data_types

August 9, 2020

```
In [1]: import math
```

0.1 Data Types in Python

The following data types can be used in base python: * **boolean** * **integer** * **float** * **string** * **list** * **None** * complex * object * set * dictionary

We will only focus on the **bolded** ones

Let's connect these data types to the the variable types we learned from the Variable Types video.

0.1.1 Numerical or Quantitative (taking the mean makes sense)

- Discrete
  - Integer (int) #Stored exactly
- Continuous
  - Float (float) #Stored similarly to scientific notation. Allows for decimal places but loses precision.

```
In [2]: type(4)
```

```
Out[2]: int
```

```
In [3]: type(0)
```

```
Out[3]: int
```

```
In [4]: type(-3)
```

```
Out[4]: int
```

```
In [5]: #try taking the mean
        numbers = [2, 3, 4, 5]
        print(sum(numbers)/len(numbers))
        type(sum(numbers)/len(numbers)) #In Python 3 returns float, but in Python 2 would retu
```

```
3.5
```

```
Out[5]: float
```

Floats

```
In [6]: 3/5
```

```
Out[6]: 0.6
```

```
In [7]: 6*10**(-1)
```

```
Out[7]: 0.6000000000000001
```

```
In [8]: type(3/5)
```

```
Out[8]: float
```

```
In [9]: type(math.pi)
```

```
Out[9]: float
```

```
In [10]: type(4.0)
```

```
Out[10]: float
```

```
In [11]: # Try taking the mean
         numbers = [math.pi, 3/5, 4.1]
         type(sum(numbers)/len(numbers))
```

```
Out[11]: float
```

0.1.2 Categorical or Qualitative

- Nominal
  - Boolean (bool)
  - String (str)
  - None (NoneType)
- Ordinal
  - Only defined by how you use the data
  - Often important when creating visuals
  - Lists can hold ordinal information because they have indices

Boolean

```
In [12]: # Boolean
         type(True)
```

```
Out[12]: bool
```

```
In [13]: # Boolean
         if 6 < 5:
             print("Yes!")
```

```
In [14]: myList = [True, 6<5, 1==3, None is None]
         for element in myList:
             print(type(element))
```

```
<class 'bool'>
<class 'bool'>
<class 'bool'>
<class 'bool'>
```

```
In [15]: print(sum(myList)/len(myList))
         type(sum(myList)/len(myList))
```

```
0.5
```

```
Out[15]: float
```

String

```
In [16]: type("This sentence makes sense")
```

```
Out[16]: str
```

```
In [17]: type("Makes sentense this sense")
```

```
Out[17]: str
```

```
In [18]: type("math.pi")
```

```
Out[18]: str
```

```
In [19]: strList = ['dog', 'koala', 'goose']
         sum(strList)/len(strList)
```

```
    ---------------------------------------------------------------------------

    TypeError                                 Traceback (most recent call last)

    <ipython-input-19-b0bd059010c7> in <module>()
      1 strList = ['dog', 'koala', 'goose']
----> 2 sum(strList)/len(strList)


    TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Nonetype

```
In [20]: # None
         type(None)

Out[20]: NoneType

In [21]: # None
         x = None
         type(x)

Out[21]: NoneType

In [22]: noneList = [None]*5
         sum(nonList)/len(nonList)


        ---------------------------------------------------------------------------

        NameError                                 Traceback (most recent call last)

        <ipython-input-22-08e0974f29ad> in <module>()
          1 noneList = [None]*5
    ----> 2 sum(nonList)/len(nonList)


        NameError: name 'nonList' is not defined
```

Lists

A list can hold many types and can also be used to store ordinal information.

```
In [23]: # List
         myList = [1, 1.1, "This is a sentence", None]
         for element in myList:
             print(type(element))

<class 'int'>
<class 'float'>
<class 'str'>
<class 'NoneType'>


In [24]: sum(myList)/len(myList)


        ---------------------------------------------------------------------------

        TypeError                                 Traceback (most recent call last)
        <ipython-input-24-01620fe6b2d4> in <module>()
    ----> 1 sum(myList)/len(myList)


        TypeError: unsupported operand type(s) for +: 'float' and 'str'
```

```
In [25]: # List
         myList = [1, 2, 3]
         for element in myList:
             print(type(element))
         sum(myList)/len(myList) # note that this outputs a float
```

```
<class 'int'>
<class 'int'>
<class 'int'>
```

```
Out[25]: 2.0
```

```
In [26]: myList = ['third', 'first', 'medium', 'small', 'large']
         myList[0]
```

```
Out[26]: 'third'
```

```
In [27]: myList.sort()
         myList
```

```
Out[27]: ['first', 'large', 'medium', 'small', 'third']
```

There are more datatypes available when using different libraries such as Pandas and Numpy, which we will introduce to you as we use them.