Информатика. Задание 27. Кластеризация. Функции.

1 Задача

Учёный решил провести кластеризацию некоторого множества звёзд по их расположению на карте звёздного неба. Кластер звёзд – это набор звёзд (точек) на графике, лежащий внутри круга радиусом R. Каждая звезда обязательно принадлежит только одному из кластеров. Истинный центр кластера, или центроид, – это одна из звёзд на графике, сумма расстояний от которой до всех остальных звёзд кластера минимальна. Под расстоянием понимается расстояние Евклида между двумя точками $A(x_1, y_1)$ и $B(x_2, y_2)$ на плоскости, которое вычисляется по формуле:

$$d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Аномалиями назовём точки, находящиеся на расстоянии более одной условной единицы от точек кластеров. При расчётах аномалии учитывать не нужно.

В файле А хранятся данные о звёздах двух кластеров, где $R = 6$ для каждого кластера. В каждой строке записана информация о расположении на карте одной звезды: сначала координата x, затем координата y. Значения даны в условных единицах, которые представлены вещественными числами. Известно, что количество звёзд не превышает 2000.

В файле Б хранятся данные о звёздах пяти кластеров, где $R = 6$ для каждого кластера. Известно, что количество звёзд не превышает 15000. Структура хранения информации о звездах в файле Б аналогична файлу А.

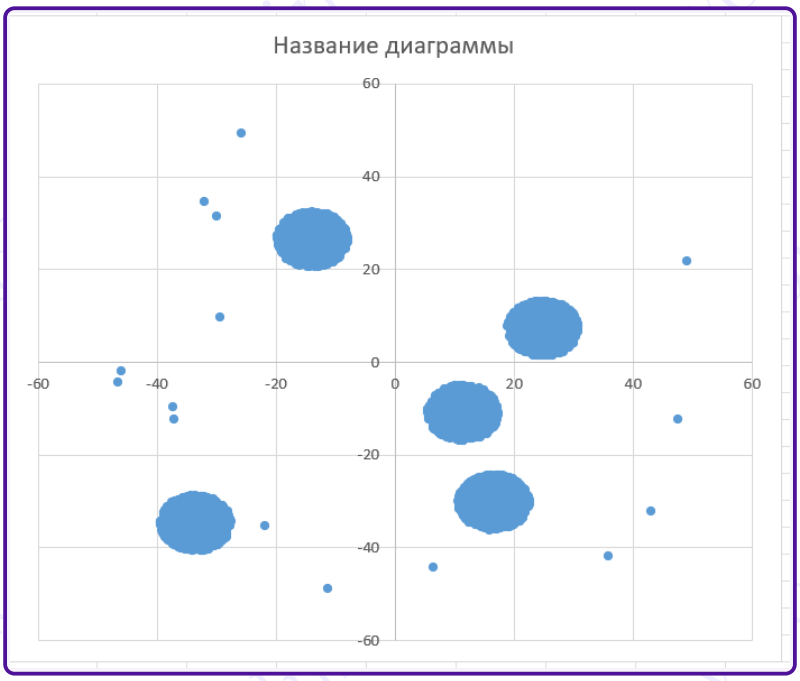
Для каждого файла определите координаты центра каждого кластера, затем вычислите два числа: P_x — среднее арифметическое абсцисс центров кластеров, и P_y – среднее арифметическое ординат центров кластеров.

В ответе запишите четыре числа через пробел: сначала целую часть произведения $P_x \cdot 1000$ для файла А, затем $P_y \cdot 1000$ для файла А, далее целую часть произведения $P_x \cdot 1000$ для файла Б и $P_y \cdot 1000$ для файла Б. Возможные данные одного из файлов иллюстрированы графиком.

Внимание! График приведён в иллюстративных целях для произвольных значений, не имеющих отношения к заданию. Для выполнения задания используйте данные из прилагаемого файла.

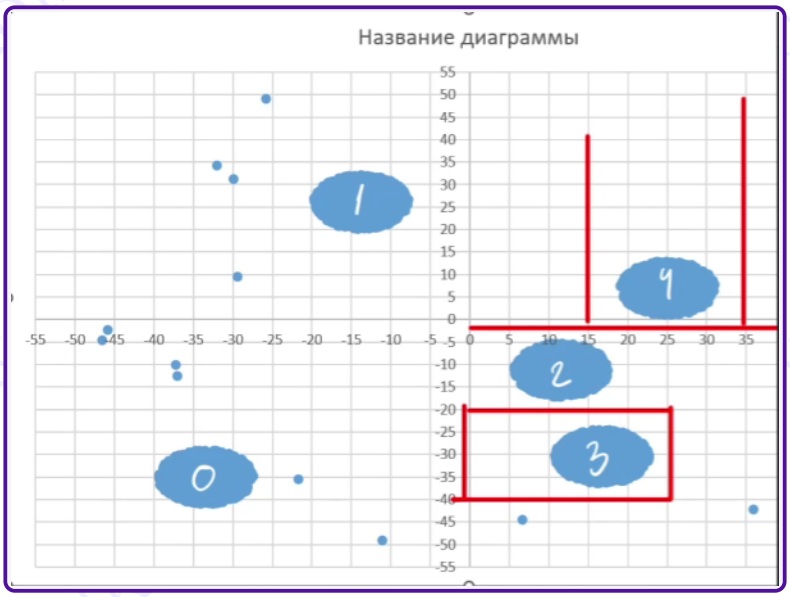
Решение:

Шаг 1 – визуализация кластеров на точечном графике в Excel



Первый вариант разделения на кластеры:

Шаг 2 – разделение на кластеры по прямым, параллельным оси абсцисс и оси ординат.



Шаг 3 – считывание данных из файла и разделение кластеров.

```
from math import dist
# модуль для нахождения расстояния между точками
f = open('E:/test1/275.txt') # открываем файл
a = [list(map(float, i.replace(',', '').split())) for i in f]
# считываем данные из файла, заменяем запятую на точку
# и разделяем число по пробелу
clusters = [[] for i in range(5)] # создаём список для кластеров
for i in a: # проходимся по звёздам, а точнее их координатам
    x, y = i # получаем координаты звезды
    # разделяем на кластеры:
    if x < -25 and y < -25:
        clusters[0].append(i)
    elif y > 15 and x > -23 and x < 0:
        clusters[1].append(i)
    elif 0 < x < 20 and 0 > y > -20:
        clusters[2].append(i)
    elif 0 < x < 25 and -20 > y > -40:
        clusters[3].append(i)
    elif y > -3 and 15 < x < 35:
        clusters[4].append(i)
```

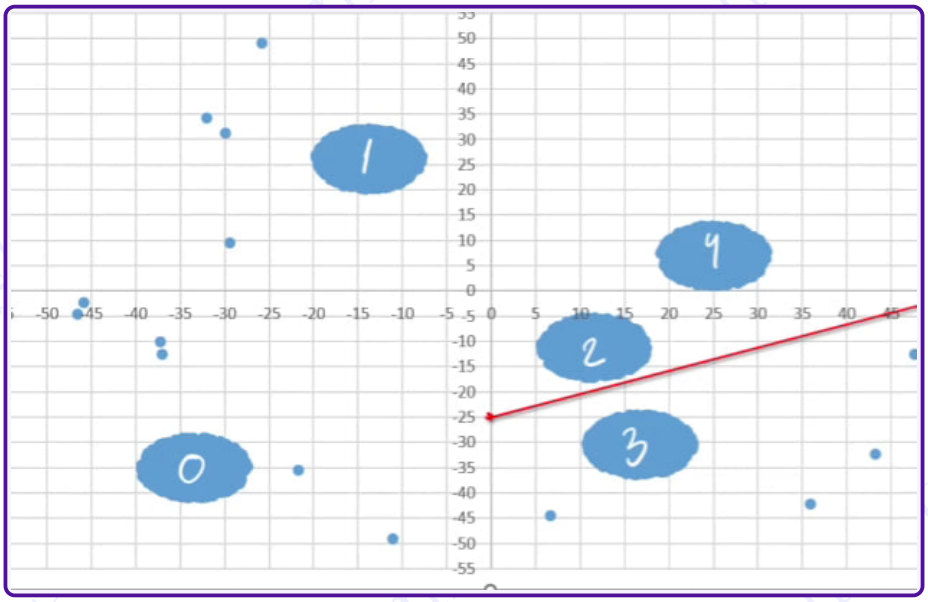
Шаг 4 – поиск центроида.

```
# Создаем пустой список для хранения центроидов (центральных точек) кластеров
centers = []
# перебираем каждый кластер из списка clusters
for j in clusters:
    # инициализируем минимальную сумму расстояний очень большим числом
    mn = 100000000500000000000
    cnt = [] # создаем переменную для хранения текущего центроида
    # перебираем каждую точку в кластере как потенциальный центроид
    for stan in j:
        sm = 0 # обнуляем сумму расстояний для текущей точки
# считаем сумму расстояний от текущей точки до всех других точек кластера
        for i in j:
            # dist() вычисляет евклидово расстояние между точками
            sm += dist(stan, i)
        # если текущая сумма расстояний меньше минимальной найденной
        if sm < mn:
            mn = sm # обновляем минимальную сумму
            cnt = stan # запоминаем точку с минимальной суммой как центроид
    centers.append(cnt) # добавляем найденный центроид в итоговый список
```

Шаг 5 – проверка.

```
# визуализация уже разделенных кластеров
from turtle import *
tracer(0)
m = 10
pu()
colors = ['red', 'green', 'blue', 'purple', 'black']
# проверка: верно ли распределны кластеры
for j in range(5):
    for i in clusters[j]:
        x, y = i
        goto(x*m, y*m)
        dot(5, colors[j])
# проверка: верно ли найдены центры кластеров
for i in centers:
    x, y = i
        goto(x*m, y*m)
        dot(15, "yellow")
done()
```

Второй вариант разделения на кластеры:



Вывод прямой по формуле:

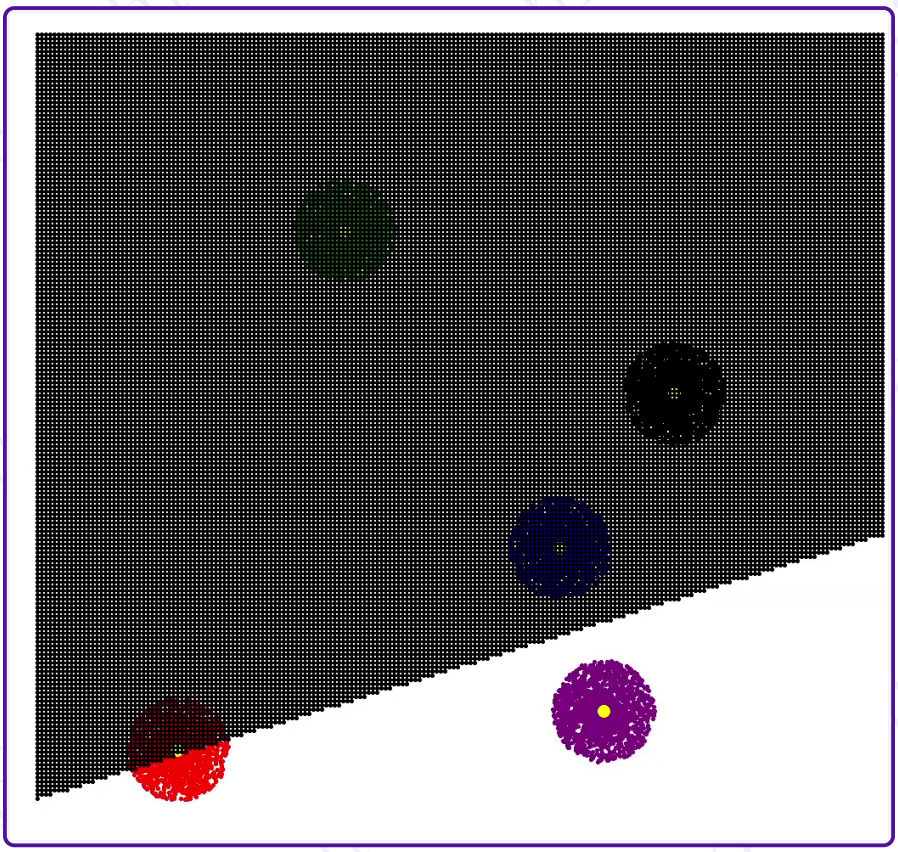
$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1};$$
$$x_1 = 0, x_2 = 55;$$
$$y_1 = -25, y_2 = 0;$$
$$\frac{y + 25}{0 + 25} = \frac{x - 0}{55 - 0};$$
$$\frac{y + 25}{25} = \frac{x}{55}.$$

Тогда можно заменить условие второго кластера на:

```
    elif 0 < x < 20 and y < 0 and (y + 25)/25 > x/80:
        clusters[2].append(i)
```

И добавить проверку на то, что прямая верно разделяет кластеры:

```
from turtle import *
tracer(0)
m = 10
for i in range(-100, 100):
    for j in range(-100, 100):
        x, y = i/10, j/10
        goto(x*m, y*m)
        if (y+25)/25 > x/80:
            dot(5)
done()
```



Третий вариант разделения на кластеры:

Отделим второй кластер от других через уравнение окружности.

Центр второго кластера: y = -12,x = 12, r = 8

Формула окружности:

$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

Тогда получим:

$$(x - 12)^2 + (y + 12)^2 = 64$$

Заменим условие второго кластера на:

```
    elif (x-12)**2 + (y+12)**2 == 64:
        clusters[2].append(i)
```

Добавим проверку на то, что верно отделили кластер окружностью:

```
for i in range(-100, 100):
    for j in range(-100, 100):
        x, y = i/2, j/2
        goto(x*m, y*m)
        if (x-12)**2 + (y+12)**2 < 64:
            dot(5, "yellow")
done()
```

Шаг 6 – выводим результаты.

```
px = py = 0
for i in centers:
        px += i[0]
        py += i[1]
print(int(px/5*1000), int(py/5*1000))
```

Ответ: 1155 – 8441