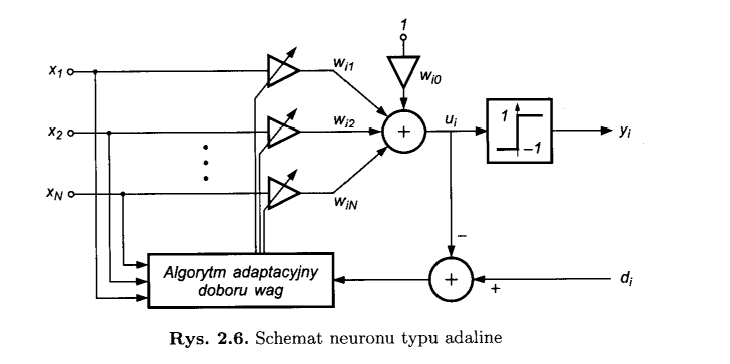
Anna Budzoń

Sprawozdanie nr 2

1. Cel ćwiczenia:
2. Zapoznanie się z algorytmami uczenia się sieci jednowarstwowej – Adaline i MCP (neuron sigmoidalny).
3. Przeprowadzenie uczenia dla sztucznego neuronu – rozróżnianie dużych i małych liter.
4. Algorytm Adaline:
5. Adaline (ang. Adaptive Linear Neuron) – oparty jest na algorytmie Widrowa-Hoffa
6. Schemat neuronu:



1. Realizacja:

- określenie losowych wag z zakresu (0,1) i ustawienie początkowo bias = 1,

- wyliczenie sumy wagowej sygnałów wejściowych,

- wyliczenie zmiennej *delta*, czyli różnicy między wartością oczekiwaną, a otrzymaną (wykorzystujemy tutaj wyliczoną wcześniej sumę – funkcja liniowa)

, gdzie:

di- wartość oczekiwana

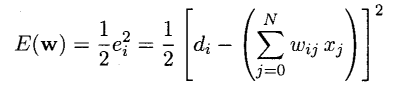
- bipolarnej funkcji aktywacji nie wykorzystujemy w procesie uczenia się,

- dyskretny sposób uaktualniania wag:



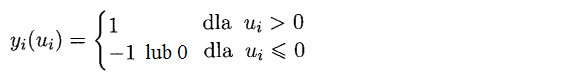
η – współczynnik uczenia się

- zdefiniowanie funkcji celu:



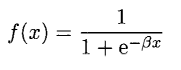
- proces uczenia neuronu Adaline polega na minimalizacji funkcji celu – uaktualniamy wagi aż wyliczona wartość wyjściowa jest odpowiednio małym przybliżeniem aktualnego wyniku, czyli aż error\_MSE (błąd średniokwadratory) będzie dążył do zera. Możemy tu przyjąć określoną, odpowiednio małą liczbę, będącą warunkiem zakończenia procesu uczenia.

- Funkcja aktywacji jest dopiero wykorzystywana przy określaniu wartości wyjściowej w procesie testowania neuronu adaline przy zadanych wartościach wejściowych.

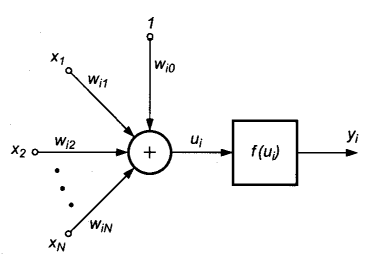


1. Algorytm neuronu sigmoidalnego:
   1. Neuron typu sigmoidalnego wykazuje strukturę podobną do modelu McCullocha-Pittsa.

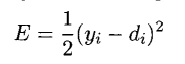
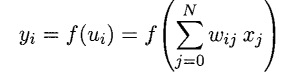
Różnicą jest funkcja aktywacji, która jest liniowa i przyjmuje postać:

, gdzie β jest ustalonym z góry parametrem, wpływającym na kształt funkcji aktywacji. Najczęściej przyjmujemy, że wynosi on 1.

* 1. Schemat neuronu:



* 1. Uczenie odbywa się z nauczycielem poprzez minimalizację funkcji celu:

, gdzie 

ui – suma wagowa sygnałów wejściowych,

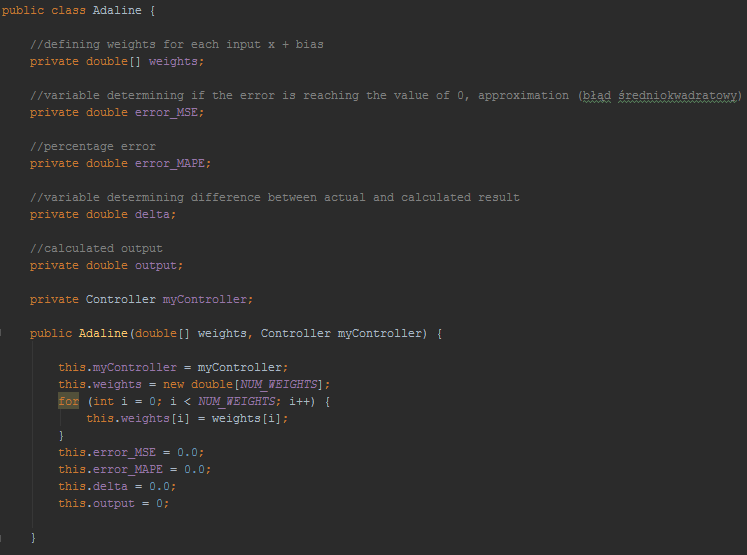
f(ui) – funkcja aktywacji od ui

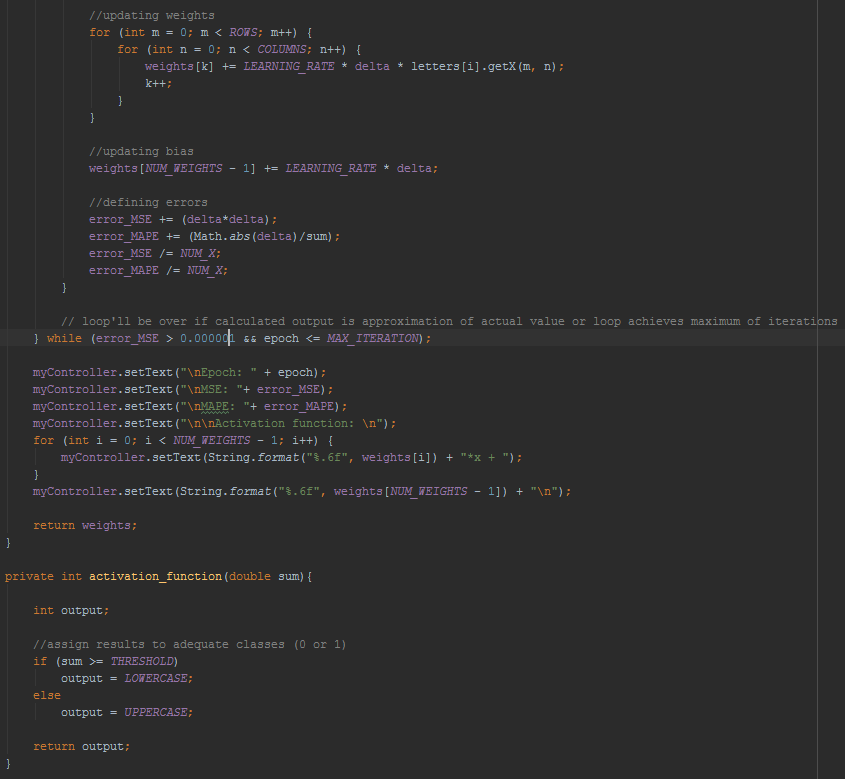
di – wartość zadana na wyjściu

- Uaktualnianie wag:

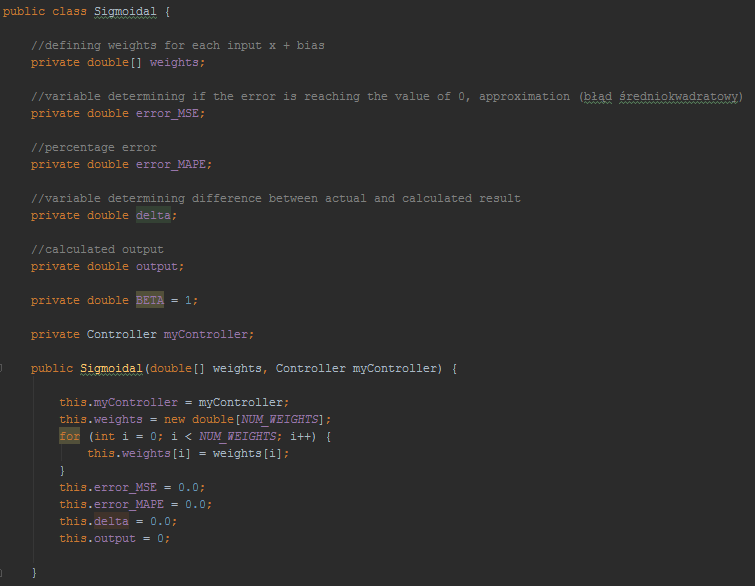


1. Listing programu:
   1. Adaline:

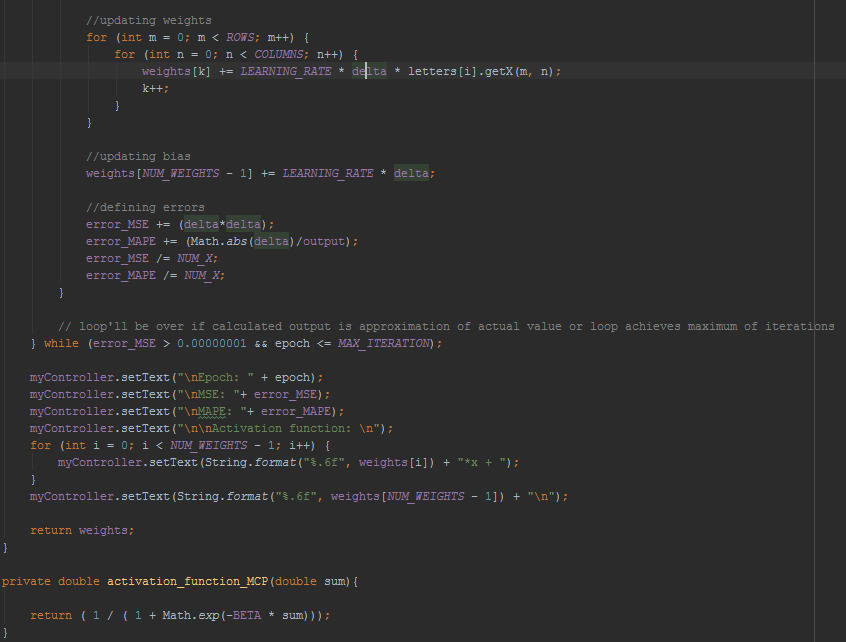




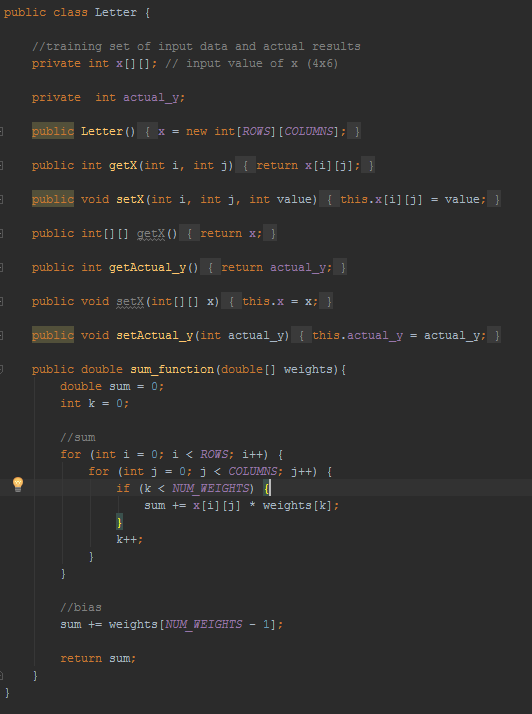
* 1. Sigmoidal:



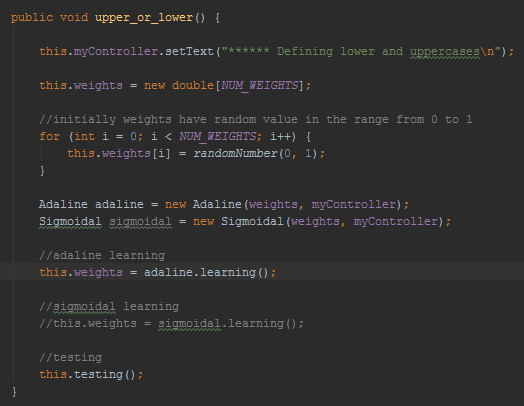




* 1. Letter:



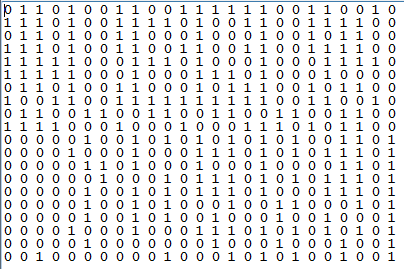
* 1. Main:



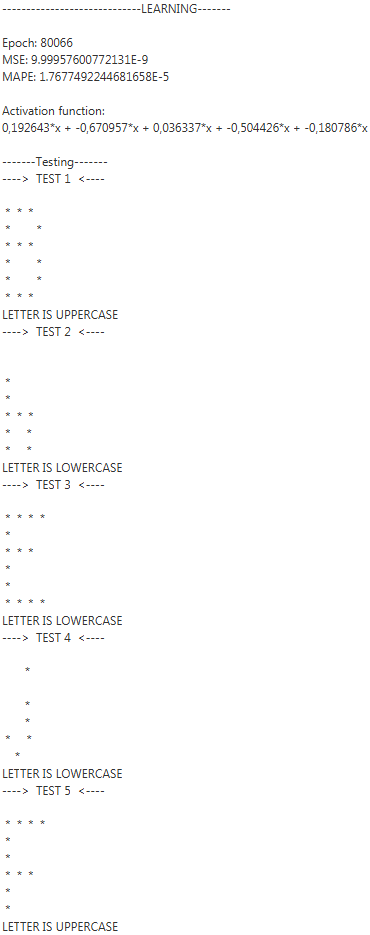
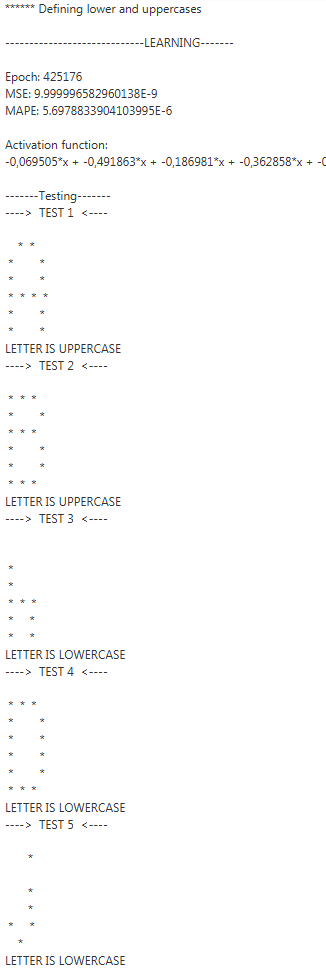
* 1. Testing:

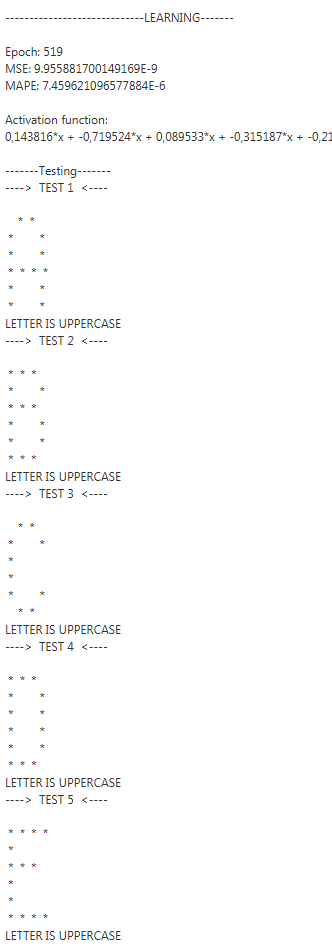
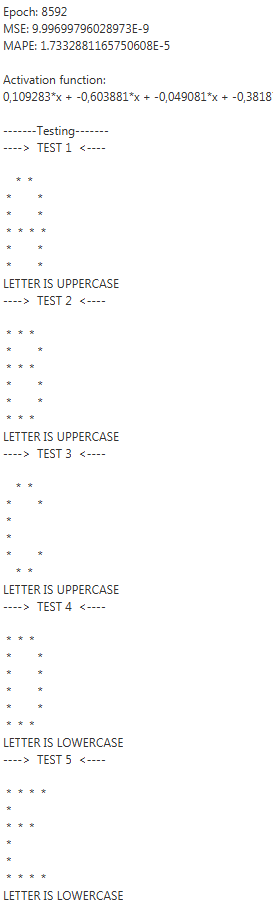


* 1. Learning set – litery A-J, a-j:

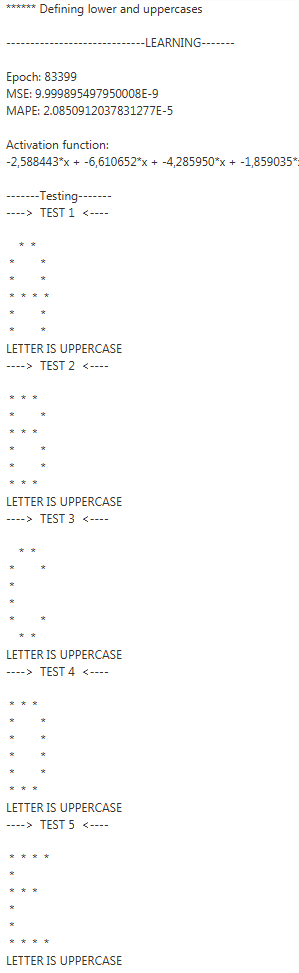
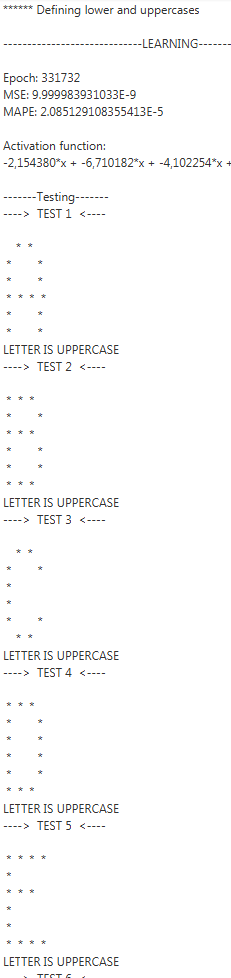


1. Przykładowe wyniki:
   1. Adaline:





* 1. Neuron sugmoidalny:



1. Wyniki i wykresy:
   1. Adaline:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ADALINE | LEARNING\_RATE | EPOCHS | CORRECT RESULTS | MSE (\*10^(-9)) | MAPE (\*10^(-5)) |
|  | 0.0001 | 425176 | 80,0% | 9.999 | 0.5698 |
|  | 0.0005 | 142827 | 90,0% | 9.99985 | 1.747 |
|  | 0.00075 | 66617 | 90,0% | 9.99972 | 1.767 |
|  | 0.001 | 80066 | 90,0% | 9.99958 | 1.768 |
|  | 0.0025 | 35594 | 90,0% | 9.99972 | 1.766 |
|  | 0.005 | 20012 | 75,0% | 9.99797 | 1.756 |
|  | 0.0075 | 4118 | 75,0% | 9.9952 | 2.01 |
|  | 0.01 | 8592 | 75,0% | 9.997 | 1.733 |
|  | 0.05 | 984 | 85,0% | 9.987 | 1.042 |
|  | 0.075 | 519 | 90,0% | 9.9559 | 7.4596 |
|  | 0.1 | 534 | 90,0% | 9.8995 | 1.975 |

* 1. Neuron sigmoidalny

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sigmoidal | LEARNING\_RATE | EPOCHS | CORRECT RESULTS | MSE (\*10^(-9)) | MAPE (\*10^(-5)) |
|  | 0.0001 | 8287044 | 100,0% | 9.99999 | 2.0851 |
|  | 0.0005 | 1641480 | 100,0% | 9.999989 | 2.085 |
|  | 0.00075 | 1108288 | 100,0% | 9.99999 | 2.085 |
|  | 0.001 | 829919 | 100,0% | 9.99999 | 2.085 |
|  | 0.0025 | 331732 | 100,0% | 9.99998 | 2.0851 |
|  | 0.005 | 165669 | 100,0% | 9.99998 | 2.0848 |
|  | 0.0075 | 111760 | 100,0% | 9.99987 | 2.0847 |
|  | 0.01 | 83399 | 100,0% | 9.999895 | 2.08509 |
|  | 0.05 | 16604 | 100,0% | 9.999957 | 2.08488 |
|  | 0.075 | 11156 | 100,0% | 9.9999 | 2.0847 |
|  | 0.1 | 8361 | 100,0% | 9.99837 | 2.08452 |

1. Analiza wykresów:
   1. Adaline:

* Testowanie przeprowadzono z różnymi współczynnikami uczenia. Na podstawie stworzonego wykresu można zaobserwować, że im większy learning rate, tym mniej uzyska się epok uczenia. Przy bardzo małej wartości tego współczynnika (0.0001) sieć potrzebowała aż 425 tysiące iteracji. Natomiast przy współczynniku uczenia równym 0.1, jedynie 534.
* Możemy również zauważyć, że sztuczny neuron uczony poprzez algorytm Adaline popełniał sporo pomyłek. Błędne wyniki stanowiły aż 15,5 % wszystkich wartości wyjściowych. Najczęściej błędne odpowiedzi dotyczyły wielkich liter, takich jak: E, I, D czy J. Jeśli chodzi o małe litery, nie zarejestrowano błędnych wyników.
  1. Neuron sigmoidalny:
* Podobnie jak w Adaline, wraz ze wzrostem współczynnika uczenia się, malała liczba epok, które neuron potrzebował do nauki. Możemy zaobserwować, że w tym algorytmie sieć potrzebowała zaskakująco dużo iteracji, by zoptymalizować wagi. Nawet przy bardzo dużym współczynniku liczba epok wyniosła aż 8361.
* Algorytm ten jednak okazał się nieomylny, ponieważ nie popełnił on ani jednego błędu w kwalifikacji danych wejściowych. Poprawność wyników wyniosła 100%.

1. Wnioski:
   1. Znacząca jest różnica pomiędzy ilością epok dla Adaline, a dla neuronu sigmoidalnego. Neuron sigmoidalny przy współczynniku uczenia 0.0001 potrzebował w przybliżeniu 8,29 mln iteracji, a Adaline potrzebowała tylko 425 tysięcy- jest to aż 19,5 razy mniej. Przy współczynniku 0.1 liczba epok dla Adaline spadła do 534, a dla sigmoidalnego neuronu nie była mniejsza niż 8 tysiący.

Przy takich wynikach nasuwa się jednoznaczny wniosek – sieć neuronowa Adaline uczy się zdecydowanie szybciej od sieci sigmoidalnej.

* 1. Na podstawie otrzymanych wyników możemy wysunąć również inny, bardzo oczywisty wniosek – bardziej niezawodnym sposobem uczenia sieci neuronowej jest algorytm sigmoidalny. Neuron sigmoidalny nie popełnił ani jednego błędu w żadnym procesie uczenia się, niezależnie od wartości współczynnika uczenia się. Przy drugiej sieci natomiast pojawiały się liczne błędy, wynoszące ogółem 15,5% wszystkich wyników, mimo że dokładność oszacowania wyników była większa niż w drugim algorytmie (mniejszy błąd procentowy MAPE). Oba algorytmy zakończyły proces uczenia się przy takim samym błędzie średniokwadratowym MSE.
  2. Neuron sigmoidalny - Biorąc pod uwagę znaczący spadek liczby epok (nawet 8 razy), możemy stwierdzić, że współczynnik uczący ma zasadniczy wpływ na szybkość uczenia się. Im większy określimy learning-rate, tym szybciej sieć neurowa nauczy się.
  3. Adaline – w tym algorytmie nie zaobserwowano aż tak drastycznych spadków ilości epok zależnie od współczynnika uczącego, jak w drugim algorytmie, jednak wykres liniowy jest w większości malejący. Dla małego współczynnika ilość epok jest przybliżona.