http://math.science.cmu.ac.th/thawinan

Introduction: Learning System

(เขียนโปรแกรม, ไม่รู้ตัวแปรจริง → หา f)

F เป็นฟังก์ชันในธรรมชาติที่ต้องการเรียนรู้ (คำตอบที่ไม่ทราบแต่ใช้จริง)

$F: X \to Y$

input / output

Ex: X ข้อมูลรูปภาพที่เป็นของสัตว์เลี้ยง.

Y . แมว, หมา, ...

(คำตอบที่ถูกต้องในโลกจริง)

ข้อมูลตัวอย่าง

$(x_1,y_1),(x_2,y_2)$

$\dots(x_N,y_N)$

<-- $\{(x_i,y_i);\ i=1,\dots,N\} \subseteq X\times Y$

ข้อมูลตัวอย่าง สามารถสร้างได้ง่าย

แต่ต้องมีจำนวนที่เพียงพอ

Learning Algorithm ⟹ output: $f \approx F$

$f: X \to Y$

จะเรียนอย่างไร.

คาดหวังว่า f ที่ได้ จะใกล้เคียงกับ F

Hypothesis set

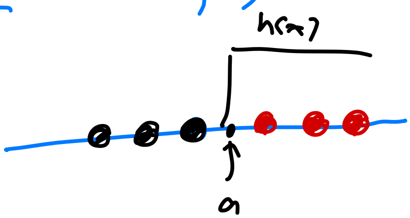
(Neuron Networks)

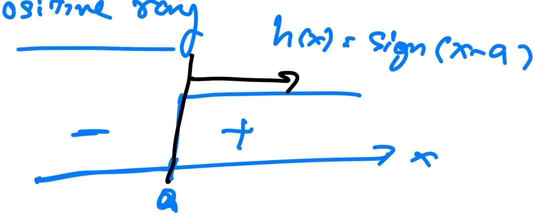
• ต้องมี Idea และ Methods จำกัดขนาดของ Hypothesis set

Ex: ให้แยกกลุ่มจุด ● และ ⊖ บนเส้นจำนวน.

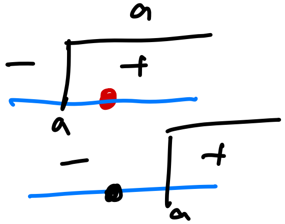
Idea ต้องการแบ่งกลุ่ม

• positive ray





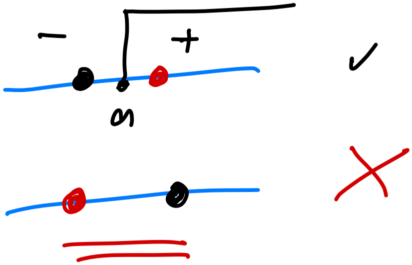
1 จุด:



breakpoints:

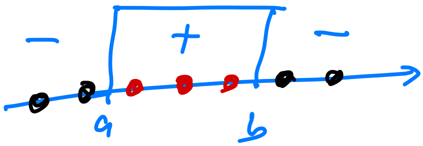
2 จุด

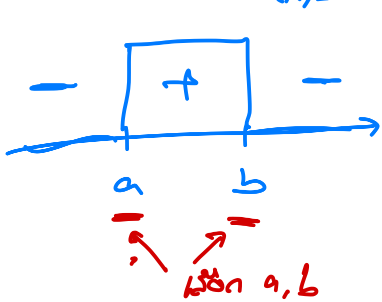
2 จุด :



$h(x) = -\text{sign}((x-a)(x-b))$

② : positive interval.

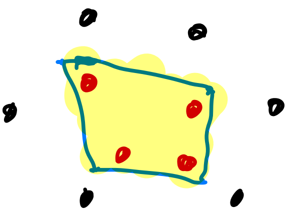


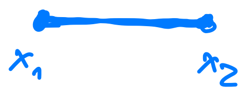


break points :

3 จุด.

③ : convex set.



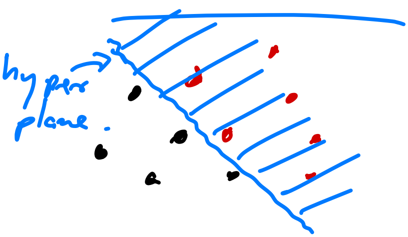


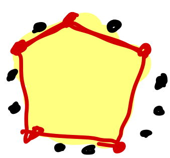
ถ้า x_1, x_2 อยู่ในขอบเขต แล้ว., $\omega \in [0,1]$

$x = \omega x_1 + (1-\omega)x_2$

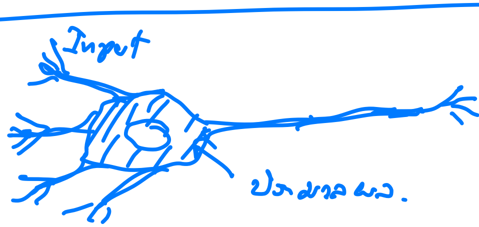
ต้องอยู่ในขอบเขตด้วย.

break points : ∞





perceptron :



Def: เรากล่าวว่า formal neuron เป็นข้อมูลในรูปแบบ

(X, Y, σ, s) ที่ซึ่ง $X \subseteq \mathbb{R}^n$ ($n > 0$ เป็นจำนวนเต็ม.)

$Y \subseteq \mathbb{R}$ โดยที่ s และ σ เป็น mappings

$$s : X \longrightarrow \mathbb{R} \quad , \quad \sigma : \mathbb{R} \longrightarrow Y$$

เราเรียก X ว่า อินพุท (Input).

Y ว่า เอาท์พุท (Output)

s ว่า activation function

σ ว่า output map.

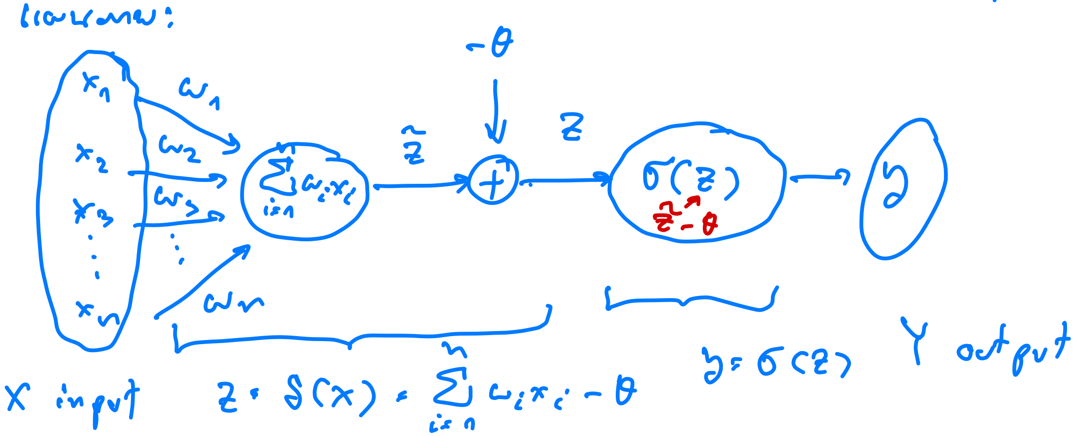
และเราสามารถเขียน formal neuron ในรูปของ transfer function นั่นคือ

$$f = \sigma \circ s : X \longrightarrow Y$$

โดยที่ $\underline{\underline{s(X)}} = s(x_1, x_2, \dots, x_n) = \sum_{i=1}^{n} w_i x_i - \theta$ เป็น activation fn

$[X = (x_1, x_2, \dots, x_n)^T]$

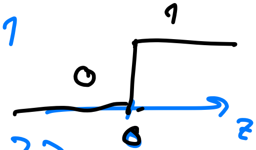
แผนภาพ:



โดยปกติแล้ว จะนิยามให้ σ อยู่ในรูป.

$\sigma : \mathbb{R} \to [0, 1]$ ที่ซึ่ง $\lim_{z \to -\infty} \sigma(z) = 0$ และ

$\lim_{z \to \infty} \sigma(z) = 1$

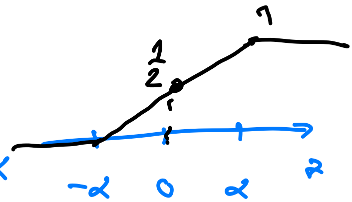


<u>Ex ของ $\sigma(z)$:</u>

1.) Heaviside function: $H(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$

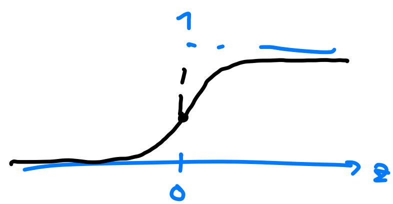
2.) ramp function: สำหรับ $\alpha > 0$

$$\sigma(z) = \begin{cases} 0, & z \leq -\alpha \\ \frac{1}{2}\left(\frac{z}{\alpha} + 1\right), & -\alpha < z < \alpha \\ 1, & \alpha < z \end{cases}$$



3.) Fermi function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



(หรืออยู่ในรูป) scaled form:

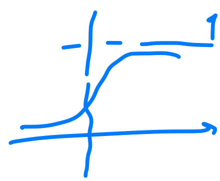
$$\sigma(z) = \frac{1}{1 + e^{-\alpha z}}$$

(ความชัน.)

ควบคุมความชันที่ (0, 1)

4.) modified hyperbolic tangent:

$$\sigma(z) = \frac{1}{2}\left(\frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} + 1\right)$$



Def: ให้ $g(x_1,\dots,x_n) = \sum_{i=1}^{n} w_i x_i - \theta$, $w_i, \theta \in \mathbb{R}$, $i=1,\dots,n$.

จะเรียก formal neuron (X, Y, σ, g) ว่า

σ-perceptron.

ถ้า σ เป็น Heaviside function จะเรียกว่า perceptron

หรือ McCulloch-Pitts neuron.

ตัวอย่าง: McCulloch-Pitts neuron ที่มี $X = \{0,1\}^n$ (Boolean set)

ที่มี $f: \{0,1\}^n \to \{0,1\}$ (switching function)

อยู่ในรูปของ

$$f(x) = f(x_1,\dots,x_n) = H\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$$

$(x_1, x_2, \dots, x_n)$ $\qquad = H(\langle w, x\rangle - \theta)$

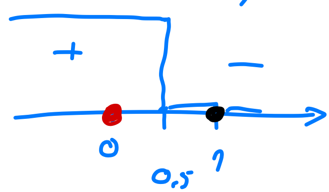
โดยกำหนด weights $w = (w_1, w_2, \dots, w_n)^T$, $w_i \in \mathbb{R}$

และ threshold $\theta \in \mathbb{R}$

Ex: 1.) f = Not :

x	$f(x)$
0	1
1	0

$H(-x+0.5)$

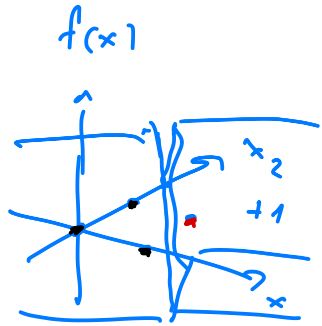


จากการเขียน f อยู่ในรูป

$f(x) = H(-x+0.5)$, $w_1 = -1$, $\theta = 0.5$.

2.) $f =$ AND:

x_1	x_2	$f(x)$
0	0	0
0	1	0
1	0	0
1	1	1

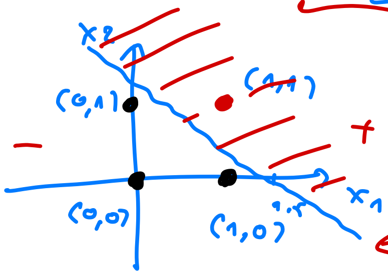


สามารถเขียน f ในรูป.

$$f(x) = H(\underbrace{x_1 + x_2 - 1.5})$$

$x_1 + x_2 - 1.5 = 0$

$\Rightarrow x_2 = -x_1 + 1.5$.

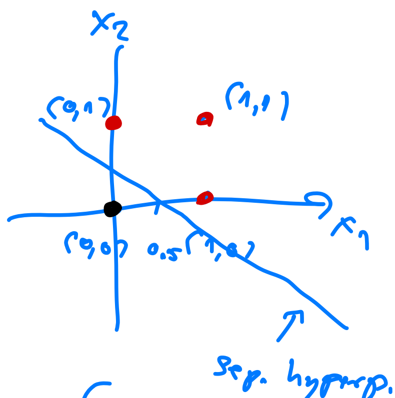


← Separating hyperplane. $(+x_1 + x_2 = 1.5)$

3.) $f =$ OR:

x_1	x_2	$f(x)$
0	0	0
0	1	1
1	0	1
1	1	1

$\Rightarrow f(x) = H(x_1 + x_2 - 0.5)$



Sep. hyperp. $(x_1 + x_2 = 0.5)$

ข้อสังเกต: โดยทั่วไปแล้ว Boolean func. สามารถเขียนได้ในรูปของตัวดำเนินการ AND, OR และ NOT

นั่นหมายความว่า เราสามารถสร้างเครือข่าย (Networks) ของ McCulloch-Pitt neuron เพื่อแทนรูปแบบของ Boolean fn ใดๆ ได้

แต่: ไม่เป็นความจริง ที่ ทุกๆ Boolean fn (switching fn) สามารถเขียน

ได้ในรูป single perceptron.

Ex: XOR (exclusive OR operator).

x_1	x_2	$f(x): x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

Lemma: ไม่มี perceptron ที่สามารถแทน XOR ฟังก์ชันได้.

พิสูจน์: สมมติให้ $(x_1, x_2) \in \{0,1\}^2$

แล้วจะมี $\omega_1, \omega_2, \theta \in \mathbb{R}$ ที่ทำให้

$$x_1 \oplus x_2 = H(\omega_1 x_1 + \omega_2 x_2 - \theta)$$

ดังนั้นสามารถ แบ่งได้เป็นกรณี:

① : $0 \oplus 0 = 0 \Rightarrow H(-\theta) = 0 \Rightarrow -\theta < 0$

② : $0 \oplus 1 = 1 \Rightarrow H(\omega_2 - \theta) = 1 \Rightarrow \omega_2 - \theta \geq 0$

③ : $1 \oplus 0 = 1 \Rightarrow H(\omega_1 - \theta) = 1 \Rightarrow \omega_1 - \theta \geq 0$

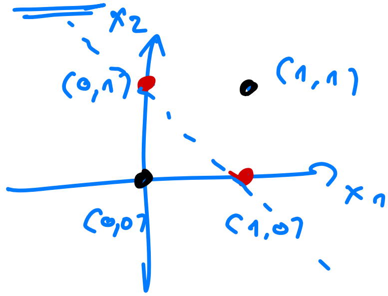
④ : $1 \oplus 1 = 0 \Rightarrow H(\omega_1 + \omega_2 - \theta) = 0 \Rightarrow \omega_1 + \omega_2 - \theta < 0$

จาก ①+④ : $\omega_1 + \omega_2 - 2\theta < 0$

②+③ : $\omega_1 + \omega_2 - 2\theta \geq 0$

↯

แผนภาพ:



ไม่มีเส้นตรงที่จะแบ่งกลุ่ม x_1, x_2 ในรูปแบบที่ต้องการได้!

=> แต่เราสามารถเขียน XOR ในรูป Networks ของ NOT, AND, OR ได้.

$$XOR(x_1,x_2) = OR(\underbrace{AND(NOT(x_1),x_2)}_{y_1}, \underbrace{AND(x_1,NOT(x_2))}_{y_2})$$

x_1	x_2	$f(x)$
0	0	0
0	1	1
1	0	1
1	1	0

=>

x_1	x_2	y_1	y_2	$f(x)$
0	0	0	0	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

=> Intro to Python. (กลุ่ม อย่างมาก 4 คน)
Labtop 1 เครื่อง.

-> ลง Anaconda python

$$\boxed{f(x): \{0,1,2\}^2 \to \{0,1\}}$$

การบ้าน!: ให้หาตัวอย่าง เซตข้อมูล ของ $(x_1, x_2) \in \{0,1,2\}^2$
ที่ 1.) สามารถแบ่งได้ด้วย perceptron, หา w_1, w_2, θ
และเขียนในรูป $f(x) = H(w_1x_1 + w_2x_2 - \theta)$
2.) ไม่สามารถแบ่งได้ด้วย perceptron.