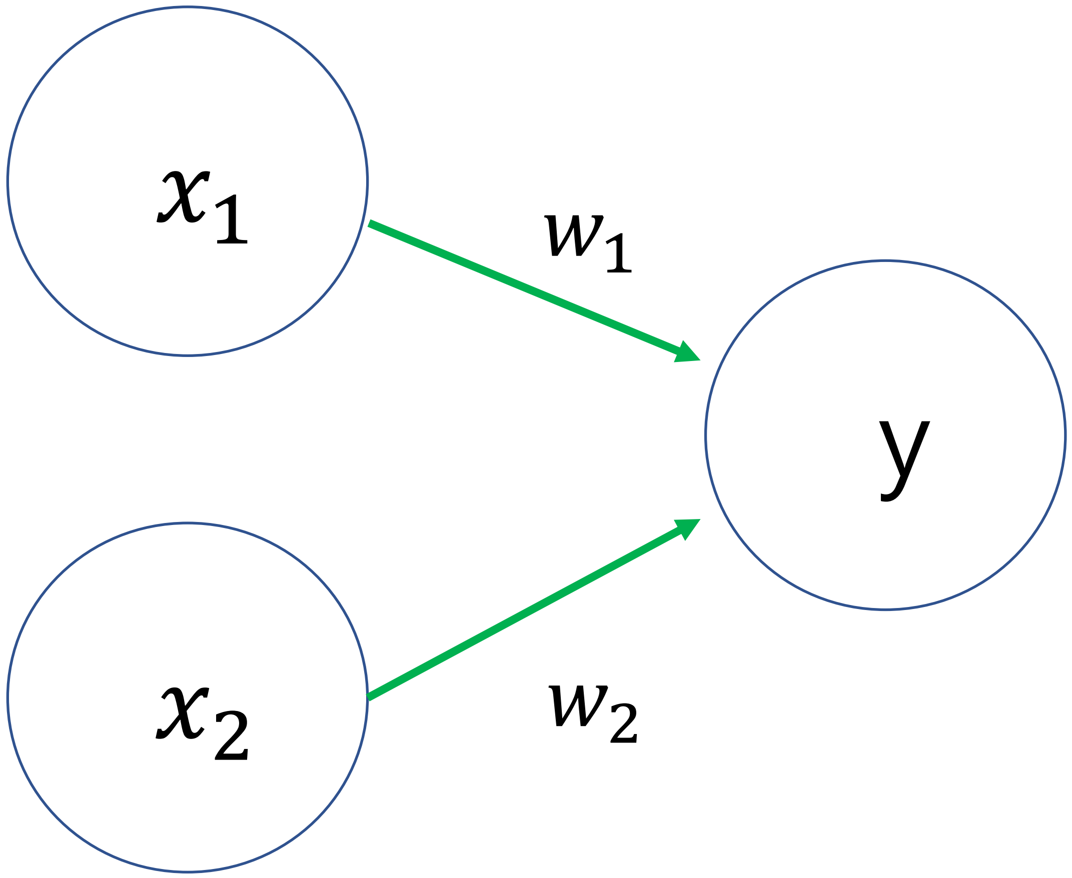
딥러닝(Deep Learning)

기본 알고리즘 알아보기

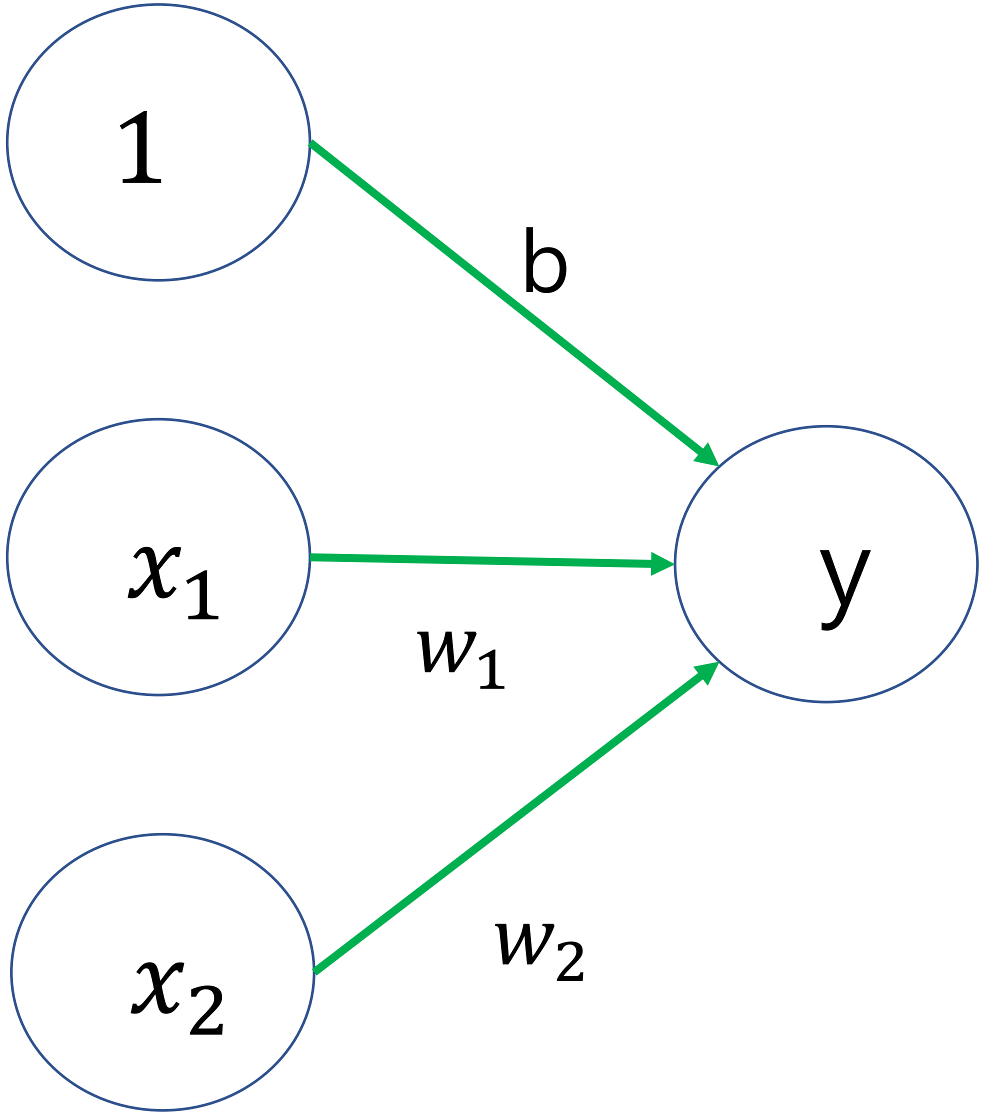
퍼셉트론 알아보기

- 가장 간단한 인공 신경망 구조의 하나.
- 1957년 프랑크 로젠블라트가 제안.
- 입력과 출력이 어떤 숫자이고 각각의 입력 연결은 가중치와 연관되어 있음.
- 입력과 가중치 합을 계산하고, 이어 계산된 합에 계단 함수(step function)을 적용

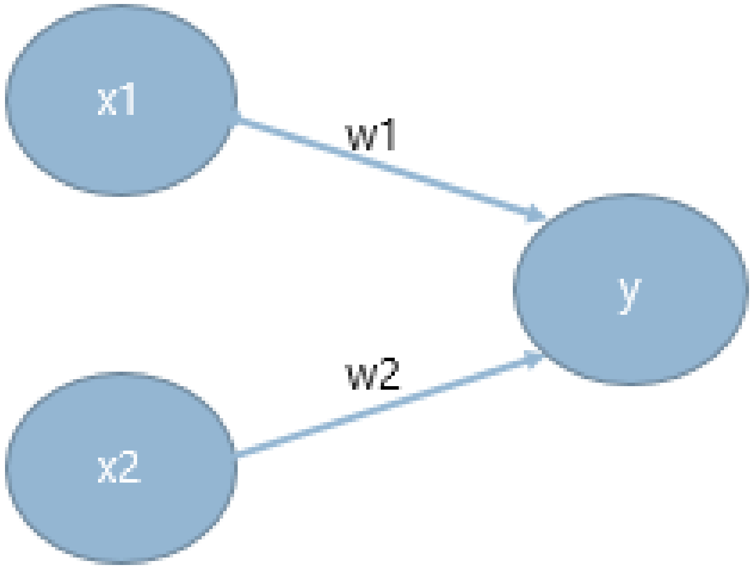
퍼셉트론 기본 구조



퍼셉트론 기본 구조



퍼셉트론 동작원리



$$y = \begin{cases} 0 & (w1*x1 + w2*x2 - \theta \leq 0) \\ 1 & (w1*x1 + w2*x2 - \theta > 0) \end{cases}$$

참고

θ(theta) : 임계값

퍼셉트론과 신경망

▶ 다층 퍼셉트론은 퍼셉트론이 여러 층으로 구성된 신경망을 가르킨다.

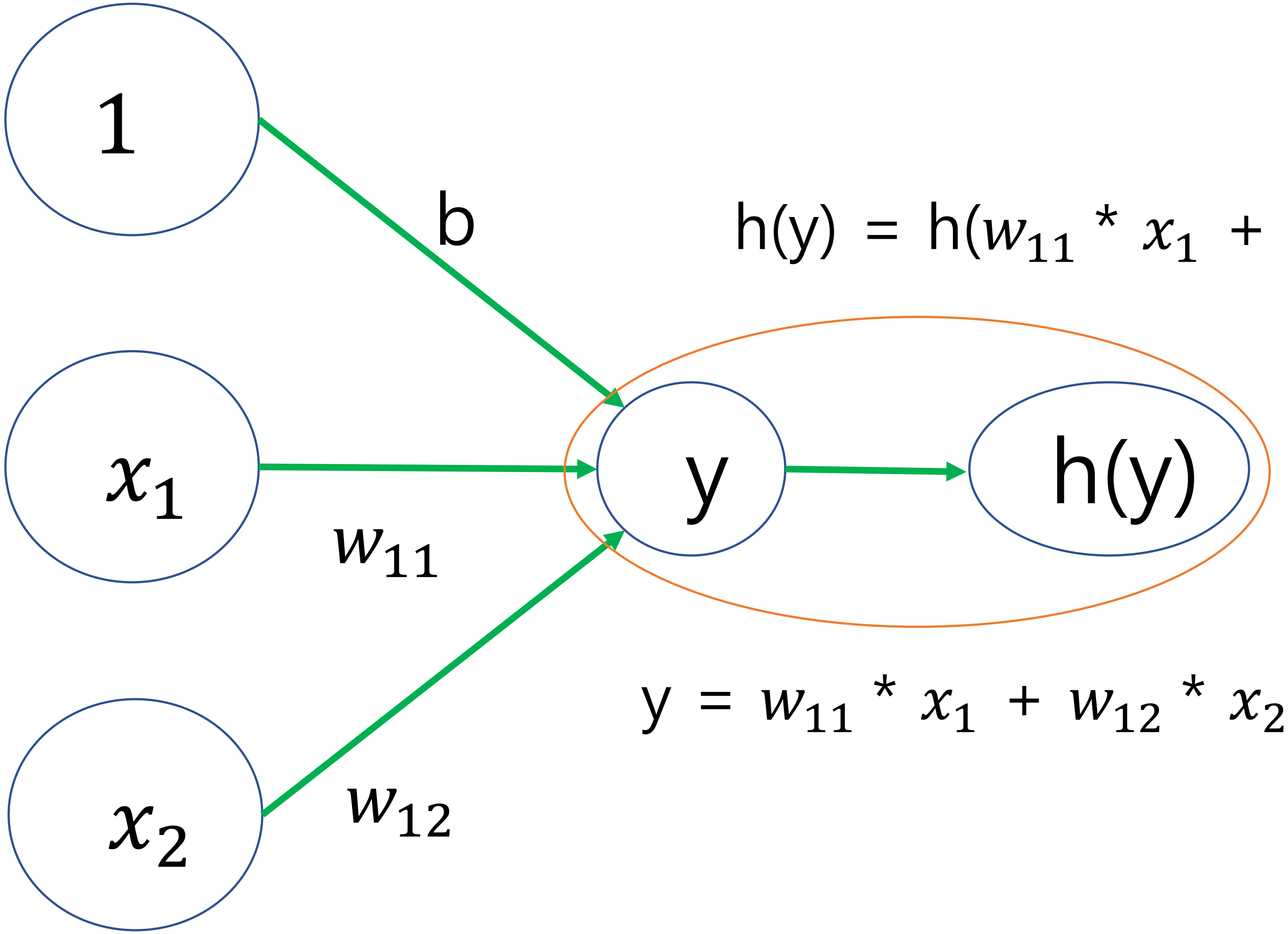
▶ 다층 퍼셉트론은 (통과) 입력층 하나와 은닉층(hidden layer)라고 불리는 하나이 이상의 TLU층과 마지막 층인 출력층(output layer)로 구성.

▶ 인공 신경망의 은닉층이 2개 이상일 때, 이를 심층 신경망(deep neural network)-DNN이라 한다.

인공 신경망이란?

- 딥러닝의 핵심이며 아주 복잡한 문제를 다루는 데 적합.
- 수백만 개의 이미지를 분류가 가능하다.
- 음성 인식 서비스의 성능을 높일 수 있다. (애플의 시리 등)
- 매일 수억 명의 사람들에게 가장 좋은 비디오를 추천해 준다. (예: 유튜브)
- 수백만 개의 기보를 익히고 자기 자신과 게임. (예 : 딥마인드 알파고)

활성화 함수의 등장



$$h(y) = h(w_{11} * x_1 + w_{12} * x_2 + b)$$

h( ) : 활성화 함수

$$y = w_{11} * x_1 + w_{12} * x_2 + b$$

활성화 함수의 기본 이해

$$h(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

활성화 함수 – sigmoid(시그모이드 함수)

$$h(x) = \frac{\mathbf{1}}{\mathbf{1}+\boldsymbol{exp}(-\boldsymbol{x})}$$

exp(-x)는 $\boldsymbol{e}^{-\boldsymbol{x}}$를 뜻한다.

e는 자연상수로 2.7182의 값을 갖는 실수이다.

h(1.0) = 0.731, h(2.0)=0.880

활성화 함수- 시그모이드 함수

$$h(x) = \frac{1}{1+exp(-x)}$$

exp(−x)는 e^{-x}를 뜻한다.

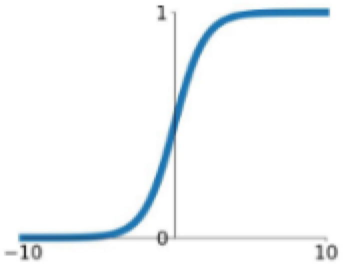
e는 자연상수로 2.7182의 값을 갖는 실수이다.

h(1.0) = 0.731, h(2.0)=0.880

활성화 함수의 종류

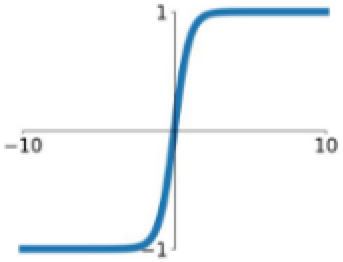
Sigmoid

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



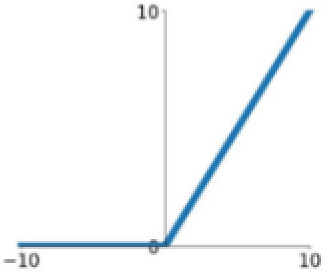
tanh

$\tanh(x)$



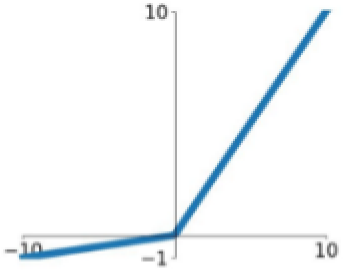
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

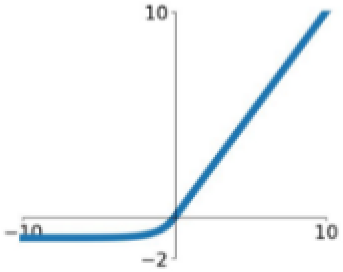


Maxout

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

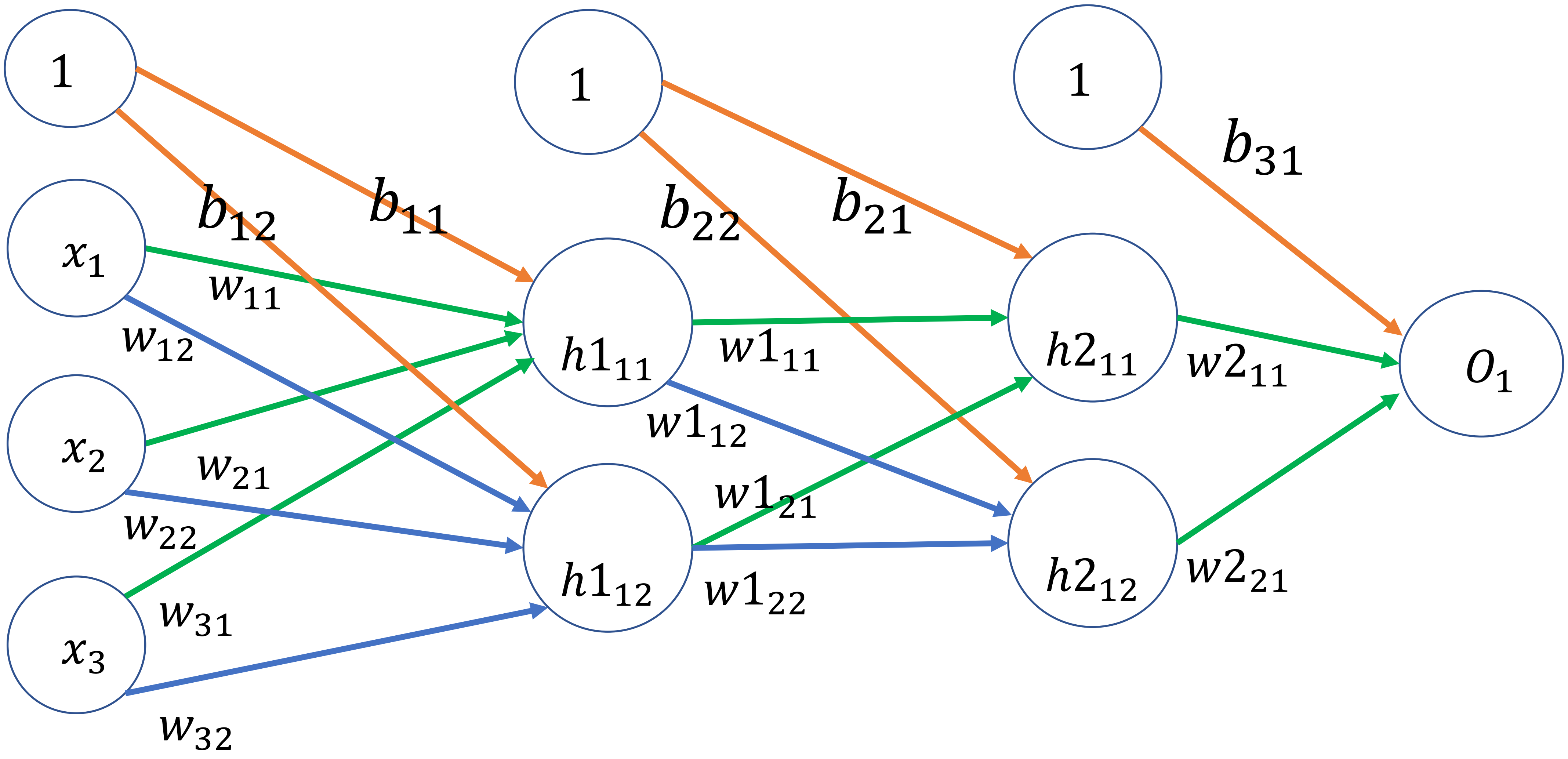
ELU

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



https://towardsdatascience.com/complete-guide-of-activation-functions-34076e95d044 참조

다층 신경망

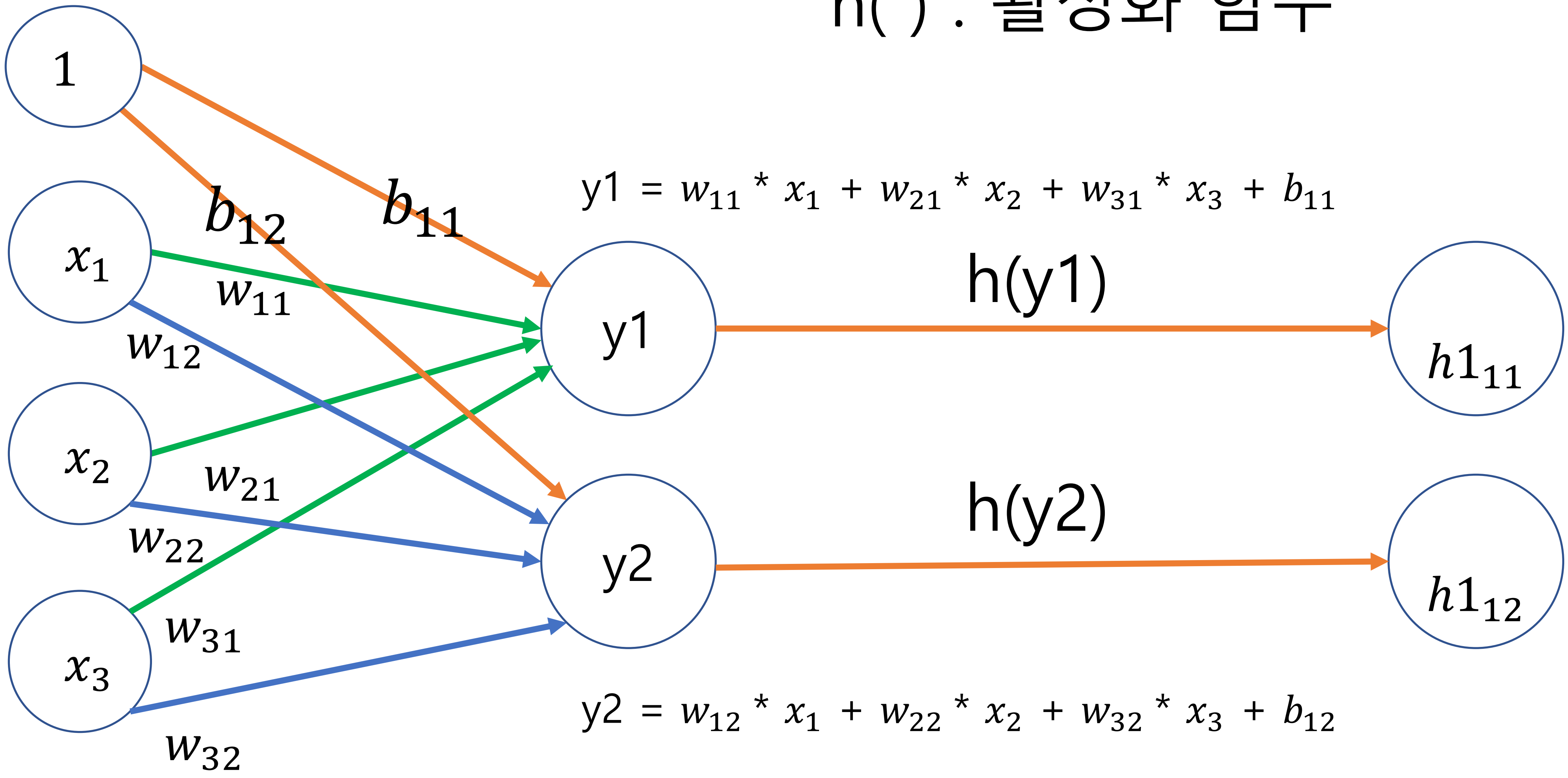


1
1
1
b_{12}
b_{11}
b_{22}
b_{21}
b_{31}
x_1
x_2
x_3
w_{11}
w_{12}
w_{21}
w_{22}
w_{31}
w_{32}
$h1_{11}$
$h1_{12}$
$w1_{11}$
$w1_{12}$
$w1_{21}$
$w1_{22}$
$h2_{11}$
$h2_{12}$
$w2_{11}$
$w2_{21}$
o_1

다층 신경망 일부 연산 확인

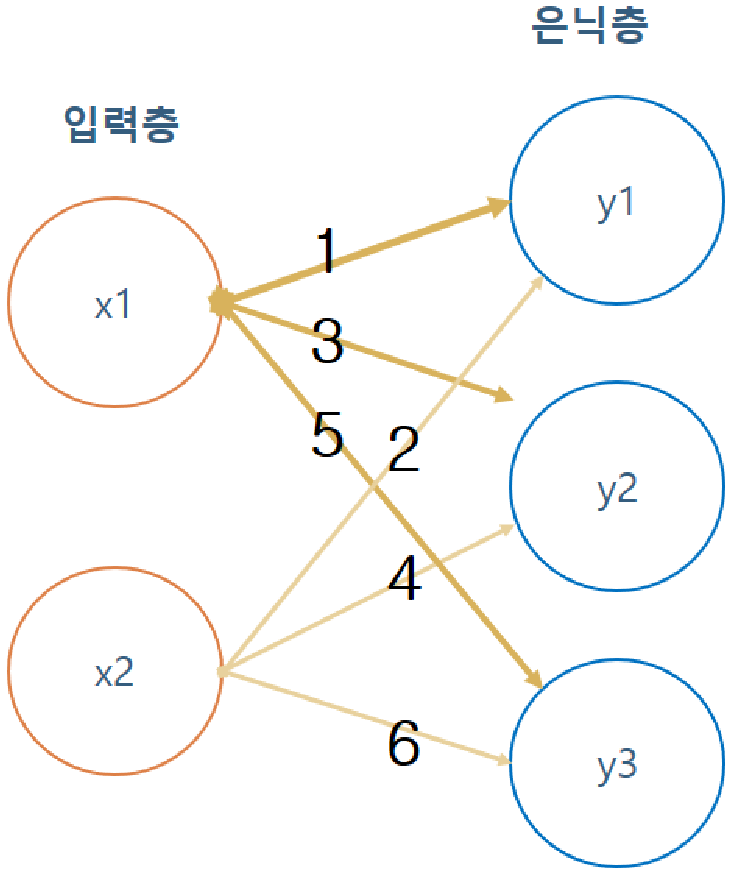
h( ) : 활성화 함수

y1 = $w_{11} * x_1 + w_{21} * x_2 + w_{31} * x_3 + b_{11}$



y2 = $w_{12} * x_1 + w_{22} * x_2 + w_{32} * x_3 + b_{12}$

신경망을 행렬로 표시(1)



$$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$$

X	W	=	Y
1 X 2	2 X 3		1 X 3

신경망을 행렬로 표시(2)

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{bmatrix} \text{X} \begin{bmatrix} ? \end{bmatrix} = \begin{bmatrix} x_{11}w_{11}+x_{12}w_{21}+x_{13}w_{31} & x_{11}w_{12}+x_{12}w_{22}+x_{13}w_{32} \\ x_{21}w_{11}+x_{22}w_{21}+x_{23}w_{31} & x_{21}w_{12}+x_{22}w_{22}+x_{23}w_{32} \\ x_{31}w_{11}+x_{32}w_{21}+x_{33}w_{31} & x_{31}w_{12}+ x_{32}w_{22}+x_{33}w_{32} \\ x_{41}w_{11}+x_{42}w_{21}+x_{43}w_{31} & x_{41}w_{12}+x_{42}w_{22}+x_{43}w_{32} \\ x_{51}w_{11}+x_{52}w_{21}+x_{53}w_{31} & x_{51}w_{12}+x_{52}w_{22}+x_{53}w_{32} \end{bmatrix}$$

[n,3] **[?, ?]** **[N,2]**

★ 첫번째 행렬의 열, 두번째 행의 행이 같다.

★ 첫번째 행렬의 행, 두번째 행의 열이 결과 행렬의 행렬이 된다.

행렬의 내적(예제)

```
A = np.array([[1,2],[3,4]])
A.shape
```

```
(2, 2)
```

```
B = np.array([[5,6],[7,8]])
B.shape
```

```
(2, 2)
```

```
np.dot(A,B)
```

```
array([[19, 22],
       [43, 50]])
```

행렬의 곱에서 유의하기

▶ 행렬의 곱에서는 대응하는 차원의 원소 수를 일치 시켜야 함.

A	B	=	C
3 X $\boxed{2}$	2 X $\boxed{4}$		3 X 4

다차원 배열 연산 예제

```
import numpy as np
A = np.array([1,2,3,4])
print(A)
```

```
[1 2 3 4]
```

```
np.ndim(A)
```

배열의 차원수 확인

```
1
```

```
A.shape
```

배열의 형상

```
(4,)
```

```
A.shape[0]
```

```
4
```

```
import numpy as np
B = np.array([[1,2],[3,4],[5,6]])
print(B)
```

```
[[1 2]
 [3 4]
 [5 6]]
```

```
np.ndim(B)
```

배열의 차원수 확인

```
2
```

```
B.shape
```

배열의 형상

```
(3, 2)
```

Cost function, loss function

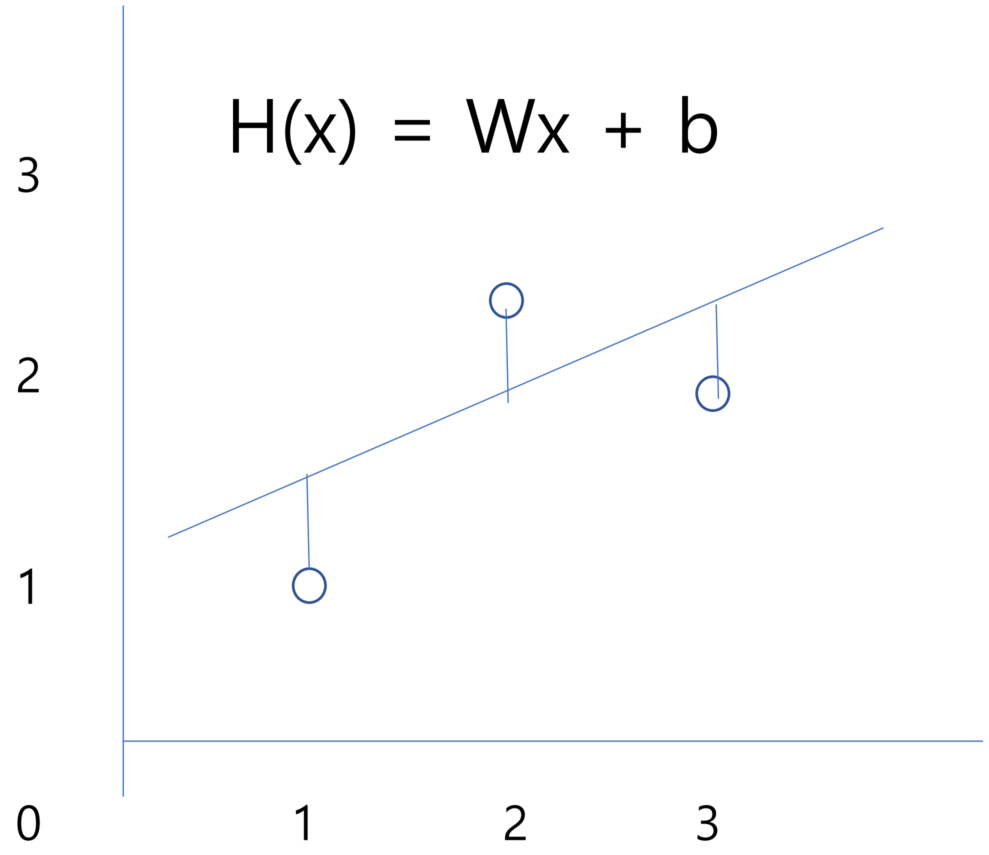
▶ 비용 함수(cost function)는 전체 훈련 세트(또는 미니 배치 경사 하강법의 경우, 미니 배치)에 대한 것.

▶ 손실 함수(loss function)는 학습용 샘플의 하나의 예제에 대한 것.

Cost Function에 대해 알아보기

▶ 비용함수는 주어진 훈련 샘플과 예상 출력과 관련하여 신경망이 얼마나 좋은가를 측정한 것.

$$\text{Cost} = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$



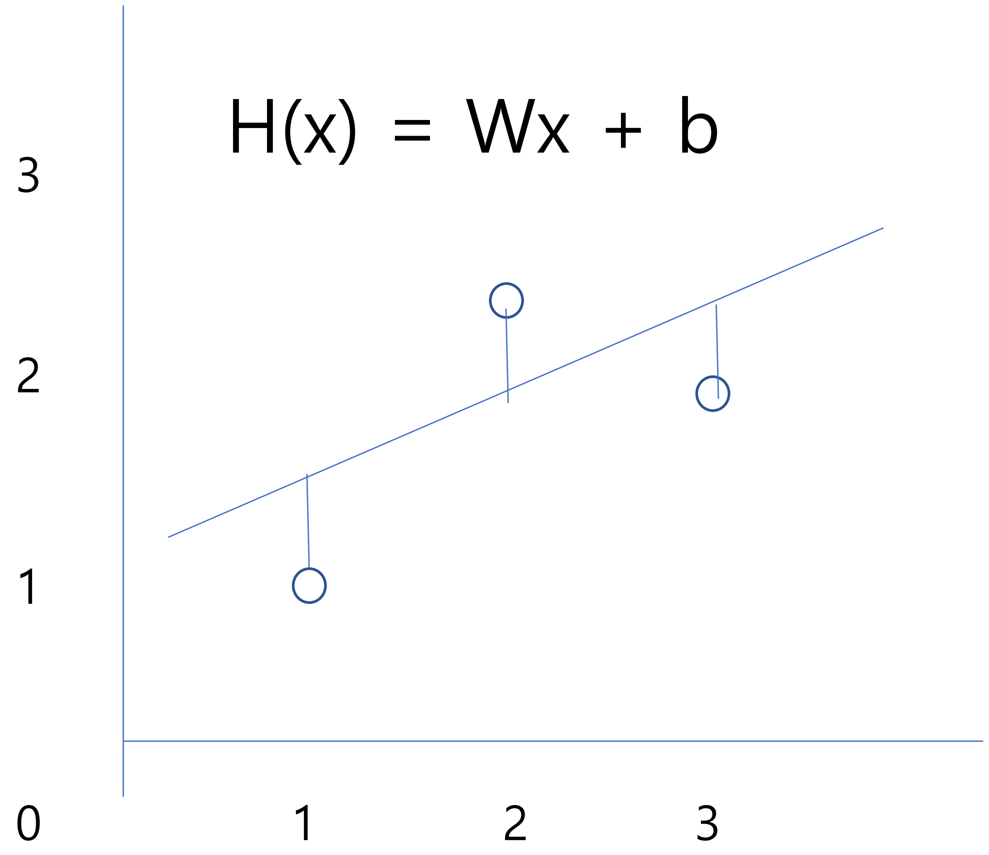
예측 모델과 cost function

▶ 모델(Hypothesis)과 cost Function

$$\text{Cost} = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$$H(x) = Wx + \cancel{b}$$

$$\text{Cost}(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$



모델의 목표는 비용을 최소화

$$\text{cost}(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Minimize cost(W,b)

▶ 만약 모델이 비용함수를 MSE로 쓴다고 가정.

▶ W와 b를 통해 cost의 값을 최소화하는 알고리즘 구하기

W값에 따른 cost 값 구하기

$H(x) = Wx$ -> $H(x) = 1*x$

W가 1일때,

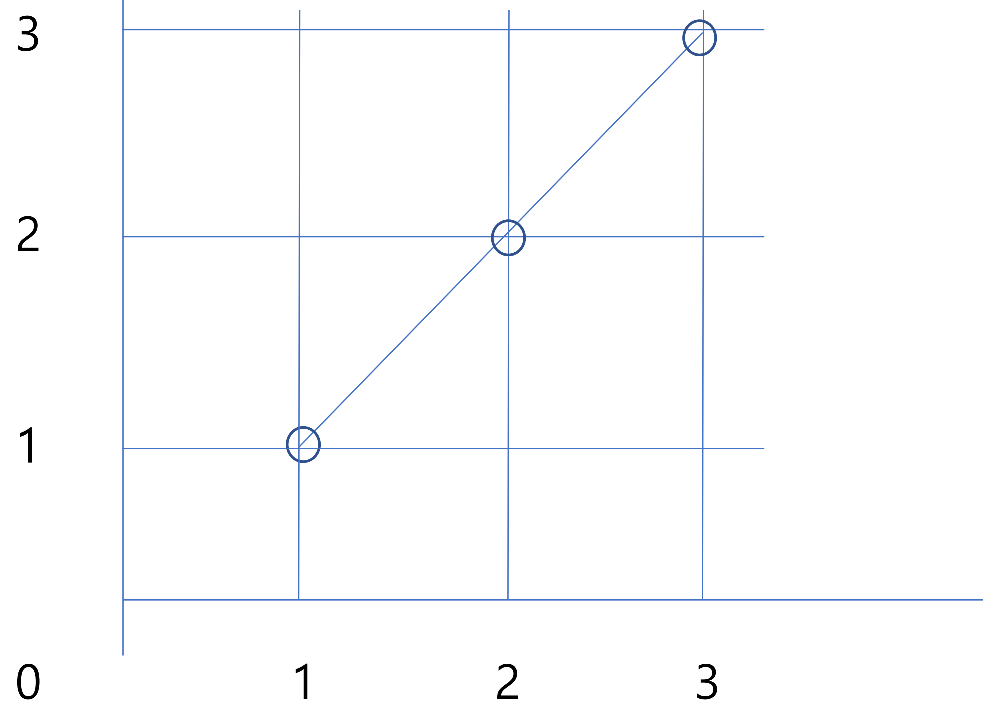
$H(x) = Wx + b$

$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$$Cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$

X	Y
1	1
2	2
3	3

$$\frac{(1*1-1)^2 + (1*2-2)^2 + (1*3-3)^2}{3}$$



W값에 따른 cost 값 구하기

H(x) = Wx -> H(x)=0*x

W가 0일때,

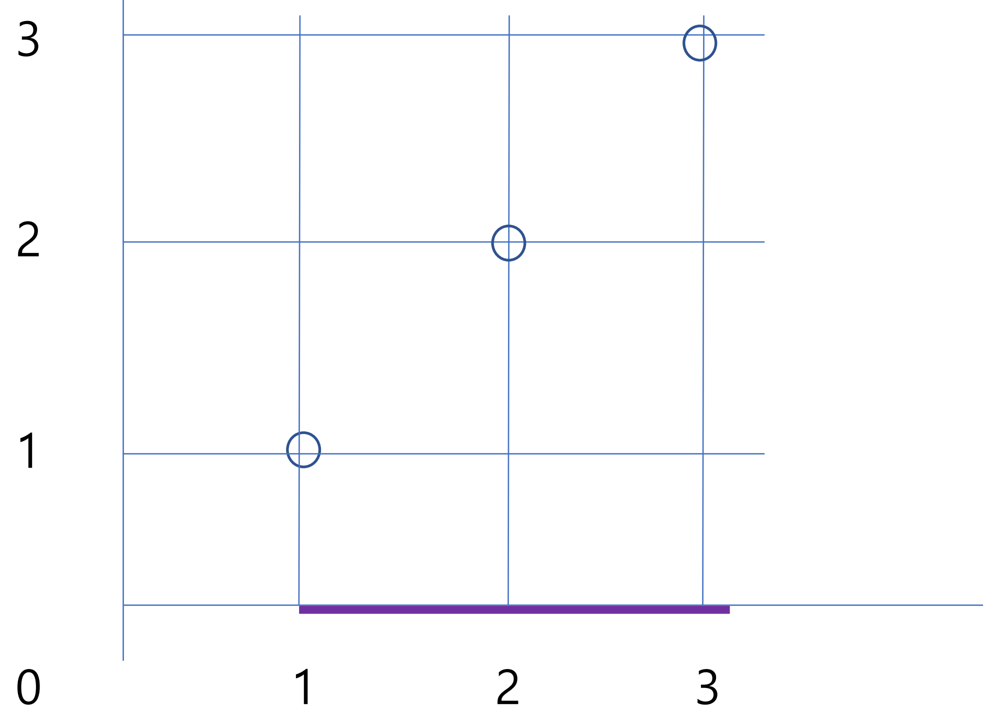
H(x) = Wx + b

$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

X	Y
1	1
2	2
3	3

$$Cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$

$$\frac{(0*1-1)^2+(0*2-2)^2+(0*3-3)^2}{3} = 4.67$$



W값에 따른 cost 값 구하기

H(x) = Wx -> H(x)=2*x

W가 2일때,

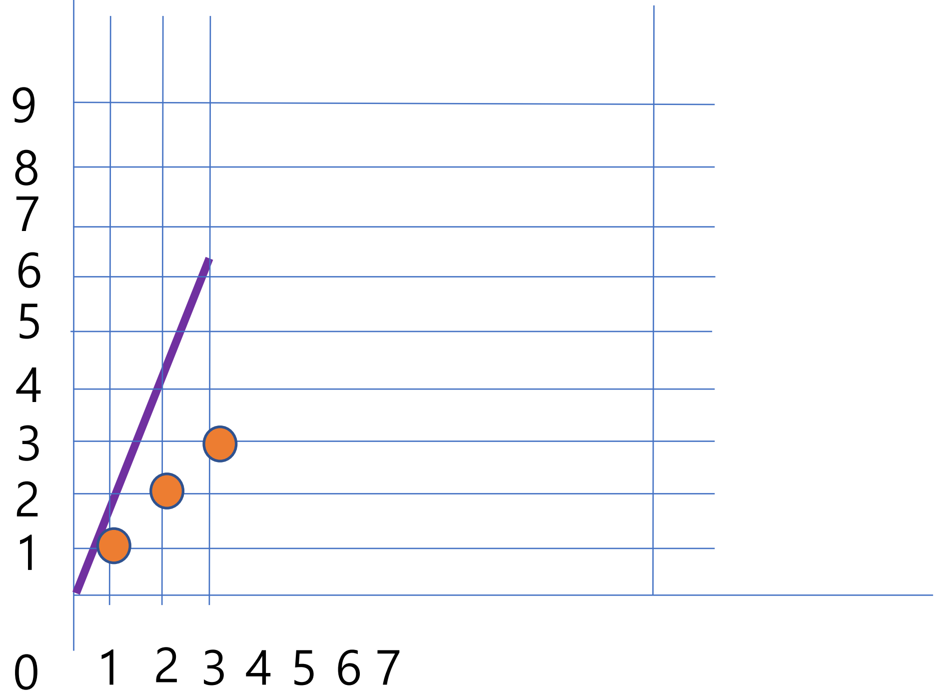
H(x) = Wx + b

$$\text{cost}(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

X	Y
1	1
2	2
3	3

$$\text{Cost}(W,b) = \frac{1}{m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$

$$\frac{(2*1-1)^2+(2*2-2)^2+(2*3-3)^2}{3}=4.67$$



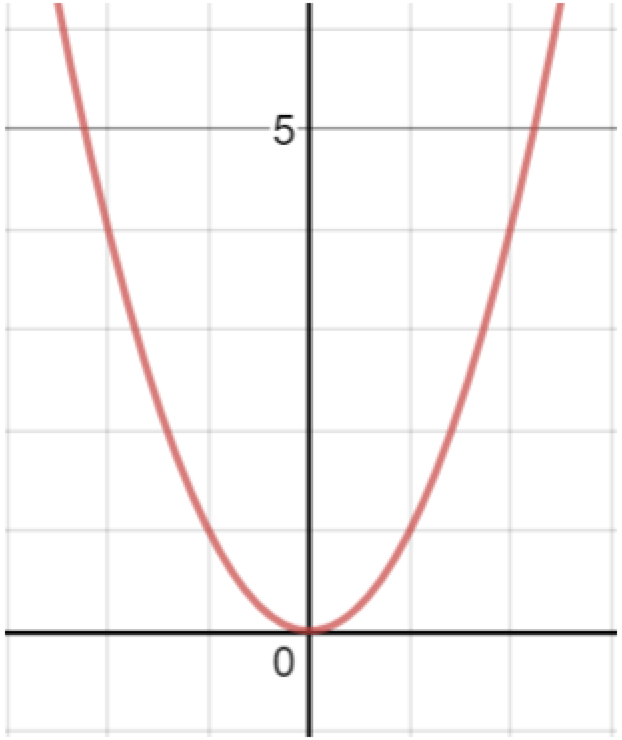
W값에 따른 cost 값 구하기

$H(x) = Wx + b$

W가 0일때, cost(W)=4.67

W가 1일때 , cost(W)=0

W가 2일때, cost(W)=4.67



Loss function – binary crossentropy/log loss

- 이 함수는 분류 과제에 사용된다.

(예) 예를 들어 개와 고양이 분류에 사용된다.

$$\text{Loss} = -\frac{1}{m}\sum_{i=1}^{m} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

Loss function – categorical crossentropy

▶ 이 함수는 분류 과제에 사용된다.

(예) 예를 들어 MNIST의 손 글씨 분류에 사용된다.

$$\text{Loss} = -\sum_{i=1}^{m} y_i \cdot \log \hat{y}_i$$

경사 하강 알고리즘

▶ 어떻게 비용 함수가 최소가 되는 W 파라미터를 구할 수 있을까?

경사를 따라 내려가는 알고리즘 중의 하나이다.

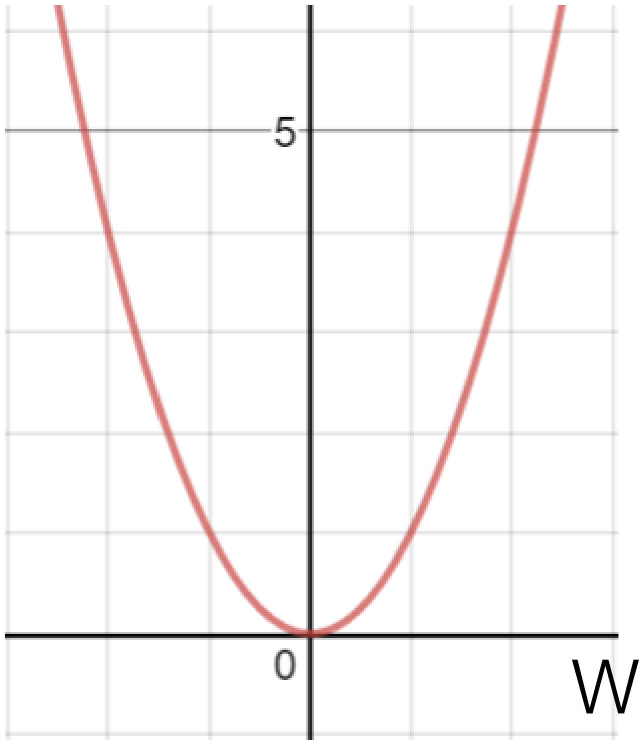
가. Cost function을 최소화시키기

나. Gradient descent 는 여러 최소화 문제를 풀기 위해 사용된다.

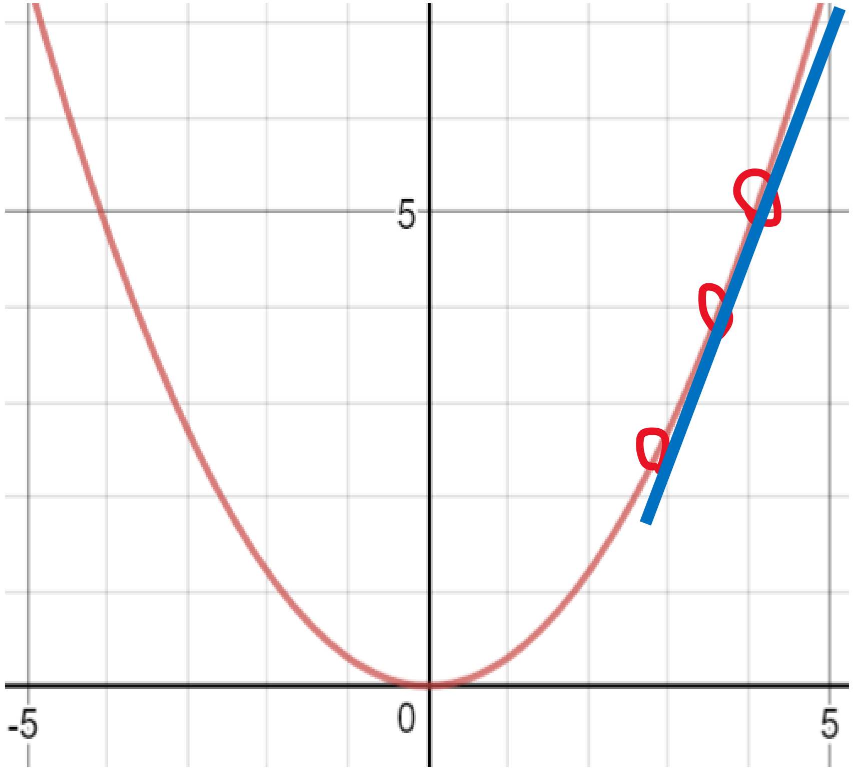
다. Cost function이 주어지고, 이 알고리즘은 cost를 최소화시키는 W와 b를 찾을 것이다.

라. W1, w2, w3…

Cost function



경사 하강 알고리즘(Gradient descent algorithm)



▶ 점진적으로 반복적인 계산을 통해 W파라미터 값을 업데이트

▶ 오류 값이 최소가 되도록 W 파라미터를 구하는 방식

▶ 예측값과 실제 값의 차이가 작아지는 방향성을 가지고 W파라미터를 보정해 간다.

▶ 오류 값이 더 작아지지 않으면 그 오류 값을 최소 비용으로 판단하고 그때의 W값을 최적 파라미터로 반환.

▶ “어떻게 하면 오류가 작아지는 방향으로 W값을 보정할 수 있을까?

▶ 예를 들어, 비용함수가 그림과 같은 포물선 형태의 2차 함수라면 경사 하강법은 최초 w에서부터 미분을 적용한 뒤, 미분값을 감소시키는 방향으로 w값을 업데이트.

비용함수를 이용해서 어떻게 w값을 업데이트 할까?

Cost(W)= $$\frac{1}{m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$

↓

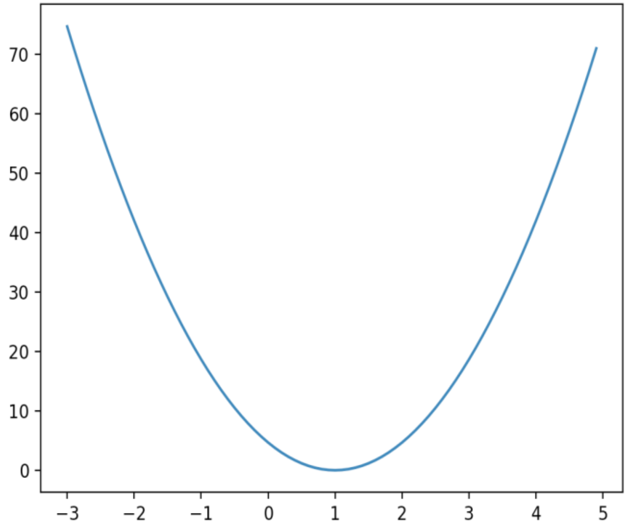
Cost(W)= $$\frac{1}{2m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$

★ 앞의 1/m이나 1/2m 같은 의미를 지닌다.

비용함수를 이용해서 어떻게 w값을 업데이트 할까?

$$\text{Cost(W)}=\frac{1}{2m}\sum_{i=1}^{m}(\left(Wx^{(i)}\right) - y^{(i)})^2$$

Cost(W)



Cost 함수를 미분하면 기울기를 구할 수 있다.

$$W := W - \alpha\frac{\partial}{\partial W}\text{cost(W)}$$

▶ W가 최소지점의 영역의 오른쪽에 있으면 –방향으로
▶ W가 최소지점의 왼쪽에 있으면 + 방향으로 움직이기 위해 –를 붙여준다.

비용함수를 이용해서 어떻게 w값을 업데이트 할까?

Cost(W)= $\frac{1}{2m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$

$$W := W - \alpha \boxed{\frac{\partial}{\partial W}cost(W)}$$

$$W := W - \alpha \boxed{\frac{\partial}{\partial W}} \frac{1}{2m}\sum_{i=1}^{m}\left(\left(Wx^{(i)}\right) - y^{(i)}\right)^2$$

$$W := W - \alpha \frac{1}{2m}\sum_{i=1}^{m}2\left(Wx^{(i)}\right) - y^{(i)})x^{(i)}$$

$$\boxed{W := W - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(Wx^{(i)}\right) - y^{(i)})x^{(i)}}$$

비용함수를 이용해서 어떻게 w값을 업데이트 할까?

데이터 개수

$$W := W - \alpha \frac{1}{m} \sum_{i=1}^{m} (Wx^{(i)}) - y^{(i)})x^{(i)}$$

Learning rate

★ 우리는 위와 같이 Gradient descent 알고리즘이 수식으로 나오고,
우리는 이를 기계적으로 적용만 시키면 이 Cost Function을 최소화하는 W를 구해내고,
이것이 바로 linear regression인 학습과정을 통해서 모델을 만든다고 할 수 있다.

정리해 보기

- **Hypothesis** H(x) = Wx + b

- **Cost function** $\text{Cost}(W,b) = \frac{1}{m}\sum_{i=1}^{m}(\left(Wx^{(i)}\right) - y^{(i)})^2$

- **Gradient descent algorithm**