Deep Learning.

AI ⇒ Applications that can do its own task without Any human intervention.

- Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions.

AI can be categorized into one of four types -

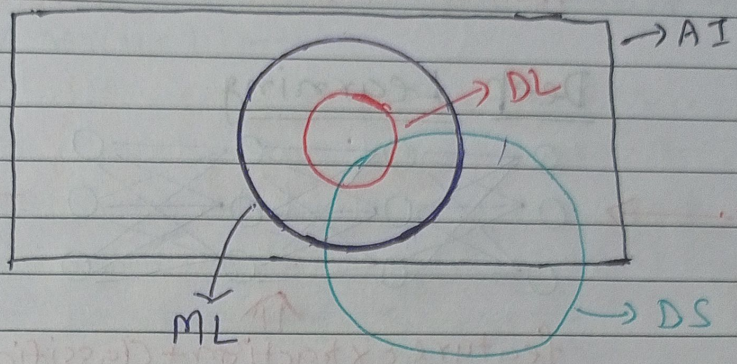
1. Reactive AI ⇒ It uses algorithms to optimize outputs based on a set of inputs. Chess playing AI's, for example, are reactive systems that optimize the best strategy to win the games.

   Reactive A.I. tends to be fairly static, unable to learn or adapt to novel situations. Thus, it will produce the same output given identical input.

2. Limited Memory AI - Can adapt past experience or update itself based on new observations or data. Often the amount of updating is limited (hence the name), and length of memory is relatively short. Autonomous vehicles, for example, can "read the road" and adapt to novel situations, even "learning" from past experience.

3. Theory-of-mind AI - Are fully adaptive and have an extensive ability to learn and retain past experiences. These types of AI includes advanced chat-bots that could pass the turing test, fooling a person and believing the AI was human being. While advanced and impressive, these AI are not self Aware.

4. Self-aware AI - As the name suggests, become sentient and aware of their own existence. Still in the realm of science fiction, some experts believe that an AI will never become concious or "alive".



ML - is a statstool to analyze the data, visualize the data, predictions, forecasting and clustering. "ML is a subset of AI".

Deep Learning => "Deep Learning is a particular kind of Machine Learning that achieves the great power and flexibility by learning to represent the world as nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones".

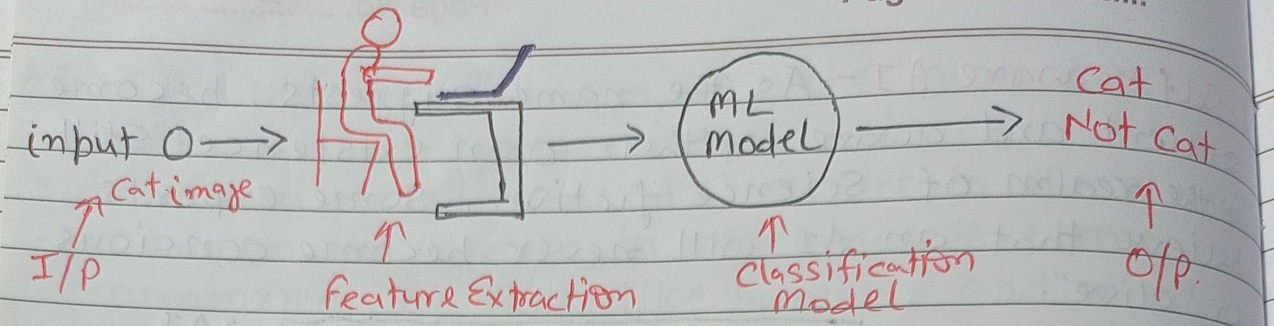
The main Aim Of DL is to mimic the Human Brain".

Machine Learning v/s Deep Learning

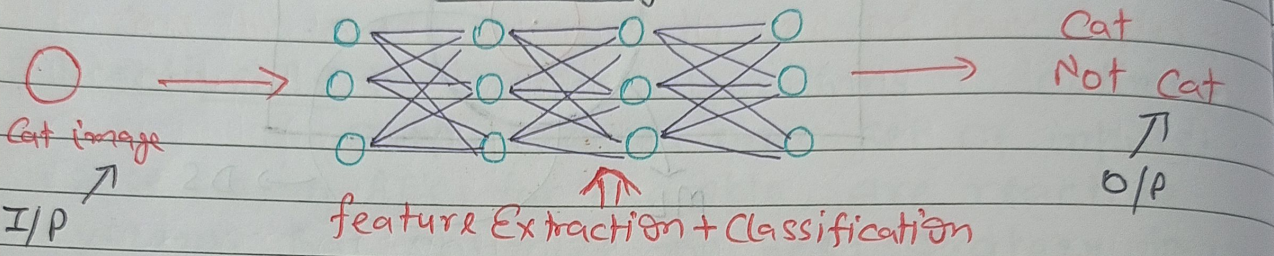
In case of Machine Learning, the data scientist needs to build the features and the more features data scientist can think of, the better is the model.

Whereas, in case of deep learning, the model itself learns the features from the training data.

Machine Learning



Deep Learning



You can see here Automatic feature extraction in Deep Learning.

⇒ For example, if we provide a deep learning model with images of various faces, it will extract several low level features like boundaries of eyes or nose or ears and then combine to build mid level features.

These mid Level features then combine to build several high level features which can relate to by seeing at these features.

Low Level features ⇒ Mid Level features ⇒ High level features

Reasons Behind popularity of Deep Learning ⇒

→ Data
→ Computational Resources
→ Applications
→ Research

Perceptron:- In the modern sense the perceptron, is an algorithm for learning a binary classifier called a threshold function: a function that maps its input x ~~(a real-world vector)~~ (a real-valued vector) to an output value f(x) (a single binary value):—

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0, \\ 0 & \text{otherwise} \end{cases}$$

$w \rightarrow$ vector of real valued weights

$w \cdot x \rightarrow$ is the dot product of $\sum_{i=1}^{m} w_i x_i$

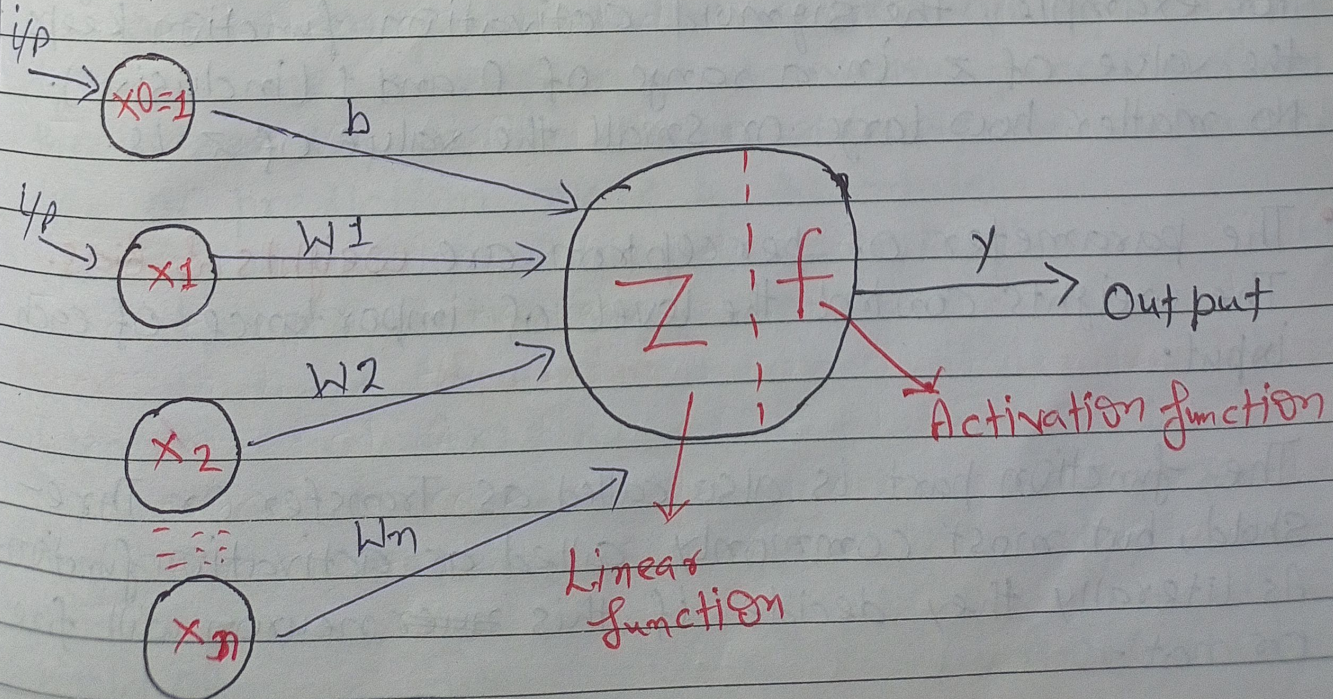
where $m \rightarrow$ is the no. of inputs to the perceptron.

$b \rightarrow$ is the bias.

The Bias shifts the decision boundary away from the ~~origion~~ Origin and does not depend on any input value.

A ~~neu~~ neuron outputs 1 (fires or activates) if the value of z exceeds the threshold value, 0. Otherwise it outputs 0.

-:- Structure Of a perceptron -:-



$$Z = (W1 X1 + W2 X2 + \ldots + Wn Xn) + b$$

$$\boxed{Y = f(Z)}$$

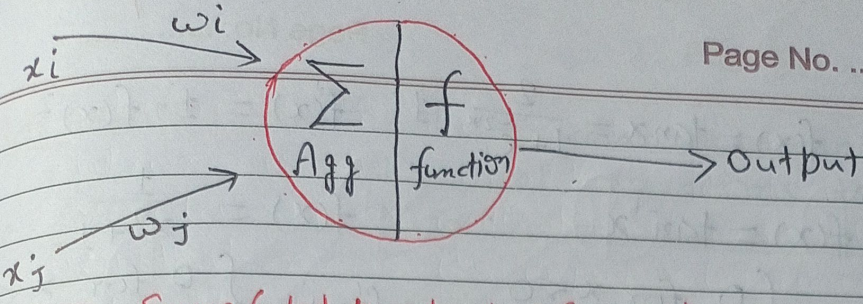
Perceptron Highlights

⇒ A perceptron is a mathmatical model in which linear function and non-linear activation function work together to calculate an output that can be an input for next perceptron or just a final O/P.

We use an activation function in the perceptron for the following purposes.

- It introduces non-linearity to the network. It enables neural networks to model non-linear relationships which are often found in real-world data.
- The type of activation function determines how a neuron should fire or activate.
- It keeps the o/p value of z in a certain range. For example, the sigmoid activation function keeps the value of z in a range of 0 and 1 (inclusive). No matter how large or small the value of z is.
- The parameter of perceptron are weights & Bias. The weights control the level of importance of each input.

The function part is also called as Transfer Or Threshold, but most commonly called as activation function As literally they decide if this ~~neuron~~ neuron will fire or not.



agg = Sum (dot product of x·w) + a bias

function = a_non-linear-function (agg)

⇒ Basically output of the function dictates if this neuron will "fire or not"—

1. The aggregate step or summation
2. The function part is also called as Transfer or threshold, but most commonly called as activation function — as literally they decide if this neuron will "fire or not".

-¦- Activation function -¦-

The activation function compares the input value to a threshold value. If the input value is greater than the threshold value, the neuron is activated.

It's disabled if the input value is less than the threshold value, which means its output isn't sent on to the next or hidden layer.

Note:- All hidden layers usually use the same activation function. However the o/p layer will typically use a different activation function from the hidden layers. The choice depends on the goal or type of prediction made by the model.

List Of Activation function.

Name	Equation	Derivative
Identity	$f(x) = x$	$f'(x) = 1$
Binary step	$f(x) = \begin{cases} 0 & \text{for } x<0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Sigmoid OR Logistic function	$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$

Tan H	$f(x) = \tanh x = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
Arc Tan	$f(x) = \tan^{-1} x$	$f'(x) = \frac{1}{x^2+1}$
Relu	$f(x) = \begin{cases} 0 & \text{for } x<0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x<0 \\ 1 & \text{for } x \geq 0 \end{cases}$
PReLU	$f(x) = \begin{cases} \alpha x & \text{for } x<0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x<0 \\ 1 & \text{for } x \geq 0 \end{cases}$
ELU	$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x<0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x)+\alpha & \text{for } x<0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus	$f(x) = \log_e(1+e^x)$	$f'(x) = \frac{1}{(1+e^{-x})}$
Soft Sign	$f(x) = \frac{x}{(1+\lvert x \rvert)}$	
Log of Sigmoid	$f(x) = \log\left(\frac{1}{1+e^{-x}}\right)$	
Swish	$f(x) = \frac{x}{(1+e^{-x})}$	
Sin c	$f(x) = \frac{\sin x}{x}$	
Leaky Relu	$f(x) = \max(0.1x, x)$	
Mish	$f(x) = x(\text{Tanh}(\text{Soft plus}(x)))$	

⇒ These activation function provide the non-linear part of story & they actually ensure that a neuron does not fall to linearity hence a whole network does not fall back to a linearity.

⇒ These also helps to keep the O/P of a neuron in certain value limit.

Hidden Layers ⇒ The Layer that performs the Mathimatical operations to generate the features are known as the hidden Layers.

Forward Propagation

Forward propagation refers to storage and calculation of input data which is fed in forward direction through the network to generate an output. Hidden Layers in neural network accepts the data from the input layer, process it on the basis of activation function and pass it to the output layer or the successive layers.

Errors in Neural Network

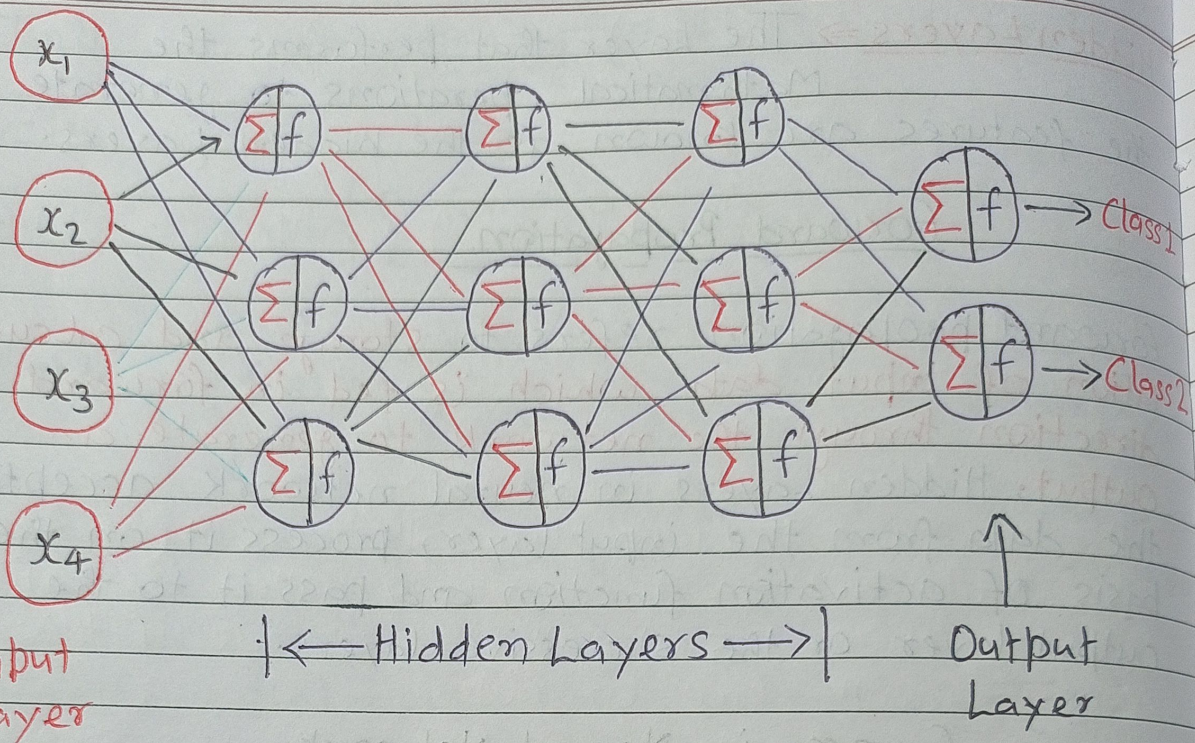
$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

There can be other error formula, but for example context 9am showing this one.

How to reduce Errors?

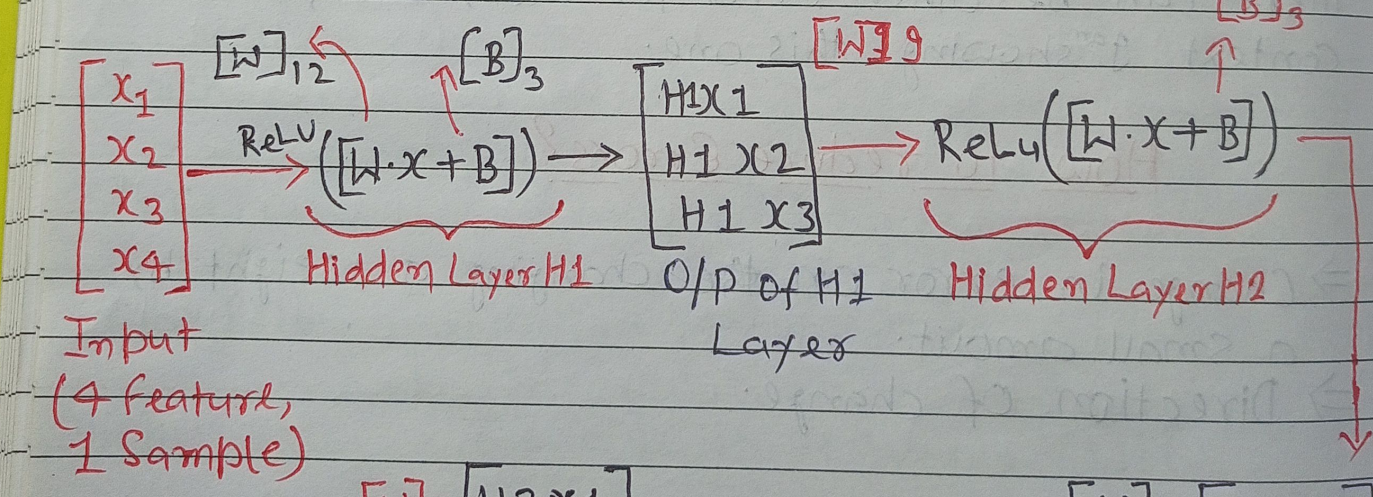
⇒ Change in error on ~~the~~ changing a weight by a small amout.
⇒ Direction of change.

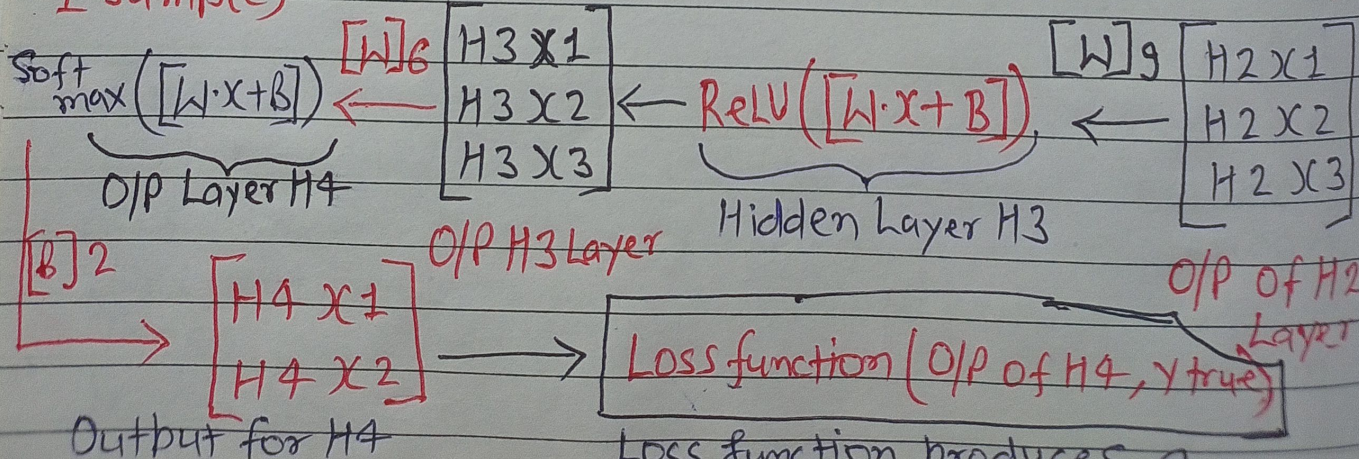
Forward Pass



$\sum agg$ = Sum(dot product of $x \cdot w$) + a bias

function = a non-linear function($\sum agg$)





*Neuron - is a things that holds Number. (0.2) or (0.5) or (0.9)
*The no. inside the neuron is known as activation.

Note => Don't forget that we have a bias per layer. which is also a learnable parameter.

So total we have - 36 weights (W) + 11 Biases (b)
= 47 W and b

Lets see, What will be the structure of this learning function is :-

$$\text{Y-pred} = f(x) = f(x, W, b) = \text{Sofmax}([W]_6 * \text{ReLU}([W]_5 * \text{ReLU}([W]_4 \cdot x + B)) + B) + B) + B)$$