Batch: SC_1 **Roll No.: 1911027**

Experiment No. : 02

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title: Perceptron learning algorithm.

Objective: To write a program to implement Logic gate using perceptron network.

Expected Outcome of Experiment:

CO1 : Identify and describe soft computing techniques and their roles

Books/ Journals/ Websites referred:

- J.S.R.Jang, C.T.Sun and E.Mizutani, "Neuro-Fuzzy and Soft Computing", PHI, 2004, Pearson Education 2004.
- Davis E.Goldberg, "Genetic Algorithms: Search, Optimization and Machine Learning", Addison Wesley, N.Y., 1989.
- S. Rajasekaran and G.A.V.Pai, "Neural Networks, Fuzzy Logic and Genetic Algorithms", PHI, 2003.
- http://library.thinkquest.org/C007395/tqweb/history.html

Pre Lab/ Prior Concepts:

Neural networks, sometimes referred to as connectionist models, are parallel-distributed models that have several distinguishing features-

1) A set of processing units;
2) An activation state for each unit, which is equivalent to the output of the unit;
3) Connections between the units. Generally each connection is defined by a weight w_{jk} that

determines the effect that the signal of unit *j* has on unit *k;*

4) A propagation rule, which determines the effective input of the unit from its external inputs;

5) An activation function, which determines the new level of activation based on the effective input and the current activation;

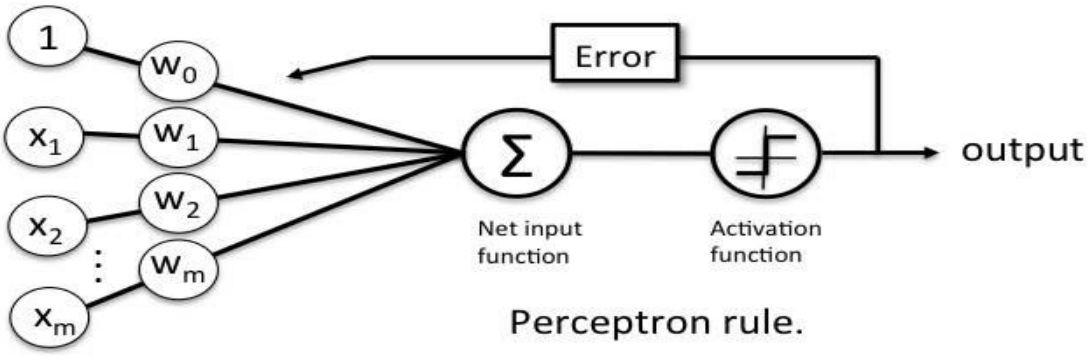
6) An external input (bias, offset) for each unit;

7) A method for information gathering (learning rule);

8) An environment within which the system can operate, provide input signals and, if necessary, error signals.

Implementation Details:

Perceptron learning concept: A perceptron, a neuron's computational prototype, is categorized as the simplest form of a neural network. Frank Rosenblatt invented the perceptron at the Cornell Aeronautical Laboratory in 1957. A perceptron has one or more than one inputs, a process, and only one output. The concept of perceptron has a critical role in machine learning. It is used as an algorithm or a linear classifier to facilitate supervised learning of binary classifiers. Supervised learning is amongst the most researched of learning problems. A supervised learning sample always consists of an input and a correct/explicit output. The objective of this learning problem is to use data with correct labels for making predictions on future data, for training a model. Some of the common problems of supervised learning include classification to predict class labels. A linear classifier that the perceptron is categorized as is a classification algorithm, which relies on a linear predictor function to make predictions. Its predictions are based on a combination that includes weights and feature vector. The linear classifier suggests two categories for the classification of training data. This means, if classification is done for two categories, then the entire training data will fall under these two categories. The perceptron algorithm, in its most basic form, finds its use in the binary classification of data. Perceptron takes its name from the basic unit of a neuron, which also goes by the same name. In some scenarios and machine learning problems, the perceptron learning algorithm can be found out, if you like. It could show limitations that you never knew existed.



Perceptron rule.

But then, this is the problem with most, if not all, learning algorithms. They are ideal for some problems, not so for others. At one point, the perceptron networks were also found to be not capable enough of implementing some basic functions. However, this problem was dealt with as soon as multi-layer perceptron networks and improved learning rules came into the picture. Perceptron today has become an important learning algorithm in the world of artificial intelligence and machine learning. It is considered a reliable and fast solution for the category of problems it has the capabilities of solving. Also, if you develop an understanding of how the perceptron works, you will find the job of understanding more complex networks a lot easier.

Primary components of a perceptron:

1. **Input:** Features are taken as inputs in the perceptron algorithm. Inputs are denoted as x1, x2, x3, x4, .xn – 'x' in these inputs indicates the feature value and 'n' the total occurrences of these features.

2. **Weights:** These are values that are calculated during the training of the model. The weights are given an initial value at the start. With every occurrence of a training error, the values of weights are updated. Weights are represented as w1, w2, w3, w4, ..wn.

3. **Bias:** As we alluded to earlier, bias is a special input type. It allows the classifier to move the decision boundary around from its original position to the right, left, up, or down. In terms of algebra, the bias allows the classifier to turn its decision boundary around.

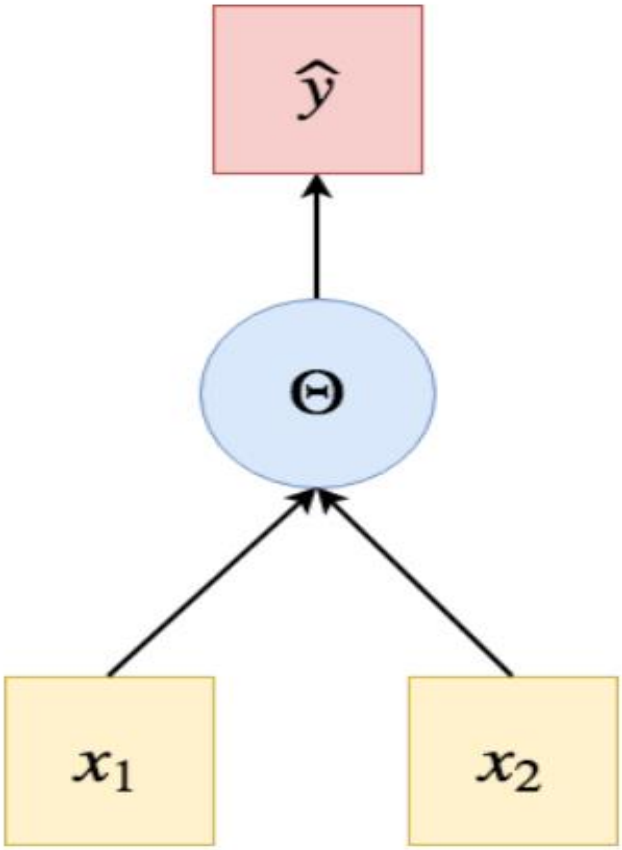
4. **Activation/step function:** Activation or step functions are used to create non-linear neural networks. These functions can change the value of neural networks to 0 or 1. The conversion of value is done to make a data set easy to classify**.**

5. **Weighted summation:** The multiplication of every feature or input value (xn) associated with corresponding weight values (wn) gives us a sum of values that are called weighted summation. Weighted summation is represented as ∑wixi for all i -> [1 to n].

AND Function: The AND logical function is a 2-variables function, AND(x1, x2), with binary inputs and output. This graph is associated with the following computation:

$\hat{y} = \Theta(w1*x1 + w2*x2 + b)$

This time, we have three parameters: w1, w2, and b. Can you guess which are three values for these parameters which would allow the perceptron to solve the AND.



Truth Table :

x1	x2	y

0	0	0
0	1	0
1	0	0
1	1	1

Algorithm:

1. Feed the features of the model that is required to be trained as input in the first layer.

2. All weights and inputs will be multiplied – the multiplied result of each weight and input will be added up.

3. The Bias value will be added to shift the output function.

4. This value will be presented to the activation function (the type of activation function will depend on the need).

5. The value received after the last step is the output value.

Code:

```
def ANDPERCEPTROM(x1,x2):
    t1=x1*w1+x2*w2
    if(t1>=threshold):
        return 1
    else:
        return 0
print("-------------------------------------------------------")
print("---------------------AND FUNCTION----------------------")
print("-------------------------------------------------------")
x1=[0,0,1,1]
x2=[0,1,0,1]
w1=1
w2=1
threshold=2
```

```
output=list()
print("\t      x1  x2  y     Status")
print("-----------------------------------------------------")
for i,j in zip(x1,x2):
    flag=ANDPERCEPTROM(i,j)
    if(flag==1):
        print("\t     ", i, " ", j, " ",flag, " : Activated")
    else:
        print("\t     ", i, " ", j, " ",flag, " : Not Activated")
    output.append(flag)
```

Output:

```
-----------------------------------------------------
--------------------AND FUNCTION---------------------
-----------------------------------------------------
             x1  x2  y     Status
-----------------------------------------------------
             0   0   0  : Not Activated
             0   1   0  : Not Activated
             1   0   0  : Not Activated
             1   1   1  : Activated
```

Conclusion: Thus, we have successfully implemented perceptron training algorithm for AND function. Implemented the same using python programming language. Also understood the concept of perceptron learning.

Post Lab Descriptive Questions :

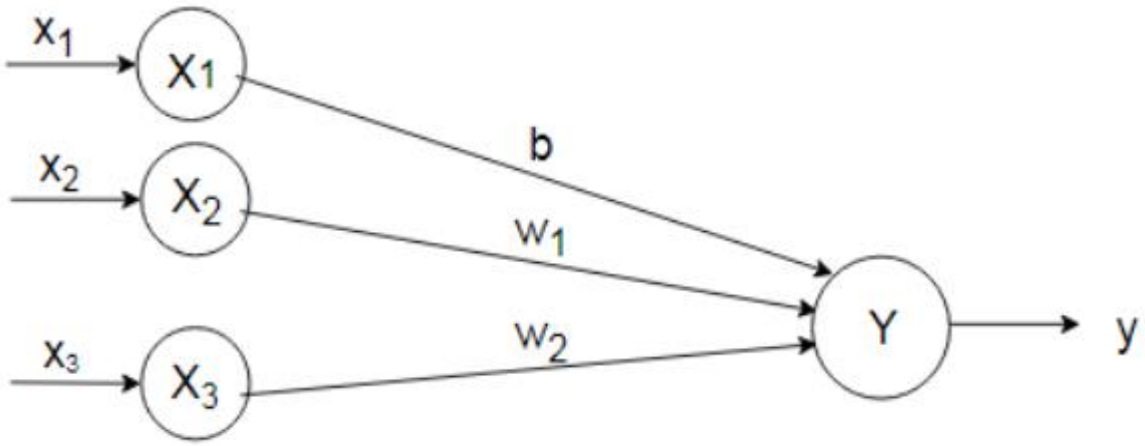
1. Explain Linear Seperability.

ANS) An ANN does not give an exact solution for a nonlinear problem. However, it provides an approximate solution to nonlinear problems. Linear separability is the concept wherein the separation of input space into regions is based on whether the network response is positive or negative. A decision line is drawn to separate positive and negative responses. The decision line may also be

called as the decision-making Line or decision-support Line or linear-separable line. The necessity of the linear separability concept was felt to clarify classify the patterns based upon their output responses. Generally, the net input calculated to the output unit is given as –

$$y_{in} = b + \sum_{i=1}^{n}(x_i w_i)$$

The linear separability of the network is based on the decision-boundary line. If there exist weight for which the training input vectors having a positive (correct) response, or lie on one side of the decision boundary and all the other vectors having negative, −1, response lies on the other side of the decision boundary then we can conclude the problem is "Linearly Separable". Consider, a single layer network as shown in the figure.



The net input for the network shown in the figure is given as-

yin=b+x1w1+x2w2

The separating line for which the boundary lies between the values x1 and x2, so that the net gives a positive response on one side and negative response on the other side, is given as,

b+x1w1+x2w2=0

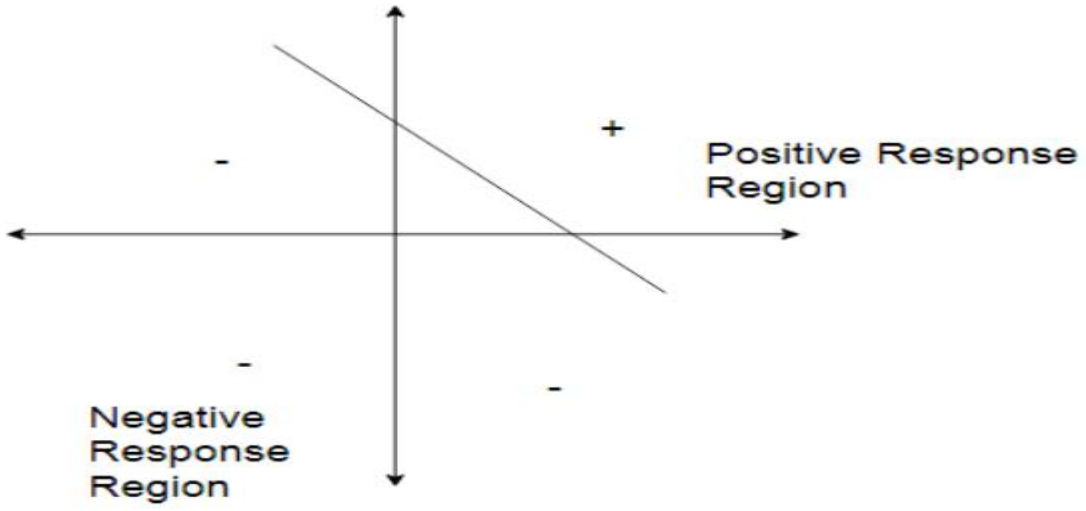
If weight w2 is not equal to 0 then we get,

x2=−w1w2,x1=−bw2

Thus, the requirement for the positive response of the net is

$b+x1w1+x2w2>0$

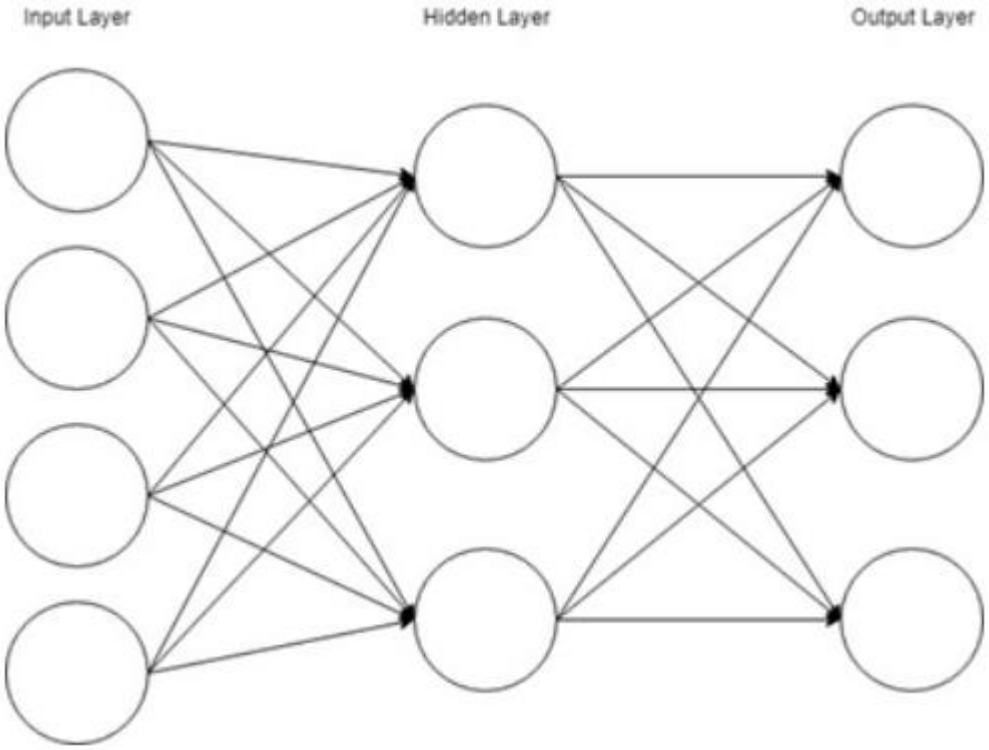
During the training process, the values of w1,w2 and b are determined so that the net will produce a positive (correct) response for the training data. If on the other hand, the threshold value is being used, then the condition for obtaining the positive response from the output unit is, Net input received $>\theta$ (threshold) $yin>\theta$ $x1w1+x2w2>\theta$ The separating line equation will then be, $x1w1+x2w2=\theta$ $x2=-w1w2x1+\theta w2$ (with $w2\neq0$) During the training process, the values of w1 and w2 have to be determined, so that the net will have a correct response to the training data. For this correct response, the line passes close through the origin. In certain situations, even for a correct response, the separating line does not pass through the origin.



2. Explain MLP.

ANS) A multilayer perceptron (MLP) is a feedforward artificial neural network that generates a set of outputs from a set of inputs. An MLP is characterized by several layers of input nodes connected as a directed graph between the input and output layers. MLP uses backpropogation for training the network. MLP is a deep learning method. A multilayer perceptron is a neural network connecting multiple layers in a directed graph, which means that the signal path through the nodes only goes one way. Each node, apart from the input nodes, has a nonlinear activation function. An MLP uses

backpropagation as a supervised learning technique. Since there are multiple layers of neurons, MLP is a deep learning technique. MLP is widely used for solving problems that require supervised learning as well as research into computational neuroscience and parallel distributed processing. Applications include speech recognition, image recognition and machine translation. The Perceptron consists of an input layer and an output layer which are fully connected. MLPs have the same input and output layers but may have multiple hidden layers in between the aforementioned layers, as seen below.



The algorithm for the MLP is as follows:

1. Just as with the perceptron, the inputs are pushed forward through the MLP by taking the dot product of the input with the weights that exist between the input layer and the hidden layer (W---H). This dot product yields a value at the hidden layer. We do not push this value forward as we would with a perceptron though.

2. MLPs utilize activation functions at each of their calculated layers. There are many activation functions to discuss: rectified linear units (ReLU), sigmoid function, tanh. Push the calculated output at the current layer through any of these activation functions.

3. Once the calculated output at the hidden layer has been pushed through the activation function, push it to the next layer in the MLP by taking the dot product with the corresponding weights.

4. Repeat steps two and three until the output layer is reached.

5. At the output layer, the calculations will either be used for a backpropagation algorithm that corresponds to the activation function that was selected for the MLP (in the case of training) or a decision will be made based on the output (in the case of testing).

MLPs form the basis for all neural networks and have greatly improved the power of computers when applied to classification and regression problems. Computers are no longer limited by XOR cases and can learn rich and complex models thanks to the multilayer perceptron.

Date: 2 / 9 / 2021 **Signature of faculty in-charge**