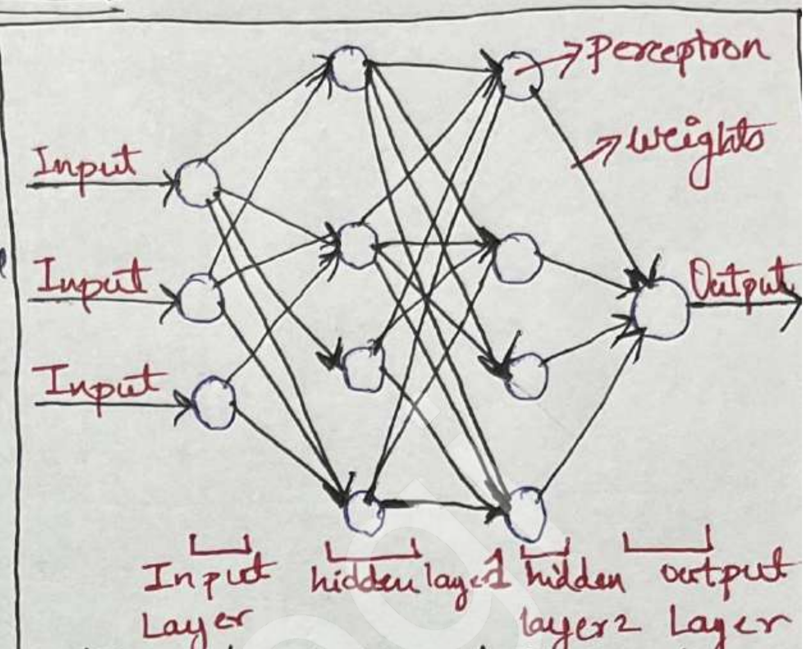
DEEP LEARNING

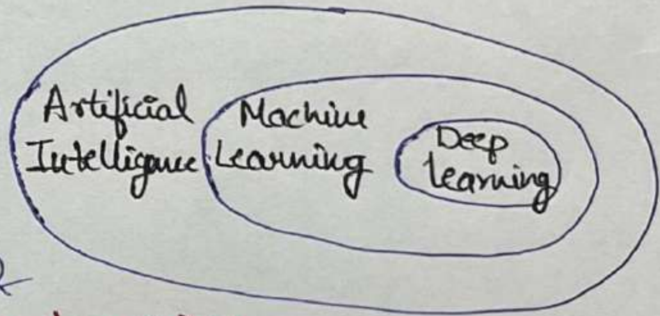


Example of Neural Network[ANN]

-> Deep Learning is a subfield of AI & ML.
-> It is inspired by the structure of human brain.
-> Uses the logical structure :- Neural Network = To analyse the data and patterns using multiple layers.
-> Examples of Neural Network : ANN, CNN, RNN, GAN.

-> Why is Deep Learning Popular?
1) Applicability - Can be applied to wide variety of real world problem
2) Performance - State of the art [Most advanced & best method model performance], even beats human experts.

-> Deep Learning is a subset of ML, that learns useful features/columns (representations) from raw data.



-> No manual feature Engineering.
-> Model learns what features matter & how to combine them - Representation Learning.
-> lower layer learns -> Simple features (edges, corner).
-> higher Layers learns -> Complex Abstraction (eg shapes, patterns).

-> Machine learning Vs Deep Learning

1) Data Dependency - DL models perform better with huge dataset
ML models perform better with smaller dataset.

2) Hardware Depndy - DL needs GPU to train complex matrix calculation
ML works on cheap hardware also.

3) Training Time – DL model needs more time for training
ML models training time is low.

4) Feature Selection – DL follows Representation Learning
In ML we manually extract features.

5) Interpratibility – DL models is not interpretable.
– ML models are easy to interpret & see.

CONCLUSION – DL cannot replace ML.
As at few places ML is better than Deep Learning.

Types of Neural Network

FNN	CNN	RNN	AE	GAN
Feed Forward Neural Network	Convolutional neural Network	Recurrent Neural Network	Auto Encoders	Generative Adversarial Network
• Data goes only forward • No memory • Used when input size fixed eg- MLP, CNN	• Image Expert • Learns shapes & edges → object • Special type of FNN	• Has Memory • Output depends on current + previous step • Types: LSTM, GRU	• Compress & Rebuild • learns what is important in data	• Generator creates fake data • Discriminator check if it's real or fake • Both compete & improve together

Applications of Deep learning

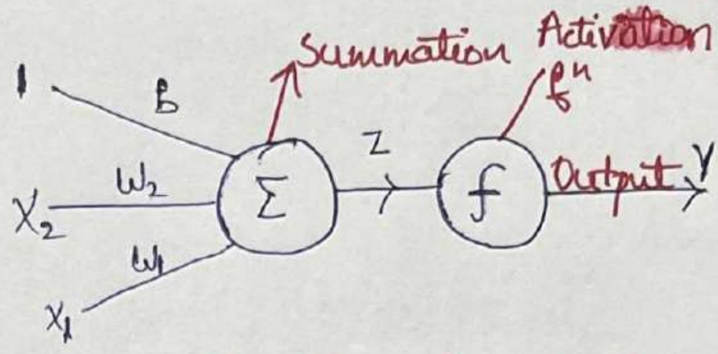
1) Self driving car
2) Game playing Agents.
3) Virtual Assistants
4) Image Colorization
5) Adding audio to mute videos
6) Image text generation
7) Text translation
8) Pixel Restoration

Deep Learning is used where data is large, complex, and unstructured like images, text, audio & video.

What is a Perceptron ?

→ It is an algorithm.
→ It is the simplest neural network can be viewed both as a algorithm or model.
→ (eg)

iq	cgpa	placement
78	7.8	1
69	5.1	0



$X_1, X_2, 1$ = Inputs f = step f^n or tanh etc
$Z = w_1 X_1 + w_2 X_2 + b.1$

PERCEPTRON

For each row/student in prediction:-
$X_1 \to IQ$, $X_2 \to CGPA$
. We will compute Z
During training the Goal is to find the value of $(X_1, X_2, X_3 \cdots, bias)$

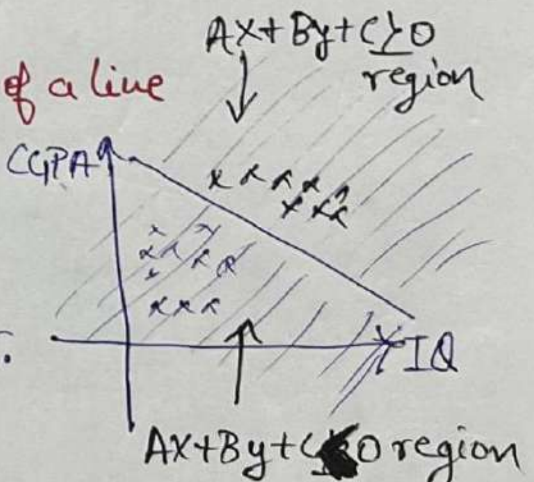
→ Neuron Vs Perceptron

- Perceptron is inspired by Neuron w.r.t inputs, outputs.
- Perceptron is the simplest version of idea of a brain cell in D.L.

→ INTUITION

- $X_1 = IQ = 78$, $X_2 = CGPA = 58$, Placement = ?
  Here the 'say' of $X_1 = IQ$ is more in comparision with $X_2 = CGPA$ as the value of IQ > CGPA. So the $\boxed{\text{weight = "say"}}$
- eg $Z = w_1 x_1 + w_2 x_2 + b \longrightarrow (Ax+By+C=0) \to Eq^n$ of a line

Step $f^n \Rightarrow Y$ (output) $= f(z) = \begin{cases} 1 & Z \geq 0 \to Ax+By+C \geq 0 \\ 0 & Z < 0 \to Ax+By+C < 0 \end{cases}$



- So, Perceptron is nothing but a binary classifier.
  It creates region & divide data in 2 region
  2-D = Line, 3-D = plane, 4-D = hyperplane.

Limitation = If perceptron can be only used on linear and sort of linear dataset.

Perceptron trick

{Perceptron is a binary classification algorithm}

→ How to train a perceptron?

- Perceptron trick is a simple way to learn a linear decision boundary that seperates 2 classes.

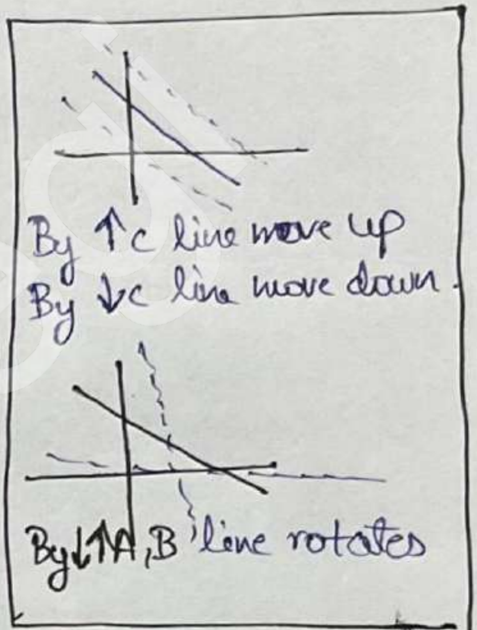
→ Core Idea - To find the weights and bias.

Suppose in 2D space:-

$AX + BY + C = 0$, A,B are the weights & C is the bias.

From perceptron diagram we can say

$x_1 = A$, $x_2 = B$, $b = C$



→ Steps -

Start with random values of A, B, C

↓

Loop for 1000 iterations

↓

With each iteration update A, B, C

↓

Repeat till max-iteration reached or no more misclassification.

→ Algorithm

```
for i in 1000
randomly select weight bias
Wn = Wo + η(Yi - Ŷi)Xi
```

→ When Pred = Actual [$\hat{Y}_i = Y_i$] then no change in weight $W_n = W_o$

→ When Pred ≠ Actual [$\hat{Y}_i \neq Y_i$] 2 cases.

Actual=1, Pred=0	Actual=0, Pred=1
Means actually '+' point but classified in '−' region	Means actually '−' point but classified in '+' region
$W_n = W_o + \eta X_i$	$W_n = W_o - \eta X_i$

We use a learning rate generally 0.01 or 0.1, otherwise the ~~two~~ line would move very quickly with each iteration.

→ Problem with Perceptron trick - It updates weight only on classification uses a hard step f^n. We cannot quantify how good the model is.

· To solve limitations of perceptron trick we use 'Loss functions'.

Perceptron Loss f^n

→ It is a mathematical f^n that measures how far the model's prediction is from the actuals.

→ We can quantify the model error & guides by telling how wrong a prediction is.

→ $$\boxed{\text{Loss } f^n = L(y_i, f(x_i)) = \max(0, -y_i f(x_i)) + \alpha \underbrace{R(w_1 w_2)}_{\text{Regularisation}}}$$

$f(x_i) = w_1 x_1 + w_2 x_2 + b$, Goal is to find best w_1, w_2, b using gradient descent

$w_1 = w_1 + \eta \frac{dL}{dw_1}$, $w_2 = w_2 + \eta \frac{dL}{dw_2}$, $b = b + \eta \frac{dL}{db}$. } for minimum Loss value.

→ Diff b/w Activation f^n & Loss f^n

★ Activation f^n –

- Introduces non-linearity – enable model to learn non-linear complex relationships (without it, a neural network is just a linear model)
- Applied Inside Neural Network both during training & inference.
- eg – RELU, Sigmoid, Tanh.

★ Loss f^n –

- Measures how wrong the prediction is compared to the true value.
- Used only during training to update weights
- eg MSE, Log-loss, Hinge loss.

Perceptron is flexible mathematical model & can use in diff ways:-

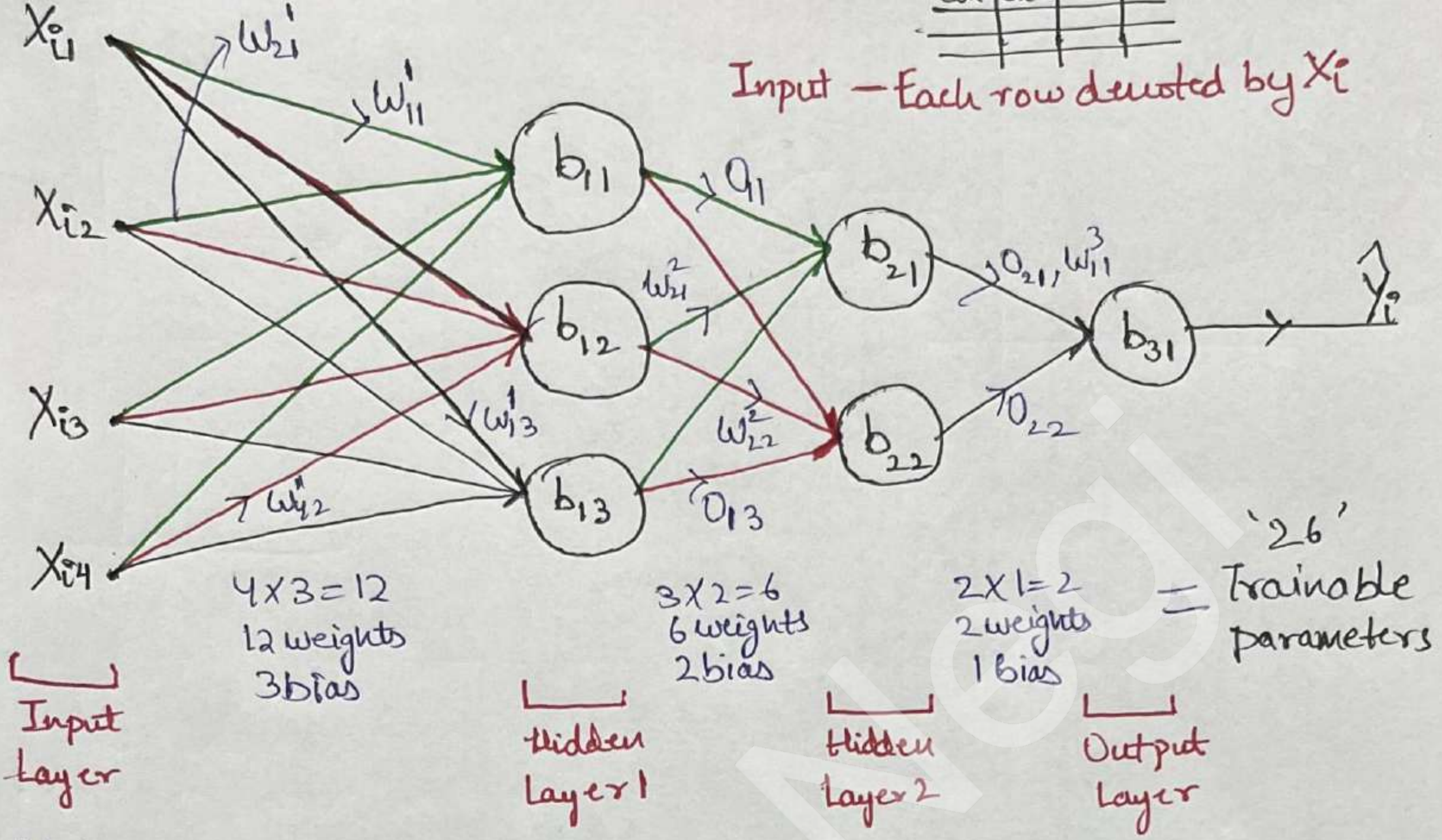
Loss f^n	Activation f^n	Output
Hinge Loss	Step f^n	Binary Classification (Perceptron) → (-1, 1)
Log-loss (Binary Cross entropy	Sigmoid	Logistic Regression Binary Classifier → 0-1
Categorical Cross-Entropy	Softmax	Softmax Regression Multiclass classification
M.S.E	Linear (No Activation f^n Used	Linear Regression Output - Numbers

MLP Notation

Data

Col1	col2	col3

Input – Each row denoted by X_i



Note

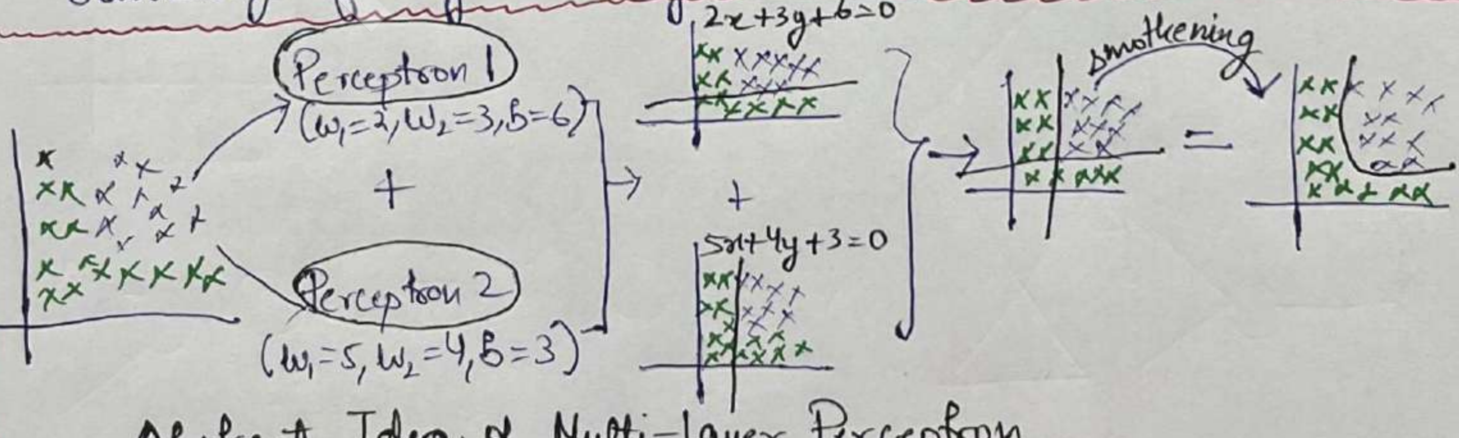
→ W_{ij}^k = K = weight entering which node, i = Current node in layer, j = target node in layer

→ O_{ij} = i = layer, j = Node No., follows same notation as its bias

Multi Layer Perceptron

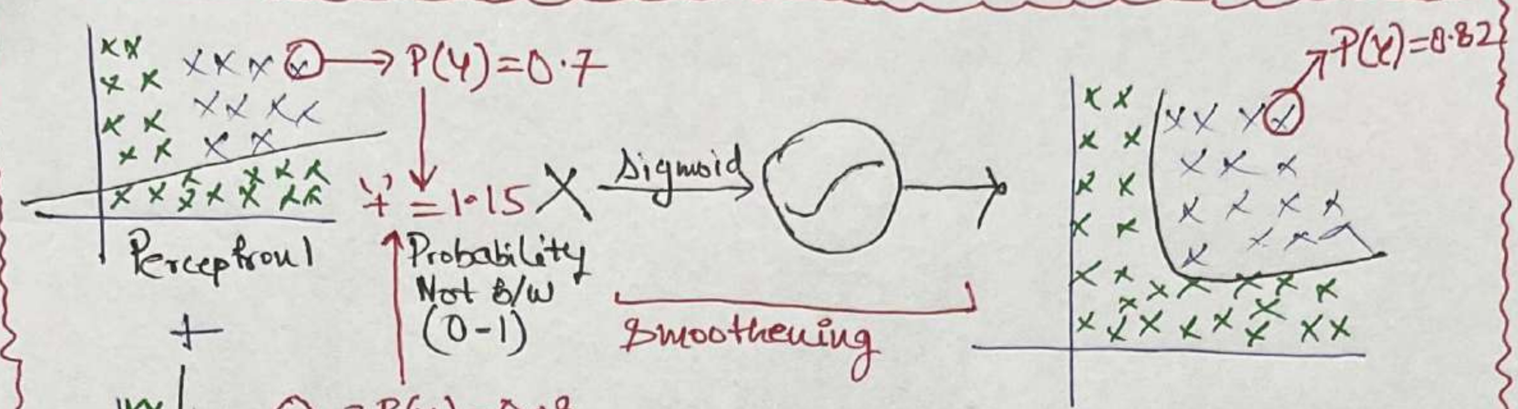
→ Problem with perceptron – can't make decision boundary for non-linear data.

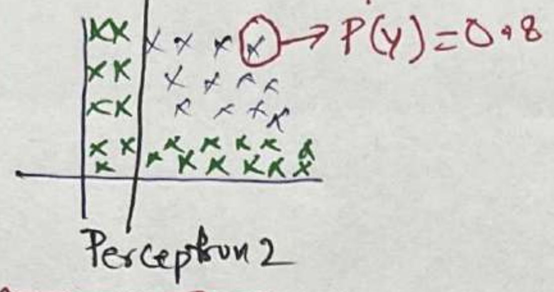
→ So we combine multiple perceptrons & it can create a decision boundary of any kind [any non-linear].



Abstract Idea of Multi-Layer Perceptron

→ MLP intuition w.r.t a data point :-

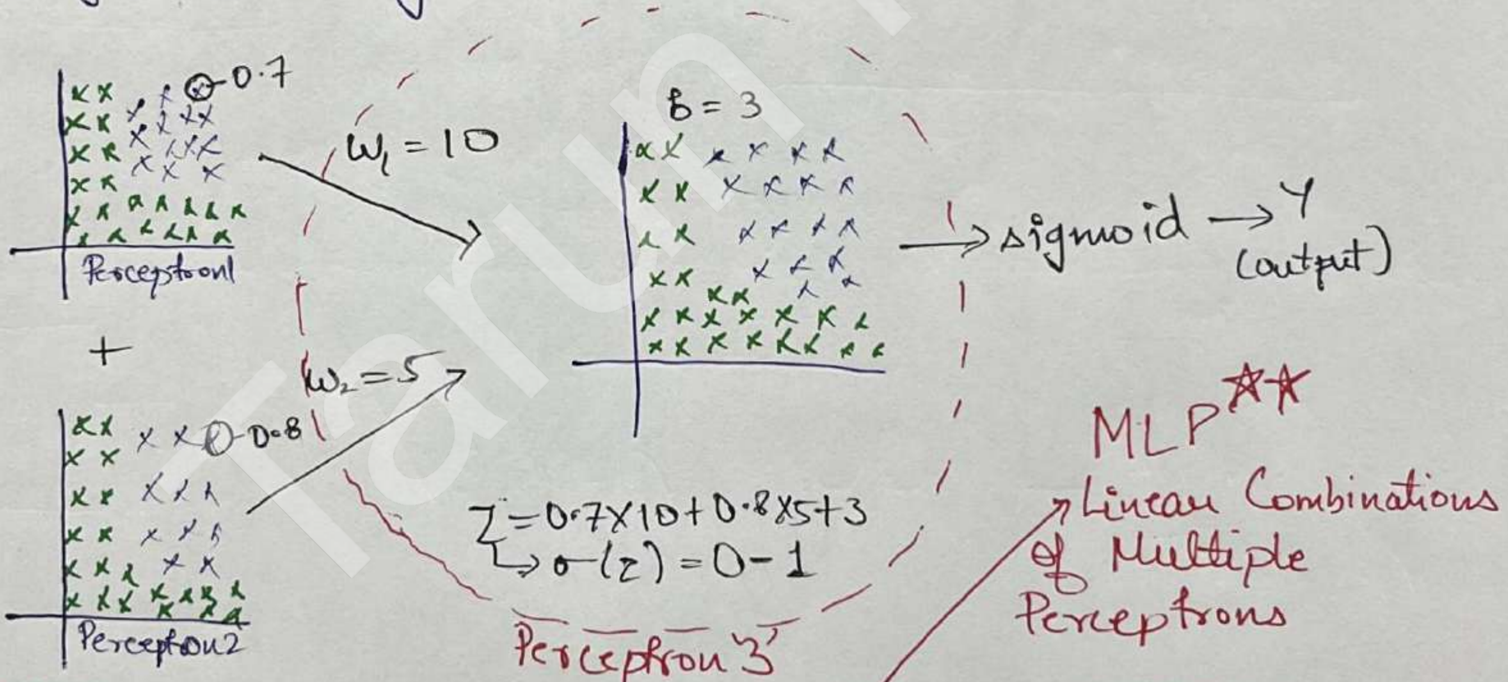


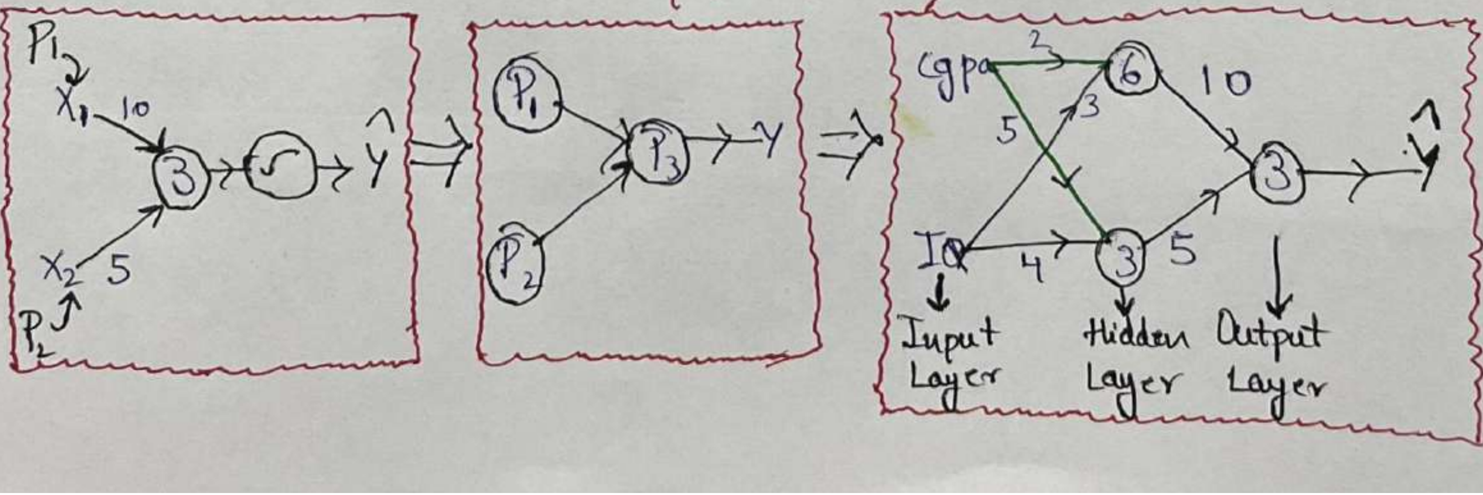


- This is done for each data point
- Both super-imposition (combining feature) & smoothening using Activation f^n is show here.

→ MLP intuition w.r.t weights & Bias:-

- One perceptron can have more effect than the other if it is assigned more weight over the other.





4 ways a perceptron can handle more Complexity

→ Add more nodes in the hidden Layer
(Model learns more patterns)

→ Add more hidden Layers
Learning happens in steps–
1st Layer learns simple things next layer combine that & learn

→ Add more input Nodes
Model gets more info & better understanding of Problem.

→ Add more output neurons (nodes)
Model gives many answ, eg multiclass classification.
eg Dog – 0.2, Cat – 0.6, human = 0.3, output = 0.6 (Cat) ✓.

→ Conclusion – Multi Layer Perceptron (MLP)
★★
Multi Layer Perceptron can solve any level of complexity by using enough input features, hidden neurons (nodes), hidden layer & output nodes & by learning patterns step by step.

IT'S NEVER TOO LATE

– TARUN NEGI

→ Upcoming Notes [Part-2]
How Neural Networks Learn?