

Taller 1

AUTHOR
Matías Montero



Introducción a R

Hola, ¡bienvenido a R! R es un lenguaje de programación de código abierto con una apasionada comunidad global y utilizado en diversos campos disciplinares. Ha sido diseñado específicamente para el análisis estadístico y la visualización de datos, pero es ampliamente flexible. Por ejemplo, sus extensiones han dado forma a la creación de sitios web, aplicaciones, e incluso el documento que estas leyendo ahora mismo.



Instalación de R y RStudio


Para empezar a utilizar R y RStudio, primero debes instalarlos en tu computador. Vamos paso a paso para instalarlo según el sistema operativo (Windows, macOS y Linux) que tengas.

Paso 1: Instalación de R

- 1. Visita el sitio oficial de CRAN:**
 - Abre tu navegador web y dirígete al sitio de descarga de R en [The Comprehensive R Archive Network](https://cran.r-project.org/) (CRAN).
- 2. Selecciona tu sistema operativo:**
 - En la página de CRAN, verás enlaces para diferentes sistemas operativos. Haz clic en el enlace de tu sistema operativo (Windows, macOS o Linux).
- 3. Descarga e instala R:**
 - Sigue las instrucciones específicas para tu sistema operativo:
 - Windows:** Haz clic en "Download R for Windows" y luego en "base". Descarga el instalador ejecutable (.exe) y ábrelo. Sigue las instrucciones del instalador para completar la instalación.
 - macOS:** Haz clic en "Download R for macOS". Descarga el archivo de paquete (.pkg) y ábrelo. Sigue las instrucciones del instalador para completar la instalación.
 - Linux:** Sigue las instrucciones específicas para tu distribución de Linux en la sección correspondiente de CRAN. Generalmente, esto implicará agregar un repositorio CRAN a tu gestor de paquetes y luego instalar R usando tu gestor de paquetes (por ejemplo, `sudo apt-get install r-base` para Ubuntu).


Paso 2: Instalación de RStudio

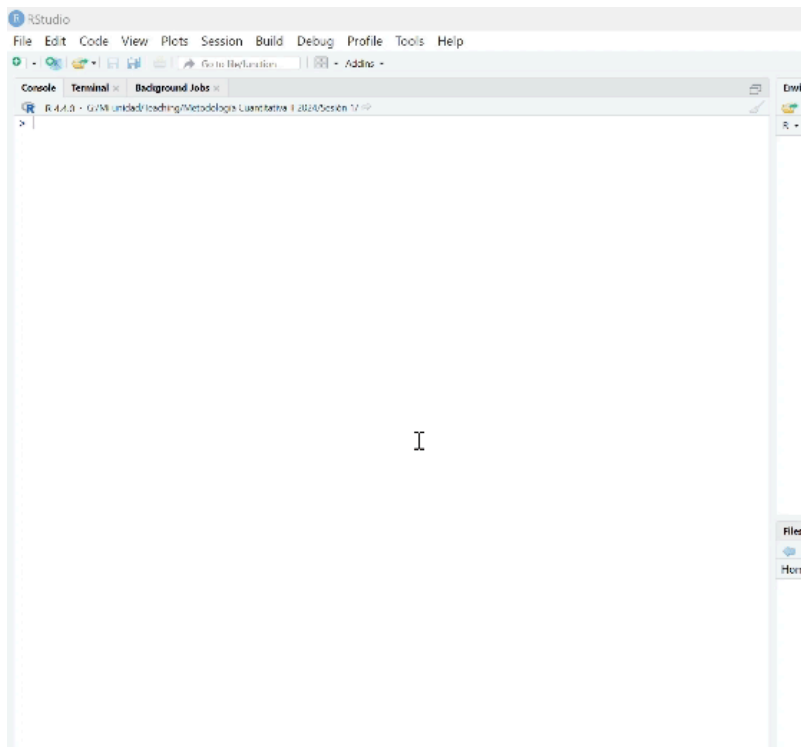
- 1. Visita el sitio oficial de RStudio:**
 - Abre tu navegador web y dirígete al sitio web Posit para descargar el instalador: [RStudio Download](https://posit.co/download/rstudio/).
- 2. Descarga el instalador de RStudio:**
 - En la página de descarga, verás diferentes opciones de instalador para distintos sistemas operativos. Haz clic en el enlace correspondiente a tu sistema operativo.
- 3. Instala RStudio:**
 - Sigue las instrucciones específicas del instalador:
 -  **Windows:** Descarga el instalador ejecutable (.exe) y ábrelo. Sigue las instrucciones del instalador para completar la instalación.
 -  **macOS:** Descarga el archivo de paquete (.dmg) y ábrelo. Arrastra el ícono de RStudio a tu carpeta de Aplicaciones.

-  **Linux:** Descarga el archivo de paquete adecuado para tu distribución (por ejemplo, .deb para Ubuntu o .rpm para Fedora). Abre una terminal y usa tu gestor de paquetes para instalar el archivo (por ejemplo, `sudo dpkg -i rstudio-x.yy.zz-amd64.deb` para Ubuntu).
- Una vez completada la instalación de R y RStudio, abre RStudio y ¡listo!

De ahora en adelante emplearemos RStudio, que es la interfaz que nos permite utilizar R de una forma algo más amigable. Digo “más amigable” porque en realidad, aunque en un principio lo percibas algo críptico esto de analizar datos usando código, en realidad verás que en estas guías de paso a paso te abrirán las puertas para descubrir este (nuevo) universo con mucha facilidad.


💡 ¿Sabías que? También puedes colaborar en R con tus colegas usando [Posit Cloud](#). Es algo así como un Google Docs en la nube, pero en R.

Te invito a que crees un guión (R Script)  en RStudio dirigiéndote en la esquina superior izquierda de la pantalla.



Como podrás haber notado, la interfaz de RStudio está organizada en varios paneles, cada uno con un propósito:

1. **Panel de Script (arriba a la izquierda):** Aquí es donde escribes y editas tu código R. Puedes guardar tus scripts como archivos .R para usarlos más tarde.
2. **Panel de Consola (abajo a la izquierda):** Muestra los resultados de la ejecución de tu código, mensajes de error y advertencias. También, puedes escribir comandos directamente en la consola.
3. **Panel de Entorno/Historia (arriba a la derecha):**
 - **Entorno:** Muestra las variables, funciones y objetos de datos que has creado en tu sesión actual de R.
 - **Historia:** Registra los comandos que has ejecutado anteriormente.
4. **Panel de Archivos/Gráficos/Ayuda/Visualizador (abajo a la derecha):**
 - **Archivos:** Explora los archivos y directorios de tu proyecto.
 - **Gráficos:** Muestra los gráficos que creas en R.
 - **Ayuda:** Accede a la documentación de R y a los archivos de ayuda de los paquetes instalados.
 - **Visualizador:** Examina objetos de datos como tablas y marcos de datos en un formato más fácil de leer.

 Note

La interfaz de RStudio es altamente personalizable, permitiéndote ordenar los paneles de la manera que prefieras y ajustar la apariencia del entorno para adaptarlo a tu gusto. ¡Sí! Puedes ponerlo en color oscuro y colores cálidos, en lugar del aspecto blanco predeterminado, y así no dañar tus lindos ojitos 🐼 ☀️

Conceptos Básicos 🛠️

Clarifiquemos primero algunos términos que comúnmente decimos en R. Te presento primero los objetos.

📌 Siéntete libre de copiar y pegar el código de este taller en tu propio script. Para ejecutar una línea de código, simplemente sitúate en la línea y teclea CNTRL + Enter, mientras que en Mac es CMD + Enter.

```
# Aquí asignamos un valor numérico a un objeto "X"
x <- 7
x
```

```
[1] 7
```

```
# Aquí creamos un vector de números concatenados
NumVar <- c(5, 3, 4)
NumVar
```

```
[1] 5 3 4
```

```
# Los vectores pueden estar concatenados con información de tipo texto o caracteres
CharVar <- c("verde", "amarillo", "verde")
CharVar
```

```
[1] "verde"    "amarillo" "verde"
```

```
# También utilizamos datos de tipo factor para representar datos categóricos
factores <- factor(c("femenino", "masculino", "masculino", "femenino"))

# Aunque los factores también pueden tener una lógica ordinal
factores_ord <- factor(c("bajo", "medio", "alto", "alto"),
                      ordered = TRUE,
                      levels = c("bajo", "medio", "alto"))

factores
```

```
[1] femenino masculino masculino femenino
Levels: femenino masculino
```

```
factores_ord
```

```
[1] bajo medio alto alto
Levels: bajo < medio < alto
```

```
# Es posible almacenar valores lógicos que asume solo dos valores posibles
hola <- c("TRUE", "FALSE", "FALSE", "TRUE")
hola
```

```
[1] "TRUE" "FALSE" "FALSE" "TRUE"
```

```
# A veces tendremos datos perdidos representados por NA
vector_con_na <- c(1, 2, NA, 4, 5)
vector_con_na
```

```
[1] 1 2 NA 4 5
```

```
# R también es capaz de manejar información de fechas y tiempo (hora)
fecha <- as.Date("2024-07-24")
fecha
```

```
[1] "2024-07-24"
```

```
fecha_hora <- as.POSIXct("2024-07-24 12:34:56")
fecha_hora
```

```
[1] "2024-07-24 12:34:56 BST"
```

```
# Oh mira, con el símbolo de gato puedes anotar en el código para que no olvides lo que haces
```

Como podrás haber intuido:

- Un vector es una secuencia de datos del mismo tipo.
- El nombre creado al objeto es completamente a nuestra elección. Como ves en el ejemplo, puede llamarse `CharVar` o `hola`.

En general, trabajarás en R usando un `data.frame`, que significa cuadro de datos. Un `data.frame` es un objeto que almacena un conjunto de vectores que, a su vez, pueden ser vectores que almacenan información de tipo numérica, factores, entre otros. Comúnmente, los vectores dentro de un `data.frame` se les conoce como variables, las cuales son las columnas que almacenan atributos de la unidad de observación de nuestro conjunto de datos representados en las filas. Por ejemplo, las filas pueden ser estudiantes, escuelas, países, mientras que las columna serán sus atributos (e.g., género).

Ejercicio 1: Creando tu primer `data.frame()`


En el guión de R que generaste en tu sesión de RStudio, crea un vector llamado `edad` que concatene cinco valores numéricos. También, crea un vector llamado `nombre` que concatene cinco nombres de personas de tipo caracteres.

 Crea los vectores como objetos. Luego, utiliza la función `data.frame()` para almacenarlos.

Sobre este `data.frame` podrás calcular diferentes estadísticos como un promedio o desviación estándar.

Ejercicio 2: Explorar un data frame

Crea un objeto llamado `CarTab` empleando los datos `mtcars`, el cual son datos de ejemplo de la revista Motor Trend US de 1974, donde cada fila es un auto y sus diferentes atributos (variables). Luego, sobre el objeto `CarTab`, emplea los comandos `class()`, `str()`, `head()`, y `summary()`. Termina calculando el promedio de la variable `hp` usando la función `mean()`.

 Escribe `CarTab` asignándolo con `<-` y luego escribe `mtcars`. Para usar los comandos, escribe el comando respectivo y entre los paréntesis escribe el nombre de objeto `CarTab`. Para el comando `mean()` tendrás que especificar la columna con `$` del objeto.

Resumiendo, puedes manejar diferentes tipos de datos en R: - Numéricos: Números enteros y decimales. - Caracteres: Cadenas de texto. - Factores: Datos categóricos ordenados o no ordenados. - Fechas: Información de tiempo en un formato de fecha, que también puede incluir hora. - Lógicos: Valores de verdad (`TRUE` o `FALSE`). - Datos perdidos: Información faltante (`NA`) en un conjunto de datos.

Ejercicio 3: Gráficos Básicos

En este ejercicio, vamos a crear gráficos básicos utilizando el objeto `CarTab`.

1. Crea un gráfico de dispersión:

- Muestra la relación entre las variables `hp` (caballos de fuerza) y `mpg` (millas por galón) del conjunto de datos `CarTab`. Para ello, usa el comando `plot()` indexando el cuadro de datos `CarTab` con el signo `$` con las variables mencionadas anteriormente, separándolos por coma.

2. Dale color :

- Muestra la relación entre las mismas variables, pero esta vez colorea los puntos de azul. Tendrás que ocupar, dentro de `plot(datos$variable1, ``datos$variable2``)`, el argumento `col = ""` con el nombre del color que desees en inglés (e.g., "red"). Recuerda anteponer una coma `,` para separar los argumentos de un comando.

3. Crea un histograma:

- Muestra la distribución de la variable `qsec` (cuarto de milla en segundos) usando el comando `hist()`.

💡 Tip

Los gráficos son útiles para visualizar relaciones y distribuciones de datos. Elige el tipo de gráfico según tus preguntas de investigación y el tipo de análisis que necesites.

Ejercicio 4: Transformación de Variables

En este ejercicio, aprenderás a crear nuevas variables a partir de las existentes en el objeto `CarTab`.

💡 Tip

Transformar variables te permite crear nuevos indicadores y categorías para análisis más sofisticados.

1. Crea una nueva variable `rhprt`:

- Calcula una división entre `hp` (caballos de fuerza) y `wt` (peso en miles de libras) y almacénala en una nueva variable `rhprt`.

💡 Tip

Usa el operador de división `/` para calcularlo y guarda el resultado en `CarTab` indexándolo con el signo `$`.

2. Crea una variable categórica `Rapido`:

- Utiliza la función `ifelse` para crear una nueva variable `Rapido` que tome el valor 1 si `qsec` es menor a 18, y 0 en caso contrario.

💡 Tip

La función `ifelse()` tiene la estructura: `ifelse(condición, valor_si_verdadero, valor_si_falso)`. Úsala para crear la variable `Rapido` basada en la condición lógica anterior. Usa el operador lógico `<`.

Ejercicio 5: Indexación o Extracción de Datos

En este ejercicio, aprenderás a extraer subconjuntos de datos del objeto `CarTab`.

1. Extrae un subconjunto de datos:

- Extrae las columnas `qsec` y `Rapido` y almacénalas en un nuevo objeto llamado `Sub_CarTab`.

💡 Tip

Usa la notación de corchetes `[]` para seleccionar columnas específicas de un data frame. La sintaxis es `nombre_de_tu_data_frame[, c("nombre_columna1", "nombre_columna2")]`.

Ejercicio 6: Regresión

En este ejercicio, estimarás un modelo de regresión lineal utilizando el objeto `CarTab`.

1. Ajusta un modelo de regresión lineal:

- Ajusta un modelo de regresión lineal que prediga `qsec` en función de `hp` y `wt` asignado con `<-` a un objeto que lo llamarás `mod1`.

💡 Tip

La función `lm()` ajusta modelos de regresión lineal. La estructura es `lm(formula, data)`, donde `formula` es una fórmula de la forma `variable_dependiente ~ predictor1 + predictor2`. Usa `summary()` para obtener las estimaciones del modelo ajustado, por ejemplo `summary(lm(qsec ~ hp + wt, data = CarTab))`.



Tip

Al final de este documento encontrarás las soluciones a los ejercicios.

Análisis Descriptivo

¿Cómo se ve un flujo de trabajo inicial de análisis de datos en R? Un flujo de trabajo típico de análisis de datos en R comienza con la configuración del directorio de trabajo. El directorio de trabajo es la carpeta en la que se almacenarán tu script, datos, y los diferentes archivos resultantes de tu análisis, como tablas y gráficos. Establecer el directorio de trabajo te permite organizar tus archivos de manera eficiente y asegura que R sepa dónde buscar y guardar archivos.

¿Cómo establecer tu directorio de trabajo?

1. Encuentra la ruta del directorio de trabajo:

- **Windows:**
 - Abre el Explorador de archivos y navega hasta la carpeta donde quieres almacenar tus archivos.
 - Haz clic en la barra de direcciones y copia la ruta completa.
- **macOS:**
 - Abre el Finder y navega hasta la carpeta donde quieres almacenar tus archivos.
 - Haz clic derecho en la carpeta, selecciona "Obtener información" y copia la ruta que aparece en "Dónde".

2. Establece el directorio de trabajo en R:

- Usa la función `setwd()` para establecer el directorio de trabajo. La ruta debe estar entre comillas y utilizar barras diagonales (`/`) en lugar de barras invertidas (`\`).

```
# setwd("C:/tu_ruta/del_computador/que_estas/usando")
# Ejemplo para Windows: setwd("C:/Users/TuNombre/Magister/MetCuant")
# Ejemplo para macOS: setwd("/Users/TuNombre/Magister/MetCuant")
```

- Puedes verificar el directorio de trabajo actual utilizando la función `getwd()`, que te mostrará la ruta del directorio de trabajo establecido.

⚠ Warning

Organiza tus archivos en subcarpetas dentro de tu directorio de trabajo, por ejemplo, `data` para datos brutos, `scripts` para tus guiones de R, y `outputs` para resultados como gráficos y tablas. Te recomiendo el protocolo IPO que puedes ver [aquí](#) para organizar tus carpetas. Con esto, tendrás un flujo de trabajo ordenado, y tu futuro yo te lo agradecerá.

Instalación de Paquetes

Para llevar a cabo análisis de datos en R, a menudo necesitamos paquetes adicionales que no vienen incluidos en la instalación base de R. Estos paquetes contienen funciones y datasets adicionales que facilitan diversas tareas.

1. Instalación de Paquetes:

- Usa la función `install.packages()` para instalar los paquetes necesarios.

```
options(repos = c(CRAN = "https://cran.r-project.org"))
install.packages("readxl") # Paquete para importar datos en formato Excel}
install.packages("janitor") # Paquete para limpieza de datos
install.packages("ggplot2") # Paquete para generar gráficos
```

2. Carga de Paquetes:

- Después de instalar los paquetes, usa la función `library()` para cargarlos en tu sesión de R.

```
library(readxl)
library(janitor)
library(ggplot2)
```

Importar Datos

Para importar datos en R desde un archivo Excel, usaremos el paquete `readxl`.

1. Importa los datos:

- Usa la función `read_excel()` para leer los datos de un archivo Excel y almacenarlos en un data frame. Asegúrate tener descargado el archivo Excel `Base-Personal-Academico-2021_SIES.xlsx` en tu directorio de trabajo.

```
personal_2021 <- read_excel("Base-Personal-Academico-2021_SIES.xlsx")
```

2. Visualiza los datos:

- Usa la función `View()` para abrir una nueva ventana y visualizar la base de datos.

```
View(personal_2021)
```

Limpieza de Datos

Los nombres de las variables en la base de datos pueden necesitar limpieza para ser más manejables.

1. Limpia los nombres de las variables:

- Usa la función `clean_names()` del paquete `janitor` para estandarizar los nombres de las variables.

```
personal_2021 <- clean_names(personal_2021)
```

2. Exploremos los datos:

- Usa diversas funciones para obtener una visión general de los datos.

```
class(personal_2021) # Muestra la clase de objeto de personal_2021
head(personal_2021)  # Muestra los primeros 6 casos en la base de datos
names(personal_2021) # Lista los nombres de las variables en la base de datos
dim(personal_2021)   # Muestra el número de variables y casos
summary(personal_2021) # Entrega estadísticos descriptivos para cada variable
str(personal_2021)   # Muestra la estructura interna de la base de datos
```

Generación de Nuevas Variables

1. Crear una nueva variable `tipo_ies`:

- Primero, genera una nueva variable `tipo_ies` que inicialmente sea igual a `codigo_institucion`.

```
personal_2021$tipo_ies <- personal_2021$codigo_institucion
```

2. Asignar valores a `tipo_ies` según `codigo_institucion`:

- Usa el operador de asignación `<-` para clasificar las instituciones en diferentes tipos.

```
### 1: Universidad Estatal CRUCH.
personal_2021$tipo_ies[personal_2021$codigo_institucion == 70] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 71] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 72] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 73] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 74] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 75] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 76] <- 1
personal_2021$tipo_ies[personal_2021$codigo_institucion == 77] <- 1
```



```
personal_2021$tipo_ies[personal_2021$codigo_institucion == 909] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 910] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 911] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 912] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 913] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 914] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 915] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 916] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 917] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 918] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 919] <- 6
personal_2021$tipo_ies[personal_2021$codigo_institucion == 920] <- 6
```

Visualización y Etiquetado de Datos

1. Visualiza los valores generados:

```
personal_2021$tipo_ies
```

2. Asigna etiquetas a los valores generados:

- Usa la función `factor()` para asignar etiquetas a los valores de `tipo_ies`.

```
personal_2021$tipo_ies_n <- factor(personal_2021$tipo_ies,
                                   levels = c(1, 2, 3, 4, 5, 6),
                                   labels = c("U. ESTATAL", "U. PRIVADA CRUCH",
                                              "U. PRIVADA", "IP", "CFT PRIVADO",
                                              "CFT ESTATAL"))
```

Selección de Subconjuntos de Datos

Para trabajar con subconjuntos específicos de datos, puedes usar la función `subset()` o la notación de corchetes `[]`.

1. Selecciona un subconjunto de datos:

- Filtra las filas del data frame según la nueva variable categórica `tipo_ies_n`.

```
personal_2021_uestatal <- personal_2021[which(personal_2021$tipo_ies_n == "U. ESTATAL"),]
```

Soluciones de los Ejercicios

Ejercicio 1

```
edad <- c(40, 5, 16, 28, 32)
nombre <- c("Daniel", "Mariela", "Miguel", "Pablo", "Javiera")
misdatos <- data.frame(nombre, edad)
misdatos
```

```
  nombre edad
1 Daniel   40
2 Mariela    5
3 Miguel   16
4 Pablo    28
5 Javiera   32
```

Ejercicio 2

```
CarTab <- mtcars
class(CarTab)
```

[1] "data.frame"

CarTab

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

str(CarTab)

```
'data.frame':  32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
 $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

head(CarTab)

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

```
summary(CarTab)
```

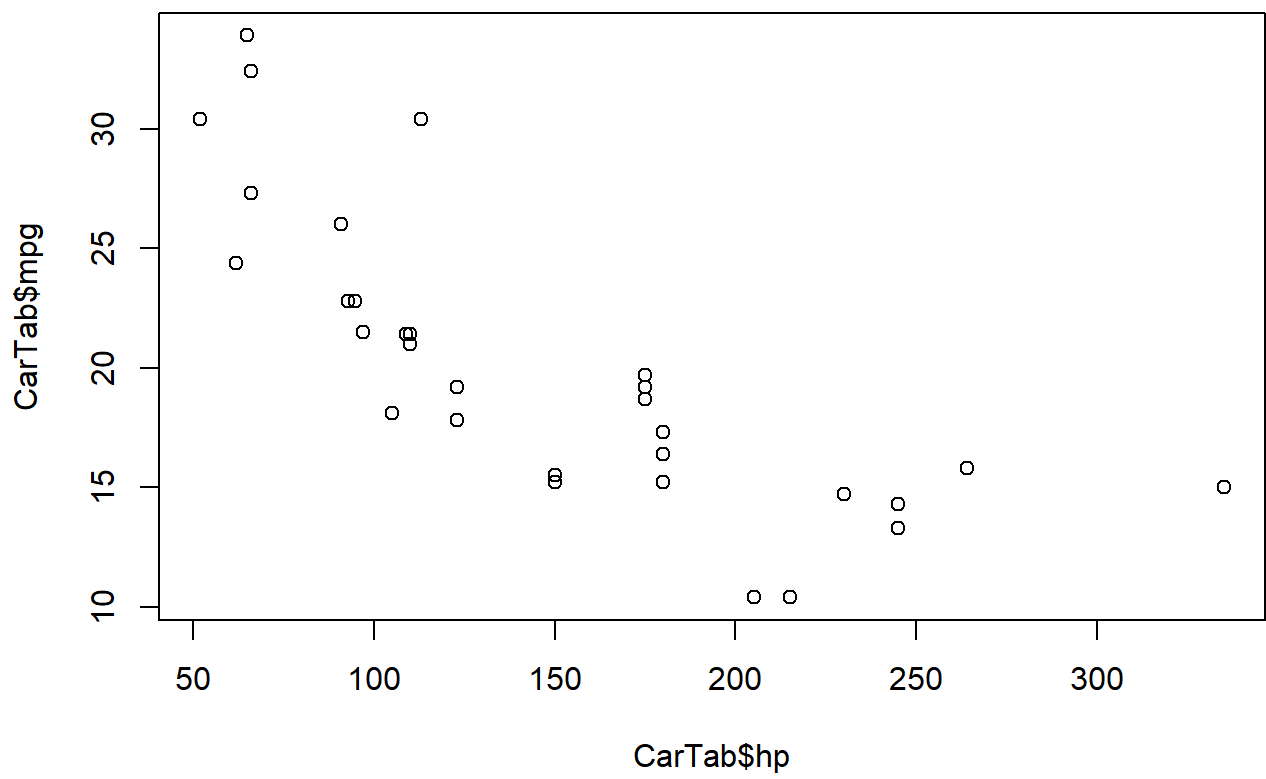
mpg	cyl	disp	hp
Min. :10.40	Min. :4.000	Min. : 71.1	Min. : 52.0
1st Qu.:15.43	1st Qu.:4.000	1st Qu.:120.8	1st Qu.: 96.5
Median :19.20	Median :6.000	Median :196.3	Median :123.0
Mean :20.09	Mean :6.188	Mean :230.7	Mean :146.7
3rd Qu.:22.80	3rd Qu.:8.000	3rd Qu.:326.0	3rd Qu.:180.0
Max. :33.90	Max. :8.000	Max. :472.0	Max. :335.0
drat	wt	qsec	vs
Min. :2.760	Min. :1.513	Min. :14.50	Min. :0.0000
1st Qu.:3.080	1st Qu.:2.581	1st Qu.:16.89	1st Qu.:0.0000
Median :3.695	Median :3.325	Median :17.71	Median :0.0000
Mean :3.597	Mean :3.217	Mean :17.85	Mean :0.4375
3rd Qu.:3.920	3rd Qu.:3.610	3rd Qu.:18.90	3rd Qu.:1.0000
Max. :4.930	Max. :5.424	Max. :22.90	Max. :1.0000
am	gear	carb	
Min. :0.0000	Min. :3.000	Min. :1.000	
1st Qu.:0.0000	1st Qu.:3.000	1st Qu.:2.000	
Median :0.0000	Median :4.000	Median :2.000	
Mean :0.4062	Mean :3.688	Mean :2.812	
3rd Qu.:1.0000	3rd Qu.:4.000	3rd Qu.:4.000	
Max. :1.0000	Max. :5.000	Max. :8.000	

```
mean(CarTab$hp)
```

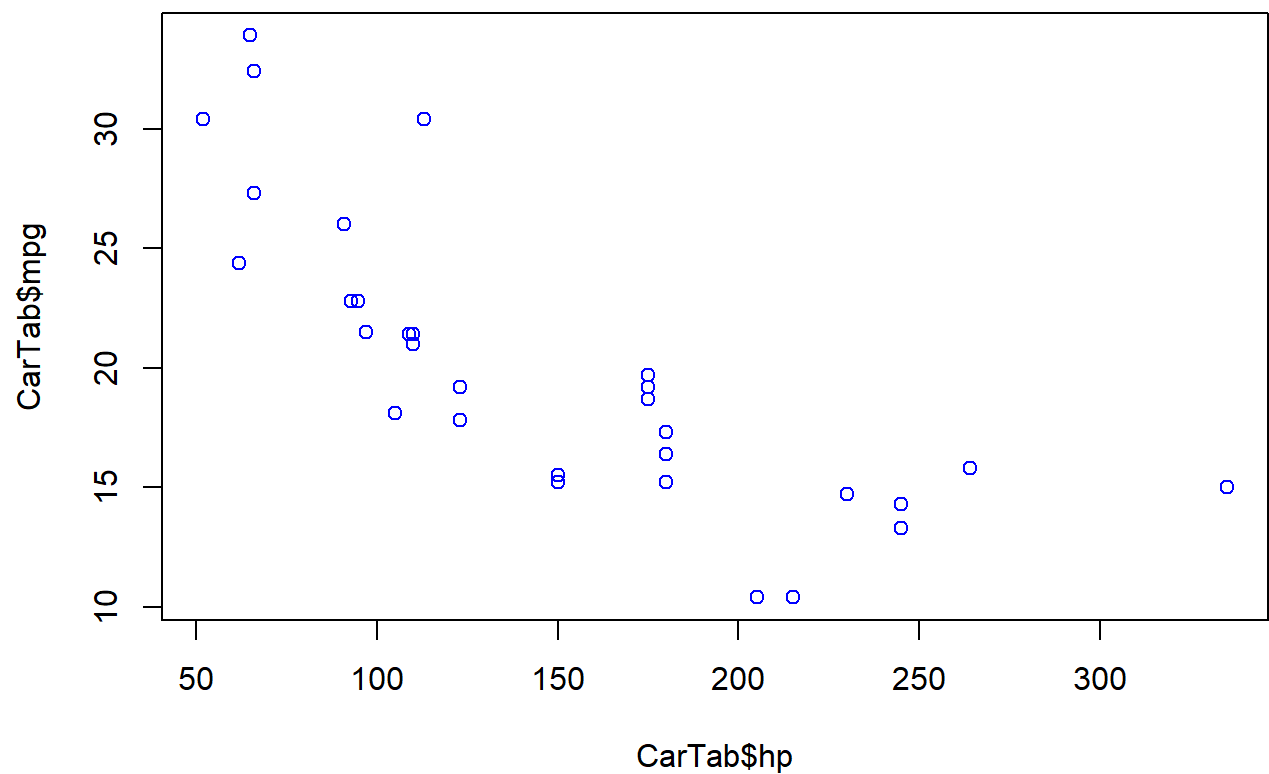
[1] 146.6875

Ejercicio 3

```
plot(CarTab$hp, CarTab$mpg)
```

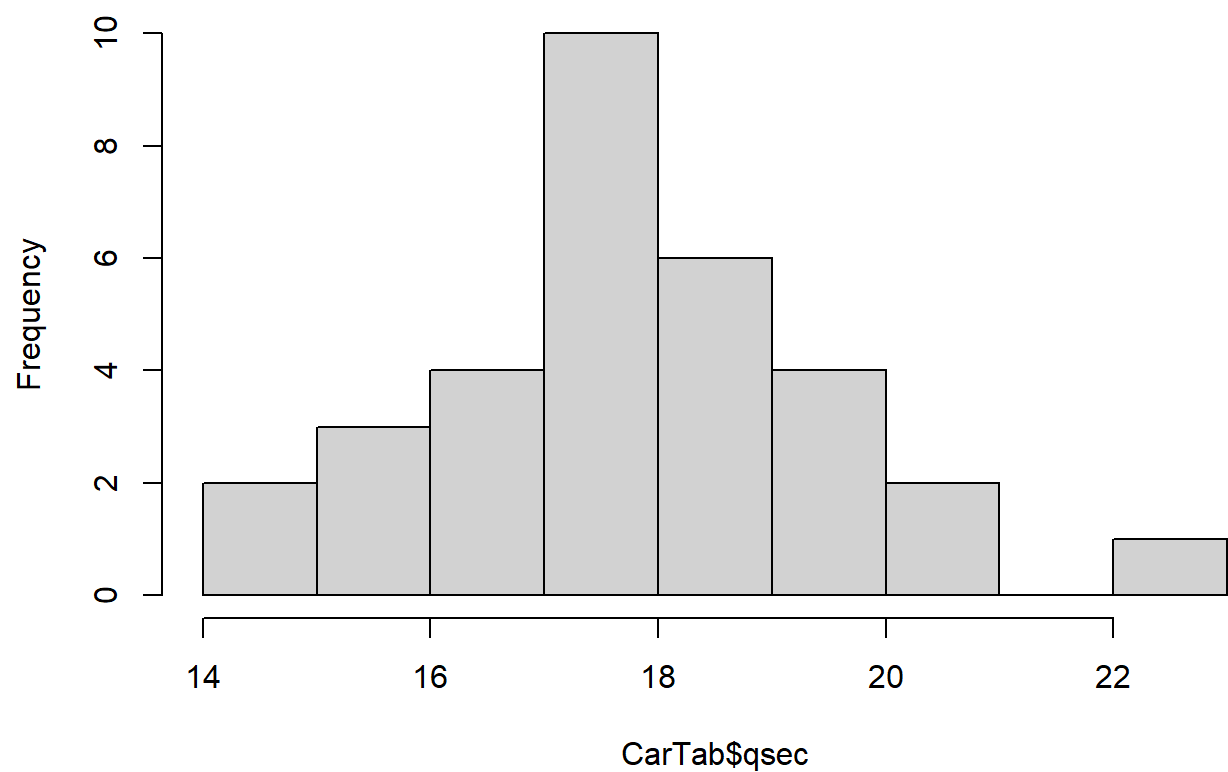


```
plot(CarTab$hp, CarTab$mpg, col = "blue")
```



```
hist(CarTab$qsec)
```

Histogram of CarTab\$qsec



Ejercicio 4

```
CarTab$rhpwat <- CarTab$hp/CarTab$wt  
CarTab$rhpwat
```

```
[1] 41.98473 38.26087 40.08621 34.21462 50.87209 30.34682 68.62745 19.43574  
[9] 30.15873 35.75581 35.75581 44.22604 48.25737 47.61905 39.04762 39.63864
```

[17] 43.03087 30.00000 32.19814 35.42234 39.35091 42.61364 43.66812 63.80208
[25] 45.51365 34.10853 42.52336 74.68605 83.28076 63.17690 93.83754 39.20863

```
CarTab$Rapido <- ifelse(CarTab$qsec<18,1,0)
CarTab
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

	rhpw	wt	Rapido
Mazda RX4	41.98	473	1
Mazda RX4 Wag	38.26	087	1
Datsun 710	40.08	621	0
Hornet 4 Drive	34.21	462	0
Hornet Sportabout	50.87	209	1
Valiant	30.34	682	0
Duster 360	68.62	745	1
Merc 240D	19.43	574	0
Merc 230	30.15	873	0
Merc 280	35.75	581	0
Merc 280C	35.75	581	0
Merc 450SE	44.22	604	1
Merc 450SL	48.25	737	1
Merc 450SLC	47.61	905	0
Cadillac Fleetwood	39.04	762	1
Lincoln Continental	39.63	864	1
Chrysler Imperial	43.03	087	1
Fiat 128	30.00	000	0
Honda Civic	32.19	814	0
Toyota Corolla	35.42	234	0
Toyota Corona	39.35	091	0
Dodge Challenger	42.61	364	1
AMC Javelin	43.66	812	1
Camaro Z28	63.80	208	1
Pontiac Firebird	45.51	365	1

Fiat X1-9	34.10853	0
Porsche 914-2	42.52336	1
Lotus Europa	74.68605	1
Ford Pantera L	83.28076	1
Ferrari Dino	63.17690	1
Maserati Bora	93.83754	1
Volvo 142E	39.20863	0

Ejercicio 5

```
Sub_CarTab <- CarTab[ , c("qsec", "Rapido")]
Sub_CarTab
```

	qsec	Rapido
Mazda RX4	16.46	1
Mazda RX4 Wag	17.02	1
Datsun 710	18.61	0
Hornet 4 Drive	19.44	0
Hornet Sportabout	17.02	1
Valiant	20.22	0
Duster 360	15.84	1
Merc 240D	20.00	0
Merc 230	22.90	0
Merc 280	18.30	0
Merc 280C	18.90	0
Merc 450SE	17.40	1
Merc 450SL	17.60	1
Merc 450SLC	18.00	0
Cadillac Fleetwood	17.98	1
Lincoln Continental	17.82	1
Chrysler Imperial	17.42	1
Fiat 128	19.47	0
Honda Civic	18.52	0
Toyota Corolla	19.90	0
Toyota Corona	20.01	0
Dodge Challenger	16.87	1
AMC Javelin	17.30	1
Camaro Z28	15.41	1
Pontiac Firebird	17.05	1
Fiat X1-9	18.90	0
Porsche 914-2	16.70	1
Lotus Europa	16.90	1
Ford Pantera L	14.50	1
Ferrari Dino	15.50	1
Maserati Bora	14.60	1
Volvo 142E	18.60	0

Ejercicio 6

```
CarTab <- mtcars
mod1 <- lm(qsec ~ hp + wt, data = CarTab)
summary(mod1)
```

Call:

lm(formula = qsec ~ hp + wt, data = CarTab)

Residuals:

Min	1Q	Median	3Q	Max
-1.8283	-0.4055	-0.1464	0.3519	3.7030

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	18.825585	0.671867	28.020	< 2e-16 ***


```
hp      -0.027310    0.003795   -7.197 6.36e-08 ***
wt       0.941532    0.265897    3.541 0.00137 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.09 on 29 degrees of freedom
Multiple R-squared:  0.652, Adjusted R-squared:  0.628
F-statistic: 27.17 on 2 and 29 DF,  p-value: 2.251e-07
```

❗ Para citar este archivo Quarto, por favor, considera este formato: Montero, M., Ortega, L. & Rodríguez, P. (2024). Metodología Cuantitativa II 2024 [Repositorio]. GitHub.
<https://github.com/monteromati/Metodologia-Cuantitativa-II-2024>