인공신경망 모형
Artificial Neural Network

Fiery_Data Analysis

이 정재
2023.07.09 (일)

Step 1

인공신경망 정의

Step 2

핵심 용어

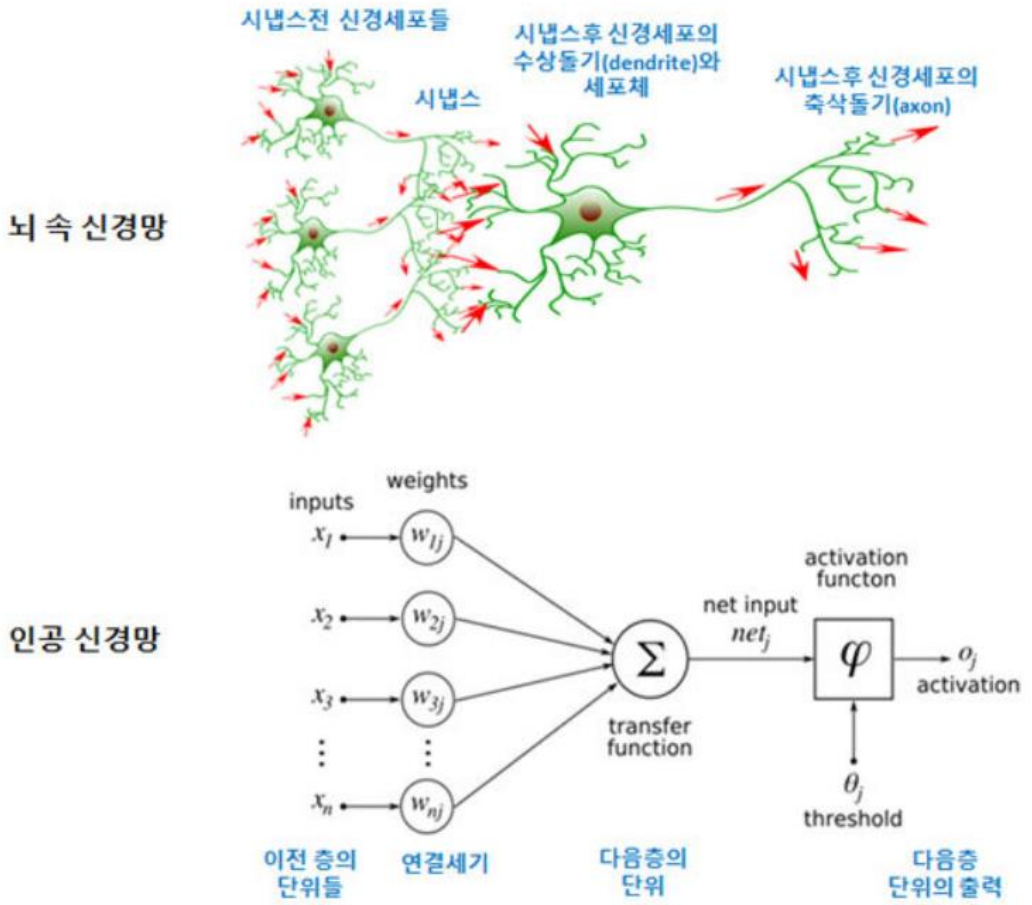
Step 3

활성화 함수

Step 4

정리

인공신경망 모형 정의



출처 https://commons.wikimedia.org/wiki/File:ANN_neuron.svg. https://en.wikibooks.org/wiki/

배경

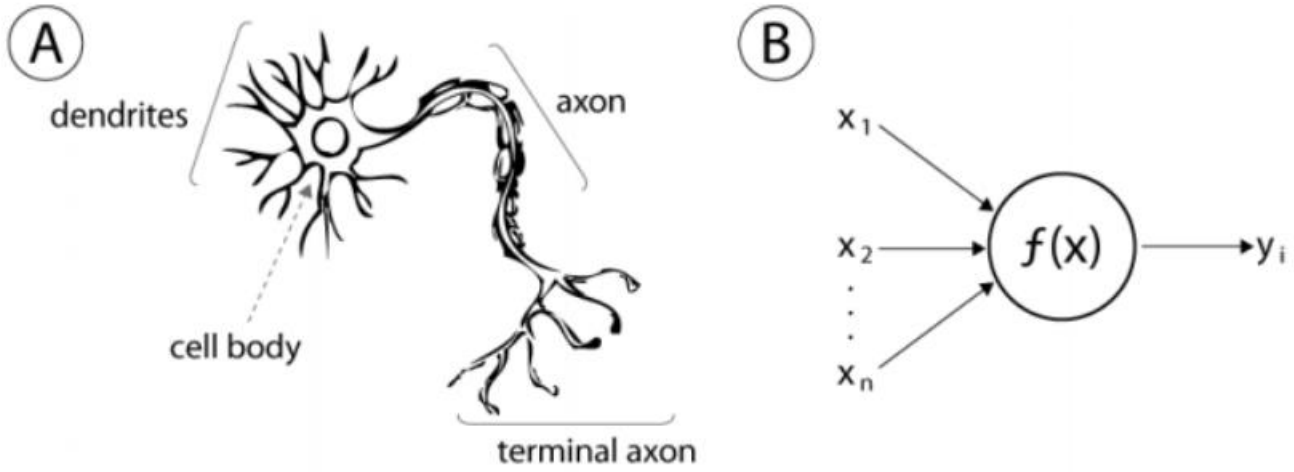
인간의 뉴런의 자극전달 과정에 아이디어를 착안하여 발생한 머신러닝 알고리즘

인간의 뉴런은 시냅스를 통하여 다른 뉴런으로부터 자극을 전달받고 시냅스를 통하여 다른 뉴런에게 자극을 전달하는 과정을 통해서 학습을 진행

이 자극 전달의 과정을 알고리즘에서 layer와 perceptron으로 뉴런과 시냅스를 구성하여 연결지은 것

- 하나의 뉴런은 하나의 perceptron과 대응
- 시냅스의 역할은 여러 layer를 잇는 weight(가중치)와 bias(편향, error)가 한다.

인공신경망 모형 정의



A 인간의 뉴런 형상화

C 뉴런들의 연결

B X1 ~ Xn의 데이터가 들어가서 Yi 하나를 출력하는 하나의 perceptron 형태

D 여러 perceptron을 여러 층에 걸쳐서 연결시킨 Multi Layer Perceptron

가중치 Weight

4월

5월

6월

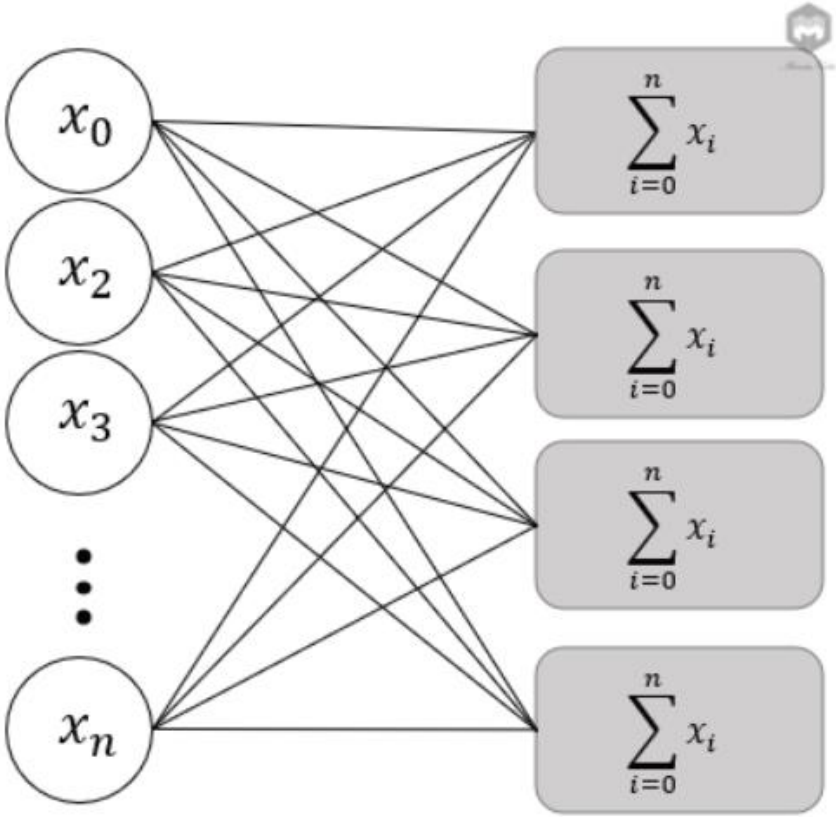
7월

8월

9월

은닉층
Hidden Layer

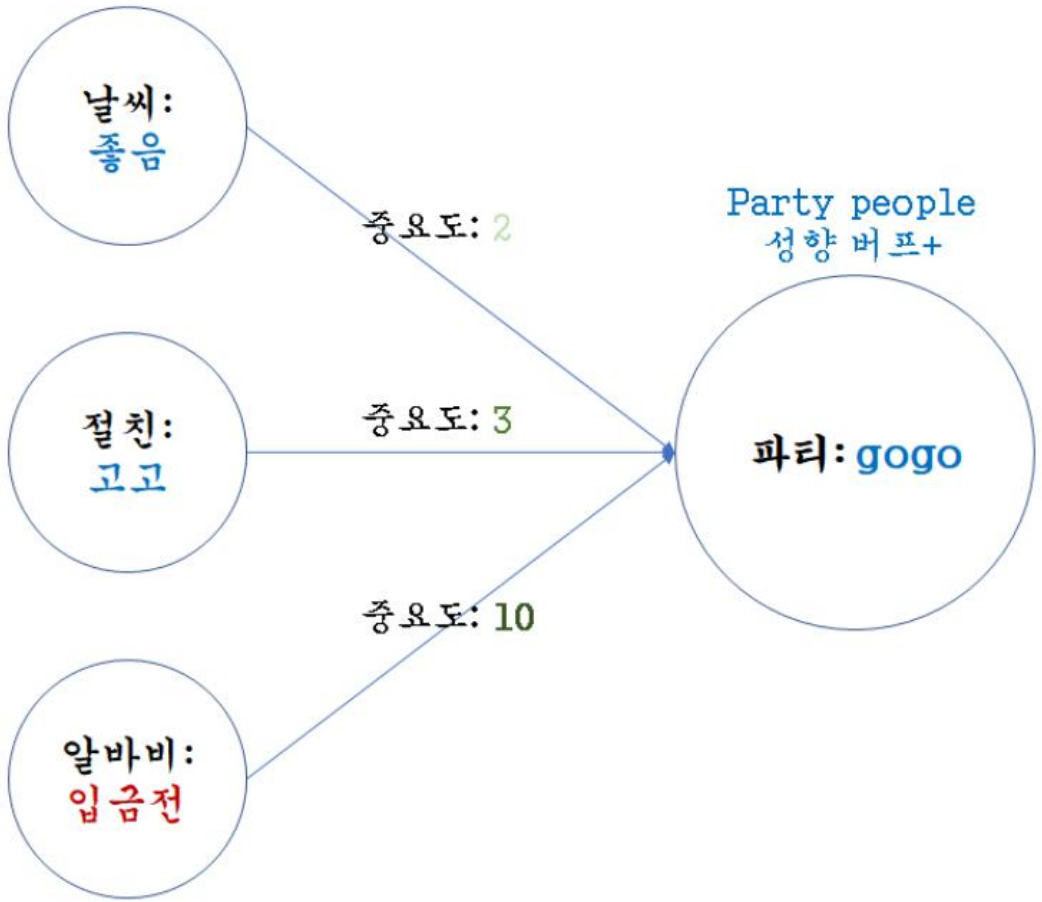
수능 점수



예시 : 가중치가 없을 때 다음 층에서 모두 같은 값이 나옴

출처: https://moonnote.tistory.com/95

편향 Bias



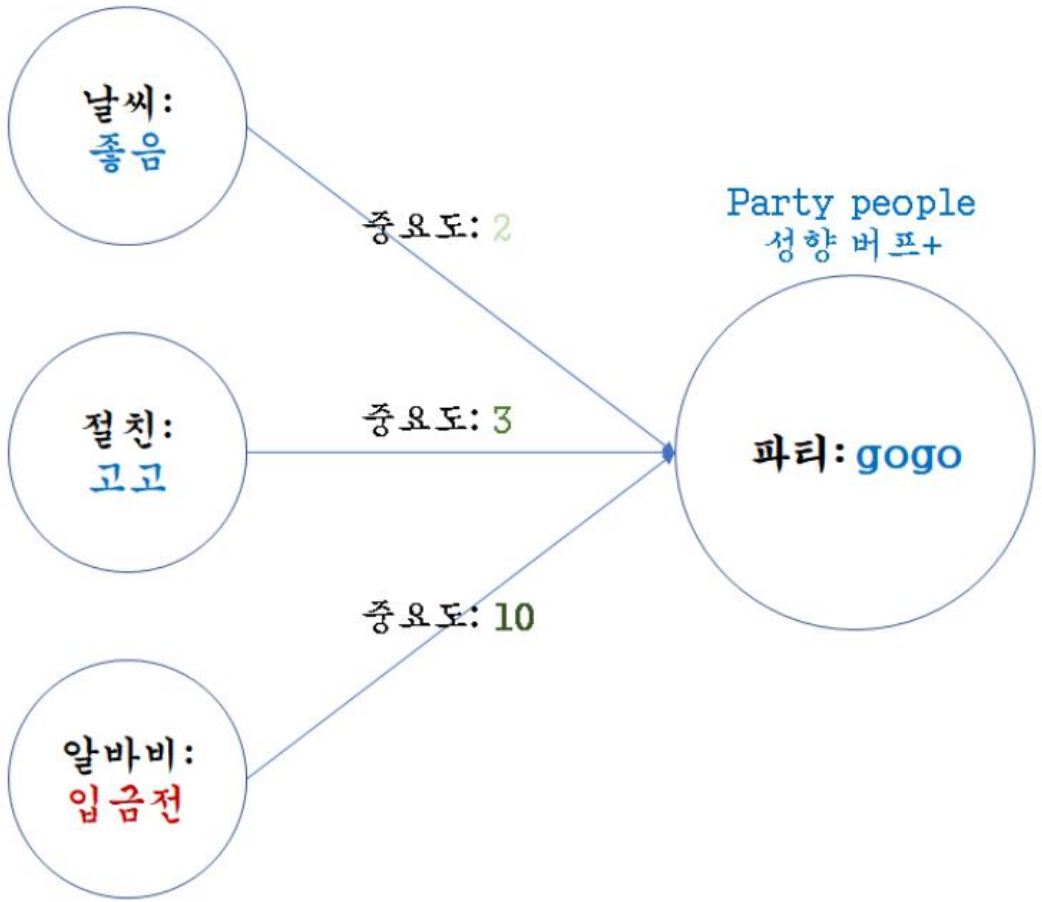
bias 예

출처: https://jh2021.tistory.com/3

편향 은 하나의 뉴런으로 입력된 모든 값을 다 더한 다음에(=가중합) 이 값에 더해주는 상수

이 값은 하나의 뉴런에서 활성화 함수를 거쳐 최종적으로 출력되는 값을 조절하는 역할

편향 Bias



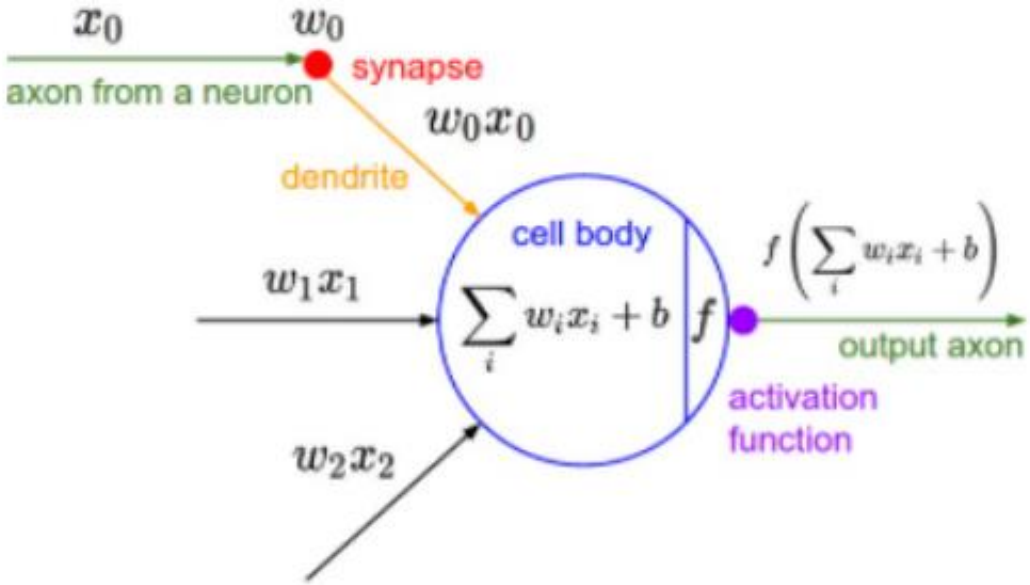
bias 예

출처: https://jh2021.tistory.com/3

편향 은 하나의 뉴런으로 입력된 모든 값을 다 더한 다음에(=가중합) 이 값에 더해주는 상수

이 값은 하나의 뉴런에서 활성화 함수를 거쳐 최종적으로 출력되는 값을 조절하는 역할

활성화 함수 Activation Function



출처: https://happy-obok.tistory.com/55

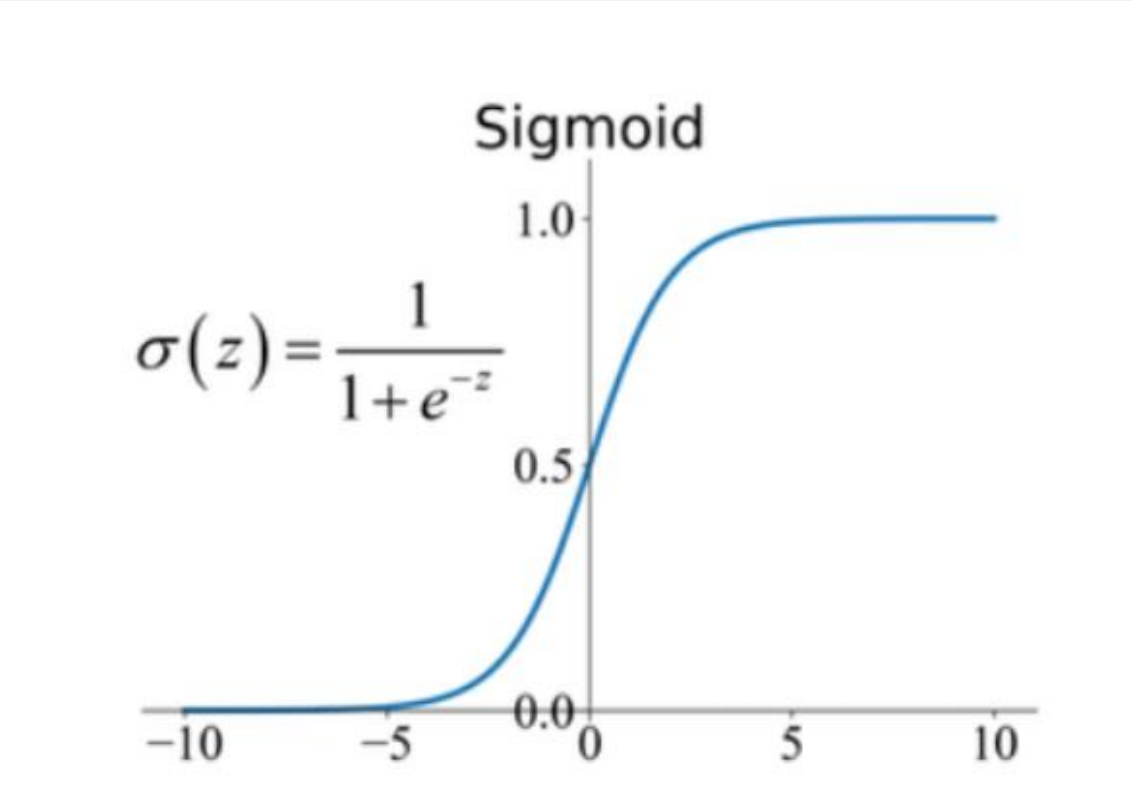
인간의 뇌는 하나의 뉴런에서 다른 뉴런으로 신호를 전달할 때 어떤 임계점을 경계로 출력값에 큰 변화가 생긴다.

인공신경망은 뇌의 구조를 모방하므로 임계점을 설정하고 출력값에 변화를 주는 함수를 이용한다.

앞서 설명한 편향은 임계점을 얼마나 쉽게 넘을지 말지를 조절해주는 값이다.

왼쪽 그림은 가중치(w)가 달린 입력 신호(x)와 편향(b)의 총합을 계산하고 함수 f에 넣어 출력하는 흐름을 보여준다.

시그모이드 Sigmoid 함수



출처: https://happy-obok.tistory.com/55

시그모이드 (sigmoid)란 'S자 모양'이라는 뜻
식에서 e 마이너스 x제곱에서 e는 자연 상수로 2.7182 … 값을 갖는 실수
실수 값을 입력 받아 0~1 사이의 값으로 압축
큰 음수 값일수록 0에 가까워지고 큰 양수 값일수록 1에 가까워진다.
딥러닝 이전에 많이 쓰였고, 미분계수가 자기 자신으로 표현될 수 있는 비선형 구조의 결과를 도출해준다는 장점이 있다.

그러나 기울기 소멸 문제 (Vanishing Gradient Problem)로 인하여 신경망의 학습 능력이 제한되는 포화(Saturation)가 발생

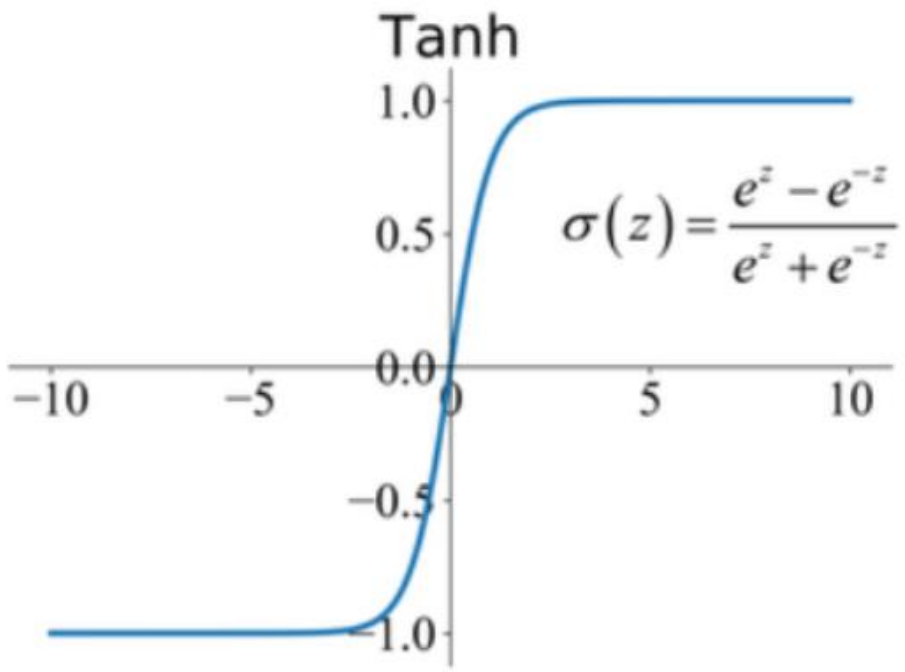
< Python Code>

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

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

#그래프 그려보기
x = np.arange(-5.0, 5.0, 0.1)
y = sigmoid(x)
plt.plot(x, y)
plt.ylim(-0.1, 1.1)
plt.show()
```

Tanh 함수



출처[1]

시그모이드와 비슷하지만 실수값을 입력 받아 -1에서 1 사이의 값으로 압축
시그모이드를 두배 해주고 -1한 값과 같다

$$\tanh(x) = 2\sigma(2x) - 1$$

시그모이드보다 좋은 최적화
하지만 여전히 기울기 소실 문제를 가지고 있음

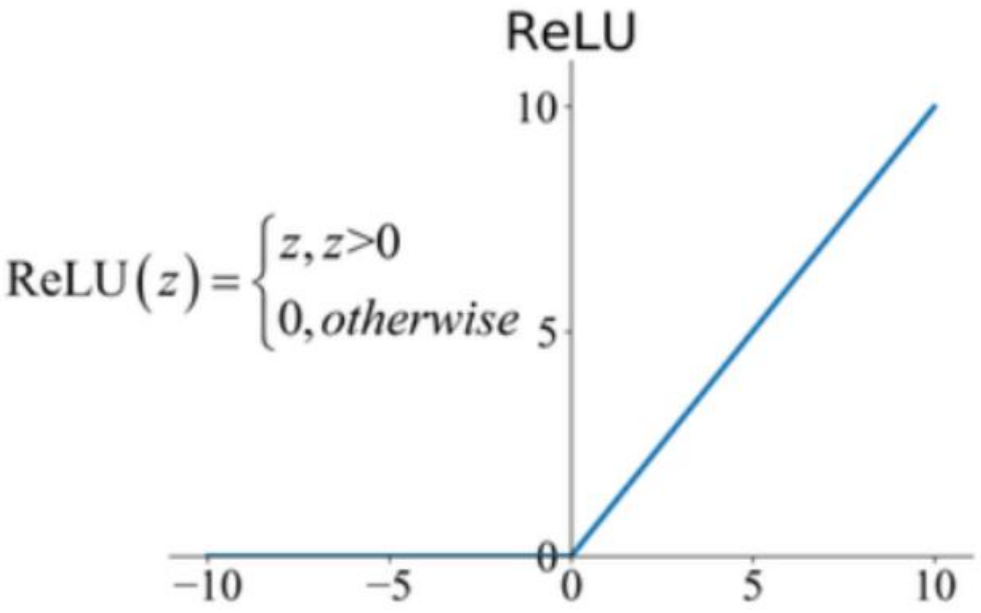
< Python Code>

```python
def tanh(x):
    return (np.exp(x)-np.exp(-x))/(np.exp(x)+np.exp(-x))

#그래프 그려보기
x = np.arange(-10.0, 10.0, 0.1)
y = tanh(x)
plt.plot(x, y)
plt.ylim(-1.1, 1.1)
plt.show()
```

출처: https://happy-obok.tistory.com/55

ReLU(Rectified Linear Unit) 함수



출처[2]

출처: https://happy-obok.tistory.com/55

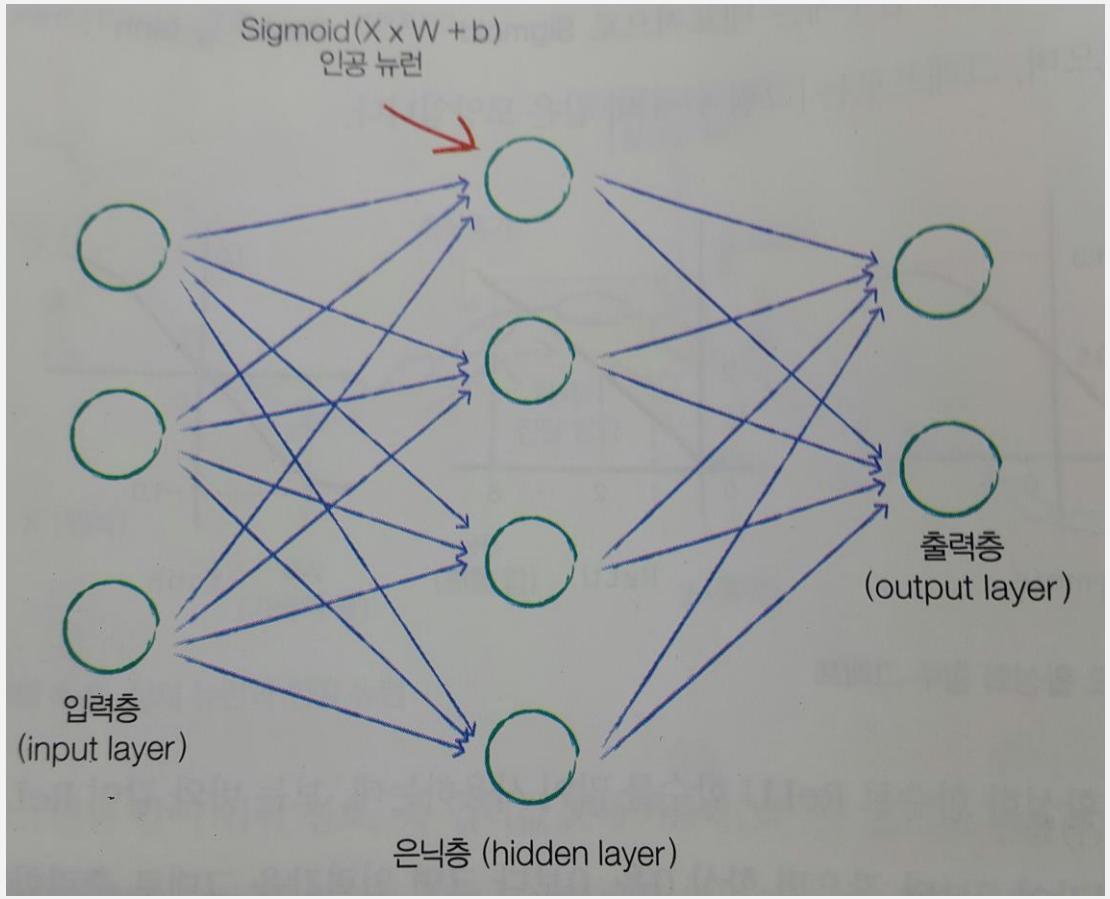
최근 가장 많이 사용하는 활성화 함수
0보다 큰 경우 기울기를 유지해주기 때문에 Gradient Vanishing 문제를 해결해 줌
함수가 매우 간단해서 계산이 간단하기에 학습속도가 매우 빠르다는 장점이 있음
입력값이 0보다 작으면 항상 0을, 0보다 크면 입력값을 그대로 출력

< Python Code>

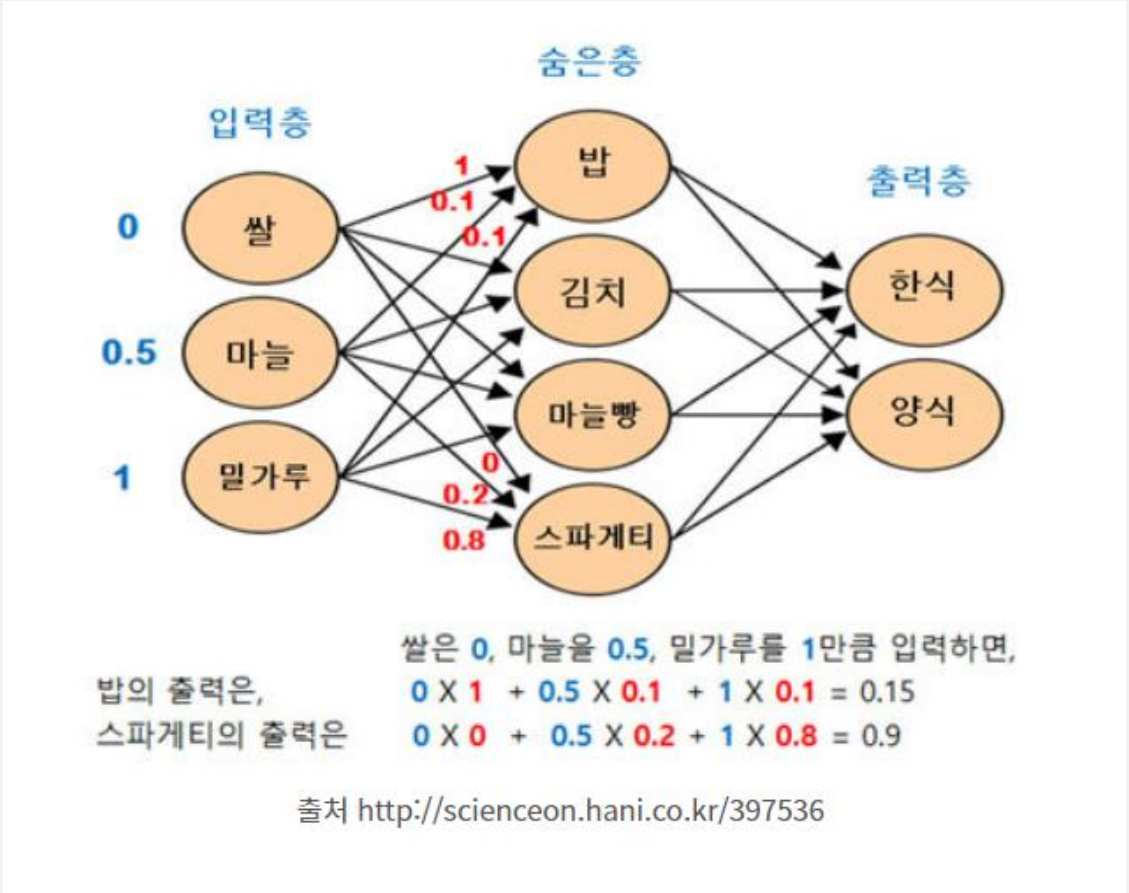
```
def relu(x):
    return np.maximum(0,x)

#그래프 그려보기
x = np.arange(-10, 10)
y = relu(x)
plt.plot(x, y)
plt.ylim(-1, 10.1)
plt.show()
```

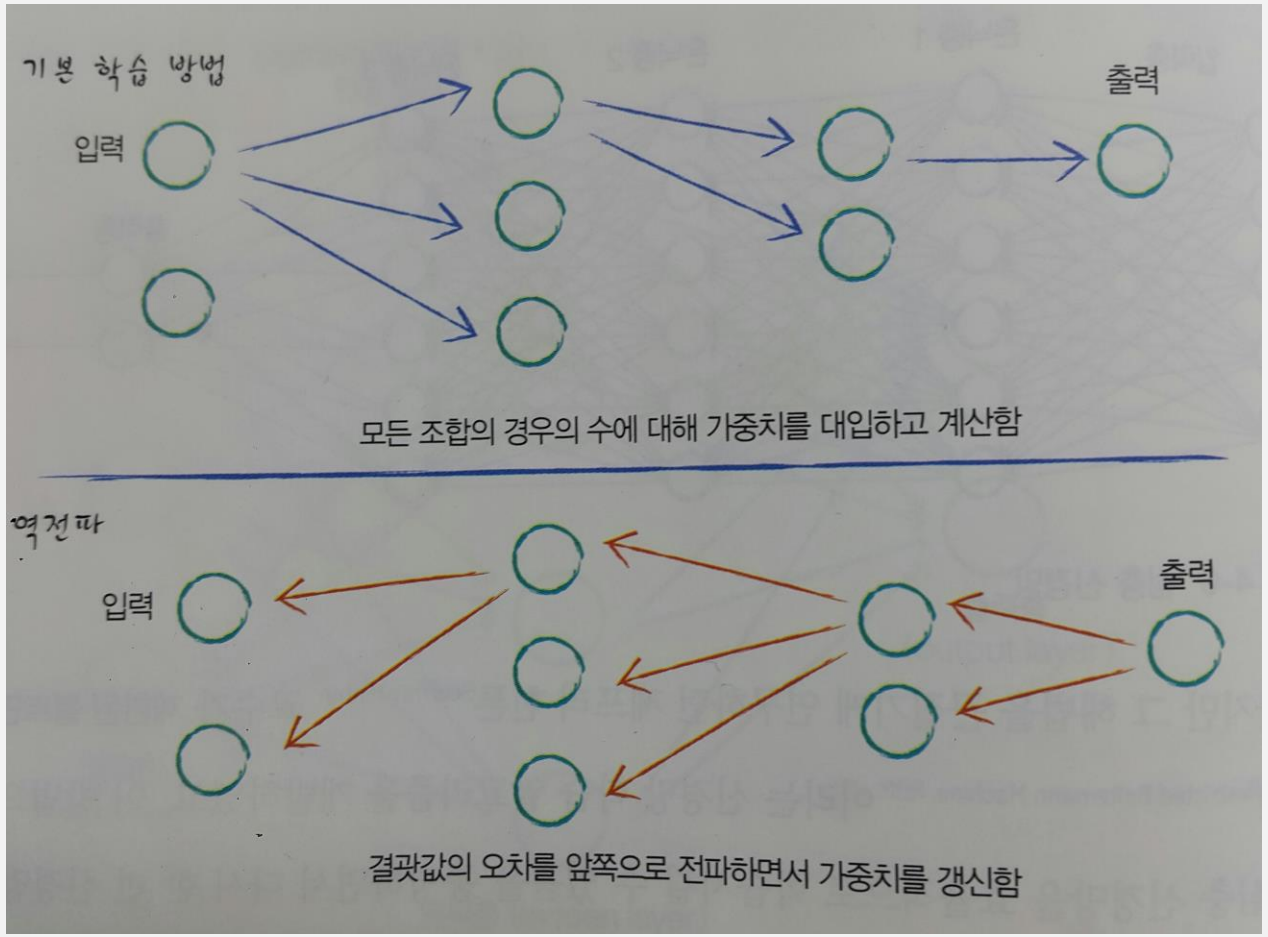
간단한 도식화



출처: https://wikidocs.net/25169



역전파 Backpropagation



출처: https://wikidocs.net/25169

출현 배경

수천 ~ 수만 개의 W와 b 값의 조합을 일일이 변경하면서 계산하려면 매우 오랜 시간이 걸리기에 신경망을 제대로 훈련시키기가 어려웠다.
특히 신경망의 층이 깊어질수록 시도해봐야 하는 조합의 경우의 수가 기하급수적으로 늘어나기 때문에 과거에는 유의미한 신경망을 만들기란 거의 불가능에 가까웠다.

역전파는 출력층이 내놓은 결과의 오차를 신경망을 따라 입력층까지 역으로 전파하며 계산해 나가는 방식이다.
이 방식은 입력층부터 가중치를 조절해가는 기존 방식보다 훨씬 유의미한 방식으로 가중치를 조절해줘서 최적화 과정이 훨씬 빠르고 정확해진다.

신경망을 구현하려면 거의 항상 적용해야 하는 알고리즘이지만, 구현하기가 제법 까다롭다. 하지만 텐서플로우 라이브러리는 활성화 함수와 학습 함수 대부분에 역전파 기법을 기본으로 제공해 주기 때문에, 이를 활용하면 다양한 알고리즘을 직접 구현하지 않고도 매우 쉽게 신경망을 만들고 학습시킬 수 있다.

Fiery_Data Analysis

이 정재
2023.07.09 (일)

감사합니다