Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

Лабораторная работа № 6

По дисциплине: «Прикладные интеллектуальные системы и экспертные системы**»**

Нейронные сети. Обучение без учителя

Студент Александрук А.М.

Группа М-ИАП-23-1

Руководитель Кургасов В.В.
к.т.н. доцент

Липецк 2023г.

Задание кафедры

Применить нейронную сеть Кохонена с самообучение для задачи кластеризации. На первом этапе сгенерировать случайные точки на плоскости вокруг 2 центров кластеризации (примерно по 20-30 точек).

Далее считать, что сеть имеет два входа (координаты точек) и два выхода – один из них равен 1, другой 0 (по тому, к какому кластеру принадлежит точка).

Подавая последовательно на вход (вразнобой) точки, настроить сеть путем применения описанной процедуры обучения так, чтобы она приобрела способность определять, к какому кластеру принадлежит точка.

Коэффициент α выбрать, уменьшая его от шага к шагу по правилу $\alpha = (50 - i)/100$, причем для каждого нейрона это будет свое значение α, а подстраиваться на каждом шаге будут веса только одного (выигравшего) нейрон.

Ход работы:

Подключаем все возможные библиотеки такие как numpy, matplotlib для выполнения лабораторной работы. Весь код лабораторной работы и вывод результатов рассматриваются на рисунках. Добавим 2 нейрона на вход и 2 на выход.

```
[1]: # Лабораторная работа 6
     # Библиотеки
     import numpy as np
     import matplotlib.pyplot as plt

     # Инициализация параметров
     input_dim = 2
     output_dim = 2
     learning_rate = 0.5
     epochs = 100
```

Рисунок 1 – Выборки для 1 варианта

Обучаем нейронную сеть по правилу $\alpha = (50 - i)/100$. И определяем выигрышный нейрон и строим график кластеризации.

```
[4]: # Обучение нейронной сети
     for epoch in range(epochs):
         np.random.shuffle(data)
         for point in data:
             # Нахождение выигравшего нейрона (ближайшего к точке)
             winner_neuron = np.argmin(np.linalg.norm(weights - point, axis=1))
             weights[winner_neuron] += learning_rate * (point - weights[winner_neuron])

         error = np.mean(np.linalg.norm(weights - data[:, np.newaxis], axis=2))
         print(f'Epoch {epoch + 1}/{epochs}, Quadratic Error: {error:.4f}, Winning Neuron: {winner_neuron}')

     # Визуализация конечного состояния после обучения
     plt.scatter(data[:, 0], data[:, 1], c=np.argmin(np.linalg.norm(weights - data[:, np.newaxis], axis=2), axis=1))
     plt.scatter(weights[:, 0], weights[:, 1], marker='o', s=100, c='green', label='Выигравший нейрон')
     plt.title(f'График кластеризации')
     plt.legend()
     plt.show()
```

Рисунок 2 – Код программы

```
Epoch 94/100, Quadratic Error: 2.9693, Winning Neuron: 1
Epoch 95/100, Quadratic Error: 3.0792, Winning Neuron: 1
Epoch 96/100, Quadratic Error: 3.2002, Winning Neuron: 0
Epoch 97/100, Quadratic Error: 3.2207, Winning Neuron: 1
Epoch 98/100, Quadratic Error: 3.9418, Winning Neuron: 0
Epoch 99/100, Quadratic Error: 3.0479, Winning Neuron: 1
Epoch 100/100, Quadratic Error: 3.0523, Winning Neuron: 0
```

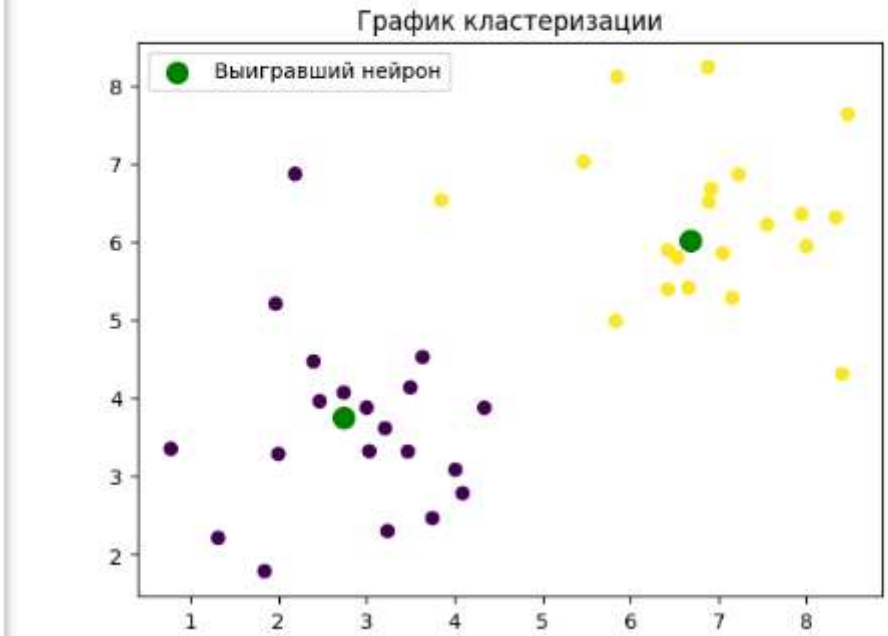

Рисунок 3 – Вывод результатов

Оцениваем точность кластеризации.

```
[5]: # Оценка точности
predicted_clusters = np.argmin(np.linalg.norm(weights - data[:, np.newaxis], axis=2), axis=1)
true_labels = np.concatenate([np.zeros(20), np.ones(20)])
accuracy = np.mean(predicted_clusters == true_labels)
print(f'Точность кластеризации: {accuracy * 100:.2f}%')

Точность кластеризации: 45.00%

[ ]:
```

Рисунок 4 – Оценка точности кластеризации

Точность кластеризации составило 45%. Выигрышный нейрон 0.

Вывод

В ходе выполнения лабораторной работы были использованы методы кластеризации с использованием нейронной сети обучение без учителя. Точность кластеризации составило 45%, а выигрышный нейрон 0. Также был изображен график кластеризации.