|  |  |
| --- | --- |
| **Pis Marcin**  **Numer indeksu: 293113**  **Inżynieria Obliczeniowa** | **Podstawy Sztucznej Inteligencji**  **Sprawozdanie 1** |
| **Budowa i działanie perceptronu** | |

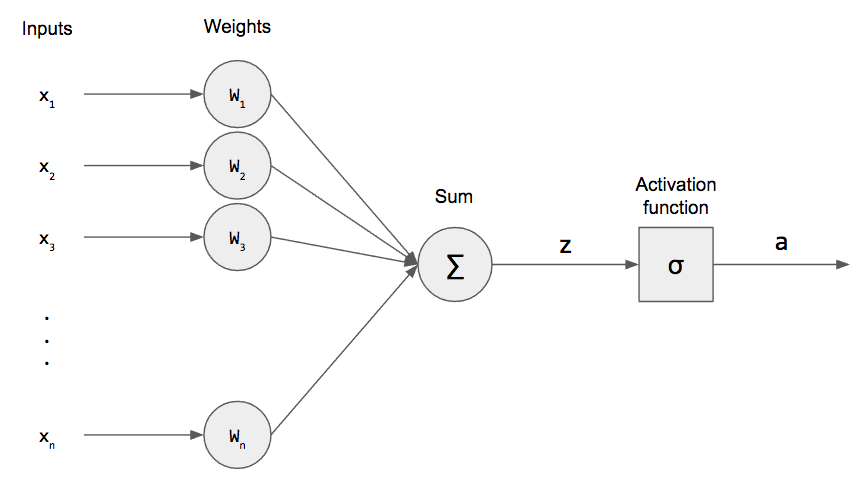
1. **Cel projektu:**

* Poznanie budowy i działania perceptronu
* Implementacja perceptronu
* Wygenerowanie danych uczących i testujących perceptron
* Uczenie i testowanie perceptronu

1. **Podstawowe pojęcia:**

**Perceptron** - najprostsza sieć neuronowa, składająca się z jednego bądź wielu niezależnych neuronów McCullocha-Pittsa, implementująca algorytm uczenia nadzorowanego klasyfikatorów binarnych. Pojęcie perceptronu zostało wprowadzone przez Franka Rosenblatta w roku 1957. Perceptron jest funkcją, która potrafi określić przynależność parametrów wejściowych do jednej z dwóch klas. Może być wykorzystywany tylko do klasyfikowania zbiorów liniowo separowalnych. Działanie perceptronu polega na klasyfikowaniu danych pojawiających się na wejściu i ustawianiu stosownie do tego wartości wyjścia. Przed używaniem perceptron należy wytrenować, podając mu przykładowe dane na wejście i modyfikując w odpowiedni sposób wagi wejść i połączeń między warstwami neuronów, tak aby wynik na wyjściu przybierał pożądane wartości. Perceptrony mogą klasyfikować dane na zbiory, które są liniowo separowalne. Własność ta uniemożliwia na przykład wytrenowanie złożonego z jednego neuronu perceptronu, który wykonywałby logiczną operację XOR na wartościach wejść. Algorytm uczenia perceptronu jest formalnie bardzo podobny do algorytmu spadku gradientowego.





System nerwowy stanowił naturalne źródło inspiracji dla sztucznej inteligencji. Stąd zainteresowanie neuronem - podstawową jednostką systemu nerwowego.

**Sztuczna sieć neuronowa** - Zbiór połączonych ze sobą jednostek wejściowo-wyjściowych. Z każdym połączeniem skojarzona jest waga, która może zostać zmieniona w trakcie uczenia.

1. **Rozwiązanie:**

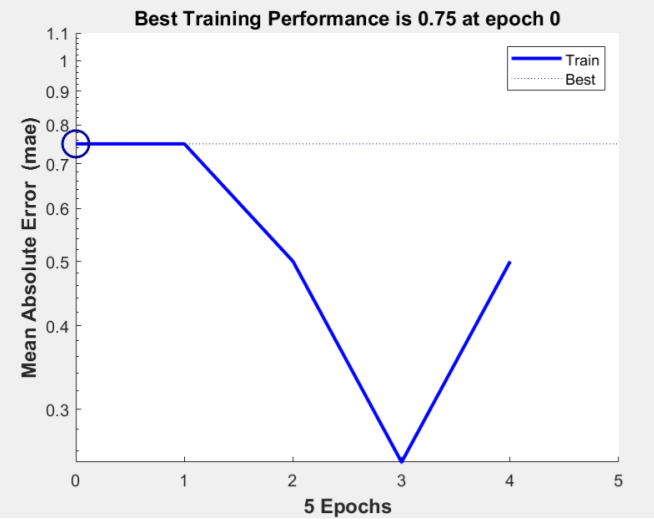
|  |
| --- |
|  |
|  |  |
|  | net=newp([0 1; -2 2],1); % tworzy perceptron składajacy sie z 2 elementow, |
|  | % gdzie pierwszy w zakresie (0,1), drugi(-2,2) i jeden neuron. |
|  |  |
|  | W=[0 0 1 1; 0 1 0 1]; |
|  | T=[0 0 0 1]; |
|  | % parametry |
|  |  |
|  | net=init(net); |
|  | % inicjalizacja dowolnymi parametrami |
|  |  |
|  | sym\_przed=sim(net,W) |
|  | % symulacja w SiMULiNK |
|  | % pierwszy argument to model |
|  | % a drugi parametry |
|  |  |
|  | net.trainParam.epochs=10; |
|  | % maksymalna wartość iteracji to 20; |
|  | net=train(net,W,T); |
|  | % uczenie sieci |
|  | sym\_po = sim(net,W) |

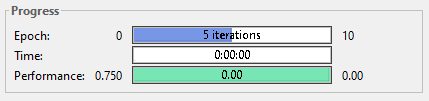
1. **Wnioski:**

1.

W=[0 0 1 1; 0 1 0 1];

T=[0 0 0 1];

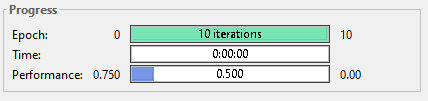




2.

W=[0 1 1 1; 0 1 0 1];

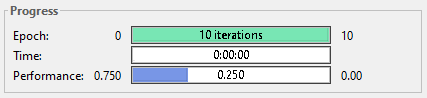
T=[0 0 0 1];



3.

W=[1 1 1 1; 0 1 0 1];

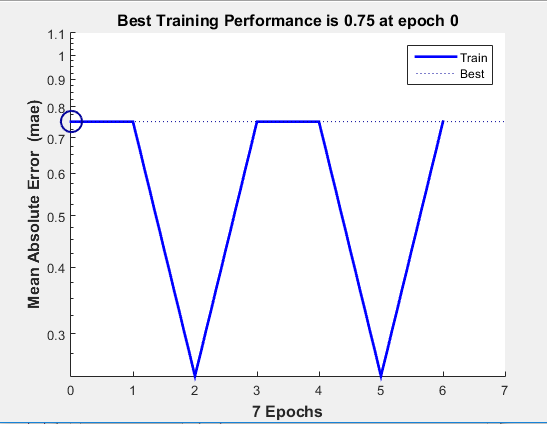
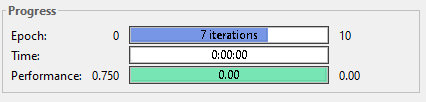
T=[0 0 0 1];



4.

W=[0 0 1 1; 1 1 0 1];

T=[0 0 0 1];



Stopień nauczania perceptronu w całości zależy od programisty – ilu iteracji użyje w swoim kodzie. Im więcej iteracji tym błąd jest mniejszy, więc programista sam musi ustalić jakiej dokładności potrzebuje. Z powyższych przykładów widać jednak, że jest to bardzo efektywne i wystarczy niewiele iteracji, jednakże można tak ustalić parametry by wykorzystywały pełen zakres iteracji.

W zależności od bramki zmienia się również czas.

Źródło: wikipedia.org, dokumentacja Matlab, mathworks.com