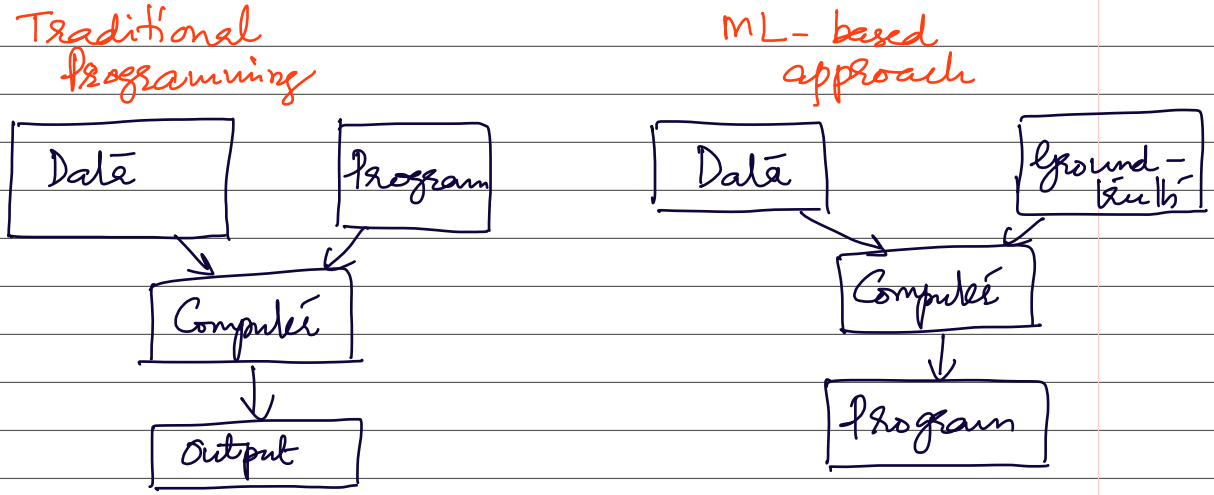
Mathematical Analysis of Artificial Neurons.

→ **What is Machine Learning?**

It is the field of study that gives computers the ability to learn from data without being explicitly programmed (Arthur Samuel, 1959). ML is the most popular AI technique.

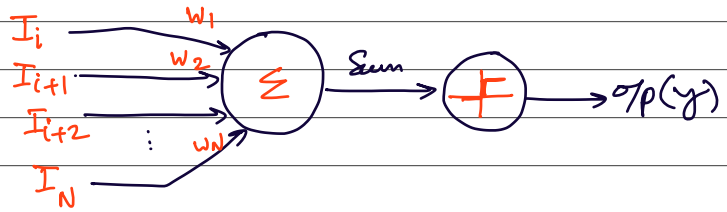


✱ A computer program is said to learn from experience E with respect to a task T and performance measure P, if the performance of the program in solving the task T, improves with experience E, as measured by P — Tom Mitchell

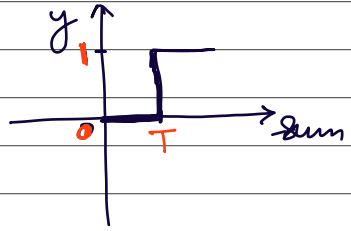
Artificial Neural Network (ANN)

→ The 1st ANN was conceptualized by McCulloch & Pitts in 1943.

→ MP model of artificial neuron is the 1st mathematical model of a biological neuron.



$$Sum = \sum_{i=1}^{N} I_i W_i$$



$$y = \begin{cases} 0, & \text{if } sum \leq T \\ 1, & \text{if } sum > T \end{cases}$$

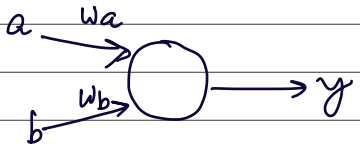
$y = f(sum)$, where f is called the "activation function."

MP model demo of logic AND function

→ All parameters need to be designed as there is no training method available.

→ 2 input AND gate

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1



→ Design

$0 \cdot W_a + 0 \cdot W_b \leq T$

$0 \cdot W_a + 1 \cdot W_b \leq T$

$1 \cdot W_a + 0 \cdot W_b \leq T$

$1 \cdot W_a + 1 \cdot W_b > T$

$\Rightarrow$

$T \geq 0$ —— ①

$W_a \leq T$ —— ②

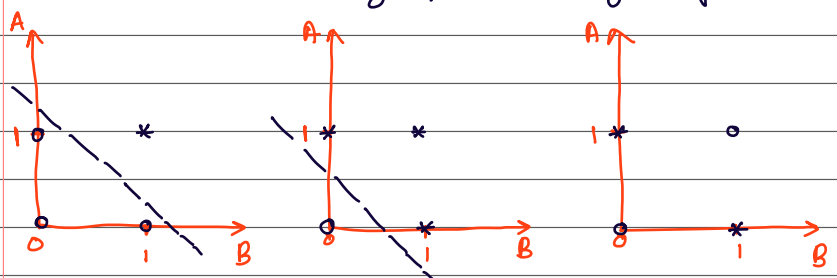
$W_b \leq T$ —— ③

$W_a + W_b > T$ —— ④

→ Solve these equations

→ $W_a = 1$, $W_b = 1$, $T = 1.5$

→ MP model works only for linearly separable data



→ Not possible to separate the data points.

→ Drawbacks of MP model

① No training/learning. So the model parameters need to be computed by solving a system of linear equations.

② Can't be used to process non-linearly separable data.

→ In 1958, Rosenblatt introduced perceptron model, which has a learning element in it.

$$\text{Sum} = \sum_{i=1}^{N} I_i W_i + b$$

→ bias, $b = -t$

→ Bias is a measure of how easy for a perceptron to reach 1.

Perceptron Learning Rule

$W_{i+1} = W_i + \Delta W$

$\Delta W = \eta \cdot \text{Error} \cdot x$ ($\eta \rightarrow$ learning Rate)

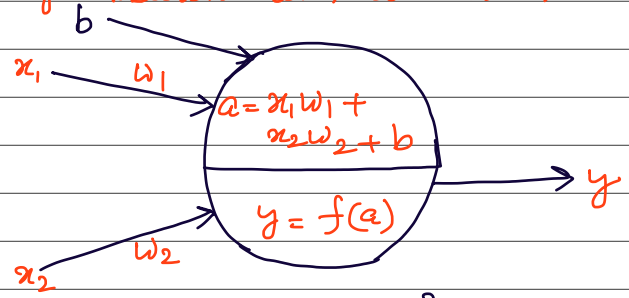
$\text{Error}, E = t - y$

$b_{i+1} = b_i + \eta \cdot \text{Error}$

This model also has issues in processing non-linear data.

→ In 1960, Widrow & Hoff proposed a neural model called Widrow & Hoff model. Gradient based learning is first introduced in this model.

Case Study ① Single Neuron with linear activation function.



Let $E = \frac{1}{2}(t-y)^2$ → mean square error

$y = a$ as f is a linear fn

→ Weight update using GD

$$w_1 = w_1 - \alpha \frac{\partial E}{\partial w_1} \quad \text{——— ①}$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w_1} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial a} \cdot \frac{\partial a}{\partial w_1} \quad \text{——— ②}$$

[Chain rule of partial derivatives]

Here $\frac{\partial E}{\partial y} = \frac{\partial\left[\frac{1}{2}(t-y)^2\right]}{\partial y} = \frac{1}{2}\frac{\partial\left[(t-y)^2\right]}{\partial y} = -(t-y)$ ——— ③

$$\frac{\partial y}{\partial a} = 1 \quad \text{——— ④}$$

$$\frac{\partial a}{\partial w_1} = \frac{\partial (x_1w_1 + x_2w_2 + b)}{\partial w_1} = x_1 \quad \text{——— ⑤}$$

Substitute eqns ③, ④ & ⑤ in ②

$$\frac{\partial E}{\partial y} = -(t-y) \cdot 1 \cdot x = \underline{\underline{-(t-y) \cdot x}} \quad \text{——— ⑥}$$

Substitute ⑥ in ①

$$w_1 = w_1 - \alpha \cdot -(t-y) \cdot x_1$$

$$w_2 = w_2 - \alpha \cdot -(t-y) \cdot x_2$$

$$b = b - \alpha \cdot -(t-y)$$

⇒

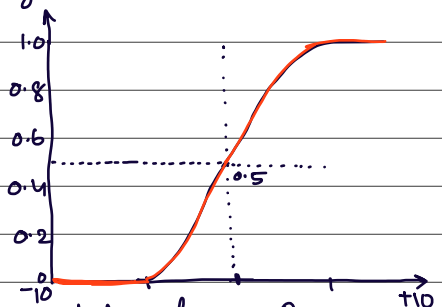
$$\boxed{\begin{aligned} w_1 &= w_1 + \alpha (t-y) \cdot x_1 \\ w_2 &= w_2 + \alpha (t-y) \cdot x_2 \\ b &= b + \alpha (t-y) \end{aligned}}$$

→ Linear neurons (neurons with linear activation fn) are not capable of capturing complex patterns in the data.

→ Solution is to use a non-linear activation fn.

Single neuron with a non-linear activation function

→ Sigmoid activation fn



→ Bounded fn :- Range is $[0,1]$

→ $f(x) = \frac{1}{1+e^{-x}}$

→ Non-linear, continuously differentiable

→ Can be used as non-linear decision boundaries.

→ Let's redo the weight update derivation for a single neuron; but with sigmoid as the activation fn.

$$E = \frac{1}{2}(t-y)^2$$

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

$$w_1 = w_1 - \alpha \frac{\partial E}{\partial w_1}$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y} \frac{\partial y}{\partial a} \frac{\partial a}{\partial w_1} \quad \text{——} ①$$

$$\frac{\partial E}{\partial y} = -(t-y) \quad \text{——} ②$$

$$\frac{\partial y}{\partial a} = \frac{\partial \left(\frac{1}{1+e^{-a}}\right)}{\partial a} = \sigma(a)(1-\sigma(a))$$

$$= y(1-y) \quad \text{——} ③$$

$$\frac{\partial a}{\partial w_1} = x_1 \quad \text{——} ④$$

Substitute ②, ③, ④ in ①

$$\frac{\partial E}{\partial w_1} = -(t-y)\, y(1-y) \cdot x_1$$

$$w_1 = w_1 + \alpha (t-y)\, y(1-y) \cdot x_1$$
$$w_2 = w_2 + \alpha (t-y)\, y(1-y) \cdot x_2$$
$$b = b + \alpha (t-y)\, y(1-y)$$

→ MSE is used as a loss fn for data that are generated by a normal distribution

→ Binary Cross Entropy is used as a loss fn for data generated from Bernoulli distrib^n

→ Single neuron with BCE as the loss fn

$$E = -[t \log(y) + (1-t)\log(1-y)]$$

$$y = \sigma(a)$$

Here both the activation and loss fns are non-linear fns.

$$w_1 = w_1 - \alpha \frac{\partial E}{\partial w_1}$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y} \frac{\partial y}{\partial a} \frac{\partial a}{\partial w_1} \quad \text{—— ①}$$

Here $\frac{\partial E}{\partial y} = \frac{\partial [-[t \log(y) + (1-t)\log(1-y)]]}{\partial y}$

$$= -\left[\frac{t}{y} - \frac{1-t}{1-y}\right] = -\left[\frac{(1-y)t - y(1-t)}{y(1-y)}\right]$$

$$= \frac{-[t - yt - y + yt]}{y(1-y)}$$

$$= \frac{-t+y}{y(1-y)} = \frac{(y-t)}{y(1-y)} \quad \text{—— ②}$$

$$\frac{\partial y}{\partial a} = y(1-y) \quad \text{—— ③}$$

$$\frac{\partial a}{\partial w_1} = \frac{\partial (x_1 w_1 + x_2 w_2 + b)}{\partial w_1} = x_1 \quad \text{—— ④}$$

Substitute ②, ③, ④ in ①

$$\therefore \frac{\partial E}{\partial w_1} = \frac{(y-t)}{y(1-y)} \cdot y(1-y) \cdot x_1 = (y-t) \cdot x_1$$

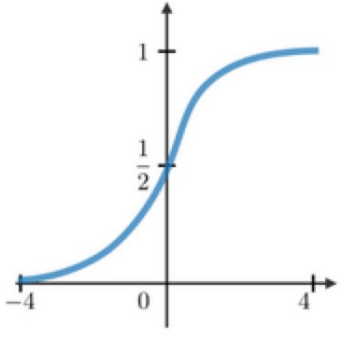
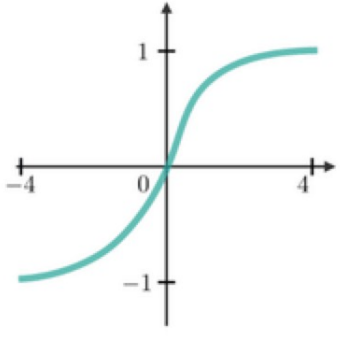
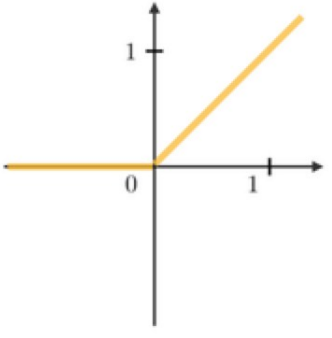
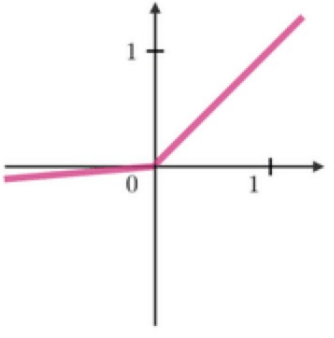
Hence

$$w_1 = w_1 + \alpha (y-t) x_1$$
$$w_2 = w_2 + \alpha (y-t) x_2$$
$$b = b + \alpha (y-t)$$

⇒ Thus Widrow Hoff model combined GD for updating neuron configurations with the help of non-linear activation and loss fns. Thus it became possible to represent complex non-linear mappings b/w i/p and o/p.

Activation fns

It is a mathematical function applied to the o/p of a neuron. The purpose is to introduce non-linearity to the n/w. Without non-linearity, the neural n/w will simply function as a linear regression model irrespective of the number of layers.

Sigmoid	Tanh	ReLU	Leaky ReLU
$g(z) = \frac{1}{1+e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$
			

→ Sigmoid, Tanh are bounded fns. Tanh is zero-centered.
→ ReLU not bounded.
→ ReLU is used in the hidden layers; whereas sigmoid is used at the last layer.
→ Learning ability
  Tanh > Sigmoid
  ReLU > Sigmoid (ReLU six times faster).