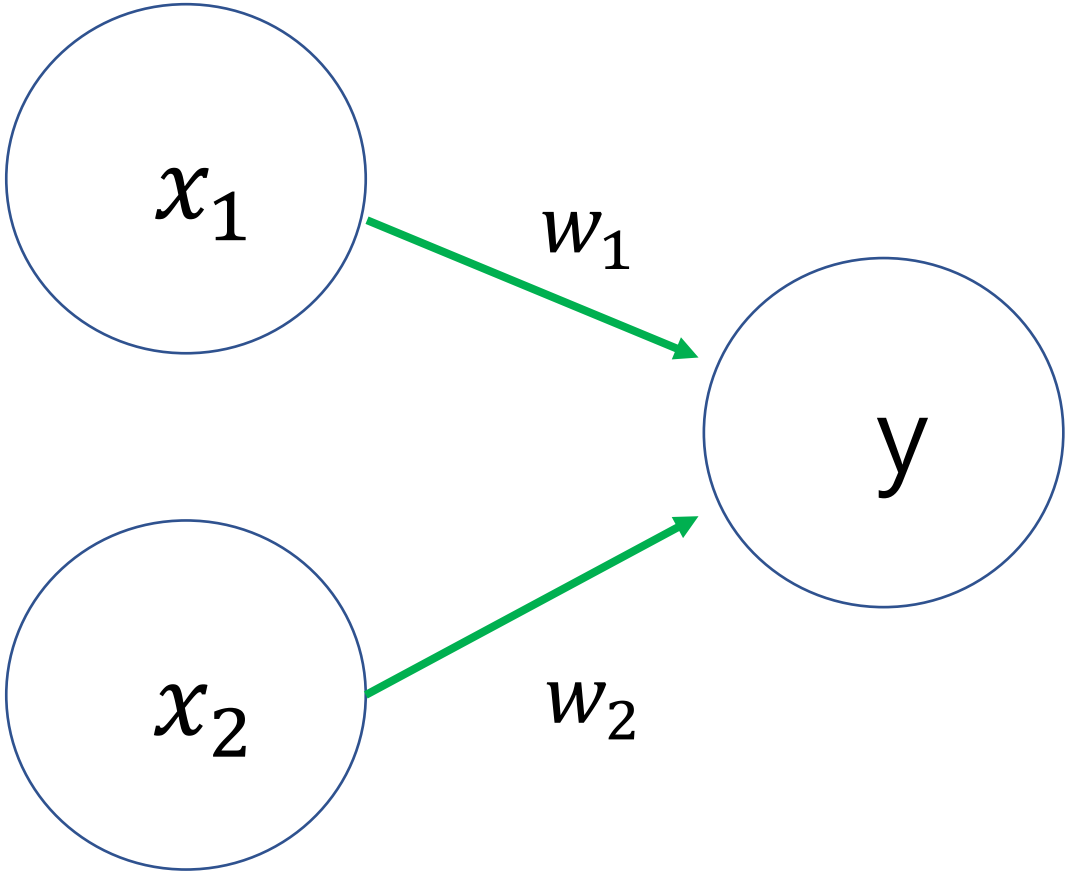
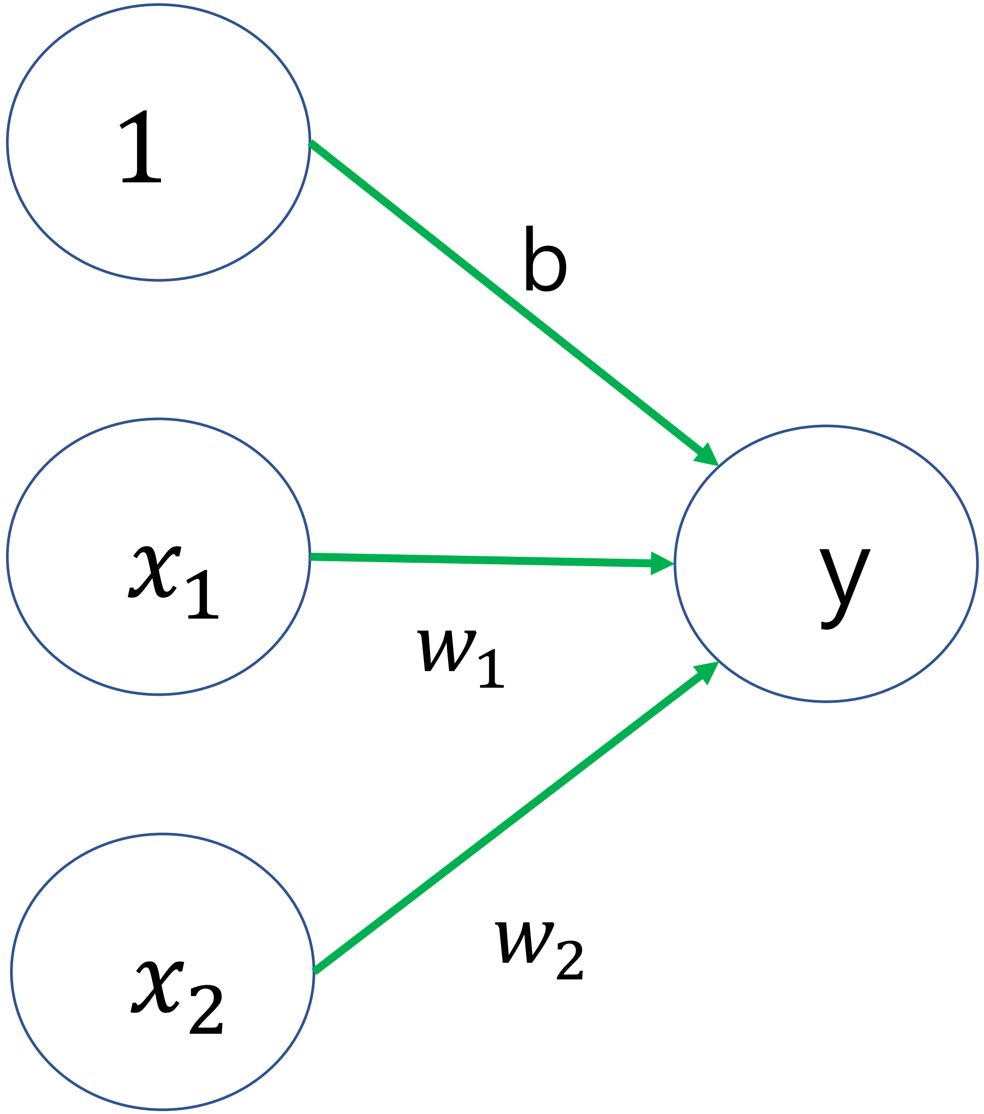
딥러닝 입문

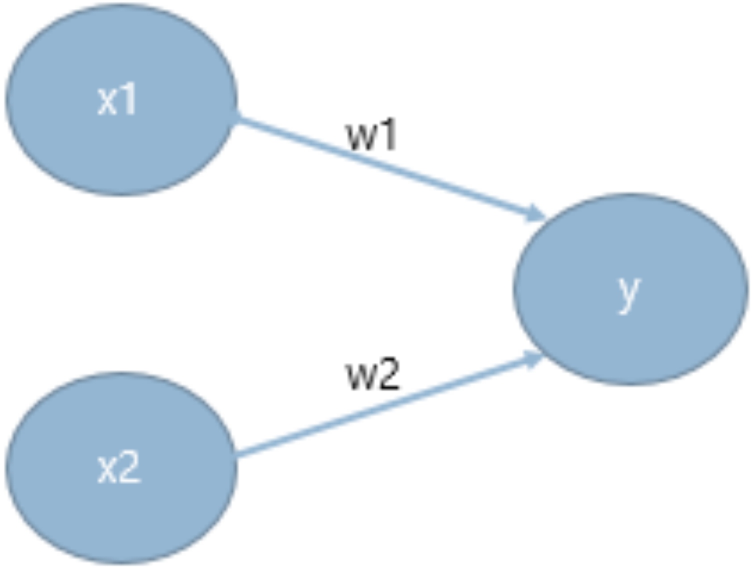
퍼셉트론(perceptron)



퍼셉트론(perceptron)



퍼셉트론 동작원리

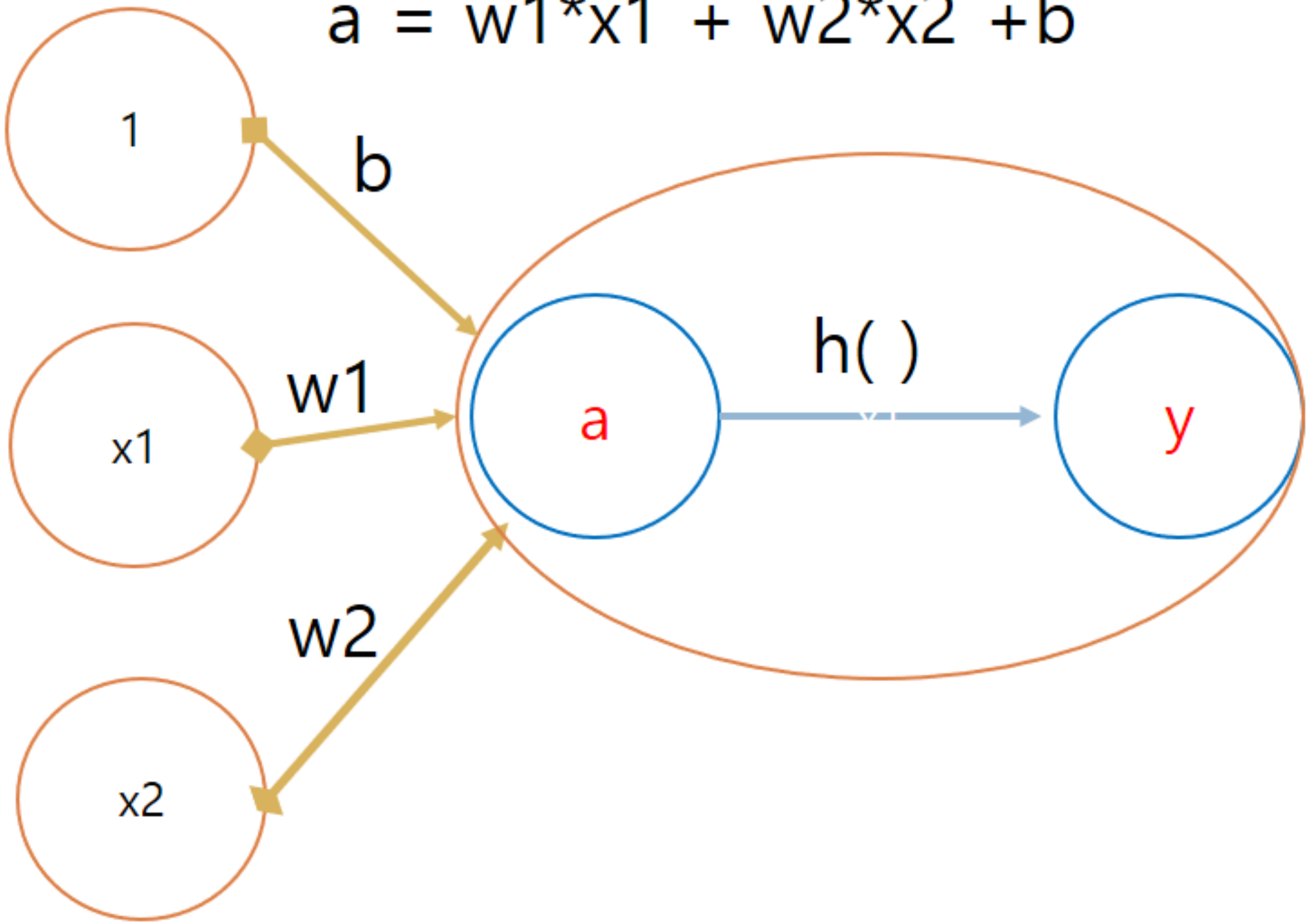


$$y = \begin{cases} 0 & (w1*x1 + w2*x2 - \theta \le 0) \\ 1 & (w1*x1 + w2*x2 - \theta > 0) \end{cases}$$

참고
θ(theta) : 임계값

활성화 함수의 등장

$$a = w1*x1 + w2*x2 + b$$



h( ) : 활성화 함수

활성화 함수

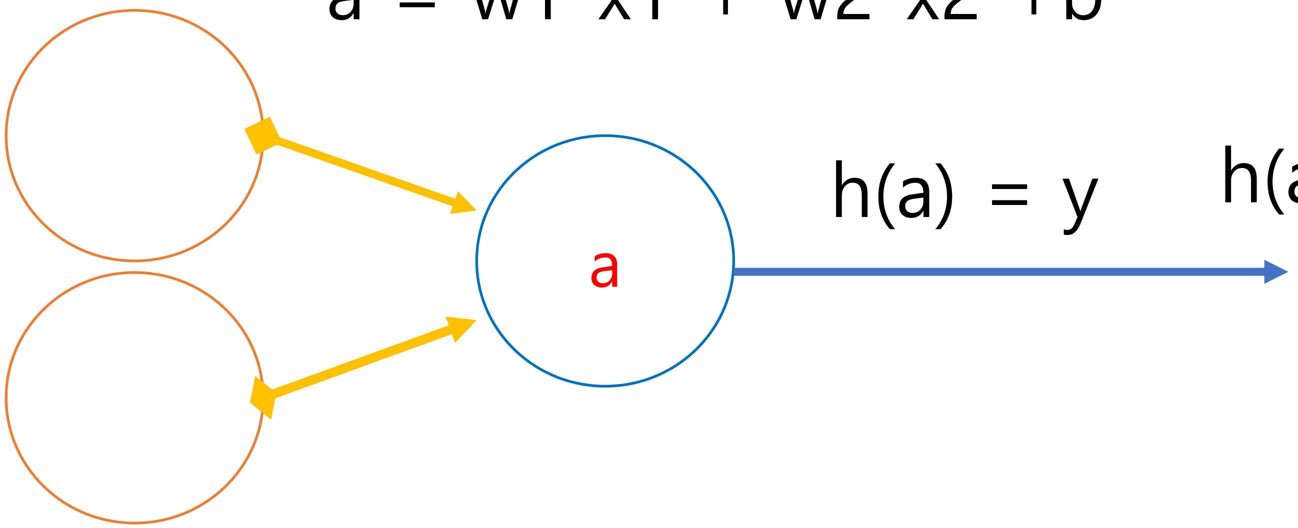
$$h(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases}$$

신경망이란?

퍼셉트론을 수식으로 표현

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

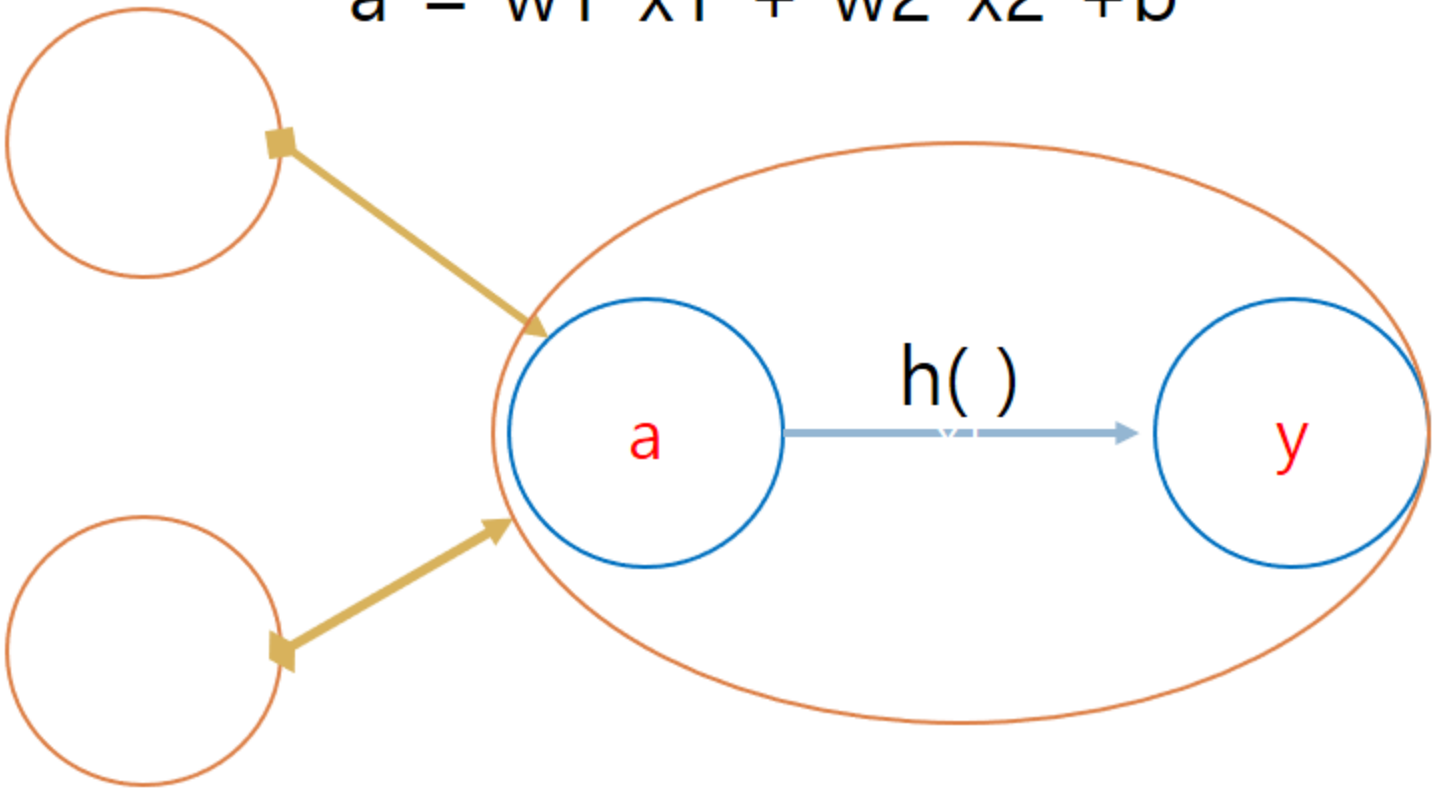
$$a = w1*x1 + w2*x2 + b$$



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

신경망이란?

$$a = w1*x1 + w2*x2 + b$$



퍼셉트론 & 신경망

단순 퍼셉트론은 단층 네트워크에서 **계단 함수를 활성화 함수로 사용**한 모델을 가리킨다.

다중 퍼셉트론은 신경망(여러 층으로 구성 시그모이드 함수 등의 활성화 함수 사용)을 **가리킨다.**

활성화 함수- 시그모이드 함수

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

exp(−x)는 $\boldsymbol{e}^{-x}$를 뜻한다.

e는 자연상수로 2.7182의 값을 갖는 실수이다.

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

다차원 배열의 연산

```
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)
```

행렬의 내적(곱)

```
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 내적(곱)

```
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)
```

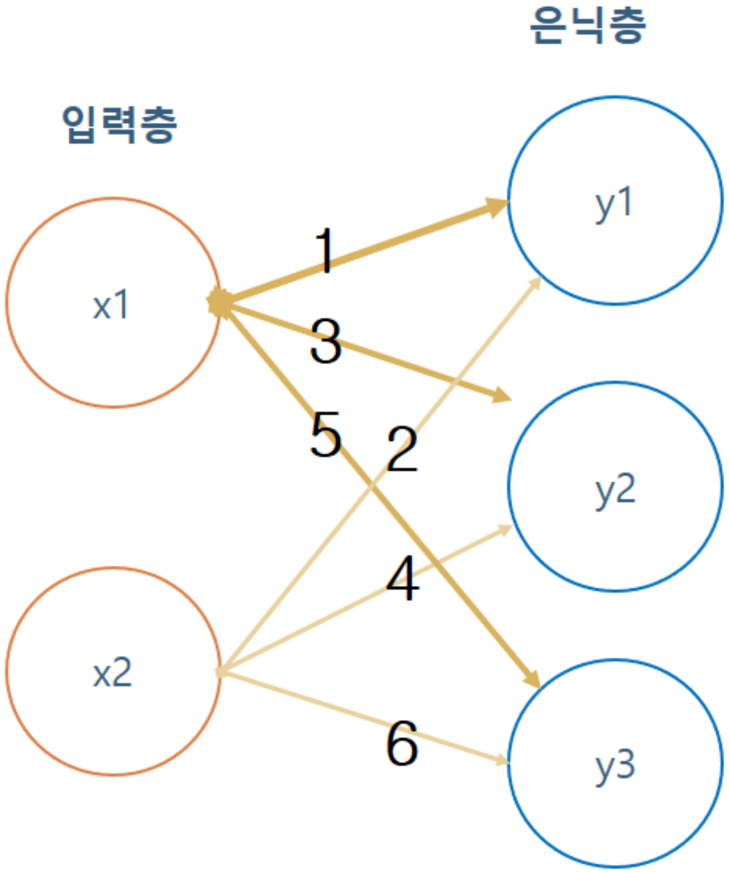
```
array([[19, 22],
       [43, 50]])
```

행렬의 내적(곱) 주의

행렬의 곱에서는 대응하는 차원의 원소 수를 일치 시켜야 함.

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

신경망의 내적



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

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

Hypothesis using matrix (n output)

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

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

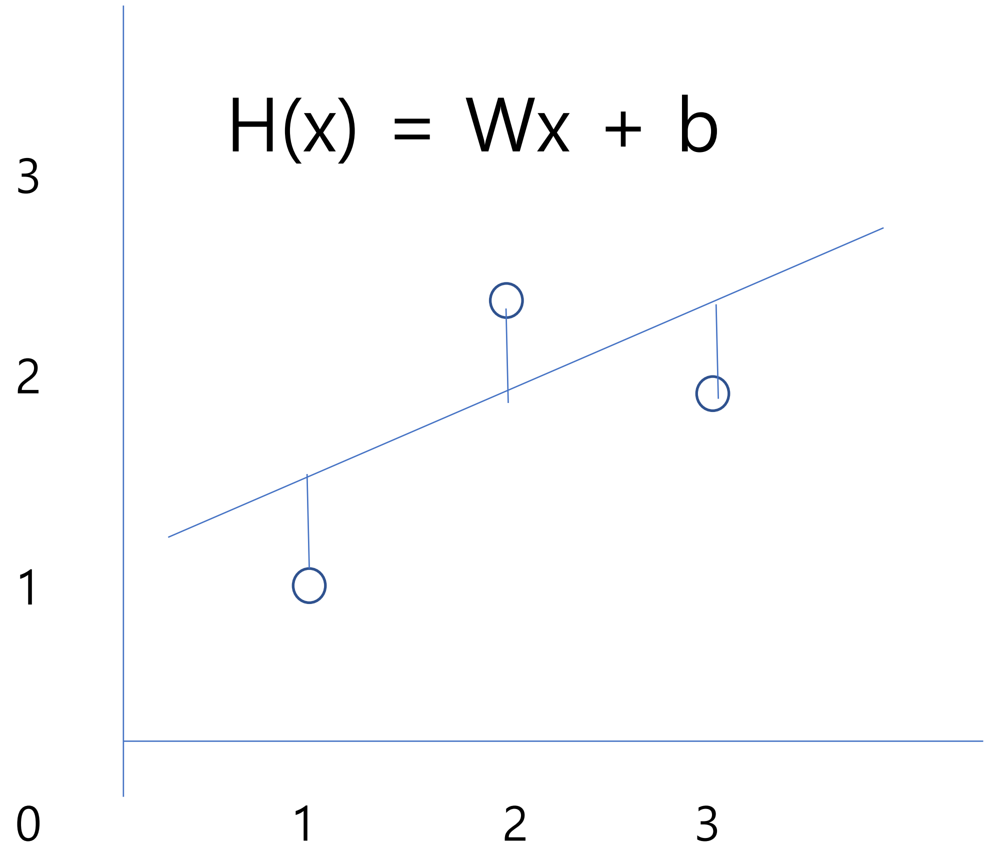
★ 첫번째 행렬의 열, 두번째 행의 행이 같다.

★ 첫번째 행렬의 행, 두번째 행의 열이 결과 행렬의 행렬이 된다.

Cost function

Hypothesis와 cost Function

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



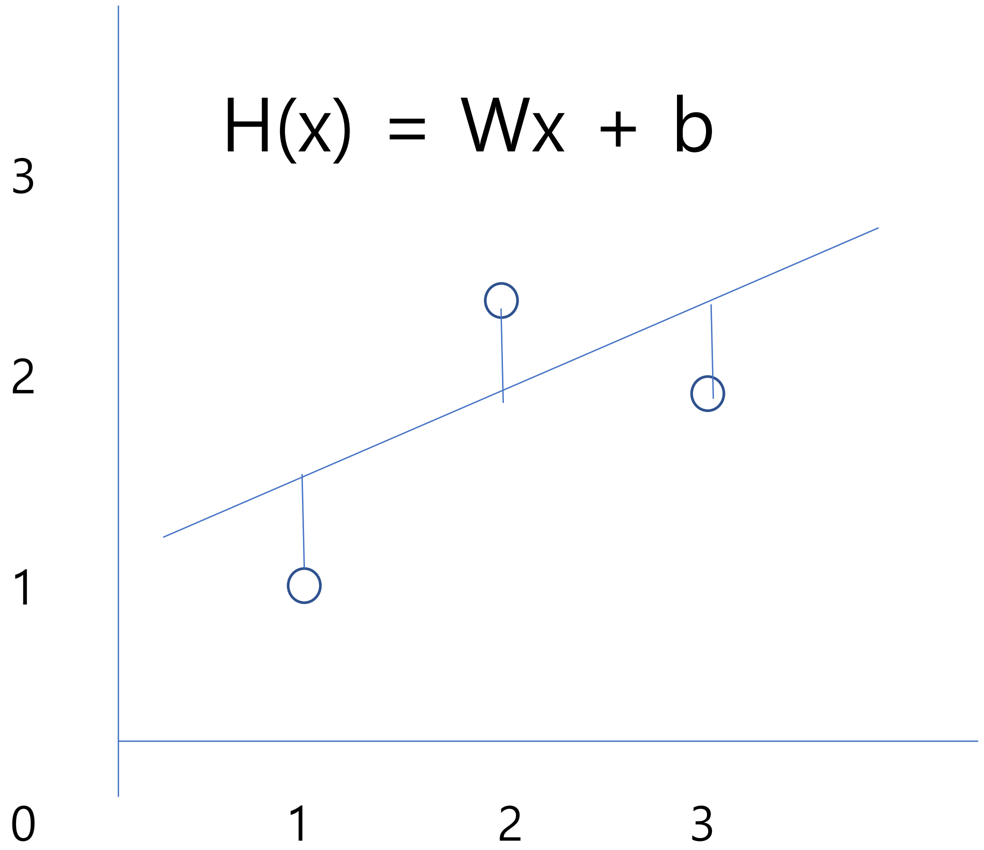
Cost function

Hypothesis와 cost Function

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

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

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



우리의 목표 : Minimize cost

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

Minimize cost(W,b)

W와 b를 통해 cost의 값을 최소화하는 알고리즘 구하기

02. Minimize cost

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

W가 1일때,

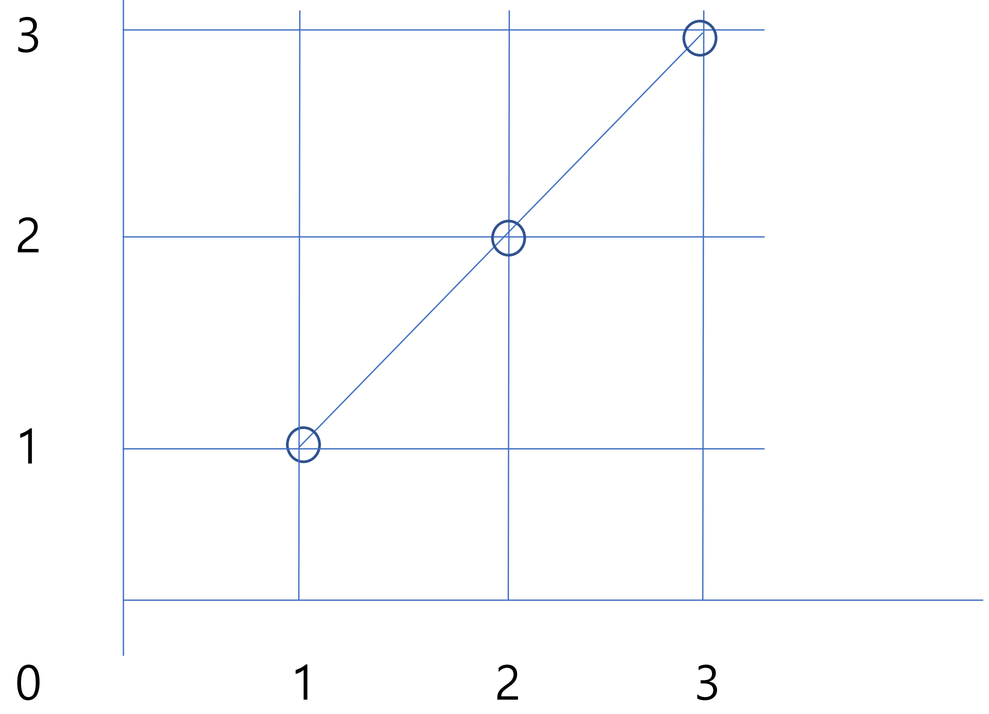
H(x) = Wx + b

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

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

X	Y
1	1
2	2
3	3

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



02. Minimize cost

$H(x) = Wx \rightarrow H(x)=0*x$

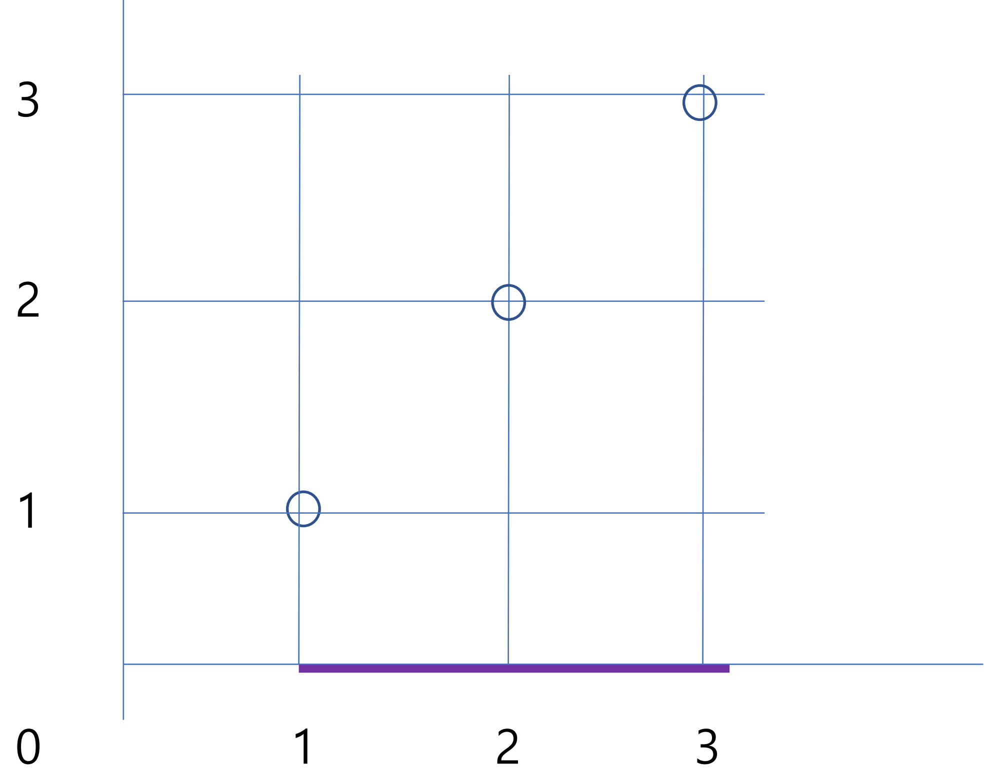
W가 0일때,

$H(x) = Wx + b$

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

X	Y
1	1
2	2
3	3

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



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

02. Minimize cost

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

W가 2일때,

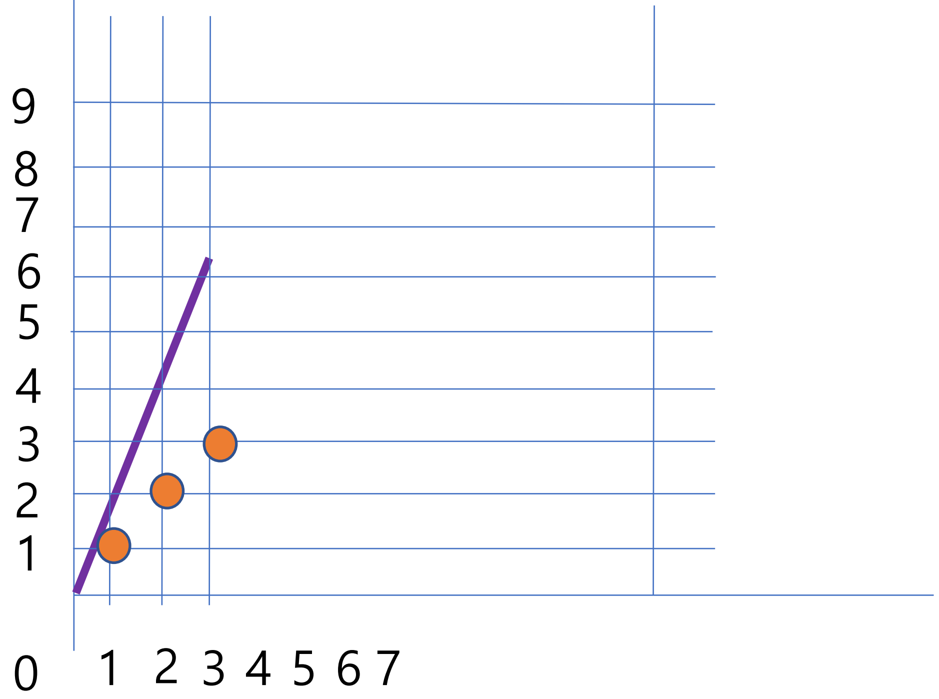
$$\frac{(2*1-1)^2+(2*2-2)^2+(2*3-3)^2}{3}=4.67$$

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

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



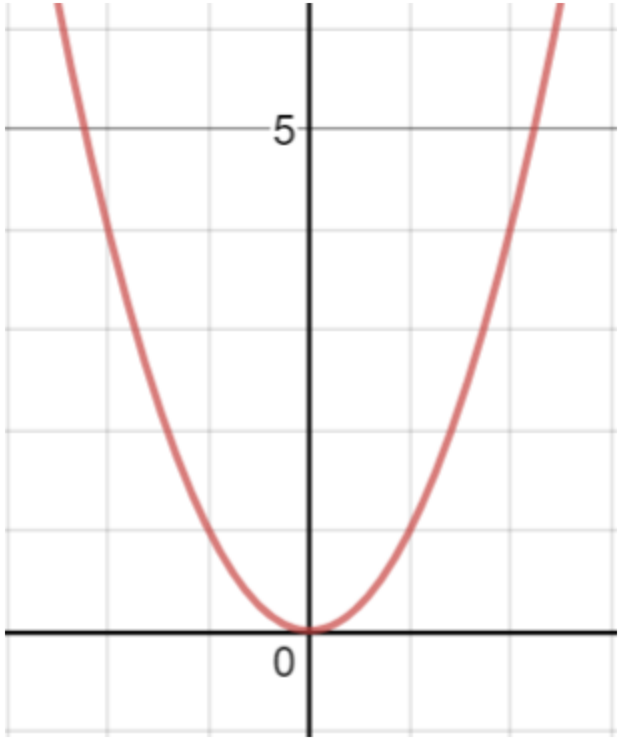
02. Minimize cost

$H(x) = Wx + b$

W가 0일때, cost(W)=4.67

W가 1일때 , cost(W)=0

W가 2일때, cost(W)=4.67



Gradient descent algorithm
경사 하강 알고리즘

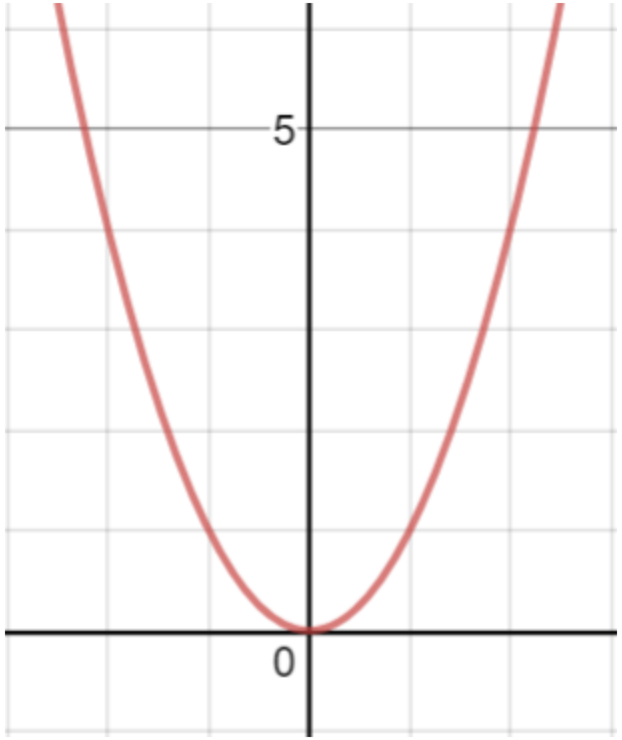
경사를 따라 내려가는 알고리즘 중의 하나이다.

가. Cost function을 최소화시키기

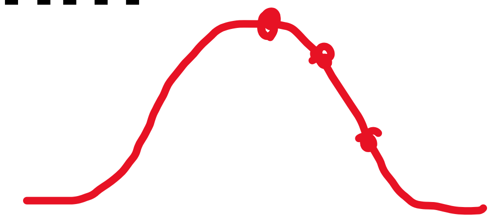
나. Gradient descent 는 여러 최소화 문제를 풀기 위해 사용된다.

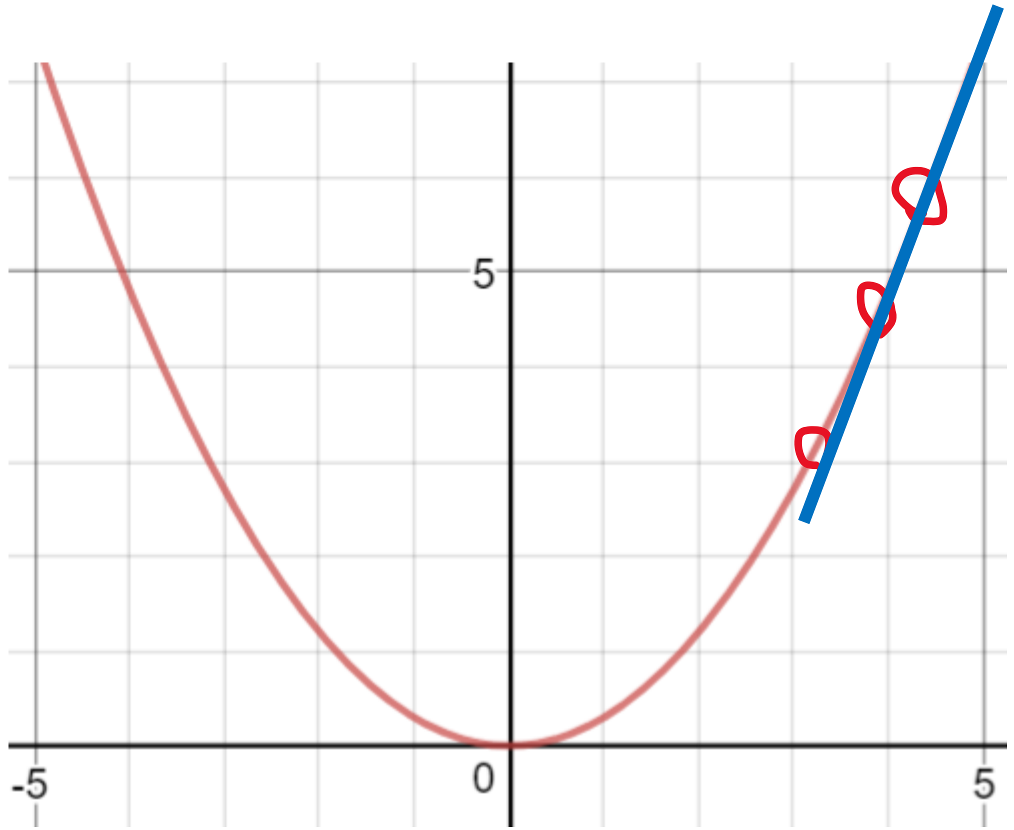
다. Cost function이 주어지고, 이 알고리즘은 cost를 최소화시키는 W와 b를 찾을 것이다.

라. W1, w2, w3...



Gradient descent algorithm
경사 하강 알고리즘





★ Cost function을 최소화 시키기

★ 어느 점에서 시작하든 항상 최저점에 도달한다.

★ 경사도는 미분을 이용하여 구할 수 있다.

★ 시작할 때, 끝나는 시점을 정할 수 있다.

How it works?(어떻게 하지)

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

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

★ 앞의 1/m이나 1/2m 같은 의미를 지닌다.

How it works?(어떻게 하지)

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

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

★ 앞의 미분시 1/m이나 1/2m에서 최소화할 때,
같은 의미를 지닌다.
미분을 할때 쉽게 하기 위해서

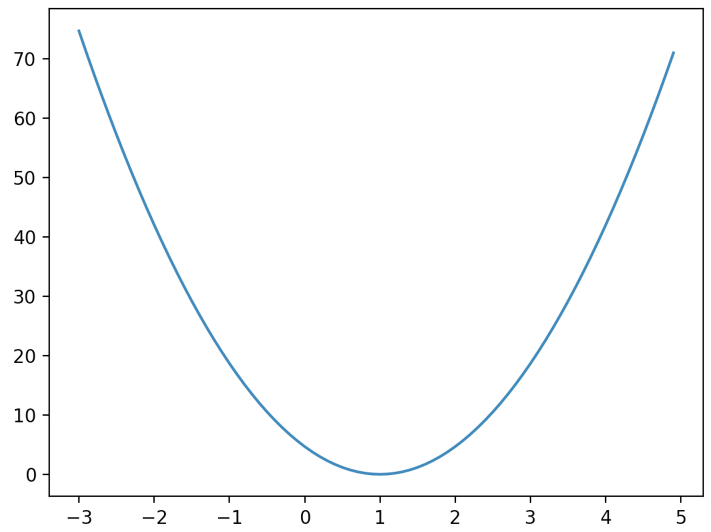
정의

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

Cost 함수를 미분하면 기울기를 구할 수 있다.

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

Cost(W)



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

W 학습 - 미분해 보기

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

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

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

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

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

Gradient descent algorithm

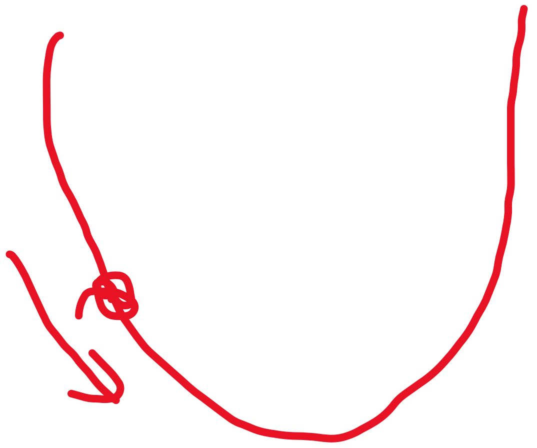
데이터 개수

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

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.

- **Hypothesis** $H(x) = Wx + b$
- **Cost function** $\mathrm{Cost}(W,b) = \frac{1}{m}\sum_{i=1}^{m}((Wx^{(i)}) - y^{(i)})^2$
- **Gradient descent algorithm**



MulitVariable

- Hypothesis

$$H(x) = Wx + b$$

$$H(x1, x2, x3) = w1x1 + w2x2 + w3x3 + b$$