Soft Computing Assignment 03

Implementation of Perceptron Algorithm for AND, OR, XOR and XNOR operation.

Md Afzal Ansari(MIT2019072)

1 Perceptron Algorithm

The Perceptron algorithm states that:

Prediction (y‘) = 1 if Wx+b $\geq$ 0 and 0 if Wx+b $<$ 0

2 Verification

Perceptron algorithm is used in order to train the network on the given data and then find the weights associated with the network. Later the value of the weights are used in order to get the predicted results.

AND Gate

The truth table for the AND gate is given as follows:

Input 1	**Input 2**	**Output**
0	0	0
0	1	0
1	0	0
1	1	1

Table 1: AND operation.

From the truth table we understood that gate output is 1 only when both the input signal are 1. We need to check the value of weight and bias obtained after running the algorithm.

In this case we obtained the weights as w1 = 0.2, w2 = 0.1 as and bias as -0.2.

Row 1

- Passing the first row of the AND logic table (x1=0, x2=0), we get; 0+0–0.2 = -0.2
- From the Perceptron rule, if Wx+b < 0, then y'=0. Therefore, this row is correct, and no need for Backpropagation.

Row 4

- Passing the first row of the AND logic table (x1=1, x2=1), we get; 0.2+0.1–0.2 = 0.1
- From the Perceptron rule, if Wx+b ≥ 0, then y'=1. Therefore, this row is correct, and no need for Backpropagation.

Similarly, other cases can be verified.

OR Gate

The truth table for the OR gate is given as follows:

Input 1	**Input 2**	**Output**
0	0	0
0	1	1
1	0	1
1	1	1

Table 2: OR operation.

From the truth table we understood that gate output is 1 when anyone of the input signal is 1. We need to check the value of weight and bias obtained after running the algorithm.
In this case we obtained the weights as w1, w2 as 0.1 and bias as -0.1.

Row 1

- Passing the first row of the OR logic table (x1=0, x2=0), we get; 0+0–0.1 = –0.1
- From the Perceptron rule, if Wx+b < 0, then y'=0. Therefore, this row is correct, and no need for Backpropagation.

Row 4

- Passing the first row of the OR logic table (x1=1, x2=1), we get; 0.1+0.1–0.1 = 0.1
- From the Perceptron rule, if Wx+b ≥ 0, then y'=1. Therefore, this row is correct, and no need for Backpropagation.

Similarly, other cases can be verified.

XOR Gate

The truth table for the XOR gate is given as follows:

Input 1	**Input 2**	**Output**
0	0	0
0	1	1
1	0	1
1	1	0

Table 3: OR operation.

From the truth table we understood that gate output is 1 when both the input signal are different. We need to check the value of weight and bias obtained after running the algorithm.
In this case we need:
1.w1 + 0.w2 cause a fire, i.e. $\geq$ t
0.w1 + 1.w2 $\geq$ t
0.w1 + 0.w2 doesn't fire, i.e. $<$ t
1.w1 + 1.w2 also doesn't fire, $<$ t

w1 $\geq$ t
w2 $\geq$ t
0 $<$ t
w1+w2 $<$ t
Contradiction.

A "single-layer" perceptron can't implement XOR. The reason is because the classes in XOR are not linearly separable. You cannot draw a straight line to separate the points (0,0),(1,1) from the points (0,1),(1,0).
We implemented a multi-layer perceptron network.

XOR(x1,x2) = AND(NOT(AND(x1,x2)),OR(x1,x2))

As we can see the network consist of combination of the AND, OR and NOT gates.
After training of the network we will test for the any random input and check the obtained predicted values.

Similarly, other cases can be verified.

XNOR Gate

The truth table for the XNOR gate is given as follows:

Input 1	**Input 2**	**Output**
0	0	1
0	1	0
1	0	0
1	1	1

Table 4: OR operation.

From the truth table we understood that gate output is 1 when both the input signal are same. We need to check the value of weight and bias obtained after running the algorithm.
Like XOR gate, XNOR gate cannot be implemented using single layer perceptron.
We implemented a multi-layer perceptron network.

XOR(x1,x2) = NOT(AND(NOT(AND(x1,x2)),OR(x1,x2)))

As we can see the network consist of combination of the AND, OR and NOT gates.
After training of the network we will test for the any random input and check the obtained predicted values.

Similarly, other cases can be verified.