Тест начат	Четверг, 21 декабря 2023, 13:40
Состояние	Завершены
Завершен	Четверг, 21 декабря 2023, 13:53
Прошло времени	12 мин. 46 сек.
Оценка	**3,00** из 3,00 (**100**%)

Вопрос **1**

Верно

Баллов: 1,00 из 1,00

Сигмоидальная функция активации

В этом задании вам нужно написать программу для вычисления результата сигмоидальной функции активации для заданных входных данных (два числа — $(x_1$ и $x_2)$) и коэффициентов. Ваша задача — дописать недостающие фрагменты кода.

Ответ: (штрафной режим: 0%)

Сбросить ответ

```
import math

# Функция сигмоид
def sigmoid(x):
    return 1 / (1 + math.exp(-x))

# Задаем коэффициенты a и b
a = 0.5
b = 0.2

# Входные данные
x1 = 1.0
x2 = 0.5

# Линейная комбинация
linear_combination = a * x1 + b * x2

# Применяем функцию активации (сигмоид)
output = sigmoid(linear_combination)

# Выводим результат
print(output)

```

	Тест	Ожидается	Получил	
✔	`print(output == 0.6456563062257954)`	`True`	`True`	✔

Все тесты пройдены! ✔

Спасибо за выполненное задание! Требуемый результат получен!

Предлагаем проверить себя с возможным вариантом решения.

```
import math
```

```
# Функция сигмоид
def sigmoid(x):
    return 1 / (1 + math.exp(-x))
```

```
# Задаем коэффициенты a и b
a = 0.5
b = 0.2
```

```
# Входные данные
x1 = 1.0
x2 = 0.5
```

```
# Линейная комбинация
linear_combination = a * x1 + b * x2
```

```
# Применяем функцию активации (сигмоид)
output = sigmoid(linear_combination)
```

```
# Выводим результат
print(output)
```

Верно

Баллы за эту попытку: 1,00/1,00.

Вопрос **2**

Верно

Баллов: 1,00 из 1,00

Вычисление свертки

Вам дана матрица **A** размером 5x5 и матрица фильтра **F** размером 3x3. Примените операцию свертки между матрицей **A** и матрицей **F** и получите выходную матрицу **C**.

Алгоритм свертки:

1. Проходя по каждой строке матрицы **A**, возьмите окно размером 3x3 начиная с первого элемента и перемножьте его с матрицей фильтра **F**. Сложите результаты умножения, чтобы получить первый элемент выходной матрицы **C**.
2. Повторите шаг 1 для всех других элементов строки **A**, чтобы заполнить первую строку матрицы **C**.
3. Повторите шаги 1–2 для всех оставшихся строк матрицы **A**, чтобы заполнить оставшиеся строки матрицы **C**.

Часть кода уже представлена, ваша задача – дописать недостающие фрагменты кода.

Ответ: (штрафной режим: 0%)

Сбросить ответ

```
import numpy as np

# создаем исходную матрицу A размером 5x5
A = np.array([[1, 2, 3, 4, 5],
              [6, 7, 8, 9, 10],
              [11, 12, 13, 14, 15],
              [16, 17, 18, 19, 20],
              [21, 22, 23, 24, 25]])

# создаем матрицу фильтра F размером 3x3
F = np.array([[1, 2, -1],
              [1, 0, -1],
              [1, 0, -1]])

# определяем размеры матриц A и F
height_a, width_a = A.shape
height_f, width_f = F.shape

# инициализируем выходную матрицу C
C = np.zeros((height_a - height_f + 1, width_a - width_f + 1))

# проходим по каждой строке матрицы A
for i in range(height_a - height_f + 1):
    # проходим по каждому столбцу матрицы A
    for j in range(width_a - width_f + 1):
        # получаем окно размером 3x3, начиная с текущей позиции
        window = A[i:i+height_f, j:j+width_f]
        # перемножаем окно с матрицей фильтра
        result = window * F
        # суммируем результаты умножения, чтобы получить новое значени
        C[i, j] = np.sum(result)

print(C)

```

	Тест	Ожидается	Получил	
✔	`print(C[0, 0] == -2)`	`True`	`True`	✔
✔	`print(C[2, 2] == 22)`	`True`	`True`	✔

Все тесты пройдены! ✔

Вы получили требуемый результат. Ниже один из возможных вариантов решения задания.

```
import numpy as np
```

```
# Создаем исходную матрицу A размером 5x5
A = np.array([[1, 2, 3, 4, 5],
              [6, 7, 8, 9, 10],
              [11, 12, 13, 14, 15],
              [16, 17, 18, 19, 20],
              [21, 22, 23, 24, 25]])
```

```
# Создаем матрицу фильтра F размером 3x3
F = np.array([[1, 2, -1],
              [1, 0, -1],
              [1, 0, -1]])
```

```
# Определяем размеры матриц A и F
height_a, width_a = A.shape
height_f, width_f = F.shape
```

```
# Инициализируем выходную матрицу C
C = np.zeros((height_a - height_f + 1, width_a - width_f + 1))
```

```
# Проходим по каждой строке матрицы A
for i in range(height_a - height_f + 1):
    # Проходим по каждому столбцу матрицы A
    for j in range(width_a - width_f + 1):
        # Получаем окно размером 3x3 начиная с текущей позиции
        window = A[i:i+height_f, j:j+width_f]
        # Перемножаем окно с матрицей фильтра
        result = window * F
        # Суммируем результаты умножения, чтобы получить новое значение в выходной матрице C
        C[i, j] = np.sum(result)
```

```
print(C)
```

Верно

Баллы за эту попытку: 1,00/1,00.

Вопрос **3**

Верно

Баллов: 1,00 из 1,00

Реализация простой нейросети

Вам даны искусственно созданные данные, представленные в виде матрицы размером 100x3. Реализуйте простую нейросеть, состоящую из одного входного слоя, одного скрытого слоя и одного выходного слоя. Размер скрытого слоя должен быть равен 5.

В качестве функции активации используйте сигмоидальную функцию. Вычислите ошибку обучения (среднеквадратичное отклонение) и выведите ее на экран.

Часть кода уже представлена, ваша задача — дописать недостающие фрагменты кода.

Ответ: (штрафной режим: 0%)

Сбросить ответ

```
# Создаем искусственные данные
import numpy as np
np.random.seed(42)

X = np.random.randn(100, 5)  # матрица 100x5

# Определяем архитектуру нейронной сети
input_size = 5
hidden_size = 3
output_size = 1

# Инициализируем веса случайным образом
W1 = np.random.randn(input_size, hidden_size)  # матрица весов 5x3
W2 = np.random.randn(hidden_size, output_size)  # матрица весов 3x1

# Определяем функцию активации
def sigmoid(x):
    return 1 / (1 + np.exp(-x))

# Прямой проход по нейронной сети
hidden_layer = sigmoid(np.dot(X, W1))  # скрытый слой
output_layer = np.dot(hidden_layer, W2)  # выходной слой

# Вычисляем среднеквадратичную ошибку
y_true = np.random.randn(100, 1)  # реальные значения
mse = round(np.mean((y_true - output_layer) ** 2),2)

print(mse)

```

	Тест	Ожидается	Получил	
✔	`print(mse == 1.1)`	`True`	`True`	✔

Все тесты пройдены! ✔

Вы получили требуемый результат. Ниже один из возможных вариантов решения задания.

```
# Создаем искусственные данные
import numpy as np
np.random.seed(42)
```

```
X = np.random.randn(100, 5)  # Матрица 100x5
```

```
# Определяем архитектуру нейронной сети
input_size = 5
hidden_size = 3
output_size = 1
```

```
# Инициализируем веса случайным образом
W1 = np.random.randn(input_size, hidden_size)  # Матрица весов 5x3
W2 = np.random.randn(hidden_size, output_size)  # Матрица весов 3x1
# Определяем функцию активации
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
```

```
# Прямой проход по нейронной сети
hidden_layer = sigmoid(np.dot(X, W1))  # Скрытый слой
output_layer = np.dot(hidden_layer, W2)  # Выходной слой
```

```
# Вычисляем среднеквадратичную ошибку
y_true = np.random.randn(100, 1)  # Реальные значения
mse = round(np.mean((y_true - output_layer)**2), 2)
```

```
print(mse)
```

Верно

Баллы за эту попытку: 1,00/1,00.