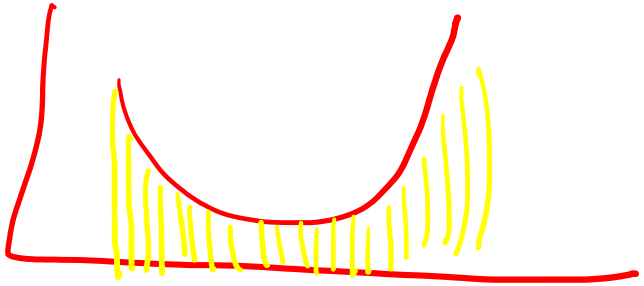
Optimization Technique:

(1) Exhaustive Search | Greedy Search | Grid Search

Time Consuming but Guarantees

eg: Man walking on a hill



Select best of all values

(2) Gradient Descent: (3 types)

Fast → Sometimes we are lost

eg: F1 car

Gradient/Slope = 0

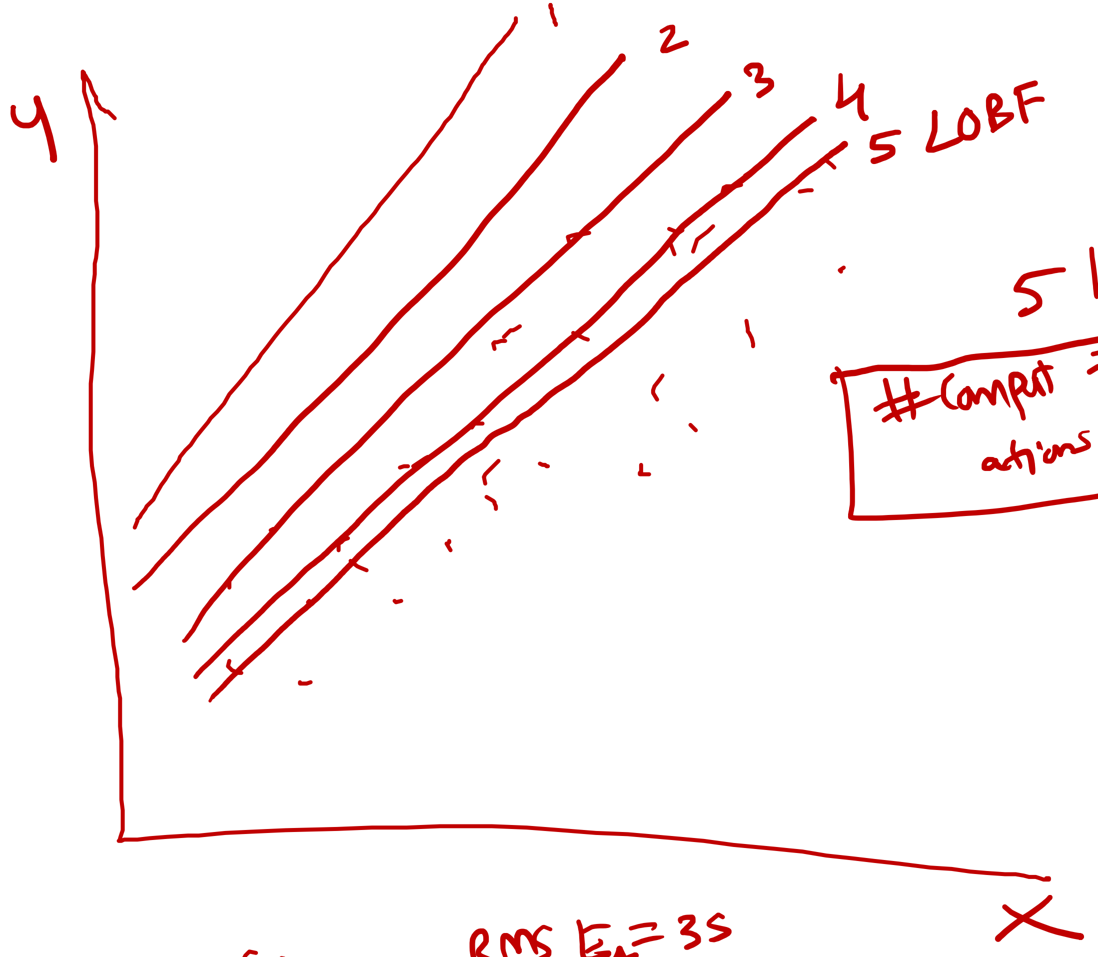
(3) Robust mtds (Simulated Annealing, Genetic Algo, Evolutionary Algo)

eg: Dirt Truck (faster than Greedy algo, but slower than G-D algo)

Types of Gradient Descent:

1. Batch Gradient Descent
2. Mini-batch Gr. Descent
3. Stochastic Gr. Descent

Batch Gradient Descent: Computes error wrt all M points (all rows) of the dataset



$RMSE_1 = 60$
$2 = 50$
$3 = 40$

$RMSE_4 = 35$
$RMSE_5 = 33$

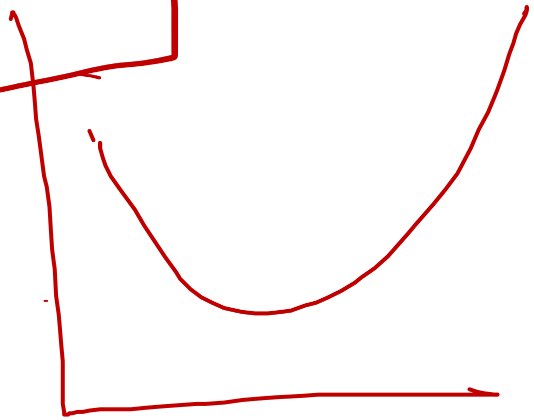
5 lines × 100

Computations = 500

X	y

100 × 2

$\boxed{M = 100}$



TV(y)	Pred($\bar{y}$)

100

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(y-\bar{y})^2}$$

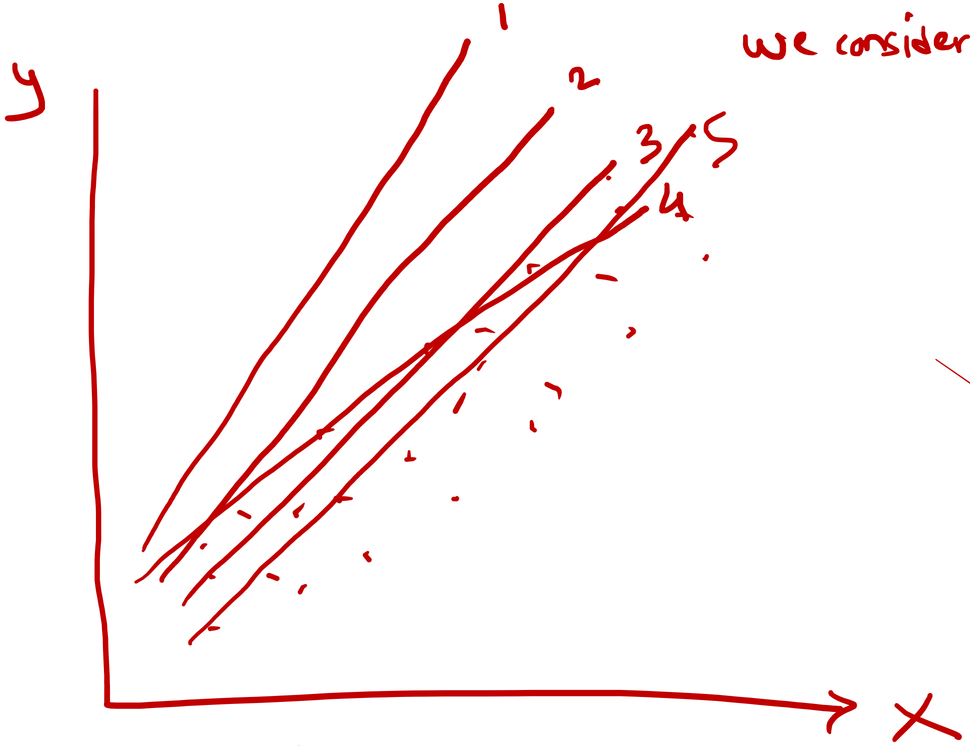
Mini_Batch GD: Divide the data into batches

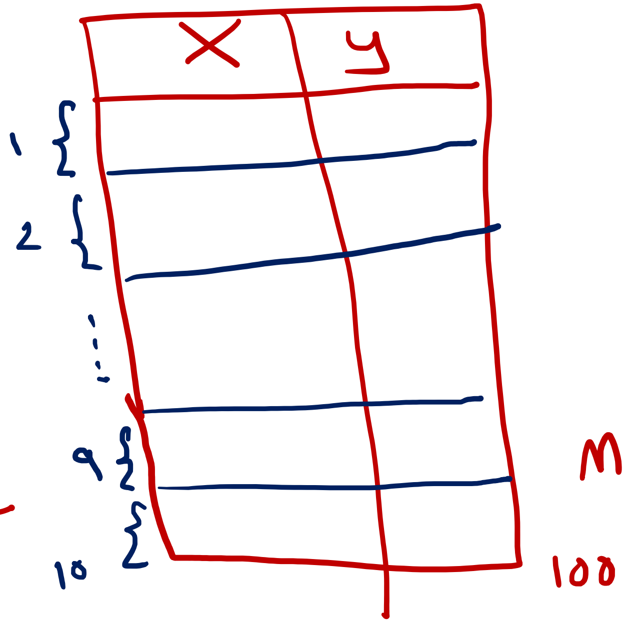
N=10 Groups

we consider only N records for calculating the error

N means ONE BATCH SIZE. Here it is 10

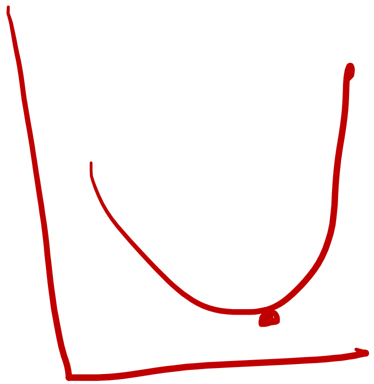
$5\times10 = 50$ computation





M=100

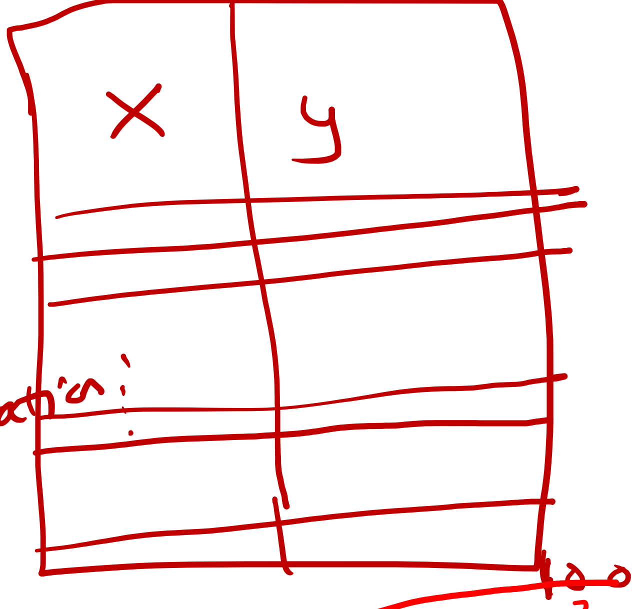
RMSE=



RMSE value/error:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=}^{N}(y-\hat{y}_{pred})^2}$$

Stochastic Gr. Descent: One record/one row for calculating error



5 × One = 5 computation

8 × 1 = 8 comp

$$RMSE = \sqrt{(y-\bar{y})^2}$$

RMSE = 70
= 65
= 60
= 58
= 55
= 50

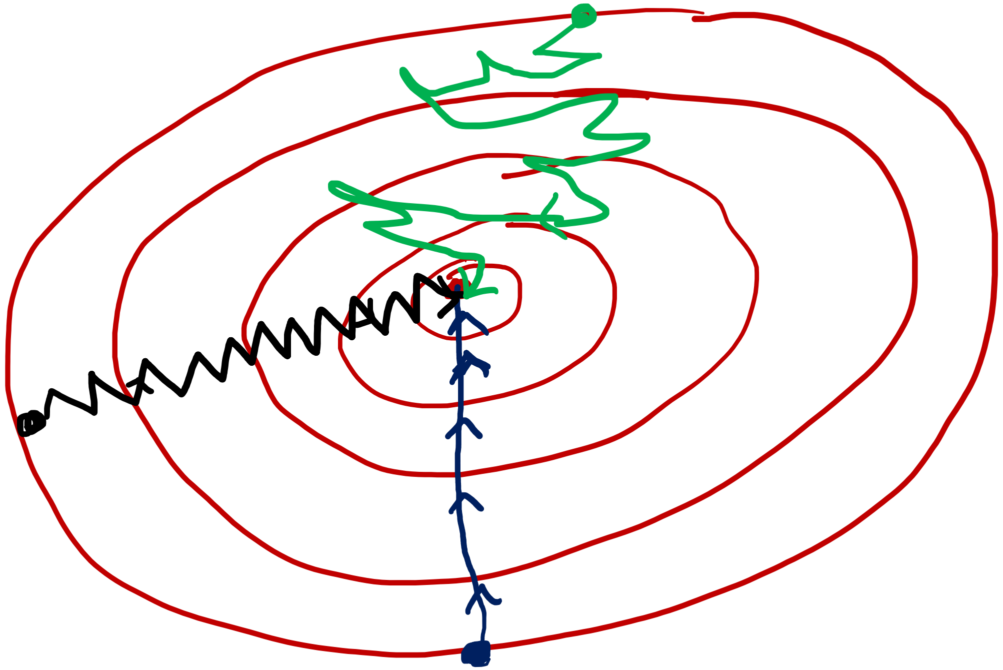
bcz we hv only one record

= ~~5~~ 2

= 48 comparison

Neural Network Learning Parameters

Compare:



① Batch GD (Sober person)

② Mini Batch G.D (Drunk person)

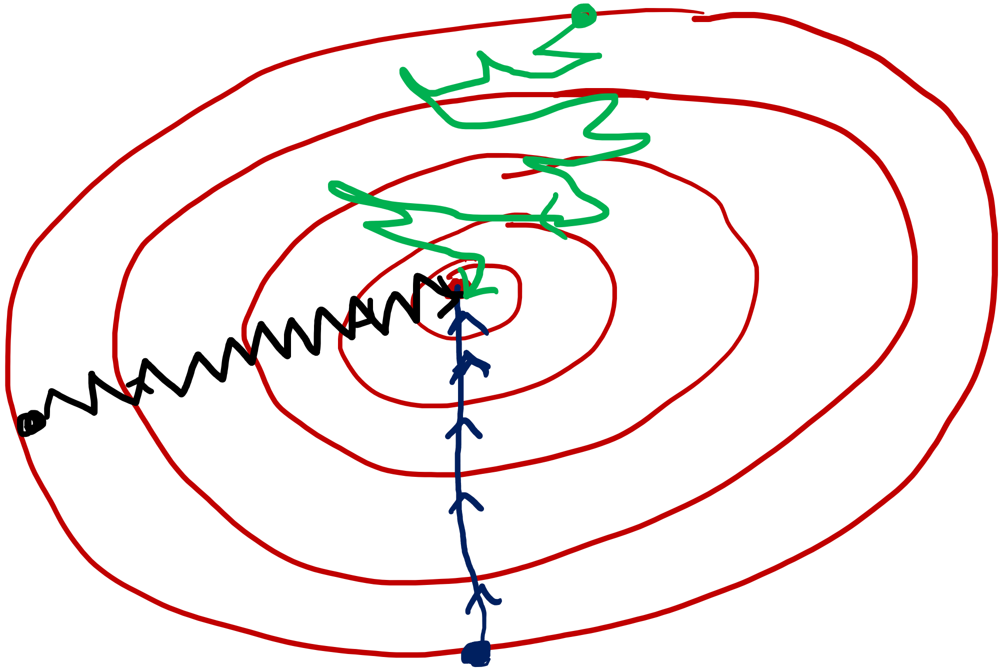
③ Stochastic GD (SGD) (Vested Person)

Researchers say SGD is the best.

SGD takes many lines for plotting to reach the LOBF; but #computations are tremendously less & it is still guaranteed to reach SOLUTION.

Neural Network Learning Parameters

Compare:



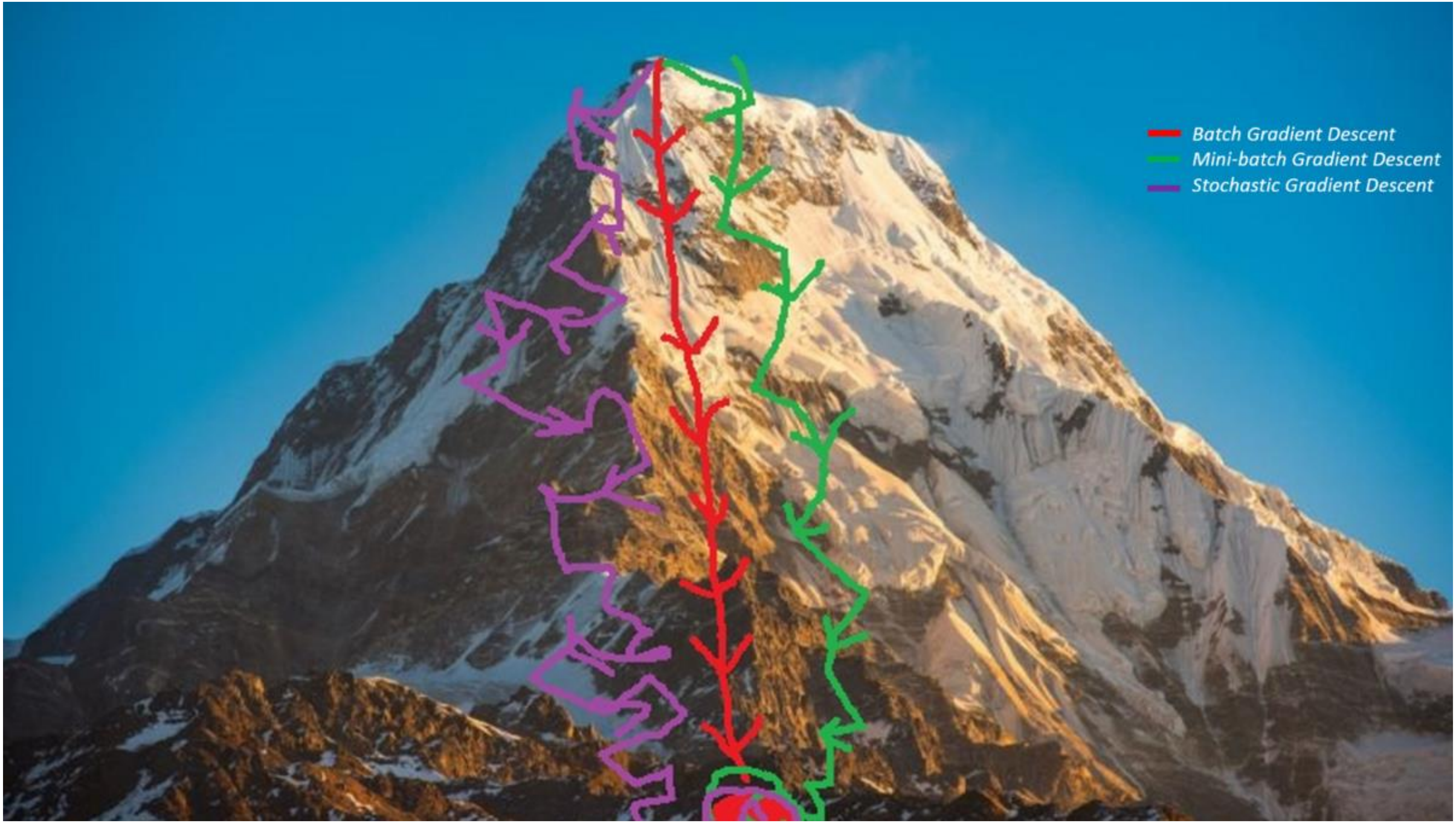
① Batch GD (Sober person)

② Mini Batch G.D (Drunk person)

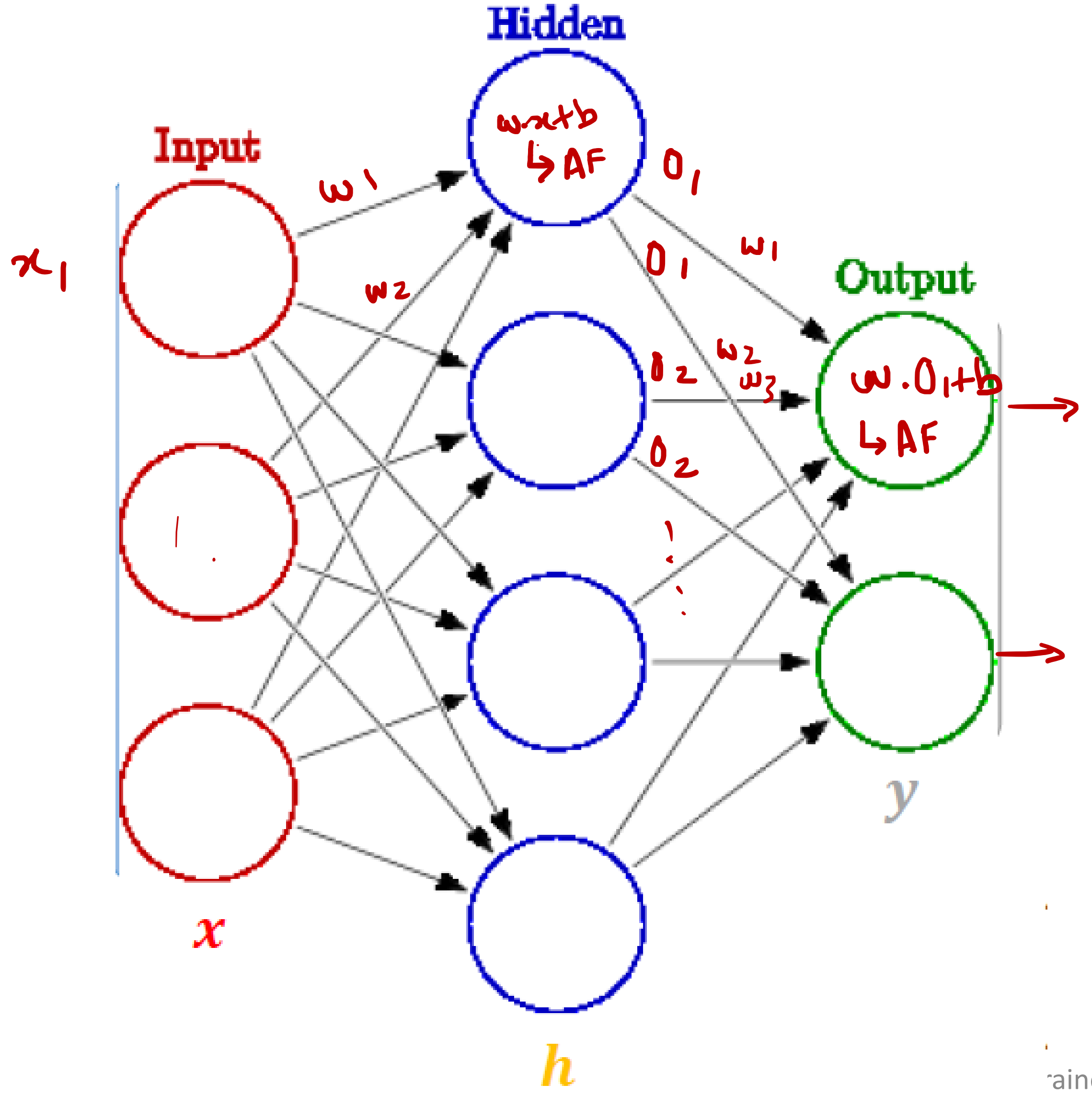
③ Stochastic GD (SGD) (Vested Person)

Researchers say SGD is the best.

SGD takes many lines for plotting to reach the LOBF; but #computations are tremendously less & it is still guaranteed to reach SOLUTION.


Batch Gradient Descent
Mini-batch Gradient Descent
Stochastic Gradient Descent

Neural Network Learning Parameters



weights

$h = \sigma(w \cdot x + b)$

$y = \sigma(O_1 \cdot w + b)$

Activation functions

How do we train?

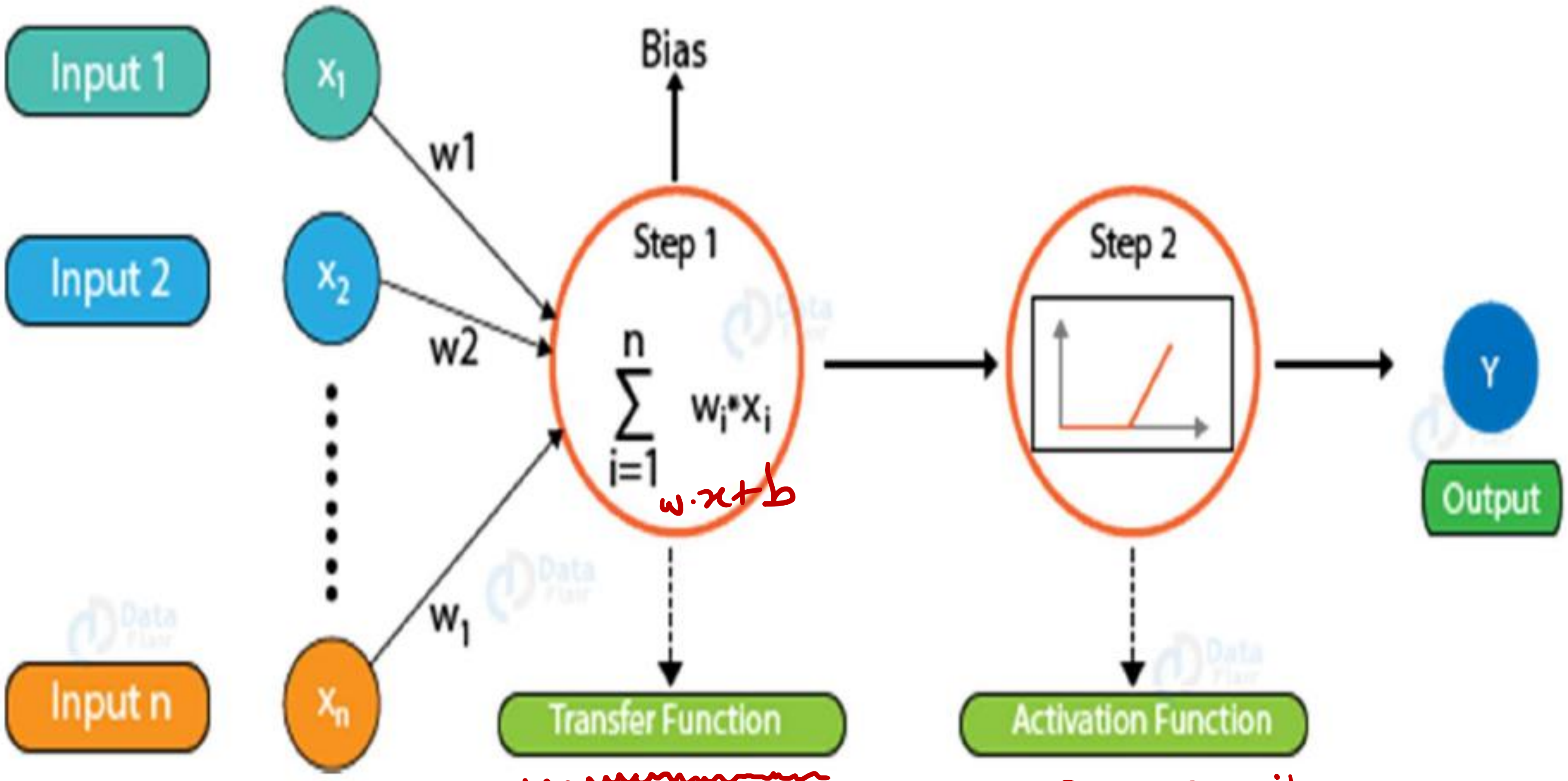
$4 + 2 = 6$ Neurons (not counting inputs)

$(3\times 4) + (4\times 2) = 12+8 = 20$ weights

$4+2 = 6$ biases (bcz I hv 6 Neurons)

$\therefore$ Learnable Parameters $= 20+6$
$= \underline{\underline{26}}$

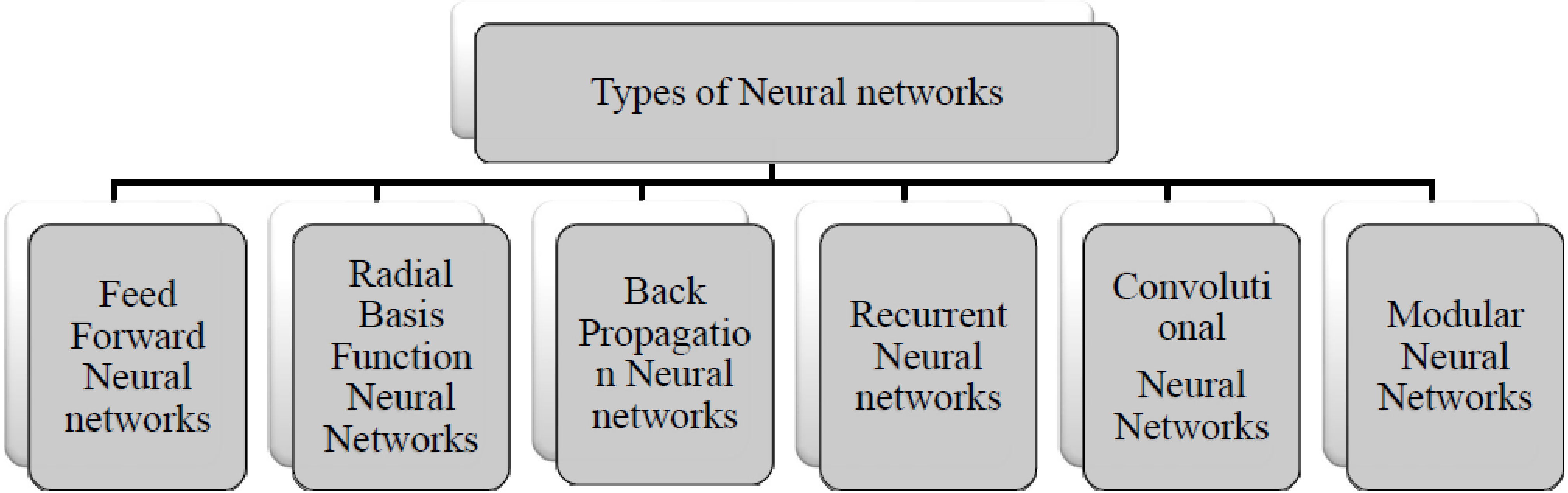
More Terminologies of a NN



w·x+b

ReLU, Sigmoid, ...

Types of Neural Network



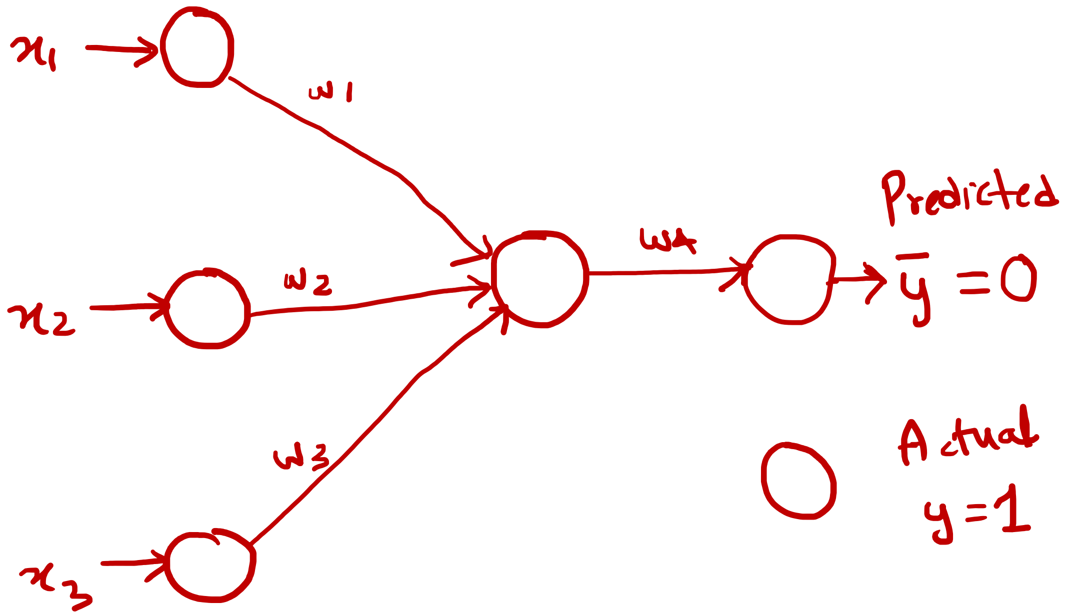
In real life, weights are initialized randomly.

Forward and Backward Propagation

x_1 Res. Score	x_2 Proj. Score	x_3 Grade Score	o/p Grant (Yes/No)
80	70	81	1

Forward Propagation →

$\boxed{EPOCH = 1FP + 1BP}$



$$z = [w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3] + b$$

$$y = \sigma(z) \quad \text{Sigmoid}$$

$\boxed{\text{Minimize Loss} \Rightarrow \text{Optimizers}}$

$SS \propto Slope$

$SS = \eta \cdot Slope$

$\boxed{\text{Assume } \eta = 0.001}$

$$Loss = (y - \bar{y})^2$$
$$= (1 - 0)^2$$
$$= 1$$

← Back Propagation

Formula:

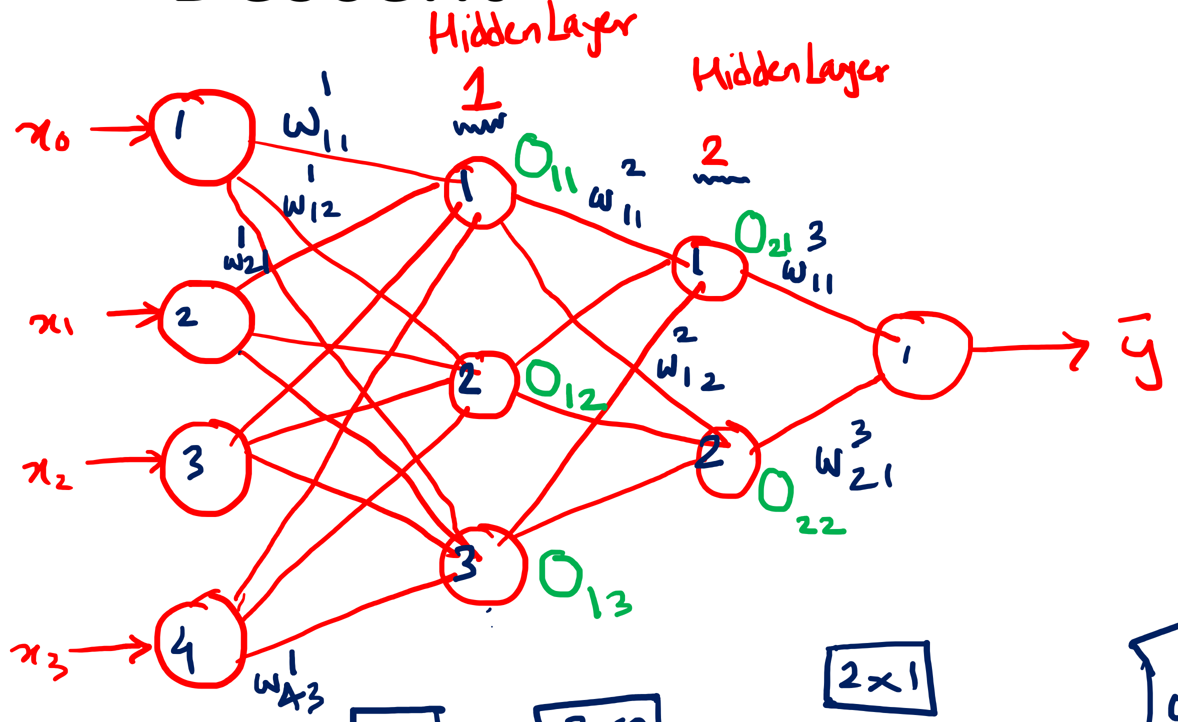
$$\boxed{w_{4_{new}} = w_{4_{old}} - \eta \cdot \frac{dL}{dw_{4_{old}}}}$$

$$w_{3_{new}} = w_{3_{old}} - \eta \cdot \frac{\delta L}{\partial w_{3_{old}}}$$

Note: ① Single o/p Node = Loss Function $\equiv (y - \bar{y})^2$

② Multi-class $\equiv \sum_{i=1}^{n} (y - \bar{y})^2 \equiv$ Cost Function

Multi Layer Neural Network Training w.r.t Gradient Descent



$\boxed{4\times3}$ $\boxed{3\times2}$ $\boxed{2\times1}$

$$\begin{bmatrix} w^1_{11} & w^1_{12} & w^1_{13} \\ w^1_{21} & w^1_{22} & w^1_{23} \\ w^1_{31} & w^1_{32} & w^1_{33} \\ w^1_{41} & w^1_{42} & w^1_{43} \end{bmatrix}_{4\times3} \qquad \begin{bmatrix} w^2_{11} & w^2_{12} \\ w^2_{21} & w^2_{22} \\ w^2_{31} & w^2_{32} \end{bmatrix}$$

Predicted

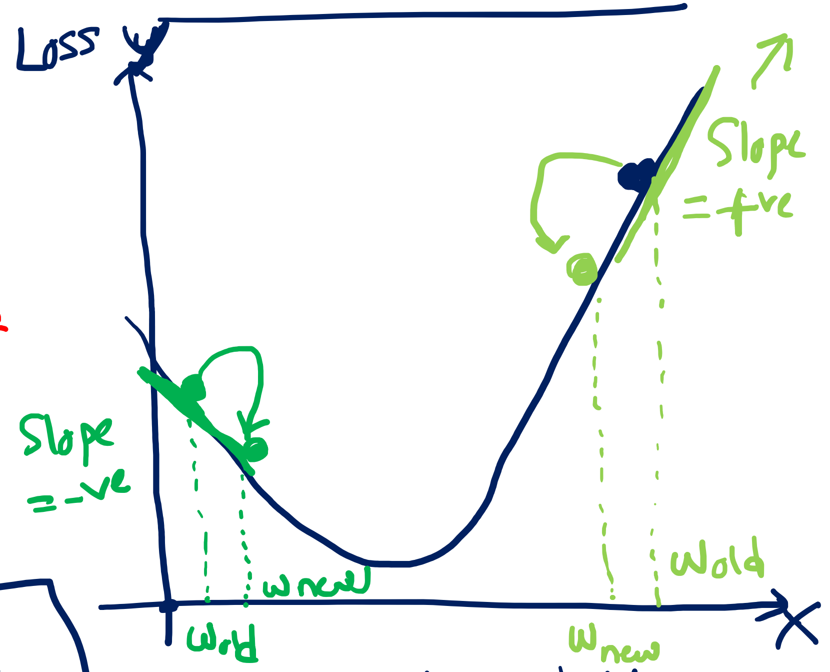
$$\text{Loss} = (y-\bar{y})^2$$

(min)

$$\eta = 0.001$$

$$\boxed{w_{new} = w_{old} - \eta \cdot \frac{\partial L}{\partial w_{old}}}$$

Gradient Descent:



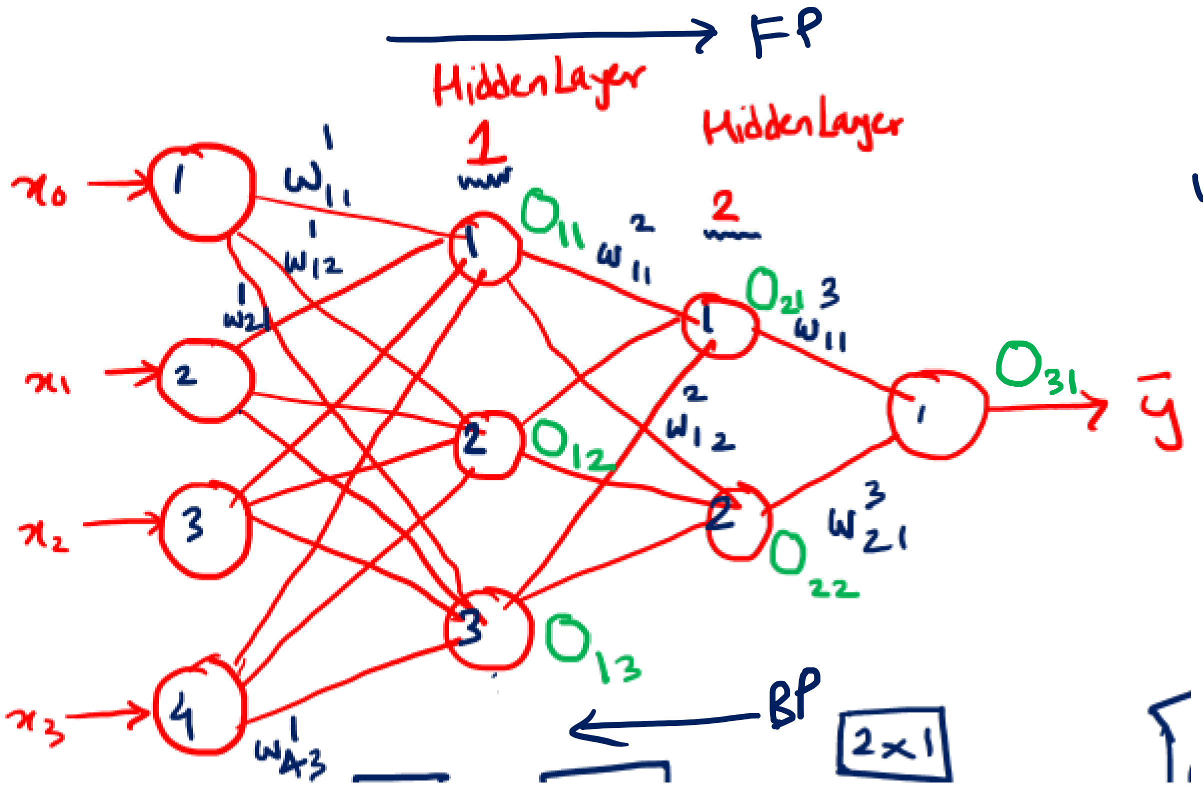
Weight Updation

$$w_{new} = w_{old} - \eta \cdot (+ve)$$
$$w_{new} = w_{old} - (+ve)$$
$$\boxed{w_{new} < w_{old}}$$

$$w_{new} = w_{old} - \eta \cdot (-ve)$$
$$w_{new} = w_{old} + (+ve)$$
$$\boxed{w_{new} > w_{old}}$$

CHAIN RULE IN BACKPROPAGATION

$$\frac{\partial L}{\partial w^3_{11\,old}} = \frac{\partial L}{\partial O_{31}} \cdot \frac{\partial O_{31}}{\partial w^3_{11}}$$



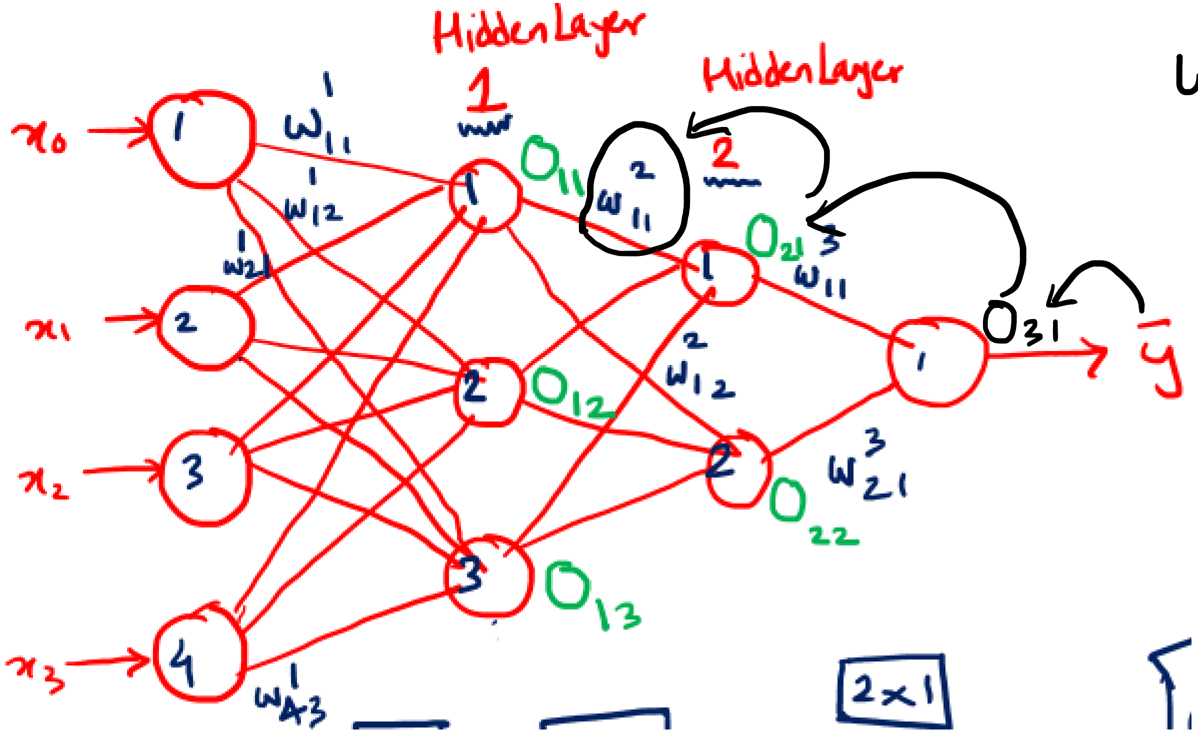
$$w^3_{11\,new} = w^3_{11\,old} - \eta \cdot \frac{\partial L}{\partial w^3_{11\,old}} \leftarrow \text{slope}$$

$\therefore$ we get w^3_{11} new (Updated weight)

|||ly, we can calculate $\boxed{w^3_{21}\text{ new.}}$

How to calculate w^2_{11}? $\Rightarrow$ Using Chain Rule of Derivatives (XI, XII Std)

CHAIN RULE IN BACKPROPAGATION



Slope

$$w^2_{11_{new}} = w^2_{11_{old}} - \eta \cdot \frac{\partial L}{\partial w^2_{11_{old}}}$$

$$\frac{\partial L}{\partial w^2_{11_{old}}} = \left[\frac{\partial L}{\partial O_{31}} \cdot \frac{\partial O_{31}}{\partial O_{21}} \cdot \frac{\partial O_{21}}{\partial w^2_{11}}\right]$$

$$+ \left[\frac{\partial L}{\partial O_{31}} \cdot \frac{\partial O_{31}}{\partial O_{22}} \cdot \frac{\partial O_{22}}{\partial w^2_{12}}\right]$$