|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)

КАФЕДРА «Информационная безопасность» (ИУ8)

Отчёт

по лабораторной работе № 2

по дисциплине «Интеллектуальные технологии информационной безопасности»

**Тема: «Применение однослойной нейронной сети с линейной функцией активации»**

Выполнил: Фалалеев Д.П.,

студент группы ИУ8-63

Проверил: Волосова Н.К,

аспирант каф. ИУ8

г. Москва,

2021 г.

1. Цель работы

Изучить возможности однослойный НС в задачах прогнозирования временных рядов методом скользящего окна (авторегрессия).

1. Постановка задачи

На временном интервале [a, b] задан дискретный набор значений функции: *cos(x).* Количество точек N = 20, расположение – равномерное. Методом “скользящего окна” спрогнозировать поведение функции на N точках последующего интервала (b, 2b – a). Для решение использовать однослойную нейронную сеть с количеством нейронов p=4 и линейной функцией активации. Обучение проводить методом Видроу-Хоффа. Исследовать количество эпох обучения и коэффициента обучения на среднеквадратичной погрешность приближения. Исследовать процесс прогнозирование при постепенным изменении (уменьшение /увеличение) размера окна. Сделать выводы по результатам численного эксперимента.

1. Ход работы
2. Сначала были сформированы наборы иксов на интервале [a, b], [a, 2b – a] и вычислены значения функции на заданном интервале.
3. При старте обучения нейронной сети (НС) были сгенерированы наборы весов в количестве **длина окна + 1** и выбрана норма обучения равная nj = 0.8.
4. В процессе самого обучения мы начинали с элемента с индексом длины окна (необходимо для начала обучения).
5. Сравнивая полученное значение с реальным, получается ошибка прогноза.
6. Далее проводится корреляция весов по формуле:



, где:





xk – значение функции на k-ом шаге.

1. Корреляция происходит до достижения минимально допустимой погрешности или прохождения заданного количества эпох.

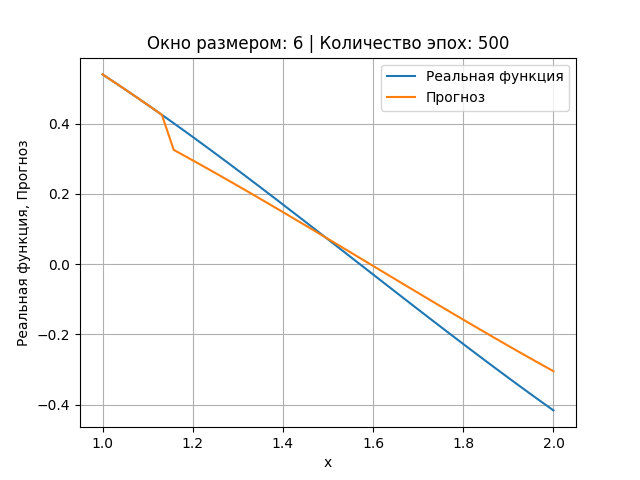
1. Выводы

Исходя из опытов обучения нейронных сетей в этой лабораторной работе можно сделать выводы, что чем больше размер окна тем вычисления более неточны, в то время как чем больше количество эпох обучения – тем точнее прогноз.

**Приложение 1 Исходный код программы.**

import math  
import matplotlib.pyplot as plt  
  
  
def fun(x):  
 return math.cos(x)  
  
  
def f\_net(x, w, w0):  
 net = w0  
 for (k, j) in zip(x, w):  
 net += k \* j  
 return net  
  
  
class NeuralActivation:  
  
 def \_\_init\_\_(self, window\_size):  
 self.a = 1  
 self.b = 1.5  
 self.c = 2 \* self.b - self.a  
 self.data = [self.a]  
 self.window\_size = window\_size  
 self.right\_function\_answer = [fun(self.a)]  
 k = (self.b - self.a)/19  
 a = self.a  
 self.next\_data = [self.a]  
 for \_ in range(0, 19):  
 a += k  
 self.right\_function\_answer.append(fun(a))  
 self.data.append(a)  
 self.next\_data.append(a)  
  
 while a <= self.c:  
 a += k  
 self.next\_data.append(a)  
 self.right\_function\_answer.append(fun(a))  
  
 def go(self, max\_epochs):  
 answer\_data = []  
 w = [0]\*(self.window\_size + 1)  
 epoch = 0  
 nj = 0.8  
 p = self.window\_size  
 sum\_error = 1  
 while epoch <= max\_epochs and sum\_error > 0.001:  
 sum\_error = 0  
 y\_net = []  
 for i in range(p, len(self.data), 1):  
 net = f\_net(self.right\_function\_answer[i - p:i], w[1:], w[0])  
 y\_net.append(net)  
 sigma = self.right\_function\_answer[i] - net  
 sum\_error += sigma\*sigma  
  
 for j in range(0, p):  
 w[j + 1] += sigma \* nj \* fun(self.right\_function\_answer[i - p + j])  
 sum\_error \*\*= 0.5  
 answer\_data.append([w, sum\_error, y\_net])  
  
 # print(epoch, sum\_error, w)  
 epoch += 1  
 print('\n\n\nWINDOW SIZE = ', self.window\_size, '\nEPOCHS = ', epoch - 1, '\n error = ', answer\_data[-1][1], '\n'  
 + ' weighs = ', answer\_data[-1][0])  
 self.print\_answer(answer\_data[-1], len(answer\_data) - 1)  
  
 def print\_answer(self, answer\_data, epochs):  
 y = []  
 y\_w = self.right\_function\_answer[:self.window\_size]  
  
 for i in range(self.window\_size, len(self.next\_data)):  
 y\_w.append(f\_net(self.right\_function\_answer[i - self.window\_size:i], answer\_data[0][1:], answer\_data[0][0]))  
  
 for i in self.next\_data:  
 y.append(fun(i))  
  
 \_, ax = plt.subplots()  
 ax.plot(self.next\_data, y, label='Реальная функция')  
 ax.plot(self.next\_data, y\_w, label='Прогноз')  
 ax.legend()  
 plt.title(f'Окно размером: {self.window\_size} | Количество эпох: {epochs}') # заголовок  
 plt.xlabel("x") # ось абсцисс  
 plt.ylabel("Реальная функция, Прогноз") # ось ординат  
 plt.grid() # включение отображение сетки  
  
 plt.show()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 for i in range(0, 3):  
 n = NeuralActivation(6 + i)  
 n.go(500)  
 n.go(1000)

**Приложение 2 Вывод.**

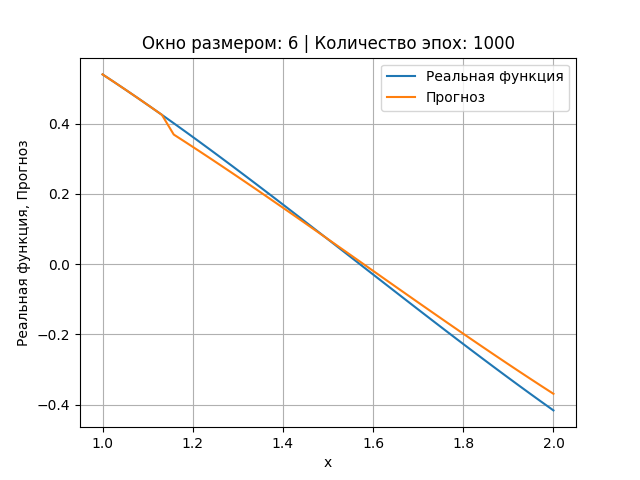


WINDOW SIZE = 6

EPOCHS = 500

error = 0.15520700746667326

weighs = [0, -0.15253531553909494, -0.04724980116447033, 0.06355095350246834, 0.179559439631636, 0.30044365410437035, 0.42584810042016463]

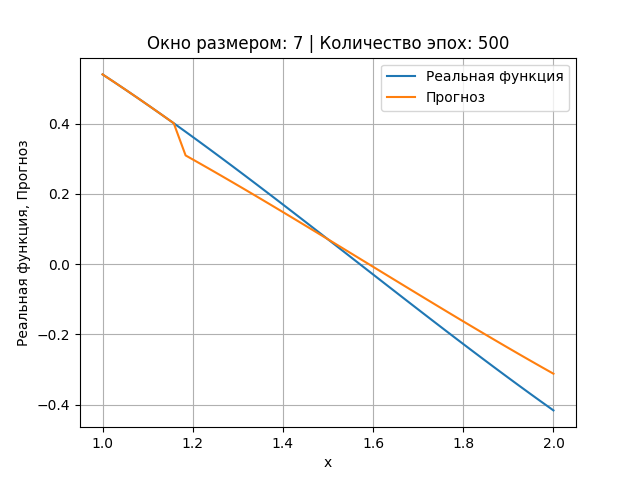


WINDOW SIZE = 6

EPOCHS = 1000

error = 0.0657346958504534

weighs = [0, -0.2482661471081124, -0.09894093685705845, 0.05823743332525878, 0.22283286886265655, 0.39437441897074055, 0.5723576952159035]

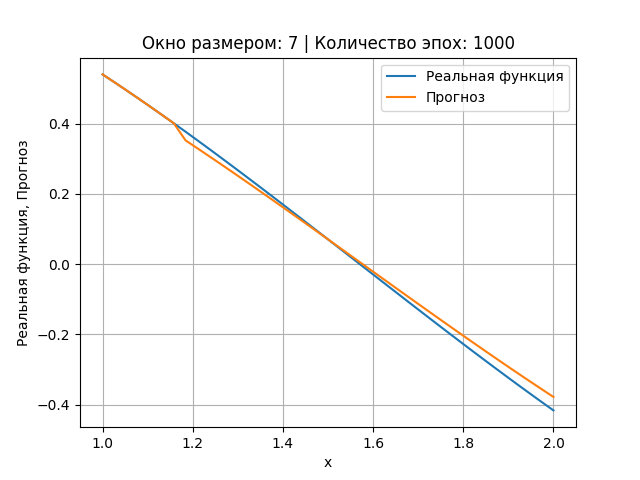


WINDOW SIZE = 7

EPOCHS = 500

error = 0.2628720855093935

weighs = [0, -0.15046894894113944, -0.0700539100783453, 0.014777850460625163, 0.1037943683291771, 0.1967441548893758, 0.2933569164268789, 0.3933444189771318]

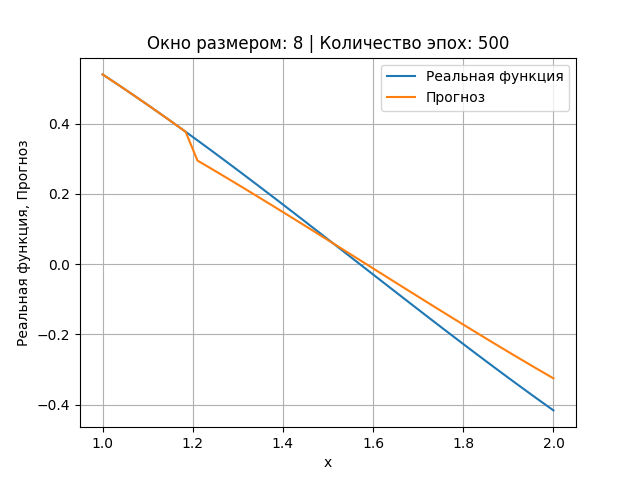


WINDOW SIZE = 7

EPOCHS = 1000

error = 0.09643147568442009

weighs = [0, -0.2261243698067085, -0.1165799921226192, -0.0009979102421705578, 0.1203059223955148, 0.24698887630402058, 0.3786826248714394, 0.5149943221149399]

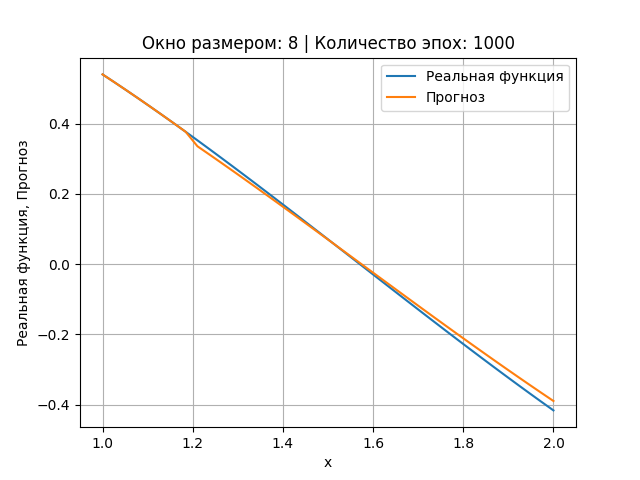


WINDOW SIZE = 8

EPOCHS = 500

error = 0.5030699931046089

weighs = [0, -0.15334840038903785, -0.08778093542876396, -0.018493463748451202, 0.05432643513680203, 0.1304747991604957, 0.20973184570778952, 0.29186265532849814, 0.376617975043901]



WINDOW SIZE = 8

EPOCHS = 1000

error = 0.14771127696139555

weighs = [0, -0.21013459260447412, -0.1255479865756273, -0.03614979907994271, 0.05781800502695612, 0.1560922791165668, 0.2583894192682599, 0.36440624604763167, 0.4738210408597662]