

Python Tutorial

Namhyuk Ahn

Setup - Window

- Window
 - <https://www.python.org/downloads/windows/>
 - Recommend to download 3.6 version
 - And install..!

- [Python 3.6.5 - 2018-03-28](#)
 - Download [Windows x86 web-based installer](#)
 - Download [Windows x86 executable installer](#)
 - Download [Windows x86 embeddable zip file](#)
 - Download [Windows x86-64 web-based installer](#)
 - Download [Windows x86-64 executable installer](#)
 - Download [Windows x86-64 embeddable zip file](#)
 - Download [Windows help file](#)

Setup - Ubuntu & macOS

- Ubuntu (Recommend)

- ```
sudo add-apt-repository ppa:deadsnakes/ppa
sudo apt-get update
sudo apt-get install python3.6
```

- Type `python --version` to check version
  - Use *virtualenv* if possible to manage package safe and efficient

- macOS (Recommend)

- Install brew, and `>> brew install python3`
  - Type `python --version` to check version
  - Use *virtualenv* if possible to manage package safe and efficient

# Python at-a-glance

- Easy quick sort

```
def quicksort(arr):
 if len(arr) <= 1:
 return arr
 pivot = arr[len(arr) // 2]
 left = [x for x in arr if x < pivot]
 middle = [x for x in arr if x == pivot]
 right = [x for x in arr if x > pivot]
 return quicksort(left) + middle + quicksort(right)

print(quicksort([3,6,8,10,1,2,1]))
Prints "[1, 1, 2, 3, 6, 8, 10]"
```

# Data types

- No need of *int*  $x = 3$ ;

```
x = 3
print(type(x)) # Prints "<class 'int'>"
print(x) # Prints "3"
print(x + 1) # Addition; prints "4"
print(x - 1) # Subtraction; prints "2"
print(x * 2) # Multiplication; prints "6"
print(x ** 2) # Exponentiation; prints "9"
x += 1
print(x) # Prints "4"
x *= 2
print(x) # Prints "8"
y = 2.5
print(type(y)) # Prints "<class 'float'>"
print(y, y + 1, y * 2, y ** 2) # Prints "2.5 3.5 5.0 6.25"
```

# Booleans and strings

```
t = True
f = False
print(type(t)) # Prints "<class 'bool'>"
print(t and f) # Logical AND; prints "False"
print(t or f) # Logical OR; prints "True"
print(not t) # Logical NOT; prints "False"
print(t != f) # Logical XOR; prints "True"
```

```
hello = 'hello' # String literals can use single quotes
world = "world" # or double quotes; it does not matter.
print(hello) # Prints "hello"
print(len(hello)) # String length; prints "5"
hw = hello + ' ' + world # String concatenation
print(hw) # prints "hello world"
hw12 = '%s %s %d' % (hello, world, 12) # sprintf style string formatting
print(hw12) # prints "hello world 12"
```

# Containers

- List (equivalent of an array in C)

```
xs = [3, 1, 2] # Create a list
print(xs, xs[2]) # Prints "[3, 1, 2] 2"
print(xs[-1]) # Negative indices count from the end of the list; prints "2"
xs[2] = 'foo' # Lists can contain elements of different types
print(xs) # Prints "[3, 1, 'foo']"
xs.append('bar') # Add a new element to the end of the list
print(xs) # Prints "[3, 1, 'foo', 'bar']"
x = xs.pop() # Remove and return the last element of the list
print(x, xs) # Prints "bar [3, 1, 'foo']"
```

# For loops

- Be pythonic! Better to use 1) and 3)

```
1) animals = ['cat', 'dog', 'monkey']
for animal in animals:
 print(animal)

>> cat
>> dog
>> monkey
```

```
2) animals = ['cat', 'dog', 'monkey']
for idx in len(range(animals)):
 print(animals[idx])

>> cat
>> dog
>> monkey
```

```
3) animals = ['cat', 'dog', 'monkey']
for idx, animal in enumerate(animals):
 print(idx, animal)

>> 0 cat
>> 1 dog
>> 2 monkey
```



# List comprehension

```
nums = [0, 1, 2, 3, 4]
squares = []
for x in nums:
 squares.append(x ** 2)
print(squares) # Prints [0, 1, 4, 9, 16]
```

- equivalent syntax

```
nums = [0, 1, 2, 3, 4]
squares = [x ** 2 for x in nums]
print(squares) # Prints [0, 1, 4, 9, 16]
```

# Dictionary

```
d = {'cat': 'cute', 'dog': 'furry'} # Create a new dictionary with some data
print(d['cat']) # Get an entry from a dictionary; prints "cute"
print('cat' in d) # Check if a dictionary has a given key; prints "True"
d['fish'] = 'wet' # Set an entry in a dictionary
print(d['fish']) # Prints "wet"
print(d['monkey']) # KeyError: 'monkey' not a key of d
print(d.get('monkey', 'N/A')) # Get an element with a default; prints "N/A"
print(d.get('fish', 'N/A')) # Get an element with a default; prints "wet"
del d['fish'] # Remove an element from a dictionary
print(d.get('fish', 'N/A')) # "fish" is no longer a key; prints "N/A"
```

# Indexing the dict

```
d = {'person': 2, 'cat': 4, 'spider': 8}
for animal in d:
 legs = d[animal]
 print(animal, legs)
>> person 2
>> cat 4
>> spider 8
```

- or use `.items()`

```
d = {'person': 2, 'cat': 4, 'spider': 8}
for animal, legs in d.items():
 print(animal, legs)
```

# Functions

```
def sign(x):
 if x > 0:
 return 'positive'
 elif x < 0:
 return 'negative'
 else:
 return 'zero'

for x in [-1, 0, 1]:
 print(sign(x))
```

```
def hello(name, loud=False):
 if loud:
 print('HELLO, %s!' % name.upper())
 else:
 print('Hello, %s' % name)

hello('Bob') # Prints "Hello, Bob"
hello('Fred', loud=True) # Prints "HELLO, FRED!"
```

# Classes

```
class Greeter(object):

 # Constructor
 def __init__(self, name):
 self.name = name # Create an instance variable

 # Instance method
 def greet(self, loud=False):
 if loud:
 print('HELLO, %s!' % self.name.upper())
 else:
 print('Hello, %s' % self.name)

g = Greeter('Fred') # Construct an instance of the Greeter class
g.greet() # Call an instance method; prints "Hello, Fred"
g.greet(loud=True) # Call an instance method; prints "HELLO, FRED!"
```

# “With”

- Example: read CSV

```
import csv
f = open("data.csv", "r")
reader = csv.reader(f)
for line in reader:
 print(line)
f.close()
```

(입력 : data.csv 파일 내용)

```
1,김정수,2017-01-19 11:30:00,25
2,박민구,2017-02-07 10:22:00,35
3,정순미,2017-03-22 09:10:00,33
```

(출력)

```
['1', '김정수', '2017-01-19 11:30:00', '25']
['2', '박민구', '2017-02-07 10:22:00', '35']
['3', '정순미', '2017-03-22 09:10:00', '33']
```

- with scope 밖으로 나가면 자동적으로 close

```
import csv
with open("data.csv", "r") as f:
 reader = csv.reader(f)
 for line in reader:
 print(line)
```

# Further reading

- 점프투파이썬 <https://wikidocs.net/book/1>

# Python packages

- To download and install packages..
  1. Install pip  
(<https://www.makeuseof.com/tag/install-pip-for-python/>)
  2. (optional) Install virtualenv  
(<https://beomi.github.io/2016/12/28/HowToSetup-Virtualenv-VirtualenvWrapper/>)
  3. `pip install <package_name>`
- In our class;
- `>> pip install numpy scipy matplotlib scikit-learn`



# Numpy

- Core package for scientific computing

```
import numpy as np

a = np.array([1, 2, 3]) # rank가 1인 배열 생성
print type(a) # 출력 "<type 'numpy.ndarray'>"
print a.shape # 출력 "(3,)"
print a[0], a[1], a[2] # 출력 "1 2 3"
a[0] = 5 # 요소를 변경
print a # 출력 "[5, 2, 3]"

b = np.array([[1,2,3],[4,5,6]]) # rank가 2인 배열 생성
print b.shape # 출력 "(2, 3)"
print b[0, 0], b[0, 1], b[1, 0] # 출력 "1 2 4"
```

# Numpy - Create an array

```
import numpy as np

a = np.zeros((2,2)) # 모든 값이 0인 배열 생성
print a # 출력 "[[0. 0.]
 # [0. 0.]]"

b = np.ones((1,2)) # 모든 값이 1인 배열 생성
print b # 출력 "[[1. 1.]]"

c = np.full((2,2), 7) # 모든 값이 특정 상수인 배열 생성
print c # 출력 "[[7. 7.]
 # [7. 7.]]"

d = np.eye(2) # 2x2 단위행렬 생성
print d # 출력 "[[1. 0.]
 # [0. 1.]]"

e = np.random.random((2,2)) # 임의의 값으로 채워진 배열 생성
print e # 임의의 값 출력 "[[0.91940167 0.08143941]
 # [0.68744134 0.87236687]]"
```

# Numpy - Indexing and slicing

```
import numpy as np

아래와 같은 요소를 가지는 rank가 2이고 shape가 (3, 4)인 배열 생성
[[1 2 3 4]
[5 6 7 8]
[9 10 11 12]]
a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])

배열의 중간 행에 접근하는 두 가지 방법이 있습니다.
정수 인덱싱과 슬라이싱을 혼합해서 사용하면 낮은 rank의 배열이 생성되지만,
슬라이싱만 사용하면 원본 배열과 동일한 rank의 배열이 생성됩니다.
row_r1 = a[1, :] # 배열a의 두 번째 행을 rank가 1인 배열로
row_r2 = a[1:2, :] # 배열a의 두 번째 행을 rank가 2인 배열로
print row_r1, row_r1.shape # 출력 "[5 6 7 8] (4,)"
print row_r2, row_r2.shape # 출력 "[[5 6 7 8]] (1, 4)"

행이 아닌 열의 경우에도 마찬가지입니다:
col_r1 = a[:, 1]
col_r2 = a[:, 1:2]
print col_r1, col_r1.shape # 출력 "[2 6 10] (3,)"
print col_r2, col_r2.shape # 출력 "[[2]
 # [6]
 # [10]] (3, 1)"
```

# Numpy - Data type

```
import numpy as np

x = np.array([1, 2]) # Numpy가 자료형을 추측해서 선택
print x.dtype # 출력 "int64"

x = np.array([1.0, 2.0]) # Numpy가 자료형을 추측해서 선택
print x.dtype # 출력 "float64"

x = np.array([1, 2], dtype=np.int64) # 특정 자료형을 명시적으로 지정
print x.dtype # 출력 "int64"
```

# Numpy - Braodcasting

- 다른 shape를 가지는 행렬끼리 연산
  - 이해하기 어렵지만 알고나면 효과적

```
import numpy as np

행렬 x의 각 행에 벡터 v를 더한 뒤,
그 결과를 행렬 y에 저장하고자 합니다
x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
v = np.array([1, 0, 1])
y = np.empty_like(x) # x와 동일한 shape를 가지며 비어있는 행렬 생성

명시적 반복문을 통해 행렬 x의 각 행에 벡터 v를 더하는 방법
for i in range(4):
 y[i, :] = x[i, :] + v

이제 y는 다음과 같습니다
[[2 2 4]
[5 5 7]
[8 8 10]
[11 11 13]]
print y
```

반복문을 통해 행렬 연산  
(매우 느림)

# Numpy - Braodcasting

- 다른 shape를 가지는 행렬끼리 연산
  - 이해하기 어렵지만 알고나면 효과적

```
import numpy as np

벡터 v를 행렬 x의 각 행에 더한 뒤,
그 결과를 행렬 y에 저장하고자 합니다
x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
v = np.array([1, 0, 1])
y = x + v # 브로드캐스팅을 이용하여 v를 x의 각 행에 더하기
print y # 출력 "[[2 2 4]
 # [5 5 7]
 # [8 8 10]
 # [11 11 13]]"
```

브로드캐스팅을 사용한 코드

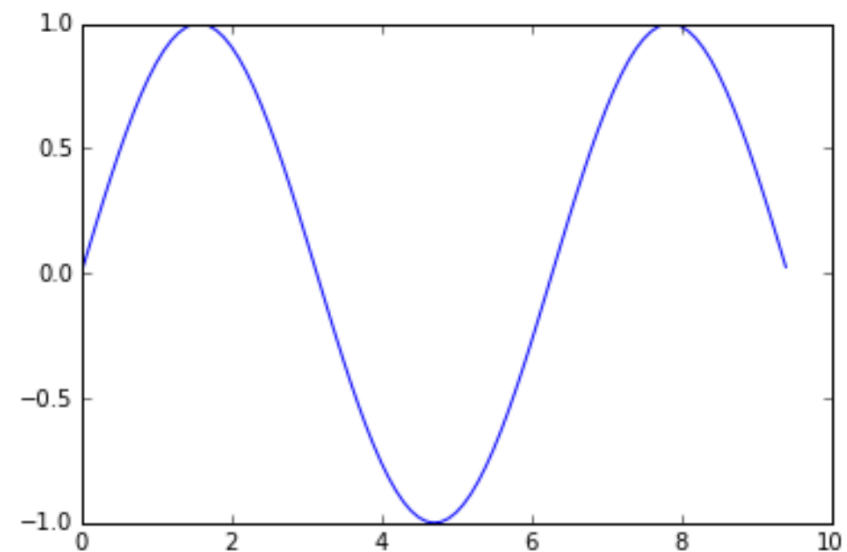
# Matplotlib

- plotting 라이브러리

```
import numpy as np
import matplotlib.pyplot as plt

사인과 코사인 곡선의 x, y 좌표를 계산
x = np.arange(0, 3 * np.pi, 0.1)
y = np.sin(x)

matplotlib를 이용해 점들을 그리기
plt.plot(x, y)
plt.show() # 그래프를 나타나게 하기 위해선 plt.show() 함수를 호출해야만 합니다.
```



# Jupyter notebook

- web에서 interactive 하게 파이썬 코딩
  - 장점: visual coding
  - 단점: 구조적으로 “잘 짜기” 어려움
  - >> pip install jupyter (설치)
  - >> jupyter notebook (실행)
- 참고:
  - <http://aikorea.org/cs231n/ipython-tutorial/>
  - <https://dojang.io/mod/page/view.php?id=1157>



# Scikit-learn

- Python package for machine learning
- Basic flow:
  1. 데이터 로드 (파일 읽기 -> numpy나 pandas 타입으로 변환)
  2. 데이터 전처리, EDA
  3. 모델 생성 및 학습
  4. 모델 평가

# 데이터 로드 & EDA

- (과제에서) 주로 CSV 파일
  1. CSV 읽기 (python tutorial 참고)
  2. List에 넣기 (append 사용)
  3. List 타입 -> numpy 타입
- EDA
  - matplotlib 등으로 feature plotting
  - 통계량, correlation 등 출력

# 모델 생성 및 학습

- 모델 생성 (SVM)

- `>> from sklearn import svm`
- `>> model = svm.SVC(gamma=0.001, C=100.)`

- 모델 클래스의 methods

- `fit(X, y)`: 입력 X와 라벨 y를 통해 해당 모델 학습
- `predict(T)`: 입력 T에 해당하는 예측값 리턴

```
>>> clf.fit(digits.data[:-1], digits.target[:-1])
SVC(C=100.0, cache_size=200, class_weight=None, coef0=0.0,
 decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
 max_iter=-1, probability=False, random_state=None, shrinking=True,
 tol=0.001, verbose=False)
```

```
>>> clf.predict(digits.data[-1:])
array([8])
```

# 모델 평가

|                                                                |                                                                   |
|----------------------------------------------------------------|-------------------------------------------------------------------|
| <code>accuracy_score</code> (y_true, y_pred[, normalize, ...]) | Accuracy classification score.                                    |
| <code>classification_report</code> (y_true, y_pred[, ...])     | Build a text report showing the main classification metrics       |
| <code>f1_score</code> (y_true, y_pred[, labels, ...])          | Compute the F1 score, also known as balanced F-score or F-measure |
| <code>fbeta_score</code> (y_true, y_pred, beta[, labels, ...]) | Compute the F-beta score                                          |
| <code>hamming_loss</code> (y_true, y_pred[, labels, ...])      | Compute the average Hamming loss.                                 |
| <code>jaccard_similarity_score</code> (y_true, y_pred[, ...])  | Jaccard similarity coefficient score                              |
| <code>log_loss</code> (y_true, y_pred[, eps, normalize, ...])  | Log loss, aka logistic loss or cross-entropy loss.                |
| <code>precision_recall_fscore_support</code> (y_true, y_pred)  | Compute precision, recall, F-measure and support for each class   |
| <code>precision_score</code> (y_true, y_pred[, labels, ...])   | Compute the precision                                             |
| <code>recall_score</code> (y_true, y_pred[, labels, ...])      | Compute the recall                                                |
| <code>zero_one_loss</code> (y_true, y_pred[, normalize, ...])  | Zero-one classification loss.                                     |

```
>>> import numpy as np
>>> from sklearn.metrics import accuracy_score
>>> y_pred = [0, 2, 1, 3]
>>> y_true = [0, 1, 2, 3]
>>> accuracy_score(y_true, y_pred)
0.5
>>> accuracy_score(y_true, y_pred, normalize=False)
2
```

[http://scikit-learn.org/stable/modules/model\\_evaluation.html](http://scikit-learn.org/stable/modules/model_evaluation.html)

# 모델 평가

- Cross-validation을 사용할 경우 (아래 예제 참고)

| Scoring               | Function                                     | Comment                                     |
|-----------------------|----------------------------------------------|---------------------------------------------|
| <b>Classification</b> |                                              |                                             |
| 'accuracy'            | <code>metrics.accuracy_score</code>          |                                             |
| 'average_precision'   | <code>metrics.average_precision_score</code> |                                             |
| 'f1'                  | <code>metrics.f1_score</code>                | for binary targets                          |
| 'f1_micro'            | <code>metrics.f1_score</code>                | micro-averaged                              |
| 'f1_macro'            | <code>metrics.f1_score</code>                | macro-averaged                              |
| 'f1_weighted'         | <code>metrics.f1_score</code>                | weighted average                            |
| 'f1_samples'          | <code>metrics.f1_score</code>                | by multilabel sample                        |
| 'neg_log_loss'        | <code>metrics.log_loss</code>                | requires <code>predict_proba</code> support |
| 'precision' etc.      | <code>metrics.precision_score</code>         | suffixes apply as with 'f1'                 |
| 'recall' etc.         | <code>metrics.recall_score</code>            | suffixes apply as with 'f1'                 |
| 'roc_auc'             | <code>metrics.roc_auc_score</code>           |                                             |

```
>>> from sklearn import svm, datasets
>>> from sklearn.model_selection import cross_val_score
>>> iris = datasets.load_iris()
>>> X, y = iris.data, iris.target
>>> clf = svm.SVC(probability=True, random_state=0)
>>> cross_val_score(clf, X, y, scoring='neg_log_loss')
array([-0.07..., -0.16..., -0.06...])
```

cross-validation을 사용해서 모델 평가

# Further readings

- cs231n <http://cs231n.github.io/python-numpy-tutorial/>
- 혹은 한글 번역본 <http://aikorea.org/cs231n/python-numpy-tutorial/>
- scikit-learn 튜토리얼  
<http://scikit-learn.org/stable/tutorial/basic/tutorial.html>
- scikit-learn 예제  
[http://scikit-learn.org/stable/auto\\_examples/index.html](http://scikit-learn.org/stable/auto_examples/index.html)