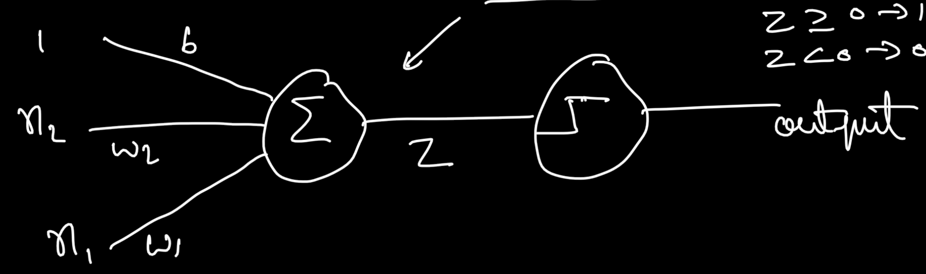
Perceptron loss function | Hinge Loss | Binary cross Entropy | Sigmoid Function

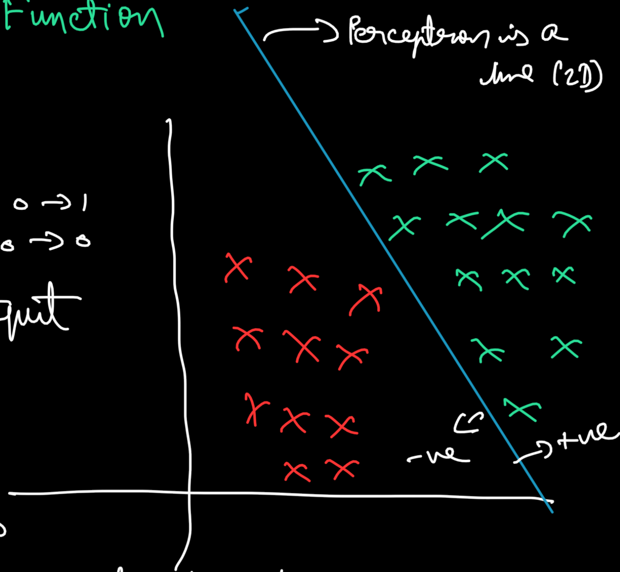
Recap:

$$Z = w_1x_1 + w_2x_2 + b$$

$z \geq 0 \rightarrow 1$

$z < 0 \rightarrow 0$



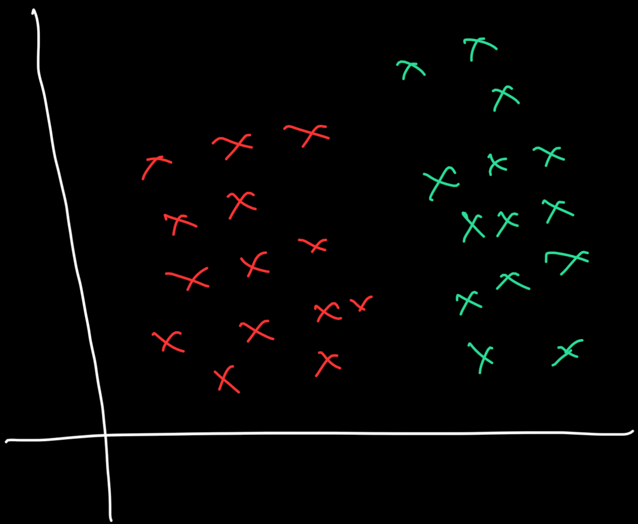


x_1 cgpa	x_2 iq	Placed
7	78	1
6	61	0
9	92	1

Perceptron is

1) Mathematical Model based on neuron

Problem with Perceptron Trick



* with perceptron trick whatever value we get for w_1, w_2 & b. we will not be very confident that these values are best to classify red & green points.

* you can not quantify your result with perceptron trick. (~~कितना अच्छा है~~)

* Since perceptron trick chooses Random point, it may possible that it will not converge. [Every time picks Same point or those point which are correctly classified.]

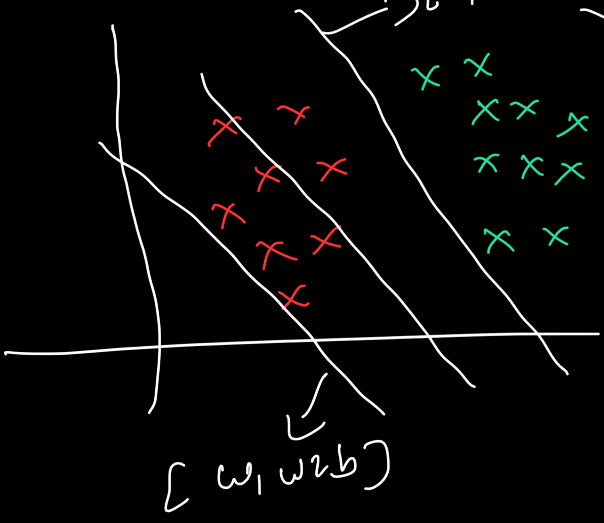
* Perceptron is a Jugad.

* 1% → It will not converge

* 99% → It will not help in quantifying result. (I got the line but ~~कितना अच्छा है~~)

Loss Function

* loss f^n is a method to tell कोई Machine Learning algorithm आपके problem पे कितना अच्छा काम कर रहा है।



Loss f^n

$\boxed{f(w_1, w_2, b)} \rightarrow$ minimum $\rightarrow [w_1, w_2, b] \rightarrow$ final line

25 ← Error

23 ← Error

$23 < 25$

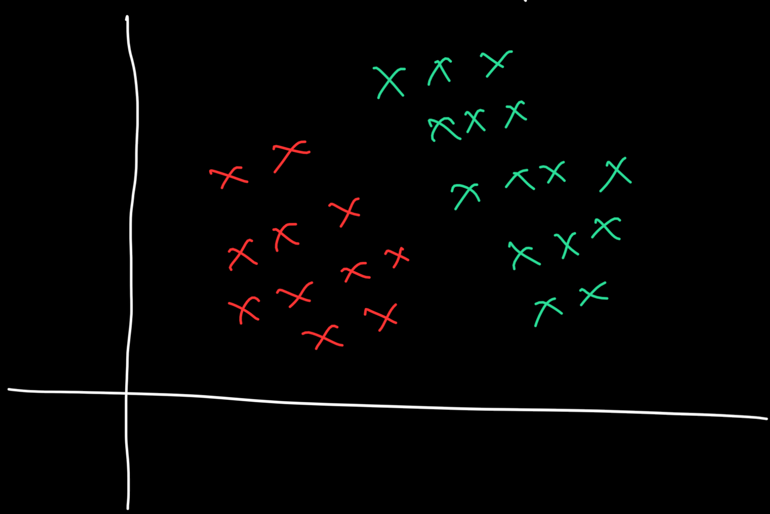
for ex:

Linear Reg → MSE $(y - \hat{y})^2$

logistic Reg → log loss

SVM → Hinge Loss

Perceptron Loss Function



$f(w_1, w_2, b) \rightarrow$ number → Error

Loss function Could be

- Count misclassified point
- Sum of distance between misclassified point & line (⊥ distance).

Here we are giving equal weightage to each point, does not care how far it is from the line.

The value we get will directly proportional to distance [Means for far point you get big value and vice versa]

It is complex operation to calculate distance rather than this we can put coordinate of that point in line eq^n and

According to Scikit learn

$$L(w_1, w_2, b) = \underbrace{\boxed{\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, f(x_i))}}_{\text{Loss function}} + \overbrace{\boxed{\alpha R(w_1, w_2)}}^{\text{Regularization}}$$

for Perceptron

$$L(y_i, f(x_i)) = \boxed{\max(0, -y_i f(x_i))}$$

$$f(x_i) = \boxed{w_1 x_1 + w_2 x_2 + b}$$

$$\boxed{L = \frac{1}{n}\sum_{i=1}^{n} \max(0, -y_i f(x_i))}$$

{ Our task is to find such value of w_1, w_2 & b so that Loss will be minimum. Because Loss depends upon w_1, w_2 & b only rest all values are constant. }

$$L = \underset{w_1, w_2, b}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} \max(0, -y_i f(x_i))$$

It is using Gradient descent.

minimum

Explanation of Loss Function

$$L = \frac{1}{n}\sum_{i=1}^{n} \max(0, -y_i f(x_i))$$

where $f(x_i) = w_1 x_{i1} + w_2 x_{i2} + b$

rows	X_1	X_2	Y
1	X_{11}	X_{12}	Y_1
2	x_{21}	X_{22}	Y_2
⋮			
i			
⋮			
n			

Breakdown loss f^n

$$\boxed{\max(0, -y_i f(x_i))}$$

if $-y_i f(x_i) = \theta > 0$ then $\max(0, \theta) = \theta$

if $-y_i f(x_i) = \theta < 0$ then $\max(0, \theta) = 0$

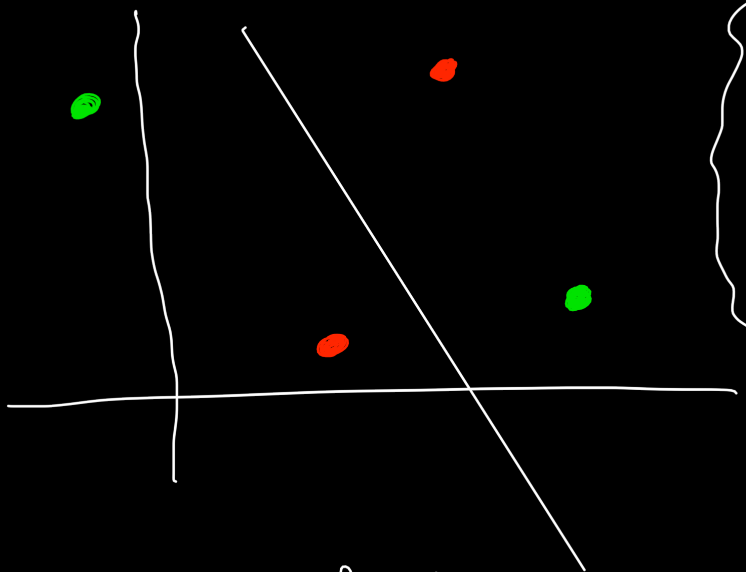
2 Points

$$L = \frac{1}{2}\left[\max(0, -y_1 f(x_1)) + \max(0, -y_2 f(x_2))\right]$$

$$f(x_1) = w_1 x_{11} + w_2 x_{12} + b$$

$$f(x_2) = w_1 x_{21} + w_2 x_{22} + b$$

Geometrical Intuition



cgpa	iq	Placed
8	81	1
6	70	-1
5	91	1
6	55	-1

With context of line, we can have four scenario:

	y_i	$\hat{y}_i$
1 $\longrightarrow$	1	1
2 $\longrightarrow$	-1	-1
3 $\longrightarrow$	1	-1
4 $\longrightarrow$	-1	1

Taking 1st

$\max(0, -y_i f(x_i))$

$y_i = 1$ (+ve)

$f(x_i) = w_1 x_1 + w_2 x_2 + b \geq 0$

$= +ve$

So $-y_i f(x_i) = -(+,+) = -ve$

$\max(0, -ve) = 0$

It means for point which is correctly classified, $\max(0, -y_i f(x_i))$ will be zero. Means for this point contribution in loss f^n would be zero.

Taking 2nd

$\max(0, -y_i f(x_i))$

$y_i = -1$ (-ve)

$f(x_i) = w_1 x_1 + w_2 x_2 + b \leq 0$
$= -ve$

So $-y_i f(x_i) = -(-,-) = -ve$

$\max(0, -ve) = 0$

Taking 4th

$\max(0, -y_i f(x_i))$

$y_i = +1$ (+ve)

$f(x_i) = -ve$

So $-y_i f(x_i) = +ve$

$\max(0, +\text{ve number})$
$= +\text{ve number}$

* for missclassified Point, $\max(0, -y_i f(x_i))$ will be some non-zero value. Means It will contribute in loss.

[So finally we come to know that here we are doing nothing but for $f(x_i)$ we are putting the coordinate of points in the eq^n of line to get a value which basically some how equal to or correspond to $\perp$ distance of that point from the line; As discussed above in Page No 2.]

Now we see, how we can find such value of w_1, w_2 & b so that Loss will be zero.

$\Downarrow$

Gradient descent

$$L = \underset{w_1 w_2 b}{\operatorname{argmin}} \boxed{\frac{1}{n} \sum_{i=1}^{n} \max(0, -y_i f(x_i))}$$

where $f(x_i) = w_1 x_{i1} + w_2 x_{i2} + b$

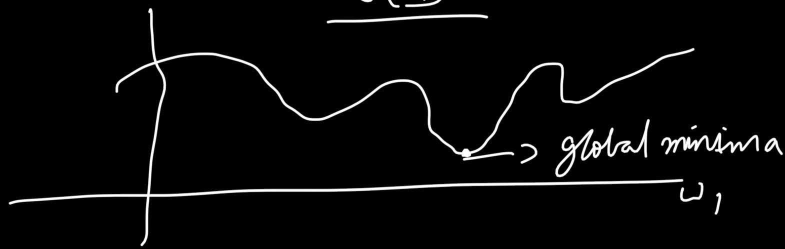
$L(w_1\ w_2\ b)$

$\downarrow\ \downarrow$

Constant

$L(w_1)$

GD



$$\text{Epochs} = 1000, \quad \eta = 0.01$$

for i in range (epochs):

$$w_1 = w_1 + \eta \frac{\partial L}{\partial w_1}$$

$$w_2 = w_2 + \eta \frac{\partial L}{\partial w_2}$$

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

$$\left[\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2}, \frac{\partial L}{b}\right]$$

Loss Function Differentiation

$$L = \frac{1}{n} \sum_{i=1}^{n} \max(0, -y_i f(x_i))$$

where $f(x_i) = w_1 x_{i1} + w_2 x_{i2} + b$

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial f(x_i)} \times \frac{\partial f(x_i)}{\partial w_1}$$

$$\frac{\partial L}{\partial f(x_i)} = \begin{cases} 0 & \text{if } y_i f(x_i) \geq 0 \\ -y_i & \text{if } y_i f(x_i) < 0 \end{cases}$$

$$\frac{\partial f(x_i)}{\partial w_1} = x_{i1}$$

$$\frac{\partial L}{\partial w_1} = \begin{cases} 0 & \text{if } y_i f(x_i) \geq 0 \\ -y_i x_{i1} & \text{if } y_i f(x_i) < 0 \end{cases}$$

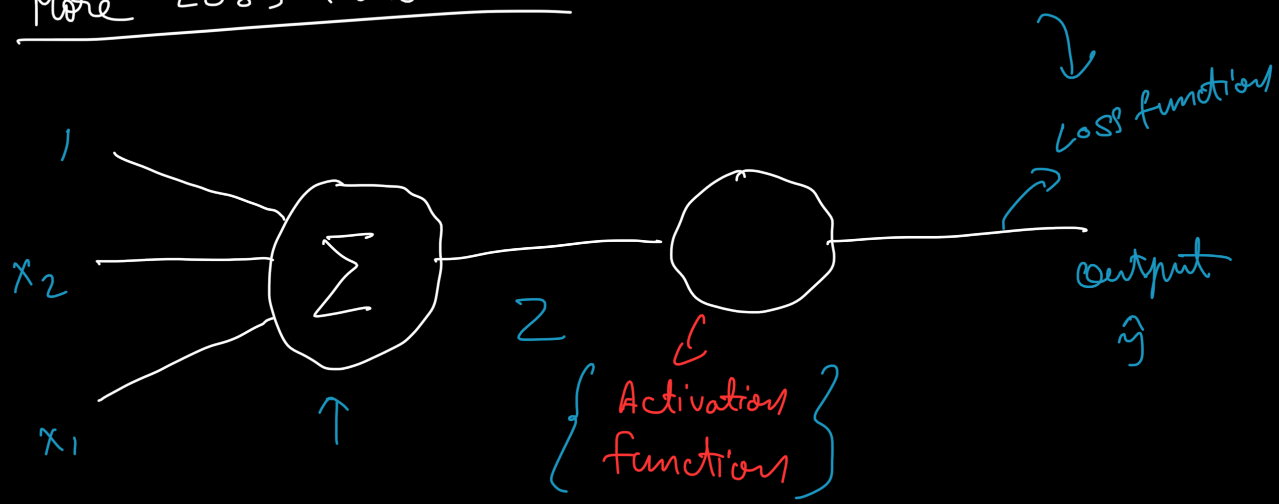
$$\frac{\partial L}{\partial w_2} = \begin{cases} 0 & \text{if } y_i f(x_i) \geq 0 \\ -y_i x_{i2} & \text{if } y_i f(x_i) < 0 \end{cases}$$

$$\frac{\partial L}{\partial b} = \begin{cases} 0 & \text{if } y_i f(x_i) \geq 0 \\ -y_i & \text{if } y_i f(x_i) < 0 \end{cases}$$

$\Downarrow$

New Code (go to code)

More Loss Functions



Perceptron is a Mathematical Model and very flexible according to the problem statement and need we can change loss fn and Activation fn also.

Suppose if we need probability in output then we can use Sigmoid activation function $\sigma(z) = \frac{1}{1+e^{-z}}$ $(z = w_1x_1 + w_2x_2 + b)$ and in loss fn we can use log loss.

log loss $= \boxed{-y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)}$ → loss f^n Binary cross Entropy

So when in perceptron we use sigmoid activation f^n and $\underbrace{\text{log loss}}_{\text{binary cross Entropy}}$, loss f^n then $\boxed{\text{perceptron} == \text{logistic Regression}}$

for multiclass classification Problem

Activation f^n = Softmax

$$f = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} = \frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}}$$

Loss function = categorical cross entropy

$$L = \sum_{j=1}^{M} y_j \log(\hat{y}_j)$$

for Regression

Activation → Linear (no activation f^n)

Loss function ↓ MSE $(y_i - \hat{y}_i)^2$

[Perceptron is a Mathematical Model and very much flexible by design.]

Loss f^n	Activation	output
Hinge loss	Step	Perceptron → binary classification -1 1
log loss	Sigmoid	logistic Reg 0 1 binary classification

Categorical cross entropy	Softmax	softmax regression → Multiclass classification output probabilities
MSE	No Activation	Linear Reg → number