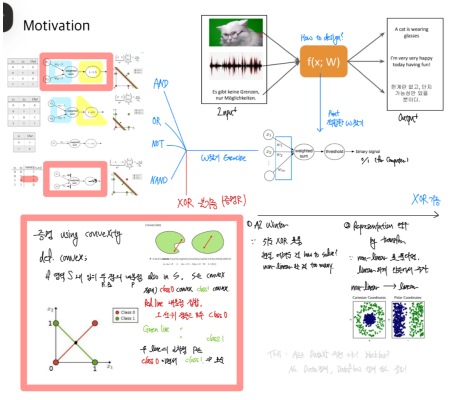
Lec2. Perceptrons

Motivation



Motivation

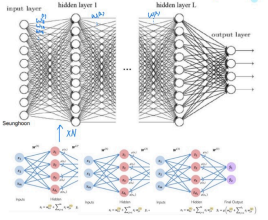
XOR-정의

Multi-layer Perceptrons

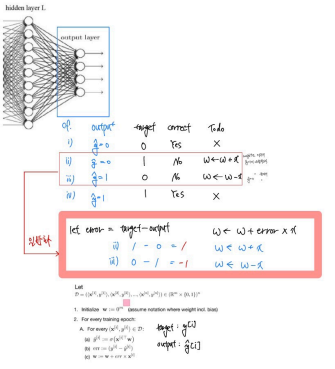
Learning MLP

Learning MLP

Motivation



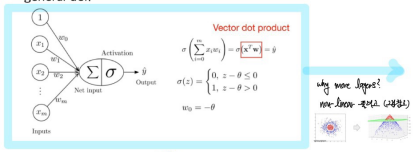
Perceptron Learning Algorithm



Multi-layer perceptons

Motivation: Single-layer perceptrons

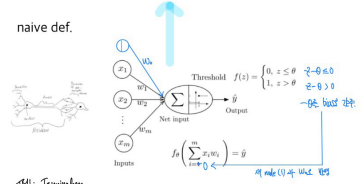
general def.



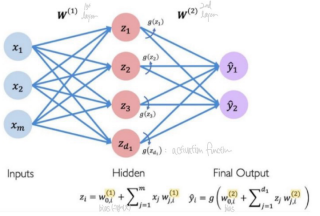
why more layers?

non-linear 변환의 (고정점)

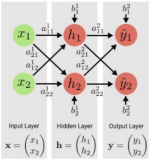
naive def.



MLP: diagram

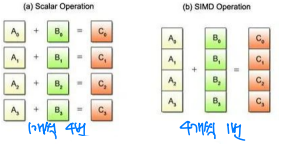


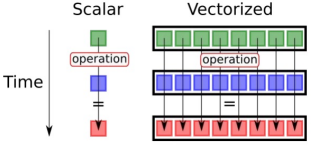
MLP: matrix notation



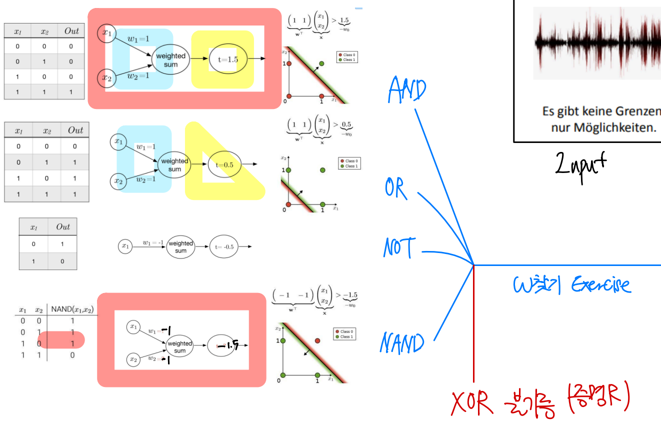
► Example MLP with $L = 2$ layers of width 2 and parameters $\{\mathbf{A}_1, \mathbf{A}_2, \mathbf{b}_1, \mathbf{b}_2\}$

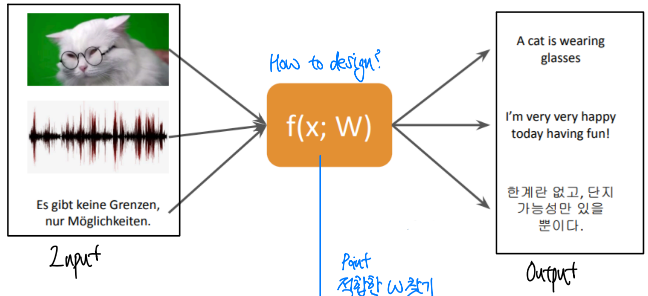
TMI: vectorization (벡터계산)

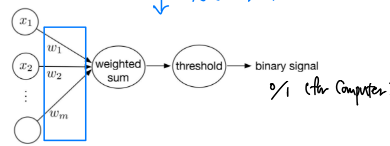


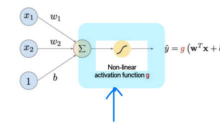


Motivation





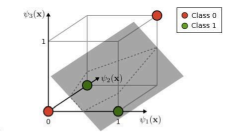




Linear classifier with non-linear features ψ:

$$\mathbf{w}^\top \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{pmatrix}}_{\psi(\mathbf{x})} > -w_0$$

x_1	x_2	$\psi_1(\mathbf{x})$	$\psi_2(\mathbf{x})$	$\psi_3(\mathbf{x})$	XOR
0	0	0	0	0	0
0	1	0	1	0	1
1	0	1	0	0	1
1	1	1	1	1	0



- Non-linear features allow linear classifier to solve non-linear classification problems!
- Analogous to polynomial curve fitting

25

XOR 가능

Sol 1. non-linear feature 사용

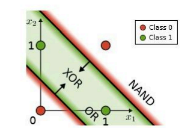
Sol 2. Stacking (Gate 중첩)

Linear Classifier:

Class 1 ⇔ $\mathbf{w}^\top \mathbf{x} > -w_0$

$\mathrm{XOR}(x_1, x_2) =$
$\mathrm{AND}(\mathrm{OR}(x_1, x_2), \mathrm{NAND}(x_1, x_2))$

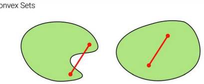
x_1	x_2	$\mathrm{XOR}(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	0



증명 using convexity

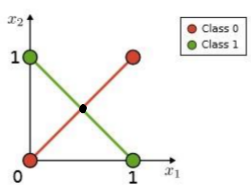
def. convex;

if 영역 S 내 임의 두 점의 내부점 also in S, S는 convex
(P_1, P_2) (P)



► A set $\mathcal{S}$ is **convex** if any line segment connecting 2 points in $\mathcal{S}$ lies entirely within $\mathcal{S}$.

$x_1, x_2 \in \mathcal{S} \Rightarrow \lambda x_1 + (1-\lambda) x_2 \in \mathcal{S}$ for $\lambda \in [0,1]$



가정) class 0 convex, class 1 convex.

Red line 내부점 집합,

그 선 위 점들은 모두 class 0

Green line " " class 1

두 line의 교차점 P는

class 0 이면서 class 1. ⇒ 모순

① AI Winter

∵ 당시 XOR 못풀

현실 여러가지 how to solve?

non-linear한 것 too many.

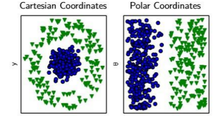
② Representation 연구

by transform

∵ non-linear 못 푼다면,

linear하게 만들어서 주자.

non-linear ⟶ linear



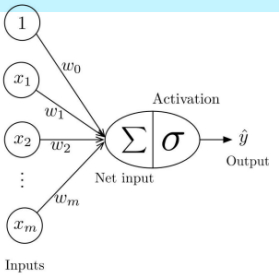
TMI : AI는 Data만 주면 OK? black box?

No. Data 정제, Data flow 설계 등도 중요!

Multi-layer perceptons

Motivation: Single-layer perceptrons

general def.



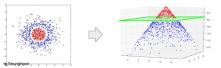
Vector dot product

$$\sigma\left(\sum_{i=0}^{m} x_i w_i\right) = \sigma(\mathbf{x}^T\mathbf{w}) = \hat{y}$$

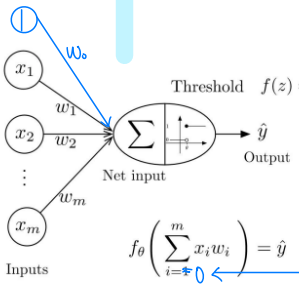
$$\sigma(z) = \begin{cases} 0, & z - \theta \le 0 \\ 1, & z - \theta > 0 \end{cases}$$

$$w_0 = -\theta$$

why more layers?
non-linear 좋아서 (고양창조)



naive def.



$$f(z) = \begin{cases} 0, & z \le \theta \\ 1, & z > \theta \end{cases}$$

$z-\theta \le 0$
$z-\theta > 0$

$-\theta$는 bias 값과.

$$f_\theta\left(\sum_{i=0}^{m} x_i w_i\right) = \hat{y}$$

x_0 node (1) 와 w_0 곱셈

TMI: Terminology

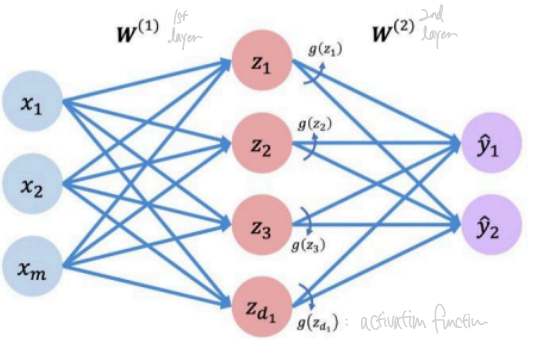
New input $z = x_1 w_1 + x_2 w_2 + \cdots + x_m w_m + b$

$= \sum_{i=1}^{m} x_i w_i + b = \mathbf{x}^T \cdot \mathbf{w} + b$

Activations $a = b(z)$ s.t b: activation function (= threshold function in MLP)

~~label output $\hat{y} = f(a)$ s.t f: threshold function~~ $\hat{y} = a$

MLP: diagram

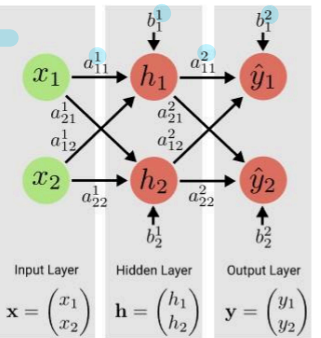


$$z_i = w_{0,i}^{(1)} + \sum_{j=1}^{m} x_j w_{j,i}^{(1)} \qquad \hat{y}_i = g\left(w_{0,i}^{(2)} + \sum_{j=1}^{d_1} z_j w_{j,i}^{(2)}\right)$$

bias (그림에서) bias

MLP: matrix notation

layer:



nonlinear function g, h (is bold h)

$$\mathbf{h} = g(\mathbf{A}_1\mathbf{x} + \mathbf{b}_1)$$
$$\mathbf{y} = h(\mathbf{A}_2\mathbf{h} + \mathbf{b}_2)$$

$$\mathbf{A}_l = \begin{pmatrix} a_{11}^l & a_{12}^l \\ a_{21}^l & a_{22}^l \end{pmatrix}$$

$$\mathbf{b}_l = \begin{pmatrix} b_1^l \\ b_2^l \end{pmatrix}$$

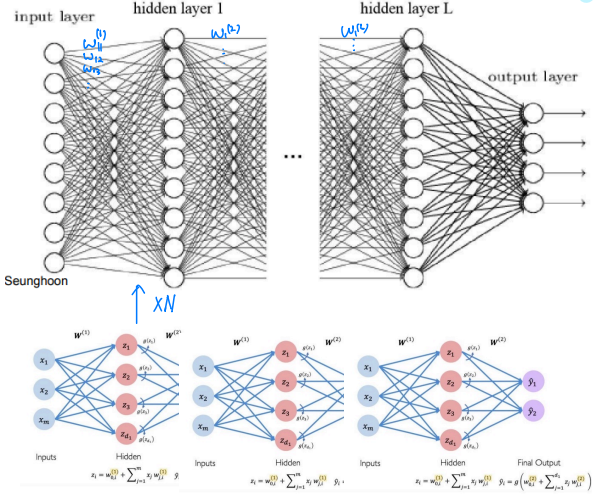
* g,h : activation function

- Example MLP with $L = 2$ layers of width 2 and parameters $\{\mathbf{A}_1, \mathbf{A}_2, \mathbf{b}_1, \mathbf{b}_2\}$

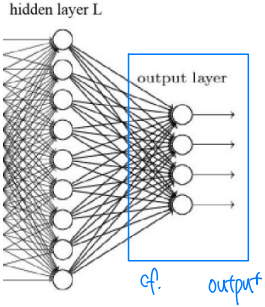
Learning MLP

Motivation

w_i는 언제 수정 할까, 자동 수정 algorithm 필요



Perceptron Learning Algorithm



cf.	output	target	correct	To do
i)	$\hat{y}=0$	0	Yes	X
ii)	$\hat{y}=0$	1	No	$w \leftarrow w + x$
iii)	$\hat{y}=1$	0	No	$w \leftarrow w - x$
iv)	$\hat{y}=1$	1	Yes	X

반대로

let error = target − output

		$w \leftarrow w + \text{error} \times x$
ii)	$1 - 0 = 1$	$w \leftarrow w + x$
iii)	$0 - 1 = -1$	$w \leftarrow w - x$

Let

$\mathcal{D} = (\langle \mathbf{x}^{[1]}, y^{[1]} \rangle, \langle \mathbf{x}^{[2]}, y^{[2]} \rangle, ..., \langle \mathbf{x}^{[n]}, y^{[n]} \rangle) \in (\mathbb{R}^m \times \{0,1\})^n$

1. Initialize $\mathbf{w} := 0^m$ (assume notation where weight incl. bias)
2. For every training epoch:
   A. For every $\langle \mathbf{x}^{[i]}, y^{[i]} \rangle \in \mathcal{D}$:
      (a) $\hat{y}^{[i]} := \sigma(\mathbf{x}^{[i]\top} \mathbf{w})$
      (b) $\text{err} := (y^{[i]} - \hat{y}^{[i]})$
      (c) $\mathbf{w} := \mathbf{w} + err \times \mathbf{x}^{[i]}$

target : $y[i]$

output : $\hat{y}[i]$