



인프런 파이토치 딥러닝 1주차 발표 —

**정보보호학과 20학번
김수현**

섹션 0. 파이토치와 구글 코랩

Colab & Pytorch



구글 계정만 있으면 됨
무료 GPU 사용가능
구글 드라이브와 연동 가능
최대 12시간 세션 유지



PyTorch



페이스북이 개발
파이썬 기반
많은 연구에 사용
(코드가 많음)

섹션 1. 파이썬 입문

타입이란?

모든 프로그래밍 언어에서 가장 중요한 개념.
사용하는 모든 것들이 특정 타입을 가지고 있고 타입
에 따라 성질이 다르기 때문에 항상 조심.
예를 들어, 정수형, 실수형, 리스트, 배열 등이 있음.
가장 중요한 것은 우리가 사용하는 타입을 알고 있어
야 함.

예시로 살펴보기

정수형(int) -> ex) a = 1
 b = 0
 c = -1

타입 확인하기 -> print(a, type(a))
 print(b, type(b))
 print(c, type(c))

-> 1 <class 'int'>
 0 <class 'int'>
 -1 <class 'int'>

str -> a = 'deep'
 b = 'learning'

Str의 덧셈 -> 문자를 붙여주는 역할. Str
은 사칙연산 중 덧셈연산만 가능하다.

print(a+b) -> deeplearning

주의할 점!

숫자형 문자 -> a = 1
b = '1'

print(a)

print(b)

-> 1

1

해결방법:

Str 을 int 로 변환하기 -> c = int(b)

print(c, type(c))

-> 1 <class 'int'>

-> 간혹 데이터에 숫자처럼 보이지만 문자인 경우들이 있다.

오류에 당황하지 말고 타입을 살펴본다.

리스트

숫자들의 모임.

ex) `list1 = [1,2,3,4]`

`print(list1, type(list1))`

-> `[1,2,3,4] <class 'int'>`

안에 있는 요소들을 알고 싶을 때 -> `print(list1[0])`

`print(list1[1])`

`print(list1[-1])`

`print(list1[:2])`

튜플

a = (1,2)

***튜플은 한번 선언하면 변경이 불가능.

덮어쓰우기를 방지할 수 있는 장점이 있음.***



딕셔너리

```
a = {"class": ['deep learning', 'machine learning'],  
     "students": [40, 20]}
```

```
print(a["class"]) -> ['deep learning',  
                     'machine learning']
```

```
print(a["students"]) -> [40, 20]
```

```
print(a["student"][0]) -> 40
```

Numpy 배열

가장 많이 쓰이고, 행렬 연산을 할 때 특화된 라이브러리

numpy 라이브러리 불러오기 -> `import numpy`

```
arr = numpy.array( [1,2,3,4] )
```

```
print(arr, type(arr)) -> [1,2,3,4] <class 'numpy.ndarray'>
```

라이브러리

Import를 통해 미리 만들어 놓은 함수와 클래스를 쉽게 사용할 수 있으며 다양한 라이브러리들이 존재한다.

`Import numpy as np` -> as를 통해서 라이브러리를 약자로 사용할 수 있다.

numpy 안에 있는 특정 함수만 불러오기 -> `from numpy import array, ones, zeros`

조건문과 반복문

조건 반복문을 통해 우리가 원하는 규칙을 구현할 수 있다.

Import numpy as np

X = 10

If x == 1:

 print("x = 1")

Else:

 print("x != 1")

for i in range(5):

-> range(5) = [0,1,2,3,4] list

print(i) -> 0 1 2 3 4

i = 0

while i<5: -> 조건이 맞는 경우에만 실행

 print(i)

 i += 1

-> 0 1 2 3 4 (break)

Break문 & try-except문

break 문

짝수가 나오면 바로 종료하도록 만들기

```
for i in pending:  
    if i % 2 == 0:  
        print(i, "입니다.")  
        break
```

try-except문

내가 원하지 않는 사항들을 배제하고 싶을 때

```
x = [1,2,3,"s",4,5]  
for i in x:  
    try:  
        y = i + 1  
        print(y)  
    except:  
        print(i, "는 오류입니다.")
```

-> 2 3 4 s는 오류입니다. 5 6

함수

코딩을 하다 보면 동일한 연산을 자주 사용하거나 문장이 길어지는 논리가 있을 수 있다.

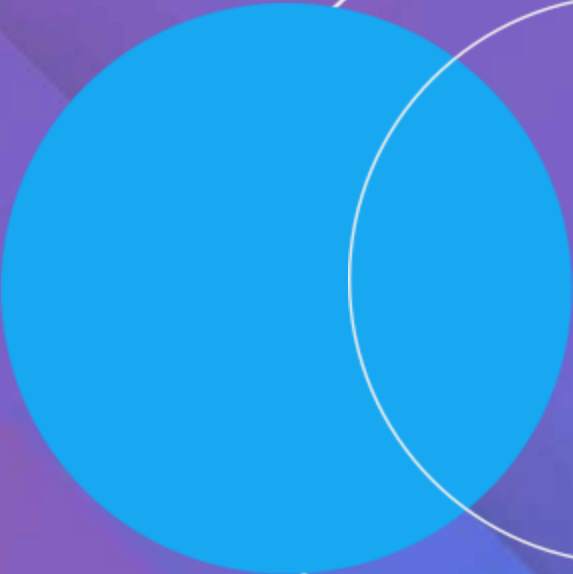
이 때 우리는 함수 하나를 정의해 코드를 쉽게 알아볼 수 있게 하고 필요할 때마다 편리하게 미리 정의된 함수를 불러와서 사용할 수 있다.

ex)

```
def objective_function(x):  
    Return np.sin(x) + np.cos(x) +.5*np.exp(x)
```

함수

1. 함수의 첫 글자는 소문자로 시작한다.
2. #으로 주석을 달 수 있다.
3. 값은 return으로 얻을 수 있다.



**지금까지
발표를 들어주셔서——
감사합니다!**