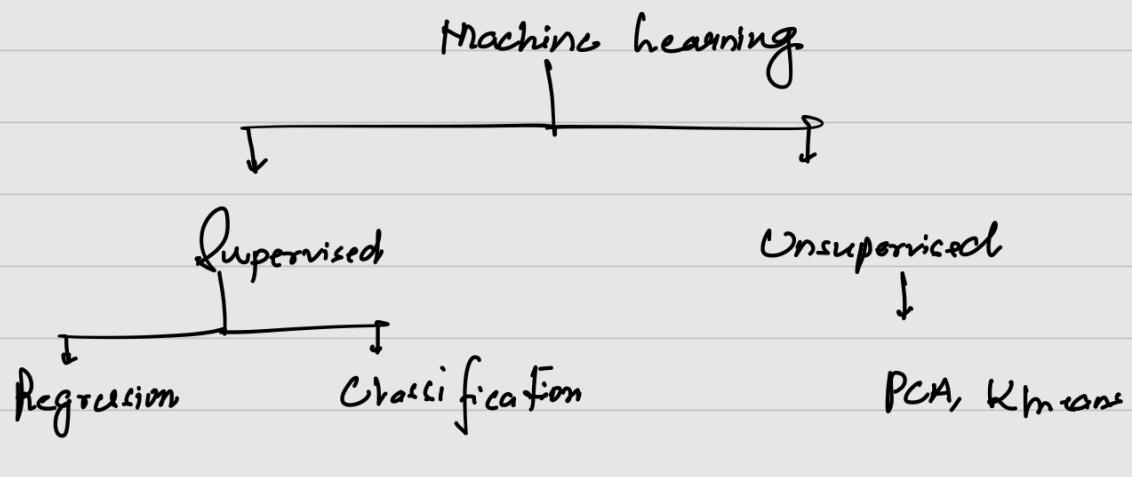


Introduction to Deep learning:



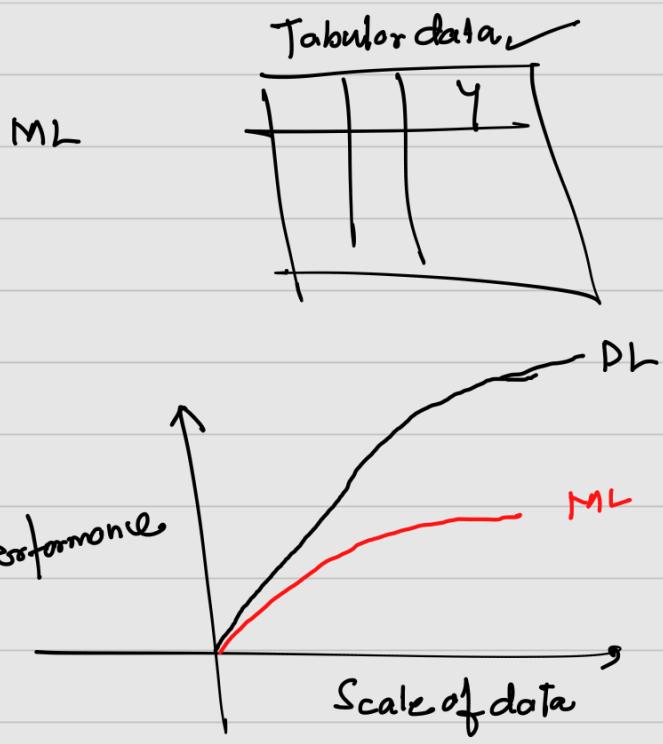
Linear Reg
Log Reg
DT, RF, Boosting

Mh → Parametric → equation $y = f(x)$
Non-Parametric process DT, RF, XGBoost

Deep learning → Parametric form:

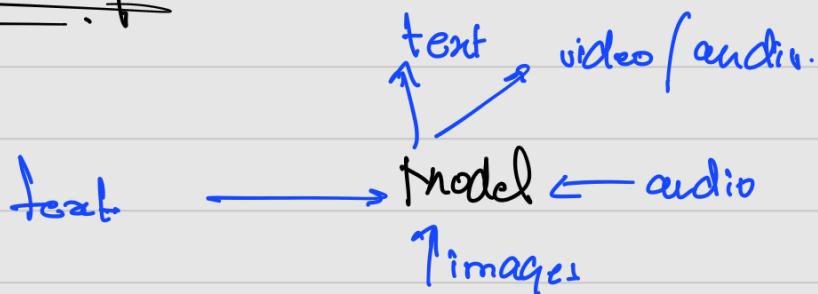
$$y \sim f(x)$$

↓
equation

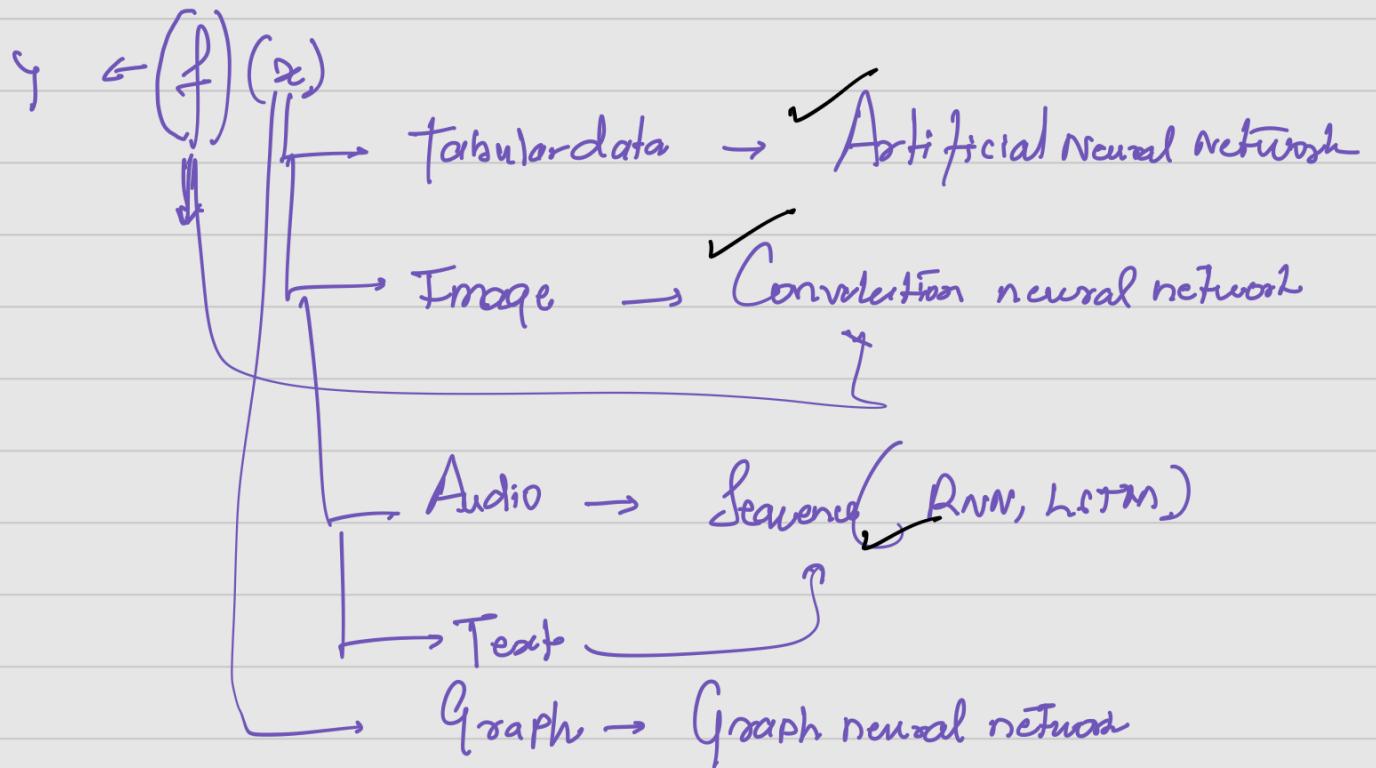


Automated feature engineering → Images, video, text,
Audio, Graph, etc.

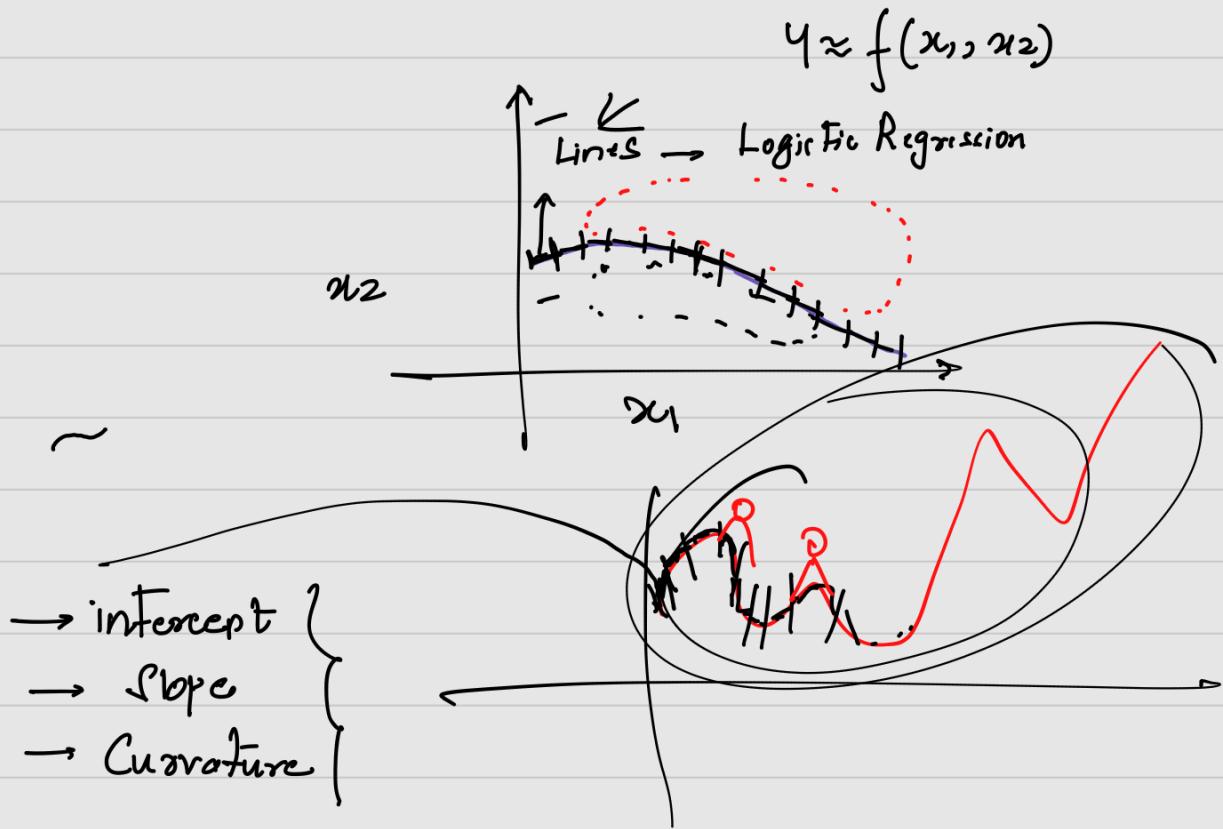
Multimodality



Deep learning a framework wherein we create a parametric model to predict y as a function of x .



Lecture-2 What is a Neuron?

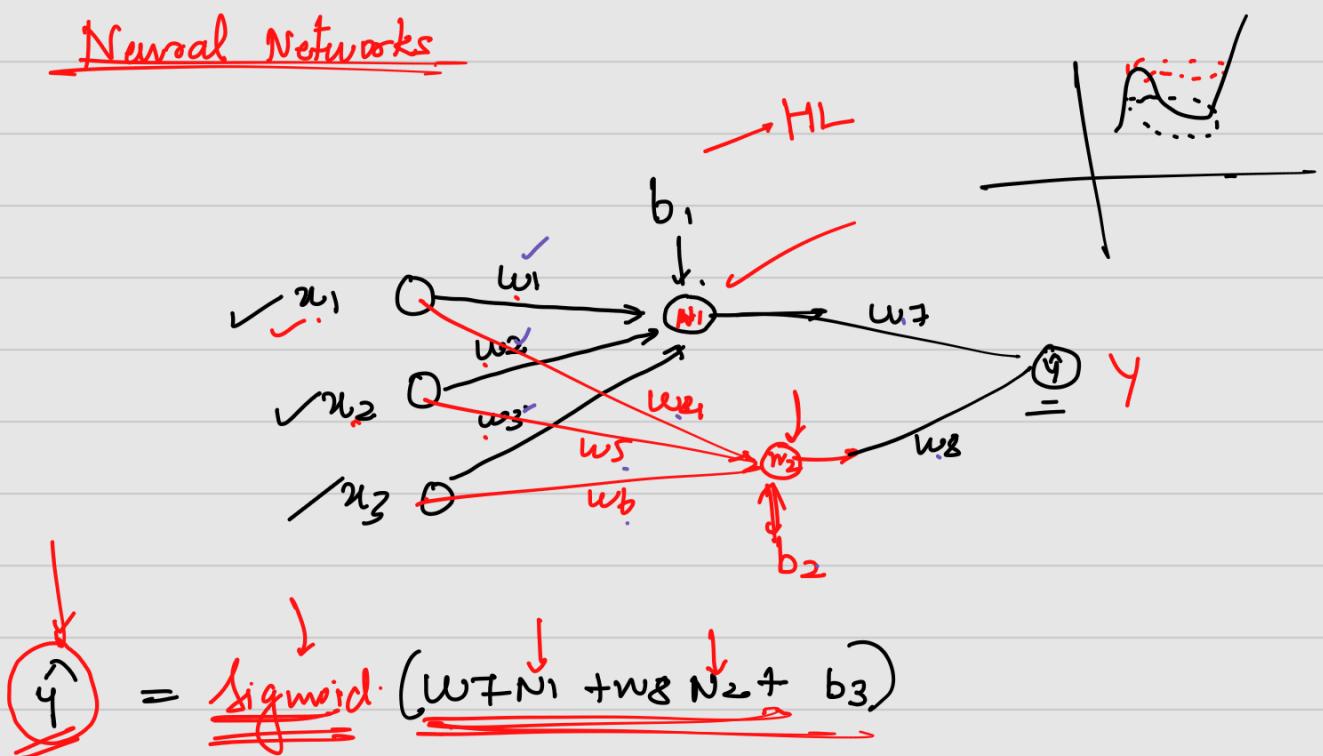




$$y = \text{NL}(w_1x_1 + w_2x_2 + w_3x_3 + b)$$

↓
activation functions

Neural Networks



x_1	x_2	x_3	y
✓	✓	✓	✓
✓	✓	✓	✓
✓	✓	✓	✓

If I want my \hat{y} to go as close as possible to y , modifying the weights and biases.

- ① Define the architecture.
- ② weights and biases \rightarrow randomly initialize.
- ③ $(\hat{y} \leftarrow y) \rightarrow \text{loss}$

- ④ Reduce the loss by modifying the weights and biases.

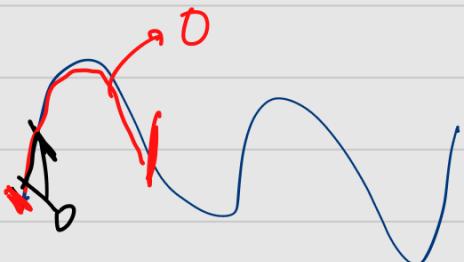
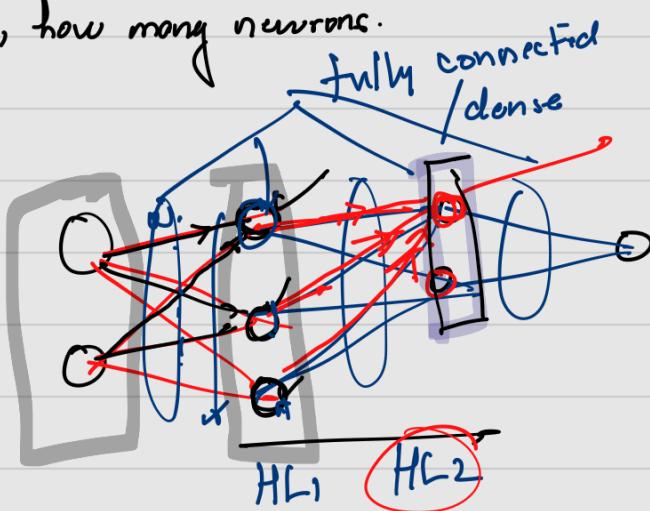
Gradient Descent

How to train Neural networks:

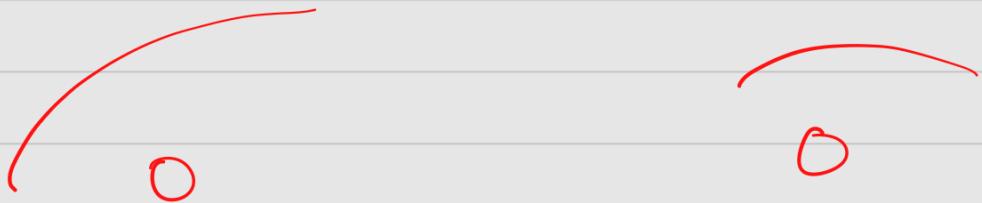
- ① Define my neural net architecture.

How many HL, how many neurons.

x_1	x_2	y

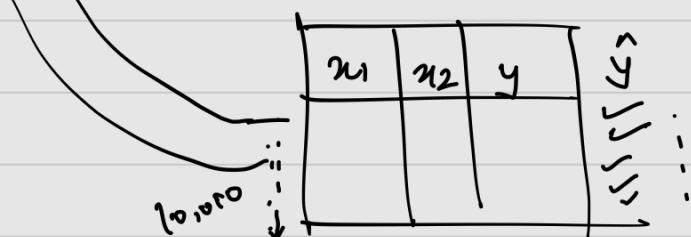
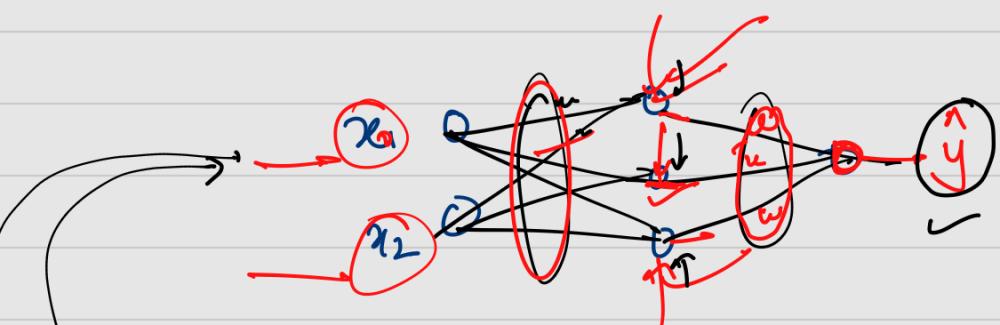


$\xrightarrow{\text{add more degree of non-linearity}}$



→ all weights and biases will be randomly initialized.

→ given these weights, f will multiply the weights to the inputs
add the biases



Regression

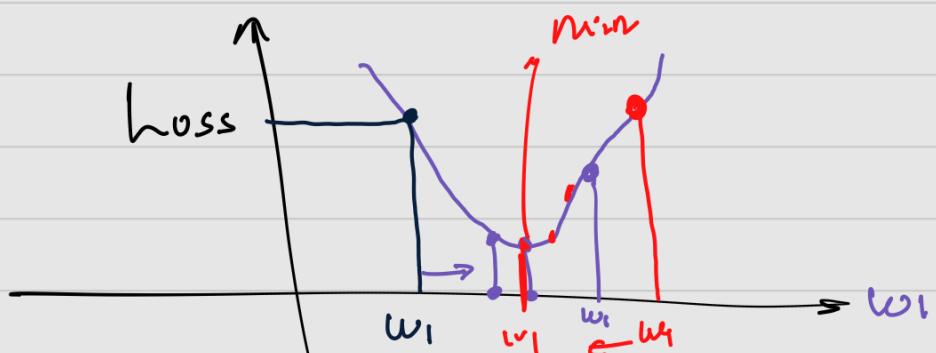
$$\text{loss} = \sum_{i=1}^n \frac{1}{n} (y_i - \hat{y}_i)^2$$

Forward Propagation

Based on the loss, we find out better set of weights and biases.

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

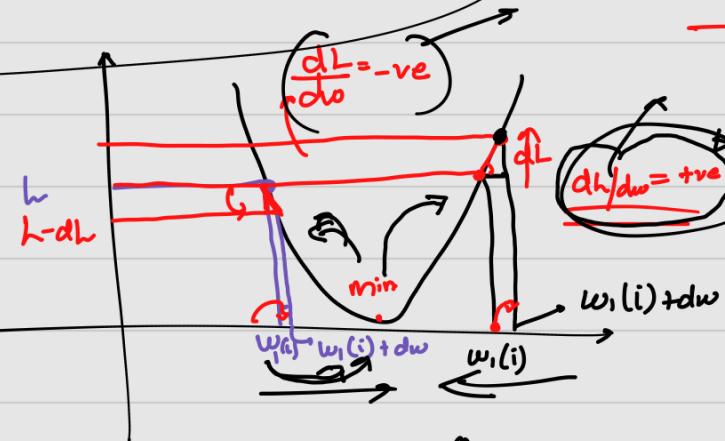
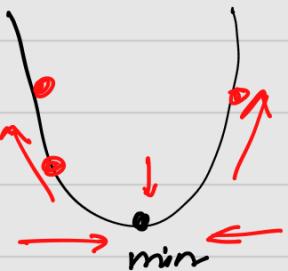
$$\text{loss} = f(\underline{w}, b)$$



for all weights and biases.

$$w_1(i+1) = \underline{w_1(i)} - d$$

$$\frac{dh}{dw(i)}$$

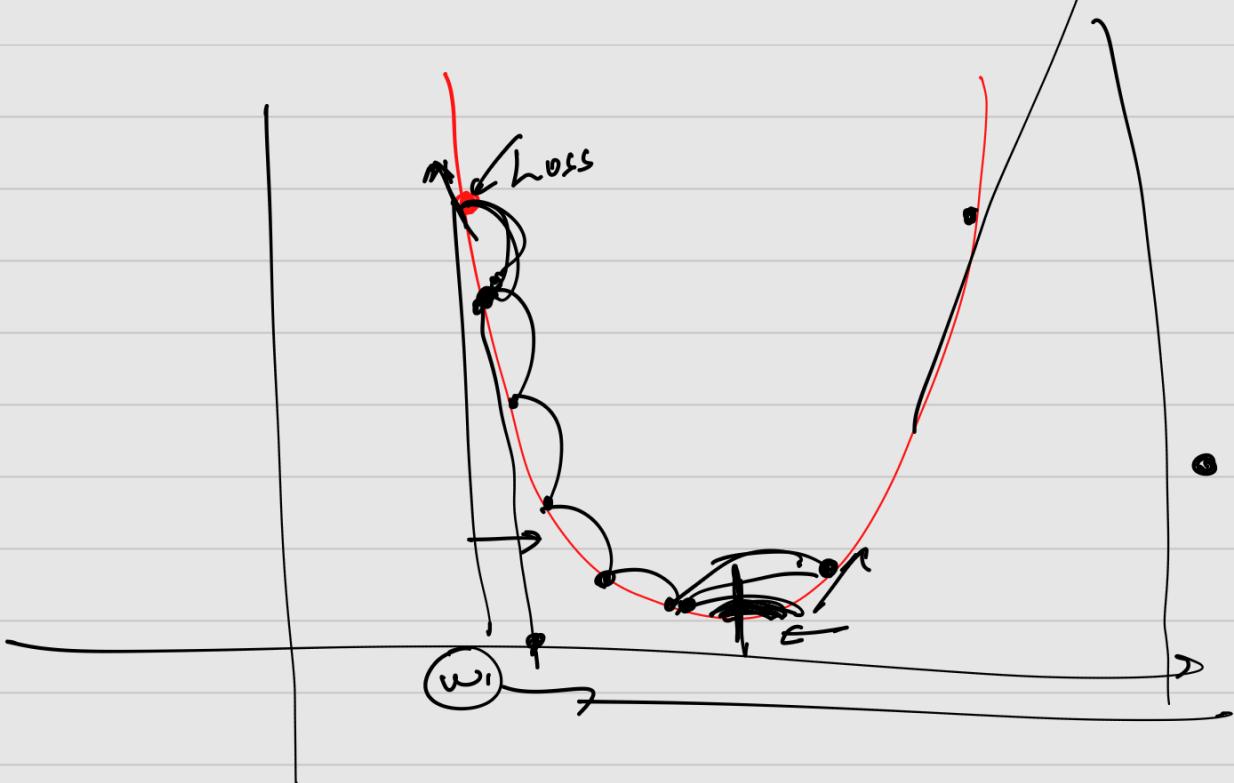


$$w_1(i+1) = \underline{w_1(i)} - \frac{dh}{dw(i)}$$

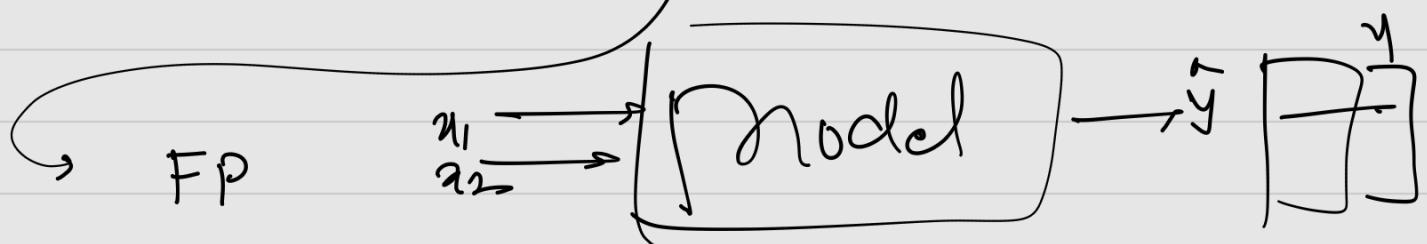
$w_1(i) + \text{amount}$

$$w_1(i+1) = \underline{w_1(i)} - \frac{d}{\cancel{h/dw}}$$

$w_1(i) - \text{amount}$



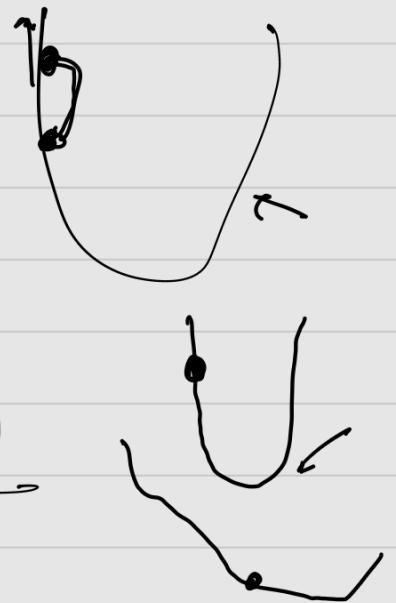
\mathcal{J} will have new set of weights and biases

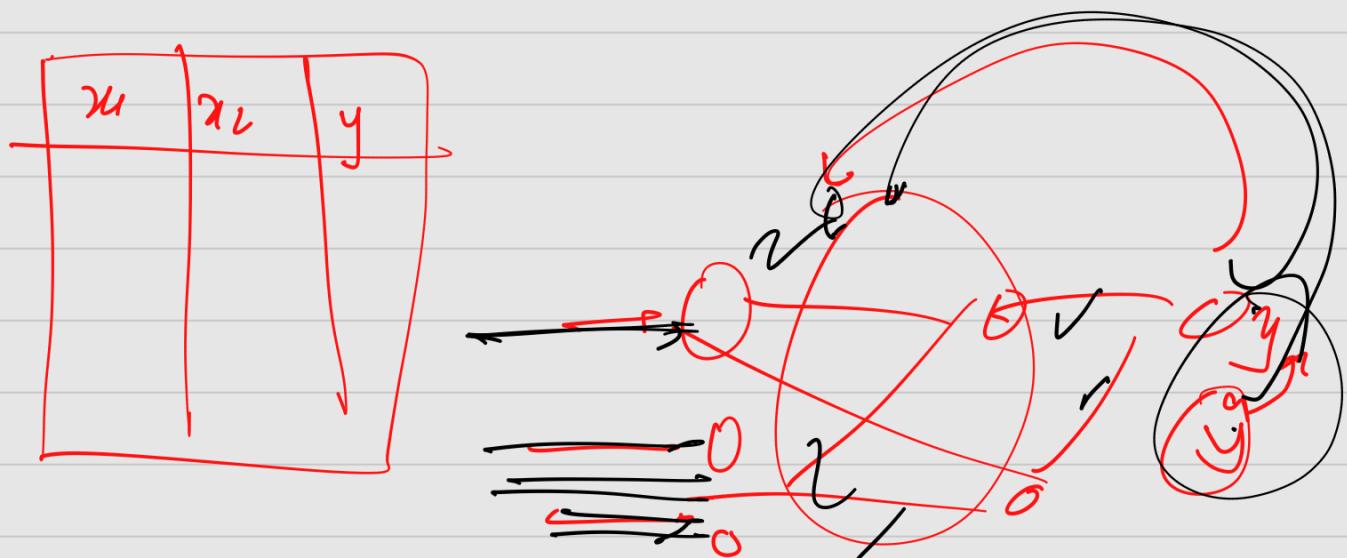
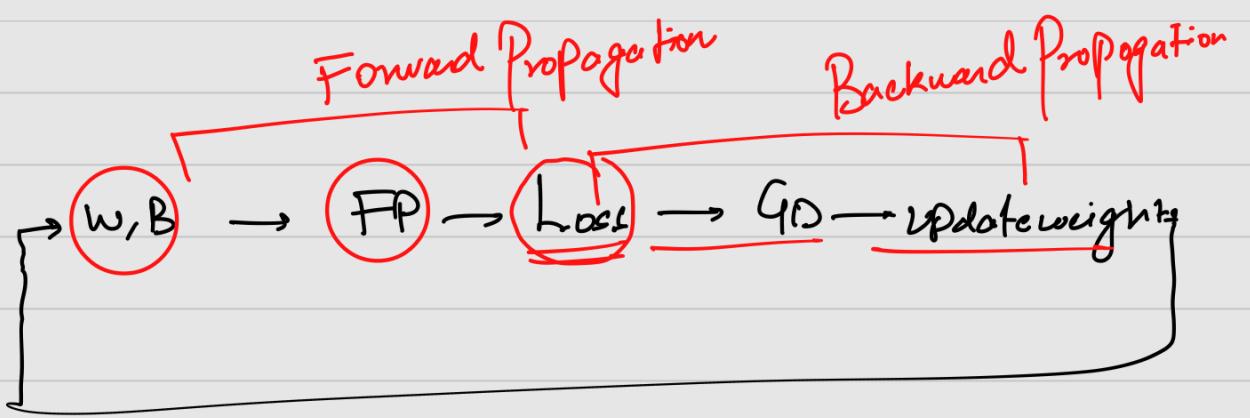


loss
gradient descent

j for all
weights

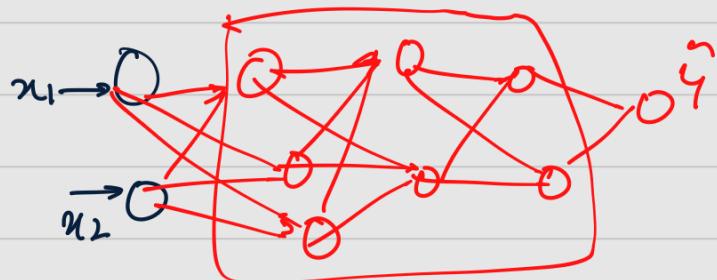
$$w_j(i+) = w_j(i) - \alpha \frac{\partial h}{\partial w_j(i)}$$





$$(1FP, 1BP) \rightarrow C_{\text{epoch}}$$

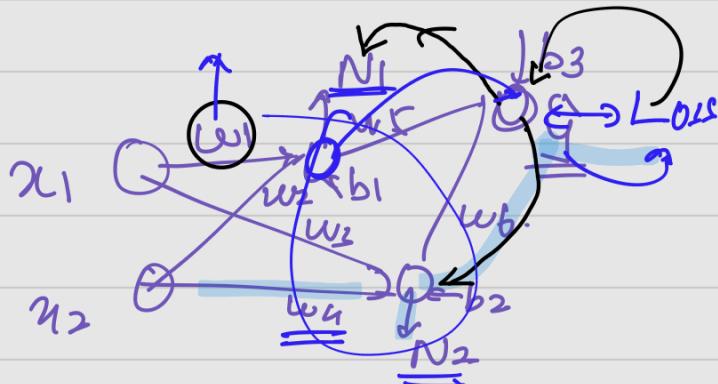
Chain Rule:



$$\underline{\underline{w_i(t+1)}} = \underline{\underline{w_i^*(t)}} - \underbrace{\frac{\partial L}{\partial w_i(t)}}_{\text{for all } i}$$

$$\left(\frac{\partial h}{\partial w_1}, \frac{\partial L}{\partial w_2}, \frac{\partial h}{\partial w_3}, \frac{\partial L}{\partial w_4} \dots \right) \left(\frac{\partial L}{\partial b_1}, \frac{\partial L}{\partial b_2} \dots \right).$$

Regression



$$N_1 = w_1 x_1 + w_2 x_2 + b_1$$

~~$$N_2 = w_3x_1 + w_4x_2 + b_2$$~~

$$\hat{y} = w_1 \underline{x_1} + w_2 \underline{x_2} + b_3 \Rightarrow \frac{\partial L}{\partial \hat{y}} = \sum_{j=1}^n \frac{1}{n} 2x(y_j - \hat{y}_j) - 1$$

$$\text{loss} = \sum_{i=1}^n \frac{1}{n} (y_i - \hat{y}_i)^2 \leftarrow \frac{\partial \text{loss}}{\partial w_2} = w_2 \frac{\partial N_2}{\partial w_2}$$

$$L = f(w, \theta)$$

$$\frac{\partial L}{\partial w} = ?$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial y} \times \frac{\partial y}{\partial N_1} \times \frac{\partial N_1}{\partial w_1}$$

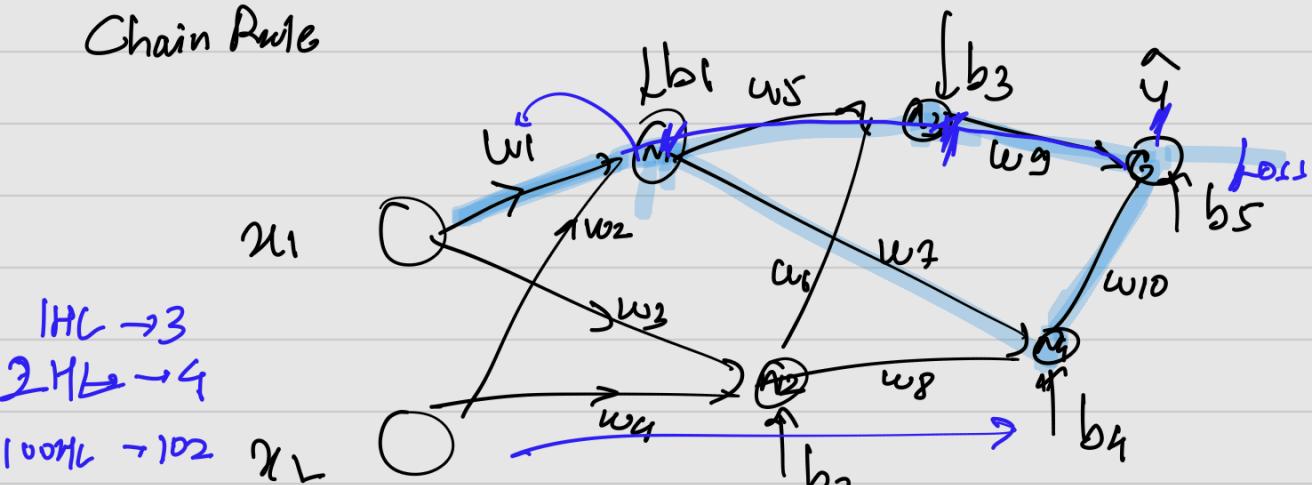
$$\begin{aligned} m &= z^2 \\ z &= m^2 \\ y &= z^2 \end{aligned}$$

$$\frac{\partial y}{\partial z} = 2z$$

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z} \times \frac{\partial z}{\partial m} \times \frac{\partial m}{\partial x}$$

$$\frac{\partial y}{\partial x} = 2z \times 2m \times 2x$$

Chain Rule



$$\begin{aligned} IHL \rightarrow 3 \\ 2HL \rightarrow 4 \\ 100HL \rightarrow 102 \end{aligned}$$

$$\begin{aligned} (0.99)^3 b_5 \\ = 0.001 b_5 \end{aligned}$$

$$\begin{aligned} \frac{\partial L}{\partial w_1} &= \frac{\partial L}{\partial y} \times \frac{\partial y}{\partial N_1} \times \frac{\partial N_1}{\partial w_1} + \\ &\quad \frac{\partial L}{\partial y} \times \frac{\partial y}{\partial N_2} \times \frac{\partial N_2}{\partial w_1} \times \frac{\partial N_1}{\partial w_1} \end{aligned}$$

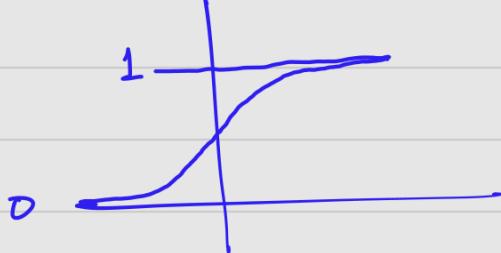
Vanishing gradient
Exploding gradient

Activations

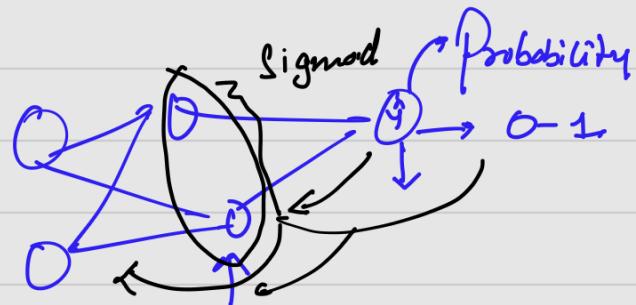
- Sigmoid.
- ReLU (Rectified Linear Unit).
- Leaky ReLU (Parametric ReLU).
- tanh (hyperbolic tangent).
- Softmax.

Sigmoid activation:

$$\begin{array}{c} \downarrow \\ y = B_0 + B_1 x_1 + B_2 x_2 \dots B_p x_p \\ \downarrow \\ -\infty, \infty \end{array}$$

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


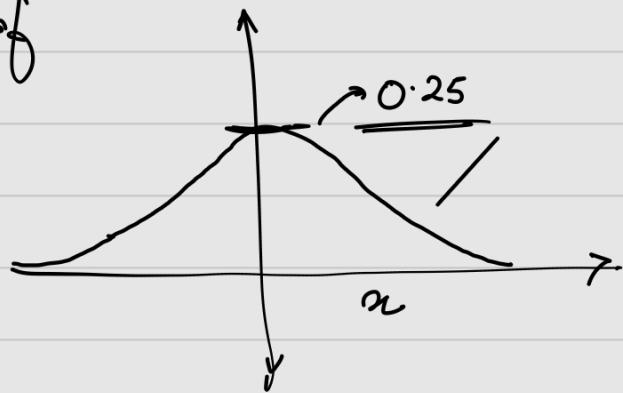
Binary Classification



$$w_{i+1} = w_i - \alpha \frac{\partial J}{\partial w_i}$$

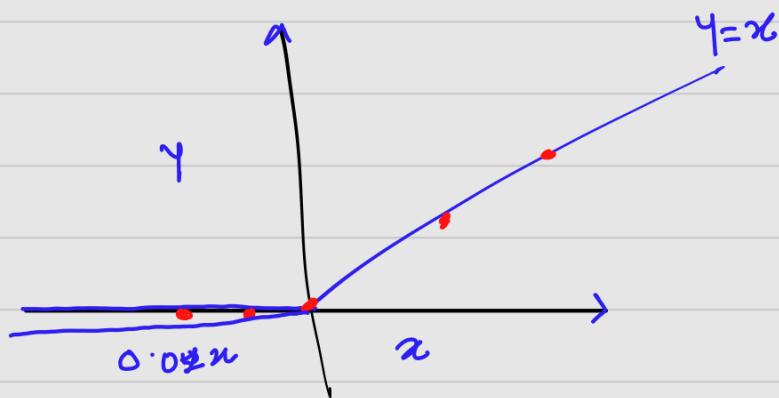
$$\sigma'(x) = \sigma(x) \times (1 - \sigma(x))$$

derivative of
Sigmoid



d. $\times \dots \times \dots \dots$

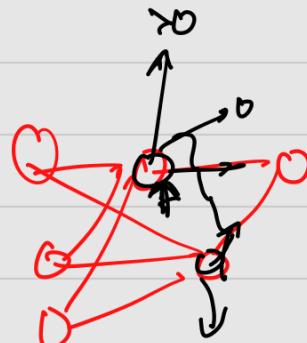
ReLU



$\text{ReLU}(x)$

$$\begin{array}{ll} x > 0 & x \\ x \leq 0 & 0 \end{array}$$

$$\text{ReLU}'(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$



Parametric ReLU.

$$\text{ReLU}(x) \quad x \quad x > 0$$

$$0 \quad x = 0$$

$$\alpha x \quad x < 0$$

$0 < \alpha < 1 \rightarrow$ learned during the training



Identity ReLU

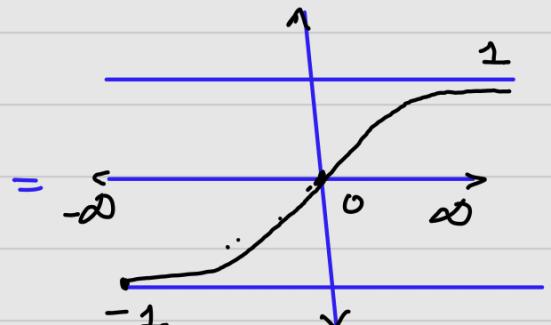
$$x > 0 \rightarrow x$$

$$x = 0 \rightarrow 0$$

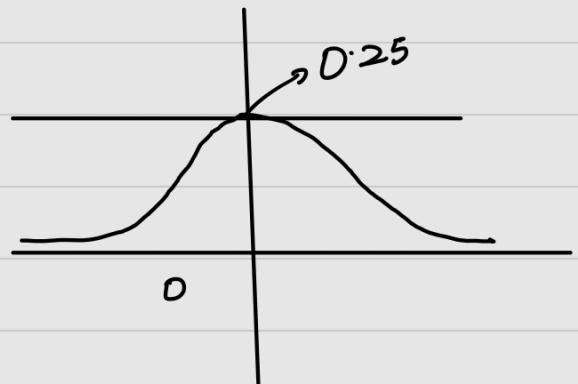
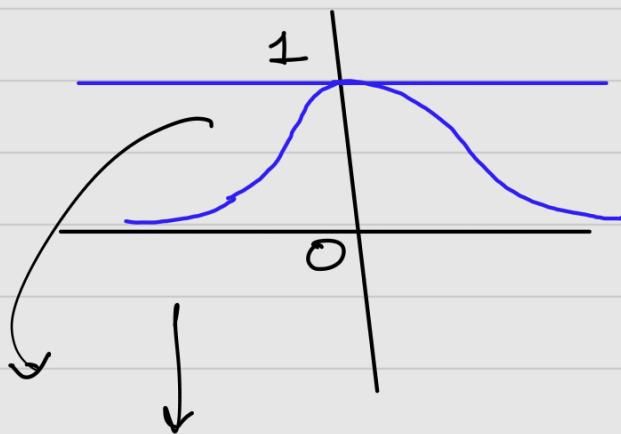
$$x < 0 \rightarrow 0.01x$$

Hyperbolic Tangent (tanh).

$$\tanh(x) = \left[\frac{e^x - e^{-x}}{e^x + e^{-x}} \right]$$



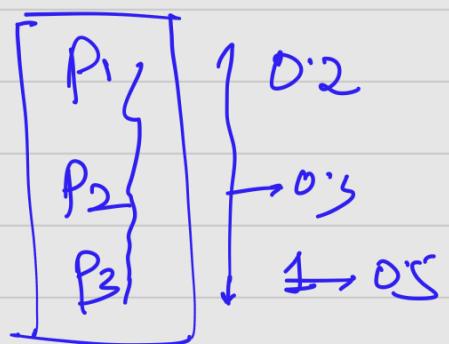
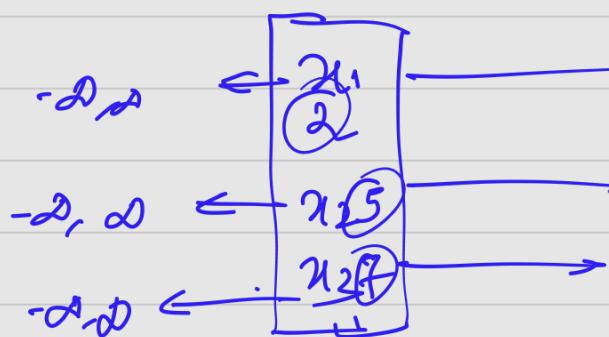
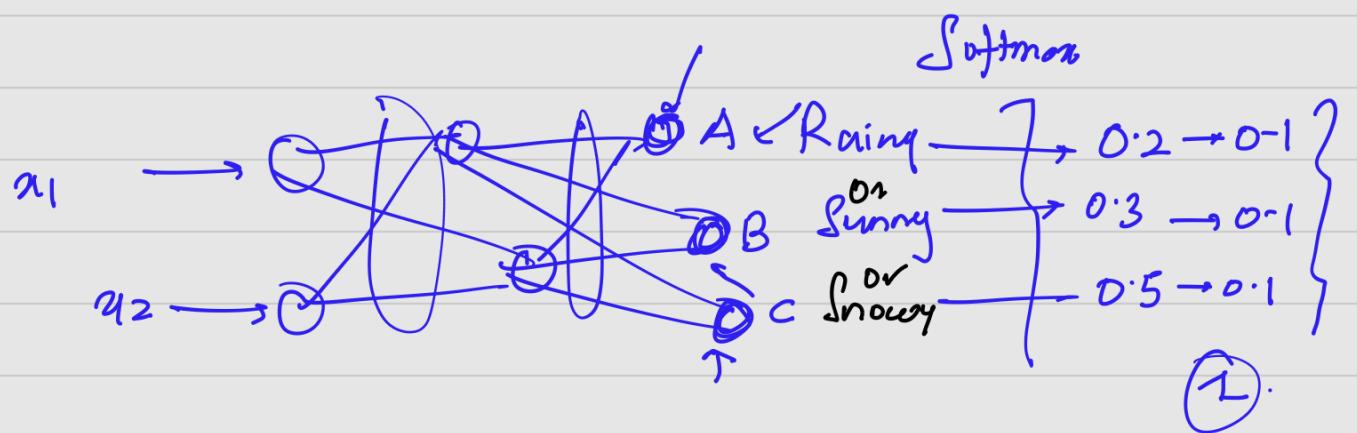
$$\tanh'(x) = 1 - \tanh^2(x)$$



Vanishing Gradient Problem

Softmax:

Sample / row



$$P_3 = \frac{e^{x_3}}{e^{x_1} + e^{x_2} + e^{x_3}}$$

$$P_1 = \frac{e^{x_1}}{e^{x_1} + e^{x_2} + e^{x_3}}$$

$$P_2 = \frac{e^{x_2}}{e^{x_1} + e^{x_2} + e^{x_3}}$$