|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **ANNFS** | **Semester:** | **VI** |
| **Date of Performance:** |  | **Batch No:** | **1** |
| **Faculty Name:** | **Dr. Jagannath Nirmal** | **Roll No:** | **1912052** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** |  |

**Experiment No: 4**

**Title:**

**ADALINE learning algorithm.**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| Write a program to implement ANDNOT logic / any logic gate using ADALINE training algorithm. |

|  |
| --- |
| **COs to be achieved:** |
| **CO1:**  **Understand the role of neural networks in engineering, artificial intelligence, and cognitive modeling.**    **CO3:**  **Develop concepts and techniques of neural networks through the study of the most important supervised and unsupervised neural network models.** |

**Apparatus / Software tools used: MATLAB**

|  |
| --- |
| **Theory:** |
| Write theory: |

|  |
| --- |
| **Block Diagram/Program flowchart:** |
| clc;  clear all;  close all;  %Enter weights, bias and threshold value  disp('Enter initial weights');  W1o = input('W1 = ');  W2o = input('W2 = ');  disp('Enter initial bias');  bo = input('b = ');  disp('Enter Threshold value');  theta = input('theta = ');  %Defining AND gate parameters  X1 = [1 1 -1 -1];  X2 = [1 -1 1 -1];  b = [1 1 1 1];  Z = [0 0 0 0];  Y = [1 -1 -1 -1];  a = 0.4;  Yin = X1(1)\*W1o + X2(1)\*W2o + bo;  %Training weights and bias  for i=1:4  W1n = W1o + X1(i)\*(Y(i)-Yin)\*a;  W1o = W1n;  W2n = W2o + X2(i)\*(Y(i)-Yin)\*a;  W2o = W2n;  bn = bo + Y(i)\*a;  bo = bn;  end  %Finding output and comparing with target output  con = 1;  while con  Yin = X1\*W1n + X2\*W2n + bn;  for i=1:4  if Yin(i) >= theta  Z(i) = 1;  else  Z(i) = -1;  end  end  disp('Output of Net');  disp(Z);  if Z==Y  con = 0;  else  disp('Network is not learning');  return  end  end  %Displaying the correct weights and threshold value  disp('Adaline Model for AND function');  disp('Weights of Neuron');  disp(W1n);  disp(W2n);  disp('Weights of Bias');  disp(bn);  disp('Threshold value');  disp(theta);  %plotting the trained network  hold on;  scatter(X1(Z>=1),X2(Z>=1),'r','filled','linewidth',10);  scatter(X1(Z<1),X2(Z<1),'b','filled','linewidth',10);  X1 = [-0.75 0 0.75 1.5];  X2 = [1.5 0.75 0 -0.75];  plot(X1,X2,'g','Linewidth',2);  hold off;  xlim([-1.2 1.2]);  ylim([-1.2 1.2]);  title('PERCEPTRON Model for AND function');  xlabel('X1');  ylabel('X2'); |

|  |
| --- |
| **Algorithm:** |
| C:\Users\Admin\Desktop\Picture2a.jpg  C:\Users\Admin\Desktop\Picture2b.jpg  C:\Users\Admin\Desktop\Picture2c.jpg |

|  |
| --- |
| **Observation:** |
| Attach program and its results after execution |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| **i) Differentiate between Adaline and Medaline learning rule**   |  |  | | --- | --- | | **Adaline:** | Medaline | | ♣ Known as Adaptive Linear Neuron. | ♣ Stands for multiple adaptive linear neuron. | | ♣ Adaline is a network with a single linear unit. | ♣ It consists of many adalines in parallel with a single output unit whose value is based on certain selection rules. | | ♣ The Adaline network is trained using the delta rule. | ♣ It uses the majority vote rule. |   **ii) Describe delta rule and gradient descent method**   * The Delta Rule uses one of the most, if not the most, popular search technique in the hypothesis space that is called Gradient Descent. * The Delta Rule, uses gradient descent as an optimization techniques, and tries different values for the weights in a neural network, and depending on how accurate the output of the network is (i.e., how close to the ground truth), it will make certain adjustments to certain weights (i.e., increase some and decrease the other). It will try to increase and decrease the weights in a way that the error of the output would go down, during training. |

|  |
| --- |
| **Conclusion:** |
|  |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |