

# Bài 14 - Biểu đồ trên Google Map

22 Oct 2019 - phamdinhkhanh

## Menu

- 1. Các biểu đồ dạng bản đồ
- 2. Biểu đồ mark
  - 2.1. Khởi tạo google map
  - 2.2. Đánh dấu điểm trên google map
- 2.3. Nối các điểm trên google map bằng polyline
  - 2.4. Tính toán khoảng cách theo đường chim bay
  - 2.5. Bản đồ heatmap
  - 2.6. Bản đồ heatmap theo thời gian
- 3. Tổng kết
- 4. Tài liệu.

## 1. Các biểu đồ dạng bản đồ

### Bản đồ google map có thể thực hiện được những gì?

Ngày nay cùng với sự phát triển của công cụ google map, việc visualize các báo cáo có liên quan đến vị trí địa lý trở nên dễ dàng hơn. Google map cũng support đa dạng các biểu đồ khác nhau như biểu đồ heatmap, biểu đồ scatter, biểu đồ line và rất nhiều các tùy biến khác trên một bản đồ có khả năng phóng to hoặc thu nhỏ theo view. Người dùng thông qua các điểm trên google map có thể đo lường được vị trí khoảng cách địa lý, phân phối mạng lưới chi nhánh theo khu vực. Các ứng dụng lớn của google map có thể kể đến đó là:

- Vẽ các biểu đồ cảnh báo thiên tai: Mức độ động đất, lũ quét, hạn hán,....
- Vẽ các biểu đồ liên quan đến dân số, sản lượng nông nghiệp, công nghiệp.
- Trong bất động sản có thể ứng dụng google map để người mua dễ hình dung vị trí của nhà ở so với các trục đường chính và hệ thống cơ sở hạ tầng xung quanh khu vực.
- Visualize mạng lưới kinh doanh để quản lý và phát triển mạng lưới.
- Đánh dấu vị trí cửa hàng, doanh nghiệp trên google map để tăng lượng khách hàng tiềm năng. ...

Và rất nhiều các ứng dụng khác của google map mà ta có thể thực hiện được. Để vẽ một biểu đồ của google map rất đơn giản vì dữ liệu đầu vào cần có tọa độ địa lý (longitude, latitude) và giá trị gắn liền với tọa độ địa lý đó.

Bài viết này sẽ hướng dẫn các bạn cách để vẽ các biểu đồ thông dụng trên google map sao cho đẹp mắt, hiệu quả và thể hiện được mục tiêu cần phân tích.

Các tài liệu mà tôi tham khảo để vẽ biểu đồ sẽ được liệt kê ở phần cuối cùng.

Bây giờ chúng ta sẽ cùng bắt tay vào việc thực hiện các biểu đồ trên google map nào.

### Có thể sử dụng những package nào để visualize bản đồ trên python?

Trên python có rất nhiều các package visualization các bản đồ của google map. Một trong số đó là: folium, gmplot, gmaps . Các package gmplot, gmaps đều sử dụng các API của google map. Gần đây việc sử dụng những thông tin này yêu cầu bạn phải có tài khoản trả phí của

google. Trái lại folium sử dụng thông tin bản đồ từ OpenStreetMap là một data mở được chia sẻ tới người dùng. Bên dưới là thông tin giới thiệu về những packages này:

- folium: Là một công cụ khai phá dữ liệu mạnh của python kế thừa các tính năng mạnh mẽ của thư viện rất nổi tiếng về map là Leaflet.js trong javascript.
- gmplot: Là một interface của matplotlib để tạo thành HTML và javascript cho phép render toàn bộ dữ liệu bạn muốn vào một bản đồ trên google map một cách dễ dàng.
- gmaps: Cũng là một Jupyter plugin cho phép nhúng google map vào một jupyter notebook. Các tính năng chính tương tự như gmplot.

Trong bài viết này chúng ta sẽ sử dụng thư viện folium là chủ yếu bởi: Biểu đồ không yêu cầu phải sử dụng API key của google map và các graphic cũng đa dạng hơn rất nhiều so với gmplot và gmaps.

Lý thuyết cũng chỉ là lý thuyết. Vậy hãy cùng xem chúng ta có thể tạo được các bản đồ đẹp mắt như thế nào trên google map nào.

## 2. Biểu đồ mark

### Ví dụ về đánh dấu chi nhánh

Giả sử công ty bạn có một mạng lưới các chi nhánh phân bố từ bắc vào nam. Bạn không biết làm cách nào để thể hiện khoảng cách giữa các chi nhánh đó. Trước khi có google map công việc thật khó khăn nhưng hiện tại bạn có thể:

- Đánh dấu vị trí địa lý các chi nhánh trên bản đồ.
- Thể hiện qui mô doanh số giữa các chi nhánh.
- Phân biệt được các chi nhánh theo vùng quản lý dựa trên màu sắc thay đổi.

Tôi sẽ lấy ví dụ một công ty (dữ liệu fake) có khoảng 200 chi nhánh được phân chia thành 7 vùng với mã số từ R01 đến R07 và 2 miền MB và MN như bên dưới. Chúng ta sẽ vẽ biểu đồ phân bố chi nhánh theo vùng và thể hiện doanh số của các chi nhánh thông qua 3 packages trên. Dữ liệu chúng ta có thể download tại DemandBranchFake.xlsx (<https://drive.google.com/file/d/1tbWWs58EedFhI5DBdnW9p8uB8oSjZSv/view?usp=sharing>).

### Khảo sát dữ liệu

```

1 import pandas as pd
2 dataset = pd.read_excel("DemandBranchFake.xlsx", sheet_name="DemandBranch")
3 names = ['DELIVERY_WARD_LATITUDE', 'DELIVERY_WARD_LONGITUDE',
4           'BRANCH_CODE', 'AREA', 'REGION', 'DI
5 encoding="utf8", sep=",", header=0)
6 print(dataset.shape)
7 dataset.head()

```



	DELIVERY_WARD_LATITUDE	DELIVERY_WARD_LONGITUDE	PROVINCE	BRANCH_CODE	AREA	REGION	DEMAND
0	9.946490	106.328108	Trà Vinh	TXT	MN	R07	399
1	10.250815	106.002399	Vĩnh Long	NMW	MN	R07	1585
2	10.678836	106.601911	Hồ Chí Minh	DRV	MN	R06	599
3	10.712544	106.695103	Hồ Chí Minh	MTM	MN	R06	77602
4	10.722063	106.525835	Hồ Chí Minh	KIY	MN	R06	Top 2290

## 2.1. Khởi tạo google map

Việc đầu tiên khi tạo một bản đồ trên google map là chúng ta phải khởi tạo một base map. Một base map sẽ yêu cầu truyền vào giá trị latitude-longtitude (vĩ độ-kinh độ) để xác định tọa độ vị trí tâm bản đồ và mức độ zoom tại thời điểm ban đầu. Từ các giá trị này ta sẽ xác định được bản đồ khởi tạo sẽ hiển thị bắt đầu trên khu vực nào với mức độ phóng đại là bao nhiêu. Chẳng hạn nếu tôi muốn đồ thị của mình hiển thị một khu vực xung quanh Hà Nội với độ zoom không quá lớn tôi có thể thiết lập latitude-longtitude vào khoảng (20.9, 105.8) và mức độ zoom là 12. Trường hợp tôi muốn nhìn toàn bộ các tỉnh ở miền bắc có thể giảm độ zoom xuống 7-8. Điều đó có nghĩa rằng zoom càng to thì càng chi tiết và zoom nhỏ sẽ ít chi tiết nhưng bao phủ một khu vực rộng lớn hơn.

Bên dưới ta sẽ khởi tạo một base map trên package gmplot thông qua hàm GoogleMapPlotter() .

```

1 import gmplot
2
3 lat = 20.97
4 long = 105.8
5 zoom = 12
6
7 # Khởi tạo một googleMapPlotter
8 gmap1 = gmplot.GoogleMapPlotter(lat, long, zoom)
9 # Lưu google map vào một html file
10 gmap1.draw('initialGmplot.html')
```

Hiện tại google đã thu phí đối với các API của google map service nên nếu không sử dụng google map key bạn sẽ chỉ có thể xem ở chế độ development mode. Bạn có thể tạo một key của google map api (<https://developers.google.com/maps/documentation/embed/get-api-key>), sau khi đã tạo được key thì cần phải enable dịch vụ Map API javascript (<https://developers.google.com/maps/documentation/javascript/tutorial>) để sử dụng được chế độ tương tác trên javascript. Thủ tục này khá mất công. Sau khi đã gen key, chúng ta cần add key vào bản đồ như bên dưới:

```

1 import gmplot
2
3 lat = 20.97
4 long = 105.8
5 zoom = 12
6
7 # Khởi tạo một googleMapPlotter
8 gmap1 = gmplot.GoogleMapPlotter(lat, long, zoom)
9 # Thêm key cho API bằng cách thay thế key vào `your_key`
10 gmap1.apikey = "your_key"
11 # Lưu google map vào một html file
12 gmap1.draw('initialGmplot.html')
```

Theo cách trên bạn sẽ mất phí vì phải sử dụng các google services. Một giải pháp khác đó là package folium. Biểu đồ của folium không đòi hỏi API key, được kế thừa từ thư viện leaflet.js bên javascript nên trông đẹp mắt và đa dạng hơn.

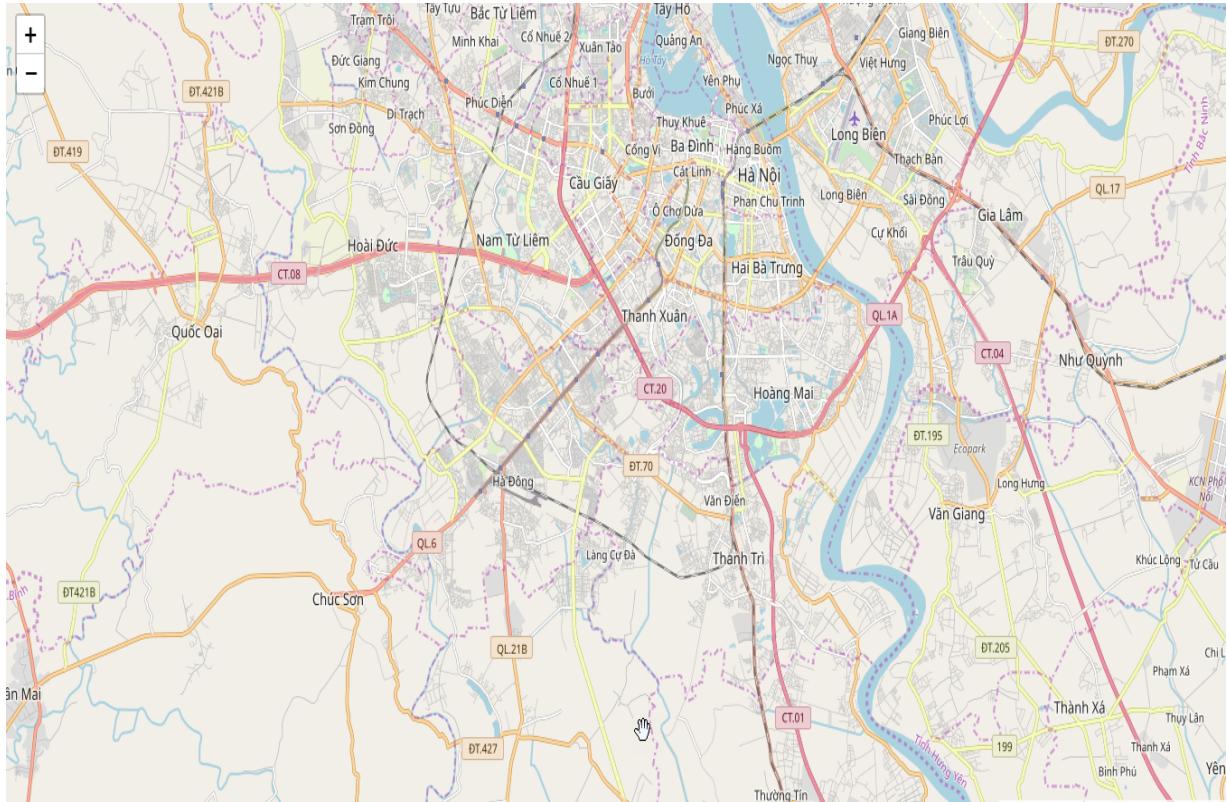
Top

```

1 import folium
2
3 lat = 20.97
4 long = 105.8
5 zoom = 12
6
7 gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
8 gmap2.save('initialFolium.html')

```

Sau khi mở các file trên các bạn sẽ thu được một bản đồ có dạng:



## 2.2. Đánh dấu điểm trên google map

google map cho phép ta đánh dấu được những vị trí quan trọng trên bản đồ. Đây là một hình thức thu thập và làm giàu dữ liệu từ người dùng rất thông minh của google. Thi thoảng bạn sẽ nhận được những câu hỏi khi đánh dấu một địa điểm trên google map như:

- Hãy cho tôi biết địa điểm bạn đang đứng là gì? Mục đích thu thập và làm giàu thông tin.
- Có phải vị trí bạn đang ở gần địa điểm A không? Mục đích là xác thực thông tin người dùng khác đã nhập.
- Hãy tham gia vào hệ thống người đánh dấu địa điểm của google để thu được các tiện ích gì đó (không nhớ rõ). Mục đích là để tạo ra những người dùng trung thành là chuyên gia và có đam mê với đánh dấu địa lý trên google map.

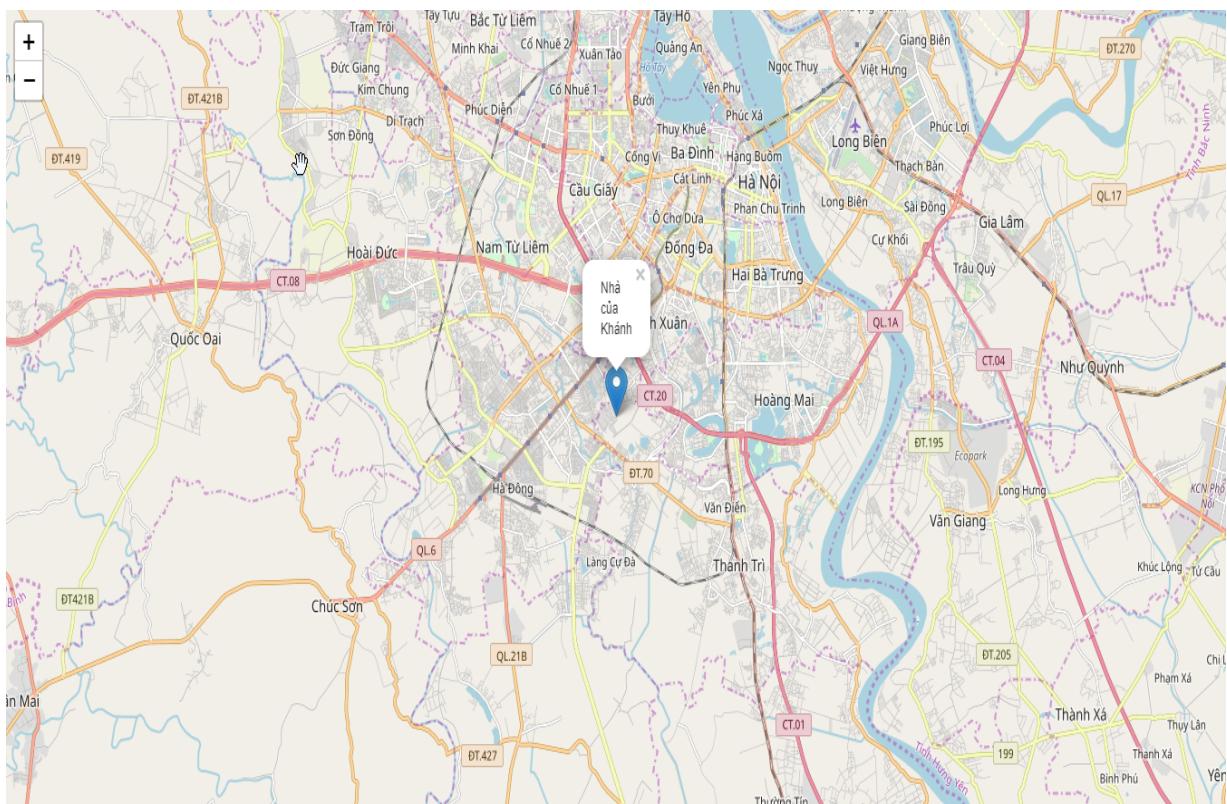
Nói tóm lại google có rất nhiều cách thức khác nhau khuyến nghị người dùng bổ sung và làm giàu dữ liệu cho bản đồ của họ. Bản thân tôi cũng là một người thường xuyên đánh dấu lại các địa điểm trên google, nhưng nơi mà tôi đã đến du lịch để lưu lại thời gian, địa điểm như một cuốn nhật ký địa lý. Việc đánh dấu các điểm quan trọng trên bản đồ cũng có thể được sử dụng dễ dàng thông qua object Marker trong folium .

Top

```

1 import folium
2
3 lat = 20.97
4 long = 105.8
5 zoom = 12
6 name = "Nhà của Khánh"
7 color="red"
8
9 # Khởi tạo base map
10 gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
11 # Thêm một marker vào base map
12 folium.Marker(location=(lat, long), popup=name).add_to(gmap2)
13 gmap2.save("foliumMarker.html")

```



Quay trở lại ví dụ về đánh dấu các chi nhánh. Chúng ta có gần 200 chi nhánh trong bộ dữ liệu. Liệu có thể đánh dấu toàn bộ những vị trí của các chi nhánh này? Câu trả lời là hoàn toàn có thể thông qua một vòng lặp đánh dấu từng điểm dữ liệu.

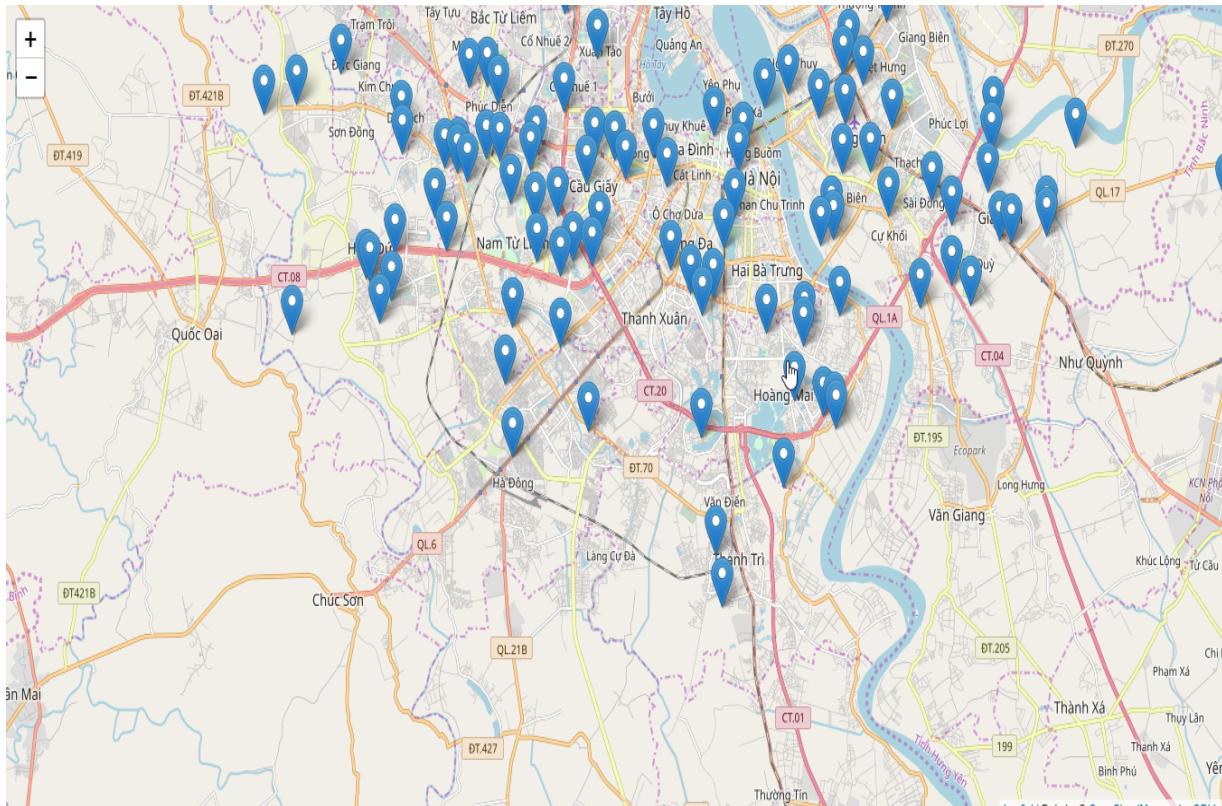
```

1 import folium
2
3 lat = 20.97
4 long = 105.8
5 zoom = 12
6 gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
7
8 def _addMarker(gmap):
9     for i, row in dataset.iterrows():
10         marker = folium.Marker(location=(row['DELIVERY_WARD_LATITUDE',
11                                         'DELIVERY_WARD_LONGITUDE']))
12         marker.add_to(gmap)
13
14 _addMarker(gmap2)
15 gmap2.save("foliumMarkerMultiple.html")

```

Top

Khi đó các điểm trên bản đồ sẽ hiển thị như sau:



## 2.3. Nối các điểm trên google map bằng polyline

### Ví dụ về giao vận

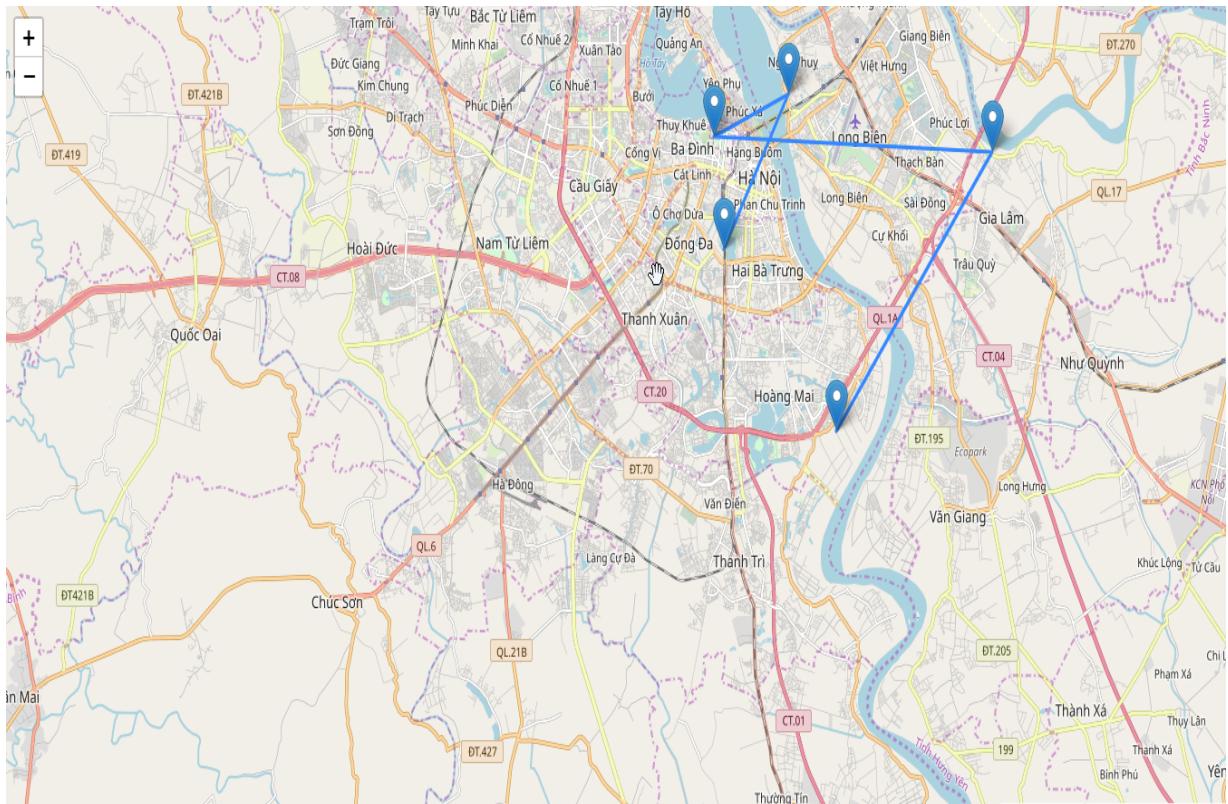
Một tình huống khác phát sinh đó là bạn là một đơn vị giao vận (logistic) và cần chuyển hàng từ điểm đến tới điểm đi. Làm sao để biết được giữa các điểm thể hiện trên bản đồ thì đâu là 2 điểm thuộc cùng một chuyến giao hàng (gồm điểm bắt đầu là kho và điểm kết thúc là địa chỉ nhận hàng). Trong folium biểu đồ PolyLine sẽ cho phép chúng ta đưa ra một lộ trình bằng cách nối các điểm trong list locations dựa trên kinh độ và vĩ độ của nó. Thực hiện như bên dưới:

```

1 import folium
2
3 lat = 20.97
4 long = 105.8
5 zoom = 12
6 gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
7
8 def _addPolyLine(gmap, dataset):
9     for i, row in dataset.iterrows():
10         marker = folium.Marker(location=(row['DELIVERY_WARD_LATITUDE'],
11                               row['DELIVERY_WARD_LONGITUDE']))
12         marker.add_to(gmap)
13         polyline = folium.PolyLine(locations=list(zip(dataset['DELIVERY_WARD_LATITUDE'],
14                                         dataset['DELIVERY_WARD_LONGITUDE'])),
15                                     line_opacity=0.5).add_to(gmap)
16
17 dataHN = dataset[dataset['PROVINCE'] == 'Hà Nội'].sample(5)
18 _addPolyLine(gmap2, dataHN)
19 gmap2.save("foliumPolyLine.html")

```

Top



## 2.4. Tính toán khoảng cách theo đường chim bay

Dựa trên cặp (latitude, longitude) truyền vào ta có thể tính toán được khoảng cách theo đường chim bay giữa các điểm với nhau. Khoảng cách được xây dựng dựa trên công thức tính khoảng cách giữa 2 điểm trên một mặt cầu. Bạn đọc quan tâm có thể xem thêm tại Great circle distance - wikipedia ([https://en.wikipedia.org/wiki/Great-circle\\_distance](https://en.wikipedia.org/wiki/Great-circle_distance)). Tôi xin phép không đi sâu lan man lý thuyết phần này mà đi vào ứng dụng luôn. Hàm tính khoảng cách khi đã biết tọa độ (latitude, longitude) của 2 điểm ( $X, Y$ ) như bên dưới.

```

1   from math import cos, asin, sqrt
2   import numpy as np
3
4   def distance(lat1, lon1, lat2, lon2):
5       R = 6371 # Bán kính trái đất (km)
6       p = np.pi/180 # giá trị pi/180
7       a = 0.5 - cos((lat2 - lat1) * p)/2 + cos(lat1 * p) * cos(lat2 * p)
8       return 2*R*asin(sqrt(a))

```

Chẳng hạn bên dưới ta muốn tính khoảng cách từ tỉnh Vĩnh Long đến thành phố Hồ Chí Minh. Ta sẽ truyền vào tọa độ tương ứng với 2 vị trí này như bên dưới:

Top

```

1  # Tọa độ Vĩnh Long
2  lat1 = 10.250815
3  lon1 = 106.002399
4  # Tọa độ thành phố Hồ Chí Minh
5  lat2 = 10.678836
6  lon2 = 106.601911
7  # Tính khoảng cách
8
9  distance(lat1, lon1, lat2, lon2)

1  81.00896435483854

```

## 2.5. Bản đồ heatmap

Bản đồ heatmap hay còn gọi là bản đồ nhiệt là một trong những bản đồ thể hiện độ lớn thông qua cường độ màu sắc, rất hiệu quả trong việc đo lường giá trị cường độ theo vùng địa lý. Heatmap thường được sử dụng trong các biểu đồ cảnh báo thiên tai hoặc biểu đồ về phân bố lượng mưa, lượng nhiệt.

Trong kinh doanh heatmap cũng rất hiệu quả trong việc thể hiện phân bố doanh số theo vùng. Giả sử dựa trên nhu cầu tiêu thụ của các vùng miền chúng ta muốn vẽ một bản đồ heatmap để hiển thị những vùng có nhu cầu tiêu thụ lớn và tập trung và những vùng có nhu cầu tiêu thụ ít. Chúng ta thực hiện như sau:

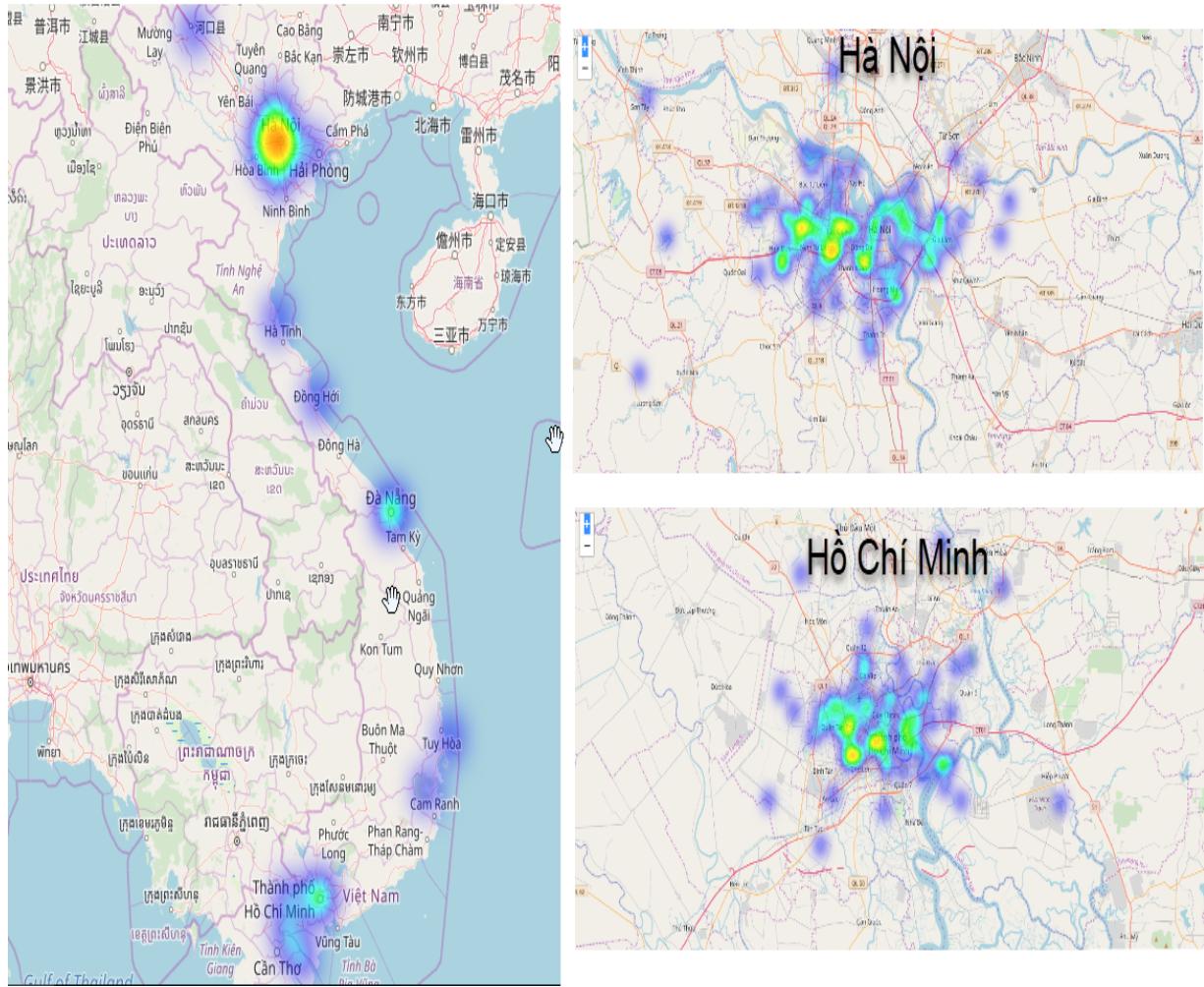
```

1  from folium.plugins import HeatMap
2
3  lat = 20.97
4  long = 105.8
5  zoom = 12
6  gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
7  maximum = max(dataset['DEMAND'])
8
9  def _addHeatMap(dataset, gmap):
10     data_loc_val = list(zip(dataset['DELIVERY_WARD_LATITUDE'], dataset['DELIVERY_WARD_LONGITUDE']))
11     heatMap = HeatMap(data=data_loc_val,
12                       min_opacity = 0.6,
13                       blur = 20,
14                       max_val = maximum,
15                       radius = 15,
16                       overlay = True)
17     heatMap.add_to(gmap)
18
19 _addHeatMap(dataset, gmap2)
20 gmap2.save("foliumHeatMap.html")

```



Top



Nhìn vào bản đồ ta có thể thấy Hà Nội và Hồ Chí Minh là hai thành phố có nhu cầu tiêu thụ lớn nhất khi màu sắc được thể hiện đậm hơn trên heatmap. Ngoài ra, khi zoom chi tiết vào các khu vực này ta còn có thể thấy được ở từng phường, quận mức độ tiêu thụ như thế nào?

### Làm thế nào để đánh dấu điểm quan trọng trên heatmap?

Khi theo dõi heatmap chúng ta có thể cần đánh dấu những vị trí quan trọng trên bản đồ để tìm vị trí đặt chi nhánh. Khi đó chỉ cần thêm một layer vào bản đồ gốc sao cho layer này cho phép khởi tạo các marker. Hàm `ClickForMarker()` trong folium cho phép ta tạo một layer như vậy.

Top

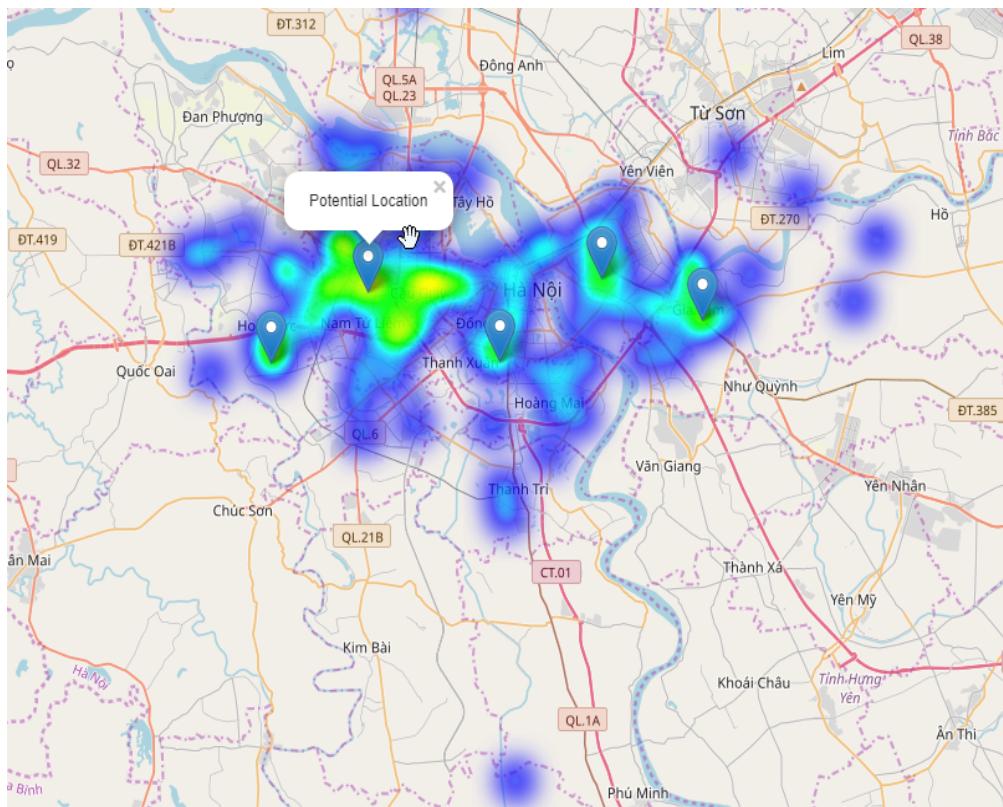
```

1   from folium.plugins import HeatMap
2
3   lat = 20.97
4   long = 105.8
5   zoom = 12
6   gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
7   maximum = max(dataset['DEMAND'])
8
9   def _addHeatMap(dataset, gmap):
10      data_loc_val = list(zip(dataset['DELIVERY_WARD_LATITUDE'], dataset['DELIVERY_WARD_LONGITUDE']))
11      heatMap = HeatMap(data=data_loc_val,
12                         min_opacity = 0.6,
13                         blur = 20,
14                         max_val = maximum,
15                         radius = 15,
16                         overlay = True)
17      heatMap.add_to(gmap)
18
19  _addHeatMap(dataset, gmap2)
20  gmap2.add_child(folium.ClickForMarker(popup='Chi nhánh tiệm nắng'))
21
22  gmap2.save("foliumHeatMapClickForMarker.html")

```



Kết quả sau khi right click vào những vị trí được đánh dấu trên bản đồ:



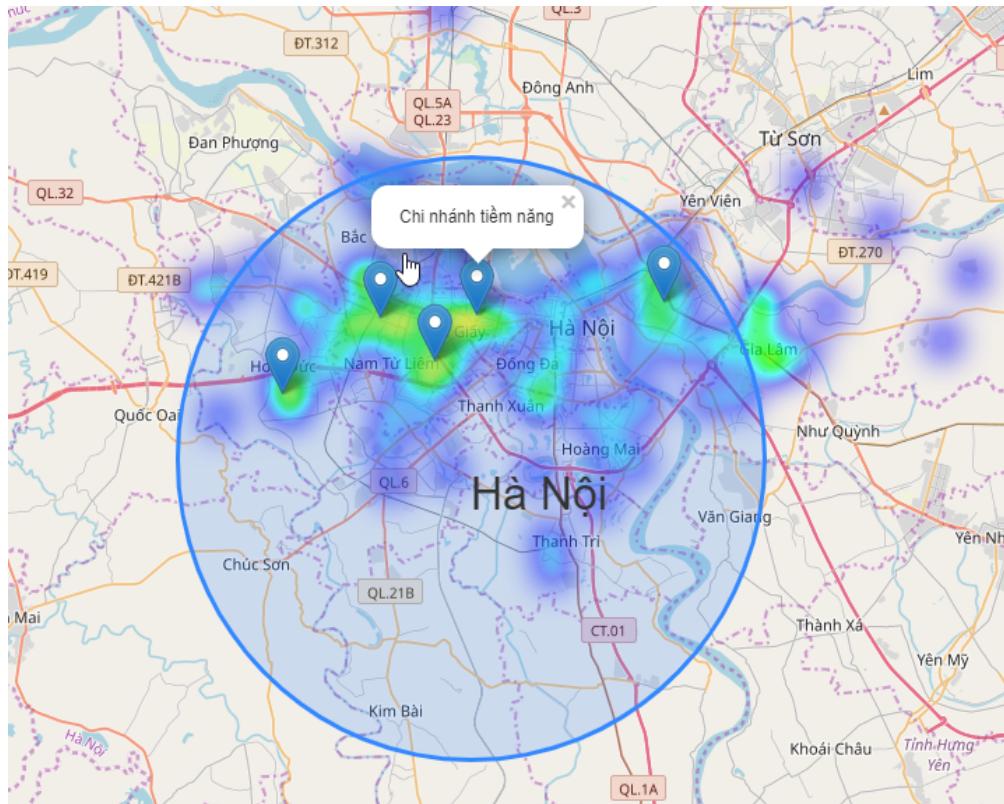
Ta có thể khoanh thêm vùng cho các vị trí lõi dựa vào layer `Circle()`.

Top

```
1  from folium.plugins import HeatMap
2  from folium.features import DivIcon
3
4  lat = 20.97
5  long = 105.8
6  zoom = 12
7  text = "Hà Nội"
8  gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
9  maximum = max(dataset['DEMAND'])
10
11 def _addHeatMap(dataset, gmap):
12     data_loc_val = list(zip(dataset['DELIVERY_WARD_LATITUDE'], dataset['DELIVERY_WARD_LONGITUDE']))
13     heatMap = HeatMap(data=data_loc_val,
14                         min_opacity = 0.6,
15                         blur = 20,
16                         max_val = maximum,
17                         radius = 15,
18                         overlay = True)
19     heatMap.add_to(gmap)
20
21 _addHeatMap(dataset, gmap2)
22 gmap2.add_child(folium.ClickForMarker(popup='Chi nhánh tiệm nắng'))
23 # Thêm circle có bán kính 15000
24 gmap2.add_child(folium.Circle([lat, long], radius = 15000, fill = True))
25 # Thêm text chú thích cho circle
26 gmap2.add_child(folium.Marker(
27     [lat, long],
28     icon=DivIcon(
29         icon_size=(150,36),
30         icon_anchor=(0,0),
31         html=<div style="font-size: 24pt">%s</div>' % text,
32         )
33     ))
34
35 gmap2.save("foliumHeatMapClickForMarker.html")
```

Kết quả:

Top



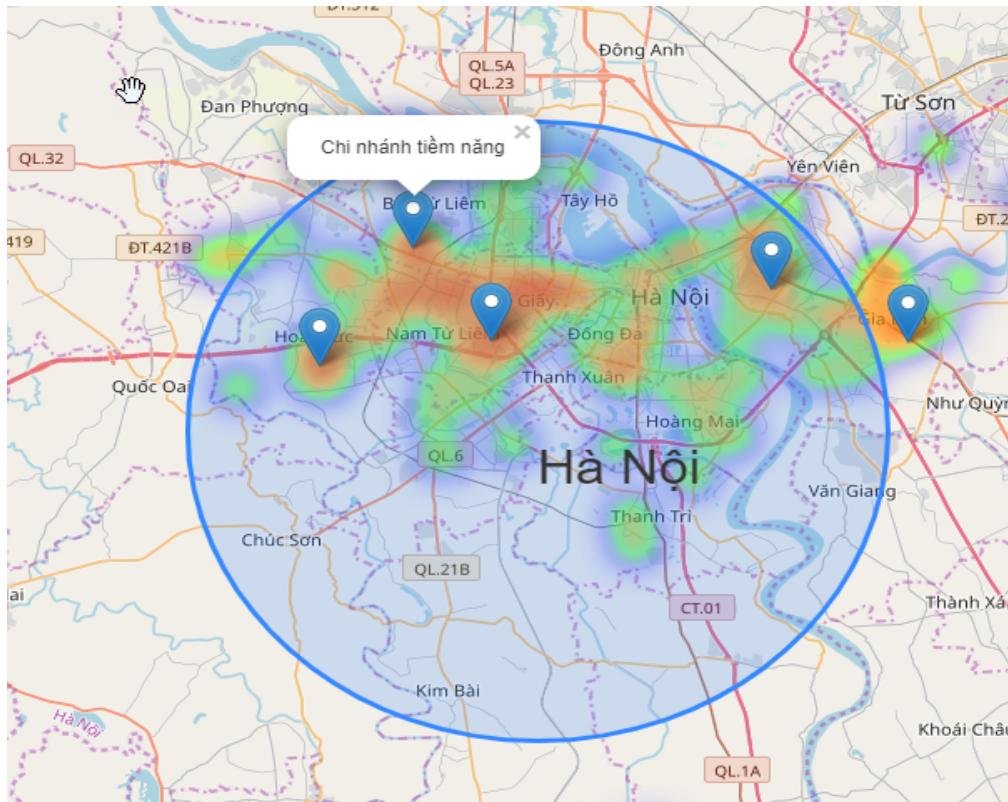
Chúng ta cũng có thể thay đổi màu sắc cho đồ thị Heatmap thông qua việc điều chỉnh gradient của hàm `Heatmap()` bằng cách thay đổi cường độ của các màu sắc bên trong. Chẳng hạn tôi muốn heatmap có một màu đỏ hơn thì cần điều chỉnh trọng số của heatmap cao hơn. Giá trị mà tôi thiết lập cho gradient là: {0.2: 'blue', 0.4: 'lime', 0.6: 'orange', 1: 'red'}

Top

```
1  from folium.plugins import HeatMap
2  from folium.features import DivIcon
3
4  lat = 20.97
5  long = 105.8
6  zoom = 12
7  text = "Hà Nội"
8  gmap2 = folium.Map(location=(lat, long), zoom_start=zoom)
9  maximum = max(dataset['DEMAND'])
10
11 def _addHeatMap(dataset, gmap):
12     data_loc_val = list(zip(dataset['DELIVERY_WARD_LATITUDE'], dataset['DELIVERY_WARD_LONGITUDE']))
13     heatMap = HeatMap(data=data_loc_val,
14                         min_opacity = 0.6,
15                         blur = 20,
16                         # Thay đổi màu sắc bằng gradient
17                         gradient = {0.2: 'blue', 0.4: 'lime', 0.6: 'orange', 1: 'red'},
18                         max_val = maximum,
19                         radius = 15,
20                         overlay = True)
21     heatMap.add_to(gmap)
22
23     _addHeatMap(dataset, gmap2)
24     gmap2.add_child(folium.ClickForMarker(popup='Chi nhánh tiệm nắng'))
25     #Thêm circle có bán kính 15000
26     gmap2.add_child(folium.Circle([lat, long], radius = 15000, fill = True))
27     #Thêm text chú thích cho circle
28     gmap2.add_child(folium.Marker(
29         [lat, long],
30         icon=DivIcon(
31             icon_size=(150, 36),
32             icon_anchor=(0,0),
33             html='<div style="font-size: 24pt">%s</div>' % text,
34         )
35     ))
36
37 gmap2.save("foliumHeatMapClickForMarker.html")
```



Top



## 2.6. Bản đồ heatmap theo thời gian

Khi xem một bản đồ heatmap thường phân phối màu sắc sẽ thay đổi theo thời gian. Cũng giống như các bạn xem bản đồ về diện tích của bắc cực và nam cực theo 4 mùa trong năm, diện tích các cực sẽ mở rộng hoặc thu hẹp theo từng mùa. Để tracking được những biến đổi này chúng ta cần sử dụng tới biểu đồ time heatmap.

Để demo cho cách tạo một biểu đồ time heatmap tôi sẽ lấy ví dụ về dữ liệu độ dài của các chuyến đi taxi trong năm 2016 được lấy từ cuộc thi NYC taxi trip duration (<https://www.kaggle.com/c/nyc-taxi-trip-duration/data>). Dựa trên các thông tin về chuyến đi người dùng cần dự báo xem chuyến đi sẽ kéo dài trong bao lâu. Dữ liệu gồm các trường như sau:

- id: Mã số chuyến đi. Là trường key xác định duy nhất.
- vendor\_id: Là mã số ám chỉ hãng taxi cung cấp chuyến đi.
- pickup\_datetime: Ngày và giờ chuyến đi bắt đầu chuyến bánh.
- dropoff\_datetime: Ngày và giờ chuyến đi kết thúc.
- passenger\_count: Số lượng hành khách trên chuyến đi.
- pickup\_longitude: Kinh độ của vị trí đón khách.
- pickup\_latitude: Vĩ độ của vị trí đón khách.
- dropoff\_longitude: Kinh độ vị trí trả khách.
- dropoff\_latitude: Vĩ độ vị trí trả khách.
- store\_and\_fwd\_flag: Đánh dấu bản ghi chuyến đi đã được lưu lại trên bộ nhớ của phương tiện trước khi gửi tới hãng taxi.
- trip\_duration: Độ dài của chuyến đi.

### Đọc và xử lý dữ liệu:

```

1 import pandas as pd
2 dataTaxiTrip = pd.read_csv('nyc-taxi-trip-duration/test/test.csv', header=0)
3 print(dataTaxiTrip.shape)
4 dataTaxiTrip.info()

```

[Top](#)

```

1      <class 'pandas.core.frame.DataFrame'>
2      RangeIndex: 625134 entries, 0 to 625133
3      Data columns (total 14 columns):
4          id           625134 non-null object
5          vendor_id    625134 non-null int64
6          pickup_datetime 625134 non-null datetime64[ns]
7          passenger_count 625134 non-null int64
8          pickup_longitude 625134 non-null float64
9          pickup_latitude   625134 non-null float64
10         dropoff_longitude 625134 non-null float64
11         dropoff_latitude   625134 non-null float64
12         store_and_fwd_flag 625134 non-null object
13         month          625134 non-null int64
14         week            625134 non-null int64
15         day             625134 non-null int64
16         hour            625134 non-null int64
17         count           625134 non-null int64
18         dtypes: datetime64[ns](1), float64(4), int64(7), object(2)
19         memory usage: 66.8+ MB

```

Để làm việc với thời gian chúng ta cần trích xuất các thành phần thời gian (month, week, day, hour) của thời điểm đón khách.

```

1  def _extract_datetime(df):
2      df.pickup_datetime = pd.to_datetime(df.pickup_datetime, format='%Y-%m-%d %H:%M:%S')
3      df['month'] = df.pickup_datetime.apply(lambda x: x.month)
4      df['week'] = df.pickup_datetime.apply(lambda x: x.week)
5      df['day'] = df.pickup_datetime.apply(lambda x: x.day)
6      df['hour'] = df.pickup_datetime.apply(lambda x: x.hour)
7
8  _extract_datetime(dataTaxiTrip)
9  print(dataTaxiTrip.shape)
10 dataTaxiTrip.info()

```

```

1      <class 'pandas.core.frame.DataFrame'>
2      RangeIndex: 625134 entries, 0 to 625133
3      Data columns (total 14 columns):
4          id           625134 non-null object
5          vendor_id    625134 non-null int64
6          pickup_datetime 625134 non-null datetime64[ns]
7          passenger_count 625134 non-null int64
8          pickup_longitude 625134 non-null float64
9          pickup_latitude   625134 non-null float64
10         dropoff_longitude 625134 non-null float64
11         dropoff_latitude   625134 non-null float64
12         store_and_fwd_flag 625134 non-null object
13         month          625134 non-null int64
14         week            625134 non-null int64
15         day             625134 non-null int64
16         hour            625134 non-null int64
17         count           625134 non-null int64
18         dtypes: datetime64[ns](1), float64(4), int64(7), object(2)
19         memory usage: 66.8+ MB

```

Top

Đầu tiên ta sẽ khởi tạo map tại vị trí trung bình của vĩ độ và kinh độ của các địa điểm đón taxi. Việc này để đảm bảo bản đồ heatmap được bao trọn trong khung hình của đồ thị.

```

1 import folium
2
3 lat = dataTaxiTrip['pickup_latitude'].mean()
4 long = dataTaxiTrip['pickup_longitude'].mean()
5 zoom = 12
6
7 gmapNYTaxi = folium.Map(location=[lat, long], zoom_start = zoom)
8 # gmapNYTaxi.save('foliumNYTaxi.html')

```

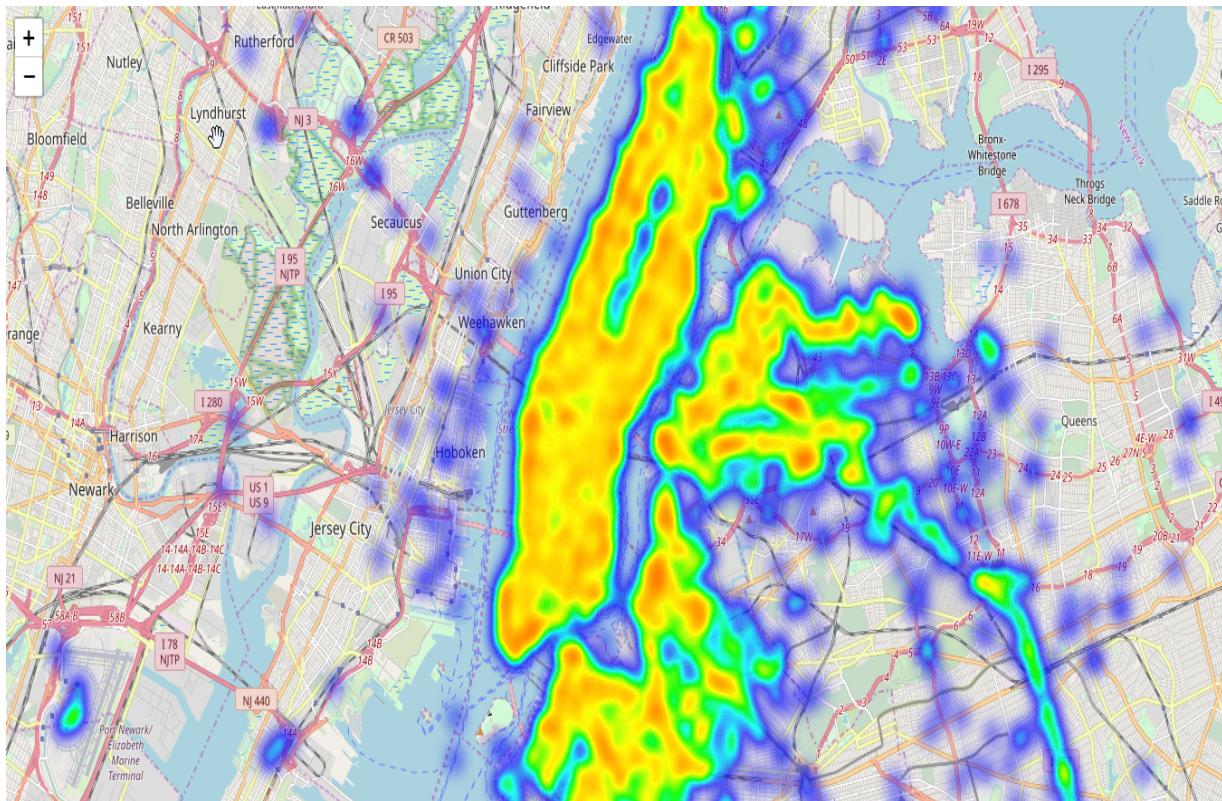
Thêm heatmap layer khởi tạo vào bản đồ.

```

1 from folium.plugins import HeatMap
2
3 dataTaxiTrip['count'] = 1
4 heatmap = HeatMap(data=dataTaxiTrip[['pickup_latitude', 'pickup_longitude']],
5                     .groupby(['pickup_latitude', 'pickup_longitude'])
6                     .sum().reset_index().values.tolist(),
7                     radius=8,
8                     max_zoom=13)
9 gmapNYTaxi.add_child(heatmap)
10 gmapNYTaxi.save('foliumHeatMapNY.html')

```

Kết quả:



Như vậy ta thu được một heatmap biểu diễn số lượng chuyển đi theo địa điểm cho toàn bộ năm 2016.

**Tạo heatmap đếm số lượng chuyến đi theo từng khung giờ cho toàn bộ năm 2016 Top**

Ta nhận thấy rằng nhu cầu về taxi sẽ thay đổi theo thời gian. Ban đêm sẽ ít hơn so với ban ngày. Do đó ta sẽ cần tạo một biểu đồ heatmap theo thời gian mà phân chia được các chuyến đi theo khung giờ trong ngày. Trước tiên ta cần chuẩn bị dữ liệu là list các thống kê số lượng chuyến đi theo từng tọa độ tại mỗi một khung giờ. List sẽ bao gồm 24 phần tử đại diện cho 24 khung giờ khác nhau. Số liệu thống kê là số chuyến taxi xảy ra trong cùng một giờ trong toàn bộ năm 2016.

```

1     dataTaxiTrip_hour_list = []
2
3     for hour in dataTaxiTrip.hour.sort_values().unique():
4         dataTaxiTrip_hour_list.append(dataTaxiTrip.loc[dataTaxiTrip.hour == hour,
5                                         ['pickup_latitude', 'pickup_longitude']].groupby(['pickup_latitude', 'pickup_longitude']).count())

```

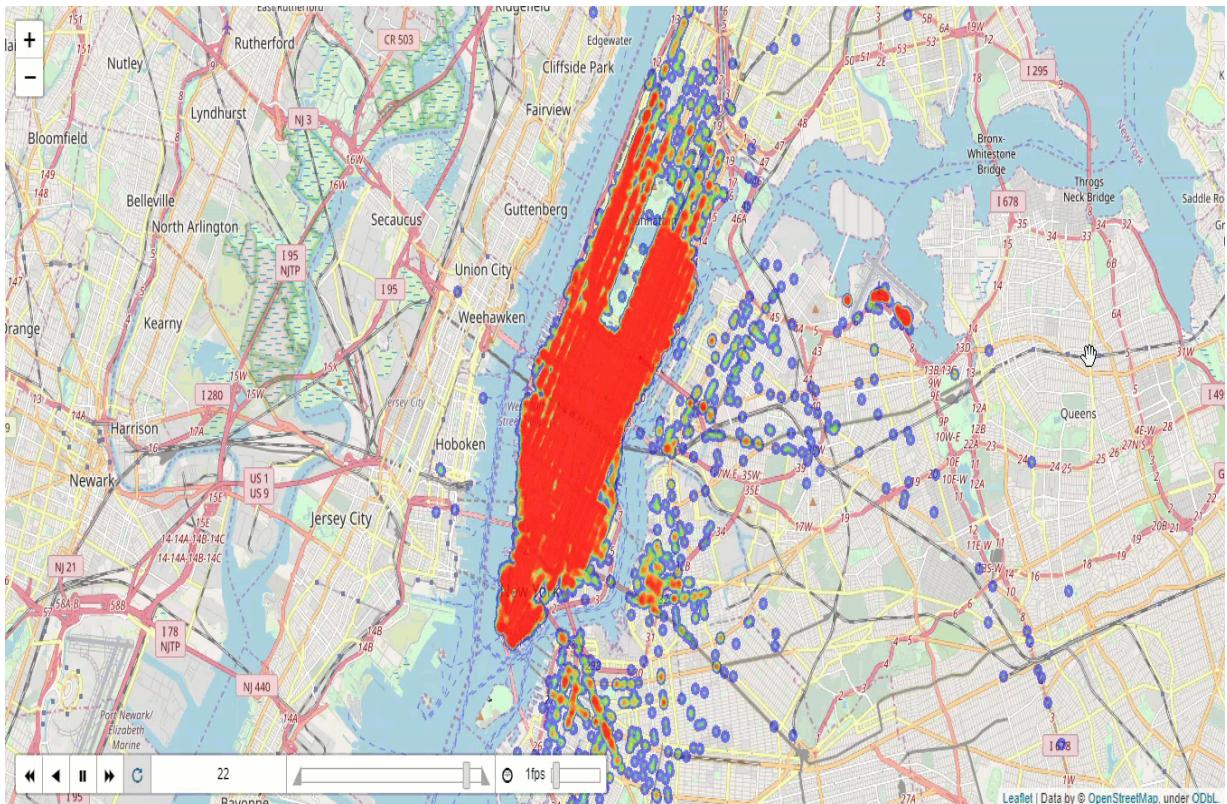
Vẽ biểu đồ HeatMap biến đổi theo thời gian

```

1     from folium.plugins import HeatMapWithTime
2
3
4     lat = dataTaxiTrip['pickup_latitude'].mean()
5     long = dataTaxiTrip['pickup_longitude'].mean()
6     zoom = 12
7
8     gmapNYTaxi = folium.Map(location=[lat, long], zoom_start = zoom)
9
10    heatmap = HeatMapWithTime(dataTaxiTrip_hour_list,
11                                radius=5,
12                                gradient={0.2: 'blue', 0.4: 'lime', 0.6: 'orange', 0.8: 'red'},
13                                min_opacity=0.5,
14                                max_opacity=0.8,
15                                use_local_extrema=True)
16
17    gmapNYTaxi.add_child(heatmap)
18    gmapNYTaxi.save('foliumHeatMapWithTimeNY.html')

```

Kết quả:



### 3. Tổng kết

Như vậy qua bài viết này tôi đã giới thiệu đến các bạn những biểu đồ cơ bản nhất trên google map bao gồm: Marker, CircleMarker, polyLine, Heatmap. Các biểu đồ này có thể được sử dụng kết hợp với nhau một cách linh hoạt để tạo ra những báo cáo ấn tượng. Tôi cũng hi vọng rằng sau khi đọc bài viết này các bạn sẽ có thể áp dụng được vào thực tiễn vào các dự án của mình.

Ngoài ra các dạng biểu đồ google map hiện nay còn được tích hợp sẵn trên power BI, tableau, data studio và nhiều phần mềm visualize khác. Có thời gian tôi sẽ giới thiệu thêm về cách thức tạo báo cáo google map trên những công cụ này.

Cuối cùng không thể thiếu là phần tổng kết những tài liệu mà tôi đã tham khảo.

### 4. Tài liệu.

1. creating interactive crime maps with folium - blog dominodatalab  
(<https://blog.dominodatalab.com/creating-interactive-crime-maps-with-folium/>)
2. folium for maps heatmaps time analysis - kaggle daveanhickey  
(<https://www.kaggle.com/daveanhickey/how-to-folium-for-maps-heatmaps-time-analysis>)
3. plotting google map using gmplot package in python - tutorialspoint  
(<https://www.tutorialspoint.com/plotting-google-map-using-gmplot-package-in-python>)

Top