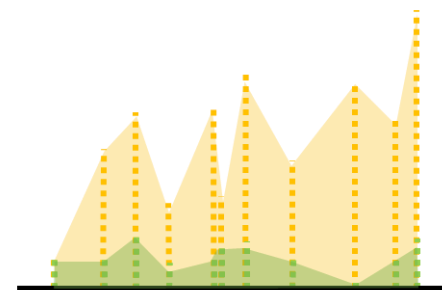
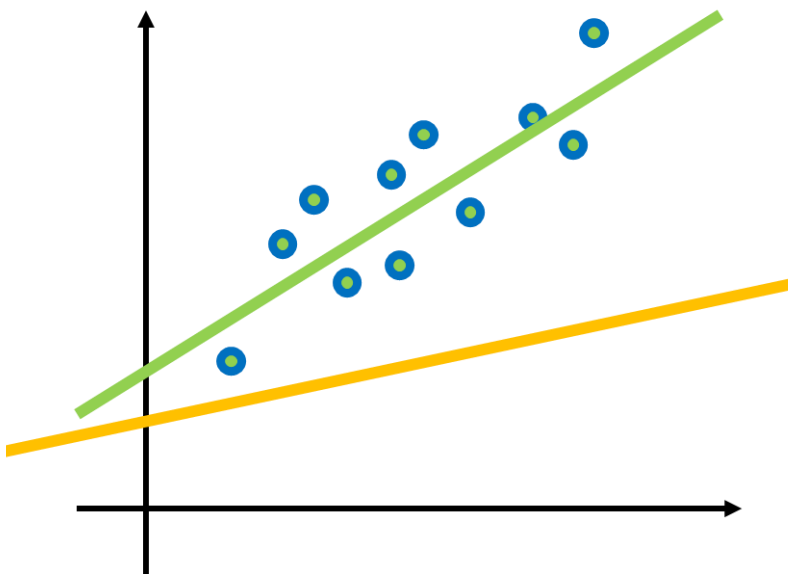
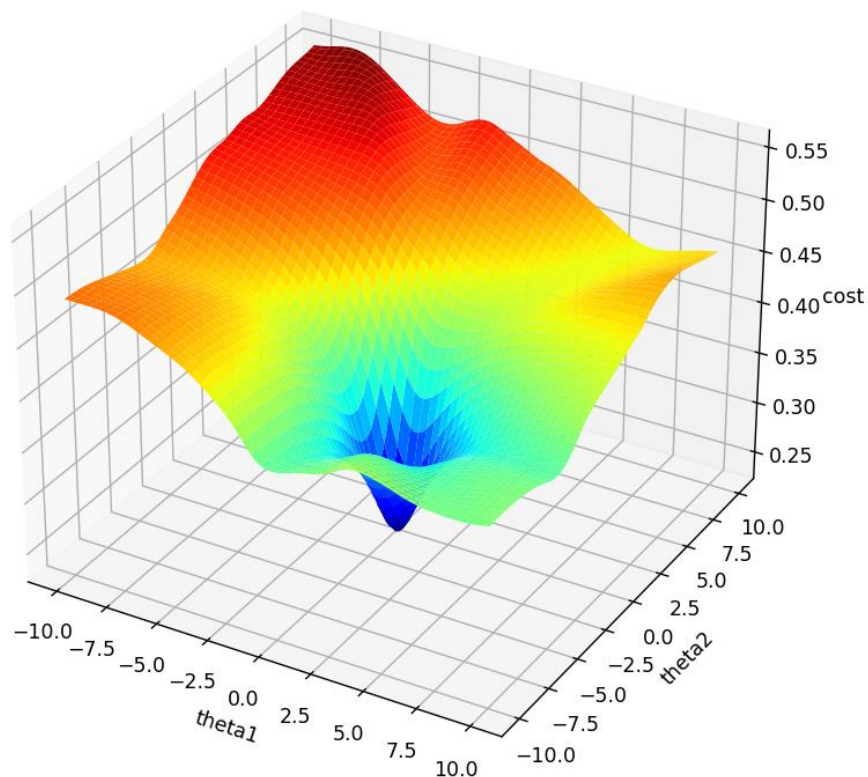


Introduction to 인공지능

경사하강법



안녕하세요 신박AI입니다

오늘은 신경망 학습에 있어서 너무너무 중요한

경사하강법에 대해서 알아보도록 하겠습니다

이 채널은 여러분의 관심과 사랑이 필요합니다

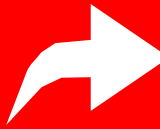
좋아요



댓글



공유



구독



‘좋아요’와 ‘구독’버튼은 강의 준비에 큰 힘이 됩니다!

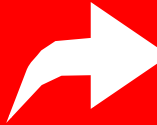
좋아요



댓글



공유



구독



Chapter 1

경사하강법의 정의

경사하강법의 사전적 정의는 다음과 같습니다

경사하강법의 사전적 정의는 다음과 같습니다

경사하강법이란? 주어진 손실함수에서 모델의 파라미터의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다

여기서 모델은 뉴럴 네트워크를 뜻합니다

경사하강법이란? 주어진 손실함수에서 **모델**의 파라미터의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다



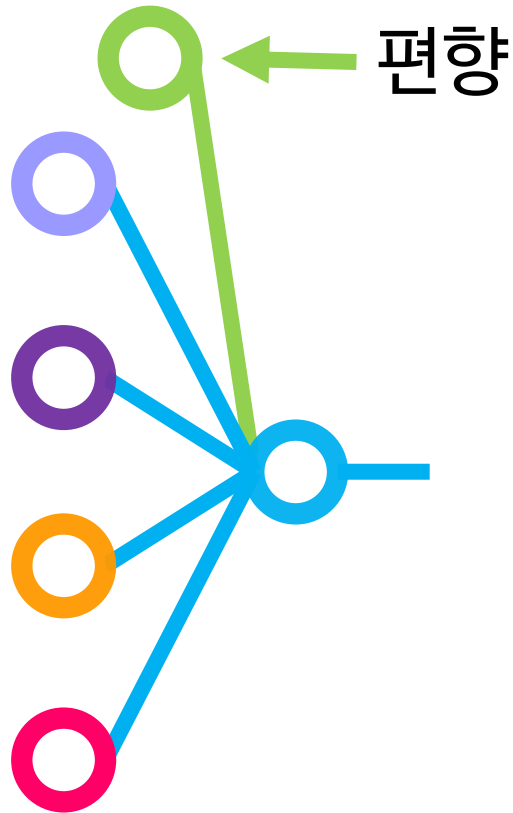
그리고 파라미터 parameter 라는 것은

경사하강법이란? 주어진 손실함수에서 모델의 **파라미터**의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다



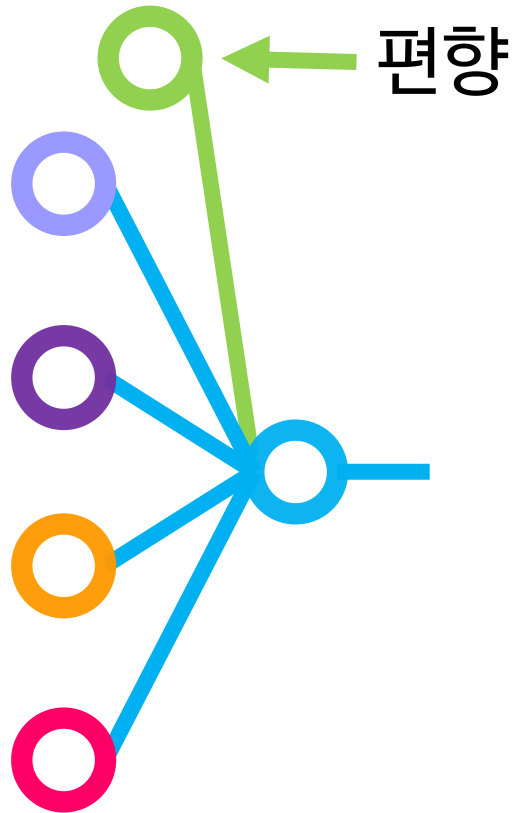
인공신경망을 구성하는 연결가중치나 편향등을 말합니다

경사하강법이란? 주어진 손실함수에서 모델의 **파라미터**의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다



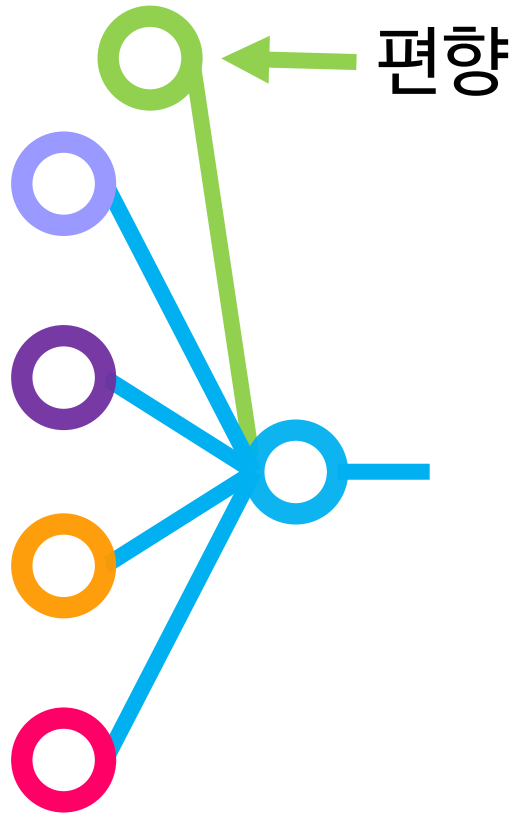
우리는 지난 시간 동안에는 연결가중치의 업데이트에 주안점을 두었기 때문에

경사하강법이란? 주어진 손실함수에서 모델의 **파라미터**의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다



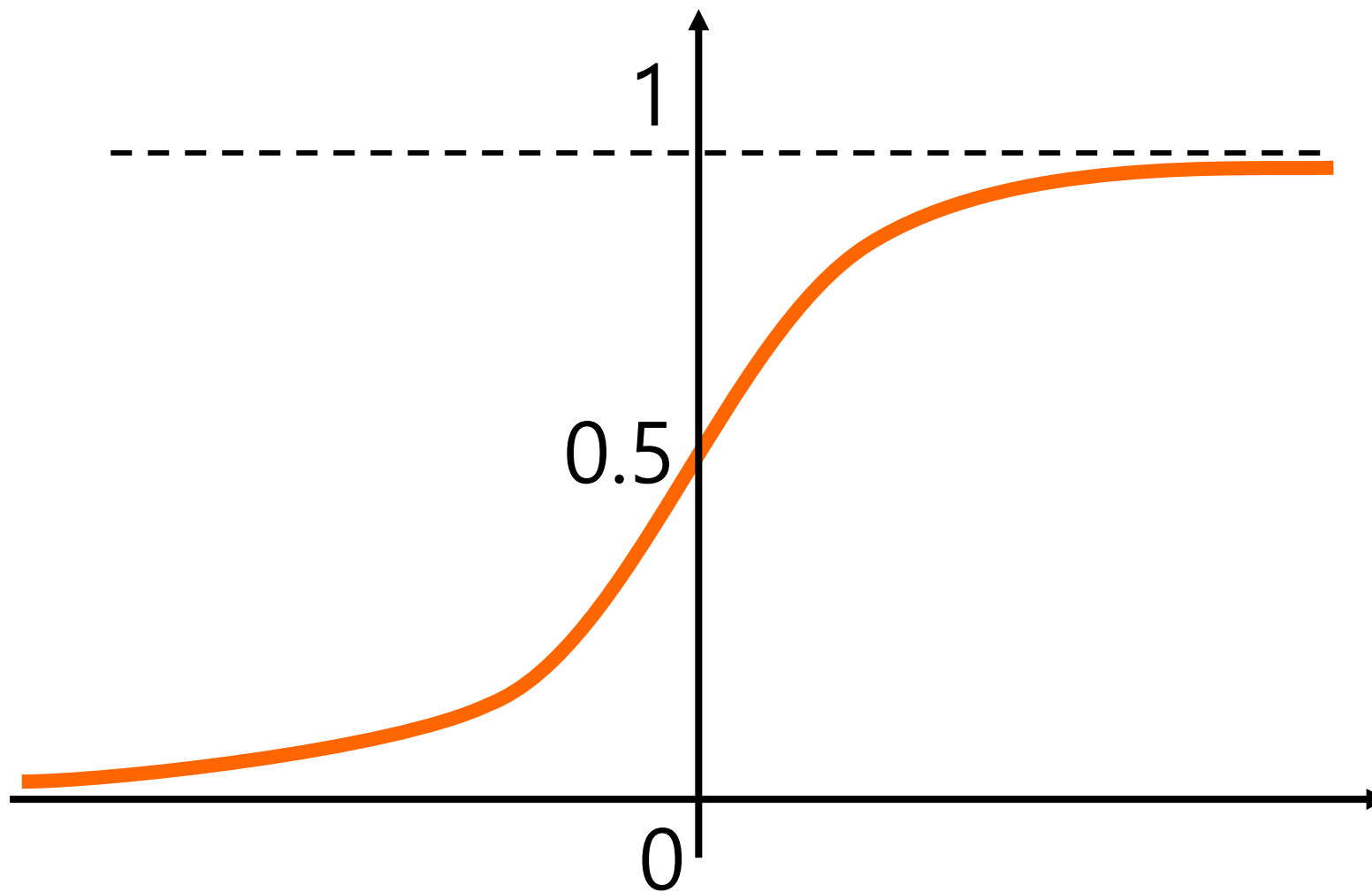
편향이라는 것을 포함시키지는 않았습니다

경사하강법이란? 주어진 손실함수에서 모델의 **파라미터**의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다

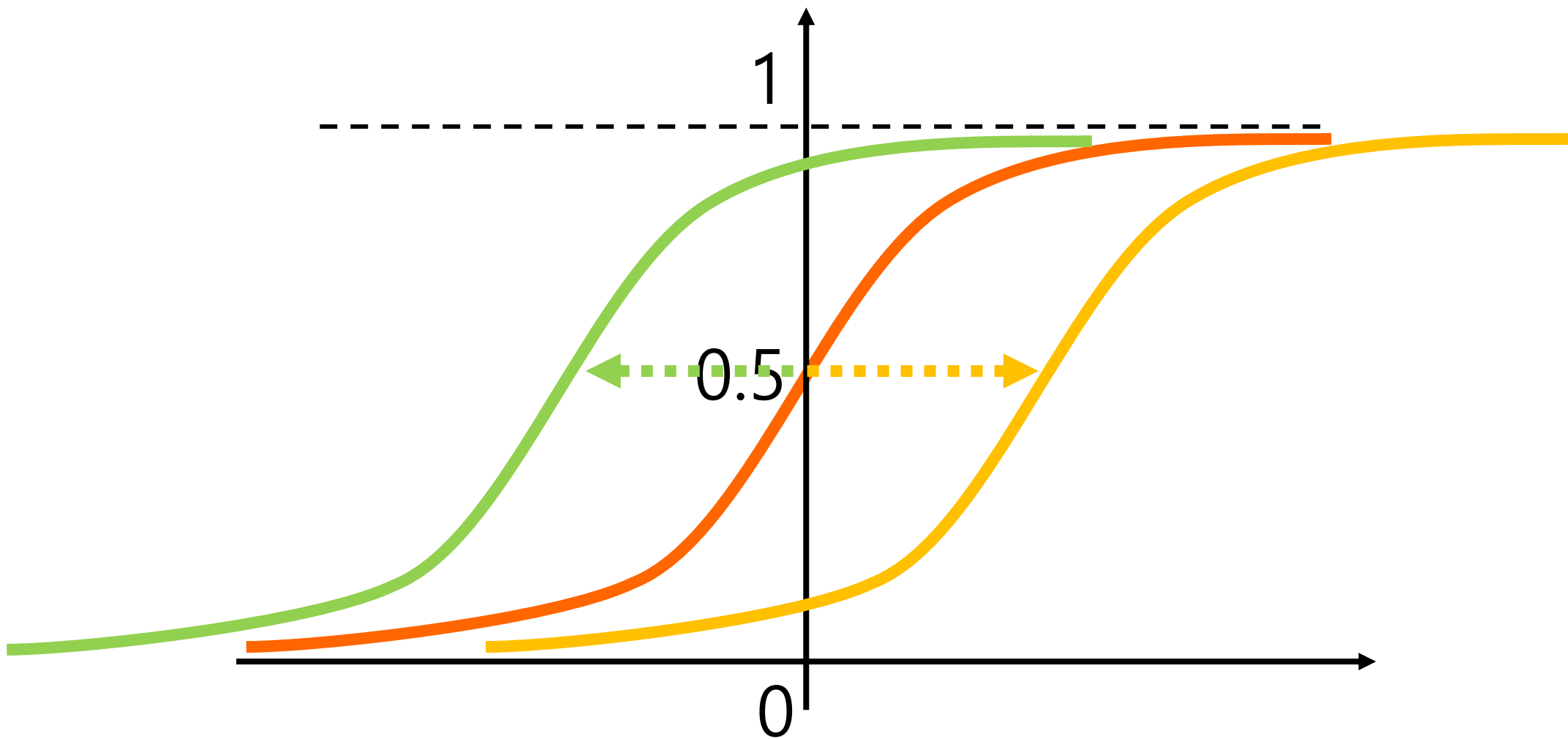


그러나 편향도 인공신경망에 중요한 파라미터입니다

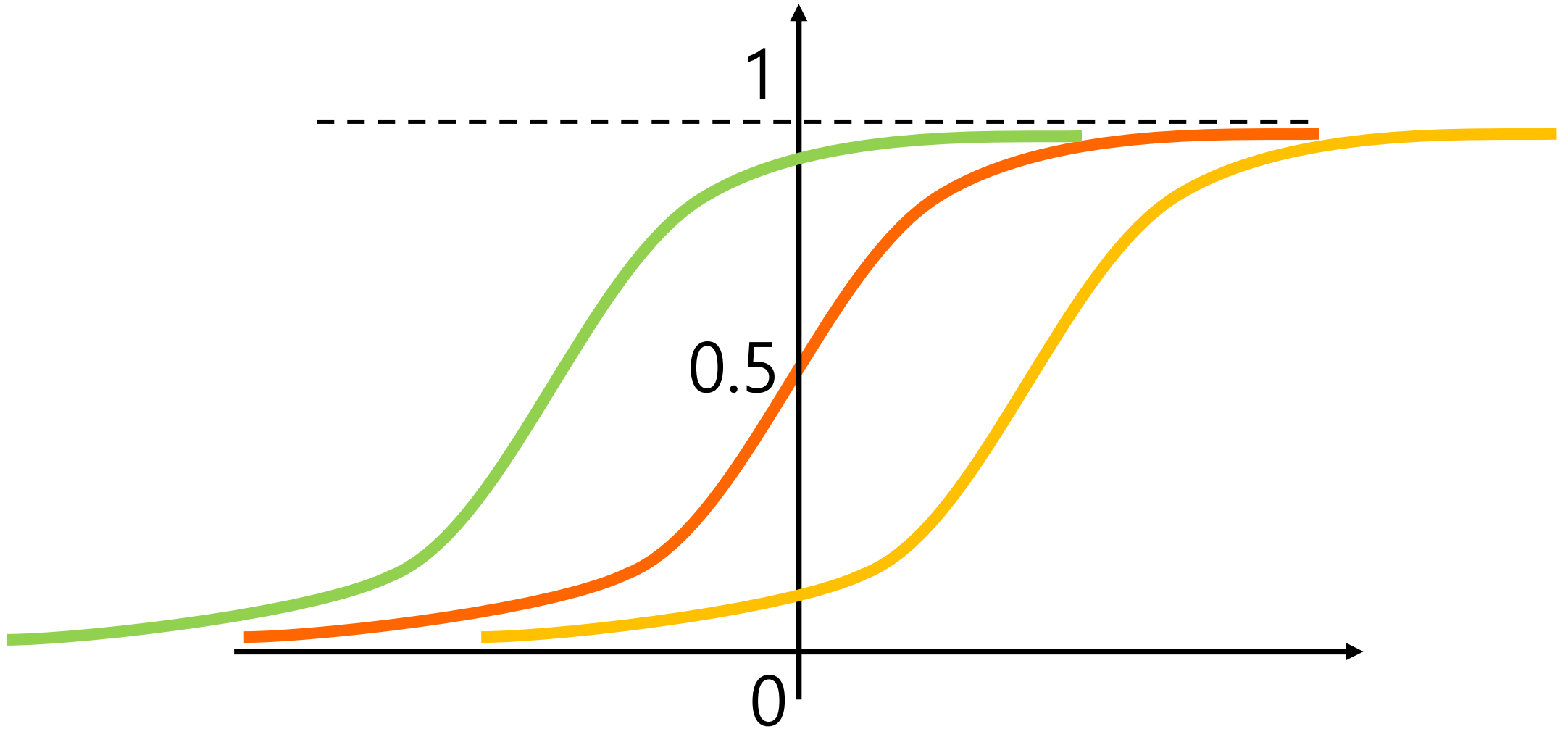
편향은 활성화 함수를 왼쪽 혹은 오른쪽으로 옮김으로서



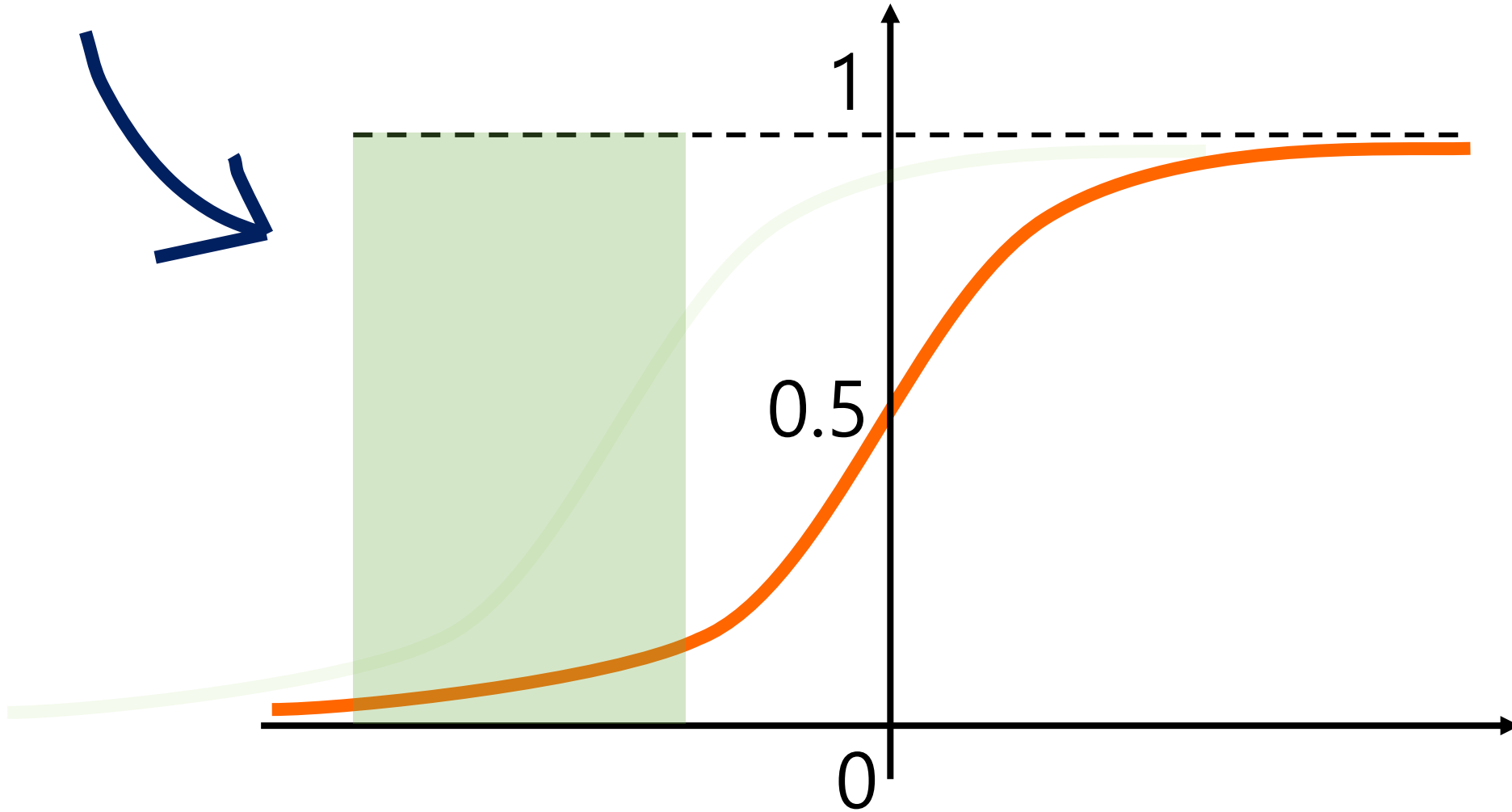
편향은 활성화 함수를 왼쪽 혹은 오른쪽으로 옮김으로서



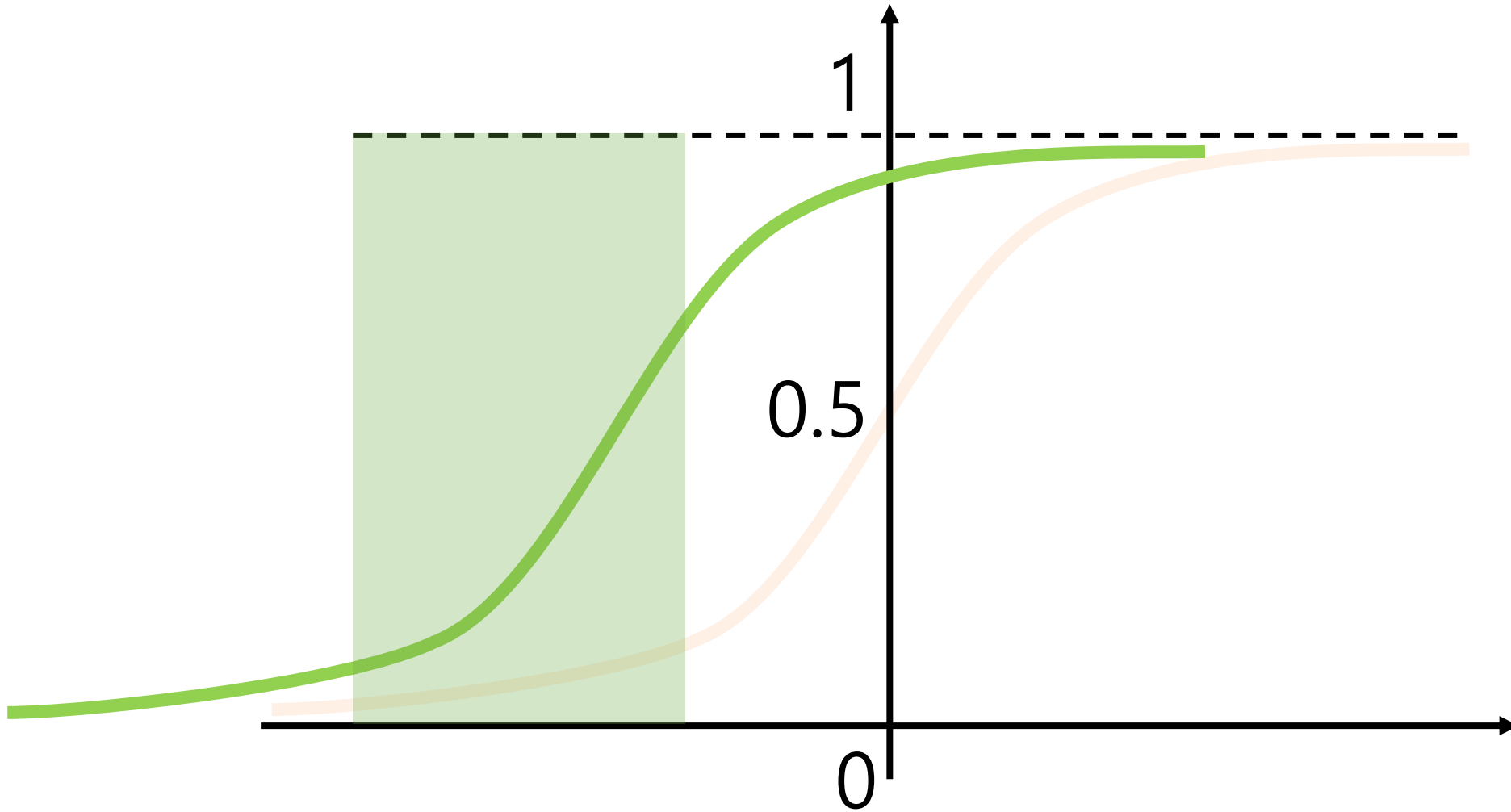
신경망 모델의 학습이 더 빨리 진행되도록 도와줍니다



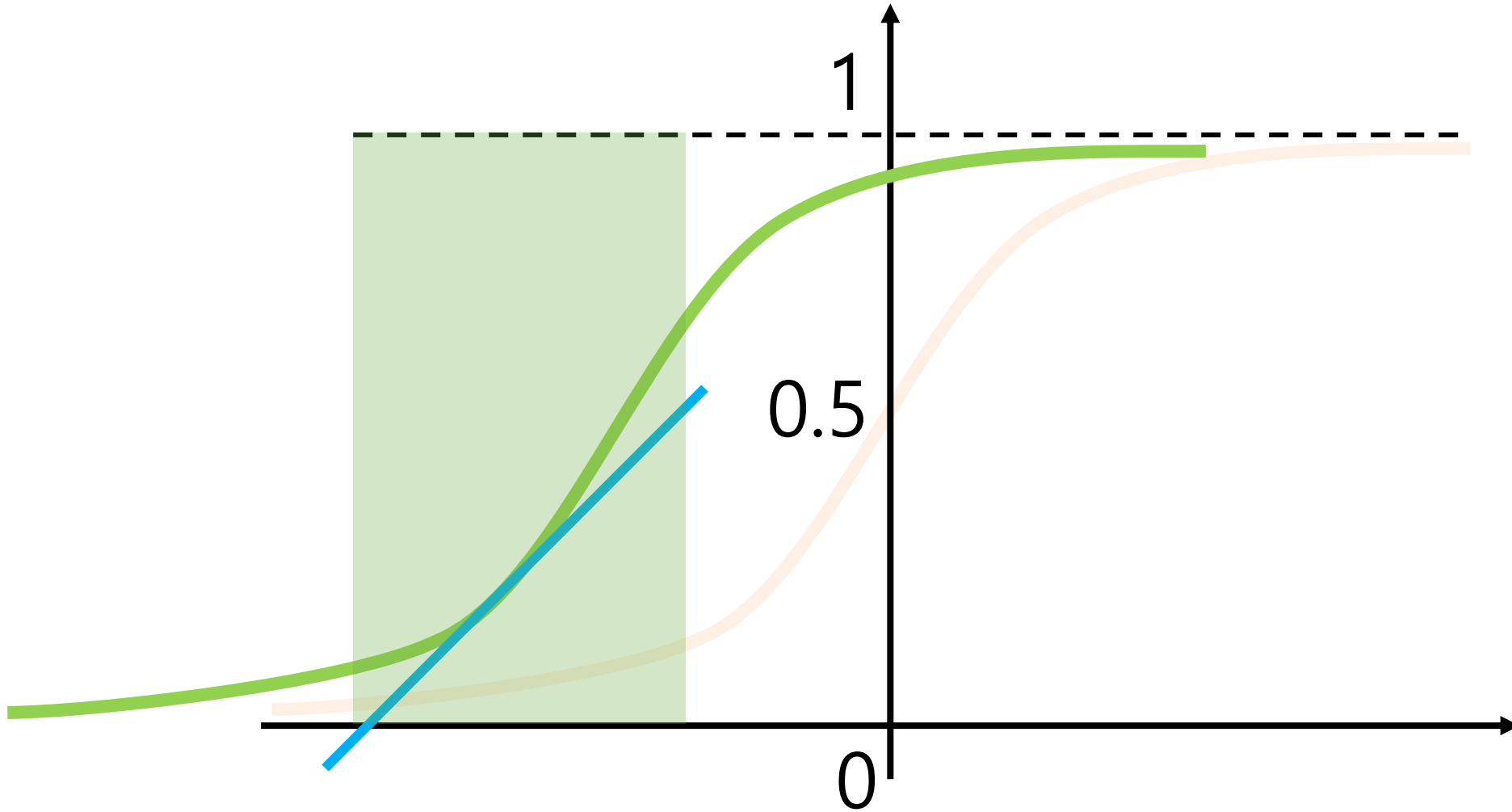
예를들자면, 어떤 이유에서든, 활성화 함수에 들어가는 입력값이 대부분
이쪽 언저리에 있다면



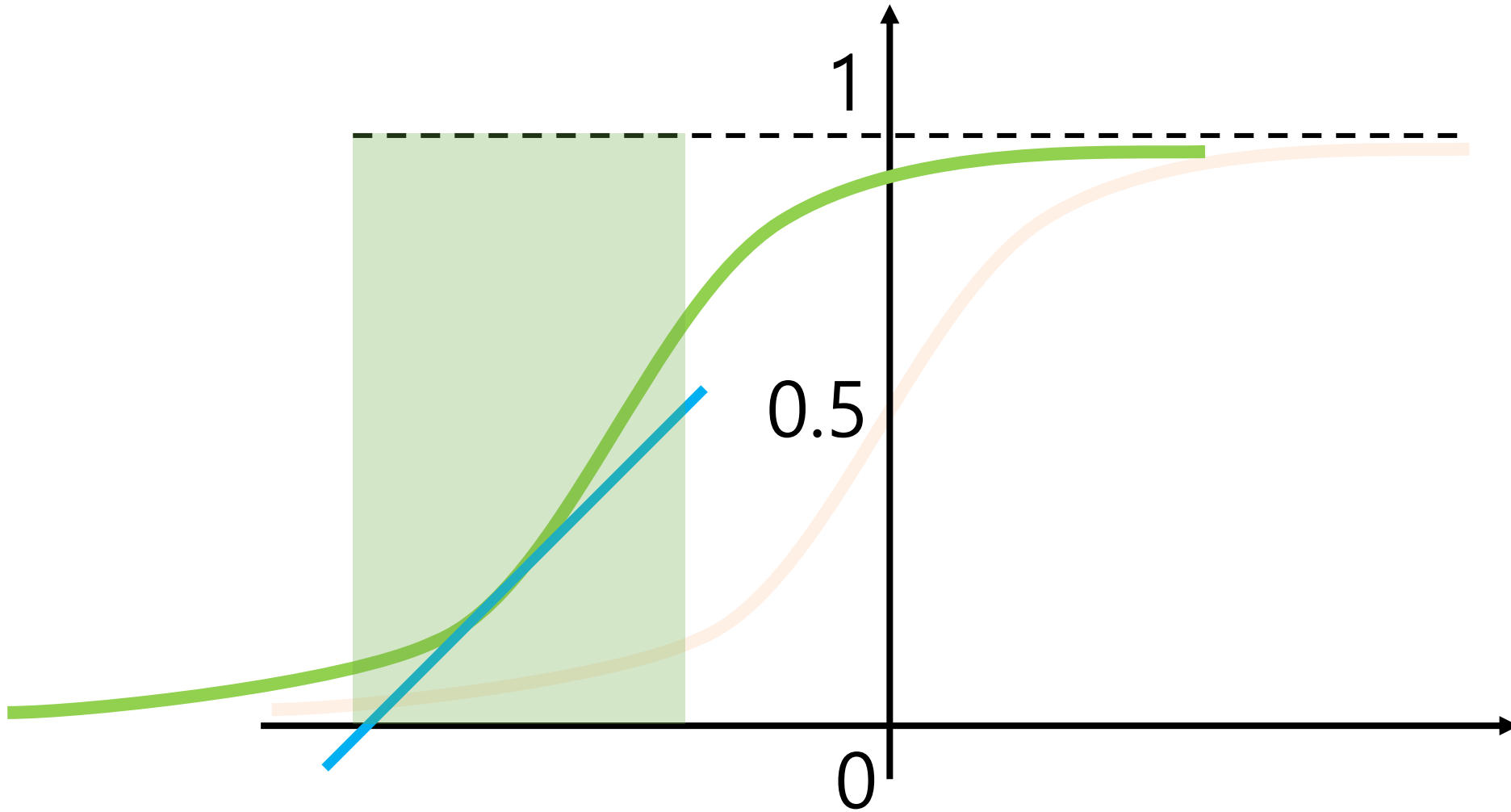
시그모이드 함수를 왼쪽으로 옮기면 학습이 훨씬 용이해집니다



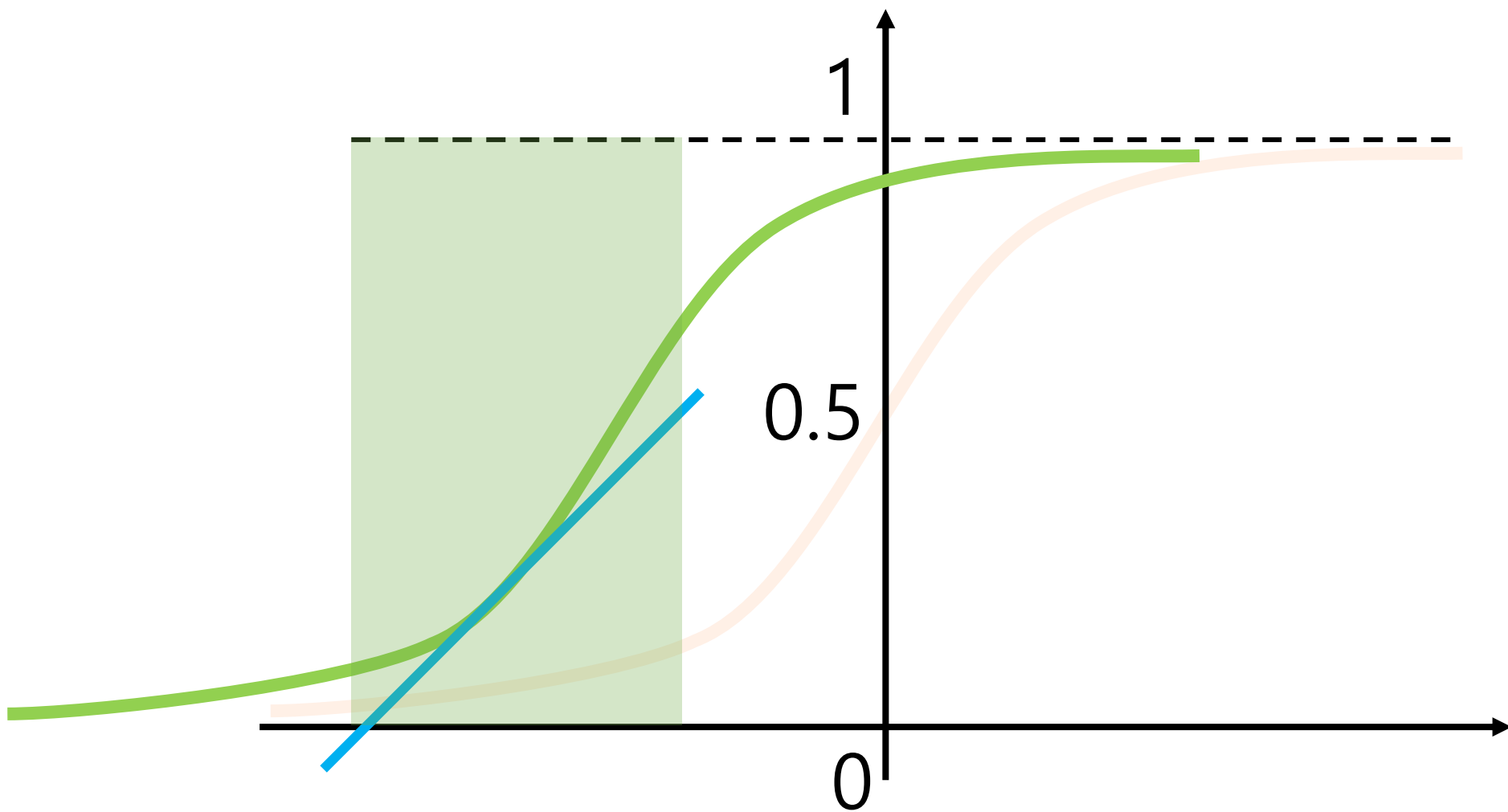
왜냐하면, 우리가 지난 영상에서 본 바와 같이 전반적으로 기울기가 급해지기 때문입니다



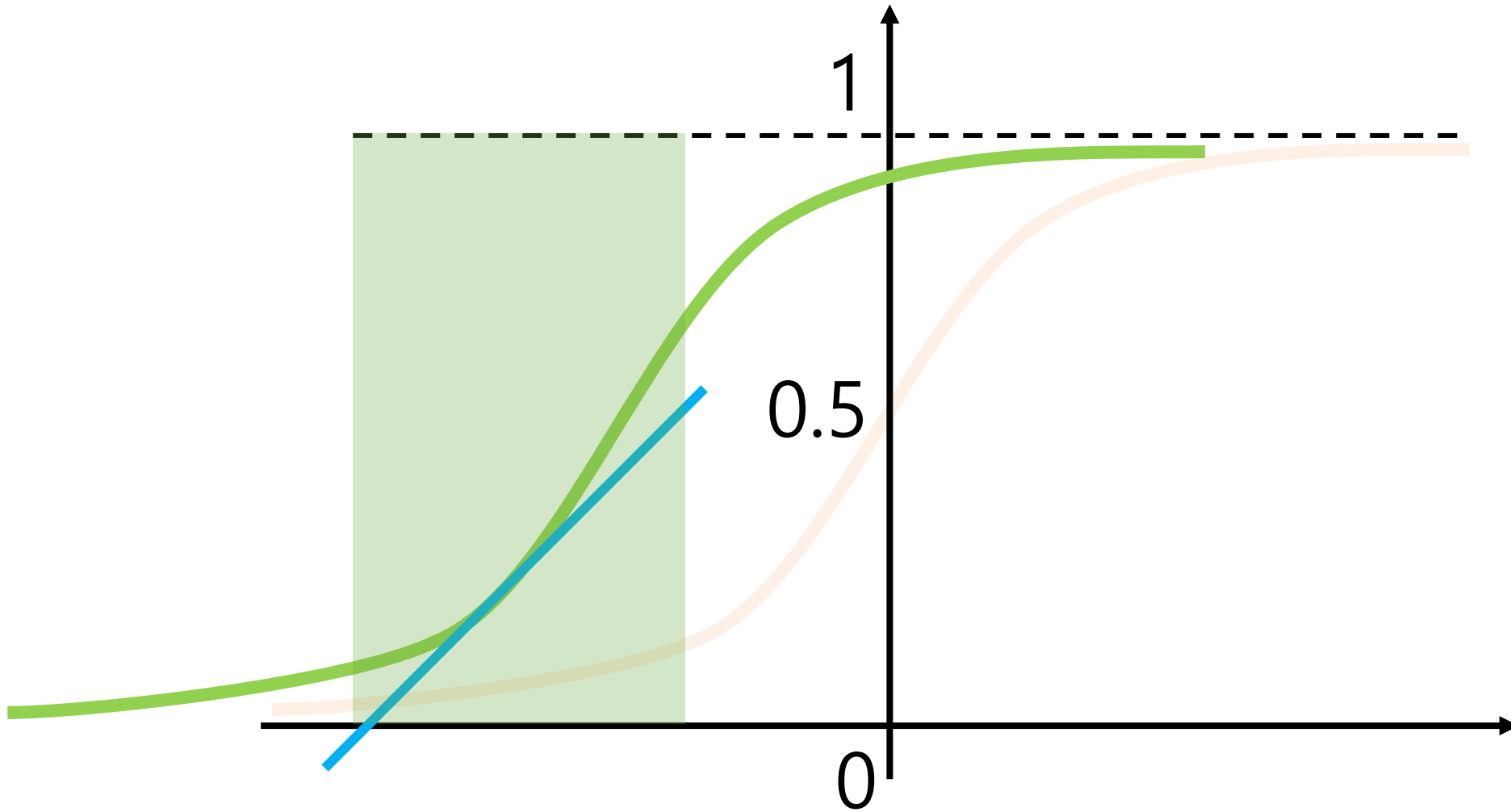
그래서 편향은 신경망 모델의 학습이 더 빨리 진행되는데 도움이 됩니다



그래서 경사하강법이 신경망 모델을 최적화 하기 위한 알고리즘인 만큼



이제부터는 신경망 모델의 편향값도 같이 넣어서 배워보도록 하겠습니다



다시 본론으로 돌아가겠습니다

다시 본론으로 돌아가겠습니다

경사하강법이란? 주어진 손실함수에서 모델의 파라미터의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다

여기 보시면 새로운 개념이 나오는데 바로 손실함수 입니다

경사하강법이란? 주어진 **손실함수**에서 모델의 파라미터의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다

손실함수는 신경망의 학습에서 아주아주 중요한 개념중의 하나입니다

경사하강법이란? 주어진 **손실함수**에서 모델의 파라미터의 최적의 값을 찾는 머신러닝과 딥러닝의 최적화 알고리즘중 하나이다

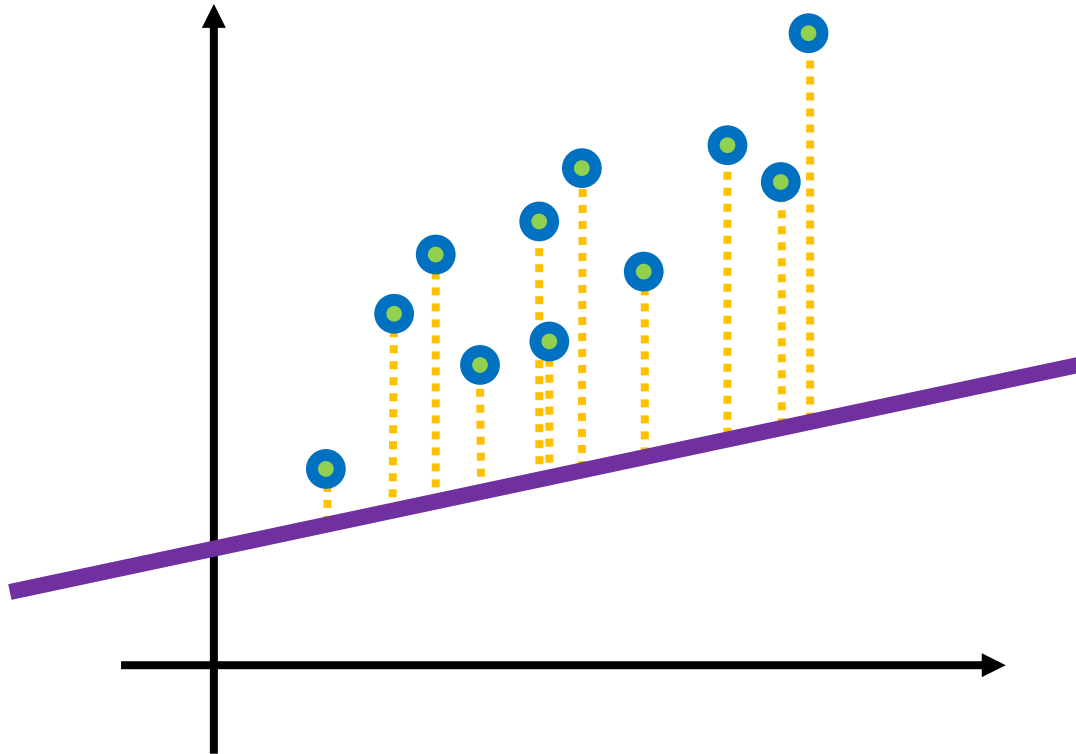
손실함수의 개념은 간단합니다

손실함수란?

신경망 모델의 예측 값과 실제 값 간의 차이를 측정하는
함수입니다

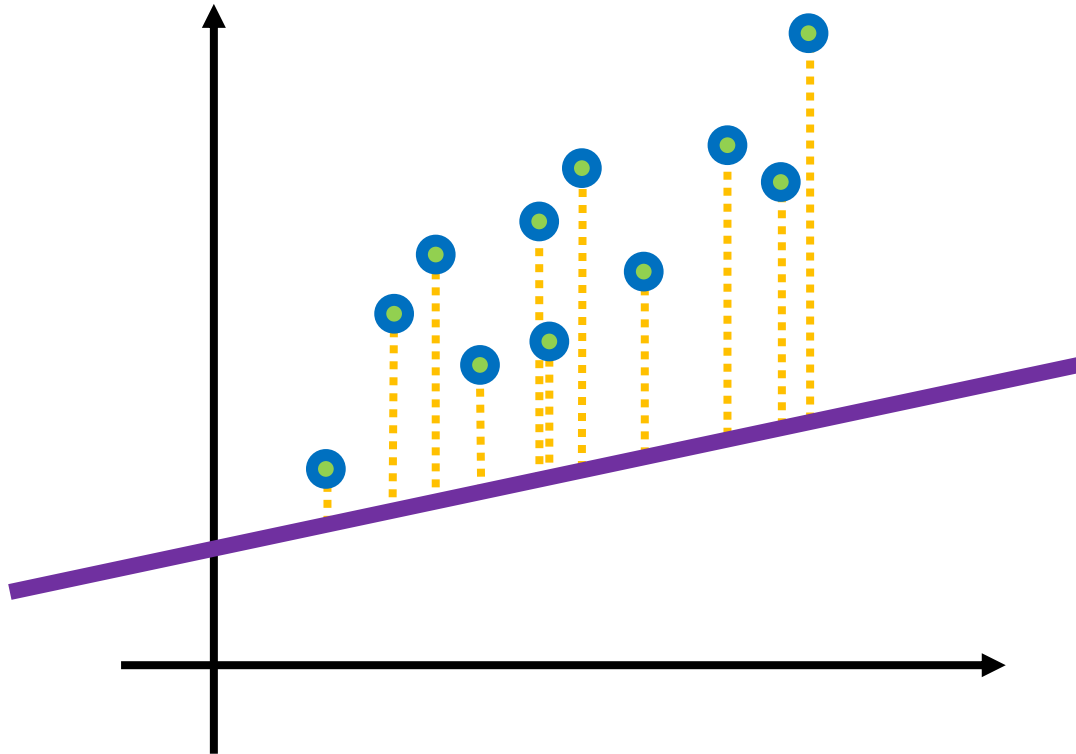
우리가 지난 영상에서 배웠던 오차와 비슷한 개념입니다

신경망 모델의 예측 값과 실제 값 간의 차이를 측정하는 함수입니다



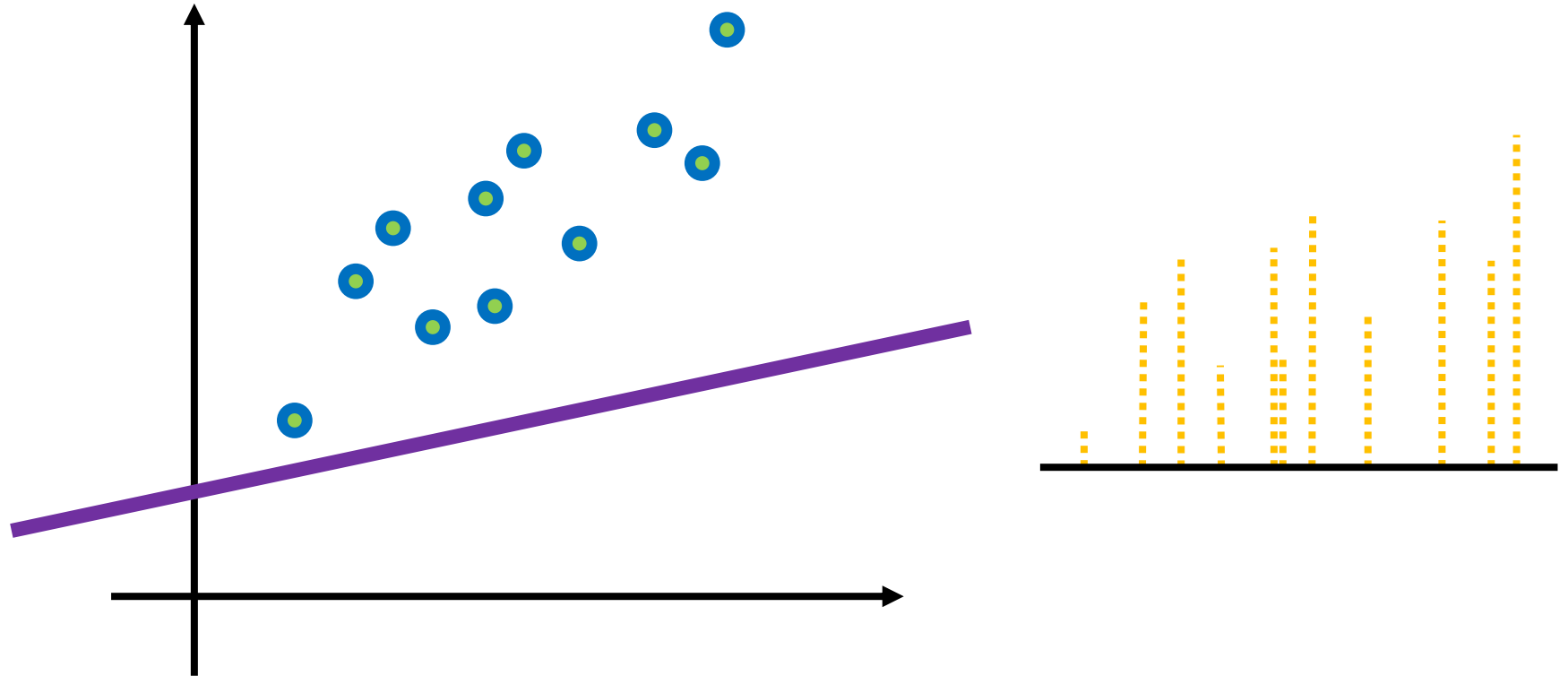
우리가 지난 영상에서 배웠던 오차와 비슷한 개념입니다

이와 같이 모델과 예측값간의 차이가 발생할 경우



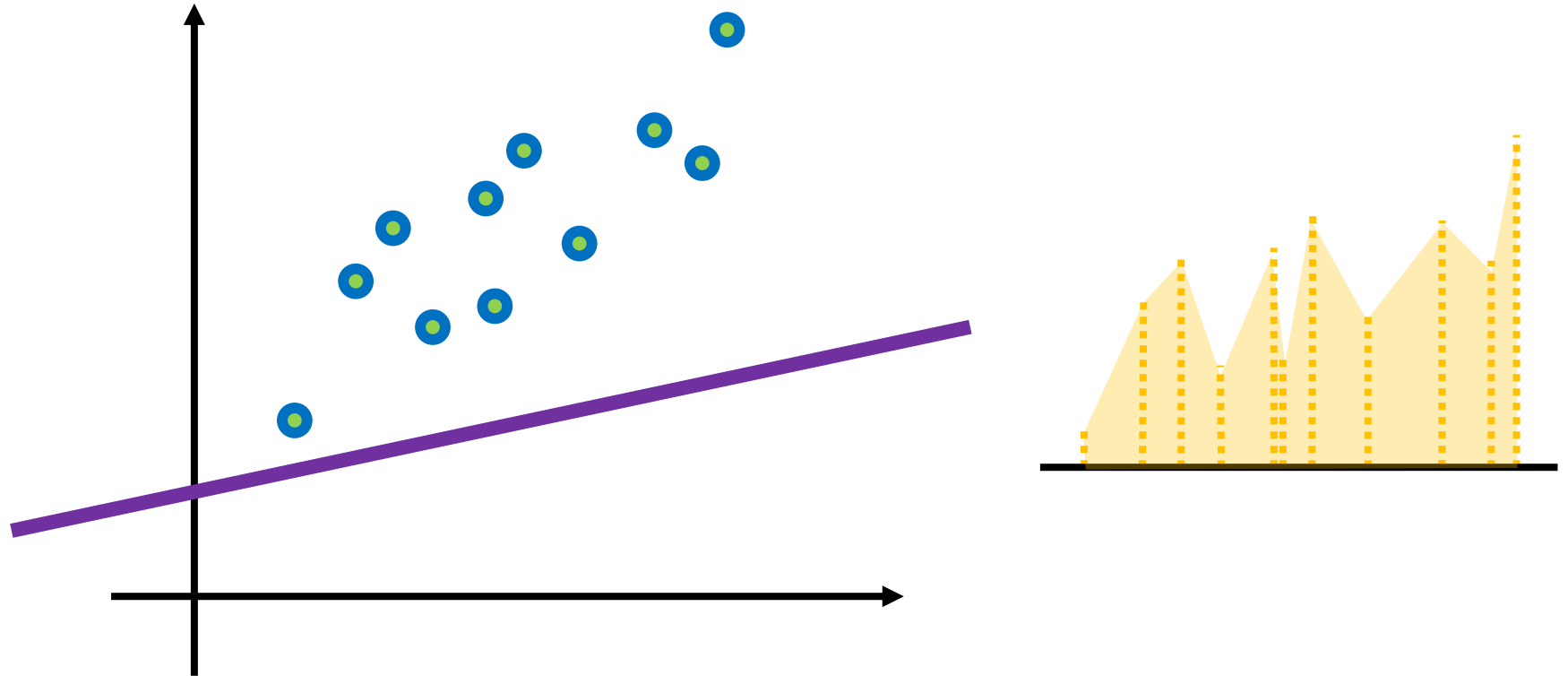
우리가 지난 영상에서 배웠던 오차와 비슷한 개념입니다

오차의 합을 다음과 같이 면적으로 표현해볼 수 있습니다

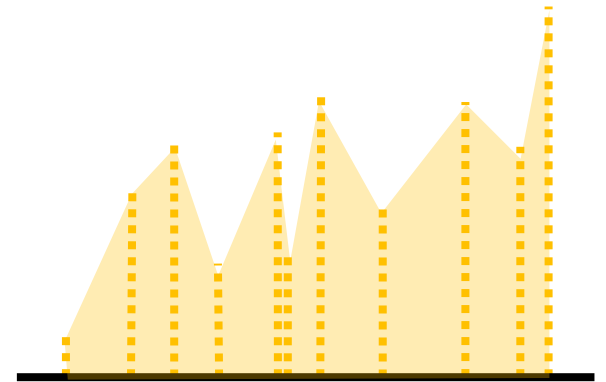
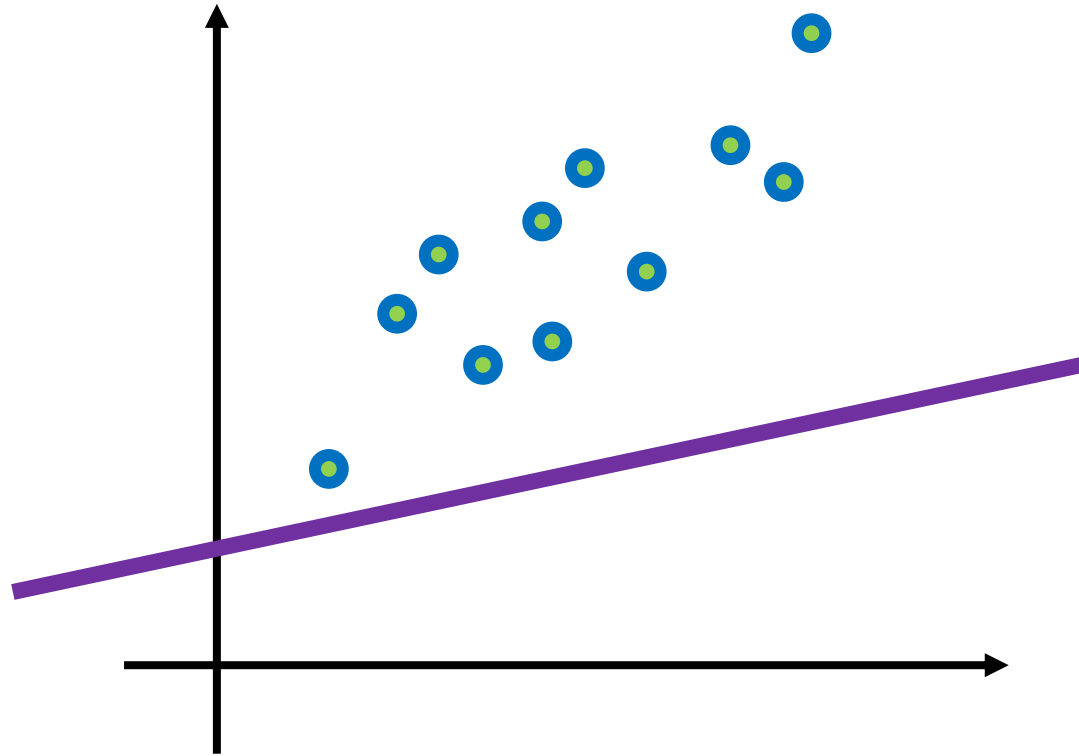


우리가 지난 영상에서 배웠던 오차와 비슷한 개념입니다

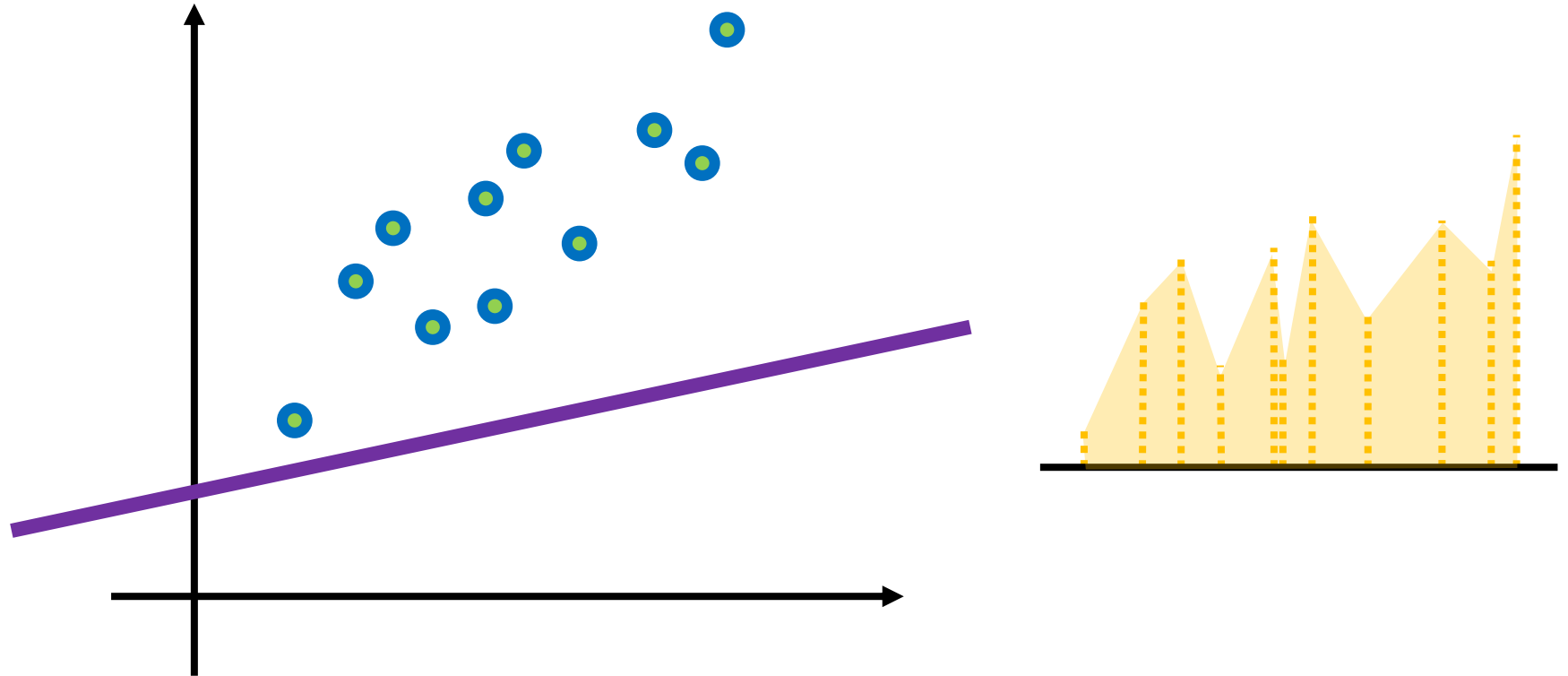
오차의 합을 다음과 같이 면적으로 표현해볼 수 있습니다



신경망의 학습이라는 것은 결국, 오차를 줄여나가도록

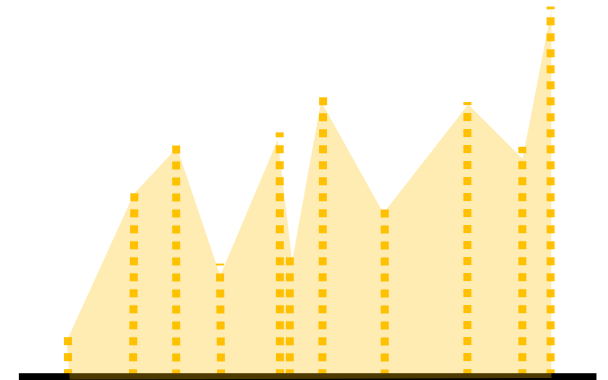
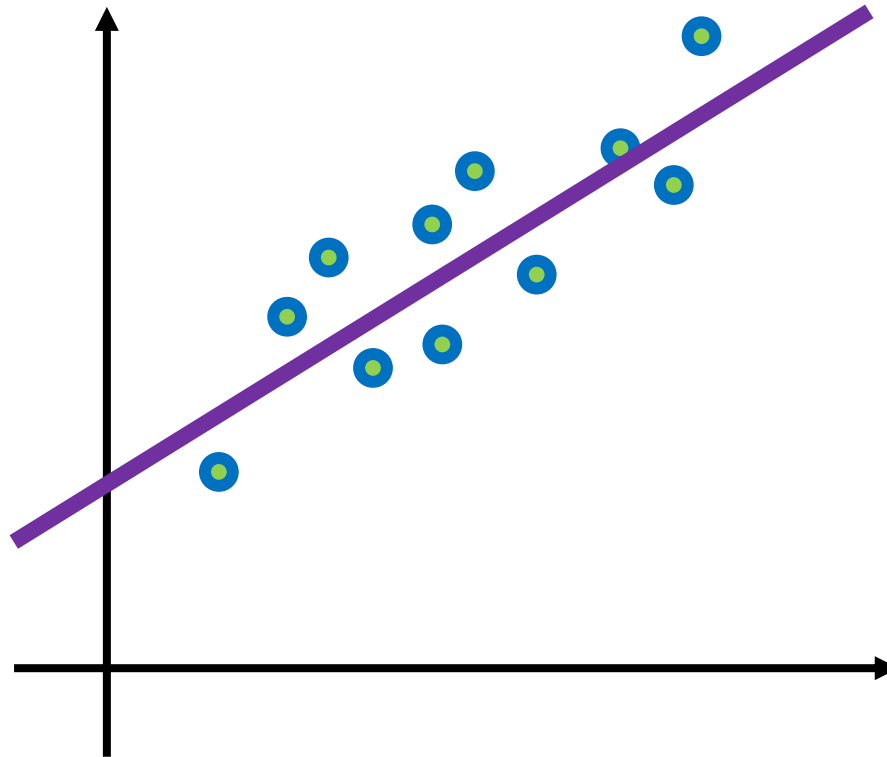


모든 파라미터들을 반복적인 방법으로 조금씩 튜닝해가는 것입니다



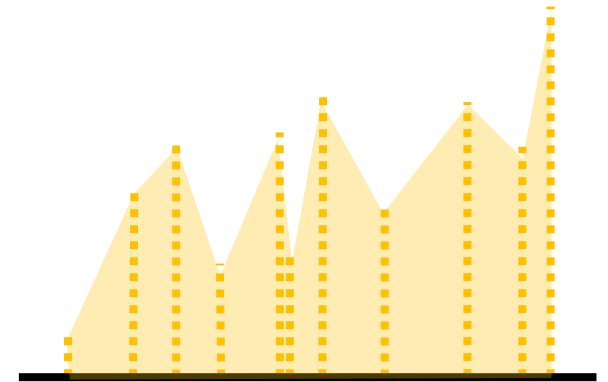
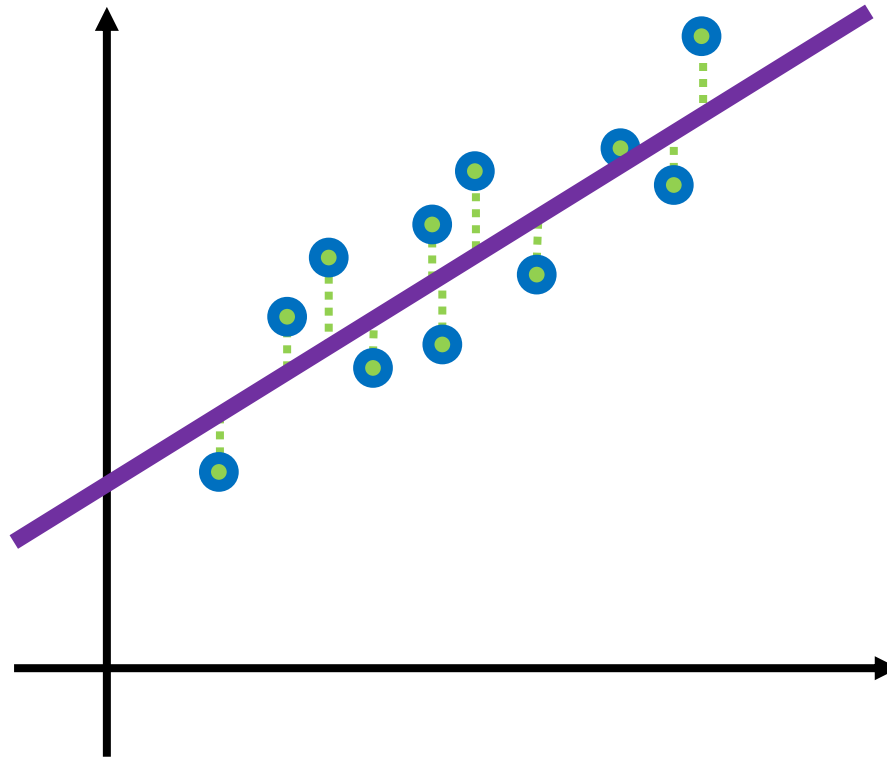
모든 파라미터들을 반복적인 방법으로 조금씩 튜닝해가는 것입니다

그래서 모델이 다음과 같이 예측했다면,



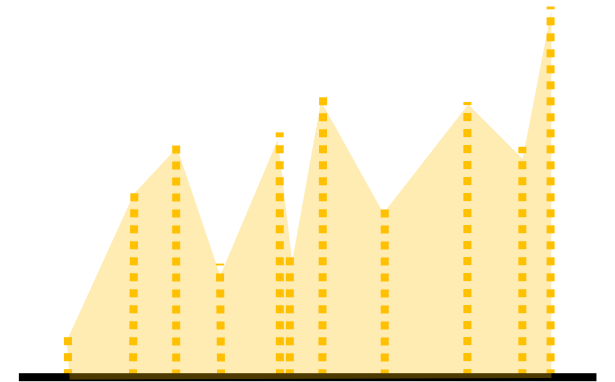
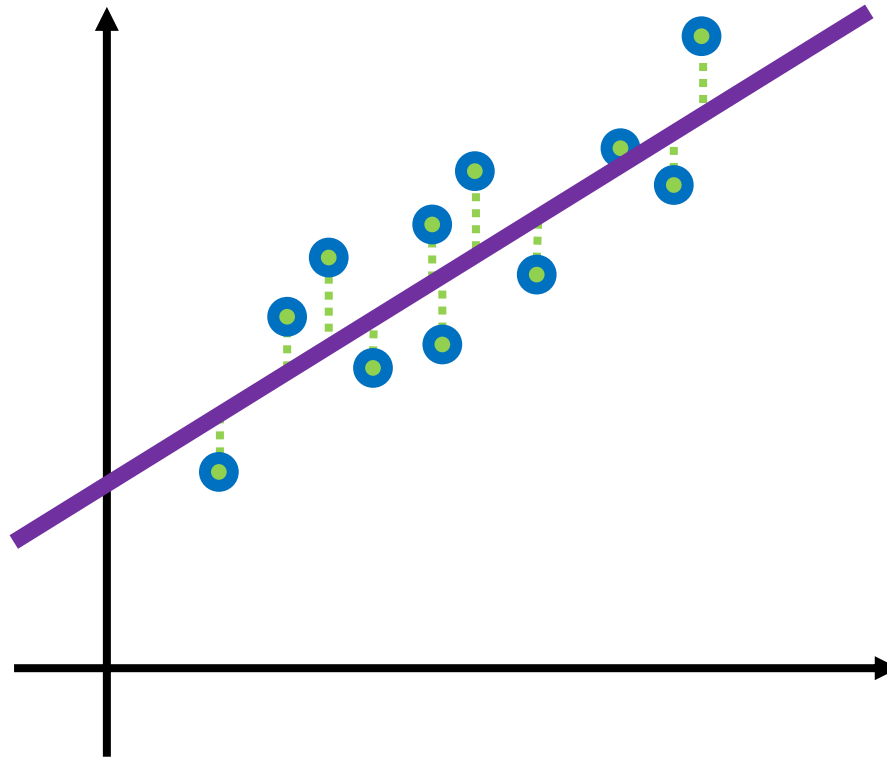
모든 파라미터들을 반복적인 방법으로 조금씩 튜닝해가는 것입니다

오차의 합은 다음과 같이 업데이트 될 수 있습니다



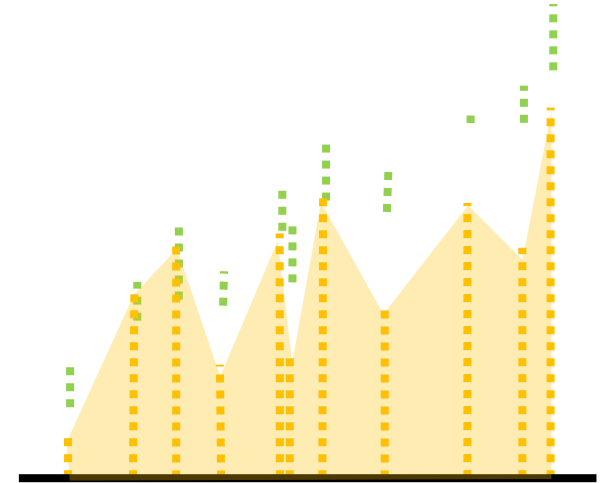
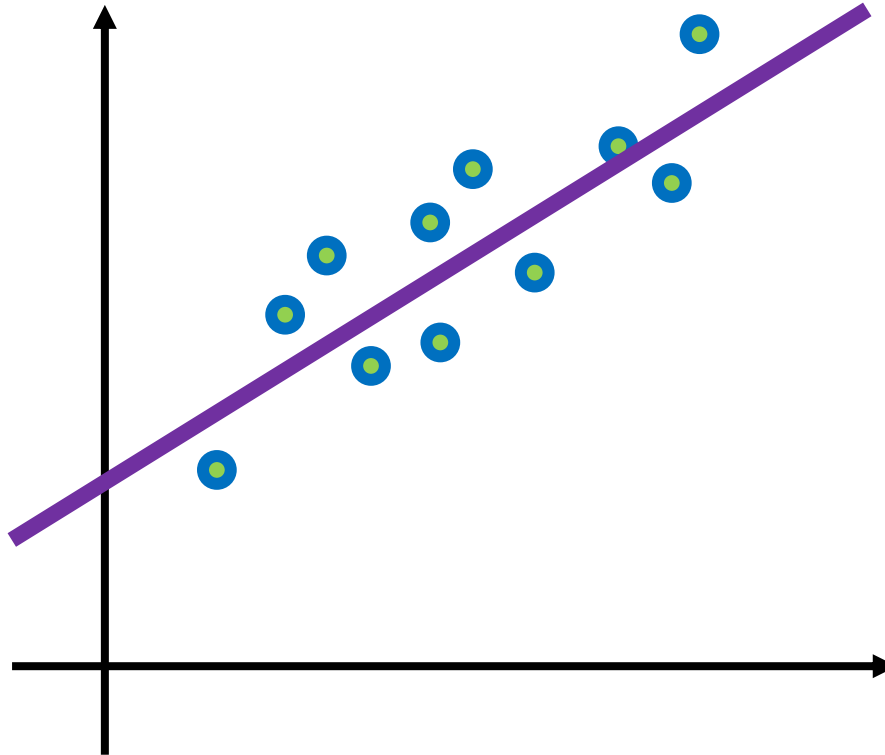
모든 파라미터들을 반복적인 방법으로 조금씩 튜닝해가는 것입니다

오차의 합은 다음과 같이 업데이트 될 수 있습니다



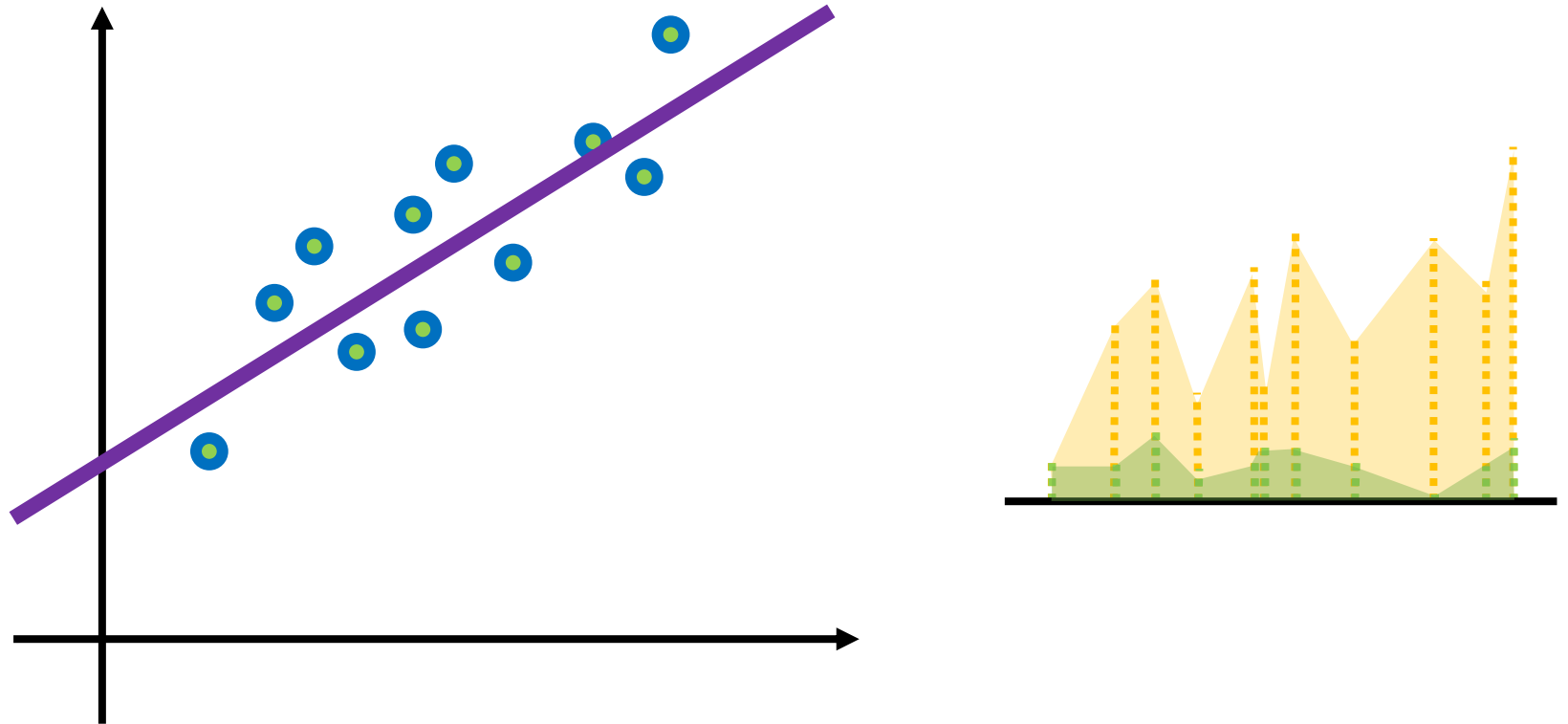
모든 파라미터들을 반복적인 방법으로 조금씩 튜닝해가는 것입니다

마찬가지로 면적으로 표현해 본다면,

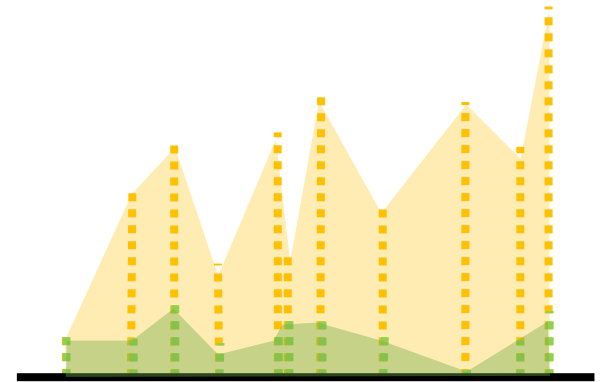
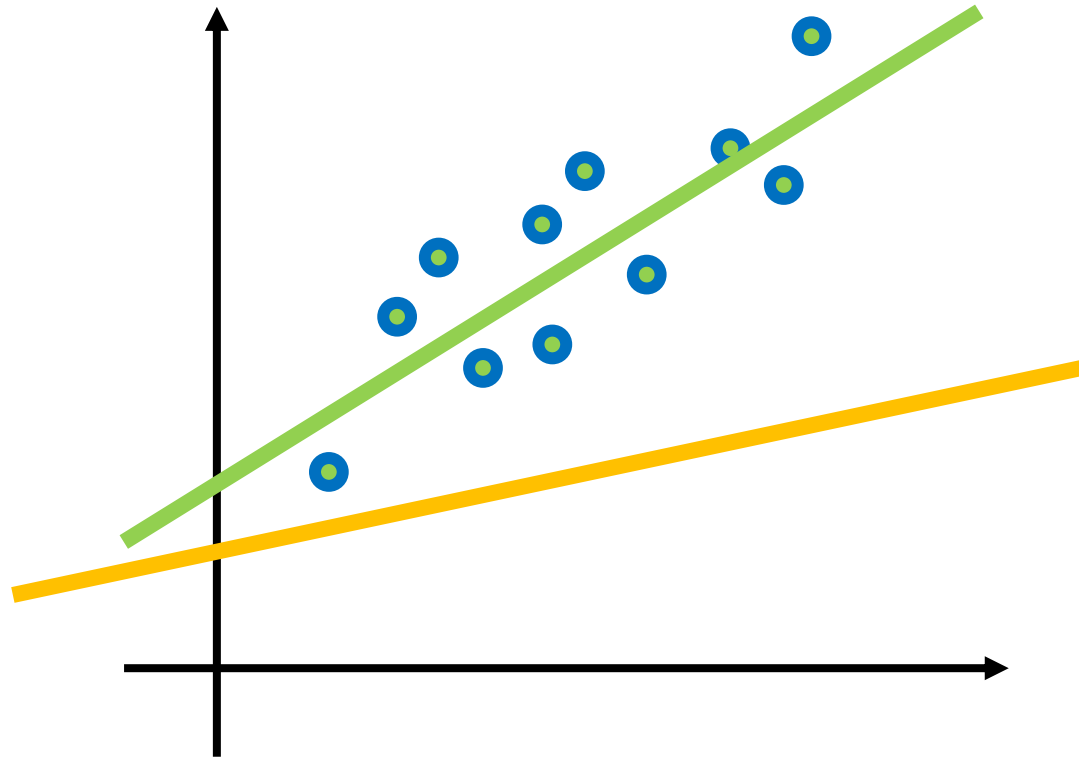


모든 파라미터들을 반복적인 방법으로 조금씩 튜닝해가는 것입니다

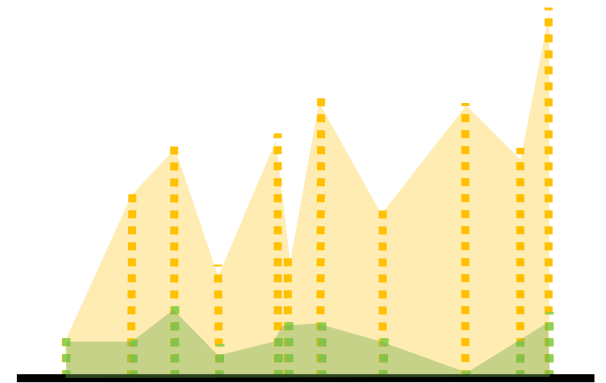
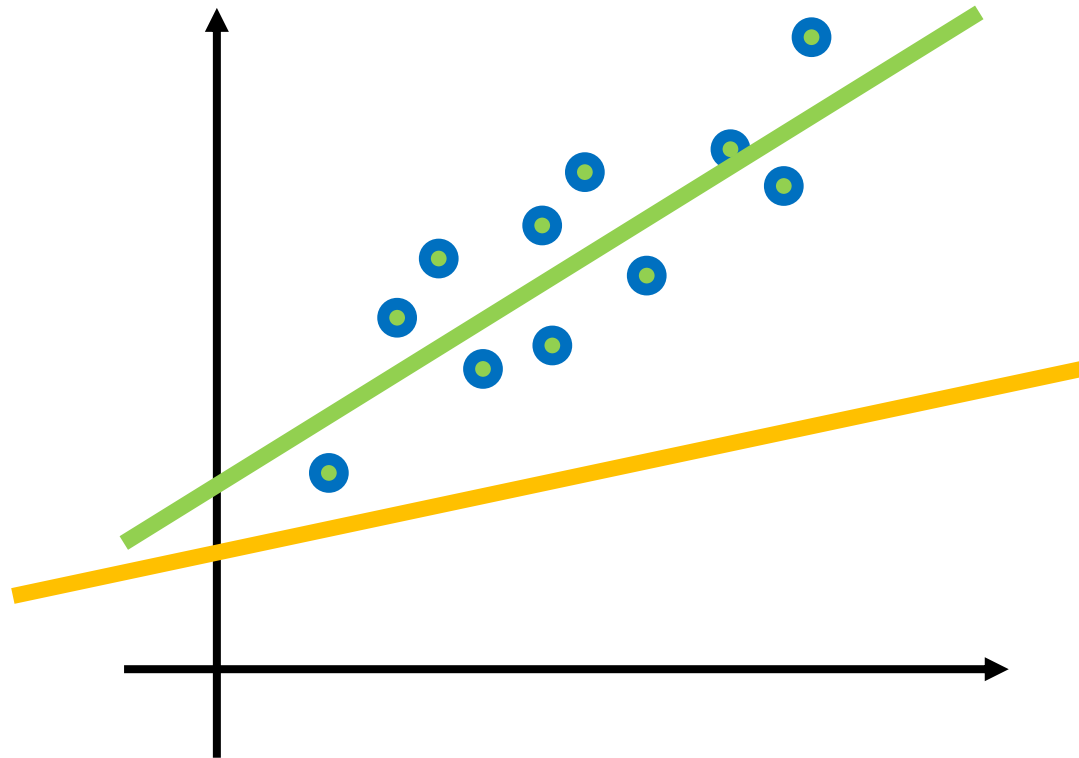
이렇게 면적 (오차의 합)이 줄어들었음을 볼수 있습니다



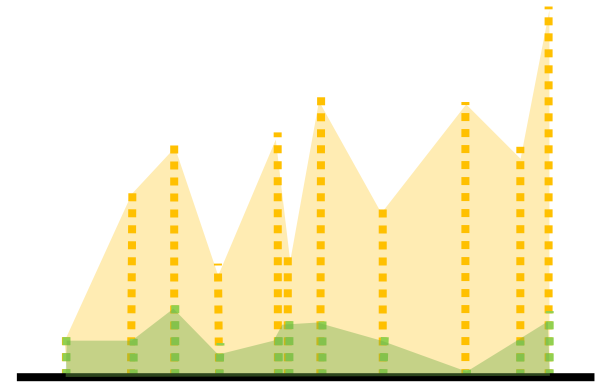
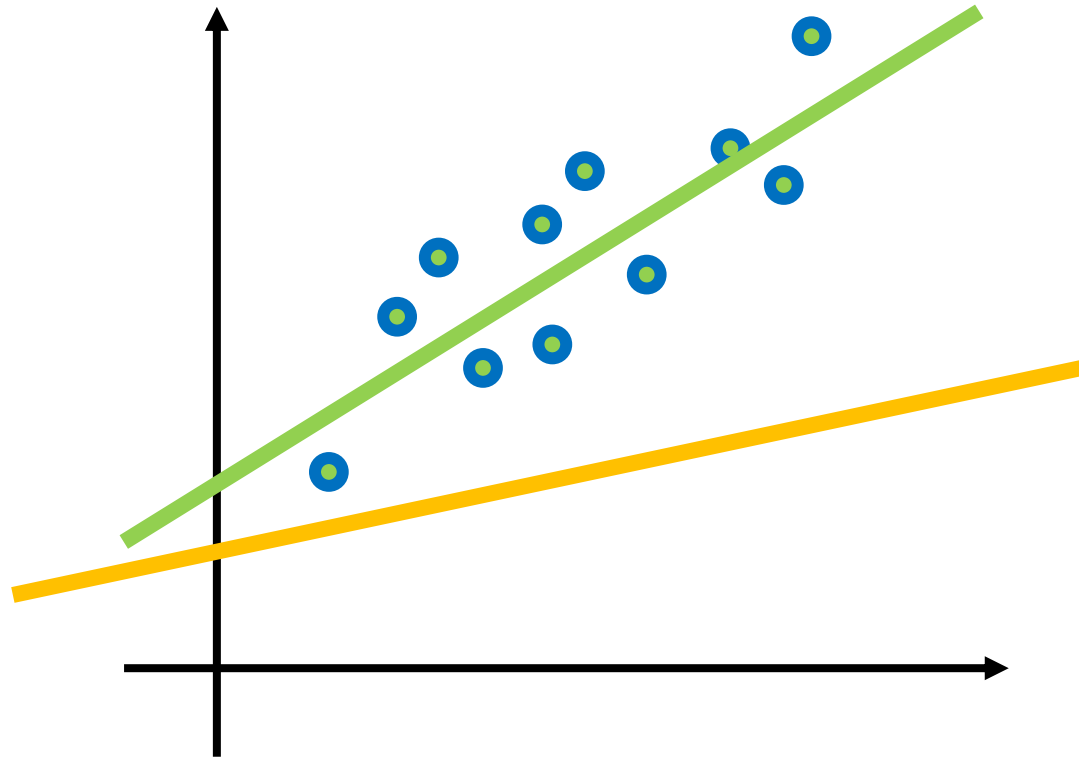
그래서 이 손실함수의 값이 최소가 되도록 하는 것이 신경망 학습에 있어서 아주 중요한 개념입니다



그러면 경사하강법을 알기 위해서 먼저 우리는



이 손실함수에 대해 좀 자세하게 알아볼 필요가 있습니다

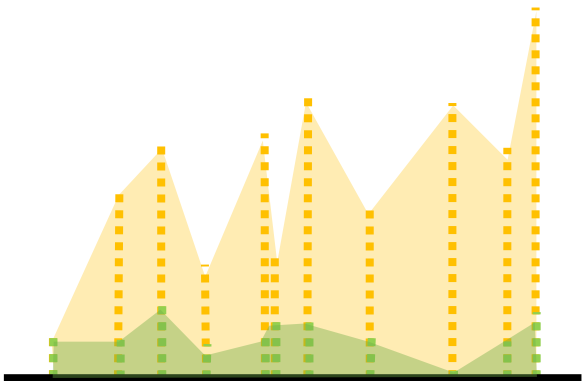
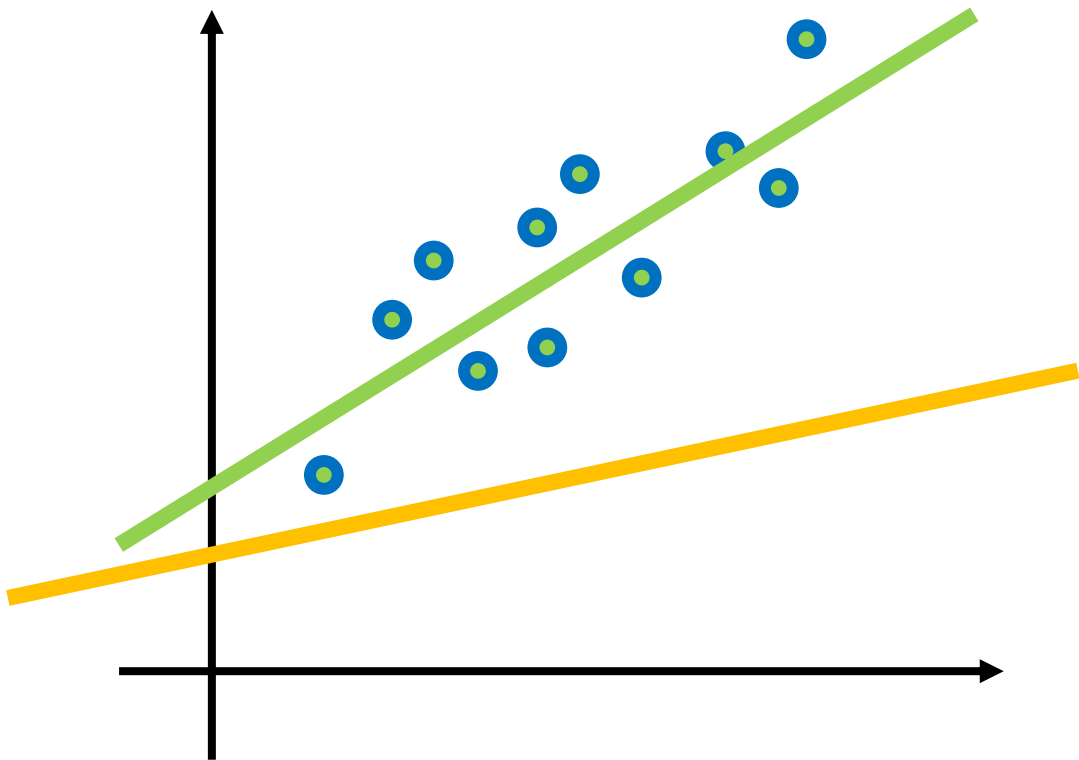


Chapter 2

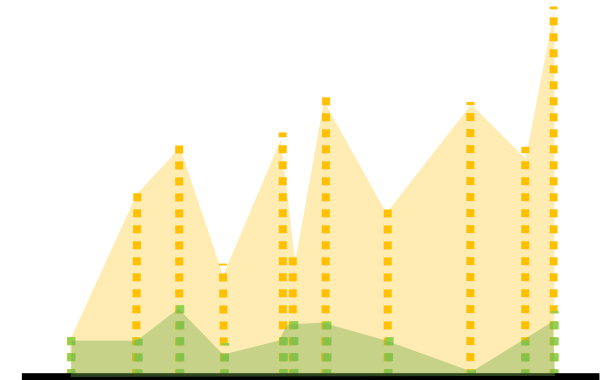
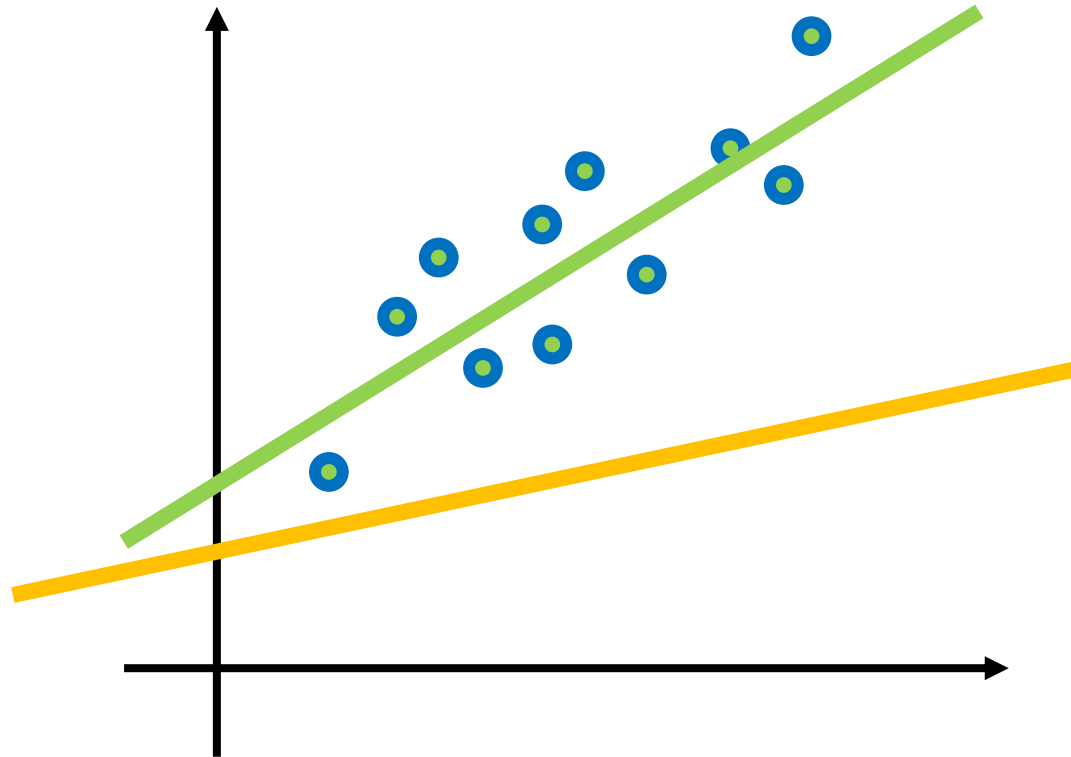
손실함수

손실함수의 개념은 좀 전에 말씀드린바와 마찬가지로

실제값과 예측값의 차이입니다



손실함수는 여러 개가 있지만,



그중 대표적인 손실함수로서는

MSE (Mean Square Error) 손실함수가 있습니다

MSE (Mean Square Error) 손실함수가 있습니다

공식은 이와 같습니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

MSE (Mean Square Error) 손실함수가 있습니다

어려워보이지만 개념은 아주 간단합니다..

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

MSE (Mean Square **Error**) 손실함수가 있습니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

MSE는 손실함수의 개념인 실제값과
예측값의 차이 **error**로 시작합니다

MSE (Mean Square Error) 손실함수가 있습니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

실제값이 클수도 있고 예측값이 클 수도 있기 때문에...


MSE (Mean Square Error) 손실함수가 있습니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

+ 혹은 - 부호의 효과를 없애기 위해


MSE (Mean **Square** Error) 손실함수가 있습니다

차이값에 제곱 (**square**)을 해줍니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$


MSE (Mean Square Error) 손실함수가 있습니다

그리고 모든 데이터들의 오차를 합
해주고


$$MSE = \frac{1}{n} \sum_i (y_i - \hat{y}_i)^2$$

MSE (**M**ean Squre Error) 손실함수가 있습니다

$$MSE = \left[\frac{1}{n} \right] \sum_i^n (y_i - \hat{y}_i)^2$$

데이터 개수만큼 나누어주어 평균 (**mean**)
을 구합니다

그래서 MSE는, 각 데이터들의 오차들의 제곱합의 평균입니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

그냥 직관적으로는, 실제값과 예측값간의 차이들의 평균정도로

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

생각하셔도 충분합니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

그래서 손실함수의 값이 크면 실제값과 예측값 차이가 크고

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

그래서 손실함수의 값이 작으면 실제값과 예측값 차이가 작은데

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

손실함수값이 작으면 작을 수록 좋은 것, 정도로 이해하시면 충분합니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

이 MSE외에도 MAE (Mean Absolute Error)도 있습니다

$$MAE = \frac{1}{n} \sum_i^n |y_i - \hat{y}_i|$$

MAE는 MSE에서 제곱 (Square)대신에 절대값 (Absolute)를 사용했
을뿐 나머지는 동일합니다

$$MAE = \frac{1}{n} \sum_i^n |y_i - \hat{y}_i|$$
$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

절대값을 쓴 이유는 제곱을 할 경우 오차가 1보다 작을 때

$$MAE = \frac{1}{n} \sum_i^n |y_i - \hat{y}_i|$$
$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

제공할 경우 그 수가 더 작아지기 때문에 오차가 왜곡될 수 있기
때문입니다

$$MAE = \frac{1}{n} \sum_i^n |y_i - \hat{y}_i|$$
$$|1 - 1.1| = 0.1$$

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$
$$(1 - 1.1)^2 = 0.01$$

또 다른 계열의 손실함수로서

교차 엔트로피 (Cross Entropy) 함수가 있습니다

교차 엔트로피 함수 공식은 다음과 같습니다

$$LogLoss = \frac{1}{n} \sum_i^n -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

*Log Loss for Binary Classification



교차 엔트로피는 딥러닝에서 일반적으로 사용되는
중요한 손실함수이므로

$$LogLoss = \frac{1}{n} \sum_i^n -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

*Log Loss for Binary Classification



제가 따로 영상을 준비하여 설명해드리고자 합니다

$$LogLoss = \frac{1}{n} \sum_i^n -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

*Log Loss for Binary Classification



다만 지금은, 교차 엔트로피 함수는 위에서 다룬 MSE에 비해서

$$LogLoss = \frac{1}{n} \sum_i^n -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

*Log Loss for Binary Classification



학습속도가 빠르다는 장점이 있다는 것을 말씀드리고 싶습니다

$$LogLoss = \frac{1}{n} \sum_i^n -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

*Log Loss for Binary Classification



자세한 부분은 교차 엔트로피 영상에서 더 깊게 다루어 보도록 하겠습니다

$$LogLoss = \frac{1}{n} \sum_i^n -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

*Log Loss for Binary Classification



Chapter 3

경사하강법

이제 본격적으로 경사하강법의 알고리즘에
대해서 알아보도록 하겠습니다

경사하강법의 개념은 의외로 간단합니다

만약 다음과 같은 손실함수 (MSE)를 사용한다고 했을 때,

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

이해를 돕기 위해 가장 간단한 경우로 생각해 보자면,

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

N=1일 경우로 한번 생각을 해보겠습니다

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

그러면 모든 n 을 1로 바꾸어 볼때..

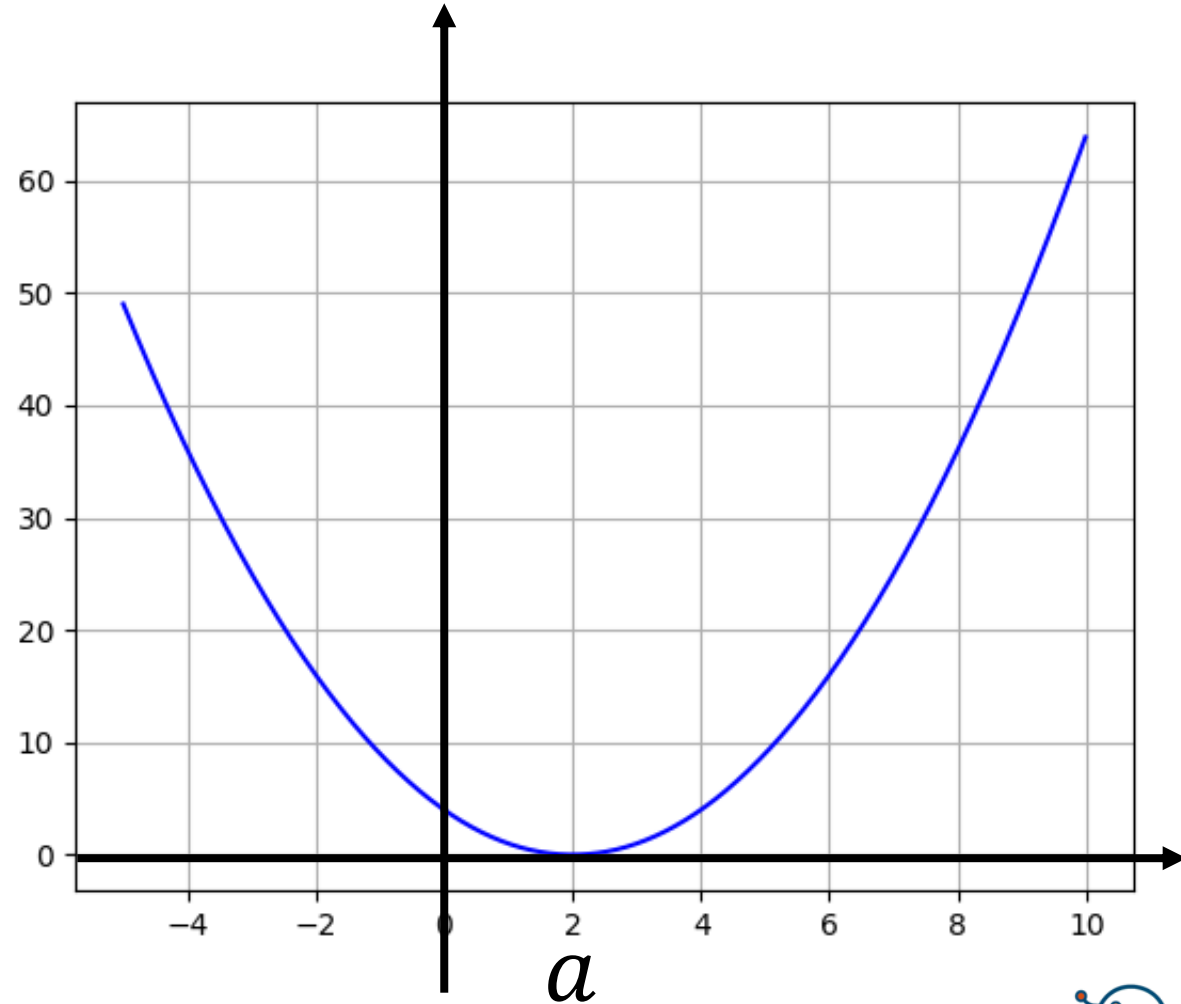
$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$

손실함수는 뭔가 아주 익숙한 2차원 함수 공식이 되는 것을 볼 수가 있습니다

$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$

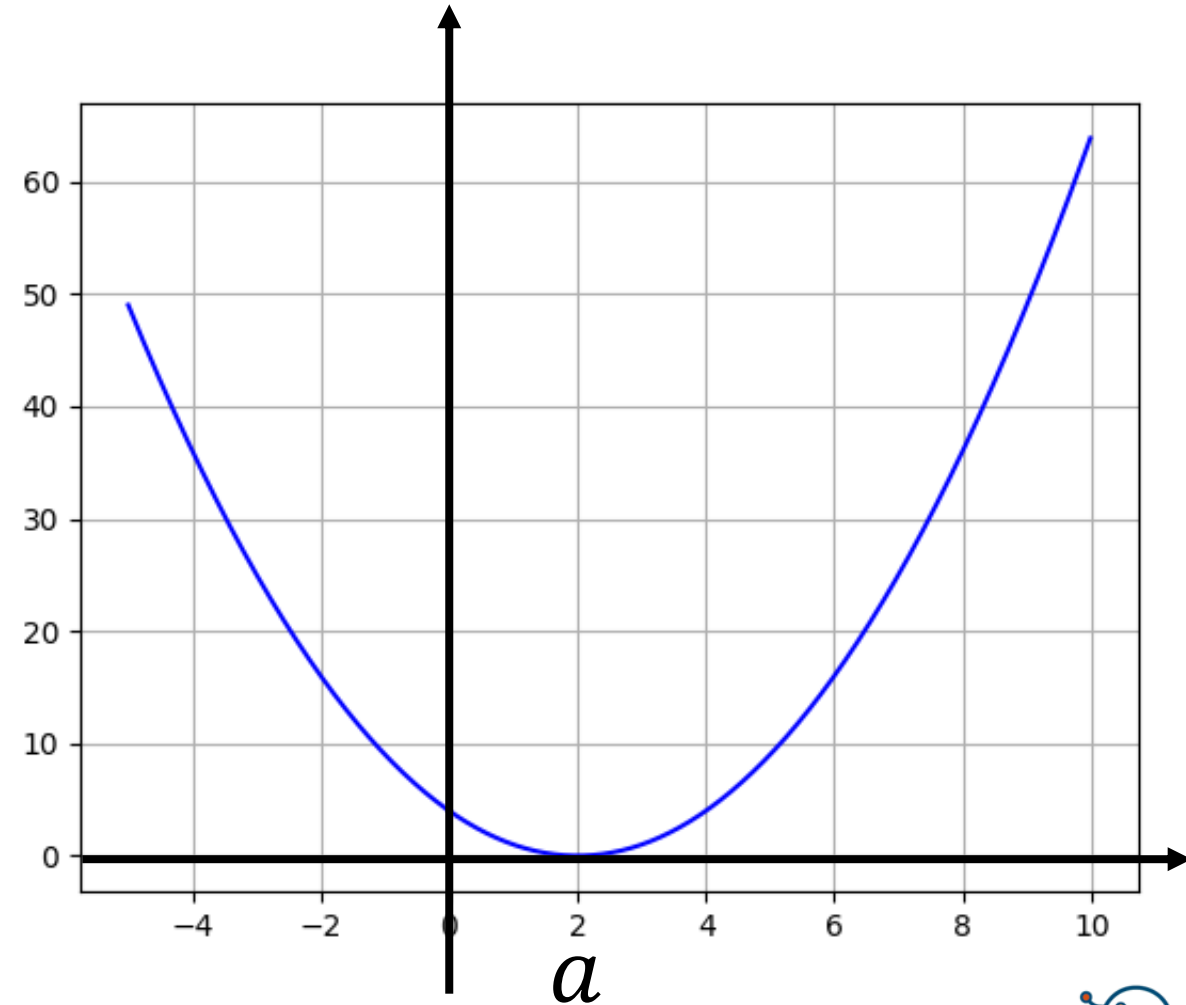
그러면 이와 같이 그래프도 그릴수가 있습니다

$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$



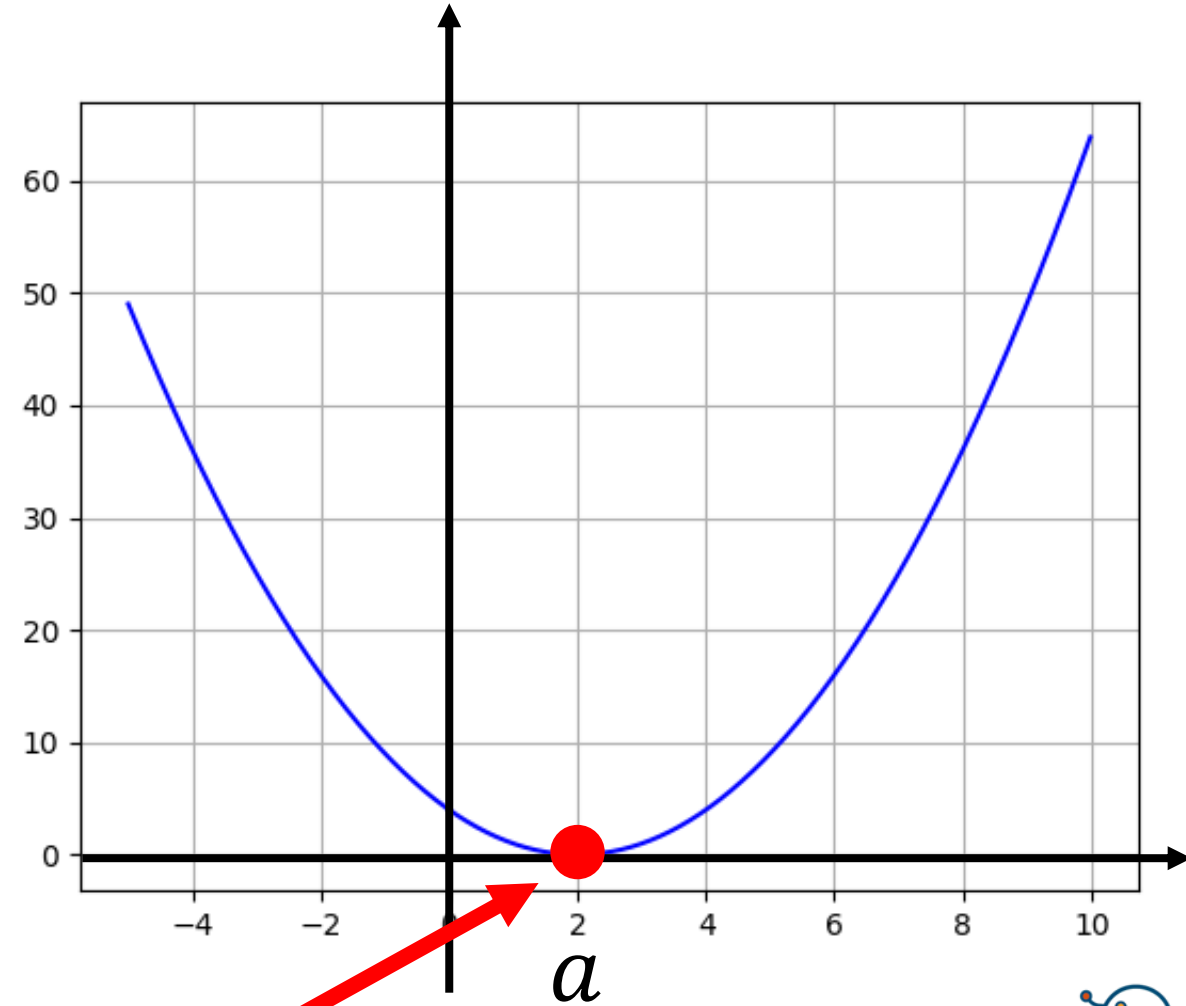
오차가 최소가 되는 지점은 한눈에 알아볼 수 있습니다

$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$



오차가 최소가 되는 지점은 한눈에 알아볼 수 있습니다

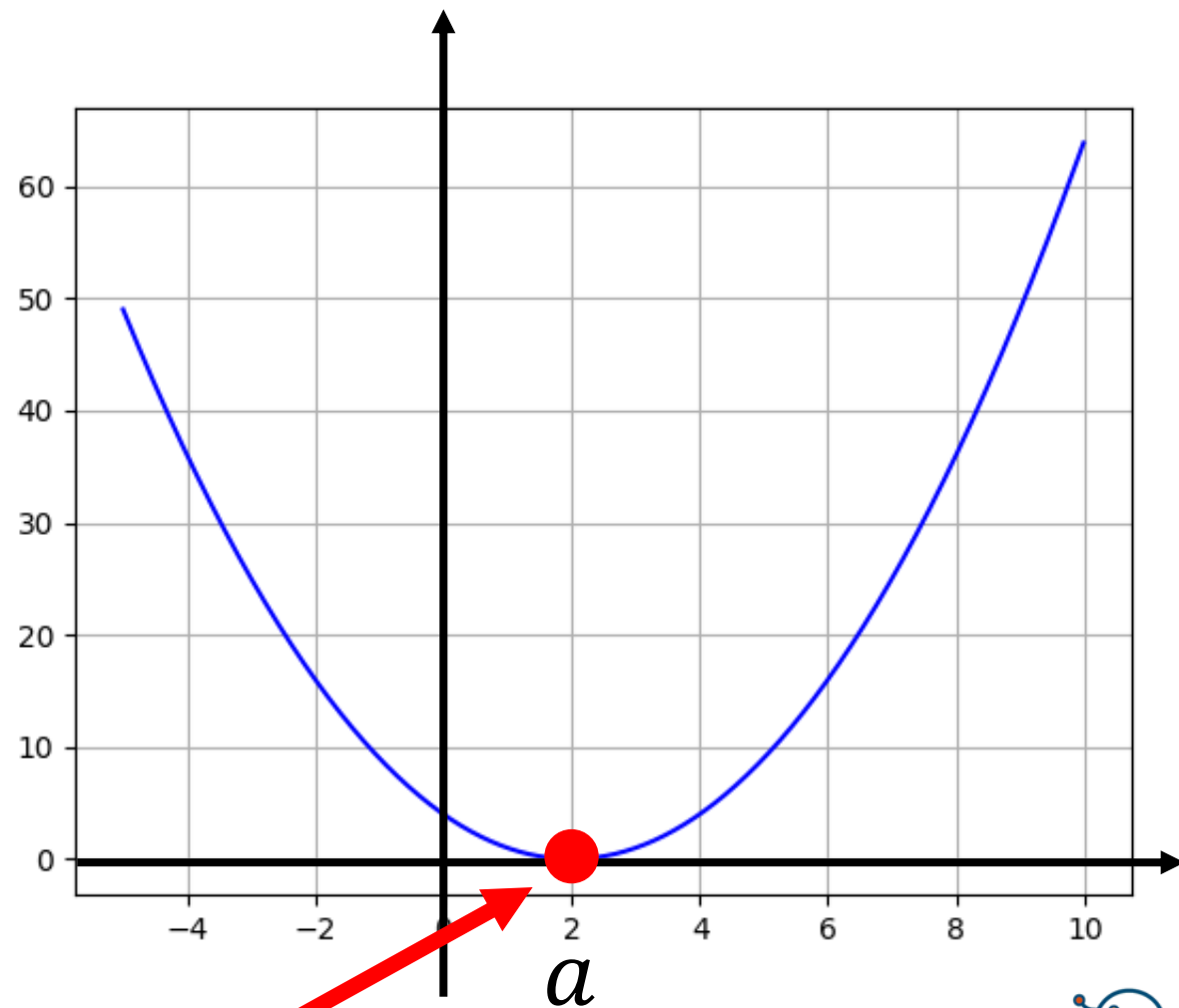
$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$



바로 여기가 되겠지요

사람의 눈은 이렇게 오차가 최소인 지점을 바로 찾을 수 있지만

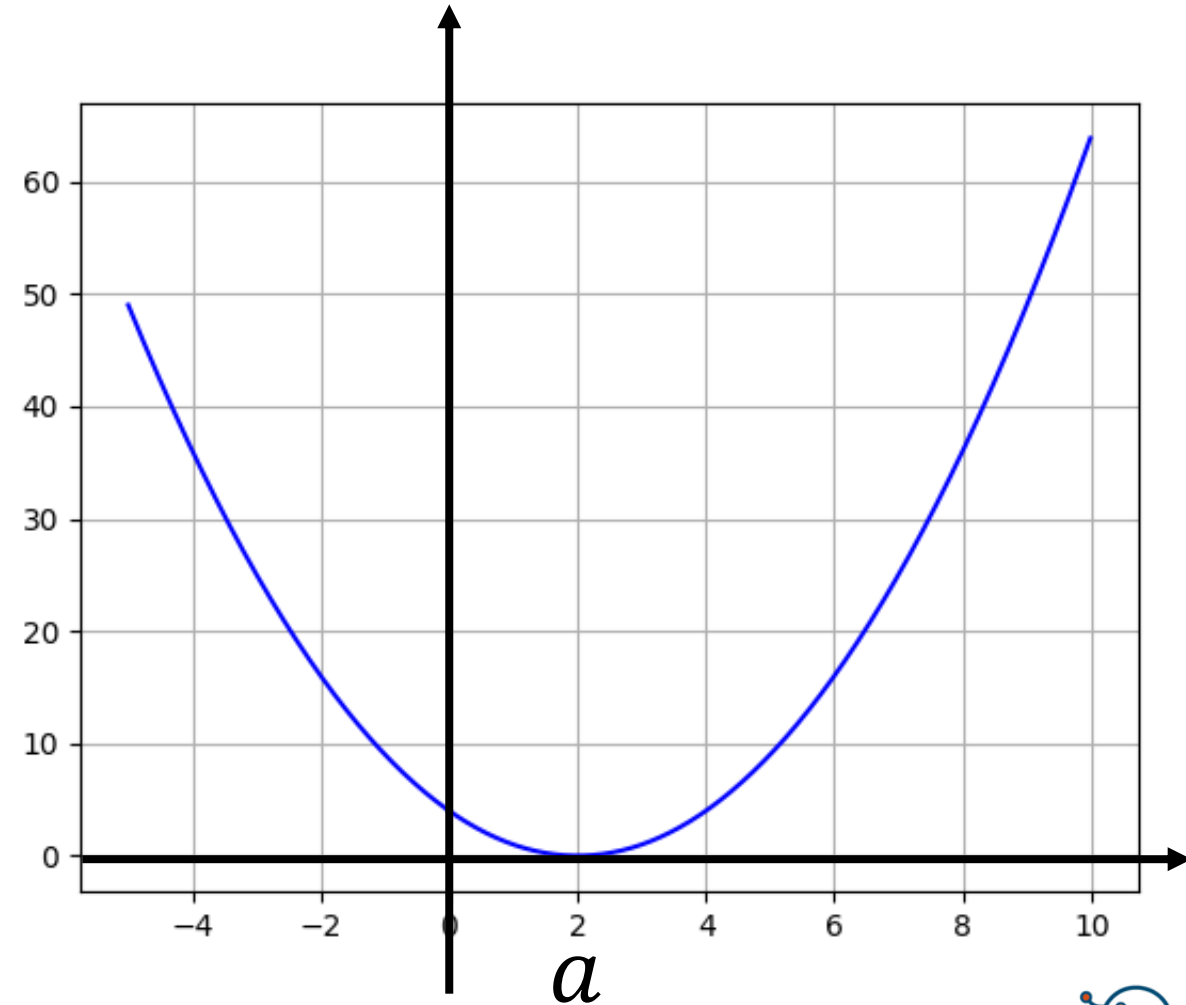
$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$



바로 여기가 되겠지요

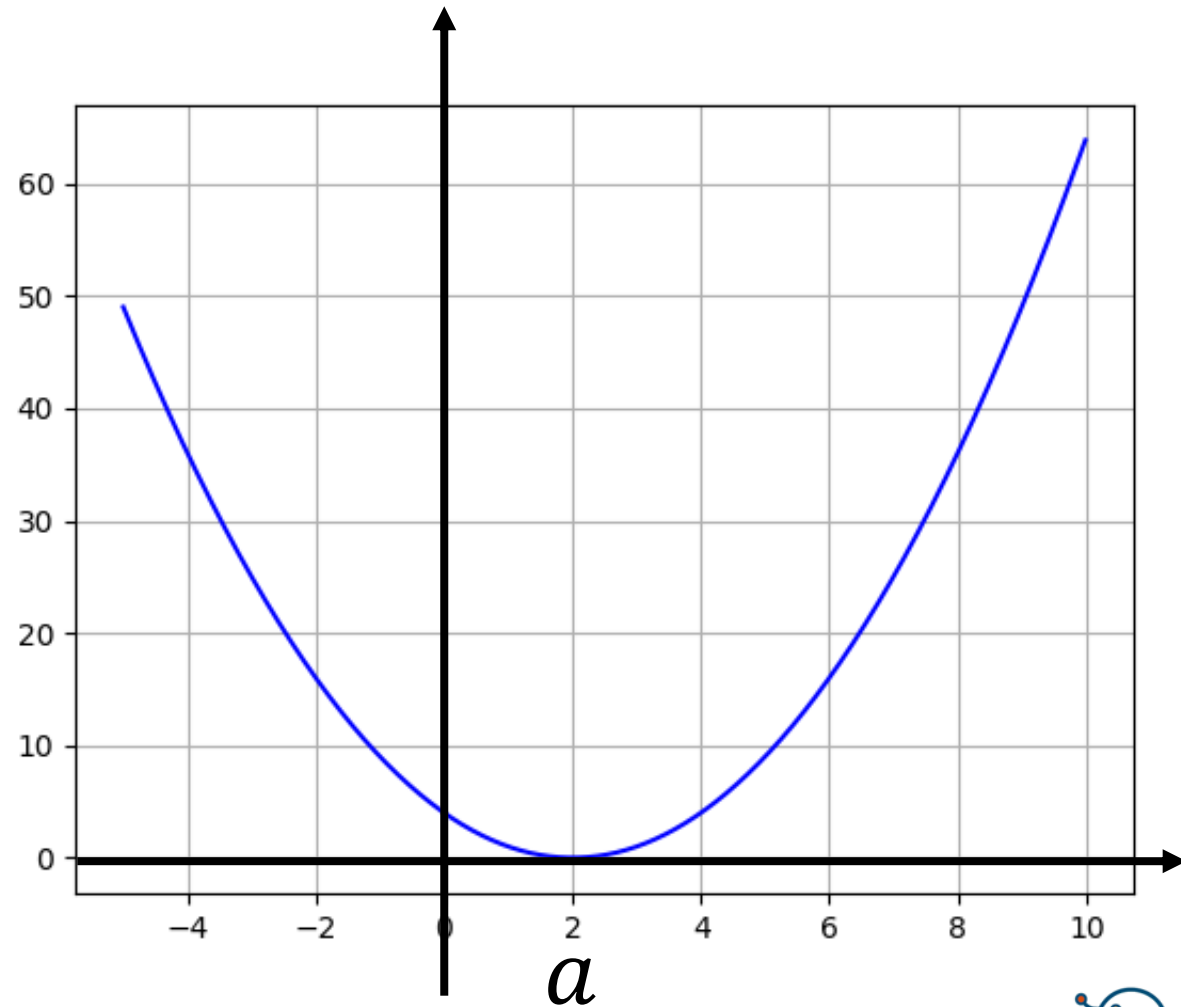
안타깝게도 기계는 그렇지 못합니다

$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$



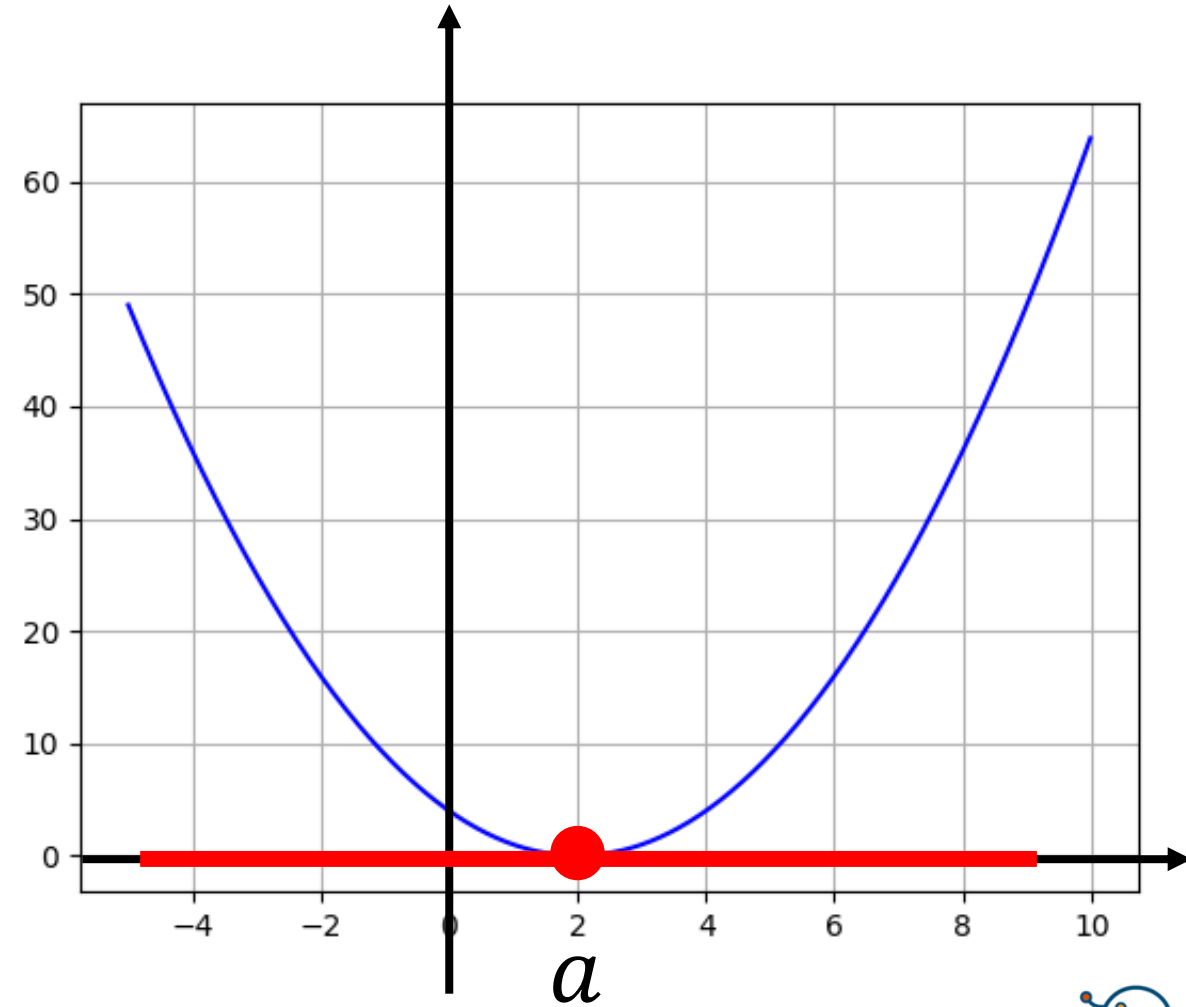
그런데 한가지 중요한 사실을 발견할 수 있는데,

$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$

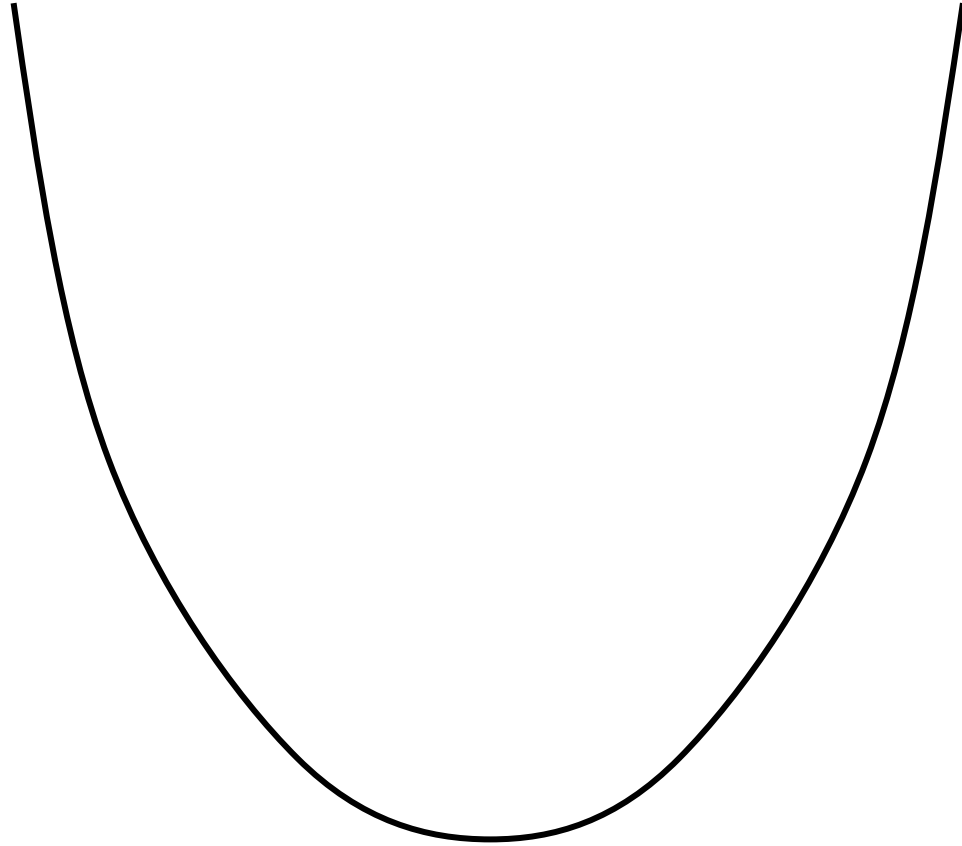


오차가 최소가 되는 지점은 늘 접선의 기울기가 0이 된다는 사실입니다

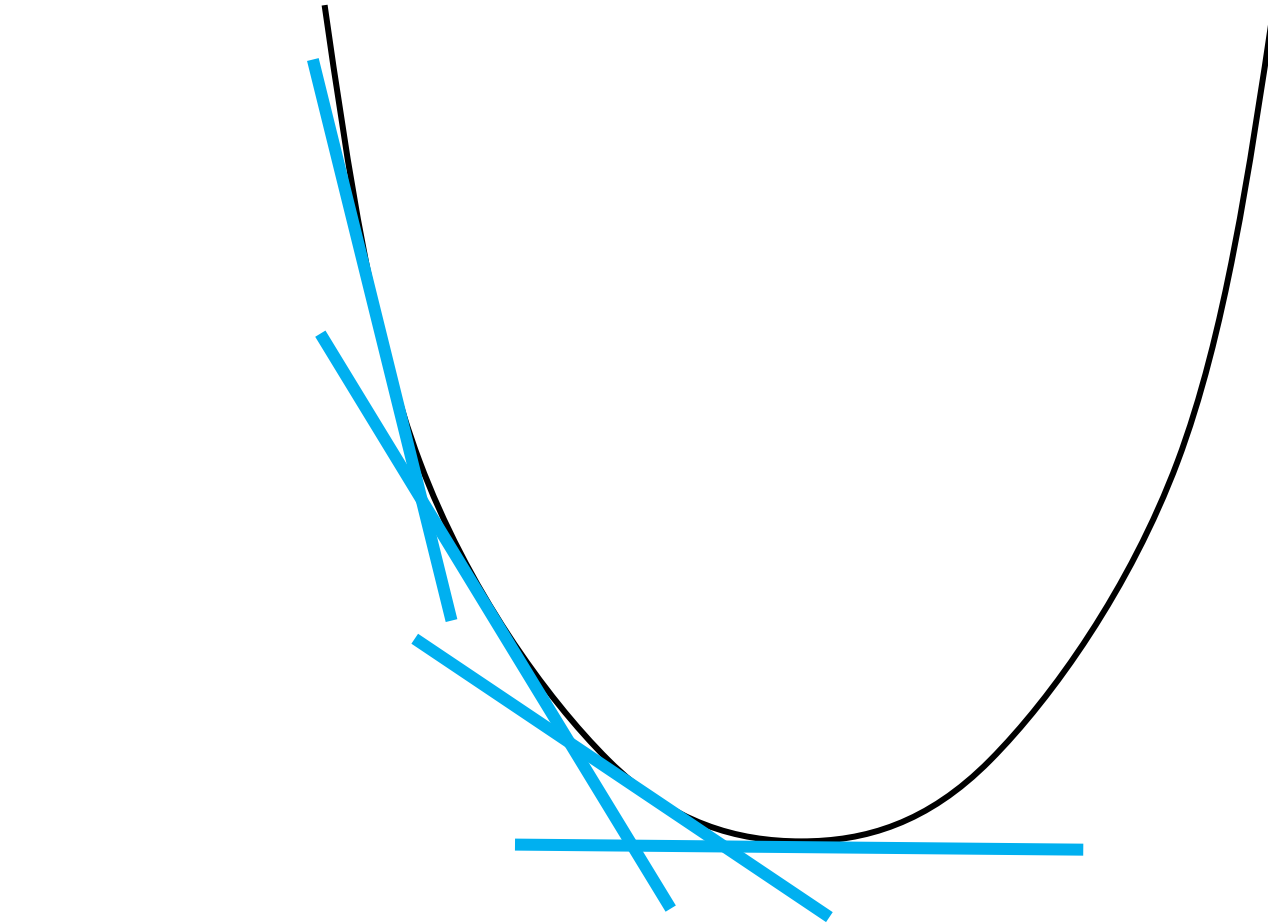
$$MSE = \frac{1}{1} \sum_1^1 (y_1 - \hat{y}_1)^2$$
$$y = \frac{1}{1} \sum_1^1 (x - a)^2$$



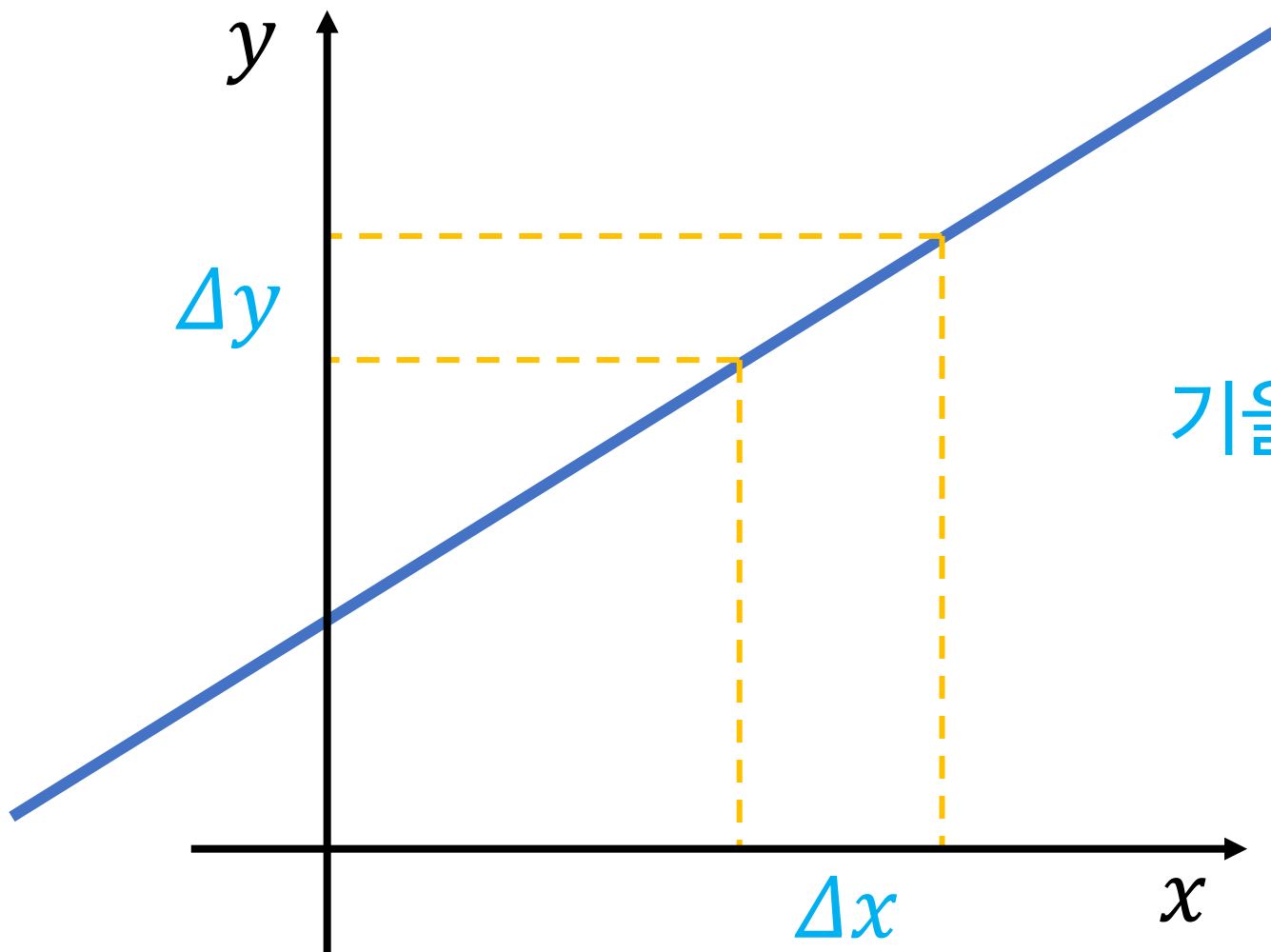
그래서 접선의 기울기가 0 이 되는 지점을 찾을 때 까지



접선의 기울기를 점진적으로 하강시키는 방법이 바로 경사하강법입니다

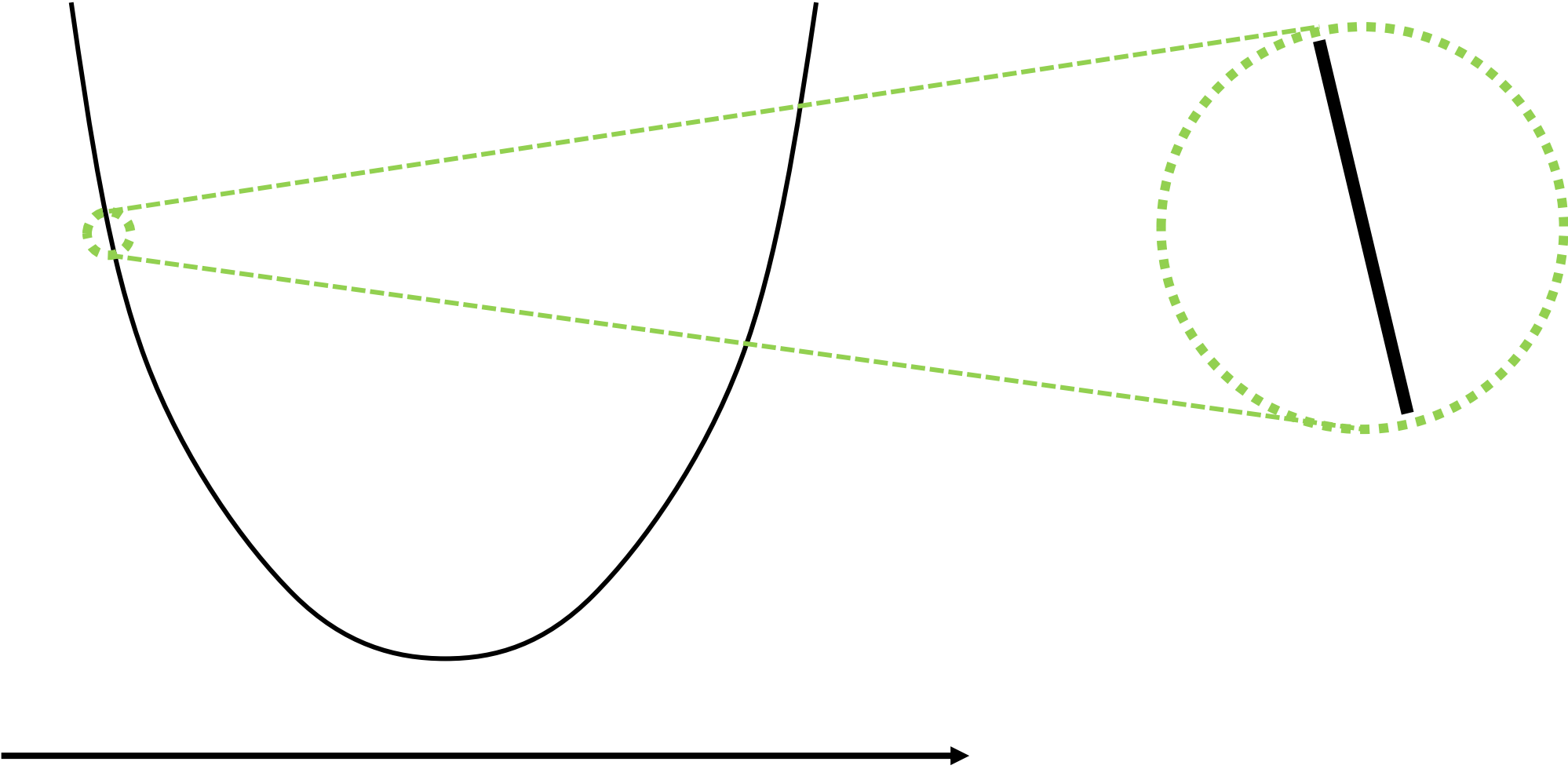


일차함수에서 기울기란 다음과 같이 표현됩니다

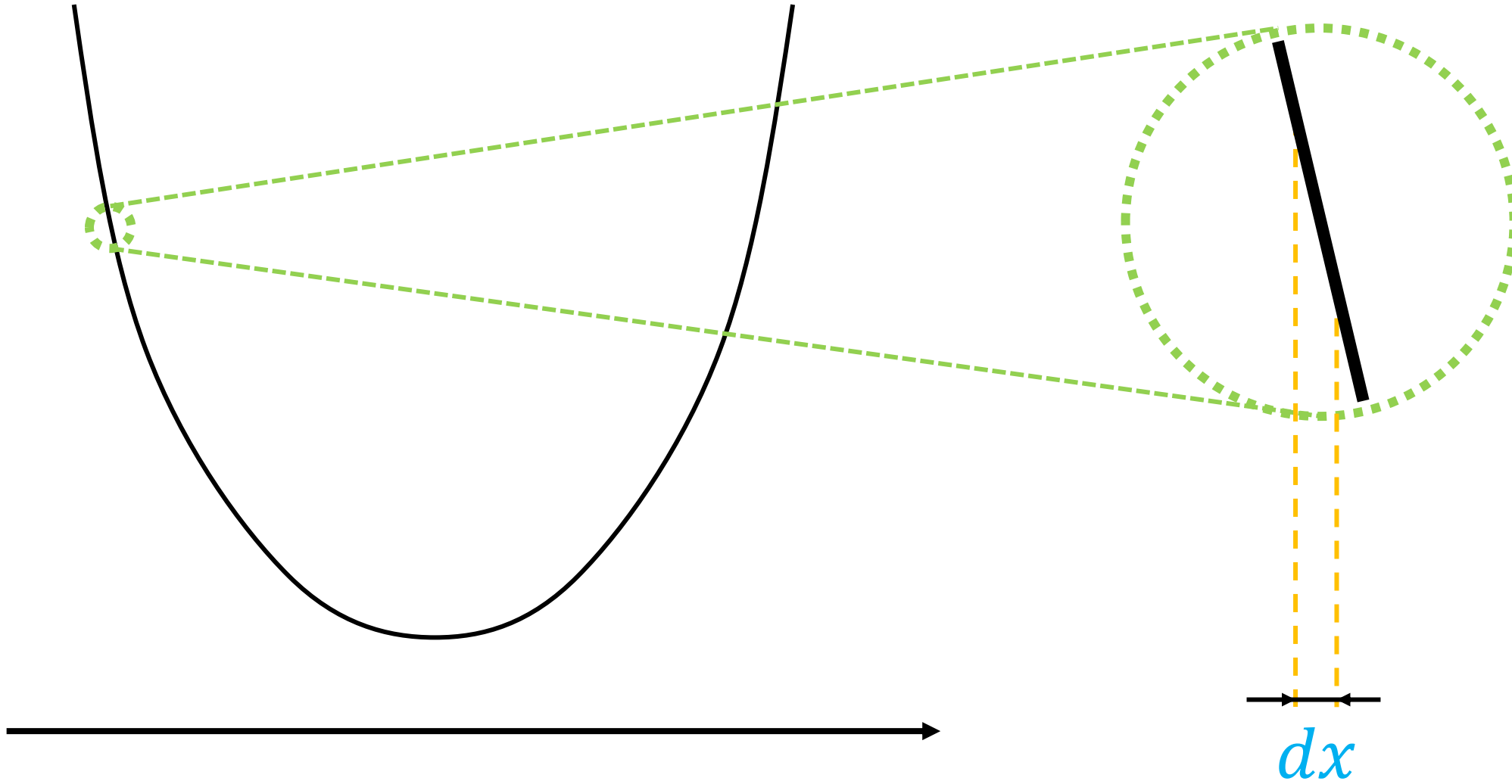


$$\text{기울기} = \frac{\Delta y}{\Delta x} = \frac{y\text{의 증가량}}{x\text{의 증가량}}$$

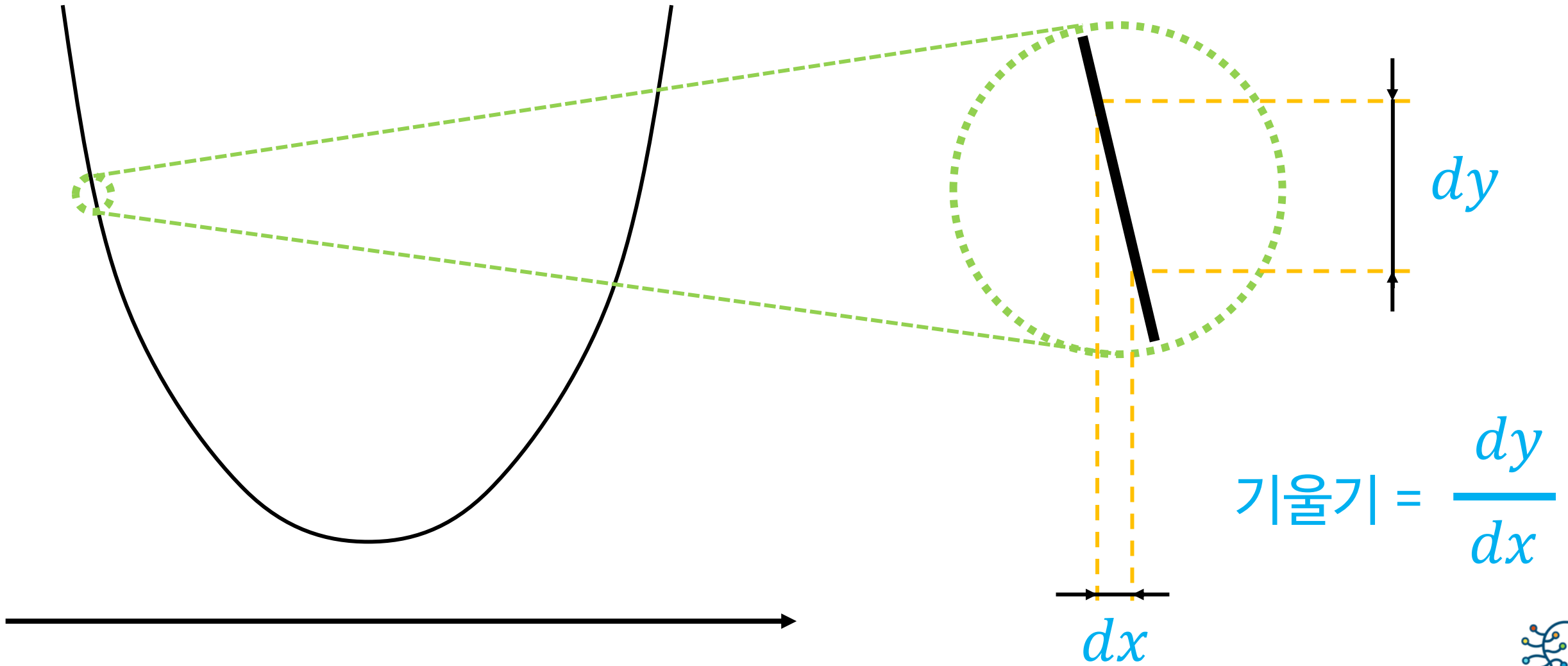
이와 같이 꼭 직선이 아닌 경우라도,



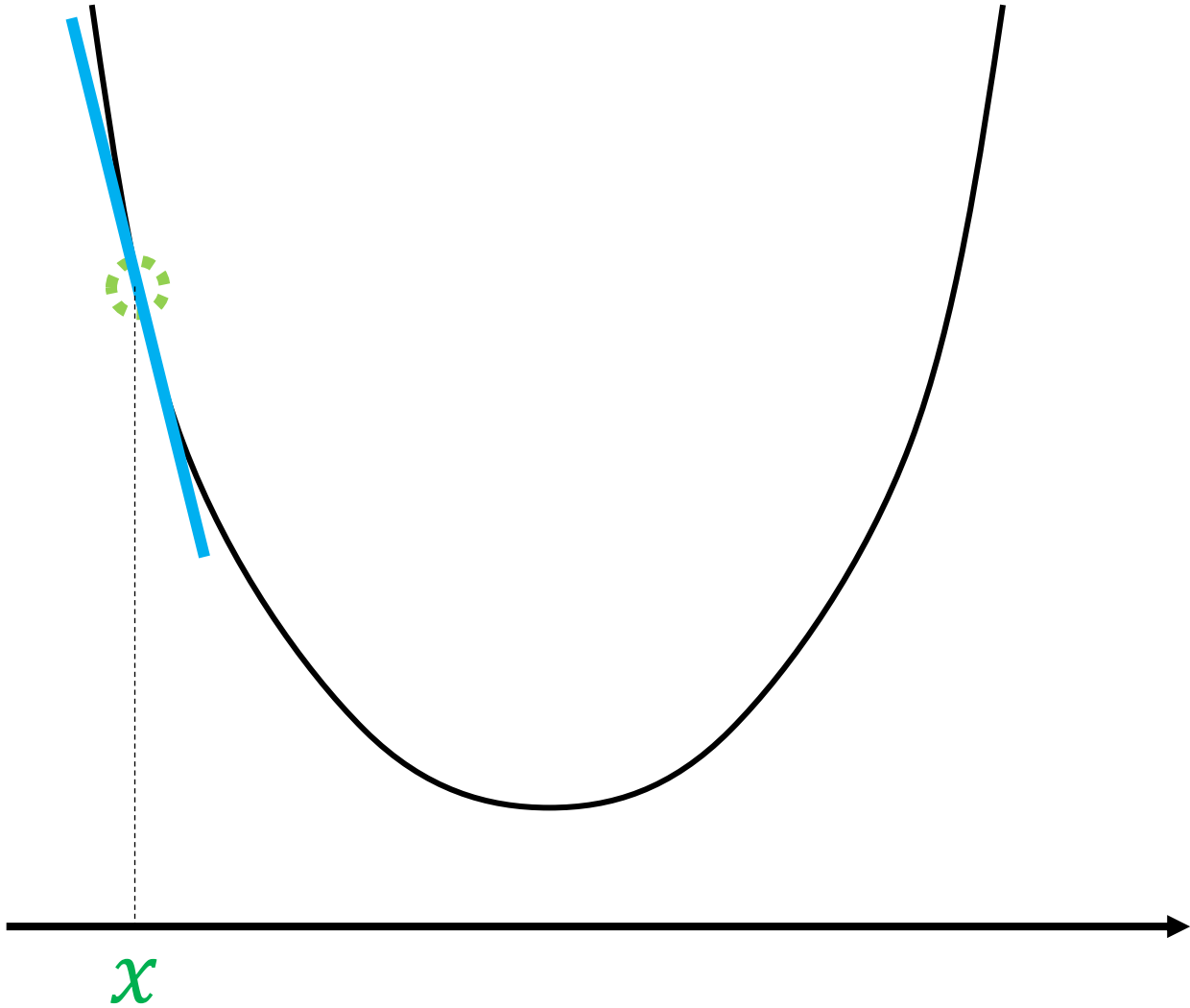
지극히 작은 x 의 변화량 dx 와



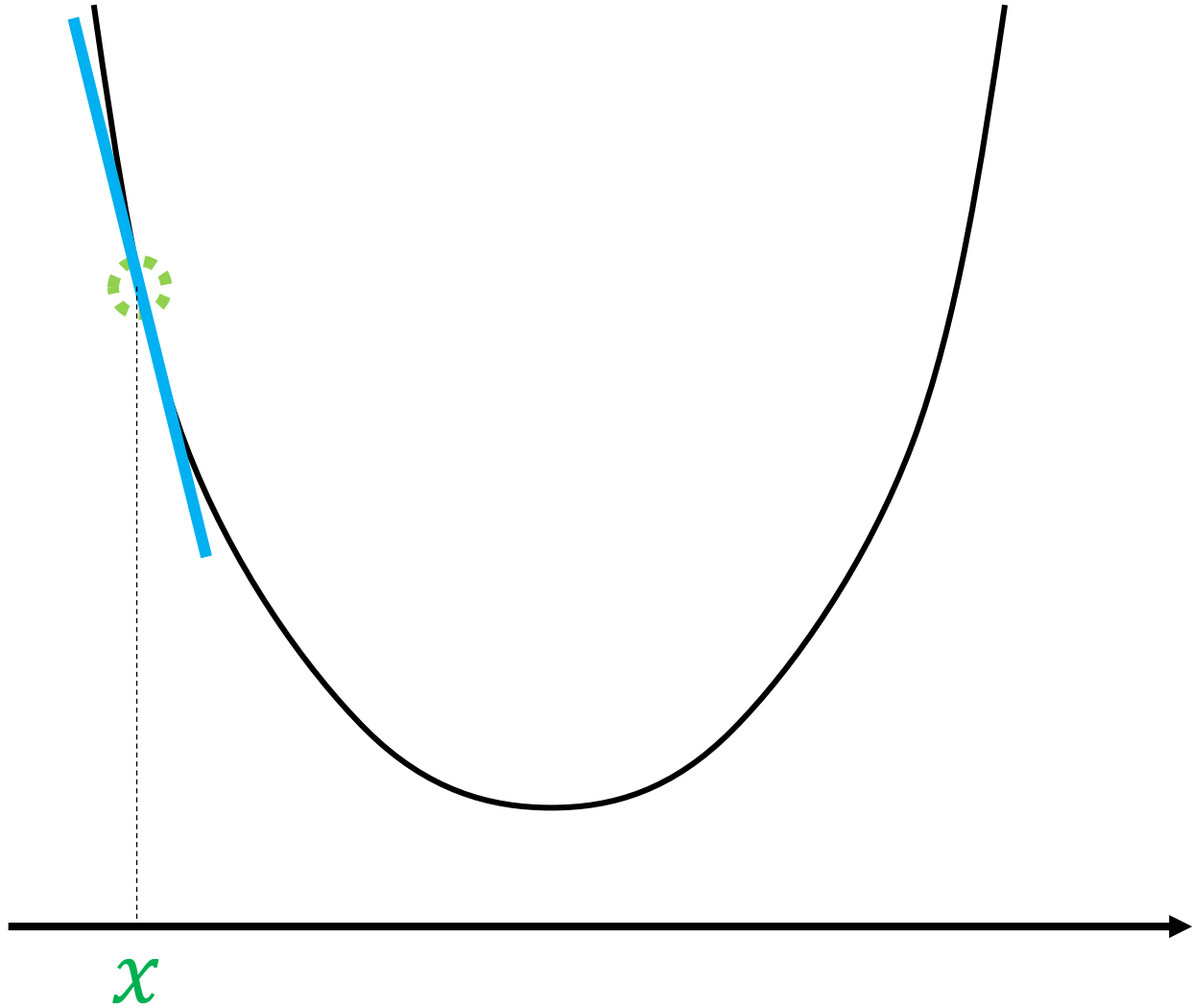
y의 변화량 dy로 구할 수 있습니다



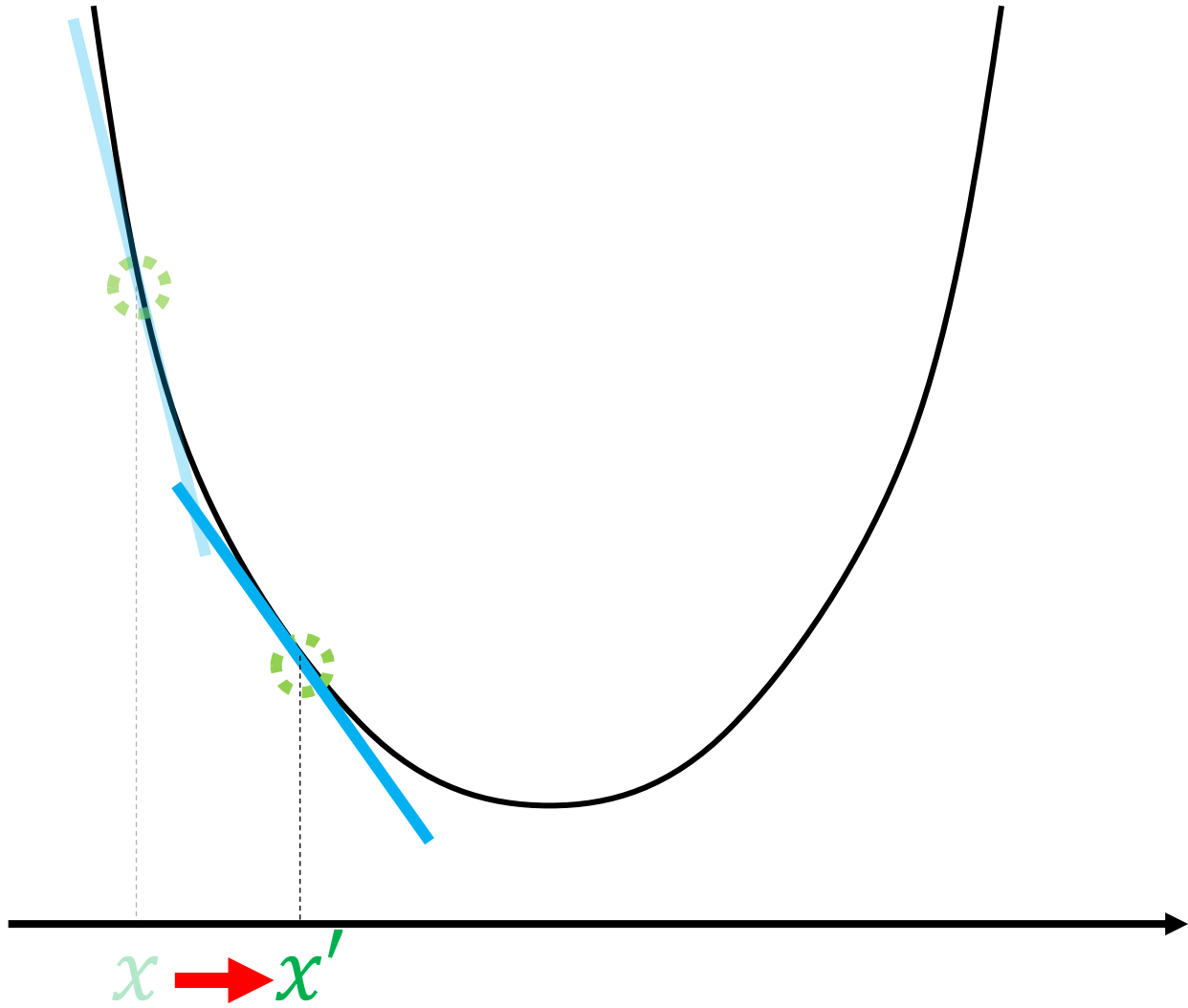
그래서 이렇게 한 점의 기울기를 구한 다음



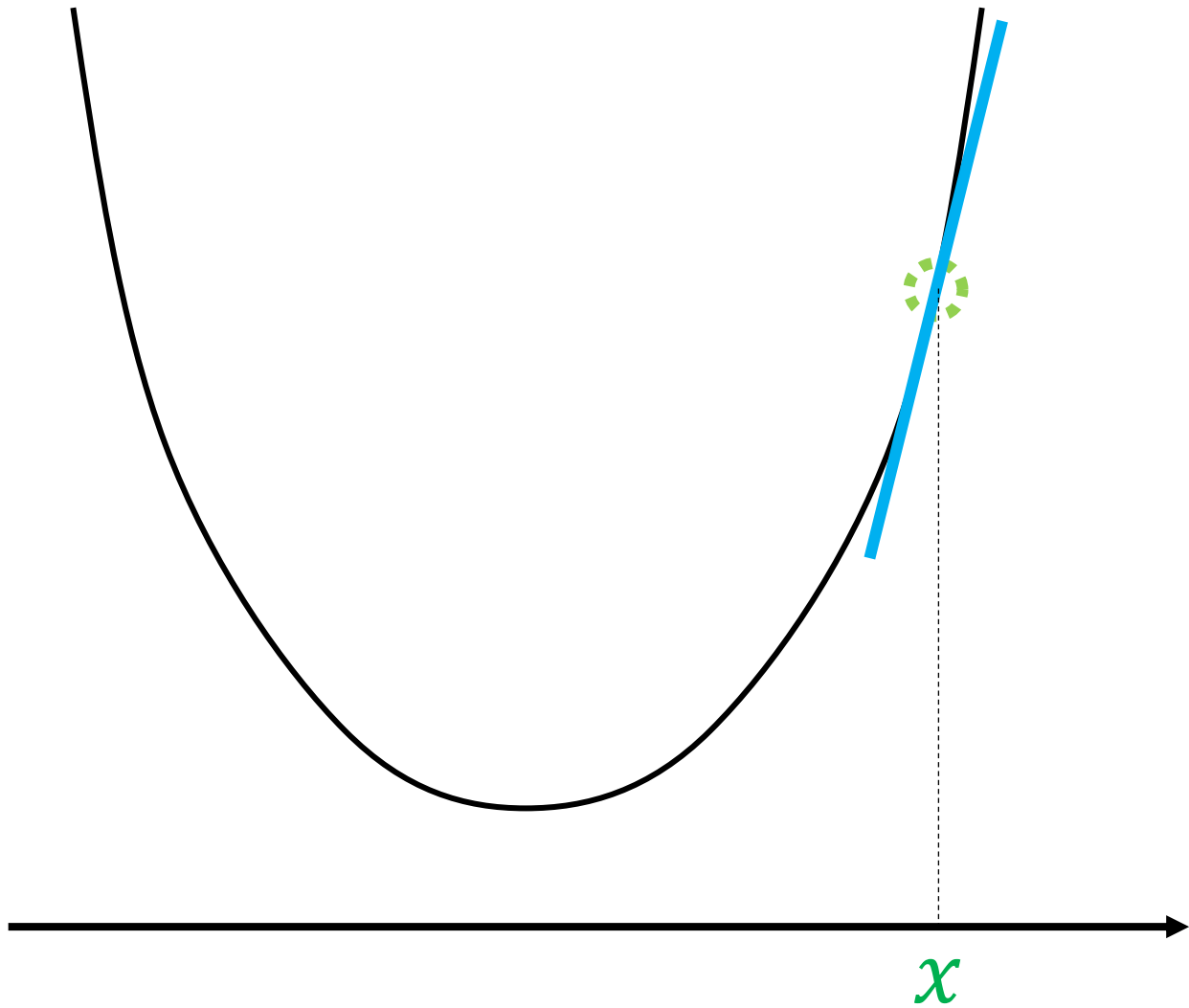
그 기울기가 음수 (-) 이면



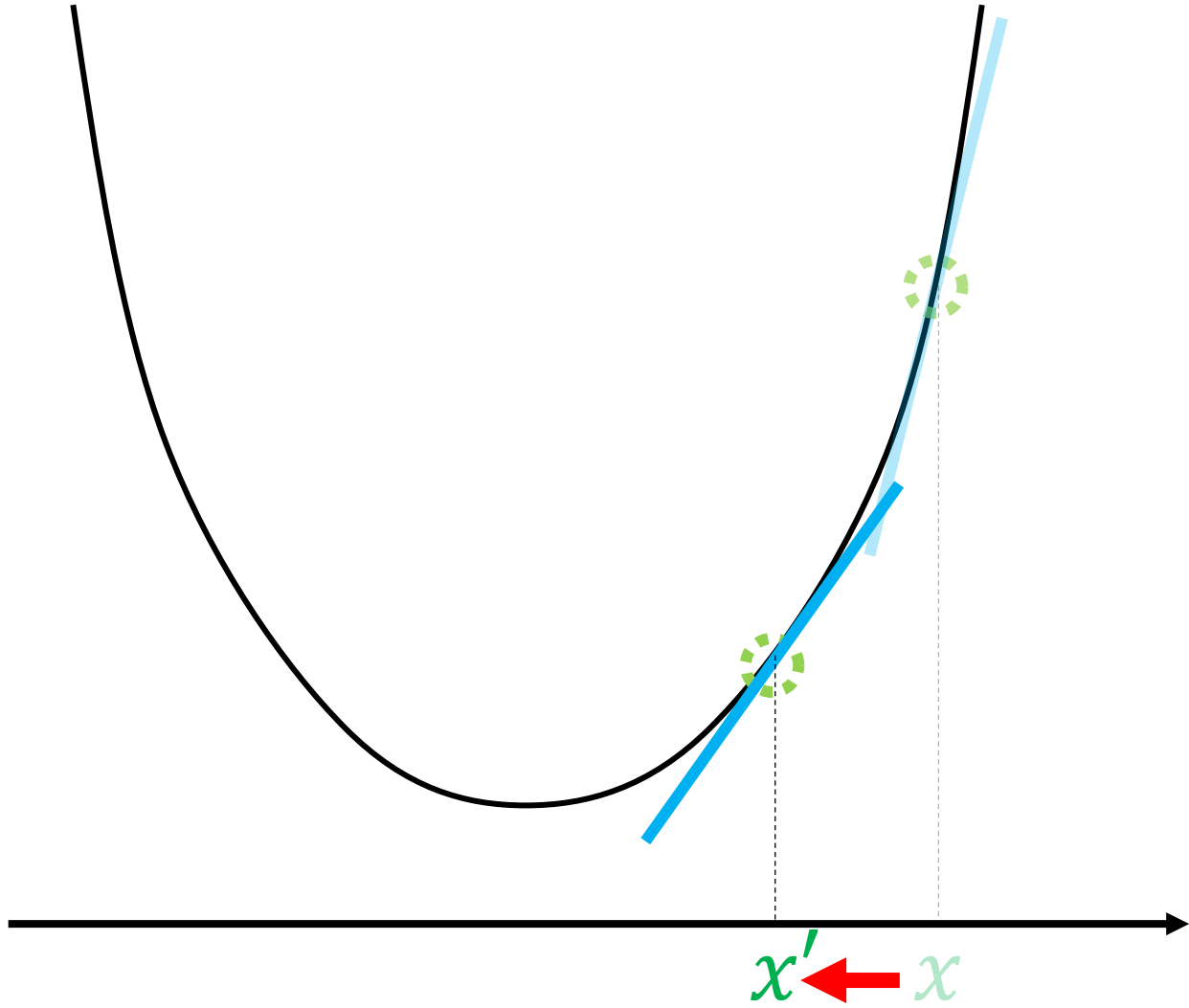
그 기울기가 음수 (-) 이면 x 를 증가시키고



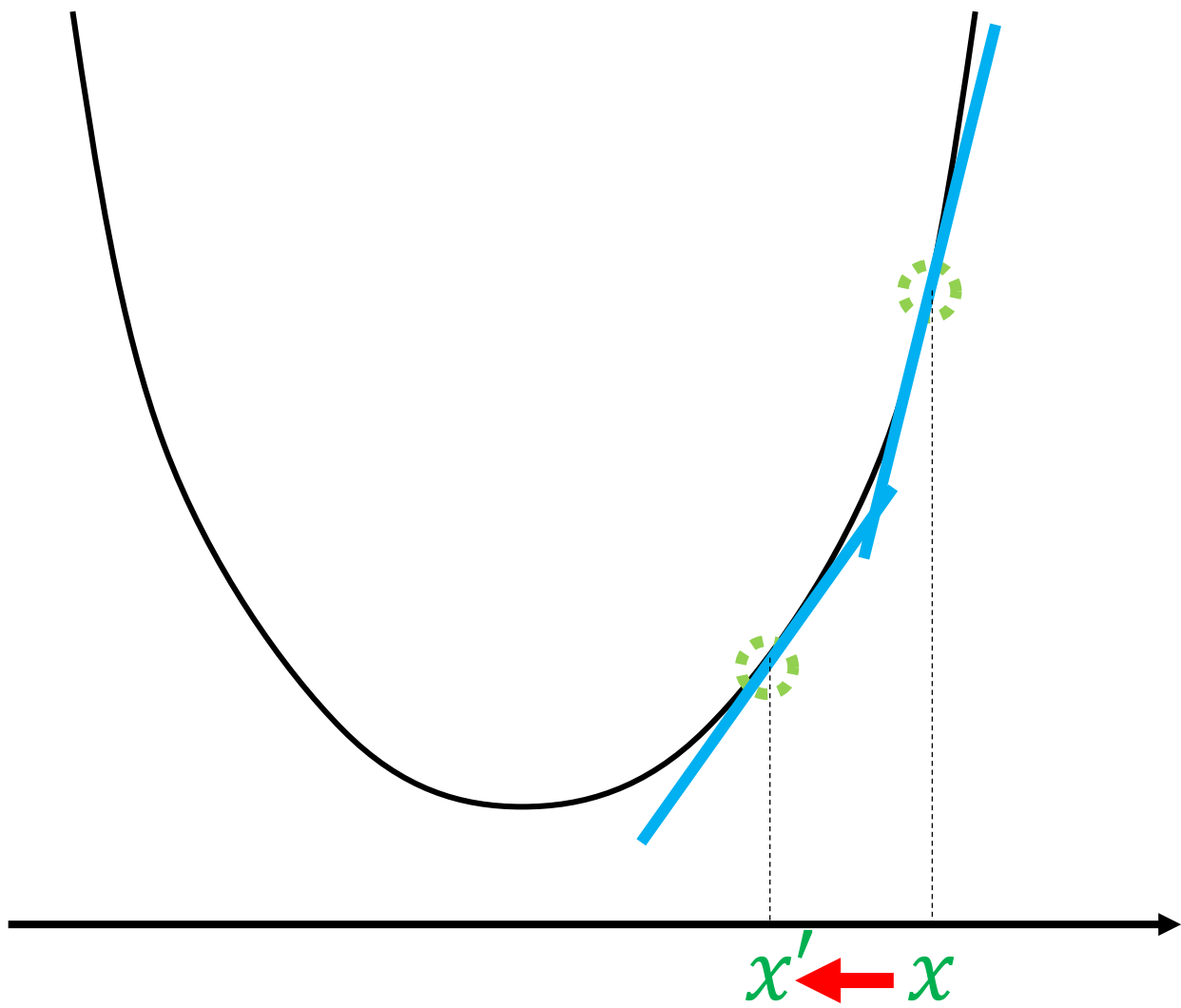
그 기울기가 양수 (+) 이면



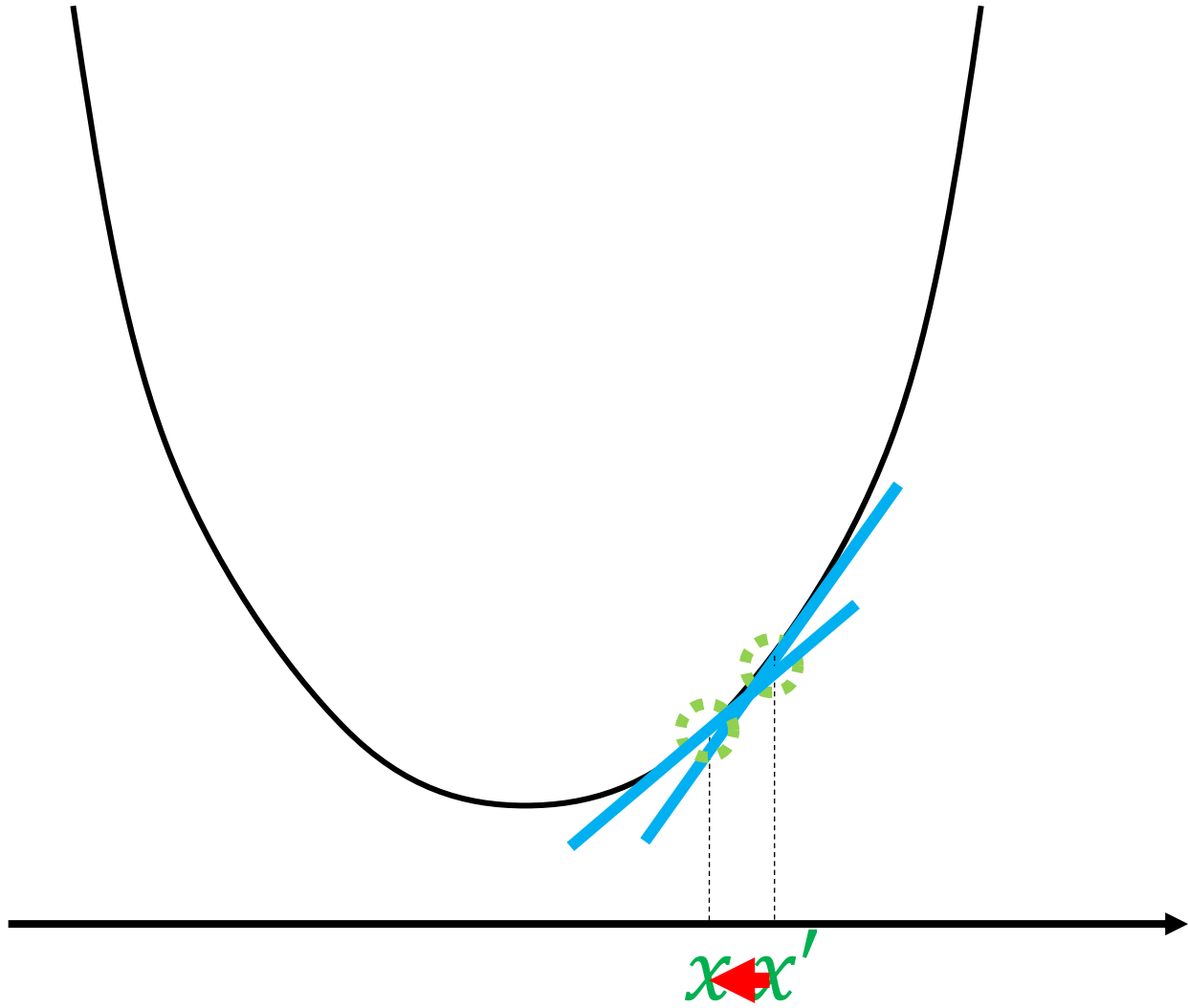
그 기울기가 양수 (+) 이면 x 를 감소시킵니다



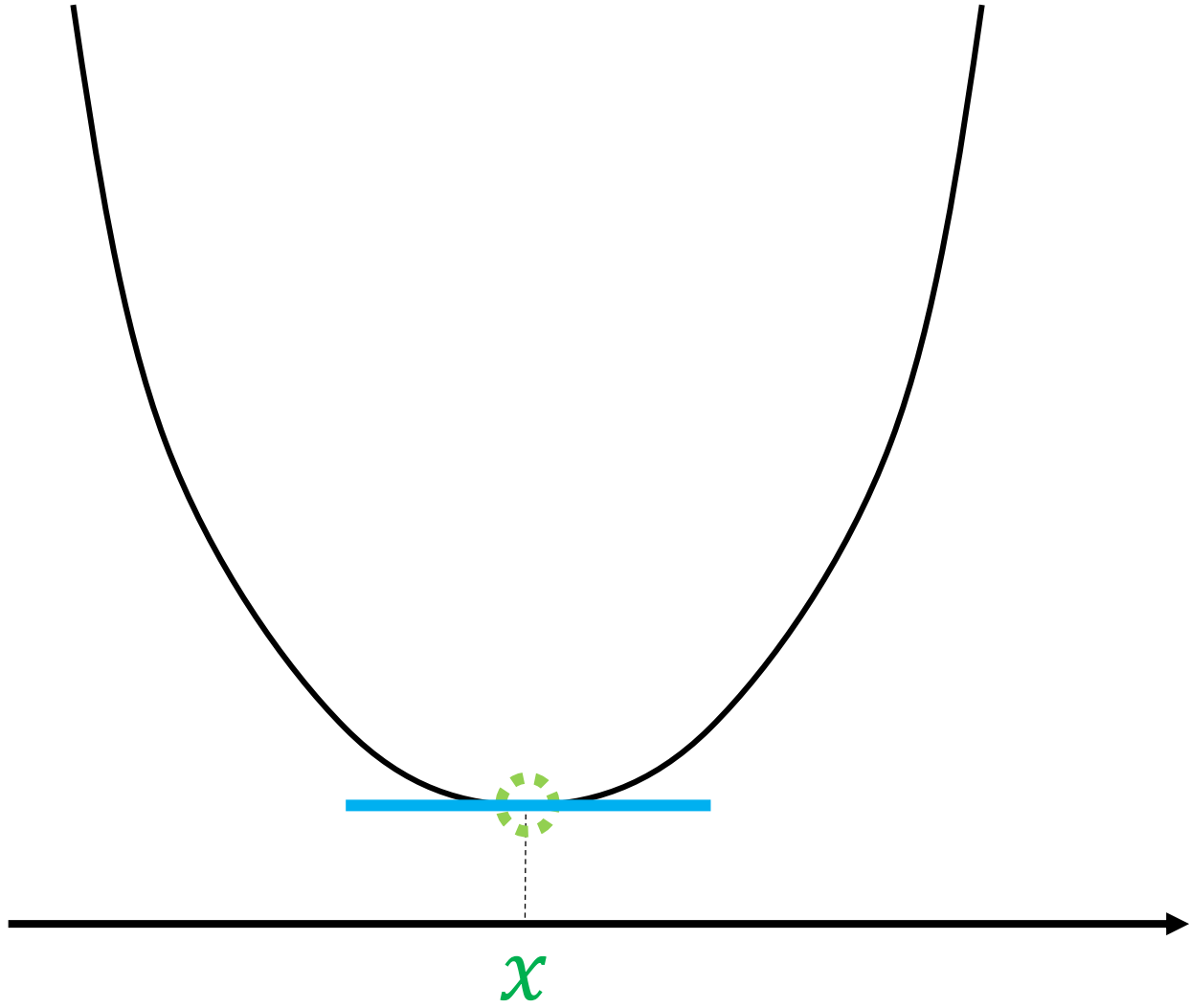
그리고 기울기가 크면 변화량도 크게 하고



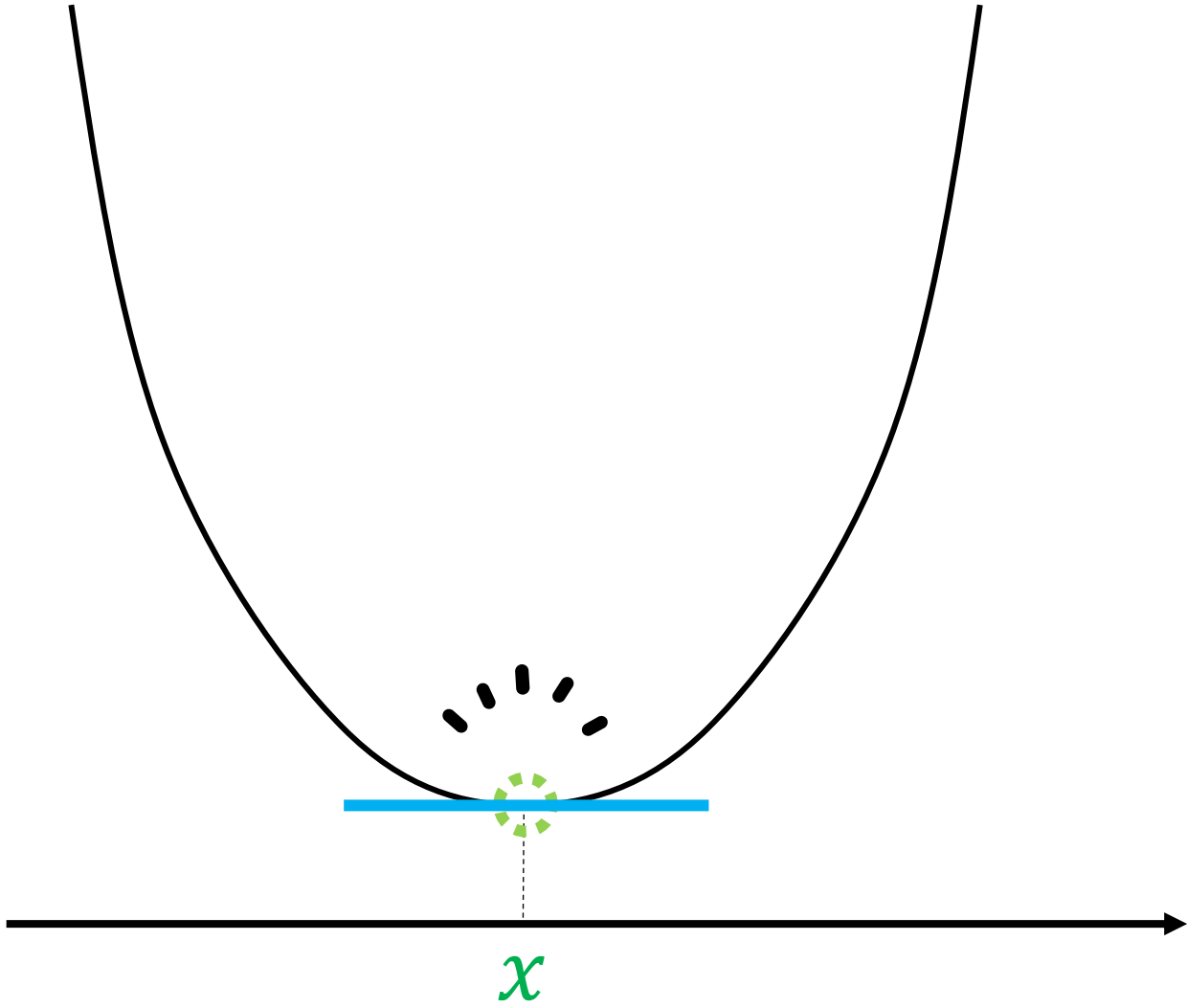
기울기가 작으면 변화량도 작게해서



기울기가 0에 수렴하는 x 를 찾으면



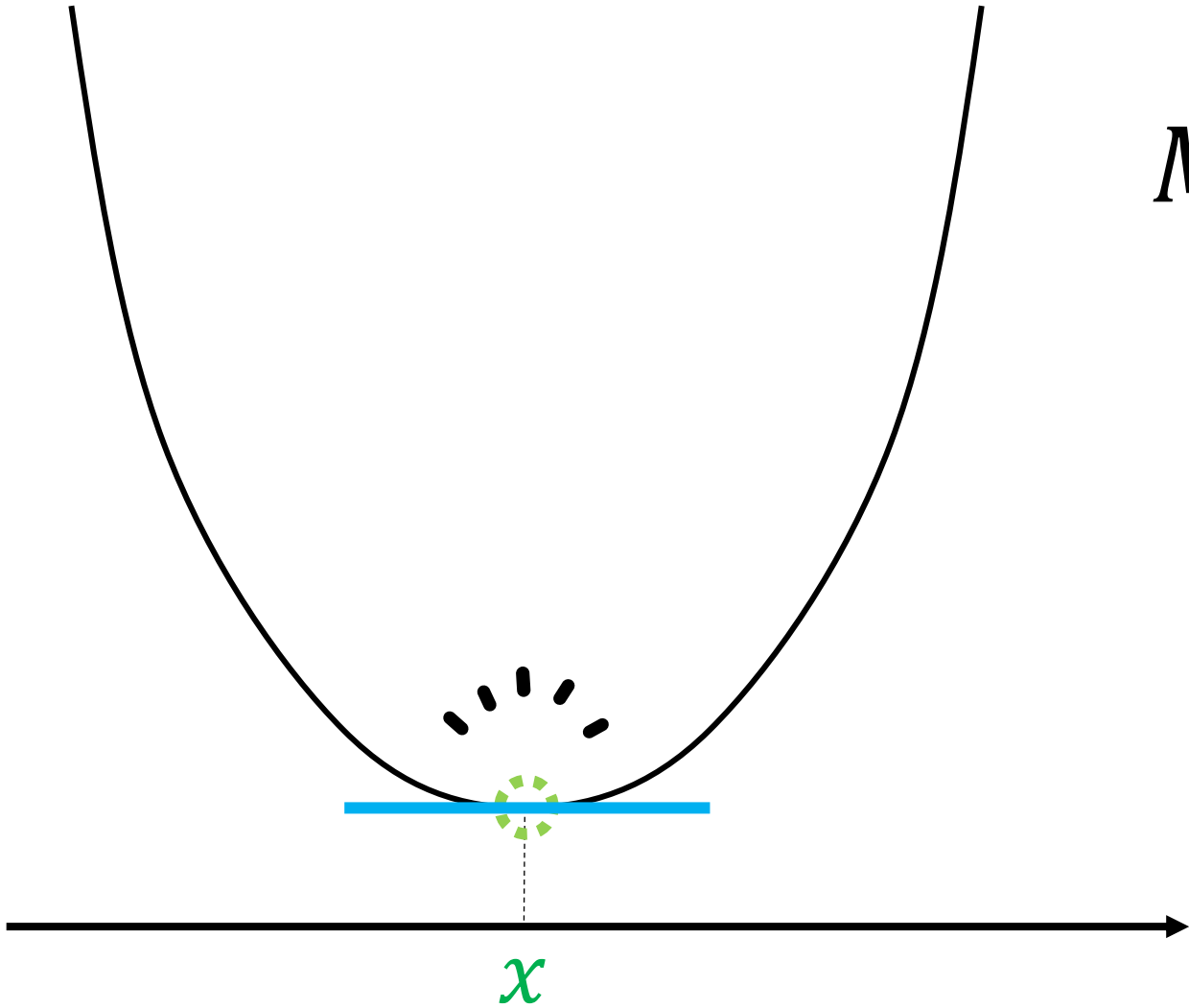
그곳이 바로 오차가 최소가 되는 지점인 것입니다



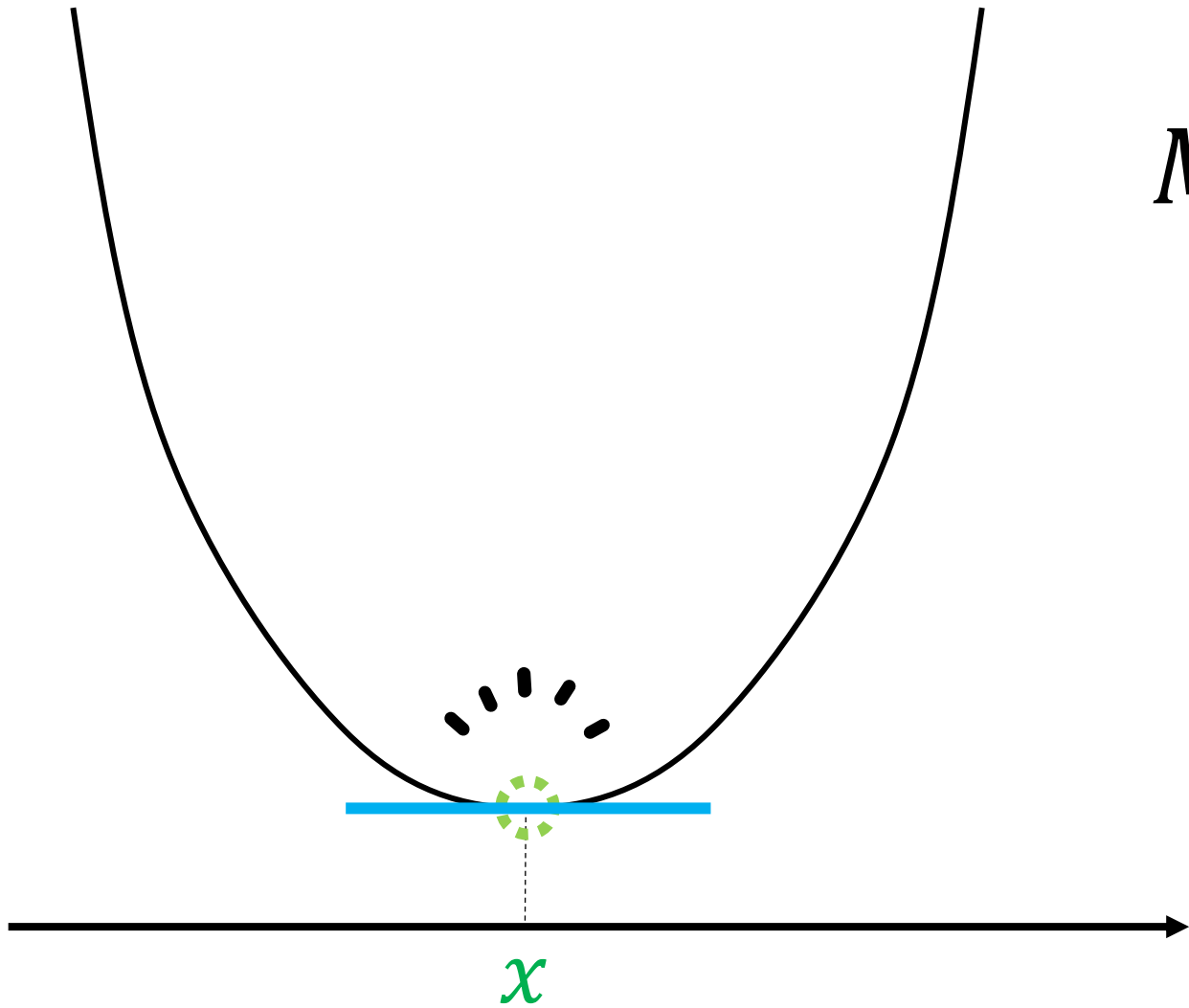
그런데 말입니다

MSE같은 경우 손실함수는 그저 2차 함수일뿐인데,

$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

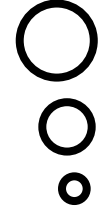


근의공식 같은걸로 최소값을 구하면 되지 않나요?



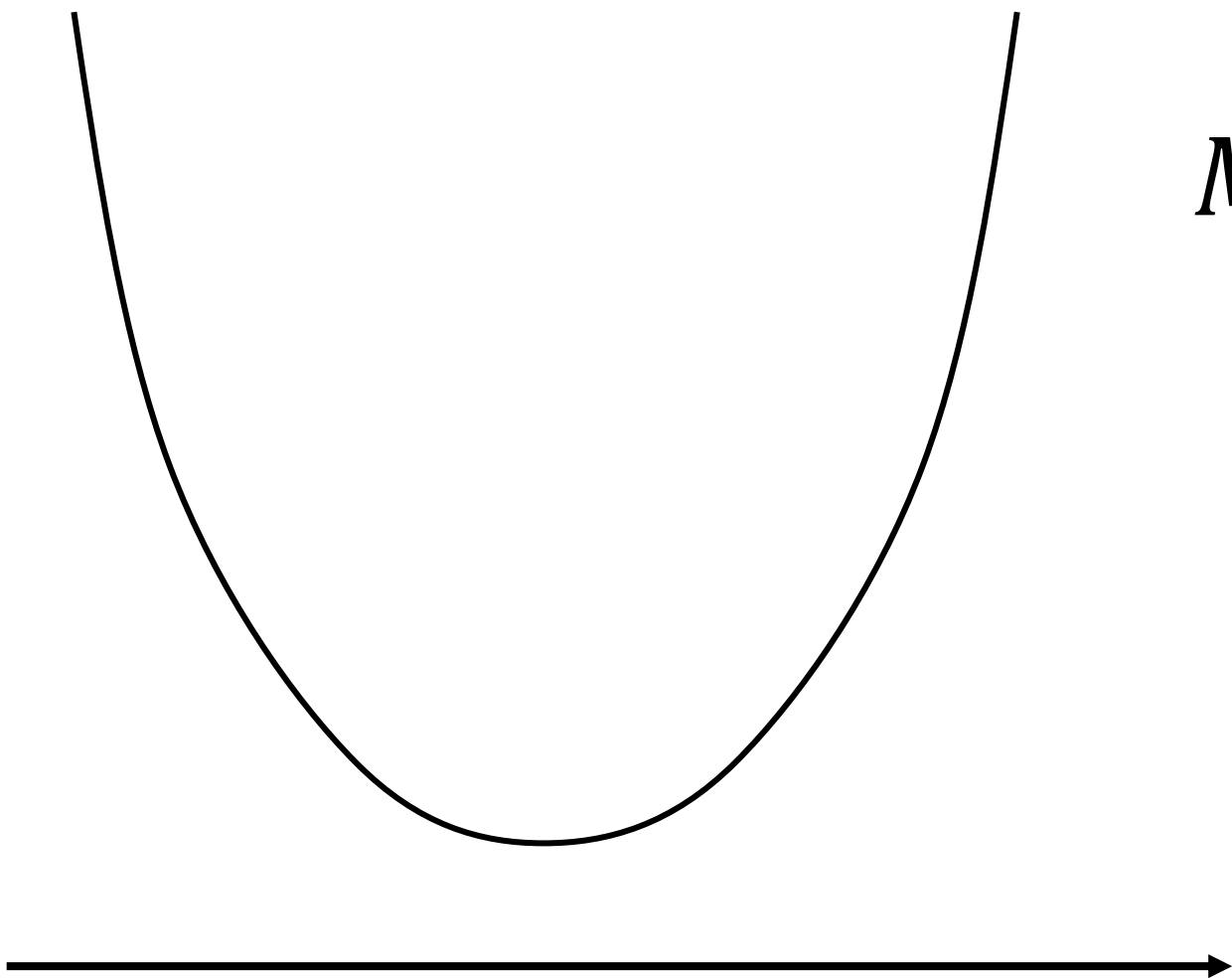
$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$



..라고 생각하실 수도 있습니다

그러나 손실함수가 늘 그렇게 단순하지는 않습니다



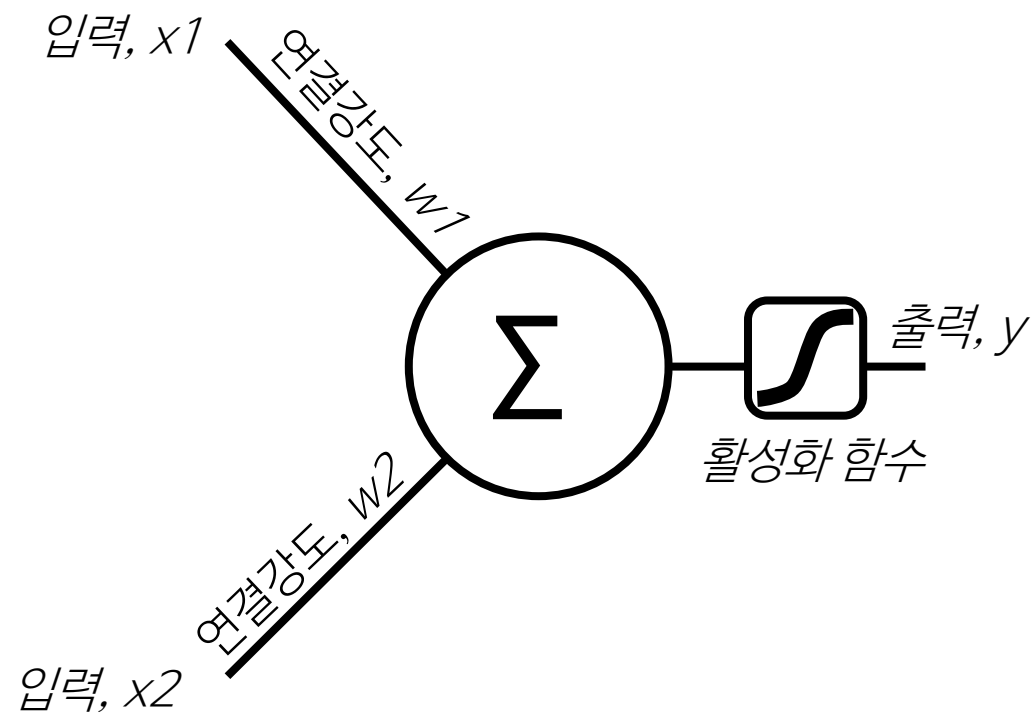
$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

대부분 딥러닝의 경우 고차원의 손실함수는 상당히 복잡합니다

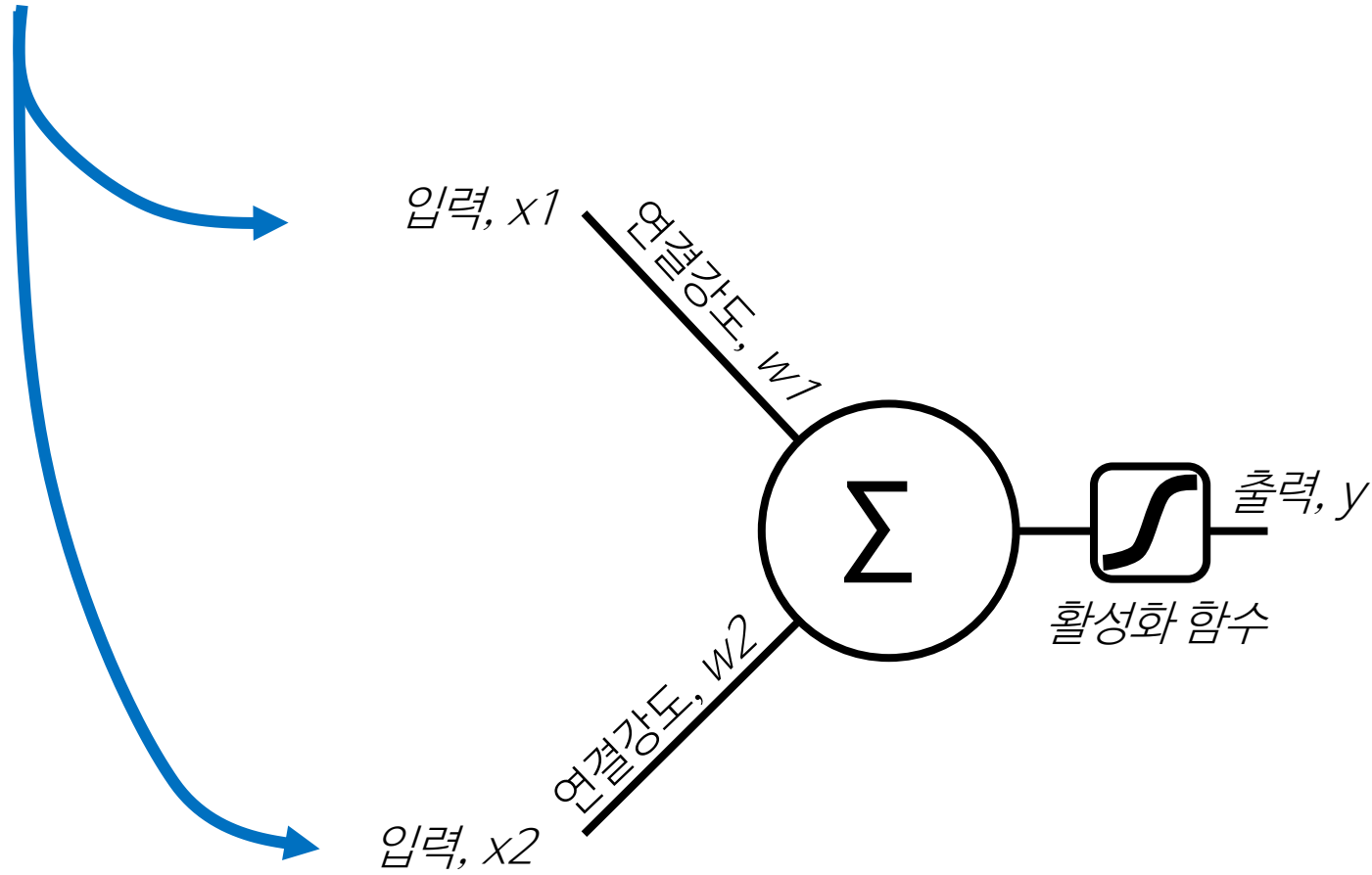


$$MSE = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2$$

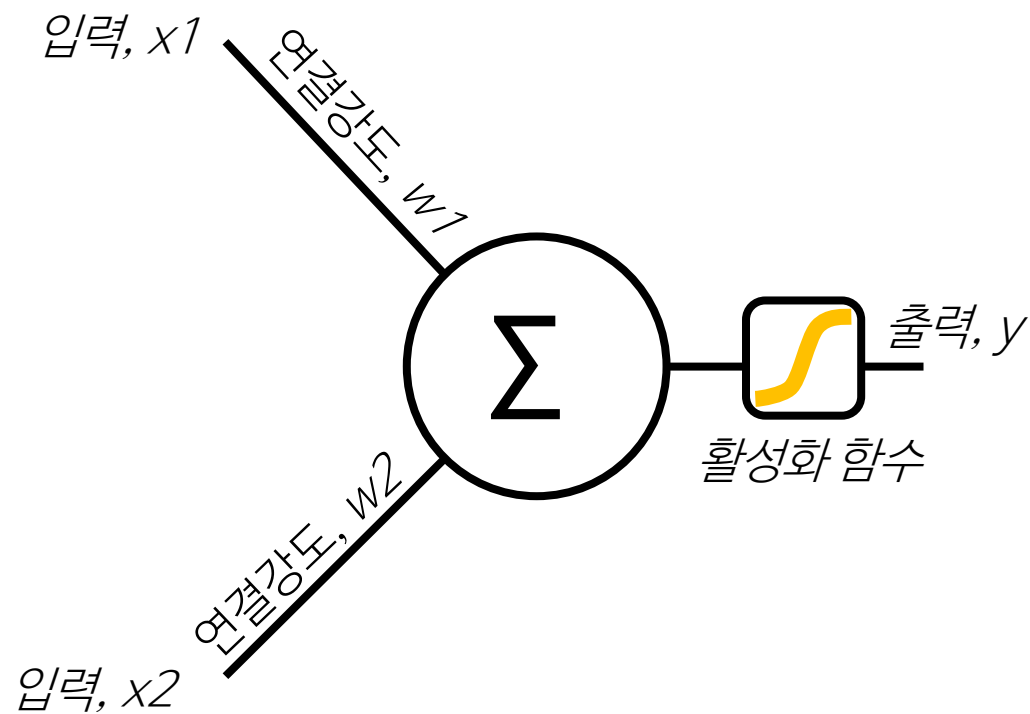
간단한 2차원 (입력이 두개) 퍼셉트론의 경우만 보더라도..



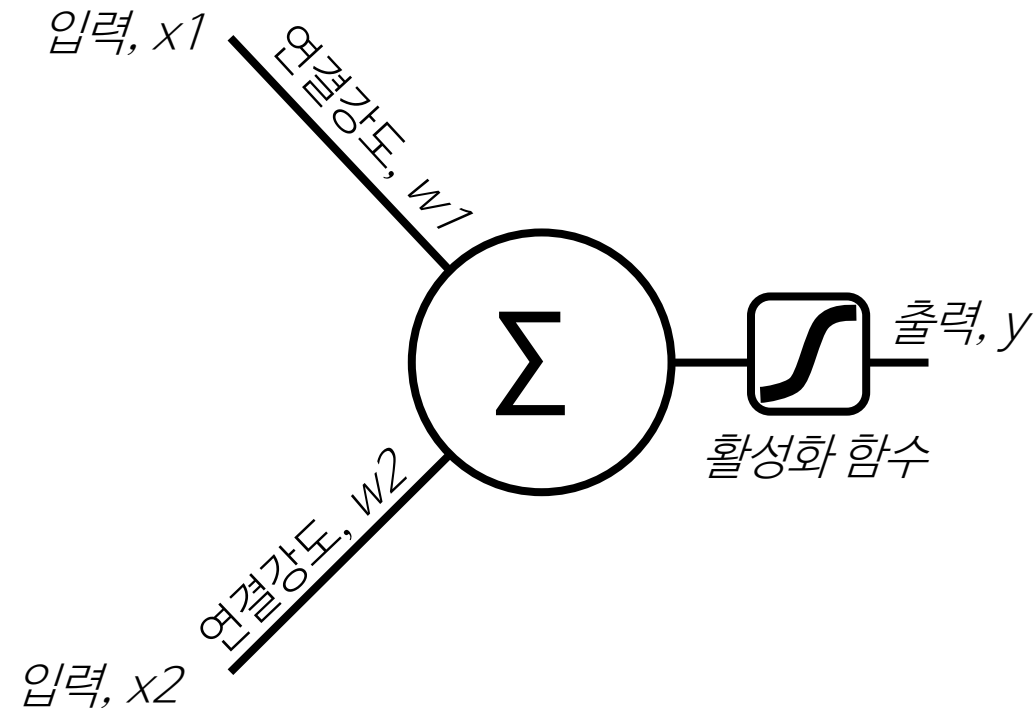
이렇게 두개의 입력을 받고



시그모이드 함수를 활성화 함수로 하는 간단한 퍼셉트론이라 할지라도

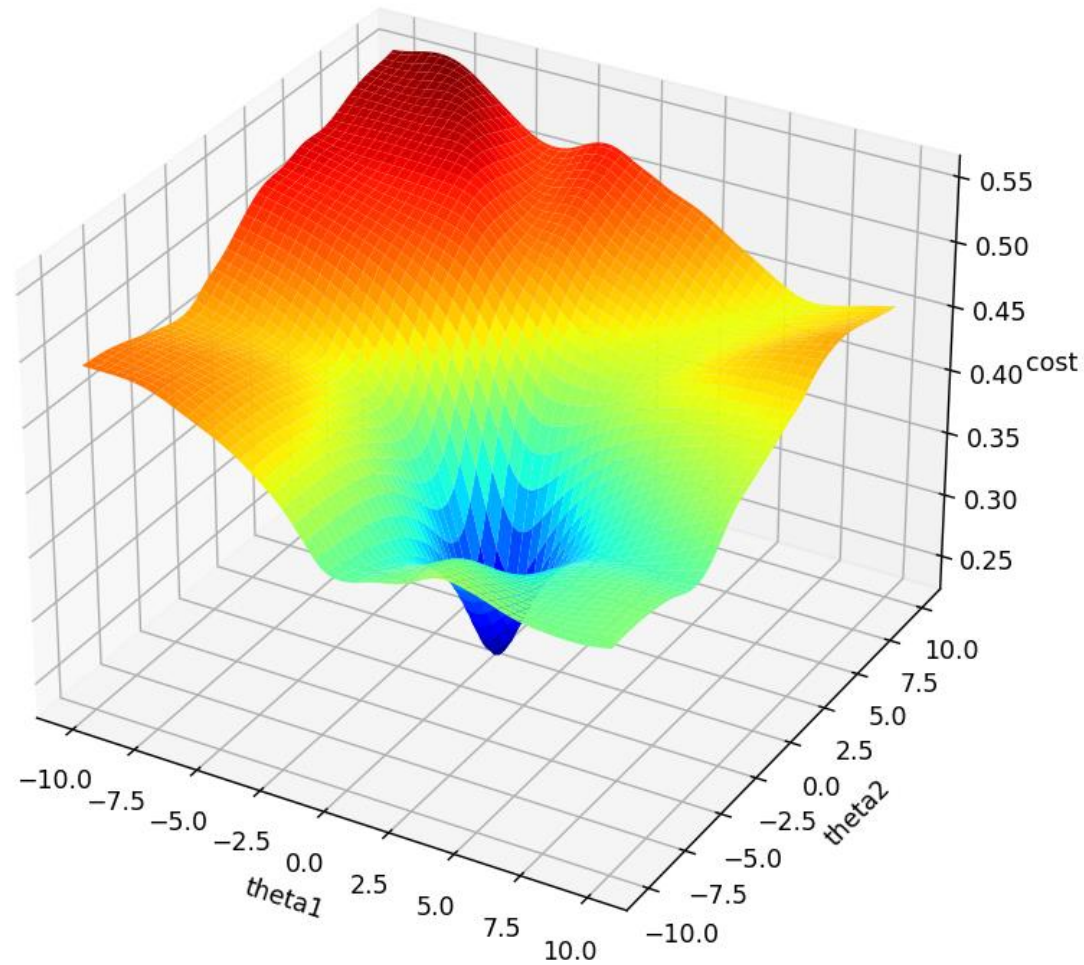


MSE 손실함수의 모습은 다음과 같습니다



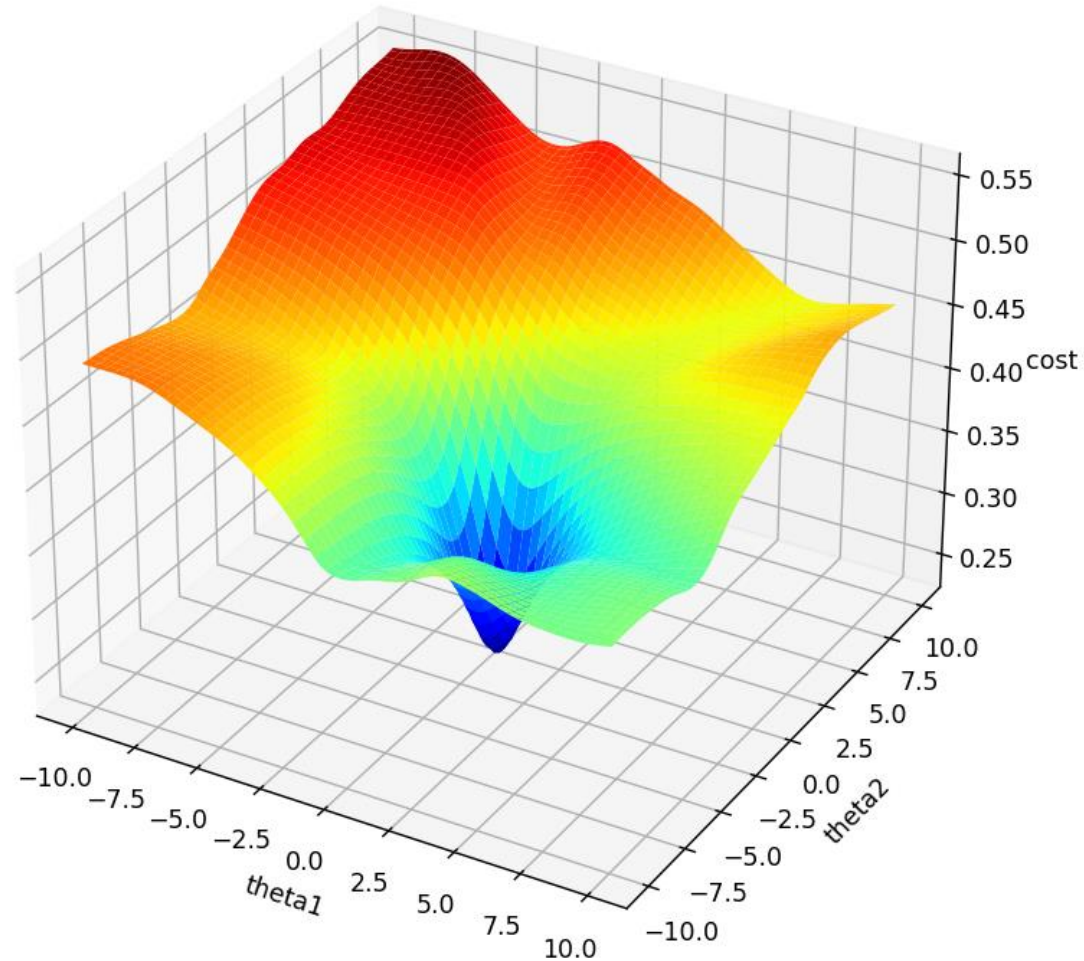
보신 바와 같이

Cost Function with Sigmoid Activation and MSE



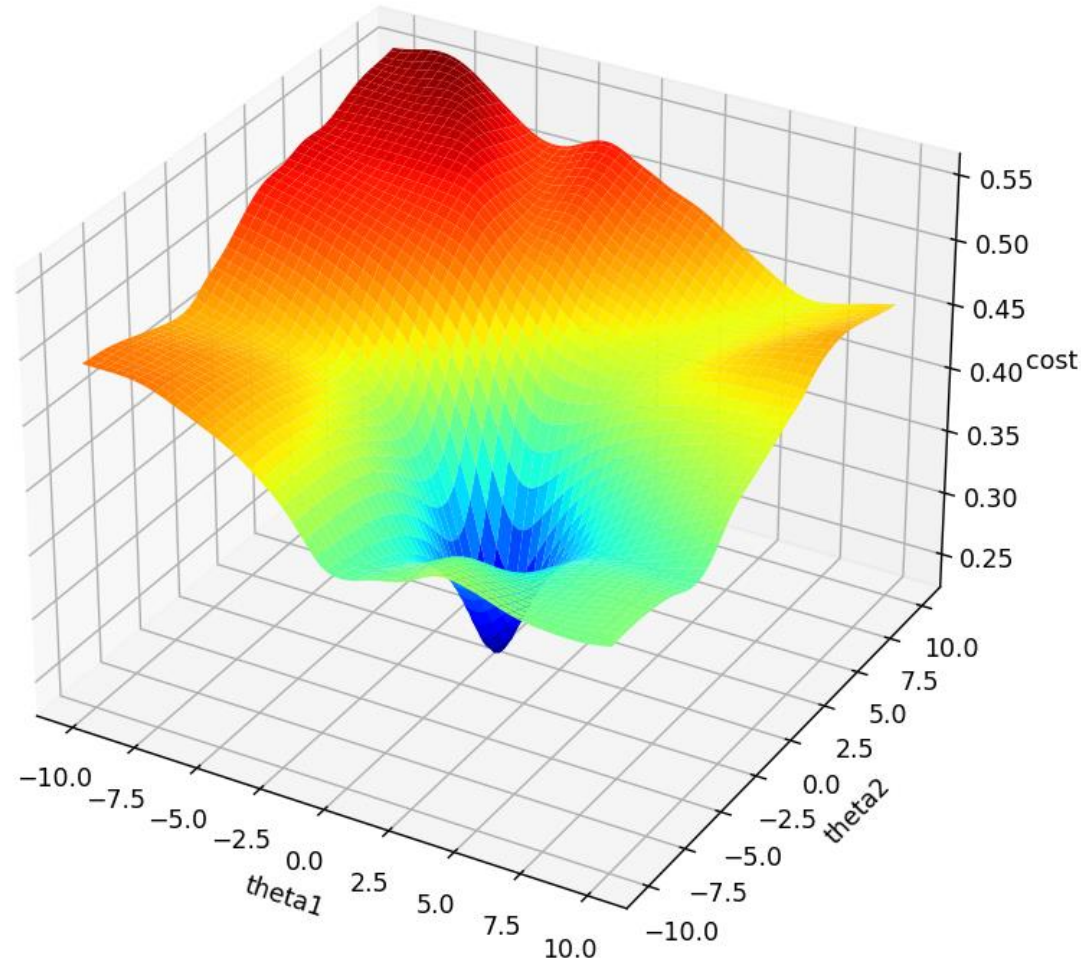
2개의 입력을 받는 간단한 퍼셉트론의 경우도

Cost Function with Sigmoid Activation and MSE



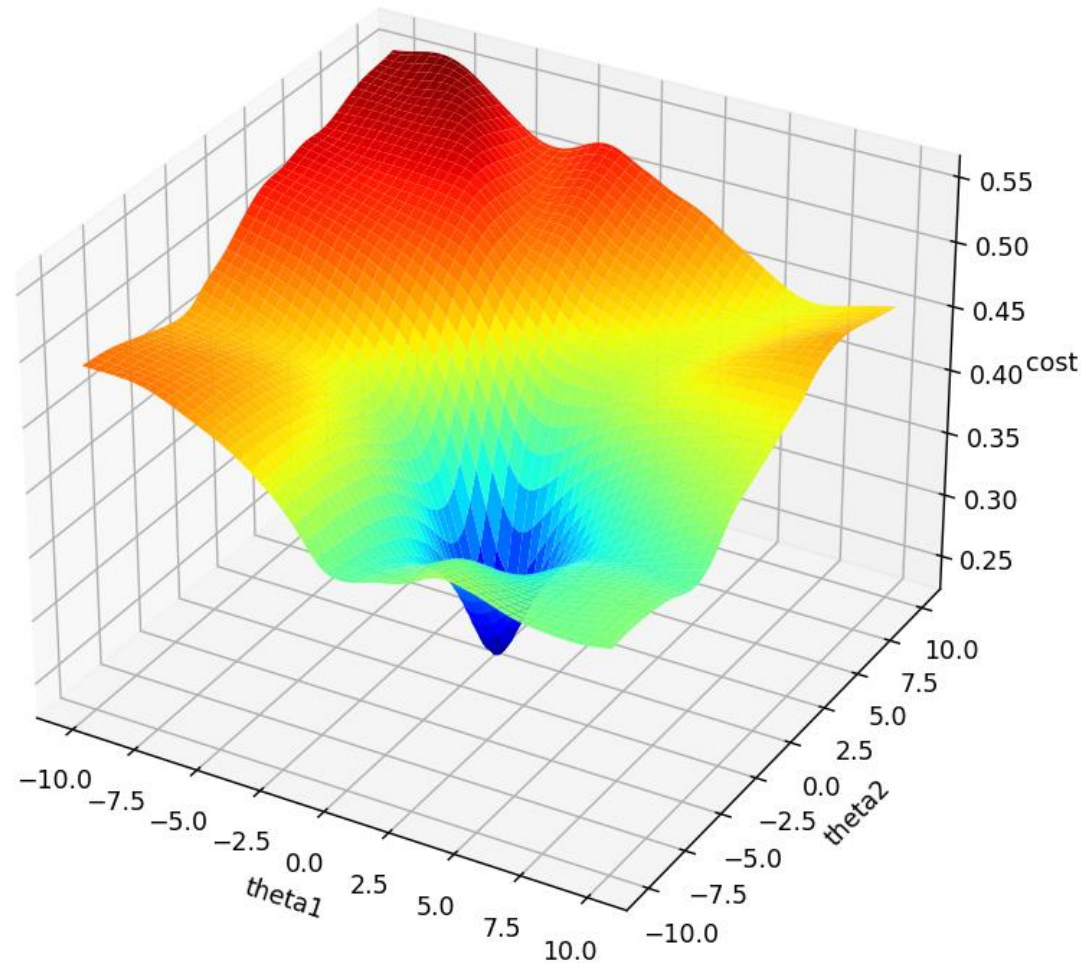
이렇게 MSE 손실함수가 복잡한데 ,

Cost Function with Sigmoid Activation and MSE



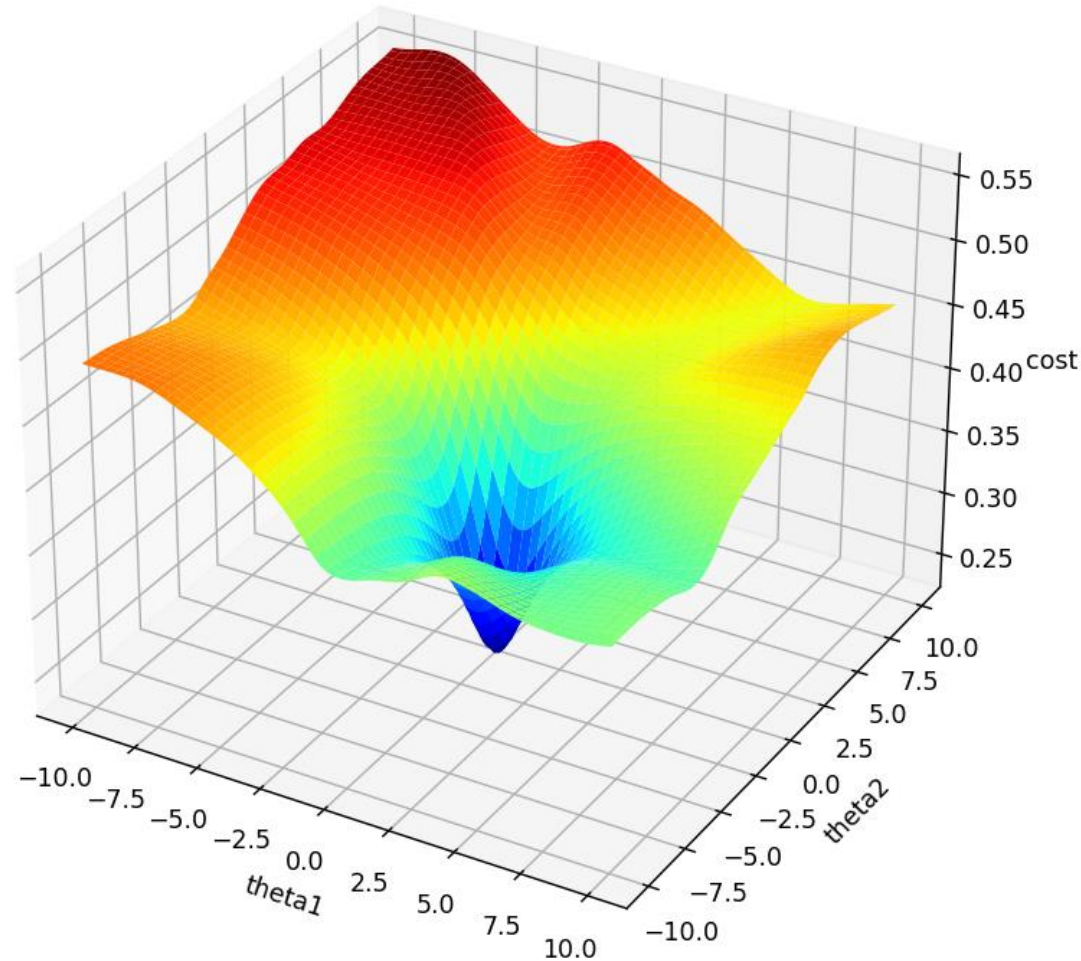
딥러닝 같은 고차원의 경우 손실함수 그래프는 상당히 복잡합니다

Cost Function with Sigmoid Activation and MSE



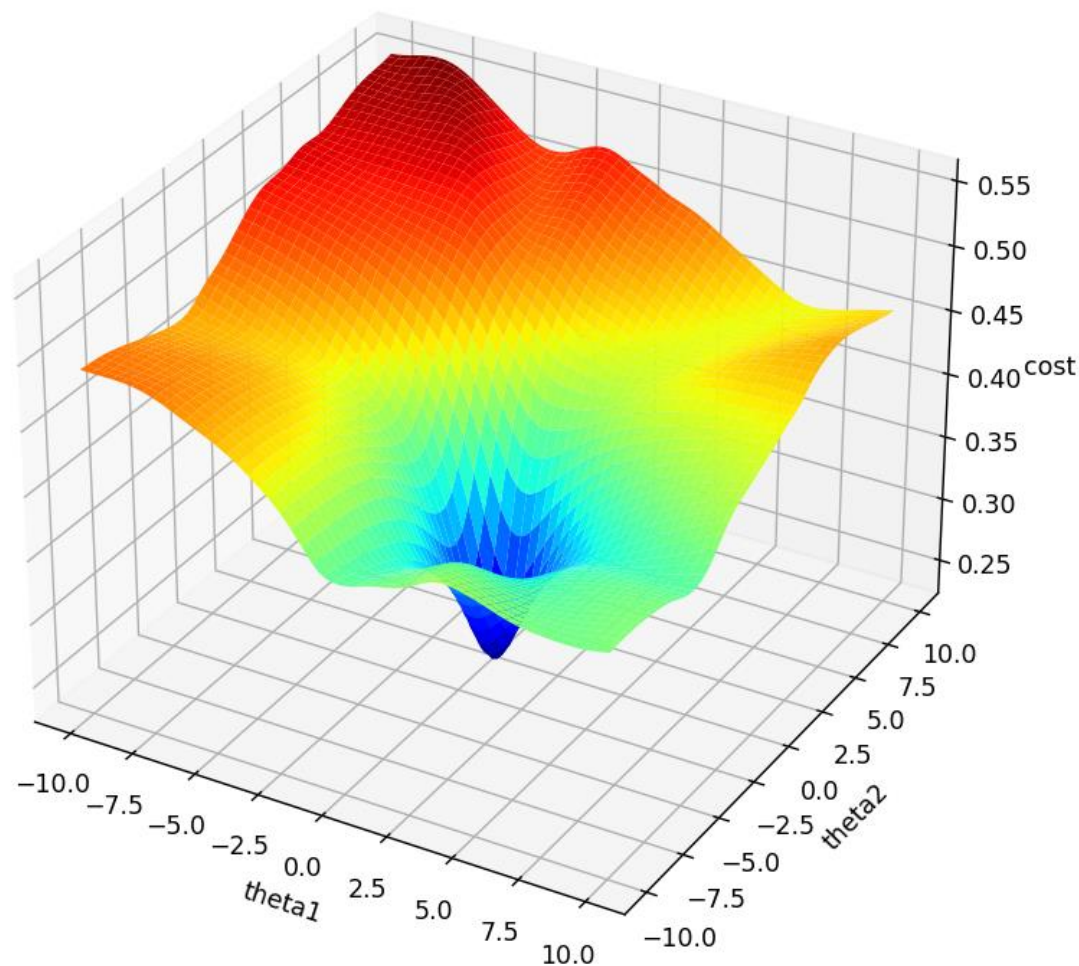
그러므로 이런 복잡한 함수의 미분 함수를 구하고

Cost Function with Sigmoid Activation and MSE



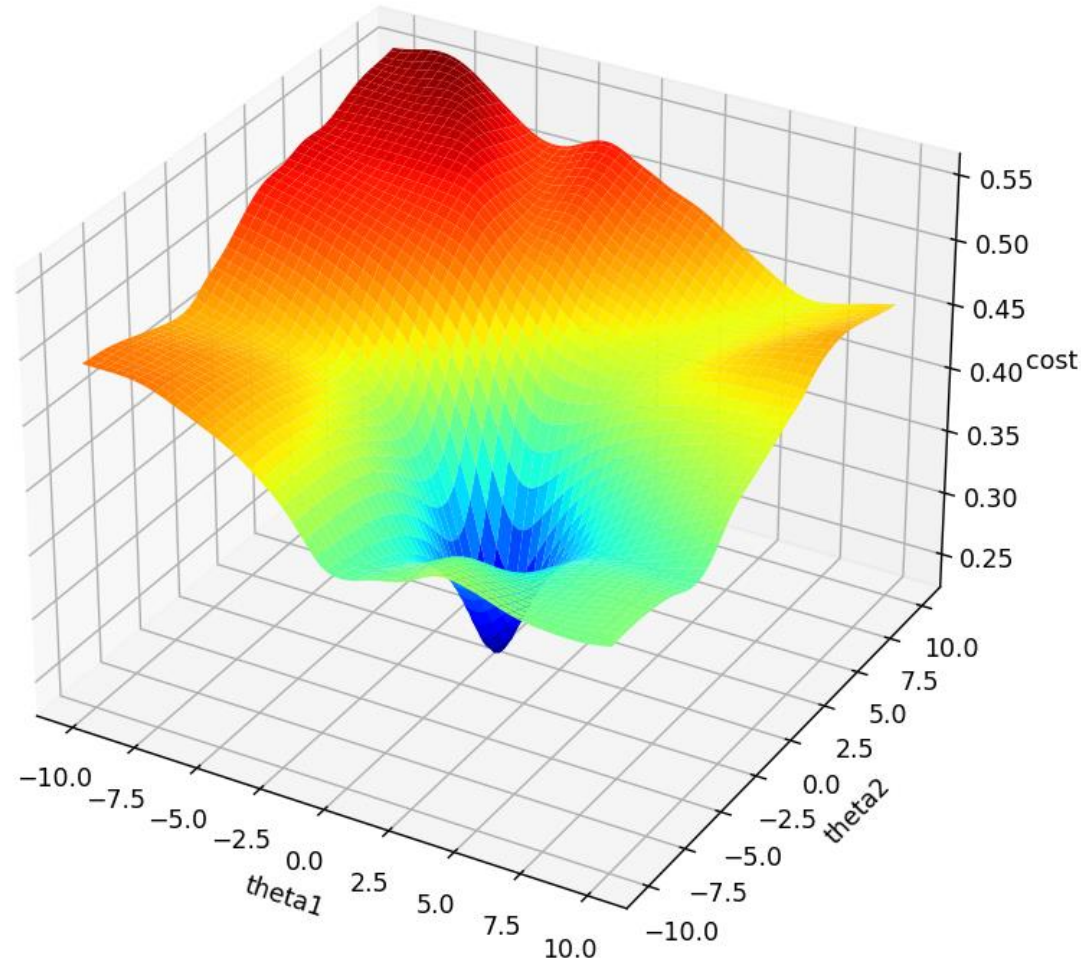
거기서 기울기가 0이 되는 점을 계산하기란 일반적으로 매우 어렵습니다

Cost Function with Sigmoid Activation and MSE



그래서 경사하강법 같은 알고리즘이 필요한 것입니다

Cost Function with Sigmoid Activation and MSE



이런 수학증명과정을 다 알필요까지는 없지만,

Taylor 근사에 의하면..

$$f(x_1 + \Delta x_1, x_2 + \Delta x_2) \approx f(x_1, x_2) + \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

이 증명에 따르면, 결국 경사하강법의 핵심은,

Taylor 근사에 의하면..

$$f(x_1 + \Delta x_1, x_2 + \Delta x_2) \approx f(x_1, x_2) + \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

$$\Delta f(x_1, x_2) \approx f(x_1 + \Delta x_1, x_2 + \Delta x_2) - f(x_1, x_2) \approx \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

손실함수 오차가 얼마나 줄어드느냐는,

Taylor 근사에 의하면..

$$f(x_1 + \Delta x_1, x_2 + \Delta x_2) \approx f(x_1, x_2) + \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

$$\Delta f(x_1, x_2) \approx f(x_1 + \Delta x_1, x_2 + \Delta x_2) - f(x_1, x_2) \approx \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

손실함수 (C)와 가중치 w_1, w_2 로 표현하면.

$$\Delta C(w_1, w_2) \approx C(w_1 + \Delta w_1, w_2 + \Delta w_2) - C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

결국 손실함수의 기울기값에 비례한다는 것을 보여줍니다

Taylor 근사에 의하면..

$$f(x_1 + \Delta x_1, x_2 + \Delta x_2) \approx f(x_1, x_2) + \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

$$\Delta f(x_1, x_2) \approx f(x_1 + \Delta x_1, x_2 + \Delta x_2) - f(x_1, x_2) \approx \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

손실함수 (C)와 가중치 w_1, w_2 로 표현하면.

$$\Delta C(w_1, w_2) \approx C(w_1 + \Delta w_1, w_2 + \Delta w_2) - C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

$$\Delta C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

이것을 보기 편하게 기호로 표현하면

Taylor 근사에 의하면..

$$f(x_1 + \Delta x_1, x_2 + \Delta x_2) \approx f(x_1, x_2) + \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

$$\Delta f(x_1, x_2) \approx f(x_1 + \Delta x_1, x_2 + \Delta x_2) - f(x_1, x_2) \approx \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

손실함수 (C)와 가중치 w_1, w_2 로 표현하면.

$$\Delta C(w_1, w_2) \approx C(w_1 + \Delta w_1, w_2 + \Delta w_2) - C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

$$\Delta C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

$$\Delta C(w_1, w_2) \approx \begin{bmatrix} \frac{\partial C}{\partial w_1} & \frac{\partial C}{\partial w_2} \end{bmatrix} \begin{bmatrix} \Delta w_1 \\ \Delta w_2 \end{bmatrix}$$

다음과 같이 정리할 수 있습니다

Taylor 근사에 의하면..

$$f(x_1 + \Delta x_1, x_2 + \Delta x_2) \approx f(x_1, x_2) + \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

$$\Delta f(x_1, x_2) \approx f(x_1 + \Delta x_1, x_2 + \Delta x_2) - f(x_1, x_2) \approx \frac{\partial f}{\partial x_1} \Delta x_1 + \frac{\partial f}{\partial x_2} \Delta x_2$$

손실함수 (C)와 가중치 w_1, w_2 로 표현하면.

$$\Delta C(w_1, w_2) \approx C(w_1 + \Delta w_1, w_2 + \Delta w_2) - C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

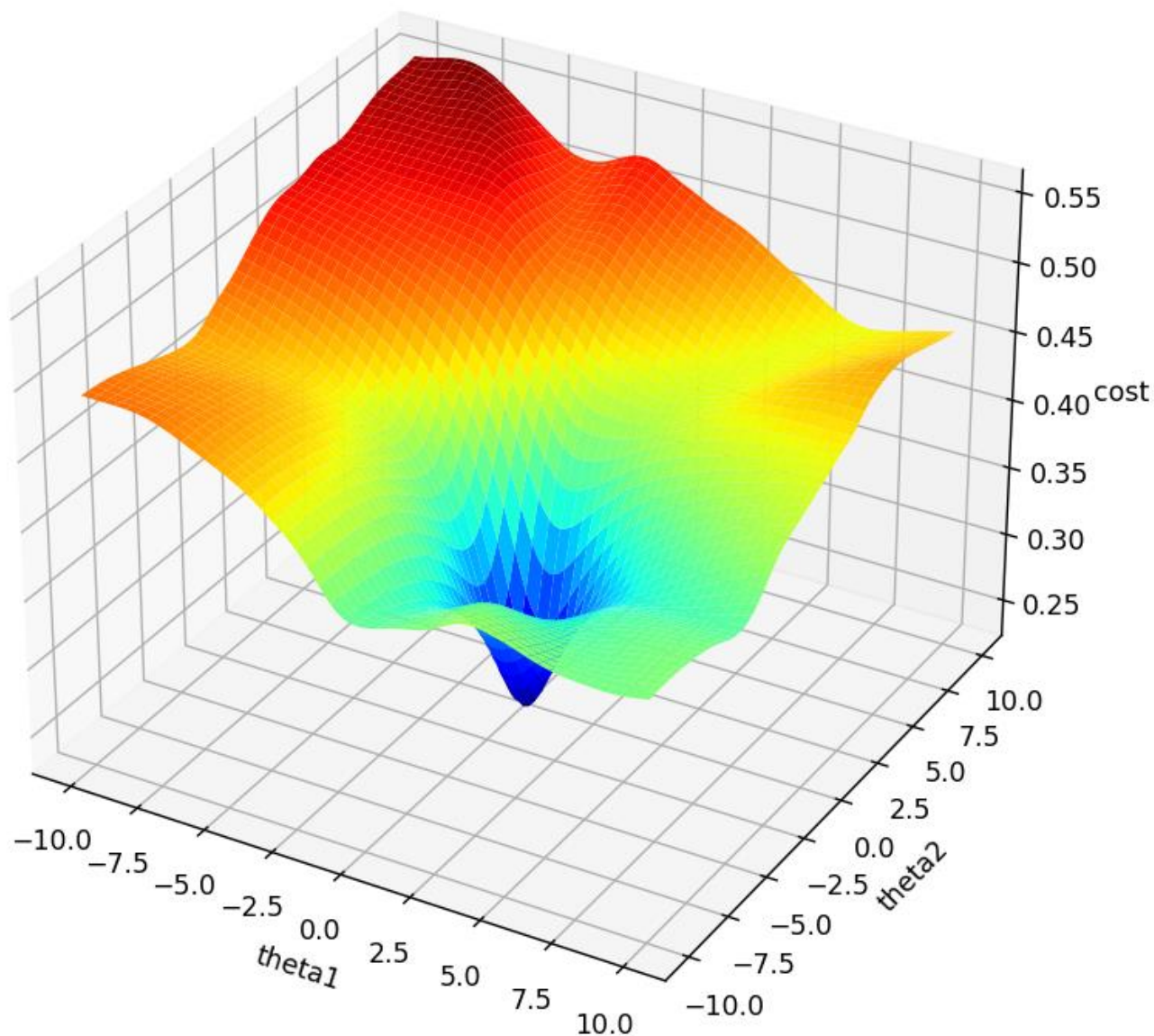
$$\Delta C(w_1, w_2) \approx \frac{\partial C}{\partial w_1} \Delta w_1 + \frac{\partial C}{\partial w_2} \Delta w_2$$

$$\Delta C \approx \nabla C \cdot \Delta W$$

단언하건데, 이런 도출과정을 몰라도 경사하강법을 이해하는데 아무런 지장이 없습니다

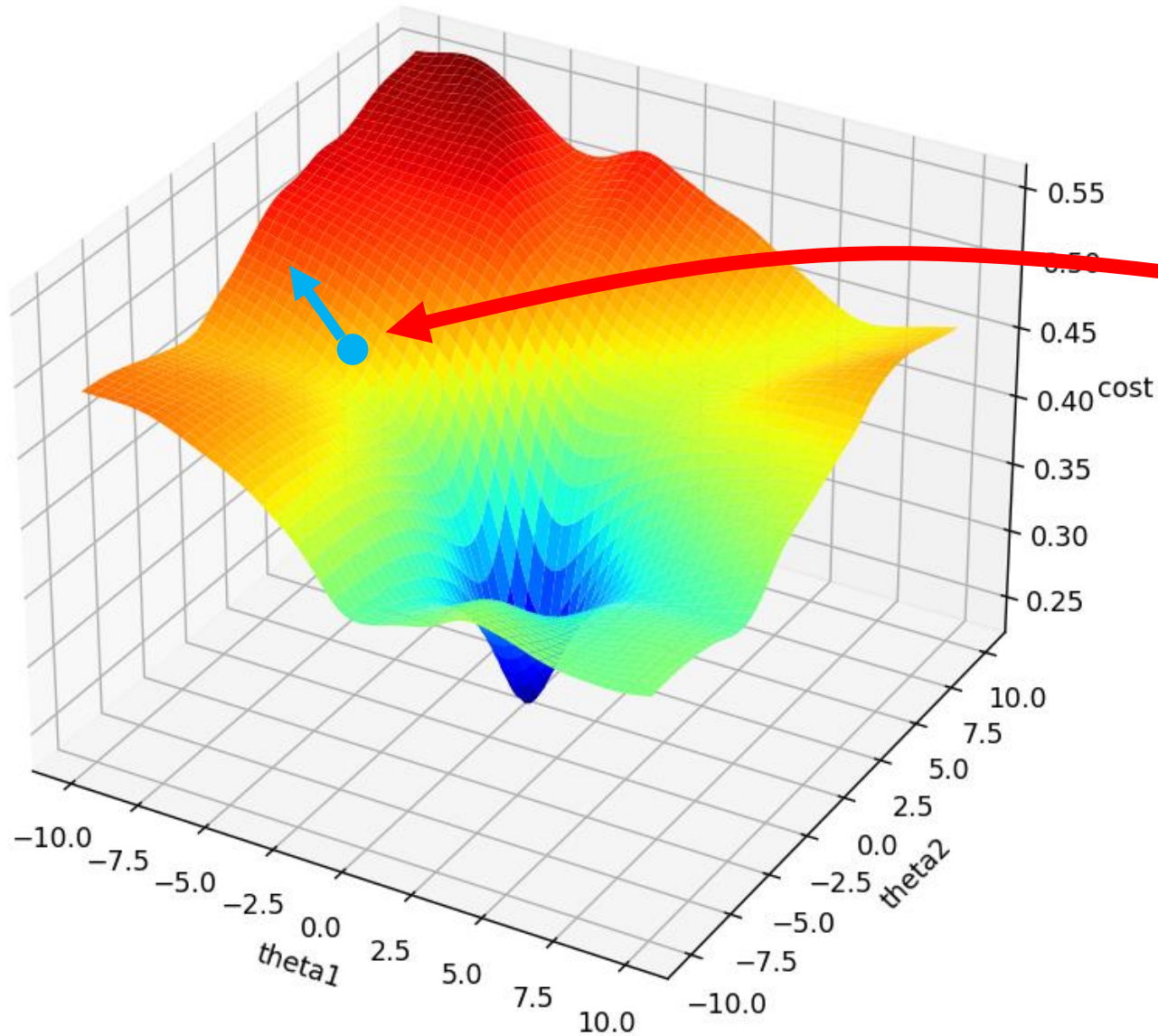
$$\Delta C \approx \nabla C \cdot \Delta W$$

하지만 이 마지막 공식은 경사하강법의 개념을 잘 표현하고 있습니다



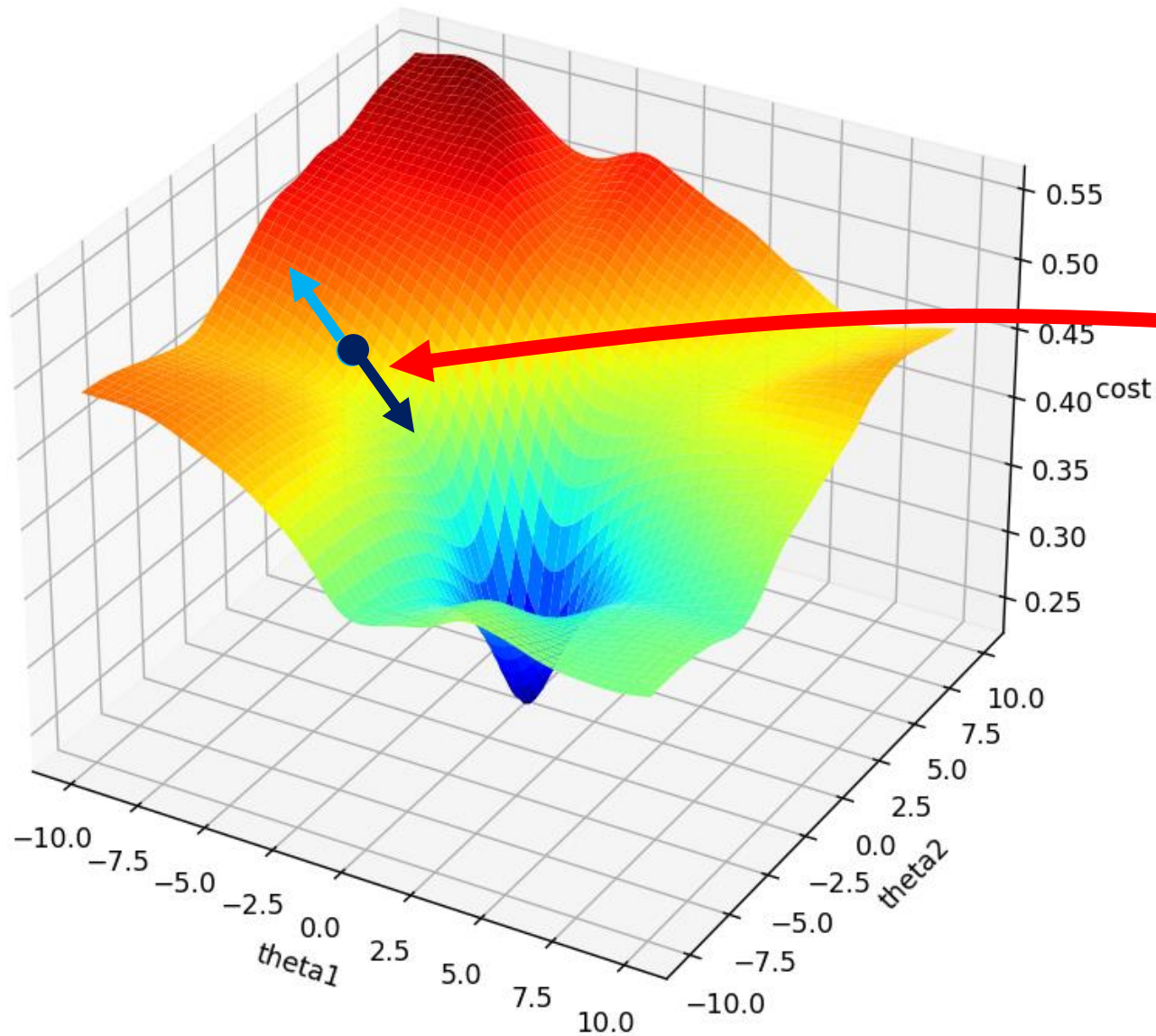
$$\Delta C \approx \nabla C \cdot \Delta W$$

이 ∇C 는 해당 지점에서의 기울기를 뜻합니다



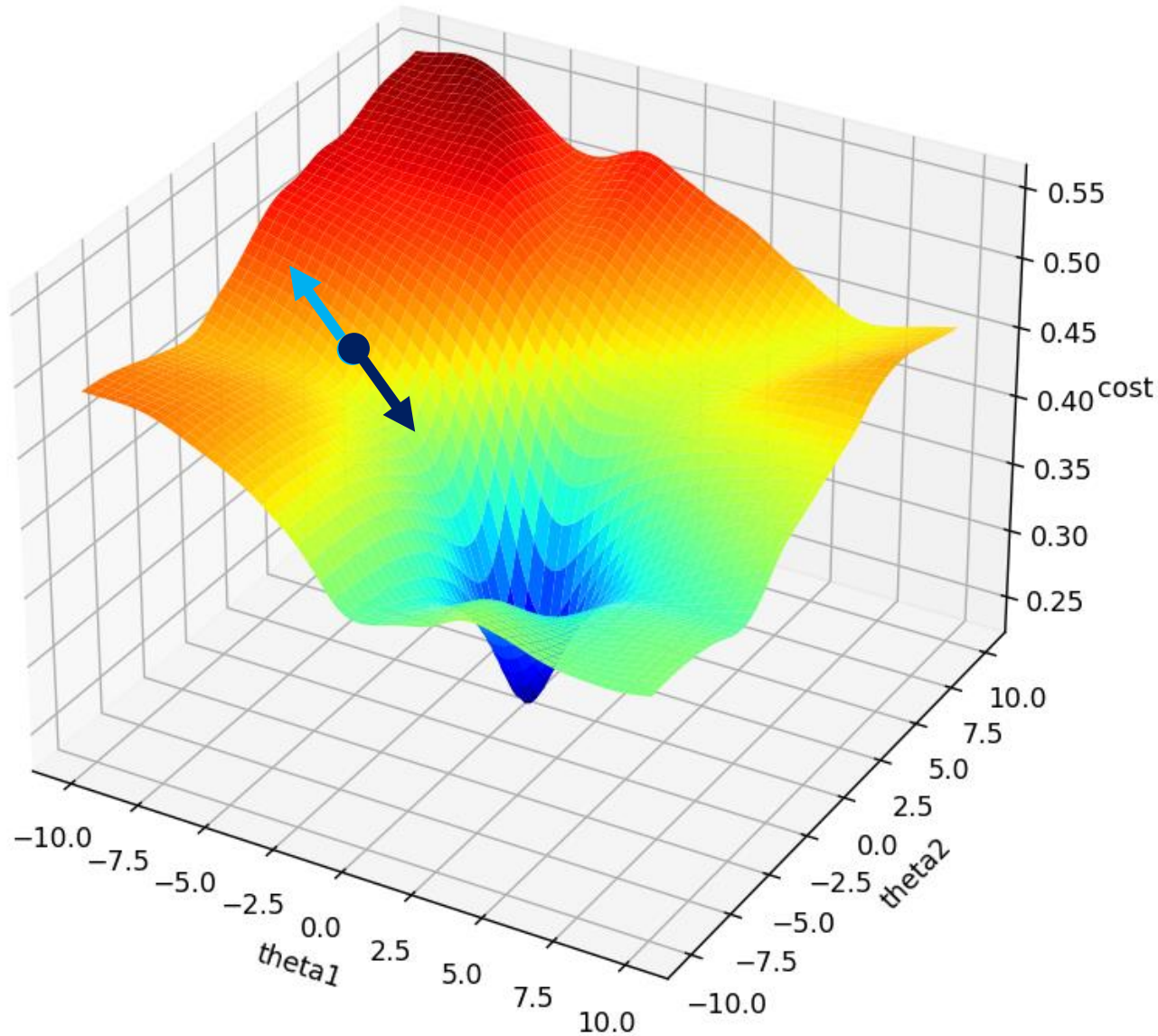
$$\Delta C \approx \nabla C \cdot \Delta W$$

그러므로 ΔW 의 값은 기울기인 ∇C 와 크기는 같고 방향이 반대인



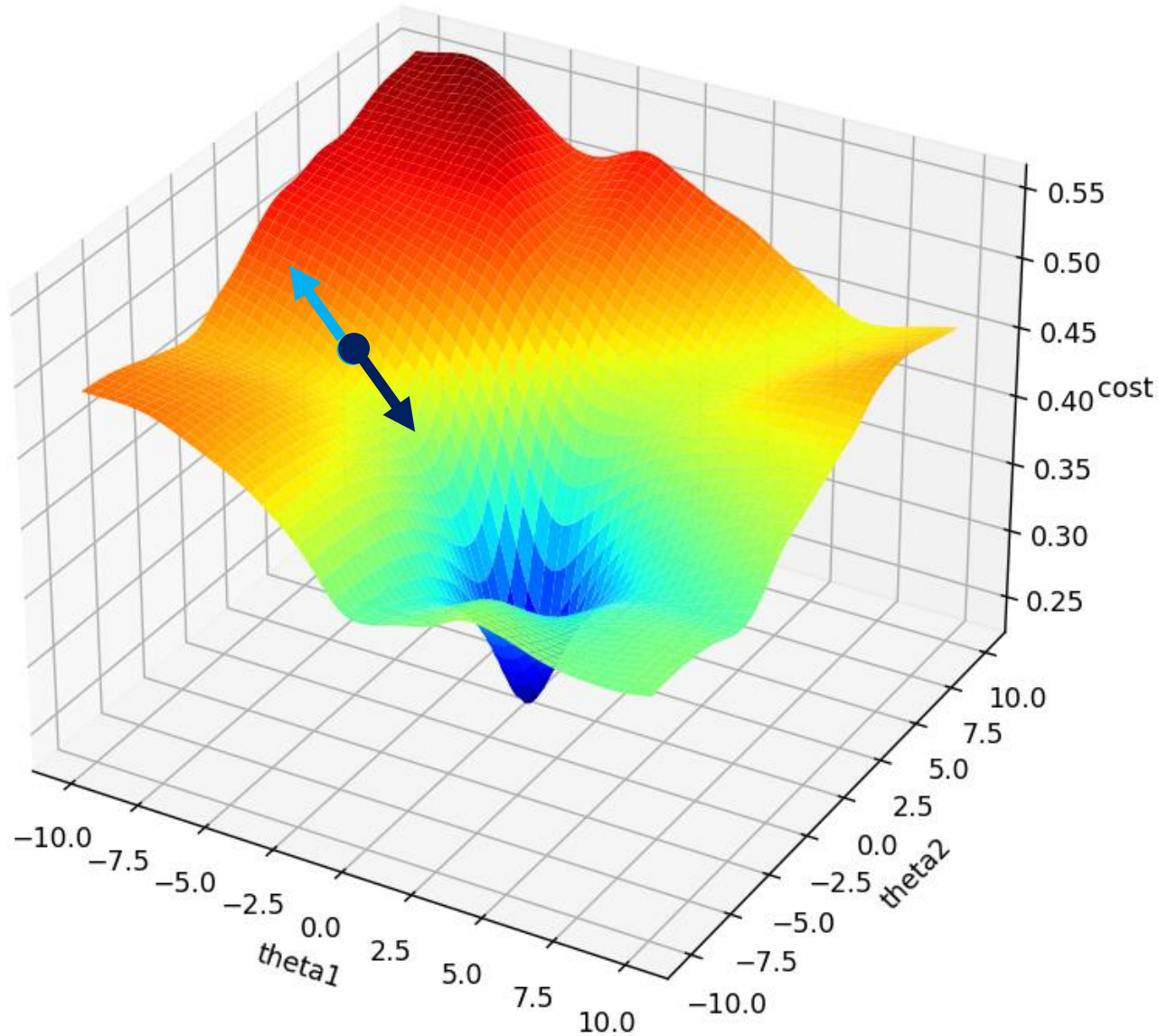
$$\Delta C \approx \nabla C \cdot \Delta W$$

– ∇C 로 생각해 볼 수가 있습니다



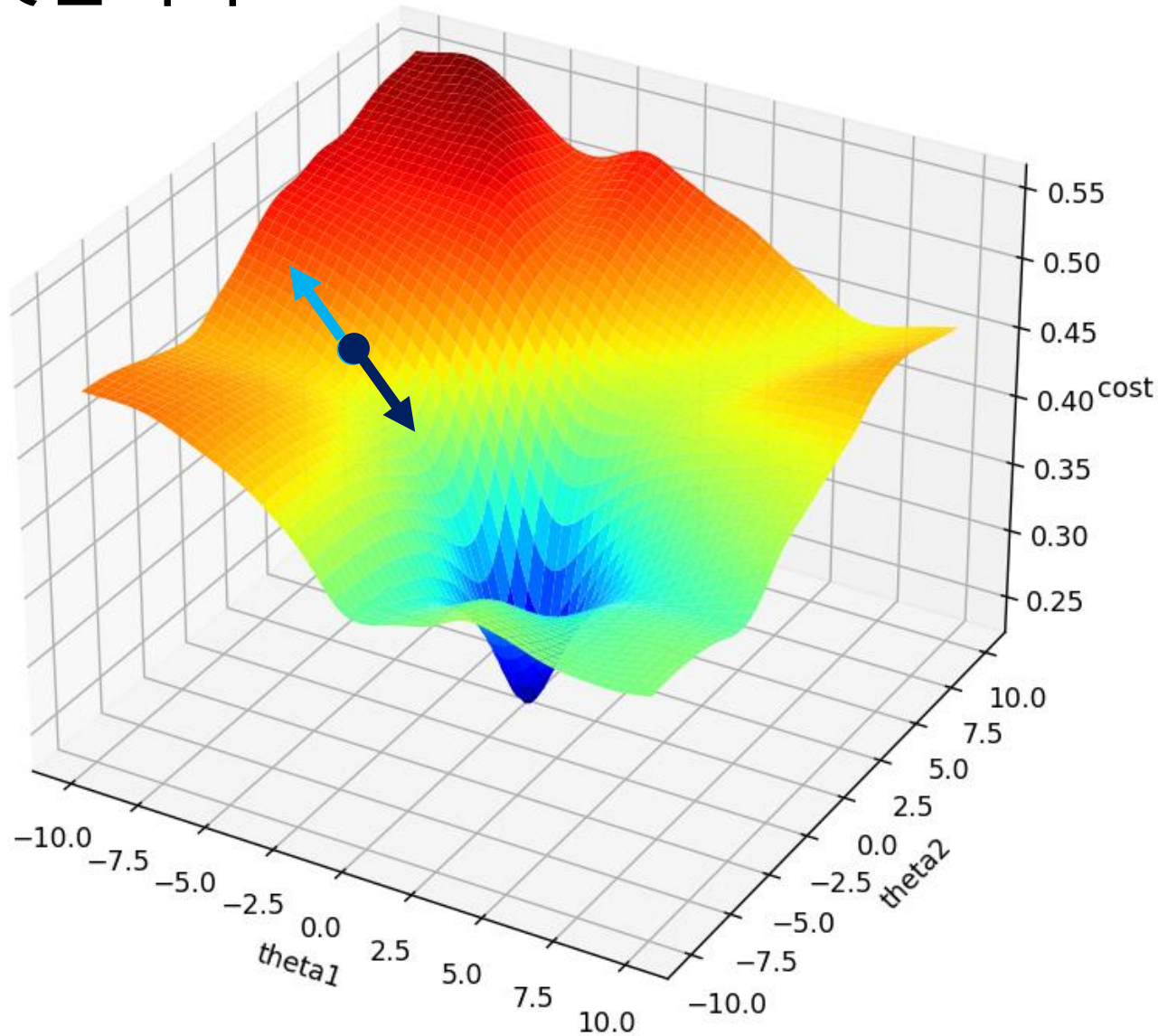
$$\begin{aligned}\Delta C &\approx \nabla C \cdot \Delta W \\ &\approx \nabla C \cdot (-\nabla C)\end{aligned}$$

왜냐하면 ΔW 의 값을 $-\nabla C$ 로 할때 ΔC 가 제일 커지기 때문입니다



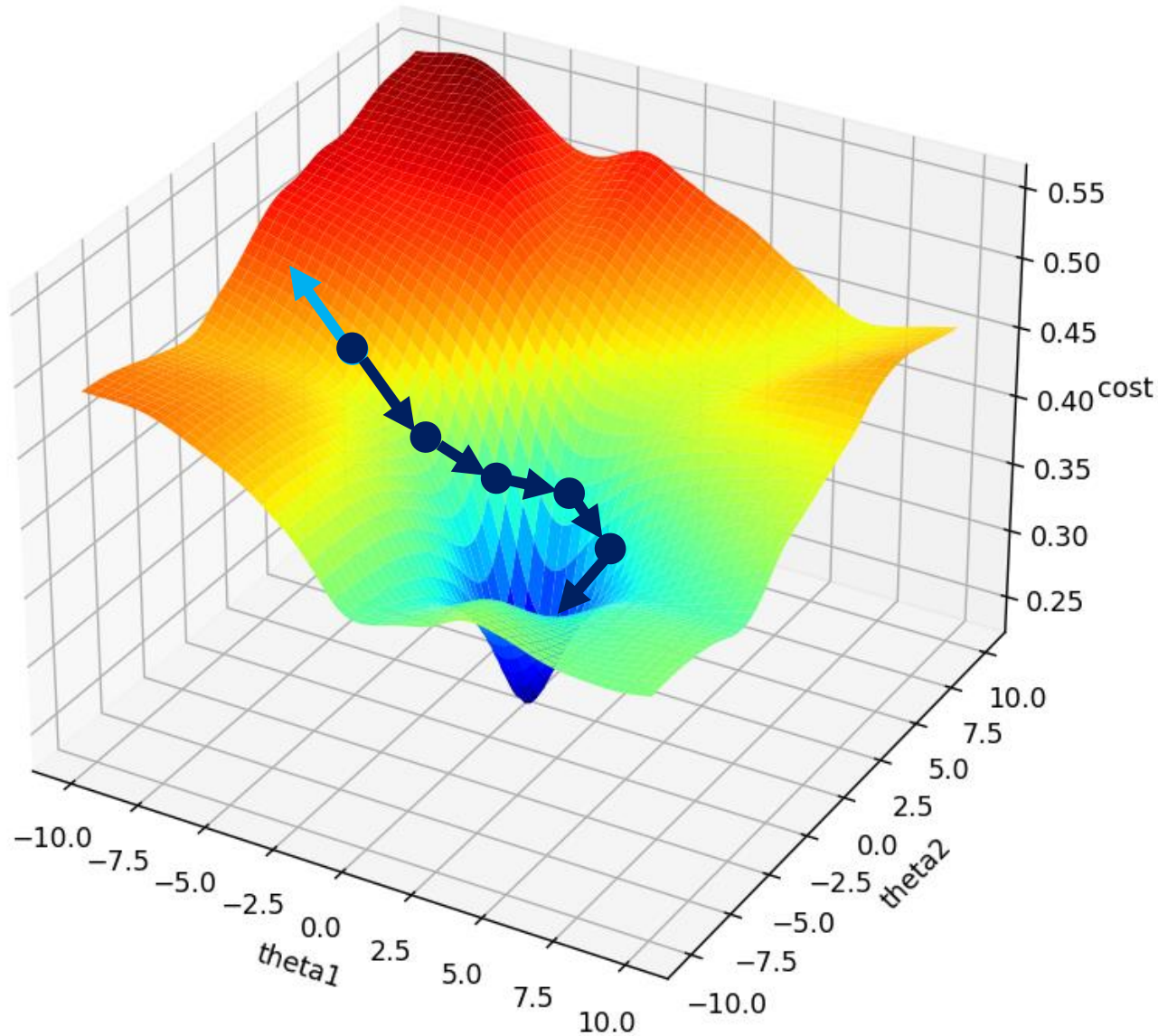
$$\begin{aligned}\Delta C &\approx \nabla C \cdot \Delta W \\ &\approx \nabla C \cdot (-\nabla C)\end{aligned}$$

이렇게 ΔW 의 값이 기울기(경사)로 결정된다 하여 경사하강법이라 부르는 것입니다



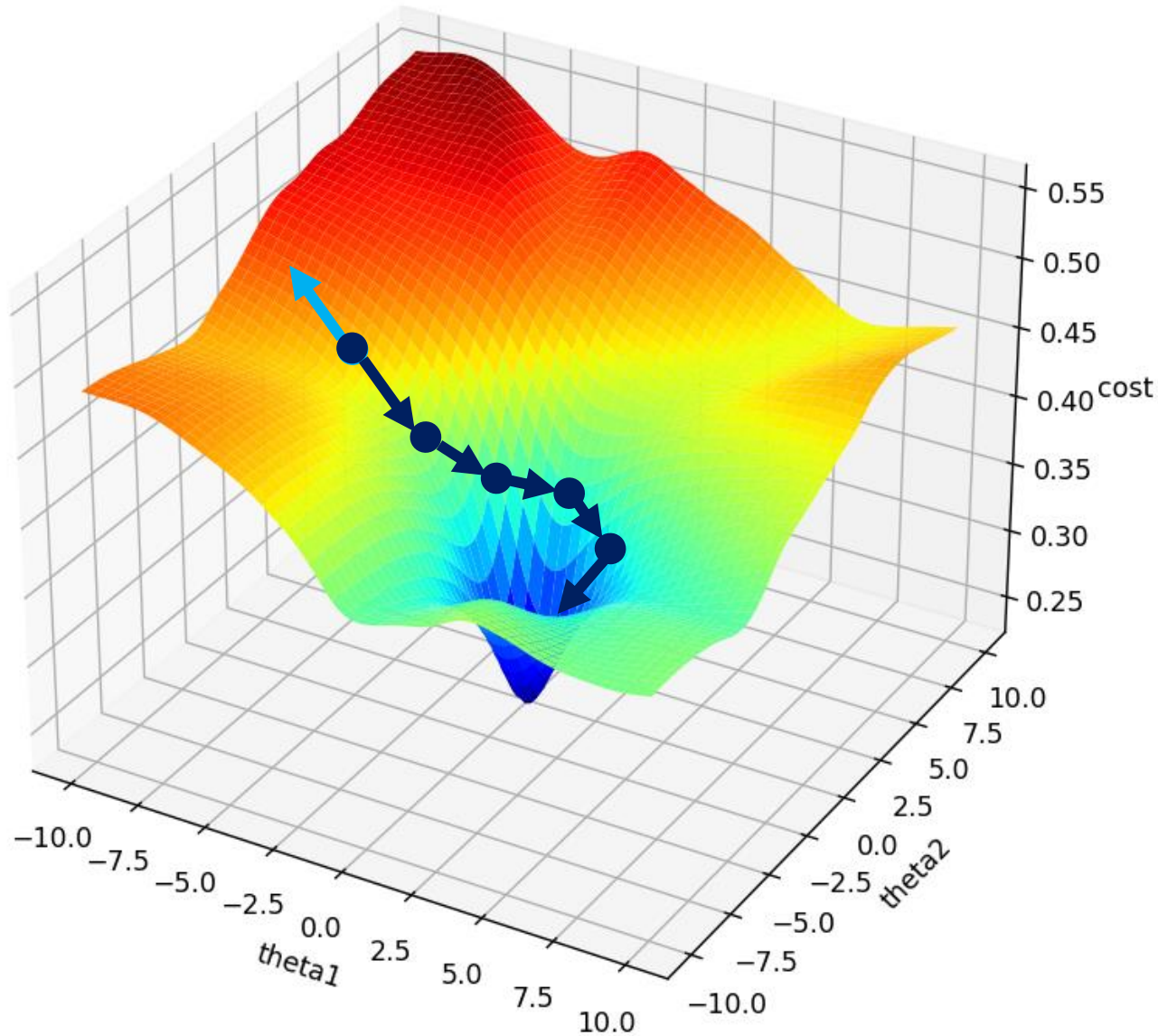
$$\begin{aligned}\Delta C &\approx \nabla C \cdot \Delta W \\ &\approx \nabla C \cdot (-\nabla C)\end{aligned}$$

그렇게 ΔW 의 값을 조절하여 최저점까지 점진적으로 내려가는 것을



$$\begin{aligned}\Delta C &\approx \nabla C \cdot \Delta W \\ &\approx \nabla C \cdot (-\nabla C)\end{aligned}$$

경사하강법의 핵심이라고 할 수가 있겠습니다



$$\begin{aligned}\Delta C &\approx \nabla C \cdot \Delta W \\ &\approx \nabla C \cdot (-\nabla C)\end{aligned}$$

마지막으로, 경사하강법을 이용한 연결가중치 업데이트 공식을 살펴보겠습니다

지난 영상에서 퍼셉트론의 연결가중치 업데이트 공식을 배웠습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

이 공식의 핵심은,

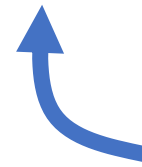
퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

입력값이 클 수록

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률



오차는 계단함수에서는 결국 0 아니면 1 이므로..

연결강도를 많이 변화시켜서..

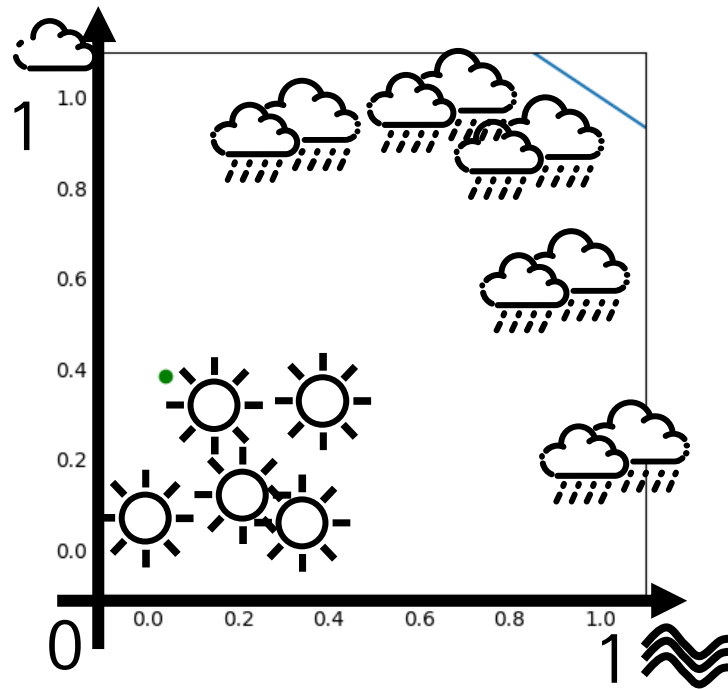
퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

최적의 파라미터를 찾아가는 것이었습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률



경사하강법을 이용한 가중치 학습법도 개념은 동일합니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

공식은 다음과 같습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 $-\nabla C$ x 오차 x 학습률

손실함수의 기울기가 클수록

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x $-\nabla C$ x 오차 x 학습률

연결강도의 변화량도 커집니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x **$-\nabla C$** x 학습률

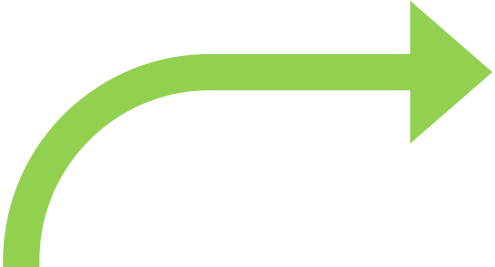
공식을 더 풀어서 쓰면 다음과 같습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x $-\nabla C$ x 학습률


$$\frac{\Delta C}{\Delta w}$$

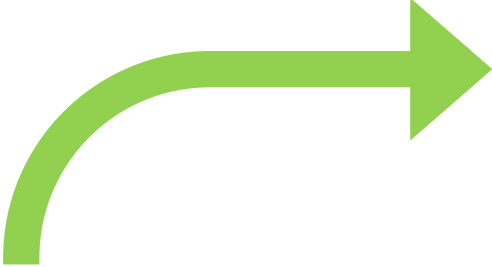
이 공식은 다음에 오류역전파를 설명할 때 꼭 필요한 부분이니 꼭 기억해주세요!

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x $-\nabla C$ x 학습률


$$\frac{\Delta C}{\Delta w}$$

퍼셉트론의 경우, 입력값이 클 수록, 연결강도의 변화가 커지듯

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + **현 입력값** x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 - **현 입력값** x 오차 x 학습률

경사하강법도, ∇C , 즉 손실함수의 기울기가 클수록

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x $-\nabla C$ x 학습률

연결강도의 변화량도 커지게 됩니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

퍼셉트론이 연결강도의 변화와 입력값의 상관관계를 말하고 있다면,

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

경사하강법은 새 연결강도와 손실함수 기울기의 관계를 말하고 있습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 - ∇C x 오차 x 학습률

새 연결강도 \propto 손실함수 기울기

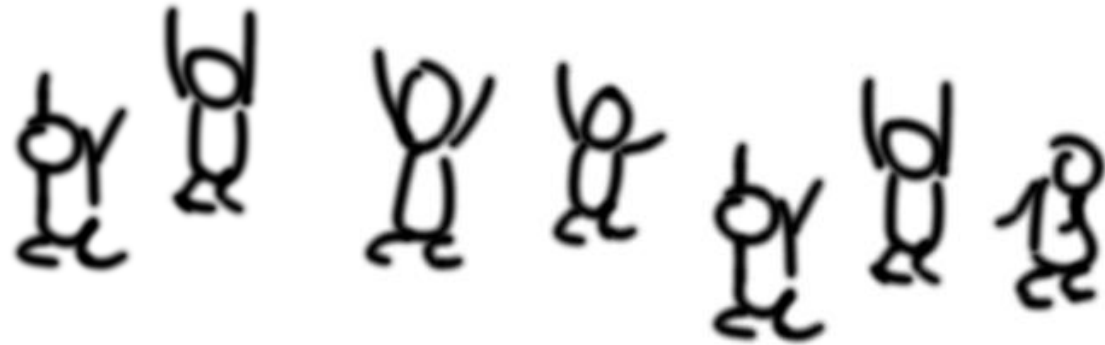
보다 더 직관적으로 말씀드려보자면..

수업시간에 학생들이 떠들어서 선생님이 결국 또 폭발하고 말았습니다

두번째야 이눔시키들..



그래서 단체기합을 또 받았다고 가정해봅시다



요즘엔 이런 단체기합 정말 없겠죠..?



그러면 꼭 쉬는시간에 이런 말들이 오갑니다..

야 너가 시끄럽게 떠들어서 혼났잖아..

웃기시네 너가 제일 시끄러웠거든?

반장, 평소에는 조용한 너까지 떠드니까
쌤이 폭발한거 아니야?

그러면 꼭 쉬는시간에 이런 말들이 오갑니다..

야 너가 시끄럽게 떠들어서 혼났잖아..

웃기시네 너가 제일 시끄러웠거든?

반장, 평소에는 조용한 너까지 떠드니까
쌤이 폭발한거 아니야?

이렇게 선생님의 폭발의 이유가 단순히
떠드는 소리의 크기에 있다기보다는,

그러면 꼭 쉬는시간에 이런 말들이 오갑니다..

야 너가 시끄럽게 떠들어서 혼났잖아..

웃기시네 너가 제일 시끄러웠거든?

반장, 평소에는 조용한 너까지 떠드니까
쌤이 폭발한거 아니야?

평소에 조용하던, 반장마저 갑자기 떠드니까
그게 선생님 폭발을 유발한
유의미한 사건이라 볼 수도 있기 때문입니다

처음에는 입력값에 따라 연결가중치를 수정하던 것이..

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 손실함수 기울기

이제는 손실함수 기울기로 연결강도를 수정하였습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 + $-\nabla C$ x 오차 x 학습률

새 연결강도 \propto 손실함수 기울기

이 방법이 일반적으로 효율적이고 발전된 방법입니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 - ∇C x 현 입력값 x 오차 x 학습률

새 연결강도 \propto 손실함수 기울기

그렇게 최적의 연결가중치를 찾아가는 방법은 지금도 발전하고 있습니다

퍼셉트론의 학습방법:

새 연결강도 = 현 연결강도 + 현 입력값 x 오차 x 학습률

새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

새 연결강도 = 현 연결강도 - ∇C x 현 입력값 x 오차 x 학습률

새 연결강도 \propto 손실함수 기울기

물론 경사하강법이 완벽한 것은 아닙니다

퍼셉트론의 학습방법:

$$\text{새 연결강도} = \text{현 연결강도} + \text{현 입력값} \times \text{오차} \times \text{학습률}$$

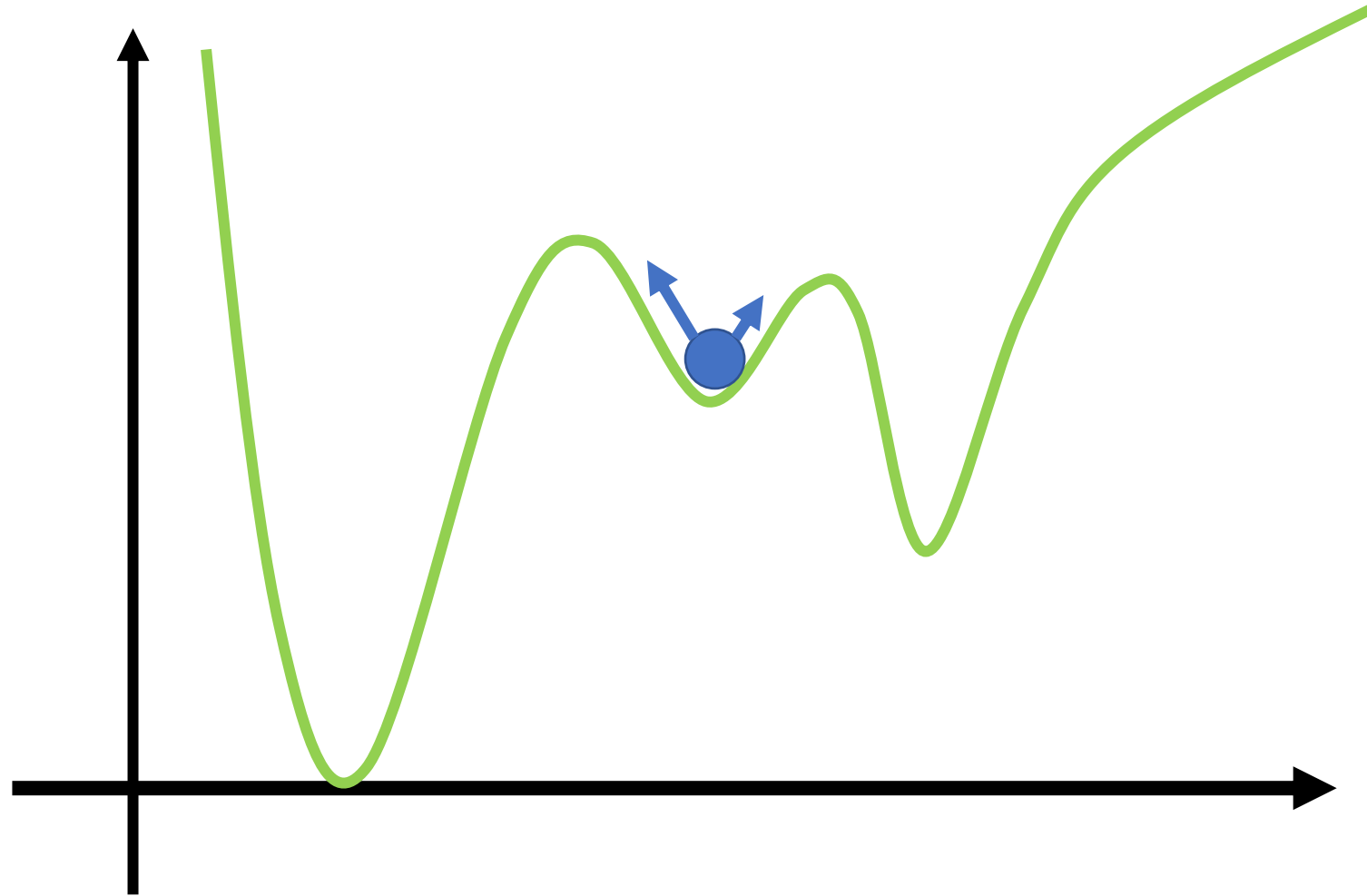
새 연결강도 \propto 현 입력값

경사하강법을 이용한 가중치 학습법:

$$\text{새 연결강도} = \text{현 연결강도} - \text{현 입력값} \times \text{오차} \times \text{학습률}$$

새 연결강도 \propto 손실함수 기울기

대표적 문제점으로, Local minima문제라든지



무엇보다도 사람의 뇌는, 이렇게 점진적인 학습도 이루어지지만,

직관적으로 단번에 깨닫는, 아하 포인트라는 것이 있습니다

頓悟

Eureka!

돈오: 일순간에
무언가를
갑자기 깨달음

유레카: 그리스 말로 무
엇인가를 깨닫거나 발견
할 때 '알았다' 라는 뜻

분명 인간의 학습과정에는 외국어처럼 점진적으로 학습해가는 과정도 있습니다

HOLA

ciao

안녕하세요?

你好

Guten Tag

bonjour

하지만 인간의 학습은 직관이나 순간적인 깨달음 같은 과정을 통해서 학습이 일어나기도 합니다



현재 딥러닝과 인공지능 분야에서는 인간 두뇌의 직관적 능력을 구현한
학습이론은 아직까지 없는 실정입니다



보다 인간처럼 생각하는 기계를 만들기 위해서는 현재의 학습 이론을 뛰어넘는 혁신이 필요한 때가 아닐까라는 생각이 듭니다

이 영상을 보고 계신 분들 중에 한 분이 그런 혁신을 이끌어내는 분이
계셨으면 좋겠습니다



오늘 경사하강법 좀 어떠셨나요?

다소 어려운 내용들도 좀 있긴 했지만

끝까지 함께 해주셔서 감사합니다

다음 시간은 신경망 역사상 가장 중요한
발명중 하나인

역전파 backpropagation 알고리즘
에 대해서 다루어 보도록 하겠습니다

시청해 주셔서..

감사합니다!

이 영상은 여러분의 관심과 사랑으로 제작됩니다
사용하실때는 출처 '신박AI'를 밝혀주세요





Copyright © 2024 by 신박AI

All rights reserved

본 문서(PDF)에 포함된 모든 내용과 자료는 저작권법에 의해 보호받고 있으며, 신박AI에 의해 제작되었습니다.

본 자료는 오직 개인적 학습 목적과 교육 기관 내에서의 교육용으로만 무료로 제공됩니다.

이를 위해, 사용자는 자료 내용의 출처를 명확히 밝히고,

원본 내용을 변경하지 않는 조건 하에 본 자료를 사용할 수 있습니다.

상업적 사용, 수정, 재배포, 또는 이 자료를 기반으로 한 2차적 저작물 생성은 엄격히 금지됩니다.

또한, 본 자료를 다른 유튜브 채널이나 어떠한 온라인 플랫폼에서도 무단으로 사용하는 것은 허용되지 않습니다.

본 자료의 어떠한 부분도 상업적 목적으로 사용하거나 다른 매체에 재배포하기 위해서는 신박AI의 명시적인 서면 동의가 필요합니다.

위의 조건들을 위반할 경우, 저작권법에 따른 법적 조치가 취해질 수 있음을 알려드립니다.

본 고지 사항에 동의하지 않는 경우, 본 문서의 사용을 즉시 중단해 주시기 바랍니다.

