Being Bilingual: coding in both R and Python

noRth 2020

Haema Nilakanta, PhD

July 14, 2020

Introduction

- The ongoing question: R or Python?

Introduction

- The ongoing question: R or Python?
- Why choose only one? Why not wrap one in the other?

Introduction

- The ongoing question: R or Python?
- Why choose only one? Why not wrap one in the other?
- Objective: introduce how to work with R and Python while in the R interface using the `reticulate` package
- I will assume minimal knowledge of Python

library(reticulate)

- First author and maintainer Kevin Ushey, RStudio
- Can use python already on your system, a virtual environment, specific versions, or Miniconda
- Works with python versions ≥ 2.7
- Build more seamless data science pipelines

Setting up python

- Some systems come downloaded with python
- If not, many ways to download (one option: Anaconda)
- Anaconda loads python and some well known packages versus miniconda which loads python and tools to install more packages (lighter weight)
- First time installing and start library, option to install miniconda
  - Happens if you don't specify python source explicity to use
  - Creates an `r-reticulate` Conda environment with python 3.6.10 with numpy version 1.18.15

Install package

```r
# Install the package
install.packages("reticulate")
```

```r
# Load the library
library(reticulate)

# and check the package version
packageVersion("reticulate")
```

```
## [1] '1.16'
```

Check python version

```
# Check what python source it's using
py_config()
```

```
python:         /Users/haema/Library/r-miniconda/envs/r-reticulate/bin/python
libpython:      /Users/haema/Library/r-miniconda/envs/r-reticulate/lib/libpython3.6m.dylib
pythonhome:     /Users/haema/Library/r-miniconda/envs/r-reticulate:/Users/haema/Library/r-miniconda/envs/r
version:        3.6.10 | packaged by conda-forge | (default, Apr 24 2020, 16:27:41)  [GCC Clang 9.0.1 ]
numpy:          /Users/haema/Library/r-miniconda/envs/r-reticulate/lib/python3.6/site-packages/numpy
numpy_version:  1.18.5
```

Translation “dictionary”

R	Python	Purpose
`library(packagename)`	`import modulename`	Load packages
`base` or `dplyr`	`pandas`	Data wrangling
`base`	`numpy`	Computations
`ggplot2`	`matplotlib`	Graphics

Interacting with python

1. Iteractive python (REPL)
2. Import python libraries
3. Load external python scripts

Interactive python

Can work with python in the console itself (REPL = Read–Eval–Print Loop)

```
# Start an interactive session
repl_python()
Python 3.6.10
(/Users/haema/Library/r-miniconda/envs/r-reticulate/bin/pyt
Reticulate 1.16 REPL -- A Python interpreter in R.
>>>
```

- The ">>>" indicates python environment
- To exit session, type `exit` and hit enter
- Whatever is defined in this session will remain in python session (version of fight club)

Import libraries

As with R, you may need functions available in other libraries. Sometimes the libraries are alredy installed (e.g., os and numpy)

```
# Load the os (operating system) module
os = import("os")

# print current working directory.
# In python keep () to run the function
os$getcwd()
```

```
## [1] "/Users/haema/Documents/noRth_reticulate_20200714"
```

```
# notice how it matches
getwd()
```

```
## [1] "/Users/haema/Documents/noRth_reticulate_20200714"
```

Import libraries

For libraries that are not installed yet, specificy the environment you want to install it to

```
# scipy popular python scientific computing library
conda_install("r-reticulate", "scipy")

# another approach
# sklearn holds many machine learning functions
py_install('sklearn', pip = TRUE)

# tensorflow popular library for deep-learning modules
# reticulate designed to install package from CRAN
install.packages("tensorflow")
```

Then we can import the module as before

```
scipy = import("scipy")
library(tensorflow)
```

Read in Python files (as functions)

Similary, we can read in a python file (e.g., load a function). Consider the following fuction stored in logitfunc.py to compute $logit(x) = \frac{e^x}{1+e^x}$.

```
import math

def logit(x):
  return math.exp(x)/(1+math.exp(x))
```

Load file with

```
source_python("logitfunc.py")
logit(0.5)
```

```
## [1] 0.6224593
```

Example with NLP

Run through example if time permits

Conclusion

- Work with python in R using library(reticulate)
- Build more seamless pipelines and leverage both systems
- Some more resources (clickable links):
  - Rstudio Reticulate
  - CRAN reticulate
  - Tutorial Rshiny + Python (virutal env)