R & Python

Ramon Ceballos

14/1/2021

LIBRERIA Reticulate

```
#Cargo la libreria reticulate instalada en R
library(reticulate)
#Con este comando cargo más de 200 librerías de python que podría usar en este script de R
use_python("C:/Users/usuario/anaconda3/python.exe")
#usa el comando de abajo:
#py_install("nombre del paquete a instalar")
#se ha importado la librería de python os a este script, guardandola en una variable
os <- import ("os")
#Se invoca el módulo listdir con el $ para ver los directorios que hay en el lugar en el que está guard
os$listdir(".")
```

```
##  [1] "01-EjemploRMD.html"
##  [2] "01-EjemploRMD.pdf"
##  [3] "01-EjemploRMD.Rmd"
##  [4] "02-Documentacion.html"
##  [5] "02-Documentacion.pdf"
##  [6] "02-Documentacion.Rmd"
##  [7] "03- R & Python empleado Reticulate.Rmd"
##  [8] "03--R---Python-empleado-Reticulate.html"
##  [9] "03--R---Python-empleado-Reticulate.pdf"
## [10] "03--R---Python-empleado-Reticulate.Rmd"
```

Se pueden llamar ficheros de python (.py) en R una vez se ha cargado Reticulate en un script de R.

Para poder llamar a funciones de dichos ficheros se emplea:

1. Comando source_python("nombre del fichero"). Así importamos dicho fichero.

Para llamar ficheros que contengan clases (librerias), se emplea el comando py_run_file ("nombre de archivo .py") para luego ejecutar el main (programa principal de una clase).

```
#Se importa la libreria de Python llamada Numpy
np <- import("numpy", convert = FALSE)

x <- np$array(c(1:4))
#Hace las sumas acumuladas del paquete numpy
sum <- x$cumsum()
#Da el resultado igual que en python
print (sum)
```

```
## [ 1  3  6 10]
```

```
#Para obtener un resultado de R se emplea:
py_to_r(sum)
```

```
## [1]  1  3  6 10
```

AYUDA EN R

Comando help(function) Ej.: help (py_to_r)

AYUDA EN PYTHON

Comando py_help()

```
py_help(os$chdir)
```

ARRAYS

Se puede crear objetos Python desde R

```
A <- np_array(c(1:10), dtype="float16")
A
```

```
## [ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
```

Escribir chunk en Python

```
import numpy as np
import pandas as pd
```

Leer datos en R y Python

```
datos <- iris
head (datos)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
#paso los datos de r a Python
datos_py <- r_to_py(datos)
```

Los dataset de R empieza en 1 mientras que los dataset de Python empiezan en 0

```
import numpy as np
import pandas as pd

r.datos_py.head()
```

```
##    Sepal.Length  Sepal.Width  Petal.Length  Petal.Width Species
## 0           5.1          3.5           1.4          0.2  setosa
## 1           4.9          3.0           1.4          0.2  setosa
## 2           4.7          3.2           1.3          0.2  setosa
## 3           4.6          3.1           1.5          0.2  setosa
## 4           5.0          3.6           1.4          0.2  setosa
```

Sparse matrix

Son matrices que se usan mucha y son la gran mayoria ceros salvo una posicion.

Depende en R del paquete Matrix.

```
library(Matrix)
n <- 6
set.seed(123)
sparse_mat <- sparseMatrix(
  i = sample(n, n, replace = F),
  j = sample(n, n, replace = F),
  x = runif(0:n),
  dims = c(n,n)
)
```

```
## Warning in sparseMatrix(i = sample(n, n, replace = F), j = sample(n, n, :
## length(i) is not a multiple of length(x)
```

```
sparse_mat
```

```
## 6 x 6 sparse Matrix of class "dgCMatrix"
##
## [1,] .         .          0.8895393 .         .        .
## [2,] .         0.04205953 .         .         .        .
## [3,] .         .          .         .         0.899825 .
## [4,] .         .          .         .         .        0.3279207
## [5,] 0.9545036 .          .         .         .        .
## [6,] .         .          .         0.2460877 .        .
```

```
sparse_mat_py <- r_to_py(sparse_mat)
```

Paso la variable de R a Python

```
r.sparse_mat_py
```

```
## <6x6 sparse matrix of type '<class 'numpy.float64'>'
##  with 6 stored elements in Compressed Sparse Column format>
```

Paso la variable de Python a R, pudiendo haberla modificado en Python

```
py_to_r(sparse_mat_py)
```

```
## 6 x 6 sparse Matrix of class "dgCMatrix"
##
## [1,] .         .          0.8895393 .         .        .
## [2,] .         0.04205953 .         .         .        .
## [3,] .         .          .         .         0.899825 .
## [4,] .         .          .         .         .        0.3279207
## [5,] 0.9545036 .          .         .         .        .
## [6,] .         .          .         0.2460877 .        .
```