Akademia Górniczo-Hutnicza

WIMiIP, Inżynieria Obliczeniowa

G01, Małgorzata Kusik

Nr. indeksu 293103

Podstawy Sztucznej Inteligencji

Sprawozdanie z Projektu 1

Budowa i działanie perceptronu

1. Cel projektu:

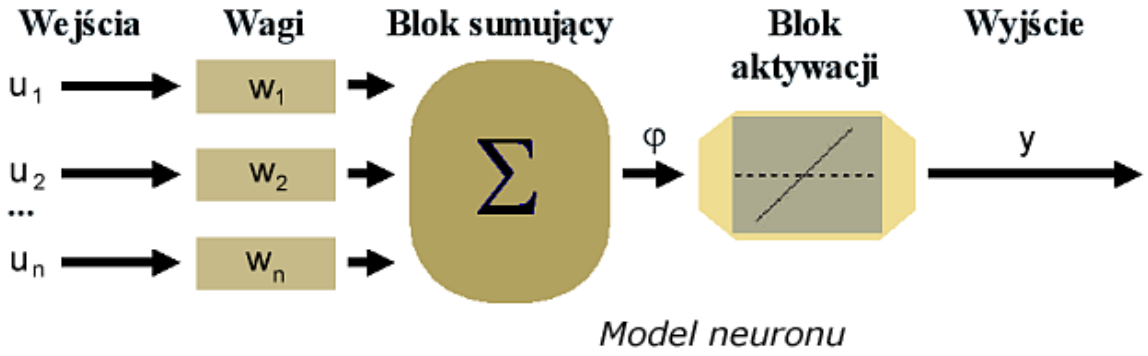
Głównym celem projektu 1 było poznanie tego jak jest zbudowany i jak działa perceptron. Możemy to zaobserwować poprzez implementacje oraz uczenie perceptronu realizującego funkcję logiczną dwóch zmiennych.

Zadania do wykonania:
– implementacja sztucznego neuronu według podanego na wykładzie algorytmu
– wygenerowanie danych uczących i testujących wybranej funkcji logicznej dwóch
zmiennych
– uczenie perceptronu dla różnej liczby danych uczących, różnych współczynników
uczenia
– testowanie perceptronu

2. Część teoretyczna

Perceptron to prosty element obliczeniowy, który sumuje ważone sygnały wejściowe i porównuje tę sumę z progiem aktywacji – w zależności od wyniku perceptron może być albo wzbudzony (wynik 1), albo nie (wynik 0).

Sztuczny neuron - prosty system przetwarzający wartości sygnałów wprowadzanych na jego
wejścia w pojedynczą wartość wyjściową, wysyłaną na jego jedynym wyjściu (dokładny sposób funkcjonowania określony jest przez przyjęty model neuronu). Jest to podstawowy element sieci neuronowych, jednej z metod sztucznej inteligencji, pierwowzorem zbudowania sztucznego neuronu był biologiczny neuron.
Schemat sztucznego neuronu:



Model neuronu

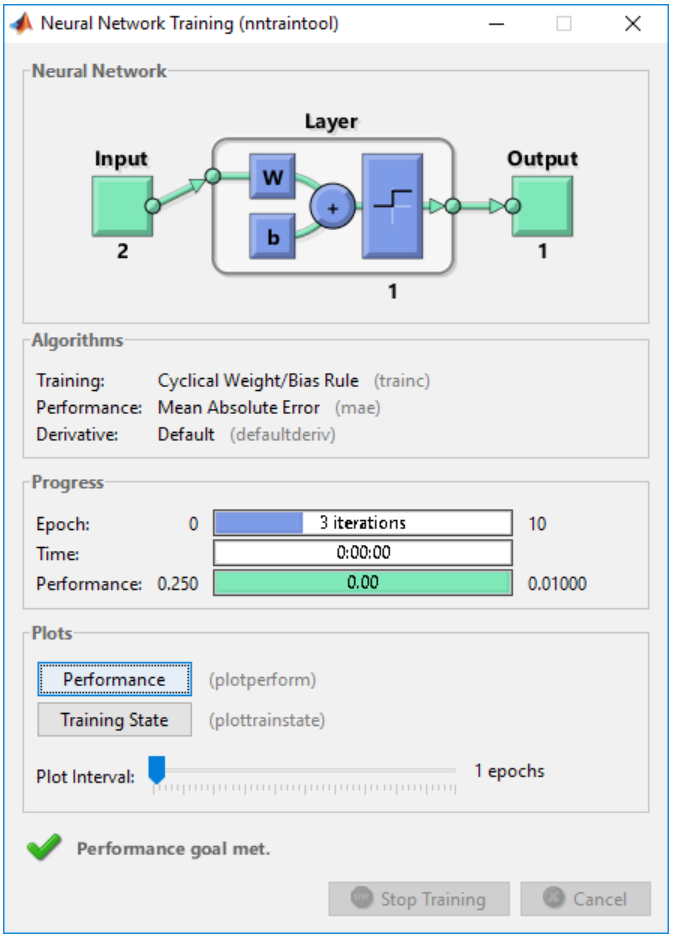
Algorytm uczenia sieci jest procesem dochodzenia wag do wartości optymalnych –
zapewniających odpowiednie reakcje sieci na pokazywane jej sygnały wejściowe, czyli prawidłowe
rozwiązanie zadań.

W programie Matlab zaimplementowałam sztuczny neuron realizujący bramkę logiczną OR.
Bramka zwraca 1 w przypadku wprowadzenia przynajmniej jednej jedynki.
Funkcje, które wykorzystałam:
– **newp** – tworzy jednowarstwową sieć neuronową
– **plotpv** – do wyświetlania wektorów
– **sim** – symuluje działanie perceptronu
– **plotpc** – wyświetla granicę decyzyjną dla sieci neuronowej

Net jest to struktura zawierająca opis sieci neuronowej.

3. Wyniki:

```
close all; clear all; clc;
% Schemat budowy i działania perceptronu o dwóch we

net = newp([0 1; -2 2], 1, 'hardlim');

% Wejście 1; Wejście 2
wejscie = [0 0 1 1; 0 1 0 1];

% Wyjście tj. warosci wyniku bramki OR
wyjscie = [0 1 1 1];
plotpv(wejscie, wyjscie);

net.name = 'Bramka OR';
net = init(net);
Y = sim(net, wejscie)

% Ustawienia domyślne uczenia perceptronu:
net.trainParam.epochs = 10;
net.trainParam.goal = 0.01;
net.trainParam.mu = 0.001;
net.adaptParam.passes = 10;
net.trainParam.show = 15;

net = train(net, wejscie, wyjscie);
% wagi:
plotpc(net.iw{1,1}, net.b{1})
Y1 = sim(net, wejscie)

% testowanie perceptronu:
test = [0 0 1 1; 0 1 0 1]
efekt = sim(net, test)

disp(net);
```



Powyższy kod przedstawia wywołanie funkcji, która tworzy sieć zawierającą pojedynczy neuron o dwóch wejściach.
Zakresy wartości:

- pierwszego wejścia to [0,1]
- drugiego wejścia to [-2,2]

Nastąpiło utworzenie sieci, a następnie wykonane zostało uczenie z maksymalną liczbą iteracji równą 20.
Wykorzystana została bramka AND, której koniunkcja dwóch zdań „*a*” i „*b*” jest zdaniem
prawdziwym wtedy i tylko wtedy, gdy oba zdania „*a*”, „*b*” są zdaniami prawdziwymi.

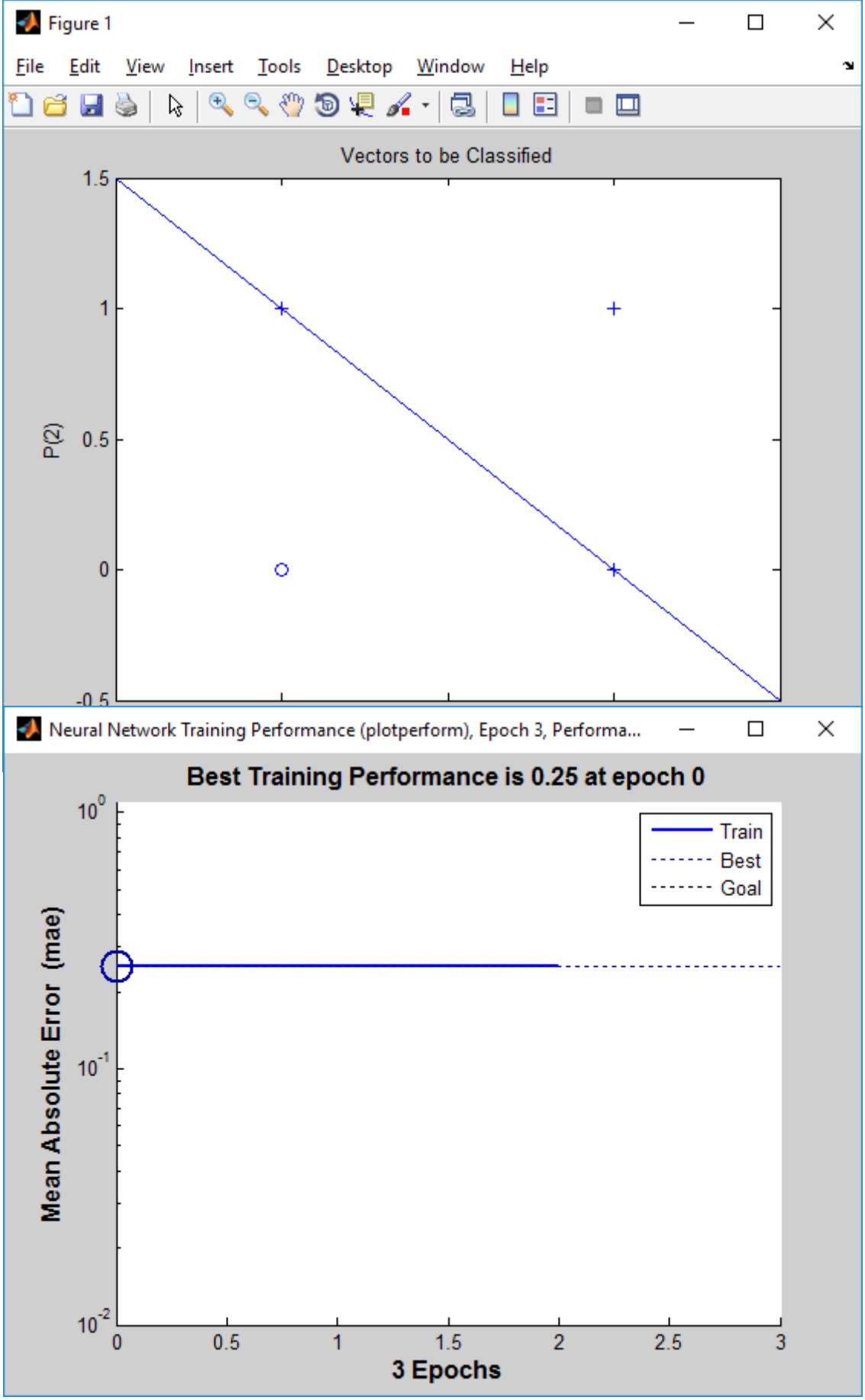


Figure 1
File Edit View Insert Tools Desktop Window Help
Vectors to be Classified
P(2)
1.5
1
0.5
0
-0.5
Neural Network Training Performance (plotperform), Epoch 3, Performa...
Best Training Performance is 0.25 at epoch 0
Train
Best
Goal
Mean Absolute Error (mae)
10^{0}
10^{-1}
10^{-2}
0
0.5
1
1.5
2
2.5
3
3 Epochs

4. Wnioski:

Po wykonaniu projektu dowiedziałam się jak działa perceptron i algorytm uczenia. Po przetestowaniu działania programu zauważyłam, że zmiana wag wpływa na długość trwania algorytmu ucznia sieci. Wysokie wagi powodują, że czas nauki wydłuża się nawet kilkukrotnie. Funkcja TRAIN wygenerowała zamierzony efekt, sieć została nauczona, zamierzony cel chciałam uzyskać w 20 epokach (iteracjach) w tym przypadku sieć dokonała tego do 3 epoki, później jednak wystąpiło przeuczenie.