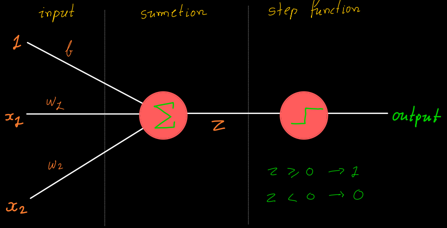
Perceptron

→ Perceptron is just a single neuron which can solve linearly separable problems.

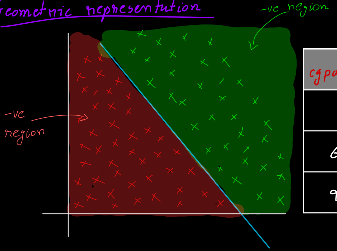
→ Output will be 0 or 1 depending on weighted sum & threshold.

→ It uses step function as activation function.

Structure



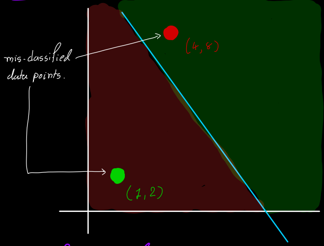
Geometric representation



cgpa	iq	placed
7	78	1
6	62	0
9	92	1

→ In 2D data, perceptron is a straight line, in 3D it would be a hyperplane.

Perceptron loss function



- loss calculation for both points.
- Suppose our straight line equation is $2x + 3y + 4 = 0$ where 2 & 3 are weights and 4 is bias.

$(4,8) = 2(4) + 3(8) + 4 = 36$
$(1,2) = 2(1) + 3(2) + 4 = 12$
$\boxed{48}$

→ So the total error is 48.
→ Take mod in case of negative output return by any data-point.

By scikit-learn......

$$E(w,b) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i)) + \underbrace{\alpha R(w)}_{\text{regularization parameter}}$$

for perceptron:

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

$\hookrightarrow f(x_i) = w_1x_1 + w_2x_2 + b = z$

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

→ The loss function completely depends on w_1, w_2, b

also x_1 = cgpa, x_2 = iq, y_i = placement.

→ Now we need to find.

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

argmin (argument of the minimum)

→ The value of the variable(s) that minimize a function.

→ We uses gradient descent for this.

Geometric intuition of loss function

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

x_{11}	x_{12}	y_1
x_{21}	x_{22}	y_2
⋮	⋮	⋮
	n	

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

→ for sometimes, let's suppose $-y_i f(x_i) = k$

hence $\max(0, -y_i f(x_i)) = \max(0, k)$.

$$k \begin{cases} k \rightarrow k \geq 0 \\ 0 \rightarrow k < 0 \end{cases}$$

$$-y_i f(x_i) \begin{cases} -y_i f(x_i) \rightarrow -y_i f(x_i) \geq 0 \\ 0 \rightarrow -y_i f(x_i) < 0 \end{cases}$$

for $n=2$

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

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

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

cgpa	iq	placed
7	8	1
6	8	-1
4	2	1
1	1	-1

1 : placement done

-1 : Not yet

4 possible combinations

y_i	$\hat{y}_i$	$\max(0, -y_i(f(x_i)))$
1	1	0
-1	-1	0
1	-1	+ve value
-1	1	+ve value

for y $\hat{y}$ $\max(0, -y_i f(x_i))$

1 1 → $\hat{y}_i = 1$, $f(x_i) \geq 0$ = y_i = +ve, $f(x_i)$ = +ve → max(0, -ve) = 0 $(w_1 1 + w_2 1 + b)$

-1 -1 → $\hat{y}_i = -1$, $f(x_i) < 0$ = y_i = -ve, $f(x_i)$ = -ve , -(+ve) = -ve → max(0, -ve) = 0 $(w_1 \cdot -1 + w_2 \cdot -2 + b)$

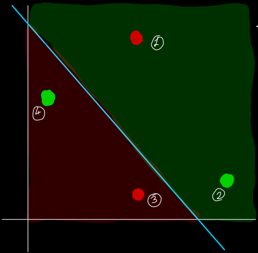
Since -ve * -ve = +ve

1 -1 → $\hat{y}_i = -1$, $f(x_i) < 0$ = y_i = +ve, $f(x_i)$ = -ve , -(-ve) = +ve → max(0, +ve) = +ve $(w_1 1 + w_2 \cdot -1 + b)$

Since -ve * +ve = -ve

-1 1 → $\hat{y}_i = -1$, $f(x_i) \geq 0$ = y_i = -ve, $f(x_i)$ = +ve , -(-ve) = +ve ⇒ max(0, +ve) = +ve $(w_1 \cdot -1 + w_2 1 + b)$

Since +ve * -ve = -ve



dp	y	$\hat{y}$	y_i	$f(x_i)$	$-y_i f(x_i)$	o/p
●	-1	1	-ve	+ve	+ve	+ve
●	1	1	+ve	+ve	-ve	0
●	-1	-1	-ve	-ve	-ve	0
●	1	-1	+ve	-ve	+ve	+ve

Classified data-point $\rightarrow 0$

Miss-classified data-points $\rightarrow$ +ve value

Gradient Descent

Step : 1

↳ Assign random values to w_1, w_2, b

Step : 2

↳ Initialize a loop & calculate partial derivatives of w_1, w_2 and b.

- $\rightarrow w_1 = w_1 + \eta \frac{\partial L}{\partial w_1}$
- $\rightarrow w_2 = w_2 + \eta \frac{\partial L}{\partial w_2}$
- $\rightarrow b = b + \eta \frac{\partial L}{\partial b}$

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

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

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

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

$$\frac{\partial L}{\partial W_2} = \begin{cases} 0 & \text{if } y_i f(x_i) \geq 0 \\ -y_i x_{i2} & \text{if } y_i f(x_i) < 0 \end{cases}$$

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

Other loss functions

Binary Cross Entropy

$$L = -y_i \log \hat{y}_i + (1-y_i) \log (1-\hat{y}_i)$$

→ if we use BCE as loss function, sigmoid as activation function & SGD as gradient descent. this combination will give probability as output.

Categorical Cross Entropy (Multi-class classification)

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

m = Number of O/P classes

→ CCE + Softmax = Multiclass classification

Loss-function	Activation fun.	Output
Hing-loss	step	Binary classifier
log-loss (BCE)	Sigmoid	Probability → Binary classification
CCE	Softmax	Softmax reg. - Multi-class classification
MSE, MAE	linear	Linear regression