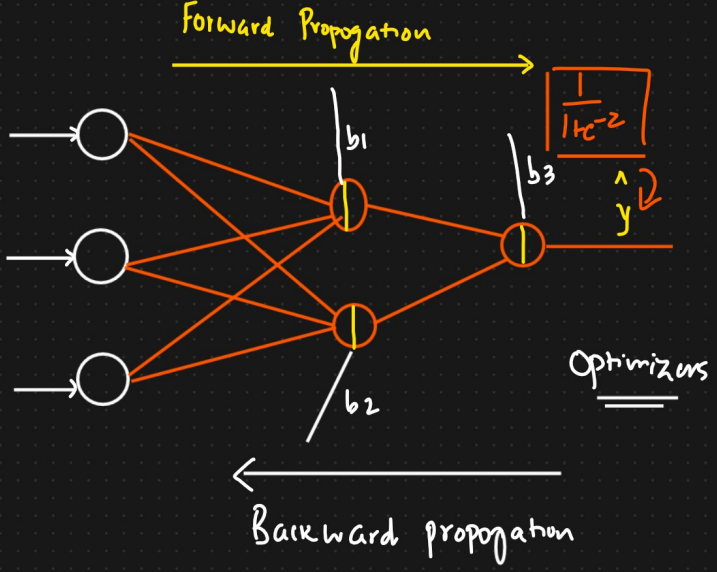
① Loss function And Cost function

② Optimizers

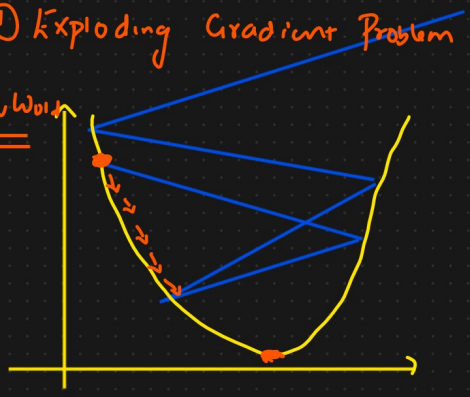


Weight Initialization

Uniform Xavier

10000

① Exploding Gradient Problem



Loss function {Single datapoint} MSE

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

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

ANN

- CLASSIFICATION
- REGRESSION
  1. MSE {Mean Squared Error}
  2. MAE {Mean Absolute Error}
  3. Huber Loss
  4. RMSE

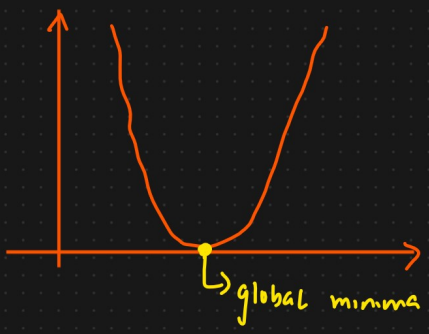
① Mean Squared Error

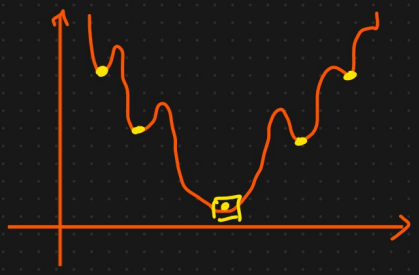
Loss fn $= (y-\hat{y})^2$

⇓

Quadratic Equation

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



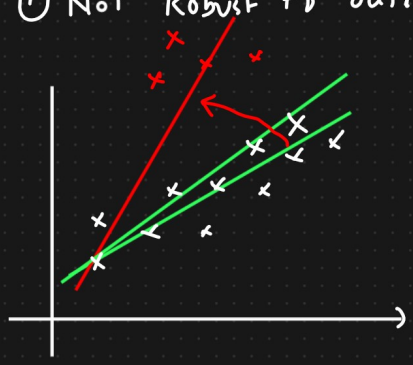


Advantages

1. MSE is Differentiable
2. It has 1 local or 1 global Minima
3. It converges faster

Disadvantages

1. Not Robust to outliers



2. Mean Absolute Error (MAE)

$$\text{Loss fn} = |y - \hat{y}|$$

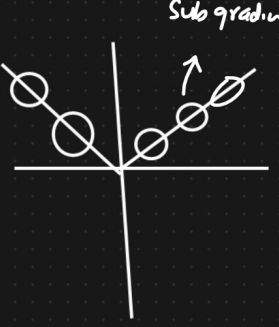
$$\text{Cost fn} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Advantages

1. Robust to outlier

Disadvantage

1. Convergence is slow



3. Huber Loss

1. MSE
2. MAE

$$\text{Cost fn} = \begin{cases} \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 & \text{if } |y-\hat{y}| \le \delta \\ \delta|y-\hat{y}| - \frac{1}{2}\delta^2, & \text{otherwise.} \end{cases}$$

$\Downarrow$ MSE

$\Downarrow$ MAE

δ ⇒ Hyperparameter

$\Downarrow$ Hyperparameter

δ → Eq

ⓐ Loss or Cost function for Classification Problem

Classification ⟶ CROSS ENTROPY
- ⟶ BINARY CROSS ENTROPY {Binary}
- ⟶ Categorical " " {Multiclass}
- ⟶ Sparse Categorical " " {Multiclass}

① Binary Cross Entropy

Log loss

$$loss = -y * \log(\hat{y}) - (1-y) * \log(1-\hat{y})$$

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

$$\hat{y} = \frac{1}{1+e^{-z}}$$

⇒ Sigmoid Activation fn

② Categorical CROSS ENTROPY (Multiclass classification)

C = No. of Categories

C = 3

	f_1	f_2	f_3	O/p	$j=1$ Good	$j=2$ Bad	$j=3$ Neutral
$i=1$	2	3	4	Good	1	0	0
$i=2$	5	6	7	Bad	0	1	0
$i=3$	8	9	10	Neural	0	0	1

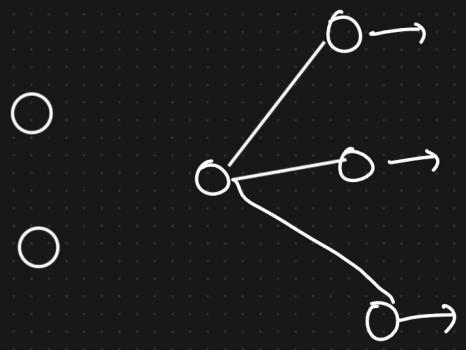
$$loss(x_i, y_i) = -\sum_{j=1}^{C} y_{ij} * \ln(\hat{y}_{ij})$$

Actual values ⇐ $y_{ij} = [\, y_{11} \quad y_{12} \quad y_{13} \cdots y_{1C} \,]$

$$[y_{21} \quad y_{22} \quad y_{23} \cdots y_{2c}]$$

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

Prediction $\Rightarrow \hat{y}_{ij} \Rightarrow$ Soft max Activation $= Sof(z) = \dfrac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$



$$= \frac{e^{z_i}}{e^1 + e^2 + e^3 + \cdots + e^K}$$

③ Sparse Categorical Entropy

$$\underset{\text{0}}{0.2}, \underset{\text{1st}}{0.3}, \underset{\text{2nd}}{0.5}$$

$$[0.2, 0.3, 0.1, 0.2, 0.2]$$

2nd Index $\Rightarrow$ Op

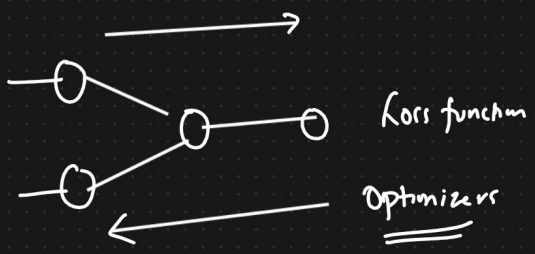
$\rightarrow 0.3 \rightarrow 1^{st}$ Index

④ Right Combination

Activation applied ⇓

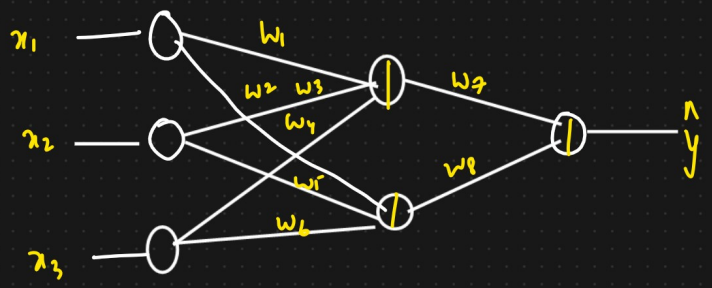
	Hidden Layers	O/P Layer	Problem Statement	Loss function
①	Relu or its Variants	Sigmoid	Binary classification	Binary Cross Entropy
②	Relu or its Variants	Softmax	Multi class	Categorical or Sparse CE
③	Relu or its Variants	linear	Regression	MSE, MAE, Huber loss, RMSE

Optimizers



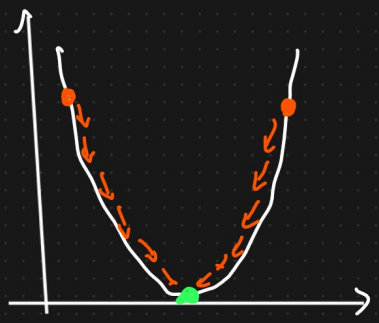
1. Gradient Descent
2. SGD (Stochastic Gradient Descent)
3. Mini batch SGD
4. SGD with Momentum
5. Adagrad and RmsProp
6. Adam optimizers

1. Gradient Descent Optimizer



Loss ↓↓

Optimizers



Weight Updation Formula

$$W_{new} = W_{old} - \eta \left[\frac{\partial h}{\partial w_{old}} \right] \Rightarrow \text{Slope} \Rightarrow \text{Chain Rule of Derivation}$$

MSE

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

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

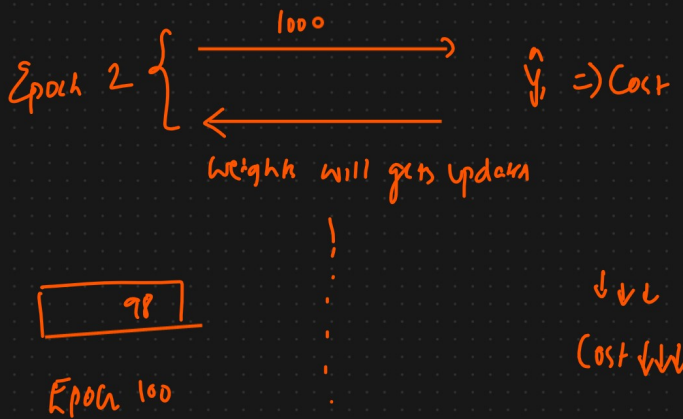
Epochs vs Iteration

DATAPOINTS = 1000 $\Rightarrow$ 1000000

Epoch 1 { 1000 datapoints → / 1 Iteration ← } $\hat{y}_i$ = Cost ↓↓

Weights will get update

100 Epochs



Advantage

8gb Ram

Disadvantage

① Convergence will happen

① Huge Ram And GPU => Resource Intensive.

② Stochastic Gradient Descent (SGD)

4th Ram

1000 datapoints

Epoche 1

Iteration 1

1

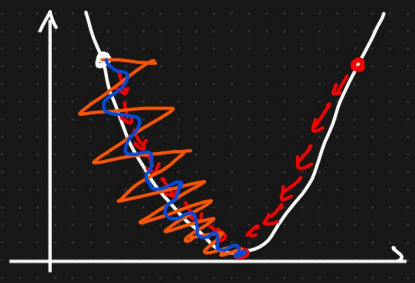
Loss

Iteration 1000

1000 x 1000 = 1

Epoch 2

Epoch 100



Advantage

④ Some Resource Issue

Disadvantage

- ④ Convergence will take time
- ② Noise will get introduced

③ Mini Batch SGD

8gb RAM

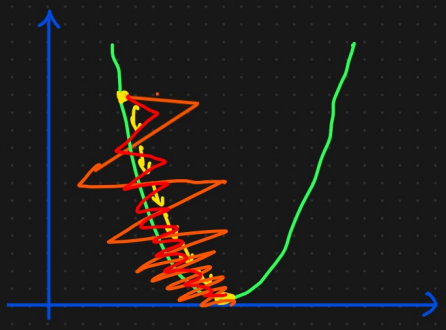
100000 / 100

Batch size

10000

100

Epoch, Iteration, Batch size



Epoch 1

Iterate

Advantage

① Convergence speed increases

② Noise will be less when compared to SGD

③ Efficient Resource USAGE

Disadvantage

① Noise still exist

④ SGD With Momentum

$$w_{new} = w_{old} - \eta \frac{\partial h}{\partial w_{old}}$$

$$b_{new} = b_{old} - \eta \frac{\partial h}{\partial b_{old}}$$

$$w_t = w_{t-1} - \eta \frac{\partial h}{\partial w_{t-1}}$$

Exponential Weight Average {Smoothening}

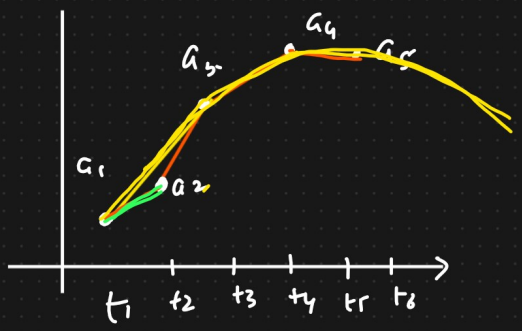
Time	t_1	t_2	t_3	t_4	...	t_n
Value	a_1	a_2	a_3	a_4	...	a_n

Exponential Weighted Average

$\beta = 0.05$ $\quad V_{t_1} = a_1$ $\quad 0.95 \Leftarrow \beta = 0.95$

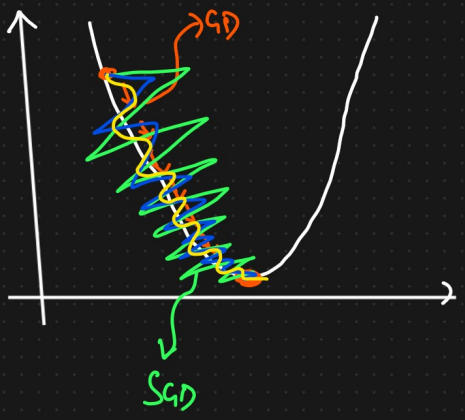
$\beta = 0$ $\quad V_{t_2} = \beta * V_{t_1} + (1-\beta) * a_2$ $\quad$ BW ↓

$= 0.95 * a_1 + 0.05 * a_2$



$V_{t_3} = \beta * V_{t_2} + (1-\beta) * a_3$

$= 0.95 [$



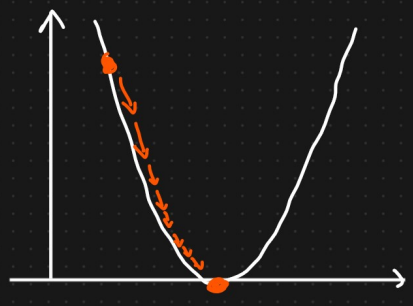
Advantage

① Reduces the noise
② Smoothen the noise
③ Quick Convergence.

⑤ **Adagrad** : {Adaptive Gradient Descent}

η = fixed ⇒ Dynamic LR

$$W_t = W_{t-1} - \eta \frac{\partial h}{\partial W_{t-1}}$$



$$W_t = W_{t-1} - \eta' \frac{\partial h}{\partial W_{t-1}}$$

$$\eta' = \frac{\eta}{\sqrt{\alpha_t + \epsilon}}$$ $\quad \eta' ↓↓↓↓$

Epsilon = Small value

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial h}{\partial W_t}\right)^2 \Rightarrow \text{Large value}$$

$$W_t \approx W_{t-1}$$

Disadvantage

① η' = Possibility to become a small value ≈ 0

⑥ Adadelta And RMS Prop [Exponent weighted Learning Rate] Exponential weighted Avg

$$\eta' = \frac{\eta}{\sqrt{Sdw_t + \epsilon}}$$

$d_t = Sdw_t$

$Sdw_t = 0$ ←

$$Sdw_t = \beta * Sdw_{t-1} + (1-\beta)\left(\frac{\partial h}{\partial w_t}\right)^2$$

Sdw_{t2}

⑦ Adam Optimizer

SGD With Momentum + RMSPROP [DYNAMIC LR]

$$W_t = W_{t-1} - \eta' V_{dw}$$

$$\eta' = \frac{\eta}{\sqrt{Sdw_t + \epsilon}}$$

$$Vdw_t = \beta * Vdw_{t-1} - (1-\beta)\frac{\partial h}{\partial w_{t-1}}$$

$$b_t = b_{t-1} - \eta' V_{db}$$

$$Vdb_t = \beta * V_{db_{t-1}} - (1-\beta)\frac{\partial h}{\partial b_{t-1}}$$

$$\eta' = \frac{\eta}{\sqrt{Sdb_t + \epsilon}}$$