Информатика. Кластеризация. DBSCAN.

Содержание

1 Разбор задач с вебинара

1.1 Пример 1

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 система – это два объекта 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 А хранятся данные о звёздах **четырех** кластеров, где где $R = 0.5$, $t = 0.01$ 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. Значения даны в условных единицах, которые представлены вещественными числами. Известно, что количество звёзд не превышает 3000.

В файле Б хранятся данные о звёздах **семи** кластеров, где $R = 0.5$, $t = 0.01$ для каждого кластера. Известно, что количество звёзд не превышает 10 000. Структура хранения информации о звездах в файле Б аналогична файлу А.

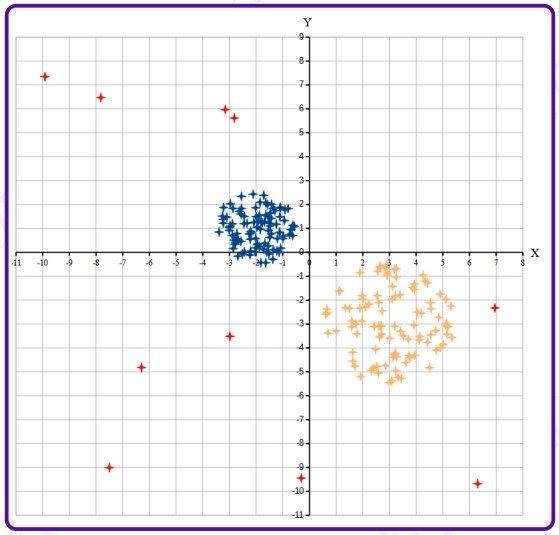
Для каждого файла в каждом кластере найдите двойную систему состоящую из двух черных дыр с минимальной разницей масс. Масса нейтронной звезды не превышает 2.7 солнечных (по модулю), дальше – черные дыры. Затем вычислите два числа: P_x — среднее арифметическое абсцисс найденных черных дыр, и P_y – среднее

арифметическое ординат найденных черных дыр.

В ответе запишите четыре числа через пробел: сначала целую часть произведения $P_x \cdot 100$ для файла А, затем $P_y \cdot 100$ для файла А, далее целую часть произведения $P_x \cdot 100$ для файла Б и $P_y \cdot 100$ 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 (файл Б):

```
from math import dist # Функция для вычисления расстояния Евклида
# Функция для кластеризации
def dbscan(a, r):
    c = [] # Создаем массив для кластеров
    while a: # Пока в списке 'a' есть звезды
        c.append([a.pop(0)]) # Создаем кластер и добавляем первую звезду
        for i in c[-1]: # Для каждой звезды в последнем кластере
            for j in a[:]: # Для каждой звезды в списке 'a'
                if dist(i[:2], j[:2]) < r: # Если рядом есть звезда
                    c[-1].append(j) # Добавляем звезду в кластер
                    a.remove(j) # Удаляем ее из исходного списка
    return c # Возвращаем список кластеров
```

```
f = open('27-1.txt') # Открываем файл
s = f.readline() # Считываем первую строку с названием столбцов
# Записываем координаты звезд и их массы в список 'a'
a = [list(map(float, i.replace(',', '.').split())) for i in f]
r = 0.5 # Радиус кластеризации
t = 0.01 # Радиус двойной системы
clusters = dbscan(a, r) # Кластеризуем звезды
px = py = 0 # Искомые средн. арифметические

for i in clusters: # Проход по кластерам
    if len(i) > 3: # Если элементов больше, чем 3 (аномалии отбрасываем)
        stars = dbscan(i, t) # Кластеризуем внутри кластера с радиусом t
        mn = 10**100 # Минимальное расстояние
        for j in stars:
            if len(j) == 2: # Если в кластере два обьекта
                x, y = j[0][2], j[1][2] # Получаем массы звезд
                if x < -2.7 and y < -2.7:  # Если обе массы меньше -2.7
                    # Если разница масс меньше минимальной
                    if abs(abs(x) - abs(y)) < mn:
                        # Обновляем минимальую разницу
                        mn = abs(abs(x) - abs(y))
                        # Сохраняем пару с минимальной разницей масс
                        syst = j
        px += syst[0][0] + syst[1][0] # Сумма абсцисс
        py += syst[0][1] + syst[1][1] # Сумма ординат

px = int((px / 14) * 100) # Среднее арифметическое абсцисс
py = int((py / 14) * 100) # Среднее арифметическое ординат

print(px, py)
```

Ответ: 244 18

1.2 Пример 2

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. 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 $A(x1, y1)$ и $B(x2, y2)$ на плоскости, которое вычисляется по формуле:

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

В файле А хранятся данные о звёздах **двух** кластеров, где где $R = 0.2$ 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. Значения даны в условных единицах, которые представлены вещественными числами. Известно, что количество звёзд не превышает 2000.

В файле Б хранятся данные о звёздах **четырех** кластеров, где $R = 0.2$ для каждого кластера. Известно, что количество звёзд не превышает 10000. 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 найденных черных дыр, и P_y – среднее арифметическое ординат найденных черных дыр.

В ответе запишите четыре числа через пробел: сначала целую часть произведения $P_x \cdot 10000$ для файла А, затем $P_y \cdot 10000$ для файла А, далее целую часть произведения $P_x \cdot 10000$ для файла Б и $P_y \cdot 10000$ для файла Б.

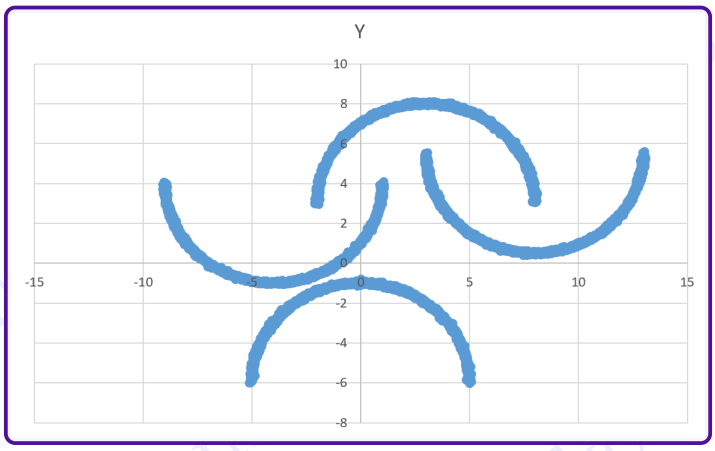
Решение (файл Б):

```
from math import dist # Функция для вычисления расстояния Евклида
# Функция для кластеризации
def dbscan(a, r):
    c = [] # Создаем массив для кластеров
    while a: # Пока в списке 'a' есть звезды
        c.append([a.pop(0)]) # Создаем кластер и добавляем первую звезду
        for i in c[-1]: # Для каждой звезды в последнем кластере
            for j in a[:]: # Для каждой звезды в списке 'a'
                if dist(i[:2], j[:2]) < r: # Если рядом есть звезда
                    c[-1].append(j) # Добавляем звезду в кластер
                    a.remove(j) # Удаляем ее из исходного списка
    return c # Возвращаем список кластеров
```

Найдем центроид первого кластера:

```
f = open('27-2.txt') # Открываем файл
s = f.readline() # Считываем первую строку с названием столбцов
# Записываем координаты звезд в список 'a'
a = [list(map(float, i.replace(',', '.').split())) for i in f]
r = 0.2 # Радиус кластеризации
clusters = dbscan(a, r) # Кластеризуем звезды
mn =10**100 # Минимальная сумма расстояний
center = [] # Координаты центроида
for star in clusters[0]: # Проходим по каждой звезде в кластере
    sm = 0  # Суммы расстояний между текущей и другими звездами
    for j in clusters[0]:  # Проходим по всем звездам в кластере
        sm += dist(star, j) # Добавляем расстояние между звездами
    if sm < mn: # Если сумма расстояний меньше минимальной
        mn = sm # Обновляем минимальную сумму
        center = star # Запоминаем координаты центроида
print(center) # Получили 3.089315787, 7.892028053
```

Построим в Excel точечную диаграмму:



И убедимся, что центроид одного из кластеров находится примерно в точке с координатами 3.089315787, 7.892028053.

Подкорректируем код для всех кластеров, добавим цикл:

```
clusters = dbscan(a, r) # Кластеризуем звезды
px = py = 0 # Искомые средн. арифметические
for i in clusters:
    mn =10**100 # Минимальная сумма расстояний
    center = [] # Координаты центроида
    for star in i: # Проходим по каждой звезде в кластере
        sm = 0  # Суммы расстояний между текущей и другими звездами
        for j in i:  # Проходим по всем звездам в кластере
            sm += dist(star, j) # Добавляем расстояние между звездами
        if sm < mn: # Если сумма расстояний меньше минимальной
            mn = sm # Обновляем минимальную сумму
            center = star # Запоминаем координаты центроида
    px += center[0]
    py += center[1]
px = int((px / 4) * 10000)  # Среднее арифметическое абсцисс
py = int((py / 4) * 10000)  # Среднее арифметическое ординат
print(px, py)
```

Ответ: 18208 16188

1.3 Пример 3

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 система – это два объекта 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 А хранятся данные о звёздах **двух** кластеров, где где $R = 0.6$, $t = 0.02$ 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. Значения даны в условных единицах, которые представлены вещественными числами. Известно, что количество звёзд не превышает 6000.

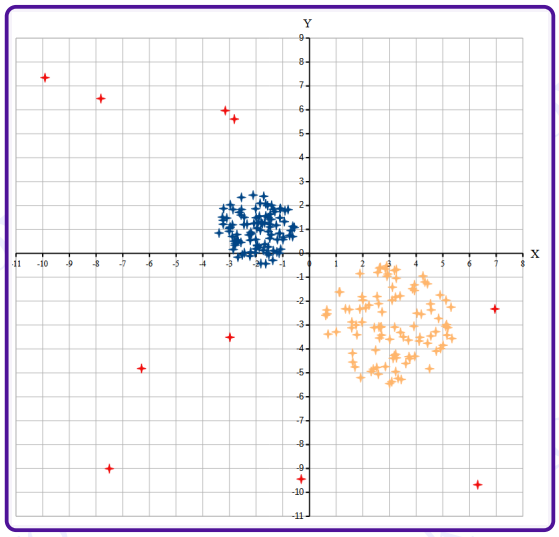
В файле Б хранятся данные о звёздах **трех** кластеров, где $R = 0.6$, $t = 0.03$ для каждого кластера. Известно, что количество звёзд не превышает 16000. Структура хранения информации о звездах в файле Б аналогична файлу А.

Для каждого файла в каждом кластере найдите двойную систему состоящую из двух нейтронных звезд с максимальной разницей масс. Масса нейтронной звезды не превышает 2.7 солнечных (по модулю), дальше идут черные дыры. Затем вычислите два числа: P_x — среднее арифметическое абсцисс найденных черных дыр, и P_y – среднее арифметическое ординат найденных черных дыр.

В ответе запишите четыре числа через пробел: сначала целую часть произведения $P_x \cdot 100$ для файла А, затем $P_y \cdot 100$ для файла А, далее целую часть произведения $P_x \cdot 100$ для файла Б и $P_y \cdot 100$ 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 (файл Б):

```
from math import dist # Функция для вычисления расстояния Евклида
# Функция для кластеризации
def dbscan(a, r):
    c = [] # Создаем массив для кластеров
    while a: # Пока в списке 'a' есть звезды
        c.append([a.pop(0)]) # Создаем кластер и добавляем первую звезду
        for i in c[-1]: # Для каждой звезды в последнем кластере
            for j in a[:]: # Для каждой звезды в списке 'a'
                if dist(i[:2], j[:2]) < r: # Если рядом есть звезда
                    c[-1].append(j) # Добавляем звезду в кластер
                    a.remove(j) # Удаляем ее из исходного списка
    return c # Возвращаем список кластеров
```

```
f = open('27-3.txt') # Открываем файл
s = f.readline() # Считываем первую строку с названием столбцов
# Записываем координаты звезд и их массы в список 'a'
a = [list(map(float, i.replace(',', '.').split())) for i in f]
r = 0.6 # Радиус кластеризации
t = 0.03 # Радиус двойной системы
clusters = dbscan(a, r) # Кластеризуем звезды
px = py = 0 # Искомые средн. арифметические
for i in clusters:
    mx =0 # Минимальная сумма расстояний
    if len(i) > 5:  # Если в кластере больше 5 звезд
        # Кластеризуем внутри кластера с радиусом t
        stars = dbscan(i, t)
        for j in stars:  # Для каждой двойной системы внутри кластера
            if len(j) == 2:  # Если в кластере два объекта (пара звезд)
                x, y = j[0][2], j[1][2]  # Получаем массы звезд
                # Если обе звезды нейтронные (-2.7 <= масса < 0)
                if (-2.7 <= x < 0) and (-2.7 <= y < 0):
                    # Если разница масс больше максимальной
                    if abs(x-y) > mx:
                        mx = abs(x-y)  # Обновляем максимальную разницу масс
                        syst = j  # Пару с максимальной разницей масс
        px += syst[0][0] + syst[1][0]  # Сумма абсцисс
        py += syst[0][1] + syst[1][1]  # Сумма ординат

px = int((px / 6) * 100)  # Среднее арифметическое абсцисс
py = int((py / 6) * 100)  # Среднее арифметическое ординат
print(px, py)
```

Ответ: −172 −474