밑바닥 부터 시작하는 딥러닝1

4장 신경망 학습

학습자: 이관형

목차

손실 함수

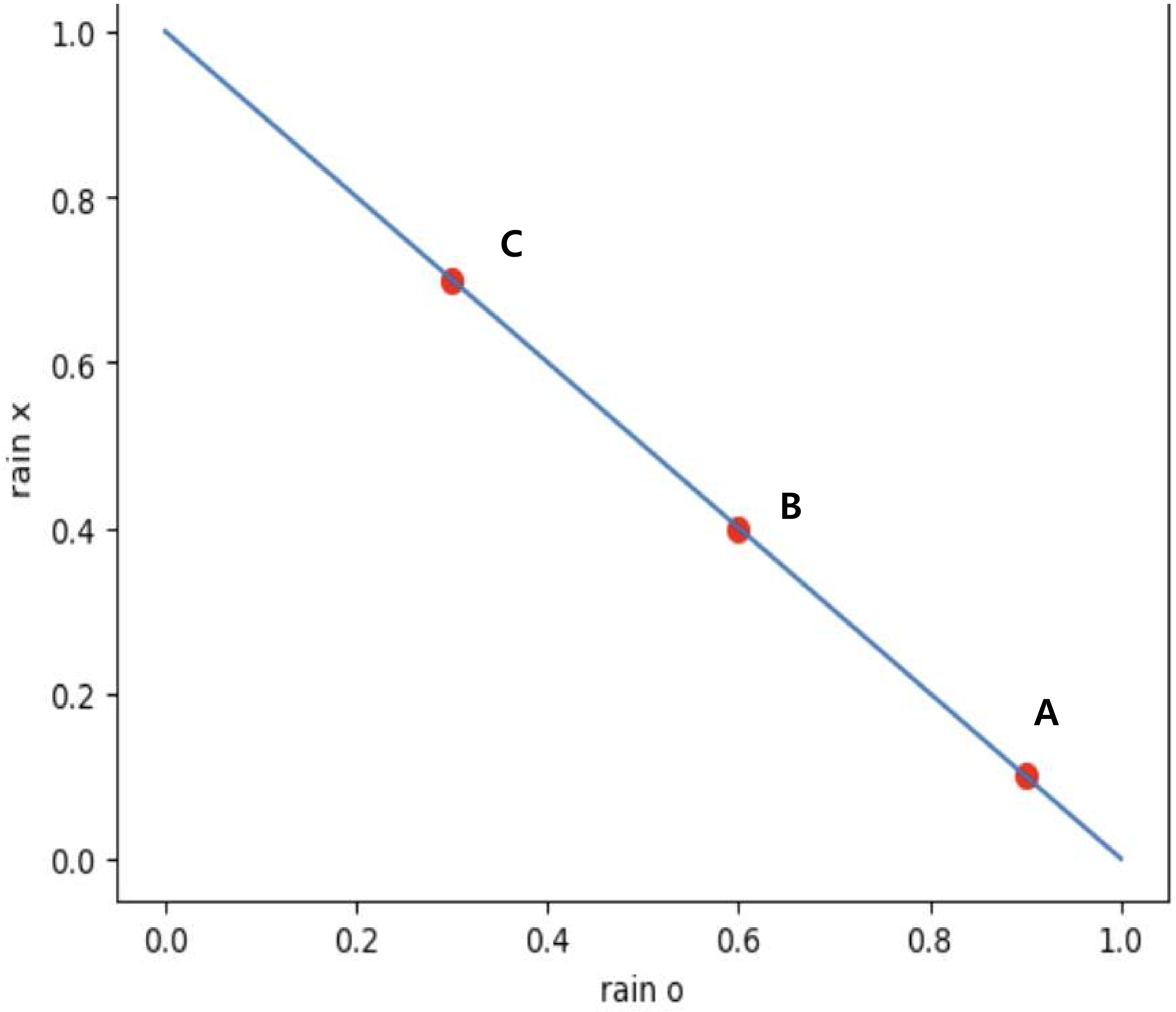
1. 두 확률 분포 사이 거리 개념

2. 오차 제곱합

3. 교차 엔트로피

4. 배치 학습

확 률 의 관 점



x: 비올 확률: (1, 0)
y: 비가 오지 않을 확률 (0, 1)

	비올	비X
A:	(0.9,	0.1)
B:	(0.6,	0.4)
C:	(0.3,	0.7)

확률 분포상으로 표현 가능

1직선 상에 모두 존재 why?
변수 비오나 비오지 않나 두개
따라서 x, y로 표현 가능하니

두 확률 분포

```
>> y = [0.1, 0.05, 0.6, 0.0, 0.05, 0.1, 0.0, 0.1, 0.0, 0.0]
>> t = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
```

결국 MNIST 데이터 값은 두 확률 분포

Y: 학습 된 가능성 높은 확률
t: 라벨값, onehotencoding으로 확률이 1인값으로 변경

우리의 목표
최적의 w, b 찾는 법

이것을 위한 객관적 지표 그중 한가지 -> 두 확률 분포의 거리의 차이가 적은 것
패널티

오 차 제 곱 합

$$E = \frac{1}{2}\sum_{k}(y_k - t_k)^2$$

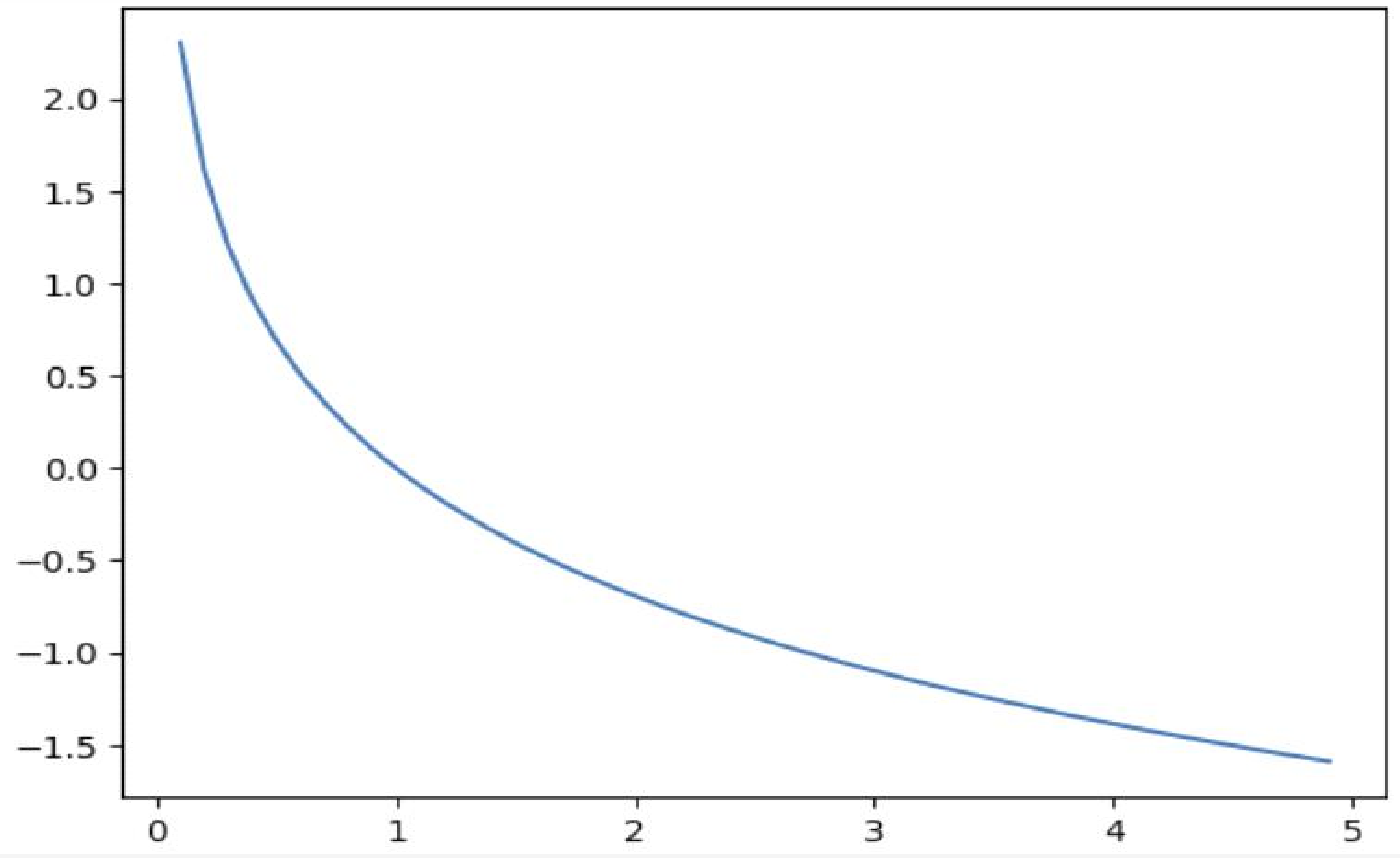
ex) 10,000장에 모두 다 적용

패널티의 평균이 작은 방향으로 ->
경사하강법을 이용

	0	1	2	3	4	5	6	7	8	9	target 0	target 1	target 2	target 3	target 4	target 5	target 6	target 7	target 8	target 9
0	2.674465e-08	8.348746e-10	2.266922e-07	3.987741e-07	3.715802e-10	1.425458e-08	5.154645e-12	3.159027e-04	2.970141e-09	2.592679e-07	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
1	1.941332e-07	4.433875e-08	3.790284e-05	5.743777e-07	2.287596e-11	2.679803e-07	1.105205e-06	5.101235e-11	5.163788e-08	1.921318e-12	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	3.538100e-12	3.378045e-05	1.465168e-07	6.090937e-08	4.499648e-09	2.593922e-08	1.601663e-08	7.753581e-08	4.228212e-08	2.961923e-09	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

교 차 엔 트 로 피 오 차

$$E = -\sum_k t_k \log y_k$$




```
>> y = [0.1, 0.05, 0.6, 0.0, 0.05, 0.1, 0.0, 0.1, 0.0, 0.0]
>> t = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
```

-> -logy(정답)

Q. 이게 왜 두 확률 분포 사이의 거리를
구하는 거?

gmr

배치 학습

```python
train_size = x_train.shape[0]
batch_size = 10


batch_mask = np.random.choice(train_size, batch_size) #인덱스를 매번 랜덤으로
x_batch = x_trian[batch_mask]
t_batch = t_train[batch_mask]
```

x_train.shape[0] == 10000

팬시 인덱싱으로 랜덤 10장씩

gmr

배치 학습(교차 엔트로피)

```
>> y = [0.1, 0.05, 0.6, 0.0, 0.05, 0.1, 0.0, 0.1, 0.0, 0.0]
>> t = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
```

```python
#원핫 인코딩이 안됐을때
def cross_entropy_error(y, t):
    if y.ndim == 1: #배치처리 x
        t = t.reshape(1, t.size) #1*10
        y = y.reshape(1, y.size) #1*10

    batch_size = y.shape[0]
    return -np.sum(np.log(y[np.arange(batch_size), t] + delta)) / batch_size #이
```

y.ndim == 1이거 이해가 잘 안됨.
배치처리 할때 어떤식으로 코드 구현?

if batch_size = 5

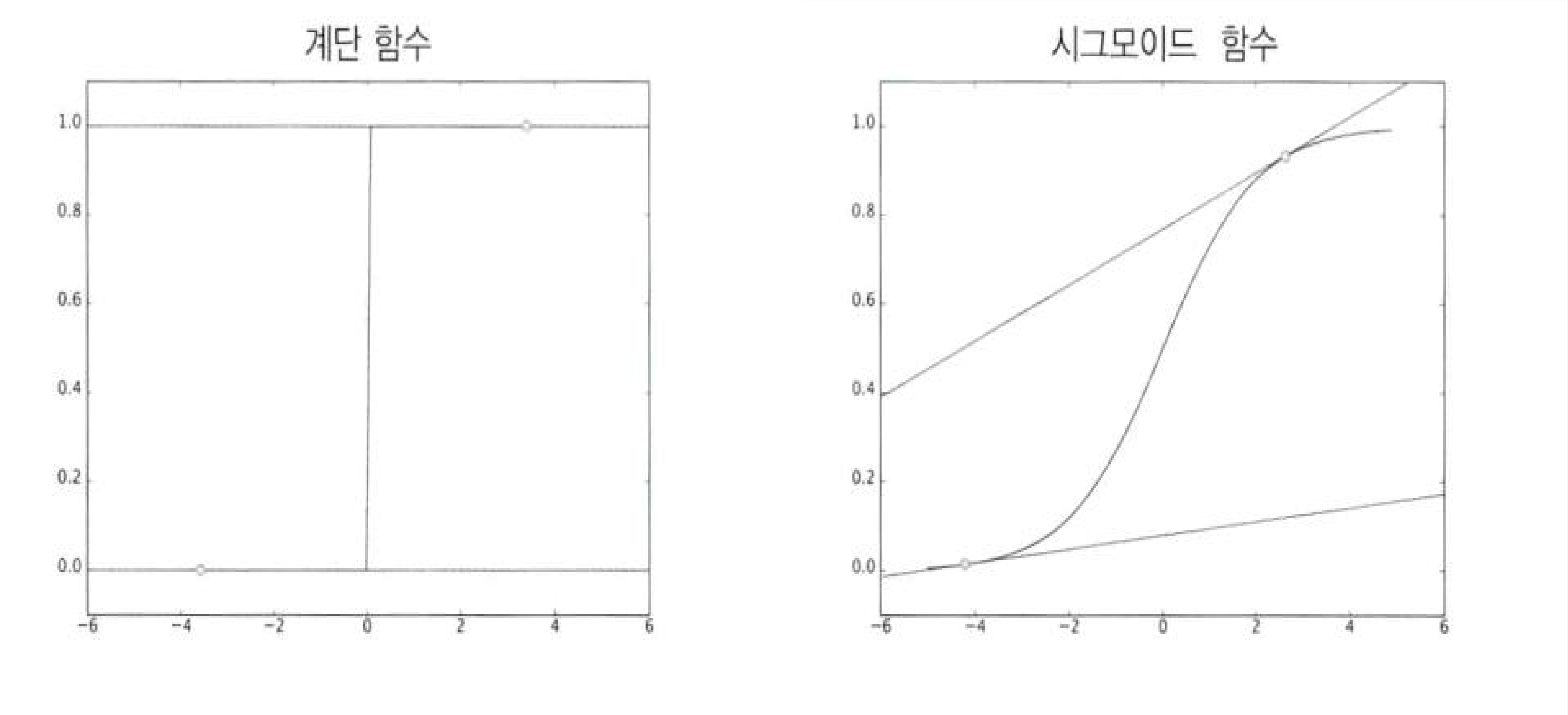
[1, 2, 3, 4, 5]

t: (정답 레이블)
y[0,2] 이런식 인덱싱 가능
y[1, 7]

손실 함수 사용의 이유

'미분'

손실함수의 값이 최소



정확도가 가장 정확 but 미분 대부분 0이 됨

ex) 100장의 훈련데이터 중 32장
32%
가중치 매개변수값을 작게 변해도 32%
개선되도 불연속적 값으로 32, 33, 34이런식으로 변함

like 계단함수

수치 미분

1. 미분의 이해

2. 수치 미분

3. 편미분

4. Gradient(다변수)

gmr

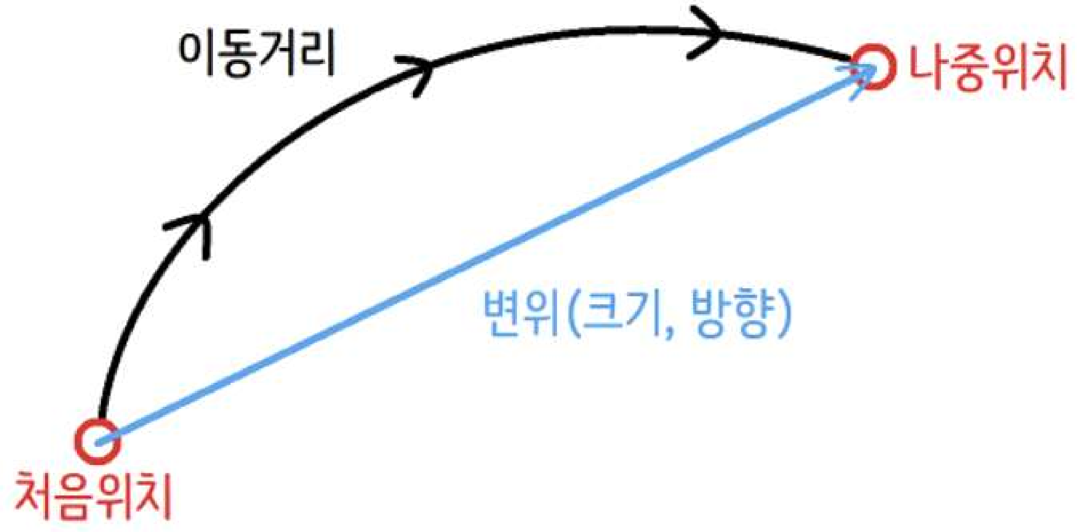
수치 미분

'미분'

순간 변화율
-> 평균 변화율을 lim->0으로 보내
한 시점에 순간 변화율(접선)

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

물리: h 시간
h시간을 0에 즉 순간
like 속도



입력해야하는 것: f(x)함수, x 값(변수)

gmr

중 심 차 분 의 이 해

전방 차분

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

중심 차분

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$

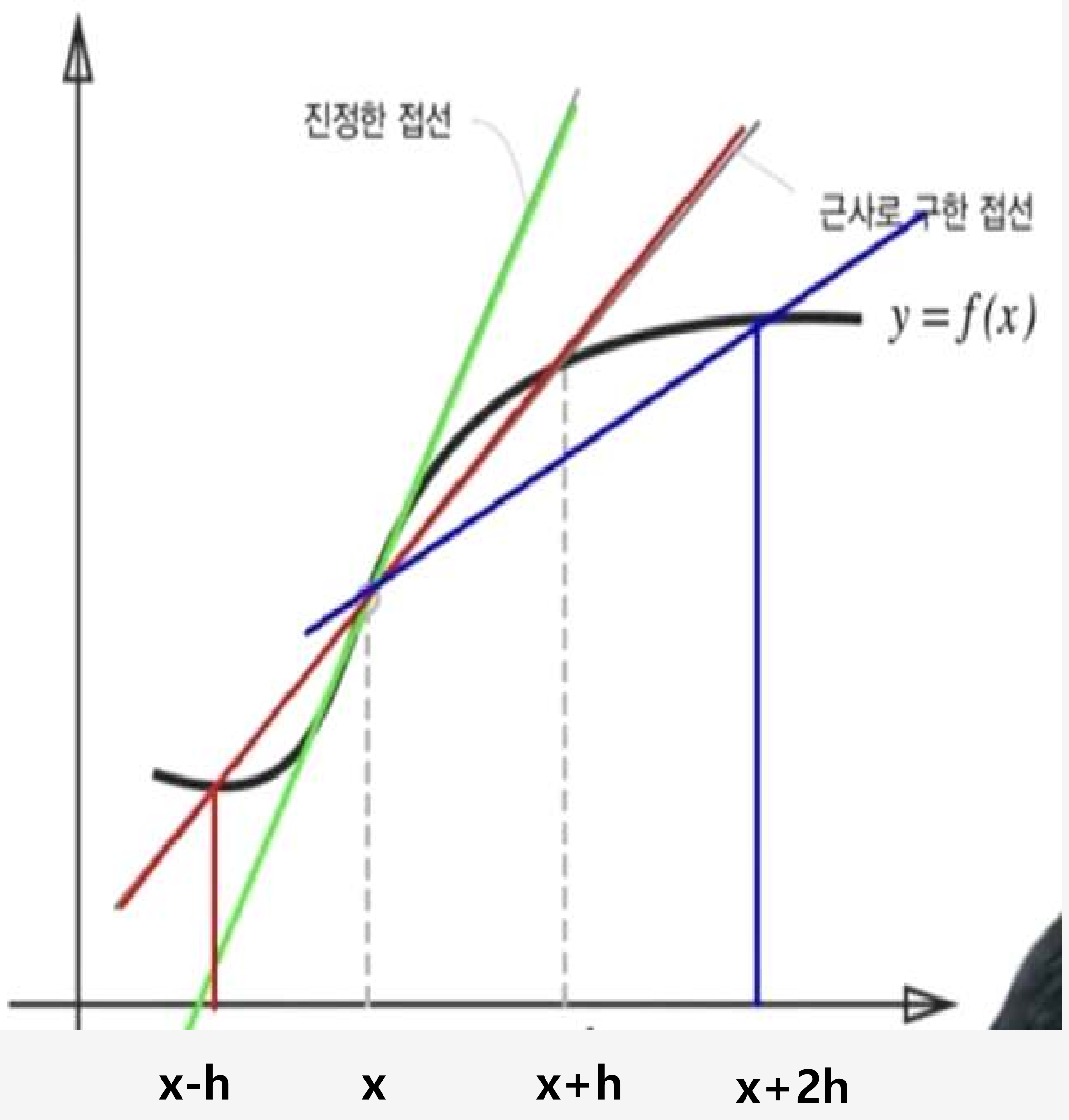
등식

$$\lim_{h \to 0} \boxed{\frac{f(x+h) - f(x)}{h}} = \lim_{h \to 0} \boxed{\frac{f(x+h) - f(x-h)}{2h}}$$

이 성립하므로 미분계수를

$$\frac{f(x+h) - f(x-h)}{2h}$$

로 근사시킬수 있는데 오차가 더 적다.



흘러가는 시간: 2h

gmr

중심 차분의 증명

전방 차분

$$\frac{df(x)}{dx} = \lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$

중심 차분

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$

$$\lim_{h\to 0}\frac{f(x+h)-f(x-h)}{2h}$$

$$= \lim_{h\to 0}\frac{1}{2}\left(\frac{f(x+h)-f(x)}{h}\right)+\frac{1}{2}\left(\frac{f(x)-f(x-h)}{h}\right)$$

$$= \frac{1}{2}\lim_{h\to 0}\underbrace{\frac{f(x+h)-f(x)}{h}}_{f'(x)} + \left(\underbrace{\frac{f(x-h)-f(x)}{-h}}_{f'(x)}\right)$$

gmr

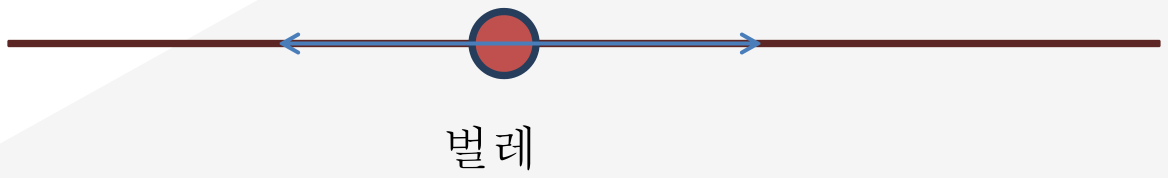
다 변 수 미 분

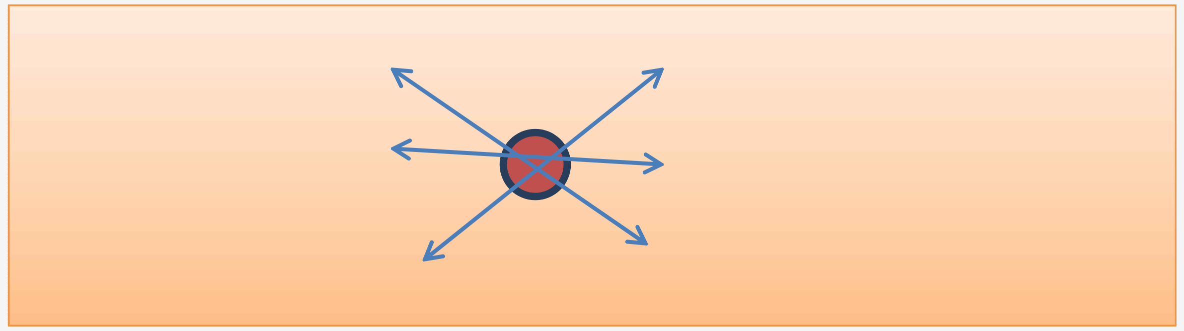
일변수 미분 (함수, 점이 필요)

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

다변수 미분 (함수, 점, 방향이 필요)

$$D_{\vec{v}} f(\mathrm{x}) = \lim_{h \to 0} \frac{f(\mathrm{x} + h\vec{v}) - f(\mathrm{x})}{h}$$





gmr

편 미 분

다변수 미분 (함수, 점, 방향이 필요)

$$D_{\vec{v}} f(\mathrm{x}) = \lim_{h \to 0} \frac{f(\mathrm{x} + h\vec{v}) - f(\mathrm{x})}{h}$$

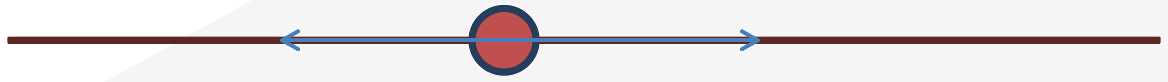
$$f(x_0, x_1) = x_0^2 + x_1^2$$

쉽다... 이건 고등학생때 했던대로 공식

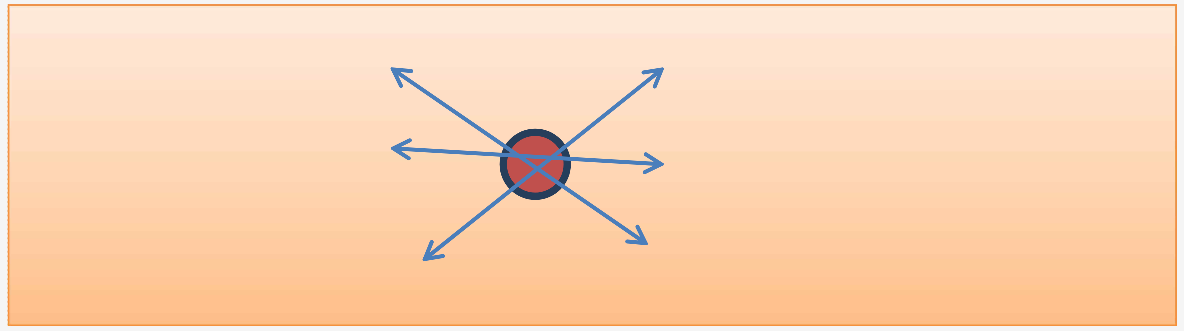
x, y의 편미분

x -> x만 변수로
y -> y만 변수로

방향 미분 중. 각 축에 대한
방향 미분



벌레



gmr

Gradient(기울기)

편미분 동시에

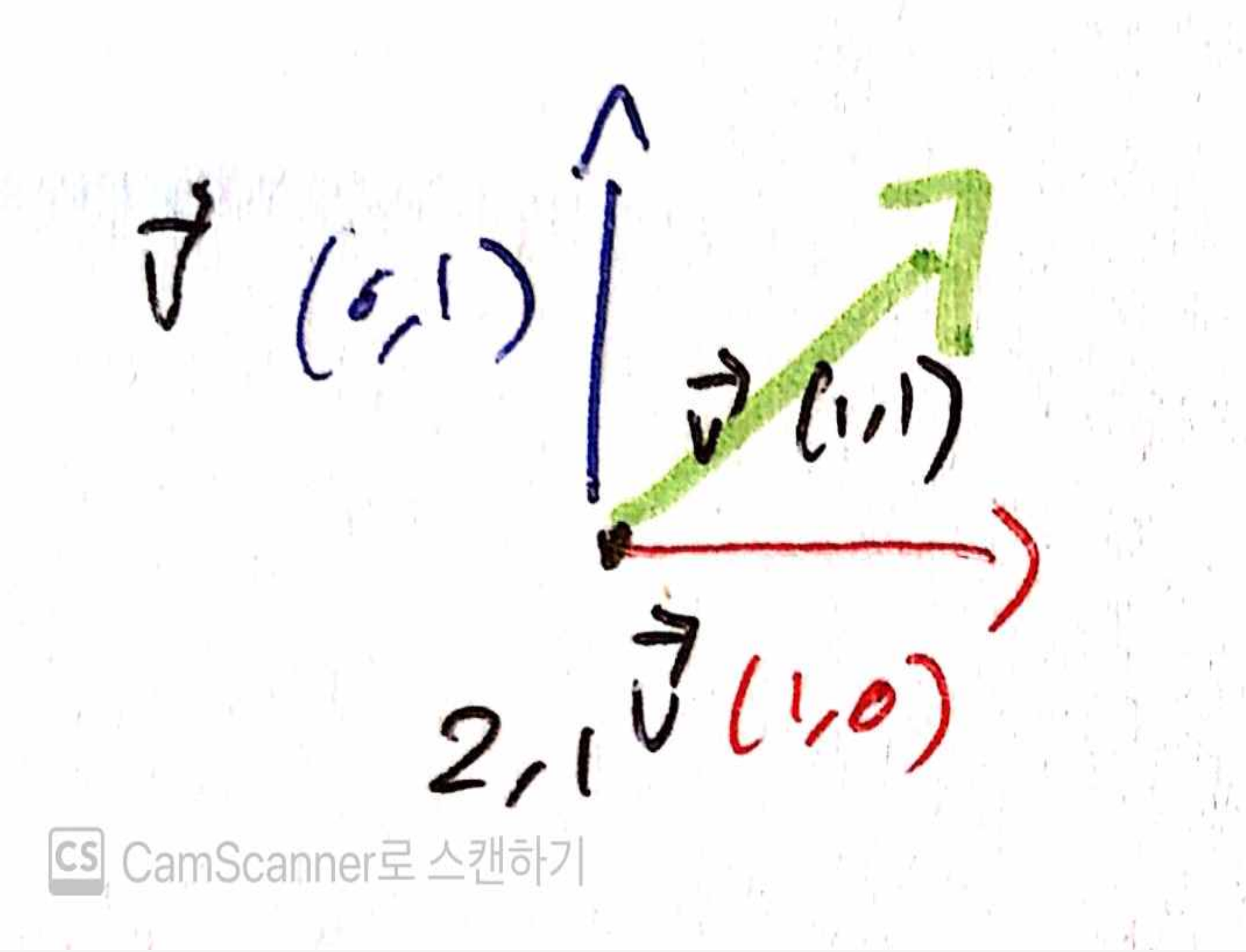
각 축으로의 방향 미분을 벡터 형태로 풀어씀.

f(x,y)=x0^ + x1^ 이때 $\left(\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial x_1}\right)$

Gradient 사용 이유

1. 내적

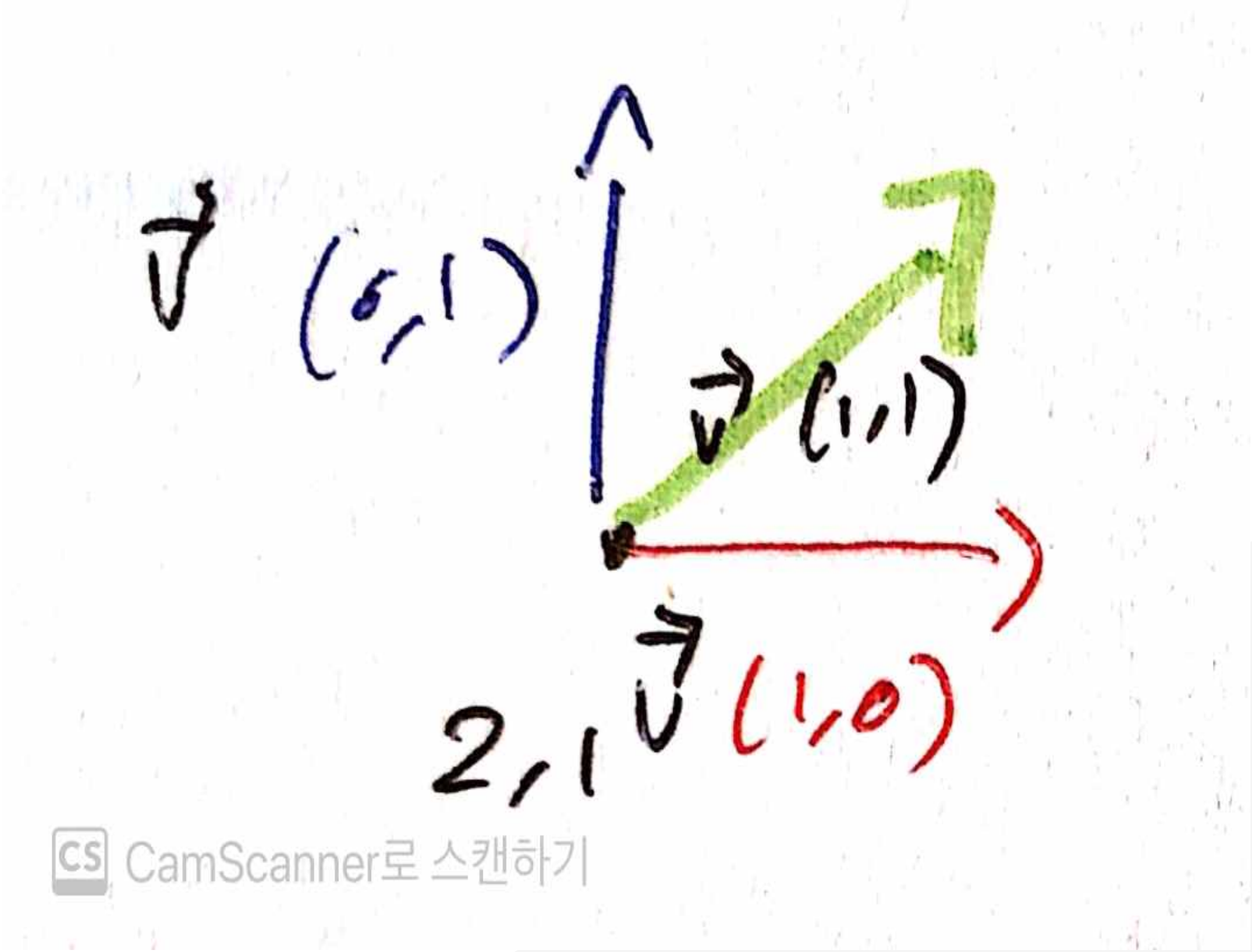
2. 접평면의 방정식



gmr

Gradient 내적

f(x,y)=x0^ + x1^



(4, 2) dot (1,1)(방향) = 6 이런식 가능

$$D_{\vec{v}} f(\mathbf{x}) = \nabla f(\mathbf{x}) \circ \vec{v}$$

모든 방향 미분 계수 내적으로 가능

즉, 손실함수 미분계수 최소? gradient 내적이 최소가 되는 점을 찾으면 됨 !

$$\vec{x} \cdot \vec{y} = x_1y_1+x_2y_2+x_3y_3 = |x|\,|y|\cos\theta$$

크기는 정해지는 값 방향벡터의 크기는 1로 둔다 -> cos 값이 최소가 되는 지점찾기 변수는 세타 나머지는 정해지는 값

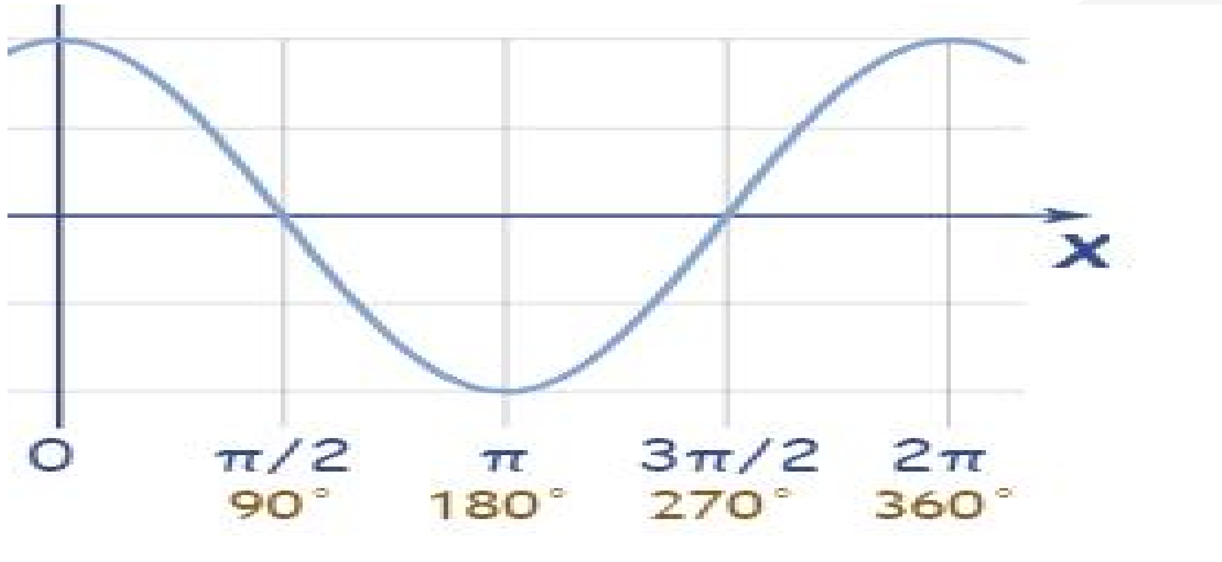
gmr

최대, 최소

즉, 손실함수 미분계수 최소? gradient 내적이 최소가 되는 점을 찾으면 됨 !

$$\vec{x}\cdot\vec{y} = x_1y_1+x_2y_2+x_3y_3 = |x|\,|y|\cos\theta$$

θ: 0-> 1 최대
θ: 180 -> -1 최소



방향 미분의 최대 최소:

θ: 0-> 1 최대

-> gradient 원래 방향, 값은 gradient 크기

θ: 180 -> -1 최소
-> gradient 반대 방향, 값은 gradient 크기

θ: 90 -> 0
-> gradient 수직 방향

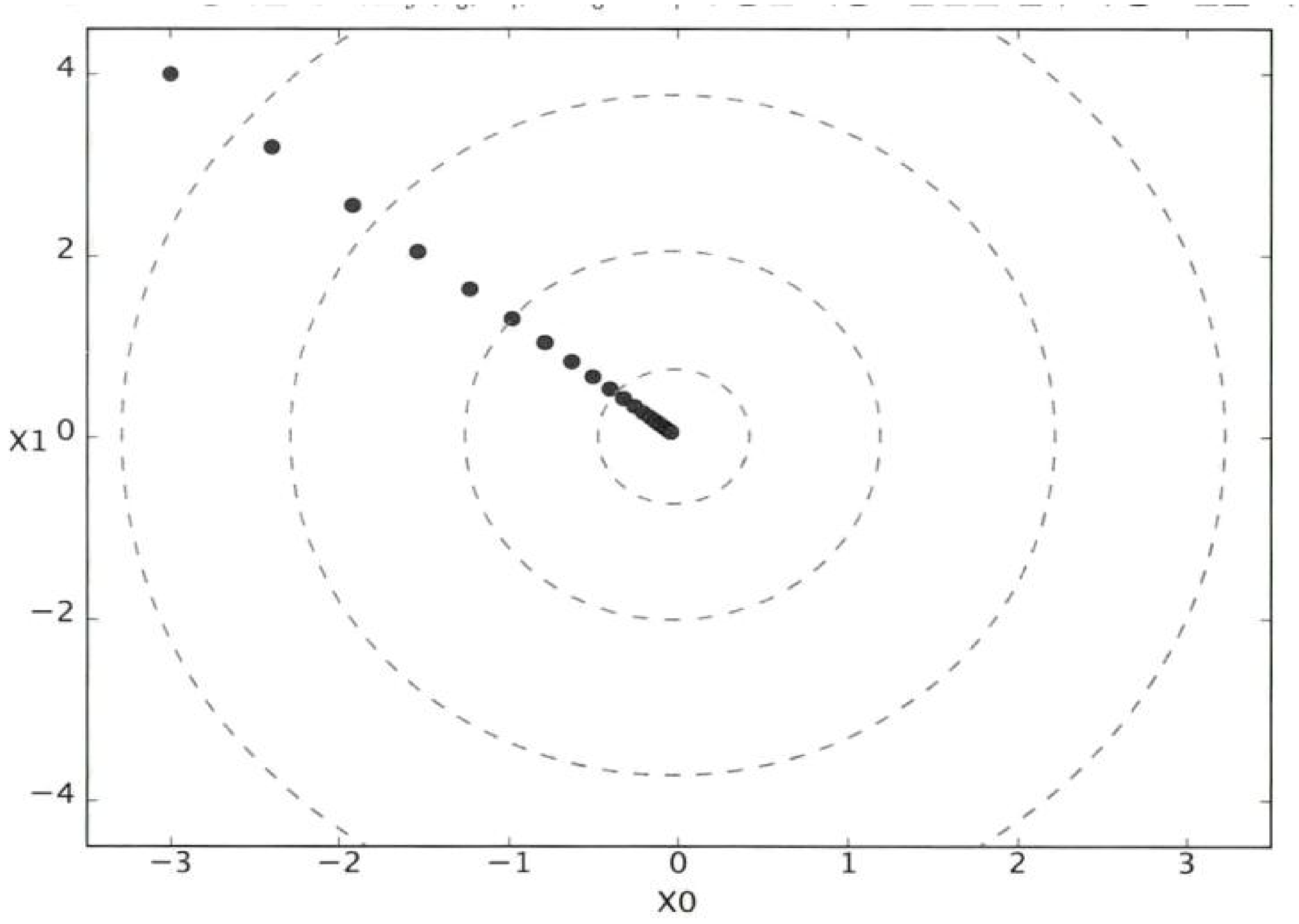
경사 하강법

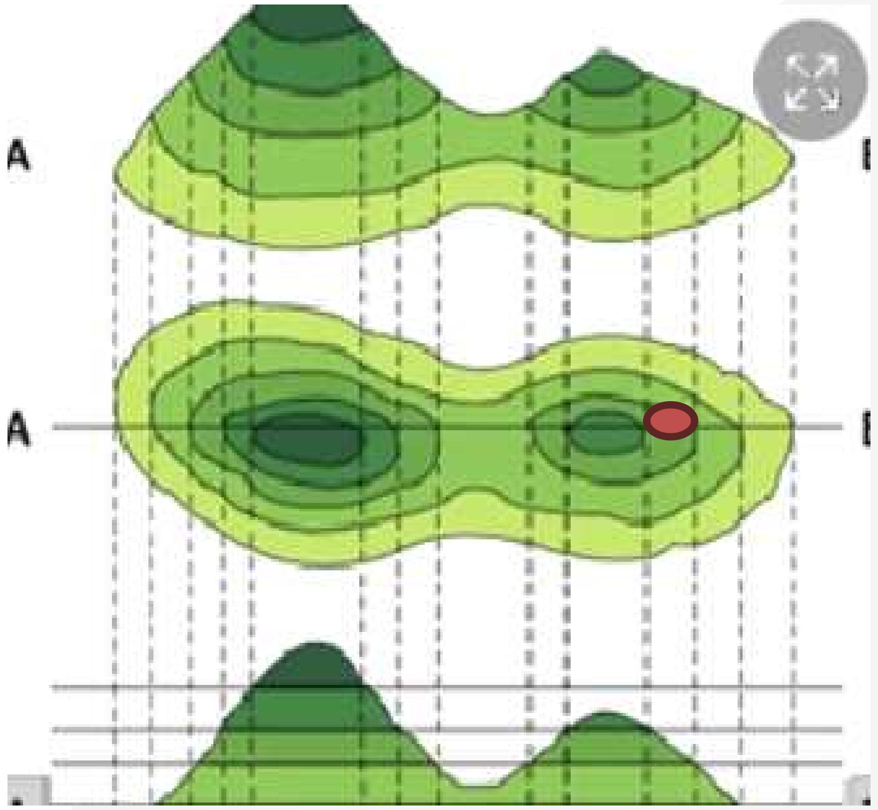
1. 등위선, 등고선의 이해

2. 경사 하강법

3. 한계

등 위 선 ,   등 고 선





정사영 x, y축에 투영

가장 빨리 산을 오르려면
등위선과 수직인 방향

경사하강법:
여러 차원의 벡터의
산을 내려오는것

gmr

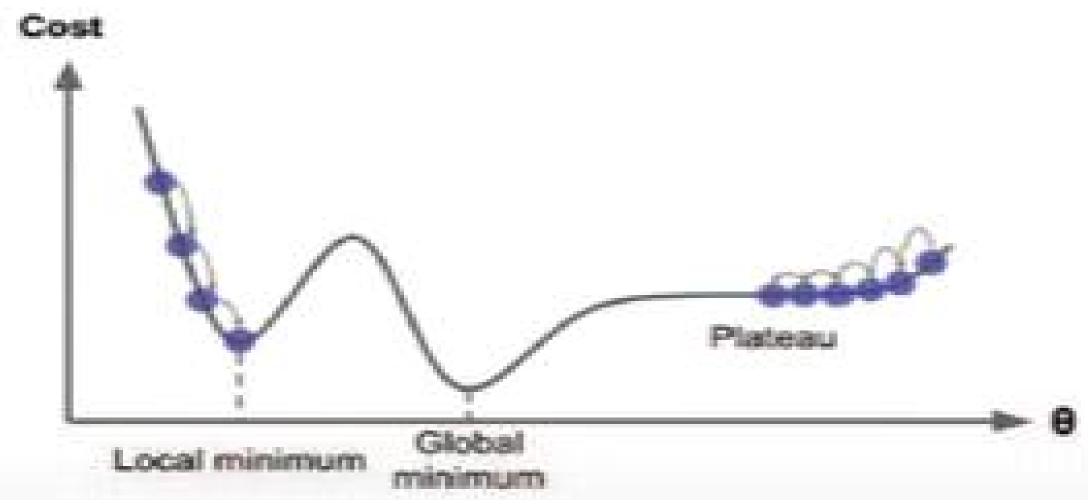
MNIST 데이터 변수

MNIST 변수 784 *
50+ 50*10+10
= 39760개의 변수

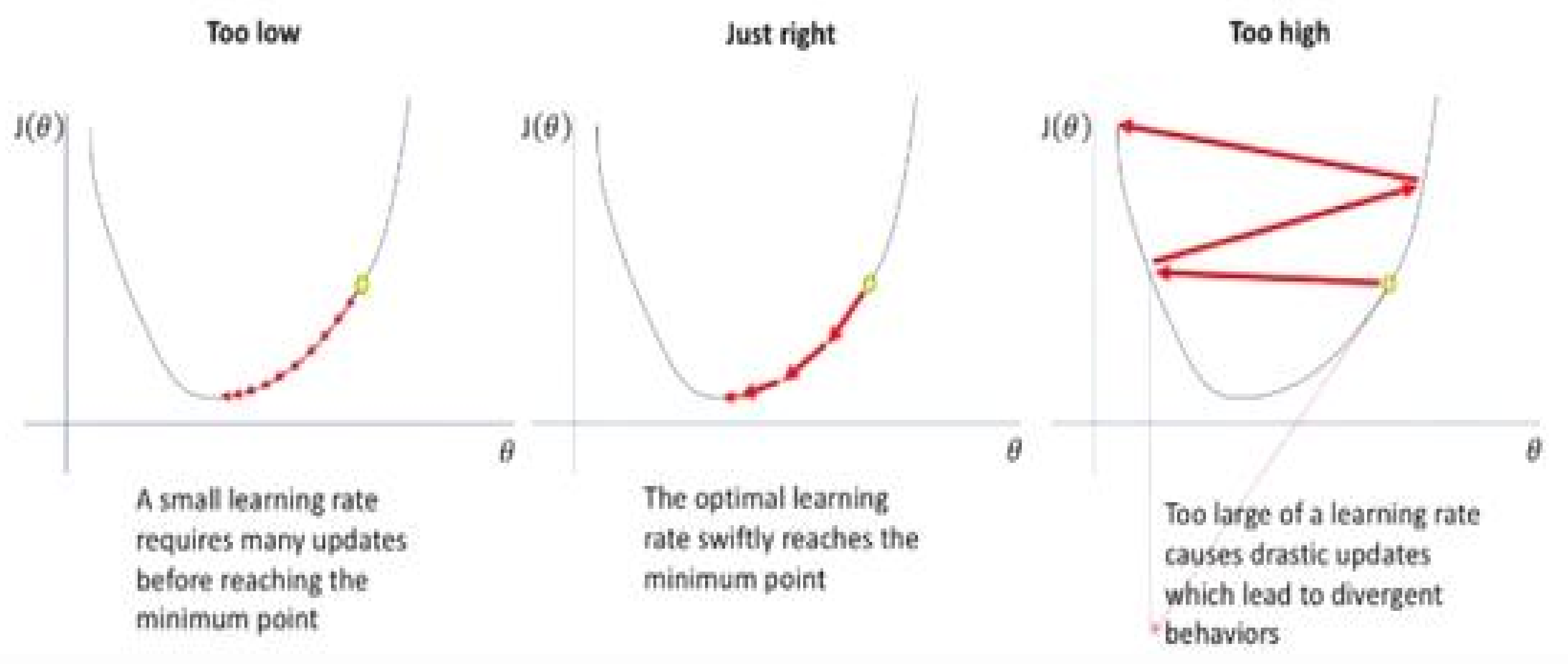
편미분 이용해서
연립으로 풀려면...

-> 경사하강법 최적
값을 예측

$$X_{n+1} = X_n - \eta \nabla f(X_n)$$



경 사 하 강 법


Too low
Just right
Too high
J(θ)
J(θ)
J(θ)
θ
θ
θ
A small learning rate requires many updates before reaching the minimum point
The optimal learning rate swiftly reaches the minimum point
Too large of a learning rate causes drastic updates which lead to divergent behaviors