PyData.Fukuoka

～ 箸休め的ＬＴ ～

福岡県の交通事故を見える化してみた(地図編)

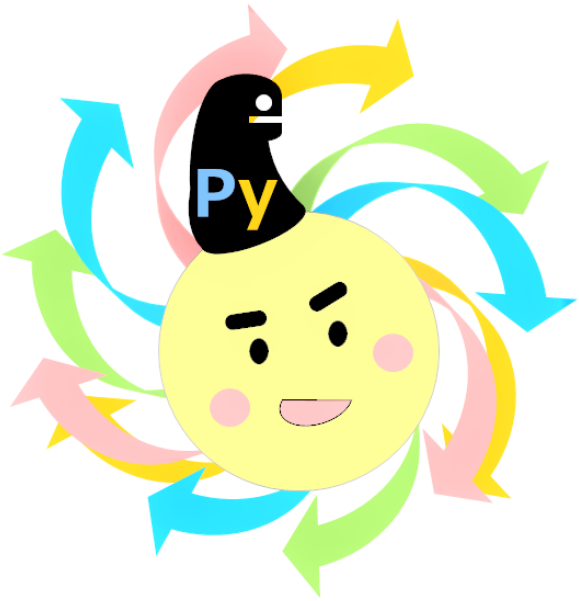
・事故多発エリアは、どこよ？

・最も危険な交差点は、どこよ？

おまえだれよ

- 田中丸 祐治 （たなかまる　ゆうじ）

- Python好きの
  日曜大工的なんちゃって
  データサイエンティスト
  （要は、ただのサラリーマン）



- 備忘録替わりにTwitter はじめました：@malo21st

いまからやること（ざっくり）

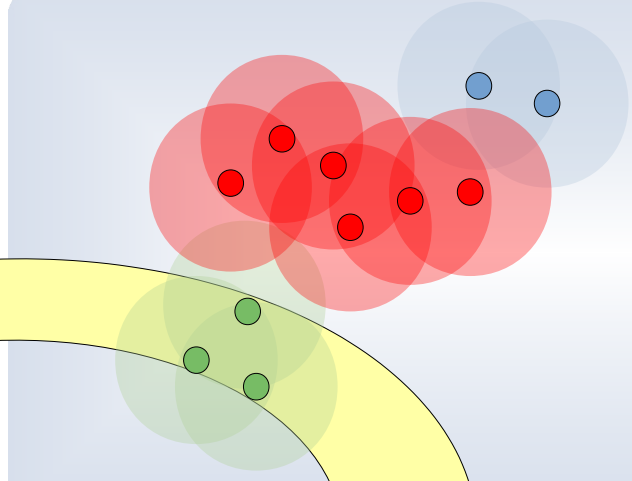
PyData.Fukuoka

SQLite

位置情報
緯度・経度

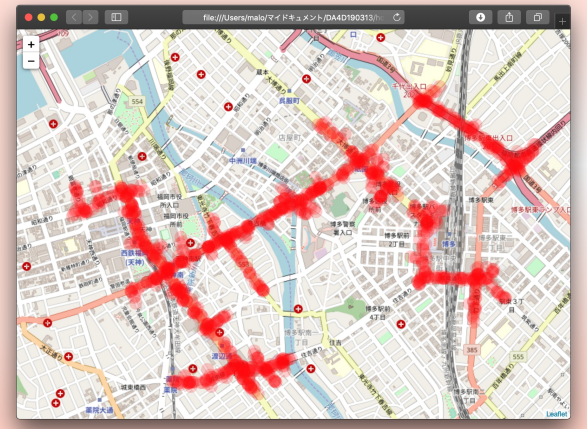
交通事故データ

・項　目　数：　25
・レコード数：72,170



データ処理

NetworkX



見える化

folium

データ処理(1/3)

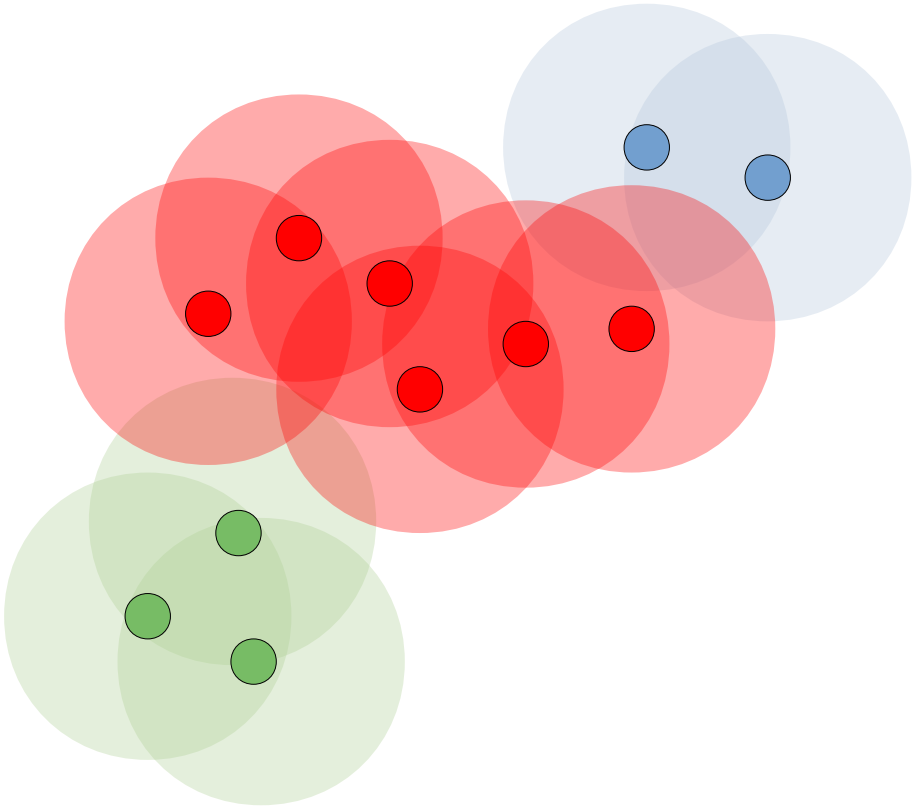
PyData.Fukuoka

・発生場所が近いなら同類とみなして同じグループ

同じグループ の条件

・危険なエリア
　条件：５０ｍ以内

・危険な交差点
　条件：１０ｍ以内



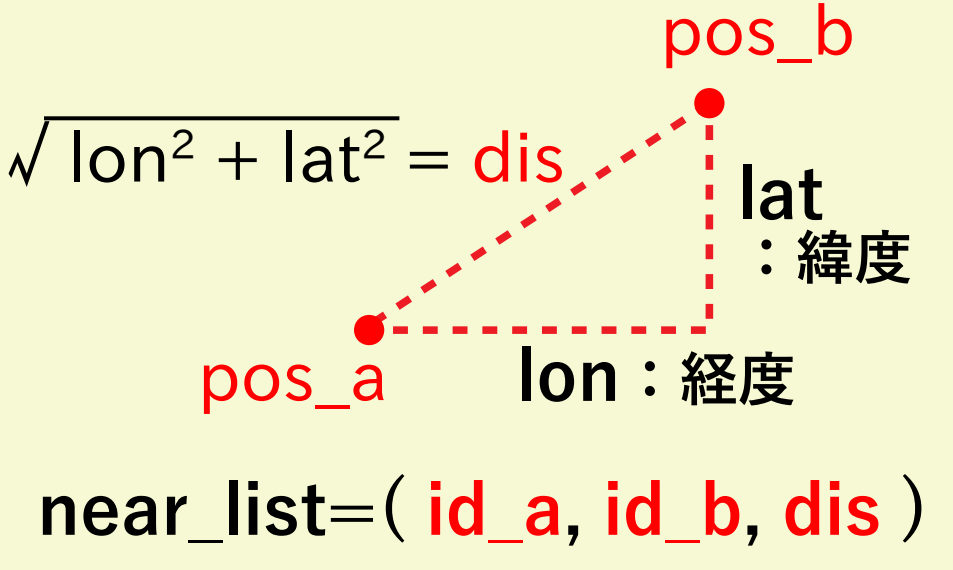
データ処理(2/3)

PyData.Fukuoka

- **発生場所が近い**なら同類とみなして同じグループ

```
1  DIS_DEG = 0.00009 # 0.00009deg:10m , 0.00045deg:50m
2  SQ_DIS_DEG = DIS_DEG * DIS_DEG
3
4  pos_list = list(pos_data.values)
5  near_list = []
6
7  while pos_list:
8      pos_a = pos_list.pop(0)
9      for pos_b in pos_list:
10         lat = abs(pos_a[1] - pos_b[1])
11         lon = abs(pos_a[2] - pos_b[2])
12         if lat > DIS_DEG:
13             continue
14         if lon > DIS_DEG:
15             continue
16         sq_dis = lat * lat + lon * lon
17         if sq_dis < SQ_DIS_DEG:
18             near_list.append([
19                 int(pos_a[0]),
20                 int(pos_b[0]),
21                 math.sqrt(sq_dis) / DIS_DEG * 10
22             ])
```

pos_? = (id_?, lat, lon)
[0] [1] [2]



near_list=(id_a, id_b, dis)

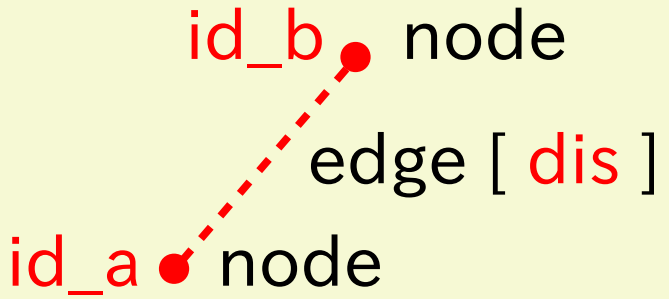
データ処理(3/3)

PyData.Fukuoka

・発生場所が近いなら同類とみなして同じグループ

⇨ NetworkX に丸投げ

near_list = (id_a, id_b, dis)
距離

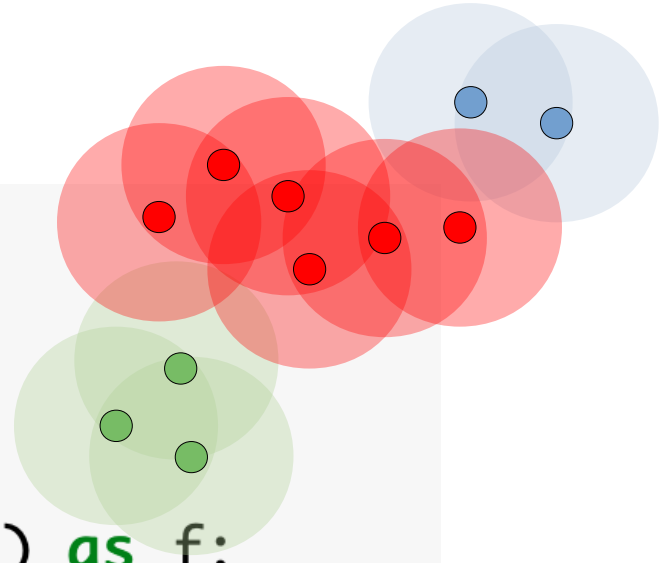



```
1  import networkx as nx
2
3  G = nx.Graph()
4
5  for nanbd in near_list:
6      G.add_edges_from([(nanbd[0], nanbd[1])])
7      G.edges[nanbd[0], nanbd[1]]['dis'] = nanbd[2]
8
9
10 import pickle
11
12 with open('near_graph10m.pkl', mode='wb') as f:
13     pickle.dump(G, f)
```

見える化(1/3)

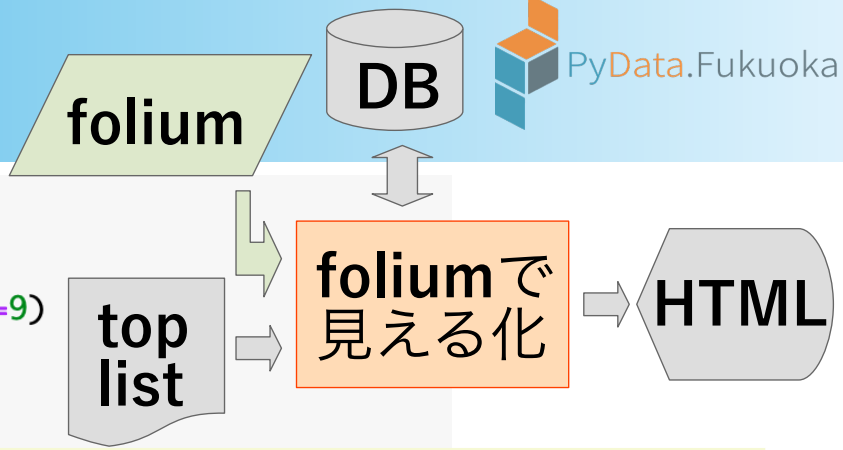
PyData.Fukuoka

・表示させたいデータの抽出



```
1  import networkx as nx
2  import pickle
3
4  G = nx.Graph()
5
6  with open('near_graph10m.pkl', 'rb') as f:
7      G = pickle.load(f)
8
9  top_list = []
10
11 for component in nx.connected_components(G):
12     top_list.append([len(component), component])
13
14 top_list.sort(reverse=True)
```

見える化(2/3)




```python
import folium

Fukuoka_city_hall = [33.58974488, 130.401803]
plot_map = folium.Map(location=Fukuoka_city_hall, zoom_start=9)

def set_marker(pos, msg):
    【 詳細は次のスライドで 】

for top in top_list[:20]:
    pos_list = []
    sum_death = 0
    sum_serious = 0
    sum_injury = 0

    for id_db in top[1]:
        sql = "SELECT latitude,longitude,death,serious,injury FROM master WHERE id ="
                + str(id_db)
        cursor.execute(sql)
        data = cursor.fetchall()
        pos_list.append([data[0][0], data[0][1]])
        sum_death += data[0][2]
        sum_serious += data[0][3]
        sum_injury += data[0][4]

    for pos in pos_list
        set_marker(
            [pos[0],pos[1]],
            [top[0], sum_death, sum_serious, sum_injury]
        )

plot_map.save('plot_map.html')
```

top, top_list = (件数 , { id, … , id })
data = (緯度, 経度, 死者, 重傷者, 軽傷者)

set_marker([緯度, 経度], [件数, 各累計人数])

見える化(3/3)

PyData.Fukuoka

・folium を使ってマーカーを設定

```
def set_marker(pos, msg):
    MONTH = 24
    POP = "発生数：{}件　月平均：{:.1f}件<br>死亡：{}　重傷：{}　軽傷：{}"

    folium.CircleMarker(
        location = [pos[0], pos[1]],
        popup = POP.format(msg[0], msg[0]/MONTH, msg[1], msg[2], msg[3]),
        radius = 10,
        color = None,
        fill_color = 'red'
    ).add_to(plot_map)
```

PyData.Fukuoka

デモンストレーション

PyData.Fukuoka

ご清聴ありがとうございました

本日の資料：

https://github.com/malo21st/PyDataFUK190416