Usar Python con R con reticulate : : GUÍA RÁPIDA

El paquete reticulate le permite usar Python y R juntos sin problemas en el código de R, en documentos de R Markdown y en el IDE de RStudio.

Python en R Markdown

(Opcional) Compile Python env para usar. Las versiones de knitr >= 1.18 usarán automáticamente el motor reticulate para los fragmentos de Python. Ver **?reticulate::eng_python** para obtener una lista de las opciones de trozos de knitr compatibles.

Sugerir el entorno de Python para usar, en los fragmentos de código.

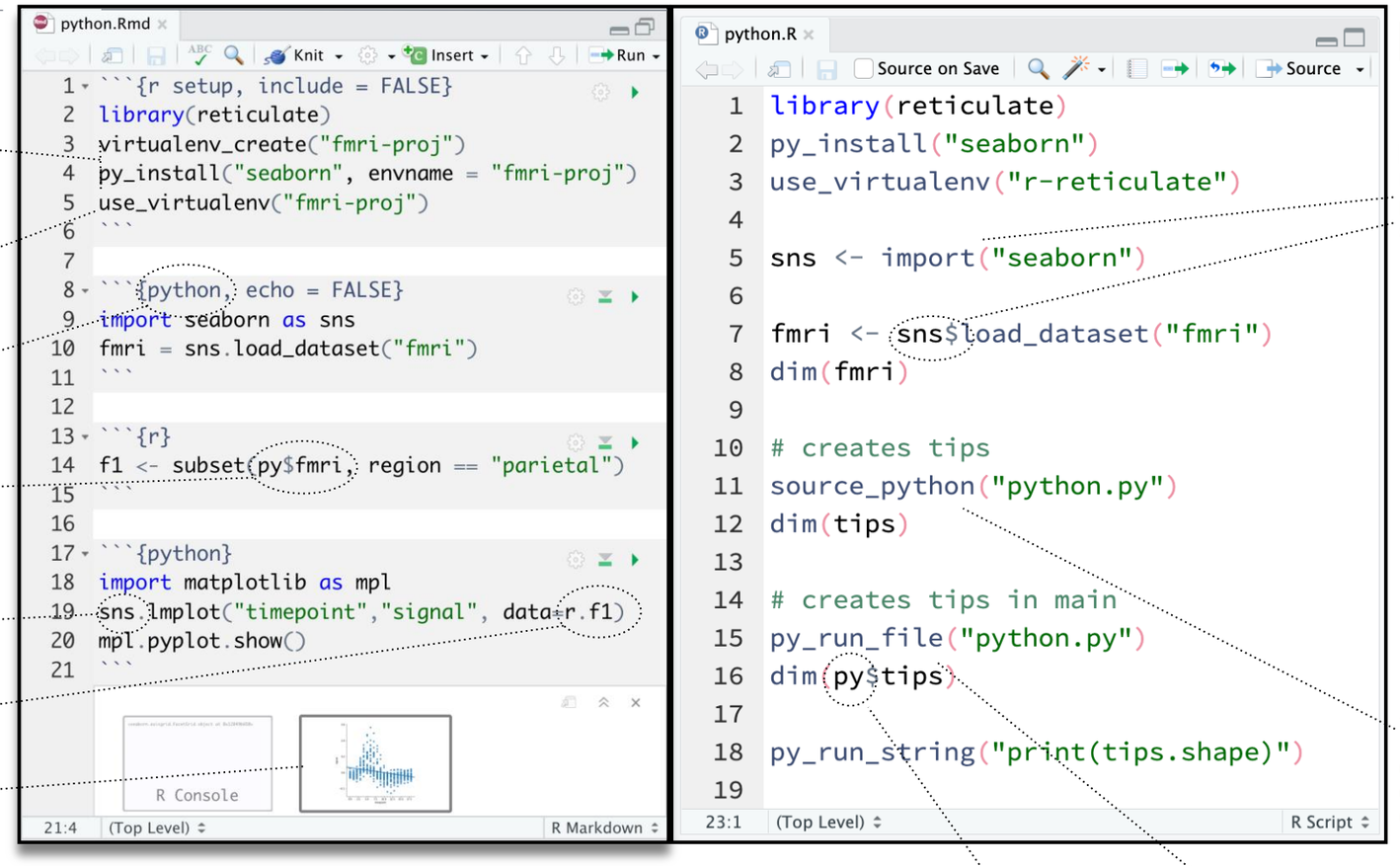
Comience los fragmentos de Python con '"{python}. Las opciones de fragmentos, como echo, include, etc., funcionan según lo esperado.

Utilice el objeto py para acceder a los objetos creados en fragmentos de Python desde fragmentos de R.

Todos los fragmentos de Python se ejecutan dentro de una sesión única de Python para que tenga acceso a todos los objetos creados en fragmentos anteriores.

Utilice el objeto r para acceder a los objetos creados en fragmentos de R desde fragmentos de Python.

La salida se muestra debajo del fragmento, incluyendo gráficas matplotlib.



Python en R

Llame a Python desde código R de tres maneras:

IMPORTAR MÓDULOS DE PYTHON

Use **import()** para importar cualquier módulo de Python. Acceda a los atributos de un módulo con **$**.

- **import(**module, as = NULL, convert = TRUE, delay_load = FALSE**)** Importar un módulo de Python. Si convert = TRUE, Los objetos de Python se convierten a sus tipos de R equivalentes. Además, **import_from_path()**. import("pandas")
- **import_main(**convert = TRUE**)**

Importar el módulo principal, donde Python ejecuta el código de forma predeterminada. import_main()

- **import_builtins(**convert = TRUE**)** Importar las funciones integradas de Python. import_builtins()

ARCHIVOS PYTHON DE ORIGEN

Utilice source_python() para obtener un script de Python y hacer que las funciones y los objetos de Python que crea estén disponibles en el entorno de R que realiza la llamada.

- **source_python(**file, envir = parent.frame(), convert = TRUE**)** Ejecutar una secuencia de comandos de Python y asignar objetos a un entorno de R especificado. source_python("file.py")

EJECUTAR CÓDIGO PYTHON

Ejecute código Python en el módulo principal de Python con **py_run_file()** o **py_run_string()**.

- **py_run_string(**code, local = FALSE, convert = TRUE**)** Ejecute código Python (pasado como una cadena) en el módulo principal. py_run_string("x = 10"); py$x
- **py_run_file(**file, local = FALSE, convert = TRUE**)** Ejecute el archivo Python en el módulo principal. py_run_file("script.py")
- **py_eval(**code, convert = TRUE**)** Correr una expresión de Python, devuelve elresultado. py_eval("1 + 1")

Acceda a los resultados, y a cualquier otra cosa en el módulo principal de Python, con **py**.

- **py** Un objeto R que contiene el módulo principal de Python y los resultados almacenados allí. py$x

Conversión de objetos

Reticulate proporciona una conversión automática integrada entre Python y R para muchos tipos de Python.

R ⟷	Python
Vector de un solo elemento	Escalar
Vector multielemento	Lista
Lista de varios tipos	Tuple
Lista con nombre	Dict
Matriz/Arreglo	NumPy ndarray
Marco de Datos	Pandas DataFrame
Función	Python function
NULL, TRUE, FALSE	None, True, False

O, si lo desea, puede convertir manualmente con

py_to_r(x**)** Convierta un objeto de Python en un objeto de R. Además **r_to_py()**.

tuple(..., convert = FALSE**)** Cree un Python tuple. tuple("a", "b", "c")

Consejo: Para indexar objetos de Python que comiencen en 0, use números enteros, por ejemplo, 0L

dict(..., convert = FALSE**)** Cree un objeto de diccionario de Python. Además **py_dict()** para crear un diccionario que use objetos de Python como claves. dict(foo = "bar", index = 42L)

np_array(data, dtype = NULL, order = "C"**)** Crea NumPy arreglos. np_array(c(1:8), dtype = "float16")

array_reshape(x, dim, order = c("C", "F"**)** Cambiar la forma de un arreglo de Python. x <- 1:4; array_reshape(x, c(2, 2))

py_func(f**)** Encapsular una función de R en una función de Python con la misma firma. py_func(xor)

py_main_thread_func(f**)** Cree una función a la que siempre se llamará en el subproceso principal.

iterate(it, f = base::identity, simplify = TRUE**)** Aplique una función de R a cada valor de un iterador de Python o devuelva los valores como un vector de R, drenando el iterador a medida que avanza. Además **iter_next()** y **as_iterator()**.

py_iterator(fn, completed = NULL**)** Creación de un iterador de Python a partir de una función de R.
seq_gen <- function(x){
n <- x; function() {n <<- n + 1; n}};
py_iterator(seq_gen(9))

Ayudantes

py_capture_output(expr, type = c("stdout", "stderr"**))** Capture y devuelva la salida de Python. Además **py_suppress_warnings()**.

py_get_attr(x, name, silent = FALSE**)** Obtener un atributo de un objeto de Python. Además **py_set_attr()**, **py_has_attr()**, y **py_list_attributes()**.

py_help(object**)** Abra la página de documentación para obtener un
Objeto Python. py_help(sns)

py_last_error() Obtenga el último error de Python encontrado. Además **py_clear_last_error()** para borrar el último error. py_last_error()

py_save_object(object, filename, pickle = "pickle", ...**)** Guarde y cargue objetos de Python con pickle. Además **py_load_object()**. py_save_object(x, "x.pickle")

with(data, expr, as = NULL, ...**)** Evaluar una expresión dentro de un gestor de contexto de Python.
py <- import_builtins();
with(py$open("output.txt", "w") %as% file,
{ file$write("Hello, there!")})

Python en el IDE

Requiere reticulate más RStudio v1.2+. Algunas funciones requieren v1.4+.

Resaltado de sintaxis para scripts y fragmentos de Python.

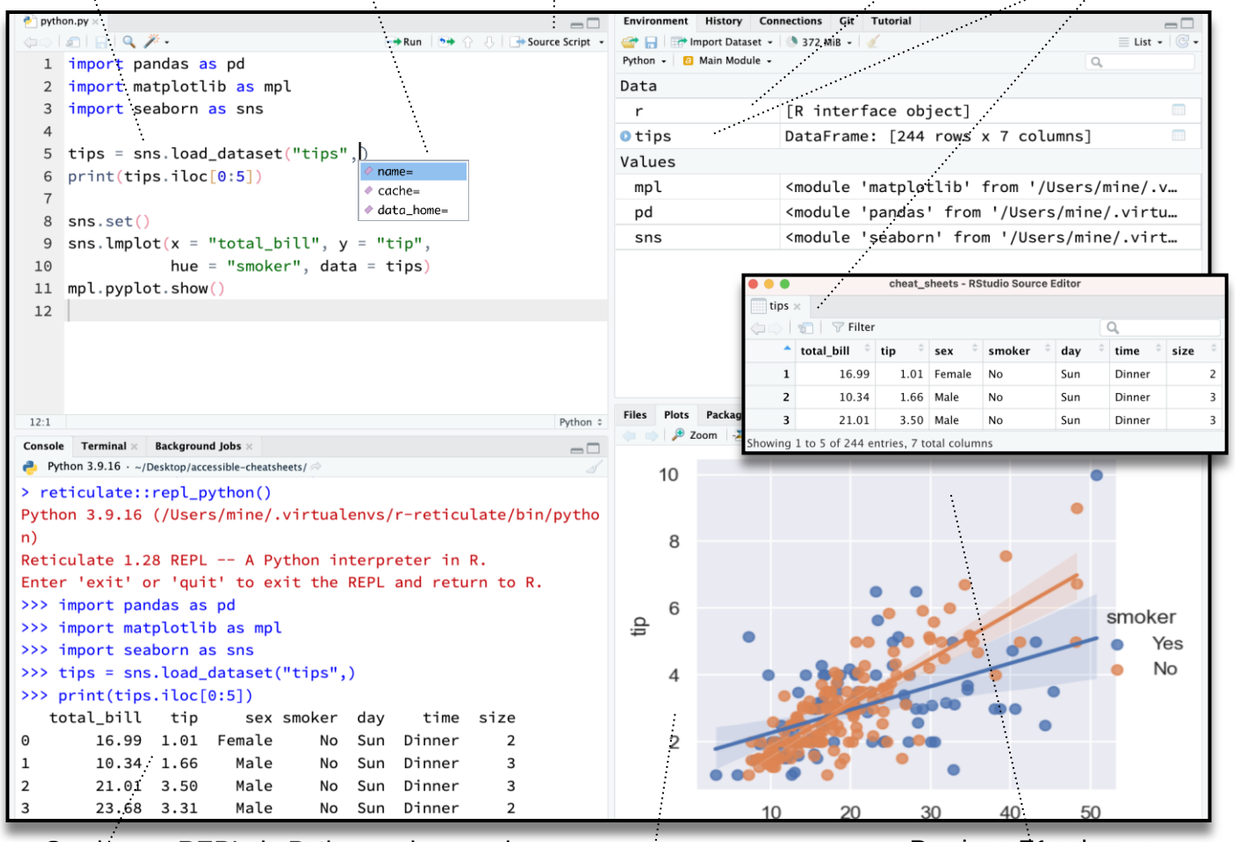
Finalización de tabulación para funciones y objetos de Python (y módulos de Python importados en scripts de R).

Scripts de Python de origen.

Ejecute el código Python línea por línea con Cmd + Enter (Ctrl + Enter).

Vea los objetos de Python en el panel Entorno.

Vea objetos de Python en el Visor de datos.



Se abre un REPL de Python en la consola cuando se ejecuta código de Python con un método abreviado de teclado. Escriba exit para cerrar.

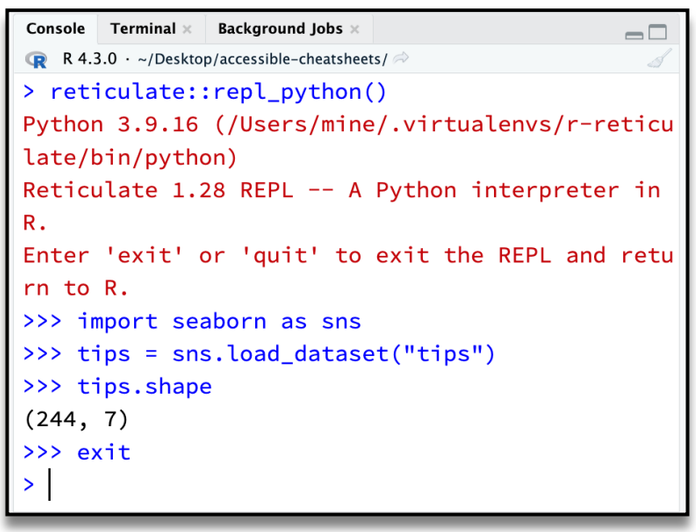
Los gráficos de Matplotlib se muestran en el panel Gráficos.

Presione F1 sobre un símbolo de Python para mostrar el tema de ayuda de ese símbolo.

Python REPL

Un REPL (bucle de lectura, evaluación, impresión) es una línea de comandos en la que puede ejecutar código Python y ver los resultados.

1. Ábralo en la consola con repl_python() o ejecutando código en un script de Python con **Cmd + Enter** (**Ctrl + Enter**).
   - **repl_python(**module = NULL, quiet = getOption("reticulate.repl.quiet", default = FALSE), input = NULL**)** Inicie un REPL de Python. Ejecute exit para cerrar. repl_python()
2. Escriba los comandos en **>>>** prompt.
3. Presione Intro para ejecutar el código.
4. Escriba exit para cerrar y volver a la consola de R.



Configurar Python

Reticulate se enlaza a una instancia local de Python cuando se llama por primera vez a import() directa o implícitamente desde una sesión de R. Para controlar el proceso, busque o cree la instancia de Python que desee. A continuación, sugiera su instancia para reticular. Reinicie R para desvincular.

Buscar Python

- **install_python(**version, list = FALSE, force = FALSE**)** Descargar e instalar Python. install_python("3.9.16")
- **py_available(**initialize = FALSE**)** Compruebe si Python está disponible en su sistema. Además, **py_module_available()** y **py_numpy_module().** py_available()
- **py_discover_config()** Devuelve la instalación detectada de Python. Use **py_config()** para comprobar qué versión se ha cargado. py_config()
- **virtualenv_list()** Enumere todos los entornos virtuales disponibles. Además, **virtualenv_root()**. virtualenv_list()
- **conda_list(**conda = "auto"**)** Enumere todos los entornos de Conda disponibles. Además, **conda_binary()** y **conda_version()**. conda_list()

Crear un entorno de Python

- **virtualenv_create(**envname = NULL, ...**)** Cree un nuevo entorno virtual. virtualenv_create("r-pandas")
- **conda_create(**envname = NULL, ...**)** Cree un nuevo entorno de Conda. conda_create("r-pandas", packages = "pandas")

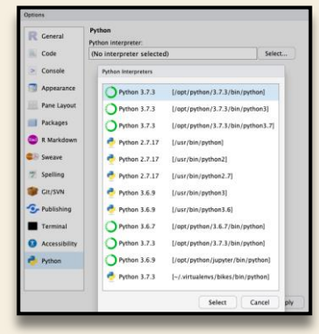
Instalar paquetes

Instale los paquetes de Python con R (abajo) o el shell:
pip install SciPy
conda install SciPy

- **py_install(**packages, envname, ...**)** Instala paquetes de Python en un entorno de Python. py_install("pandas")
- **virtualenv_install(**envname, packages, ...**)** Instale un paquete dentro de un virtualenv. Además, **virtualenv_remove()**. virtualenv_install("r-pandas", packages = "pandas")
- **conda_install(**envname, packages, ...**)** Instale un paquete dentro de un entorno de Conda. Además, **conda_remove()**. conda_install("r-pandas", packages = "plotly")

Sugerir un entorno para usar

Establecer un intérprete de Python predeterminado en las opciones globales o de proyecto del IDE de RStudio.



Ve a **Tools > Global Options... > Python** para Global Options.

Dentro de un proyecto, vaya a **Tools > Project Options... >Python**.

De lo contrario, reticulate examina las instancias del equipo en el siguiente orden:

1. La instancia a la que hace referencia la variable de entorno **RETICULATE_PYTHON** (si se especifica). **Tip: set in .Renviron file.**
   - **Sys.setenv(**RETICULATE_PYTHON = PATH**)** Establezca el binario predeterminado de Python. ¡Persiste a lo largo de las sesiones! Deshacer con **Sys.unsetenv()**. Sys.setenv(RETICULATE_PYTHON = "/usr/local/bin/python")
2. Las instancias a las que hace referencia con **use_** functions si se llama before import().
   - **use_python**(python) Ruta de acceso a un binario de Python. use_python("/usr/local/bin/python")
   - **use_virtualenv**(virtualenv) Ruta o nombre de un virtualenv de Python. use_virtualenv("~/myenv") use_virtualenv("r-keras")
3. Un entorno virtual que se encuentra en el directorio de trabajo actual: "./.venv"
4. Entornos que llevan el nombre del módulo importado. e.g. ~/.virtualenvs/r-scipy/ for import("scipy")
5. El paquete predeterminado virtualenv, "r-reticulate".
6. En la ubicación del binario de Python descubierto en el sistema PATH(i.e. Sys.which("python"))