Министерство образования Республики Беларусь

УО «Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №2**

По дисциплине: “Методы и алгоритмы принятия решений”

Тема: “Линейная искусственная нейронная сеть.

Адаптивный шаг обучения”

**Вариант №9**

**Выполнил**: студент 2 курса группы ПО-7 Крупенков Михаил Дмитриевич

**Проверил:** Крощенко А.А.

Брест 2021

**Цель работы:**

Изучить обучение и функционирование линейной ИНС с применением адаптивного шага.

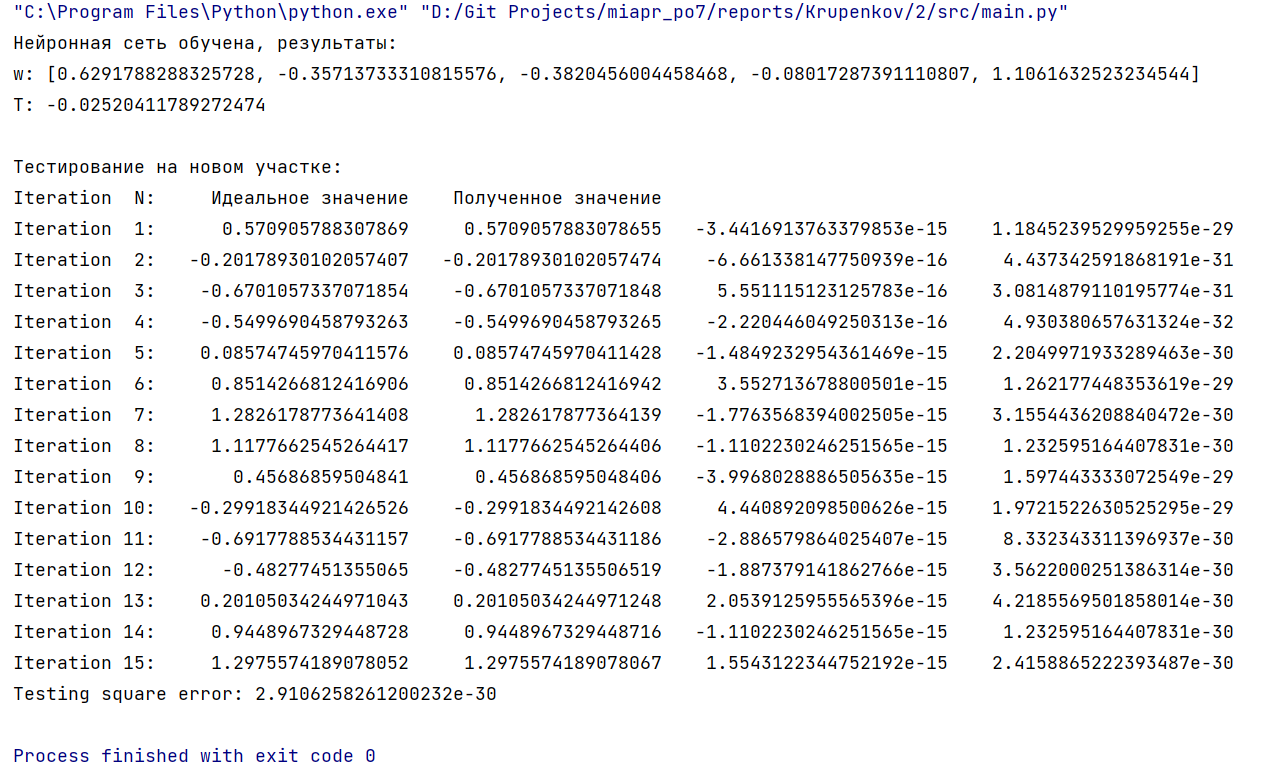
**Постановка задачи:**

Модифицировать программу из лабораторной работы №1, используя правило адаптивного шага обучения. Произвести исследование получившейся модели ИНС на задачах прогнозирования, согласно варианту лабораторной работы №1.

**Код программы:**

**from** math **import** sin  
**from** random **import** uniform  
**from** typing **import** List  
  
*# Гиперпараметры обучения*MIN\_SQUARE\_ERROR = 1e-30 *# Минимальная ошибка для остановки обучения*TRAINING\_EPOCH\_AMOUNT = 30 *# Количество значений функции (эпох) для обучения*TESTING\_EPOCH\_AMOUNT = 15 *# Количество значений функции (эпох) для прогнозирования*MAX\_ITERATIONS\_AMOUNT = 18 *# Максимальное количество повторений обучения  
  
  
# Функция по условию (Вариант 9)***def** function(x: float) -> float:  
 **return** sin(8 \* x) + 0.3  
  
  
*# Вывод нс***def** calculate\_output(  
 inputs\_amount: int, w: List[float], training\_outputs: List[float], epoch: int, T: float  
) -> float:  
 *# Вычисление выходного значения (Формула 1.2)* output: float = 0  
 **for** j **in** range(inputs\_amount):  
 output += w[j] \* training\_outputs[epoch + j]  
  
 **return** output - T  
  
  
**def** main() -> **None**:  
 inputs\_amount = 5 *# Количество входов* step = 0.1 *# Шаг табуляции функции  
  
 # Значения функции для обучения* training\_outputs: List[float] = [function(i \* step) **for** i **in** range(TRAINING\_EPOCH\_AMOUNT + inputs\_amount)]  
  
 *# Значения функции для прогнозирования* testing\_outputs: List[float] = [  
 function(i \* step) **for** i **in** range(TRAINING\_EPOCH\_AMOUNT, TRAINING\_EPOCH\_AMOUNT + TESTING\_EPOCH\_AMOUNT + inputs\_amount)  
 ]  
  
 w: List[float] = [uniform(0, 1) **for** \_ **in** range(inputs\_amount)]  
 T: float = uniform(0, 1) *# Порог  
  
 # \_\_\_\_\_\_\_\_\_\_ Начало \_\_\_\_\_\_\_\_\_\_* **try**:  
 iteration = 0  
 square\_error: float = MIN\_SQUARE\_ERROR *# Среднеквадратичная ошибка  
  
 # \_\_\_\_\_\_\_\_\_\_ Обучение \_\_\_\_\_\_\_\_\_\_* **while** square\_error >= MIN\_SQUARE\_ERROR **and** iteration < MAX\_ITERATIONS\_AMOUNT:  
  
 square\_error\_sum = 0  
 iteration += 1  
  
 **for** epoch **in** range(TRAINING\_EPOCH\_AMOUNT):  
 output = calculate\_output(inputs\_amount, w, training\_outputs, epoch, T)  
 ideal\_output: float = training\_outputs[epoch + inputs\_amount] *# Истинное значение функции* error: float = output - ideal\_output *# Отклонение от функции  
  
 # Изменение скорости обучение* training\_speed = 1 / (1 + sum([y \*\* 2 **for** y **in** training\_outputs[epoch:epoch + inputs\_amount]]))  
  
 *# Обновление весов и порога (Формула 1.7, 1.8)* **for** t **in** range(inputs\_amount):  
 w[t] -= training\_speed \* error \* training\_outputs[epoch + t]  
 T += training\_speed \* error  
  
 *# Сумма среднеквадратичных ошибок (Фейк формула 1.3)* square\_error\_sum += error \*\* 2  
 square\_error = square\_error\_sum / TRAINING\_EPOCH\_AMOUNT  
  
 *# \_\_\_\_\_\_\_\_\_\_ Результаты обучения \_\_\_\_\_\_\_\_\_\_* print(**'Нейронная сеть обучена, результаты:'**)  
 print(**f'w: {**w**}\n'  
 f'T: {**T**}\n'**)  
 print(**'Тестирование на новом участке:'**)  
 print(**'Iteration N: Идеальное значение Полученное значение '**)  
 square\_error\_sum = 0  
  
 *# \_\_\_\_\_\_\_\_\_\_ Тестирование \_\_\_\_\_\_\_\_\_\_* **for** epoch **in** range(TESTING\_EPOCH\_AMOUNT):  
 output = calculate\_output(inputs\_amount, w, testing\_outputs, epoch, T)  
  
 ideal\_output: float = testing\_outputs[epoch + inputs\_amount] *# Истинное значение функции* error: float = output - ideal\_output *# Отклонение от функции* square\_error\_sum += error \*\* 2 *# Суммарная ошибка (Формула 1.3)* print(**f'Iteration {**epoch + 1**:2}: {**ideal\_output**:21} {**output**:21} '  
 f'{**error**:24} {**error \*\* 2 **if** error **else " are the same":24}'**)  
  
 square\_error = square\_error\_sum / TRAINING\_EPOCH\_AMOUNT  
 print(**f'Testing square error: {**square\_error**}'**)  
  
 **except** OverflowError:  
 print(**'Слишком большая скорость обучения, выход из программы'**)  
  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 main()

**Результаты программы:**



**Вывод:**

Я изучил обучение и функционирование линейной ИНС с применением адаптивного шага.