**Podstawy Sztucznej Inteligencji – Laboratorium nr 2**

Wykonał: Kamil Wieniecki

Temat ćwiczenia: Budowa i działanie sieci jednowarstwowej

1. Cel ćwiczenia

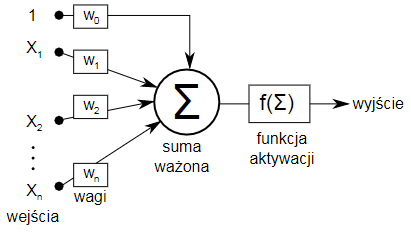
Celem ćwiczenia jest poznanie budowy i działania jednowarstwowych sieci neuronowych oraz uczenie rozpoznawania wielkości liter

1. Realizacja ćwiczenia

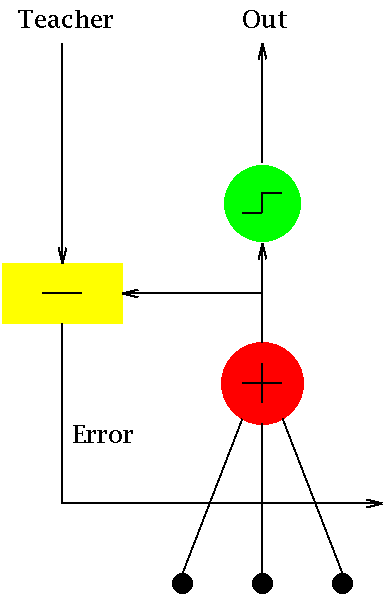
Wybrany przeze mnie język programowania do realizacji laboratorium to **Python**.

Do realizacji ćwiczenia stworzyłem dwie sieci jednowarstwowe wykorzystując z modelu neuronu McCullocha – Pittsa, natomiast druga implementacja to Adaline.

Graficzne przedstawienie modelu McCullocha – Pittsa:



Graficzne przedstawienie modelu Adaline:



Nauka sieci opiera się na algorytmie Widrowa – Hoffa.

Progowa funkcja aktywacji dla modelu McCullocha-Pittsa wygląda następująco:



Natomiast dla modelu Adaline:



Metoda sumowania sygnałów wejściowych wygląda następująco:

y = ∑ wixi

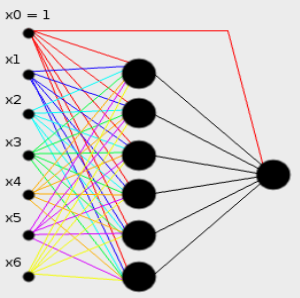
gdzie wi – waga, xi – sygnał wejściowy.

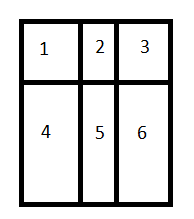
Metoda uczenia opisana jest w następujący sposób:

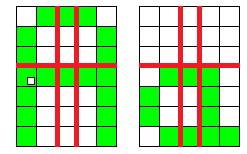
Wi = wi + (y – y’) \* learning\_rate \* xi

Gdzie w – dana waga, y –wielkość litery (0, 1), y’ – wielkość litery „nauczona”, learning\_rate – współczynnik szybkości nauki, xi – sygnał wejściowy.  
Budowa Adaline i Perceptronu niewiele się różni tzn. głównie różnica jest przy funkcji aktywacji oraz Adaline ma dodatkowo metodę test, która zwraca funkcję aktywacji z parametrem sumowania wag (jako argument).

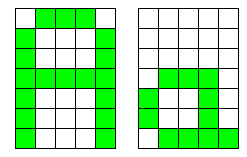
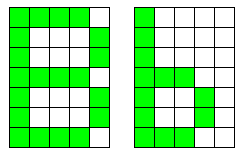
Sieci składają się z siedmiu neuronów, 6 neuronów wysyła sygnały wyjściowe do neuronu nr 7 jako do neuronu wyjściowego. Każdy neuron dostaje 7 sygnałów wejściowych (pierwszy to bias, pozostałe to obliczenia czy w danym sektorze znajduje się fragment litery). Schemat połączeń:

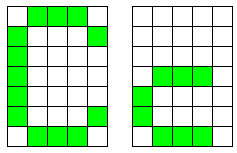
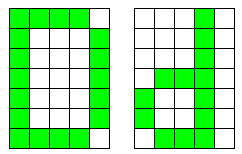
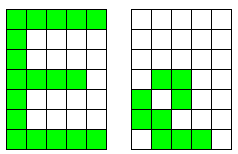
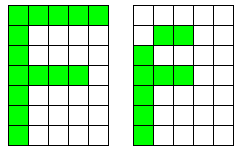
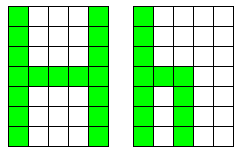
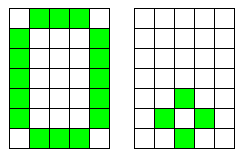


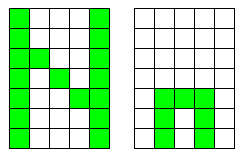
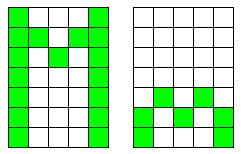
Przygotowany przeze mnie zestaw danych do nauki sieci to macierze 5x7 - 10 dużych i 10 małych liter alfabetu. Każda litera została podzielona na ponumerowane obszary wg. schematu:



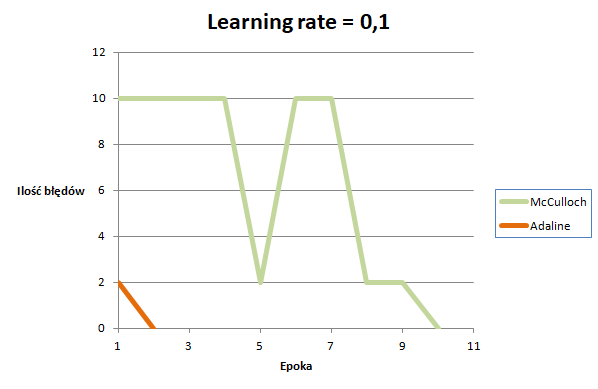
Czerwonymi liniami został zaznaczony rzeczywisty podział na przykładzie liczby „A” i „a”.

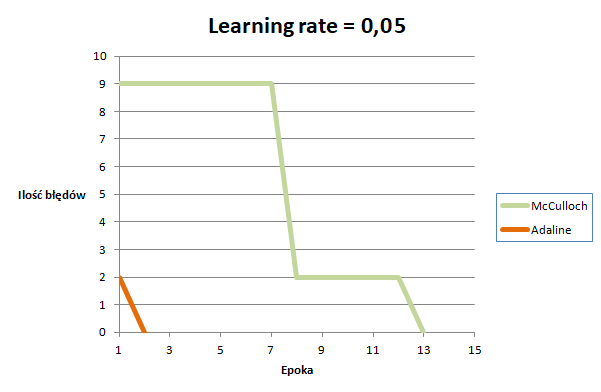
Schematy liter:

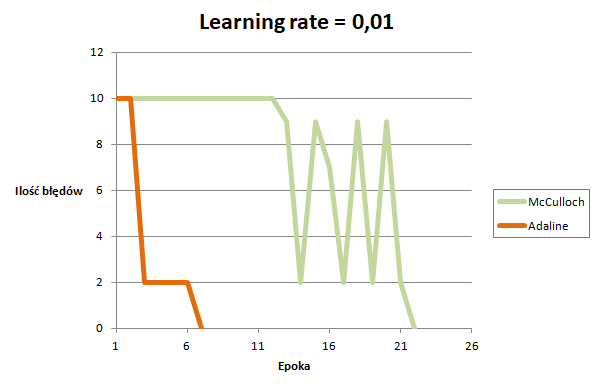
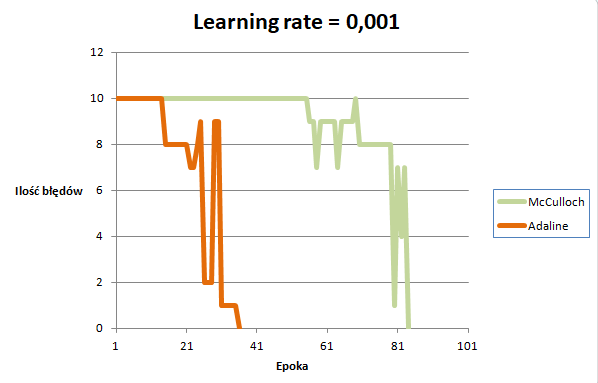
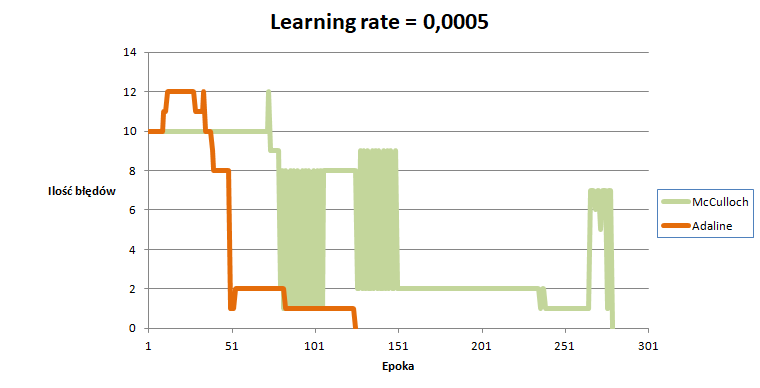




Zielone obszary zostały opisane wartością 1, natomiast białe (puste) wartością 0.

1. Wyniki:





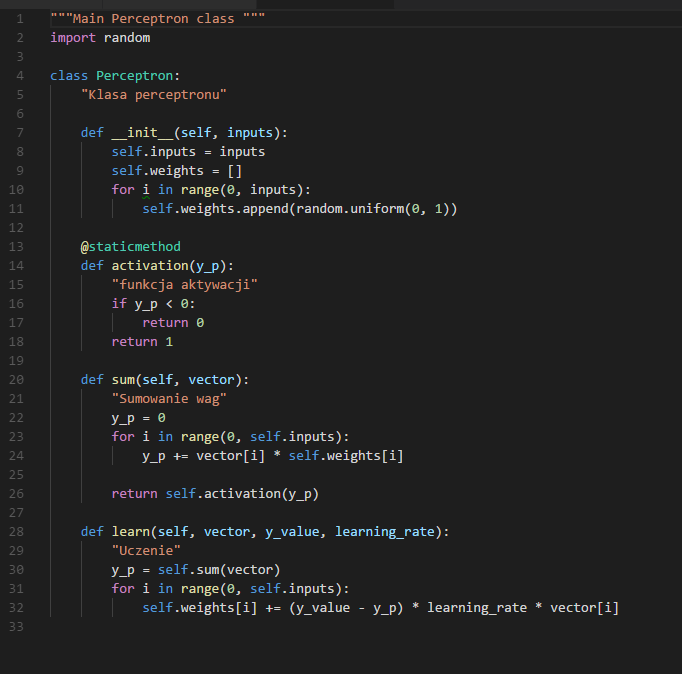
1. Analiza wyników

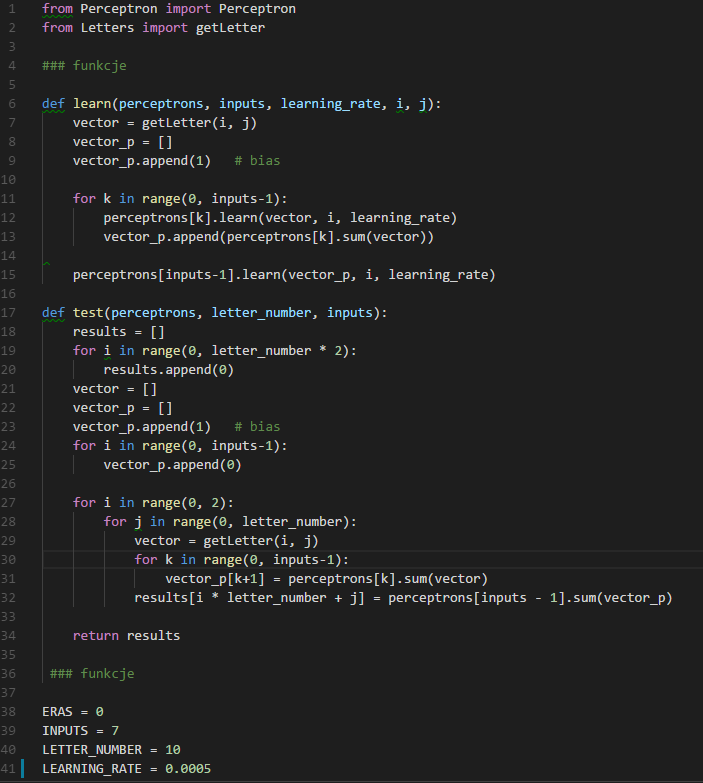
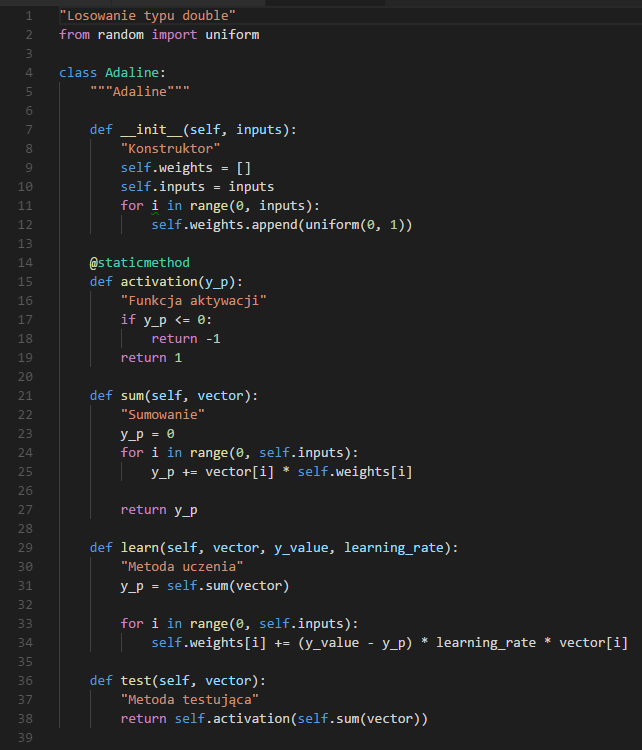
Jak wynika z wykresów powyżej współczynnik uczenia znacząco wpływa na szybkość uczenia się. Im większa wartość learning\_rate tym szybciej przebiega nauka. Model Adaline jest również dużo bardziej wydajny (od 2 do 10-krotnie) od modelu McCullocha-Pittsa.

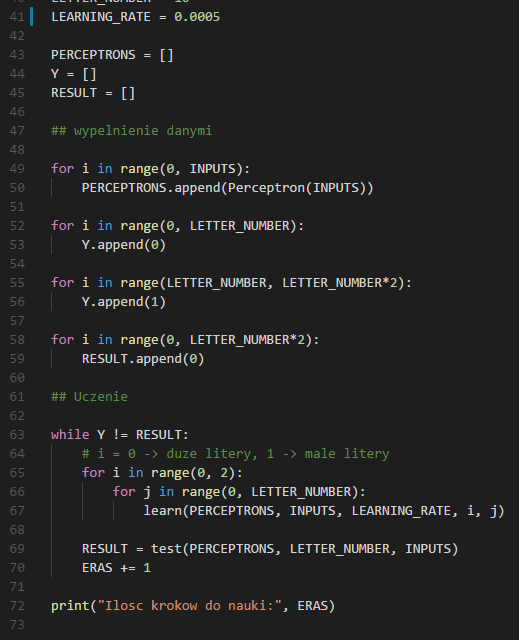
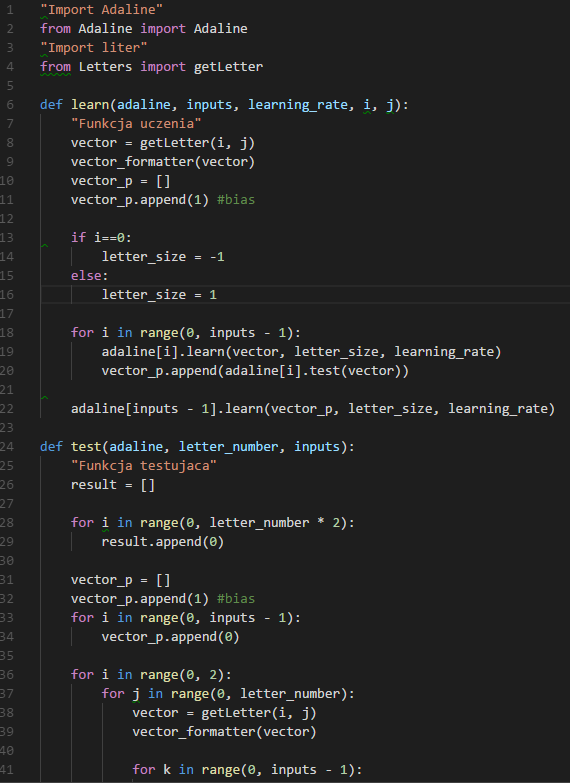
Dla wysokich wartości learning rate Adaline potrzebował zaledwie 1-2 epoki do nauki przy każdej próbie. Przy czym dla tej samej wartości McCulloch-Pitts potrzebował około 10 epok, aby nauczyć się prawidłowo rozróżniać litery. Dla wartości 0,01 Adaline potrzebował 7 epok do nauczenia się, a McCulloch aż 22. Następnie dla 0,001 Adaline 36, McCulloch – 84, dla 0,0005 -> Adaline – 125 epok, McCulloch – 279.

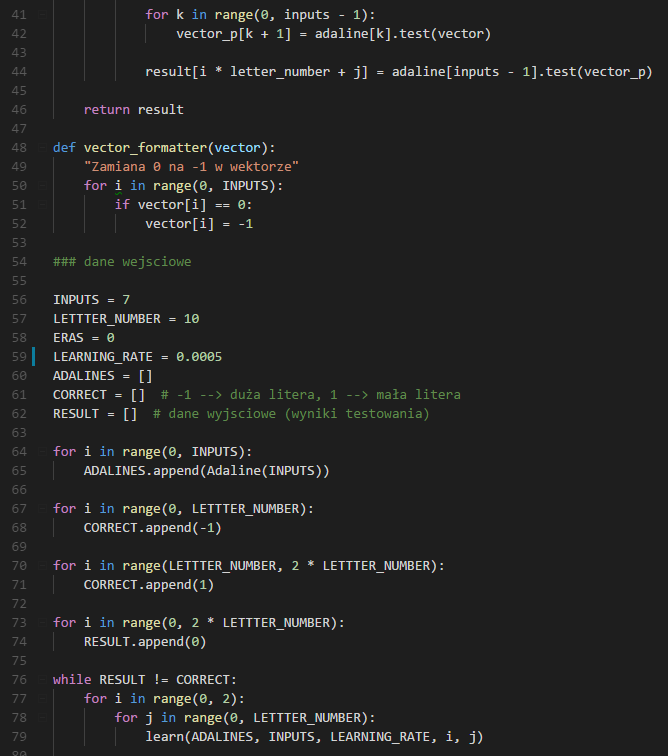
1. Wnioski

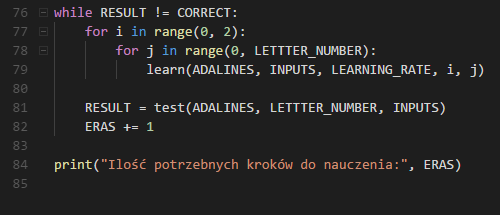
Model Adaline jest znacznie wydajniejszy i efektywniej się uczy. Zestawiając różne współczynniki uczenia zawsze był w stanie nauczyć się znacznie szybciej od modelu McCullocha-Pittsa. Możliwość zmniejszania wartości wag w modelu Adaline oraz funkcja aktywacji dla wartości -1 i 1 zwiększa efektywność nauki neuronu. Wprowadzane dane testowe również mają ogromne znaczenie (empirycznie przekonałem się, że budowa litery „L” oraz „b” jest dla modelów prawie tym samym przez co nie można było rozpoznać, czy wprowadzona litera jest duża, czy też nie). W przypadku niepoprawnego wprowadzenia danych możliwe jest zapętlenie programu w nieskończoność.

1. Listing kodu

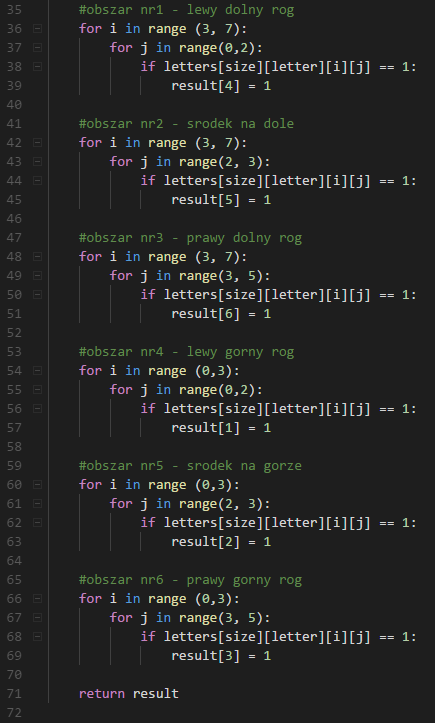












1. Bibliografia

https://pl.wikipedia.org/wiki/Neuron\_McCullocha-Pittsa

https://en.wikipedia.org/wiki/ADALINE