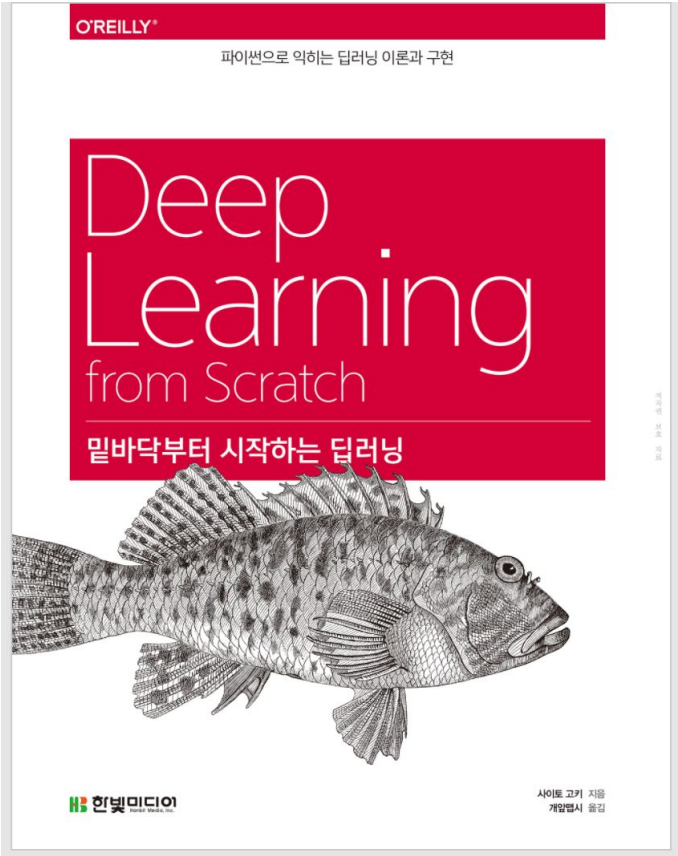
딥러닝을 함께 공부해보세~
영차영차 1일차

0. 밑바닥 부터 시작하는 딥러닝



1. 설명이 꼼꼼하다

2. 책이 얇다

3. TensorFlow를 설치하지 않아도 된다.

- 바탕이 되는 이론들을 Python으로 간단하게 구현하고 이미지트레이닝(MNIST)까지 시켜볼 수 있다.

1. 파이썬 기본

1. 파이썬 기본

산술 연산

```
>>> 1 - 2
-1
>>> 4 * 5
20
>>> 7 / 5
1
>>> 3 ** 2
9
```

1. 파이썬 기본

변수

```
>>> x = 10
>>> print(x)
 10
>>> y = 3.14
>>> print(y)
 3.14
```

1. 파이썬 기본

리스트

```
>>> a = [1, 2, 3, 4, 5] #리스트 생성
>>> print(a) #리스트의 내용 출력
 [1, 2, 3, 4, 5]
>>> a[0]
 1
>>> a[4]
 5
>>> a[4] = 99
>>> print(a)
 [1, 2, 3, 4, 99]
```

```
>>> a[0:2]
 [1, 2]
>>> a[1:]
 [2, 3, 4, 99]
>>> a[:3]
 [1, 2, 3]
>>> a[:-1]
 [1, 2, 3, 4]
>>> a[:-2]
 [1, 2, 3]
```

1. 파이썬 기본

딕셔너리

```
>>> me = {'weight' : 70}
>>> me['weight']
 70
>>> me['height'] = 180
>>> print(me)
 {'weight': 70, 'height': 180}
```

1. 파이썬 기본

함수

```
>>> def hello() :
...     print("hello charlie")
...
>>> hello()
hello charlie
```

1. 파이썬 기본

클래스

```python
class Man:
    def __init__(self, name):
        self.name = name
        print("Initialized!")

    def hello(self):
        print("Hello " + self.name + "!")

    def goodbye(self):
        print("Good-bye " + self.name + "!")

m = Man("David")
m.hello()
m.goodbye()
```

```
HiJiGOO-MacBook-Pro:seminar Hi.JiGOO$ python lec1.py
Initialized!
Hello David!
Good-bye David!
```

1. 파이썬 기본

넘파이 (외부 라이브러리) : 다차원 배열을 다루는 편리한 메서드를 많이 제공한다

```
>>> import numpy as np
>>>
>>> x = np.array([1.0, 2.0, 3.0])
>>> print(x)
 [ 1.  2.  3.]
>>> y = np.array([2.0, 4.0, 6.0])
>>> print(y)
 [ 2.  4.  6.]
>>> x - y
 array([-1., -2., -3.])
>>> x * y
 array([  2.,   8.,  18.])
>>> x / y
 array([ 0.5,  0.5,  0.5])
>>> x / 2.0
 array([ 0.5,  1. ,  1.5])
```

1. 파이썬 기본

넘파이 (외부 라이브러리) - N 차원 배열

```
>>> A = np.array([[1, 2], [3, 4]])
>>> print(A)
 [[1 2]
  [3 4]]
>>> B = np.array([[3, 0], [0, 6]])
>>> print(B)
 [[3 0]
  [0 6]]
```

```
>>> A + B
 array([[ 4,  2],
        [ 3, 10]])
>>> A * B
 array([[ 3,  0],
        [ 0, 24]])
```

```
>>> A.shape
 (2, 2)
>>> B.shape
 (2, 2)
```

1. 파이썬 기본

넘파이 (외부 라이브러리) -
브로드캐스트

1	2
3	4

*

10	20

=

1	2
3	4

*

10	20
10	20

=

10	40
30	80

```
>>> A = np.array([[1, 2], [3, 4]])
>>> B = np.array([10, 20])
>>> A * B
array([[10, 40],
       [30, 80]])
```

1. 파이썬 기본

넘파이 (외부 라이브러리) - 원소 접근

```
>>> X = np.array([[51, 55], [14, 19], [0, 4]])
>>> print(X)
 [[51 55]
  [14 19]
  [ 0  4]]
>>> X[0]
array([51, 55])
>>> X[0][1]
55
```

```
>>> for row in X :
...     print(row)
...
[51 55]
[14 19]
[0 4]
```

```
>>> X = X.flatten()          #X를 1차원 배열로 변환
>>> print(X)
 [51 55 14 19  0  4]
>>> X[np.array([0, 2, 4])]   #인덱스가 0, 2, 4인 원소 얻기
array([51, 14,  0])
```

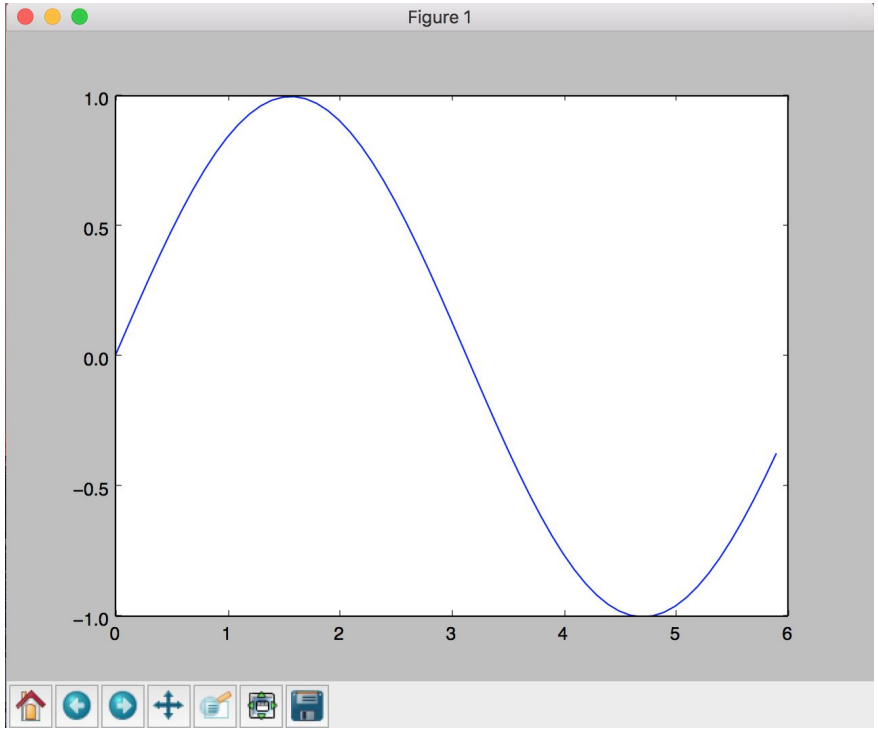
1. 파이썬 기본

matplotlib - 단순한 그래프 그리기

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

# 데이터 준비
x = np.arange(0, 6, 0.1) #0에서 6까지 0.1 간격으로 생성
y = np.sin(x)

# 그래프 그리기
plt.plot(x, y)
plt.show()
```



1. 파이썬 기본

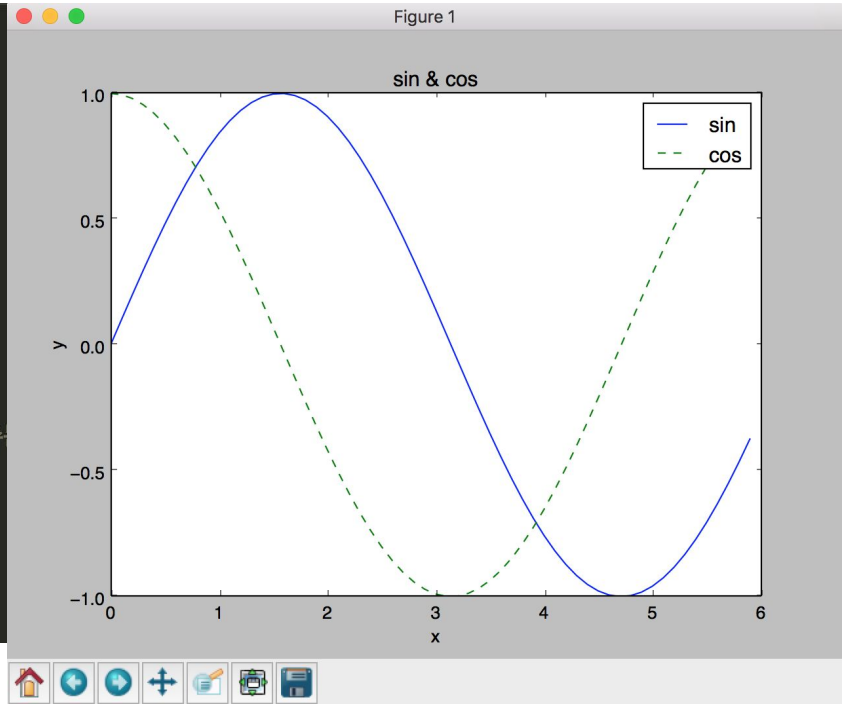
matplotlib - 추가 기능

```python
# coding: utf-8

import numpy as np
import matplotlib.pyplot as plt

# 데이터 준비
x = np.arange(0, 6, 0.1) #0에서 6까지 0.1 간격으로 생성
y1 = np.sin(x)
y2 = np.cos(x)

# 그래프 그리기
plt.plot(x, y1, label="sin")
plt.plot(x, y2, linestyle="--", label="cos") #cos 함수
plt.xlabel("x") # x축 이름
plt.ylabel("y") # y축 이름
plt.title("sin & cos") #제목
plt.legend()
plt.show()
```



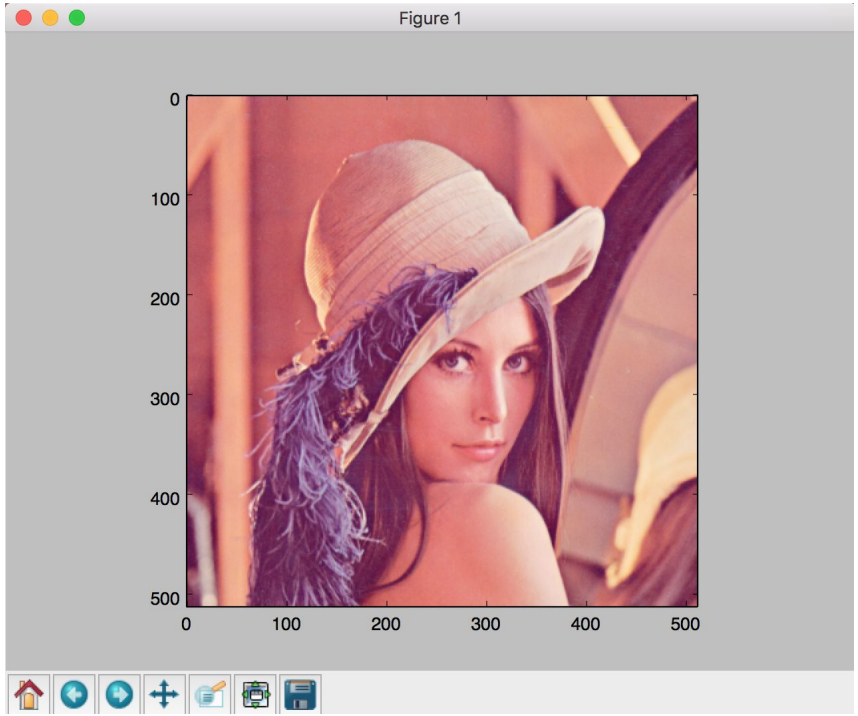
1. 파이썬 기본

이미지 표시하기

```
import matplotlib.pyplot as plt
from matplotlib.image import imread

img = imread('lena.png')

plt.imshow(img)
plt.show()
```



2. 퍼셉트론

2. 퍼셉트론

퍼셉트론(perceptron)은 인공신경망의 한 종류로서, 1957년에 코넬 항공 연구소(Cornell Aeronautical Lab)의 프랑크 로젠블라트 (Frank Rosenblatt)에 의해 고안되었다.

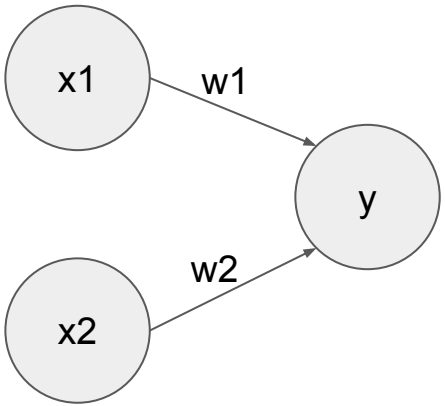
> 신경망(딥러닝)의 기원이 되는 알고리즘

2. 퍼셉트론

퍼셉트론이란?

> 다수의 신호를 입력으로 받아 하나의 신호를(흐른다(1)/안 흐른다(0)) 출력한다.

> x1, x2는 입력신호에 대해 w1, w2가중치를 갖는다.
총합이 정해진 한계(임계값: θ)를 넘어설 때만 1을 출력한다.



입력이 2개인
퍼셉트론

$$y = \begin{cases} 0 & (w1x1 + w2x2 \le \theta) \\ 1 & (w1x1 + w2x2 > \theta) \end{cases}$$

수식으로 나타냄

2. 퍼셉트론

퍼셉트론을 활용한 논리회로 : AND 게이트

> AND 게이트를 퍼셉트론으로 표현하고 싶다. 진리표대로 작동하는 w1, w2, θ의 값을 정의해야한다.

x1	x2	y
0	0	0
0	1	0
1	0	0
1	1	1

ex) (w1, w2, θ) = (0.5, 0.5, 0.7)

$$y = \begin{cases} 0 & (0.5 * x1 + 0.5 * x2 <= 0.7) \\ 1 & (0.5 * x1 + 0.5 * x2 > 0.7) \end{cases}$$

ex) (w1, w2, θ) = (0.5, 0.5, 0.7),
(0.5, 0.5, 0.8),
(1.0, 1.0, 1.0)

2. 퍼셉트론

퍼셉트론을 활용한 논리회로 : NAND 게이트

> NAND 게이트를 퍼셉트론으로 표현하고 싶다. 진리표대로 작동하는 w1, w2, θ의 값을 정의해야한다.

x1	x2	y
0	0	1
0	1	1
1	0	1
1	1	0

ex) (w1, w2, θ) = (-0.5, -0.5, -0.7)

$$y = \begin{cases} 0 & (-0.5 * x1 + -0.5 * x2 <= -0.7) \\ 1 & (-0.5 * x1 + -0.5 * x2 > -0.7) \end{cases}$$

2. 퍼셉트론

퍼셉트론을 활용한 논리회로 : OR 게이트

> OR 게이트를 퍼셉트론으로 표현하고 싶다. 진리표대로 작동하는 w1, w2, θ의 값을 정의해야한다.

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

ex) (w1, w2, θ) = (0.5, 0.5, 0.2)

$$y = \begin{cases} 0 & (0.5 * x1 + 0.5 * x2 <= 0.2) \\ 1 & (0.5 * x1 + 0.5 * x2 > 0.2) \end{cases}$$

2. 퍼셉트론

간단하게 퍼셉트론 구현하기 : AND

```
def AND(x1, x2):
    w1, w2, theta = 0.5, 0.5, 0.7
    tmp = x1*w1 + x2*w2
    if tmp <= theta:
        return 0
    elif tmp > theta:
        return 1

AND(0, 0) # 0을 출력
AND(1, 0) # 0을 출력
AND(0, 1) # 0을 출력
AND(1, 1) # 1을 출력
```

2. 퍼셉트론

NAND게이트와 OR을 구하기 전에..

>앞으로를 생각해서 .. θ 를 -b 로 치환 & 배열로 연산

$$y = \begin{cases} 0 & (w1x1 + w2x2 <= \boldsymbol{\theta}) \\ 1 & (w1x1 + w2x2 > \boldsymbol{\theta}) \end{cases} \Rightarrow y = \begin{cases} 0 & (w1x1 + w2x2 \boldsymbol{+ b} <= 0) \\ 1 & (w1x1 + w2x2 \boldsymbol{+ b} > 0) \end{cases}$$

$$w1x1 + w2x2 \boldsymbol{+ b} \Rightarrow [x1, x2] * [w1, w2] + b$$

2. 퍼셉트론

다시 만든 AND 게이트 그리고 NAND, OR 게이트

```python
def AND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.7
    tmp = np.sum(w * x) + b
    if tmp <= 0:
        return 0
    else:
        return 1
```

```python
def NAND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([-0.5, -0.5])
    b = 0.7
    tmp = np.sum(w * x) + b
    if tmp <= 0:
        return 0
    else:
        return 1
```

AND와는 가중치 (W와 b) 만 다르다

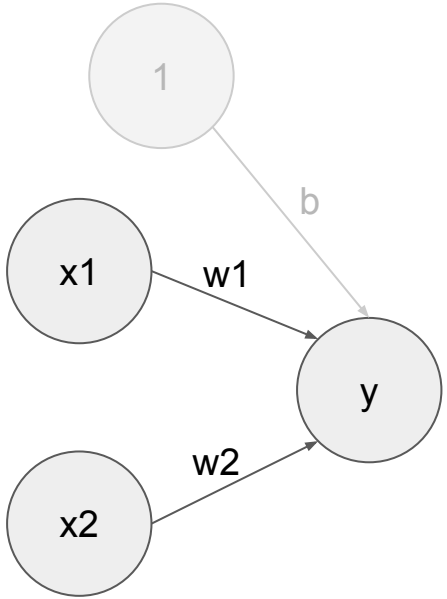
```python
def OR(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.2
    tmp = np.sum(w * x) + b
    if tmp <= 0:
        return 0
    else:
        return 1
```

AND와는 가중치 (W와 b) 만 다르다

2. 퍼셉트론

한계 > XOR는 표현할 수 없다 ?

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	0



$$y = \begin{cases} 0 & (w1x1 + w2x2 + \mathbf{b} <= 0) \\ 1 & (w1x1 + w2x2 + \mathbf{b} > 0) \end{cases}$$

2. 퍼셉트론

한계 > XOR는 표현할 수 없다 ? 조합하면 가능하다.

그림 2-9 AND, NAND, OR 게이트 기호

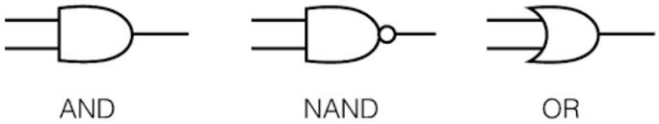
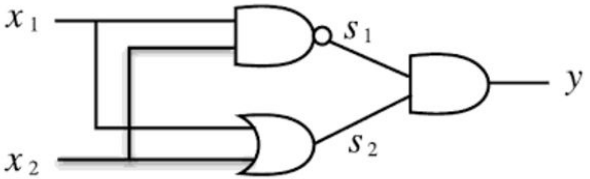


그림 2-11 AND, NAND, OR 게이트를 조합해 구현한 XOR 게이트



x1	x2	s1	s2	y
0	0	1	0	0
0	1	1	1	1
1	0	1	1	1
1	1	0	1	0

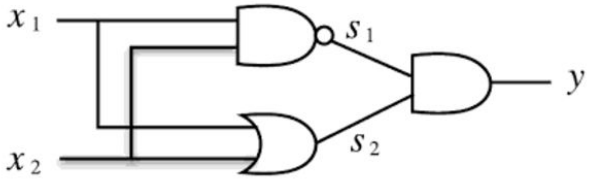
2. 퍼셉트론

조합으로 퍼셉트론 구현하기 : XOR

```
def XOR(x1, x2):
    s1 = NAND(x1, x2)
    s2 = OR(x1, x2)
    y = AND(s1, s2)
    return y
```

```
XOR(0, 0) # 0을 출력
XOR(1, 0) # 1을 출력
XOR(0, 1) # 1을 출력
XOR(1, 1) # 0을 출력
```

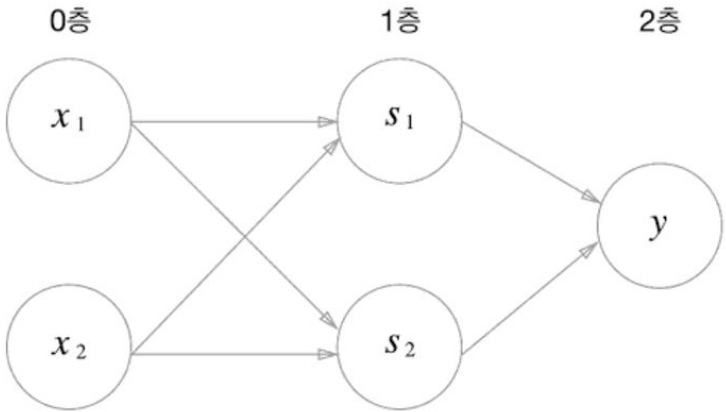
그림 2-11 AND, NAND, OR 게이트를 조합해 구현한 XOR 게이트



2. 퍼셉트론

XOR의 퍼셉트론 : 다층 퍼셉트론

그림 2-13 XOR의 퍼셉트론



1. 0층의 두 뉴런이 입력 신호를 받아 1층의 뉴런으로 신호를 보낸다.
2. 1층의 뉴런이 2층의 뉴런으로 신호를 보내고, 2층의 뉴런은 이 입력 신호를 바탕으로 y를 출력한다.

> 단측 퍼셉트론으로는 표현하지 못한 것을 층을 하나 늘려 구현할 수 있다.

2. 퍼셉트론

퍼셉트론은 신경망의 한 분류 : 다음이야기는 **신경망**

가중치 W를 매번 사람이 수동으로 해야 하나? **신경망이 자동으로 학습**

https://ko.wikipedia.org/wiki/%EC%9D%B8%EA%B3%B5%EC%8B%A0%EA%B2%BD%EB%A7%9D