Информатика. Задания на кластеризацию. Повторение материала.

Учёный решил провести кластеризацию некоторого множества звёзд по их расположению на карте звёздного неба. Кластер звёзд – это набор звёзд (точек) на графике, каждая из которых находится от хотя бы одной другой звезды на расстоянии не более R условных единиц. Каждая звезда обязательно принадлежит только одному из кластеров.

Двойная звездная система – это две звезды на расстоянии менее t. При этом других звезд на расстоянии менее t у этих двух звезд быть не должно.

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 хранятся данные о звёздах кластеров, где $R = 0.4$, $t = 0.04$ для каждого кластера. В каждой строке записана информация о расположении на карте одной звезды, а также ее масса (в солнечных массах): сначала координата x, затем координата y, затем масса m. В случае, если масса представлена положительным числом, объект является звездой, если отрицательным – объект является нейтронной звездой либо черной дырой.

Для каждого файла в каждом кластере найдите двойную звездную систему состоящую из нейтронной звезды карлика (масса по модулю от 1.5 до 2.8 солнечных масс) и черной дыры (масса по модулю от 2.9 солнечных масс).

Решение

Для удобства восприятия разобъем наш код по частям:

1. Создадим функцию *dbscan*(), так как в задаче она нам понадобится несколько раз:

```python
from math import dist
# Функция для кластеризации звёзд с использованием алгоритма DBSCAN
# a: Список звёзд, каждая из которых представлена как [x, y, масса]
# r: Радиус для определения соседства звёзд
# return: Список кластеров, каждый из которых представляет собой список звёзд
def dbscan(a, r, tipulya = 0):
    cl = []  # Список для хранения кластеров
    while a:  # Пока есть звёзды в списке
        if tipulya == 1 and a[0][2] > 0:
            a.pop(0)
        else:
            cl.append([a.pop(0)])  # Начинаем новый кластер с первой звезды
            for i in cl[-1]:  # Для каждой звезды в текущем кластере
                for j in a:  # Проверяем каждую оставшуюся звезду
    # Если расстояние между звёздами i и j меньше или равно r
                    if dist(i[:2], j[:2]) <= r:
                        cl[-1].append(j)  # Добавляем j в текущий кластер
                        a.remove(j)  # Удаляем j из списка оставшихся звёзд
    return cl  # Возвращаем список кластеров
```

2. Откроем наш файл, считаем его в список a, а затем воспользуемся созданной функцией *dbscan*(), чтобы разбить звезды на кластеры:

```python
f = open('27-1b.txt')
a = [list(map(float, i.split())) for i in f]
r = 0.4
clusters = dbscan(a, r)
```

3. Теперь переходим к поиску двойной звездной системы.

Основная мысль заключается в том, что нам нужно пройти по каждой точке в четырех найденных кластерах и с помощью уже созданной функции $dbscan()$ для каждого кластера найти списки звезд, расстояние между которыми менее 0.04.

Далее в каждом кластере нужно оставить только те списки, в которых количество звезд равно двум – то есть только двойные звездные системы, а также в звездной системе содержится черная дыра и нейтронная звезда.

```
c = 0
for i in clusters: # Для каждого кластера
    stars = dbscan(i, 0.04, 1) # Находим двойные системы внутри кластера
    for j in stars:
        if len(j) == 2: # Проверяем, что в кластере два объекта
            mass = sorted([j[0][2], j[1][2]])
            if (-10000 <= mass[0] <= -2.9) and (-2.8 <= mass[1] <= -1.5):
                c += 1
print(c)
```

В результате совмещения всех трех частей в полноценный код получаем решение задачи:

```
from math import dist

def dbscan(a, r, tipulya = 0):
    cl = []
    while a:
        if tipulya == 1 and a[0][2] > 0:
            a.pop(0)
        else:
            cl.append([a.pop(0)])
            for i in cl[-1]:
                for j in a:
                    if dist(i[:2], j[:2]) <= r:
                        cl[-1].append(j)
                        a.remove(j)
    return cl


f = open('27-1b.txt')
a = [list(map(float, i.split())) for i in f]
r = 0.4
clusters = dbscan(a, r)

c = 0
for i in clusters:
    stars = dbscan(i, 0.04, 1)
    for j in stars:
        if len(j) == 2:
            mass = sorted([j[0][2], j[1][2]])
            if (-10000 <= mass[0] <= -2.9) and (-2.8 <= mass[1] <= -1.5):
                c += 1
```

```
print(c)
```