0 . Python basic

Sogang University Department of Business Administration
Professor Myungsuk Kim

PyCharm editor

◆ Automatic construction of virtual environment
To avoid Python version conflicts

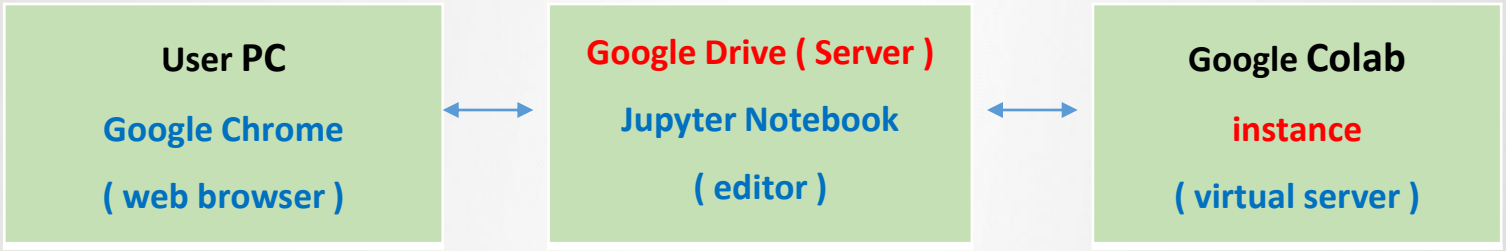
◆ Install package _ _
pip install --upgrade pip
terminal at the bottom Open a window and [pip install package name] input
To check the currently installed packages, type [pip list]
Or File>Settings>Project: Project name >Python interpreter
The requirements.txt file provides information about all package versions
pip install -r requirements.txt

◆ create new project
File>New Project> Create a project name , click Start in a new window : Remember location on PC
Create a Python file directly under the new project (created at the level where main.py is installed)

◆ Execution method
Open a terminal window and execute [python filename.py] or click the icon ▷ on the top right
If you want to execute only some commands, select the area with the mouse and press the [Alt+Shift+E]
The execution result can be checked by opening the console panel at the bottom.

Google Colab oratory (Colab)

◆ Visualize your environment



◆ Restrictions

- Storage : Up to 360GB/ 1 pc laptop Size : 20MB
- Automatic reset after 12 hours of operation / Reset after 90 min if no response lasts
- After the reset, the saved notebook is intact, but the data created (on Colab) will disappear unless you do not save them to PC or Google Drive

◆ Advantages of Colab

1) No need to build environment
2) Edit file : Jupyter Notebook Units (* .ipynb file) management
   (Google Drive> +New> more> Google Colaboratory)
3) GPU

Google Colaboratory (Colab)

◆ Upload local pc files to Colab

```
from google.colab import files
uploaded = files.upload () # Opens a window where you can select a specific file
!dir # uploaded data check , sample data is the default created data
```

◆ Save files to your PC

```
from google.colab import files
files.download ('dataframe6.xlsx')
```

◆ Save files to your Google drive

```
from google.colab import drive
drive.mount ('/content/drive')
!cp dataframe6.xlsx /content/drive/My₩ Drive/dataframe6.xlsx
```

◆ Main (Linux) Commands

command	explanation	command	explanation	command	explanation
dir	Check current path folder	cd..	Move a path up one level	rm - rf *	file Delete entire folder
pwd	Check current condensation	cp <A> <B>	Copy file (A) (B)	zip –r xxxx.zip <A>	Zip compression
cd <path>	go to route (change directory)	rm <A>	File (A) Delete	unzip xxxx.zip	Unzip _

Google Colaboratory (Colab)

◆ Install package

Input [!pip install package name]

To see which packages are currently installed [! Enter pip list] and check

The requirements.txt file provides information about all package versions required for running

!pip install -r requirements.txt

◆ Install Github file as below

! git clone "https://github.com/ultralytics/yolov5.git"

◆ Execution method

[!python path/filename.py] as Python file executable

If you want to execute only some commands, select the area with the mouse and press the [Contl+Shift+Enter]

◆ file path Setting method (Three)

(1) "C:/doit/newfile.txt "

(2) "C:₩₩doit₩₩ newfile.txt "

(3) r"C :₩doit₩ new file .txt"

Python basics

◆ modules and packages

Module : A Python program in which functions, variables, and classes are defined
(Standard module / Third party module / Custom module)

Package : A bundle made up of several modules.

(1) import < module name >

(2) import < module name > as < alias >

```
import numpy as np
a = np.array [[1,2,3], [4,5,6]]
print(a )
```

(3) from < module name > import < specific function >

```
from numpy import array
a = array[[1,2,3], [4,5,6]]
print(a )
```

(4) from < module name > import *

Python basics

```
#I will import a library (component or package made by others) called bs4 and use the functions in it .
import bs4
bs4. BeautifulSoup ( result.content , " html.parse ")


#The function called BeautifulSoup from a library called bs4: By calling it, I will use it right away
from bs4 import BeautifulSoup
Beautifulsoup ( result.content , " html.parse ")


#By calling urllib.request, a function [urlopen] will be used
import urllib.request
url = "https://www.naver.com/"
html = urllib.request. urlopen ( url )
```

Python basics

❖ Reserved words : Cannot be used as a variable or function name because the purpose of use has already been determined (check the method below)

```
import keyword
keyword.kwlist
```

❖ Basic input/output functions (print(), input())

```
print(10+20)
n = input('type a number : ')
print(n)
```

❖ Types of variables : integer (int), string (str), real number (float), ...
=> Using type(), you can check the type with a function

```
n = '3'
int (n); type(n)
n1=float (n); type(n1)
str (n)
```

Python basics

0. Main modules

```
import os
os.getcwd () # current working directory
os.listdir ('/content')
# Notice the names of files in that directory

os.mkdir ('XX') # create subdirectories
os.chdir ('/content/XX') # change current working directory
os.getcwd ()


import sys
sys. prefix
sys.version
sys. byteorder
sys.modules # Print all installed modules ( long )
```

```
import random
random.randint (1,5)
random.random ()
list = ['a', 'b', 'c']
random.choice (list)
random.shuffle (list)


import time
time.time ()
time.asctime ()


import calendar
cal = calendar. month (2020,10)
print( cal )
```

Python basics

1. Variables and Constants

```
out1 = 10
type(out1)
print(out1)
print(10)

out2="10"
type(out2)
print(out2)
print("10")
```

2. Conditional statements

```
score = 90
if score >=90:
print("A")
elif score >= 80:
print("B")
elif score >= 70:
print("C")
else:
print("F")
```

3. Function

```
def print_hello ():
print("hello")

print_hello ()

def print_message (phrase):
print(phrase)

input0="hello"
type(input0)
print_message ("hello")

input1=10
type(input1)
print_message (input1)

input2=[10]
type(input2)
print_message (input2)

input3={10}
type(input3)
print_message (input3)
```

```
def game(number):
if number == 1:
print('attack')
    elif number == 2:
print('defense')

game(2)

def plus(v1, v2):
return v1+v2

result1 = plus(10,15)
print("result = ", result1)

### lamda expression function : assigning a function to a function as an expression
from math import *
def radian(x):
   return x / 270 * pi

for x in range(0, 360, 90):
   print(' angle : {}, radians : {:.2f}'.format(x, radian(x)))

lambda_radian = (lambda x: x /270 * pi)
for x in range(0, 360, 90):
   print(' degrees : {}, radians : {:.2f}'. format(x, lamda_radian (x)))
```

Python basics

4. For and While statements

```
range(0,10)
list(range(0,10))

for item in range(0,10):
    print(item)

for item in range(0,10):
    index = item+1
    print(index, item)


num_list = list(range(0,10))
num_list

result = [num+1 for num in num_list ]
# result = [ operation result for element in list ]
print(result)


for n in [1,2,3,4,5]:
    print(n)
    print(n*10)
```

```
# enumerate function
for num , fruit in enumerate(['apple',' banna ', 'orange']):
    print('{}:{}'. format( num , fruit))

# list comprehensions
data=[]
for x in range(10):
    data. append (x**2)
    print(data)

data1 = [x**2 for x in range(10)]


# While
i = 0
while i < 20:
  i += 1
if i % 2 == 0:
continue
if i % 3 == 0:
print( i )
```

Python basics

5. Numbers

```
####### number type
# int (integer)
# float (real number)
# bool ( logical value)
# complex
  (complex number)


#### Integer
a = 20
b = -10
c = 15
total1 = a + b + c
print(total1)
type(total1)

#### mistakes
a = 3.14
b = 5
c = 2.2
total2 = a + b + c
total2 = total2 /2
print(total2)
type(total2)
```

```
#### operator
print(2 ** 3) # multiplier 2 to the 3
print(100 / 40) # division : quotient
print(100 % 40) # division : remainder
print(100 // 40)
# division : extract only integers from th
e quotient

####### comparison operators
# a = b (a equals b )
# a!= b (a and b are different )
# a<b (a is less than b )
# a>=b (a is greater than or equal to b )
# a<>b (a and b are different )
# a is b (a and b are equal )
# a is not b (a and b are different )
# a in b (a is included in b )
# a not in b (a is not included in b )
```

6. string

```
# string concatenation
ai = 'AI Big Data'
inov = 'Innovation MBA'
print( ai +" "+ inov )
```

```
# extract part of string
print( ai [0:3]) # first 3 characters
print( inov [7:]) # characters from 7th to last
print( inov [-2:]) # 2 characters from end
print( ai [0:2+2]) # first 2 characters + 2 characters
c = 10
pie = 3.14
diameter = 2 * pie * c
print(' circle circumference : '+ str (diameter))

# Assign variable to string
a = ' exercise '
b = 20
c = pie
print(' string = {}'.format(a))
print(' integer = {}'.format(b))
print(' decimal point = {}'.format(c))
print(' Decimal first = {:.1f}'.format(c))
print(' Multiple values = {}, {}, {:.2f}'.format(a, b, c))

# string , number conversion
num = 123
str1 = str ( num )
type(str1)
type( int (str1))
type(float(str1))
```

Python basics

7. Useful functions

```
### find(): Tells you where
# a specific string exists

sentence1 = 'AI big data
innovation MBA is a real AI data
program'

print(sentence1. find('AI'))
print(sentence1. find('data'))
print(sentence1. find('innovation'))
print(sentence1. find('parsing'))
```

```
### split(): split a string
# ( into a list conversion )
sentence2 = '
apple;banana;cherry;strawberry;tomato '

print(sentence2)
sentence3 = sentence2.split(';')
print(sentence3)


for i in range(0, len(sentence3)):
print(i+1, sentence3[ i ])


score = "{math:90}, {english:95},
{science:85}, {ethics:90}"
total = 0
score1 = score.split (",")
for i in range(0, len (score1)):
score2 = score1[ i ].split(":")[1].split("}")[0]
  num_score = int (score2)
total += num_score
print(total)
```

```
### replace(): find a specific string
# and replace it with another string
sentence1 = 'AI big data
innovation MBA is a real AI data
program'

print(sentence1.replace("AI","
metaverse "))
```

```
### count(): Shows the number of
# occurrences of a specific string
sentence1 = 'AI big data
innovation MBA is a real AI data
program'

print(sentence1. count("AI"))
print(sentence1. count(" mba "))
```

Python basics

8. List

```
value = [1,2,3,4,5,6]
print(value)
print(value[0])
print(value[1:3])
value[0] = 10
# Randomly change the
# values in the list
print(value[1:4])

list0 = ['apple', 'orange']
print(list0)
list0. append (' banna ')
print(list0)
list0. insert (0, ' straberry ')
print(list0)
list0. remove ('apple')
print(list0)
del list0[0]
print(list0)

list1 = [3,2,4,2,1,7,6,2]
list1.sort()
print(list1)
list1.count(2)
How many times does #2
occur ?
sum(list1)
```

```
list2 = ['apple', 'cherry', 'banana',
'strawberry']
list2.sort()
print(list2)

print(list(range(10)))
print(list(range(1, 4)))
print(list(range(1, 10, 3)))
```

9. Dictionary

```
dic = {'apple': 500, ' banna ': 350, 'cake':
15000}
print( dic ['apple'])

dic ['apple'] = 700
print( dic )

dic1 = {'apple': 500}
print(dic1)
dic1[' banna '] = 400
print(dic1)
del dic1['apple']
print(dic1)
```

```
dic2 = {'apple': 'red', 'banana': 'yellow',
'orange': 'orange'}
print(dic2['apple'])
print(dic2.keys())
print(dic2.values())
print(dic2. items())

for key in dic2.keys():
print(key)

### get("key") = value function
dic2 = {'apple': 'red', 'banana': 'yellow',
'orange': 'orange'}
print(dic2. get('apple'))

insert = 'cherry'
insert = 'banana'
if (dic2.get(insert) == None):
print(insert + no color information from ')
else:
print(dic2. get(insert))
```

Python basics

10. Set

```
### different from set: dictionary
fruits = {'apple', 'banana', 'orange'}
companies = {'apple', ' microsoft ', 'google'}

fruits & companies # intersection
fruits | companies # union

# followings are not working
s1 = {1, 2, 3, 4, 5, 6, 7}
s2 = {3, 6, 9}
s1 + s2
s1 - s2
```

11. Tuple

```
tuple = (1,2,3,4,5,6,7)
print(tuple)
print(tuple[0])
print(tuple[1:4])
```

12. Class

```
# Class : A collection of [ data ] and [ operations on that data ]
# The [ data ] of the class is called [ member variable ], and [ data manipulation ] is called [ method ]
# class class name :
# def __ init __(self, < argument 1>, < argument 2>, ...):
# < processing to be executed in the constructor >
#def _ Method (self, < argument 1>, < argument 2>, ...):
# < processing to be executed in the method >

### ex1 ###
class Readers:
  def __ init __(self, name):
self.name = name
  def read_book (self):
print(self.name+' is reading!')

reader = Readers('Mike')
reader.read_book ()
```

```
### ex2 ###
class Hello1:
  def __ init __(self, msg ):
self. msg = msg

  def output(self):
print(self. msg)

hello = Hello1("Hello World")
hello.output ()

### ex3 ###
class subway:
  def __ init __(self, line, size):
    self. line = line
    self. size = size

  def sizeup (self, q):
    self.size = self.size + q
return self. size

s1 = subway(1, 6)
s1.sizeup(3)
```

Python basics

13. Matrix

```
import pandas as pd
d = {'x1': [1, 2, 3, 4, 5, 2], 'x2': [0, 0, 1, 1, 0, 0], 'x3': [7, 8, 9, 6, 5, 6] }
df = pd.DataFrame (data=d)

grouped = df ['x1']. groupby ( df ['x2']) #group by x2 for aggregation
grouped.mean () # mean of x1

# (1) Paste the two arrays left to right ( https://rfriend.tistory.com/352)
import numpy as np
a = np.array ([1, 2, 3])
b = np.array ([4, 5, 6])
np.r _[a, b]
np. hstack ([a, b])
np.concatenate ((a, b), axis = 0)

# (2) Paste two arrays from top to bottom
np.r _[[a], [b]]
np.vstack ([a, b])
c = np.array ([[0, 1, 2], [3, 4, 5,]])
d = np.array ([[6, 7, 8], [9, 10, 11]])
np.concatenate ((c, d), axis = 1)

# (3) Create a 2D array by pasting two 1D arrays vertically by column
np.c _[a, b]
np.column_stack ([a, b])
c = np.array ([[0, 1, 2], [3, 4, 5,]])
d = np.array ([[6, 7, 8], [9, 10, 11]])
np.concatenate (( cT , dT ), axis = 1)
```

Python basics

14. (Text) file to make

```
###### file path Setting method ( Three )
# "C:/doit/newfile.txt "
# "C:₩₩doit₩₩ newfile.txt "
# r"C :₩doit₩ newfile.txt "
###### How to set the mode
# r: read mode ( used to read files only )
# w: write mode ( used to write content to a file )
# a: Append mode ( append new content to end of file )
# r+: read-write mode
# b : binary mode ( binary mode )
# t : text mode ( default )
import os
os. getcwd()
os.cwdir('/content')

f = open(" new file .txt", 'w', encoding= " utf-8 " )
f. close ()

# Often used with the with statement
f = open("foo.txt", 'w')
f. write ("Life is too short, you need python ₩n
Is Python better than R?")
f.close ()

with open("foo2.txt", "w") as f:
    f. write ("Life is too short, you need python ₩n
Python is better than R? ")
```

```
# readline use function
# read only the first line
f = open("foo.txt", 'r')
line = f. readline ()
print(line)
f.close ()

# read all lines
f = open("foo.txt", 'r')
while True:
line = f. readline ()
if not line: break
print(line)
f.close ()

#readlines _ Read all lines ( content ) with a function
f = open("foo.txt", 'r')
lines = f. readlines ()
for line in lines:
print(line)
f.close ()

# Read everything using the read function
f = open("foo.txt", 'r')
data = f. read ()
print(data)
f.close ()
```

Python basics

15. Packages , Components

```
# Modules : Python programs in which components are defined
# package : a bundle consisting of several modules

# import < package name > as <alias>
import numpy as np
a = np.array([[1,2,3], [4,5,6]])
print(a)

# from < package name > import < component name >
from numpy import array
a = array([[1,2,3], [4,5,6]])
print(a)
```

16. Colab Save files (PC, Instance , Google Drive)

```
# Upload files from PC to instance
from google.colab import files
uploaded = files.upload ()
! dir    # Check uploaded data


# Download the file of the instance to the PC
from google.colab import files
files.download ('foo.txt')


# Save the instance's files to my Google Drive
from google.colab import drive
drive.mount ('/content/drive')
! cp foo.txt '/content/drive/My₩ Drive/foo.txt'
```

Python basics

17. Github import file

```
from google.colab import drive
drive.mount ('/content/ gdrive ')

! pwd    # Check your current location
#gibhub _ Specify where to store data
%cd "/content/ gdrive /My Drive/ Colab Notebooks/ Textmining / download"

# Copy github repository
! git clone "https://github.com/ attardi / wikiextractor.git "
!ls # check if saved properly
```


[Reference]

1. Crawl the web in one bite (Kim Kyung-rok , Seo Young-deok) BJ Public
2. Artificial intelligence that learns by analyzing AlphaZero (Furukawa Hidekazu / Kam Yeon-soo) J-Pub
3. Web resource : https://rfriend.tistory.com/352
4. Web resource : https://wikidocs.net/26