Deep Learning

* Deep learning is a technique which basically memic human brain
* machine learning can work and learn in same like human learn

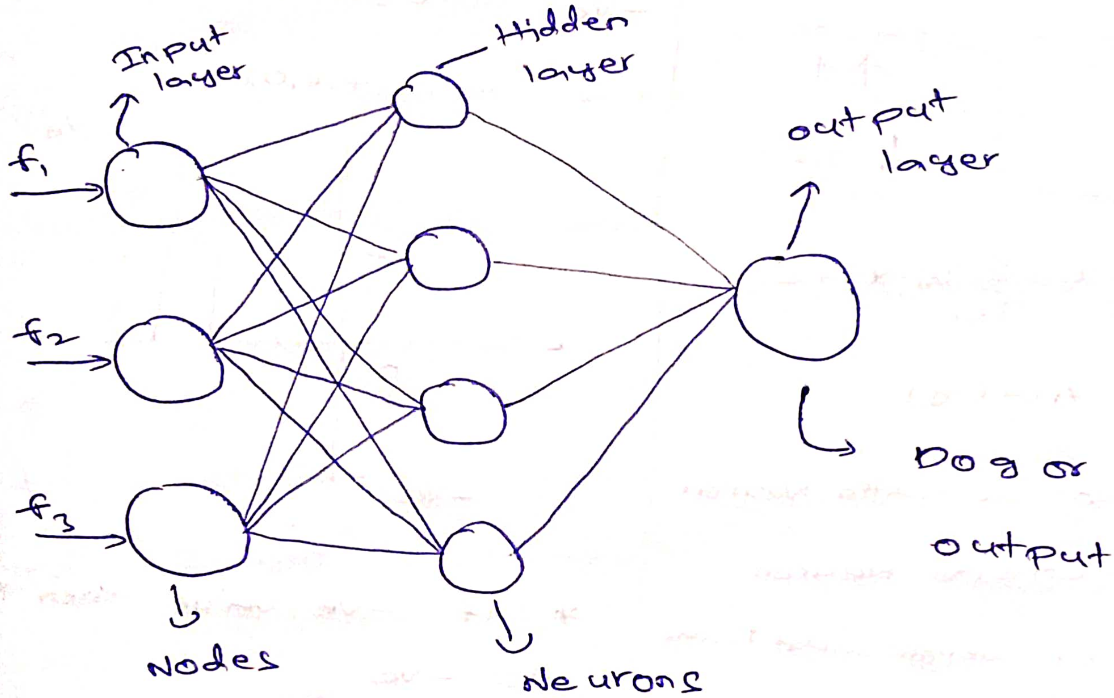
Example:

① ANN
② CNN
③ RNN

Input — Information

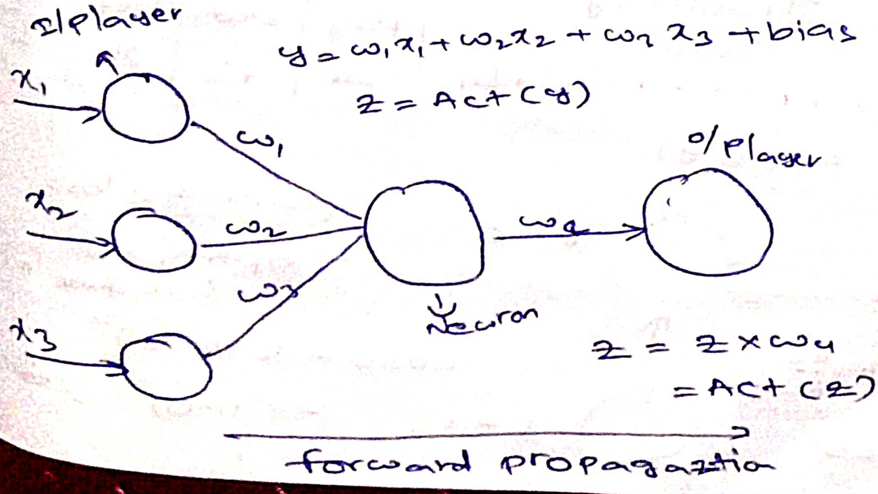
Dog
* big size
* voice different
* eye diff

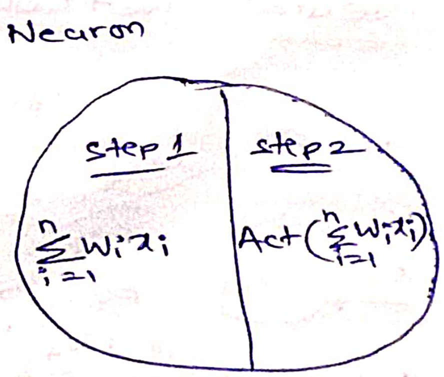
Cat
* small size
* diff eye
* voice diff



basic Neural Networks

How Neural Network works





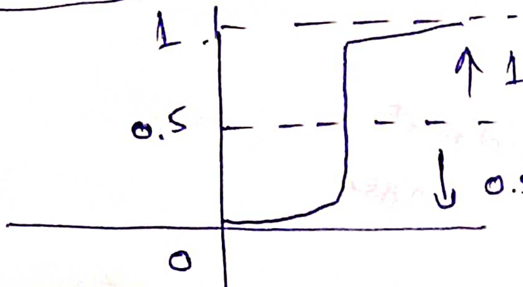
sigmoid

$\rightarrow \frac{1}{1+e^{-y}} \Rightarrow 0 \text{ to } 1$

Activation Function :

① SIGMOID AF

$$\frac{1}{1+e^{-y}}$$



$$y = \sum_{i=1}^{n} W_i X_i + b_i$$

$Act(y)$

0 to 0.5 Not activated Neuron

0.5 to 1 activated Neuron

* ① Summation of weight, input & bias

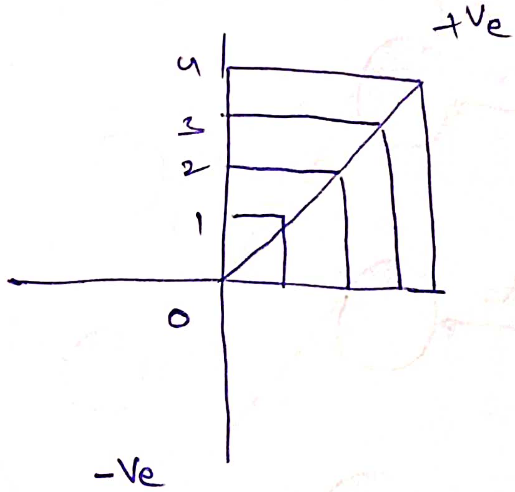
② Activation function

② RELU AF

$max(y, 0)$

$-Ve$ $max(-ve, 0)$

$+Ve$ $max(+ve, 0)$



* if −ve value then pass to −ve

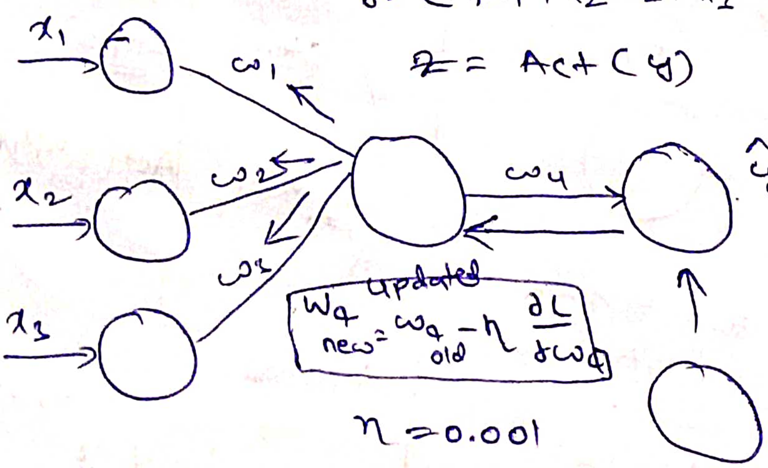
* if +ve value then pass to +ve

Neural Network Training

play	study	sleep	o/p
2h	4h	8h	1

$y = [x_1 w_1 + x_2 w_2 + x_3 w_3] + b_1$

$z = Act(y)$



updated

$$\boxed{W_{4\,new} = W_{4\,old} - \eta \frac{\partial L}{\partial W_4}}$$

$\eta = 0.001$

$\hat{y} = 0$

$y = 1$

minimize loss

↳ optimizer

↳ reduce loss

$Loss = (y - \hat{y})^2$

$= (1-0)^2$ ↳ reduced

* Forward propagation :-

* we pass input to input layer after input layer to Hiden layer Pass that time some weights and bias are added and activation function also add after that we pass to output layer that time weight, bias & Action functo added af ter that predicted value is show 0 or 1 on binary class classification

$$loss = (\overset{\text{Actual}}{y} - \overset{\text{predicted}}{\hat{y}})^2$$

$$= (1-0)^2$$

$$loss = 1$$

↓ reduc loss by using optim izers

↓ minimize the loss

* backward propagation :-

* reverse process
* used to reduce loss
* Actual to predicted
* predicted to Hiden layer (neuron) in that time weight is updated

learning rate

derivative of loss

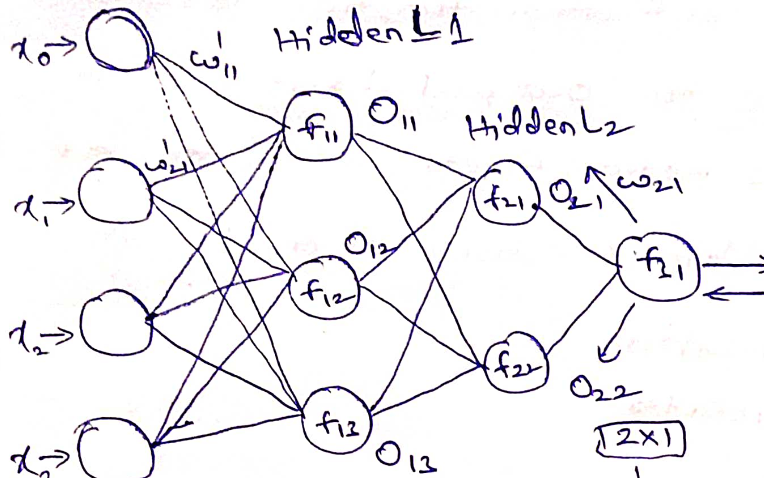
$$\boxed{w_{4\,new} = w_{4\,old} - \eta \frac{\partial L}{\partial w_4}}$$

derivative of w_4

same weight updated on w_3 w_2, w_1 also

$$\boxed{w_{3\,new} = w_{3\,old} - \eta \frac{\partial L}{\partial w_3}}$$

Multilayer Neural Network :

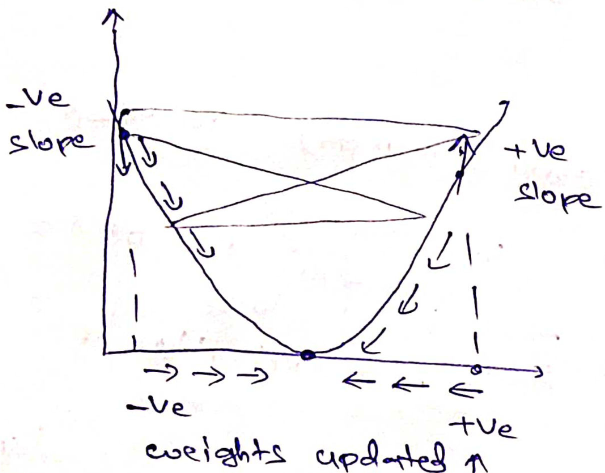


loss $= (y-\hat{y})^2$

decrese loss

* multilayer neural network

Gradient Desent :



$$W_{new} = w_{old} - \eta \frac{dL}{dw_{21}}$$

$\eta = 0.001$

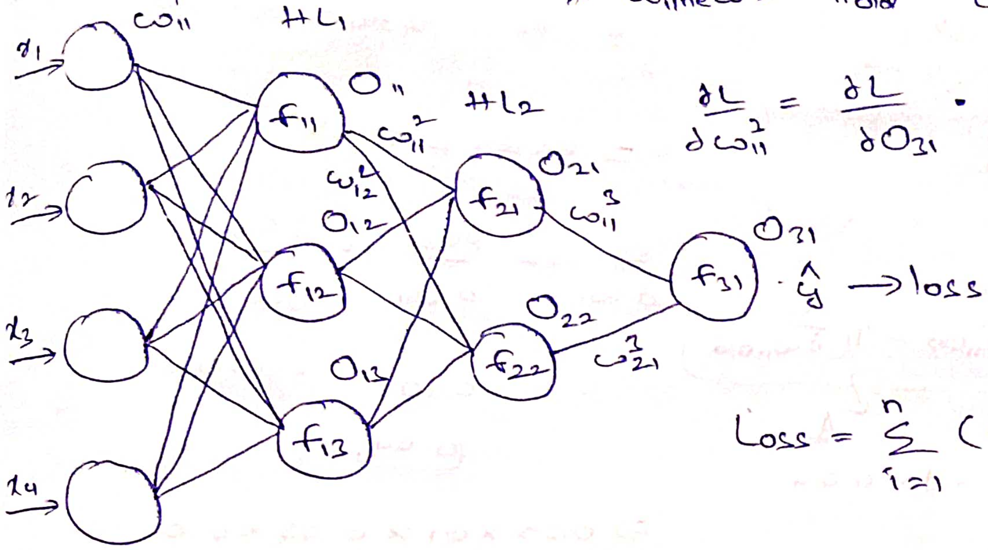
$\eta \rightarrow$ learning rate

* $\eta \rightarrow$ learning rate is used to control the Gradient decent value as lower not higher

* $\eta \rightarrow$ seleted by using hyper parameter optimization

Chain Rule in Backpropagation

* $w^3_{11new} = w^3_{11old} - \eta \boxed{\frac{\partial L}{\partial w^3_{11}}}$



$$\frac{\partial L}{\partial w^3_{11}} = \frac{\partial L}{\partial O_{31}} \cdot \frac{\partial O_{31}}{\partial w^3_{11}}$$

$$Loss = \sum_{i=1}^{n} (y-\hat{y})^2$$

* $w^3_{21new} = w^3_{21old} - \eta \boxed{\frac{\partial L}{\partial w^3_{21}}}$

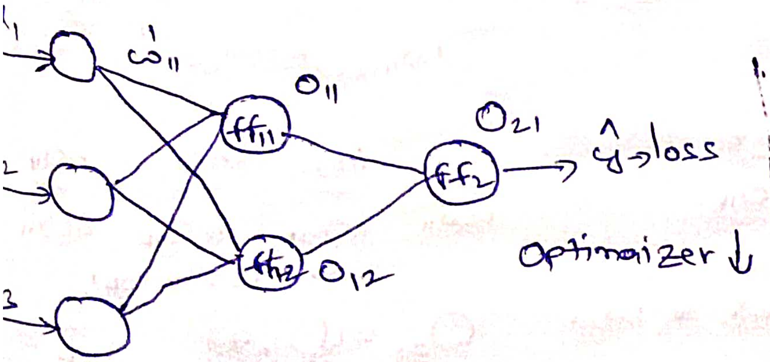
$$\frac{\partial L}{\partial w^3_{21}} = \frac{\partial L}{\partial O_{21}} \times \frac{\partial O_{31}}{\partial w^3_{21}}$$

* derivative's updated
* weight is updated
* loss reduced

* $w^2_{11new} = w^2_{11old} - \eta \boxed{\frac{\partial L}{\partial w^2_{11}}}$

$$\frac{\partial L}{\partial w^2_{11}} = \left[\frac{\partial L}{\partial O_{31}} \times \frac{\partial O_{31}}{\partial O_{21}} \times \frac{\partial O_{21}}{\partial w^2_{11}}\right] + \left[\frac{\partial L}{\partial O_{31}} \times \frac{\partial O_{31}}{\partial O_{22}} \times \frac{\partial O_{22}}{\partial w^2_{12}}\right]$$

Vanishing Gradient Problem:



weight updation

$$w^1_{11new} = w^1_{11}old - \eta \frac{\partial L}{\partial w^1_{11}old}$$

$$\frac{\partial L}{\partial w^1_{11}old} = \frac{\partial O_{21}}{\partial O_{11}} \cdot \frac{\partial O_{11}}{\partial w^1_{11}}$$

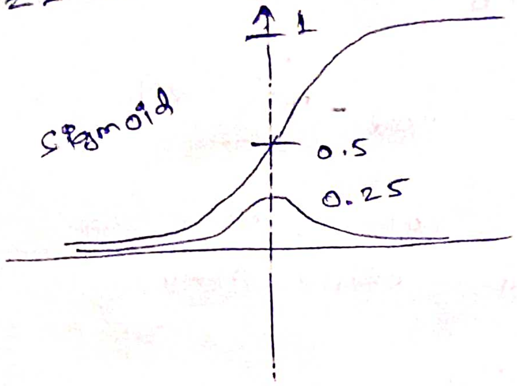
$\boxed{0.20} \times \boxed{0.02}$

↳ chain rule

$\boxed{0 \text{ to } 0.25}$

↑ sigmoid derivative

$z = \sum x w + b$



$\frac{1}{1+e^{-z}}$

$\frac{\partial \sigma(z)}{\partial z} = 0 \text{ to } 0.25$

$0 \le \sigma(z) \le 0.25$

$\frac{\partial L}{\partial w'_{11}} = \frac{\partial O_{21}}{\partial O_{11}} \cdot \frac{\partial O_{11}}{\partial w'_{11}}$

0.20×0.02

$\boxed{0.04}$

$$\boxed{w_{11new} = w'_{11old} - \eta \frac{\partial L}{\partial w'_{11old}}}$$

2.5

1

$\Rightarrow 2.5 - 1 \times 0.04$

$\Rightarrow 2.46$

$\Rightarrow 0.25 \times 0.1 \times 0.05 \times 0.01$

$\Rightarrow 10^{-4}$

$w_{11new} \approx w_{11old}$

$= 2.4999$

* vanishing gradient is very small

* layer increse number is decrese

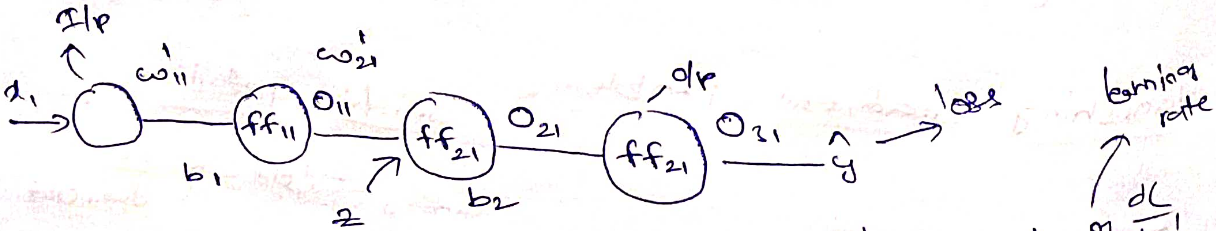
* no.of layers increses then the number is very small

*

Exploding Gradient Problem:

↳ happen because of weight because high value weight

↳ It will never come to global minima



$w'_{11new} = w'_{11old} - \eta \frac{\partial L}{\partial w'_{11}}$

$\frac{\partial O_{21}}{\partial O_{11}} = \frac{\partial \sigma(z)}{\partial z} \times \frac{\partial z}{\partial O_{11}}$

$\frac{\partial L}{\partial w'_{11}} = \frac{\partial O_{31}}{\partial O_{21}} \cdot \frac{\partial O_{21}}{\partial O_{11}} \cdot \frac{\partial O_{11}}{\partial w'_{11}}$

$2.00 \times 125 \times 100$

$= 0 \le \sigma(z) \le 0.25 * w_{21}$

$O_{21} = \sigma(z)$ $\frac{1}{1+e^{-z}}$

$= 0.25 \times 500 = 125$

$z = w_{21} \cdot O_{11} + b_2$