Práctica 6. Estadística descriptiva 2. Representaciones gráficas

Jesús Martín Fernández

Contenidos

1. Introducción

Los métodos gráficos son de suma importancia en la estadística descriptiva porque ofrecen una manera clara y directa de visualizar datos, facilitando su interpretación. Al representar información de forma visual, es posible resumir grandes volúmenes de datos de manera eficiente, destacando patrones, tendencias y anomalías que, de otra manera, podrían pasar desapercibidos en representaciones numéricas. Estas representaciones gráficas no solo permiten comprender mejor los datos, sino que también mejoran la comunicación de los resultados, haciendo que sean más accesibles para audiencias diversas, independientemente de su nivel de experiencia en estadística.

En términos de utilidad, algunos gráficos se enfocan principalmente en describir la distribución de una variable. Por ejemplo, los histogramas muestran la frecuencia de los datos en intervalos específicos, mientras que los diagramas de caja y bigote (llamados también boxplot) permiten visualizar la mediana, los cuartiles y los valores atípicos. Por otro lado, existen gráficos que permiten intuir relaciones entre variables, como los gráficos de dispersión, que revelan posibles correlaciones entre dos variables continuas, o los diagramas de barras agrupadas, que permiten comparar distribuciones entre categorías. Estas herramientas son clave no solo para entender la

estructura de los datos, sino también para formular hipótesis sobre las relaciones subyacentes entre variables.

2. Gráficos en R, principales funciones.

R ofrece muchas formas de crear gráficos para visualizar datos, lo que es muy útil para entender y comunicar información. Cuando se usa una función gráfica en R, como `plot()`, el gráfico se muestra en una ventana que se abre automáticamente si no hay una activa. Esta ventana es lo que R llama un "dispositivo gráfico". Si prefieres guardar el gráfico en un archivo (como un PDF o una imagen PNG), también puedes hacerlo usando funciones como `pdf()` o `png()`.

En R, hay dos tipos principales de funciones gráficas: las de alto nivel y las de bajo nivel. Las de alto nivel crean un gráfico nuevo desde cero, como cuando dibujas un gráfico de dispersión o un histograma. Las de bajo nivel, en cambio, añaden elementos a un gráfico ya existente, como puntos o líneas adicionales.

Presentamos una tabla con las principales funciones para crear gráficos:

Función	Descripción
`plot(x)`	Gráfico de los valores de x (eje y) ordenados en el eje x.
`plot(x, y)`	Gráfico bivariado de x (eje x) e y (eje y).
`sunflowerplot(x, y)`	Gráfico donde los puntos con coordenadas similares se dibujan como flores, representando la cantidad de puntos.
`pie(x)`	Gráfico circular (diagrama de sectores).
`boxplot(x)`	Gráfico de cajas (box-and-whiskers) que muestra la distribución de los datos.
`stripchart(x)`	Gráfico de los valores de x en línea (alternativa a boxplot para tamaños de muestra pequeños).
`matplot(x, y)`	Gráfico bivariado de columnas de x contra columnas de y.
`hist(x)`	Histograma de las frecuencias de x.
`barplot(x)`	Histograma de los valores de x (gráfico de barras).
`pairs(x)`	Dibuja todos los gráficos bivariados posibles entre las columnas de x.
`qqnorm(x)`	Gráfico de cuantiles de x respecto a los valores esperados bajo una distribución normal.
`contour(x, y, z)`	Gráfico de contornos (los datos se interpolan para dibujar las curvas).
`filled.contour(x, y, z)`	Gráfico de contornos coloreado con leyenda de colores.
`persp(x, y, z)`	Gráfico en perspectiva de datos tridimensionales.

Y otra con las principales funciones para modificar los gráficos:

Función	Descripción	Opciones Principales
`points(x, y)`	Añade puntos al gráfico existente.	`col=`, `pch=`, `cex=`
`lines(x, y)`	Añade líneas al gráfico existente.	`col=`, `lty=`, `lwd=`
`text(x, y, labels, ...)`	Añade texto en las coordenadas (x,y).	`pos=`, `col=`, `cex=`
`mtext(text, side, line, ...)`	Añade texto en los márgenes especificados.	`col=`, `cex=`
`segments(x0, y0, x1, y1)`	Dibuja líneas entre los puntos (x0, y0) y (x1, y1).	`col=`, `lwd=`
`arrows(x0, y0, x1, y1, angle=30, code=2)`	Dibuja flechas entre los puntos especificados.	`col=`, `lwd=`, `length=`
`abline(a, b)`	Dibuja una línea con pendiente b y ordenada en el origen a.	`col=`, `lwd=`
`rect(x1, y1, x2, y2)`	Dibuja un rectángulo delimitado por los puntos (x1, y1) y (x2, y2).	`col=`, `border=`, `lwd=`
`polygon(x, y)`	Dibuja un polígono conectando los puntos (x, y).	`col=`, `border=`, `lwd=`
`legend(x, y, legend)`	Añade una leyenda al gráfico.	`col=`, `pch=`, `lwd=`, `box=TRUE`
`title()`	Añade un título y subtítulo al gráfico.	`main=`, `sub=`, `xlab=`, `ylab=`
`axis(side, vect)`	Añade un eje al gráfico en el lado especificado.	`las=`, `col.axis=`, `line=`
`box()`	Añade un marco alrededor del gráfico.	`lwd=`, `col=`
`rug(x)`	Dibuja pequeñas líneas verticales en el eje x para mostrar la distribución de los datos.	`col=`, `lwd=`
`locator(n, type="n", ...)`	Devuelve las coordenadas (x,y) después de hacer clic en el gráfico.	`type=` (define cómo se dibuja el gráfico)

Ambas tablas están tomadas y modificadas de Paradis, E. (2005). R for Beginners. Institut des Sciences de l'Evolution. Université Montpellier II. No vamos a usar sino las funciones más importantes.

En esta práctica vamos a trabajar sobre cinco tipos de gráficos: de barras y sectores, histogramas, boxplot, Q-Q plot (frente a la normal) y scatterplot (o diagramas de dispersión)

3. Gráficos de barras y sectores.

Los gráficos de barras son útiles para comparar la frecuencia o el tamaño de distintas categorías en una variable categórica. Permiten visualizar rápidamente las diferencias entre grupos,

facilitando la comparación directa entre ellos. Los gráficos de sectores (o gráficos de pastel) se emplean para mostrar la proporción que representa cada categoría respecto al total. Son útiles para entender la contribución relativa de cada parte en un conjunto de datos.

En primer lugar, y como siempre, vamos crear el directorio de trabajo y comprobar que estamos en él

```
#setwd("~/Práctica 6")

#La ruta o pathway es diferente para cada uno.
#getwd ()
```

Ahora recuperamos la base de datos iam (df_iam2 Aula Virtual, Práctica 6), es un archivo.csv

```
df_iam2 <- read.csv("df_iam2.csv")
```

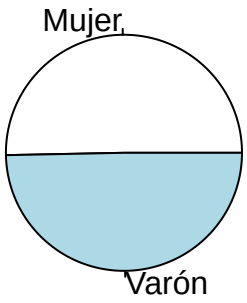
Y como siempre, comprobamos qué variables tiene el archivo

```
head(df_iam2)
```

```
  edad   sex   alt peso      imc hta fum colesterol clas_soc iam    imc_r
1   40 Mujer 157.2 54.2 21.93281  Sí  Sí        192     Alta  Sí   Normal
2   40 Varón 172.5 70.5 23.69250  No  Sí        176     Alta  No   Normal
3   84 Mujer 167.4 63.4 22.62447  Sí  No        226     Baja  No   Normal
4   87 Varón 173.3 96.3 32.06485  Sí  No        188     Baja  No Obesidad
5   83 Varón 172.7 62.7 21.02243  No  No        178     Alta  Sí   Normal
6   60 Mujer 154.3 51.3 21.54694  Sí  No        157     Alta  No   Normal
```

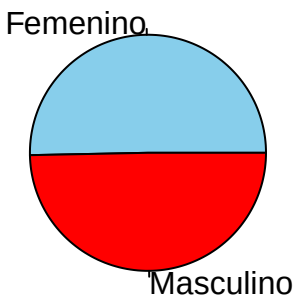
Vamos a dibujar un diagrama de sectores con los hombres y mujeres con la función `pie` , aunque previamente habrá que contar las ocurrencias de cada caso con las funciones `table` y `prop.table` . Se puede hacer todo en una secuencia

```
pie(prop.table(table(df_iam2$sex)))
```



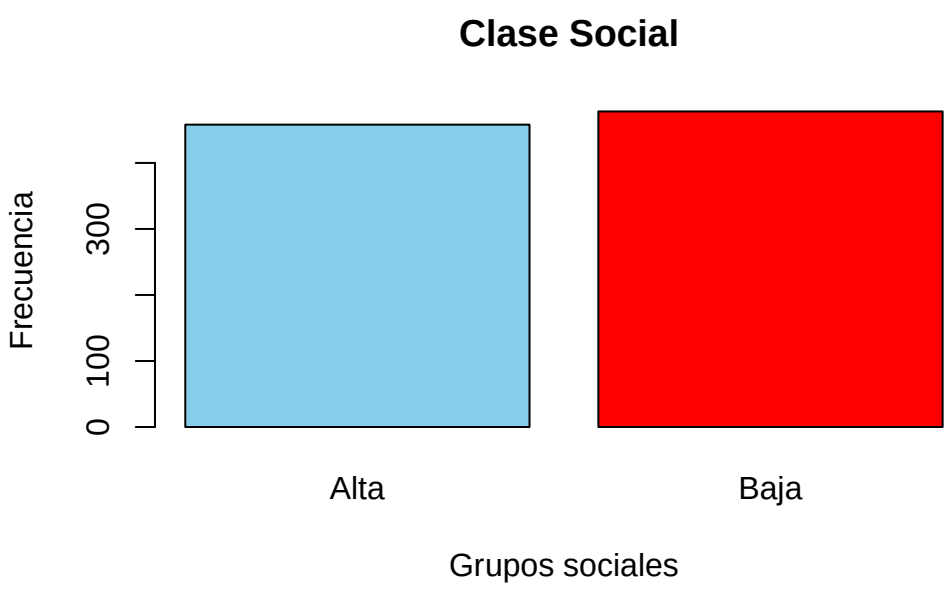
La función `pie` tiene algunas opciones básicas. Vamos a cambiar los colores (`col`), las etiquetas por "Femenino" "Masculino" (`labels`) , a poner un título de gráfico (`main`) y a cambiar el color (`border`) y el grosor (`lwd`) del borde.

```
pie(prop.table(table(df_iam2$sex)), col = c("skyblue", "red"),
    labels = c("Femenino", "Masculino"),
    main = "Distribución por género", border = "black",
    lwd = 2)
```



Ahora crearemos un gráfico de barras con la variable clas_soc

```
# Crear el diagrama de barras
barplot(table(df_iam2$clas_soc),
        main = " Clase Social",
        xlab = "Grupos sociales",
        ylab = "Frecuencia",
        col = c("skyblue", "red"), #puedes cambiar el color de las barras
        border = "black")  # Opcional: color del borde de las barras
```



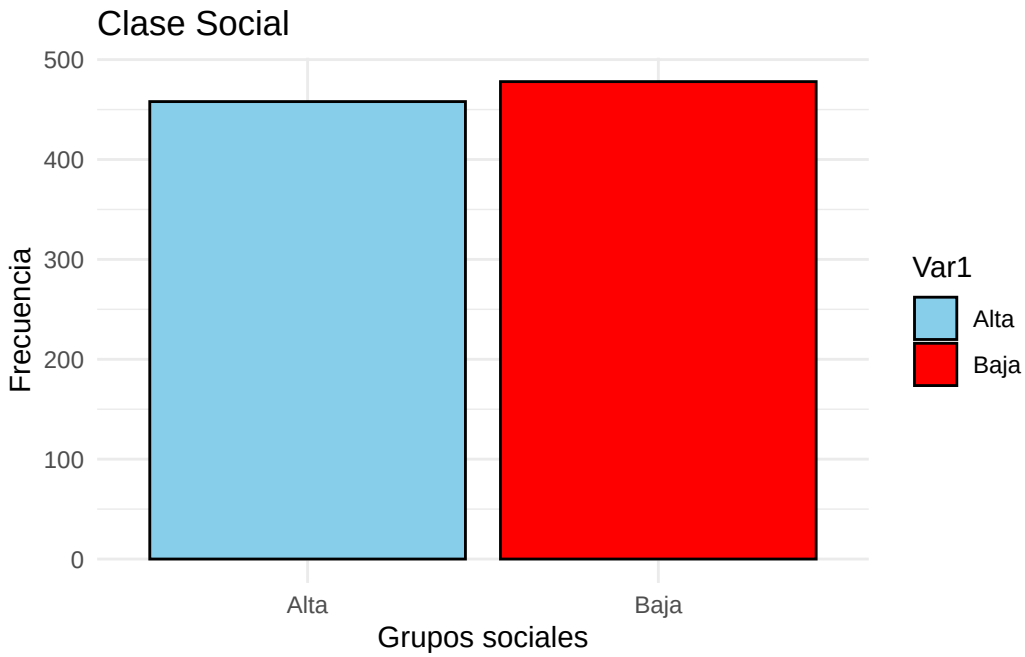
El paquete `ggplot` tiene muchas funcionalidades para hacer gráficos. Este sería el código para hacer el mismo gráfico con este paquete

```
#install.packages ("ggplot2")
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.4.1
```

```
# Convertimos la tabla en un data frame para poder usarla en ggplot2
df_clas_soc <- as.data.frame(table(df_iam2$clas_soc))

# Creamos el gráfico de barras
ggplot(df_clas_soc, aes(x = Var1, y = Freq, fill = Var1)) +
  geom_bar(stat = "identity", color = "black") +
  # "color" define el borde de las barras
  scale_fill_manual(values = c("skyblue", "red")) +
  # Define los colores de las barras
  labs(title = "Clase Social", x = "Grupos sociales", y = "Frecuencia") +
  theme_minimal()  # Opcional: el estilo del gráfico
```

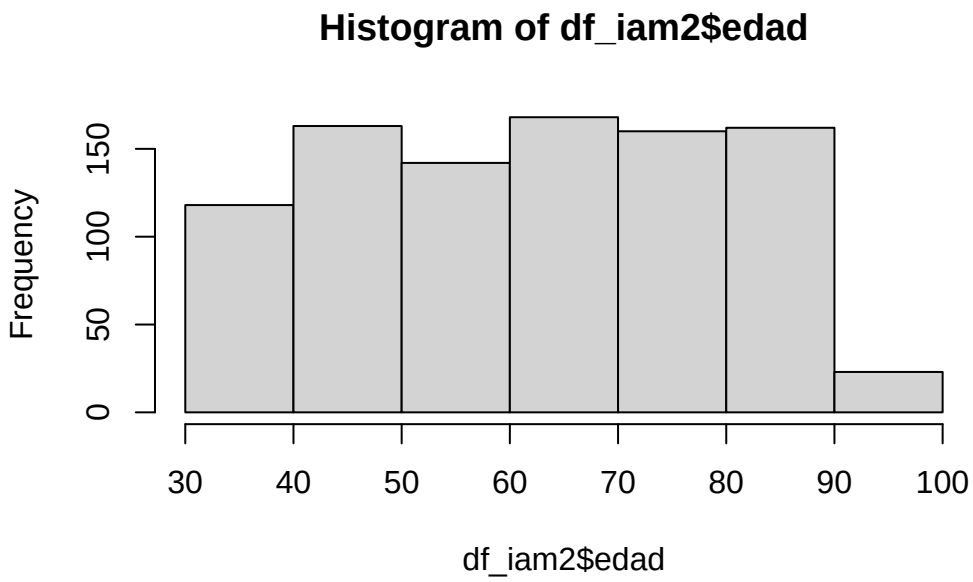


4. Histogramas

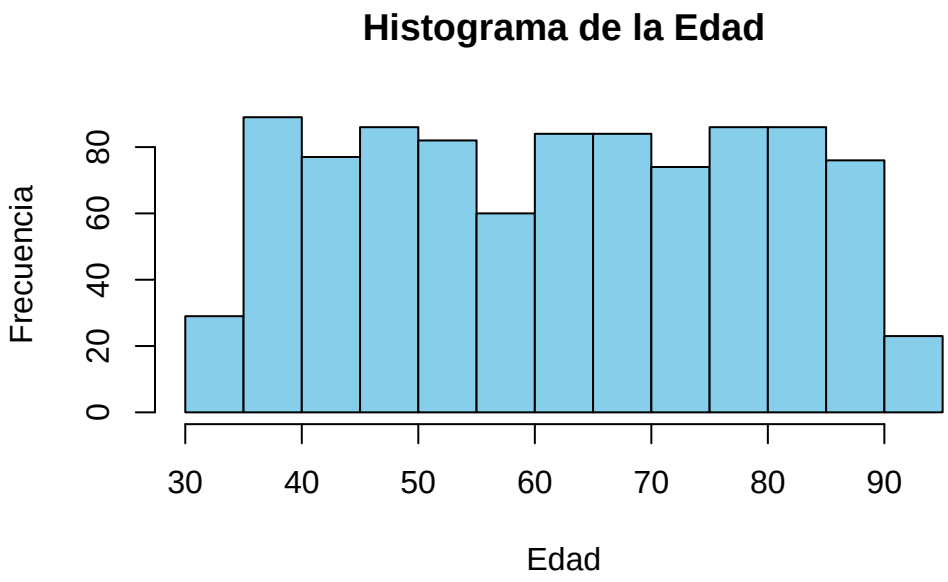
Un histograma es una representación gráfica que muestra la distribución de un conjunto de datos continuos. Se utiliza para visualizar cómo se distribuyen los valores de una variable, agrupándolos en intervalos y mostrando la frecuencia de los datos dentro de cada uno de estos intervalos.

Vamos a hacer un histograma de la variable edad, en intervalos de 10 años, con la función `hist`

```
hist(df_iam2$edad,breaks = 6)
```




```r
hist(df_iam2$edad,
     breaks = 10,
     main = "Histograma de la Edad",
     xlab = "Edad",
     ylab = "Frecuencia",
     col = "skyblue")
```

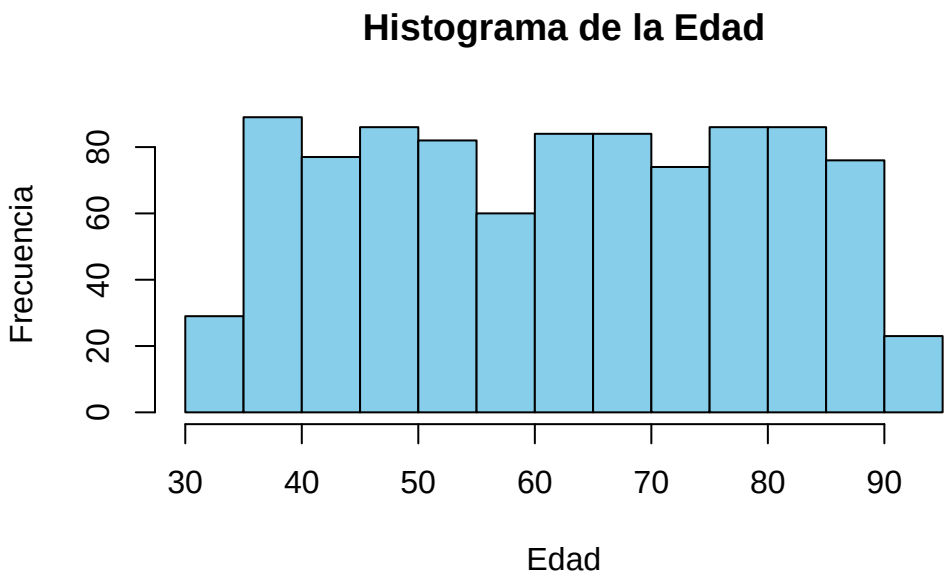


Si quisiésemos guardar la salida gráfica en un formato png, procederiamos así

```
#png("histograma_edad.png", width = 800, height = 400)

#widht = and height= pueden ser cambiados

hist(df_iam2$edad,
     breaks = 10,
     main = "Histograma de la Edad",
     xlab = "Edad",
     ylab = "Frecuencia",
     col = "skyblue")
```



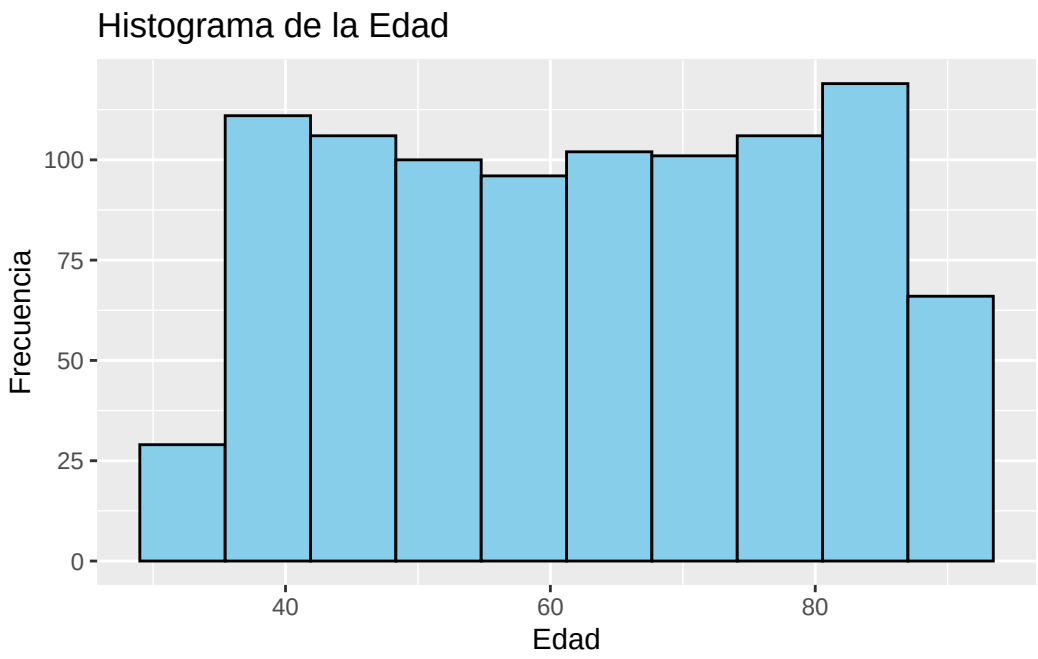

```
#dev.off()
```

Habrá guardado el archivo en el directorio en el que estés trabajando, Si queremos guardarlo en otro lugar se puede pedir al principio con la instrucción:

png("/ruta/completa/histograma_edad_por_sex.png", width = 800, height = 400)

También puede hacerse el histogrrama con `ggplot`

```
# Histograma con 10 intervalos
ggplot(df_iam2, aes(x = edad)) +
  geom_histogram(bins = 10, fill = "skyblue", color = "black") +
  labs(title = "Histograma de la Edad",
       x = "Edad",
       y = "Frecuencia")
```



5. Gráfico de tallo y hoja

El diagrama de tallo y hoja (steam and leaf" es una técnica gráfica que descompone los datos numéricos en dos partes: el tallo, que representa las cifras más significativas (por ejemplo, las decenas), y las hojas, que representan las cifras menos significativas (por ejemplo, las unidades). Introducido por John Tukey en los años 70, este gráfico permite visualizar la distribución de los datos mientras conserva cada valor individual, lo que facilita identificar patrones, concentraciones y la forma general de la distribución. Aunque es fácil de construir y útil para conjuntos de datos pequeños, su uso se vuelve poco práctico con grandes volúmenes de datos. A diferencia de un histograma que agrupa datos en intervalos, el diagrama de tallo y hoja muestra los valores exactos, lo que ofrece una vista detallada de la distribución sin perder la integridad de los datos originales.

Vamos a hacer un diagrma de tallo y hoja con la variable `edad`para que puedas compararlo con la información visual que aporta el histograma

```
stem(df_iam2$edad)
```

```
The decimal point is 1 digit(s) to the right of the |

3 | 44444444444
```

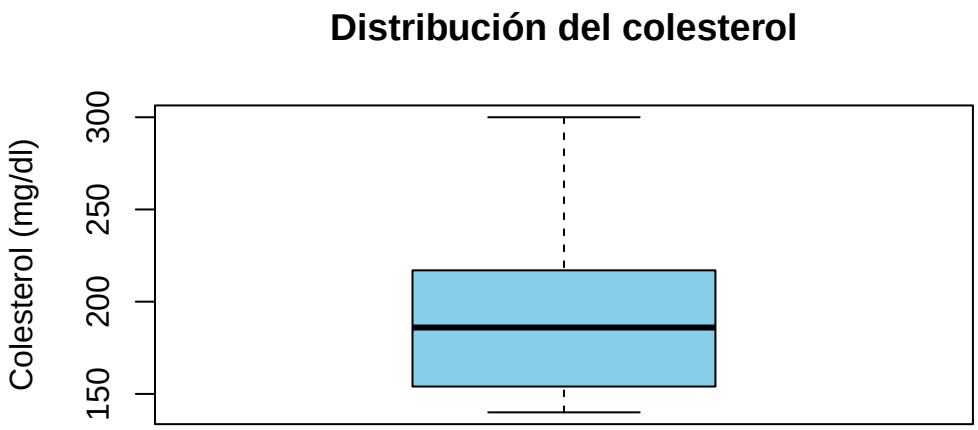
```
  3 | 55555555555555555566666666666666666666666677777777777777778888888888+14
  4 | 00000000000001111111111111111111112222222222333333333333333333444444
  4 | 55555555555555666666666666666666667777777777777788888888888888888999
  5 | 000000000000000000001111111111111111112222222222233333333333333333+6
  5 | 55555555555555555666666666666667777777777778888888888899999999999999
  6 | 00000000111111111111111111122222222222222233333333333333333444444444
  6 | 555555555555555555666666666666666666677777777777777777888888888899+6
  7 | 0000000000000000111111111111111112222222222222233333333333344444444
  7 | 5555555555555555555566666666666667777777777777777778888888888889+5
  8 | 000000000000000000001111111111111112222222222222222333333333333333+17
  8 | 55555555556666666666666677777777777777777778888888888888888899999999
  9 | 00000000000000111111111111111111222222
```

6. Boxplot

Un boxplot, también conocido como diagrama de caja o diagrama de caja y bigotes, es un gráfico que representa la distribución de una variable numérica mediante cinco valores clave: mínimo, primer cuartil (Q1), mediana (Q2), tercer cuartil (Q3) y máximo. La caja muestra el rango intercuartílico (IQR), que abarca del primer al tercer cuartil, con la línea central representando la mediana. Los "bigotes" se extienden hasta los valores más extremos dentro de 1.5 veces el IQR, mientras que los valores fuera de este rango se consideran outliers y se representan como puntos individuales. El boxplot es útil para visualizar la dispersión, asimetría y posibles valores atípicos en los datos, y es ideal para comparar distribuciones entre grupos.

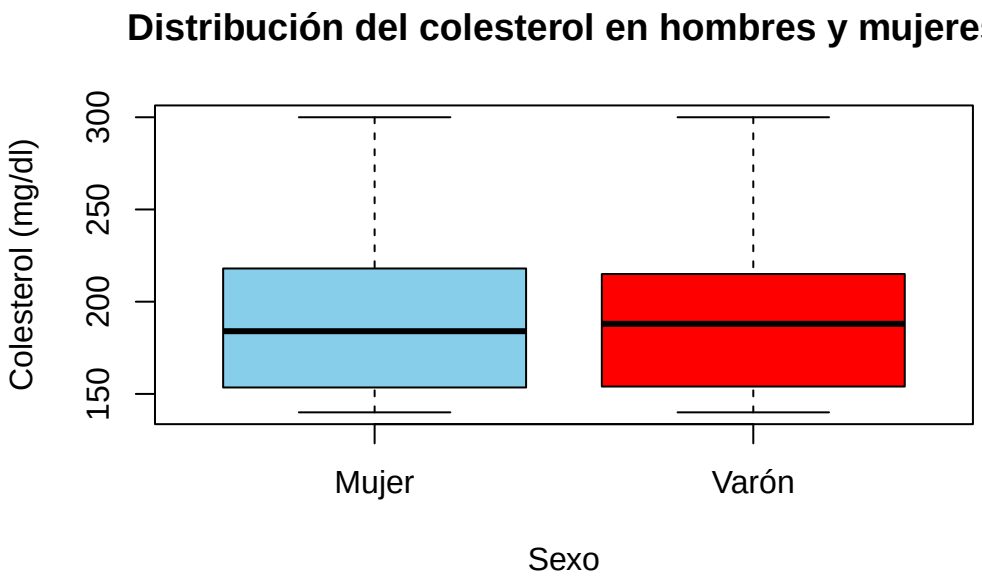
Vamos a construir un boxplot de la variable col con la función `boxplot`

```r
boxplot(df_iam2$col,
main = "Distribución del colesterol",
ylab = "Colesterol (mg/dl)",
col = "skyblue",
border = "black")
```



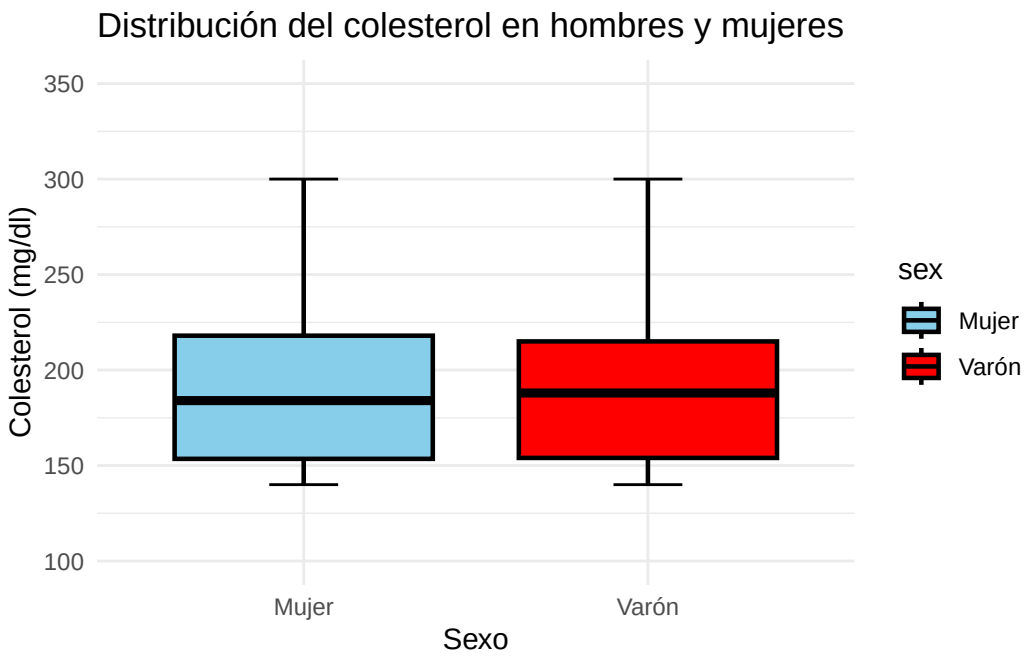
También podríamos comparar la distribución del `colesterol` en cada uno de los dos valores de `sex`

```
boxplot(df_iam2$col ~ df_iam2$sex,
main = "Distribución del colesterol en hombres y mujeres",
xlab = "Sexo",
ylab = "Colesterol (mg/dl)",
col = c("skyblue", "red"),
border = c("black", "black"))
```



Vamos a hacer esta ultima gráfica con ggplot

```
ggplot(df_iam2, aes(x = sex, y = colesterol, fill = sex)) +
  stat_boxplot(geom = "errorbar", width = 0.2, color = "black") +
  # Añadir bigotes correctamente
  geom_boxplot(color = "black", lwd = 0.8) +  # Dibujar la caja con grosor
  scale_fill_manual(values = c("skyblue", "red")) +
  labs(title = "Distribución del colesterol en hombres y mujeres",
       x = "Sexo",
       y = "Colesterol (mg/dl)") +
  coord_cartesian(ylim = c(100, 350)) +  # Ajustar el rango sin excluir datos
  theme_minimal()
```



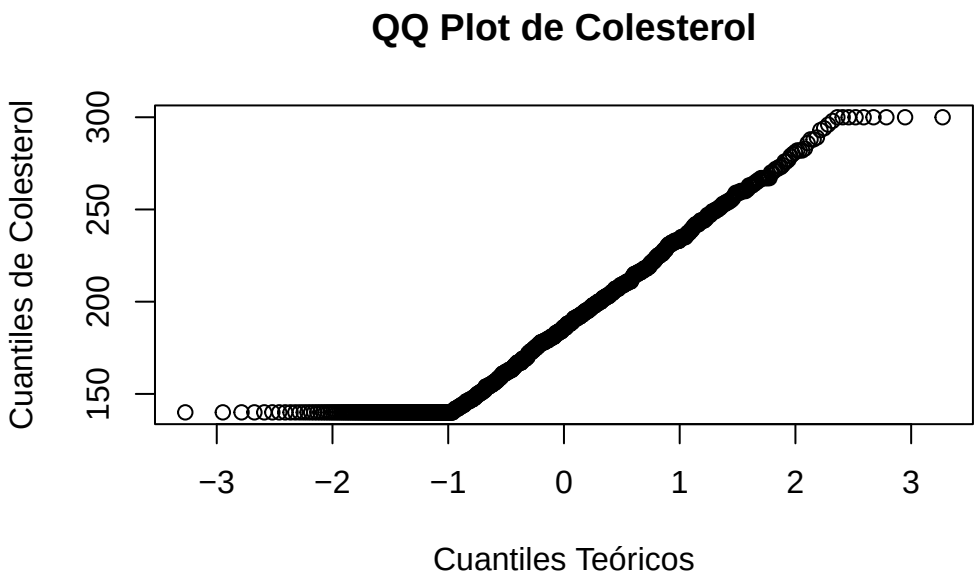

```
#theme_minimal(): Usa un tema más limpio y minimalista
#para mejorar la apariencia (opcional).
```

7. Q-Q plot

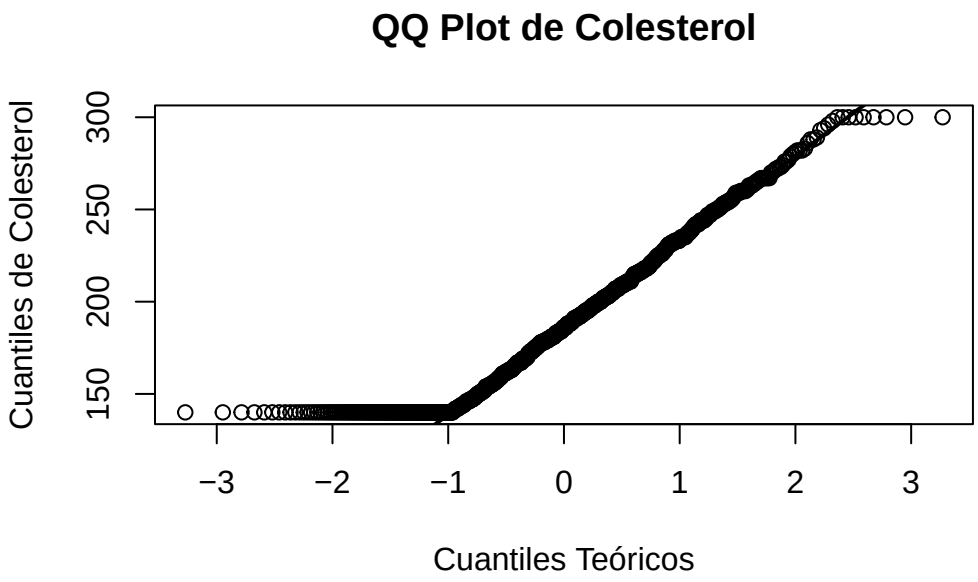
Un QQ plot (gráfico cuantílico-cuantílico) es una herramienta gráfica utilizada para comparar la distribución de dos conjuntos de datos, o para evaluar si un conjunto de datos sigue una distribución teórica específica, como la normal. En un QQ plot, los cuantiles de una distribución se trazan contra los cuantiles de otra distribución. Si los puntos caen aproximadamente en una línea recta, indica que las distribuciones comparadas son similares. Cualquier desviación significativa de esta línea sugiere que las distribuciones difieren, permitiendo detectar asimetrías, colas largas u otros patrones en los datos.

Vamos a hacer un QQ plot frente a una distribución normal de la variable `colesterol`

```
qqnorm(df_iam2$colesterol,
       main="QQ Plot de Colesterol",
       ylab="Cuantiles de Colesterol",
       xlab="Cuantiles Teóricos")
```




```
qqnorm(df_iam2$colesterol,
       main="QQ Plot de Colesterol",
       ylab="Cuantiles de Colesterol",
       xlab="Cuantiles Teóricos")
qqline(df_iam2$colesterol, col="black", lwd=2)# Agrega una línea de referencia
```

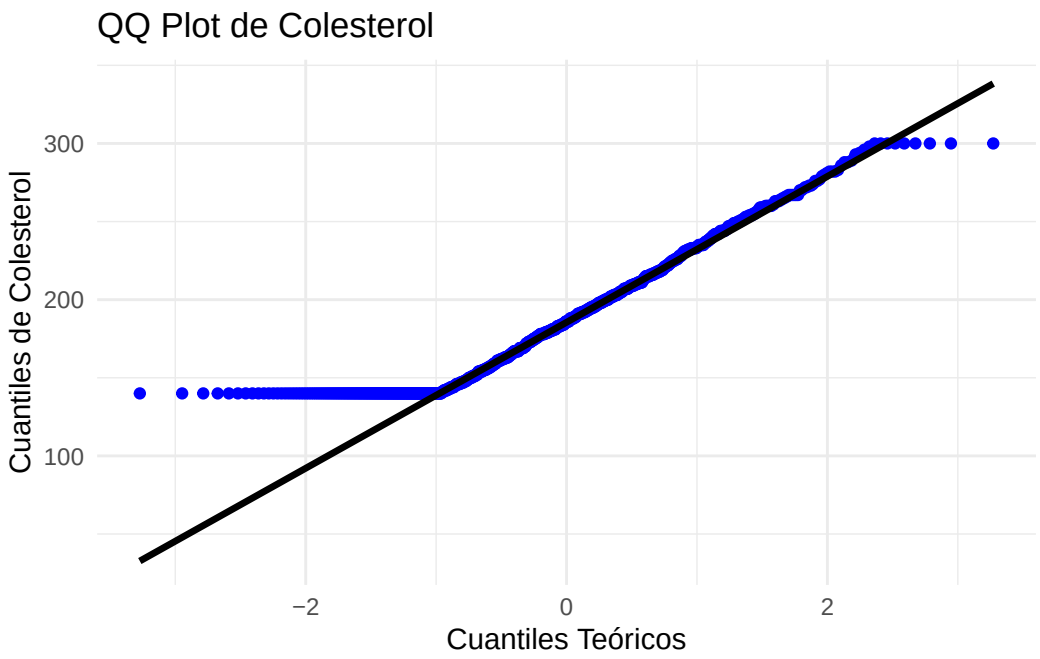


Cuando los puntos caen sobre la línea negra, indica que los datos del colesterol están normalmente distribuidos. Como en este caso hay muchos puntos en ambos extremos de la curva , implica que tenemos unas colas muy largas (en ambos lados).

Vamos a repetir la misma gráfica con `ggplot`

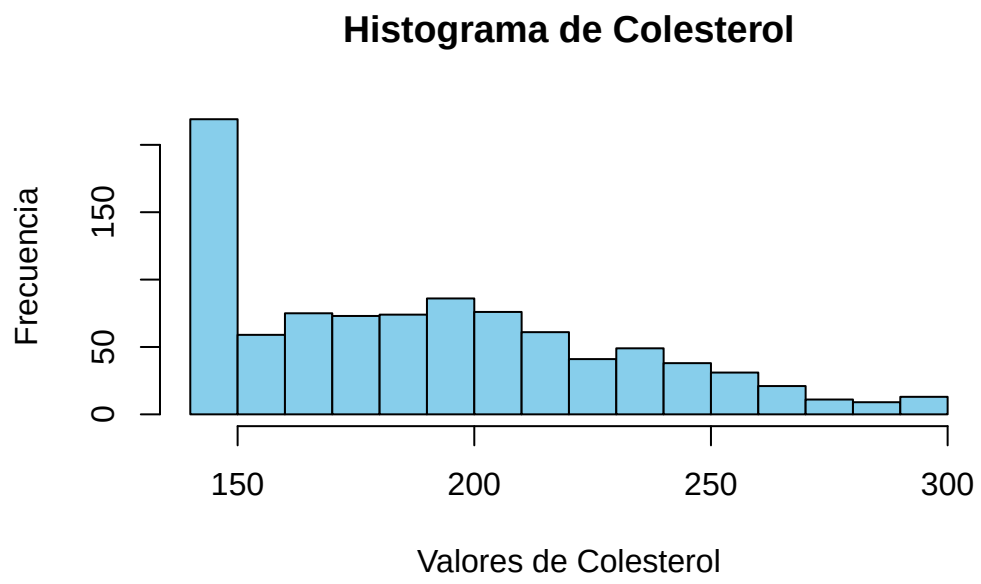
```
ggplot(df_iam2, aes(sample = colesterol)) +
  stat_qq(color = "blue") +
  stat_qq_line(color = "black", size = 1.2) +
  labs(title = "QQ Plot de Colesterol",
       x = "Cuantiles Teóricos",
       y = "Cuantiles de Colesterol") +
  theme_minimal()
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



Mira el histograma de la variable `colesterol`para entender mejor el significado del QQ-plot

```
hist(df_iam2$colesterol,
     main="Histograma de Colesterol",
     xlab="Valores de Colesterol",
     ylab="Frecuencia",
     col="skyblue",
     border="black",
     breaks=15)
```



8. Scatterplot

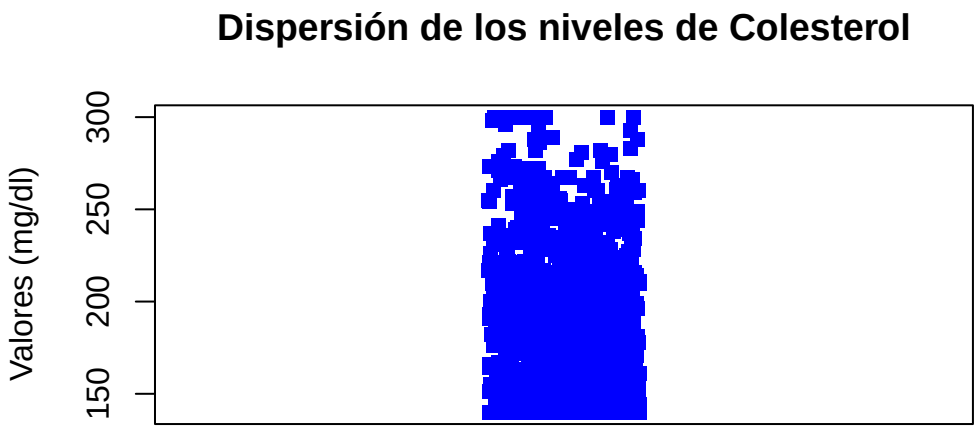
Un scatterplot es un gráfico que muestra la relación entre dos variables numéricas mediante puntos en un plano cartesiano. Cada punto representa una observación, con su valor en el eje X correspondiente a una variable y su valor en el eje Y a la otra. Este gráfico es útil para identificar la dirección y fuerza de la relación entre las variables (correlación positiva, negativa o inexistente), además de detectar patrones, relaciones no lineales o valores atípicos. Si los puntos siguen una tendencia ascendente o descendente, sugieren una correlación; si están dispersos sin un patrón claro, no hay relación evidente.

Una variante del scatterplot, que requiere solo una variable, el stripchart (o dotplot o diagrama de puntos) organiza los puntos a lo largo de un solo eje y puede utilizarse para detectar la densidad, concentración y posibles outliers en los datos.

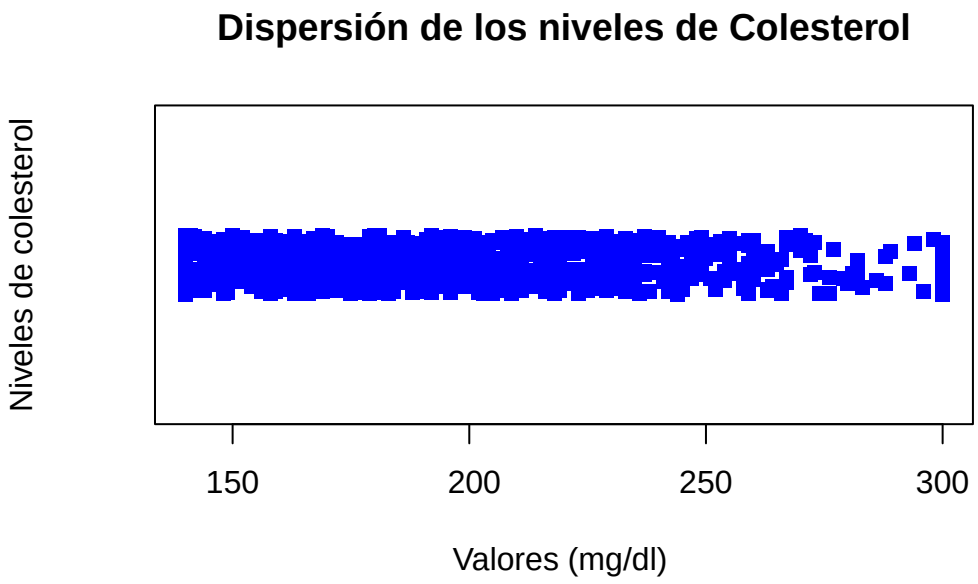
Vamos a hacer un `stripchart` de la variable `colesterol` (primero en vertical y luego en horizontal)‘

```
stripchart(df_iam2$colesterol,
           main= "Dispersión de los niveles de Colesterol",
           method = "jitter",
           vertical = TRUE,
           pch = 15,
           col = "blue",
```

```
           xlab = "Niveles de colesterol",
           ylab = "Valores (mg/dl)")
```

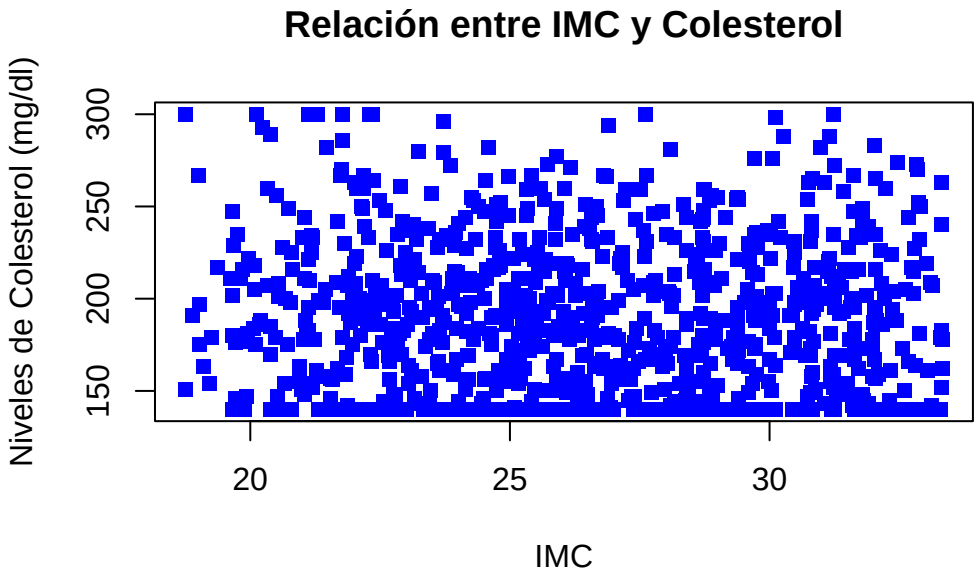



```
stripchart(df_iam2$colesterol,
           main= "Dispersión de los niveles de Colesterol",
           method = "jitter",
           vertical = FALSE,
           pch = 15,
           col = "blue",
           xlab = "Valores (mg/dl)",
           ylab = "Niveles de colesterol")
```



Finalmente haremos un `scaterplott` entre las variables `imc` ‘colesterol

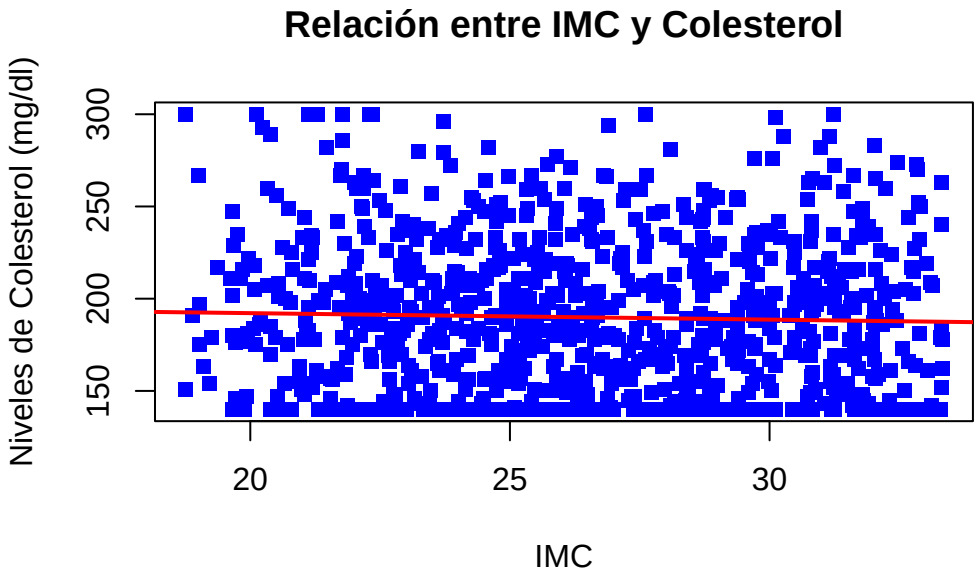
```
plot(df_iam2$imc, df_iam2$colesterol,
     main="Relación entre IMC y Colesterol",
     xlab="IMC",
     ylab="Niveles de Colesterol (mg/dl)",
     pch=15,     # Tipo de punto, sólido
     col="blue") # Color de los puntos
```



Se puede añadir una línea de tendencia, pero para ello hay que "modelizar" la relación entre `imc` y `colesterol` , por ejemplo con una regresión lineal

```
plot(df_iam2$imc, df_iam2$colesterol,
     main="Relación entre IMC y Colesterol",
     xlab="IMC",
     ylab="Niveles de Colesterol (mg/dl)",
     pch=15,
     col="blue")
# Ajustar un modelo de regresión lineal
modelo <- lm(colesterol ~ imc, data=df_iam2)

# Añadir la línea de tendencia
abline(modelo, col="red", lwd=2) # Línea roja, grosor 2
```

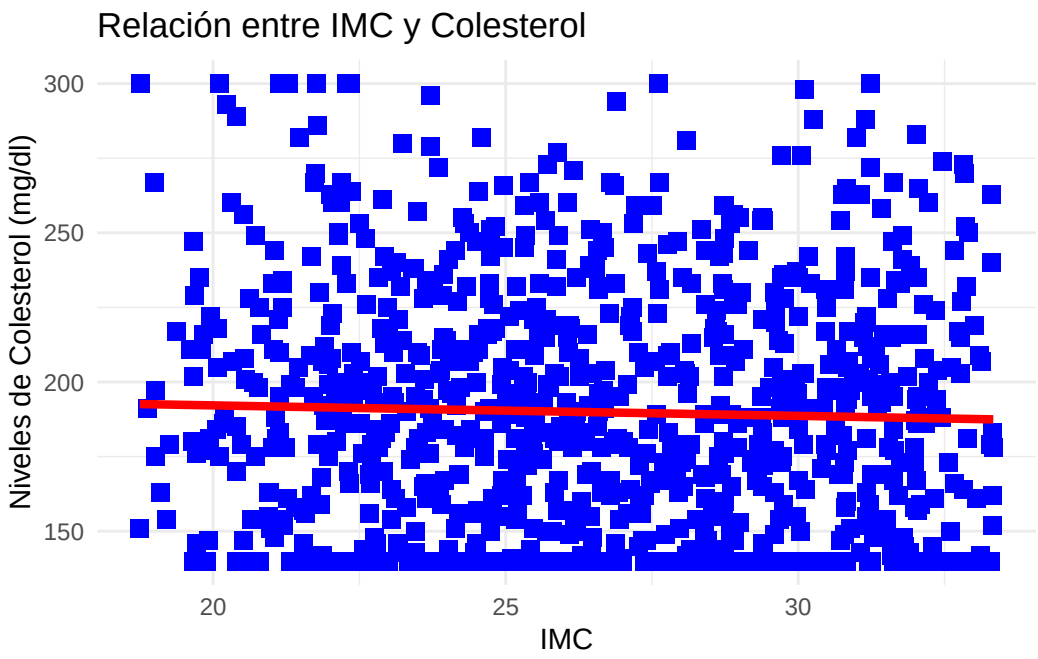


En este caso parece que no hay una asociación entre ambas variables.

Repetimos la gráfica con `ggplot`

```
ggplot(df_iam2, aes(x = imc, y = colesterol)) +
  geom_point(color = "blue", size = 3, shape = 15) +
  # Puntos azules, forma cuadrada
  geom_smooth(method = "lm", color = "red", se = FALSE, size = 1.5) +
  # Línea de regresión
  labs(title = "Relación entre IMC y Colesterol",
       x = "IMC",
       y = "Niveles de Colesterol (mg/dl)") +
  theme_minimal()
```

```
`geom_smooth()` using formula = 'y ~ x'
```


Relación entre IMC y Colesterol
Niveles de Colesterol (mg/dl)
300
250
200
150
20
25
30
IMC