Piotr Wiśniewski

Sprawozdanie 1

Data oddania: 20.10.2017r.

Temat ćwiczenia:

Budowa i działanie perceptronu prostego, jednowarstwowego.

Cel ćwiczenia:

Poznanie budowy i działania perceptronu poprzez implementację i uczenie perceptronu realizacji wybranej funkcji logicznej.

Operacje zachodzące wewnątrz sztucznego neuronu:

* 1. wymnożenie danych wejściowym przez odpowiadające im wagi- to właśnie one są zasadniczym elementem w procesie uczenia perceptronu. Początkowo są wybrane losowo, a następnie w kolejnych epokach, ich wartość jest optymalizowana.
  2. następnie w bloku sumującym iloczyny *xn* i wag *w1,w2,…wn* są sumowane ze sobą. Musimy także uwzględnić w sumie bias (*b*), czyli wartość odchylenia odpowiadająca za nieliniowe przekształcenie wejść w wyjście. Bias jest stałą losową na wejściu, tak jak w przypadku wag.
  3. w bloku aktywacji określa się funkcję aktywacji, która jest prostą służącą do dzielenia przestrzeni, klasyfikująca dane wejściowe do odpowiednich grup. Do naszego zadania wykorzystamy funkcję progową unipolarną:

gdzie θ jest zadaną wartością progową (zmienna threshold) i przyjmujemy, że wynosi 0.

* 1. W ten sposób otrzymujemy wartość wyjściową y, która wynosi 0 lub 1, zależną od sumy ∑.

Listing kodu programu:

from random import choice  
from numpy import array, dot, random  
  
unit\_step = lambda x: 0 if x < 0 else 1  
  
""" 2 element to bias"""  
training\_data = [  
 (array([0, 0, 1]), 0),  
 (array([0 ,1, 1]), 1),  
 (array([1, 0, 1]), 1),  
 (array([1, 1, 1]), 1),  
]  
  
w = random.rand(3)  
errors = []  
"""wspolczynnik nauki, krok uczenia sie """  
eta = 0.2  
""" liczba iteracji nauki """  
n = 50  
iteration = 1  
"""range zwraca iterator"""  
for i in range(n):  
 print("Test nr",iteration)  
 """losowe wejscie"""  
 x, expected = choice(training\_data)  
 print("Random input:")  
 print(x)  
 print(expected)  
 """mnozenie obiektow"""  
 result = dot(w, x)  
  
 error = expected - unit\_step(result)  
  
 errors.append(error)  
  
 w += eta \* error \* x  
 iteration = iteration +1  
print(" ")  
for x, \_ in training\_data:  
 result = dot(x, w)  
 print("{}: {} -> {}".format(x[:2], result, unit\_step(result)))

Analiza wyników:

eta = 1

liczba iteracji = 10

[0 0]: -0.13451558686677434 -> 0

[0 1]: 0.21820780282691676 -> 1

[1 0]: 0.5257288040030969 -> 1

[1 1]: 0.8784521936967881 -> 1

Przy wysokim współczynniku nauki nawet po małej liczbie prób program poprawnie potrafi oszacować wynik

eta = 0.2

liczba iteracji = 10

[0 0]: 0.4123005912622348 -> 1

[0 1]: 1.3671171404568605 -> 1

[1 0]: 0.853086984787754 -> 1

[1 1]: 1.8079035339823797 -> 1

Przy mniejszym współczynniku nauki program ma już większe problemy.

eta = 0.2

liczba iteracji = 20

[0 0]: -0.1960514059296195 -> 0

[0 1]: 0.6023421036412839 -> 1

[1 0]: -0.07336873599294519 -> 0

[1 1]: 0.7250247735779582 -> 1

Dopiero przy współczynniku nauki (eta) na poziomie równym około 30 program poprawnie potrafi przewidzieć wyniki.

Wnioski:

Współczynnik uczenia się perceptronu ma decydujący wpływ na szybkość jego uczenia się - im wyższy tym szybciej on przebiega. Przy eta= 1 lub eta = 0.5 sztuczny neuron potrzebował jedynie 3 iteracji, by wyliczane dane wyjściowe były prawidłowe.

Podczas każdego wykonanego testowania, nie wykryto błędu w obliczeniach perceptronu. Można podejrzewać, że stało się tak, ponieważ funkcje logiczne są jednymi z najprostszych operacji i prawdopodobieństwo otrzymania dobrego wyniku jest równe 50%.

Wszystkie 3 czynniki wpływają na skuteczność uczenia się perceptronu: wagi początkowe, współczynnik uczenia się a także ilość powtórzeń danych uczących.