

Deep learning

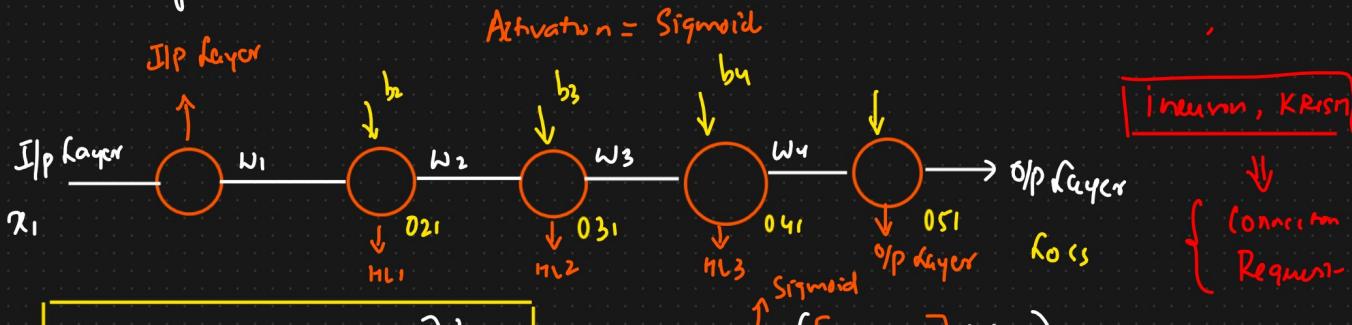
- ① Vanishing Gradient Problem ✓ }
- ② Activations Functions ✓ }
- ③ Loss functions or Cost function = }

[Optimizers] → ANN.

[inblog.in]

[Medium]

④ Vanishing Gradient Problem



$$w_{1,\text{new}} = w_{1,\text{old}} - \eta \frac{\partial L}{\partial w_{1,\text{old}}}$$

$$o_{51} = \sigma([o_{41} * w_4] + b_5)$$

↑ → Sigmoid

$$\text{Sigmoid} = \frac{1}{1+e^{-z}}$$

↓ Chain Rule of Derivatives.

$$L = (w^T x + b)$$

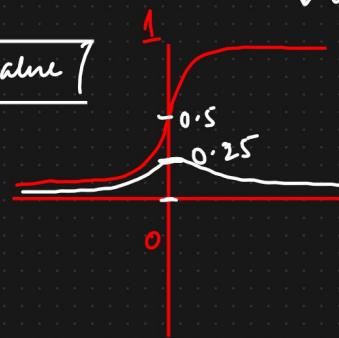
$$\frac{\partial L}{\partial w_{1,\text{old}}} = \left[\frac{\partial L}{\partial o_{51}} \right] * \left[\frac{\partial o_{51}}{\partial o_{41}} \right] + \frac{\partial o_{41}}{\partial o_{31}} + \frac{\partial o_{31}}{\partial o_{21}} * \frac{\partial o_{21}}{\partial w_{1,\text{old}}}$$

$$\text{val} * 0.25 * 0.10 * 0.05 * 0.001 * 0.0005$$

↓ ↓ ↓ ↓

$$\text{Sigmoid} = \frac{1}{1+e^{-z}}$$

| Smaller value |



$$\frac{d \sigma(z)}{dz} = [0 \text{ to } 0.25]$$

Sigmoid

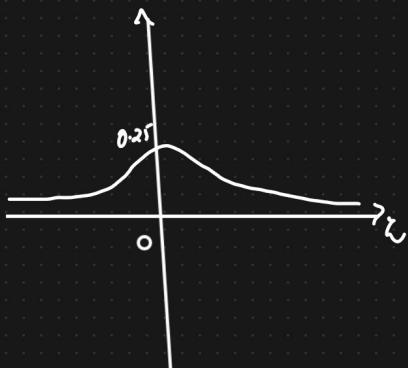


$$z = (w^T x + b)$$

$$y = \sigma(z)$$

$$y = \sigma(w^T x + b)$$

Derivative of Sigmoid



① Weights will not get updated

$$w_{i,\text{new}} = w_{i,\text{old}} - \eta \left(\frac{\partial L}{\partial w_{i,\text{old}}} \right) \Rightarrow \text{small}$$

$$\eta = 0.001$$

$$= w_{i,\text{old}} - (\text{small}) (\text{small}) (0.001) \neq (0.000005)$$

$$w_{i,\text{new}} \approx w_{i,\text{old}}$$

\Rightarrow Vanishing GRADIENT PROBLEM



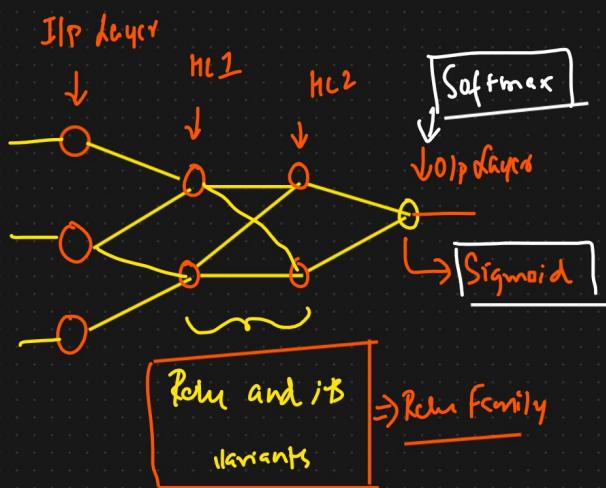
[Weights are not getting updated]

② To fix this problem we change the Activation function

Multiclass Classification

Binary Classification

[Which Activation function To use when]



$$\max(0, x) \times$$

① 0 or 1

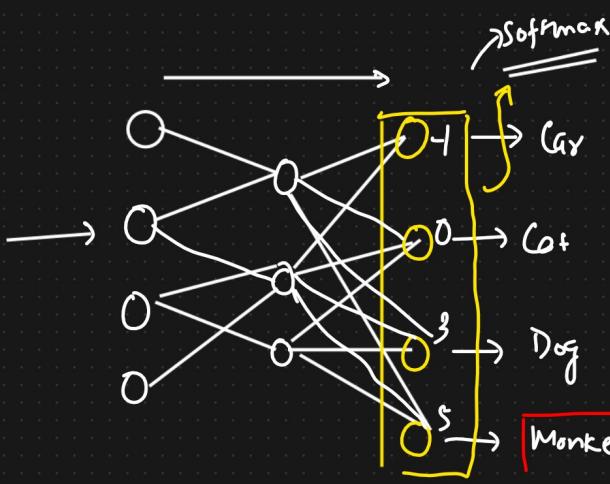
Vanishing Gradient Problem

x Sigmoid or Tanh

↓
Sigmoid

$$= \frac{1}{1 + e^{-x}} = 0 \text{ or } 1$$

Softmax



+1

$$\text{Softmax} = \left[\frac{y_i}{e^{\sum_{k=0}^n y_k}} \right] \quad \begin{cases} \Rightarrow 0.002 \\ \Rightarrow 1 \Rightarrow 0.005 \\ \Rightarrow 0.118 \\ \Rightarrow 0.874 \end{cases}$$

$$= \frac{1}{e^{-1} + e^0 + e^3 + e^5} = \frac{0.36}{169.87}$$

$$\text{Car} = \frac{e^{-1}}{[e^{-1} + e^0 + e^3 + e^5]} = \frac{0.3e}{169.87} = \boxed{0.002} \Rightarrow \text{probability}$$

$$\text{Cat} = \frac{e^0}{[e^{-1} + e^0 + e^3 + e^5]} = 0.006$$

$$\text{Dog} = \frac{e^3}{169.87} = 0.118$$

$$\text{Monkey} = \frac{e^5}{169.87} = 0.874$$

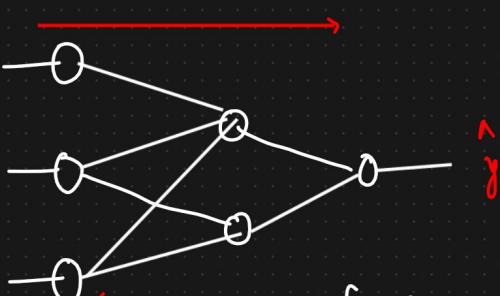
Linear Regression ✓

Sigmoid }
Softmax } [Output layer]



① Loss functions And Cost functions

① Regression }
② Classification }



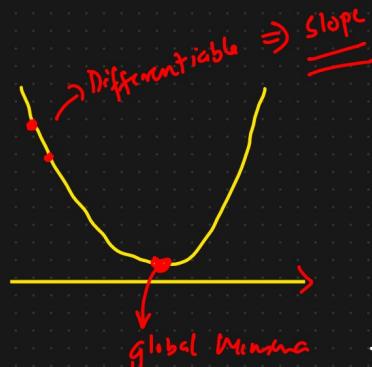
$$\text{Loss function} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

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

① Mean Squared Error (MSE)

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

↓
 Quadratic Equation



Advantages

- ① Differentiable ✓
- ② It has 1 local or global Minima
- ③ It converges faster

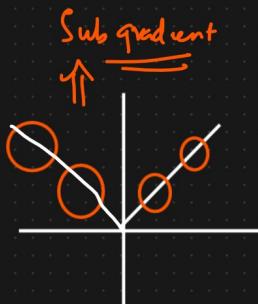
Disadvantages

- ① Not Robust to outliers

② Mean Absolute Error (MAE).

$$\text{Loss} = \frac{1}{n} |y - \hat{y}|$$

$$\text{Cost fn} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$



Adv

- ④ Robust to outliers

Rmse ←

③ Huber Loss ←

MSE

MAE

than an outlier

$$\text{Loss} = \begin{cases} \frac{1}{n} (y - \hat{y})^2 & \text{if } |y - \hat{y}| \leq \delta \\ \delta |y - \hat{y}| - \frac{1}{2} \delta^2, & \text{otherwise} \end{cases}$$

↓
hyperparameter

① Classification



① Binary Cross Entropy [log loss] → Logistic Regression

$$\text{loss} = -y * \log(\hat{y}) - (1-y) * \log(1-\hat{y}) \Rightarrow \text{Logistic Regression}$$

$$\text{loss} = \begin{cases} -\log(1-\hat{y}) & \text{if } y=0 \\ -\log(\hat{y}) & \text{if } y=1 \end{cases}$$

$$\boxed{\hat{y} = \frac{1}{1+e^{-x}}} \Rightarrow \text{Sigmoid} \Rightarrow \hat{y}$$

$$\uparrow \text{Softmax} \Rightarrow \hat{y}$$

② Categorical Cross Entropy {Multi Classification}

				y_{11}	y_{12}	y_{13}	
				$j=1$	$j=2$	$j=3$	
f_1	f_2	f_3	0 1	Good	Bad	Neutral	$i = \text{Row}$
→ 2	3	4	Good	1	0	0	$j = \text{Column}$
5	6	7	Bad	0	1	0	
8	9	10	Neutral	0	0	1	

$$d(x_i, y_i) = - \sum_{j=1}^c y_{ij} * \ln(\hat{y}_{ij})$$



$$Y_{ij} = [y_{11}, y_{12}, y_{13}, \dots, y_{1j}]$$

$$[y_{21}, y_{22}, y_{23}, \dots, y_{2j}]$$

$$Y_{ij} = \begin{cases} 1 & \text{if the element is in the class} \\ 0 & \text{otherwise} \end{cases}$$

Predicted \leftarrow $\hat{Y}_{ij} = \frac{\text{Softmax Activation}}{\text{Output Layer}}$

$$\boxed{T(z) = \frac{e^{z_i}}{\sum_{j=1}^k e^{z_j}}} \Rightarrow \underline{\text{Softmax}}$$

Conclusion

$[ReLU, \text{Softmax}] \Rightarrow \text{Categorical Cross Entropy}$

$[ReLU, \text{Sigmoid}] \Rightarrow \text{Binary Cross Entropy}$

$[ReLU, \text{Linear}] \Rightarrow \text{MSE, MAE, Huber Loss, RMSE}$

Regression

$$\begin{matrix} 0 & \downarrow R_{cm} & \downarrow R_{cm} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} \xrightarrow{\text{sigmoid.}}$$