**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ**

**СІКОРСЬКОГО»**

**Навчально-науковий інститут прикладного системного аналізу**

**Кафедра штучного інтелекту**

**Комп’ютерний практикум № 1**

**з дисципліни «Інтелектуальний аналіз даних»**

**Тема «Отримання навичок роботи в середовищі Python»**

Виконали:

студенти 3 курсу

групи КІ-32

Бондарєва А. С.,

Власенко В. Ю.,

Монагарова А. М.,

Нємцов О. В.

Прийняв: Андросов Д.В.

**Київ – 2025**

**Варіант 6**

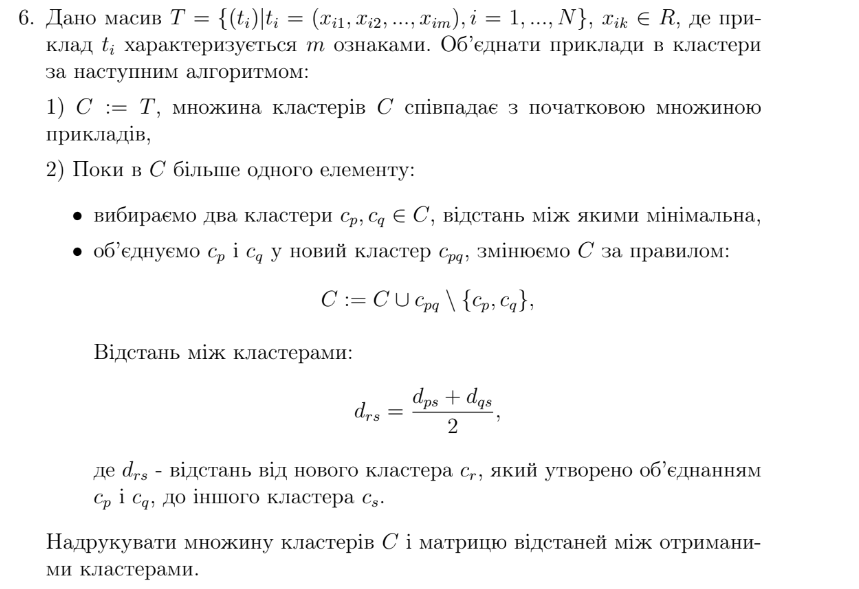


Рисунок 1 – Умова завдання

**Теоретична частина**

У даній лабораторній роботі реалізовано алгоритм ієрархічної кластеризації. Початково кожен приклад вважається окремим кластером. Далі на кожному кроці знаходяться два найближчі кластери та об'єднуються у новий. Відстань між новим кластером та іншими обчислюється за формулою:

де drs — відстань від нового кластера до іншого кластера cs, dps та dqs — відстані від об'єднуваних кластерів cp та cq до cs.

Для виконання поставленої задачі, було розроблено програму мовою Python в середовищі розробки Visual Studio Code.

**Лістинг програми**

import numpy as np

def euclidean\_distance(x, y):

# евклідова відстань між двома векторами

return np.sqrt(np.sum((x - y) \*\* 2))

def build\_distance\_matrix(clusters):

# матриця відстаней між усіма кластерами

n = len(clusters)

dist\_matrix = np.zeros((n, n))

for i in range(n):

for j in range(i+1, n):

d = min(euclidean\_distance(a, b) for a in clusters[i] for b in clusters[j])

dist\_matrix[i, j] = d

dist\_matrix[j, i] = d

return dist\_matrix

def format\_cluster(cluster):

# форматуємо кластер для нормального виведення

return "{" + ", ".join(f"({p[0]:.0f}, {p[1]:.0f})" for p in cluster) + "}"

def hierarchical\_clustering(data):

clusters = [[point] for point in data]

dist\_matrix = build\_distance\_matrix(clusters)

step = 1

while len(clusters) > 1:

n = len(clusters)

# знаходження двох найближчих кластерів

np.fill\_diagonal(dist\_matrix, np.inf)

p, q = np.unravel\_index(np.argmin(dist\_matrix), dist\_matrix.shape)

print(f"\nКрок {step}. об'єднуємо кластери {p} і {q} (відстань = {dist\_matrix[p, q]:.3f})")

# об'єднуємо кластери

new\_cluster = clusters[p] + clusters[q]

# оновляємо список кластерів

new\_clusters = [clusters[k] for k in range(n) if k not in (p, q)]

new\_clusters.append(new\_cluster)

# оновляємо матрицю відстаней

new\_n = len(new\_clusters)

new\_dist = np.zeros((new\_n, new\_n))

mask = [k for k in range(n) if k not in (p, q)]

for i in range(len(mask)):

for j in range(i+1, len(mask)):

new\_dist[i, j] = dist\_matrix[mask[i], mask[j]]

new\_dist[j, i] = new\_dist[i, j]

for s in range(len(mask)):

dps = dist\_matrix[p, mask[s]]

dqs = dist\_matrix[q, mask[s]]

drs = (dps + dqs) / 2

new\_dist[s, new\_n-1] = drs

new\_dist[new\_n-1, s] = drs

clusters = new\_clusters

dist\_matrix = new\_dist

# друк кластерів

print("кластери:")

for idx, cl in enumerate(clusters):

print(f" {idx}: {format\_cluster(cl)}")

# друк матриці відстаней

print("Матриця відстаней:")

for row in dist\_matrix:

print(" ", [f"{x:.2f}" for x in row])

step += 1

T = np.array([

[1.0, 2.0],

[2.0, 1.0],

[5.0, 4.0],

[6.0, 5.0]

])

hierarchical\_clustering(T)

**Результати роботи програми**

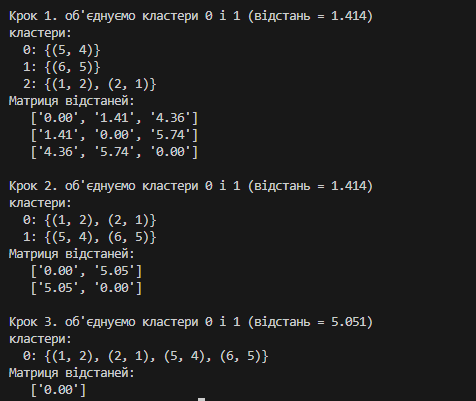


Рисунок 2 – Результат виконання коду програми

**Реалізація програми**

Отже, наведена вище програма реалізує агломеративну ієрархічну кластеризацію. На самому початку кожна точка розглядається як окремий кластер, далі покроково будується матриця відстаней між усіма поточними кластерами, і щоразу обирається пара з мінімальною відстанню. Вони об’єднуються в новий кластер. Відстань від цього новоствореного кластера до решти визначається шляхом того, що ми беремо середнє значення відстаней до кожного з двох попередніх.

Такий підхід дозволяє, так би мовити, «нарощувати» кластери поступово, бо спершу групуються найближчі точки, а далі вже сформовані невеликі кластери, і потім все об’єднується в одну групу. На кожному кроці виводиться поточний склад нових кластерів та оновлену матрицю відстаней, тому наочно можна побачити процес кластеризації, як і згодом фінальний результат. Також, після кожного злиття матриця відстаней повністю перебудовується, і це має гарантувати правильність вибору найближчих кластерів на кожному наступному кроці.

**Висновки**

У ході виконання лабораторної роботи, було реалізовано алгоритм агломеративної ієрархічної кластеризації з покроковим відстеженням результатів. Кластери об’єднуються один з одним, формуючи ієрархічну структуру, тобто, процес відбувається від маленьких пар точок до єдиної групи. І отже, ми бачимо, що отримані результати є очікуваними і не суперечать теоретичній частині, оскільки алгоритм знаходить найближчі кластери, послідовно об’єднує їх і наочно показує свій хід роботи.