Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет Інформатики та Обчислювальної Техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 3
з дисципліни «Методи та технології штучного інтелекту»
На тему:
«Дослідження алгоритму нечіткої кластеризації»

Виконала:
студентка групи ІС-12.
Мельникова К.О.

Перевірив:
Шимкович В. М.

Київ – 2023

Мета роботи: Вирішення практичного завдання кластеризації методами нечіткої логіки.

Завдання:

1. Необхідно сформулювати завдання в галузі обчислювальної техніки або програмування, для якої була б необхідна автоматична класифікація множини об'єктів, які задаються векторами ознак в просторі ознак.

2. Вирішити сформульовану задачу з використанням механізму кластеризації методами нечіткої логіки за допомогою програмних засобів моделювання або мови програмування високого рівня.

3. Знайти центри кластерів і побудувати графік зміни значень цільової функції.

4. Оформіть звіт по лабораторній роботі.

Хід роботи:

1. Виконаємо задачу на прикладі датасету iris бібліотеки sklearn, що містить опис 3 різних сортів квіток ірису за різними ознаками.

Для виконання даної задачі буде достатньо лише двох ознак: ширини та довжини чашолистку.

Імпотуємо бібліотеки та датасет. Окрім даних про параметри чашолистків ірисів імпортуємо також значення міток, які потім порівняємо зі значеннями отриманими в результаті кластеризації.

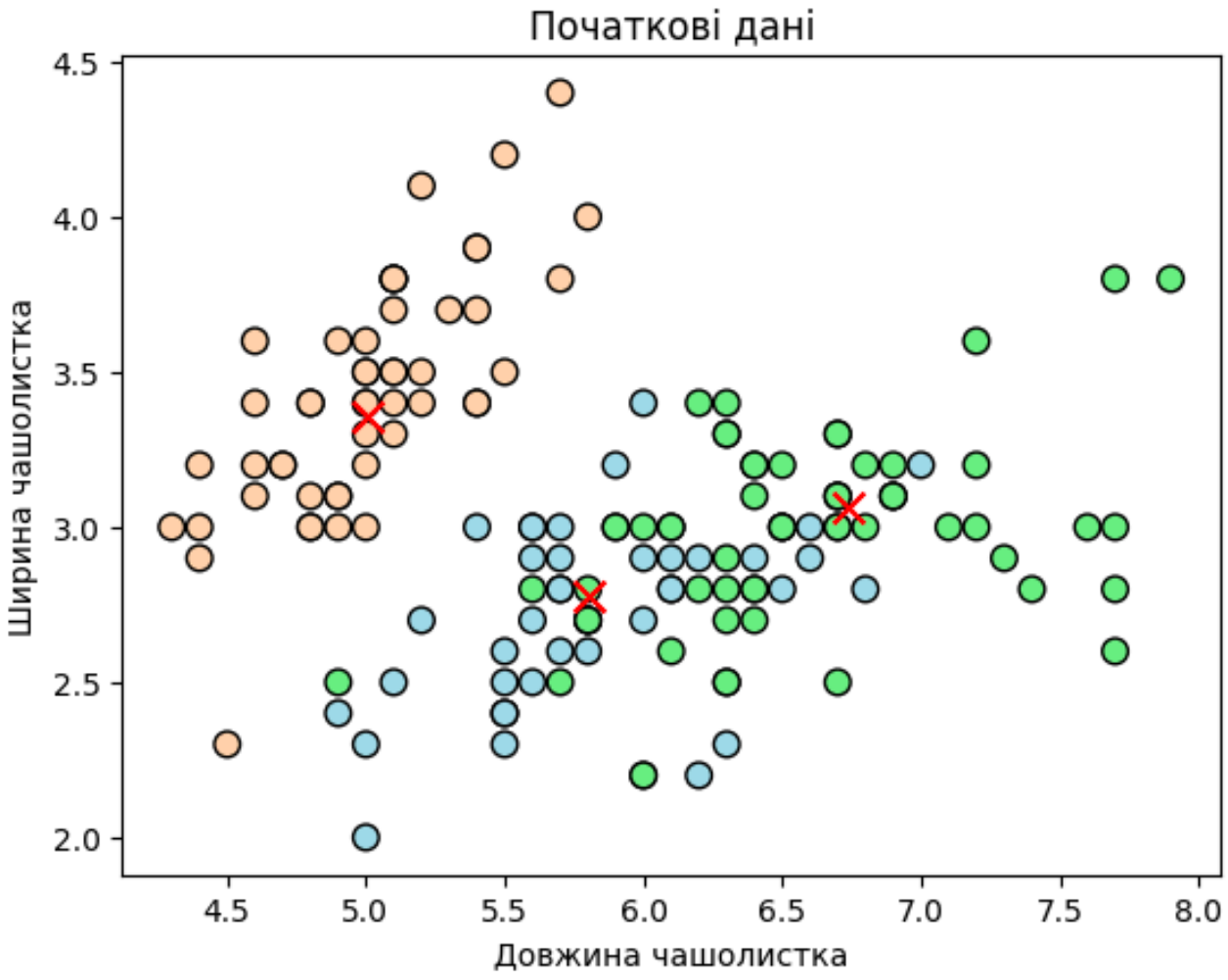
```
from matplotlib.colors import ListedColormap
import numpy as np
import skfuzzy as fuzz
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris

iris = load_iris()
x= iris.data[:, :2] # значення X – ширина та довжина чашолистка квітки ірису
y = iris.target # значення сортів, пізніше ми порівняємо їх з результатами отриманими при кластеризації
```

Виконаємо алгоритм нечіткої кластеризації використовуючи метод cmeans:

```
number_of_clasters = 3 # к-кість сортів ірису

# виконаємо кластеризацію за допомогою методу cmeans
cntr, distribution_matrix, _, _, jm, _, _ = fuzz.cluster.cmeans(x.T, number_of_clasters, 3, error=0.005, maxiter=20, init=None)
```

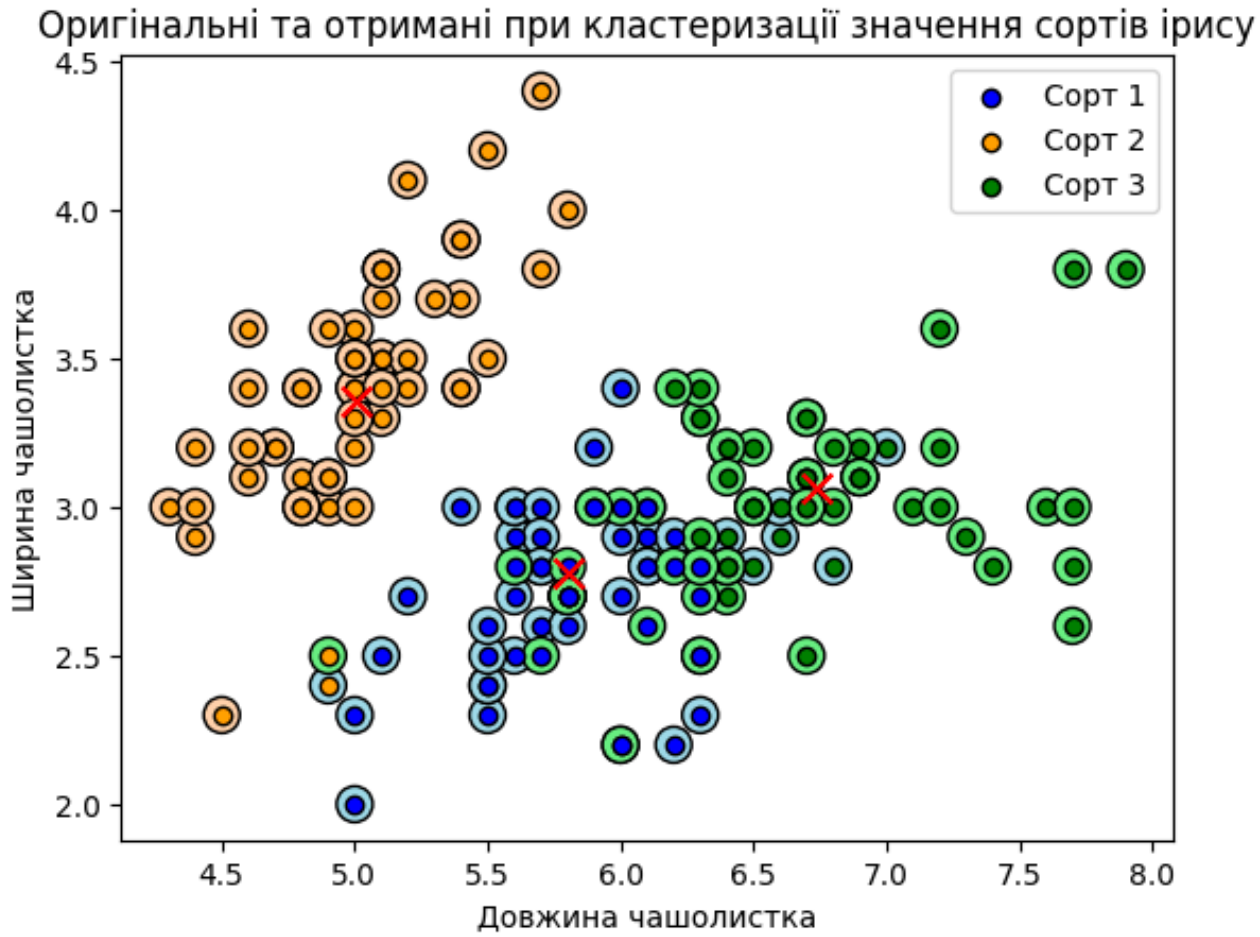
Візуалізуємо початкові дані та центри кластерів:

```
# визначимо кольори
colors_def = ((0.99, 0.84, 0.69), (0.68, 0.85, 0.9), 'lightgreen')
def_color_map = ListedColormap(colors_def[:len(np.unique(y))])

# візуалізуємо початкові дані та центри кластерів
plt.title('Початкові дані')
plt.scatter(x[:, 0], x[:, 1], c=y, cmap=def_color_map, edgecolor='black', s=70)
plt.scatter(cntr[:, 0], cntr[:, 1], marker='x', s=100, color='red')
plt.xlabel('Довжина чашолистка')
plt.ylabel('Ширина чашолистка')
plt.show()
```



Визначимо якому кластеру належить кожна точка (до якого сорту належить квітка ірису), а також візуалізуємо та порівняємо оригінальні та отримані при виконанні алгоритму нечіткої кластеризації значення сортів ірису:

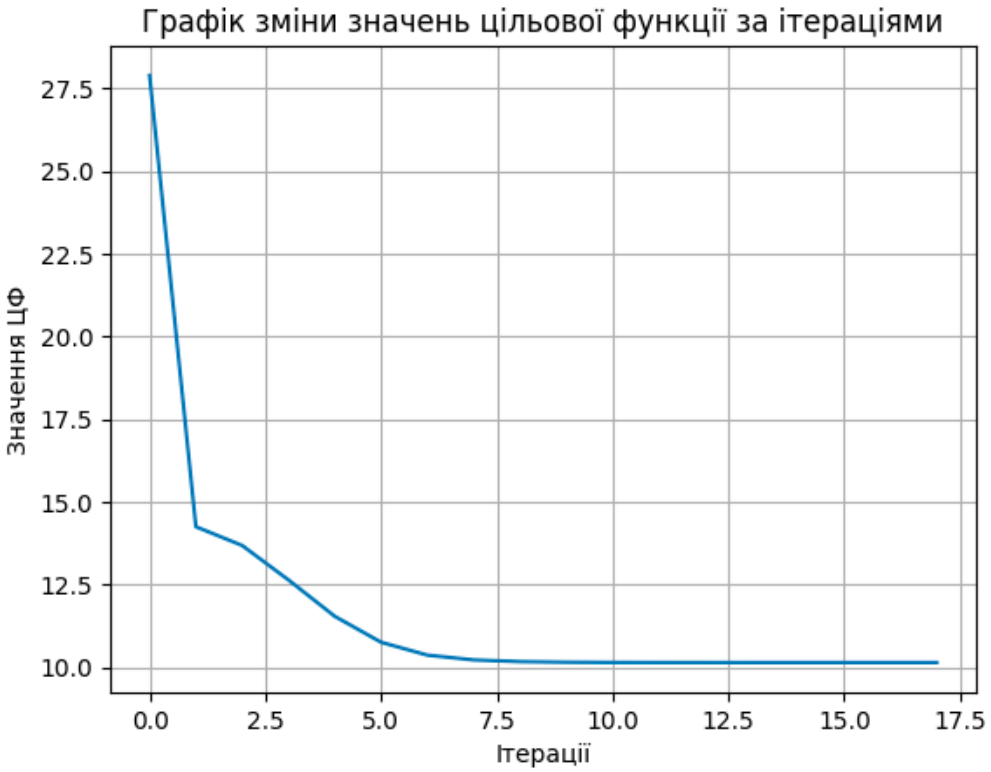
```python
# визначимо якому кластеру належить кожна точка (до якого сорту належить квітка ірису)
fuzzy_labels = np.argmax(distribution_matrix, axis=0)
col_dictionary = {5.1: 'orange', 7.0: 'green', 5.5: 'blue'}
plt.scatter(x[:, 0], x[:, 1], c=y, cmap=def_color_map, s=150, edgecolor='black')
for i in range(number_of_clasters):
    cluster_points = x[fuzzy_labels == i]
    plt.scatter(cluster_points[:, 0], cluster_points[:, 1], label=f'Сорт {i + 1}', edgecolor='black', color=col_dictionary[cluster_points[0][0]])

plt.scatter(cntr[:, 0], cntr[:, 1], marker='x', s=100, color='red')

plt.title('Оригінальні та отримані при кластеризації значення сортів ірису')
plt.xlabel('Довжина чашолистка')
plt.ylabel('Ширина чашолистка')
plt.legend()
plt.show()
```



Як бачимо, отримані результати майже ідентичні вихідним міткам.

Тепер виведемо графік зміни значень цільової функції за ітераціями:

```
# Графік зміни значень цільової функції за ітераціями
plt.title('Графік зміни значень цільової функції за ітераціями')
plt.plot(jm)
plt.xlabel('Ітерації')
plt.ylabel('Значення ЦФ')
plt.grid(True)
plt.show()
```



Висновок:

Отже, під час виконання даної лабораторної роботи було розроблено програмне забезпечення на мові програмування високого рівня Python з використанням вбудованих бібліотек та функцій, що на прикладі датасету iris бібліотеки sklearn виконує задачу кластеризації з використанням алгоритму нечіткої кластеризації FCA. Крім цього, було виконано необхідну візуалізацією даних, центрів кластерів та цільової функції FCA, а також порівняно отримані дані з очікуваними. Результат даного порівняння свідчить про коректну роботу алгоритму.