신경망

목차

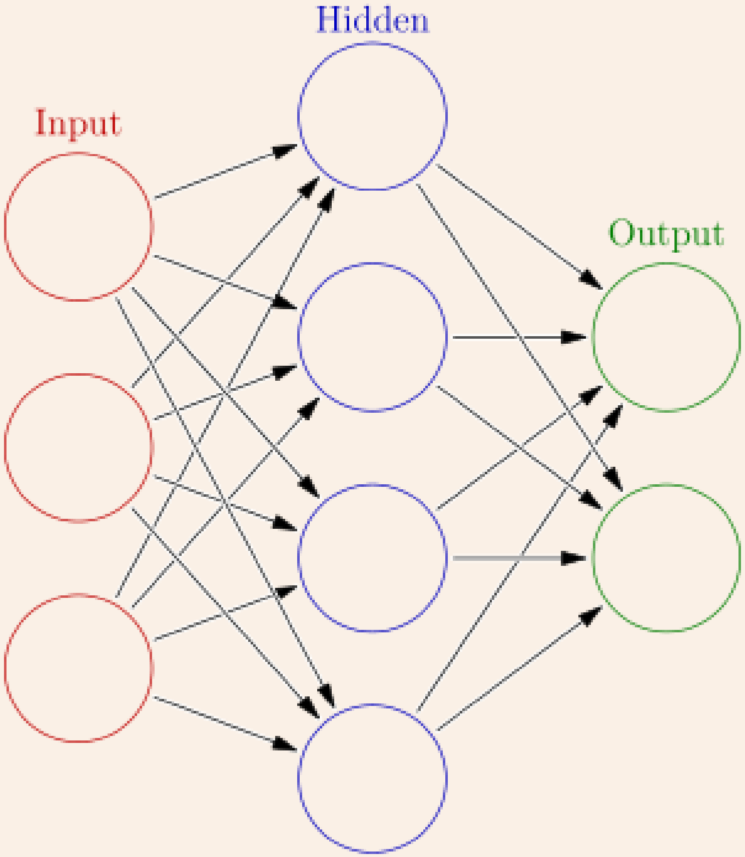
신경망

활성화함수

3층 신경망 구현

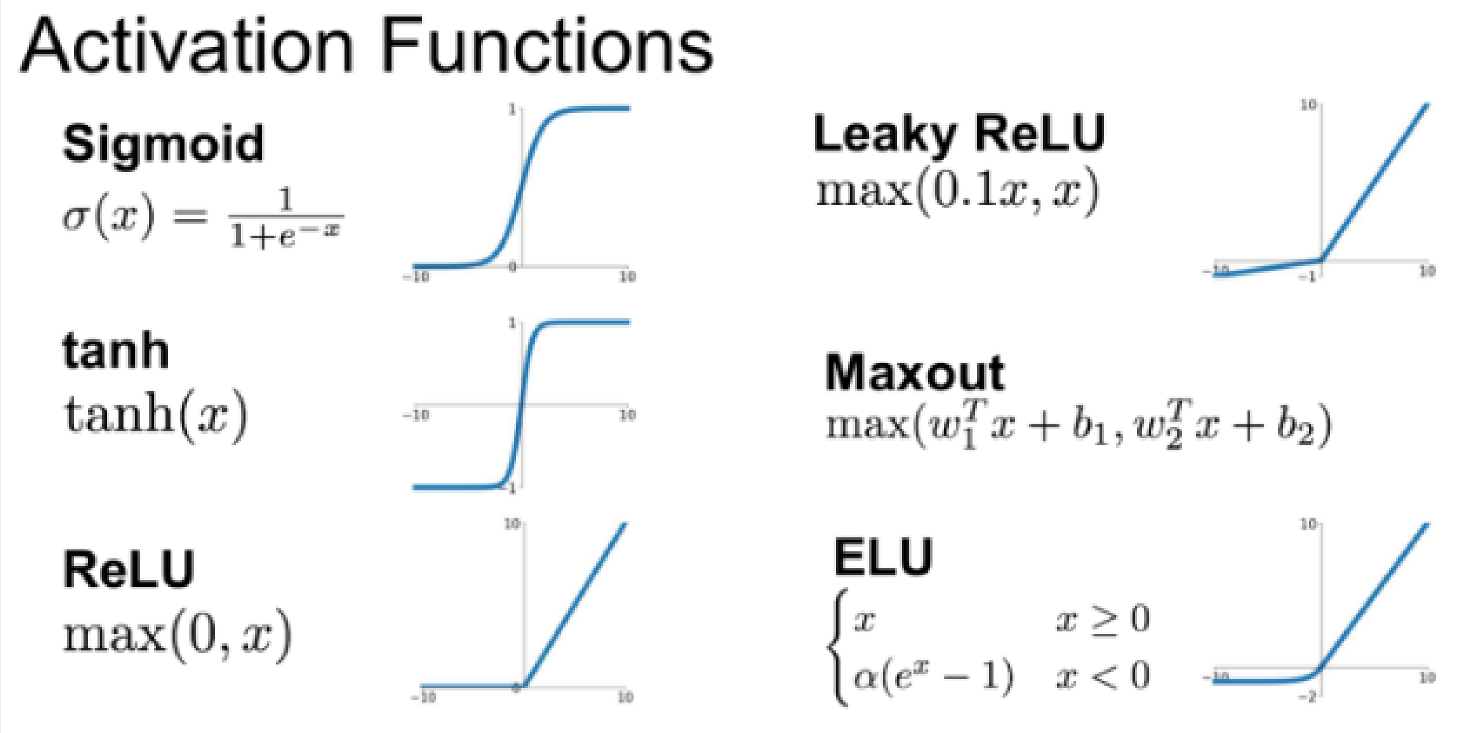
신경망

신경망? 다층 퍼셉트론?



활성화 함수

활성화 함수(Activation Function)



Activation Functions

Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$

tanh

$\tanh(x)$

ReLU

$\max(0, x)$

Leaky ReLU

$\max(0.1x, x)$

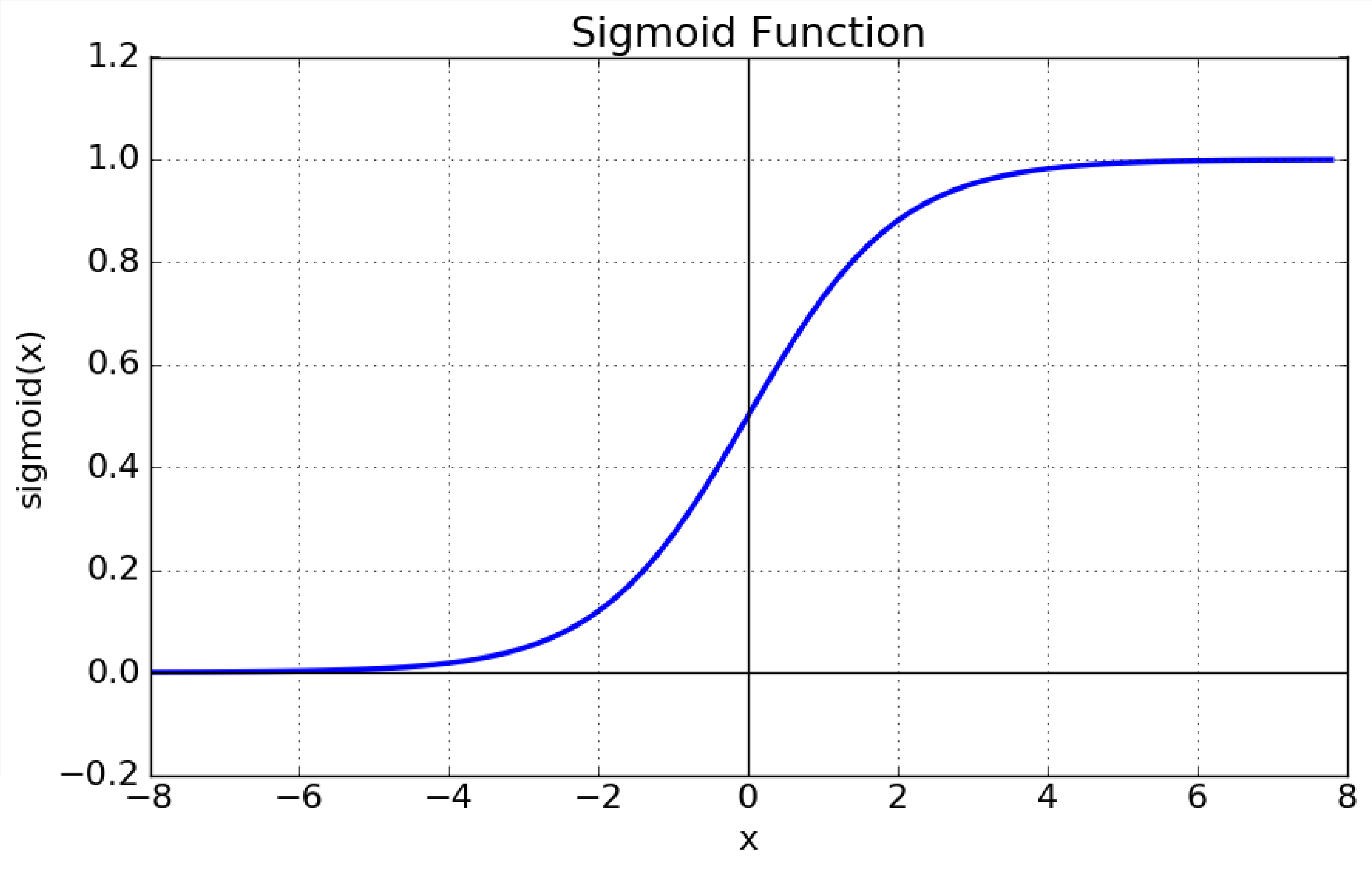
Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

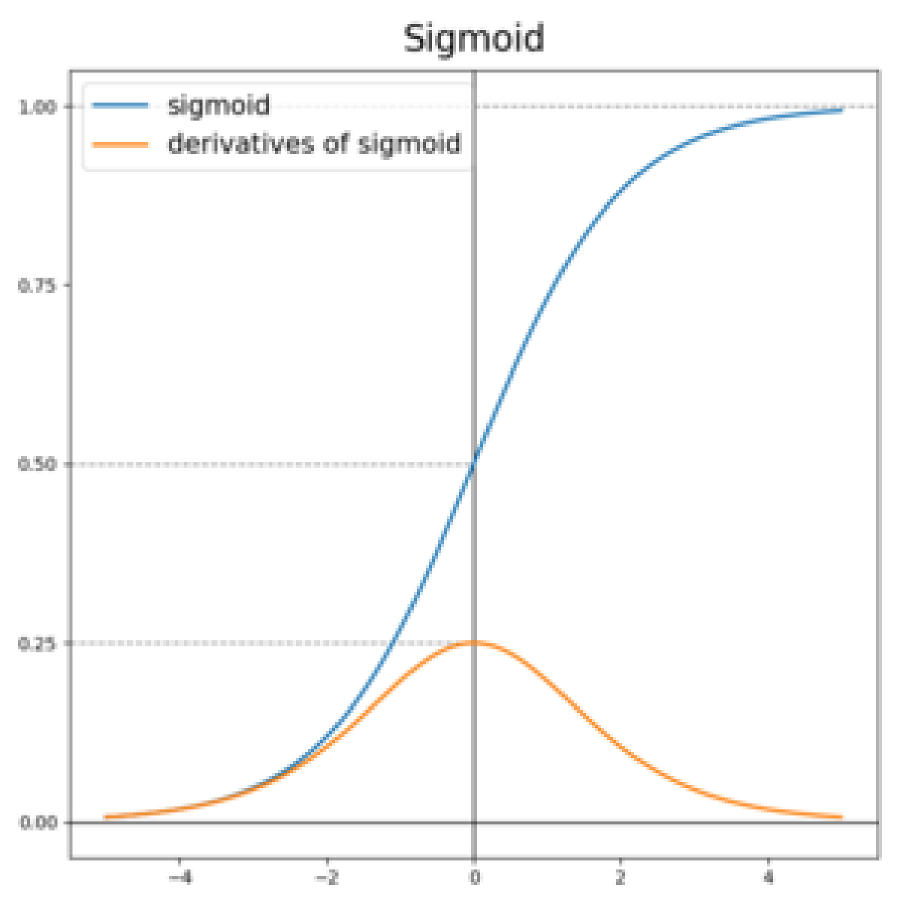
활성화 함수



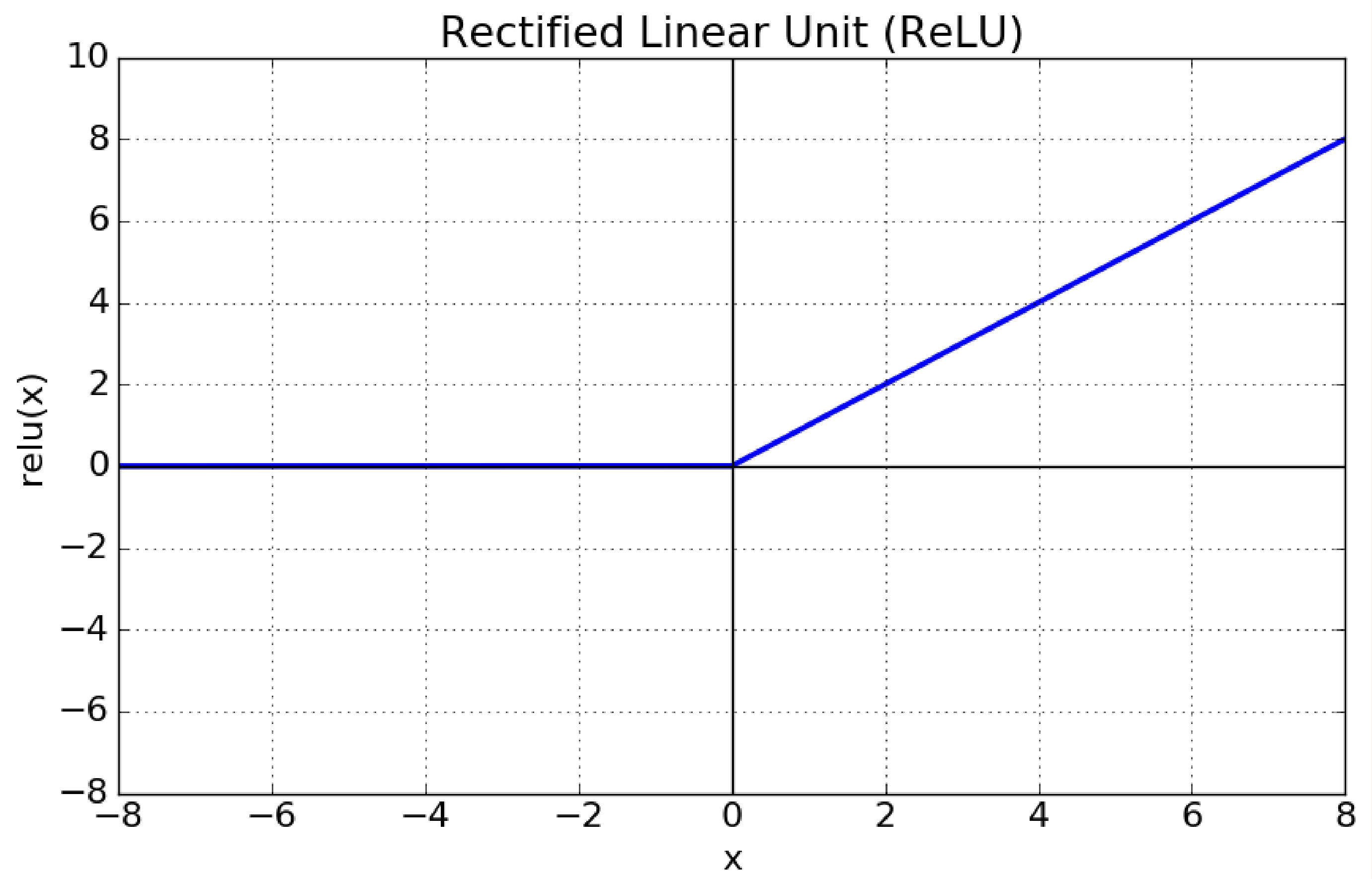
Sigmoid

- $a = \dfrac{1}{1+e^{-z}}$

내 용


Sigmoid
sigmoid
derivatives of sigmoid
1.00
0.75
0.50
0.25
0.00
−4
−2
0
2
4

활성화 함수

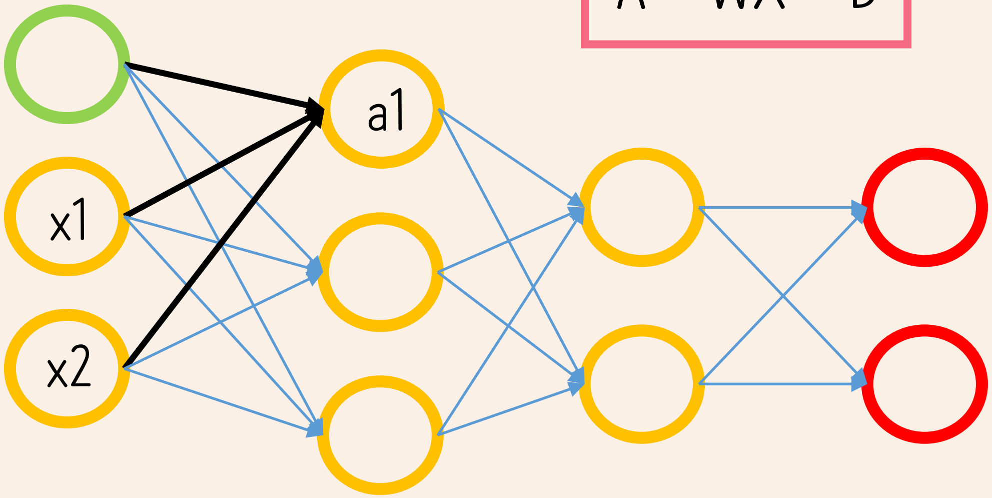


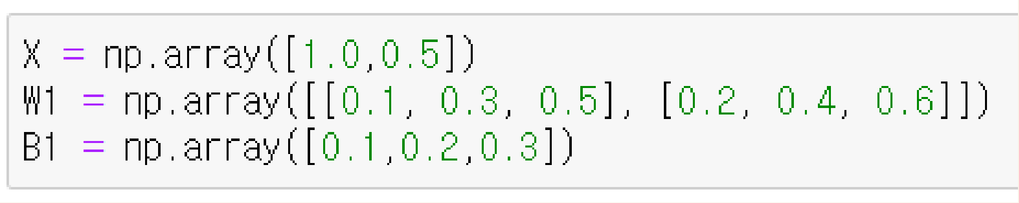
ReLU

- $a = max(0, z)$

3층 신경망 구현

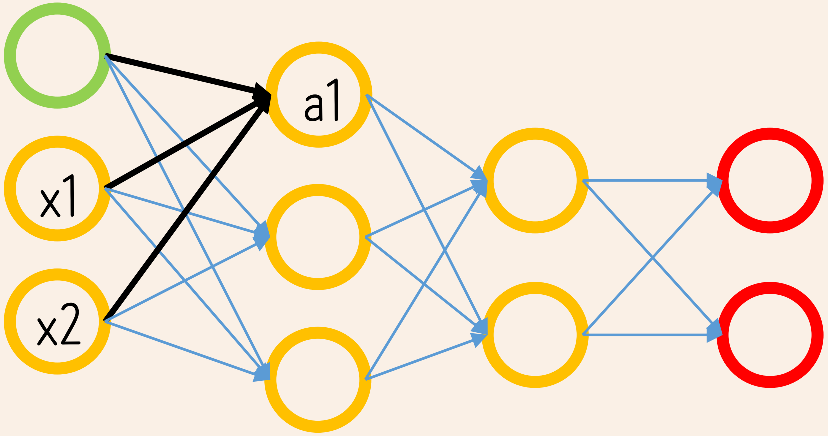
$$A = WX + B$$

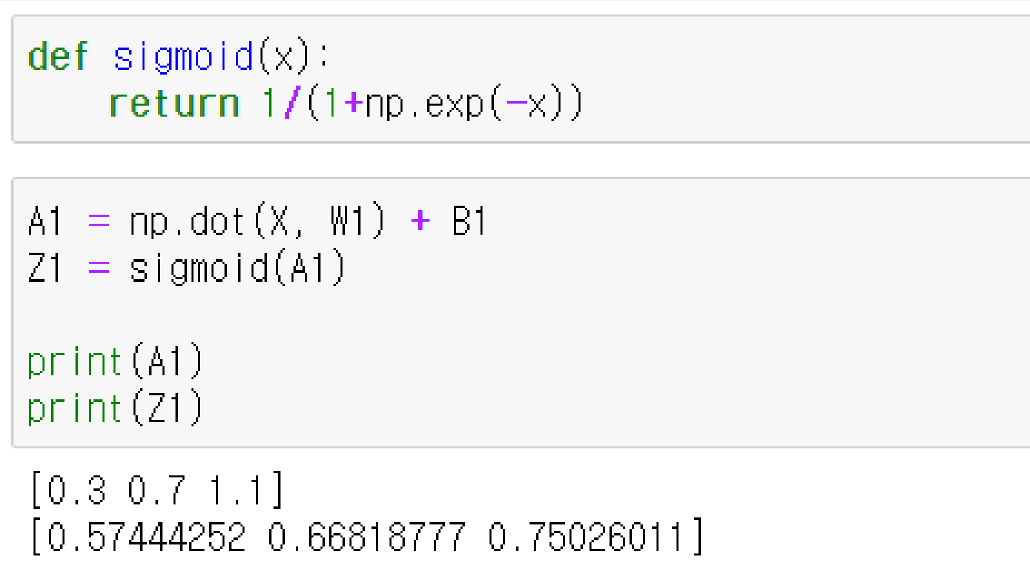




```
X = np.array([1.0,0.5])
W1 = np.array([[0.1, 0.3, 0.5], [0.2, 0.4, 0.6]])
B1 = np.array([0.1,0.2,0.3])
```

3층 신경망 구현





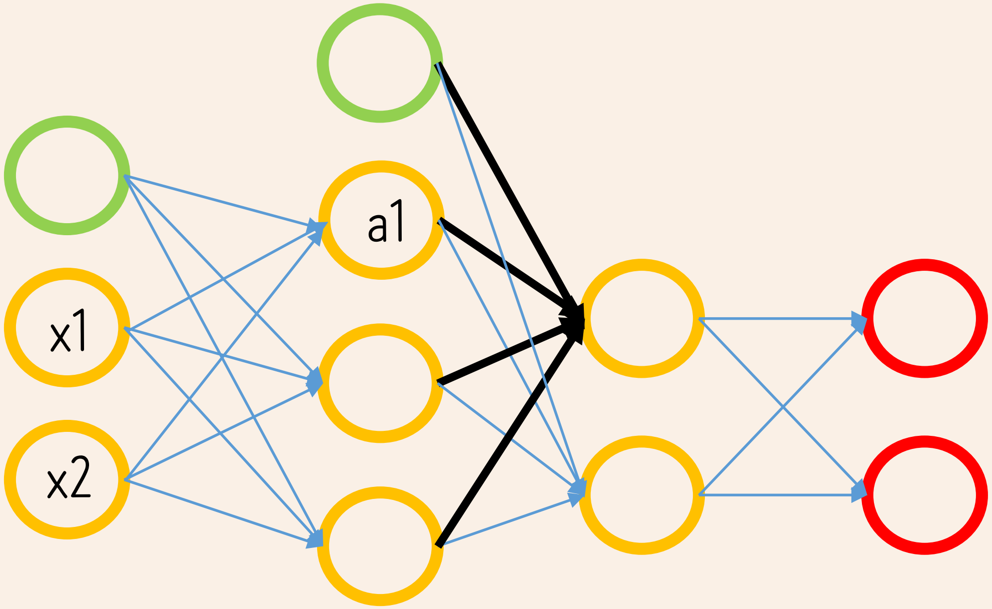
```
def sigmoid(x):
    return 1/(1+np.exp(-x))
```

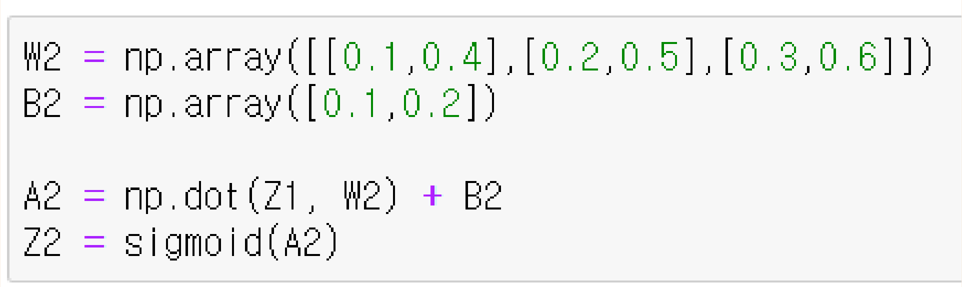
```
A1 = np.dot(X, W1) + B1
Z1 = sigmoid(A1)

print(A1)
print(Z1)
```

```
[0.3 0.7 1.1]
[0.57444252 0.66818777 0.75026011]
```

3층 신경망 구현

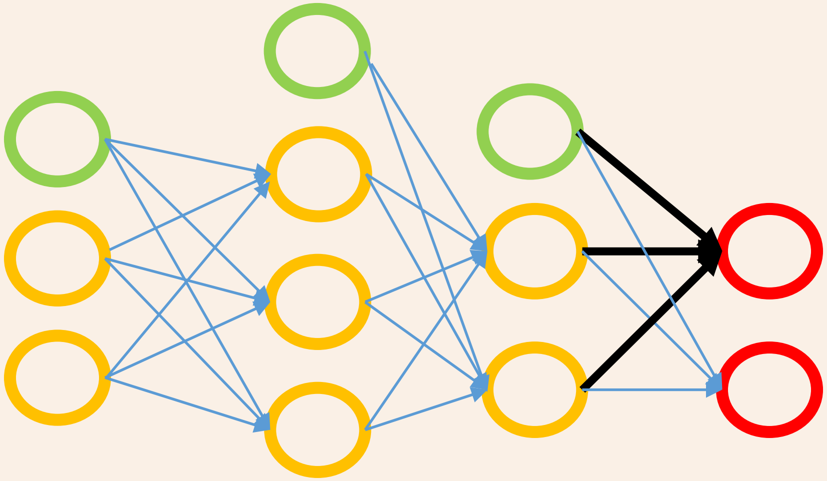


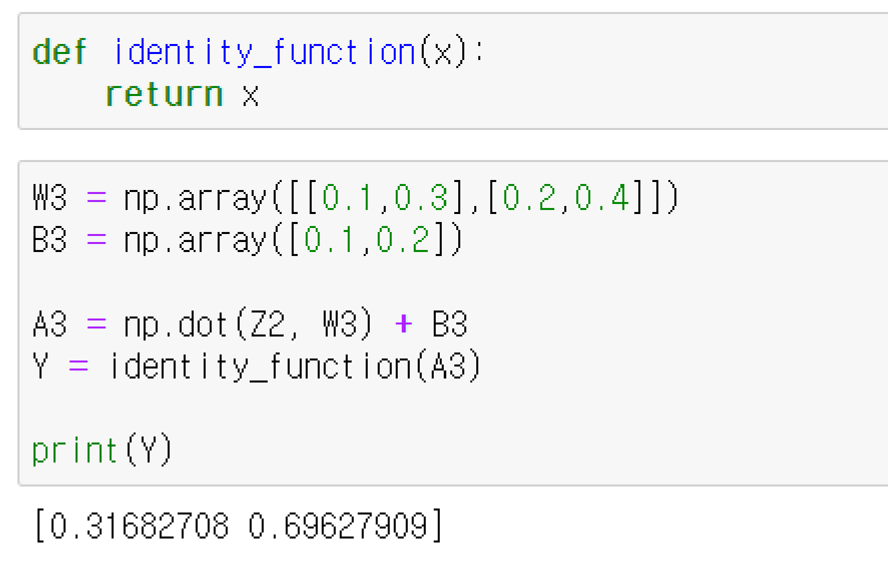


```
W2 = np.array([[0.1,0.4],[0.2,0.5],[0.3,0.6]])
B2 = np.array([0.1,0.2])

A2 = np.dot(Z1, W2) + B2
Z2 = sigmoid(A2)
```

3층 신경망 구현





```
def identity_function(x):
    return x
```

```
W3 = np.array([[0.1,0.3],[0.2,0.4]])
B3 = np.array([0.1,0.2])

A3 = np.dot(Z2, W3) + B3
Y = identity_function(A3)

print(Y)
```

```
[0.31682708 0.69627909]
```

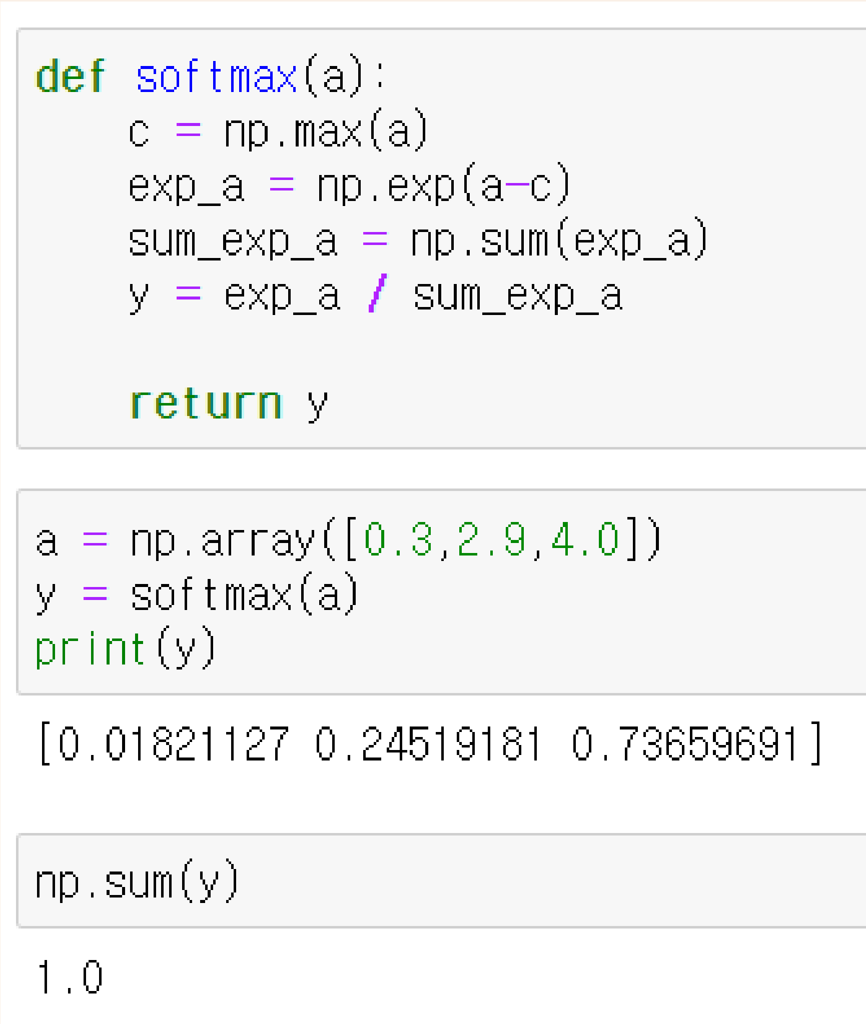
3층 신경망 구현

회귀 – 항등 함수

2클래스 분류 – 시그모이드 함수

다중 클래스 분류 – **소프트맥스 함수**

$$f(\vec{x})_i = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}} \qquad \text{for } i = 1, \ldots, K$$



```
def softmax(a):
    c = np.max(a)
    exp_a = np.exp(a-c)
    sum_exp_a = np.sum(exp_a)
    y = exp_a / sum_exp_a

    return y
```

```
a = np.array([0.3,2.9,4.0])
y = softmax(a)
print(y)
```

```
[0.01821127 0.24519181 0.73659691]
```

```
np.sum(y)
```

```
1.0
```

Thank You