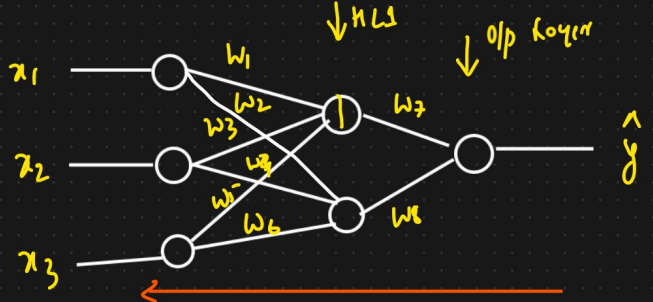
Optimizers

1. Gradient Descent
2. Stochastic Gradient Descent (SGD)
3. Mini batch SGD
4. SGD With Momentum
5. Adagrad and RMSPROP
6. Adam Optimizers

1. Gradient Descent Optimizer



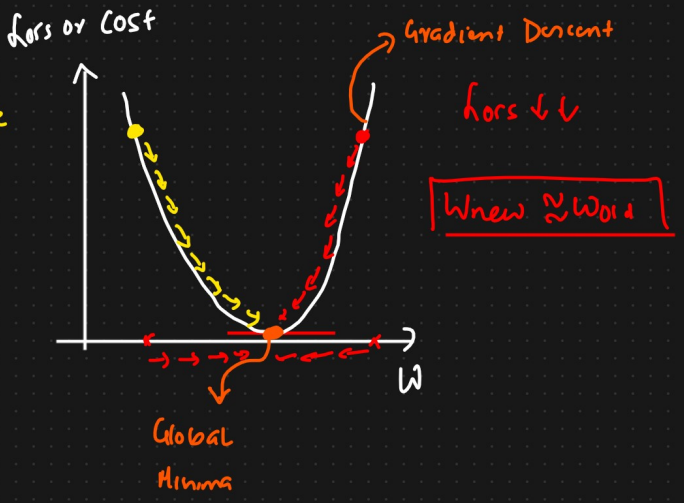
Backward Propogation

$$Loss = [\quad] \downarrow\downarrow\downarrow$$

Optimizers

Learning Rate

$$W_{new} = W_{old} - \eta \frac{\partial L}{\partial W_{old}}$$



MSE

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

Dataset = 1000 Data points

Epochs, Iteration

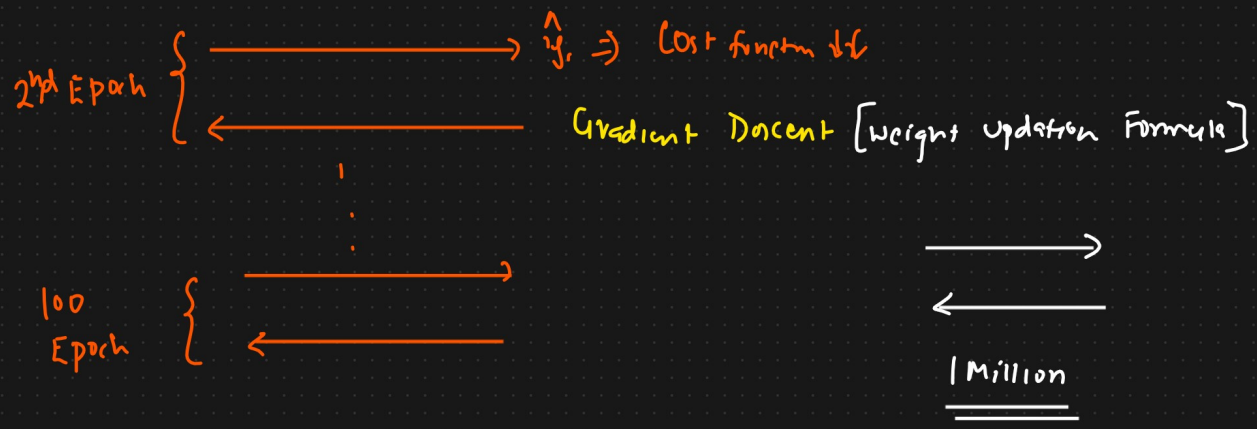
1 Epoch = 1 Iteration

10 Iteration

$100/10 = 100$

1 Epoch { 1000 datapoint → $\hat{y}_i$ ⇒ Cost function ↓↓ ; Weights will get updated ← } Gradient Descent optimizer [Weight updation]

100



Advantages

① Convergence will happen.

Disadvantage

① Huge Resource RAM, GPU

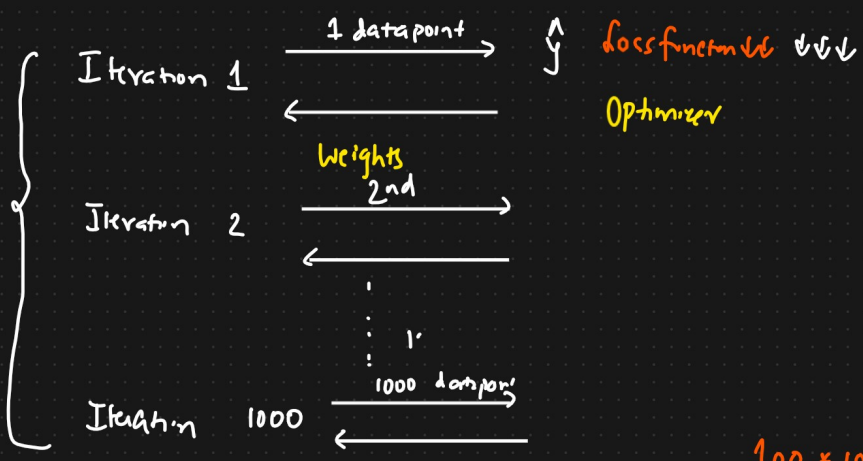
⇓

Resource Intensive.

② Stochastic Gradient Descent (SGD)

1000 datapoint

Epochs & Iteration



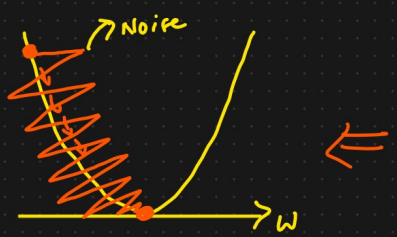
100 × 1000 = 100000

Epochs

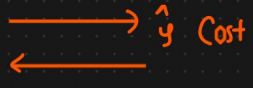
100000 datapoint

Advantage

Ⓐ Solve Resource Issue



Disadvantage



Ⓐ Time Complexity ↑↑

Ⓐ Convergence will also take more time.

Ⓐ Noise gets Introduced

③ Mini Batch SGD

No. of iterations = $\frac{100000}{1000}$ = 100 iterations

Epoch, Iteration, Batch_size

Datapoints = 100000

batch_size = 1000

MSE

Cost fn = $\sum_{i=1}^{1000} (y_i - \hat{y}_i)^2$ ↓↓

Epoch 1

Iteration 1 — 1000 →

← Optimizer ⇒ Mini Batch SGD

change the weights

Iteration 2 — 1000 →

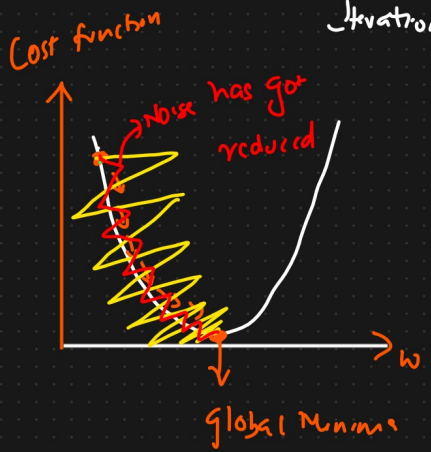
↓↓

8gb	1
16gb	5000

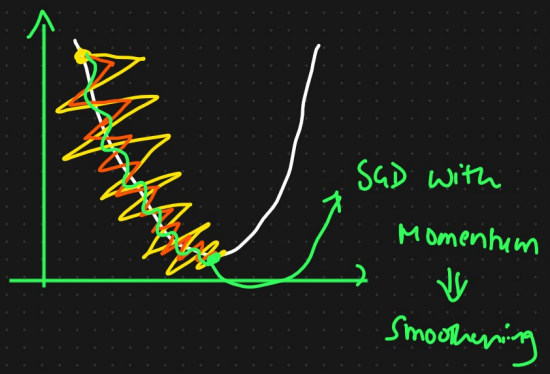
Iteration 3

!

Iteration 100



Batch Size



Advantages

① Convergence speed will increase =

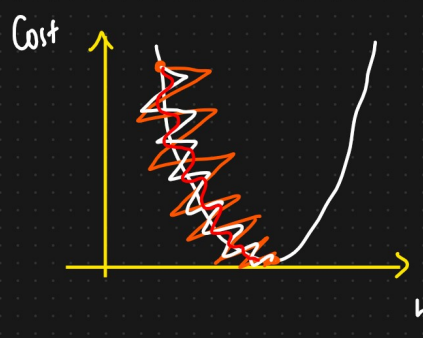
② Noise will be less when compared to SGD

③ Efficient Resource Usage (RAM)

Disadvantage

① Noise still exists

④ SGD With Momentum



Weight Updation formula

learning rate

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

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

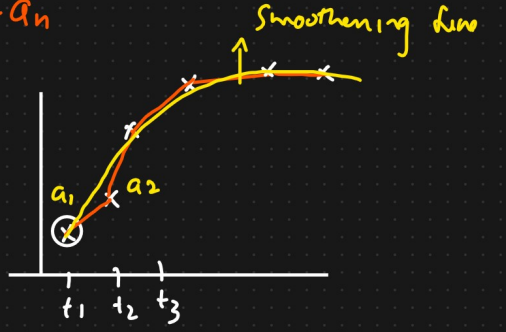
$$W_t = W_{t-1} - \eta \frac{\partial h}{\partial W_{t-1}}$$

Exponential Weight Average. {Smoothening} ⇒ ARIMA, SARIMAX

Time = t_1 t_2 t_3 t_4 - - - t_n

Values = a_1 a_2 a_3 a_4 - - - - a_n

Time Series



$$V_{t_1} = a_1$$

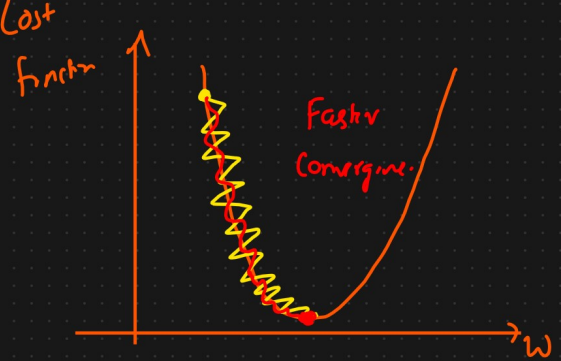
$$V_{t_2} = \boxed{\beta} * V_{t_1} + (1-\beta) * a_2$$

$\beta = 0$ to 1

$$= 0.95 * a_1 + (0.05)\, a_2$$

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

$$= 0.95 \left[0.95 * a_1 + (0.05)\, a_2 \right] + (0.05) * a_3$$



Advantage

① Reduces the noise

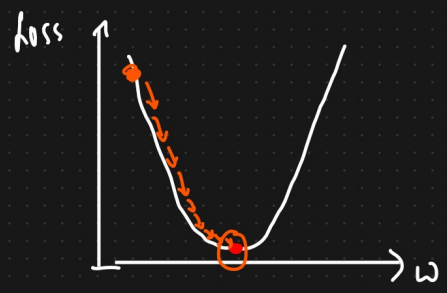
② Quick Convergence

⑤ Adagrad : Adaptive Gradient Descent

η = fixed ⇒ Dynamic learning Rate

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

→ Learning Rate = 0.001



As the convergence happen the learning rate should change

$0.00001 \approx 0$

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

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

$\eta' \downarrow\downarrow$ $\alpha_t \uparrow\uparrow$

ϵ → Epsilon ≈ small value

$\alpha_t \uparrow\uparrow$

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial h}{\partial w_t}\right)^2$$

$W_t \approx W_{t-1}$

$t=1 \rightarrow t=2 \rightarrow t=3$

$\eta = 0.01$ $\eta = 0.005$ $\eta = 0.003$

Disadvantage

① η' → Possibility to become a very small value ≈ 0

⑥ Adadelta And RMSPROP

Exponential Weight Average

$\beta = 0.95$ $\quad Sdw_t = 0$

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

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

[Dynamic LR + Smoothening EWA]

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

⑦ Adam Optimizer

SGD With Momentum + RMSPROP [Dynamic LR + Smoothening]

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

Weight Updation

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

→ Bias Updation

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

$Sdw_t = 0$

EWA

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

[EWA]

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

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

⇒ Momentum
↳ Smoothening