introduction_python

September 6, 2022

Lab 1: Introduction to Python

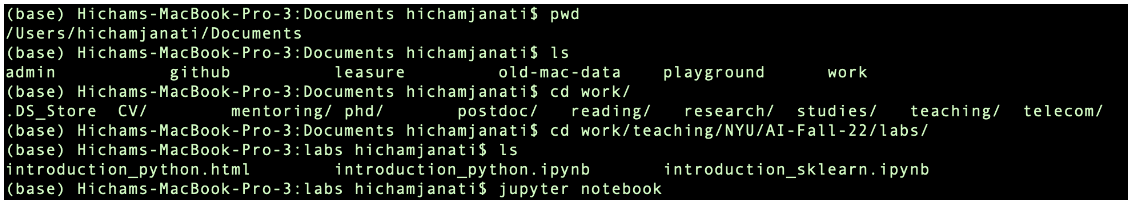
Goals:

- Have a working Python / Jupyter environment with packages installed
- Know where to look for information: external resources & documentation
- Discover core packages of the Python ecosystem:
    1. `Numpy` for scientific computing
    2. `Matplotlib` for plotting and visualization
    3. `Pandas` for manipulating heterogenous dataframes
    4. `scikit-learn` for Machine learning models

0.0.1 1 - Getting started with Jupyter

1. Anaconda is a large Python distribution software that includes several Python packages and tools and simplifies their management. If you do not have it installed on your laptop go over to the official documentation and follow the OS dependent installation guidelines.
2. Create a labs folder for the Python labs of this course. For me, this path is `/Users/hichamjanati/Documents/work/teaching/NYU/labs`. Download the notebook `introduction_python.ipynb` from the class website and save it in that folder.
3. Now we need to launch Jupyter from the labs folder. To do so, open the Anaconda Terminal (Anaconda Prompt). A command line window should pop up. If you are not familiar with line commands, for our purposes all you need to know is:

- Know what folder your terminal is in: `pwd` (print working directory) for Linux/Mac | `cd` for windows.
- List the content of the current folder: `ls` for Linux/Mac | `dir` for Windows.
- Change the current folder: `cd` followed by the subfolder you want to go to. Below, we complete the path with `work/teaching/NYU/code`:



```
(base) Hichams-MacBook-Pro-3:Documents hichamjanati$ pwd
/Users/hichamjanati/Documents
(base) Hichams-MacBook-Pro-3:Documents hichamjanati$ ls
admin           github          leasure         old-mac-data    playground      work
(base) Hichams-MacBook-Pro-3:Documents hichamjanati$ cd work/
.DS_Store  CV/        mentoring/ phd/       postdoc/   reading/   research/  studies/   teaching/  telecom/
(base) Hichams-MacBook-Pro-3:Documents hichamjanati$ cd work/teaching/NYU/AI-Fall-22/labs/
(base) Hichams-MacBook-Pro-3:labs hichamjanati$ ls
introduction_python.html        introduction_python.ipynb       introduction_sklearn.ipynb
(base) Hichams-MacBook-Pro-3:labs hichamjanati$ jupyter notebook
```

4. Now launch Jupyter by running: `jupyter notebook` which should open the browser. You should be able to access and run this notebook.

0.0.2 2 - Getting started: Python in Jupyter

2.1 running cells, magic commands and for loops The following Python cell is a naive loop that stores the first 10 million numbers. Click-on then hit `Shift+Enter` to run it.

[17]:
```
%%time

N = 10_000_000 # underscores in whole numbers are ignored by Python

numbers = []  # create an empty list
for ii in range(N):
    numbers.append(ii) # add the number to the list
total = sum(numbers)

print(f"The total is {total}")
```

```
The total is 49999995000000
CPU times: user 1.2 s, sys: 54.9 ms, total: 1.26 s
Wall time: 1.26 s
```

The `%%time` in the beginning of the cell is called a *magic command* that keeps track of the time it took the CPU to run the entire cell. Magic commands with one percentage apply for one lines only:

[18]:
```
%time print("The total is", sum([ii for ii in range(N)]))
```

```
The total is 49999995000000
CPU times: user 609 ms, sys: 105 ms, total: 714 ms
Wall time: 713 ms
```

Question 1:

Can you guess why is this one-liner 2x faster than the code above ?

2.2 Jupyter cells: code, markdown and shortcuts One of the main advantages of Jupyter is the ability to alternate between text cells (such as the one you are reading right now) and Python cells. Double-click on this text to edit it.

Text cells are actually *Markdown* cells. Markdown is a *super* light formatting language. As you probably noticed by editing some of these cells, **double asterisks make text bold**, while *un-*

derscores make text italic. But I digress, Markdown is a long story for another day, check the documentation for more. More importantly, here is how to create one:

1. Enable the *command mode* by hitting `Esc`. Command mode in Jupyter makes the cursor disappear.
2. Press `M` to switch from code to Markdown
3. Or press `Y` to switch from Markdown to code.

Several other shortcuts exist in the *command mode*. The ones I usually are: 1. `A` to add a new cell above 2. `B` to add a new cell below 3. `H` to open the help and check all other shortcuts

Keep in mind these shortcuts only work in Command Mode (i.e after pressing `Esc`).

2.3 The Numpy library: speed and clarity Let's write code performing the same counting operation above but using the `Numpy` library.

[10]:
```python
import numpy as np
```

[16]:
```python
%%time
N = 10_000_000
numbers = np.arange(N)  # creates an array of integers from 0 to N-1
total = numbers.sum()    # sums the array
print(f"The total is {total}")
```

```
The total is 49999995000000
CPU times: user 136 ms, sys: 81.3 ms, total: 218 ms
Wall time: 218 ms
```

As you can see `Numpy` is 3x faster than list comprensions and code is much shorter. `Numpy` should always be preferred to native Python lists when dealing with nothing but numbers.

[ ]:
```

```