**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №5

по дисциплине «Прикладные интеллектуальные системы и экспертные системы»

«Нейронные сети. Обучение без учителя»

Студент Цыганов Н.А.

Группы М-ИАП-23

Руководитель Кургасов В.В.

Доцент

Липецк 2023 г

Цель работы

Использование нейронной сети Кохонена с самообучением для задачи кластеризации случайных точек на плоскости вокруг двух центров кластеризации.

Задание кафедры

Применить нейронную сеть Кохонена с самообучением 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 выбрать, уменьшая его от шага к шагу по правилу $a = (50\text{-}i)/100$, причем для каждого нейрона это будет своѐ значение $a$, а подстраиваться на каждом шаге будут веса только одного (выигравшего) нейрона.

Ход работы

Сгенерируем случайные точки вокруг двух центров кластеризации. Это представлено на рисунке 1.

```
: import numpy as np
  import matplotlib.pyplot as plt
```

```
: # Шаг 1: Генерация случайных точек вокруг двух центров кластеризации
  np.random.seed(42)
  center1 = np.array([2, 2])
  center2 = np.array([-2, -2])
  cluster1 = center1 + np.random.randn(30, 2)
  cluster2 = center2 + np.random.randn(30, 2)
  # Объединение точек в один набор данных
  data = np.vstack([cluster1, cluster2])
```

```
: # Визуализация после генерации точек
  plt.figure(figsize=(8, 6))
  plt.scatter(cluster1[:, 0], cluster1[:, 1], label='Кластер 1')
  plt.scatter(cluster2[:, 0], cluster2[:, 1], label='Кластер 2')
  plt.title('Сгенерированные точки')
  plt.legend()
  plt.show()
```

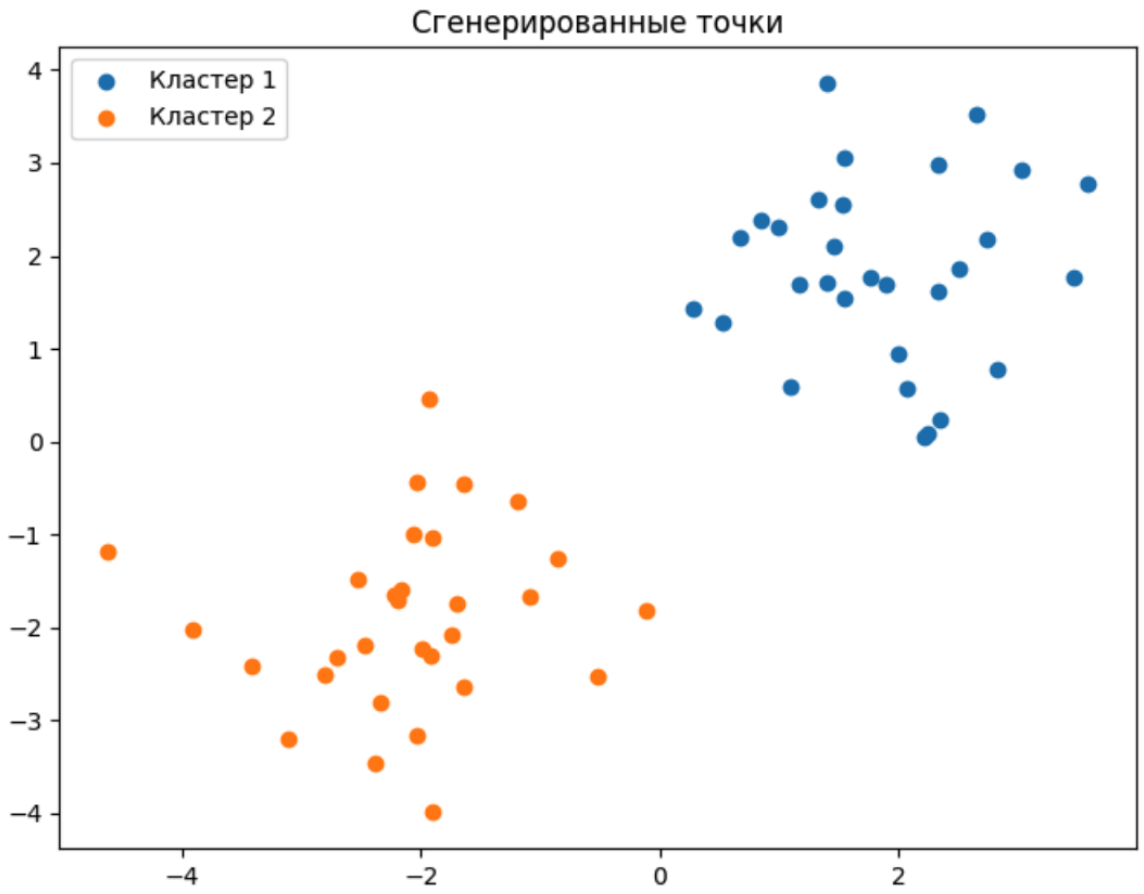


Рисунок 1 - Сгенерированные точки

Далее зададим параметры нейронной сети Кохонена, проинициализируем веса и запустим обучение, код для обучения, график первой эпохи представлен на рисунке 2.

```
# Нейронная сеть Кохонена
input_size = 2
output_size = 2
learning_rate = 0.5
```

```
# Инициализация весов
weights = np.random.rand(output_size, input_size)
```

```
# Визуализация для указанных интервалов эпох
for epoch in range(50):
    if epoch in range(0, 6) or epoch in range(45, 49):
        plt.figure(figsize=(8, 6))
        plt.scatter(cluster1[:, 0], cluster1[:, 1], label='Кластер 1')
        plt.scatter(cluster2[:, 0], cluster2[:, 1], label='Кластер 2')
        plt.scatter(weights[:, 0], weights[:, 1], marker='*', s=200, c='red', label='Центры нейронов')
        plt.title(f'Эпоха {epoch + 1}')
        plt.legend()
        plt.show()

    for point in data:
        # Рассчитываем расстояние до каждого нейрона
        distances = np.linalg.norm(weights - point, axis=1)

        # Находим выигравший нейрон (с минимальным расстоянием)
        winner_neuron = np.argmin(distances)

        # Обновляем веса только для выигравшего нейрона
        weights[winner_neuron] += learning_rate * (point - weights[winner_neuron])

        # Уменьшаем коэффициент обучения
        learning_rate = (50 - epoch) / 100
```

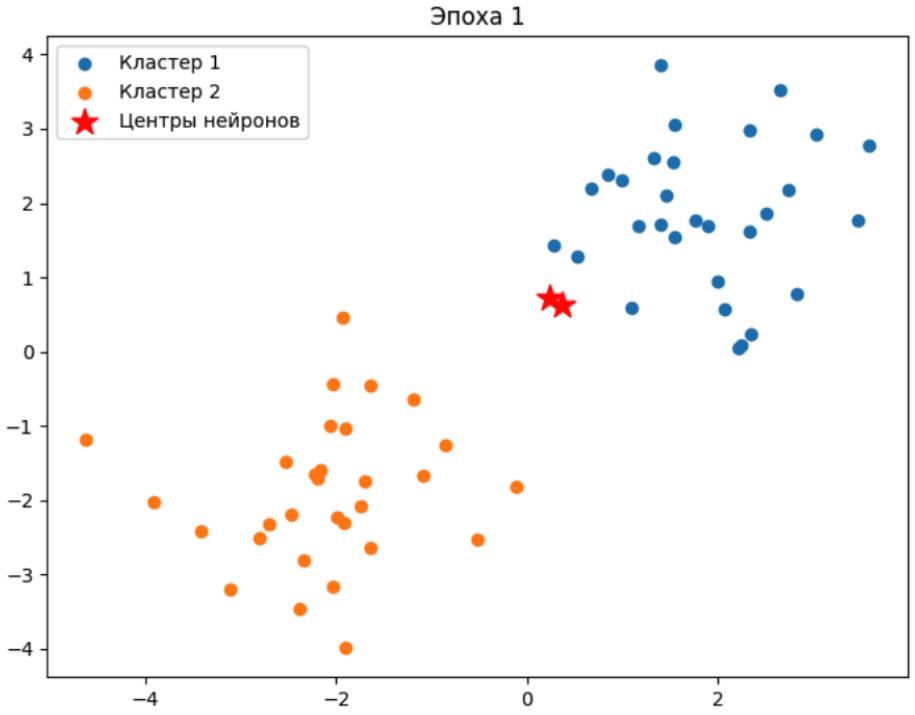


Рисунок 2 - Код и визуализация первой эпохи

Визуализация последней эпохи представлена на рисунке 3.

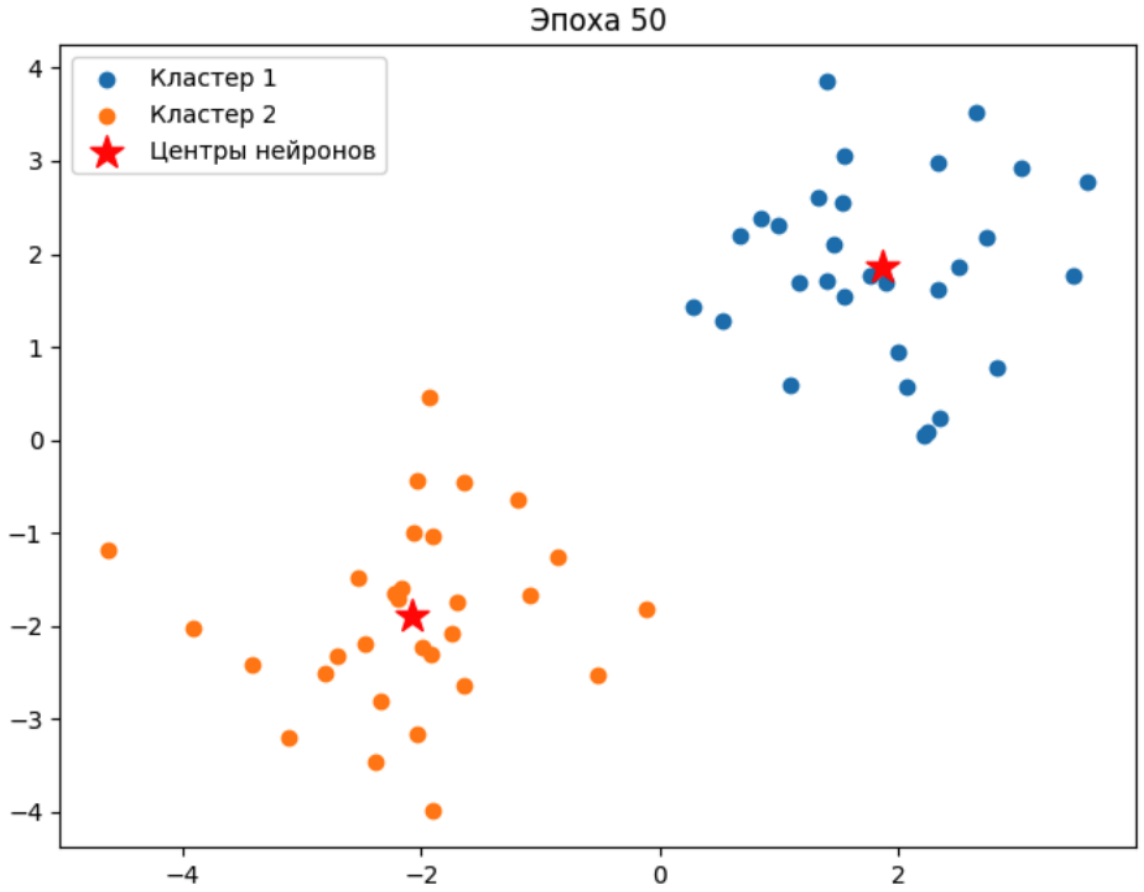


Рисунок 3 - Эпоха 50

После обучения можно использовать сеть для классификации точек, это показано на рисунке 4.

```python
# После обучения можно использовать сеть для классификации точек
def predict(point):
    distances = np.linalg.norm(weights - point, axis=1)
    return np.argmin(distances)
```

```python
# Проверка точек
for point in data:
    prediction = predict(point)
    print(f"Точка {point} принадлежит кластеру {prediction}")
```

```
Точка [2.49671415 1.8617357 ] принадлежит кластеру 0
Точка [2.64768854 3.52302986] принадлежит кластеру 0
Точка [1.76584663 1.76586304] принадлежит кластеру 0
Точка [3.57921282 2.76743473] принадлежит кластеру 0
Точка [1.53052561 2.54256004] принадлежит кластеру 0
Точка [1.53658231 1.53427025] принадлежит кластеру 0
Точка [2.24196227 0.08671976] принадлежит кластеру 0
Точка [0.27508217 1.43771247] принадлежит кластеру 0
Точка [0.98716888 2.31424733] принадлежит кластеру 0
Точка [1.09197592 0.5876963 ] принадлежит кластеру 0
Точка [3.46564877 1.7742237 ] принадлежит кластеру 0
Точка [2.0675282  0.57525181] принадлежит кластеру 0
Точка [1.45561728 2.11092259] принадлежит кластеру 0
Точка [0.84900642 2.37569802] принадлежит кластеру 0
Точка [1.39936131 1.70830625] принадлежит кластеру 0
Точка [1.39829339 3.85227818] принадлежит кластеру 0
Точка [1.98650278 0.94228907] принадлежит кластеру 0
Точка [2.82254491 0.77915635] принадлежит кластеру 0
Точка [2.2088636  0.04032988] принадлежит кластеру 0
Точка [0.67181395 2.19686124] принадлежит кластеру 0
Точка [2.73846658 2.17136828] принадлежит кластеру 0
Точка [1.88435172 1.6988963 ] принадлежит кластеру 0
Точка [0.52147801 1.28015579] принадлежит кластеру 0
Точка [1.53936123 3.05712223] принадлежит кластеру 0
Точка [2.34361829 0.23695984] принадлежит кластеру 0
Точка [2.32408397 1.61491772] принадлежит кластеру 0
Точка [1.323078   2.61167629] принадлежит кластеру 0
Точка [3.03099952 2.93128012] принадлежит кластеру 0
Точка [1.16078248 1.69078762] принадлежит кластеру 0
Точка [2.33126343 2.97554513] принадлежит кластеру 0
Точка [-2.47917424 -2.18565898] принадлежит кластеру 1
Точка [-3.10633497 -3.19620662] принадлежит кластеру 1
Точка [-1.18747418 -0.64375997] принадлежит кластеру 1
Точка [-2.07201012 -0.9964671 ] принадлежит кластеру 1
Точка [-1.63836397 -2.64511975] принадлежит кластеру 1
Точка [-1.63860439 -0.46196343] принадлежит кластеру 1
Точка [-2.03582604 -0.43535634] принадлежит кластеру 1
Точка [-4.6197451 -1.1780975] принадлежит кластеру 1
Точка [-1.91295293 -2.29900735] принадлежит кластеру 1
Точка [-1.90823922 -3.98756891] принадлежит кластеру 1
Точка [-2.21967189 -1.64288743] принадлежит кластеру 1
Точка [-0.52210596 -2.51827022] принадлежит кластеру 1
Точка [-2.8084936  -2.50175704] принадлежит кластеру 1
Точка [-1.08459788 -1.67124889] принадлежит кластеру 1
Точка [-2.5297602  -1.48673257] принадлежит кластеру 1
Точка [-1.90292245 -1.03135501] принадлежит кластеру 1
Точка [-2.70205309 -2.32766215] принадлежит кластеру 1
Точка [-2.39210815 -3.46351495] принадлежит кластеру 1
Точка [-1.70387972 -1.73894473] принадлежит кластеру 1
Точка [-1.99488654 -2.23458713] принадлежит кластеру 1
Точка [-3.41537074 -2.42064532] принадлежит кластеру 1
Точка [-2.34271452 -2.80227727] принадлежит кластеру 1
Точка [-2.16128571 -1.59594914] принадлежит кластеру 1
Точка [-0.1138141  -1.82542219] принадлежит кластеру 1
Точка [-1.74244961 -2.07444592] принадлежит кластеру 1
Точка [-3.91877122 -2.02651388] принадлежит кластеру 1
Точка [-1.93976979  0.46324211] принадлежит кластеру 1
Точка [-2.19236096 -1.69845266] принадлежит кластеру 1
Точка [-2.03471177 -3.16867804] принадлежит кластеру 1
Точка [-0.85717719 -1.24806697] принадлежит кластеру 1
```

Рисунок 4 - Принадлежность точек

Сделаем визуализацию результатов на рисунке 5.

```
: # Визуализация результатов
plt.figure(figsize=(8, 6))
plt.scatter(cluster1[:, 0], cluster1[:, 1], label='Кластер 1')
plt.scatter(cluster2[:, 0], cluster2[:, 1], label='Кластер 2')
plt.scatter(weights[:, 0], weights[:, 1], marker='*', s=200, c='red', label='Центры нейронов')
plt.title('Финальные центры нейронов')
plt.legend()
plt.show()
```

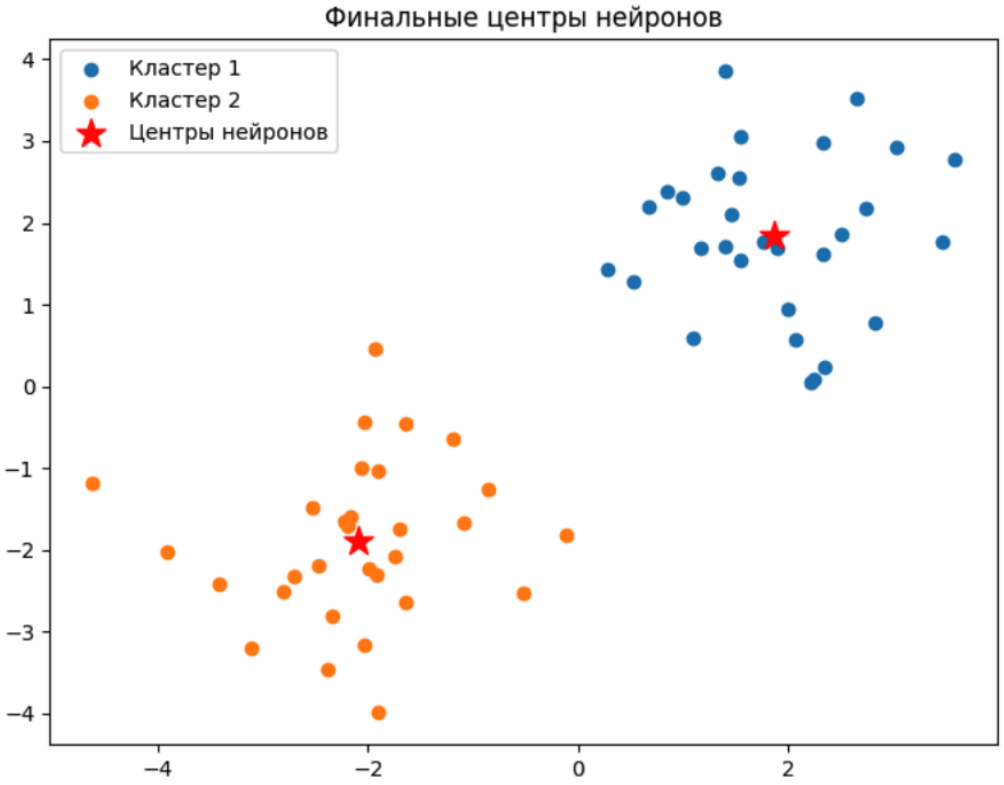


Рисунок 5 - Визуализация результатов

Вывод

В результате выполнения работы были получены практические навыки использования нейронной сети Кохонена с самообучением для задачи кластеризации случайных точек на плоскости вокруг двух центров кластеризации.