Optimizers

Introduction

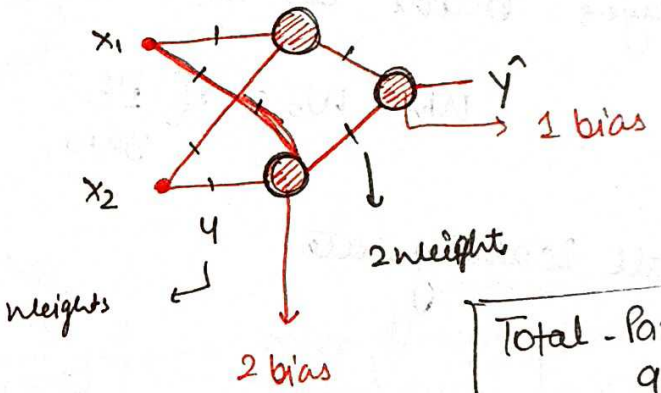
Performance of NN
↳ How to speed the training

Technique

→ Weight init
→ Batch Norm
→ Activation function
→ Optimizers

Role of optimizers

Cgpa	iq	placed
8	80	1
9	90	1
7	70	0

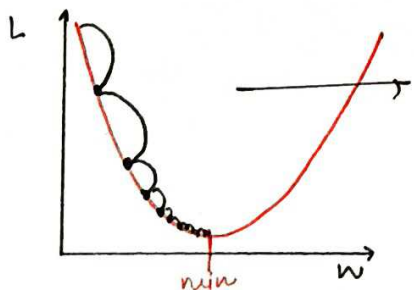


Total - Parameter is 9

Hume 9 parameter ki aisi value dhundni hai ki jab hum data NN mai dale to pred output close ho real output se.

Loss → $\hat{y} - y$ → final b value.
↳ w, b kis value pr optimum value hai.
final w value

Start with Random Value (w, b) ⟶ end with Optimize Value (w, b)



Optimize value
Optimize technique → (gradient descent)

Gradient Descent Update rule :

$$W_n = W_0 - \eta \frac{\partial L}{\partial W}$$

Types of Gradient descent

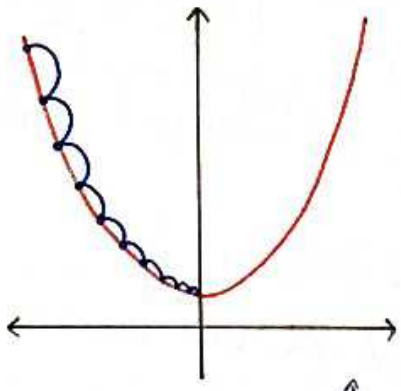
→ Batch GD
→ Stochastic GD
→ Mini ~~Batch~~ batch GD

Challenge No.1

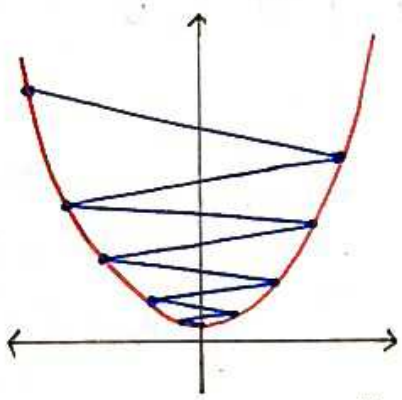
1. challenge → Decide correct ~~value~~ learning rate value.

$$W_n = W_0 - \eta \frac{\partial L}{\partial W_0}$$

Small learning rate



Large Learning rate



↑ Not perfect with every dataset

2.) Solution of this problem is → Learning rate scheduling

Pre-define ← means before training define

threshold → some range

kis time pe LR ghata ya aur kabb badhaya

3) learning rate same → every direction.

for eg:- 9 weights and bias

↳ graph → 10-D

If one weight want to decrease the value with high speed and another weight want low speed for reach best solution. This statement is true if learning for weight one is different to learning rate of another weight. This is not possible.

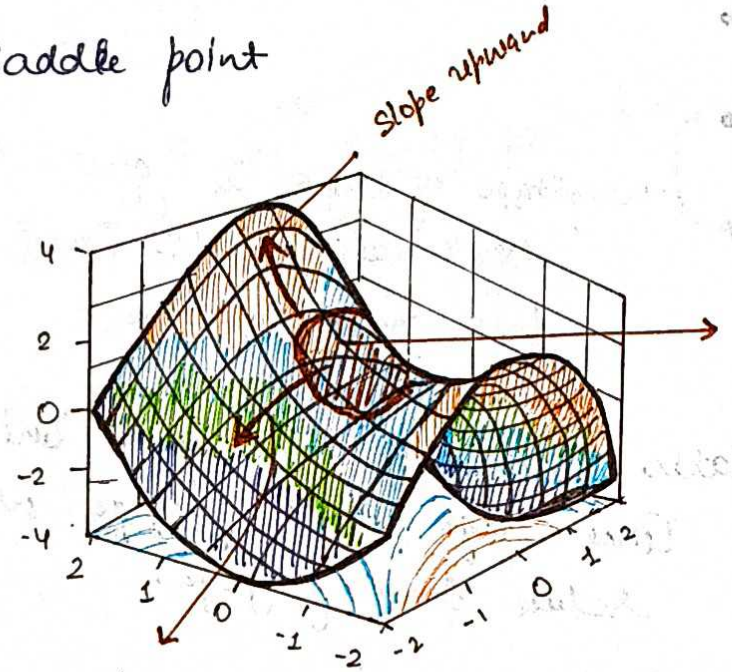
→ Seperate LR

↳ LR Seperate

Cz learning rate is same for every weight.

4. Local minima

5. Saddle point



every direction slop is same (not change in slope)

which mean $\frac{\partial L}{\partial w} = 0$

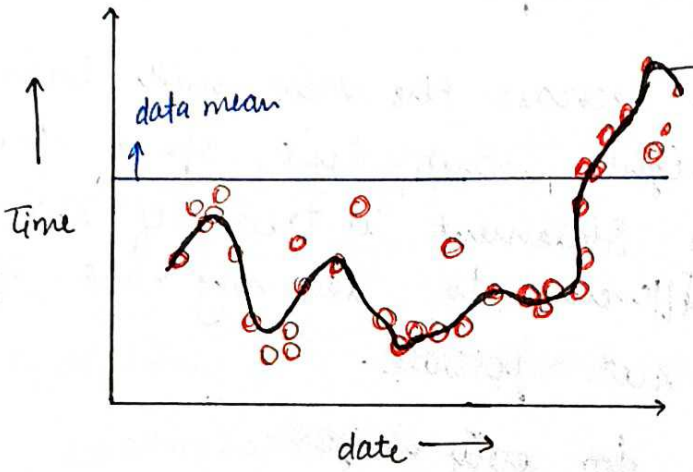
↳ differentiation

$W_n = W_0 - \eta \left[\frac{\partial L}{\partial w}\right] \to 0$

old and New weight is same.

Exponential Weight moving Average (EWMA)

Time Series Data



→ to find hidden trends in data

EWMA → Uses

- → time series
- → financial
- → Signal process
- → Deep learning
- → optimization

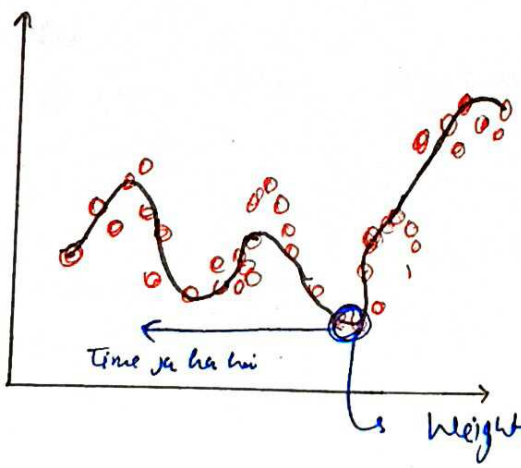
(1)

EWMA → Day 1 → → weightage

Day 2 →
Day 3 → } Day 3 Weightage > Day 2 Weightage

Day 4 →
Day 5 → } Weightage of Day 5 is greater than Day 4 weightage. (Cz Day 5 Weightage baad mai aaya hai)

(2) But time ke sath weightage reduce bhi hota jayega. Time age badhta jayega weightage reduce hota jayega.



Mathematical Formula

$$V_t = \beta V_{t-1} + (1-\beta)\,\theta_t$$

- V_t → EWMA at particular time t.
- V_{t-1} → pervious time of EWMA
- θ_t → time instence pe jo data hai.
- β → $\beta \Rightarrow 0 < \beta < 1$

ex:→

index	temp(θ)
D1	25
D2	13
D3	17
D4	31
D5	43

Every time stamp we will calculate EWMA.

$V_0 = 0$ or $V_0 = \theta_0$ → use this one some text book

let $\beta = 0.9$

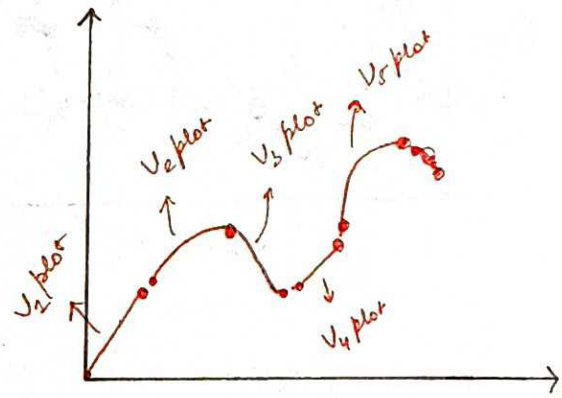
$V_1 = 0.9 \times V_0 + 0.1 \times 13$ (V_0 → 0, 13 → θ_1)

$V_1 = 1.3$

$V_3 = 0.9 \times 1.3 + 0.1 \times 17 =$

⋮

$V_5 = \ldots$



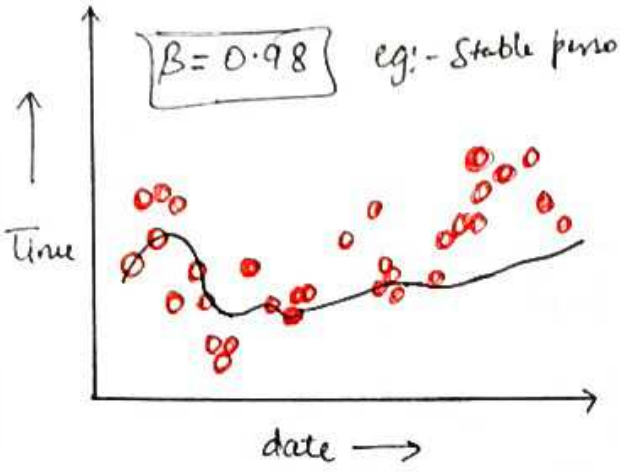
$\beta \Rightarrow \dfrac{1}{1-\beta}$ →

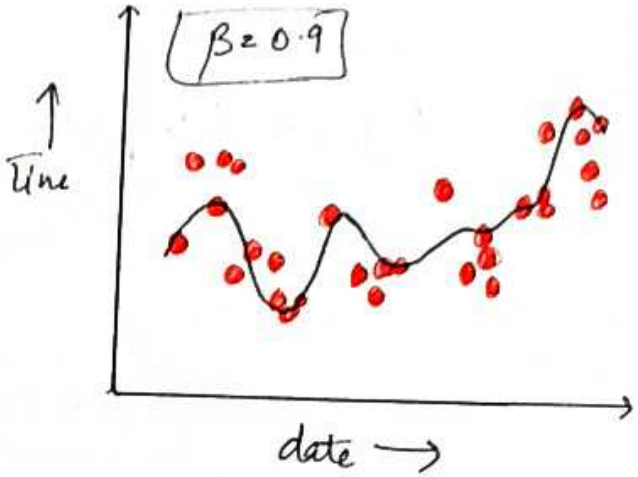
$\beta = 0.9$

$\dfrac{1}{1-0.9} = 10$ → So, β act as the average of last 10 days data

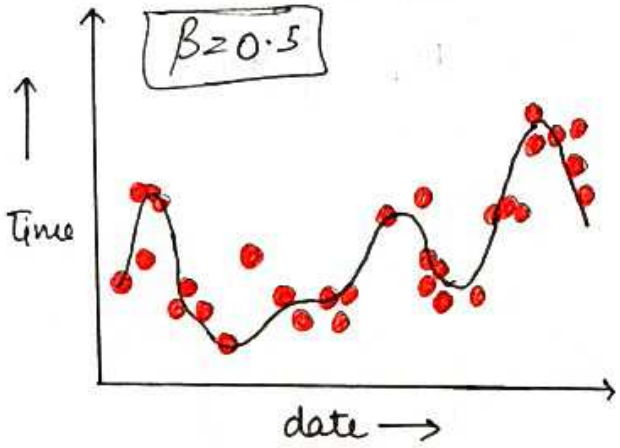
let $\beta = 0.5$

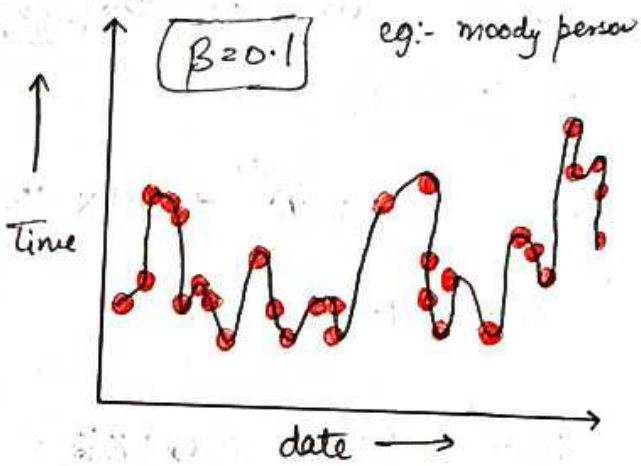
$\dfrac{1}{1-0.5} = 2$ → β act as the avg of last 2 days data





β is very high which means past ke points ko weightage de rahe ho.





When β value is very low and graph is very spiky. Because Moderate Graph on the base of current point.

Moody person → affect on moody on the basis of today's day

Stable person → Person not focus on today. Person focus on past and decide mood.

Sweet spot is → 0.9
Most of ~~the~~ time you see 0.9

Mathematical Intuition

$$V_t = \beta V_{t-1} + (1-\beta)\theta_t$$

$$V_0 = 0$$

$$V_1 = (1-\beta)\theta_1$$

$$V_2 = \beta V_1 + (1-\beta)\theta_2$$

$$= \beta(1-\beta)\theta_1 + (1-\beta)\theta_2$$

$$V_3 = \beta V_2 + (1-\beta)\theta_2 \Rightarrow \beta^2(1-\beta)\theta_1 + \beta(1-\beta)\theta_2 + (1-\beta)\theta_3$$

$$V_4 = \beta V_3 + (1-\beta)\theta_4 \Rightarrow \beta^3(1-\beta)\theta_1 + \beta^2(1-\beta)\theta_2 + \beta(1-\beta)\theta_3 + (1-\beta)\theta_4$$

$$= [1-\beta]\left[\beta^3\theta_1 + \beta^2\theta_2 + \beta\theta_3 + \theta_4\right]$$

$\beta^3\theta_1$: oldest term; $\beta^2\theta_2$: new the θ_1; θ_4: newest term

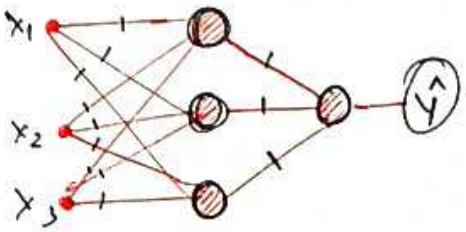
$\beta^3 < \beta^2 < \beta^1$ ← β lies left → $0 < \beta < 1$

point jitna aakota hota ja raha hum aur chhoti value se multiply kar rahe hai.
This means that u (purani value ko kam weightage de rahe aur new value ko jayada weitage)

[code]
[ewMA file]

Understanding Graphs

revise some NN concept ☺



Loss function → $f(x) \rightarrow (y-\hat{y})^2$

$L = f(w_1, w_2, \ldots b)$

↳ Lets say weights are 24

So, Graph → (25D)

but human → max 3-D

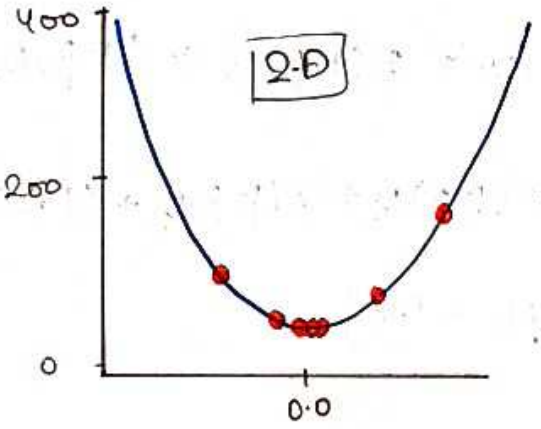
$y = f(x)$

↳ x mai change hone pr y kitna change ho rha.

Let assume, single neuron and then cal.

⇒ $L = f(w)$ → This make 2-D Graph

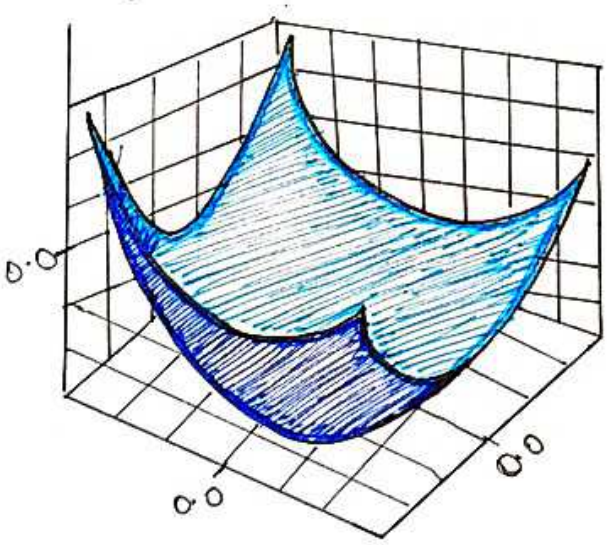
↳ w mai change hone se L mai kitna change ho rha.



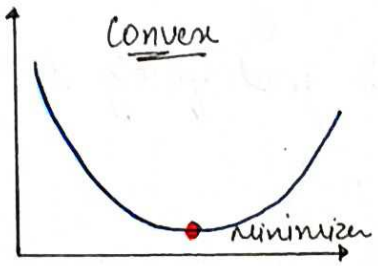
Let assume, single Neuron

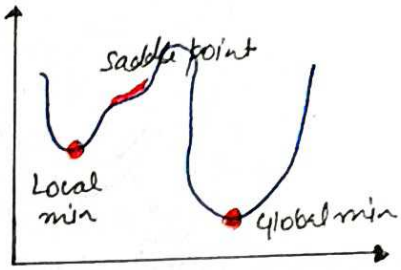
Now we make

3-D Graph



Convex Vs Non-Convex optimization





Non-Convex

↓

1) Local minima

2) Saddle point

is small. So, it take time to reach Global min

3) High curvature.

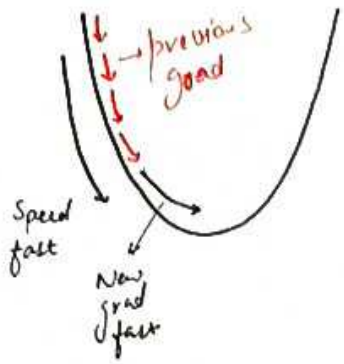
Momentum Optimization — The why?

Non-Convex optimization Problem

- High curvature
- Constant grad (small slope)
- Noisy grad (local min)

→ Momentum can solve all 3 problems

Momentum



→ If previous 4 grad downward hai to new grad fastly downward jayega.

Same in Physics →

Momentum = mv (m: mass, v: velocity)

Momentum Optimization - Mathematics (The How?)

Normal Grad descent algo for update weight →

$$W_{t+1} = W_t - \eta \Delta W_t$$

Momentum →

$V \rightarrow$ velocity

$$W_{t+1} = W_t - V_t$$

where $V_t = \beta * V_{t-1} + \eta \nabla W_t$

history of velocity using for momentum

$0 < \beta < 1$

SGD

SGD Momentum

Effect of beta (β)

$$W_{t+1} = W_t - V_t$$

$$V_t = \beta V_{t-1} + \eta \nabla W_t$$

Let say, $\beta = 0$

$$\boxed{V_t = \eta \nabla W_t}$$

$$\boxed{W_{t+1} = W_t - \eta \nabla W_t}$$

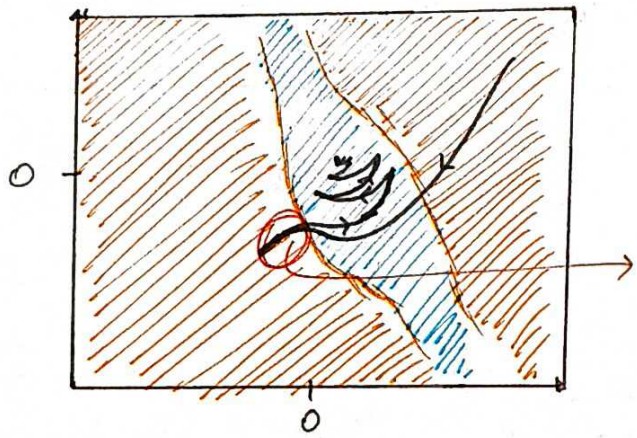
↳ Normal SGD

So, if $\beta = 0$ Then momentum act like Normal SGD.

If $\beta = 0.9 \rightarrow \frac{1}{1-\beta} \rightarrow \frac{1}{1-0.9} = 10$ which means

current velocity = avg of past (10) velocity

Problem with Momentum Optimization



Time waste in this Oscilation That's why momentum is not fastest technique

Nesterov Accelerated Gradient (NAG)

Mathematical Intuition

$$W_{t+1} = W_t - V_t$$

where

$$V_t = \beta V_{t-1} + \eta \nabla W_t$$

β = decay factor

$$W_{t+1} = W_t - (\beta V_{t-1} + \eta \nabla W_t)$$

for new weight depend → past velocity + Gradient at that point

But in Simple GD

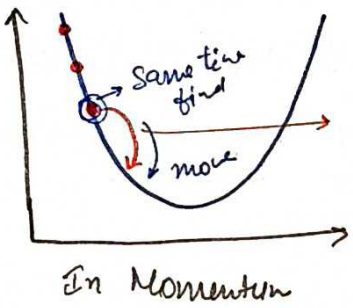
$$W_{t+1} = W_t - \eta \nabla W_t$$

New weight depend → Gradient at that point

Nesterov Accelerated Gradient (NAG)

$$W_{t+1} = W_t - (\beta V_{t-1} + \eta \nabla W_t)$$

ye kis position pe calculate krega (NAG)



In Momentum

Calculate

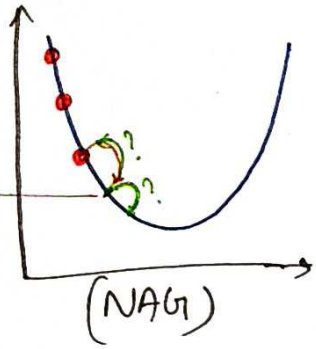
History of velocity + gradient at that point

find both → Same time then move.

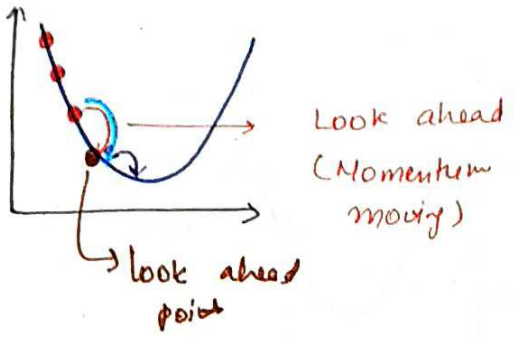
Where the Momentum move from that point, we calculate "Gradient at the point". And then again move.

After Gradient point → decide to move upward? or downward?

At this point we first calculate Momentum and move.



(NAG)



formula → $\boxed{W_{la} = W_t - \beta V_{t-1}}$ - ①

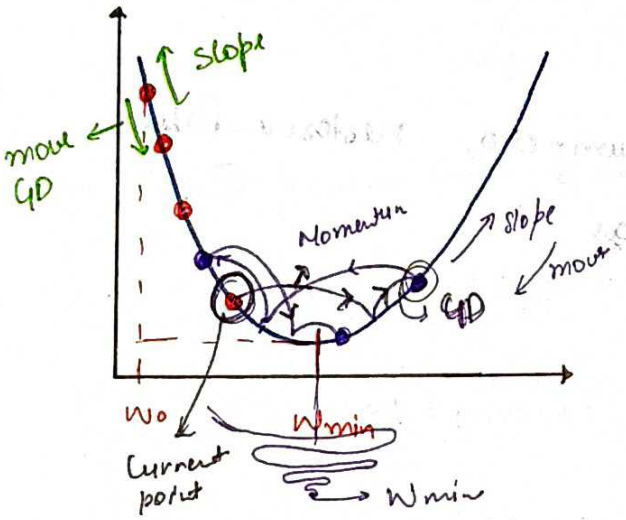
$V_t = \beta V_{t-1} + \eta \nabla W_{la}$ - ②

eq① put in eqn ②

$V_t = (W_t - W_{la}) + \eta \nabla W_{la}$

$W_{t+1} = W_t - V_t$

Geometric Intuition

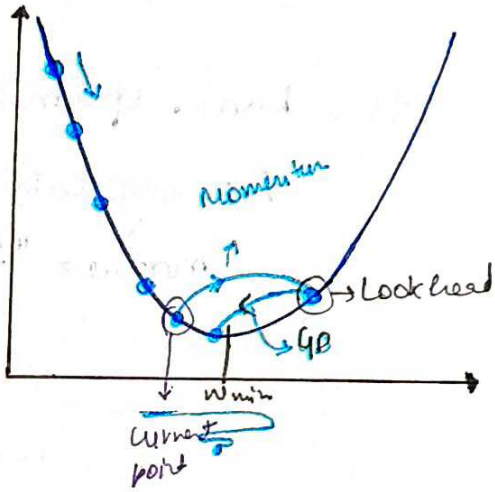


Slope ke opposite gradient descent move kita

first move kiya → Momentum Cause of ~~previous~~ current point

Gradient Descent move ~~from~~ Cause of ~~previous~~ current point not momentum point.

finding Momentum and Gradient descent at a same time.



Here's we find first Moment and move the line then find Gradient descent and again move the line.

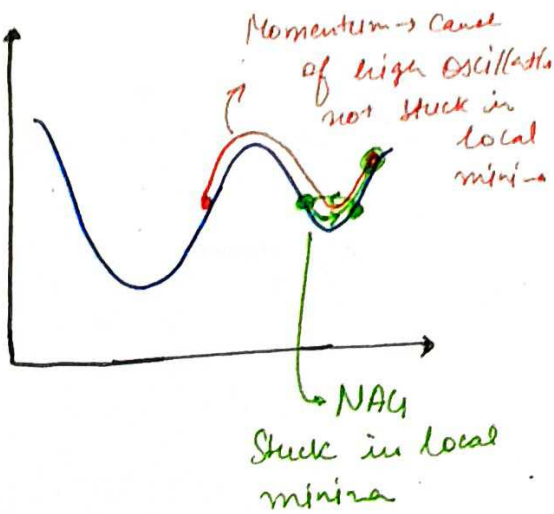
Gradient descent on the basis of look ahead point

but Inn Momentum Gradient descent find on the basis of ~~previous~~ current point

Disadvantage

NAG → Oscillation
↓
dampen

May be stuck in local minima



Keras Code → Momentum

```
tf.keras.optimizers.SGD(
    learning_rate = 0.01, Momentum = 0.0, nesterov = False,
    name = "SGD", **kwargs
)
```

Simple SGD → Momentum = 0 , Nesterov = False

Momentum → Momentum = 0.9 , Nesterov = False

NAG → Momentum = 0.9 , Nesterov = True

AdaGrad (Adaptive Gradient)

AdaGrad → learning rate is not fix

Change or adapt according to the situation.

Where we use:

→ Input features → Scale is different

eg:-

cgpa	Salary	iq
1-10	0-100k	0-500

↑ range → Scales are different

↳ In this Normalize or Scaling can be use.

→ Input features → Sparse → most of values are zero.

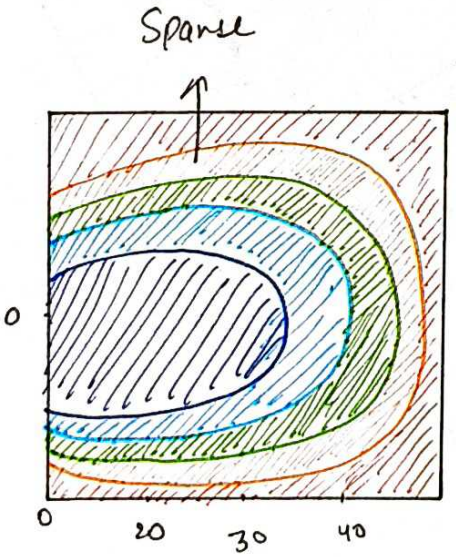
eg:-

iq	cgpa	from iit
:	:	0 / 1

→ package of most of the value is 0.

Problem

Elongated bowl problem

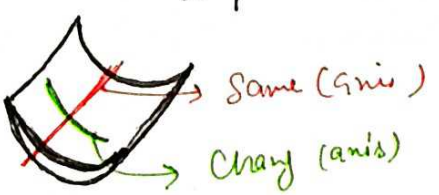


Half contour plot

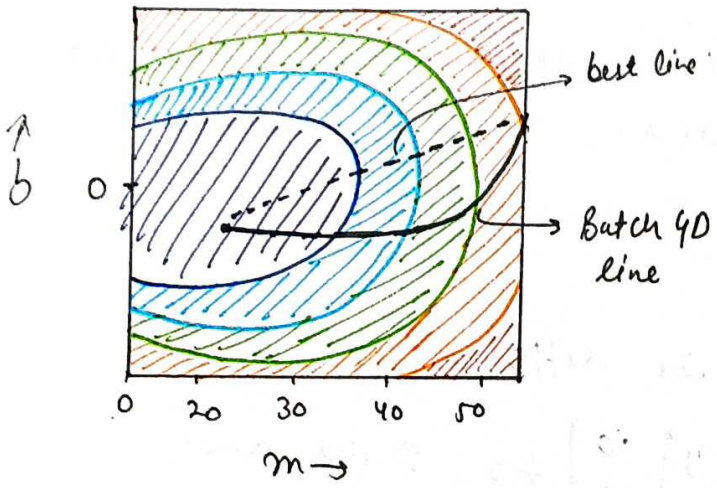
Cause of maximum zeros in data Half contour plot made.

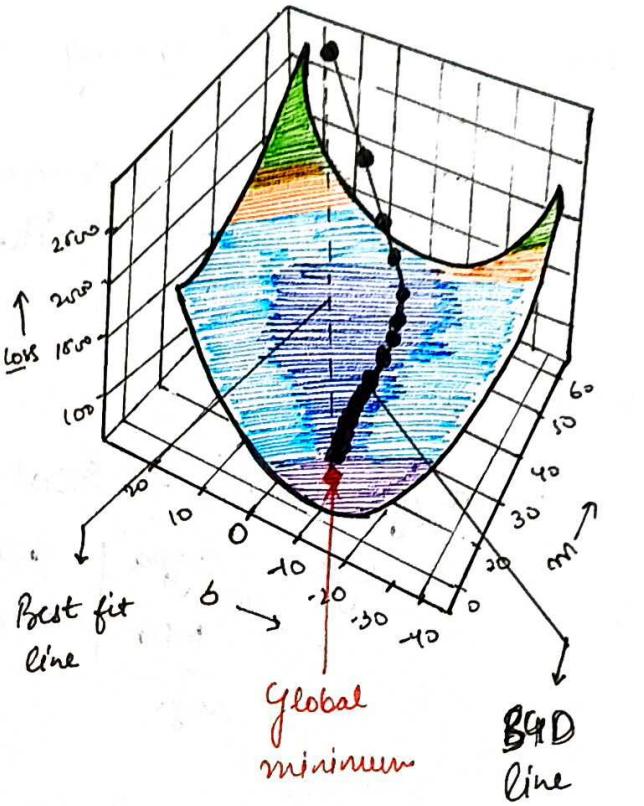
Elongate → slope change along 1-axis

slop same at 2nd axis



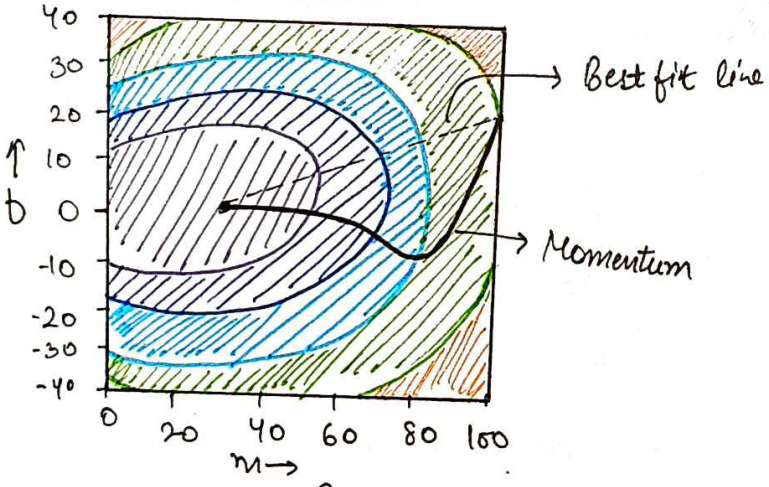
Batch Gradient Descent and Momentum is not good for Sparse data.





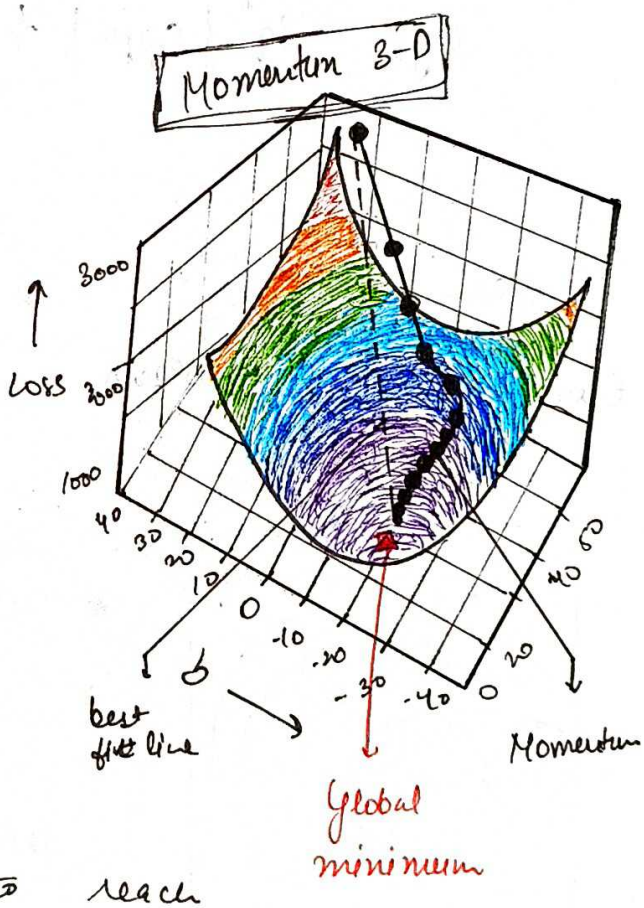
Cause of curve → Batch Gradient Descent take more time to reach Global minima.

Momentum



Contour Plot

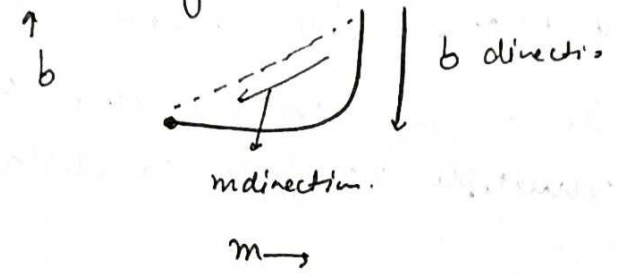
Momentum 3-D



Cause of Overshoot curve ↓

Momentum take more time to reach Global minimum.

Why we moving toward b direction not m?



Because sparse data present in m

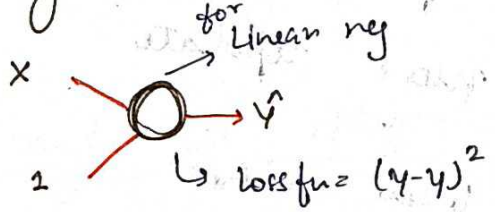
Correlated with b ↓ ← m correlated

Let assume b col (1)	X	Y
1	0.72 (⊗)	-
1	0.81	-
1	0	-
1	0	-
1	0	-
1	0	-
1	0.2	-

input

Sparse value present in ~~x~~ and X correlated with m. That's why line not move to m.

Explaining Why?



BGD → Sparse

Pseudocode?

for i in epochs:

$$w = w - \eta \frac{\partial L}{\partial w}$$

$$b = b - \eta \frac{\partial L}{\partial b}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w}$$

$$\boxed{\frac{\partial L}{\partial w} = -2(y-\hat{y})x}$$

$$\boxed{\frac{\partial L}{\partial b} = -2(y-\hat{y})(1)}$$

(2 input is 1)

Let assume 100 rows

So, we calculate 100 times $\frac{\partial L}{\partial w}$ and add all 100 ~~repeated~~ weights $\frac{\partial L}{\partial w}$ term. We will get big number for $\eta\left(\frac{\partial L}{\partial L}\right)$ this

→ but if lots of 0 value present in x and when we calculat $\frac{\partial L}{\partial w} = -2(y-\hat{y})x$ and whole term will be 0 because x is 0 in data. And there are multiple zero (2 multiple zero in x data. After that we add whole $\frac{\partial L}{\partial w}$ values → Small values cause of zero present in data. Now update also small if $\frac{\partial L}{\partial w}$ is small value. Small update in every epoch

→ In b, $\frac{\partial L}{\partial b} = -2(y-\hat{y})*1$ we will get big values because whole term multiply with 1 not x. So, $\frac{\partial L}{\partial b}$ never be 0. So after addition, value will be bigger and see more update

m →

Adagrad Mathematics + Intuition

$$W = W - \eta \frac{\partial L}{\partial W}$$

$$b = b - \eta \frac{\partial L}{\partial b}$$

→ Check whether W get higher update or b get higher update.

→ According to previous discussion b are getting higher updates. And W getting lower updates.

So maintain the both update → will reduce learning rate of b (higher updates).

low learning rate × higher update = low value (low update)

Though this we can match will W update.

⇒ " for every parameter learning rate is different "

Formula for Adagrad

$$V_t = V_{t-1} + (\nabla W_t)^2$$

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \nabla W_t \quad \left(\nabla W_t \to \frac{\partial L}{\partial W}\right)$$

$$b_{t+1} = b_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \nabla b_t \quad \left(\nabla b_t \to \frac{\partial L}{\partial b}\right)$$

→ if $V_t = 0$ then whole not to be 0.

Normal Grad Descent

$$W_{t+1} = W_t - \eta \nabla W_t$$

Let's discuss about $V_t = V_{t-1} + (\nabla W_t)^2$

V_{t-1} → Past V_t

$(\nabla W_t)^2$ → gradient ka square $\left(\frac{\partial L}{\partial w}\right)^2$ or $\left(\frac{\partial L}{\partial b}\right)^2$

→ Why gradient Square?

Because gradient may be negative or positive.
That's why we squared gradient for negative to positive

→ Why not use mod?

because differentiation nhi hoga mod ka.

Disadvantage (Adagrad → not use in complex NN)

→ use in linear regression

Reason

→ Adagrad
Never reach to global minimum

Global maximum

→ Why Adagrad Never reach global minimum?

Because learning rate divide with past updates

$$\frac{\eta}{\sqrt{V_t}}$$

V_t → past updates

After sometime or many epochs V_t getting bigger and learning rate smaller. So, after divide learning rate become more smaller and no more updates. So never reach to global minimum.

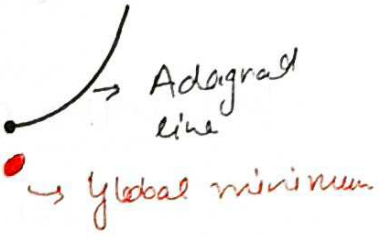
RMSProp

RMSProp → Root Mean Square Propagation

↳ improvement

Adagrad ← Sparse data

Disadvantage → reduce learning rate



Mathematical Formulation

Adagrad

past gradient → gives high value

$$V_t = V_{t-1} + (\nabla W_t)^2$$

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \nabla W_t$$

∇W_t multiply with small so, whole term will be more small and not getting any updates

because of past gradient this term will be bigger.

learning rate divide with bigger term so value is small

RMSProp

Old value ko kam value de rahe ← expo decay avg → Recent value ko jaya value de rahe

$$V_t = \beta V_{t-1} + (1-\beta)(\nabla W_t)^2$$

where $\beta = 0.95$ ↳ Hyperparameter

any time change ←

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \nabla W_t$$

↳ so, V_t will not be much bigger

$V_0 = 0$

$V_1 = 0.95 \times 0 + 0.05 (\nabla W_1)^2$

$V_2 = 0.95 \times 0.05 (\nabla W_1)^2 + 0.05 (\nabla W_2)^2$

$V_3 = 0.95 \times 0.95 \times 0.5 (\nabla W_1)^2 + 0.95 \times 0.05 (\nabla W_2)^2 + 0.5 (\nabla W_3)^2$

Largest

Smallest ↑ epoch 1 ↓ old ↑ epoch 2 ↑ epoch 3 recent

→ RMSProp → V_t shoot → X

η → very small X

recent → accordly move → updates acc relatively

global minimum → reach

Adagrad → good with → convex / optimization problem ad regression

→ bad with → non-convex optimization and complex NN

In RMSProp → No disadvantage

ADAM Optimizer

Adam → Adaptive Moment Estimation

↳ Most powerful technique → ANN, CNN, RNN

SGD / BGD / MBGD

→ Momentum → NAG → Momentum
→ Adagrad } → learning rate decay
→ RMSProp }

Momentum, Learning rate decay → Merge (Adam)

Mathematical formulation

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \times m_t$$

Where,

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) \nabla W_t$$ → Momentum

$$V_t = \beta_2 V_{t-1} + (1-\beta_2) (\nabla W_t)^2$$ → Adagrad

Bias Correction → find bias correction after m_t and V_t

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

t → different from epoch Number

$\beta_1 = 0.9$

$$\hat{V}_t = \frac{V_t}{1-\beta_2^t}$$

t → epoch Number

$\beta_2 = 0.99$ according to keras

Why bias correction?

→ Because during starting → m_t start from 0 and V_t also start from 0. So, biased can be in the m_t and V_t. That's why we find bias correction.