Cheatsheet: Carga de Archivos CSV en R

1. Carga de Archivos Utilizando Path Completo

Esta opción es **poco recomendada** debido a su baja reproducibilidad, especialmente cuando se comparte código con otros usuarios o se trabaja en diferentes equipos.

Sintaxis Básica

```
# Usando read.csv
datos <- read.csv("C:/Usuarios/NombreUsuario/Documentos/datos.csv")

# Usando readr::read_csv (del tidyverse, más rápido)
library(readr)
datos <- read_csv("C:/Usuarios/NombreUsuario/Documentos/datos.csv")
```

Manejo de Rutas en Windows

En Windows, las rutas utilizan por defecto backslashes (`\`), pero R interpreta estos como caracteres de escape. Por lo tanto, existen dos opciones:

1. Invertir las barras (usar forward slashes)

```
datos <- read.csv("C:/Usuarios/NombreUsuario/Documentos/datos.csv")
```

2. Usar doble backslash

```
datos <- read.csv("C:\\Usuarios\\NombreUsuario\\Documentos\\datos.csv")
```

3. Usar el prefijo r para raw strings (R ≥ 4.0.0)

```
datos <- read.csv(r"(C:\Usuarios\NombreUsuario\Documentos\datos.csv)")
```

Cómo Obtener el Path Completo de un Archivo

En Windows 10/11:

1. Navega hasta el archivo en el Explorador de Archivos
2. Mantén presionada la tecla Shift y haz clic derecho sobre el archivo
3. Selecciona “Copiar como ruta de acceso” en el menú contextual
4. La ruta se copiará al portapapeles con comillas dobles

Alternativamente:

1. Haz clic derecho sobre el archivo
2. Selecciona “Propiedades”
3. La ruta completa aparecerá en “Ubicación” (tendrás que añadir el nombre del archivo manualmente)

En macOS:

1. Haz clic derecho sobre el archivo en Finder
2. Mantén presionada la tecla Option
3. La opción "Copiar [nombre del archivo] como ruta de acceso" aparecerá en el menú
4. Selecciónala para copiar la ruta completa al portapapeles

2. Carga de Archivos Usando file.path y Working Directory

Esta opción es **más recomendada** porque mejora la reproducibilidad del código y facilita la organización de los proyectos.

Concepto de Working Directory

El directorio de trabajo (working directory) es la carpeta desde la cual R leerá y escribirá archivos por defecto. Al utilizar rutas relativas en lugar de absolutas, el código se vuelve más portable.

Verificar y Definir el Working Directory

```
# Verificar el working directory actual
getwd()

# Definir un nuevo working directory
setwd("C:/Usuarios/NombreUsuario/Proyectos/ProyectoActual")
```

Utilizar file.path para Rutas Relativas

```
# Definir una variable para la ruta de entrada
instub <- "datos"

# Cargar archivo usando file.path
datos <- read.csv(file.path(instub, "datos.csv"))
```

Esto buscará el archivo en la carpeta "datos" dentro del directorio de trabajo actual.

Ventajas de usar file.path

1. **Independencia de plataforma**: file.path usa automáticamente el separador de directorios apropiado para cada sistema operativo
2. **Código más limpio**: separa la lógica de las rutas del código de análisis
3. **Mayor reproducibilidad**: facilita cambiar las ubicaciones de los archivos sin modificar todo el código

Ejemplo de Estructura Recomendada

```
# Al inicio del script
base_dir <- getwd()
data_dir <- file.path(base_dir, "datos")
output_dir <- file.path(base_dir, "resultados")

# Cargar datos
datos <- read.csv(file.path(data_dir, "datos.csv"))
```

```
# Guardar resultados
write.csv(resultados, file.path(output_dir, "resultados.csv"))
```

Consejos Adicionales

Paquetes que Facilitan la Gestión de Rutas

- **here**: Simplifica las referencias a archivos dentro de un proyecto

```
library(here)
datos <- read.csv(here("datos", "datos.csv"))
```

- **fs**: Proporciona funciones más intuitivas para trabajar con el sistema de archivos

```
library(fs)
path_data <- path(getwd(), "datos", "datos.csv")
datos <- read.csv(path_data)
```

Uso de Parámetros al Cargar CSVs

```
datos <- read.csv(file.path(data_dir, "datos.csv"),
                  header = TRUE,          # Primera fila contiene nombres de columnas
                  sep = ",",              # Separador de columnas
                  dec = ".",              # Separador decimal
                  stringsAsFactors = FALSE, # No convertir strings a factores automáticam
                  na.strings = c("NA", "", "N/A") # Valores que se considerarán como NA
)
```

Para Archivos Excel

Para archivos Excel, se pueden utilizar los paquetes `readxl` o `openxlsx`:

```
library(readxl)
datos <- read_excel(file.path(data_dir, "datos.xlsx"), sheet = 1)
```