



Lab 04: Gom nhóm

BÁO CÁO BÀI TẬP MÔN HỌC
KHAI THÁC DỮ LIỆU VÀ ỨNG DỤNG

Người thực hiện

Vũ Hồng Anh – 1412019
Trần Thiên Hoàng – 1412185
Võ Phương Hòa - 1412192

Giáo viên phụ trách

Bùi Thị Danh
Lê Hoài Bắc
Nguyễn Hoàng Khai
Nguyễn Ngọc Thảo



Mục lục

I. Báo cáo thực nghiệm	1
1. Kết quả gom cụm theo thuật toán k-means	1
1.1. Kết quả thực nghiệm tham số k và random seed.....	1
1.2. Kết quả thực nghiệm độ đo khoảng cách	2
2. Phân tích hiệu quả phân tách cụm của các cặp thuộc tính khác nhau	2
2.1. Phân tích hiệu quả phân tách cụm	2
2.2. Ý nghĩa của cụm dữ liệu	4
3. Kết quả gom cụm theo thuật toán DBScan.....	4
3.1. Kết quả thực nghiệm.....	4
3.2. Nhận xét kết quả	5
II. Hướng dẫn sử dụng	6
1. Thuật toán k-means.....	6
1.1. Các hàm chính	6
1.2. Cách sử dụng	7
2. Chương trình vẽ scatter plot	8
2.1. Các hàm chính	8
2.2. Cách sử dụng	8
3. Thuật toán DBScan	10
3.1. Các hàm chính	10
3.2. Cách sử dụng	12
III. Nội dung phân công	13

I. Báo cáo thực nghiệm

Các kết quả thực nghiệm dưới đây được thực hiện trên tập dữ liệu “animal-dental.csv”.

1. Kết quả gom cụm theo thuật toán k-means

1.1. Kết quả thực nghiệm tham số k và random seed

K	random seed					best
	10	20	30	40	50	
2	327.57	327.57	485.54	313.13	313.13	313.13
3	305.03	206.5	261.78	296.73	206.5	206.5
4	192.9	171.96	245.7	188.89	188.89	171.96
5	169.31	161.05	137.75	155.36	171.82	137.75
6	124.83	114.75	106.87	101.42	144.93	101.42

Bảng 1. Kết quả thực nghiệm tham số k và random seed (dùng độ đo khoảng cách Euclidean)

Theo như kết quả thực nghiệm trong bảng các seed tốt nhất đối với các giá trị k:

k2: 40 và 50

k3: 20 và 30

k4: 20

k5: 30

k6: 40

Do đó, số lượng cụm tốt nhất $K_1 = 6$.

1.2. Kết quả thực nghiệm độ đo khoảng cách

K	best random seed	best
2	30, 40, 50	313.13
3	20, 50	209.27
4	20	171.68
5	30	139.27
6	40	111.57

Bảng 3. Kết quả thực nghiệm với độ đo khoảng cách Manhattan

k	best random seed	best
2	40	313.13
3	20, 30, 50	206.50
4	30	154.82
5	30	137.69
6	40	101.88

Bảng 2. Kết quả thực nghiệm với độ đo khoảng cách Cosine

Số lượng cụm tốt nhất $K_2 = 6$. Tương tự số lượng cụm tốt nhất $K_3 = 6$.

Qua các kết quả thực nghiệm trên ta thấy các giá trị K_1, K_2, K_3 có giá trị bằng nhau và bằng 6.

2. Phân tích hiệu quả phân tách cụm của các cặp thuộc tính khác nhau

2.1. Phân tích hiệu quả phân tách cụm

Sau khi phát sinh đồ thị phân bố dữ liệu cho các cặp thuộc tính X và Y như sau:

1. X = răng tiền hàm trên, Y = răng tiền hàm dưới $(X, Y) = (5, 6)$
2. X = răng hàm trên, Y = răng hàm dưới $(X, Y) = (7, 8)$
3. X = răng cửa dưới, Y = răng nanh dưới $(X, Y) = (2, 4)$
4. X = răng cửa trên, Y = răng nanh trên $(X, Y) = (1, 3)$

cho tập dữ liệu “animal-dental.csv” với số lượng gom cụm tốt nhất K_1, K_2, K_3 , kết quả thu được như sau :

Dưới đây là bảng thống kê số điểm mà mỗi điểm trong số đó được biểu diễn bởi nhiều cụm khác nhau.

VD : Số mẫu của tập dữ liệu là 66, tuy nhiên :

- Với $K_1, (X, Y) = (5, 6)$, trên biểu đồ chỉ biểu diễn 10 điểm khác nhau, trong đó có 5 điểm mà mỗi điểm trong số đó được biểu diễn bởi nhiều ký hiệu

khác nhau, đại diện cho các cụm khác nhau, điểm có tọa độ (3, 3) được biểu diễn bởi 4 cụm khác nhau.

- Với K_1 , $(X, Y) = (7, 8)$, trên biểu đồ chỉ biểu diễn 9 điểm khác nhau, trong đó có 2 điểm mà mỗi điểm trong số đó được biểu diễn bởi nhiều cụm khác nhau, điểm có tọa độ (3, 3) được biểu diễn bởi 4 cụm khác nhau.

STT	(X, Y)	Số điểm khác nhau	K_1	K_2	K_3
1	(5, 6)	10	5(4)	5(4)	5(4)
2	(7, 8)	9	2(4)	1(4)	2(4)
3	(2, 4)	9	1(2)	2(3)	2(2)
4	(1, 3)	8	4(3)	4(3)	5(3)

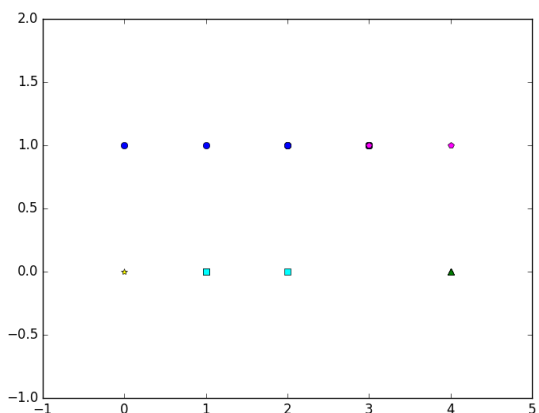
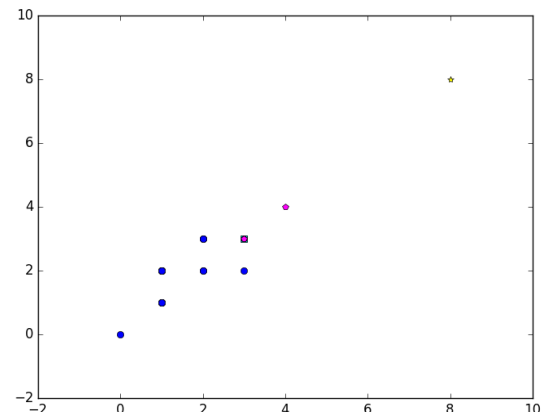
a(b) : trong đó a là số điểm được biểu diễn bởi nhiều cụm khác nhau (tính từ 2 cụm trở lên)

b là số cụm nhiều nhất cùng biểu diễn một điểm

Từ kết quả thống kê, dễ thấy với K_1 và K_3 , cặp thuộc tính $(X, Y) = (2, 4)$ tức (răng cửa dưới, răng nanh dưới) có sự chồng lấn trong biểu diễn cụm là ít nhất. Do đó, cặp thuộc tính (răng cửa dưới, răng nanh dưới) cho sự phân tách cụm tốt nhất.

Với K_2 , tại $(X, Y) = (7, 8)$, chỉ có một điểm được biểu diễn bởi nhiều cụm, nhưng tới 4 cụm, tại $(X, Y) = (2, 4)$, có 2 điểm được biểu diễn bởi nhiều cụm, số cụm cùng biểu diễn một điểm nhiều nhất là 3 cụm. Sau quan sát đồ thị, dễ nhận thấy $(X, Y) = (2, 4)$ phân tách cụm tốt hơn, do đó cặp thuộc tính (răng cửa dưới, răng nanh dưới) cho sự phân tách cụm tốt nhất.

Như vậy, kết quả phân tách cụm tốt nhất ở cả 3 trường hợp đều giống nhau, với cặp thuộc tính cho sự phân tách cụm tốt nhất là (răng cửa dưới, răng nanh dưới).



Đồ thị gom cụm với K_2 , (X, Y) lần lượt là (7, 8) và (2, 4)

2.2. Ý nghĩa của cụm dữ liệu

Chọn kết quả gom cụm với số cụm tốt nhất là K_1 (6 cụm), ta thấy :

- Nhóm 1 : đa phần là thú ăn thịt, sống trên cạn, trừ các động vật thuộc họ rái cá, hải cẩu (5 loài) nên xếp vào nhóm 4
- Nhóm 2 : hầu hết là thú ăn cỏ, trừ sói đồng cỏ (coyote) nên xếp vào nhóm 1
- Nhóm 3 : hầu hết là loài gặm nhấm, trừ hai loài thuộc họ thỏ nên xếp vào nhóm 2
- Nhóm 4 : gồm hai con vật, thuộc họ hải cẩu
- Nhóm 5 : hầu hết thuộc họ dơi, một số khác gồm chồn, chuột chũi, lợn lòi nên xếp vào nhóm 1.
- Nhóm 6 : chỉ một con vật và nó thuộc lớp thú có mai

Như vậy, đa phần các loài thuộc cùng họ, chi (tức gần nhau về mặt sinh học) được xếp vào cùng một nhóm. Các sai sót trong gom cụm chủ yếu gặp ở loài rái cá, hải cẩu (5 loài bị xếp sai vào nhóm 1), còn lại 1 loài thuộc họ chồn, 1 thuộc họ chuột chũi, 1 thuộc họ lợn lòi, 1 thuộc họ sói, 2 loài thuộc họ thỏ cũng bị xếp sai nhóm.

Nhìn lại cấu tạo số răng các loại của loài rái cá, hải cẩu, dễ nhận thấy số răng mỗi loại của chúng tương đương với các loài thú ăn thịt (chồn, báo, linh miêu...) và khác hẳn với các loài hải cẩu được xếp đúng vào nhóm 4.

Tương tự, hai loài thỏ bị xếp sai vào nhóm toàn họ gặm nhấm vì số răng mỗi loại của chúng tương tự với các loài gặm nhấm hơn so với các loài thú ăn cỏ khác. Mặt khác, bản thân thỏ ăn thức ăn xanh (các loại rau củ, lá...) chứ ít khi ăn cỏ nên cấu tạo răng của nó khác với đa phần thú ăn cỏ.

Tóm lại, kết quả phân cụm nhìn chung phản ánh đúng mặt tương đồng về sinh lý của các loài vật, trừ một số sai biệt do sự khác biệt về mặt cấu tạo của một số loài vật.

3. Kết quả gom cụm theo thuật toán DBScan

3.1. Kết quả thực nghiệm

- Với $\epsilon = 1$ và $\text{minPoints} = 8$ ta thu được kết quả thực nghiệm là 3 cụm và 1 giá trị nhiễu (Noise). 34 phần tử nhóm 0, 22 phần tử ở nhóm 1, 9 phần tử ở nhóm 2 và 1 phần tử bị nhiễu.

- Với $\epsilon = 0.5$ và $\text{minPoints} = 8$ ta thu được kết quả thực nghiệm là 3 cụm và 20 giá trị nhiễu. 20 phần tử ở nhóm 0, 17 phần tử ở nhóm 1, 9 phần tử ở nhóm 2 và 20 phần tử bị nhiễu.
- Với $\epsilon = 0.5$ và $\text{minPoints} = 3$ ta thu được kết quả thực nghiệm là 7 cụm và 3 giá trị nhiễu. 20 phần tử ở nhóm 0, 5 phần tử ở nhóm 1, 6 phần tử ở nhóm 2, 3 phần tử ở nhóm 3, 3 phần tử ở nhóm 4, 17 phần tử ở nhóm 5, 9 phần tử ở nhóm 6 và 3 phần tử bị nhiễu.
- Với $\epsilon = 1$ và $\text{minPoints} = 3$ ta thu được kết quả thực nghiệm là 3 cụm và 1 giá trị nhiễu (Noise). 34 phần tử ở nhóm 0, 22 phần tử ở nhóm 1, 9 phần tử ở nhóm 2 và 1 phần tử bị nhiễu.
- Với $\epsilon = 0.07$ và $\text{minPoints} = 5$ ta thu được kết quả thực nghiệm là 5 cụm và 9 giá trị nhiễu (Noise). 20 phần tử ở nhóm 0, 5 phần tử ở nhóm 1, 6 phần tử ở nhóm 2, 17 phần tử ở nhóm 3, 9 phần tử ở nhóm 4 và 9 phần tử bị nhiễu.

3.2. Nhận xét kết quả

Xét về giá trị nhiễu: dựa vào kết quả từ 1 và 4 (có cùng ϵ nhưng khác minPoints) so với kết quả từ 1 và 2 (khác ϵ nhưng cùng minPoints) ta rút ra được giá trị nhiễu phụ thuộc rất nhiều vào ϵ .

Xét về số lượng nhóm: dựa vào kết quả từ 2 và 3 (cùng ϵ nhưng khác minPoints) ta thấy rằng có sự chênh lệch nhóm khá lớn (3 và 7). Mặt khác khi xét kết quả từ 3 và 4 (khác ϵ nhưng cùng minPoints) ta cũng thấy có sự chênh lệch nhóm khá lớn.

Xét về phân bố trong các nhóm: dựa vào kết quả từ 1 và 2 (khác ϵ nhưng cùng minPoints) ta thấy rằng có sự phân hóa không rõ rệt giữa các phần tử trong nhóm. Tuy nhiên khi xét kết quả từ 1 và 4 (khác ϵ và minPoints) ta lại thấy rằng số lượng phần tử giữa các nhóm không thay đổi.

Từ các kết quả gom cụm trên ta nhận thấy rằng sau khi chạy thuật toán DBSCAN thì kết quả gom cụm phụ thuộc rất nhiều vào cả 2 tham số ϵ và minPoints . Tuy nhiên với tham số ϵ và minPoints thích hợp thì sẽ giảm thiểu được các phần tử gây nhiễu cũng như tối ưu hóa được kết quả gom cụm hơn. Mặt khác dựa vào ý nghĩa của tham số ϵ (khoảng cách tối đa giữa 2 điểm) và minPoints (số điểm có thể tới được dựa vào minPoints) ta rút ra được 1 vấn đề về

khuyết điểm của thuật toán DBSCAN là không hiệu quả khi phân nhóm 1 tập hợp dữ liệu có mật độ phân tán khác nhau.

Nhận xét về tính hợp lý về mặt sinh học: ta chọn kết quả 5 (20 phần tử ở nhóm 0, 5 phần tử ở nhóm 1, 6 phần tử ở nhóm 2, 17 phần tử ở nhóm 3, 9 phần tử ở nhóm 4 và 9 phần tử bị nhiễu).

- Ở nhóm 0 có 20 phần tử. Tất cả đều có chung đặc điểm là có 1 răng nanh trên và 1 răng nanh dưới. Có từ 3-4 răng tiền hàm trên và tất cả đều có 3 răng cửa trên và 3 răng cửa dưới. Phần lớn đều là loài ăn thịt và ăn tạp (báo, chồn, cáo, sói, gấu, ...)
- Ở nhóm 1 có 5 phần tử. Điểm chung là có 3 răng cửa trên và 2 răng cửa dưới. Phần lớn các loài này đều có số răng khá giống nhau. Phần lớn là những loài sống hoặc săn bắt dưới nước (trừ chuột chũi Common-mole).
- Ở nhóm 2 có 6 phần tử. Điểm chung là có 2 răng cửa trên, 3 răng cửa dưới, có 1 răng nanh trên, 1 răng nanh dưới, 3 răng hàm trên và 3 răng hàm dưới. Phần lớn thuộc họ dơi (trừ lợn Pecari)
- Ở nhóm 3 có 17 phần tử. Tất cả đều có chung đặc điểm là có 1 răng nanh trên và 1 răng nanh dưới, 3 răng hàm trên và 3 răng hàm dưới. Đặc biệt không có răng nanh. Phần lớn là thuộc họ gặm nhấm (sóc, chuột, guinea pig, sóc chuột, ...).
- Ở nhóm 4 có 9 phần tử. Tất cả đều có chung đặc điểm là chỉ có 4 răng cửa dưới mà không có răng cửa trên, phần lớn không có răng nanh. Các loại răng còn lại đều là 3. Tất cả đều là loài ăn cỏ, có guốc và sừng (hươu, nai, bò, dê, ...)
- Có 9 phần tử bị nhiễu nên ta không phân tích các phần tử này.

II. Hướng dẫn sử dụng

Các chương trình đều thực thi dưới dạng tham số dòng lệnh.

1. Thuật toán k-means

1.1. Các hàm chính

- Cụm hai hàm tìm trung tâm cho các cụm
 - Hàm **cent_gen**: phát sinh các trung tâm khác nhau cho các cụm một cách ngẫu nhiên

- Hàm **find_cent**: tìm trung tâm mới cho các cụm
- Cụm các hàm tính các khoảng cách Euclidean, Manhattan, Cosine
 - Hàm **Euclid_dist**: tính khoảng cách euclidean
 - Hàm **Man_dist**: tính khoảng cách Manhattan
 - Hàm **Cos_sim**: tìm độ tương tự Cosine
- Hàm **k_mean**: thực thi thuật toán k-means

1.2. Cách sử dụng

Cú pháp gọi thực thi chương trình:

python mykmeans.py <input> <output> <k> <random seed> <dist>

trong đó:

<input>: tập dữ liệu, định dạng *.csv

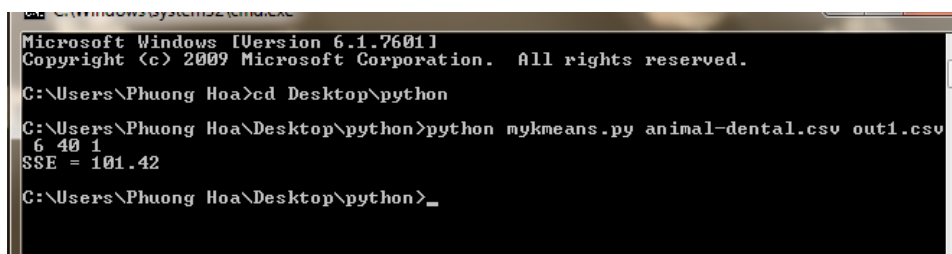
<output>: tập dữ liệu sau phân lớp, định dạng *.csv

<k>: số phân lớp

<random seed>: số nguyên dùng để khởi động bộ gieo số ngẫu nhiên

<dist>: độ đo sự tương tự (1: Euclidean, 2: Manhattan, 3: Cosine)

VD: với tập dữ liệu “animal-dental.csv”, đặt output là “out1.csv”, chọn $k = 6$, random seed = 40, dist = 1 (Euclidean), ta thực hiện lệnh như sau:



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Phuong Hoa>cd Desktop\python

C:\Users\Phuong Hoa\Desktop\python>python mykmeans.py animal-dental.csv out1.csv
6 40 1
SSE = 101.42

C:\Users\Phuong Hoa\Desktop\python>_
```

Kết quả thu được là $SSE = 101.42$ và tập tin “out1.csv” với dữ liệu đã được phân lớp.

2. Chương trình vẽ scatter plot

2.1. Các hàm chính

- Hàm **format_data**: định dạng lại kiểu dữ liệu cho các giá trị trong tập dữ liệu
Input: tập dữ liệu (các giá trị đều thuộc kiểu string)
Output: tập dữ liệu (các giá trị có kiểu dữ liệu thực sự là số thì chuyển sang kiểu float)
- Cụm các hàm xác định kích thước đồ thị:

Mục đích: Để biểu diễn điểm dữ liệu trên đồ thị, ta cần xác định các giá trị xmin, xmax, ymin, ymax của hai trục x, y của đồ thị (việc xác định dựa vào tập dữ liệu). Khi vẽ, một số điểm có thể nằm trên các trục x, y của đồ thị, gây khó khăn cho việc đánh giá sau này cũng như mất thẩm mỹ, do đó cần tăng các giá trị xmax, ymax và giảm các giá trị xmin, ymin để khắc phục nhược điểm đã nêu.

- Hàm **find_delta_max**: tìm giá trị phù hợp để tăng/giảm các giá trị xmax, ymax, xmin, ymin
 - Hàm **update_value_xy**: cập nhật xmax, xmin, ymax, ymin sau khi tăng/giảm giá trị delta tìm được từ hàm find_delta_max
 - Hàm **find_xy**: gọi thực hiện hai hàm trên, kết quả trả về là biến kiểu list gồm 4 phần tử [xmax, xmin, ymax, ymin].
- Hàm **draw_scatter_plot**: vẽ đồ thị gom cụm
Input:
x_list, y_list: có thể coi là tọa độ các điểm cần biểu diễn trên đồ thị
cluster_list: phân lớp của các điểm
li_value: chứa 4 giá trị xmax, xmin, ymax, ymin
picture_name: tên của ảnh lưu đồ thị

Đồ thị có thể biểu diễn tối đa 10 phân lớp với màu sắc và ký hiệu khác nhau ứng với mỗi phân lớp.

2.2. Cách sử dụng

Mở cmd, nhập đường dẫn tới file chứa chương trình thực thi, cú pháp lệnh như sau:

python myplots.py <input> <output> <x> <y>

trong đó:

<input>: tập tin có dạng *.csv, chứa dữ liệu đã phân lớp

<output>: tập tin ảnh có dạng *.png (hoặc *.jpeg), mô tả đồ thị phân lớp

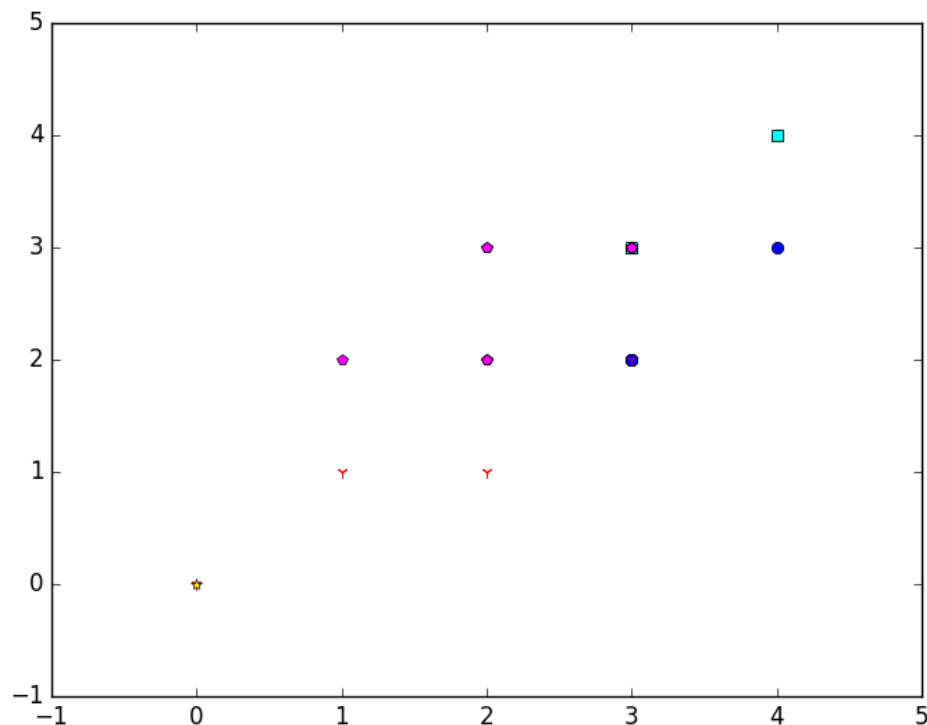
<x>, <y>: chỉ mục của thuộc tính trục x và trục y (tính từ 0 đến n-1, với n là số lượng thuộc tính của dữ liệu)

VD: với tập input và output lần lượt là “out1.csv” và “new.png”, x = 5, y = 6, ta gọi lệnh thực thi như sau:

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Phuong Hoa>cd Desktop\python
C:\Users\Phuong Hoa\Desktop\python>python myplots.py out1.csv new.png 5 6
C:\Users\Phuong Hoa\Desktop\python>_
```

“out1.csv” là kết quả gom cụm tốt nhất của tập dữ liệu “animal-dental.csv” dùng độ đo khoảng cách Euclidean. x và y lần lượt là các thuộc tính thể hiện răng tiền hàm trên và răng tiền hàm dưới. Kết quả thu được là ảnh “new.png” như sau:



3. Thuật toán DBScan

3.1. Các hàm chính

- Hàm **setPoints**: dùng để gán giá trị các thuộc tính cluster vào data

```
# gán giá trị của thuộc tính Cluster
def setPoints(self, dataset):
    m = copy.deepcopy(dataset)
    for item in self.pList:
        for i in range(0, len(dataset)):
            a = map(str, item)
            b = map(str, dataset[i])
            if (a == b):
                m[i].append(int(self.name))
    return m
```

- Hàm **regionQuery**: xét giá trị epsilon có thỏa hay không

```
# Xét epsilon
def regionQuery(self, P, eps):
    result = []
    for d in self.dataSet:
        if (((d[0]-P[0])**2 + (d[1] - P[1])**2)**0.5) <= eps:
            result.append(d)
    return result
```

- Hàm **dbscan**: đây là hàm thực thi thuật toán DBScan

```
#khởi chạy thuật toán DBScan
def dbscan(self,D,eps,MinPts):
    self.dataSet = D[:]
    self.outputDataSet = D[:]
    C = -1
    Noise = cluster('Noise')
    maxitem = 0
    for point in D:
        if point not in self.visited:
            self.visited.append(point)
            NeighbourPoints = self.regionQuery(point,eps)

            if len(NeighbourPoints) < MinPts:
                Noise.addPoint(point)
            else:
                name = str(self.count)
                C = cluster(name)
                self.count+=1;
                self.expandCluster(point,NeighbourPoints,C,eps,MinPts)
    # gán giá trị Noise cho các bộ bi nhiễu
    for item in self.outputDataSet:
        if (len(item)>maxitem):
            maxitem = len(item)
    for item in self.outputDataSet:
        if (len(item)<maxitem):
            item.append('Noise')
    return self.outputDataSet
```

- Hàm **expandCluster**: xét các điểm ghé thăm

```
# ghé thăm các điểm
def expandCluster(self,point,NeighbourPoints,C,eps,MinPts):
    C.addPoint(point)
    for p in NeighbourPoints:
        if p not in self.visited:
            self.visited.append(p)
            np = self.regionQuery(p,eps)
            if len(np) >= MinPts:
                for n in np:
                    if n not in NeighbourPoints:
                        NeighbourPoints.append(n)

    for c in self.Clusters:
        if not c.has(p):
            if not C.has(p):
                C.addPoint(p)

    if len(self.Clusters) == 0:
        if not C.has(p):
            C.addPoint(p)

    self.Clusters.append(C)

    self.outputDataSet = C.setPoints(self.outputDataSet)
```

3.2. Cách sử dụng

Lệnh thực thi có dạng:

python myDBScan.py <input> <output> <epsilon> <minPoints>

trong đó:

<input>: tập tin dữ liệu (định dạng *.csv)

<output>: tập tin dữ liệu sau phân lớp (định dạng *.csv)

<epsilon>, <minPoints>: các tham số của thuật toán

VD: Với tập dữ liệu đầu vào là “animal-dental.csv”, tập tin đầu ra là “new.csv”, các tham số epsilon và minPoints tùy chọn lần lượt là 0.5 và 4, ta thực thi như sau:

Mở cửa sổ cmd, nhập đường dẫn tới file chứa chương trình cần thực thi rồi nhập lệnh như hình dưới

```
C:\WINDOWS\system32\cmd.exe

J:\KTDL & UD\script\gomnhom 2>python myDBScan.py animal-dental.csv new.csv 0.5 4
J:\KTDL & UD\script\gomnhom 2>
```

Kết quả trong file “new.csv” có dạng như sau:

Animal										
A	B	C	D	E	F	G	H	I	J	K
1	Animal	i	C	c	P	p	M	m	Cluster	
2	Opossum	5	4	1	1	3	3	4	4	Noise
3	Hairy-tail-	3	3	1	1	4	4	3	3	0
4	Common-	3	2	1	0	3	3	3	3	1
5	Star-nose-	3	3	1	1	4	4	3	3	0
6	Brown-ba	2	3	1	1	3	3	3	3	2
7	Silver-hai	2	3	1	1	2	3	3	3	2
8	Pigmy-bat	2	3	1	1	2	2	3	3	2
9	House-bat	2	3	1	1	1	2	3	3	2
10	Red-bat	1	3	1	1	2	2	3	3	Noise
11	Hoary-bat	1	3	1	1	2	2	3	3	Noise
12	Lump-nos	2	3	1	1	2	3	3	3	2
13	Armadillo	0	0	0	0	0	0	8	8	Noise
14	Pika	2	1	0	0	2	2	3	3	Noise
15	Snowshoe	2	1	0	0	3	2	3	3	Noise
16	Beaver	1	1	0	0	2	1	3	3	3
17	Marmot	1	1	0	0	2	1	3	3	3
18	Groundho	1	1	0	0	2	1	3	3	3

III. Nội dung phân công

STT	Họ và tên	MSSV	Nhiệm vụ	Kết quả
1	Vũ Hồng Anh	1412019	1. Cài đặt thuật toán k-means 2. Báo cáo kết quả thực nghiệm tham số k và random seed	100%
2	Trần Thiên Hoàng	1412185	1. Báo cáo kết quả thực nghiệm độ đo khoảng cách 2. Cài đặt thuật toán DBScan	100%
3	Võ Phương Hòa	1412192	1. Cài đặt chương trình scatter plot để phân tích hiệu quả phân tách cụm của các cặp thuộc tính khác nhau 2. Nhận xét ý nghĩa của cụm dữ liệu	100%