

c:\>git clone <https://github.com/yungbyun/mllecture.git>

AI and Deep Learning

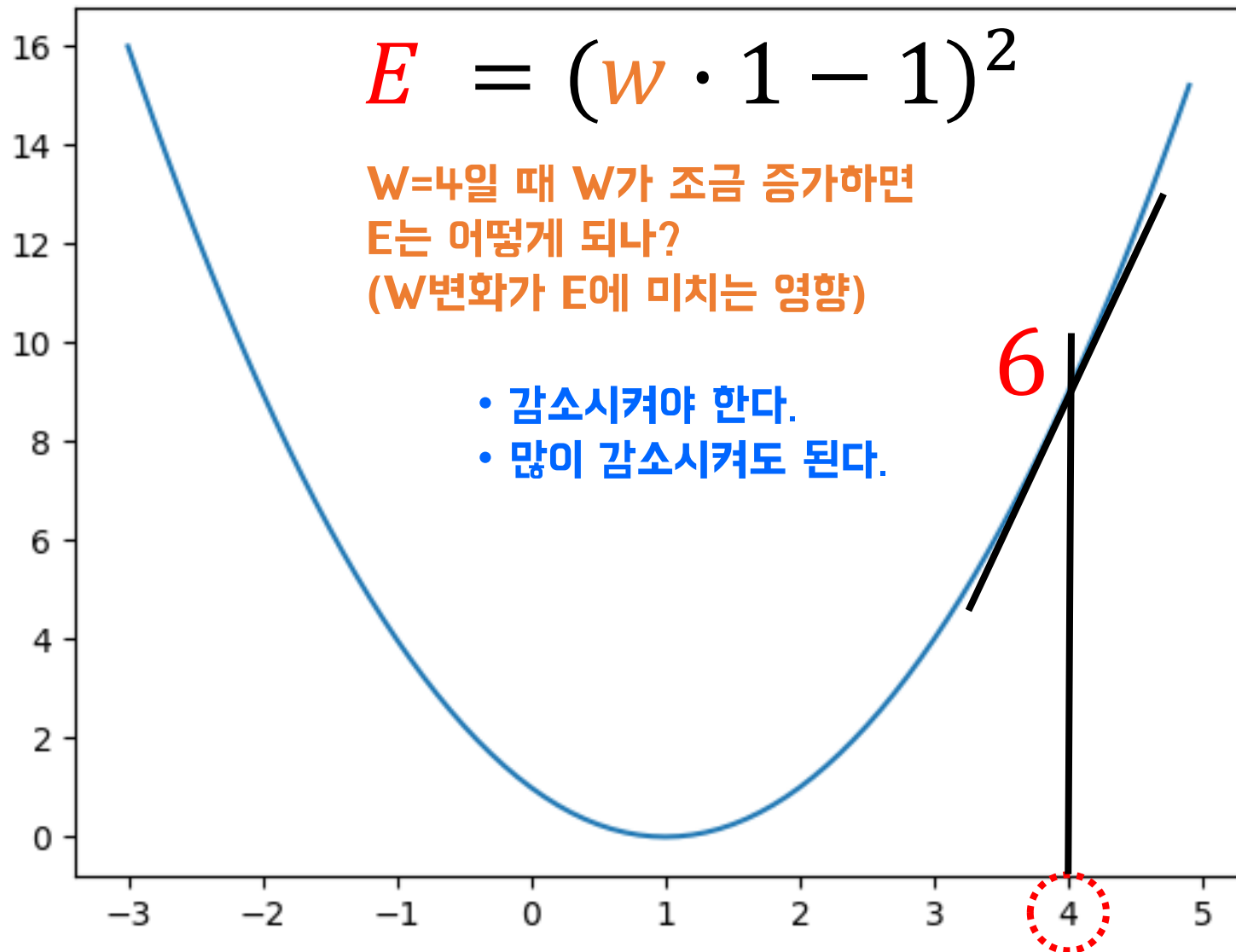
# Linear Regression & Back-propagation (2)

Jeju National University

Yung-Cheol Byun

# 어떻게 '자동으로'

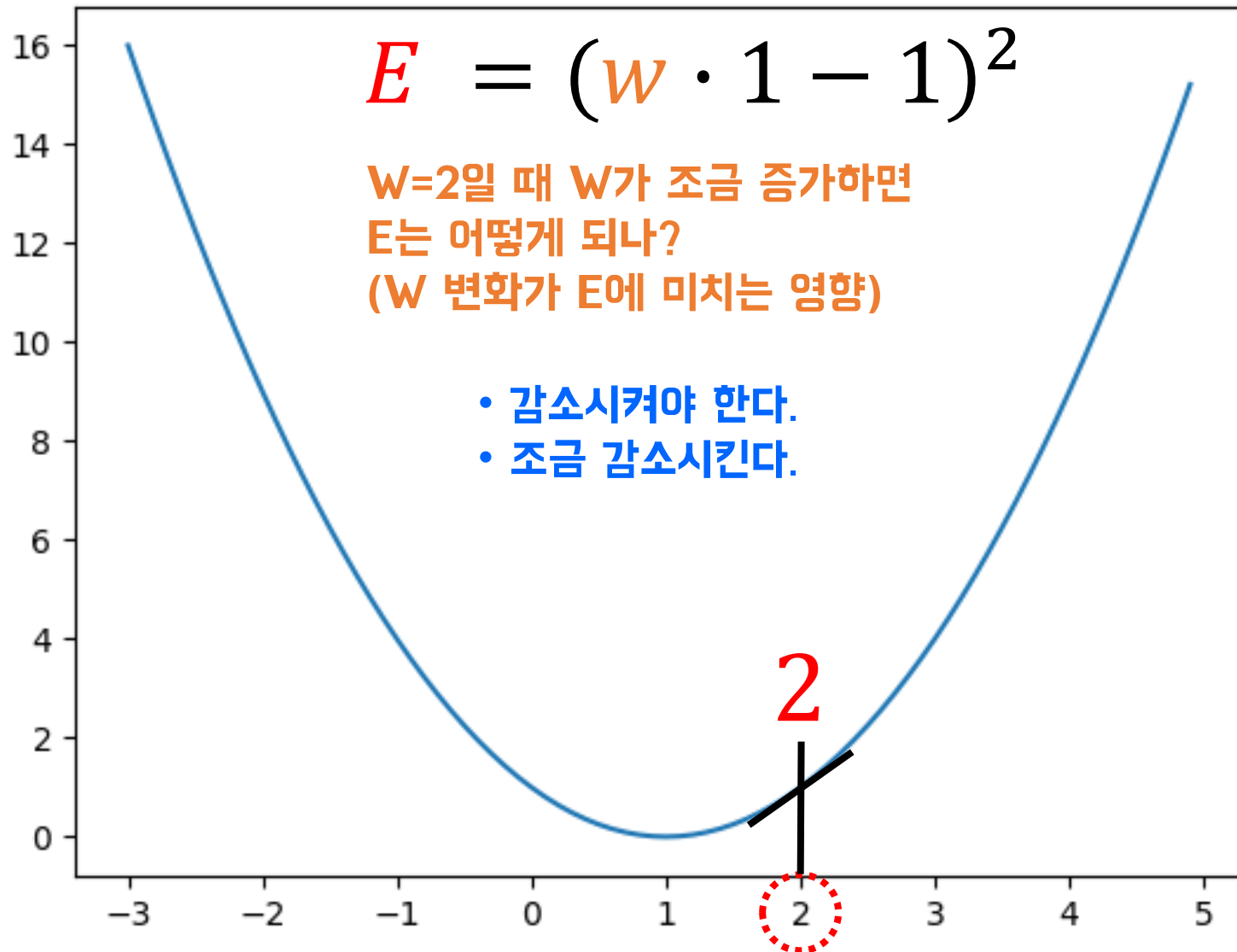
- 오류  $E$ 를 최소화하는  $w$   
값(시냅스 연결 강도,  
파라미터)을 찾을 수 있을까?

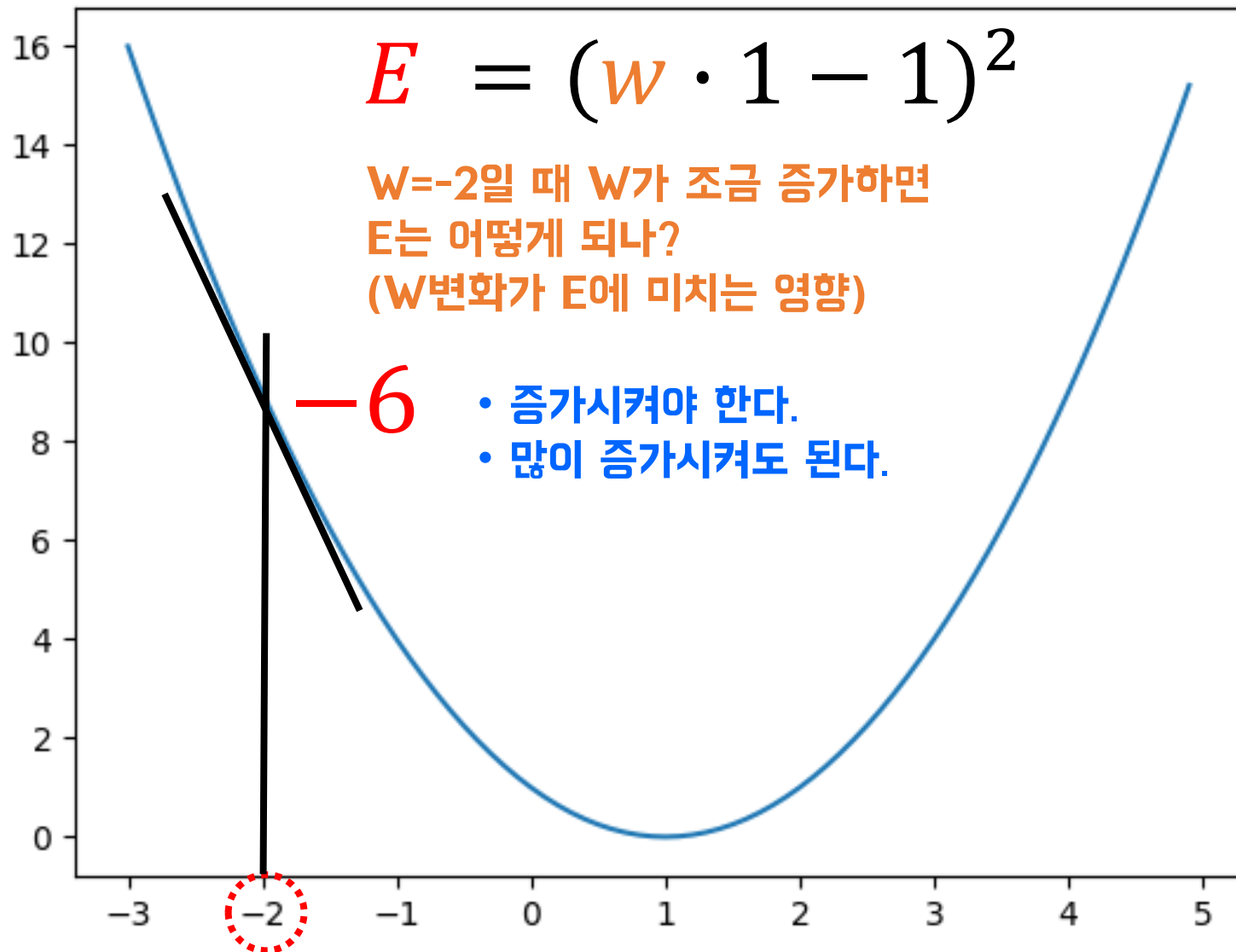


$$E = (w \cdot 1 - 1)^2$$

W=2일 때 W가 조금 증가하면  
E는 어떻게 되나?  
(W 변화가 E에 미치는 영향)

- 감소시켜야 한다.
- 조금 감소시킨다.

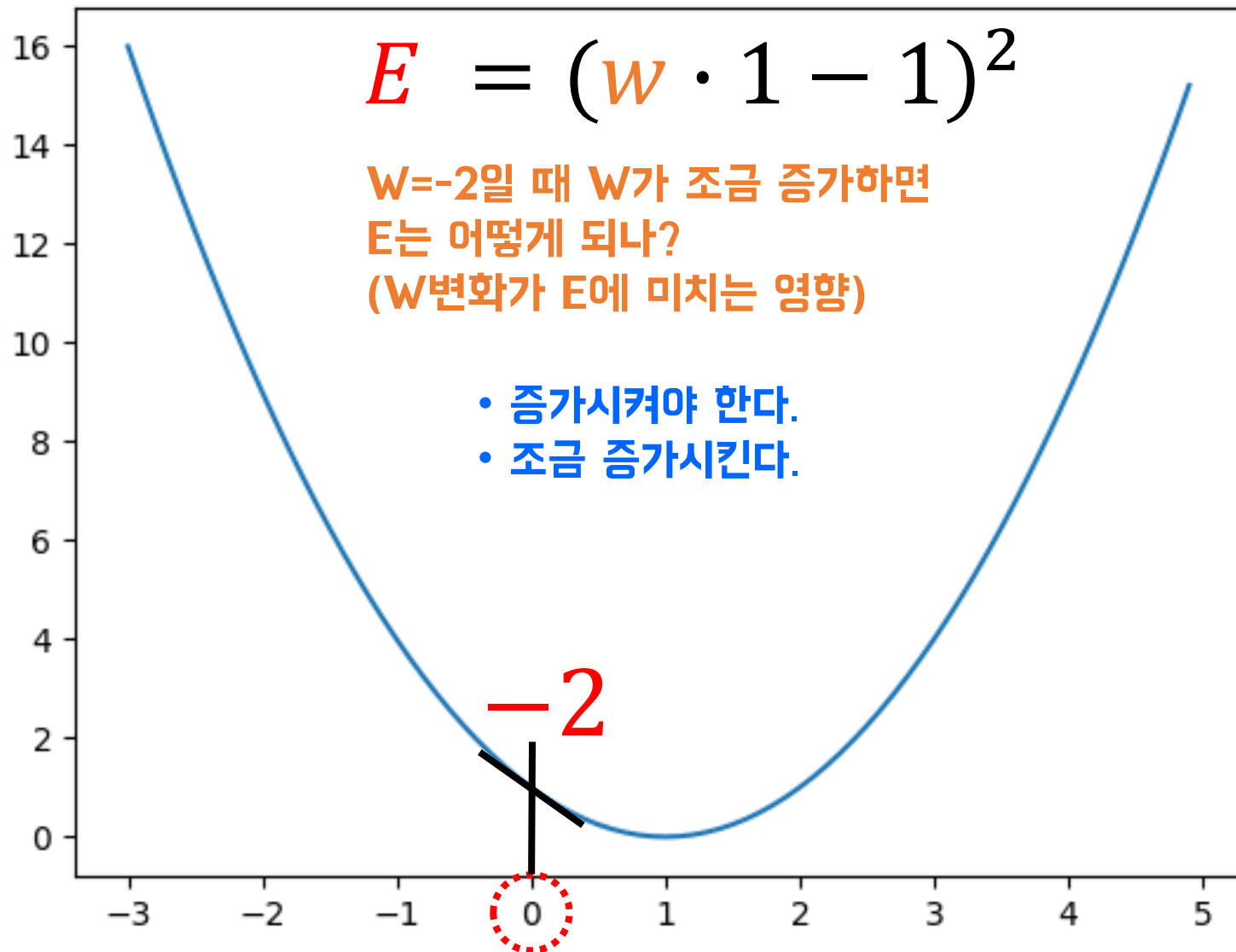




$$E = (w \cdot 1 - 1)^2$$

W=-2일 때 W가 조금 증가하면  
E는 어떻게 되나?  
(W변화가 E에 미치는 영향)

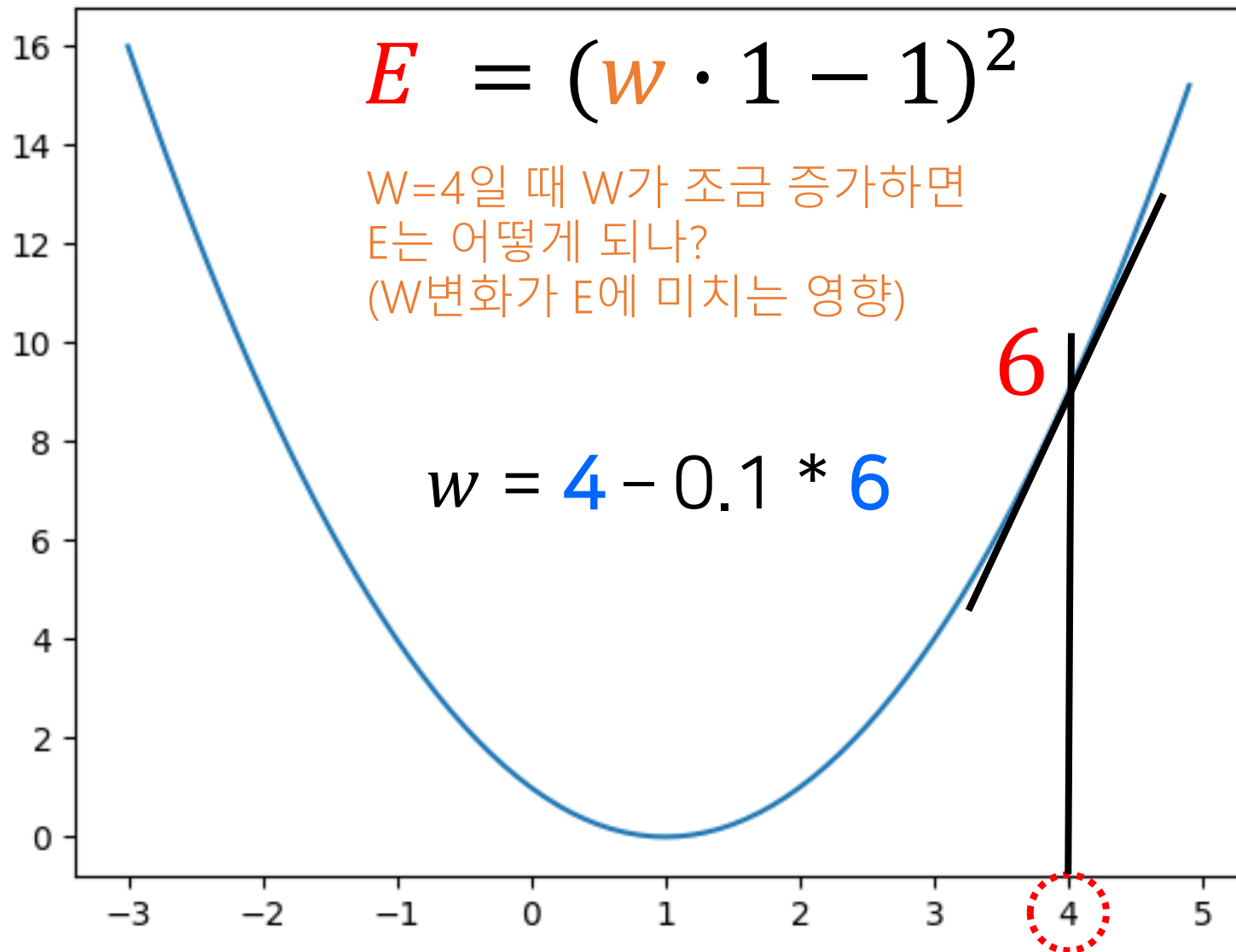
- 증가시켜야 한다.
- 조금 증가시킨다.



아래와 같이 하면 어떨까?

$$W = W - \alpha * 기울기$$

$\alpha$  = 얼마나 반영할 지를 의미하는 상수(가령, 0.1)

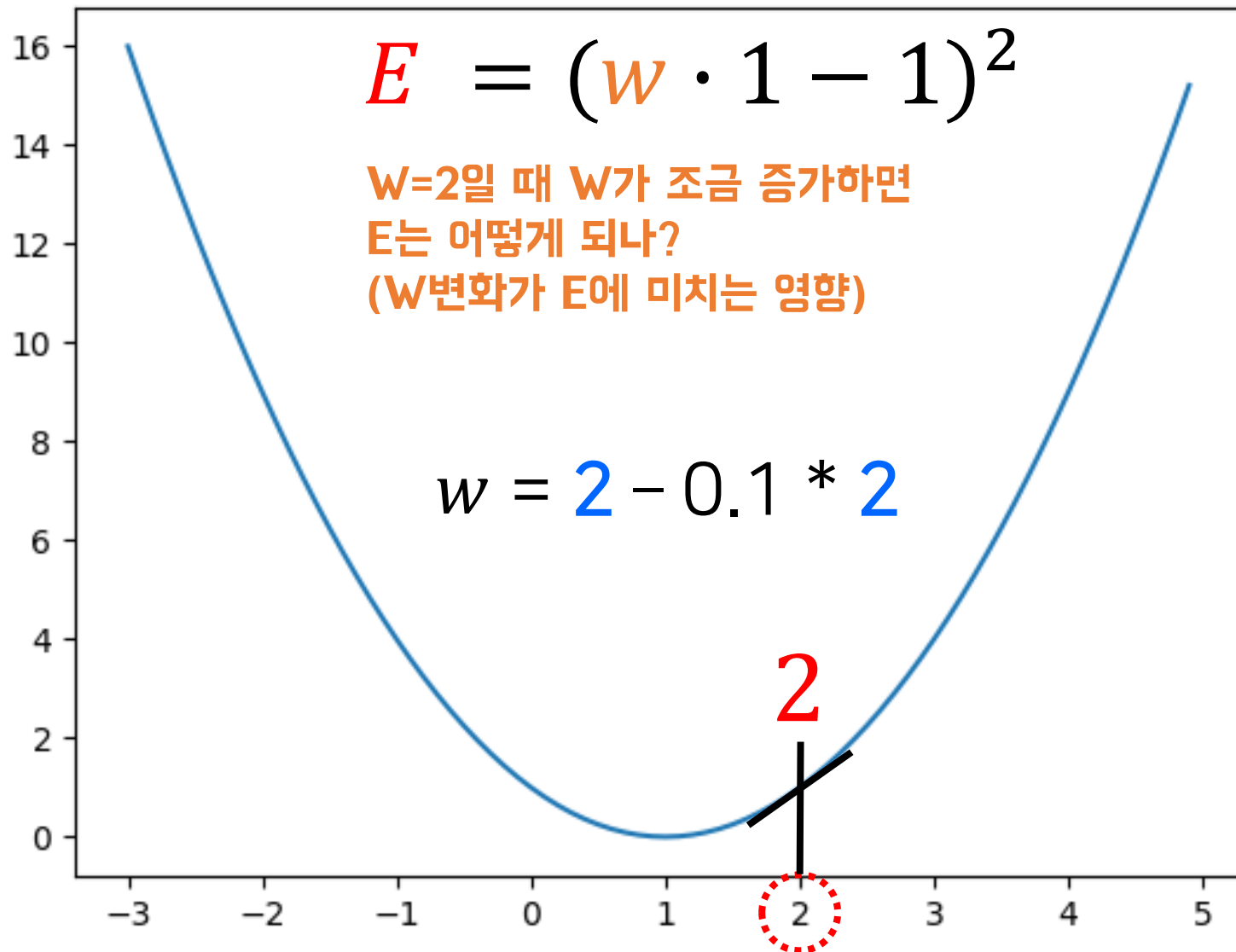


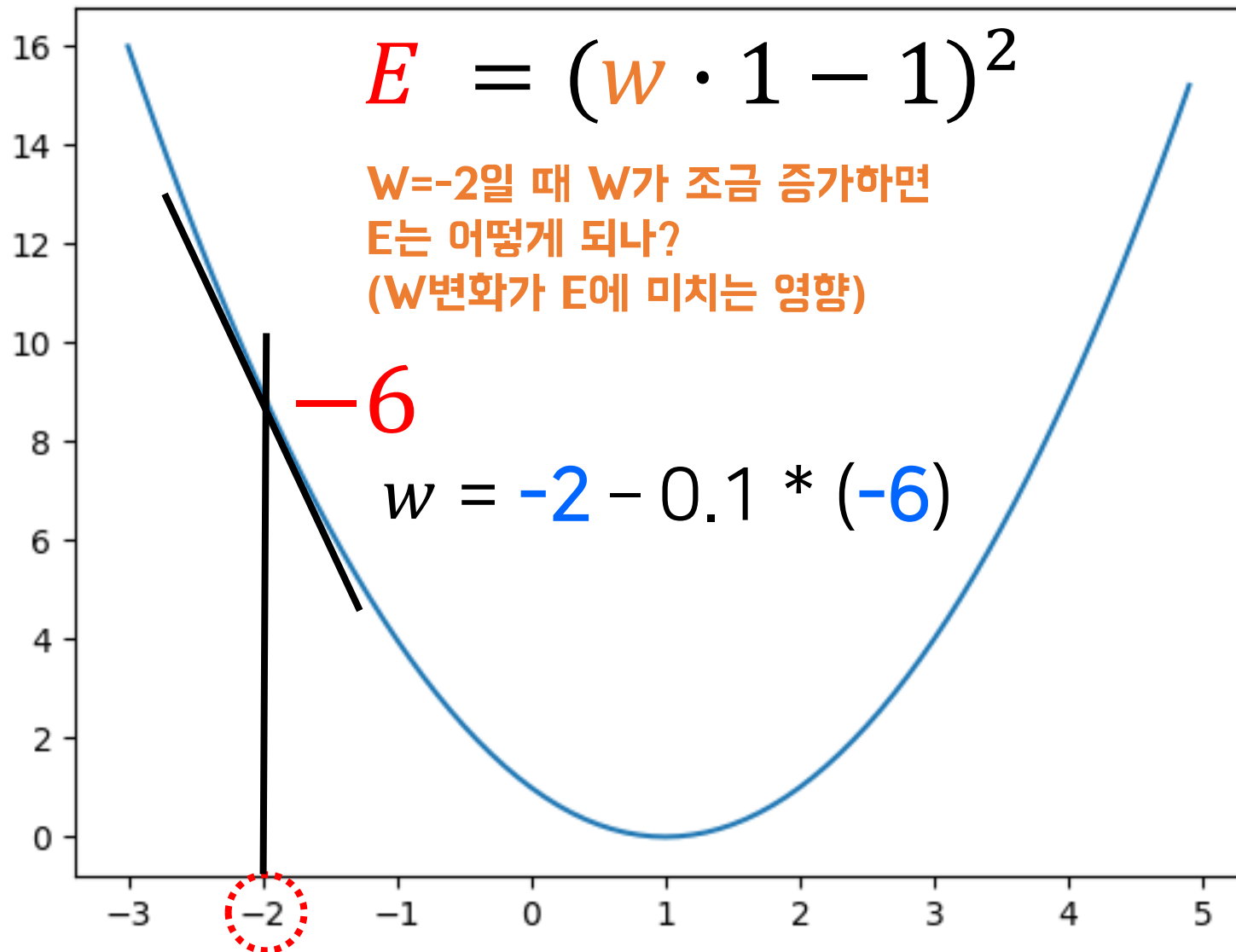


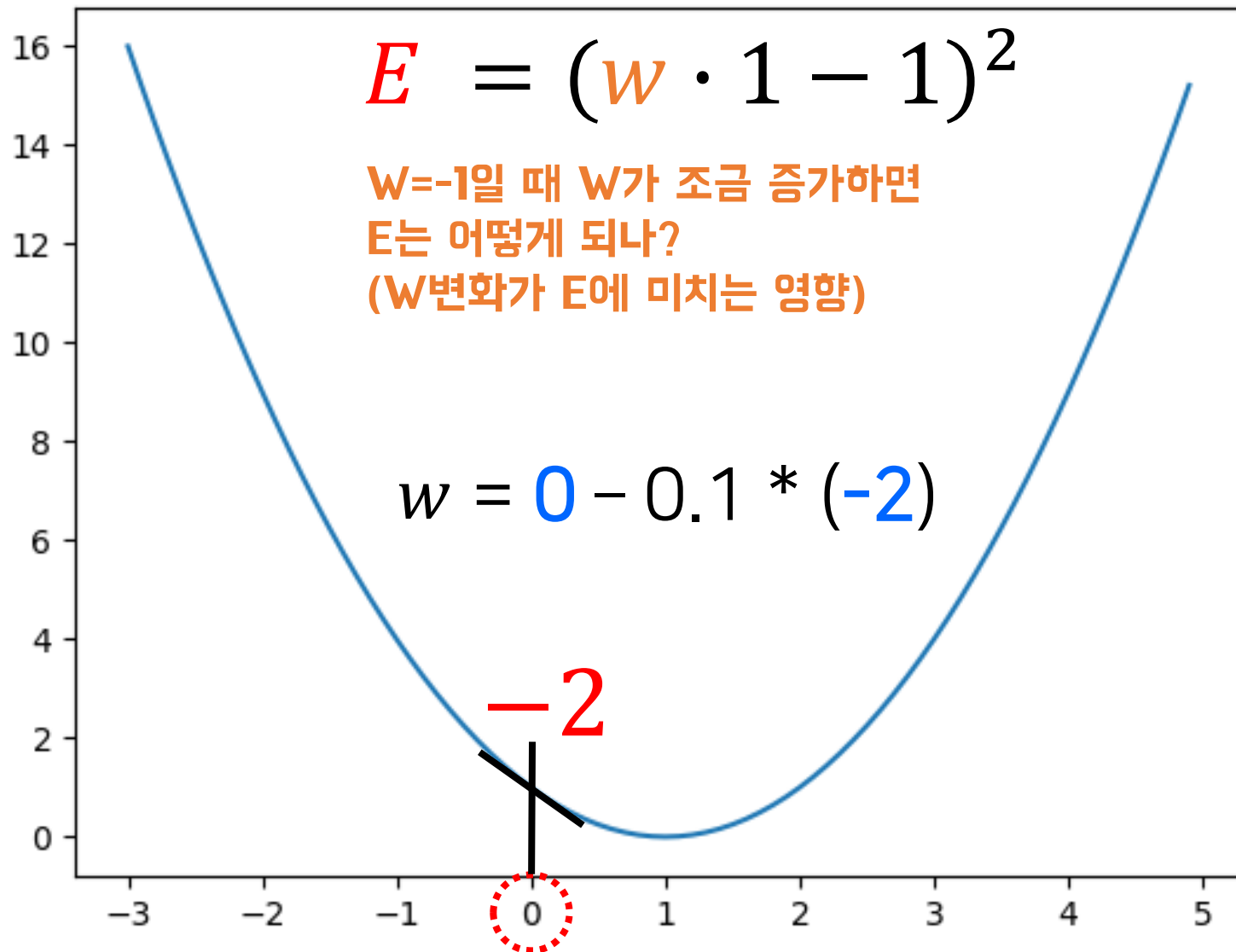
$$E = (w \cdot 1 - 1)^2$$

W=2일 때 W가 조금 증가하면  
E는 어떻게 되나?  
(W변화가 E에 미치는 영향)

$$w = 2 - 0.1 * 2$$







# 학습 방법( $w$ 업데이트)

- 난수로  $w$  값 초기화 (ex, -3)
- $w$  에서 기울기 구함
- 기울기로  $w$  를 업데이트

$$W = W - \alpha * (\text{기울기})$$

$\alpha$  : 반영 비율 (learning rate)

# TensorFlow

- 텐서플로우 프레임워크에서 가중치  $w$  (파라미터)를 자동으로 찾음(튜닝).
- 우리가 업데이트(튜닝) 하는 것이 아님.
- 이를 위해  $w$  를 텐서플로우 프레임워크 내에서 관리할 수 있도록 정의
- 또한 hypothesis와 cost\_function( $E$ )도 텐서플로우에서 계산할 수 있도록 정의

# TF를 이용한 선형 회귀 학습

③

```
W = tf.Variable(tf.random_normal([1]))
```

$x$

$x$

$w$

①

```
x = [1]
```

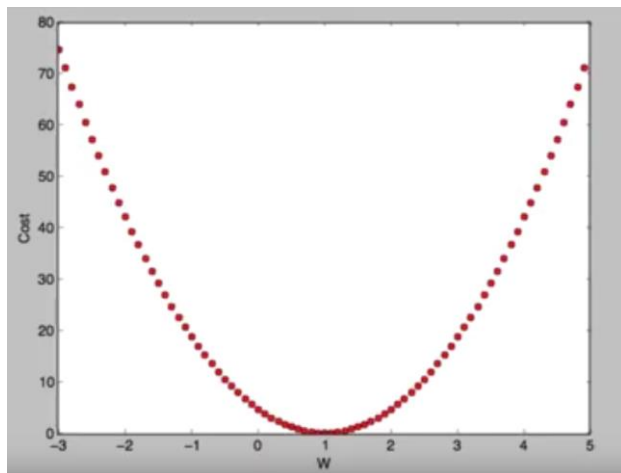
$\text{hypo} = w * x$

$h$

④

$y = [1]$

②



```
cost_function = (hypo - y) ** 2
```

⑤

$$E = (\text{hypo} - y)^2$$

01.py

Finding  $w$  in  
linear regression

```
import tensorflow as tf

#----- training data
x_data = [1]
y_data = [1]

#----- a neuron
w = tf.Variable(tf.random_normal([1]))
hypo = w * x_data

#----- learning
cost = (hypo - y_data) ** 2

train = tf.train.GradientDescentOptimizer(learning_rate=0.01).minimize(cost)

sess = tf.Session()
sess.run(tf.global_variables_initializer())

for i in range(1001):
    sess.run(train)

    if i % 100 == 0:
        print('w:', sess.run(w), 'cost:', sess.run(cost))

#----- testing(prediction)
x_data = [2]
print(sess.run(x_data * w))
```



02.py

Drawing  
cost function

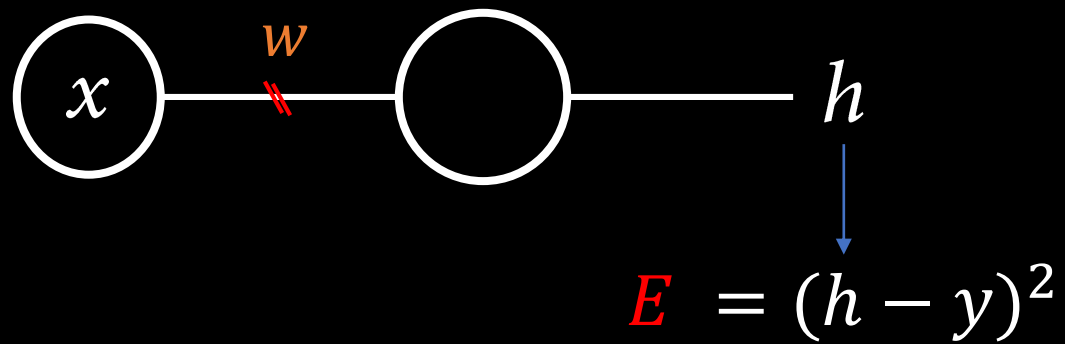
# 오류 함수 생각하기

$$E = \frac{1}{3} \sum_{i=1}^3 (wx_i - y_i)^2$$

# 오류 함수 생각하기

$$E = (wx - y)^2$$

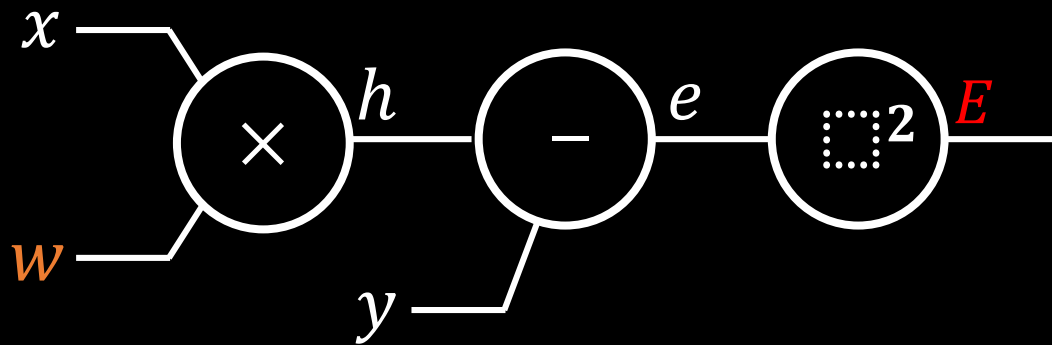
- 어느 부분이 뉴런인가?
- 뉴런의 모습 상상하기
- 입력 데이터는?
- 출력 데이터는?
- 시냅스는?
- 가설(hypothesis)은?
- 뉴런의 출력
- 오류 함수의 의미는?
- 뉴런 입력이 여러 개일 경우



# 오류 계산 그래프

$$E = (wx - y)^2$$

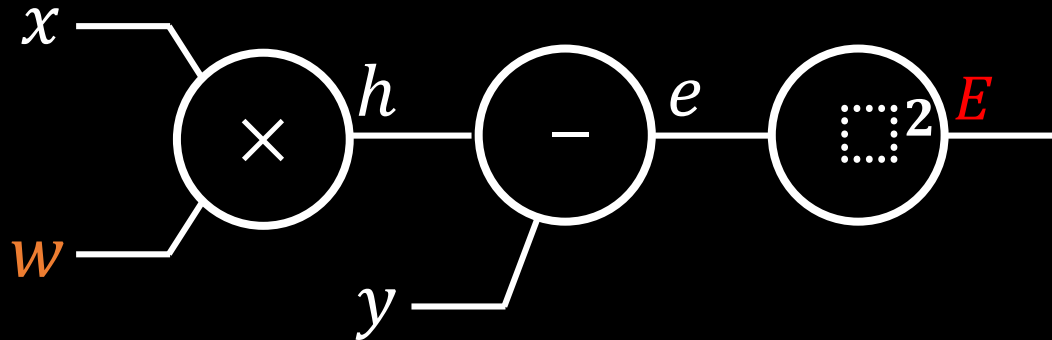
```
hypo = tf.multiply(W, x)  
cost_function = (hypo - y) ** 2
```



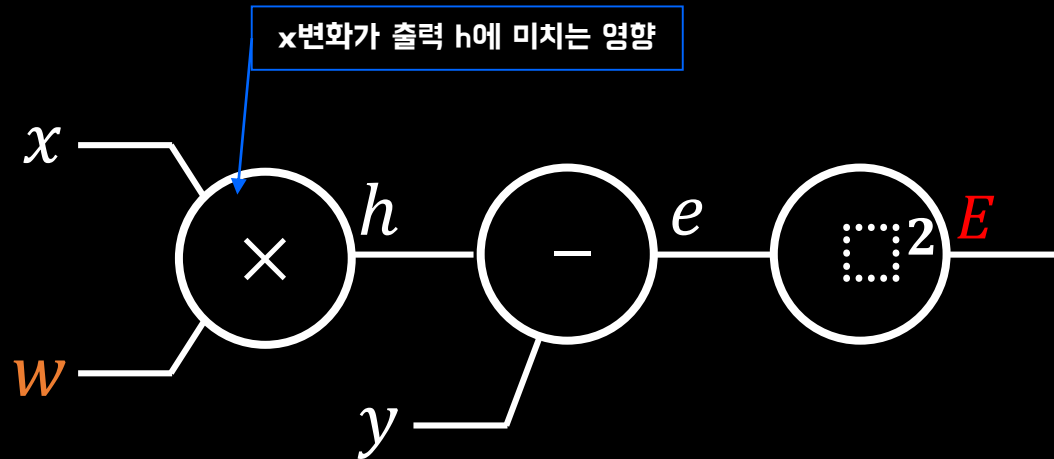
텐서란 무엇이고, 텐서 플로우란 무엇인가?  
텐서플로우 프레임워크가 파라미터 튜닝

# 계산 그래프와 미치는 영향

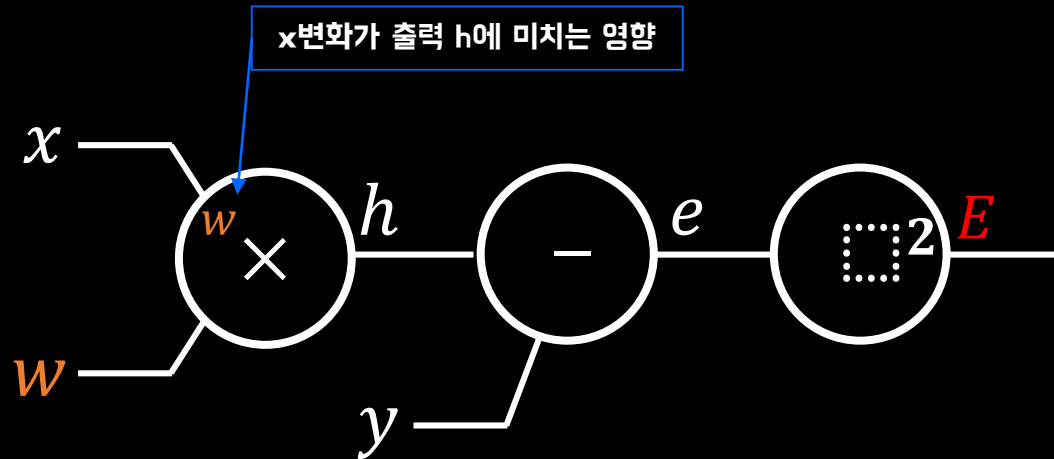
계산 그래프에서  $w$  가 E에 미치는 영향을 쉽게 알 수 있다.  
그러면 오류를 줄일 수 있도록  $w$  를 조절할 수 있다.



# 계산 그래프와 미치는 영향

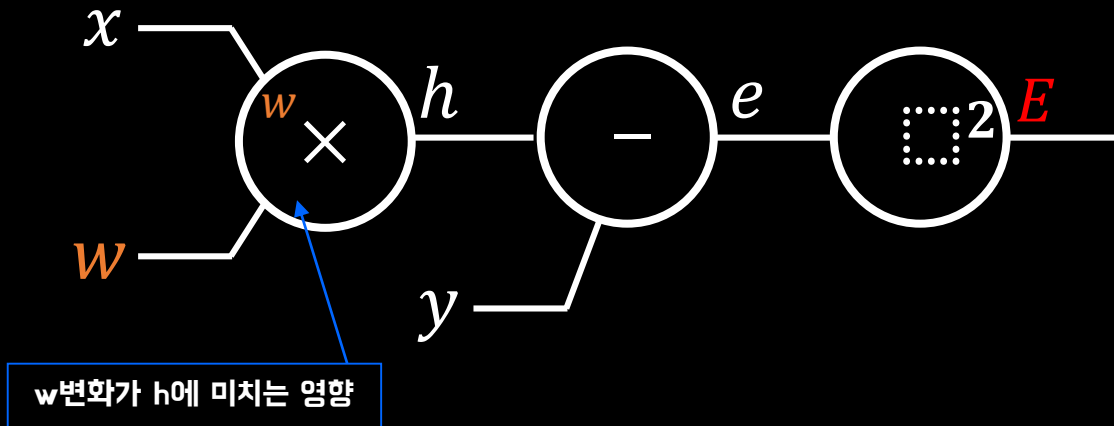


# 계산 그래프와 미치는 영향

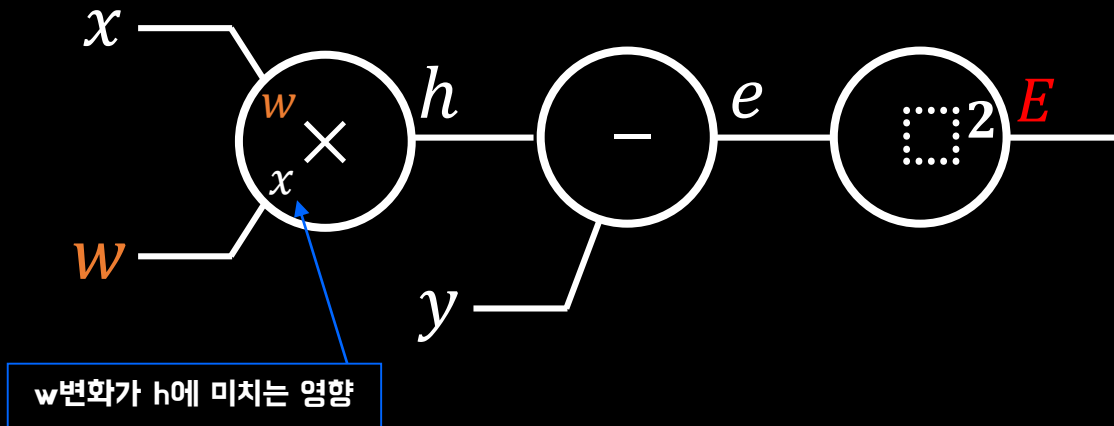




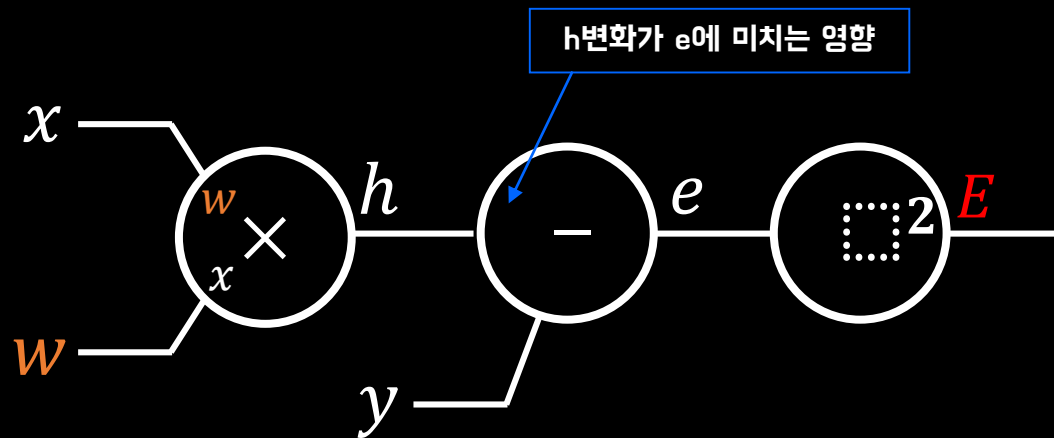
# 계산 그래프와 미치는 영향



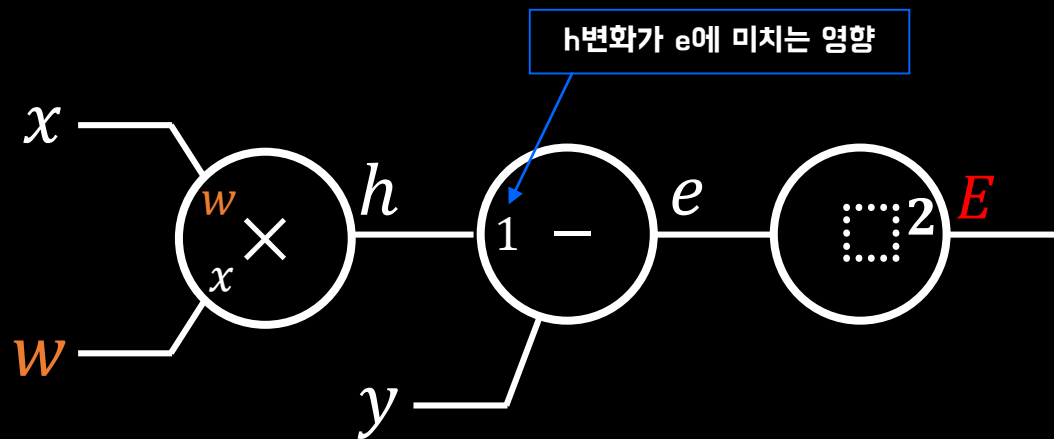
# 계산 그래프와 미치는 영향



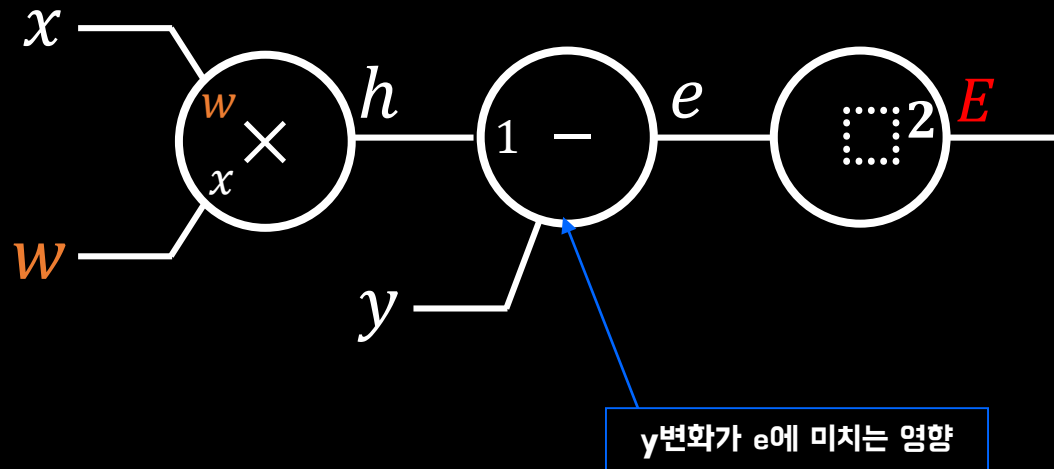
# 계산 그래프와 미치는 영향



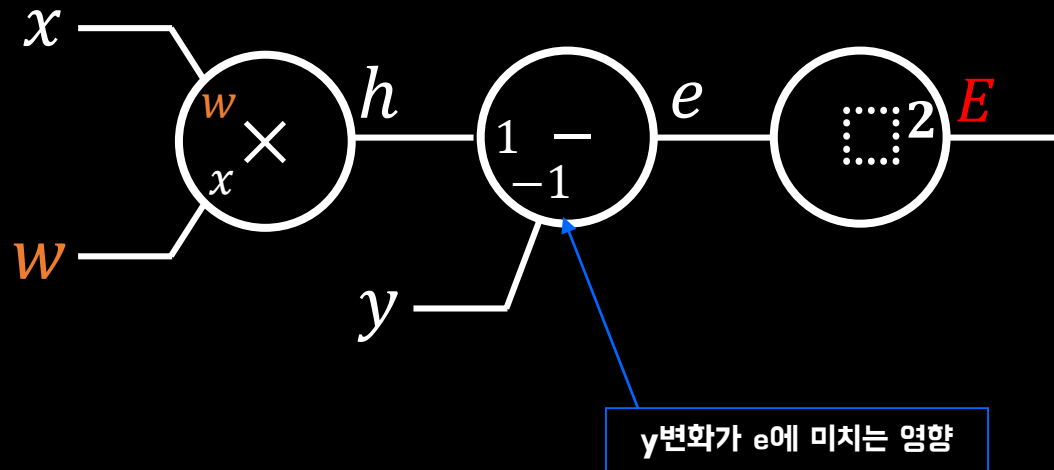
# 계산 그래프와 미치는 영향



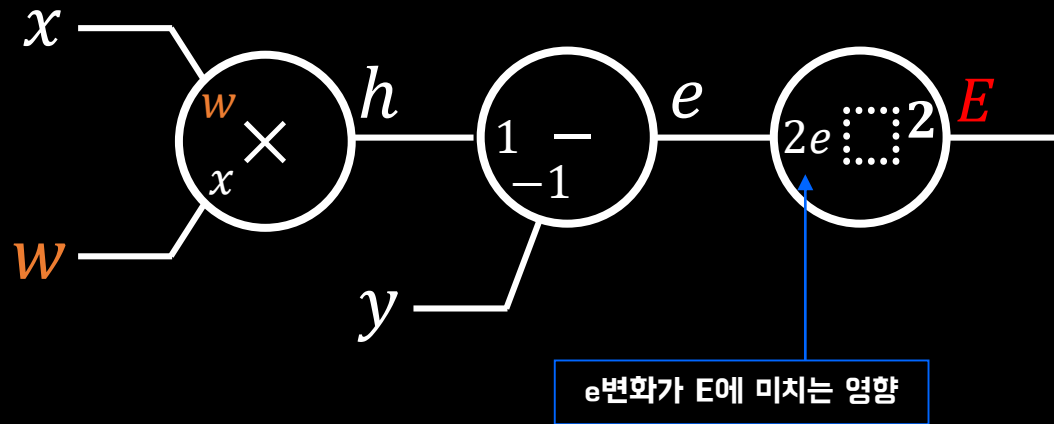
# 계산 그래프와 미치는 영향



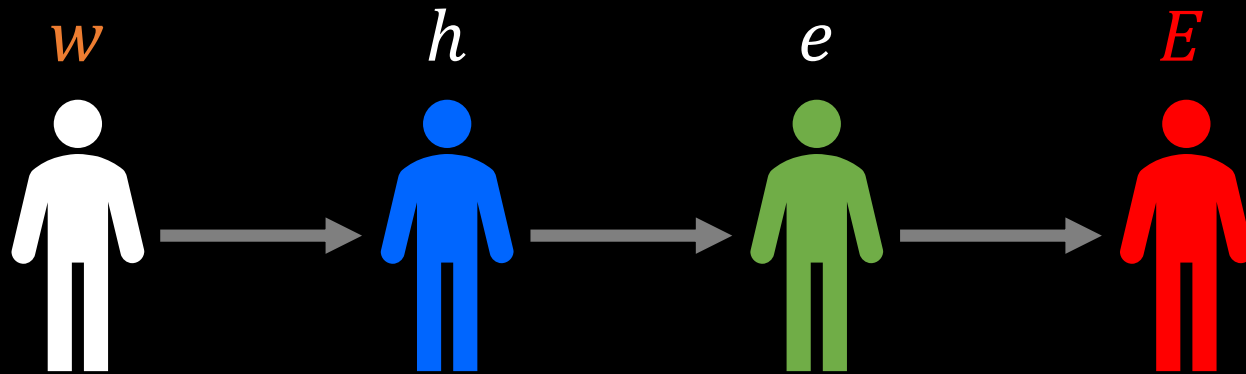
# 계산 그래프와 미치는 영향



# 계산 그래프와 미치는 영향



# 사람 사이의 영향력



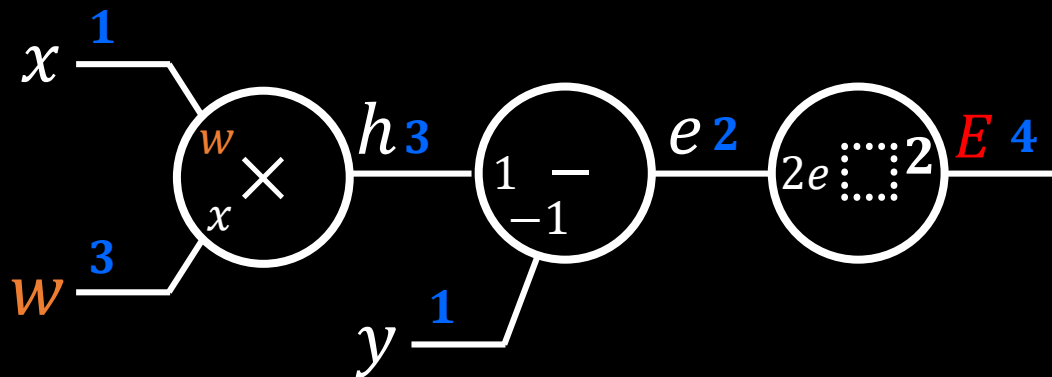
체인 룰 (chain rule)



$w$  변화가  
 $E$  에 미치는 영향력은?

# 앞으로 전파

$(x, y) = (1, 1)$ 이고  $w$ 는 3일 경우 **에러** 값은?



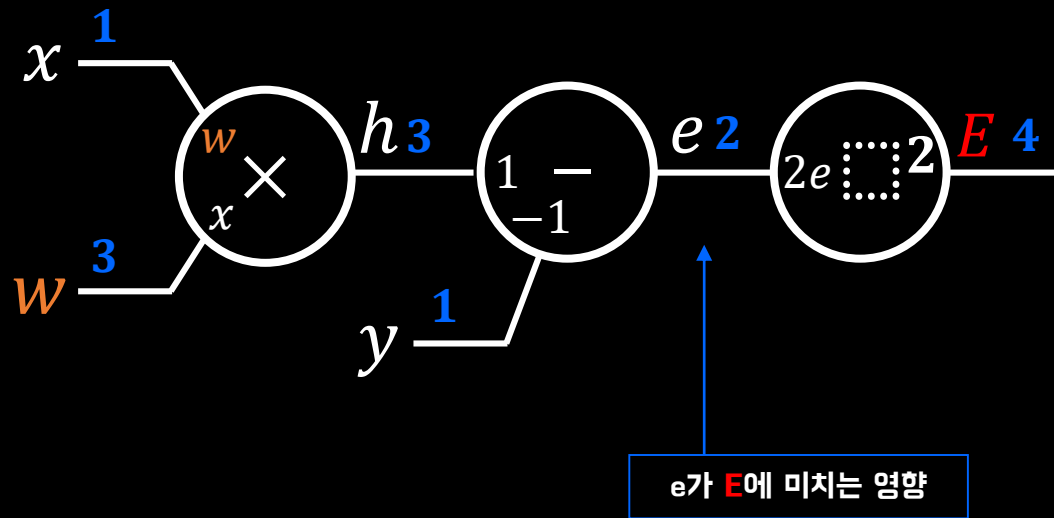
**에러(E)**가 크다.

**에러(E)**가 줄어들도록  $W$ 를 조절하자. 어떻게???

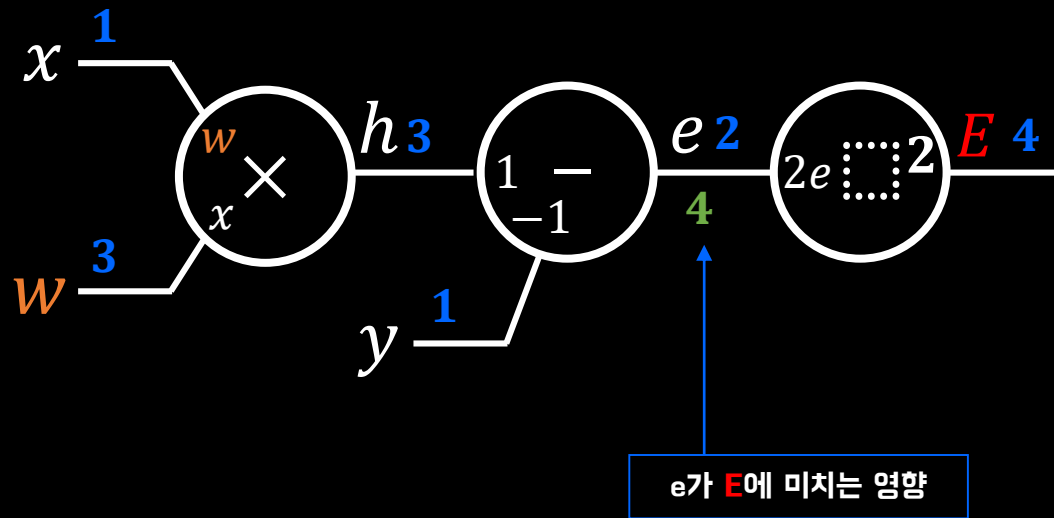
$W$ 변화가 **E**에 미치는 영향(기울기)을 구한 후  $W = W - \alpha * (\text{기울기})$

미치는 영향, 기울기를 구하  
자.

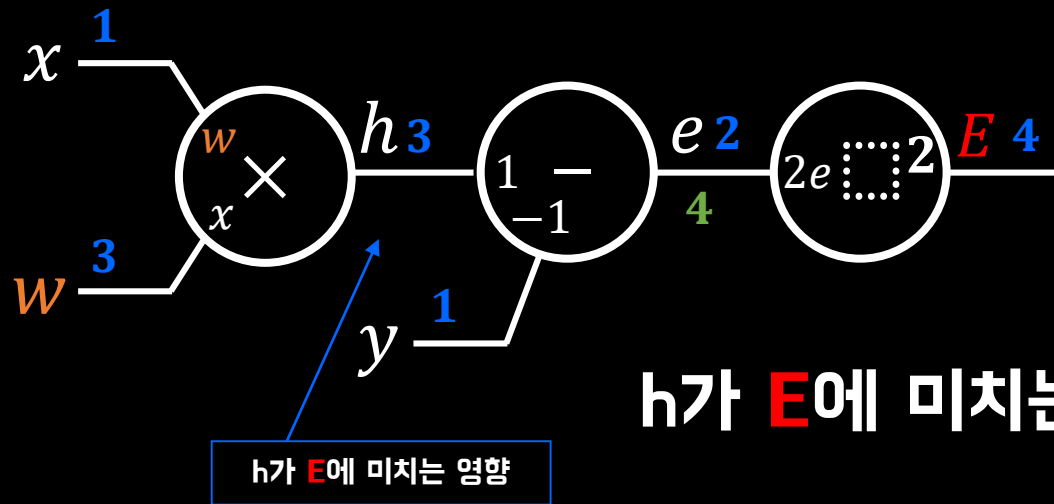
# 뒤로(역) 전파



# 뒤로(역) 전파

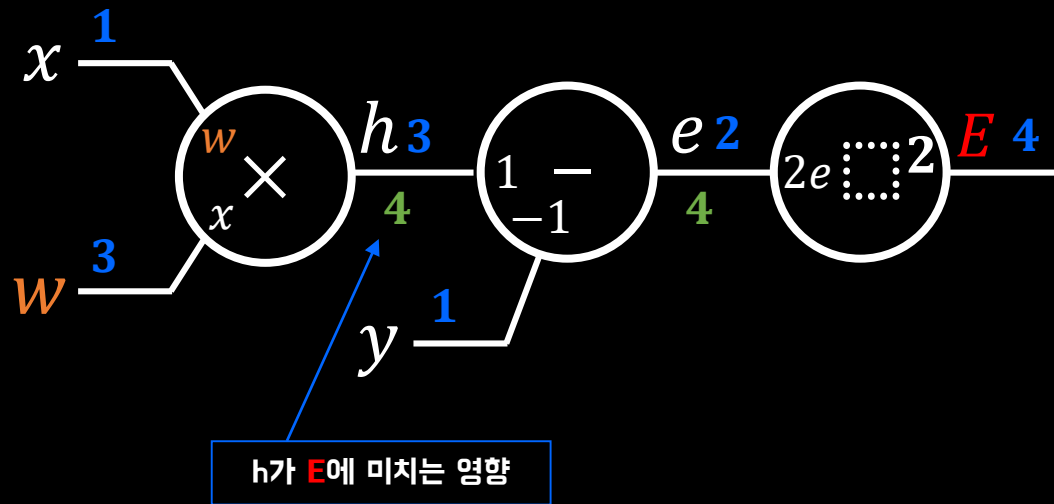


# 뒤로(역) 전파

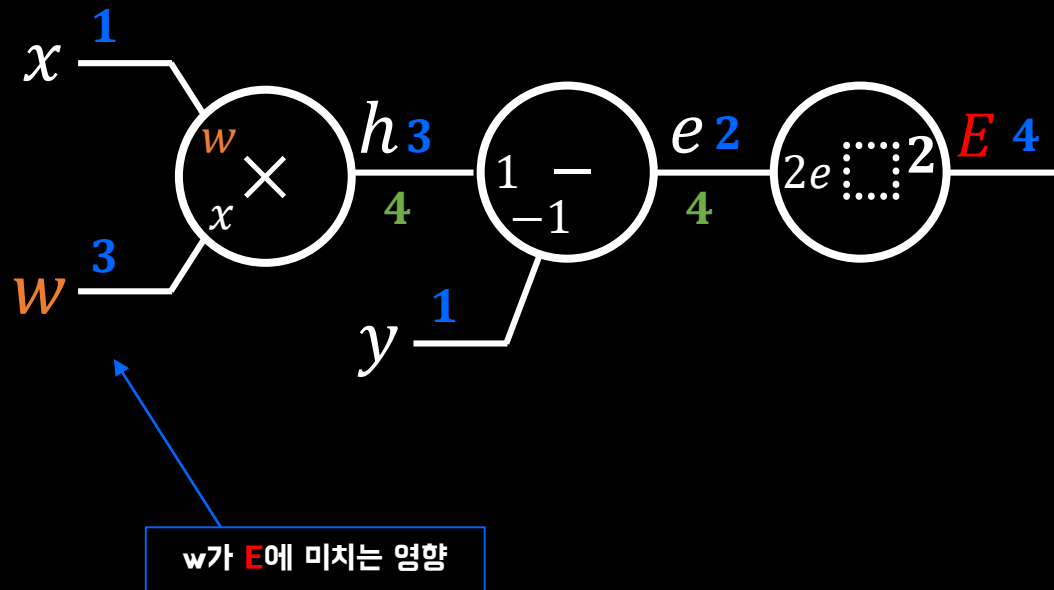


h가 E에 미치는 영향

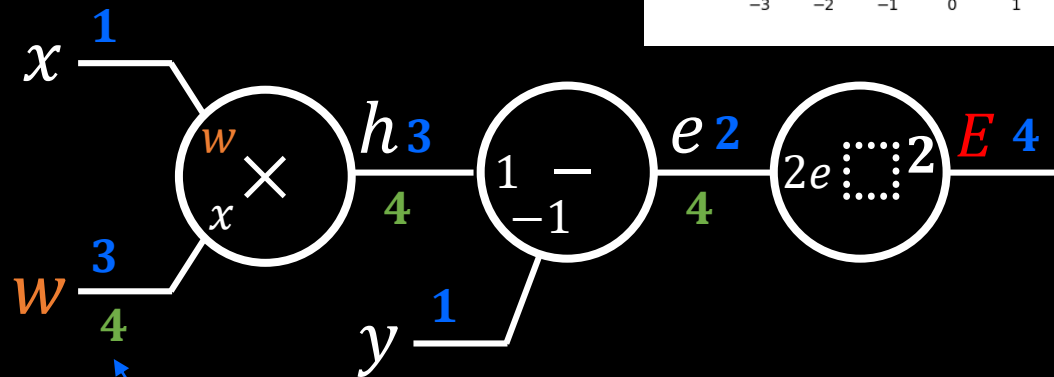
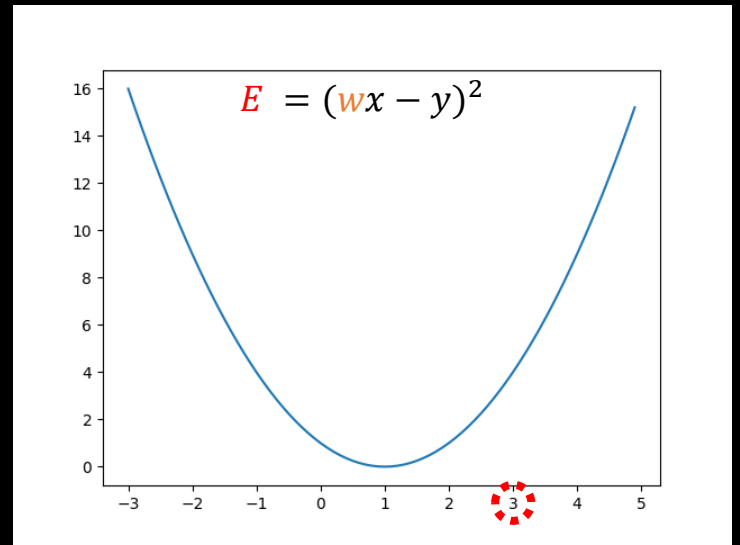
# 뒤로(역) 전파



# 뒤로(역) 전파







w가 E에 미치는 영향

$$W = 3 - 0.1 * 4$$

$$W = 2.6$$



Tuned parameter  
after 1 step learning.

# 미치는 영향을 구하는 방법

현재  $w$  값에서  $w$  변화가 오류  $E$ 에 미치는 영향 구하기

[방법1]  $w$ 가 아주 조금 변할 때  $E$ 는 얼마나 변하는지 계산

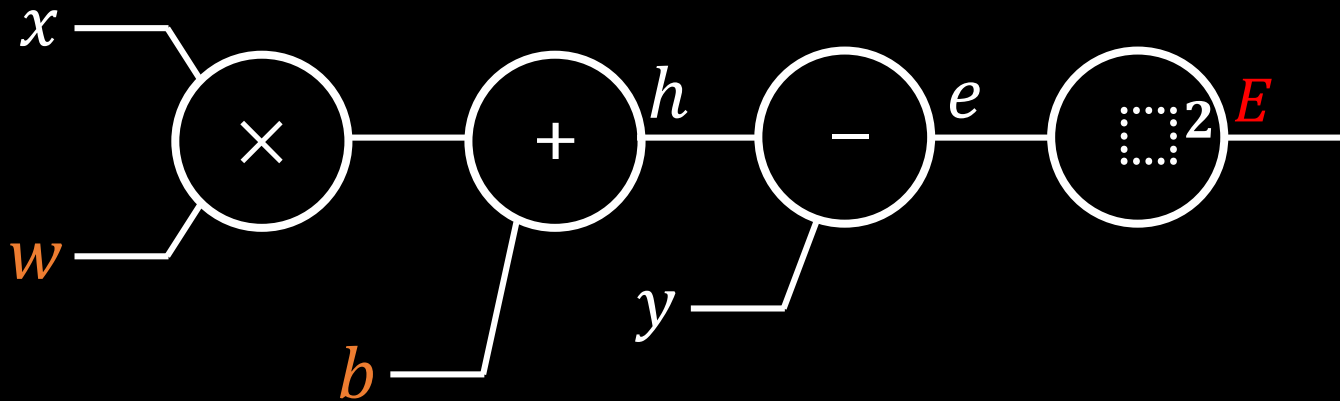
[방법2] 계산 그래프에서 역전파와 체인 룰을 이용한 방법

[방법3] 고등학교 때 배운 방법(?)

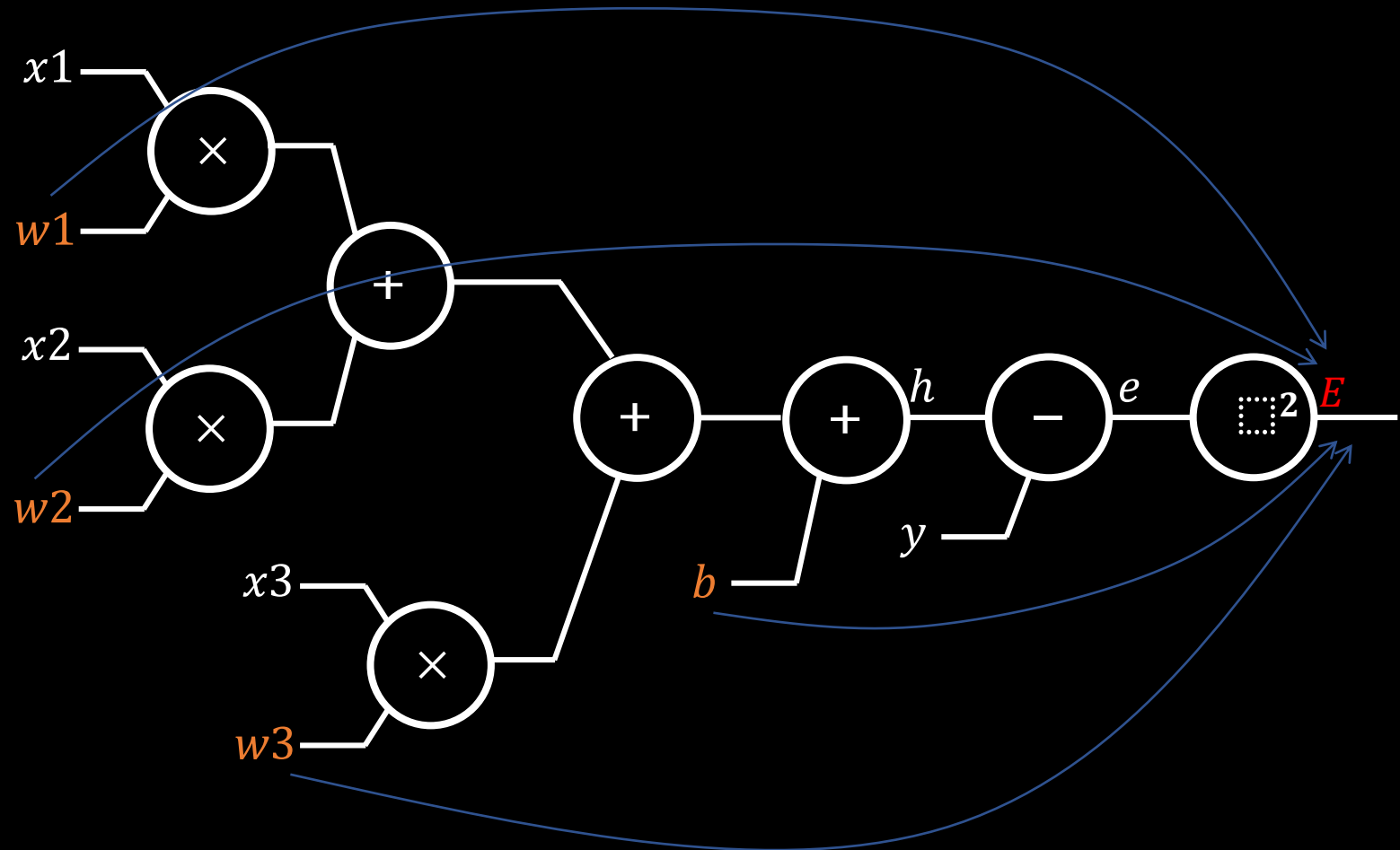
# 계산 그래프 확장

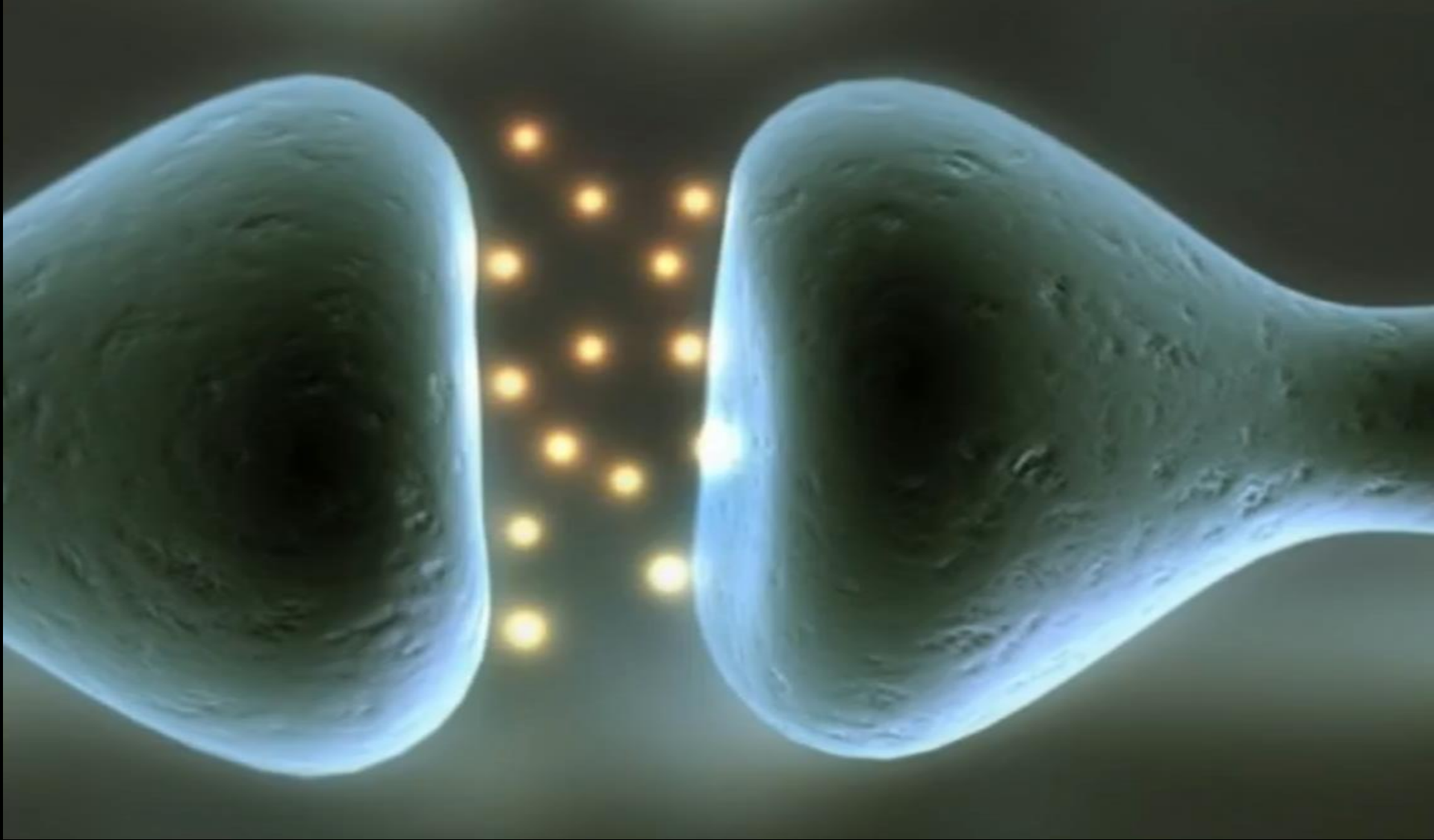
- bias가 있을 경우 (+ 게이트)
- 뉴런 입력이 3개일 때 (+ 게이트)
- 뉴런이 2개일 때
- 튜닝할 파라미터는 모두 몇 개?

$$E = ((wx + b) - y)^2$$



$$E = ((w1x1 + w2x2 + w3x3 + b) - y)^2$$





학습, 더 새롭고 좋은 연결을 만드는 것

# Meaning of cost(error)

- 기울기가 큼  $\rightarrow$  **bad!**  $\rightarrow$  big penalty(아주 힘들다)  $\rightarrow$  big update(w)
- 기울기가 작음  $\rightarrow$  **not bad!**  $\rightarrow$  small penalty(많이 힘들지 않다)  $\rightarrow$  small update(w)
- 기울기 0  $\rightarrow$  **great!**  $\rightarrow$  no penalty  $\rightarrow$  no update(w)  $\rightarrow$  learning ended!



# 우리 마음 속의 cost(error, loss) function

‘좋다’, ‘나쁘다’를 느끼게 하는 기저 그래프

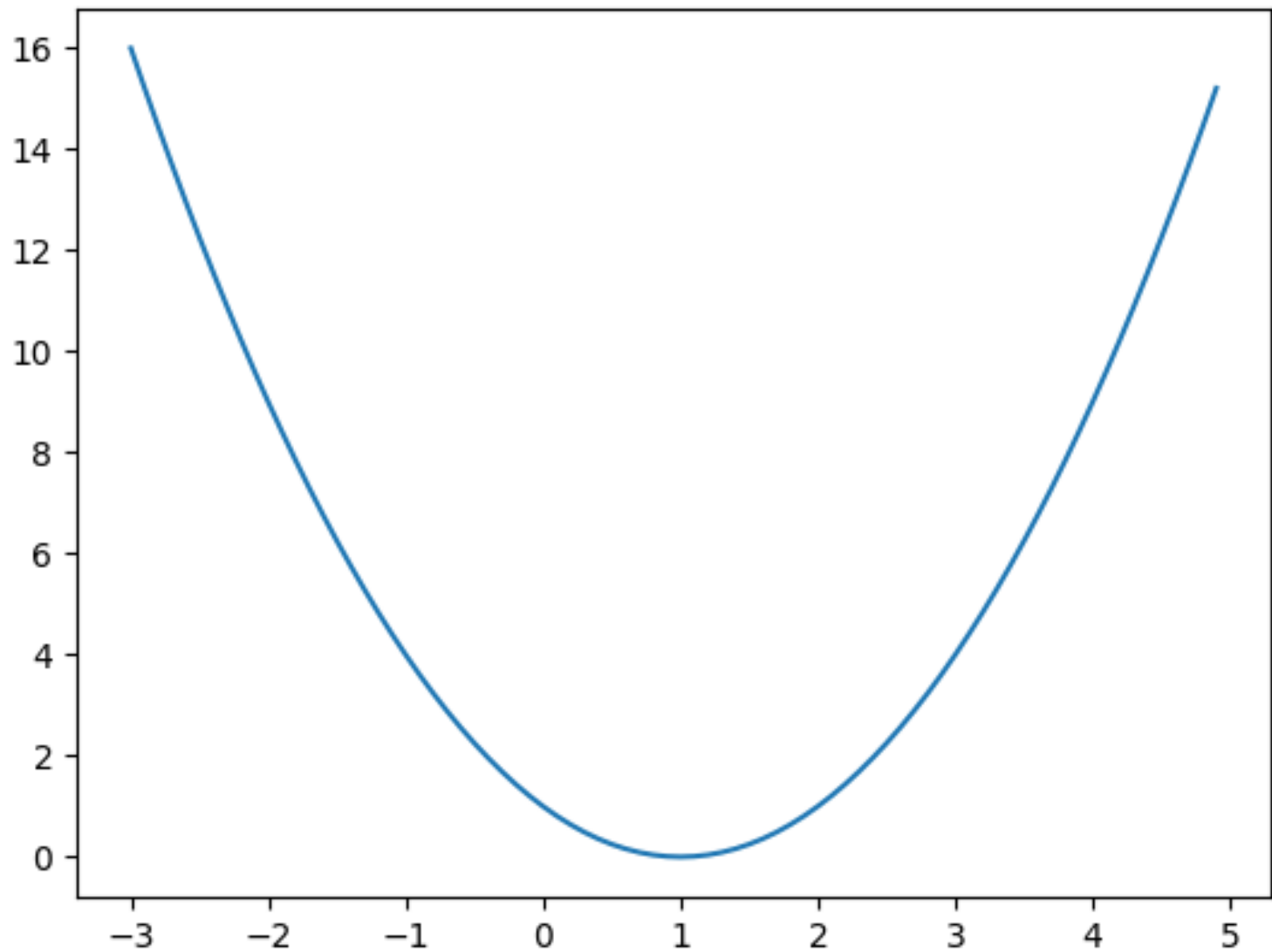
- **좋다.**

- 열심히 공부해서 알게 되니 기분이 좋다.
- 낚시가 너무 재미있다.
- 물건 훔치니 기분이 짜릿하다.

- **나쁘다.**

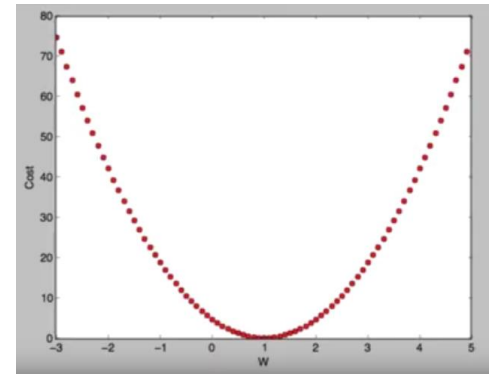
- 많이 틀리니 기분이 나쁘다.
- 과식하니 속이 쓰리다.
- 지렁이를 밟았다. 기분이 별로이다.

## 스트레스

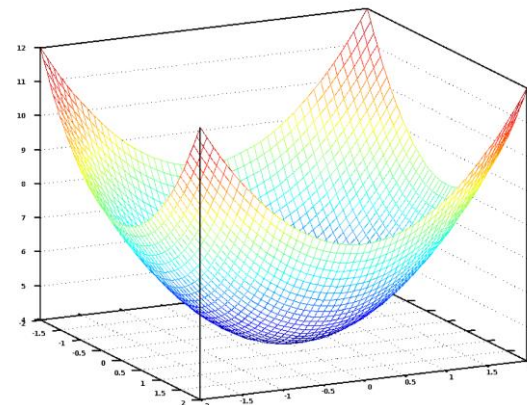
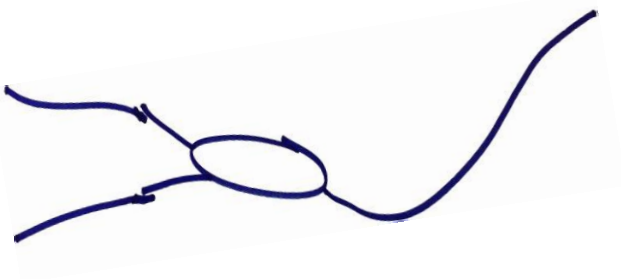


일

# Cost(error) graph



convex function



convex function