Resum curs R

Curso Estadística descriptiva Udemy

Roger Ruiz i Andrés

6 de mayo, 2020

Índice

1. Trabajando con R

2. Documentación con R Markdown

3. Librería `reticulate`

Para usar `Python` en `R`, primero hay que cargar la librería `reticulate` e indicar que versión de `Python` se debe usar. Es recomendable que se use la misma que la que se usa cuando se ejecuta **Anaconda**.

```
library(reticulate)
use_python("/Users/rogerruiziandres/opt/anaconda3/bin/python3", required = TRUE)
py_config()
```

```
## python:         /Users/rogerruiziandres/opt/anaconda3/bin/python3
## libpython:      /Users/rogerruiziandres/opt/anaconda3/lib/libpython3.7m.dylib
## pythonhome:     /Users/rogerruiziandres/opt/anaconda3:/Users/rogerruiziandres/opt/anaconda3
## version:        3.7.4 (default, Aug 13 2019, 15:17:50)  [Clang 4.0.1 (tags/RELEASE_401/final)]
## numpy:          /Users/rogerruiziandres/opt/anaconda3/lib/python3.7/site-packages/numpy
## numpy_version:  1.17.2
##
## NOTE: Python version was forced by use_python function
```

```
knitr::knit_engines$set(python = reticulate::eng_python)
```

Una vez cargada,

1. Podemos ejecutar *chunks* de `Python`:

```
#Chunk Python
def fsuma(x):
  return x + 2
```

```
#Chunk Python
fsuma(2)
```

```
## 4
```

2. Podemos invocar scrips de `Python`, donde podamos tener funciones guardadas, desde un *chunk* de `R`:

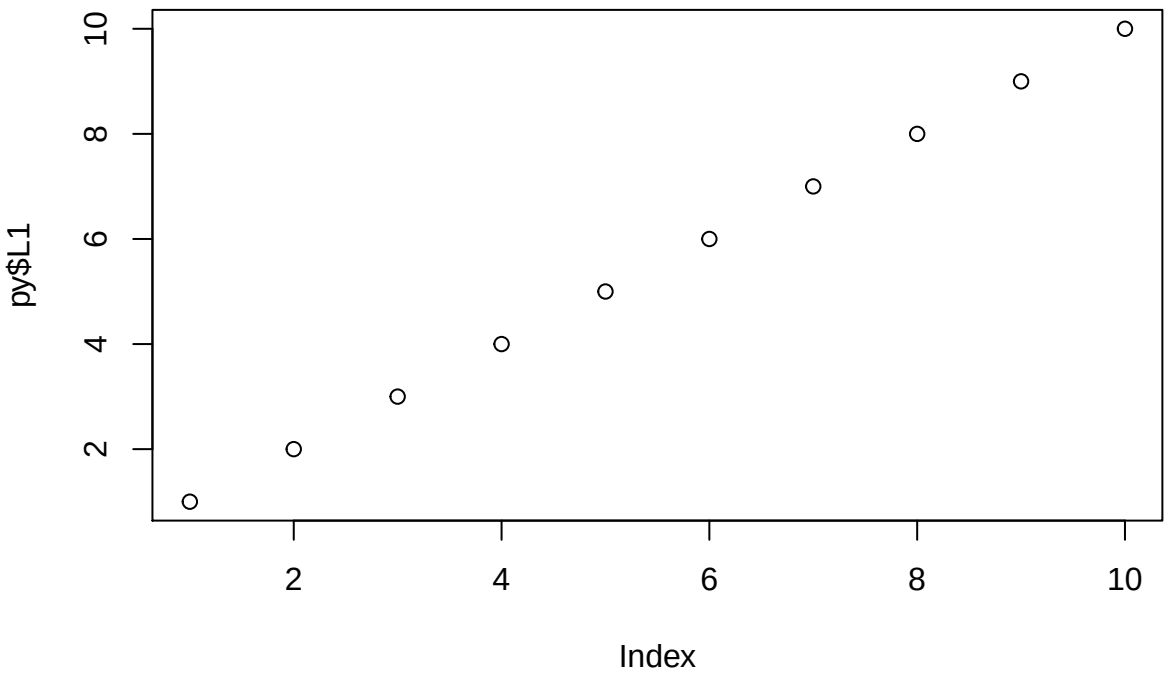
```
#Chunk R
source_python("script_python.py")#Tenemos definida la función "add" que suma dos números
add(3,4)
```

```
## [1] 7
```

3. Podemos declarar algún objeto desde un chunk de Python y recuperarlo en un chunk de R con **py$...*:

```
#Chunk Python
L1 = [1,2,3,4,5,6,7,8,9,10]
```

```
#Chunk R
plot(py$L1)
```



4. Lo mismo con un ejemplo de pandas:

```
#Chunk de Python
import pandas as pd

df = pd.read_csv("../data/bodyfat.txt", delimiter = " ")
df.head()
```

```
##    Density   Fat  Age  Weight  Height  ...  Knee  Ankle  Biceps  Forearm  Wrist
## 1   1.0708  12.3   23  154.25   67.75  ...  37.3   21.9    32.0     27.4   17.1
## 2   1.0853   6.1   22  173.25   72.25  ...  37.3   23.4    30.5     28.9   18.2
## 3   1.0414  25.3   22  154.00   66.25  ...  38.9   24.0    28.8     25.2   16.6
## 4   1.0751  10.4   26  184.75   72.25  ...  37.3   22.8    32.4     29.4   18.2
## 5   1.0340  28.7   24  184.25   71.25  ...  42.2   24.0    32.2     27.7   17.7
##
## [5 rows x 15 columns]
```

```
df.describe().transpose()
```

```
##           count        mean        std  ...       50%       75%       max
## Density   252.0    1.055574   0.019031  ...    1.0549    1.0704    1.1089
## Fat       252.0   19.150794   8.368740  ...   19.2000   25.3000   47.5000
## Age       252.0   44.884921  12.602040  ...   43.0000   54.0000   81.0000
## Weight    252.0  178.924405  29.389160  ...  176.5000  197.0000  363.1500
## Height    252.0   70.148810   3.662856  ...   70.0000   72.2500   77.7500
## Neck      252.0   37.992063   2.430913  ...   38.0000   39.4250   51.2000
## Chest     252.0  100.824206   8.430476  ...   99.6500  105.3750  136.2000
```

```
## Abdomen  252.0   92.555952  10.783077  ...   90.9500   99.3250  148.1000
## Hip      252.0   99.904762   7.164058  ...   99.3000  103.5250  147.7000
## Thigh    252.0   59.405952   5.249952  ...   59.0000   62.3500   87.3000
## Knee     252.0   38.590476   2.411805  ...   38.5000   39.9250   49.1000
## Ankle    252.0   23.102381   1.694893  ...   22.8000   24.0000   33.9000
## Biceps   252.0   32.273413   3.021274  ...   32.0500   34.3250   45.0000
## Forearm  252.0   28.663889   2.020691  ...   28.7000   30.0000   34.9000
## Wrist    252.0   18.229762   0.933585  ...   18.3000   18.8000   21.4000
##
## [15 rows x 8 columns]
```

```
#Desde este chunk de R recuperamos el df anterior para tratarlo con R

df.1 = py$df

str(df.1)
```

```
## 'data.frame':    252 obs. of  15 variables:
##  $ Density: num  1.07 1.09 1.04 1.08 1.03 ...
##  $ Fat    : num  12.3 6.1 25.3 10.4 28.7 20.9 19.2 12.4 4.1 11.7 ...
##  $ Age    : num  23 22 22 26 24 24 26 25 25 23 ...
##  $ Weight : num  154 173 154 185 184 ...
##  $ Height : num  67.8 72.2 66.2 72.2 71.2 ...
##  $ Neck   : num  36.2 38.5 34 37.4 34.4 39 36.4 37.8 38.1 42.1 ...
##  $ Chest  : num  93.1 93.6 95.8 101.8 97.3 ...
##  $ Abdomen: num  85.2 83 87.9 86.4 100 94.4 90.7 88.5 82.5 88.6 ...
##  $ Hip    : num  94.5 98.7 99.2 101.2 101.9 ...
##  $ Thigh  : num  59 58.7 59.6 60.1 63.2 66 58.4 60 62.9 63.1 ...
##  $ Knee   : num  37.3 37.3 38.9 37.3 42.2 42 38.3 39.4 38.3 41.7 ...
##  $ Ankle  : num  21.9 23.4 24 22.8 24 25.6 22.9 23.2 23.8 25 ...
##  $ Biceps : num  32 30.5 28.8 32.4 32.2 35.7 31.9 30.5 35.9 35.6 ...
##  $ Forearm: num  27.4 28.9 25.2 29.4 27.7 30.6 27.8 29 31.1 30 ...
##  $ Wrist  : num  17.1 18.2 16.6 18.2 17.7 18.8 17.7 18.8 18.2 19.2 ...
##  - attr(*, "pandas.index")=Int64Index([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,
##                ...
##               243, 244, 245, 246, 247, 248, 249, 250, 251, 252],
##              dtype='int64', length=252)
```

```
head(df.1)
```

```
##   Density  Fat Age Weight Height Neck Chest Abdomen   Hip Thigh Knee Ankle
## 1  1.0708 12.3  23 154.25  67.75 36.2  93.1    85.2  94.5  59.0 37.3  21.9
## 2  1.0853  6.1  22 173.25  72.25 38.5  93.6    83.0  98.7  58.7 37.3  23.4
## 3  1.0414 25.3  22 154.00  66.25 34.0  95.8    87.9  99.2  59.6 38.9  24.0
## 4  1.0751 10.4  26 184.75  72.25 37.4 101.8    86.4 101.2  60.1 37.3  22.8
## 5  1.0340 28.7  24 184.25  71.25 34.4  97.3   100.0 101.9  63.2 42.2  24.0
## 6  1.0502 20.9  24 210.25  74.75 39.0 104.5    94.4 107.8  66.0 42.0  25.6
##   Biceps Forearm Wrist
## 1   32.0    27.4  17.1
## 2   30.5    28.9  18.2
## 3   28.8    25.2  16.6
## 4   32.4    29.4  18.2
## 5   32.2    27.7  17.7
## 6   35.7    30.6  18.8
```

5. Ejemplo de plot con Python:

```
import matplotlib.pyplot as plt
import numpy as np
```

```
def f(x):
    return np.exp(-x)*np.cos(2*np.pi*x)
```

```
x1 = np.arange(0, 5.0, 0.1)
x2 = np.arange(0, 5.0, 0.2)
```

```
plt.figure(1)

_ = plt.subplot(2,1,1)# con la sintaxis _ = evitamos que aparezcan comentarios en el resultado final
_ = plt.plot(x1, f(x1), 'ro', x2, f(x2), 'k')

_ = plt.subplot(2,1,2)
_ = plt.plot(x2, f(x2), 'g--')

plt.show()
```

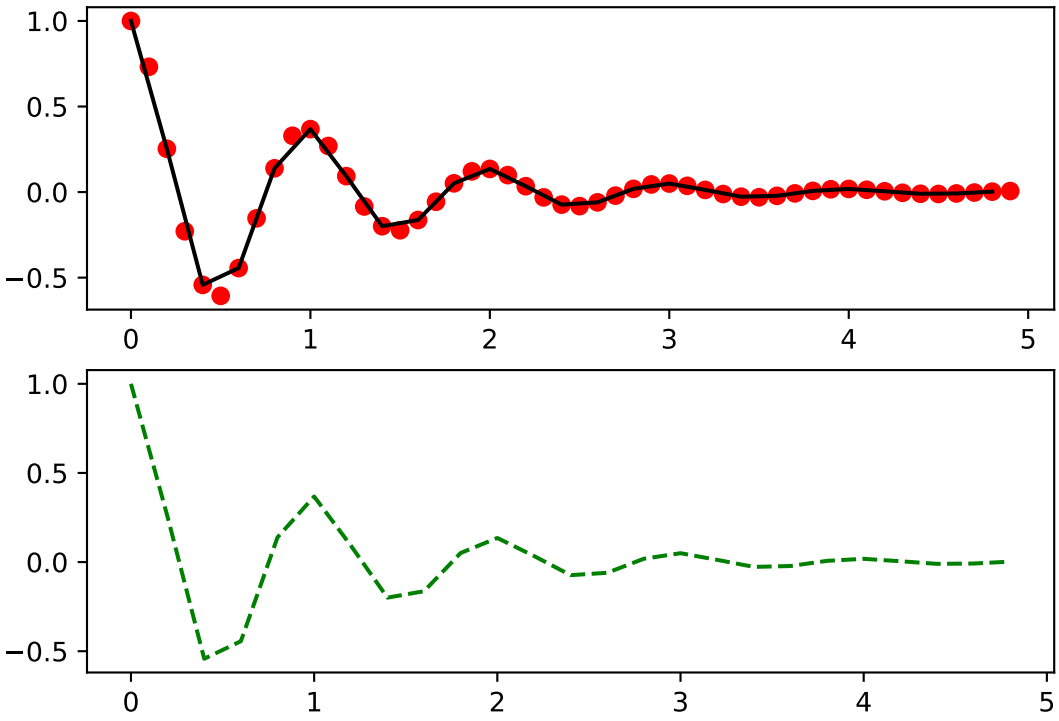
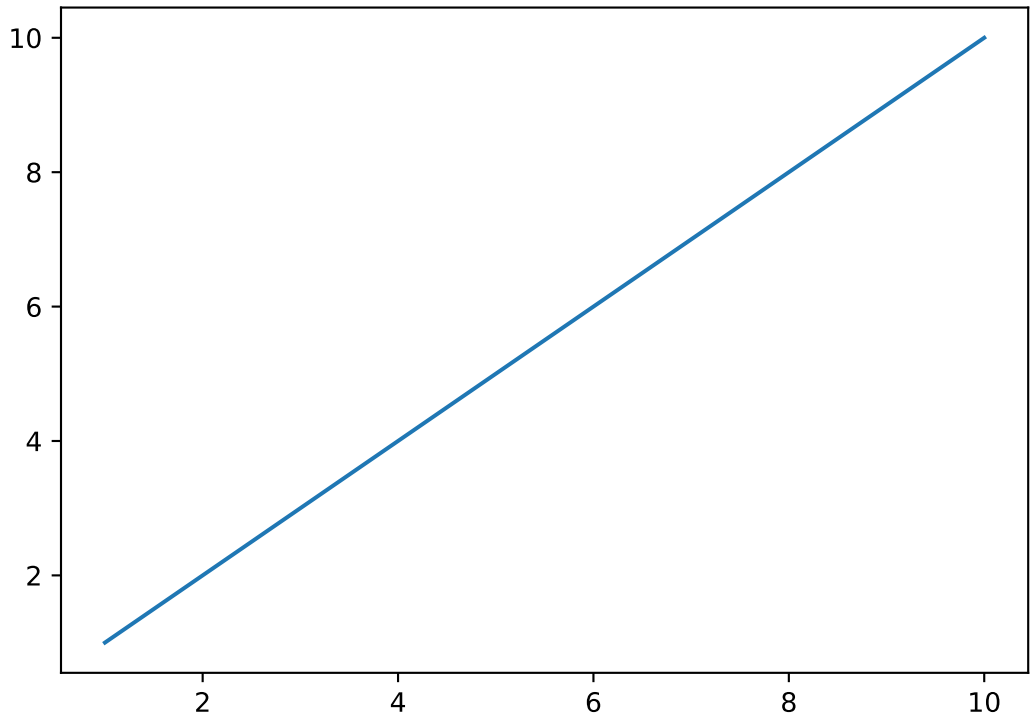


Figura 1: Hola

6. Pasar objetos de R a Python:

```
#Chunk de R
vector1 <- c(1,2,3,4,5,6,7,8,9,10)
vector2 <- c(1,2,3,4,5,6,7,8,9,10)
```

```
#Chunk de Python
_ = plt.plot(r.vector1, r.vector2)
plt.show()
```



Vamos a ejecutar la función **add** definida en un script de Python con algún parámetro proveniente de R sin r_to_py:

```
#Chunk de R
a <- 3
add(a,3)
```

```
## [1] 6
```

4. Estructuras de datos con R

5. Gráficos con la la función `plot`

6. Data Frames