

02. 활성화 함수

1. Sigmoid

$$y = \frac{1}{1 + e^{-x}}$$

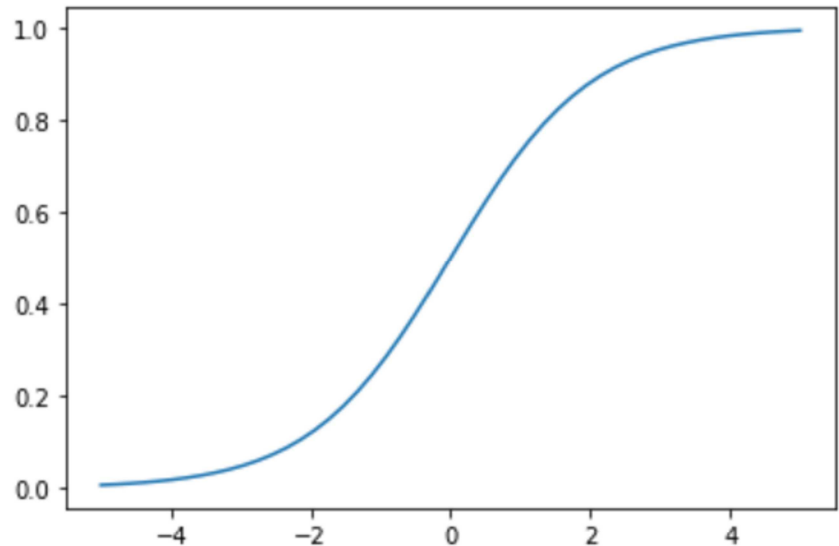
$$0 < y < 1$$

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

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

x = np.linspace(-5, 5)
y = sigmoid_function(x)

plt.plot(x, y)
plt.show()
```



● 시그모이드 함수 y 의 도함수는 $y' = (1 - y)y$

손쉽게 미분가능하여 신경망분야에서 많이 사용

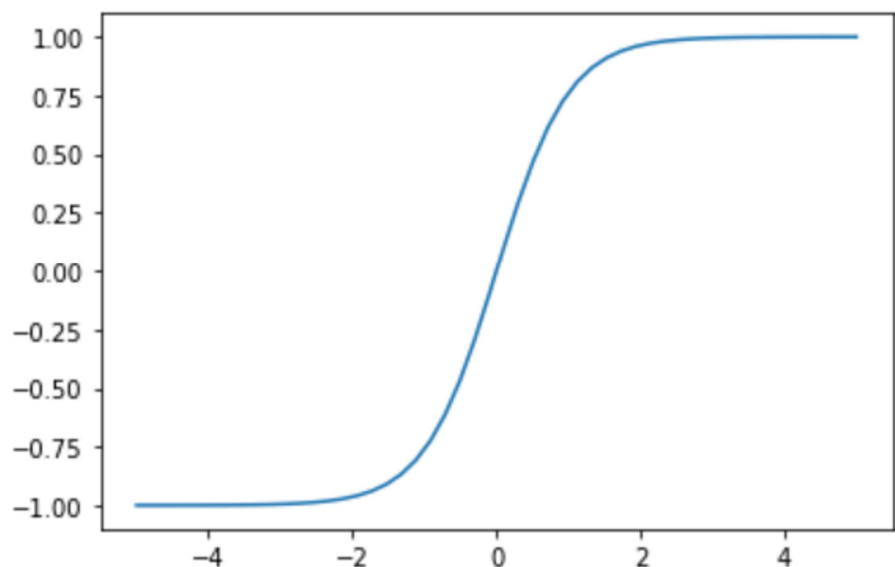
2. tanh

$$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$-1 < y < 1$$

```
def tanh_function(x):
    return np.tanh(x)

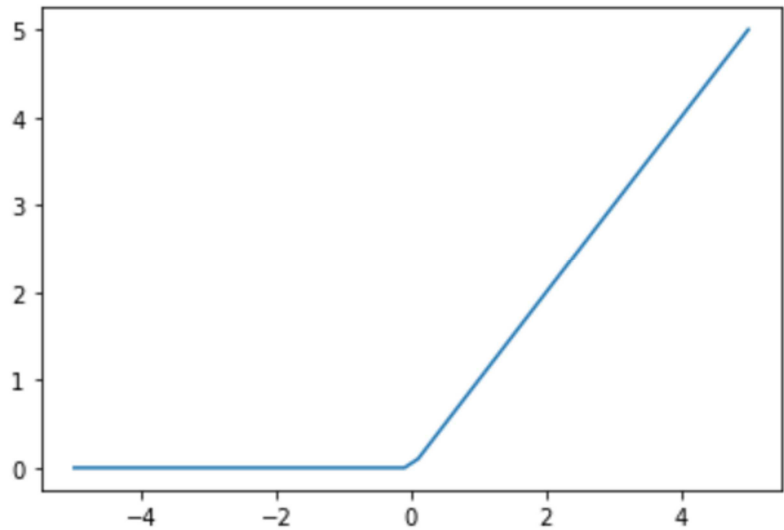
x = np.linspace(-5, 5)
y = tanh_function(x)
plt.plot(x, y)
plt.show()
```



3. ReLU(Rectified Linear Unit)

$$y = \begin{cases} 0 & (x \leq 0) \\ x & (x > 0) \end{cases}$$

```
def relu_function(x):  
    return np.where(x <= 0, 0, x)  
  
x = np.linspace(-5, 5)  
y = relu_function(x)  
  
plt.plot(x, y)  
plt.show()
```

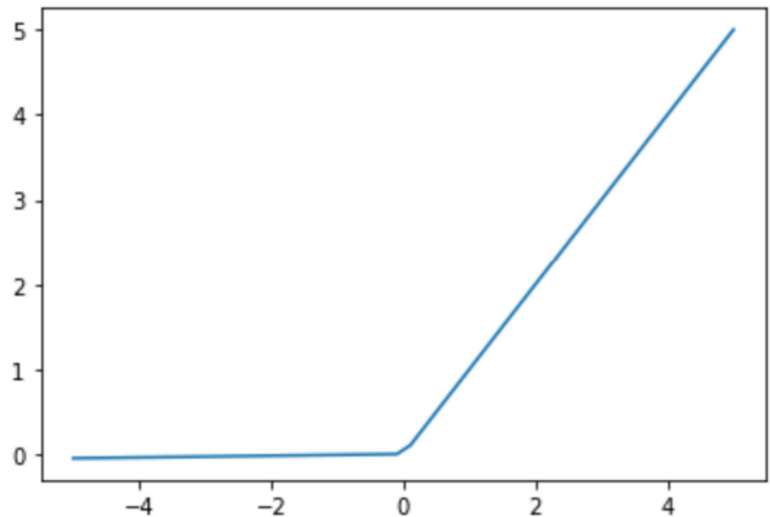


4. Leaky ReLU

- x가 음수인 영역에서 아주 작은 기울기를 생성
- 출력이 0이 되어 더 이상 학습이 진행되지 않는 뉴런이 발생하는 dying ReLU 현상을 피하기 위해 사용

$$y = \begin{cases} 0.01x & (x \leq 0) \\ x & (x > 0) \end{cases}$$

```
def leaky_relu_function(x):  
    return np.where(x <= 0, 0.01*x, x)  
  
x = np.linspace(-5, 5)  
y = leaky_relu_function(x)  
  
plt.plot(x, y)  
plt.show()
```



5. Softmax

- 입력받은 값을 출력으로 0~1사이의 값으로 모두 정규화하며 출력 값들의 총합은 항상 1이 되는 특성을 가진 함수 (확률 산출해주는 함수)

```
def softmax_function(x):  
    return np.exp(x)/np.sum(np.exp(x)) # 소프트맥스함수  
  
y = softmax_function(np.array([1,2,3]))  
print(y)
```

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