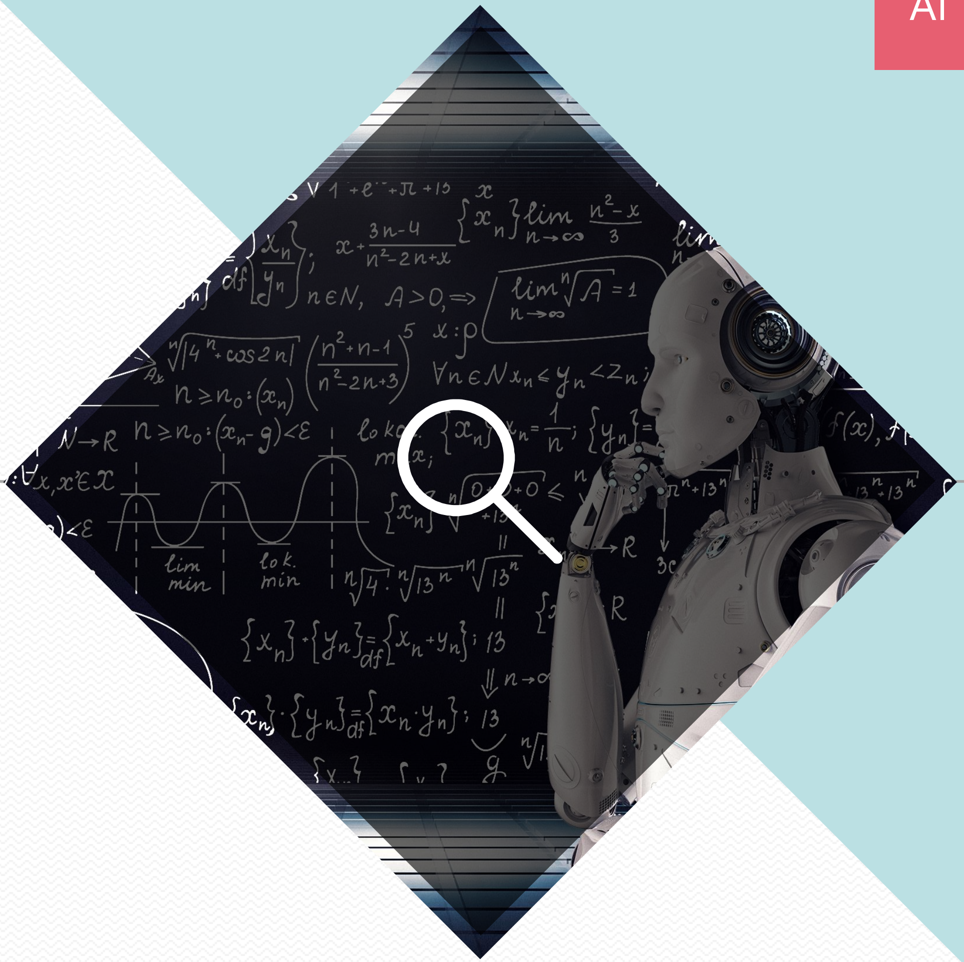
AI TRAINING SERIES



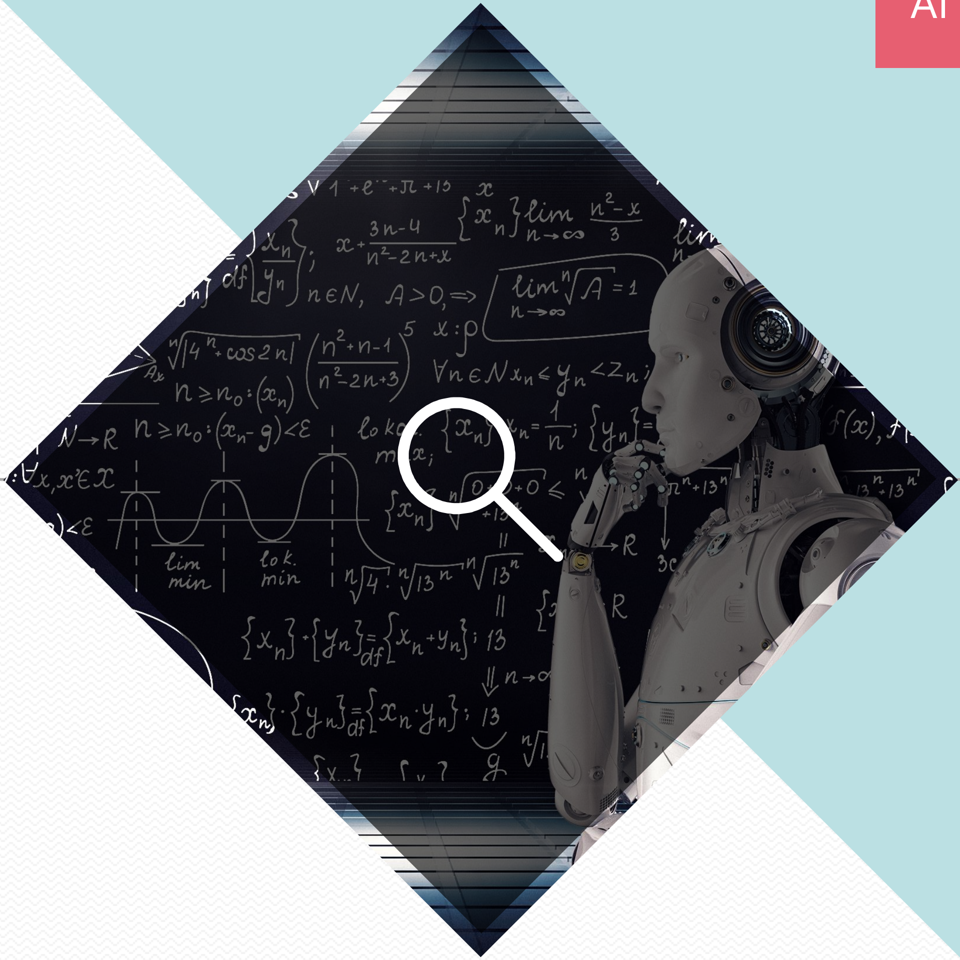
Vector/Matrix/ Entropy

Vector & matrix

CONTENTS

Vector & Matrix

AI TRAINING SERIES



Vector

What is Vector? How is it Defined?

Vector vs Scalar

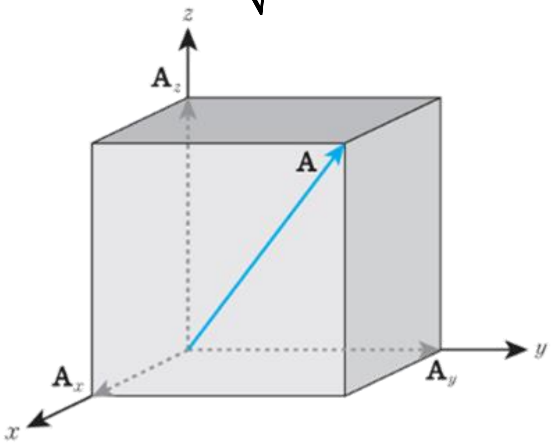
물리량을 표현하는 방법으로 크기와 방향을 모두 갖는 양

벡터(vector)

- 크기와 방향을 모두 갖는 양

예 – 힘
– 속도
– 가속도

$$|A| = A = \sqrt{A_x^2 + A_y^2 + A_z^2}$$



스칼라(scalar)

- 하나의 실수만으로 표시되는, 방향을 갖지 않는 양
- 크기 또는 수치로만 표현될 수 있는 물리량

예 5[kg], 30[cm], 20[㎥], 18[℃], 5[V]

- 두 벡터의 스칼라곱도 스칼라양

Vector의 표현

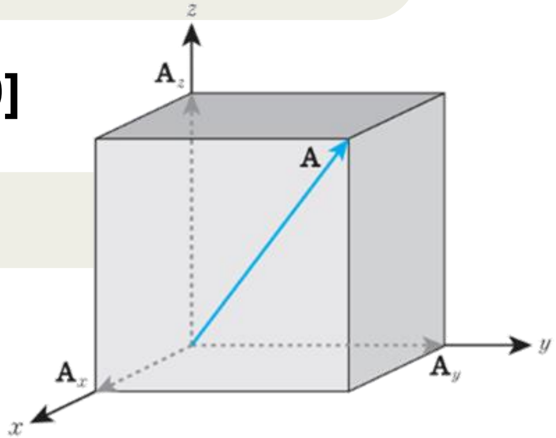
벡터는 좌표점을 이용하거나 기본 벡터를 이용해 표현가능

- 스칼라(고딕체)와 구별하여 굵은 글자나 윗부분에 화살표를 붙인 글자로 표시함
- 공간상의 임의의 벡터를 수식으로 표현하는 방법
- 종점좌표에서 시작점 좌표를 빼는 방식으로 표시

1 좌표점을 이용하여 표시

$$\vec{A} = [2,2,4] = [2-0, 2-0, 4-0]$$

2 기본 벡터를 이용



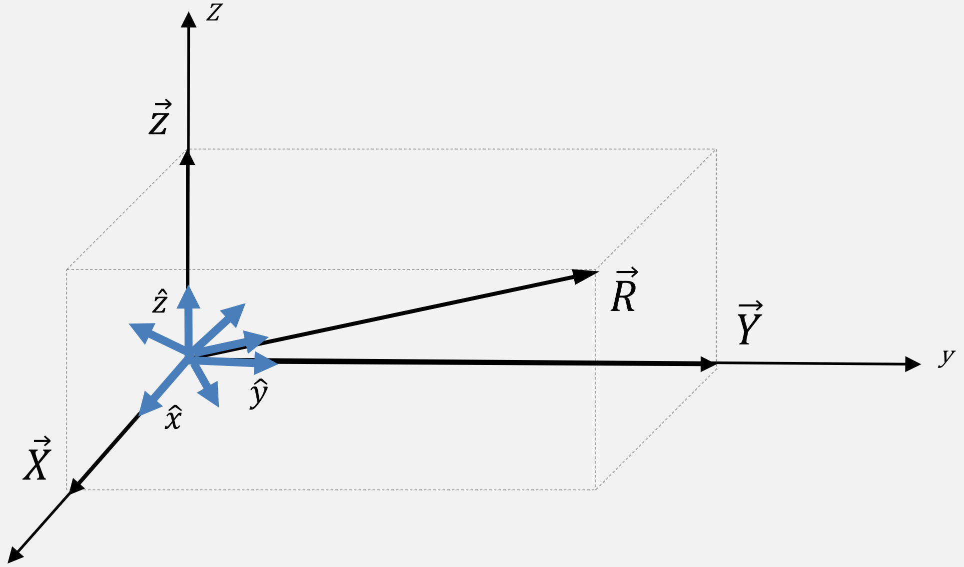
$$\vec{A} = 2\hat{x} + 2\hat{y} + 4\hat{z}$$

Vector의 크기

벡터의 크기와 단위 벡터

단위벡터

어떤 방향을 가지고 크기가 1인 벡터



"벡터를
그 벡터의 크기로
나눈 값"

$$\hat{a} = \frac{\vec{A}}{|\vec{A}|}$$

$$|A| = A = \sqrt{A_x^2 + A_y^2 + A_z^2}$$

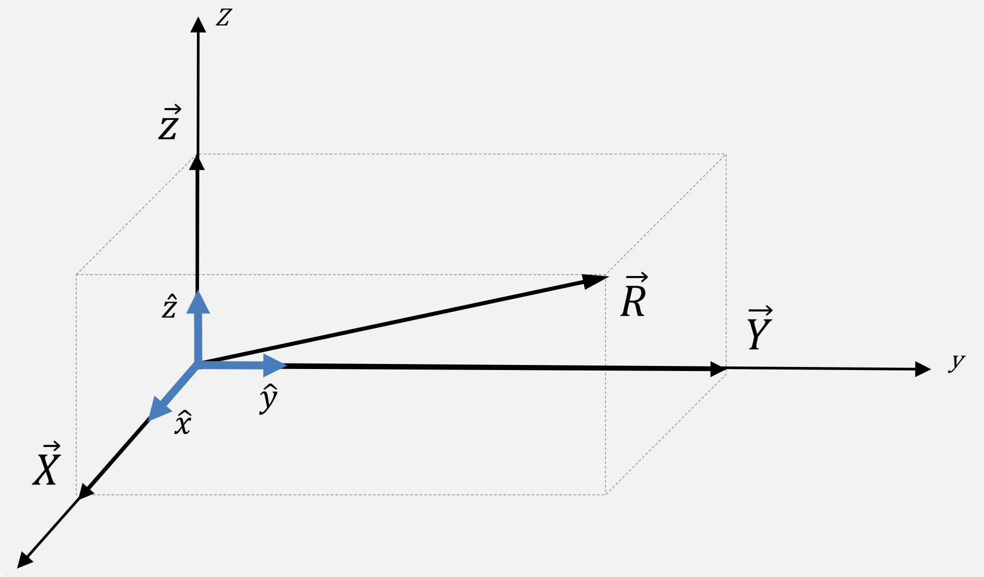
Vector의 크기

기본벡터

기본벡터

각 x, y, z축위에 존재하는 크기가 단위 벡터

→ 통상적으로 소문자로 표시하며, $\hat{x}, \hat{y}, \hat{z}$로 표시함



기본벡터도 단위 벡터에 포함됨

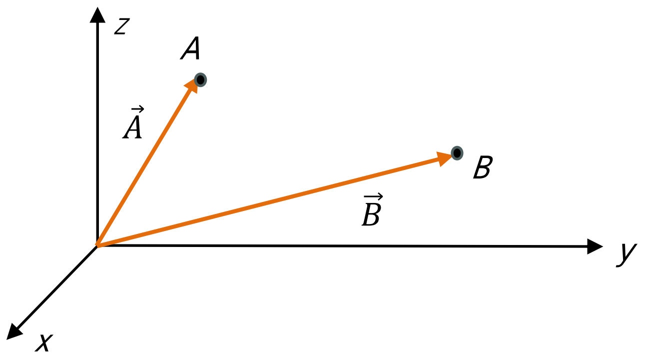
$$\hat{x} = (1,0,0)$$
$$\hat{y} = (0,1,0)$$
$$\hat{z} = (0,0,1)$$

Vector의 연산

벡터의 덧셈과 뺄셈

덧셈/뺄셈은 두 좌표의 각 축의 값을 더하거나 빼서 계산

[벡터의 합성]



$$\vec{A}+\vec{B}=A_x\hat{x}+A_y\hat{y}+A_z\hat{z}+B_x\hat{x}+B_y\hat{y}+B_z\hat{z}=(A_x+B_x)\hat{x}+(A_y+B_y)\hat{y}+(A_z+B_z)\hat{z}$$

$$\vec{A}-\vec{B}=(A_x\hat{x}+A_y\hat{y}+A_z\hat{z})-(B_x\hat{x}+B_y\hat{y}+B_z\hat{z})=(A_x-B_x)\hat{x}+(A_y-B_y)\hat{y}+(A_z-B_z)\hat{z}$$

$$\vec{A}+\vec{B}=[A_x,A_y,A_z]+[B_x,B_y,B_z]=[A_x+B_x,\ A_y+B_y,\ A_z+B_z]$$

$$\vec{A}-\vec{B}=[A_x,A_y,A_z]-[B_x,B_y,B_z]=[A_x-B_x,A_y-B_y,A_z-B_z]$$

Vector의 곱셈

곱셈(내적) : 점곱(dot product)

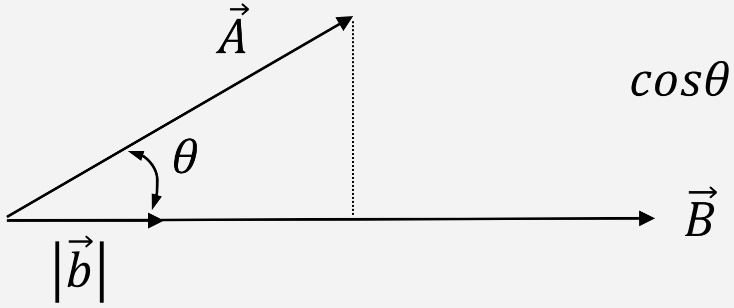
$$A \cdot B = |A||B|cos\theta = A_xB_x + A_yB_y + A_zB_z$$

→ 각 기본 $\hat{x}, \hat{y}, \hat{z}$ 간의 각도는 90도이므로 내적은 0이 됨.

$$A \cdot B = (A_x\hat{x} + A_y\hat{y} + A_z\hat{z}) \cdot (B_x\hat{x} + B_y\hat{y} + B_z\hat{z}) = A_xB_x + A_yB_y + A_zB_z$$

스칼라 곱(scalar product) : 점곱의 다른 말

- $\vec{b}$ 가 단위 벡터일 경우는 $\vec{A}$의 $\vec{b}$방향의 성분 벡터가 됨



$$cos\theta = \frac{A_xB_x + A_yB_y + A_zB_z}{|A||B|}$$

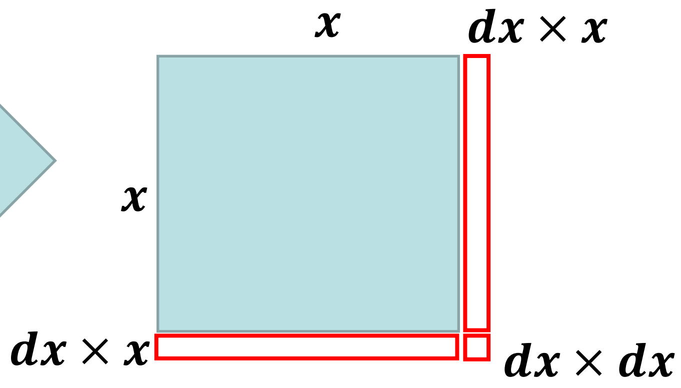
1 벡터(vector)

6) Derivatives : 미분

- x의 dx만큼 변화 → f(x)의 변화량 관찰

$$\frac{\partial f(x_0, y_0)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x, y_0) - f(x_0, y_0)}{\Delta x}$$

$$y = x^2$$



$$\begin{aligned} dy &= (x + dx)^2 - y \\ &= \cancel{x^2} + 2xdx + (dx)^2 \\ &\quad - \cancel{x^2} \\ &= \boxed{2xdx} + (dx)^2 \end{aligned}$$

$$y = x^3$$

$$\begin{aligned} dy &= (x + dx)^3 - y \\ &= \cancel{x^3} + 3x^2dx + 3x(dx)^2 + (dx)^3 \\ &\quad - \cancel{x^3} \\ &= \boxed{3x^2dx} + 3x(dx)^2 + (dx)^3 \end{aligned}$$

1 벡터(vector)

6) **Derivatives** : 1 variable → 2 variable

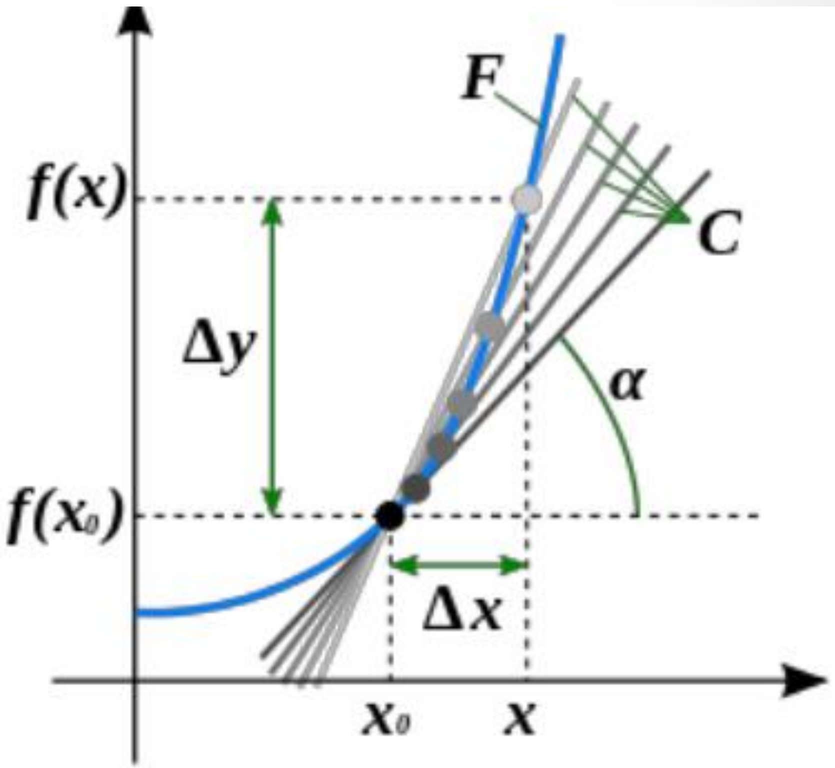
- First order

$$f'(x_0) = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

"편미분으로 바로 차원을 늘릴 수 있을까?"

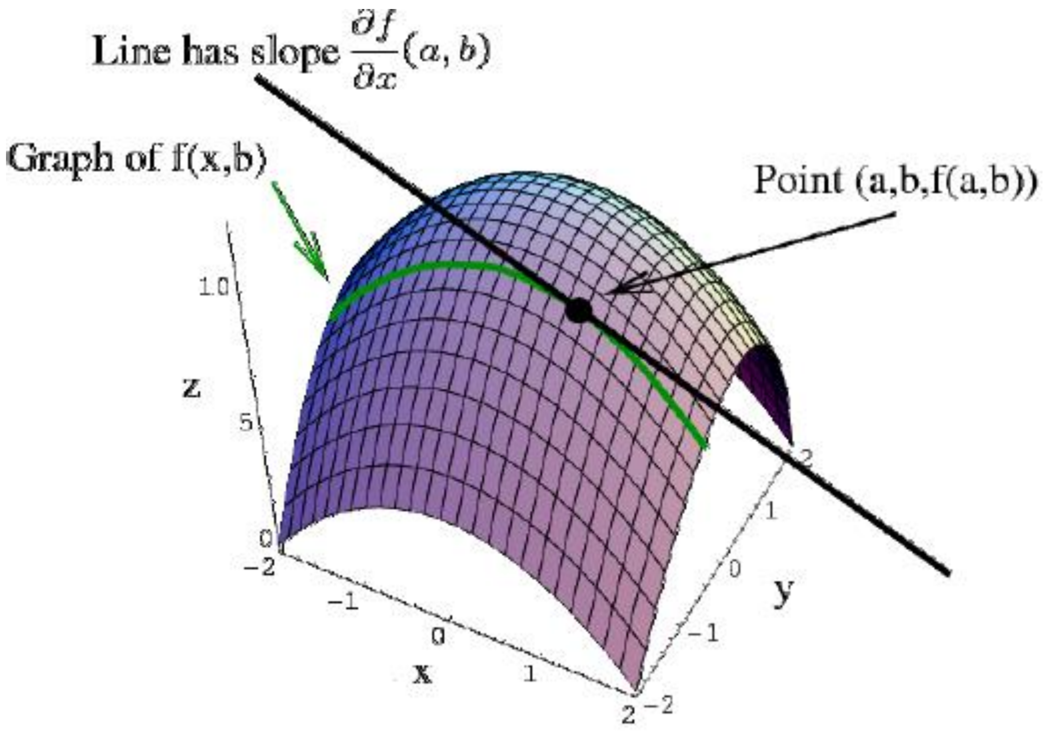
$$\frac{\partial f(x_0, y_0)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x, y_0) - f(x_0, y_0)}{\Delta x}$$

$$\frac{\partial f(x_0, y_0)}{\partial y} = \lim_{\Delta y \to 0} \frac{f(x_0, y_0 + \Delta y) - f(x_0, y_0)}{\Delta y}$$



1 벡터(vector)

6) **Derivatives** : multivariable of partial derivatives



$$\nabla A(x,y) = \frac{\partial A}{\partial x}\hat{x} + \frac{\partial A}{\partial y}\hat{y}$$

"각 좌표축이 독립적이므로
편미분 사용가능"

$$\frac{\partial f(x_0,y_0)}{\partial x} = \lim_{\Delta x\to 0}\frac{f(x_0+\Delta x,y_0)-f(x_0,y_0)}{\Delta x}$$

$$\frac{\partial f(x_0,y_0)}{\partial y} = \lim_{\Delta y\to 0}\frac{f(x_0,y_0+\Delta y)-f(x_0,y_0)}{\Delta y}$$

1 벡터(vector)

7) Gradient(∇) : Derivatives of vector

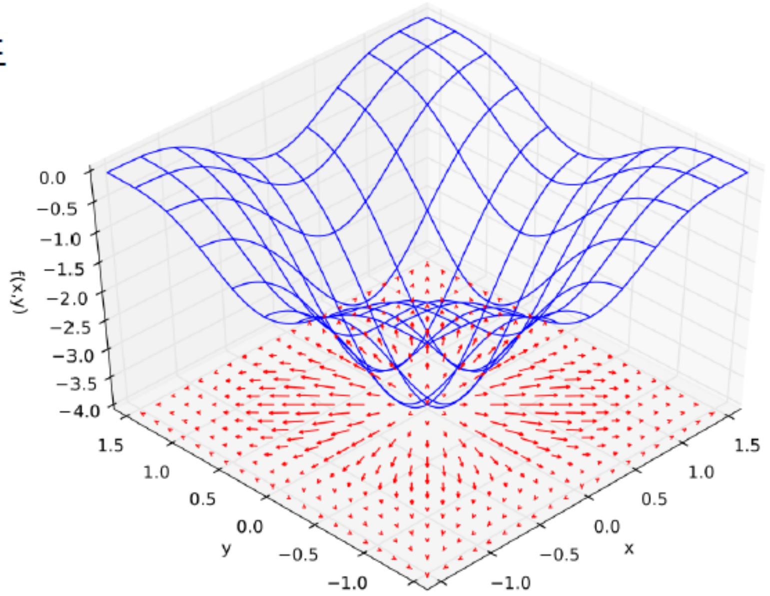
- The gradient is a vector of partial derivatives representing the rate and direction of the steepest ascent of a function.

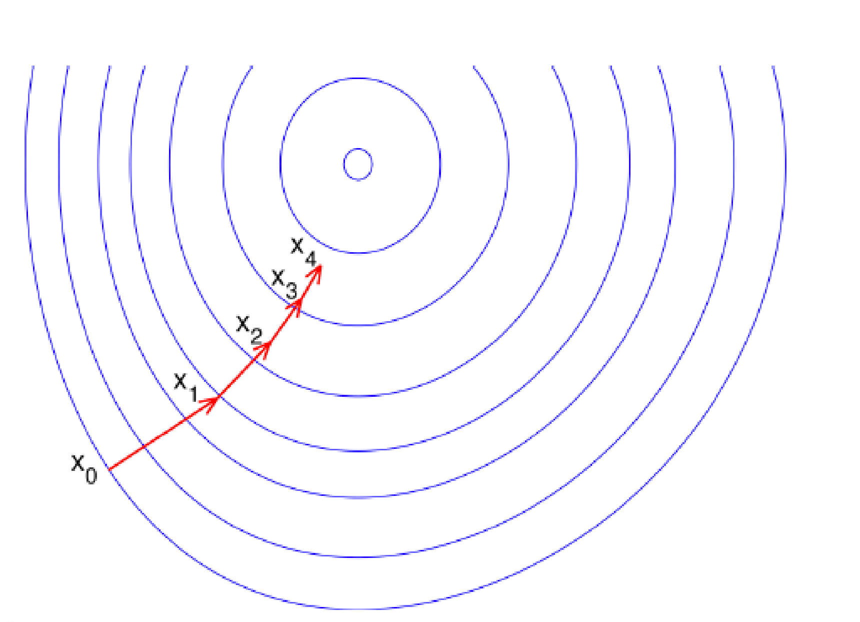
- Gradient는 편미분값의 벡터 표현입니다.

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- n차원에서는

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$



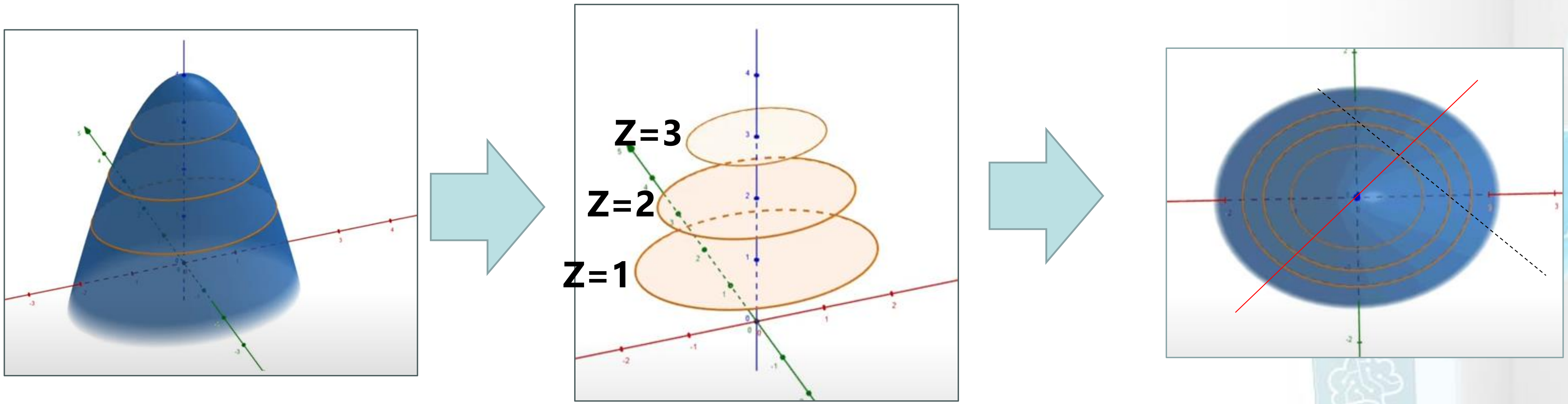


1 벡터(vector)

7) Gradient(∇) : Derivatives of vector

$$f(x,y)=4-x^2-y^2$$

→ X=1,y=1에서 가장 가파른 방향은 어느 방향일까?



1 벡터(vector)

7) Gradient(∇) : Derivatives of vector

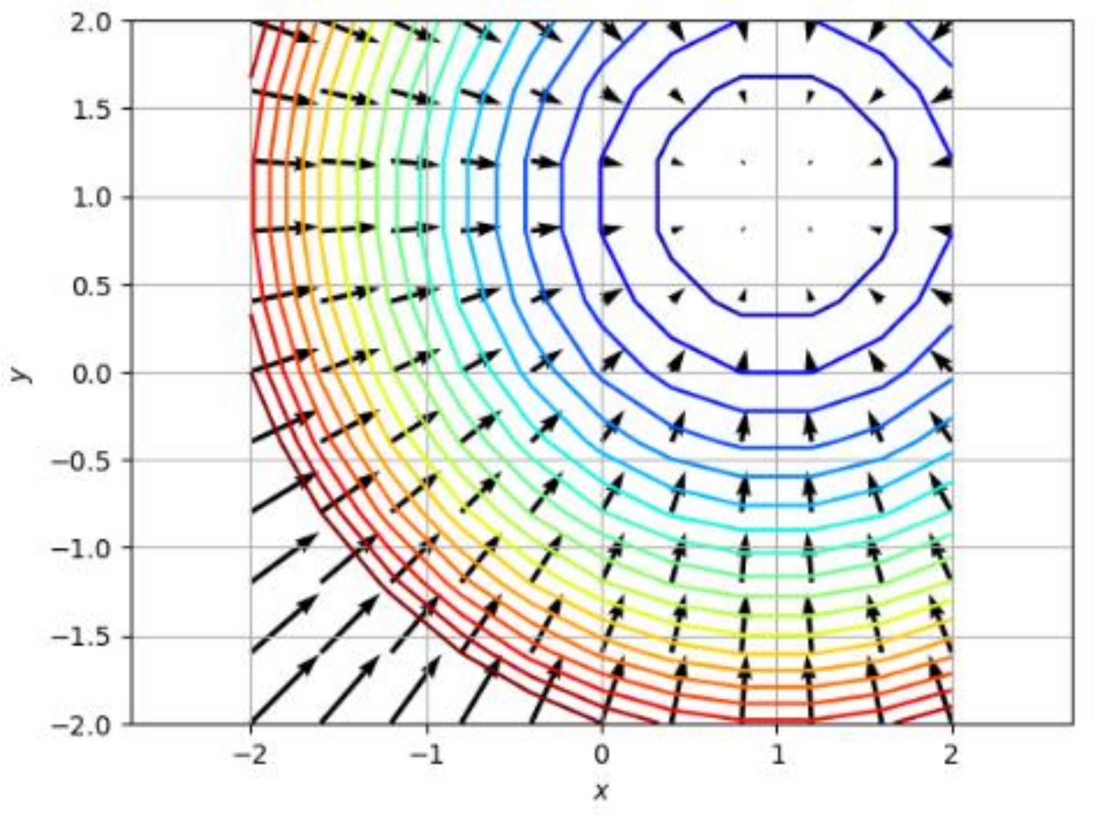
```
import numpy as np
import matplotlib.pylab as plt
x = np.linspace(-2,2, 11)
y = np.linspace(-2,2, 11)

print(x)
print(y)

x,y = np.meshgrid(x,y)
print(x)
print(y)

f = lambda x,y : (x-1)**2 + (y-1)**2
z = f(x,y)
print(z)
```

```
from mpl_toolkits.mplot3d import Axes3D
```

```
grad_f_x = lambda x, y: 2 * (x-1)
grad_f_y = lambda x, y: 2 * (y-1)
```

```
dz_dx = grad_f_x(x,y)
dz_dy = grad_f_y(x,y)
```

```
ax = plt.axes()
ax.contour(x, y, z, levels=np.linspace(0, 10, 20), cmap=plt.cm.jet)
ax.quiver(x, y, -dz_dx, -dz_dy)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```



1 벡터(vector)

8) Chain rule

$$x \longrightarrow f(x) = \left(x^2+1\right)^2$$

$$x \rightarrow x^2 \rightarrow x^2+1 \rightarrow \left(x^2+1\right)^2$$

$$\frac{d\left(\left(x^2+1\right)^2\right)}{dx} = \frac{d\left(\left(x^2+1\right)^2\right)}{d\left(x^2+1\right)} \frac{d\left(x^2+1\right)}{dx^2} \frac{dx^2}{dx}$$

$$f = \left(x^2+1\right)^2,\ g = x^2+1,\ h = x^2$$

$$\frac{df}{dx} = \frac{df}{dg}\frac{dg}{dh}\frac{dh}{dx}$$

$$\frac{d\left(\left(x^2+1\right)^2\right)}{dx} = \frac{df}{dg}\frac{dg}{dh}\frac{dh}{dx} = 2\left(x^2+1\right)\cdot 1 \cdot 2x = 4x\left(x^2+1\right)$$

8) Chain rule

$$\frac{\partial f\left(g\left(h(x)\right)\right)}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial h}\frac{\partial h}{\partial x} = f'\left(g\left(h(x)\right)\right)g'\left(h(x)\right)h'(x) = (f \circ g \circ h)'$$

$$f(x) = \left(x^2 + 1\right)^2$$

$$f(g) = g^2 \qquad g(h) = h + 1 \qquad h(x) = x^2$$

$$f'(g) = 2g \qquad g'(h) = 1 \qquad h'(x) = 2x$$

Direct method

$$f(x) = \left(x^2 + 1\right)^2 = x^4 + 2x^2 + 1$$

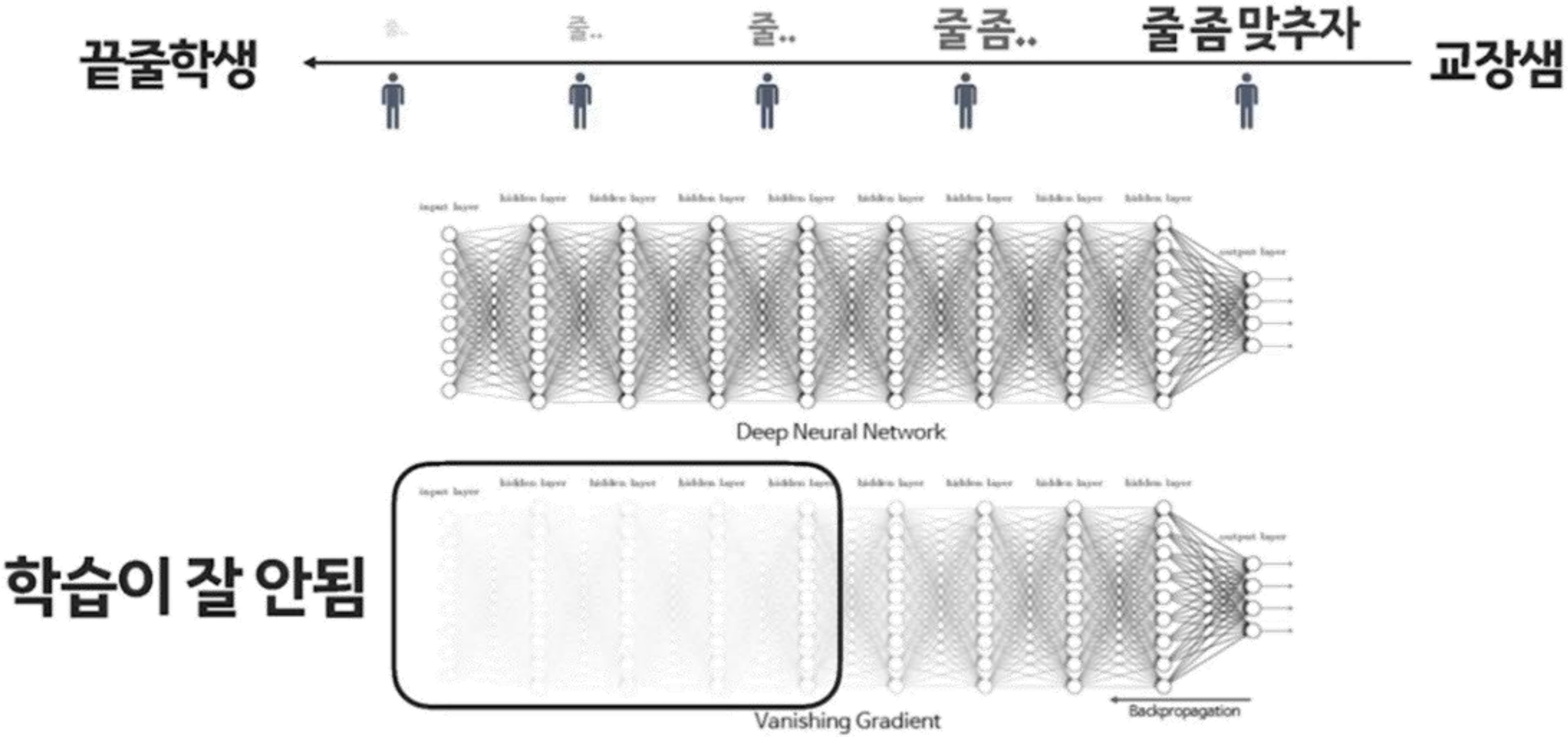
$$f'(x) = 4x^3 + 4x = 4x\left(x^2 + 1\right)$$

Chain rule

$$f'(x) = 2\left(x^2 + 1\right) \cdot 1 \cdot 2x$$
$$= 4x\left(x^2 + 1\right)$$

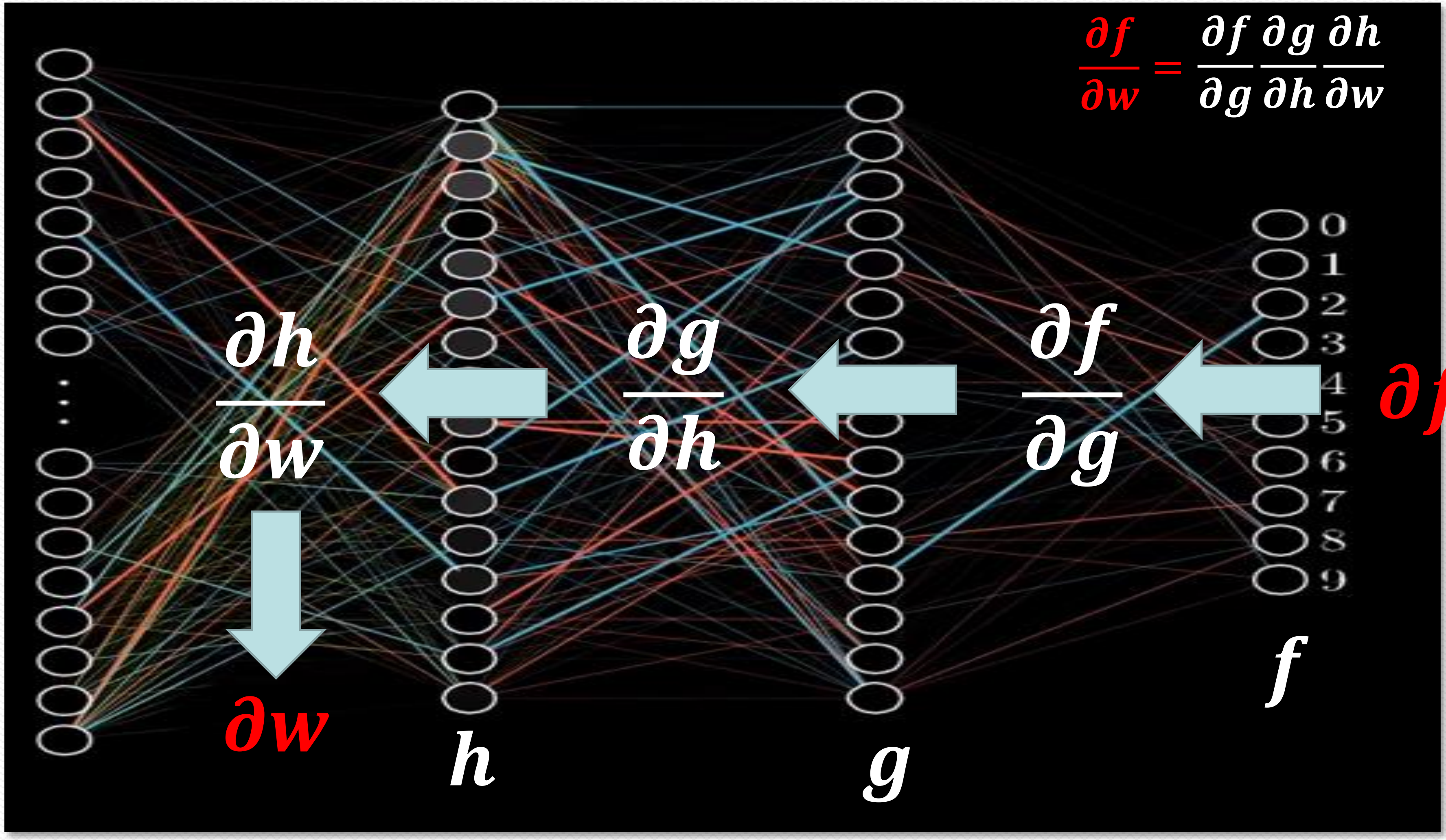
Vanishing gradient

레이어가 깊을 수록 업데이트가 사라져감.



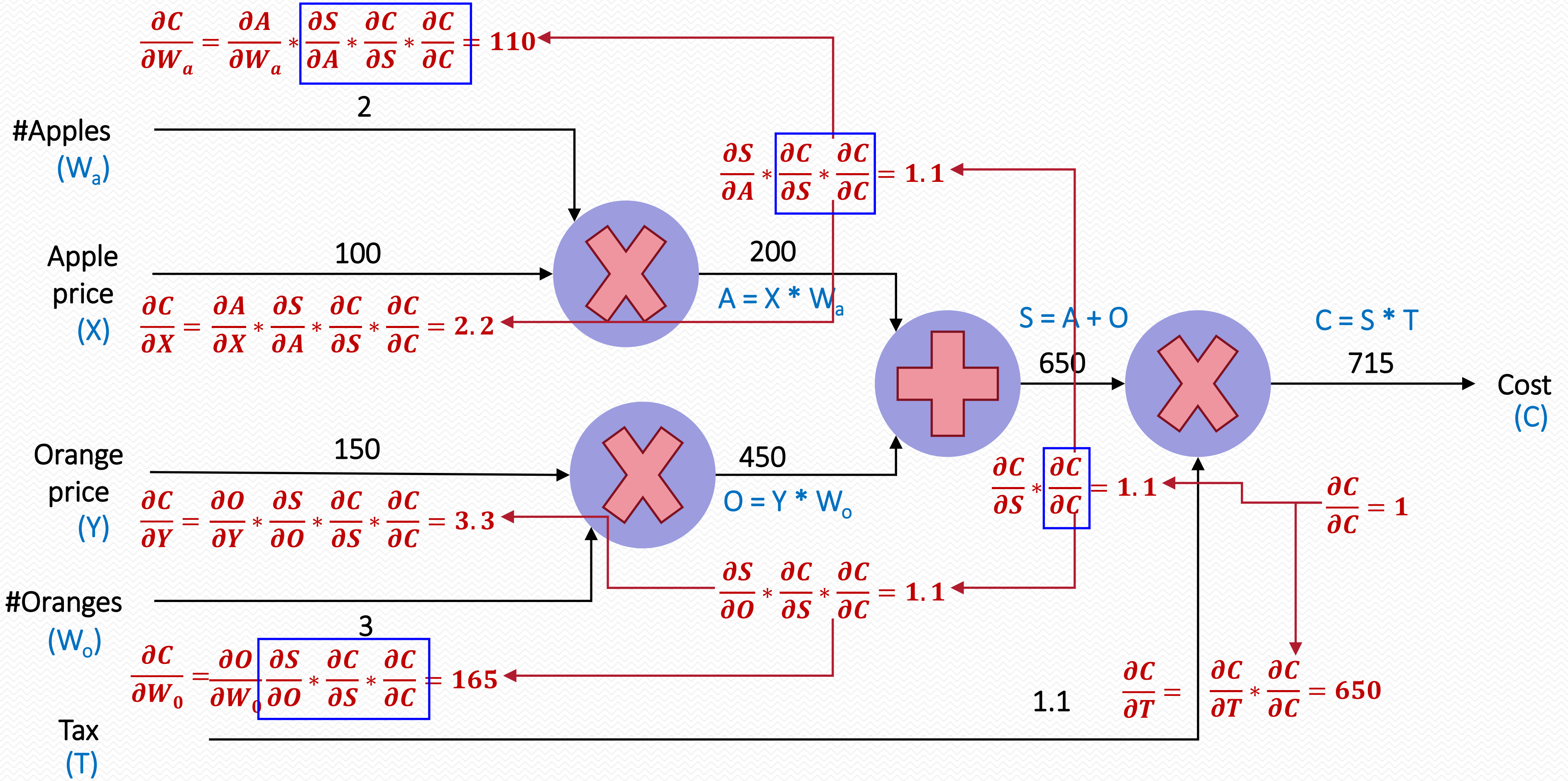
Error Back Propagation

CALCULATE THE ∂w from ∂f



Error Back Propagation

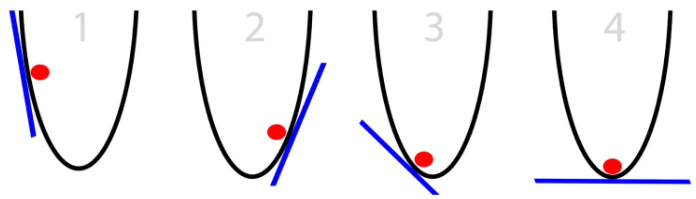
- Contributions of each parameter to final cost can be calculated from partial derivatives.



Gradient descent algorithm

Solve the following minimization problem : for $x \in \mathbf{R}^n$,

$$\min_x f(x)$$



To solve the problem, we use descent method framework : For given initial $x^{(0)}$,

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x^{(k)}$$

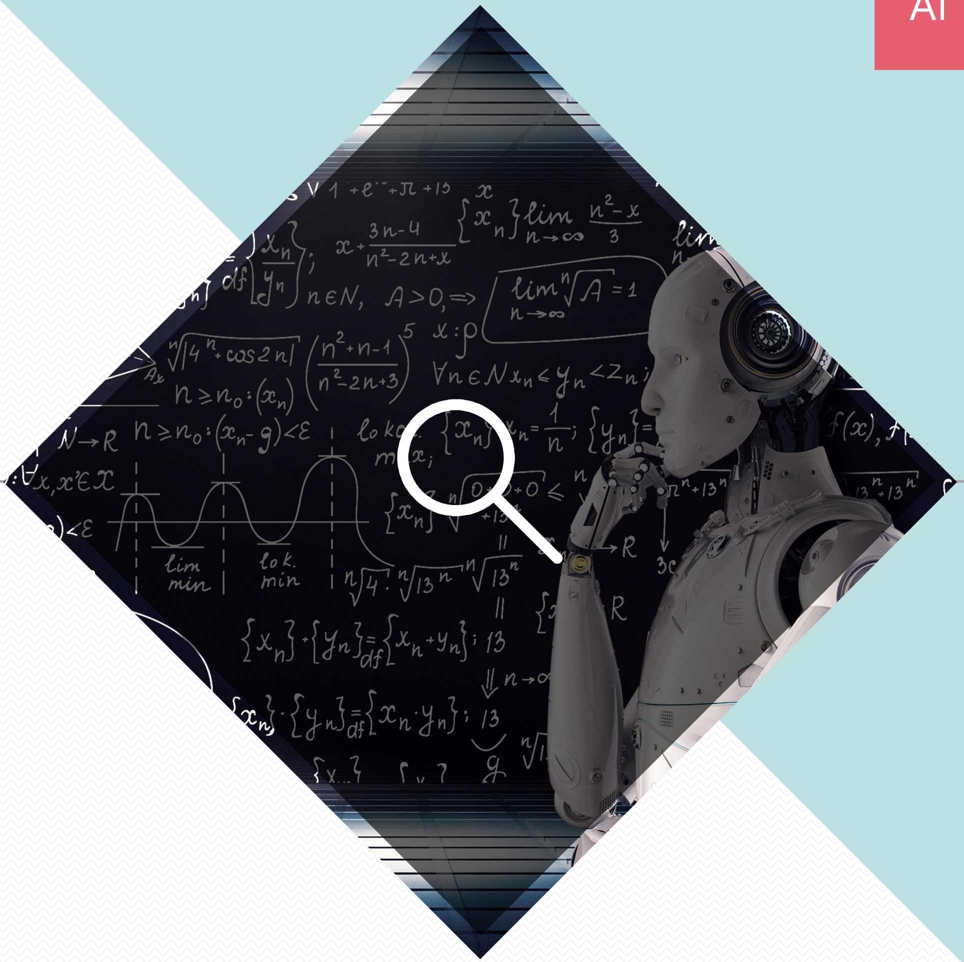
1. $\alpha^{(k)}$: Learning Rate / Step Size
2. $\Delta x^{(k)}$: Search Direction

$\boldsymbol{\alpha}$**(constant)**

$-\nabla f(x^k)$

- 비용 함수 최소화
- Gradient descent는 주로 최적화 문제에 사용된다.
- 주어진 비용 함수, 비용 (W, b)에 대해 비용을 최소화하기 위해 W, b를 찾는다.
- 일반적인 함수 : 비용 (w1, w2,...)에 적용 가능

AI TRAINING SERIES



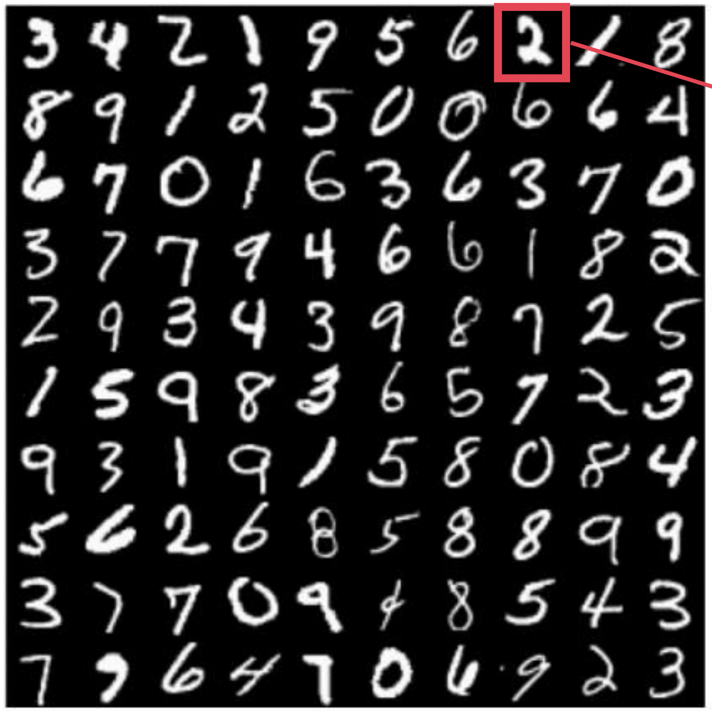
Matrix

차원의 확장

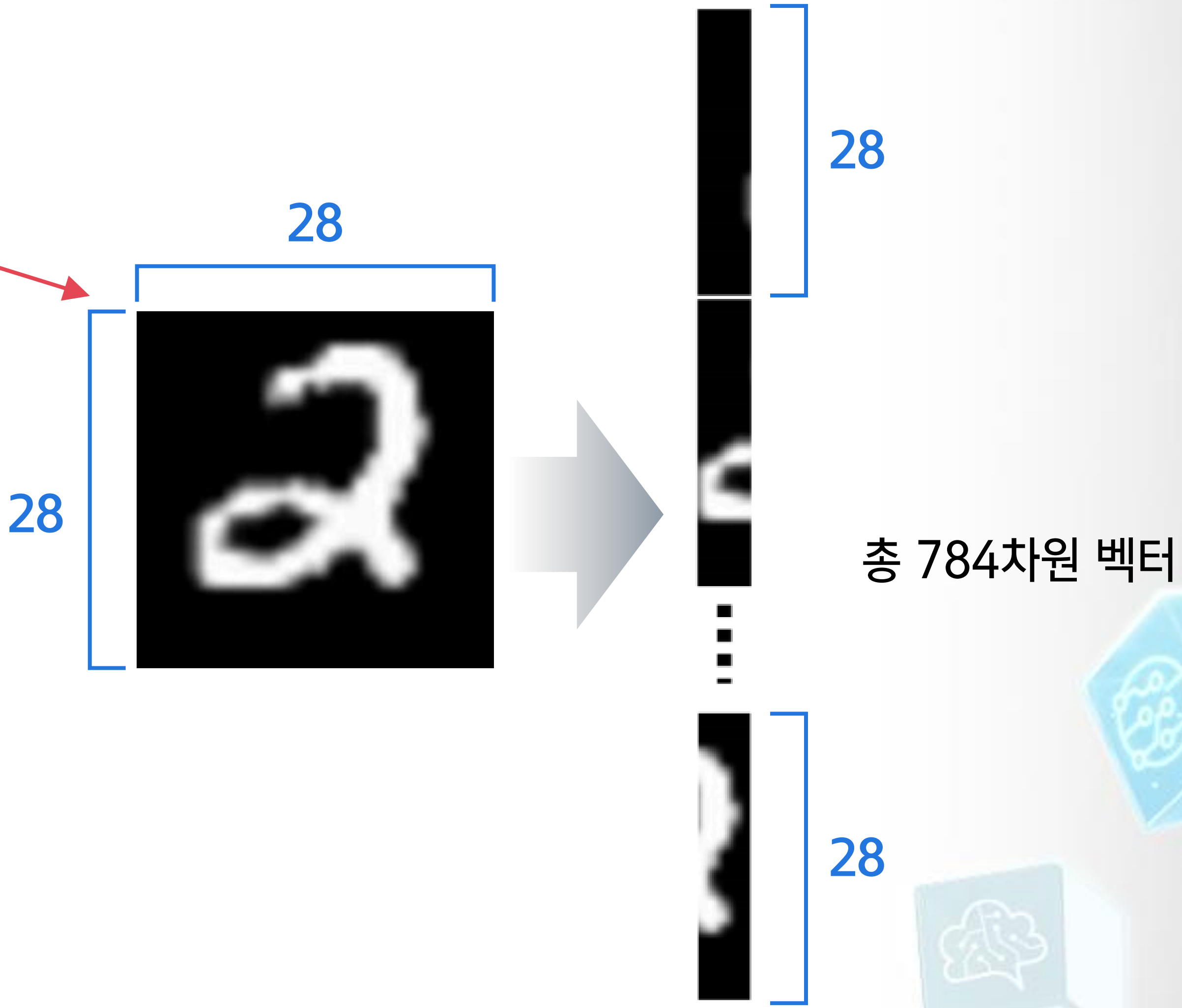
2 matrix

1) 차원의 확장

- **왜 차원을 높여야 하는가?**



〈출처1〉



2 matrix

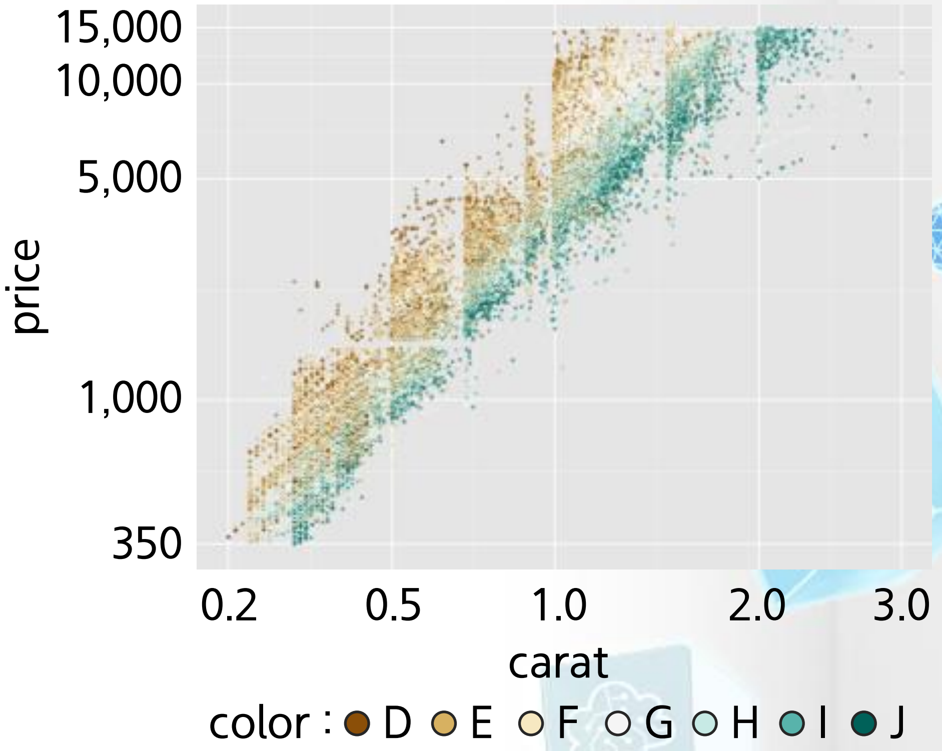
1) 차원의 확장

- **왜 차원을 높여야 하는가?**
  - 다이아몬드의 가격 예측

① 캐럿 ② 컷 ③ 색깔 ④ 투명도 ⑤ 깊이 ⑥ x방향 길이 ⑦ y방향 길이 ⑧ z방향 길이

	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
2	0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
3	0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
4	0.29	Premium	I	VS2	62.4	58	334	4.2	4.23	2.63
5	0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
6	0.24	Very Good	J	VVS2	62.8	57	336	3.94	3.96	2.48
7	0.24	Very Good	I	VVS1	62.3	57	336	3.95	3.98	2.47
8	0.26	Very Good	H	SI1	61.9	55	337	4.07	4.11	2.53
9	0.22	Fair	E	VS2	65.1	61	337	3.87	3.78	2.49

[price(log 10) by cube-root of carat and color]

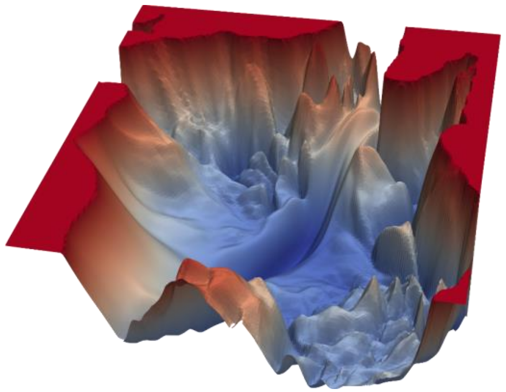


〈출처2〉

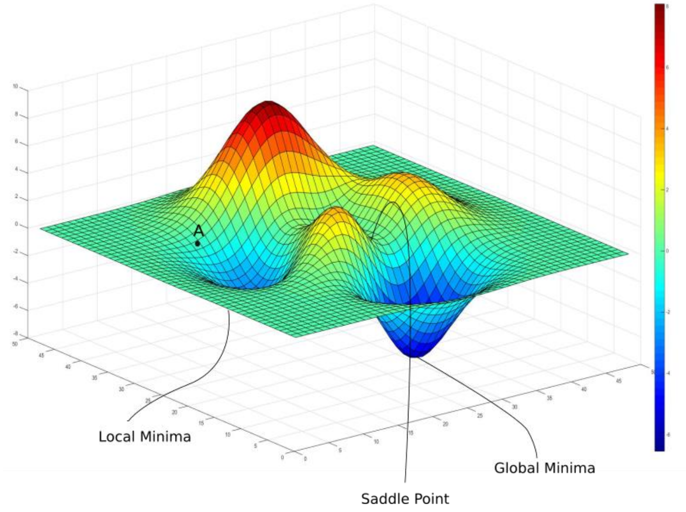
2 matrix

1) 차원의 확장

- **Local minima**



loss contour of a VGG-56 DNN's loss function



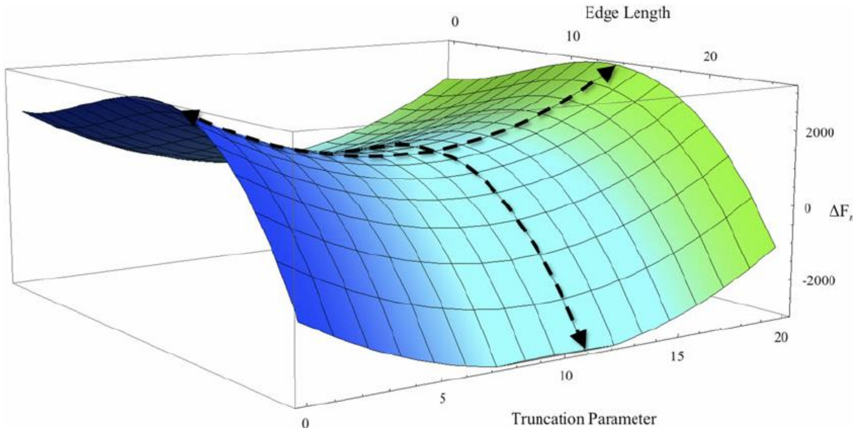
- Gradient of local and global minima is zero
- Improper initialization point may cause convergence to a local minima – you're doomed!

2 matrix

1) 차원의 확장

- **Multi dimension**

Saddle Points



- A minima in one direction, a maxima in another direction
- Occurs where two maxima meet

2 matrix

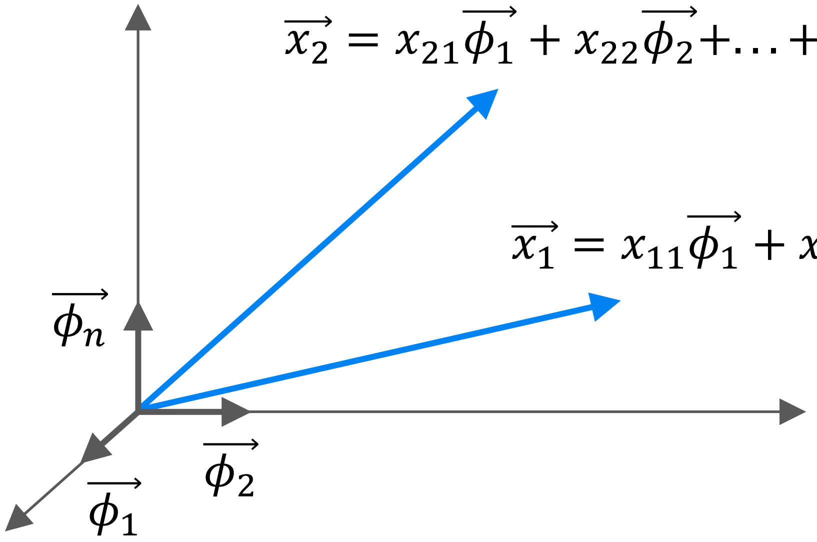
1) 차원의 확장

- **다차원 벡터의 내적**
  - 두 벡터 $\overrightarrow{x_1}, \overrightarrow{x_2}$ 의 내적은 두 벡터 간의 projection 에너지의 적분으로 정의됨
  - 두 벡터가 신호이고, 같은 신호일 경우 신호의 에너지가 됨

$$\langle \overrightarrow{x_1}, \overrightarrow{x_2} \rangle = x_{11}x_{21} + x_{12}x_{22} + \ldots + x_{1n}x_{2n}$$

$$\overrightarrow{x_2} = x_{21}\overrightarrow{\phi_1} + x_{22}\overrightarrow{\phi_2} + \ldots + x_{2n}\overrightarrow{\phi_n}$$

$$\overrightarrow{x_1} = x_{11}\overrightarrow{\phi_1} + x_{12}\overrightarrow{\phi_2} + \ldots + x_{1n}\overrightarrow{\phi_n}$$



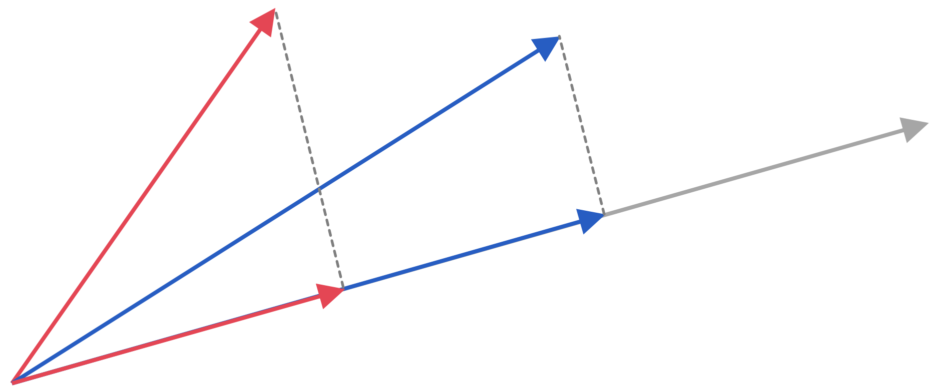
2 matrix

1) 차원의 확장

- **방향성분 추출**
  - $\overrightarrow{x_n}$ 에 관한 기저함수 $\overrightarrow{\emptyset_n}$의 가중치 혹은 계수는 $\overrightarrow{x_n}$ 와 $\overrightarrow{\emptyset_n}$의 내적으로 추출됨
  - $\overrightarrow{x_n}$ 와 $\overrightarrow{\emptyset_n}$ 의 내적은 $\overrightarrow{x_n}$ 의 $\overrightarrow{\emptyset_n}$ 방향의 정사형 성분으로 $\overrightarrow{\emptyset_n}$ 방향의 **가중치**로 해석됨

$$\vec{x} = x_1\overrightarrow{\phi_1} + x_2\overrightarrow{\phi_2} + \ldots + x_n\overrightarrow{\phi_n}$$

$$\overrightarrow{x_n} = \langle \overrightarrow{x_n}, \overrightarrow{\phi_m} \rangle$$



2 matrix

1) 차원의 확장

- **직교성과 기본 벡터(basis function)**
  - 직교성(orthogonal & orthonormal)
    - 신호 $x(t)$에 관한 식의 기저함수를 $\overrightarrow{\phi_n}$라고 할 때, 다른 기저함수 간의 내적이 0일 경우 orthogonal이라고 함
    - 동일 기저함수 간의 내적이 1일 경우, orthonormal이라고 함

$$\vec{x} = x_1\overrightarrow{\phi_1} + x_2\overrightarrow{\phi_2} + \ldots + x_n\overrightarrow{\phi_n}$$

$$\langle\overrightarrow{\phi_n}, \overrightarrow{\phi_m}\rangle = \begin{cases} 1 & n = m \\ 0 & n \neq m \end{cases} \Rightarrow \text{Set } \{\overrightarrow{\phi_n}\} \text{ are Orthonormal}$$

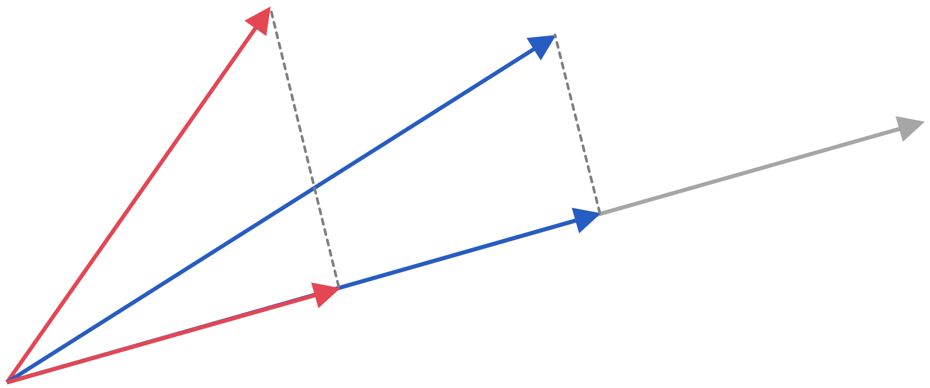
2 matrix

1) 차원의 확장

- **sin/cos함수 기본 벡터**
  - 지수 기저함수(파동의 주파수 기본으로 모델링)

$$\vec{x} = x_1\overrightarrow{\phi_1} + x_2\overrightarrow{\phi_2} + \ldots + x_n\overrightarrow{\phi_n}$$

$$\overrightarrow{x_n} = \langle \overrightarrow{x_n}, \overrightarrow{\phi_m} \rangle$$



$$\phi_m(t) = e^{-i2\pi f_0 t}$$

$$e^{-i2\pi f_0 t} = \cos 2\pi f_0 t + \mathrm{i}\sin 2\pi f_0 t \qquad f_0 = {}^{1}/_{T_0}$$

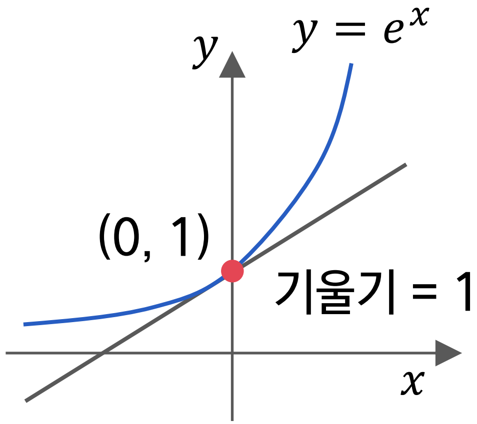
1) 차원의 확장

- **자연지수함수**

변화율이 함수가 되는 함수를 구할 수 있을까?

- 전염병의 증가, 세균의 번식, 자연에서의 불길의 번짐, 대기권에서의 유전율의 변화, …
- 자연지수함수(natural exponential function)로 모델링됨

$$f(x) = f'(x) \Leftrightarrow f(x) = e^x$$



$$e^x = \lim_{n\to\infty}\left(1 + \frac{x}{n}\right)^n$$

오힐러의 공식

$$e^{ix} = \cos x + i\sin x$$
$$e^{-ix} = \cos x - i\sin x$$

2 matrix

1) 차원의 확장

- **다차원 벡터의 내적**
  - 벡터 간의 연산이므로 차원이 같아야 함
    - **내적의 결과는 스칼라!!**

$$\langle x, y\rangle = x^T y$$

$$x = \begin{bmatrix}1\\2\\3\end{bmatrix}, y = \begin{bmatrix}5\\-2\\1\end{bmatrix} \qquad x^T y = \begin{bmatrix}1 & 2 & 3\end{bmatrix}\begin{bmatrix}5\\-2\\1\end{bmatrix} = 4$$

np.dot(A,x)

2 matrix

1) 차원의 확장

- **matrix 확장**
  - matrix로 확장하면 여러 벡터의 내적을 한꺼번에 진행 가능

$$\langle x, y\rangle = x^T y$$

$$x^T y = \begin{bmatrix}1 & 2 & 3\end{bmatrix}\begin{bmatrix}5\\-2\\1\end{bmatrix} = 4 \quad \Rightarrow \quad \underbrace{\boxed{\begin{bmatrix}1 & 2 & 3\\1 & 2 & 3\\1 & 2 & 3\end{bmatrix}}}_{\text{matrix}}\begin{bmatrix}5\\-2\\1\end{bmatrix} = \begin{bmatrix}4\\4\\4\end{bmatrix}$$

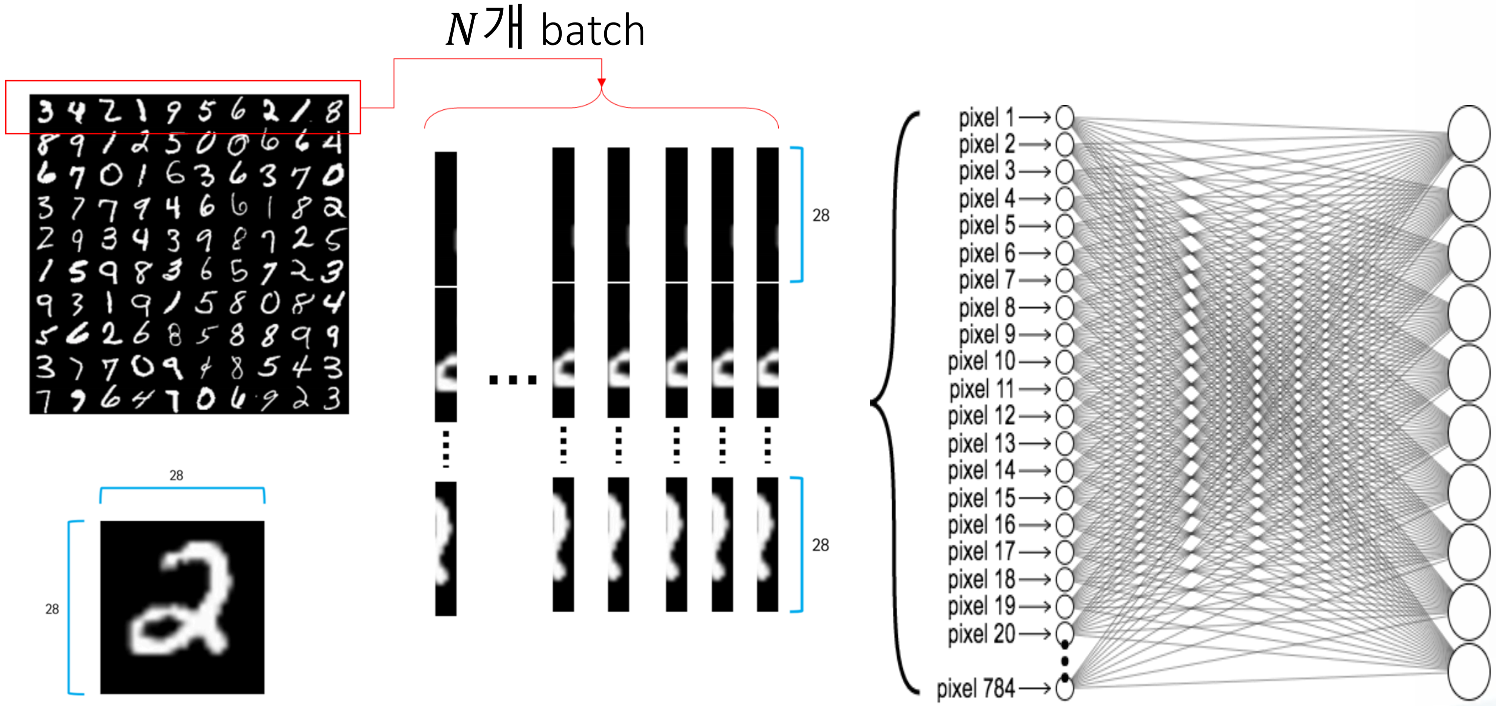
$$\underbrace{\begin{bmatrix}\mathbf{1} & \mathbf{2} & \mathbf{3}\\\mathbf{1} & \mathbf{2} & \mathbf{3}\\\mathbf{1} & \mathbf{2} & \mathbf{3}\end{bmatrix}}_{\text{matrix}}\underbrace{\begin{bmatrix}\mathbf{5} & \mathbf{5} & \mathbf{5}\\\mathbf{-2} & \mathbf{-2} & \mathbf{-2}\\\mathbf{1} & \mathbf{1} & \mathbf{1}\end{bmatrix}}_{\text{matrix}} = \underbrace{\begin{bmatrix}\mathbf{4} & \mathbf{4} & \mathbf{4}\\\mathbf{4} & \mathbf{4} & \mathbf{4}\\\mathbf{4} & \mathbf{4} & \mathbf{4}\end{bmatrix}}_{\text{matrix}}$$

np.dot(A,x)

2 matrix

1) 차원의 확장

- **matrix 확장**
  - matrix로 확장하면 여러 벡터의 내적을 한꺼번에 진행 가능



$$W\begin{bmatrix}x_1\\ \vdots\\ x_{784}\end{bmatrix}=\begin{bmatrix}y_1\\ \vdots\\ y_{784}\end{bmatrix} + \dots + W\begin{bmatrix}x_N\\ \vdots\\ x_{784}\end{bmatrix}=\begin{bmatrix}y_N\\ \vdots\\ y_{784}\end{bmatrix} = W\begin{bmatrix}x_1 & \cdots & x_N\\ \vdots & \cdots & \vdots\\ x_{784} & \cdots & x_{N,784}\end{bmatrix}=\begin{bmatrix}y_1 & \cdots & y_N\\ \vdots & \cdots & \vdots\\ y_{784} & \cdots & y_{N,784}\end{bmatrix}$$

2 matrix

2) matrix 곱셈

Numpy lib array의 A.dot(B)를 이용해 A와 B의 행렬곱(내적)을 구할 수 있음

행! 렬(열)!

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \boxed{a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4} \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 4 & 2 & 0 \\ 9 & 5 & 0 & 0 \\ 4 & 0 & 2 & 4 \\ 6 & 1 & 8 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 1\cdot 1 + 4\cdot 2 + 2\cdot 3 + 0\cdot 4 \\ 9\cdot 1 + 5\cdot 2 + 0\cdot 3 + 0\cdot 4 \\ 4\cdot 1 + 0\cdot 2 + 2\cdot 3 + 4\cdot 4 \\ 6\cdot 1 + 1\cdot 2 + 8\cdot 3 + 3\cdot 4 \end{bmatrix} = \begin{bmatrix} 15 \\ 19 \\ 26 \\ 44 \end{bmatrix}$$

2 matrix

3) matrix 선형 방정식

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$
$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2$$
$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

$$Ax = b$$

b=np.matmul(A,x)

b=A@x

4) 역행렬

역행렬을 이용하여 선형 방정식의 해를 찾을 수 있음(np.linalg.inv(A))

- 원래행렬 A에 역행렬 A^{-1}를 곱하면 단위행렬(I)을 얻을 수 있음
- 역행렬은 np.linalg.inv(A)를 이용하여 구할 수 있음
- 판별식(determinant)값이 0이면 역행렬은 존재하지 않음
- 고차원 행렬의 판별식을 구하는 방법은 np.linalg.det를 사용함
- 단위행렬은 판별식이 1이고, 어떤 행렬을 곱해도 자기 자신이 나오는 행렬임

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}^{-1} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$AA^{-1} = A^{-1}A = I$$

2 matrix

4) 역행렬

- **가우스-조단 소거법**

$$\begin{matrix} x+3y=5 \\ 3x+5y=11 \end{matrix} \rightarrow \begin{matrix} 3x+9y=15 \\ 3x+5y=11 \end{matrix} \rightarrow \begin{matrix} 3x+9y=15 \\ 4y=4 \end{matrix} \rightarrow \begin{matrix} \frac{1}{3}x+y=\frac{5}{3} \\ y=1 \end{matrix} \rightarrow \begin{matrix} \frac{1}{3}x=\frac{2}{3} \\ 4y=4 \end{matrix} \rightarrow \begin{matrix} x=2 \\ y=1 \end{matrix}$$

$$\left[\begin{array}{cc|c} 1 & 3 & 5 \\ 3 & 5 & 11 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 3 & 9 & 15 \\ 3 & 5 & 11 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 3 & 9 & 15 \\ 0 & 4 & 4 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} \frac{1}{3} & 1 & \frac{5}{3} \\ 0 & 1 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 1 & 0 & 2 \\ 0 & 1 & 1 \end{array}\right]$$

$$\left[\begin{array}{cc|cc} a & b & 1 & 0 \\ c & d & 0 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cc|cc} a & b & 1 & 0 \\ 0 & d-\frac{bc}{a} & -\frac{c}{a} & 1 \end{array}\right] \rightarrow \left[\begin{array}{cc|cc} 1 & \frac{b}{a} & 1 & 0 \\ 0 & 1 & -\frac{c}{ad-bc} & \frac{a}{ad-bc} \end{array}\right] \rightarrow \left[\begin{array}{cc|cc} 1 & 0 & \frac{d}{ad-bc} & -\frac{b}{ad-bc} \\ 0 & 1 & -\frac{c}{ad-bc} & \frac{a}{ad-bc} \end{array}\right]$$

2 matrix
4) 역행렬

역행렬을 이용하여 선형 방정식의 해를 찾을 수 있음(np.linalg.inv(A))

$$A^{-1} = \frac{1}{Det(A)} \begin{bmatrix} C_{11} & \cdots & C_{1n} \\ \vdots & \ddots & \vdots \\ C_{n1} & \cdots & C_{nn} \end{bmatrix}$$

$$\boldsymbol{C_{ij} = (-1)^{i+j} Det(M_{ij})}$$

$$M_{ij} = \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{n1} & \cdots & A_{nn} \end{bmatrix}$$

i th row 제외

i th column 제외

5) 전치행렬

- **matrix A의 transpose**

$$\begin{bmatrix} 4 & 5 & 2 & 1 \\ 2 & 3 & 8 & 0 \\ 1 & 0 & 7 & 2 \end{bmatrix}^T = \begin{bmatrix} 4 & 2 & 1 \\ 5 & 3 & 0 \\ 2 & 8 & 7 \\ 1 & 0 & 2 \end{bmatrix}$$

- **symmetric matrix A**

$$A^T = A$$

$$\begin{bmatrix} 4 & 2 & 1 \\ 2 & 3 & 0 \\ 1 & 0 & 7 \end{bmatrix}^T = \begin{bmatrix} 4 & 2 & 1 \\ 2 & 3 & 0 \\ 1 & 0 & 7 \end{bmatrix}$$

$$\left(A_{ij}\right)^T = A_{ji}$$

A.T

2 matrix

1) 차원의 확장

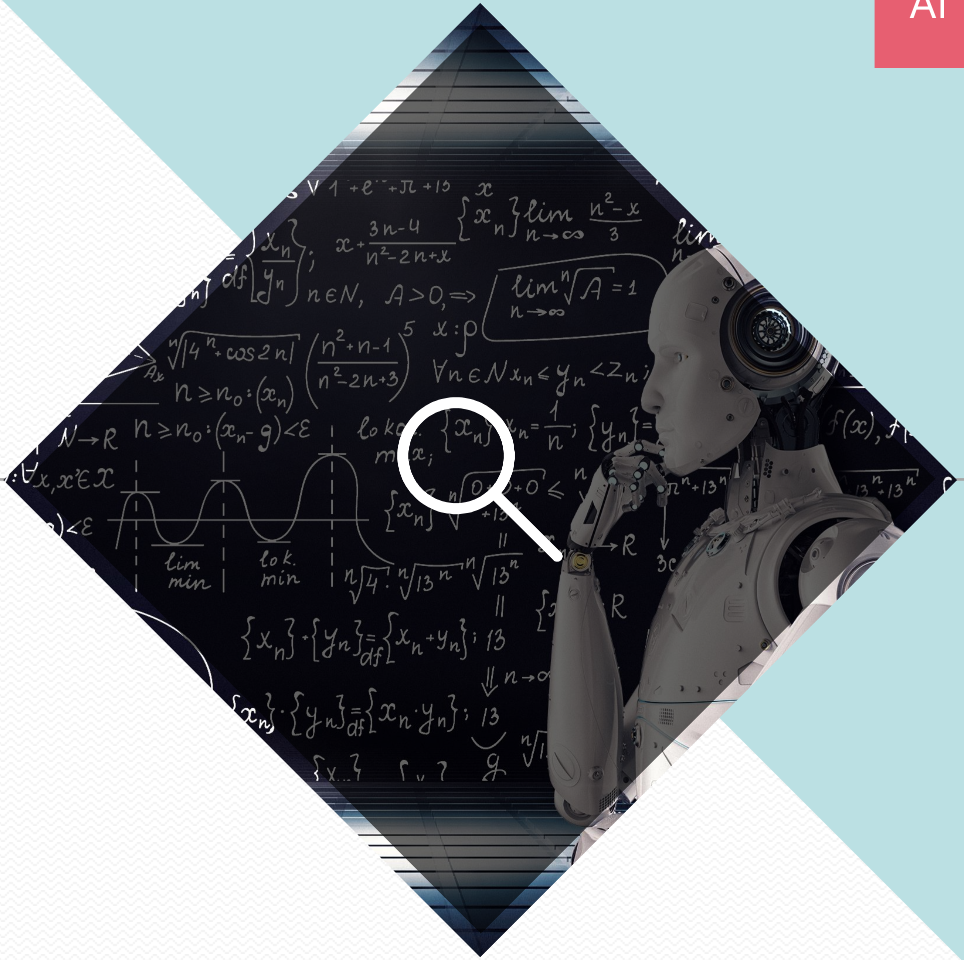
- **norm**

$$\|x\|_p = \left(\sum_i |x_i|^p\right)^{\frac{1}{p}} \qquad \text{for } p \in \mathbb{R}, \mathrm{p} \geq 1.$$

np.linalg.norm(A,ord=1)

L1 놈	L2 놈	무한 놈
벡터와 원소의 절대값의 합	유클리디안 거리로 계산된 백터의 길이	벡터의 원소 중 절대값이 가장 큰 값

AI TRAINING SERIES



Entropy

What is Vector? How is it Defined?

2 Entropy

1) 확률

표본 공간 (정의역)	X: 확률변수	P_X
값 1	1	1/6
값 2	2	1/6
값 3	3	1/6
값 4	4	1/6
값 5	5	1/6
값 6	6	1/6

확률 사상

- 실험 결과들에 대해 어떤 규칙을 정하여 값을 부여하는데, 이러한 규칙을 확률변수라 함
- 가능한 모든 상황에서 어떤 이벤트가 발생할 수 있는 빈도수가 확률

2) 평균과 분산

평균

$$m_X = E\left[X^n\right] = \sum_{i=-\infty}^{\infty} i^n P(X = i)$$

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

분산

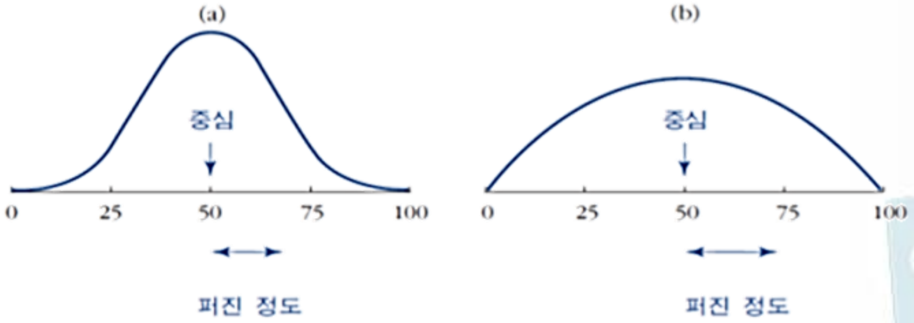
$$\sigma^2 = Var[X] = E\left[(X - m)^2\right]$$

$$= E\left[X^2 - 2mX + m^2\right]$$

$$= E\left[X^2\right] - E[2mX] + E\left[m^2\right]$$

$$= E\left[X^2\right] - m^2$$

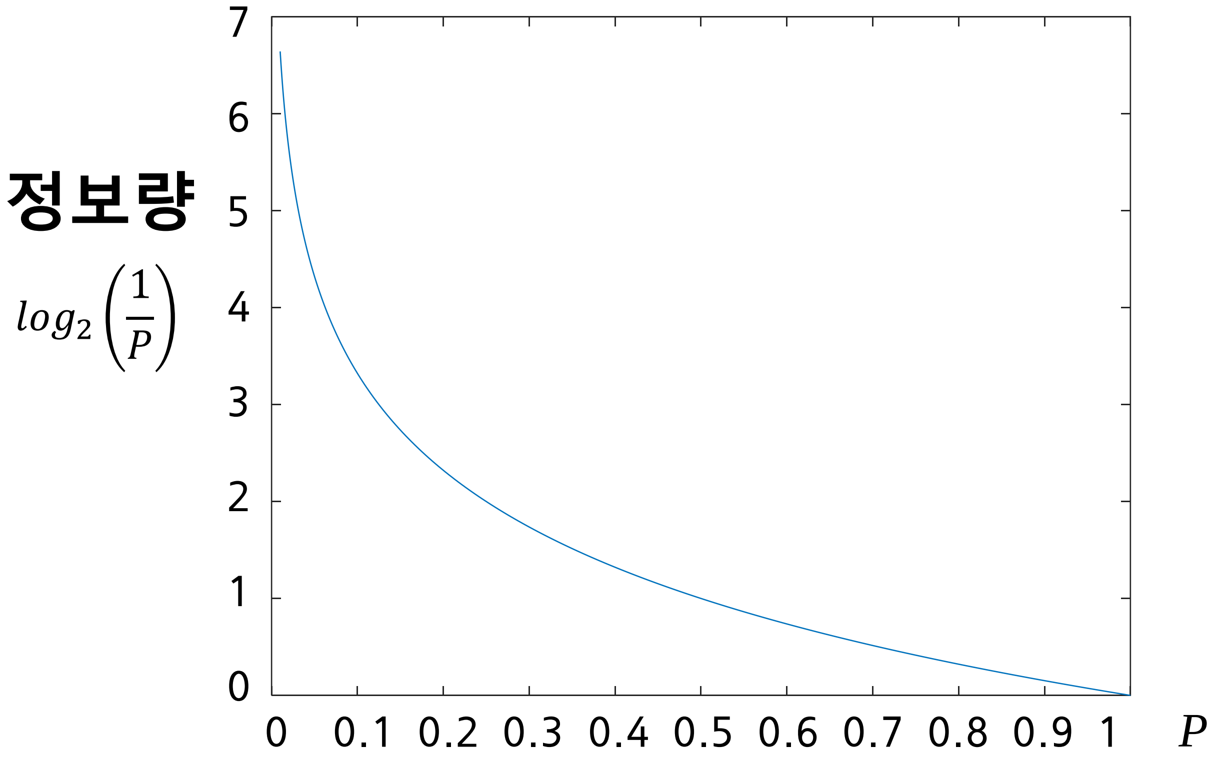
σ : 표준편차
σ^2 : 분산



3) 소스 심볼의 정보량

- 소스 심볼에 대한 정보량 I 는 심볼의 발생 확률(의외성) P로부터 계산됨

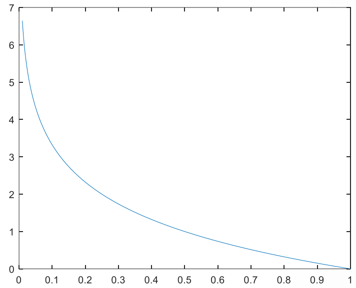
$$p = \frac{1}{2^I} \Rightarrow I = log_2\left(\frac{1}{P}\right) = -log_2 P \text{ [비트]}$$



2 Entropy

3) 소스 심볼의 정보량

- 소스 심볼의 정보량($log_2\left(\frac{1}{P}\right)$)
  - $X_1, X_2, X_3, \dots, X_M$으로 나타내어지는 M개의 심볼이 있는 경우
  - 심볼이 일어날 수 있는 확률 : $P_1, P_2, P_3, \dots, P_M$
  - 심볼 X_1이 M개의 총 정보 속에서 차지하는 비율 : $-P_1 log_2 P_1$
  - **심볼의 평균 정보량**
    - 엔트로피(entropy, 무질서도)로 표현
    - 샤논(Shannon)이 수학적으로 정의



$$E(X) = \sum_{m=0}^{M-1} p_m \log_2(1/p_m) = -\sum_{m=0}^{M-1} p_m \log_2 p_m \text{ [비트/심볼]}$$

- Entropy : $-\sum_{m=0}^{M-1} p_m \log_2 p_m$
- Cross-Entropy: $-\sum_{m=0}^{M-1} p_m \log_2 q_m$
- KL-divergence: $-\sum_{m=0}^{M-1} p_m \log_2 p_m + \sum_{m=0}^{M-1} p_m \log_2 q_m$

- 전혀 정보를 가지지 못하는 $M = 1$인 경우 심볼의 확률은 1이고, $X = 0$
- $\boxed{C(H(x), y) = -y * log(H(X)) - (1-y) * log(1 - H(X))}$

2 Entropy

3) 소스 심볼의 정보량 : Logistic regression 의 cost 와 각 분류모델의 기대값

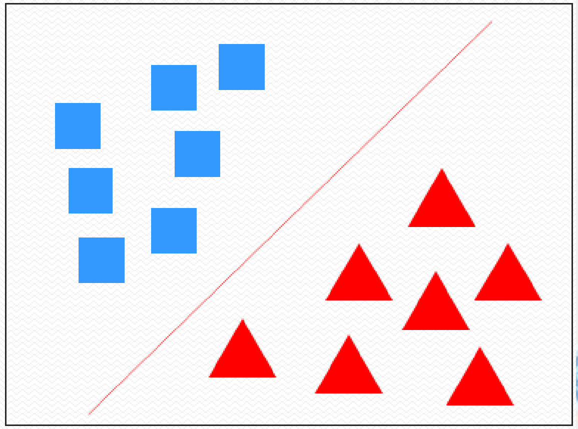
- Cost function with Condition

$$C(H(x),y) = \begin{cases} -\log(H(X)) & if\ y = 1 \\ -\log(1-H(x)) & if\ y = 0 \end{cases}$$

Single Function expression

$$\boxed{C(H(x),y) = -y * log(H(X)) - (1-y) * log(1-H(X))}$$

Categorical crossentropy func.



2 Entropy

4) 조건부 확률

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{\sum_{j=1}^{n} P(B \mid A_j)P(A_j)}$$

베이지안 이론

- 베이지안 이론은 두 종류의 조건부 확률 사이의 관계를 정의함
- 다양한 분야에서 활용되는 대표적인 기법

Likelihood

Class Prior Probability

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Posterior Probability

Predictor Prior Probability

P(A), 사전확률(prior probability) : 결과가 나타나기 전에 결정되어 있는 A(원인)의 확률.

P(B|A), 우도확률(likelihood probability) : A(원인)가 발생하였다는 조건하에서 B(결과)가 발생할 확률.

P(A|B), 사후확률(posterior probability) : B(결과)가 발생하였다는 조건하에서 A(원인)가 발생하였을 확률.

$$P(\text{질병} \mid \text{양성}) = \frac{P(\text{양성} \mid \text{질병})P(\text{질병})}{P(\text{양성})} = \frac{0.99 \times 0.01}{0.01 \times 0.99 + 0.99 \times 0.10} = 0.091$$

THANK YOU