NumPy Essential for Data Science

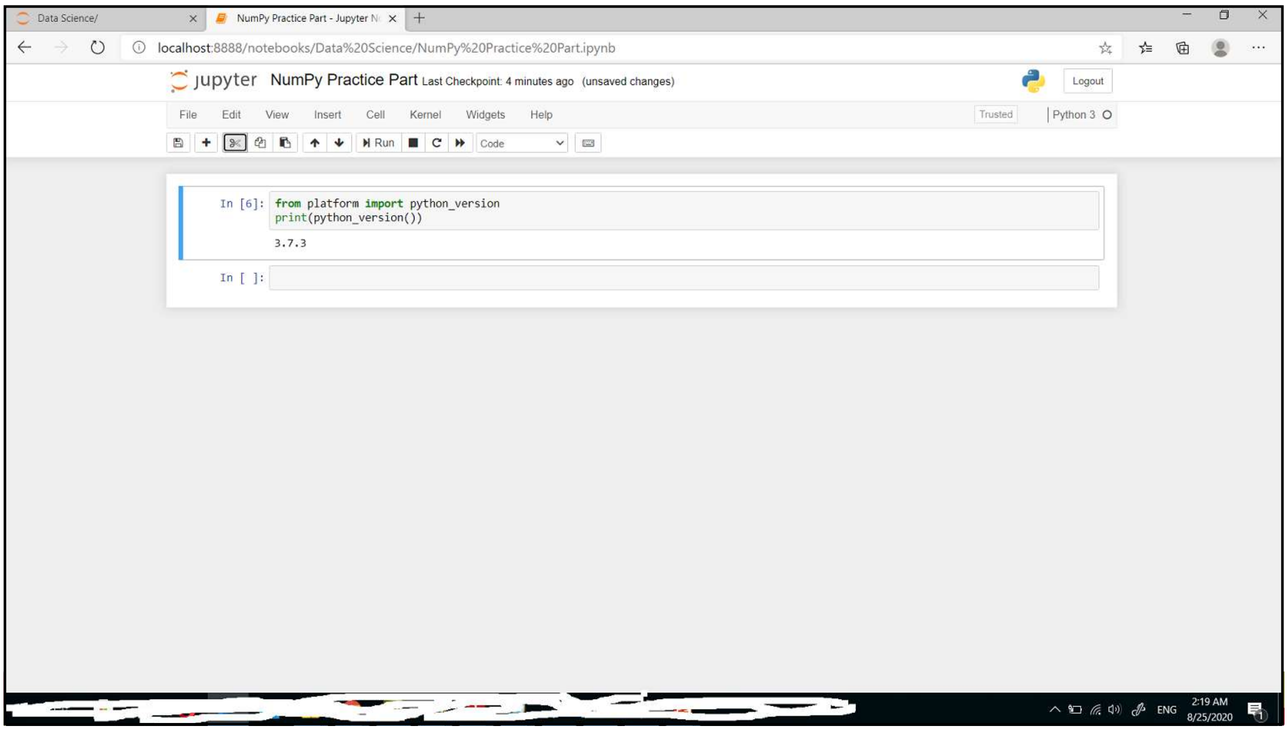
Muhammad Affan Alim

Brief introduction of Python

- Invented in the Netherlands, early 90s by Guido van Rossum
- Open sourced from the beginning
- Considered a scripting language, but is much more
  - No compilation needed
  - Scripts are evaluated by the interpreter, line by line
  - Functions need to be defined before they are called

Installing the Anaconda

- Watch the video
- https://www.youtube.com/watch?v=G3Lt1JWBvL8

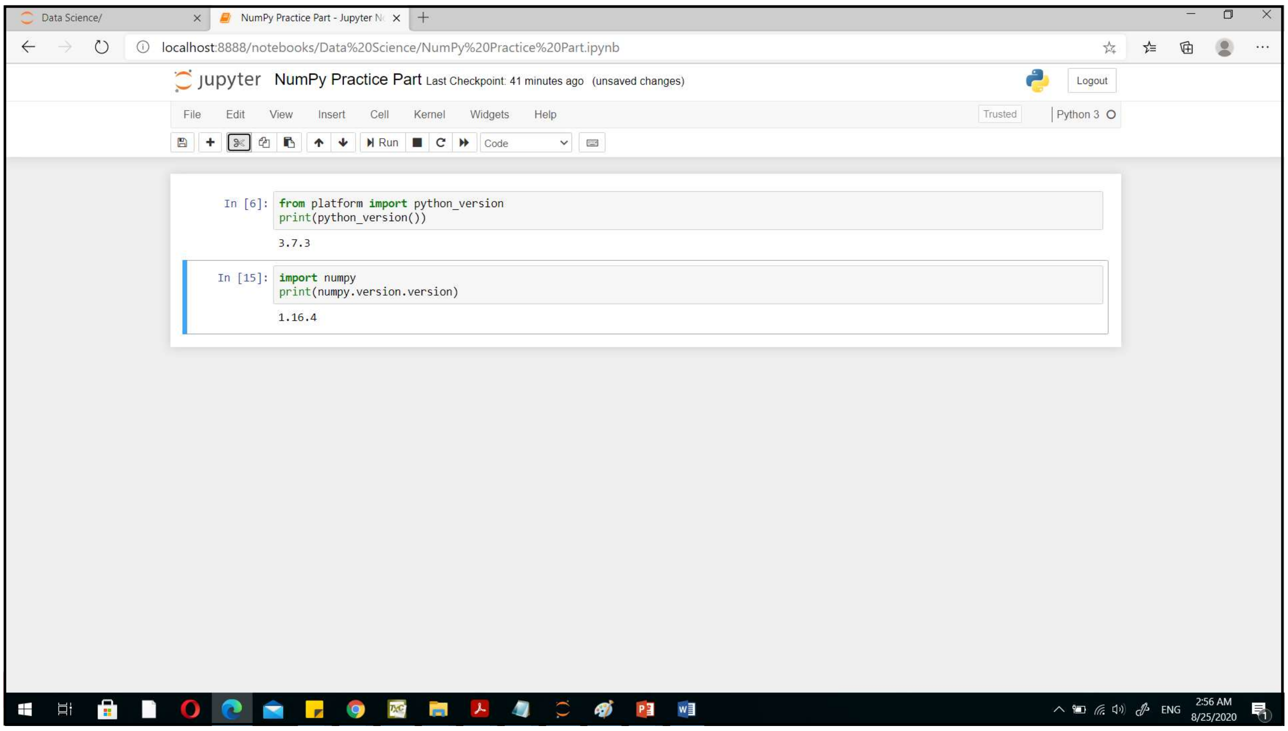

Data Science/
NumPy Practice Part - Jupyter N
localhost:8888/notebooks/Data%20Science/NumPy%20Practice%20Part.ipynb
jupyter NumPy Practice Part Last Checkpoint: 4 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
ENG
2:19 AM
8/25/2020

```
In [6]: from platform import python_version
        print(python_version())

        3.7.3

In [ ]:
```

[redacted]

Introduction to NumPy

- NumPy (short for Numerical Python) provides an efficient interface to store and operate on dense data buffers.
- NumPy arrays from the core of nearly the entire ecosystem of data science tools in Python
- If you followed the installation the Anaconda stack, you already have NumPy



NumPy Practice Part Last Checkpoint: 41 minutes ago (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3

```
In [6]: from platform import python_version
        print(python_version())

        3.7.3

In [15]: import numpy
         print(numpy.version.version)

         1.16.4
```

NumPy cont…

- By convention, you'll find that most people in the SciPy/PyData world will import NumPy using np as an alias:

- In[2]: import numpy as np

Why is NumPy Faster Than Lists?

- NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently.

- This behavior is called locality of reference in computer science.

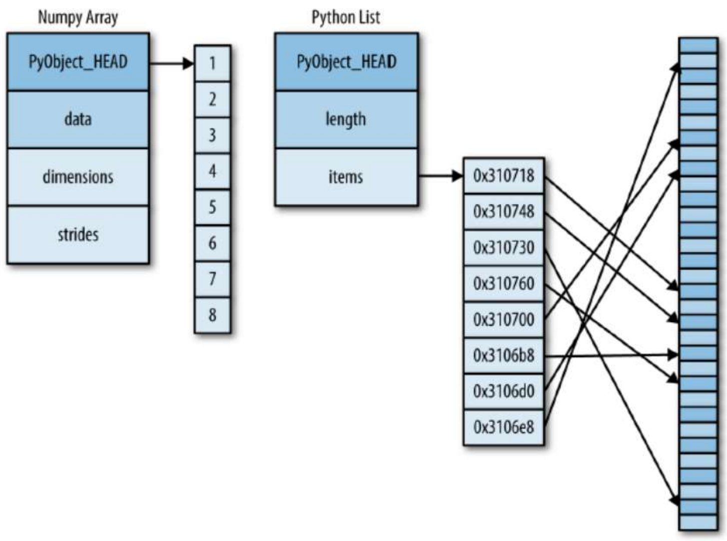
- This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.

c

- The Python list, on the other hand, contains a pointer to a block of pointers, each of which in turn points to a full Python object like the Python integer.
- Fixed-type NumPy-style arrays lack this flexibility, but are much more efficient for storing and manipulating data

Difference between NumPy array and Python List

-



Creating arrays using NumPy

- First, we can use np.array to create arrays from Python lists:

```
In[8]: # integer array:
       np.array([1, 4, 2, 5, 3])

Out[8]: array([1, 4, 2, 5, 3])
```

- Remember that unlike Python lists, NumPy is constrained to arrays that all contain the same type.

Creating arrays using NumPy cont…

- If types do not match, NumPy will upcast if possible (here, integers are upcast to floating point):

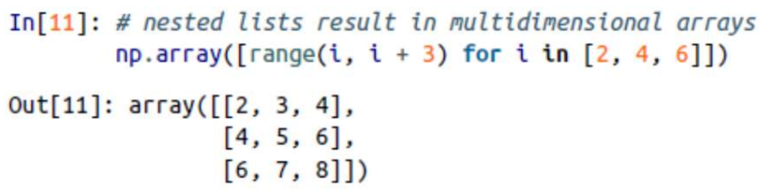
```
In[9]: np.array([3.14, 4, 2, 3])

Out[9]: array([ 3.14,  4.  ,  2.  ,  3.  ])
```

- If we want to explicitly set the data type of the resulting array, we can use the dtype keyword:

```
In[10]: np.array([1, 2, 3, 4], dtype='float32')

Out[10]: array([ 1.,  2.,  3.,  4.], dtype=float32)
```

Creating arrays using NumPy cont…

- NumPy arrays can explicitly be multidimensional; here's one way of initializing a multidimensional array using a list of lists:



```
In[11]: # nested lists result in multidimensional arrays
        np.array([range(i, i + 3) for i in [2, 4, 6]])

Out[11]: array([[2, 3, 4],
                [4, 5, 6],
                [6, 7, 8]])
```