Exportar e importar datos

Importar

En R, puedes importar archivos desde diversas fuentes, incluyendo URL, ficheros en formato de tabla (por ejemplo, CSV), archivos Excel y archivos JSON. Aquí te muestro cómo hacerlo, junto con algunas funciones y librerías asociadas, así como las funciones `head`, `file.show`, y `readLines`.

1. Desde una URL:

Para importar datos desde una URL, puedes utilizar la función `read.table` o `read.csv` directamente. Si los datos están en formato CSV, podrías usar:

```
url <- "<https://ejemplo.com/datos.csv>"
datos <- read.csv(url)
```

2. Desde un Fichero en Formato de Tabla (CSV, por ejemplo):

Para importar datos desde un fichero en formato de tabla, puedes usar `read.table` o `read.csv`:

```
# Usando read.csv para un archivo CSV
datos_csv <- read.csv("ruta/al/archivo.csv")

# Usando read.table para un archivo de texto delimitado por tabulaciones (por ejemplo)
datos_tabulado <- read.table("ruta/al/archivo.txt", sep = "\\t", header = TRUE)
```

3. Desde un Archivo Excel:

Para trabajar con archivos Excel en R, la librería `readxl` es útil. Puedes instalarla y luego usar la función `read_excel`:

```
# Instalar la librería si no está instalada
# install.packages("readxl")

# Cargar la librería
library(readxl)
```

```
# Leer datos desde un archivo Excel
datos_excel <- read_excel("ruta/al/archivo.xlsx")
```

4. Desde un Archivo JSON:

Para archivos JSON, puedes usar la librería `jsonlite`. Instálala si no lo has hecho:

```
# Instalar la librería si no está instalada
# install.packages("jsonlite")

# Cargar la librería
library(jsonlite)

# Leer datos desde un archivo JSON
datos_json <- fromJSON("ruta/al/archivo.json")
```

Funciones Adicionales:

- `head`: Muestra las primeras filas de un conjunto de datos.

  ```
  head(datos_csv)  # Muestra las primeras filas del archivo CSV
  ```

- `file.show`: Muestra el contenido de un archivo en la consola.

  ```
  file.show("ruta/al/archivo.txt")  # Muestra el contenido del archivo de texto
  ```

- `readLines`: Lee líneas de un archivo de texto.

  ```
  lineas <- readLines("ruta/al/archivo.txt")  # Lee líneas del archivo de texto
  ```

Estas funciones y librerías te proporcionan herramientas versátiles para importar datos desde diferentes fuentes en R. Asegúrate de tener las librerías instaladas antes de usarlas (`install.packages("nombre_de_la_libreria")`).

Al utilizar funciones para abrir, leer o guardar ficheros en R, es esencial comprender los parámetros más importantes que puedes especificar para adaptar el comportamiento de estas funciones a tus necesidades. Aquí tienes una descripción de los parámetros clave para algunas de las funciones mencionadas:

1. `read.table` y `read.csv` (Leer Datos desde un Archivo de Texto):

- `file` **(obligatorio):** Ruta o nombre de archivo del fichero que se va a leer.
- `header` **(opcional):** Especifica si la primera fila del archivo contiene nombres de columnas (`TRUE` por defecto).
- `sep` **(opcional):** Carácter que delimita las columnas en el archivo (coma para CSV, por ejemplo).
- `quote` **(opcional):** Carácter usado para citar campos que pueden contener el delimitador (comillas dobles por defecto).
- `stringsAsFactors` **(opcional):** Indica si se deben tratar las cadenas de texto como factores (`TRUE` por defecto).

2. `read_excel` (Leer Datos desde un Archivo Excel):

- `path` **(obligatorio):** Ruta o nombre de archivo del fichero Excel que se va a leer.
- `sheet` **(opcional):** Nombre de la hoja de cálculo que se va a leer.
- `range` **(opcional):** Rango de celdas que se va a leer (p. ej., "A1:C10").

3. `fromJSON` (Leer Datos desde un Archivo JSON):

- `file` **o** `txt` **(uno de los dos es obligatorio):** Ruta o nombre de archivo del fichero JSON o una cadena de texto con formato JSON.

4. `write.table` y `write.csv` (Escribir Datos en un Archivo de Texto):

- `file` **(obligatorio):** Ruta o nombre de archivo donde se guardará el fichero.
- `sep` **(opcional):** Carácter que delimita las columnas en el archivo (coma para CSV, por ejemplo).
- `quote` **(opcional):** Carácter usado para citar campos que pueden contener el delimitador (comillas dobles por defecto).
- `row.names` **(opcional):** Indica si se deben incluir o no los nombres de fila en el archivo.

5. `write_excel` (Escribir Datos en un Archivo Excel):

- `x` **(obligatorio):** Datos que se van a escribir.

- `path` **(obligatorio):** Ruta o nombre de archivo donde se guardará el fichero Excel.
- `sheetName` **(opcional):** Nombre de la hoja de cálculo que se va a escribir.

Estos son algunos de los parámetros más comunes, pero cada función puede tener más opciones según las necesidades específicas. Consulta la documentación oficial de R y las librerías relevantes para obtener información detallada sobre los parámetros y sus opciones.

Puedes descargar un archivo HTML desde una URL en R utilizando la función `download.file`. Aquí tienes un ejemplo básico:

```
# URL del archivo HTML que deseas descargar
url <- "<https://www.ejemplo.com/archivo.html>"

# Ruta local donde deseas guardar el archivo descargado
ruta_local <- "ruta/donde/guardar/archivo.html"

# Descargar el archivo HTML desde la URL
download.file(url, destfile = ruta_local, mode = "wb")
```

En este ejemplo:

- `url`**:** Es la dirección URL del archivo HTML que deseas descargar.
- `ruta_local`**:** Es la ruta local en tu sistema donde deseas guardar el archivo descargado. Asegúrate de tener permisos para escribir en esa ubicación.
- `download.file`**:** Es la función que realiza la descarga del archivo. `destfile` es el parámetro que especifica la ruta local y `mode = "wb"` indica que el archivo se abrirá en modo binario.

Después de ejecutar este código, tendrás el archivo HTML descargado en la ruta local especificada. Puedes abrirlo y leerlo con funciones adicionales según tus necesidades.

Exportar

En R, puedes exportar datos a diferentes formatos, como ficheros de tabla (por ejemplo, CSV), archivos Excel y archivos JSON. Aquí te explico cómo hacerlo y las librerías asociadas, junto con ejemplos en cada caso.

1. Exportar a un Fichero de Tabla (CSV):

Para exportar a CSV, puedes usar las funciones `write.table` o `write.csv`. Estas funciones son muy similares, pero `write.csv` es una versión especializada para archivos CSV.

```
# Datos de ejemplo
datos <- data.frame(
  Nombre = c("Juan", "María", "Pedro"),
  Edad = c(25, 30, 22),
  Puntuacion = c(85, 92, 78)
)

# Exportar a CSV
write.csv(datos, file = "ruta/donde/guardar/datos.csv", row.names = FALSE)
```

Parámetros clave:

- `file`: Ruta o nombre de archivo donde se guardará el fichero.
- `row.names`: Indica si se deben incluir o no los nombres de fila en el archivo.

2. Exportar a un Archivo Excel:

Para exportar a un archivo Excel, puedes usar la librería `writexl` o `openxlsx`.

```
# Instalar y cargar la librería si no está instalada
# install.packages("writexl")
library(writexl)

# Datos de ejemplo
datos <- data.frame(
  Nombre = c("Juan", "María", "Pedro"),
  Edad = c(25, 30, 22),
  Puntuacion = c(85, 92, 78)
)

# Exportar a Excel
write_xlsx(datos, path = "ruta/donde/guardar/datos.xlsx")
```

Parámetros clave:

- `path`: Ruta o nombre de archivo donde se guardará el fichero Excel.

3. Exportar a un Archivo JSON:

Para exportar a un archivo JSON, puedes usar la librería `jsonlite`.

```
# Instalar y cargar la librería si no está instalada
# install.packages("jsonlite")
library(jsonlite)

# Datos de ejemplo
datos <- list(
  Nombre = c("Juan", "María", "Pedro"),
  Edad = c(25, 30, 22),
  Puntuacion = c(85, 92, 78)
)

# Exportar a JSON
write_json(datos, "ruta/donde/guardar/datos.json")
```

Parámetros clave:

- `json`: Datos que se van a escribir.
- `file`: Ruta o nombre de archivo donde se guardará el fichero JSON.