데이터 사이언스

Programming Project3 : DBSCAN Clustering

학생: 최윤석

학번: 2015005169

작성일자: 2022.06.04

교수님: 김상욱 교수님

실행 환경

- Window
- Python 3.10.1
- numpy 1.22.3
- pandas 1.4.2

Summary of program + algorithm

▶ `if __name__ == "main"`

clustering.py를 실행할 때 넣어줘야 하는 매개 변수들을 argparse를 이용해 입력받고

main 함수에 넘겨준다.

▶ `def main(input_filename, n, eps, min_pts)`

클러스터 수, '밀도'로 인정 받을 거리 eps, 거리 안에 몇 개 이상 object가 있어야 cluster로 인정할 것인지 정하는 min_pts 값들을 매개변수로 받는다.

input_filename을 이용해 data frame을 만들고

DBSCAN object를 생성한다.

▶ `class Point`

input file에 저장되어 있는 object_id, x_coordinate, y_coordinate 값을 저장하기 위한 class이다.

▷ `def __init__(self, object_id, x_pos, y_pos):`

object_id, x_pos, y_pos 값을 넘겨받아 객체를 생성하고,

클러스터링을 위한 label값은 None으로 초기화한다.

▷ `def __ne__(self, other):`

`make_neighbor_list` 함수에서 point끼리의 비교를 하는데, 이 때 !=연산을 하기 위한 함수

▷ `def calculate_distance(self, point):`

다른 점을 input으로 받으면 점과 점 사이의 거리를 구해 반환한다.

► `class DBSCAN`

아래 알고리즘을 수행할 수 있는 함수들과, 데이터가 포함된 class이다.

DBSCAN 알고리즘

#1 point를 하나 선택한다 (label이 있다면 넘어간다)

#2 Eps와 MinPts값을 이용하여 density reachable한 모든 포인트를 탐색하며 label을 채워나간다

#3 기준 point와의 거리가 Eps이하인 point 수가 MinPts 미만이라면 NOISE로 처리한다 (label 값 -1)

#4 (#3)에 해당하지 않으면 self.cluster_label을 이용하여 (class label은 0부터 시작)

density reachable한 point가 없을 때까지 하나의 cluster로 묶는다

#5 모든 포인트에 대해 위 과정을 수행한다

▷ `def __init__(self, input_data_frame, n_cluster, Eps, MinPts):`

input_data_frame을 순회하며 Point객체를 만들어 self.point_list에 저장하고

나머지 매개변수들도 self 변수에 저장한다.

label을 지정하기 위해 self.cluster_label을 선언하여 0으로 초기화한다.

▷ `def make_neighbor_list(self, core):`

core point를 기준으로 self.point_list를 순회하며 core가 아니고, core와의 거리가 Eps이하인 point들을 찾아 list형태로 반환한다.

▷ `def retrieve_cluster(self, neighbor_list, label):`

density reachable한 모든 Point들을 하나의 cluster로 묶기 위한 함수이다.

넘겨받은 neighbor_list를 순회하며 label이 NOISE라면 넘겨받은 label로 지정해준다

만약 label이 없는 Point(neighbor)가 나온다면 label을 지정해주고,

새로 지정한 point를 P라고 할 때 P가 또다른 core가 될 수 있다면 P의 neighbor들도

neighbor_list에 추가해준다.

▷ `def make_cluster_list(self):`

clustering이 끝난 후 호출했을 때 cluster 결과를 반환하는 함수이다.

self.cluster_label수 만큼 이중 list를 만든 후

NOISE가 아닌 Point들을 label에 따라 분류한다.

그 후 sort와 reverse를 이용해 앞에서부터 cluster내에 point가 많은 순으로

프로그램 실행 시 지정한 cluster수 만큼 list형태로 반환한다.

▷ `def clustering(self):`

clustering을 수행하는 함수이다.

label이 없는 point들에 대해 make_neighbor_list로 neighbor list를 구해

retrieve_cluster를 호출하여 cluster를 형성한다.

실행 결과

```
최 운 석 @DESKTOP-ANR4SCC MINGW64 /d/programming/Github/2022_ite4005_2015005169/ass
ignment3/result (assignment3)
$ PA3.exe input1
98.90902점
최 운 석 @DESKTOP-ANR4SCC MINGW64 /d/programming/Github/2022_ite4005_2015005169/ass
ignment3/result (assignment3)
$ PA3.exe input2
94.60035점
최 운 석 @DESKTOP-ANR4SCC MINGW64 /d/programming/Github/2022_ite4005_2015005169/ass
ignment3/result (assignment3)
$ PA3.exe input3
99.97736점
최 운 석 @DESKTOP-ANR4SCC MINGW64 /d/programming/Github/2022_ite4005_2015005169/ass
ignment3/result (assignment3)
$
```

주의 사항

- 실행 전 numpy와 pandas를 설치해야 합니다.
- input#.txt에 대한 output file은 input#_result 폴더에 생성됩니다.
- clustering.py를 실행하는 폴더에 input#.txt파일이 존재해야 합니다.