0    0.56  0.00   0   1.0   1.0 -0.18
##          kurtosis   se
## age         -0.57 0.52
## sex         -1.39 0.03
## cp          -1.21 0.06
## trtbps       0.87 1.01
## chol         4.36 2.98
## fbs          1.88 0.02
## restecg     -1.37 0.03
## thalachh    -0.10 1.32
## exng        -1.46 0.03
## oldpeak      1.50 0.07
## slp         -0.65 0.04
## caa          0.78 0.06
## thall        0.25 0.04
## output      -1.97 0.03
```

4 Limpieza de los datos

Primero de todo, hacemos una matriz de correlaciones de todo el conjunto, para ver la relación lineal entre variables y comprender como se relacionan entre sí.

```
# Convertimos todas las variables a numéricas
heart_numeric <- sapply(heart, as.numeric)

 # Calculamos la matriz de correlación
correlation_tab <- cor(heart_numeric)

# Definimos una nueva paleta de colores
new_col <- colorRampPalette(c("#0000CD", "#7D26CD", "#FFFFFF","#FF6347","#FF0000"))

# Crear la matriz de correlación con corrplot y personalización adicional
corrplot(correlation_tab,
         method = "color",
         tl.col = "black",
         tl.srt = 30,
         tl.cex = 0.8,
         cl.cex = 0.8,
         col = new_col(200),
         addCoef.col = "black",
         order = "AOE",
         number.cex = 0.8)
```



Observaciones de la matriz de correlación

Según la matriz de correlaciones inicial, presentada anteriormente, donde las correlaciones positivas se muestran en color rojo y las negativas en color azul fuerte podemos ver cuales son las variables con mayores intensidad de color, guiando así la significación de los coeficientes de correlación.

- La variable de interés **output** se **relaciona positivamente** con **slp**, **thalachh** y **cp**, y **negativamente** con **caa**, **oldpeal**, **exng** y **thall**.
- **No existe ningún tipo de relación entre la variable output y la variable chol ni la variable fbs**.
- Hay una **correlación moderada negativa entre slp y oldpeak** (-0.58), y una **correlación moderada positiva entre slp y thalachh**.
- Las variables **cp y exng estan correlacionadas negativamente, igual que la edad (age) y la variable thalachh**.

4.1 Valores nulos y/o vacíos

Debido que algunas de las variables del conjunto de datos tienen tipos de datos incorrectos, debemos transformar los tipos de datos antes del análisis.

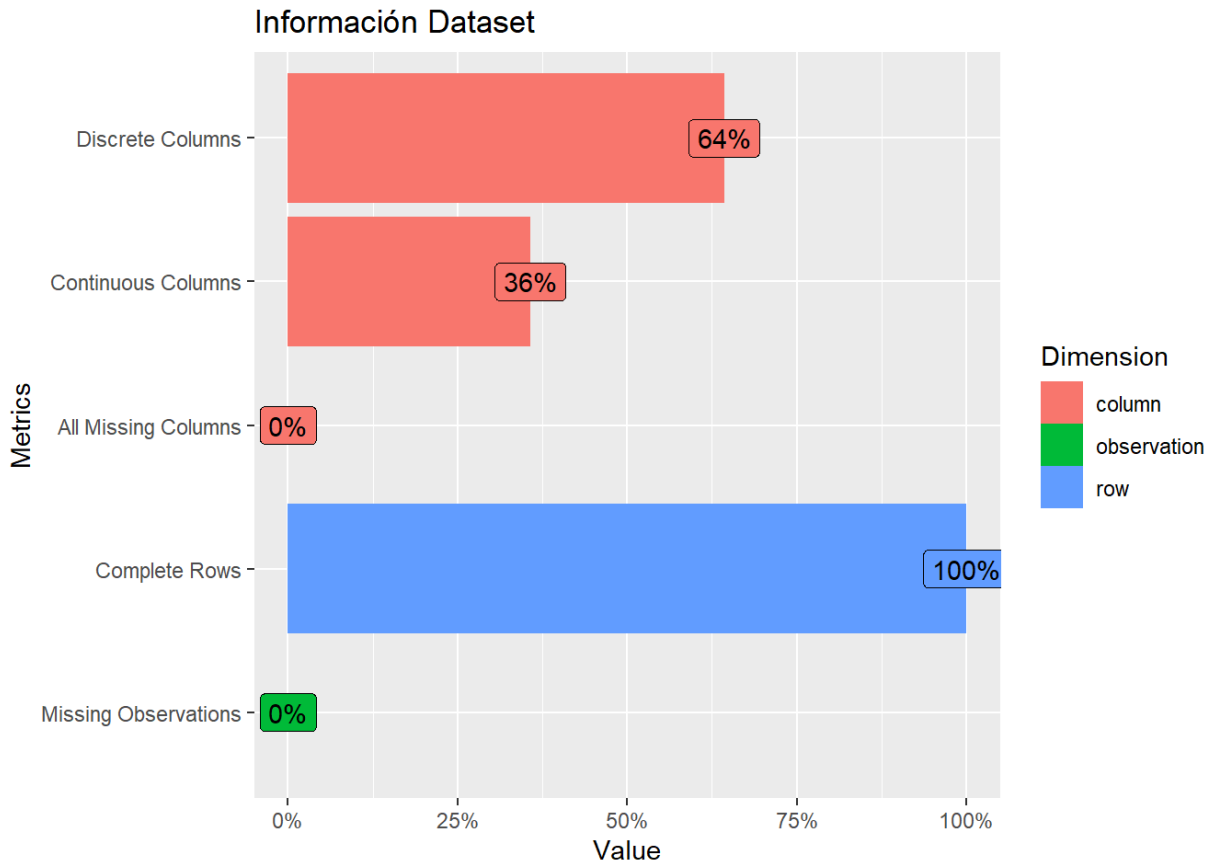
```
# Clasificamos las variables en numéricas o en categóricas
# Numéricas
heart <- heart%>%
  mutate_at(vars(age,trtbps,chol, thalachh, oldpeak), as.numeric)

# Categóricas
heart <- heart%>%
  mutate_at(vars(sex, cp, fbs, restecg, exng, slp, thall, caa, output), as.factor)

introduce(heart)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1  303      14                9                  5                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                    0           303               4242        31880
```

Según la tabla y el barplot creado, podemos ver como el conjunto de datos heart tiene 14 atributos y 303 observaciones, y contiene un total de 9 columnas discretas y 5 columnas continuas.

Con la librería DataExplorer vemos una vista general del conjunto de datos de análisis una vez seleccionados, basandonos en los valores faltantes, columnas discretas y continuas.

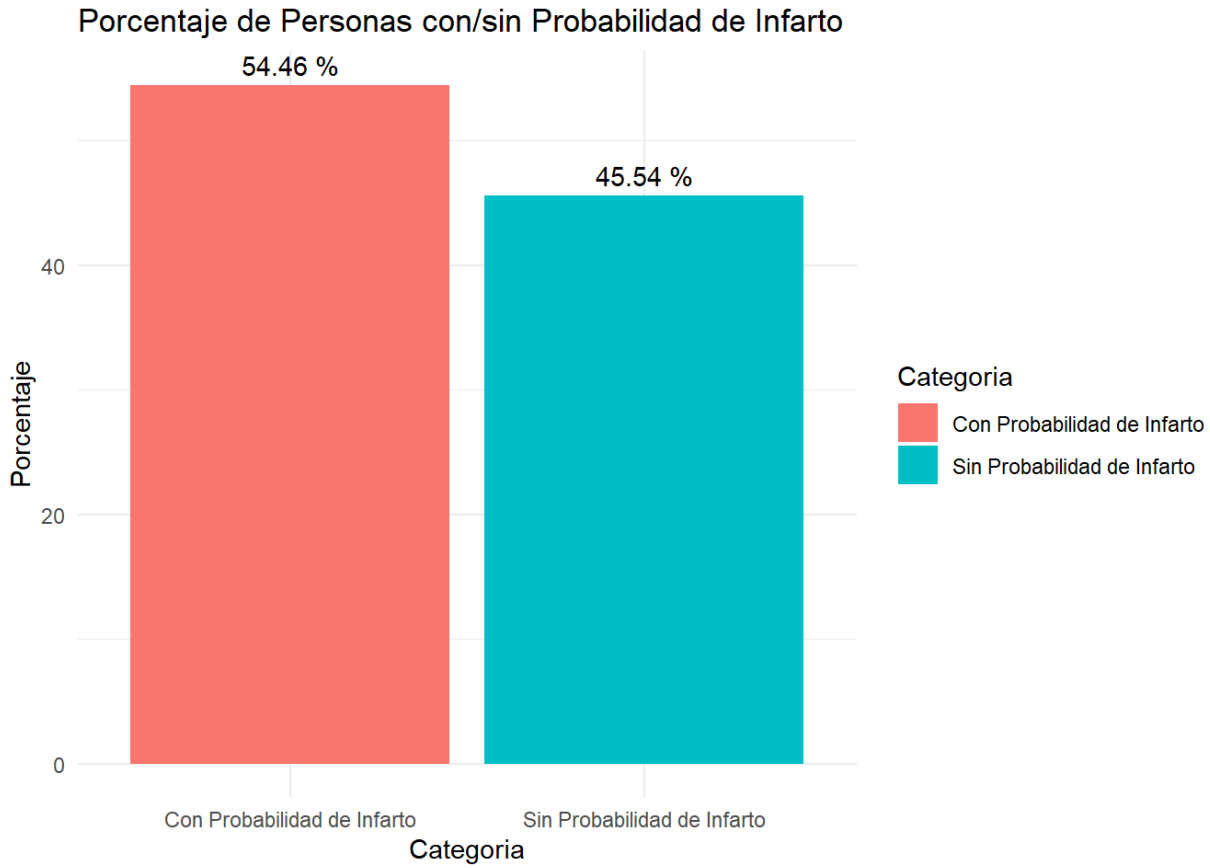
```
plot_intro(heart, title = "Información Dataset")
```



Tal y como vemos en el barplot creado, podemos ver la distirbución del conjunto como en la tabla anterior y también observamos como **no hay valores faltantes**.

Ahora vamos a visualizar la información básica del conjunto de datos en función de la variable 'output' de interés. La variable output nos va indicar quien tiene o no una mayor probabilidad de sufrir un ataque al corazón, por lo que primero calculamos el porcentaje de pacientes que tienen mayor probabilidad y luego el resto.

```
# Calcula los porcentajes
porcentaje_infarto <- round((sum(heart$output == 1)/nrow(heart)) * 100, 2)
porcentaje_sin_infarto <- round((sum(heart$output == 0)/nrow(heart))*100, 2)

# Crea un dataframe para usar en ggplot
df_porcentajes <- data.frame(
  Categoria = c("Con Probabilidad de Infarto", "Sin Probabilidad de Infarto"),
  Porcentaje = c(porcentaje_infarto, porcentaje_sin_infarto)
)

# Crea el gráfico de barras con porcentajes
ggplot(df_porcentajes, aes(x = Categoria, y = Porcentaje, fill = Categoria)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste(Porcentaje, "%")), vjust = -0.5) +  # Agrega etiquetas de porcentaje
  labs(title = "Porcentaje de Personas con/sin Probabilidad de Infarto") +
  theme_minimal()
```



Observamos que es un **conjunto de datos relativamente equilibrado**.

4.2 Valores Duplicados

Vamos a comprobar si existen valores duplicados en nuestro conjunto de datos y si existen los eliminaremos.

```
# Valores duplicados
get_duplicates <- function(heart){
    total_rows = dim(heart)[1]
    unique_rows = dim(heart %>% group_by_all %>% count)[1]
    n_duplicates = (total_rows - unique_rows)
    cat('n duplicates -> ', n_duplicates)
}

get_duplicates(heart) # Vemos que hay un valor duplicado
```

```
## n duplicates ->  1
```

```
heart = unique(heart)


cat('Eliminamos la fila duplicada')
```

```
## Eliminamos la fila duplicada
```

```
get_duplicates(heart)
```

```
## n duplicates ->  0
```

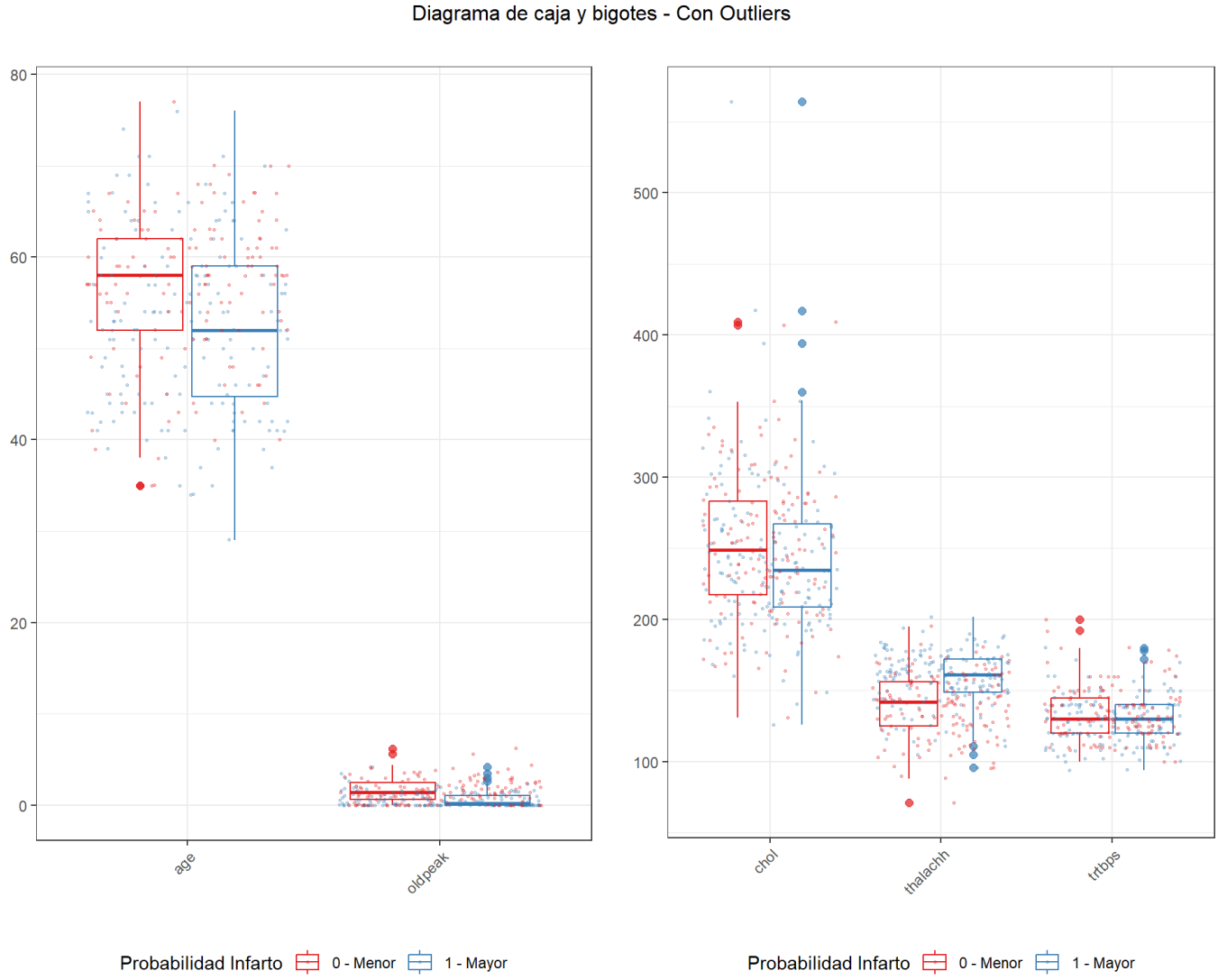
Ya no tenemos valores duplicados en el conjunto de datos. Vamos a estudiar ahora los valores extremos (*outliers*) y cómo afectan a nuestro conjunto de datos.

4.3 Valores Extremos

Seguidamente es importante estudiar la posibilidad de valores outliers para las variables numéricas del conjunto de datos.

```
generate_boxplot <- function(data, title) {
  # Gráfico 1
  selected_vars1 <- data[, c("age", "oldpeak")]
  selected_vars1 <- stack(selected_vars1)
  selected_vars1$output <- data$output

  plot1 <- ggplot(selected_vars1, aes(x = ind, y = values, color = as.factor(output))) +
    geom_boxplot(alpha = 0.7, outlier.size = 2) +
    geom_jitter(alpha = 0.3, size = 0.5) +
    theme_bw() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    labs(x = "", y = "") +
    scale_color_brewer(palette = "Set1", name = "Probabilidad Infarto", labels = c('0 - Menor', '1 - Mayor')) +
    ggtitle("") +
    theme(legend.position = "bottom",
          legend.direction = "horizontal",
          legend.box = "horizontal")

  # Gráfico 2
  selected_vars2 <- data[, c("chol", "thalachh", "trtbps")]
  selected_vars2 <- stack(selected_vars2)
  selected_vars2$output <- data$output

  plot2 <- ggplot(selected_vars2, aes(x = ind, y = values, color = as.factor(output))) +
    geom_boxplot(alpha = 0.7, outlier.size = 2) +
    geom_jitter(alpha = 0.3, size = 0.5) +
    theme_bw() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    labs(x = "", y = "") +
    scale_color_brewer(palette = "Set1", name = "Probabilidad Infarto", labels = c('0 - Menor', '1 - Mayor')) +
    ggtitle("") +
    theme(legend.position = "bottom",
          legend.direction = "horizontal",
          legend.box = "horizontal")

  # Organizar gráficos en una cuadrícula
  grid.arrange(plot1, plot2, ncol = 2, top = title)
}
```

```
generate_boxplot(heart, "Diagrama de caja y bigotes - Con Outliers")
```



Observaciones Podemos ver como hay puntos muy alejados de las medias y de los boxplots creados en las variables chol, thalachh , oldpeak y trtbps, por lo que, vamos a estudiar si mediante una función creada según las desviaciones estandard, se eliminan estos valores y dejamos el conjunto de datos sin potenciales valores outliers.

Para limpiar los valores atípicos (outliers) usamos un enfoque del rango intercuartílico (IQR), el cual es una medida de dispersión que se basa en la diferencia entre el tercer (Q3) y primer cuartil (Q1) del conjunto de datos.

```
# Creamos un dataframe de las variables de las cuales hemos de quitar outliers, excluyendo age y oldpeak
df_outliers<-as.data.frame(heart %>%
                  select("trtbps","thalachh","chol","oldpeak"))

# Función para quitar outliers
outliers <- function(x) {
 # IQR
  Q1 <- quantile(x, probs=.25)
  Q3 <- quantile(x, probs=.75)
  iqr = Q3-Q1

 # Rango Superior
 upper_limit = Q3 + (iqr*1.5)
 # Rango Inferior
 lower_limit = Q1 - (iqr*1.5)

 x > upper_limit | x < lower_limit
}

# Quitamos valores atípicos
remove_outliers <- function(df_outliers, cols = names(df_outliers)) {
  for (col in cols) {
    df_outliers<- df_outliers[!outliers(df_outliers[[col]]),]
  }
  df_outliers
}

# Una vez creada la función para quitar los outliers, creamos un nuevo conjutno de datos sin los outliers
heart_clean<-remove_outliers(heart,c("trtbps","thalachh", "chol"))

# Observamos el cambio de las dimensiones
dim_clean<- dim(heart_clean)
```

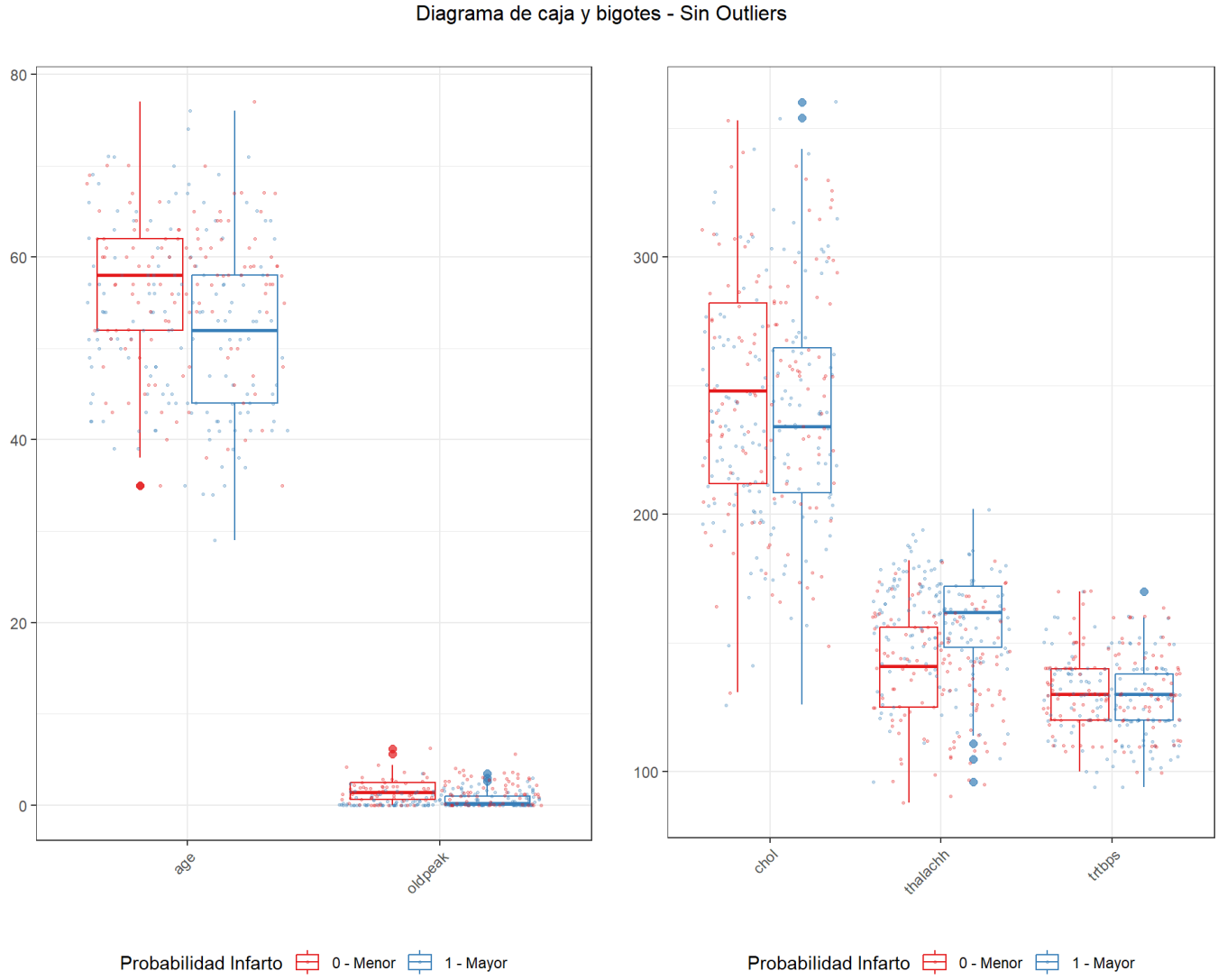
Los valores potenciales outliers han sido eliminados y ahora tenemos nuestro **conjunto de datos limpio** con un total de **287 filas** y **14 variables**.

Guardamos nuestro fichero con la limpieza realizada.

```
write.csv(heart_clean, file = "heart_clean.csv")
```

Resultado de Limpieza de Outliers

```
generate_boxplot(heart_clean, "Diagrama de caja y bigotes - Sin Outliers")
```



4.4 Comprobación de la normalidad y homogeneidad de la varianza

Normalizamos los datos

La normalización típica se realiza para que los datos estén en una escala común y no dominada por variables con rangos mucho mayores que otras. Una técnica común es la normalización estándar, que escala los datos para que tengan una media de 0 y una desviación estándar de 1. A continuación normalizamos nuestros datos para poder hacer análisis posteriores.

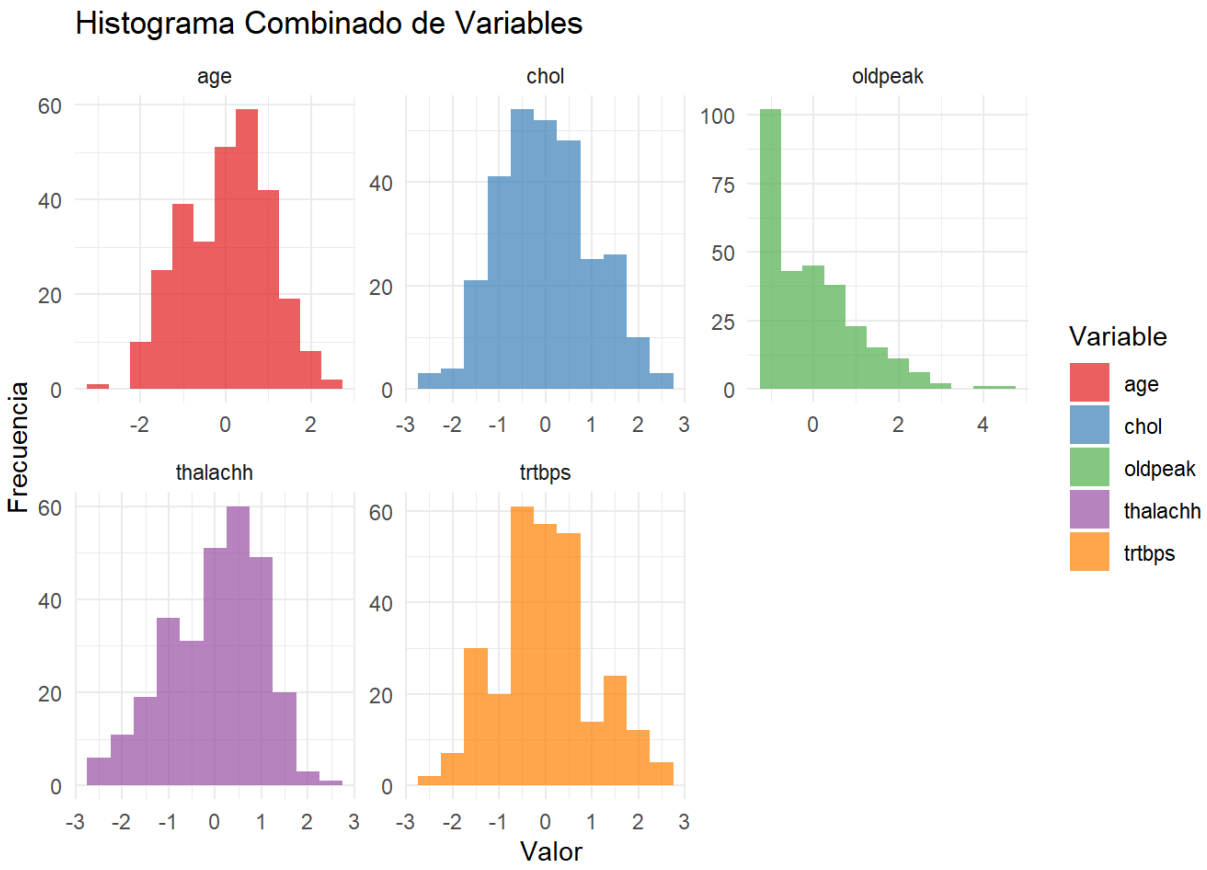
```
# Seleccionar solo las columnas numéricas para normalizar
# heart_clean_numeric <- heart_clean[sapply(heart_clean, is.numeric)]
heart_clean_numeric <- sapply(heart_clean, as.numeric)


# Normalizar utilizando la función scale del paquete base
heart_norm <- as.data.frame(scale(heart_clean_numeric))

# Agregar la columna 'output' al conjunto de datos normalizado
heart_norm$output <- heart_clean$output
head(heart_norm)
```

```
##          age        sex          cp        trtbps       chol        fbs
## 1  0.9818481  0.6529701  1.96764460  0.9716159546 -0.2093614  2.4806218
## 2 -1.8812042  0.6529701  0.99734563 -0.0002257472  0.1688223 -0.4017201
## 3 -1.4407346 -1.5261277  0.02704666 -0.0002257472 -0.8544982 -0.4017201
## 4  0.2110264  0.6529701  0.02704666 -0.6481202150 -0.1426231 -0.4017201
## 5  0.3211438 -1.5261277 -0.94325231 -0.6481202150  2.4824166 -0.4017201
## 6  0.3211438  0.6529701 -0.94325231  0.6476687206 -1.1214514 -0.4017201
##      restecg    thalachh       exng    oldpeak        slp        caa      thall
## 1 -1.0174164  0.01426375 -0.6856756  1.1378906 -2.3018848 -0.7126315 -2.1126920
## 2  0.9036264  1.64293866 -0.6856756  2.1961935 -2.3018848 -0.7126315 -0.4871086
## 3 -1.0174164  0.98266505 -0.6856756  0.3441635  0.9605396 -0.7126315 -0.4871086
## 4  0.9036264  1.24677449 -0.6856756 -0.1849879  0.9605396 -0.7126315 -0.4871086
## 5  0.9036264  0.58650088  1.4533341 -0.3613717  0.9605396 -0.7126315 -0.4871086
## 6  0.9036264 -0.07377273 -0.6856756 -0.5377555 -0.6706726 -0.7126315 -2.1126920
##   output
## 1      1
## 2      1
## 3      1
## 4      1
## 5      1
## 6      1
```

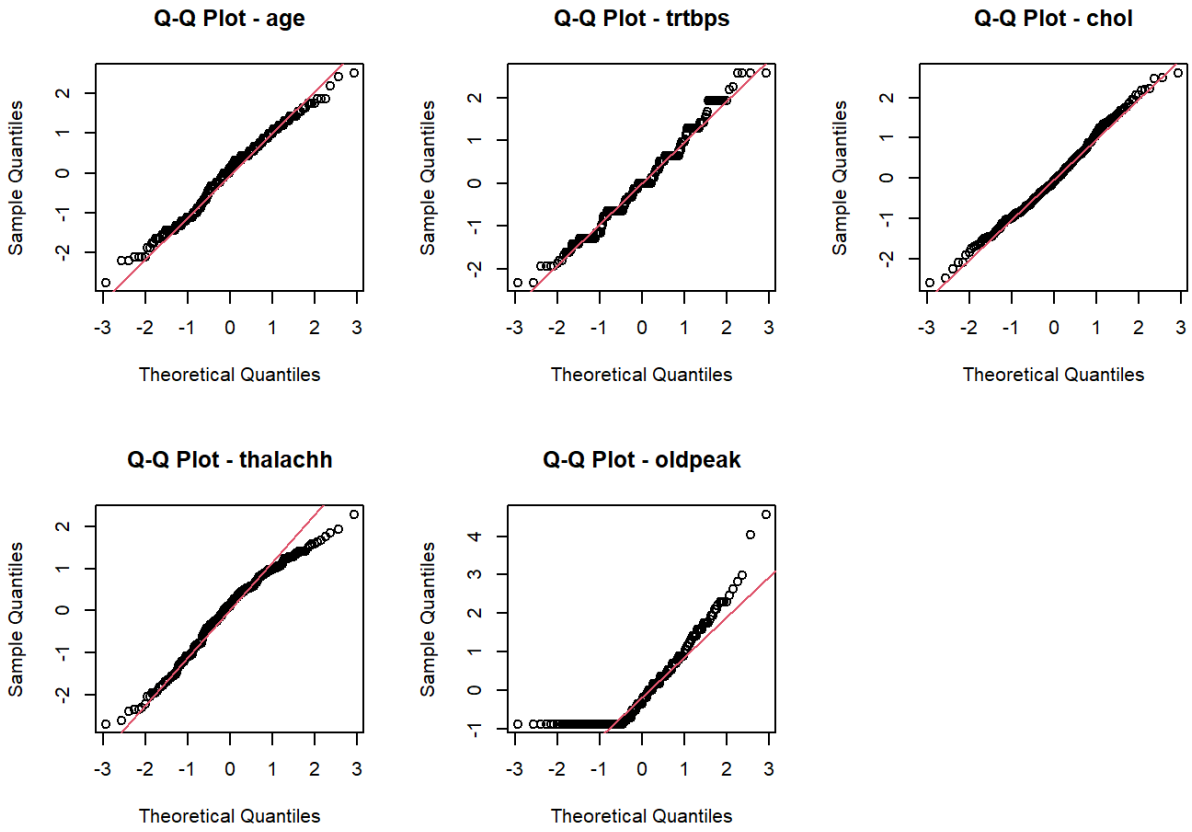
4.4.1 Comprobación de la normalidad

Vamos a comprobar gráficamente la normalidad de mis variables mediante histogramas y gráficos Q-Q. Los **histogramas** brindan una visualización de la distribución de tus datos y pueden ayudarte a identificar patrones, simetrías o sesgos en las distribuciones. El **gráfico Q-Q** (quantile-quantile) es una herramienta gráfica que compara la distribución de nuestros datos con la distribución teórica que estamos probando, en este caso, la distribución normal.

```
# Seleccionar las variables de interés
variables_interes <- c("age", "trtbps", "chol", "thalachh", "oldpeak")

# Crear un nuevo dataframe con las variables seleccionadas
df_variables <- heart_norm[, variables_interes]

# Convertir el dataframe a formato largo (tidy) para ggplot
df_long <- tidyr::gather(df_variables, key = "Variable", value = "Valor")

# Crear el histograma combinado
ggplot(df_long, aes(x = Valor, fill = Variable)) +
  geom_histogram(binwidth = 0.5, position = "identity", alpha = 0.7) +
  facet_wrap(~ Variable, scales = "free") +
  theme_minimal() +
  labs(title = "Histograma Combinado de Variables", x = "Valor", y = "Frecuencia") +
  scale_fill_brewer(palette = "Set1")
```




```
# Selecciona solo las variables numéricas
variables_numericas <- heart_norm[, variables_interes]

# Gráficos Q-Q para cada variable
par(mfrow = c(2, 3))  # Ajusta el diseño para mostrar varios gráficos

for (variable in colnames(variables_numericas)) {
  qqnorm(variables_numericas[[variable]], main = paste("Q-Q Plot -", variable))
  qqline(variables_numericas[[variable]], col = 2)
}

par(mfrow = c(1, 1))  # Restablece el diseño a uno solo
```



Una vez realizados ya podemos observar de manera gráfica la normalidad de nuestros datos. Esto no son pruebas formales pero proporcionan una indicación visual. Por ejemplo, en el gráfico Q-Q si todos los puntos están cerca de la línea diagonal, sugiere que tus datos siguen una distribución normal. Y si los puntos se desvían de la línea en los extremos, podría indicar colas pesadas o ligeros sesgos en tus datos.

Para evaluar la normalidad de las variables seleccionadas mediante un prueba estadística formal empleamos la prueba de **Shapiro-Wilk**.

```
shapiro_test_results <- sapply(heart_norm[variables_interes], shapiro.test)
print(shapiro_test_results[c("statistic", "p.value"), ])
```

```
##           age        trtbps      chol      thalachh    oldpeak
## statistic 0.9880388  0.9840512   0.9928493 0.9755678   0.8417848
## p.value   0.01789985 0.002776064 0.1856207 8.09197e-05 1.853554e-16
```

Los resultados de la normalidad muestran que para cada una de las variables numéricas del conjunto de datos, **excepto la variable chol**, hay evidencia suficiente para rechazar la hipótesis nula y afirmar que **los datos no siguen una distribución normal**. De todas maneras, con conjuntos de datos grandes, **se puede asumir normalidad** para fines prácticos, incluso si las pruebas de normalidad dan como resultado p-values bajos.

4.4.2 Comprobación de la varianza

Calculamos las varianzas de las variables numéricas ‘age’, ‘trtbps’, ‘chol’, ‘thalachh’, y ‘oldpeak’, del conjunto, agrupadas según la probabilidad de sufrir o no un infarto.

```r
# Calculamos la varianza para cada variable según los niveles de 'output'
heart_var <- heart_norm

calcular_varianza <- function(data, output_category) {
  subset_data <- subset(data, output == output_category)
  subset_data <- subset_data[, c("age", "trtbps", "chol", "thalachh", "oldpeak")]
  return(diag(var(subset_data)))
}

# Calcular varianzas para output = 0
varianzas_output_0 <- calcular_varianza(heart_var, output_category = 0)

# Calcular varianzas para output = 1
varianzas_output_1 <- calcular_varianza(heart_var, output_category = 1)

# Combina los resultados en un dataframe
resultados_varianzas <- data.frame(
  Varianza_Output_0 = varianzas_output_0,
  Varianza_Output_1 = varianzas_output_1
)

# Visualizar los resultados
resultados_varianzas
```

```
##          Varianza_Output_0 Varianza_Output_1
## age              0.7855903         1.0895781
## trtbps           1.0385980         0.9477493
## chol             1.0628628         0.9347773
## thalachh         0.9162405         0.7388003
## oldpeak          1.2811090         0.4191531
```

Estos resultados muestran la media de cada variable numérica (normalizada) para dos grupos distintos definidos por los niveles '0' y '1' de la variable 'output', los cuales indican menor o mayor probabilidad de ataque cardíaco. Podemos observar que las varianzas son diferentes entre los dos grupos para cada variable. Esto podría indicar falta de homogeneidad de varianzas entre los grupos. Sin embargo, para obtener una conclusión más precisa sobre la homogeneidad de varianzas, sería mejor realizar pruebas específicas, como la prueba de Levene.

Por tanto, para determinar si hay diferencias significativas en las varianzas entre las dos categorías, podrías realizar un test de homogeneidad de varianza, como el **test de Levene** o el **test de Bartlett**. Estos tests evalúan la hipótesis nula de que las varianzas en los diferentes grupos son iguales. En este caso particular vamos a utilizar el test de Bartlett que es menos sensible a la normalidad de los datos.

```r
perform_bartlett_test <- function(data, variables_interes, group_var) {
  results <- list()

  for (var in variables_interes) {
    test_result <- bartlett.test(data[[var]] ~ data[[group_var]])
    results[[var]] <- test_result
  }

  return(results)
}

# Uso de la función
bartlett_test_results <- perform_bartlett_test(heart_norm, variables_interes, "output")

# Mostrar resultados
for (var in names(bartlett_test_results)) {
  cat(paste("Bartlett Test for", var, "\n"))
  print(bartlett_test_results[[var]])
  cat("\n")
}
```

```
## Bartlett Test for age
##
##  Bartlett test of homogeneity of variances
##
## data:  data[[var]] by data[[group_var]]
## Bartlett's K-squared = 3.7017, df = 1, p-value = 0.05436
##
##
## Bartlett Test for trtbps
##
##  Bartlett test of homogeneity of variances
##
## data:  data[[var]] by data[[group_var]]
## Bartlett's K-squared = 0.29518, df = 1, p-value = 0.5869
##
##
## Bartlett Test for chol
##
##  Bartlett test of homogeneity of variances
##
## data:  data[[var]] by data[[group_var]]
## Bartlett's K-squared = 0.58144, df = 1, p-value = 0.4457
##
##
## Bartlett Test for thalachh
##
##  Bartlett test of homogeneity of variances
##
## data:  data[[var]] by data[[group_var]]
## Bartlett's K-squared = 1.6365, df = 1, p-value = 0.2008
##
##
## Bartlett Test for oldpeak
##
##  Bartlett test of homogeneity of variances
##
## data:  data[[var]] by data[[group_var]]
## Bartlett's K-squared = 43.283, df = 1, p-value = 4.736e-11
```

Si el valor p es menor que el nivel de significancia (0.05 es comúnmente utilizado), podemos rechazar la hipótesis nula de igualdad de varianzas. En nuestros resultados, el grupo **oldpeak** tienen un **valor p significativamente bajo**, lo que sugiere que las **varianzas son diferentes**.

Si el valor p es mayor que el nivel de significancia, no hay suficiente evidencia para rechazar la hipótesis nula. Por tanto podemos asumir homogeneidad de varianzas en nuestras variables **age**, **trtbps**, **chol** y **thalachh**.

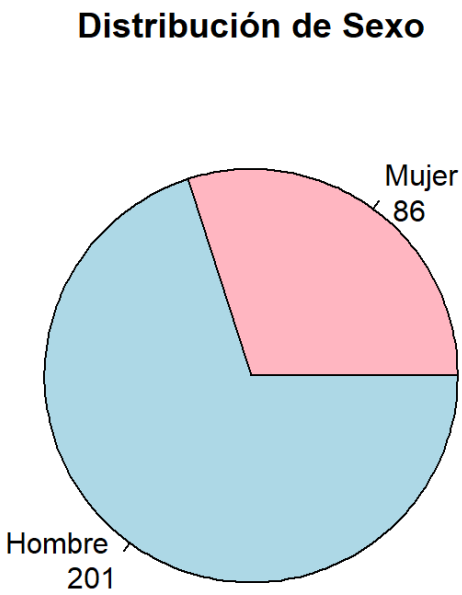
Estadístico: Indica el tamaño del efecto. Un valor más alto sugiere una mayor diferencia en las varianzas. Por ejemplo, el grupo “oldpeak” tiene un estadístico K-squared significativamente alto.

5 Análisis de Datos

5.1 Análisis Univariante

Comprobamos los datos que tenemos para cada uno de los sexos.

```
# Creación de la tabla de frecuencias para la variable "sex"
sex_frequency <- table(heart_clean$sex)
# Configuración de colores para el piechart
colors <- c("#FFB6C1", "lightblue")
# Creación del piechart
pie(sex_frequency, col = colors, main = "Distribución de Sexo", labels = paste0(c("Mujer ", "Hombre "), "\n ",
sex_frequency))
```

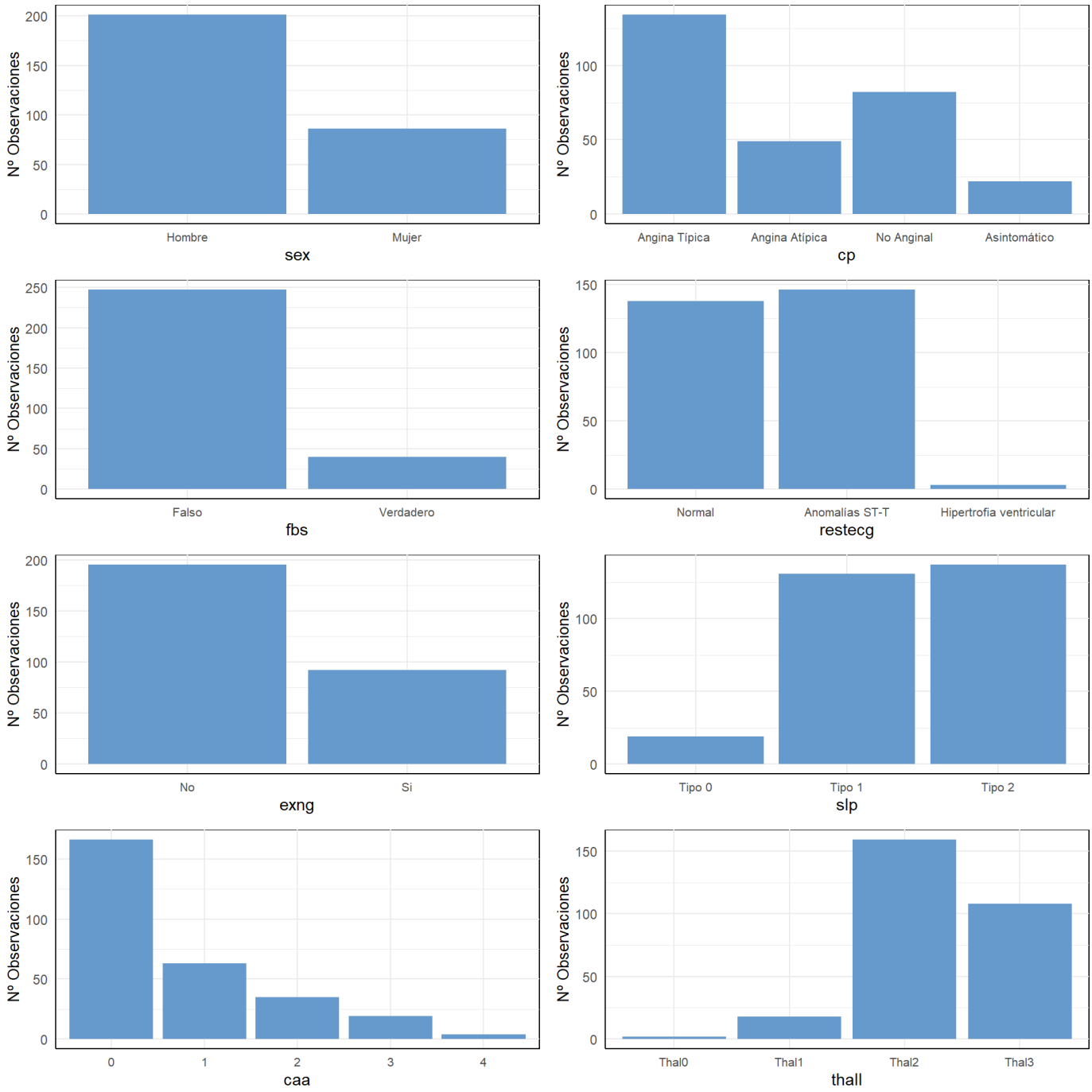


La distribución de género es desigual en tu conjunto de datos, ya que hay más observaciones para “Hombre” (201) que para “Mujer” (86).

Estudiamos un poco más nuestras variables categóricas. Primero discretizaremos las variables categóricas, asignando a cada valor la correspondiente definición de la variable.

```r
# Hacemos cópia del conjunto para usarlo solamente en este análisis:
heart_discr<-heart_clean

# Sexo del paciente (sex)
heart_discr$sex <- ifelse(heart_discr$sex == 0, "Mujer", "Hombre")

# Dolor Torácico (cp)
heart_discr$cp <- factor(heart_discr$cp, levels = c(0, 1, 2, 3), labels = c("Angina Típica", "Angina Atípica",
"No Anginal", "Asintomático"))

# Resultados del Electrocardiograma en Reposo (restecg)
heart_discr$restecg <- factor(heart_discr$restecg, levels = c(0, 1, 2), labels = c("Normal", "Anomalías ST-T",
"Hipertrofia ventricular"))

# Angina Inducida por Esfuerzo (exng)
heart_discr$exng <- ifelse(heart_discr$exng == 1, "Si", "No")

# Número de Buques Principales (caa) (0-3)
heart_discr$caa <- as.character(heart_discr$caa)

# Glucemia en Ayunas (fbs)
heart_discr$fbs <- ifelse(heart_discr$fbs == 1, "Verdadero", "Falso")

# Pendiente del Segmento ST Máximo del Ejercicio (slp)
heart_discr$slp <- factor(heart_discr$slp, levels = c(0, 1, 2), labels = c("Tipo 0", "Tipo 1", "Tipo 2"))

# Tasa de Mortalidad (thall)
heart_discr$thall <- factor(heart_discr$thall, levels = c(0, 1, 2, 3), labels = c("Thal0", "Thal1", "Thal2", "T
hal3"))



# Ouput/Target (0= menor probabilidad de infarto 1= mayor probabilidad de infarto)
heart_discr$output <- factor(heart_discr$output, levels = c(0, 1), labels = c("Menor probabilidad", "Mayor prob
abilidad"))
```

```r
categorical_var <- c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")
plots <- list()

for (i in categorical_var) {
  plot_data <- as.data.frame(table(heart_discr[[i]]))
  colnames(plot_data) <- c(i, "Freq")

  plot <- ggplot(plot_data, aes_string(x = i, y = "Freq")) +
    geom_bar(stat = "identity", fill = "#6699CC") +
    labs(x = i, y = "Nº Observaciones") +
    theme_minimal() +
    theme(panel.background = element_rect(fill = "white"),
          axis.line = element_line(color = "black"),
          axis.title.y = element_text(size = 10),  # Ajustar tamaño del texto del eje Y
          axis.text.x = element_text(size = 8, angle = 0, hjust = 0.5))  # Ajustar tamaño del texto del eje X

  plots[[i]] <- plot
}

grid.arrange(grobs = plots, ncol = 2)
```



Observamos que el dolor torácico (**cp**) más común en nuestros datos es el de Angina típica (TA) seguido del caso No Anginal.

La mayoría de casos de Glucemia en ayunas (**fbs**) presentan Glucemia en ayunas <= 120 mg/dl, es decir, tienen el valor Falso.

Los valores predominantes de los resultados del electrocardiograma en reposo (**restecg**) son el valor 0 y 1 que corresponden a “Normal” y “Anomalías de onda ST-T”. Casi no hay casos de Hipertrofia ventricular.

Existen el doble de casos aproximadamente donde la Angina no ha sido inducida por esfuerzo (valor 0 de **exng**).

Los datos de la pendiente del segmento ST máximo del ejercicio (**slp**) nos hace ver que son predominantes los caso de Tipo 1 y 2.

EL número de observaciones del número de vasos principales (**caa**) es mucho mayor para el valor 0 y va disminuyendo conforme el número de vasos principales incrementa. Con el valor 4 casi no hay observaciones.

Por último, en la variable que indica la tasa de mortalidad (**thall**) observamos muchas más muestras para los casos 2 y 3.

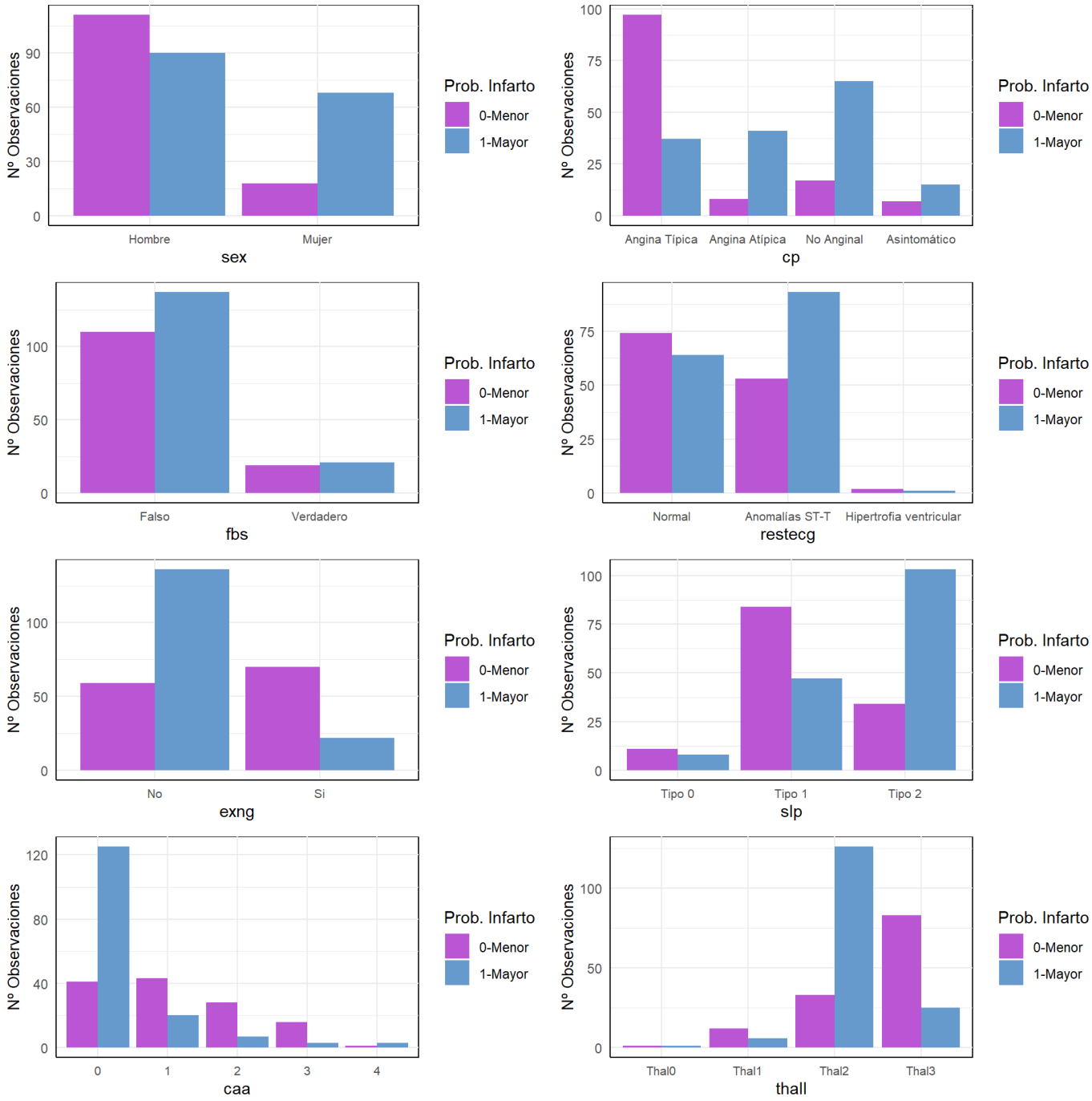
5.2 Análisis Bivariante

Vamos a completar nuestro estudio con un análisis Bivariante de las variables categóricas y numéricas.

5.2.1 Variables Categóricas

A continuación estudiamos la estadística básica de las variables categóricas respecto a output del conjunto heart.

```
categorical_var <- c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")
plots <- list()


for (i in categorical_var) {
  plot_data <- as.data.frame(table(heart_discr[[i]], heart_discr$output))
  colnames(plot_data) <- c(i, "output", "Freq")

  plot <- ggplot(plot_data, aes_string(x = i, y = "Freq", fill = "output")) +
    geom_bar(stat = "identity", position = position_dodge()) +
    scale_fill_manual(values = c("#BA55D3", "#6699CC"),
                      name = "Prob. Infarto",
                      labels = c('0-Menor', '1-Mayor')) +
    labs(x = i, y = "Nº Observaciones") +
    theme_minimal() +
    theme(panel.background = element_rect(fill = "white"),
          axis.line = element_line(color = "black"),
          axis.title.y = element_text(size = 10),  # Ajustar tamaño del texto del eje Y
          axis.text.x = element_text(size = 8, angle=0, hjust=0.5), # Ajustar tamaño del texto del eje X
          legend.title = element_text(size = 10) +
          theme(legend.position = c(0.8, 0.8)))   # Ajustar tamaño del texto de la leyenda

  # print(plot)
  plots[[i]] <- plot
}


grid.arrange(grobs = plots, ncol = 2)
```



Observaciones

Según la variable **thall**, el riesgo de infarto se alcanza en las personas con frecuencia cardíaca máxima(clase 2).

En la característica **sexo**, la clase 1 (Hombre) tiene más posibilidades de sufrir un infarto que la clase 0 (Mujer). Las probabilidades de sufrir un infarto son mayores en la clase 0 de sexo.

Comparando con el análisis de correlación, la característica **fbs** muestra la menor correlación con la salida.

En **ca**, las personas con clase 0 y 1 son más propensas a sufrir un ataque al corazón que las personas con clase 4, 3 y 2.

Según la característica **cp**, las personas con dolor no anginoso tienen más probabilidades de sufrir un infarto que las personas con dolor anginoso atípico y típico.

En **exng**, las personas con clase 1 (tienen Angina inducida por el esfuerzo) tienen altas probabilidades de riesgo de infarto, mientras que las personas con clase 0 (no tienen Angina inducida por el esfuerzo) son menos propensas al infarto.

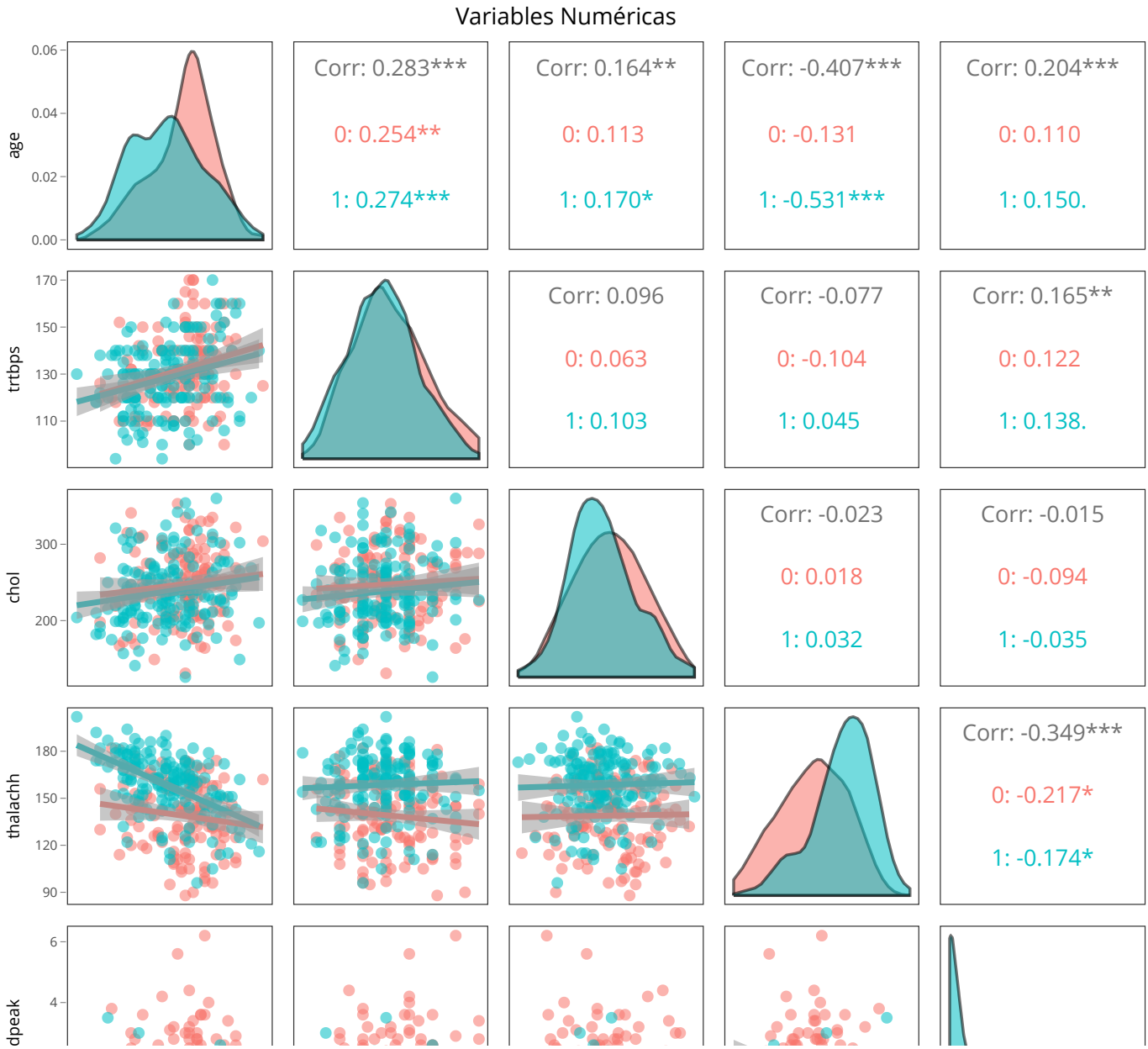
La característica **slp** muestra que la clase 0 tiene menos correlación con el resultado que las clases 1 y 2.

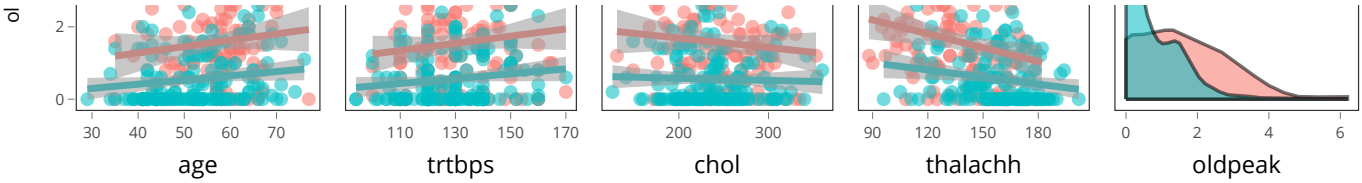
5.2.2 Variables Numéricas

A continuación vamos a observar la relación entre las variables y buscar patrones, tendencias o relaciones lineales de las variables numéricas del conjunto de datos. Al utilizar colores diferentes para cada clase de salida, podemos identificar visualmente cómo la distribución de las variables numéricas difiere entre las dos clases. Esto es especialmente útil para

comprender si hay diferencias notables en la relación entre las variables y la variable de salida, lo que puede tener implicaciones importantes en el análisis y modelado posterior.

```
# Creamos un gráfico de pares con las variables numéricas del conjunto de datos "heart"
# Establecer opciones para el tamaño del gráfico
options(repr.plot.width = 20, repr.plot.height = 20)

# Creamos un gráfico de pares con las variables numéricas del conjunto de datos "heart"
pair_plot <- ggpairs(heart_clean, columns = c("age", "trtbps", "chol", "thalachh", "oldpeak"),
                     aes(color = as.factor(output), alpha = 0.5),
                     lower = list(continuous = "smooth"),
                     palette = c('blue', 'red')) +  # Usamos la misma paleta de colores
  theme_bw() +
  theme(text = element_text(size = 8),
        panel.grid = element_blank(),
        legend.position = "right",
        legend.title = element_text(face = "bold")) +
  ggtitle("Variables Numéricas") +
  labs(color = "Output", alpha = "Transparencia")


# Convertimos el gráfico a un gráfico interactivo con plotly
interactive_plot <- ggplotly(pair_plot)

# Mostramos el gráfico interactivo
interactive_plot
```





Por un lado, el gráfico anterior viene a corroborar lo visto en el gráfico de correlaciones en apartados anteriores observándose la correlación negativa moderada entre la edad (**age**) y la variable **thalachh**. Podemos observar como esta correlación es algo más grande en los casos que existen mayor probabilidad de infarto.

5.3 Análisis Multivariado

5.3.1 Modelo No Supervisado: Clustering

Un modelo no supervisado de clustering busca agrupar datos similares en conjuntos o clústeres. En este caso, la distancia Manhattan es una medida de distancia utilizada en clustering para calcular la diferencia entre dos puntos en un espacio multidimensional.

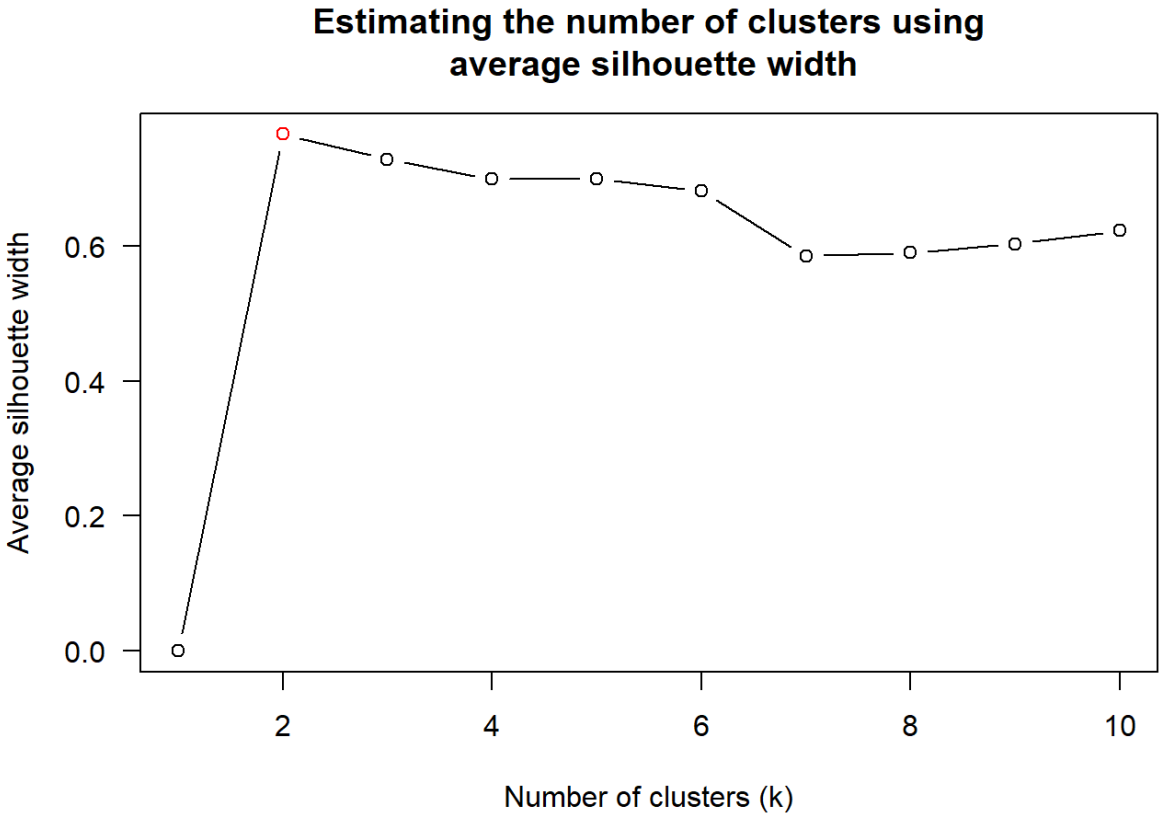
En este caso, puede ser útil para la identificación de subgrupos de pacientes con perfiles de riesgo similares de enfermedad cardíaca, lo que podría ser útil para personalizar tratamientos, identificar factores de riesgo comunes o incluso guiar futuras investigaciones médicas.

Para poder aplicar el logaritmo, escalamos los datos y mostramos el dendograma para saber donde determinar el corte y escoger el número de clústers.

```
heart_norm <- as.data.frame(sapply(heart_norm, as.numeric))

# Escalamos los datos
heart_scale<-scale(heart_norm)

#Calculamos la distancia manhattan
dist_manh<-dist(heart_scale, method = 'manhattan')

#A continuación usamos la variable creada para minimizar las diferencias dentro de los conglomerados mediante e
l método Ward
hcluster<-hclust(dist_manh, method = "ward.D")
hcluster
```

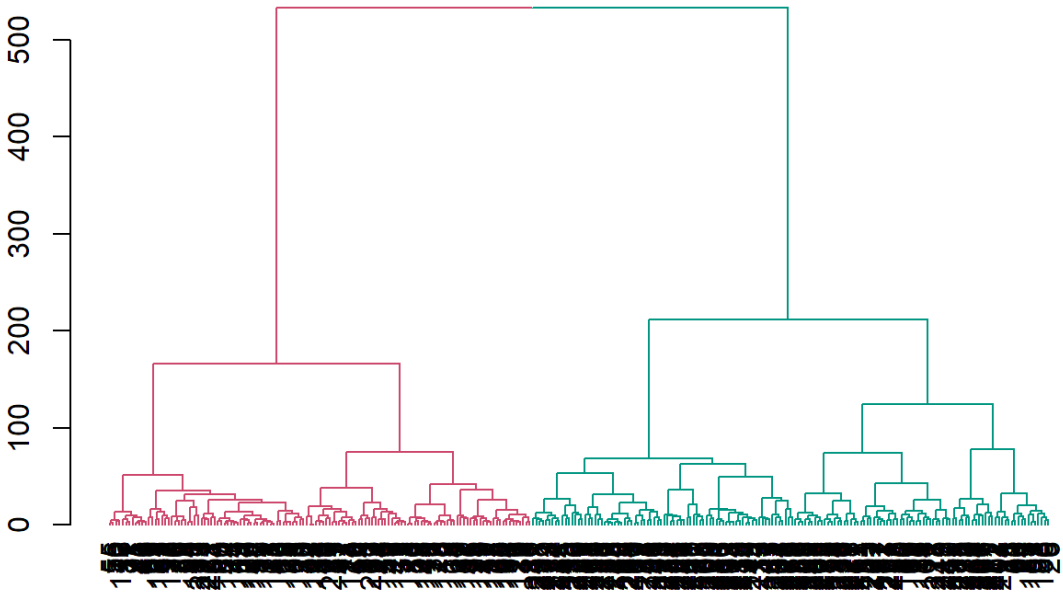
```
##
## Call:
## hclust(d = dist_manh, method = "ward.D")
##
## Cluster method   : ward.D
## Distance         : manhattan
## Number of objects: 287
```

```
dendo<-as.dendrogram(hcluster)
dendo_k<-find_k(dendo)
plot(dendo_k)
```



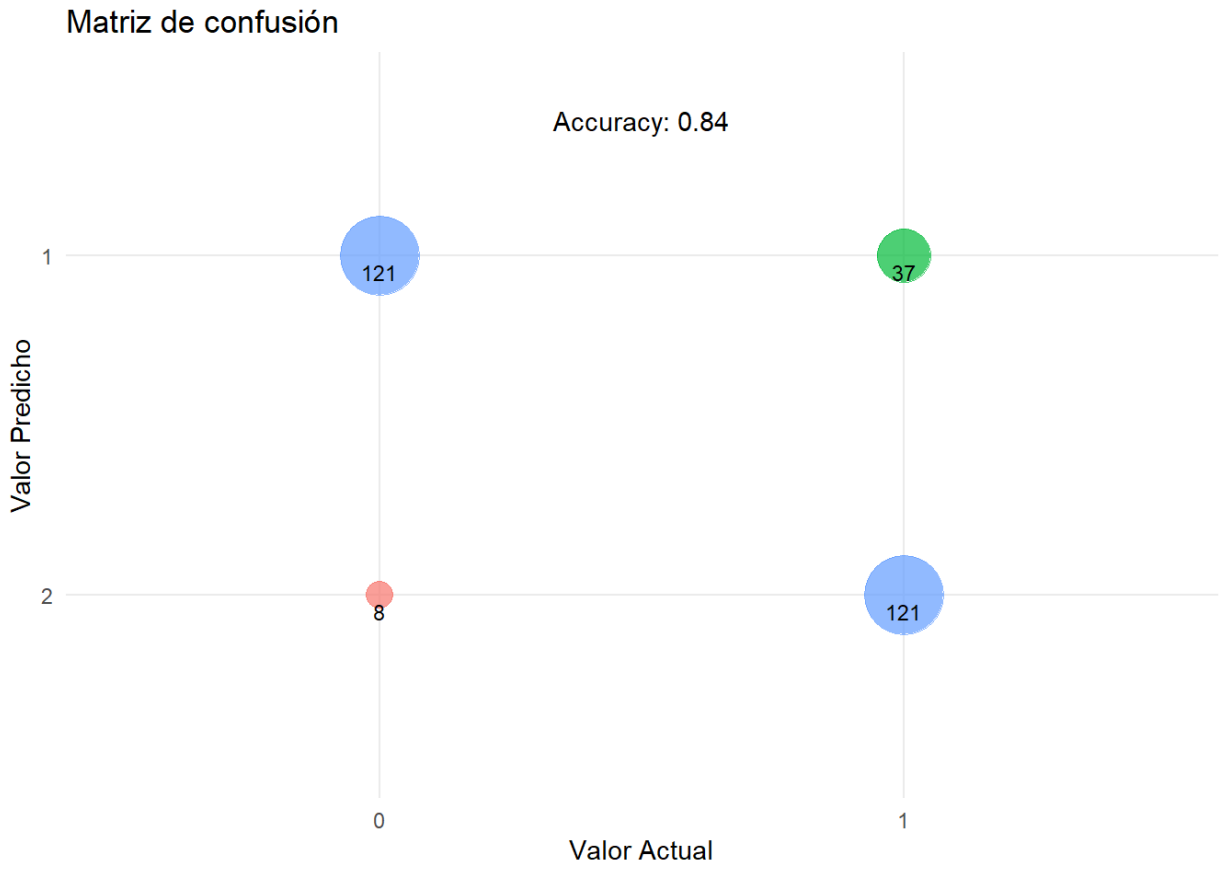
Observamos que el número de clusters óptimo es de 2. Vamos a dibujar nuestro dendograma.

```
plot(color_branches(dendo,k=dendo_k$nc))
```



Creamos la matriz de confusión.

```
grupos<-cutree(hcluster,k=dendo_k$nc)
# table(grupos,heart_clean$output)
conf_matrix <- table(grupos, heart_clean$output)
# Calcula la precisión global
precision_kmeans <- sum(diag(conf_matrix)) / sum(conf_matrix)

# Crea un data frame
conf_matrix_df <- as.data.frame(as.table(conf_matrix))
colnames(conf_matrix_df) <- c("Predicted", "Actual", "Count")

# Ajusta las etiquetas si es necesario
conf_matrix_df$Actual <- factor(conf_matrix_df$Actual, levels = c("0", "1"))
conf_matrix_df$Predicted <- factor(conf_matrix_df$Predicted, levels = c("2", "1"))


# Crea el gráfico de dispersión con ggplot2
ggplot(conf_matrix_df, aes(x = Actual, y = Predicted, size = Count, color = as.factor(Count))) +
  geom_point(alpha = 0.7) +
  geom_text(aes(label = Count), vjust = 1.5, color = "black", size = 3) +
  scale_size_continuous(range = c(5, 15)) +
  labs(title = "Matriz de confusión",
       x = "Valor Actual",
       y = "Valor Predicho") +
  theme_minimal() +
  theme(legend.position = "none") +
  annotate("text", x = 1.5, y = 2.4, label = paste("Accuracy:", round(precision_kmeans, 2)))
```



Mediante la matriz de confusión creada, extraeremos el número de casos que se han clasificado correctamente y el porcentaje de precisión del modelo

```
cat("Número de observaciones clasificadas correctamente =", sum(diag(conf_matrix)), "\n")
```

```
## Número de observaciones clasificadas correctamente = 242
```

```
cat("Número de observaciones clasificadas incorrectamente =", sum(conf_matrix) - sum(diag(conf_matrix)), "\n")
```

```
## Número de observaciones clasificadas incorrectamente = 45
```

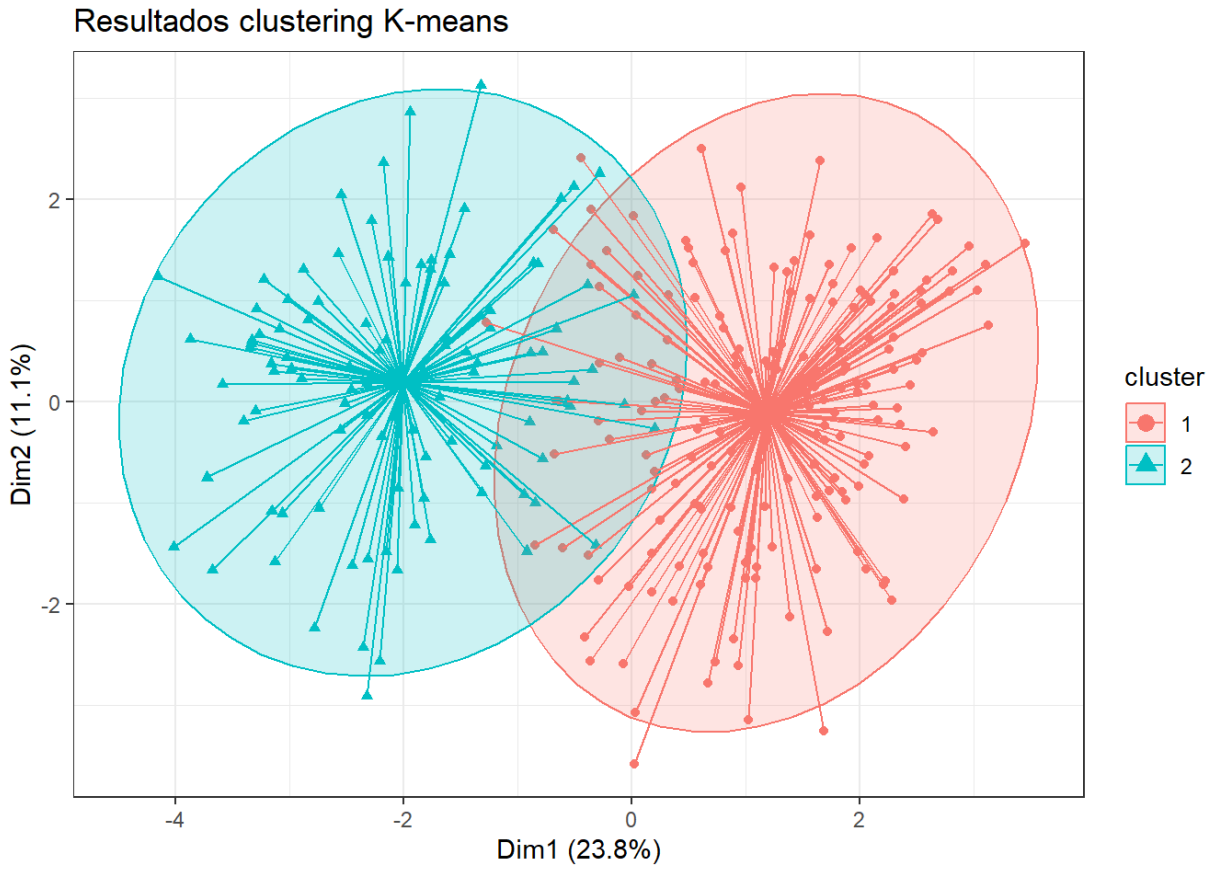
```
cat("Precisión del modelo:", sprintf("%.2f%%", precision_kmeans * 100), "\n")
```

```
## Precisión del modelo: 84.32%
```

Con una **precisión del modelo del 84.32%**, se puede decir que este modelo de clasificación logra predecir correctamente un elevado número de los casos en el conjunto de datos evaluado. Esto indica una capacidad razonablemente buena para predecir la enfermedad cardíaca en función de las características utilizadas en el modelo. Sin embargo, en un futuro también sería valioso evaluar otras métricas de rendimiento para obtener una comprensión más completa de su eficacia, como la sensibilidad, la especificidad u otras métricas según el contexto médico específico.

A continuación, visualizamos el cluster con la función fviz_cluster.

```
pam.res <- pam(heart_scale, dendo_k$nc)
#pam.res <- pam(heart_norm, 2)

fviz_cluster(pam.res, geom = "point", ellipse.type = "norm",
             show.clust.cent = TRUE,star.plot = TRUE)+
  labs(title = "Resultados clustering K-means")+ theme_bw()
```



En este gráfico observamos que cada grupo obtenido del algoritmo K-means se representa de un color diferente. Podemos ver líneas que conectan los puntos con sus respectivos centroides. Estas líneas nos dan una idea de la distancia entre los puntos y los centroides, lo que refleja cómo se han agrupado los datos.

5.3.2 Modelo Supervisado: Regresión Logística

Este análisis busca entender cómo diferentes variables pueden influir en una variable de salida específica. En este caso, se trata de predecir un cierto resultado,es decir, si alguien tiene cierta enfermedad cardíaca o no (ouput).

```
# Codificamos la variable objetivo como factor
heart_clean$output <- factor(heart_clean$output)

heart_clean <- as.data.frame(sapply(heart_clean, as.numeric))

# Reemplazamos los valores en la columna "output"
heart_clean$output[heart_clean$output == 1] <- 0
heart_clean$output[heart_clean$output == 2] <- 1

# Dividimos el conjunto de datos en conjunto de entrenamiento y conjunto de prueba
set.seed(123)
split = sample.split(heart_clean$output, SplitRatio = 0.75)
training_set = subset(heart_clean, split == TRUE)
test_set = subset(heart_clean, split == FALSE)

# Escalamos de características
# training_set[-14] = scale(training_set[-14])
training_set[, -which(names(training_set) == 'output')] <- scale(training_set[, -which(names(training_set) ==
'output')])

#test_set[-14] = scale(test_set[-14])
test_set[, -which(names(test_set) == 'output')] <- scale(test_set[, -which(names(test_set) == 'output')])


# Ajustamos la regresión logística al conjunto de entrenamiento
classifier = glm(formula = output ~ .,
                 family = binomial,
                 data = training_set)
# Mostrarnos resultados
summary(classifier)
```

```
##
## Call:
## glm(formula = output ~ ., family = binomial, data = training_set)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.196009   0.217916   0.899 0.368402
## age          0.028971   0.266135   0.109 0.913314
## sex         -0.844252   0.262799  -3.213 0.001316 **
## cp           0.784661   0.228264   3.438 0.000587 ***
## trtbps      -0.213332   0.228084  -0.935 0.349623
## chol        -0.455822   0.252511  -1.805 0.071050 .
## fbs          0.009286   0.217132   0.043 0.965889
## restecg      0.393506   0.213860   1.840 0.065766 .
## thalachh     0.532906   0.292425   1.822 0.068399 .
## exng        -0.287279   0.238710  -1.203 0.228796
## oldpeak     -0.726294   0.302640  -2.400 0.016401 *
## slp          0.494799   0.255935   1.933 0.053199 .
## caa         -0.666232   0.237150  -2.809 0.004965 **
## thall       -0.549061   0.211196  -2.600 0.009328 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 296.00  on 214  degrees of freedom
## Residual deviance: 148.91  on 201  degrees of freedom
## AIC: 176.91
##
## Number of Fisher Scoring iterations: 6
```

Una vez generado el modelo de regresión logística vamos a predicir los resultados y a calcular la matriz de confusión.

```
# Predecimos los resultados del conjunto de prueba
prob_pred_glm = predict(classifier, type = 'response', newdata = test_set[-14])
y_pred = ifelse(prob_pred_glm > 0.5, 1, 0)

# Creamos la Matriz de Confusión
cm = table(test_set[, 14], y_pred > 0.5)
cm
```

```
##
##     FALSE TRUE
##   0    24    8
##   1     6   34
```

```
# calculamos la precisión
precision_glm <- sum(diag(cm)) / sum(cm)
```

Con todas las variables vemos que conseguimos una precisión del modelo del 80.56%. Vamos a probar una segunda aproximación con las variables más significativas.

```
# Ajustamos la regresión logística al conjunto de entrenamiento
classifier = glm(formula = output ~ sex + cp + chol + restecg + thalachh + oldpeak + slp + caa + thall,
                 family = binomial,
                 data = training_set)
# Mostramos resultados
summary(classifier)
```

```
##
## Call:
## glm(formula = output ~ sex + cp + chol + restecg + thalachh +
##     oldpeak + slp + caa + thall, family = binomial, data = training_set)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.2077     0.2150   0.966 0.334021
## sex          -0.8809     0.2545  -3.461 0.000538 ***
## cp            0.8216     0.2113   3.889 0.000101 ***
## chol         -0.5077     0.2411  -2.106 0.035212 *
## restecg       0.4048     0.2069   1.956 0.050470 .
## thalachh      0.6345     0.2571   2.468 0.013569 *
## oldpeak      -0.7855     0.2949  -2.664 0.007717 **
## slp           0.4783     0.2455   1.948 0.051368 .
## caa          -0.6615     0.2297  -2.880 0.003979 **
## thall        -0.5263     0.2033  -2.589 0.009639 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 296.00  on 214  degrees of freedom
## Residual deviance: 151.14  on 205  degrees of freedom
## AIC: 171.14
##
## Number of Fisher Scoring iterations: 6
```

```
# Predecimos los resultados del conjunto de prueba
prob_pred = predict(classifier, type = 'response', newdata = test_set[-14])
y_pred = ifelse(prob_pred > 0.5, 1, 0)

# Creamos la Matriz de Confusión
cm = table(test_set[, 14], y_pred > 0.5)

# calculamos la precisión
precision <- sum(diag(cm)) / sum(cm)
```

En este caso hemos conseguido un modelo con una precisión del 77.78% pero el valor de AIC (Criterio de Información de Akaike) es menor que en el anterior modelo. AIC es un indicador que busca encontrar un equilibrio entre la bondad del ajuste del modelo y la simplicidad del modelo. En general, un modelo con un AIC más bajo se considera mejor, ya que indica que está proporcionando un buen ajuste con un número menor de parámetros.

Si nos fijamos en el primer modelo de regresión logística podemos decir que fue construido para predecir la probabilidad de ocurrencia de enfermedades cardíacas (output) en función de varias variables predictoras (age, sex, cp, trtbps, chol, fbs, restecg, thalachh, exng, oldpeak, slp, caa, thall).

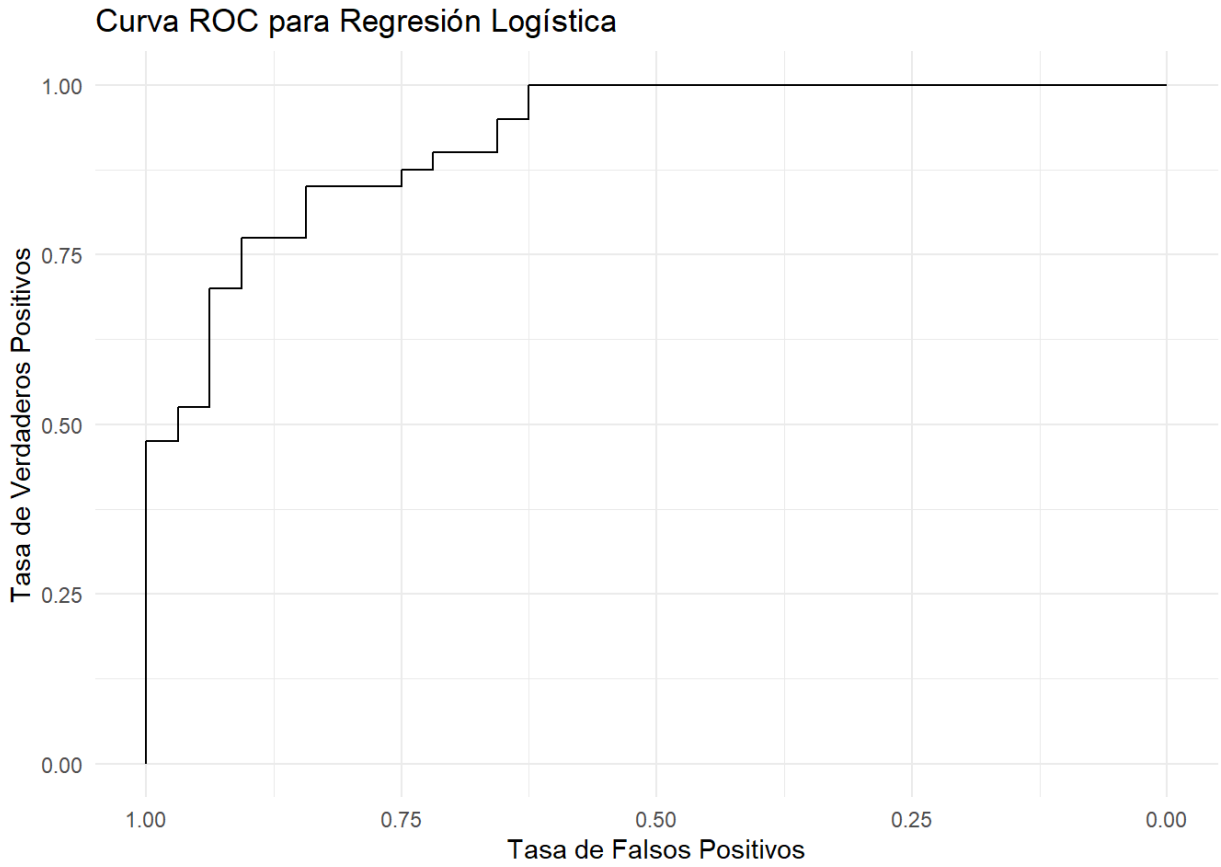
- **age (Edad)**: No se encuentra una relación significativa (coeficiente 0.028971, p-valor 0.913314) entre la edad y la probabilidad de enfermedades cardíacas en este modelo.
- **sex (Género)**: Existe una relación significativa y negativa (coeficiente -0.844252, p-valor 0.001316) entre el género y la probabilidad de enfermedades cardíacas. Las mujeres tienden a tener una menor probabilidad de padecer enfermedades cardíacas en comparación con los hombres.
- **cp (Tipo de dolor torácico)**: Se encuentra una relación positiva y significativa (coeficiente 0.784661, p-valor 0.000587). Mayores niveles de este tipo de dolor se asocian con un aumento en la probabilidad de enfermedades cardíacas.
- **trtbps (Presión arterial en reposo), chol (Colesterol en suero), fbs (Azúcar en sangre en ayunas), restecg (Resultados electrocardiográficos en reposo) y thalachh (Frecuencia cardíaca máxima alcanzada)**: No se encuentran asociaciones significativas con la probabilidad de enfermedades cardíacas en este modelo.
- **exng (Angina inducida por el ejercicio) y slp (Pendiente del segmento ST máximo del ejercicio)**: Aunque no son estadísticamente significativas al 95%, exng muestra una relación negativa y slp indica una posible asociación positiva con la probabilidad de enfermedades cardíacas.
- **oldpeak (Depresión del ST inducida por el ejercicio)**: Existe una relación negativa significativa (coeficiente -0.726294, p-valor 0.016401). Mayor depresión del ST se asocia con una disminución en la probabilidad de enfermedades cardíacas. caa (Número de vasos sanguíneos principales) y thall (Resultado de prueba de esfuerzo cardíaco): Muestran asociaciones significativas con la probabilidad de enfermedades cardíacas. Caa tiene una relación negativa y thall tiene una relación negativa con la probabilidad de enfermedades cardíacas.

En resumen, las variables más influyentes para predecir la ocurrencia de enfermedades cardíacas en este modelo son “cp” (Tipo de dolor torácico), “sex” (Género), “caa” (Número de vasos sanguíneos principales) y “thall” (Resultado de prueba de esfuerzo cardíaco).

El AIC del modelo es 176.91, lo que sugiere que este modelo podría mejorar con ajustes adicionales o la inclusión de más variables predictoras. Además, la deviance residual es significativamente menor que la deviance nula, indicando que el modelo con las variables predictoras explica parte de la variabilidad en la variable de salida (enfermedades cardíacas).

Vamos a obtener otra métrica de nuestro modelo como podría ser el valor de AUC a través de la curva ROC. Creamos una función que podrá ser utilizada en otro métodos.

```
plot_roc_curve <- function(test_set, prob_pred, title) {
  # Crear el objeto roc
  roc_curve <- roc(test_set$output, prob_pred)

  title_graph = paste0("Curva ROC para ", title)
  # Crear la curva ROC con ggplot2
  roc_plot <- ggroc(roc_curve) +
    ggtitle(title_graph) +
    labs(x = "Tasa de Falsos Positivos", y = "Tasa de Verdaderos Positivos") +
    theme_minimal()

  # Mostrar el gráfico
  print(roc_plot)

  # Obtener el área bajo la curva (AUC)
  auc_value <- auc(roc_curve)
  title_auc = paste0("Área bajo la curva (AUC) en ", title, ": ")
  cat(title_auc, round(auc_value, 2), "\n")

  # Devolver el valor de AUC
  return(auc_value)
}

auc_glm<- plot_roc_curve(test_set = test_set, prob_pred = prob_pred_glm, title="Regresión Logística")
```




```
## Área bajo la curva (AUC) en Regresión Logística:  0.92
```

5.3.3 Random forest

Random Forest, un algoritmo de aprendizaje automático que es efectivo para la clasificación.

```
set.seed(123)  # Para reproducibilidad
random_frst <- randomForest( output ~ .,
                             data=training_set)

y_pred_rf <- predict(random_frst, test_set)

# Matriz de confusión
cm = table(test_set[, 14], y_pred_rf > 0.5)
cm
```

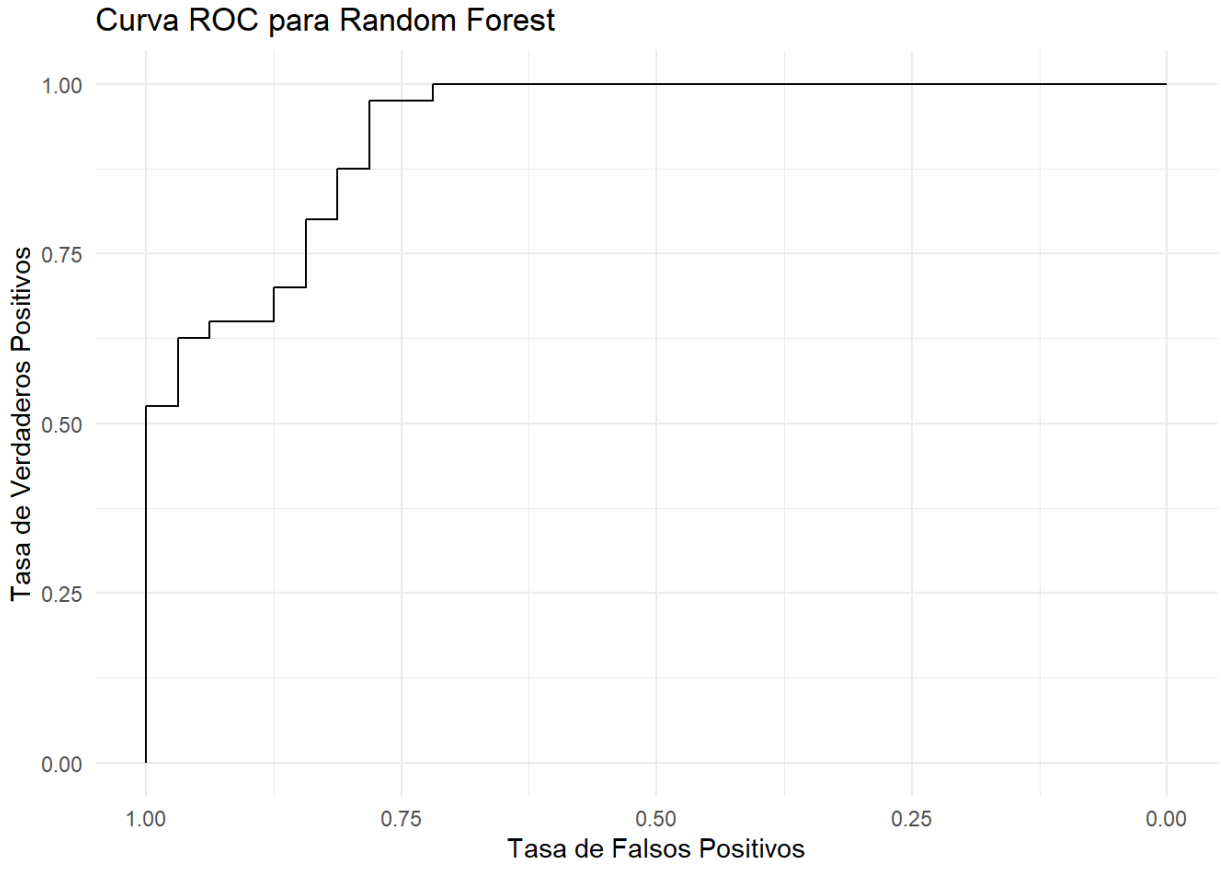
```
##
##     FALSE TRUE
##   0    25    7
##   1     5   35
```

```
precision_rf <- sum(diag(cm)) / sum(cm)
```

Para el modelo de RandomForest hemos obtenido una precisión del 83.33%.

Calculamos el valor de AUC.

```
auc_rf<- plot_roc_curve(test_set = test_set, prob_pred = y_pred_rf, title="Random Forest")
```




```
## Área bajo la curva (AUC) en Random Forest:  0.93
```

5.3.4 k-Nearest Neighbors

k-NN (k-Nearest Neighbors) es un algoritmo de aprendizaje automático supervisado utilizado tanto para clasificación como para regresión. En el contexto de clasificación, k-NN asigna una etiqueta a un nuevo punto de datos basándose en la mayoría de las etiquetas de sus k vecinos más cercanos en el espacio de características. En regresión, k-NN predice el valor de un nuevo punto de datos tomando el promedio de los valores de sus k vecinos más cercanos.

```
set.seed(123)  # Para reproducibilidad
# Aplicar el algoritmo KNN
k_value <- 5  # Ajustar el valor
knn_model <- knn(train = training_set[0:14], test = test_set[0:14], cl = training_set[[14]], k = k_value,  prob
= TRUE)

# Crear la matriz de confusión
conf_matrix_knn <- table(Actual = test_set[[14]], Predicted = knn_model)

# Calcular la precisión del modelo
precision_knn <- sum(diag(conf_matrix_knn)) / sum(conf_matrix_knn)

# Imprimir la matriz de confusión y la precisión
print(conf_matrix_knn)
```

```
##       Predicted
## Actual  0  1
##      0 25  7
##      1  1 39
```

```
print(paste("Precisión del modelo k-NN:", round(precision_knn * 100, 2), "%"))
```

```
## [1] "Precisión del modelo k-NN: 88.89 %"
```

Para el modelo de k-Nearest Neighbors hemos obtenido una precisión del 88.89%.

6 Resolución del problema

Como conclusión final podemos comentar que se realizaron análisis descriptivos detallados para comprender la distribución y características de las variables en nuestro conjunto de datos realizando tareas de limpieza. Asimismo se exploraron visualmente las relaciones entre las variables, identificando posibles patrones o tendencias.

Se realizaron pruebas estadísticas y gráficos para explorar las relaciones entre pares de variables y se evaluó la homogeneidad de varianzas y se tomaron decisiones sobre la aplicabilidad de ciertos análisis.

Se aplicaron técnicas avanzadas, como el **Clustering (k-means)**, para identificar patrones y grupos en los datos.

Se utilizaron modelos predictivos, como **Random Forest**, **Regresión Logística** y **k-Nearest Neighbors**, para predecir la variable de salida (output).

Se han construido unos modelos con una precisión superior al 80% siendo el mejor k-NN con una precisión del 88.89%.

```
# Crear un data frame con los valores
df <- data.frame(Modelo = c("K-means", "Regresión Logística", "Random Forest", "KNN"),
                 Precision = c(precision_kmeans, precision_glm, precision_rf, precision_knn))

ggplot(df, aes(x = Modelo, y = Precision, fill = Modelo)) +
  geom_bar(stat = "identity", color = "black") +
  geom_text(aes(label = paste0(round(Precision * 100), "%")),
            position = position_dodge(width = 0.9),
            vjust = -0.5) +
  labs(title = "Comparación de Precisiones de Modelos",
       x = "Modelo",
       y = "Precision") +
  theme_minimal() +
  theme(legend.position = "none")
```