Python, IPython, Jupyter

Pritam Dalal

Python is Interpreted

- Python is an interpreted language. (So are R and Matlab.)
- When you installed Anaconda, you installed a program called the *interpreter*.
- The interpreter sits on your computer and waits for Python commands. Upon receiving one, it immediately performs the computations encoded in that command.
- Thus far, our experience of issuing commands to the interpreter has been to type code into the cell of a Juypter Notebook, and then hitting shift + enter.
- Let's explore some other ways to send commands to the interpreter.

Running Python from the Shell (Part 1 of 3)

1. Open up the shell on your machine:
   - Mac OSX - Terminal
   - Windows - ??
2. At the prompt, type `python` and then press -enter-.
3. You should see some text get printed and then following prompt: >>>.
4. You are now in the Python interactive shell.
5. You can type Python commands at the prompt, similar to a Jupyter notebook code cell.

Running Python from the Shell (Part 2 of 3)

Let's try a few commands:

6. `>>> import pandas_datareader as pdr`
7. `>>> df_spy = pdr.get_data_yahoo('SPY')`
8. `>>> df_spy.head()`
9. `>>> df_spy['Adj Close'].plot()`
10. `>>> quit()`

Running Python from the Shell (Part 3 of 3)

- Obviously, this interface leaves a lot to be desired, especially for interactive scientific computing.
- For example, plot most likely won't work on your machine (but there is a way to get them to open in a separate window).
- This was the motivation for IPython.
- IPython is an enhanced wrapper around Python, one that is more suitable for scientific computing.

Running IPyton from the Shell

1. `$ ipython`
2. You will see a few lines of text printed and then: `In: [1]`
3. `In: [2] dir()`
   - notice that `df_spy` is gone from the session
4. `In: [3] import pandas_datareader as pdr`
5. `In: [4] df_spy = pdr.get_data_yahoo('SPY')`
6. `In: [5] df_spy['Adj Close'].plot()` **still didn’t work**
7. `In: [6] quit()`

IPython Qt Console

1. The next step in the evolution of IPython was the Qt console, which has a more modern feel and allows for in-line graphics.
2. `$ jupyter qtconsole`
3. `In: [1] import pandas_datareader as pdr`
4. `In: [2] df_spy = pdr.get_data_yahoo('SPY')`
5. `In: [3] %matplotlib inline`
6. `In: [4] df_spy['Adj Close'].plot()` **Success!**
7. `In: [5] quit()`

History (Part 1 of 3)

- Both spreadsheets and computational notebooks started being developed and released in the 1980s.
- Early spreadsheets were VisiCalc, Lotus 1-2-3, and Excel.
- Early notebooks were Maple and Mathematica.
- Python was created in 1990 Guido van Rossum.
- The IPython project was started in 2001 by Fernando Perez.
- IPython began as an enhanced interactive wrapper to Python, designed to facilitate scientific computing.

History (Part 2 of 3)

- IPython was heavily influence by the early computational notebooks, and in the mid-2000s started developing its own.
- After several stalled attempts, in 2011, the first version of IPython Notebook was developed
- By 2014, the IPython project had come to encompass the interactive shell, the Qt Console, the notebooks, and several other projects - all in a single repository.
  - That's a lot of code.
- Moreover, IPython Notebooks began supporting other languages: first Julia, then R, and now many others. (But Julia, Python, and R are the main three.)

History (Part 3 of 3)

- In 2014, all the language agnostic parts of the IPython project were spun-off and rebranded as Project Jupyter.
- Due to the shared history of Jupyter and IPython, the two often get conflated.
- Today, precise use of the term IPython only has two meanings:
  - the interactive IPython shell
  - the Python backend (kernel) of Jupyter Notebook
- Jupyter has many projects under it's umbrella:
  - Notebooks (you're familiar with)
  - JupyterHub (shared notebooks + more)
  - **JupyterLab** (an IDE for interactive computing projects)

Ways to Execute a Script in JupyterLab

1. Navigate to `module/` folder of the Supplemental Materials folder.
2. Let's examine the contents of `01_first_module.py`.
3. You can run this script from a Jupyter notebook:
   - launch the `01_for_running_module.ipynb` and type along.
   - use the `%run` magic
4. You can also run this script from an IPython console.
   - use the `%run` magic
5. You can also run this script from the shell - brave souls can follow along.

Observations About JupytyerLab

1. The left sidebar give a view to file structure your computer, which allows for easy access and organization of files.
2. An integrated environment for writing and running python code in various ways: scripts, notebook, consoles.
3. At various stages in a large data analysis project, you may need to interact with Python code in these different ways.
4. This makes JupyterLab an ideal IDE for data analysis and scientific computing (very much inline with with vision of Project Jupyter).