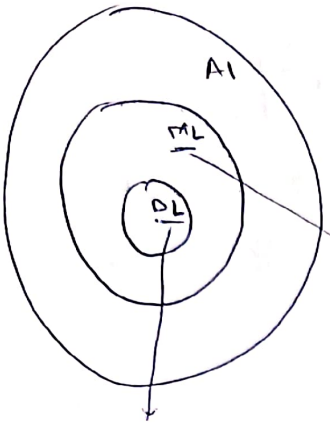
Deep Learning (ineuron)

Artificial Intelligence : Humans want to automate rudimentary tasks



To deal with search uncertainty, we need AI. When we are writing search codes, it may go endless and may result in a situation where this is not an optimal idea

Robot → Image (CNN) → picking pen

more useful for data available in structured format

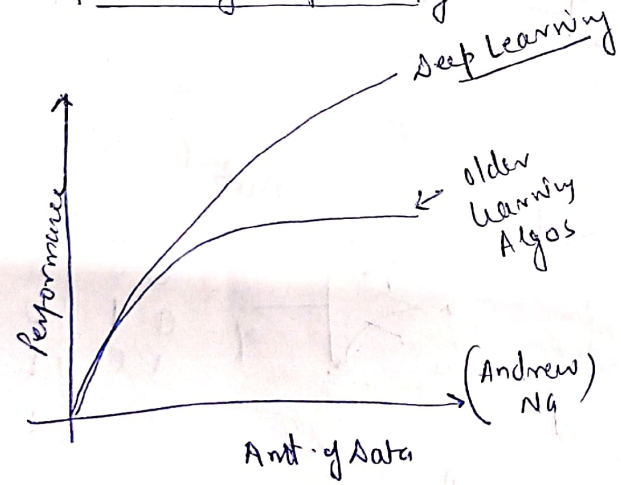
- live feed :
- audio file RNN
- unstructured data (text (NLP))

DL → ANN (Artificial Neural Network)

→ Supervised Learning (Labelled data)
  → Computer vision
    → Object classification
    → object detection
    → Object tracking
    → ~~Object~~ Image Segmentation
  → NLP
    → Text generation
    → Sentiment Analysis
    → Neural Machine Translation
    → Attention mechanisms

→ Unsupervised Learning (Unlabelled data)
  → Representation and generative learnings
    → autoencoders
      → field of AI in which we try to find out latent representation of data
    → GAN
      → special kind of auto-encoders.
  → Reinforcement learning

* Importance of Deep Learning



* NeuraLink * OpenBCI * NVIDIA blogs * NVIDIA broadcast app

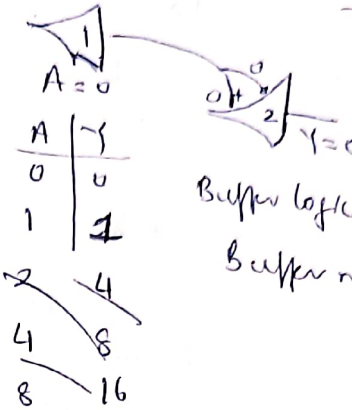
(2)

ANN vs BNN

- Artificial Neural Network
- Biological Neural Network

1st Artificial Neuron] Read this paper if you want
1943

threshold = 2



A	Y
0	0
1	1

Buffer logic gate or Buffer neuron

2 4
4 8
8 16

* Perceptron

→ invented by Frank Rosanblatt in 1957 :- paper

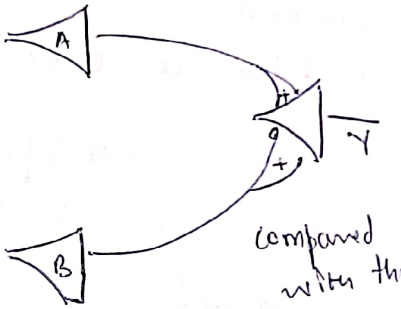
"The perceptron: A probabilistic model for information storage & organisation in the brain".

different from 1st artificial neuron in 1943.

1st neuron → input was binary

Perceptron architecture is also known as LTU or TLU

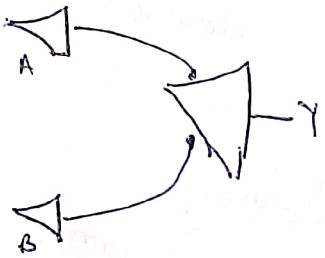
- LTU → Linear Threshold Unit
- TLU → Threshold Logic Unit



if sum > threshold = 1
else = 0

OR A∨B

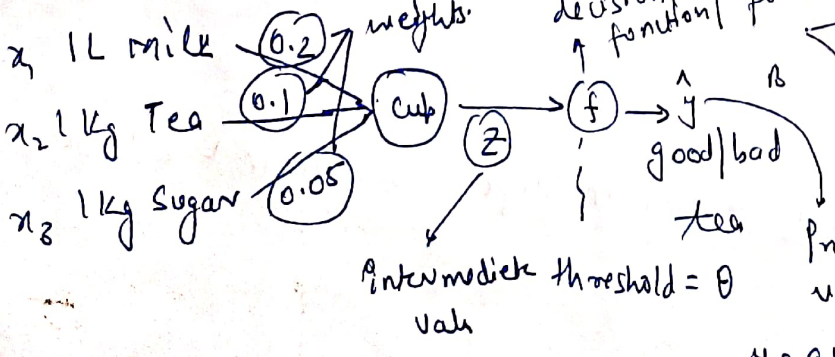
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

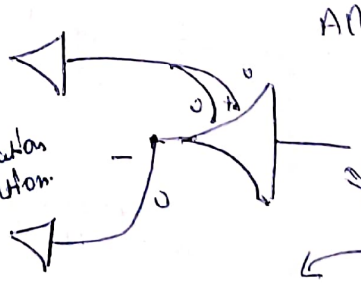


AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Task :- To make a good cup of Tea
↳ binary classification → good tea / bad tea



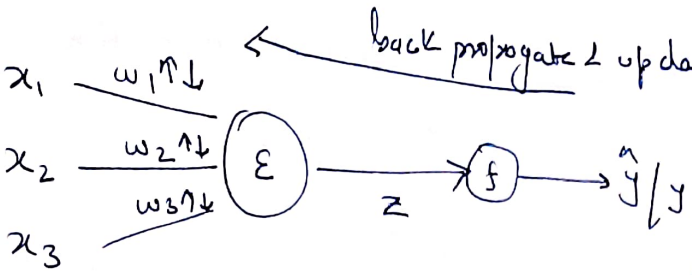


A = 1+1
B = -1

A∧B (A-B)

A	B	Y
0	0	0
0	1	0
1	0	1
1	1	1

A - B = 2 - (-1)
= 3 > threshold
= 1

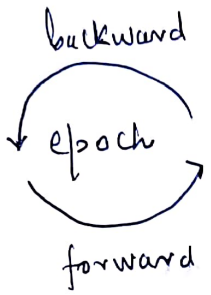


Keep doing these iterations and thats how we learn

$error = y - \hat{y}$

forward pass.

one iteration is called as an epoch.



$$z = w_1 x_1 + w_2 x_2 + w_3 x_3$$
$$\hat{y} = f(z)$$
$$error = y - \hat{y}$$

Perceptron weight update rule →

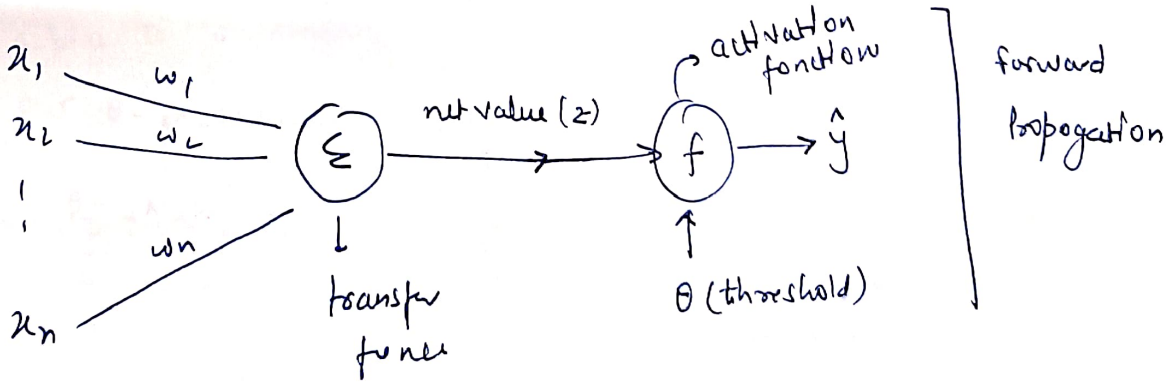
$$w_{new} = w_{old} + \eta \; error \; x_i$$

learning rate

Perceptron learning rule

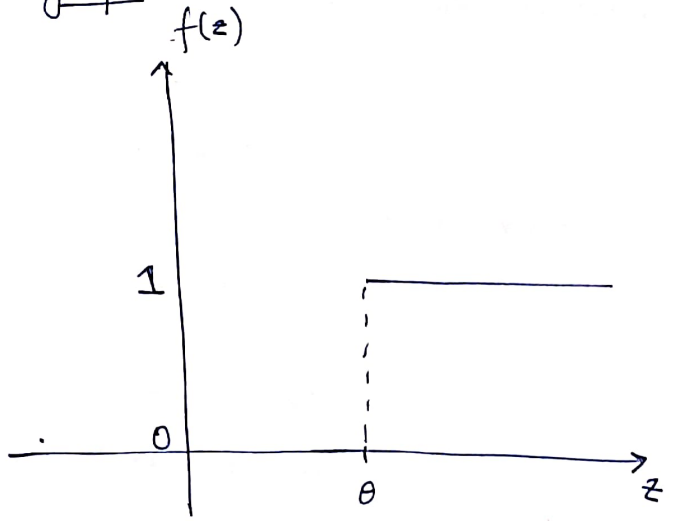
$$w_1 = w_1 + \eta (y - \hat{y}) x_1 \rightarrow \Delta w$$
$$w_2 = w_2 + \eta (y - \hat{y}) x_2$$
$$w_3 = w_3 + \eta (y - \hat{y}) x_3$$

$$f(z) = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$$ → learning function.



$$z = \sum_{i=1}^{n} w_i x_i \qquad \hat{y} = f(z) = \begin{cases} 0 & z < \theta \\ 1 & z \geq \theta \end{cases}$$

(4) graph



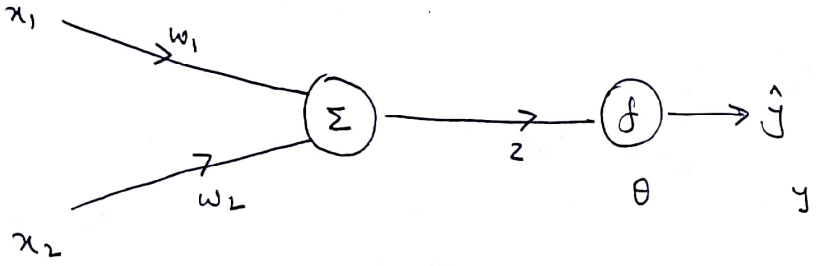
Back ward Propagation

error = $y - \hat{y}$

~~$w_{old} = w$~~

$$w_{new} = w_{old} + \eta (y - \hat{y}) x_{input}$$

* follow this cycle till error = 0.



$$z = w_1 x_1 + w_2 x_2 \quad -(1)$$

$$\hat{y} = f(z) = \begin{cases} 1 & z \geq \theta \\ 0 & z < \theta \end{cases}$$

$$= \begin{cases} 1 & w_1 x_1 + w_2 x_2 \geq \theta \\ 0 & w_1 x_1 + w_2 x_2 < \theta \end{cases}$$

$$= \begin{cases} 1 & w_1 x_1 + w_2 x_2 - \theta \geq 0 \\ 0 & w_1 x_1 + w_2 x_2 - \theta < 0 \end{cases}$$

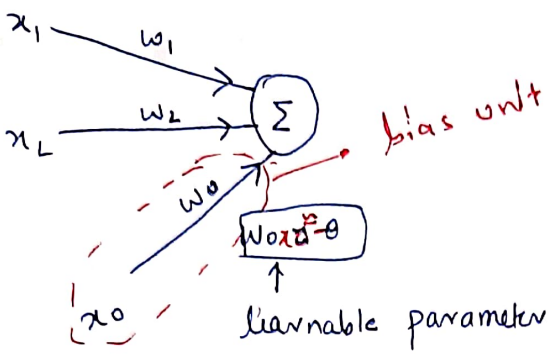
$w_1 x_1 + w_2 x_2 - \theta$

Let $w_0 = \theta$, $x_0 = -1$

$$w_0 x_0 = -\theta \quad -(2)$$

(5)

→ $w_1 x_1 + w_2 x_2 + w_0 x_0$ look like a third variable → lets modify our network now



learnable parameter not a manual parameter we need to set. It will learn on its own

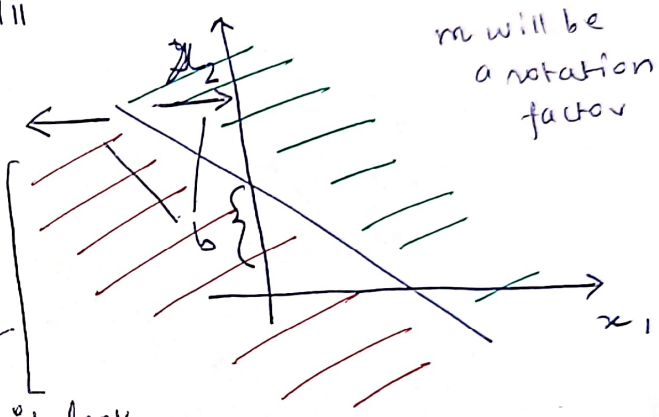
Assumption

$w_1 x_1 + w_2 x_2 - \theta = 0$

$x_2 = -\frac{w_1}{w_2} x_1 + \frac{\theta}{w_2}$

$m = \frac{w_1}{w_2}$, $b = \frac{\theta}{w_2}$

$x_2 = -m x_1 + b$



m will be a rotation factor

Now $b = \frac{\theta}{w_2}$ → it is helping in providing a shift and ensuring classification

doesn't it look like classification

$m = \frac{w_1}{w_2}$ → weights are responsible for rotation

b will be a shifting factor for the line aka translation factor

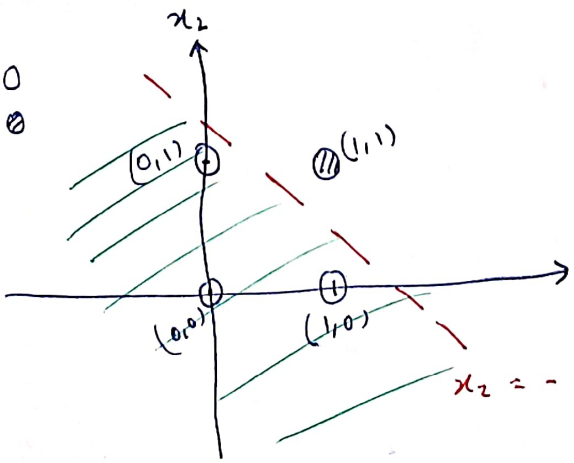
if $\theta = 0$ → $b \to 0$ $\quad x_1 = -m x_1$ line pass through origin and this may not be helpful as bias is helpful us to properly segregate classes.

* To make a model more generic we need to provide a bias to a model.

AND Data

	x_1	x_2	y
A	0	0	0
B	0	1	0
C	1	0	0
D	1	1	1

$0 = 0$
$1 = 0$



$b = \frac{\theta}{w_2}$

$m = \frac{w_1}{w_2}$

$x_2 = -m x_1 + b$