Machine Learning 2018
Python Tutorial #1

Kien C Nguyen

November 10, 2018

The Python Interpreter

- Python is an interpreted language (as opposed to compiled languages such as C++).
- Commands can be executed interactively, or a series of commands can be saved in a plain text file known as source code or a script.
- For Python, source code is normally stored in a file with the .py suffix (e.g., hello.py).
- We will use Python 2 (2.7 and above) for this class.

How to run Python code

We can use one of the following ways

- Start Python at the command line by typing
  $ python
- Edit source.py and run
  $ python source.py
- Make the python file executable and run it directly
  - Make the Python source code executable
    $ chmod +x source.py
  - Put the path to the Python interpreter at the very first line of the Python source code, e.g.
    #!/usr/local/bin/python
  - Run the Python source code directly
    $ ./source.py
- Use an IDE (Integrated Development Environment) such as Pycharm.

Sample Python source code

```
def calculate_mean(f_list):
    l_len = len(f_list)
    # Calculate the mean
    l_sum = 0
    for e in f_list:
        l_sum += e
    l_mean = float(l_sum) / l_len
    return l_mean

my_list = [4, 5, 20]
print calculate_mean(my_list)
```

Identifiers, Objects, and the Assignment Statement

- Python is an object-oriented language and all data types are based on classes.
- Consider the assignment

```
score = 10.5
```

- This associates the identifier *score* with a floating-point object with value 10.5.
- Identifiers in Python are case-sensitive, so score and Score are distinct names.
- Identifiers can be composed of almost any combination of letters, numerals, and underscore characters.
- An identifier, however, cannot begin with a numeral (E.g., '4runner' is not a valid identifier.

Identifiers, Objects, and the Assignment Statement

- An identifier cannot be one of the reserved words (Table from [1] Goldwasser, Goodrich, and Tamassia)

Reserved Words								
False	as	continue	else	from	in	not	return	yield
None	assert	def	except	global	is	or	try	
True	break	del	finally	if	lambda	pass	while	
and	class	elif	for	import	nonlocal	raise	with	

Table 1.1: A listing of the reserved words in Python. These names cannot be used as identifiers.

- Unlike statically typed languages such as C++, Python is a dynamically typed language.
- We do not need to associate an identifier with a particular data type during declaration.
- We can associate an identifier with any type of object, and later reassign it to another object of the same (or different) type.
- Although an identifier has no declared type, the object to which it refers has a definite type.

Calling Methods

- In Python, we can call independent (non-class) functions that are invoked with a syntax such as calculate_mean(my_list), in which case my_list is a parameter passed into the function.
- We can also define methods (member functions) for Pythons classes, which can be invoked on an instance of the class using the dot (.) operator.

A Python class

```
class MyList(object):
    def __init__(self, in_list):
        # Data members (attributes)
        self._list = in_list

    # Member functions (methods)
    def calculate_mean(self):
        # Here we use the independent function
        # calculate_mean() provided earlier
        return calculate_mean(self._list)

scores = MyList([4, 5, 10])
print scores.calculate_mean()
```

Pythons Built-In Classes

- The table below [1] provides a summary of commonly used, built-in classes in Python
- There are mutable and immutable classes.
- A class is immutable if each object of that class has a fixed value upon instantiation that cannot be changed later on.

Class	Description	Immutable?
bool	Boolean value	✓
int	integer (arbitrary magnitude)	✓
float	floating-point number	✓
list	mutable sequence of objects	
tuple	immutable sequence of objects	✓
str	character string	✓
set	unordered set of distinct objects	
frozenset	immutable form of set class	✓
dict	associative mapping (aka dictionary)	

Table 1.2: Commonly used built-in classes for Python

References

[1] M. H. Goldwasser, M. T. Goodrich, and R. Tamassia – Data Structures and Algorithms in Python, John Wiley & Sons, 2013