Министерство образования Республики Беларусь

УО «Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №1**

По дисциплине: “Методы и алгоритмы принятия решений ”

Тема: «Линейная искусственная нейронная сеть. Правило обучения Видроу-Хоффа.»**Вариант №10**

**Выполнил**:

студент 2 курса

группы ПО-7

Курмыса Е.Е.

**Проверил:**

Крощенко А.А.

Брест 2021

**Цель работы:** изучить обучение и функционирование линейной ИНС при решении задач прогнозирования.

**Постановка задачи:** написать на любом ЯВУ (здесь — Python) программу моделирования прогнозирующей линейной ИНС. Для тестирования использовать функцию , количество входов ИНС .

**Код:**

from math import sin

from random import uniform

def f(x):

return 2 \* sin(9 \* x) + 0.4

omega = [uniform(0, 1) for \_ in range(3)] # коэффициенты

T = 0 # порог обучения

count\_of\_iterations = 1 # порядковый номер очередного прогона

Err\_min = float(input('Введите максимально возможную квадратичную погрешность при обучении: '))

error\_sum = 0 # для высчитывания итоговой среднеквадратической ошибки

last\_err = 0 # для того чтобы прервать обучение при том случае, когда погрешности не изменятся

alpha = float(input('Введите скорость обучения: '))

print('Изначальные данные:\nИзначальные коэфф. -', omega, '\nПорог функции T -', T, '\nШаг функции alpha -', alpha)

# обучение на тридцати итерациях, после которых мы получаем коэффициенты и T для непосредственного прогноза

print(f'Обучение:\nЭпоха # {"N":2}: {"Идеальное значение t":21} {"Полученное значение":21} {"Отклонение между знач.":26} {"Квадр. погр.":26}')

while count\_of\_iterations <= 100:

for i in range(30):

x = [f(j / 10) for j in range(i, i + 3)] # для обучения, проверять на практике будем другие значения

X = (i + 3) / 10 # аргумент функции для проверки погрешности

S = 0

for j in range(3):

S += omega[j] \* x[j] # приближённое значение f(X) ч/з нейронные сети

t = f(X) # идеальное значение f(X) напрямую

y = S - T

delta = y - t

err = delta\*\*2 / 2

error\_sum = err

for j in range(3):

omega[j] -= alpha \* delta \* x[j]

T += alpha \* delta

if last\_err == err:

break

print(f'Эпоха # {count\_of\_iterations:2}: {t:21} {y:21} {delta:24} {err:24}')

if error\_sum / 30 <= Err\_min:

break

last\_err = err

count\_of\_iterations += 1

print('\nПосле обучения имеем следующие конфигурации:\nКоэффициенты =', omega, '\nПорог функции T =', T)

print('\nПроводим тестирование/предсказание значения функции на 15-ти случайных рациональных значениях от -100 до 100:')

test\_number = 1 # номер теста

error\_sum = 0 # для высчитывания итоговой среднеквадратической ошибки

print(f'Тестирование:\nЭпоха # {"N":2}: {"Идеальное значение t":21} {"Полученное значение":21} {"Отклонение между знач.":26} {"Квадр. погр.":26}')

while test\_number <= 15:

X = uniform(-100, 100) # аргумент для тестирования

x = [f(X - 0.3), f(X - 0.2), f(X - 0.1)] # значения x для генерации ответа

t = f(X) # идеальное значение функции

S = 0 # сумматор произведений

for i in range(3):

S += x[i] \* omega[i]

y = S - T

delta = y - t

err = delta\*\*2 / 2

error\_sum += err

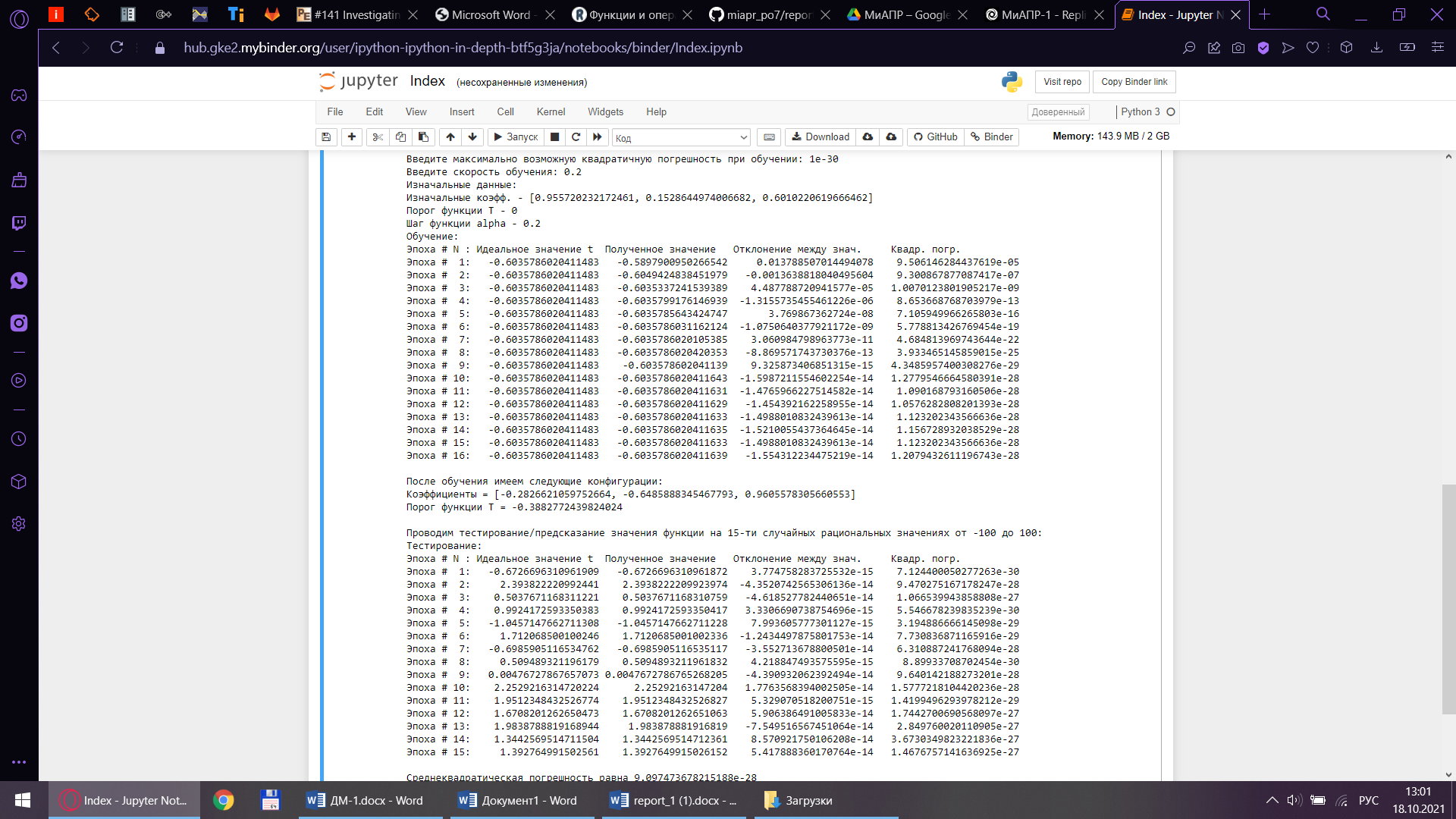
print(f'Эпоха # {test\_number:2}: {t:21} {y:21} {delta:24} {err:24}')

test\_number += 1

avg\_error = error\_sum / 15 # среднеквадратическая погрешность

print('\nСреднеквадратическая погрешность равна', avg\_error)

**Вывод**



Как мы видим, при заданной нами максимальной среднеквадратической погрешности в программа выводит результат с погрешностью . В данном случае это особой проблемой не является, однако в случаях, когда указанные дольные будут решающими, требуется повышать точность вычислений, увеличивая, например, число эпох или улучшая код.

**Вывод:** я изучил обучение и функционирование линейной ИНС при решении задач прогнозирования.