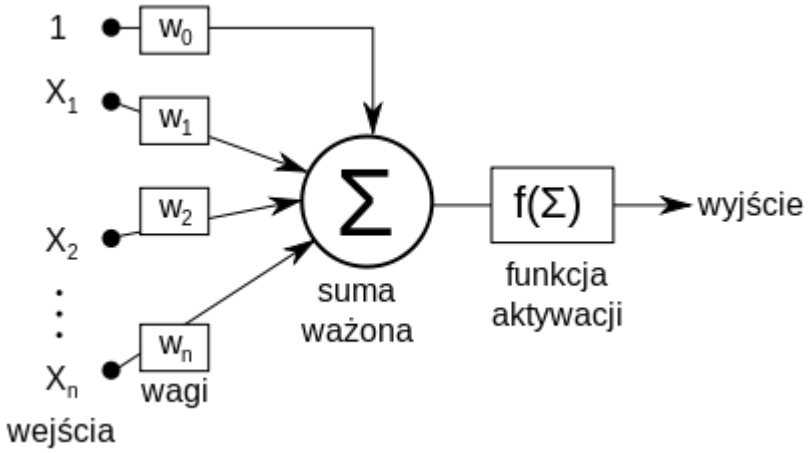
Tomasz Tomala
Podstawy Sztucznej Inteligencji
Sprawozdanie z projektu nr 2

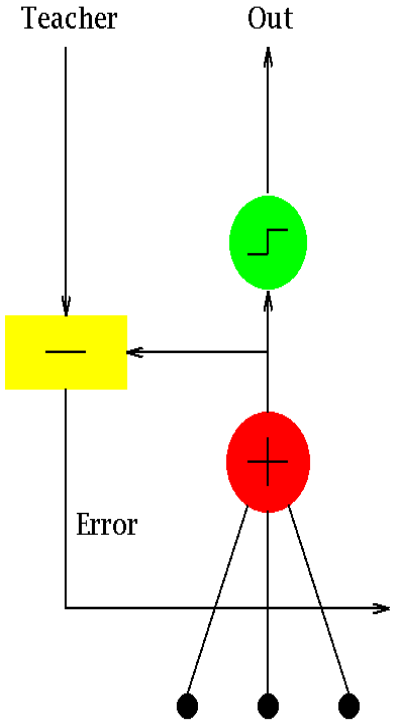
Celem ćwiczenia buło poznanie budowy i działania jednowarstwowych sieci neuronowych oraz uczenie rozpoznawania wielkości liter.

1) Syntetyczny opis budowy wykorzystanego algorytmu uczenia:

Aby wykonać ćwiczenie stworzyłem dwie jednowarstwowe sieci – każdą z wykorzystaniem innego algorytmu. Pierwsza z nich korzysta z modelu perceptronu McCullocha-Pittsa, druga z kolei wykorzystuje neuron typu Adaline.

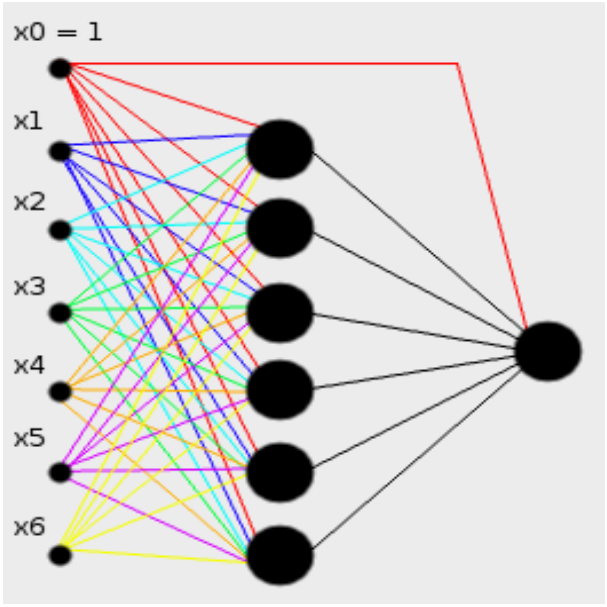


Ilustracja 1: Model perceptronu McCullocha-Pittsa



Ilustracja 2: Model neuronu Adaline

Obie sieci składają się z siedmiu neuronów. Sześć pierwszych stanowi pierwszą warstwę i przesyła sygnały wyjściowe do siódmego wyjściowego neuronu. Każdy z neuronów otrzymuje po siedem sygnałów wejściowych. Sposób w jaki wszystko zostało połączone przedstawiam na poniższej grafice:



Ilustracja 3: Graficzne zilustrowanie połaczeń w sieci

Do nauki perceptronów oraz adaline wykorzystałem algorytm Widrow-Hoffa.

Do budowy perceptronu wykorzystałem podany na wykładzie model McCullocha-Pittsa. Zaimplementowana przeze mnie klasa Perceptron składa się z trzech metod: active, process oraz learn.

Metoda ***active*** wykorzystuje unipolarną funkcję progową która zwraca wynik 0 lub 1 dla podanego argumentu.

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

Metoda ***process*** sumuje iloczyn sygnałów wejściowych i odpowiadających im wag. Uruchamia metodę activate z otrzymaną sumą iloczynów jako parametrem, oraz zwraca wynik tej metody.

$$\sum_{i=1}^{m} w_i x_i$$

Metoda ***learn*** wywołuje metodę process dla otrzymanych wejść jako parametrów, po czym na podstawie otrzymanego wyniku modyfikuje wszystkie wagi dla odpowiednich wejść.

$$\mathbf{W_i = W_i + (y - y') * x_i * \alpha}$$

Budowa adaline jest bardzo podobna do budowy perceptronu. Różni się tylko tym, że modyfikowanie wag wykonuje się z pominięciem funkcji aktywacji. Zaimplementowana przeze mnie klasa Adaline składa się z czterech metod: active, process, learn oraz test.

Metoda ***active*** wykorzystuje unipolarną funkcję progową która zwraca wynik -1 lub 1 dla podanego argumentu.

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ -1 & \text{otherwise} \end{cases}$$

Metoda ***process*** sumuje iloczyn sygnałów wejściowych i odpowiadających im wag, oraz zwraca ten wynik.

$$\sum_{i=1}^{m} w_i x_i$$

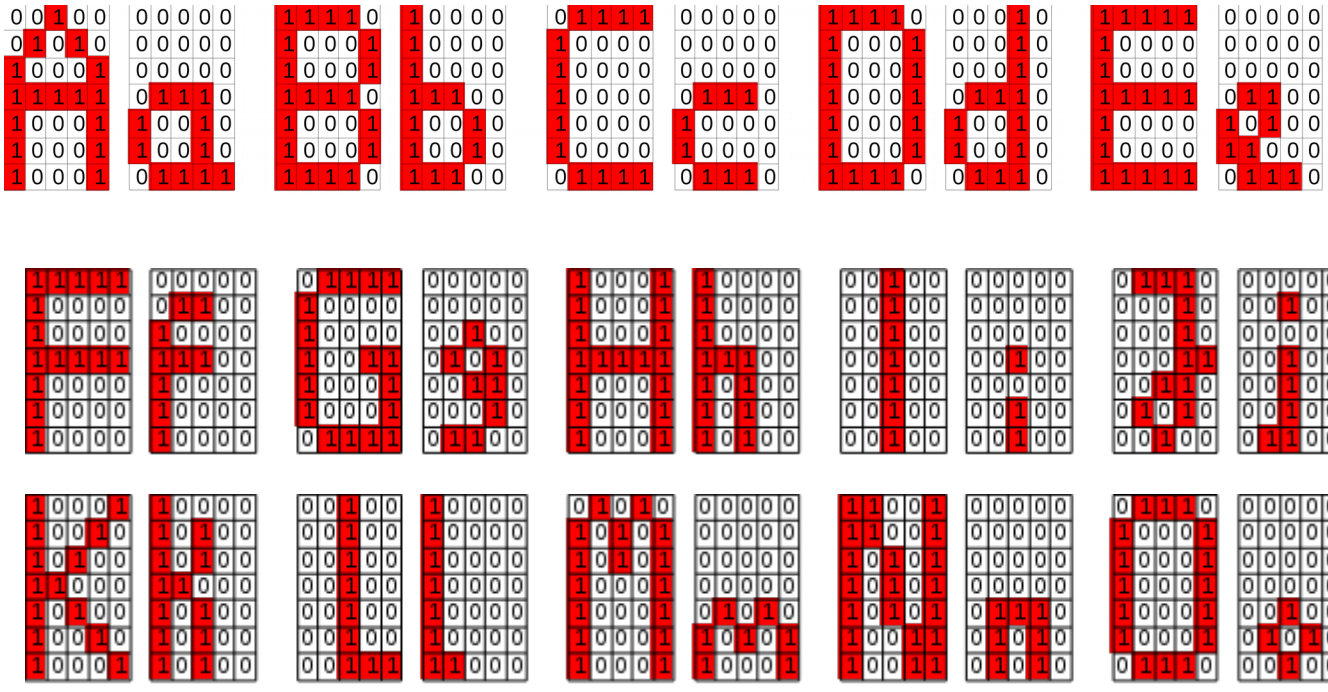
Metoda ***learn*** wywołuje metodę process dla otrzymanych wejść jako parametrów, po czym na podstawie otrzymanego wyniku modyfikuje wszystkie wagi dla odpowiednich wejść.

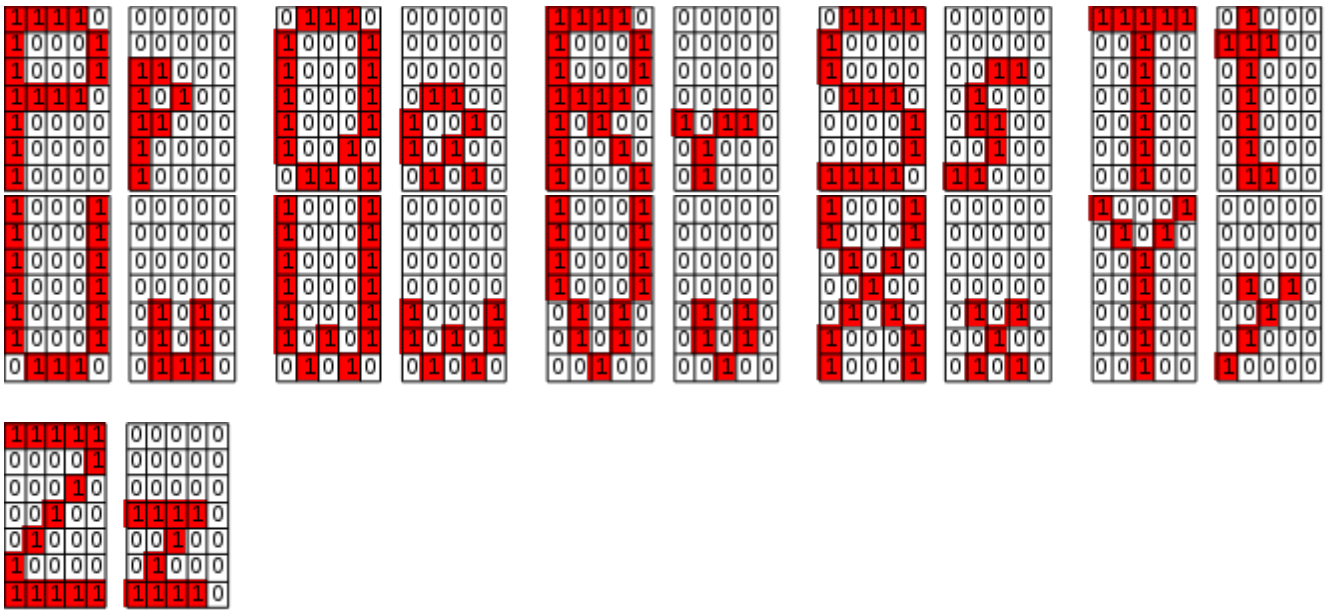
$$\mathbf{W_i = W_i + (y - y') * x_i * \alpha}$$

Metoda ***test*** uruchamia metodę active z wynikiem metody process jako parametrem.

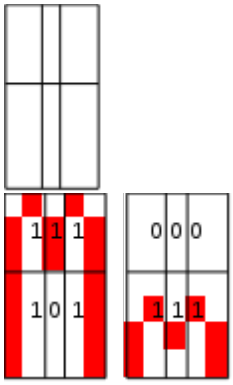
2) Zestawienie otrzymanych wyników:

Jako dane uczące i testujące wykorzystałem własnoręcznie stworzone litery alfabetu. Są one przedstawione jako dwuwymiarowa tablica o rozmiarach 7 x 5. Przedstawiam je poniżej:





Tablicę 7 x 5 podzieliłem na 6 podobszarów. Każdy z nich to jeden sygnał wejściowy do neuronów. Jeżeli w danym obszarze pojawił się choćby jeden piksel z litery to obszar zwracał sygnał 1, w przeciwnym wypadku zwracał sygnał 0. Poniżej sposób podziału tablicy na obszary, oraz przykład wyniku dla dużej i małej litery „M”:



3) Analiza i dyskusja błędów uczenia i testowania opracowanego perceptronu w zależności od wartości współczynnika uczenia oraz liczby danych uczących:

4) Sformułowanie wniosków:

5) Listing całego kodu

Bibliografia:

https://pl.wikipedia.org/wiki/Neuron_McCullocha-Pittsa
https://en.wikipedia.org/wiki/ADALINE
https://en.wikipedia.org/wiki/Perceptron