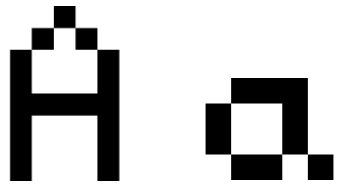
PSI-Scenariusz 2

Kacper Pawlikowski

IS. Gr.3

Dane wejściowe:

Po 10 małych i dużych liter na matrycy 8x7 w postaci plików csv składających się ze znaków -1, 1 . Przykładowo:



Dane wyjściowe:

-1, 1 w zależności od neuronu i czy na wejściu jest duża/mała litera

Zastosowałem sieć jednowarstwową składającą się z dwóch Perceptronów McCullocha-Pittsa z 56 wejściami. Jeden Perceptron odpowiada za rozpoznawanie wielkich liter natomiast drugi rozpoznaje małych.

Działanie perceptronu:

Dla każdego wejścia x_i perceptronu przypisana jest waga w_i. Dla stanów wejściowych liczymy sumę ważoną:

$$s=\sum_{i=1}^{n} x_i w_i + b$$

b-wartość odchylenia, odpowiada za nieliniowe przekształcenie wejść w wyjście

Funkcję progową unipolarną w postaci:

$$y = \begin{cases} -1 \; dla \; s < 0 \\ 1 \; dla \; s \geq 0 \end{cases}$$ y-wyjście neuronu

Oraz funkcję liniową:

$$a*x+b$$

użyłem jako funkcje aktywacji.

Algorytm uczenia:

Skorzystałem z następującego algorytmu uczenia dla pojedynczego neuronu:

- Początkowe wagi zostały wylosowane z zakresu <-0.5, 0.5>
- Sprawdzam czy na podstawie przygotowanych danych wejściowych otrzymam oczekiwany wynik. Jeżeli nie:
    - Obliczam błąd: e=uzyskany_wynik - oczekiwany_wynik
    - Modyfikuję wagi:
      Waga=Waga+współczynnik_uczenia*e*dana_wejściowa, oraz b=b+ współczynnik_uczenia *e
- Procedurę powtarzam dla wszystkich przygotowanych zestawów danych

- Dla każdego z Neuronów (kolejność użycia zestawów jest losowa) a następnie sprawdzam błąd średniokwadratowy:

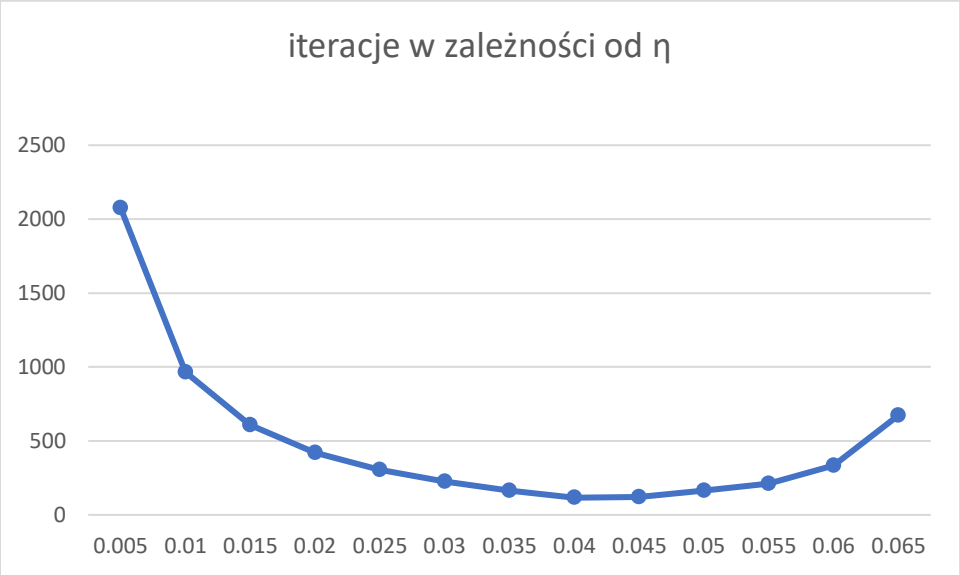
$$E=\frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{k}(d_{i,j}-y_{i,j})^2$$

p-liczba przykładów do nauki
$d_{i,j}$-oczekiwana odpowiedź dla i-tego przykładu na j-tym wyjściu
$y_i j$-uzyskana odpowiedź

- Jeżeli e>0 to powtarzam proces uczenia

Sieć przetestowałem dla kilku funkcji aktywujących:

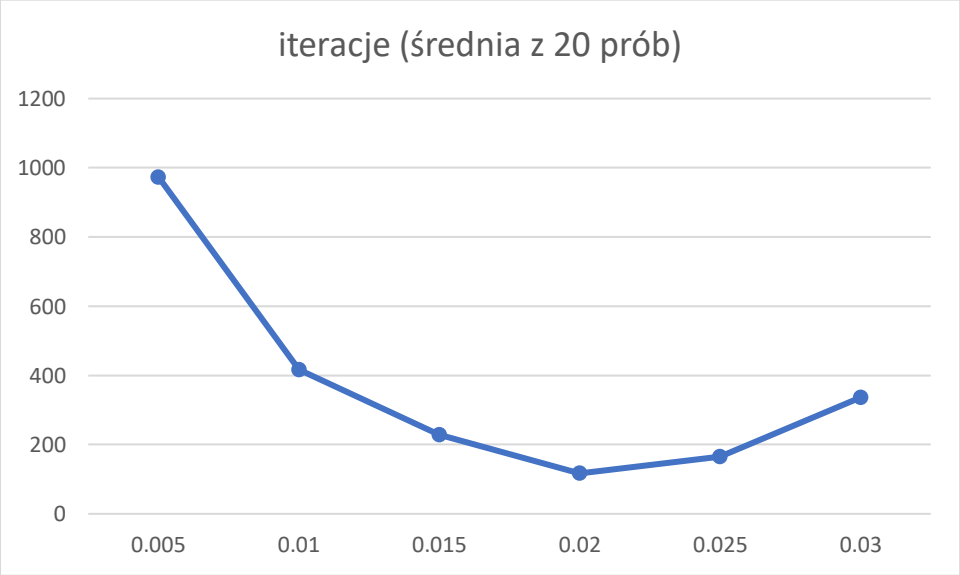
1. **Liniowa funkcja aktywująca w postaci 0.5x:**
   Każdy przedstawiony na wykresie pomiar to wartość średnia obliczona na podstawie 20 prób.



```
numer iteracji: 2624   blad sredniokwadratowy: 2.66853e+306
numer iteracji: 2625   blad sredniokwadratowy: 3.74129e+306
numer iteracji: 2626   blad sredniokwadratowy: 5.16685e+306
numer iteracji: 2627   blad sredniokwadratowy: 7.01001e+306
numer iteracji: 2628   blad sredniokwadratowy: 9.32901e+306
numer iteracji: 2629   blad sredniokwadratowy: 1.21701e+307
numer iteracji: 2630   blad sredniokwadratowy: 1.55647e+307
numer iteracji: 2631   blad sredniokwadratowy: 1.9532e+307
numer iteracji: 2632   blad sredniokwadratowy: 2.40935e+307
numer iteracji: 2633   blad sredniokwadratowy: 2.9304e+307
numer iteracji: 2634   blad sredniokwadratowy: 3.53104e+307
numer iteracji: 2635   blad sredniokwadratowy: 4.24436e+307
numer iteracji: 2636   blad sredniokwadratowy: 5.13561e+307
numer iteracji: 2637   blad sredniokwadratowy: 6.32101e+307
numer iteracji: 2638   blad sredniokwadratowy: 7.99193e+307
numer iteracji: 2639   blad sredniokwadratowy: inf
numer iteracji: 2640   blad sredniokwadratowy: inf
numer iteracji: 2641   blad sredniokwadratowy: inf
numer iteracji: 2642   blad sredniokwadratowy: inf
numer iteracji: 2643   blad sredniokwadratowy: inf
```

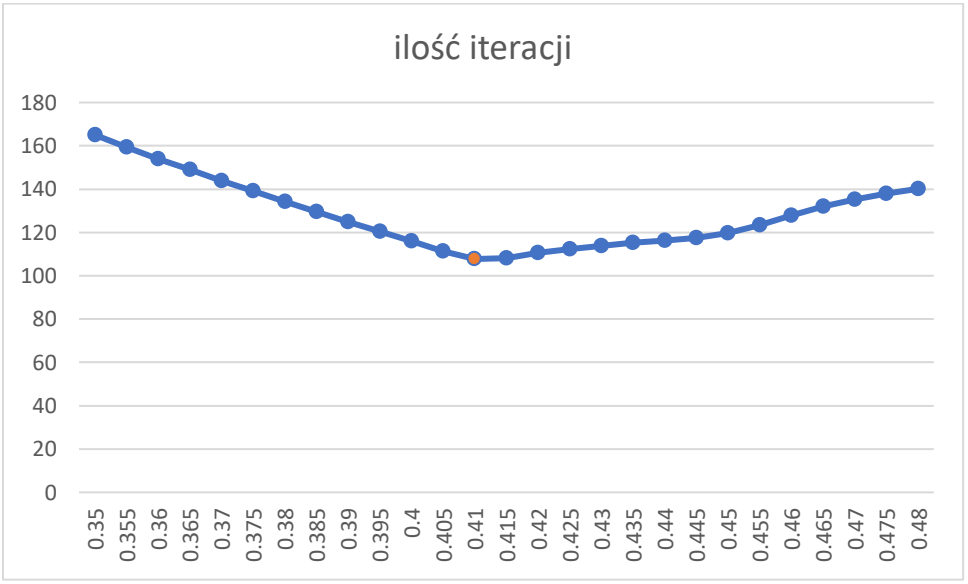
Przy większych wartościach współczynnika uczenia (>0.065) sieć przestawała się uczyć. Błąd średniokwadratowy rósł bardzo szybko poza zakres obsługiwany przez program.

2. **Liniowa funkcja aktywująca w postaci 1x:**
   Każdy przedstawiony na wykresie pomiar to wartość średnia obliczona na podstawie 20 prób.



Podobnie Jak w poprzednim przypadku powyżej pewnej wartości współczynnika uczenia (>0.03)program przestawał działać.

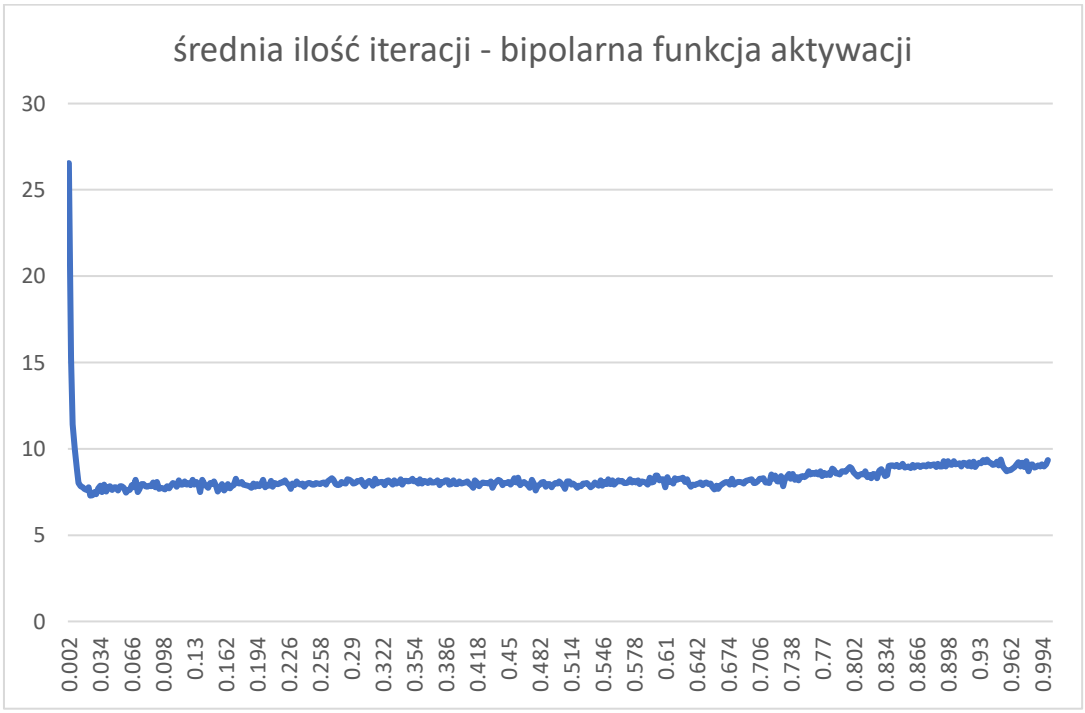
3. **Liniowa funkcja aktywująca w postaci 0.05x:**
4. Każdy przedstawiony na wykresie pomiar to wartość średnia obliczona na podstawie 20 prób.



Dla powyższego przypadku współczynnik uczenia równy 0.0415 okazał się optymalny.

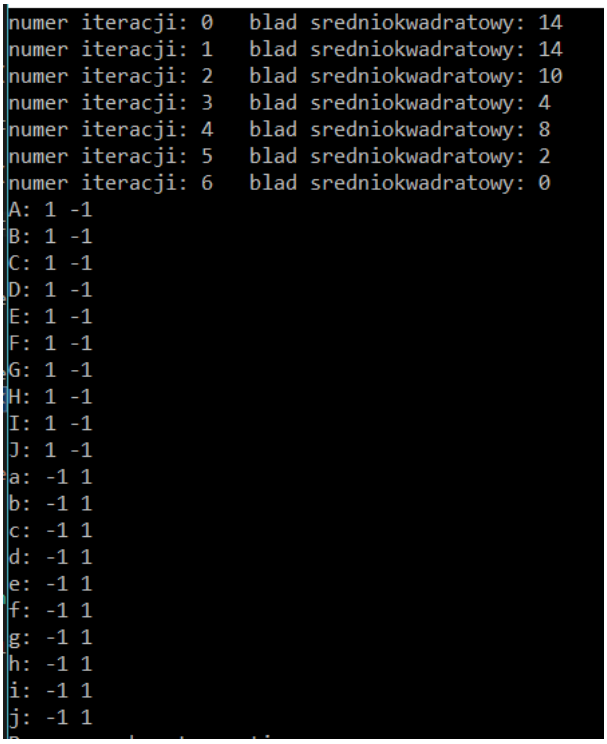
4.Bipolarna funkcja aktywacji:k

Ponieważ wynik w przypadku funkcji bipolarnej zależy mocno od początkowych wag a te są ustawiane losowo, każdy przedstawiony na wykresie pomiar to wartość średnia obliczona na podstawie 200 prób.



Bardzo mały współczynnik uczenia znacznie zwiększa ilość niezbędnych iteracji, natomiast przy pewnej granicy zwiększanie współczynnika nie wpływa już znacząco na tempo uczenia,

Przykładowy output programu:



Wnioski:

- Im współczynnik uczenia jest większy, tym większa poprawka wag jest wykonywana przy takiej samej wartości błędu. Dlatego dla małych wartości η potrzeba znacznie więcej iteracji. Należy jednak pamiętać, że dla dużych wartości η poprawka wag może być za duża i nie przybliży nas do rozwiązania, tak jak to miało miejsce w przypadku dwóch pierwszych rozpatrywanych przypadkach.

- Gdy stosujemy liniową funkcję aktywacyjną musimy pamiętać, że w przypadku dużego współczynnika a początkowo zwracane przez nią wartości mogą być bardzo duże, szczególnie gdy neuron ma dużo wejść.
- Jeżeli od sieci chcemy tylko sklasyfikowania danego elementu na zasadzie należy/nie należy to lepiej zastosować funkcję bipolarną. Funkcja liniowa umożliwia otrzymanie wielu różnych wyników a nie tylko wyniku binarnego ale równocześnie nauka zajmuje znacznie więcej iteracji.