Trabajar con proyectos en RStudio

Trabajar con proyectos de RStudio no solo hace tus análisis más ordenados y reproducibles, también hacen tu vida más simple.

Al comienzo posiblemente tengan un script y uno o dos archivos con datos, pero es posible que rápidamente te encuentres con una docena de archivos con nombres parecidos pero que pertenecen a análisis totalmente disntintos. Antes de que la cosa comience a complicarse te proponemos trabajar con proyectos.

¿Qué ventajas tiene?

- Te permite "cuidar" los datos que usas al ordenarnos en carpetas que diferencien entre la versión original o cruda y los datos limpios o los resultados finales.
- Te permite compartir tu trabajo fácilmente con otras personas. Solo tendrías que compartir la carpeta del proyecto sabiendo que incluye todo lo necesario para que cualquier persona reproduzca tu análisis.
- Te permite publicar de manera ordenada tu código si vas a presentar o publicar tu trabajo.
- Te permite continuar con lo que estabas haciendo hace una semana o hace un mes como si el tiempo no hubiera pasado. De alguna manera es un regalo para tu yo futuro.

Primer desafío: Crea un nuevo proyecto en RStudio

1. Hacé click en el menú "Archivo" ("File") y luego en "Nuevo Proyecto" ("New > Project").
2. Hacé click en "Nueva Carpeta" ("New Directory").
3. Hacé click en "Nuevo Proyecto" ("New Project").
4. Escribí el nombre de la carpeta que alojará a tu proyecto, por ejemplo > "mi_proyecto"
5. Si aparece (y sabés usarlo), seleccioná "Crear un repositorio de git" ("Create a > git repository").
6. Hacé click en "Crear Proyecto" ("Create Project").

Si todo salió bien, ahora deberías tener una nueva carpeta que se llama *mi_proyecto*. Pero si bien es una carpeta común y corriente, le llamamos proyecto porque además contiene un archivo con el mismo nombre *mi_proyecto.Rproj* (o solo *mi_proyecto* si en tu computadora no ves la extensión de los archivos).

Abrir un proyecto

La manera más simple de abrir un proyecto es abriendo la carpeta que lo contiene y haciendo doble click sobre el archivo *mi_proyecto.Rproj*. Al hacer esto se abrirá RStudio y la sensión de R en la misma carpeta. Por defecto, R buscará cualquier archivo que quieras abrir en esa misma ubicación. Esto ayuda a mantener tu trabajo ordenado y que luego sea simple retomar o compartir lo que hiciste.

RStudio permite tener varios proyectos abiertos, y esto es posible porque justamente cada proyecto tiene su propia carpeta. Si en algún momento trabajas con proyectos en paralelo vas a poder hacerlo sin que el código o los resultados de un análisis interfieran con otro.

Segundo desafío: Abrí tu nuevo proyecto desde el explorador de archivos

1. Cerrá RStudio
2. Desde el explorador de archivos, buscá la carpeta donde creaste tu proyecto.
3. Hacé doble click en el archivo que tiene el nombre de tu proyecto (y que termina > con *.Rproj*) que encontrarás en esa carpeta.

¿Cómo se organiza?

No existe una "mejor" forma de organizar un proyecto pero acá van algunos principios generales que nos hacen la vida más simple::

- **Tratar los datos como sólo de lectura** Es posible que la toma de los datos que querés analizar te haya costado mucho trabajo, o te haya costado conseguirlos. Trabajar con datos de forma interactiva (por ejemplo, en Excel) tiene la ventaja de permitirte hacer algunos análisis rápidamente pero al mismo tiempo tiena la desventaja de que esos datos pueden ser modificados fácilmente. Esto significa que a veces no conozcas de la procedencia de los datos, o no recuerdes como los modificastes desde que los obtuviste. Por lo tanto, es una buena idea tratar los datos como "sólo de lectura" y nunca modificar los archivos originales.
- **Limpieza de datos** En muchos casos tus datos estarán "sucios", necesitarán un preprocesamiento importante para organizarlos en un formato que R (o cualquier otro lenguaje de programación) pueda analizados fácilmente. Esta tarea se denomina a veces "amasado" o "masticado de datos". Es una buena costumbre guardar el código que te permitió limpiar estos datos por si los volvieras a necesitar. También es recomendable guardar esa versión de los datos limpios, de "sólo lectura", para que puedas usarlos en tu análisis sin necesidad de repetir cada vez todo el proceso de limpieza de los datos.
- **Tratar las salidas o resultados generados como descartables** Cualquier resultado (gráficos, tablas, valores) debe poder repetirse o rehacerse a partir del código guardado. Si bien las pruebas rápidas para *ver si el código funciona* se pueden hacer en la consola, es importante guardar el código que genera los resultados y asegurarnos de que sean reproducibles. Aún mejor, si organizas esos resultados en distintas sub-carpetas, luego tendrás todo aún más ordenado.

{here} es la solución

Si tenés alguna experiencia programando con R es posible que tengas estás lineas al comienzo de alguno de tus scripts o si nunca las usaste, seguro viste que alguien más lo hacia:

```
setwd("/Users/pao/una_carpeta/al/proyecto_importante")
rm(list = ls())
```

La primera línea *setea* o le avisa a R cual será la carpeta donde va a trabajar. Con el uso de proyectos esto está prácticamente solucionado porque al abrir el proyecto ya sea desde el explorador de archivos haciendo doble click en el archivo con extensión *Rproj* o desde RStudio, R sabrá que ese directorio será el de trabajo.

Pero también dijimos que era una buena práctica organizar los datos en una subcarpeta "datos", los informes en otra y tal vez las figuras en una subcarpeta distinta dentro del proyecto. ¿Cómo hacemos para que R lea un archivo que no está en la carpeta de trabajo? Podríamos escribir el camino hacia ese archivo, por ejemplo `"datos/mi_archivo_de_datos.csv"` pero si queremos compartir el código a otra persona que tal vez tiene un sistema operativo distinto y usa la barra invertida \ va a estar en problemas al intentar correr esa línea.

Sale al rescate el paquete `{here}`, que funciona independientemente del sistema operativo. Su función principal `here()` recibe como argumentos el camino hacia el archivo que se quiere leer, siempre entre comillas y separados por comas, así:

```
mis_datos <- read_csv(here("datos", "mi_archivo_de_datos.csv"))
```

Internamente este paquete es suficientemente inteligente para identificar cual es el directorio de trabajo (por ejemplo detectando que hay un archivo *.Rproj*) y busca a partir de ahí la subcarpeta "datos" y adentro de ella el archivo "mi_archivo_de_datos.csv".

La segunda línea se usa para borrar los elementos que creamos en el análisis normalmente cuando cambiamos de tema o empezamos a trabajar con algo distinto. Esto está bien porque no queremos arrastrar analisis que hicimos en un proyecto a otro, necesitamos que sean autocontenidos y *reproducibles*. El problema es que este comando **no** borra los paquetes activados o las opciones usuario que hayamos seteado.

Borrón y cuenta nueva... todos los días!

¿Cómo nos aseguramos de que el análsis sea realmente reproducible? Esta es una pregunta bastante amplia y hay muchas herramientas para resolver este problema. Por ahora nos vamos a concentrar en que al menos en tu computadora puedas repetir los cálculos o el análisis desde cero. Y además de organizar proyectos y no modificar los datos originales, ¿cómo podés asegurarte de que guardaste todo el código que estuviste escribiendo y usaste? La manera más directa es reiniciar la sesión de R y correr el código de nuevo, si da error o no devuelve lo que esperabas significa que te faltó guardar algún paso.

> *Tip: Podés reiniciar la sesión de R con el atajo* `Ctrl+Shif+F10`

Esto puede pasar si por ejemplo leemos una base de datos en memoria pero no guardamos el código que lo hace. Mientras estemos trabajando, R tendrá esa base de datos en memoria y podremos hacer cálculos y gráficos. Por defecto además RStudio va a recordar las variables que estes usando mañana o pasado en un archivo oculto (.RData) a menos que le indiques lo contrario. Y si bien suena práctico volver a R al otro día y tener el análisis tal cual lo dejamos, esto puede significar que nunca nos demos cuenta que nos faltó guardar una línea de código clave en nuestro análisis.

Tercer desafío: Configurá RStudio

1. Hacé click en el menú "Herramientas ("Tools") y luego "Opciones globales" ("Global > Options").
2. Destildá la opción "Recuperar .RData al inicio de la sesión" ("Restore .RData into > workspace at startup").
3. Hacé click en "Aplicar" ("Apply").
