PYTHON FOR R USERS

Introduction

Daniel Chen
Instructor

The Basics

- Variables
- Assignment
- Strings
- Lists
- Dictionaries

Programming Fundamentals

- Control flow (if-statments)
- loops
- Functions

Pandas

- Selecting and subsetting data
- Data types
- Data manipulation and processing techniques

Plotting

- pandas
- matplotlib
- seaborn

Putting it all together

- NYC Flights 2013 Dataset
- Planes
- Flight Delays

Python Data types

R

- numeric: integers, floating point numbers
- logical: TRUE or FALSE values
- character: strings

PYTHON

- int: integers
- float: floating point numbers
- bool: True or False values
- str: strings

Assignment

R

```
> x <- 3

> y = 0.5

> z <- TRUE
```

PYTHON

```
In [1]: x = 3

In [2]: y = 0.5

In [3]: z = True
```

Print Statements

R

```
> print('R is awesome!')
[1] "R is awesome!"
```

PYTHON

```
In [1]: print('Python too!')
Python too!
```

What's your data type?

R

```
> x <- 3

> class(x)
[1] "numeric"
```

PYTHON

```
In [4]: x = 3

In [5]: type(x)
Out[5]: int

In [6]: y = 0.5

In [7]: type(y)
Out[7]: float
```

Operators

R

```
> 1 + 1
[1] 2
> '1' + '1'
Error in "1" + "1" : non-numeric argument to binary operator
> '1' * 5
Error in "1" * 5 : non-numeric argument to binary operator
> '1' '1'
Error: unexpected string constant in "'1' '1'"
```

Operators

PYTHON

```
In [7]: 1 + 1
Out[7]: 2

In [8]: '1' + '1'
Out[8]: '11'

In [9]: '1' * 5
Out[9]: '11111'

In [10]: '1' '1'
Out[10]: '11'
```

PYTHON FOR R USERS

Let's practice!

PYTHON FOR R USERS

Containers - Lists and Dictionaries

Daniel Chen
Instructor

Lists

- Similar to the R vector and list
- Holds heterogeneous information (e.g., [3, 2.718, True, 'hello'])

Creating Lists

R

```
> l <- list(1, "2", TRUE)
> l
[[1]]
[1] 1

[[2]]
[1] "2"

[[3]]
[1] TRUE
```

PYTHON

```
In [8]: l = [1, '2', True]

In [9]: l
Out[9]: [1, '2', True]
```

Python is 0-indexed

Python is a 0-indexed language

Subsetting Lists

R

Get the 1st element from R list

```
> l[[1]]  # use 1
[1] 1
```

PYTHON

Get the 1st element from Python list

```
In [10]: l[0] # use 0
Out[10]: 1
```

Negative Indices

Negative indices start counting from the end

Negative Indicies

R

```
> l <- list(1, "2", TRUE)
> l[-1]
[[1]]
[1] "2"

[[2]]
[1] TRUE
```

PYTHON

```
In [8]:  l = [1, "2", True]
In [11]: l[-1]
Out[11]: True
```

Left Inclusive Right Exclusive

Left inclusive, right exclusive

Think Fence Posts

```
    index 0           1           2           3           4           5
          |           |           |           |           |           |
          | element 0 | element 1 | element 2 | element 3 | element 4 |
          |           |           |           |           |           |
```

Slicing Part 1

PYTHON

```
In [12]: l = [0, 1, 2, 3, 4]

In [14]: l[0:3]
Out[14]: [0, 1, 2]

In [10]: l[0:5:2]
Out[10]: [0, 2, 4]
```

Slicing Part 2

IMPLICIT

```
In [14]: l[:3]
Out[14]: [0, 1, 2]

In [13]: l[1:]
Out[13]: [1, 2, 3, 4]

In [10]: l[::2]
Out[10]: [0, 2, 4]
```

EXPLICIT

```
In [14]: l[0:3]
Out[14]: [0, 1, 2]

In [14]: l[1:5]
Out[14]: [1, 2, 3, 4]

In [10]: l[0:5:2]
Out[10]: [0, 2, 4]
```

Dictionaries

- Lists are unlabeled
- Dictionaries provide key:value pairs

Python: Creating and working with dictionaries

```
In [19]: d = {'int_value':3, 'bool_value':False, 'str_value':'hello'}

In [21]: d
Out[21]: {'bool_value': False, 'int_value': 3, 'str_value': 'hello'}

In [20]: d['str_value']
Out[20]: 'hello'
```

Finding the length

```
In [5]: l = [0, 1, 2, 3, 4]
In [4]: d = {'int_value':3, 'bool_value':False, 'str_value':'hello'}

In [7]: len(l)
Out[7]: 5

In [6]: len(d)
Out[6]: 3
```

PYTHON FOR R USERS

Let's practice!

PYTHON FOR R USERS

Functions, Methods, and Libraries

Daniel Chen
Instructor

Calling Functions

- R and Python functions work the same
- Python: Functions and Methods

Methods vs Functions

- Python is object oriented
  - Attributes
  - Methods
- Methods are functions that an **object can call** on itself
- Functions are **called on an object**

```
l = [0, 1, 2, 3, 4]
len(l)   # call a function
l.len()  # not a method
```

Periods have a special meaning in Python

Periods have very specific meanings and uses

Append to List

```
In [22]: l = [1, "2", True]

In [23]: l.append('appended value')

In [24]: l
Out[24]: [1, '2', True, 'appended value']
```

Update a Dicitonary

```
In [25]: d = {'int_value':3, 'bool_value':False, 'str_value':'hello'}

In [26]: d.update({'str_value':'new_value', 'new_key':'new_value'})

In [27]: d
Out[27]:
{'bool_value': False,
 'int_value': 3,
 'new_key': 'new_value',
 'str_value': 'new_value'}
```

Methods are specific to the object

```
In [4]: d.append('hello')
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-4-42922a99ec56> in <module>()
----> 1 d.append('hello')

AttributeError: 'dict' object has no attribute 'append'
```

Libraries

- Libraries provide more functionality
- Arrays and dataframes are not built into Python
- Arrays and matricies come from numpy
- Dataframes come from pandas

Numpy ndarray

R

```
> library(readr)
> dat <- read_csv('my_file.csv')
```

PYTHON

```
import numpy
arr = numpy.loadtxt('my_file.csv', delimiter=',')

import numpy as np
arr = np.loadtxt('my_file.csv', delimiter=',')
```

Pandas DataFrame

```
import pandas as pd

df = pd.read_csv('my_file.csv')
df.head()
```

PYTHON FOR R USERS

Let's practice!