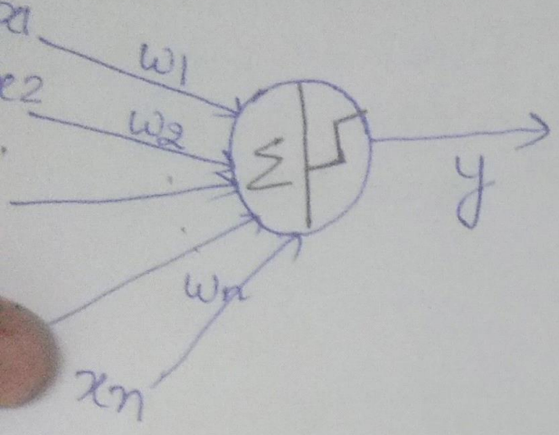
Unit-2 (Neural Networks ...

Back Propogation networks

Architecture :- Perceptron Model



$y = 1$ if $\sum_{i=1}^{n} w_i * x_i \geq \theta$

$= 0$ if $\sum_{i=1}^{n} w_i x_i < \theta$

Rewriting the above,

$y = 1$ if $\sum_{i=1}^{n} w_i x_i - \theta \geq 0$

$= 0$ if $\sum_{i=1}^{n} w_i * x_i - \theta < 0$

The perceptron model is a more general computational model than McCulloch-Pitts neuron. It takes an input, aggregates it (weighted sum) and returns 1 only if the aggregated sum is more than some threshold else returns 0.

A single perceptron can only be used to implement "linearly separable" functions. It takes both real & boolean inputs and associates a set of weights to them, along with a bias. We learn the weights, we get the function. Let's use a perceptron to learn an OR funct

Perceptron:- 1) Frank Rosenblatt, an Americal Psychologist, proposed the clasical perceptron model (1958).

2) More general computational model than McCulloch-Pitts neurons.

3) Main differences :- Introduction of numerical weights for inputs and a mechanism for learning these weights.

4) Refined & carefully analyzed by Minsky and Papert (1969) - their model is referred to as the perceptron model here.

OR function using a Perceptron

x_1	x_2	OR	
0	0	0	$w_0 + \sum_{i=1}^{2} w_i x_i < 0$
1	0	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
0	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
1	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$

$w_0 + w_1 \cdot 0 + w_2 \cdot 0 < 0 \Rightarrow w_0 < 0$

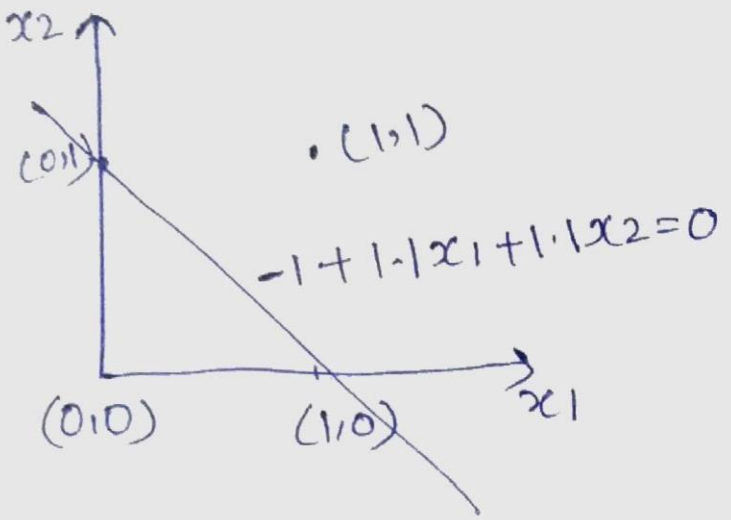
$w_0 + w_1 \cdot 0 + w_2 \cdot 1 \geq 0 \Rightarrow w_2 > -w_0$

$w_0 + w_1 \cdot 1 + w_2 \cdot 0 \geq 0 \Rightarrow w_1 > -w_0$

$w_0 + w_1 \cdot 1 + w_2 \cdot 1 \geq 0 \Rightarrow w_1 + w_2 > -w_0$

One possible solution is

$w_0 = -1, \; w_1 = 1.1, \; w_2 = 1.1$



In the above, the weighted sum has to be more than or equal to 0 when the o/p is 1. Based on the OR function output for various set of inputs, solved for weights based on those conditions & we get a line that perfectly separates positive inputs from those of negative.

Assignment:

Find out the decision boundary for AND, XOR, Tautology (always ON)

Single layer artificial Neural Network:-

One input layer & one output layer of processing units. No feedback connections (eg a Perceptron).

Perceptron learning Algorithm:-

P ← inputs with label 1
N ← inputs with label 0
initialize w randomly;
while ! convergence do
Pick random $x \in P \cup N$
if $x \in P$ and $\sum_{i=0}^{n} w_i * x_i < 0$ then
$w = w + x$
end
if $x \in N$ and $\sum_{i=0}^{n} w_i * x_i \geq 0$ then
$w = w - x$
end
end
// the algorithm converges when all the inputs are classified correctly.

$$\cos\alpha = \frac{w^T x}{\|w\| \|x\|}$$

case 1

for $x \in P$ if $w \cdot x < 0$ it means that the angle (α) between the x and the current w is greater than 90°. but we want α to be less than 90°

what happens to the new angle (α_{new}) when $w_{new} = w + x$

$$\cos(\alpha_{new}) \propto w_{new}^T x$$
$$\propto (w+x)^T x$$
$$\propto w^T x + x x^T$$
$$\propto \cos\alpha + x x^T$$
$$\cos(\alpha_{new}) > \cos\alpha$$

* Thus α_{new} will be less than α and this is exactly what we want.

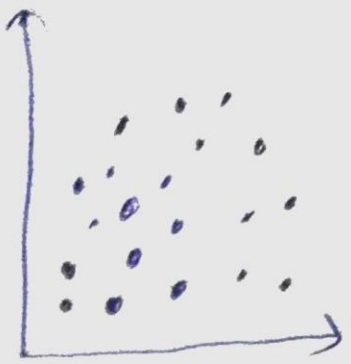
case 2 :- for $x \in N$ if $w \cdot x \geq 0$ then it means that the angle (α) between this x and the current w is less than 90° (but we want α to be greater than 90°

what happens to the new angle (α_{new}) when $w_{new} = w - x$

$$\cos(\alpha_{new}) \propto w_{new}^T x$$
$$\propto (w-x)^T x$$
$$\propto w^T x - x x^T$$
$$\propto \cos\alpha - x^T x$$
$$\cos(\alpha_{new}) < \cos\alpha$$

will be greater than α and this is exactly

1) Most real world data is not linearly separable and will always contain some outliers.



2) In fact, sometimes there may not be any outliers but still the data may not be linearly separable

3) We need computational units (models) which can deal with such data.

4) While a single perceptron can't deal with such data, we will show that a network of perceptrons can indeed deal with such data.

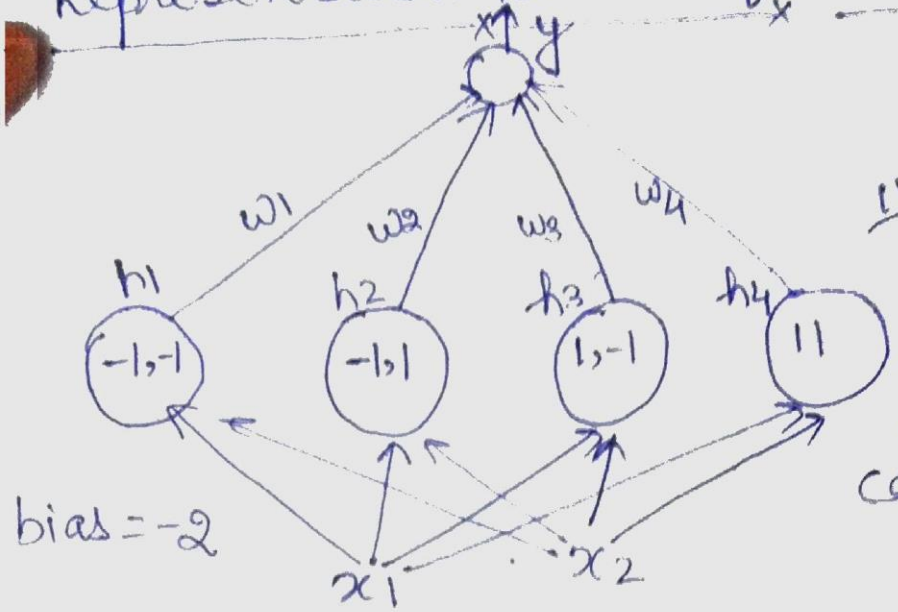
Before seeing how a network of perceptrons can deal with linearly inseparable data, we will discuss boolean functions in some more detail..

How many boolean functions can you design from 2 inputs?

x_1	x_2	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

* Of these, how many are linearly separable?
* In general, how many boolean functions can you for n inputs? 2^{2^n}
* How many of these 2^{2^n} functions are not linearly separable?

Representation Power of a Network of Perceptrons



Terminology:-

1) This network contains 3 layers

2) The layer containing the i/ps (x_1, x_2) is called the i/p layer.

3) The middle layer containing the 4 perceptrons is called the hidden layer.

4) The final layer containing one output neuron is called the output layer.

5) The outputs of the 4 perceptrons in the hidden layer are denoted by h_1, h_2, h_3, h_4

6) w_1, w_2, w_3, w_4 are called layer 2 weights.

7) We claim that this network can be used to implement any boolean function linearly

separable or not)!

8) In other words, we can find w_1, w_2, w_3, w_4 such that the truth table of any boolean function can be represented by this network.

9) Each perceptron in the middle layer fires only for a specific input (and no two perceptron fire for the same input)
the first perceptron fires for $\{-1,-1\}$
The second perceptron fires for $\{-1,1\}$
The third perceptron fires for $\{1,-1\}$
The fourth perceptron fires for $\{1,1\}$

10) Let us see why this network works by taking an example of the XOR function.

11) Let w_0 be the bias output of neuron (i.e, it will fire if $\sum_{i=1}^{4} w_i h_i \geq w_0$)

x_1	x_2	XOR	h_1	h_2	h_3	h_4	$\sum_{i=1}^{4} w_i h_i$
0	0	0	1	0	0	0	w_1
0	1	1	0	1	0	0	w_2
1	0	1	0	0	1	0	w_3
1	1	0	0	0	0	1	w_4

This results in the following four conditions to implement XOR: $w_1 < w_0$, $w_2 \geq w_0$, $w_3 \geq w_0$, $w_4 < w_0$

* Unlike before, there are no contradictions now and the system of inequalities can be satisfied.
* Essentially each w_i is now responsible for one of the 4 possible inputs and can be adjusted to get the desired O/P for that input.

Theorem:- Any boolean function of n inputs can be represented exactly by a network of perceptrons containing 1 hidden layer with 2^n perceptrons and one O/P layer containing 1 perceptron.

Note:- A network of 2^n+1 perceptrons is not necessary but sufficient. For eg. we already saw how to represent AND function with just 1 Perceptron.

Catch:- As n increases the no. of perceptrons in the hidden layers obviously increases exponentially.

Ex:- What do we do about functions which are not linearly separable.

XOR

x_1	x_2	XOR	
0	0	0	$w_0 + \sum_{i=1}^{2} w_i x_i < 0$
1	0	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
0	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
1	1	0	$w_0 + \sum_{i=1}^{2} w_i x_i < 0$

$w_0 + w_1 \cdot 0 + w_2 \cdot 0 < 0 \Rightarrow w_0 < 0$ — (1)

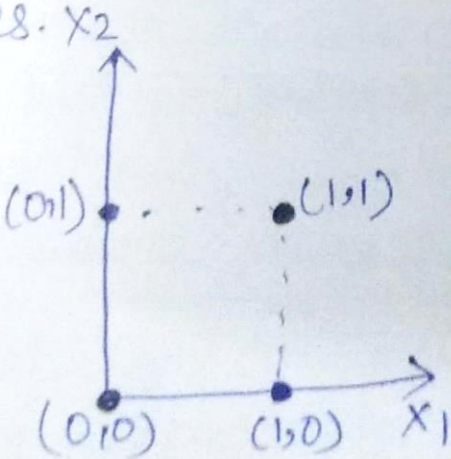
$w_0 + w_1 \cdot 0 + w_2 \cdot 1 \geq 0 \Rightarrow w_2 \geq -w_0$ — (2)

$w_0 + w_1 \cdot 1 + w_2 \cdot 0 \geq 0 \Rightarrow w_1 \geq -w_0$ — (3)

$w_0 + w_1 \cdot 1 + w_2 \cdot 1 < 0 \Rightarrow w_1 + w_2 < -w_0$ — (4)

The fourth condition contradicts conditions 2 & 3.

Hence we can't have a solution to this set of inequalities.



And indeed you can see that it is impossible to draw a line which separates the black points from the blue points.