МИНЕСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧЕРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«Брестский государственный технический университет»

Кафедра «Интеллектуальные информационные технологии»

Лабораторная работа №4

По дисциплине «Интеллектуальные методы обработки данных»

За 7 семестр

Тема: «Нелинейные искусственные нейронные сети в задачах прогнозирования»

Выполнила:  
студентка 4 курса  
группы АС-56  
Карпенко М.В.

Проверил:

Савицкий Ю.В.

Брест 2022

*Цель работы:* Изучить обучение и функционирование нелинейной искусственной нейронной сети (ИНС) при решении задач прогнозирования

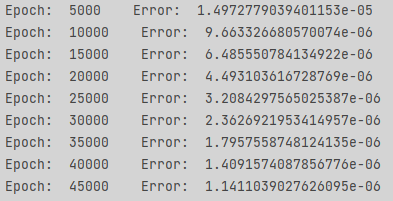
*Задание:*   
Написать на любом ЯВУ программу моделирования прогнозирующей нелинейной ИНС. Для тестирования использовать функцию *.* Для прогнозирования использовать многослойную ИНС с одним скрытым слоем. В качестве функций активации НЭ использовать сигмоидную функцию, для выходного - линейную.

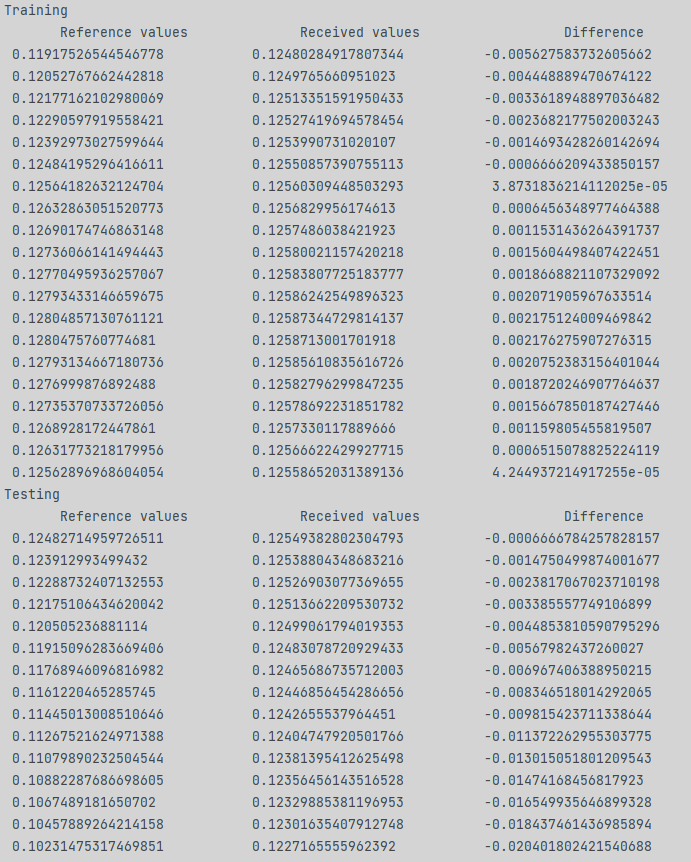
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № варианта | *a* | *b* | *с* | *d* | Кол-во входов ИНС | Кол-во НЭ в скрытом слое |
| 9 | 0.1 | 0.3 | 0.08 | 0.3 | 10 | 4 |

*Текст программы:*

import random  
from math import sin, cos, exp  
  
def hiddenS(hiddenArrY, hiddenArrW, ethalonArr, hiddenArrT, input, hidden, iter):  
 for i in range(len(hiddenArrY)):  
 hiddenArrY[i] = 0  
  
 for i in range(hidden):  
 for j in range(input):  
 hiddenArrY[i] += hiddenArrW[i][j] \* ethalonArr[iter + j]  
 hiddenArrY[i] -= hiddenArrT[i]  
 return hiddenArrY  
  
def hiddenY(hiddenS):  
 return 1 / 1 + exp(-abs(hiddenS))  
  
def Y(hiddenArrY, arrOutW, outT, hidden):  
 y = 0  
 for i in range(hidden):  
 y += hiddenArrY[i] \* arrOutW[i]  
 return y - outT  
  
def wHiddenChanging(hiddenArrW, gammaHidden, hiddenArrY, ethalonArr, input, hidden, iter, step):  
 for i in range(hidden):  
 for j in range(input):  
 hiddenArrW[i][j] -= (step \* gammaHidden[i] \* hiddenArrY[i] \* (1 - hiddenArrY[i]) \* ethalonArr[j + iter])  
 return hiddenArrW  
  
def main():  
 a, b, c, d = 0.1, 0.3, 0.08, 0.3  
 input = 10  
 NE = 4  
 step = 0.1  
 training\_count = 30  
 testing\_count = 15  
 table\_array = []  
 reference\_array = []  
  
 for x in range(training\_count):  
 reference\_array.append(a \* cos(b \* x \* step) + c \* sin(d \* x \* step))  
  
 NEarr = []  
  
 for i in range(NE):  
 row = []  
 for j in range(input):  
 row.append(random.uniform(0, 1))  
 NEarr.append(row)  
  
 gammaOut = 0  
 hiddenArrT, hiddenArrY, gammaHidden = [], [], []  
  
 for i in range(NE):  
 hiddenArrT.append(0)  
 hiddenArrY.append(0)  
 gammaHidden.append(0)  
  
 arrOutW = [random.uniform(0, 1) for i in range(NE)]  
 outT = random.uniform(0, 1)  
  
 epoch = 0  
 y = 0  
 error = 1  
 min\_error = 1e-6  
 while error > min\_error:  
 error = 0  
 for j in range(training\_count - input):  
 hiddenArrY = hiddenS(hiddenArrY, NEarr, reference\_array, hiddenArrT, input, NE, j)  
 for i in range(NE):  
 hiddenArrY[i] = hiddenY(hiddenArrY[i])  
  
 y = Y(hiddenArrY, arrOutW, outT, NE)  
 gammaOut = y - reference\_array[j + input]  
  
 for i in range(NE):  
 gammaHidden[i] = gammaOut \* y \* (1 - y) \* arrOutW[i]  
 arrOutW[i] -= step \* gammaOut \* hiddenArrY[i]  
 hiddenArrT[i] += step \* gammaHidden[i] \* hiddenArrY[i] \* (1 - hiddenArrY[i])  
  
 outT += step \* gammaOut  
 NEarr = wHiddenChanging(NEarr, gammaHidden, hiddenArrY, reference\_array, input, NE, j, step)  
  
 error += ((y - reference\_array[j + input]) \*\* 2) / 2  
 epoch += 1  
 error /= training\_count - input  
 if epoch % 5000 == 0:  
 print('Epoch: ', epoch, ' Error: ', error)  
 table\_array.append(error)  
  
 print("Training")  
 print("{:^30}{:^30}{:^30}".format("Reference values", "Received values", "Difference"))  
  
 for j in range(training\_count - input):  
 hiddenArrY = hiddenS(hiddenArrY, NEarr, reference\_array, hiddenArrT, input, NE, j)  
 for i in range(NE):  
 hiddenArrY[i] = hiddenY(hiddenArrY[i])  
  
 y = Y(hiddenArrY, arrOutW, outT, NE)  
 print("{:< 30}{:< 30}{:< 30}".format(reference\_array[j + input], y, reference\_array[j + input] - y))  
  
 reference\_test\_array = []  
 for x in range(training\_count - input, testing\_count + training\_count):  
 reference\_test\_array.append(a \* cos(b \* x \* step) + c \* sin(d \* x \* step))  
  
 print("Testing")  
 print("{:^30}{:^30}{:^30}".format("Reference values", "Received values", "Difference"))  
 for j in range(testing\_count):  
 hiddenArrY = hiddenS(hiddenArrY, NEarr, reference\_test\_array, hiddenArrT, input, NE, j)  
 for i in range(NE):  
 hiddenArrY[i] = hiddenY(hiddenArrY[i])  
  
 y = Y(hiddenArrY, arrOutW, outT, NE)  
 print("{:< 30}{:< 30}{:< 30}".format(reference\_test\_array[j + input], y, reference\_test\_array[j + input] - y))  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

*Результаты работы программы:*





*Выводы по лабораторной работы:* Изучила обучение и функционирование нелинейной искусственной нейронной сети (ИНС) при решении задач прогнозирования.