NumPy

INTRODUCTION TO PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

Lists Recap

- Powerful
- Collection of values
- Hold different types
- Change, add, remove
- Need for Data Science
  - Mathematical operations over collections
  - Speed

Illustration

```
height = [1.73, 1.68, 1.71, 1.89, 1.79]
height
```

```
[1.73, 1.68, 1.71, 1.89, 1.79]
```

```
weight = [65.4, 59.2, 63.6, 88.4, 68.7]
weight
```

```
[65.4, 59.2, 63.6, 88.4, 68.7]
```

```
weight / height ** 2
```

```
TypeError: unsupported operand type(s) for ** or pow(): 'list' and 'int'
```

Solution: NumPy

- Numeric Python
- Alternative to Python List: NumPy Array
- Calculations over entire arrays
- Easy and Fast
- Installation
  - In the terminal: `pip3 install numpy`

NumPy

```python
import numpy as np
np_height = np.array(height)
np_height
```

```
array([1.73, 1.68, 1.71, 1.89, 1.79])
```

```python
np_weight = np.array(weight)
np_weight
```

```
array([65.4, 59.2, 63.6, 88.4, 68.7])
```

```python
bmi = np_weight / np_height ** 2
bmi
```

```
array([21.85171573, 20.97505669, 21.75028214, 24.7473475 , 21.44127836])
```

Comparison

```
height = [1.73, 1.68, 1.71, 1.89, 1.79]
weight = [65.4, 59.2, 63.6, 88.4, 68.7]
weight / height ** 2
```

```
TypeError: unsupported operand type(s) for ** or pow(): 'list' and 'int'
```

```
np_height = np.array(height)
np_weight = np.array(weight)
np_weight / np_height ** 2
```

```
array([21.85171573, 20.97505669, 21.75028214, 24.7473475 , 21.44127836])
```

NumPy: remarks

```
np.array([1.0, "is", True])
```

```
array(['1.0', 'is', 'True'], dtype='<U32')
```

- NumPy arrays: contain only one type

NumPy: remarks

```python
python_list = [1, 2, 3]
numpy_array = np.array([1, 2, 3])
```

```python
python_list + python_list
```

```
[1, 2, 3, 1, 2, 3]
```

```python
numpy_array + numpy_array
```

```
array([2, 4, 6])
```

- Different types: different behavior!

NumPy Subsetting

```
bmi
```

```
array([21.85171573, 20.97505669, 21.75028214, 24.7473475 , 21.44127836])
```

```
bmi[1]
```

```
20.975
```

```
bmi > 23
```

```
array([False, False, False,  True, False])
```

```
bmi[bmi > 23]
```

```
array([24.7473475])
```

Let's practice!

INTRODUCTION TO PYTHON

2D NumPy Arrays

INTRODUCTION TO PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

Type of NumPy Arrays

```python
import numpy as np
np_height = np.array([1.73, 1.68, 1.71, 1.89, 1.79])
np_weight = np.array([65.4, 59.2, 63.6, 88.4, 68.7])
```

```python
type(np_height)
```

```
numpy.ndarray
```

```python
type(np_weight)
```

```
numpy.ndarray
```

2D NumPy Arrays

```
np_2d = np.array([[1.73, 1.68, 1.71, 1.89, 1.79],
                  [65.4, 59.2, 63.6, 88.4, 68.7]])
np_2d
```

```
array([[ 1.73,  1.68,  1.71,  1.89,  1.79],
       [65.4 , 59.2 , 63.6 , 88.4 , 68.7 ]])
```

```
np_2d.shape
```

```
(2, 5) # 2 rows, 5 columns
```

```
np.array([[1.73, 1.68, 1.71, 1.89, 1.79],
          [65.4, 59.2, 63.6, 88.4, "68.7"]])
```

```
array([['1.73', '1.68', '1.71', '1.89', '1.79'],
       ['65.4', '59.2', '63.6', '88.4', '68.7']], dtype='<U32')
```

Subsetting

```
             0        1        2        3        4


array([[  1.73,    1.68,    1.71,    1.89,    1.79],     0
        [  65.4,    59.2,    63.6,    88.4,    68.7]])    1
```

```
np_2d[0]
```

```
array([1.73, 1.68, 1.71, 1.89, 1.79])
```

Subsetting

```
            0        1        2        3        4

array([[  1.73,    1.68,    1.71,    1.89,    1.79],     0
       [  65.4,    59.2,    63.6,    88.4,    68.7]])    1
```

```
np_2d[0][2]
```

```
1.71
```

```
np_2d[0, 2]
```

```
1.71
```

Subsetting

```
          0       1       2       3       4


array([[  1.73,   1.68,   1.71,   1.89,   1.79],     0
       [  65.4,   59.2,   63.6,   88.4,   68.7]])    1
```

```
np_2d[:, 1:3]
```

```
array([[ 1.68,  1.71],
       [59.2 , 63.6 ]])
```

```
np_2d[1, :]
```

```
array([65.4, 59.2, 63.6, 88.4, 68.7])
```

Let's practice!

INTRODUCTION TO PYTHON

NumPy: Basic Statistics

INTRODUCTION TO PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

Data analysis

- Get to know your data
- Little data -> simply look at it
- Big data -> ?

City-wide survey

```python
import numpy as np
np_city = ... # Implementation left out
np_city
```

```
array([[1.64, 71.78],
       [1.37, 63.35],
       [1.6 , 55.09],
       ...,
       [2.04, 74.85],
       [2.04, 68.72],
       [2.01, 73.57]])
```

NumPy

```python
np.mean(np_city[:, 0])
```

```
1.7472
```

```python
np.median(np_city[:, 0])
```

```
1.75
```

NumPy

```
np.corrcoef(np_city[:, 0], np_city[:, 1])
```

```
array([[ 1.      , -0.01802],
       [-0.01803,  1.      ]])
```

```
np.std(np_city[:, 0])
```

```
0.1992
```

- sum(), sort(), ...
- Enforce single data type: speed!

Generate data

- Arguments for `np.random.normal()`
    - distribution mean
    - distribution standard deviation
    - number of samples

```
height = np.round(np.random.normal(1.75, 0.20, 5000), 2)

weight = np.round(np.random.normal(60.32, 15, 5000), 2)

np_city = np.column_stack((height, weight))
```

Let's practice!

INTRODUCTION TO PYTHON