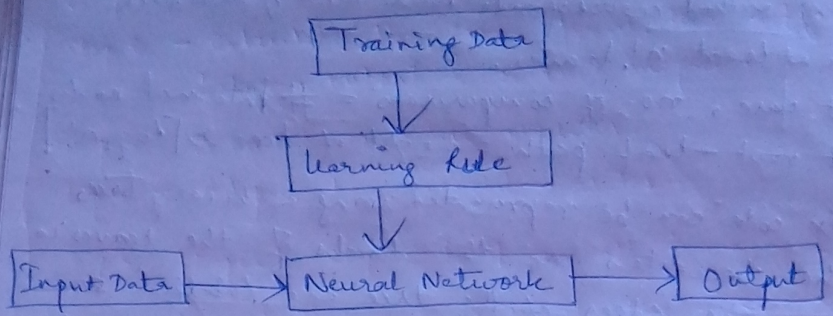
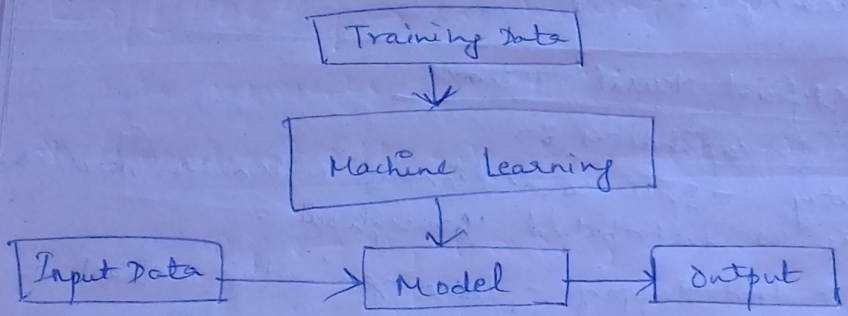
* How neural network and machine learning are related :—



Neural Network



Machine learning

We can implement the model of machine learning in various ways. One of them is neural network.

→ The information of the neural network is stored in the form of weights & biases.

→ The input signals are multiplied by the weights before entering the node. The output of the node is the weighted sum.

$$v = (w_1 x_1) + (w_2 x_2) + (w_3 x_3) + b$$

Higher the weights greater the effect

→ output of a node of NN is :—

$$y = \phi(v) = \phi(wx + b)$$

ϕ = activation function

→ No calculation is needed for input nodes.
→ For hidden nodes, we have to calculate the weighted sum & also we have to pass the signal through the activation function.

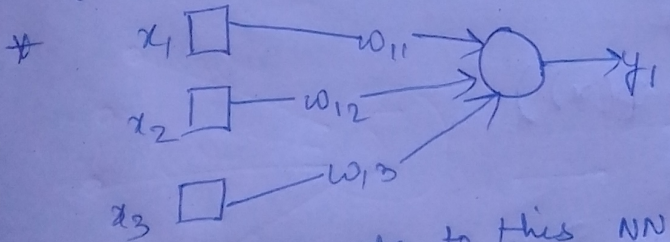
* NN Training — Supervised learning:-
- In supervised learning we first initialize the weights with adequate values.
- Then we take input from training data.

Step 1:- Initialize the weights
Step 2:- Calculate the error
Step 3:- Adjust the weight to reduce the error.

The systematic way of modifying the weights is called 'Learning Rule'.
The representative learning rule of single layer neural network is 'Delta Rule'.
Difference between the output & correct output of a NN is the error. Based on the error, we adjust the weights of the network.

$$w_{ij} \leftarrow w_{ij} + \alpha e_i x_j$$

- x_j → input/output value of input layer (node j)
- e_i → error of node 'i'
- w_{ij} → previous weight
- α → learning rate (0,1)
- w_{ij} → updated weight



$e_1 = d_1 - y_1$

Applying delta rule to this NN gives:-

$w_{11} \leftarrow w_{11} + \alpha e_1 x_1$
$w_{12} \leftarrow w_{12} + \alpha e_1 x_2$
$w_{13} \leftarrow w_{13} + \alpha e_1 x_3$

* The training process:-

Step 1:- Initialize the weights.

Step 2:- Calculate the error from the difference between output & correct output.

Step 3:- Calculate weights updates.

Step 4:- Adjust the weight updates.

Step 5:- Repeat step 2 to 4 for all training data.

Step 6:- Repeat step 2 to 5 until the error reaches an acceptable level.

→ EPOCH → One training iteration for all data represents one epoch.

* Generalized delta rule for weight update:-

$$w_{ij} \leftarrow w_{ij} + \alpha \delta_i x_j$$

$$\delta_i \leftarrow \phi'(v_i) e_i$$

$$\phi' = \frac{d}{dx}(\text{Activation function})$$

for linear activation function, $\phi'(v_i) = 1$

hence, $w_{ij} \leftarrow w_{ij} + \alpha e_i x_j$

eg: for sigmoid (nonlinear) activation function:-

$$\phi(x) = \frac{1}{1+e^{-x}}$$

$$\phi'(x) = \phi(x)(1-\phi(x))$$

$$\delta_i = \phi'(v_i) e_i$$

~~$\Rightarrow \delta_i = \phi(x)(1-\phi(x)) e_i$~~

$$\Rightarrow \delta_i = \phi(v_i)(1-\phi(v_i)) e_i$$

* Ways to update the weights :-

(1) Stochastic Gradient Descent → Errors are calculated for each training data & weights are immediately updated.

(2) Batch Gradient Descent → Updation is done for all the training examples once, i.e., errors are calculated for all training egs then updation is performed.

(3) Mini-batch gradient descent → Selects certain set of training egs then performs weight updation on that followed by weight updation on the next set. The training data is divided into certain sets or groups.