

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**



BÁO CÁO BÀI TẬP LỚN

DATA ENGINEER

Analyzing user interests by their stops

Thành phố Hồ Chí Minh, Tháng 11 năm 2023



Mục lục

1	Giới thiệu đề tài	3
2	Mô tả tập dữ liệu	3
2.1	Tổng quan bộ dữ liệu	3
2.2	Dịnh dạng tệp Trajectory	5
2.3	Dịnh dạng tệp Transportation	5
3	Các lý thuyết liên quan	5
3.1	StayPoint	5
3.2	Giải thuật DBSCAN	6
4	Các công nghệ sử dụng	7
4.1	Apache Hadoop & HDFS	7
4.1.1	Hadoop	7
4.1.2	HDFS	8
4.2	Apache Spark	9
4.3	MySQL	12
4.4	PowerBI	13
5	Hướng giải quyết bài vấn đề	14
5.1	Xây dựng hệ thống lưu trữ dữ liệu	14
5.2	Thống kê dữ liệu của người dùng	18
5.3	Phân cụm các vị trí GPS của người dùng	19
5.4	Trực quan hóa dữ liệu GPS của 1 người dùng theo thời gian	20
5.4.1	Trực quan hóa thông qua heatmap	21
5.5	Trên toàn bộ tệp dữ liệu - xác định các địa điểm được ghi nhận nhiều nhất	24
5.6	Trên tập dữ liệu của mỗi người dùng - xác định các điểm được ghi nhận nhiều nhất	27
5.7	Trên toàn bộ tập dữ liệu - xác định thời gian nhiều users nhất cùng xuất hiện	31
5.8	Trên toàn bộ tập dữ liệu - xác định vị trí nhiều user đi qua nhau trên bản đồ	35
5.9	Các loại hình giao thông hay sử dụng	38
5.9.1	Thời gian sử dụng phương tiện của mỗi user	38
5.9.2	Nhận xét	40
6	Kết luận	40
6.1	Kết quả	40
6.2	Khó khăn	40
6.2.1	Độ lớn của dữ liệu	40
6.2.2	Lựa chọn công cụ	41
6.2.3	Cách tiếp cận bài toán	41
6.3	Ý nghĩa của đề tài	41



Danh mục hình ảnh

1	Hệ thống định vị toàn cầu GPS	3
2	Tổng quan dữ liệu ở Bắc Kinh	4
3	Tổng quan dữ liệu ở đường vành đai 5, Bắc Kinh	4
4	Phân bố lộ trình theo khoảng cách và thời lượng hoạt động	4
5	Phân bố người dùng theo khoảng thời gian ghi nhận và số lượng lộ trình	5
6	StayPoint	6
7	Giải thuật DBSCAN	6
8	Hadoop	7
9	Hệ sinh thái của Hadoop	8
10	Kiến trúc của HDFS	8
11	Spark	9
12	Thành phần của Spark	10
13	Đặc điểm của Spark	11
14	MySQL	12
15	Kiến trúc của MySQL	12
16	PowerBI	13
17	Giao diện cơ bản của PowerBI	14
18	Luồng chuyển đổi dữ liệu	15
19	Kiểm tra dữ liệu root của HDFS	15
20	Kiểm tra dữ liệu bên trong folder Data	16
21	2 databases: trajectory và label đã được tạo	17
22	Database trajectory	18
23	Database label	18
24	Tổng quan về dữ liệu người dùng	18
25	Biểu đồ thể hiện số ngày tham dự GPS của user	19
26	Biểu đồ thể hiện quãng đường tham dự của mỗi user	19
27	Phân cụm của người dùng 70 trong tháng 7 năm 2007	20
28	Dữ liệu vị trí của người dùng 70 trong tháng 9 năm 2008	21
29	Quảng đường di chuyển của người dùng 70	22
30	Heatmap của người dùng 70 tại Bắc Kinh	22
31	Heatmap của người dùng 70 tại Trường Sơn	23
32	Heatmap của người dùng 70 tại Loan Nam	23
33	Dếm số lần xuất hiện của các điểm	25
34	Thêm cột thể hiện loại địa điểm	25
35	Danh sách các loại địa điểm xuất hiện nhiều nhất	26
36	Một địa điểm có type là "yes"	26
37	Các địa điểm có số lần xuất hiện nhiều nhất	27
38	Các địa điểm có số lần xuất hiện nhiều nhất của mỗi người	28
39	Loại địa điểm ưa thích nhất của mỗi người	29
40	Những loại địa điểm ưa thích nhất của người dùng	29
41	Tỉ lệ người dùng đến địa điểm ưa thích	30
42	Trực quan hóa địa điểm ưa thích của từng người dùng	31
43	Thống kê những ngày có số lượng người dùng được ghi nhận nhiều nhất	33
44	Những người dùng xuất hiện trong ngày 17/02/2009 và địa điểm đầu tiên họ xuất hiện	34
45	Những loại địa điểm xuất hiện đầu tiên của mỗi người dùng trong ngày 17/02/2009	34
46	Hai ngày có nhiều người dùng ghi nhận nhất	35
47	Những địa điểm có số lượng user đi qua nhiều nhất	36
48	Loại địa điểm của những điểm có số lượng user đi qua nhiều nhất	37
49	Những loại địa điểm xuất hiện	37
50	Trực quan hóa các địa điểm ghi nhận nhiều user	38
51	Thời gian sử dụng phương tiện của mỗi user	39
52	Thời gian sử dụng phương tiện của tất cả user	40

1 Giới thiệu đề tài

GPS (Global Positioning System) hay còn gọi là Hệ thống định vị toàn cầu là hệ thống vệ tinh có khả năng cung cấp dữ liệu về tọa độ, cao độ, dấu thời gian thậm chí là cả những thông tin về hướng di chuyển, tốc độ tức thời,... Hiện nay, hầu hết các thiết bị thông minh đều có GPS, chính vì vậy việc thu thập thông tin về vị trí của người dùng cho những mục đích chính đáng là hoàn toàn khả thi. Trong bài báo cáo này, nhóm tập trung nghiên cứu về việc phân tích người dùng thông qua vị trí GPS của họ. Dựa vào tệp data GPS của người có sẵn, nhóm đã xử lý, phân tích và trực quan hóa dữ liệu để có thể rút ra được những thông tin ý nghĩa.



Hình 1: Hệ thống định vị toàn cầu GPS

2 Mô tả tập dữ liệu

2.1 Tổng quan bộ dữ liệu

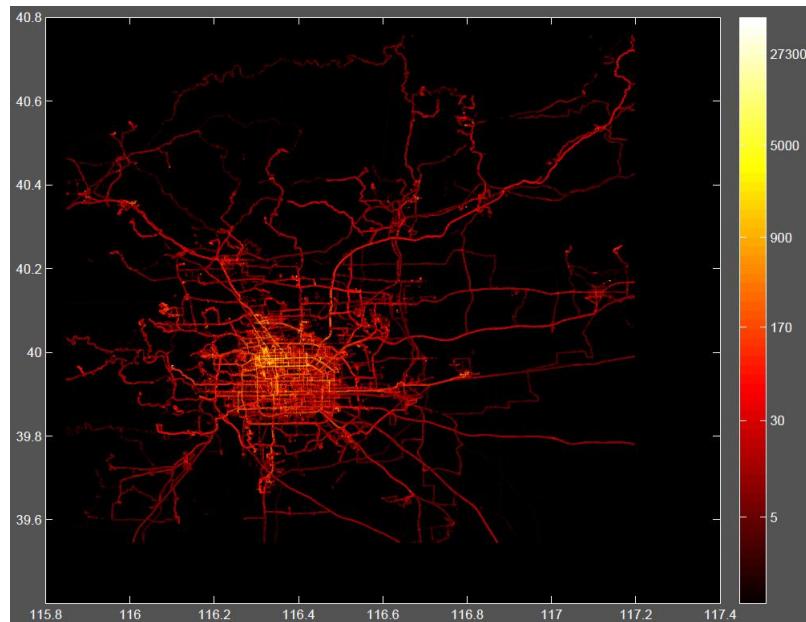
Nhóm sử dụng bộ dữ liệu quỹ đạo GPS này được thu thập trong dự án Geolife (Microsoft Research Asia) GeoLife: Building Social Networks Using Human Location History: Downloads - Microsoft Research (<https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/>)

Bộ dữ liệu này được thu thập bởi 182 người dùng trong khoảng thời gian hơn 5 năm (từ tháng 4 năm 2007 đến tháng 8 năm 2012). Quỹ đạo GPS của bộ dữ liệu này được biểu thị bằng một chuỗi các điểm được đánh dấu thời gian, mỗi điểm trong đó chứa thông tin về vĩ độ, kinh độ và độ cao.

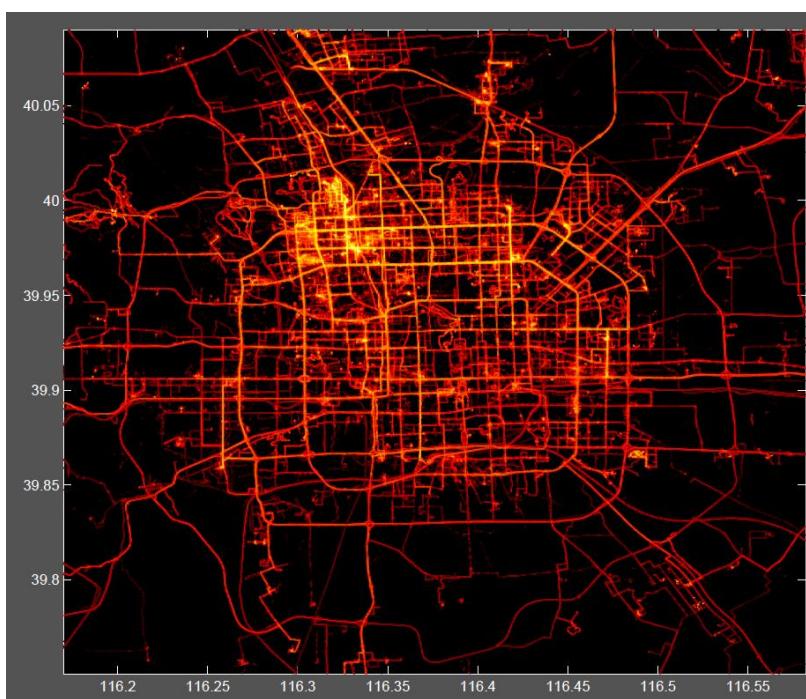
Bộ dữ liệu này chứa 17.621 quỹ đạo với tổng khoảng cách là 1.292.951 km và tổng thời gian là 50.176 giờ. Các quỹ đạo này được ghi lại bởi các máy ghi GPS và điện thoại GPS khác nhau và có nhiều tốc độ lấy mẫu khác nhau. 91.5% quỹ đạo được ghi lại trong một biểu diễn dày đặc, ví dụ: mỗi 1-5 giây hoặc cứ sau 5-10 mét mỗi điểm.

Bộ dữ liệu này đã mã hóa lại một loạt các chuyển động ngoài trời của người dùng, không chỉ bao gồm các thói quen trong cuộc sống như về nhà và đi làm mà còn có một số hoạt động giải trí và thể thao, chẳng hạn như mua sắm, tham quan, ăn uống, đi bộ đường dài và đi xe đạp. Bộ dữ liệu này có thể được sử dụng trong nhiều lĩnh vực nghiên cứu, chẳng hạn như khai thác mô hình di động, nhận dạng hoạt động của người dùng, mạng xã hội dựa trên vị trí mạng, bảo mật vị trí và đề xuất vị trí.

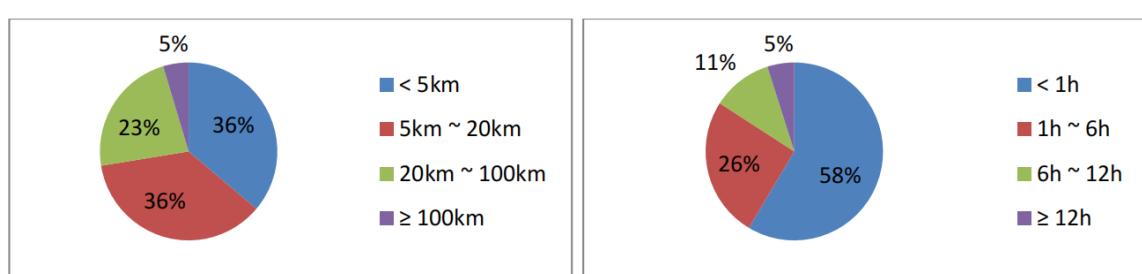
Mặc dù bộ dữ liệu này được phân phối rộng rãi ở hơn 30 thành phố của Trung Quốc và thậm chí ở một số thành phố ở Hoa Kỳ và Châu Âu, phần lớn dữ liệu được tạo ra ở Bắc Kinh, Trung Quốc. Ngoài ra, bộ dữ liệu còn đính kèm tệp 73 người dùng đã gắn nhãn quỹ đạo của họ bằng phương thức di chuyển, chẳng hạn như lái xe, đi xe buýt, đi xe đạp và đi bộ. Ở đó là tệp nhãn lưu trữ riêng trong thư mục của mỗi người dùng.



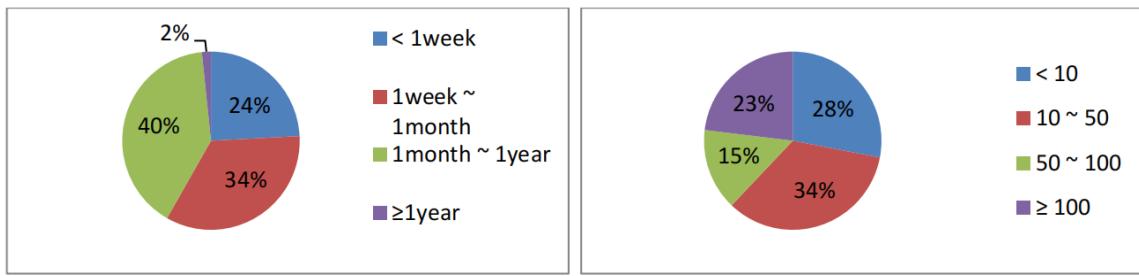
Hình 2: Tổng quan dữ liệu ở Bắc Kinh



Hình 3: Tổng quan dữ liệu ở đường vành đai 5, Bắc Kinh



Hình 4: Phân bố lô trình theo khoảng cách và thời lượng hoạt động



Hình 5: Phân bố người dùng theo khoảng thời gian ghi nhận và số lượng lô trình

2.2 Định dạng tệp Trajectory

Trong tệp Trajectory, mỗi thư mục của bộ dữ liệu này lưu trữ các tệp nhật ký GPS của người dùng, các tệp này đã được chuyển đổi sang định dạng PLT. Mỗi tệp PLT chứa một quỹ đạo duy nhất và được đặt tên theo thời gian bắt đầu của nó. Để tránh nhầm lẫn nếu giờ có thể xảy ra, thời gian sử dụng GMT trong ngày/giờ thuộc tính của từng điểm.

Định dạng PLT:

Dòng 1 . . . 6 là giới thiệu trong tập dữ liệu này và có thể bỏ qua. Các điểm được mô tả trong các dòng sau, mỗi điểm cho một dòng.

Cột 1: Vĩ độ theo độ thập phân.

Cột 2: Kinh độ theo độ thập phân.

Cột 3: Tất cả được đặt thành 0 cho tập dữ liệu này.

Cột 4: Độ cao tính bằng feet (-777 nếu không hợp lệ).

Cột 5: Ngày - số ngày (có phần phân số) đã trôi qua kể từ ngày 30/12/1899.

Cột 6: Ngày ở dạng chuỗi.

Cột 7: Thời gian dưới dạng chuỗi.

Lưu ý rằng cột 5 và cột 6&7 biểu thị cùng ngày-giờ trong tập dữ liệu này. Bạn có thể sử dụng một trong hai.

Ví dụ:

39.906631,116.385564,0,492,40097.5864583333,2009-10-11,14:04:30

39.906554,116.385625,0,492,40097.5865162037,2009-10-11,14:04:35

2.3 Định dạng tệp Transportation

Trong tệp Transportation , Các phương thức vận chuyển có thể là: đi bộ, xe đạp, xe buýt, ô tô, tàu điện ngầm, xe lửa, máy bay, thuyền, chạy bộ và xe máy. Thời gian ngày/giờ của tất cả các nhãn được ghi nhận theo GMT, mặc dù hầu hết chúng được tạo ở Trung Quốc.

Ví dụ:

Thời gian bắt đầu, Thời gian kết thúc, Phương thức di chuyển 2008/04/02 11:24:21 2008/04/02 11:50:45 bus

2008/04/03 01:07:03 2008/04/03 11:31:55 train

2008/04/03 11:32:24 2008/04/03 11:46:14 walk

2008/04/03 11:47:14 2008/04/03 11:55:07 car

Các nhãn khác nhau để sử dụng trong tương lai. Thứ hai, một người dùng có thể gắn nhãn phương thức vận chuyển của đường sắt nhẹ là tàu hỏa trong khi những người khác có thể sử dụng tàu điện ngầm làm nhãn. Trên thực tế, không có quỹ đạo có thể được ghi lại trong hệ thống tàu điện ngầm vì thiết bị ghi GPS không thể nhận được bất kỳ tín hiệu nào ở đó. Ở Bắc Kinh, đường ray nhẹ và hệ thống tàu điện ngầm được kết nối liền mạch, ví dụ: tuyến 13 (đường sắt nhẹ) được kết nối với tuyến 10 và tuyến 2, là tàu điện ngầm các hệ thống. Đôi khi, một tuyến (như tuyến 5) bao gồm một phần tàu điện ngầm và một phần đường ray nhẹ. Vì vậy, người dùng có thể có nhiều hiểu biết về phương thức vận chuyển của họ.

3 Các lý thuyết liên quan

3.1 StayPoint

Một công bố khoa học mà nhóm có tham khảo:

- Yu Zheng, Xing Xie, Wei-Ying Ma - Mining Interesting Locations and Travel Sequences From GPS

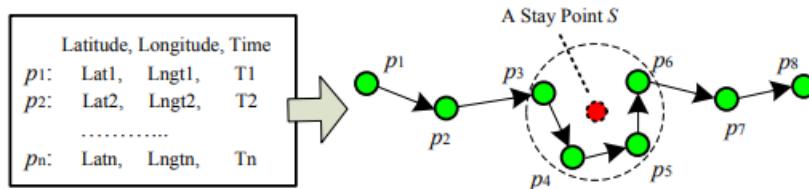


Figure 1. a GPS log, a GPS trajectory and a stay point

Hình 6: StayPoint

Một staypoint s đại diện cho một vùng trên bản đồ mà người dùng ở lại trong một khoảng thời gian nhất định. Việc xác định 1 staypoint phụ thuộc vào việc ta định nghĩa hai tham số là ngưỡng thời gian (T_{thresh}) và ngưỡng khoảng cách (D_{thresh}).

Để hình thành một staypoint, ta cần phải xác định được một tập các vị trí liên tiếp, và ngưỡng thời gian/khoảng cách sao cho khoảng cách giữa bất kỳ vị trí nào trong tập con đó so với vị trí đầu tiên của tập con đó không lớn hơn ngưỡng khoảng cách và thời gian người dùng ở lại trong tập con các vị trí đó không nhỏ hơn ngưỡng thời gian. Sau khi xác định được những yếu tố trên, ta có thể tìm được một điểm ảo trên bản đồ, đó chính là staypoint, một trung tâm của tập con được chọn, có tọa độ là trung bình cộng của các tọa độ trong tập con đó.

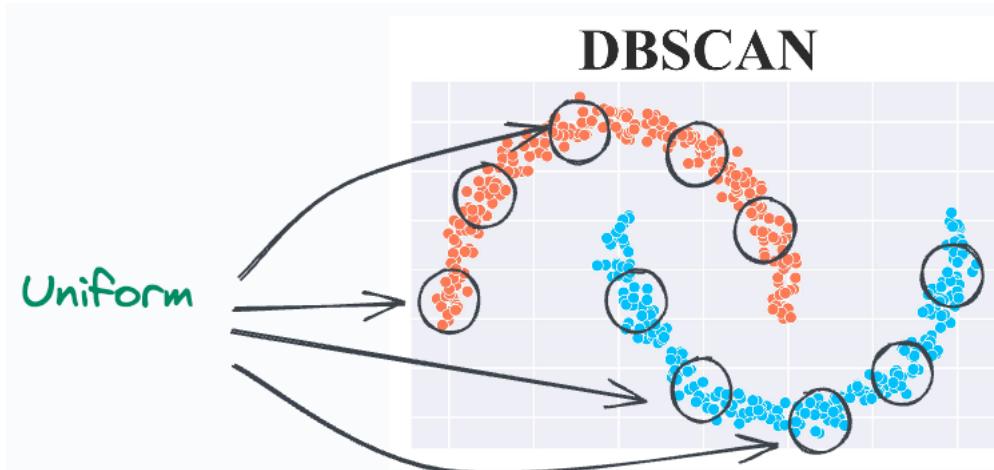
Đối với ví dụ hình ảnh ở trên, tập con P đó chính là các điểm màu xanh $P = \{p_3, p_4, p_5, p_6\}$ và điểm màu đỏ sẽ là staypoint.

Trong thực tế, các staypoint sẽ xảy ra trong 2 trường hợp:

- Người dùng đứng yên trong một ngưỡng thời gian. Trong hầu hết các trường hợp, trạng thái này xảy ra khi mọi người nhập một tòa nhà và mất tín hiệu vệ tinh trong một khoảng thời gian cho đến khi đến trở lại ngoài trời.
- Tình huống khác là khi người dùng đi loanh quanh trong một phạm vi không gian địa lý nhất định trong một khoảng thời gian. Trong hầu hết các trường hợp, điều này tình huống xảy ra khi mọi người đi du lịch ngoài trời và bị thu hút bởi môi trường xung quanh.

3.2 Giải thuật DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) là một thuật toán phân cụm dữ liệu trong Machine Learning, dùng để phân cụm dữ liệu dựa trên mật độ của chúng. Thuật toán này được giới thiệu bởi Martin Ester, Hans-Peter Kriegel, Jörg Sander và Xiaowei Xu vào năm 1996.



Hình 7: Giải thuật DBSCAN

Ý tưởng chính của DBSCAN là phát hiện và phân nhóm các điểm dữ liệu gần nhau dựa trên mật độ của chúng, mà không cần định nghĩa trước số lượng cụm. Thuật toán này xác định được các điểm nhiễu (noise) và cụm (cluster) dựa trên hai thông số chính là bán kính (eps) và số lượng điểm dữ liệu tối thiểu (min_samples) trong một cụm. Các điểm có mật độ dày hơn được coi là thuộc cùng một cụm, trong khi các điểm có mật độ thấp hơn sẽ được coi là điểm nhiễu.

Các bước chính của thuật toán DBSCAN bao gồm:

1. Chọn một điểm dữ liệu ngẫu nhiên trong tập dữ liệu.
2. Tìm tất cả các điểm trong bán kính eps của điểm đó.
3. Nếu số lượng điểm trong bán kính eps của điểm đó lớn hơn hoặc bằng min_samples, ta sẽ tạo ra một cụm mới và thêm các điểm này vào cụm đó.
4. Lặp lại quá trình tìm kiếm các điểm kề cận và thêm vào cụm cho tất cả các điểm mới được thêm vào cụm.
5. Nếu không còn điểm nào được tìm thấy trong bán kính eps của một điểm, ta chuyển sang điểm tiếp theo và lặp lại quá trình.
6. Khi tất cả các điểm đã được phân vào các cụm, các điểm còn lại được coi là điểm nhiễu.

DBSCAN là một thuật toán mạnh mẽ trong việc phân cụm dữ liệu và có thể áp dụng cho nhiều lĩnh vực khác nhau, nhưng cần phải chú ý đến việc định nghĩa các thông số như eps và min_samples để đạt được kết quả tối ưu.

4 Các công nghệ sử dụng

4.1 Apache Hadoop & HDFS

4.1.1 Hadoop

Hadoop là một dự án được phát triển từ đầu thế kỷ XXI. Đến năm 2008, Hadoop được Yahoo công bố là một dự án mã nguồn mở cho cộng đồng.

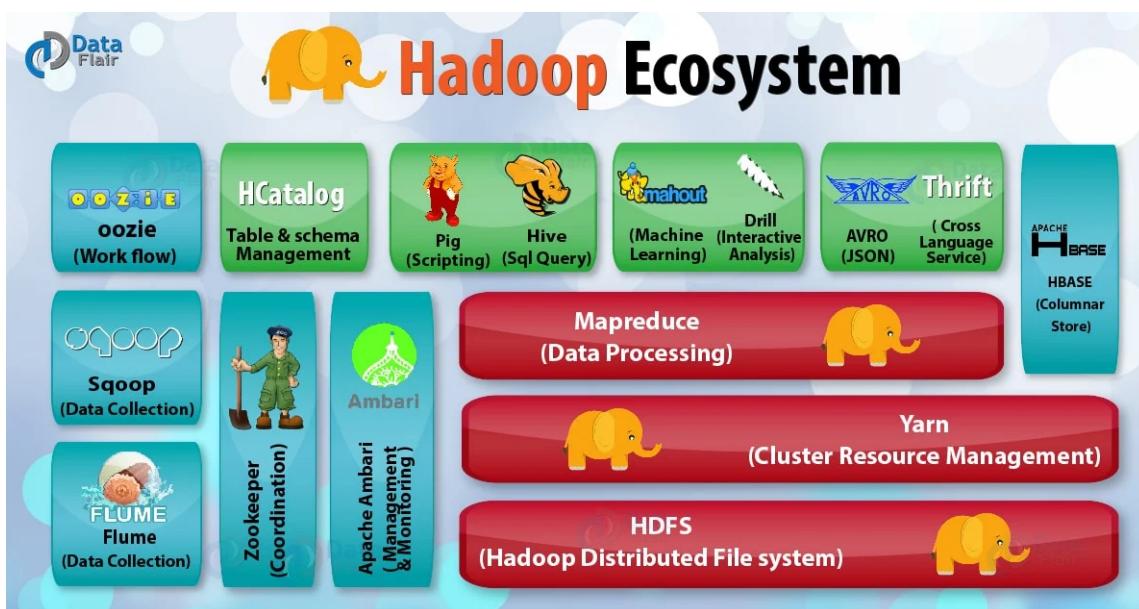


Hình 8: Hadoop

Hadoop là một framework mã nguồn mở dùng để xử lý dữ liệu lớn. Hadoop gồm có 3 thành phần chính bao gồm Hệ thống tệp phân tán Hadoop Distributed File System (HDFS) để lưu trữ, sử dụng MapReduce để xử lý dữ liệu song song và Yet Another Resource Negotiator (YARN). Hadoop được sử dụng trong xử lý hàng loạt, lưu trữ dữ liệu và phân tích dữ liệu lớn. Một số đặc điểm của Hadoop

1. **Lưu Trữ Dữ Liệu Lớn (Big Data):** Hadoop giải quyết vấn đề lưu trữ và xử lý dữ liệu lớn, cho phép lưu trữ và xử lý các tập dữ liệu có kích thước lớn trên nhiều máy tính.
2. **Phân Tán và Mở Rộng:** Hadoop chia nhỏ dữ liệu thành các phần nhỏ và phân tán chúng trên nhiều nút máy tính trong một cụm. Điều này giúp mở rộng quy mô của hệ thống theo nhu cầu.
3. **MapReduce:** Hadoop sử dụng mô hình lập trình MapReduce để xử lý dữ liệu phân tán. Nó chia tác vụ xử lý thành hai pha: pha "Map" (áp dụng hàm map cho từng phần dữ liệu) và pha "Reduce" (tổng hợp kết quả từ các pha "Map").
4. **Hadoop Distributed File System (HDFS):** HDFS là hệ thống tệp phân tán được Hadoop sử dụng để lưu trữ dữ liệu. Dữ liệu được chia thành các khối nhỏ và phân tán trên nhiều nút máy tính trong cụm Hadoop.

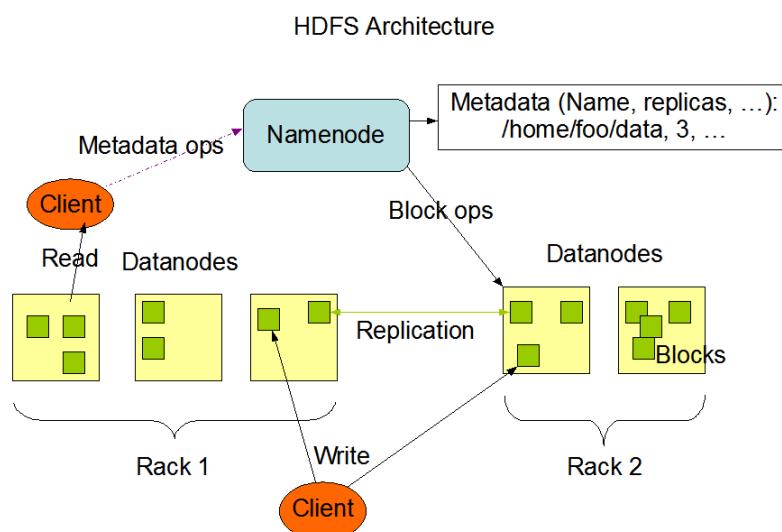
5. **Cộng Đồng Lớn và Hỗ Trợ Mạnh Mẽ:** Hadoop có một cộng đồng người dùng lớn và tích cực, cung cấp nhiều tài liệu, diễn đàn, và nguồn thông tin học tập.
6. **Hỗ Trợ Đa Ngôn Ngữ:** Hadoop hỗ trợ nhiều ngôn ngữ lập trình như Java, Python, và Ruby, giúp người phát triển sử dụng ngôn ngữ ưa thích của họ.
7. **Tích Hợp Với Các Công Nghệ Khác:** Hadoop có thể tích hợp với nhiều công nghệ khác như Apache Spark, Apache Hive, Apache HBase, và nhiều dự án khác để cung cấp các tính năng mở rộng và đa dạng.
8. **Scalability (Khả Năng Mở Rộng):** Hadoop có khả năng mở rộng linh hoạt, có thể thêm máy tính vào cụm để tăng khả năng xử lý và lưu trữ dữ liệu.



Hình 9: Hệ sinh thái của Hadoop

4.1.2 HDFS

Hadoop Distributed File System (HDFS) là lớp lưu trữ của Hadoop. HDFS là một hệ thống lưu trữ file phân tán.



Hình 10: Kiến trúc của HDFS

Các thành phần của HDFS bao gồm:

- Namenode: Giải quyết các vấn đề về lưu trữ, bảo trì và quyền truy cập tới hệ thống file và chứa metadata của các Datanode nó quản lý
- Secondary namenode: Dùng để backup trong trường hợp Namenode bị lỗi.
- Datanode: Dùng để lưu trữ và xử lý các công việc.

Các tính năng của HDFS:

- Khả năng chịu lỗi cao: Khi namenode bị lỗi, có secondary namenode để backup. Khi một Datanode bị lỗi sẽ có một datanode khác được phân công công việc để xử lý
- Có thể lưu trữ dữ liệu lớn lên đến hàng petabytes.
- Hiệu quả về mặt chi phí lưu trữ.
- Tính mở rộng cao: Dễ dàng để mở rộng khi cần thiết.

4.2 Apache Spark

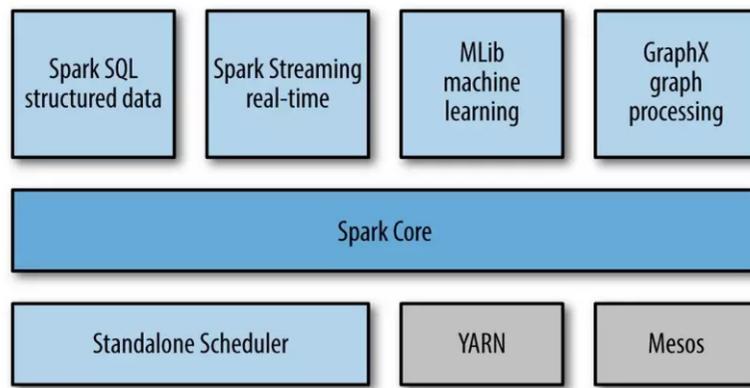
Apache Spark là một framework mã nguồn mở tính toán cụm được phát triển vào năm 2009 như một dự án nghiên cứu tại AMPLab của UC Berkeley, một sự hợp tác giữa sinh viên, nhà nghiên cứu và giảng viên, tập trung vào các lĩnh vực ứng dụng sử dụng nhiều dữ liệu. Mục tiêu của Spark là tạo ra một khung mới, được tối ưu hóa để xử lý lặp lại nhanh như học máy và phân tích dữ liệu tương tác, trong khi vẫn giữ được khả năng mở rộng và khả năng chịu lỗi của Hadoop MapReduce. Bài báo đầu tiên có tựa đề “Spark: Cluster Computing with Working Sets” được xuất bản vào tháng 6 năm 2010 và Spark có nguồn mở theo giấy phép BSD. Vào tháng 6 năm 2013, Spark đã bắt đầu giai đoạn ươm tạo tại Quỹ phần mềm Apache (ASF) và được thành lập như một Dự án cấp cao nhất của Apache vào tháng 2 năm 2014. Spark có thể chạy độc lập, trên Apache Mesos hoặc thường xuyên nhất là trên Apache Hadoop.

Ngày nay, Spark đã trở thành một trong những dự án tích cực nhất trong hệ sinh thái Hadoop, với nhiều tổ chức áp dụng Spark cùng với Hadoop để xử lý dữ liệu lớn. Năm 2017, Spark có 365.000 thành viên buổi gặp mặt, thể hiện mức tăng trưởng gấp 5 lần trong hai năm. Nó đã nhận được sự đóng góp của hơn 1.000 nhà phát triển từ hơn 200 tổ chức kể từ năm 2009.



Hình 11: Spark

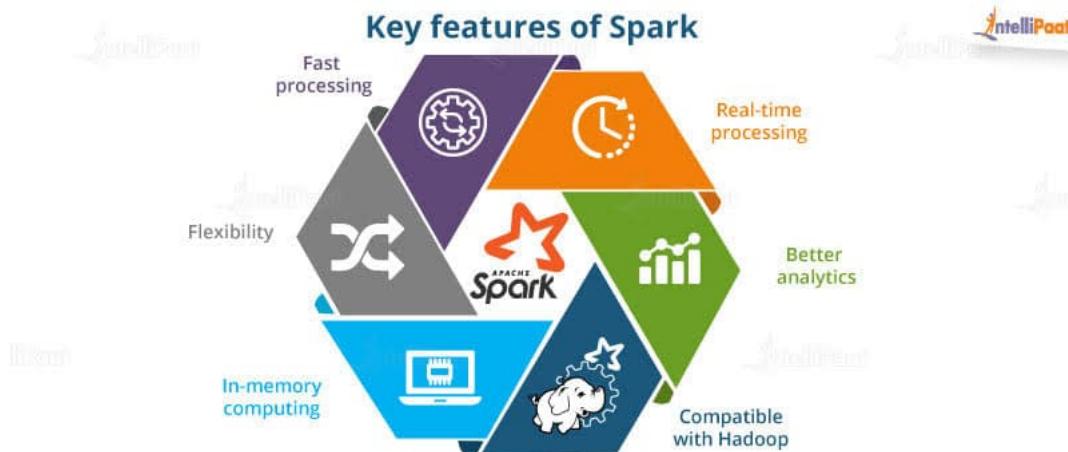
Apache Spark cho phép ta xây dựng những mô hình dự đoán nhanh chóng với khả năng thực hiện tính toán cùng lúc trên một nhóm các máy tính hay trên toàn bộ các tập dữ liệu mà không cần thiết phải trích xuất các mẫu tính toán thử nghiệm. Tốc độ xử lý dữ liệu của Apache Spark có được là do khả năng thực hiện các tính toán trên nhiều máy khác nhau cùng một lúc tại bộ nhớ trong (in-memories) hay hoàn toàn trên RAM.



Hình 12: Thành phần của Spark

Apache Spark gồm có 5 thành phần chính: Spark Core, Spark Streaming, Spark SQL, MLlib và GraphX.

- Spark Core là nền tảng cho các thành phần còn lại và các thành phần này muốn khởi chạy được thì đều phải thông qua Spark Core do Spark Core đảm nhận vai trò thực hiện công việc tính toán và xử lý trong bộ nhớ, đồng thời nó cũng tham chiếu các dữ liệu được lưu trữ tại các hệ thống lưu trữ bên ngoài, cung cấp các chức năng cơ bản để xử lý dữ liệu phân tán (distributed data processing - DDP) bao gồm quản lý bộ nhớ, thiết lập lịch tác vụ và khôi phục lỗi.
- Spark SQL cung cấp một kiểu data abstraction mới (Schema ADD) nhằm hỗ trợ cho cả kiểu dữ liệu có cấu trúc (structured data) và dữ liệu nửa cấu trúc (semi-structured data – thường là dữ liệu dữ liệu có cấu trúc nhưng không đồng nhất và cấu trúc của dữ liệu phụ thuộc vào chính nội dung của dữ liệu ấy). Spark SQL hỗ trợ DSL (Domain-specific language) để thực hiện các thao tác trên DataFrames bằng ngôn ngữ Scala, Java hoặc Python và nó cũng hỗ trợ cả ngôn ngữ SQL với giao diện command-line và ODBC/JDBC server. Ngoài ra, Spark SQL còn tích hợp tốt với các công cụ ETL và Spark Streaming, điều đó giúp nó trở thành một phần quan trọng trong hệ thống phân tích dữ liệu phân tán và các ứng dụng trực tuyến trong thời gian thực
- Spark Streaming được sử dụng để thực hiện việc phân tích stream bằng việc coi stream là các mini-batches và thực hiện kỹ thuật RDD transformation đối với các dữ liệu mini-batches này. Qua đó cho phép các đoạn code được viết cho xử lý batch có thể được tận dụng lại vào việc xử lý stream, làm cho việc phát triển lambda architecture được dễ dàng hơn. Tuy nhiên điều này lại tạo ra độ trễ trong xử lý dữ liệu (độ trễ chính bằng mini-batch duration) và do đó nhiều chuyên gia cho rằng Spark Streaming không thực sự là công cụ xử lý streaming giống như Storm hoặc Flink.
- MLlib (Machine Learning Library) là một nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ. Theo các so sánh benchmark Spark MLlib nhanh hơn 9 lần so với phiên bản chạy trên Hadoop (Apache Mahout).
- GraphX là nền tảng xử lý đồ thị dựa trên Spark. GraphX đi kèm với lựa chọn các thuật toán phân tán để xử lý cấu trúc đồ thị. Để thực hiện các tính toán trên đồ thị, GraphX cung cấp RDD, VertexRDD và EdgeRDD trong đó Vertex (đỉnh) và Edge (cạnh).



Hình 13: Đặc điểm của Spark

Apache Spark có một số đặc điểm nổi bật như:

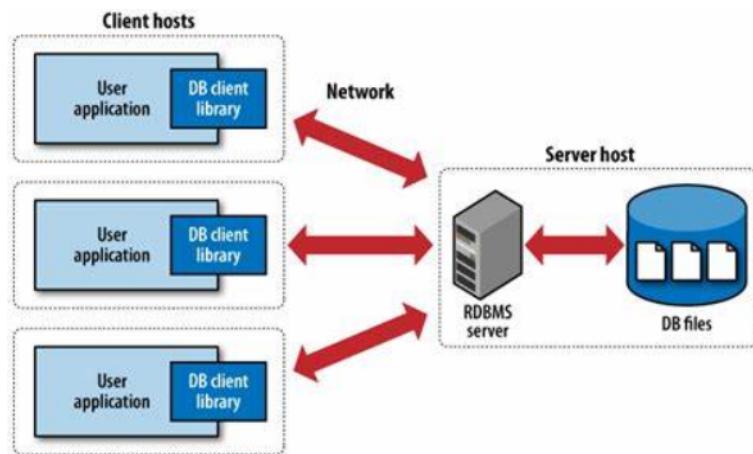
1. **Xử lý Dữ liệu Lớn (Big Data):** Apache Spark được thiết kế để xử lý dữ liệu lớn, cung cấp khả năng xử lý dữ liệu trong quy mô petabyte.
2. **Tốc Độ Xử Lý Cao:** Spark cung cấp hiệu suất xử lý cao bằng cách sử dụng mô hình tính toán trong bộ nhớ (in-memory computing), giảm thiểu việc đọc và ghi dữ liệu trên đĩa, làm tăng tốc độ xử lý.
3. **Đa Ngôn Ngữ Hỗ Trợ:** Spark hỗ trợ nhiều ngôn ngữ lập trình như Scala, Java, Python, và R, giúp người phát triển sử dụng ngôn ngữ ưa thích của họ để viết ứng dụng Spark.
4. **Hỗ Trợ Đa Dạng Công Cụ Xử Lý Dữ Liệu:** Spark cung cấp API cho nhiều công cụ xử lý dữ liệu như SQL (Spark SQL), machine learning (MLlib), đồ đồ thị (GraphX), và xử lý luồng dữ liệu (Spark Streaming).
5. **Phân Tán và Linh Hoạt:** Spark có khả năng phân tán tính toán trên nhiều node, giúp mở rộng quy mô của hệ thống một cách linh hoạt.
6. **Resilient Distributed Datasets (RDDs):** RDDs là một khái niệm quan trọng trong Spark, là cấu trúc dữ liệu phân tán và có thể tự phục hồi sau khi có lỗi, giúp đảm bảo tính ổn định và đồng nhất của dữ liệu.
7. **Hỗ Trợ Cộng Đồng Mạnh Mẽ:** Spark có một cộng đồng người dùng rộng lớn và tích cực, cung cấp nhiều tài liệu, hướng dẫn, và diễn đàn để hỗ trợ người sử dụng.
8. **Tích Hợp VỚI Hadoop:** Spark có thể tích hợp chặt chẽ với Hadoop Distributed File System (HDFS) và có thể chạy trên nền tảng Hadoop.
9. **Hỗ Trợ Streaming Dữ Liệu Thời Gian Thực:** Spark Streaming cho phép xử lý dữ liệu streaming (dữ liệu thời gian thực), mở rộng khả năng ứng dụng Spark sang các tình huống yêu cầu xử lý liên tục.
10. **Tích Hợp VỚI Nhiều Nguồn Dữ Liệu:** Spark có khả năng tích hợp dữ liệu từ nhiều nguồn khác nhau như HDFS, Apache HBase, Apache Hive, và nhiều nguồn dữ liệu khác.

4.3 MySQL



Hình 14: MySQL

Công ty Thụy Điển MySQL AB phát triển MySQL vào năm 1994. Đến năm 2010, được Oracle mua lại. MySQL là một hệ quản trị cơ sở dữ liệu dạng quan hệ mã nguồn mở (Relational Database Management System, viết tắt là RDBMS) hoạt động theo mô hình client-server.



Hình 15: Kiến trúc của MySQL

Một số đặc điểm nổi bật của MySQL

- Mã nguồn mở:** MySQL là hệ thống quản trị cơ sở dữ liệu quan hệ mã nguồn mở, có nghĩa là người dùng có thể sử dụng, thay đổi và phân phối mã nguồn mà không cần phải trả bất kỳ chi phí nào.
- Độ tin cậy cao:** MySQL được biết đến với độ ổn định và độ tin cậy cao. Nó thường được sử dụng trong các hệ thống yêu cầu tính sẵn sàng cao như các trang web lớn, ứng dụng doanh nghiệp quan trọng.
- Tương thích chuẩn SQL:** MySQL tuân theo các tiêu chuẩn SQL, giúp người phát triển chuyển đổi giữa các hệ quản trị cơ sở dữ liệu một cách dễ dàng hơn.
- Hỗ trợ nhiều loại lưu trữ:** MySQL hỗ trợ nhiều loại lưu trữ như InnoDB, MyISAM, MEMORY, ARCHIVE, và nhiều loại khác, mỗi loại phù hợp cho một mục đích sử dụng cụ thể.
- Hỗ trợ ACID:** MySQL thực hiện các nguyên tắc ACID (Atomicity, Consistency, Isolation, Durability), giúp đảm bảo tính nhất quán và an toàn của cơ sở dữ liệu trong mọi tình huống.
- Quản lý đồng thời:** MySQL hỗ trợ quản lý đồng thời, cho phép nhiều người dùng truy cập cùng một cơ sở dữ liệu mà không làm ảnh hưởng đến tính nhất quán.

7. **Tiện ích dòng lệnh và giao diện đồ họa:** MySQL cung cấp cả tiện ích dòng lệnh và giao diện đồ họa (qua các công cụ như MySQL Workbench) để quản lý cơ sở dữ liệu.
8. **Hỗ trợ nhiều ngôn ngữ lập trình:** MySQL được tích hợp tốt với nhiều ngôn ngữ lập trình phổ biến như PHP, Java, Python, C++, và nhiều ngôn ngữ khác.
9. **Hỗ trợ giao thức mạng:** MySQL hỗ trợ nhiều giao thức mạng như TCP/IP, socket, và named pipes, giúp kết nối và tương tác với cơ sở dữ liệu từ xa.
10. **Cộng đồng lớn và hỗ trợ mạnh mẽ:** Với cộng đồng người dùng rộng lớn và sự hỗ trợ mạnh mẽ từ cộng đồng mã nguồn mở, MySQL có sẵn nhiều tài liệu, diễn đàn và tài nguyên học tập.

Tóm lại, MySQL là một hệ quản trị cơ sở dữ liệu mạnh mẽ, linh hoạt và được sử dụng rộng rãi trên toàn cầu.

4.4 PowerBI

PowerBI là một phần mềm BI (business intelligence) của Microsoft. PowerBI cung cấp nhiều tính năng mạnh mẽ để phân tích dữ liệu, bao gồm:

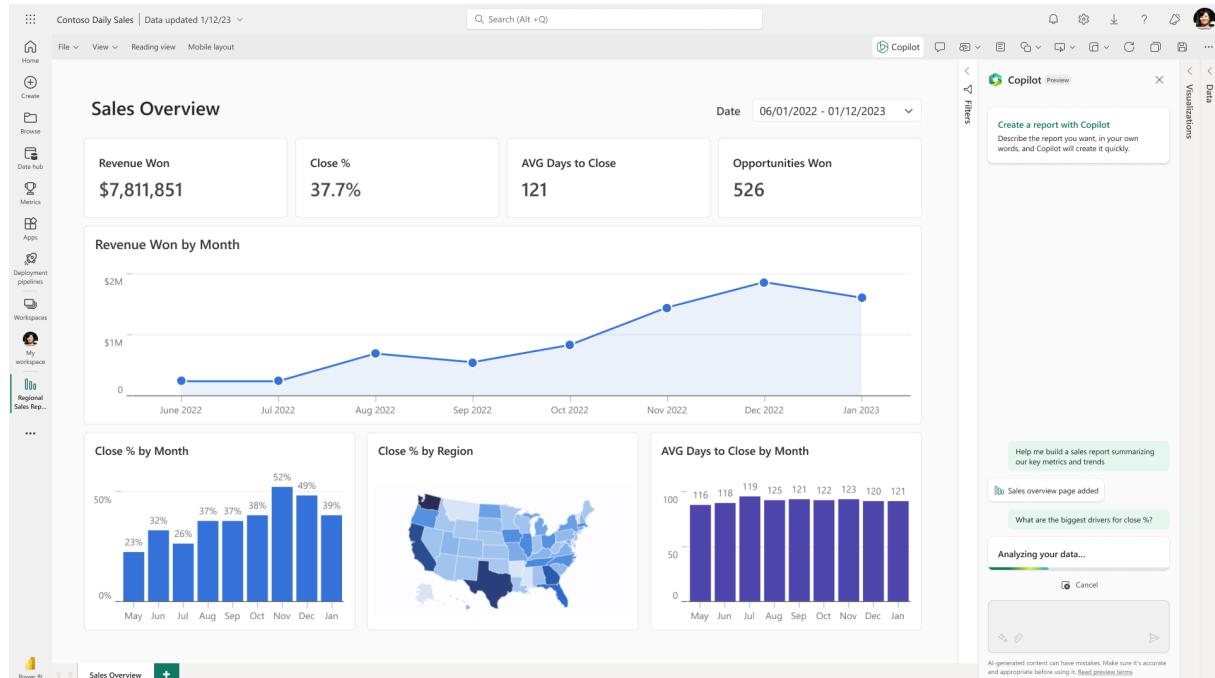
- **Tạo biểu đồ và báo cáo trực quan:** PowerBI cung cấp nhiều loại biểu đồ và báo cáo trực quan để giúp người dùng dễ dàng hiểu được dữ liệu.
- **Phân tích dữ liệu theo nhiều cách khác nhau:** PowerBI cung cấp nhiều cách khác nhau để phân tích dữ liệu, bao gồm phân tích thống kê, phân tích chuỗi thời gian, và phân tích dữ liệu theo vị trí.
- **Chia sẻ dữ liệu và báo cáo với người khác:** PowerBI cho phép người dùng chia sẻ dữ liệu và báo cáo với người khác một cách dễ dàng.



Hình 16: PowerBI

Các tính năng chính của PowerBI:

- **Tạo biểu đồ và báo cáo trực quan:** PowerBI cung cấp nhiều loại biểu đồ và báo cáo trực quan, bao gồm: Biểu đồ tròn, Biểu đồ đường, Bản đồ... PowerBI cũng cho phép người dùng tùy chỉnh các biểu đồ và báo cáo trực quan để phù hợp với nhu cầu của mình.
- **Phân tích dữ liệu theo nhiều cách khác nhau:** PowerBI cung cấp nhiều cách khác nhau để phân tích dữ liệu, bao gồm: Phân tích thống kê - PowerBI cung cấp các tính năng phân tích thống kê cơ bản, bao gồm tính toán trung bình, trung vị, phương sai, độ lệch chuẩn, v.v.; Phân tích chuỗi thời gian - PowerBI cung cấp các tính năng phân tích chuỗi thời gian, bao gồm phân tích xu hướng, phân tích chu kỳ, và phân tích mùa vụ; Phân tích dữ liệu theo vị trí - PowerBI cung cấp các tính năng phân tích dữ liệu theo vị trí, bao gồm phân tích cụm, phân tích phân tán, và phân tích đường dẫn.
- **Chia sẻ dữ liệu và báo cáo với người khác:** PowerBI cho phép người dùng chia sẻ dữ liệu và báo cáo với người khác một cách dễ dàng. Người dùng có thể chia sẻ dữ liệu và báo cáo dưới dạng tệp PDF, tệp PowerPoint, hoặc tệp HTML. Ngoài ra, PowerBI cũng cho phép người dùng chia sẻ dữ liệu và báo cáo thông qua cổng thông tin Power BI.



Hình 17: Giao diện cơ bản của PowerBI

PowerBI là một công cụ BI mạnh mẽ với nhiều lợi ích, bao gồm:

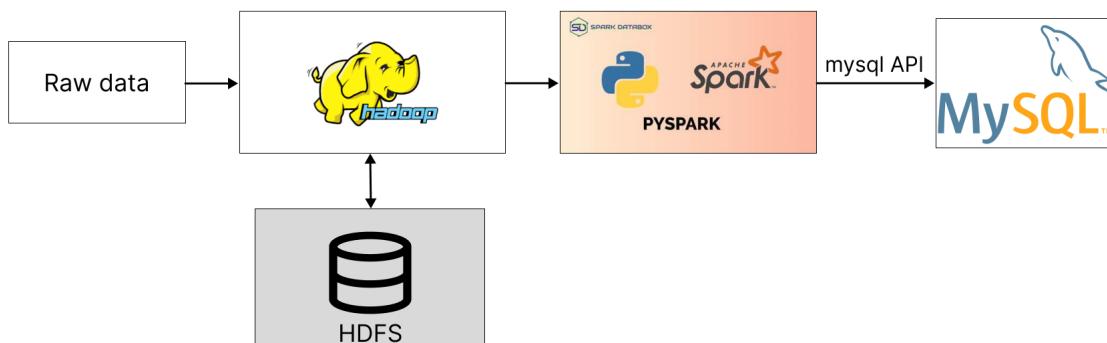
- **Giúp người dùng dễ dàng hiểu được dữ liệu:** PowerBI cung cấp nhiều loại biểu đồ và báo cáo trực quan giúp người dùng dễ dàng hiểu được dữ liệu.
- **Giúp người dùng phân tích dữ liệu theo nhiều cách khác nhau:** PowerBI cung cấp nhiều cách khác nhau để phân tích dữ liệu, giúp người dùng có thể tìm hiểu sâu hơn về dữ liệu.
- **Giúp người dùng chia sẻ dữ liệu và báo cáo với người khác:** PowerBI cho phép người dùng chia sẻ dữ liệu và báo cáo với người khác một cách dễ dàng, giúp người dùng thảo luận và đưa ra quyết định dựa trên kết quả phân tích.

Tóm lại, PowerBI là một công cụ BI mạnh mẽ với nhiều tính năng phù hợp cho việc phân tích dữ liệu. PowerBI có thể được sử dụng để phân tích dữ liệu từ nhiều nguồn khác nhau, bao gồm dữ liệu bảng tính, dữ liệu cơ sở dữ liệu, và dữ liệu từ các ứng dụng web.

5 Hướng giải quyết bài vấn đề

5.1 Xây dựng hệ thống lưu trữ dữ liệu

Nhóm sẽ xử lý dữ liệu thông qua hadoop, sau đó sử dụng pyspark để chuyển dữ liệu phi cấu trúc sang có cấu trúc được lưu trữ trên mysql



Hình 18: *Luồng chuyển đổi dữ liệu*

Các công cụ sử dụng:

- Hadoop
 - Spark và Pyspark
 - Mysql Workbench

Vì dữ liệu có những dạng file khác nhau gồm file dưới format plt. Do đó để có thể chuẩn hóa dữ liệu về dạng có cấu trúc (structured data) thì trước đó ta phải lưu dữ liệu vào một hệ thống có thể lưu trữ được 2 loại format file trên. Do đó nhóm quyết định sử dụng Hadoop với lớp lưu trữ của nó là HDFS giữ vai trò như một "data lake". Nơi mà có thể lưu trữ tất cả các loại dữ liệu.

Sau khi tải dữ liệu về, Ta đẩy dữ liệu từ máy local lên HDFS bằng câu lệnh sau:

§ HDES DES -PUT <PATH-TO-LOCAL-DATA> <PATH-IN-HDES>

Sau khi chạy câu lệnh này, chúng ta có thể kiểm tra dữ liệu đã nhập thành công hay chưa thông qua localhost:9870

Browse Directory

File List							Actions
File ID	Name	Type	Size	Last Modified	Replication	Block Size	Actions
1	datafile1	File	0 B	Nov 11 22:37	0	0 B	Data2 Delete

Hình 10: Kép lồng trang để kiểm thử một ứng dụng HDFS



/Data2									Go!	File	Upload	Print	Export
Show 25 entries									Search:				
	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name					
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	154	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	155	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	156	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	157	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	158	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	159	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	160	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	161	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	162	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:24	0	0 B	163	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:26	0	0 B	164	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:26	0	0 B	165	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:26	0	0 B	166	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:26	0	0 B	167	Trash				
□	drwxr-xr-x	ADMIN	supergroup	0 B	Nov 11 22:27	0	0 B	168	Trash				

Hình 20: Kiểm tra dữ liệu bên trong folder Data

Sau khi lưu dữ liệu thô vào HDFS, nhóm tiến hành sử dụng PySpark cùng với Mysql API để tiền xử lý dữ liệu đưa dữ liệu về dạng có cấu trúc.

Trước khi thực hiện xử lý dữ liệu cần install và import các thư viện cần thiết

```
1 import pyspark
2 pip install Flask-SQLAlchemy
3 pip install pymysql==0.9.3
4
5 from pyspark.sql import SparkSession
6 from pyspark.sql import SQLContext
7 from pyspark.sql.types import StructType
8 from pyspark.sql import Row
9 import pandas as pd
10 from functools import reduce
11 from sqlalchemy import create_engine
12 import pymysql
```

Tạo một Spark Session để thực thi:

```
1 spark1 = SparkSession.builder.master("local").appName("read_hdfs").getOrCreate()
```

Tạo API kết nối HDFS với mysql lưu trữ dữ liệu vào mysql

```
1 sqlEngine1 = create_engine('mysql+pymysql://root:12345678@127.0.0.1/trajectory')
2 sqlEngine2 = create_engine('mysql+pymysql://root:12345678@127.0.0.1/label')
```

Xử lý dữ liệu các file trajectory dưới dạng plt. Ở đây, chúng ta sẽ xử lý cho từng người với toàn bộ dữ liệu của họ.

```
1 for k in range(182):
2     directory, saveloc = getDir(k)
3     print(directory)
4     rdd = spark1.sparkContext.wholeTextFiles(directory)
```

```
5     rdd_val = rdd.values()
6     rdd_val = rdd_val.collect()
7     d = 0
8     lst = []
9     for tmp in rdd_val:
10         data = tmp.splitlines(True)
11         data = data[6:]
12         for i in range(len(data)):
13             tmpstr = data[i][-2]
14             str = reduce(lambda prev, curr: prev + curr, tmpstr)
15             lst += [str.split(",")]
16     df = pd.DataFrame(lst, columns = ["Latitude", "Longitude", "Nonuse", "Altitude",
17     ↪ "UnixTime", "Date", "Time"])
18     df["Latitude"] = df["Latitude"].astype(float)
19     df["Longitude"] = df["Longitude"].astype(float)
20     df["Altitude"] = df["Altitude"].astype(float)
21     df["UnixTime"] = df["UnixTime"].astype(float)
22     df = df.drop("Nonuse", axis = 1)
23     df.to_sql(saveloc,sqlEngine1,if_exists='append')
```

Xử lý file label dưới dạng txt file:

```
1 for k in range(182):
2     directory, saveloc = fileDir(k)
3     try:
4         rdd = spark1.sparkContext.textFile(directory)
5         rdd.collect()
6         lst = rdd.collect()
7         for i in range(len(lst)):
8             lst[i] = lst[i].split('\t')
9             column = lst[0]
10            lst = lst[1:]
11            df = pd.DataFrame(lst, columns = column)
12            df.to_sql(saveloc,sqlEngine2,if_exists='replace')
13            print(directory)
14     except:
15         pass
```

Kết quả sau khi chuyển dữ liệu từ HDFS sang mysql

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| label |
| mysql |
| performance_schema |
| sakila |
| sys |
| trajectory |
| world |
+-----+
8 rows in set (0.02 sec)

mysql> |
```

Hình 21: 2 databases: trajectory và label đã được tạo

Kiểm tra dữ liệu trong 2 databases:

```
mysql> SELECT COUNT(*) FROM information_schema.tables WHERE table_schema = 'trajectory';
+-----+
| COUNT(*) |
+-----+
|      182 |
+-----+
1 row in set (0.00 sec)

mysql> |
```

Hình 22: Database trajectory

```
mysql> SELECT COUNT(*) FROM information_schema.tables WHERE table_schema = 'label';
+-----+
| COUNT(*) |
+-----+
|      69 |
+-----+
1 row in set (0.00 sec)

mysql> |
```

Hình 23: Database label

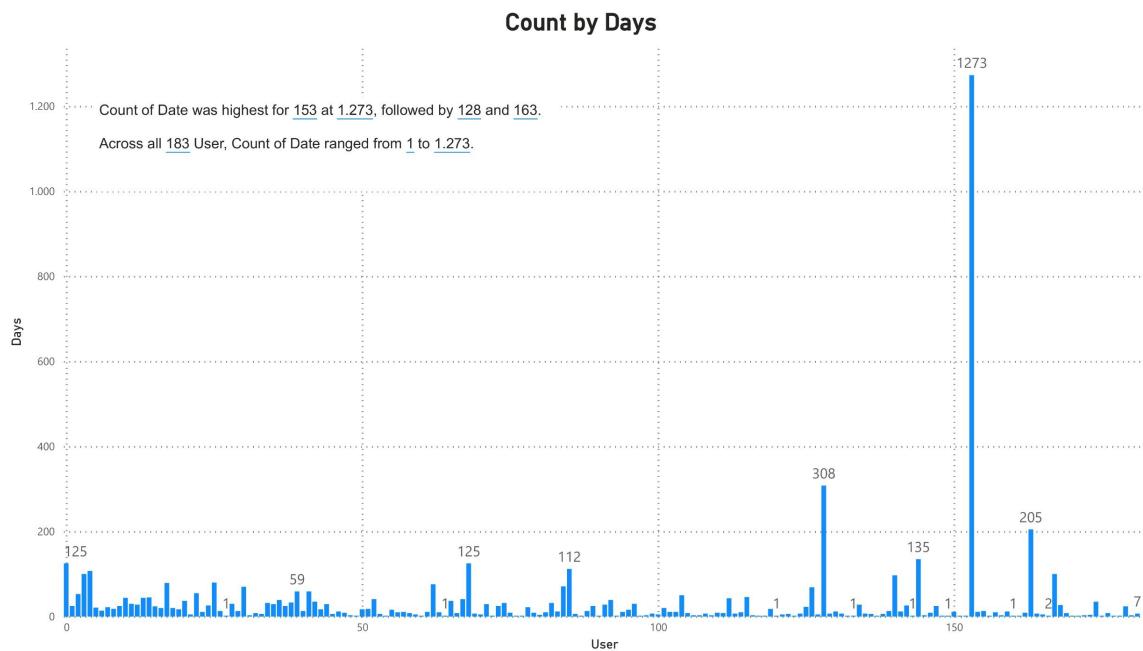
5.2 Thống kê dữ liệu của người dùng

Tổng ngày, quảng đường và quảng đường trung bình của mỗi người dùng:

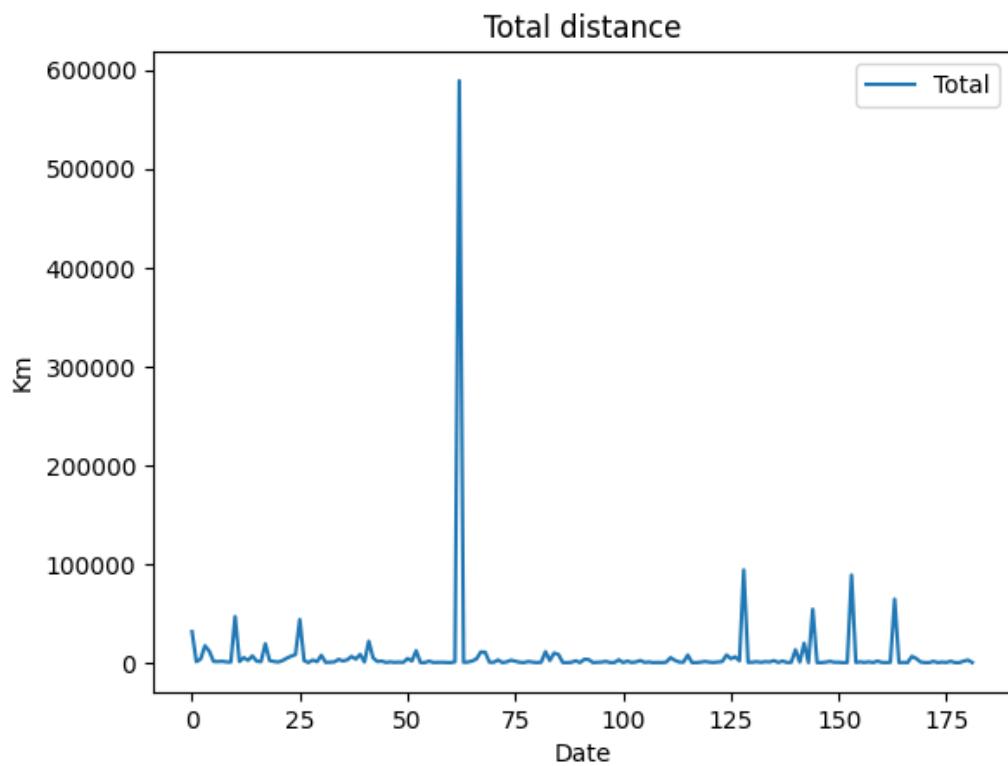
User	Date	Total	Average
0	544	31438.656231	57.791647
1	45	1003.664988	22.303666
2	118	3556.112660	30.136548
3	239	17113.235450	71.603496
4	263	11580.709696	44.033117
...
177	1	28.428026	28.428026
178	1	0.971681	0.971681
179	45	1533.615302	34.080340
180	6	2594.151864	432.358644
181	15	135.534470	9.035631
182 rows × 3 columns			

Hình 24: Tổng quan về dữ liệu người dùng

Biểu đồ thể hiện số ngày tham gia của mỗi User:



Hình 25: Biểu đồ thể hiện số ngày tham dự GPS của user



Hình 26: Biểu đồ thể hiện quãng đường tham dự của mỗi user

5.3 Phân cụm các vị trí GPS của người dùng

Ở phương diện này, nhóm tiến hành sử dụng thuật toán DBSCAN để phân cụm các điểm trong giữ liệu GPS của người dùng theo mật độ các điểm.

Hai số thông eps và min_samples của mô hình này được tính toán thông qua thuật toán Nearest Neighbor.

Với bộ dữ liệu dành cho người dùng số 70 trong tháng 9 năm 2008, ta có $\text{min_samples} = 2$ và $\text{eps} = 0.0023502363947231686$

```
1 # create color list for set of labels
2 import webcolors
3 colors = list(webcolors.HTML4_NAMES_TO_HEX)
4 colors[:len(labels)]
5 colors

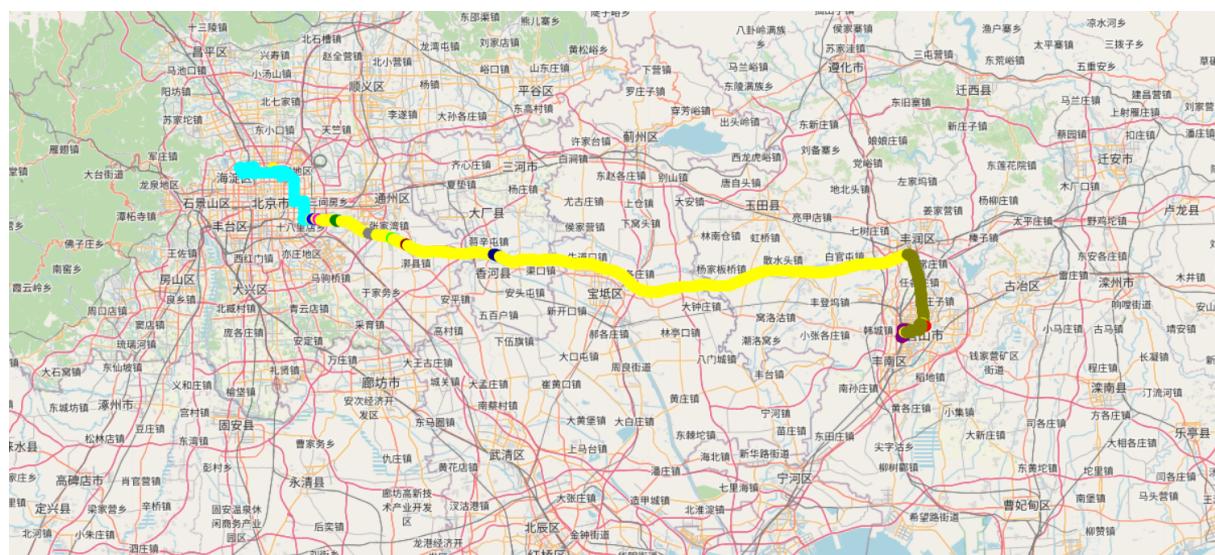
# labeling for each location
dbSCAN=DBSCAN(eps=eps_sample, min_samples=optimal_min_samples)
dbSCAN.fit(df[['Latitude', 'Longitude']])
df['DBSCAN_labels']=dbSCAN.labels_

# displaying colored location on map
m = folium.Map(location=[points[0].latitude,points[0].longitude])
mapDots = folium.map.FeatureGroup()
for i in range(len(df)):
    mapDots.add_child(folium.CircleMarker(
        [df['Latitude'][i + 1], df['Longitude'][i + 1]],
        radius=4,
        tooltip=points[i].dateTime,
        color=colors[df['DBSCAN_labels'][i + 1]],
    ))

```

Từ đây ta có thể biết được những nhóm nào là nhóm vị trí có mật độ dày nhất (tức là người dùng thường xuyên đi lại nhất). Và có thể thống kê số điểm của mỗi cụm.

Ví dụ dưới đây thể hiện việc gom cụm các điểm GPS của người dùng số 70 trong tháng 9 năm 2008.



Hình 27: Phân cum của người dùng 70 trong tháng 7 năm 2007

Biểu đồ trên thể hiện những nơi người dùng hay đi qua (màu aqua) và những nơi người dùng 70 ít đi qua (màu maroon)

5.4 Trục quan hóa dữ liệu GPS của 1 người dùng theo thời gian

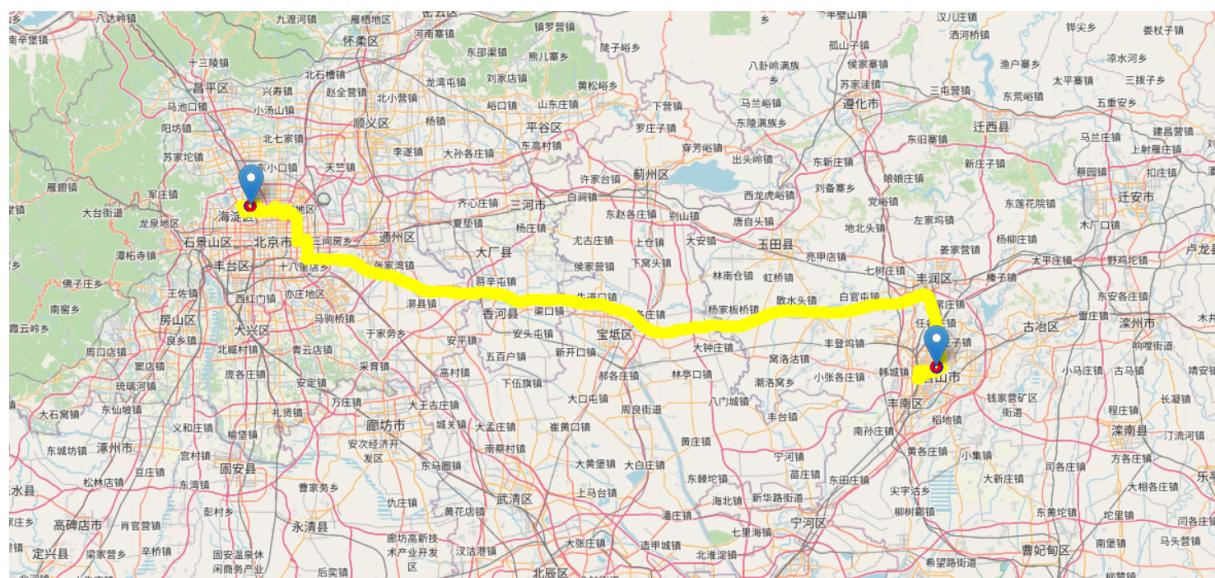
Như đã định nghĩa lại ở phần 4, dưới đây là hàm xác định các loại điểm của nhóm

```

1  def stayPointExtraction(points, distThres, timeThres):
2      stayPointList = []
3      stayPointCenterList = []
4      pointNum = len(points)
5      i = 0
6      while i < pointNum:
7          j = i + 1
8          while j < pointNum:
9              if getDistanceOfPoints(points[i], points[j]) > distThres:
10                  if getTimeIntervalOfPoints(points[i], points[j-1]) > timeThres:
11                      latitude, longitude = computMeanCoord(points[i:j])
12                      arriveTime = time.mktime(time.strptime(points[i].dateTime,
13                                              time_format))
14                      leaveTime = time.mktime(time.strptime(points[j-1].dateTime,
15                                              time_format))
16                      dateTIme = time.strftime(time_format, time.localtime(arriveTime)),
17                                              time.strftime(time_format, time.localtime(leaveTime)))
18                      stayPointCenterList.append(Point(latitude, longitude, dateTIme,
19                                              arriveTime, leaveTime))
20                      stayPointList.extend(points[i:j])
21                  break
22          j += 1
23      i = j
24  return stayPointCenterList, stayPointList

```

Kết quả với người dùng 70 trong tháng 9 năm 2008, với bán kính 200m, t_threshold = 3 giờ như sau:

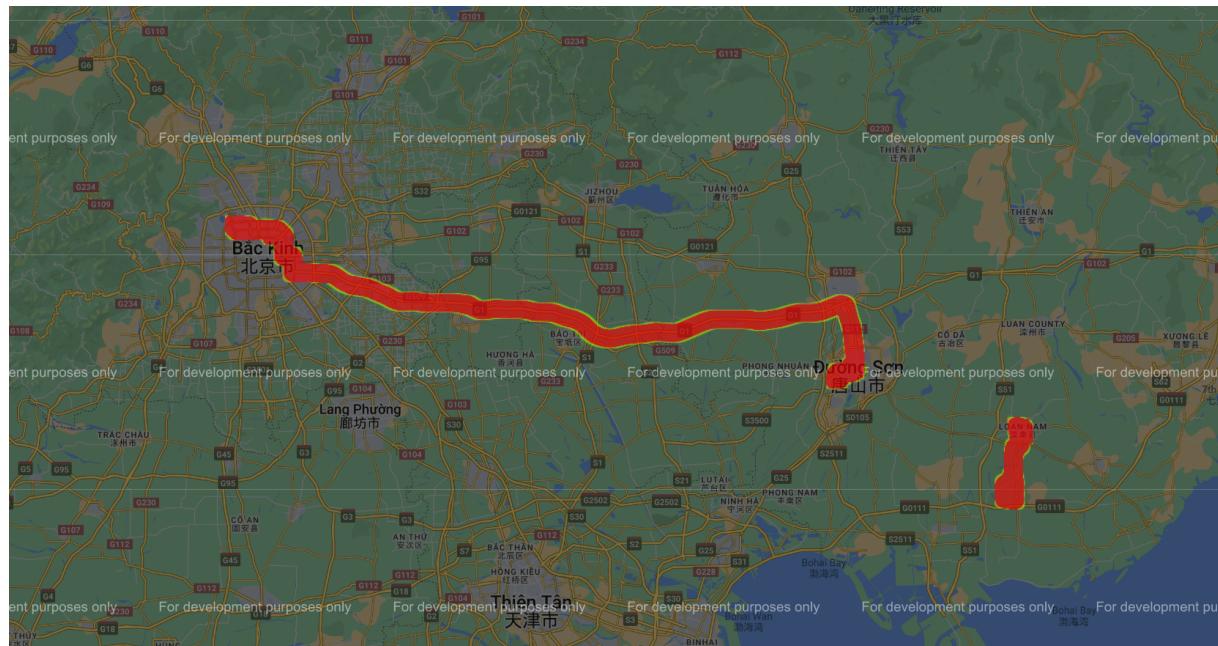


Hình 28: Dữ liệu vị trí của người dùng 70 trong tháng 9 năm 2008

- Điểm màu vàng: GPS của người dùng
- Điểm màu xanh: Các điểm stayPoint
- Điểm màu đỏ và marker: Điểm stayPointCenter

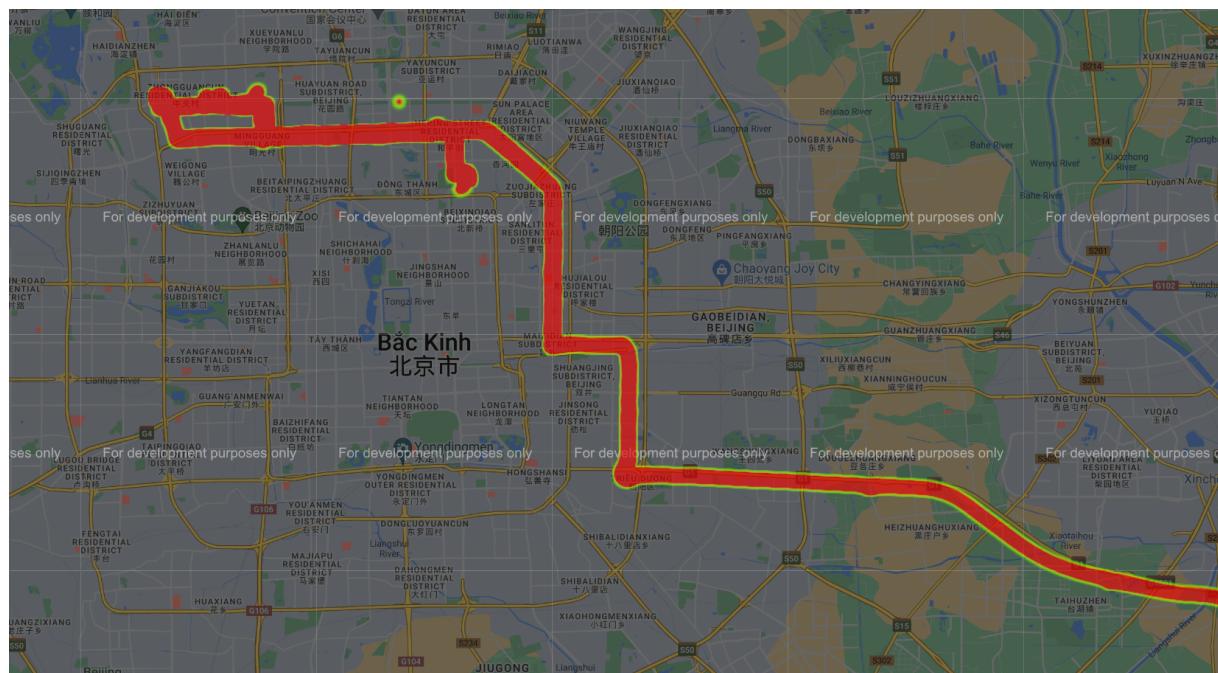
5.4.1 Trực quan hóa thông qua heatmap

Ở mục này, nhóm sử dụng thư viện gmplot của Python để visualize heatmap của người dùng. Ví dụ sau đây thể hiện heatmap của người dùng số 0 dựa theo GPS của họ.

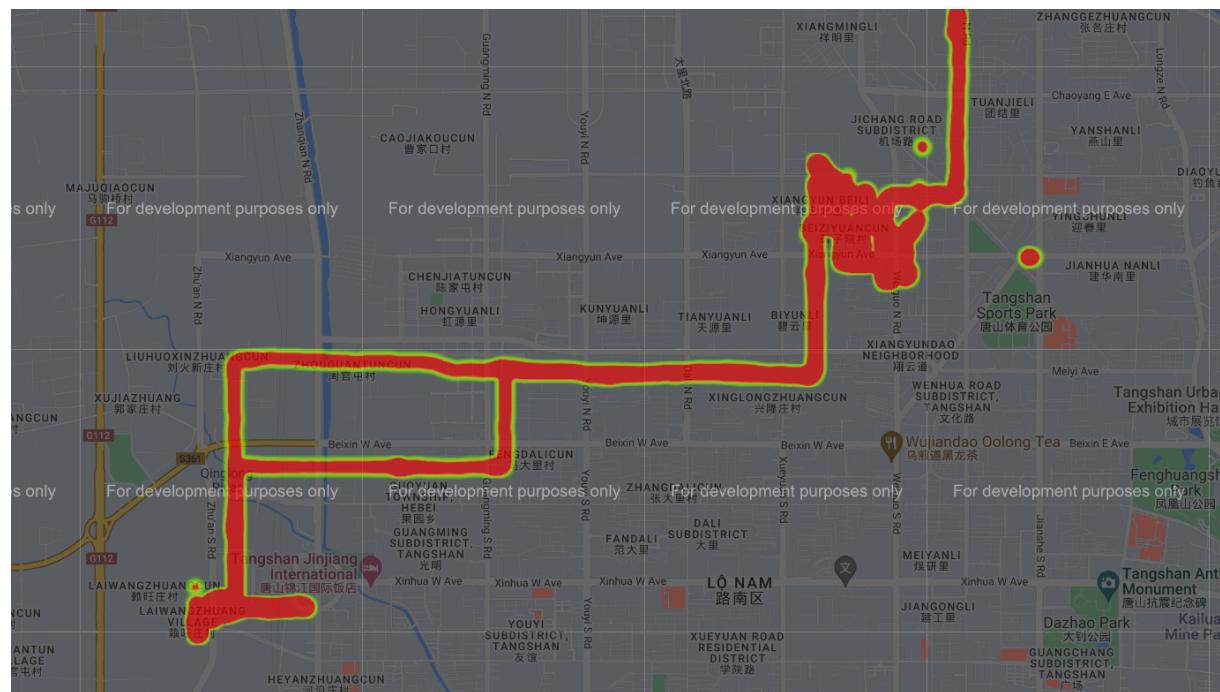


Hình 29: Quãng đường di chuyển của người dùng 70

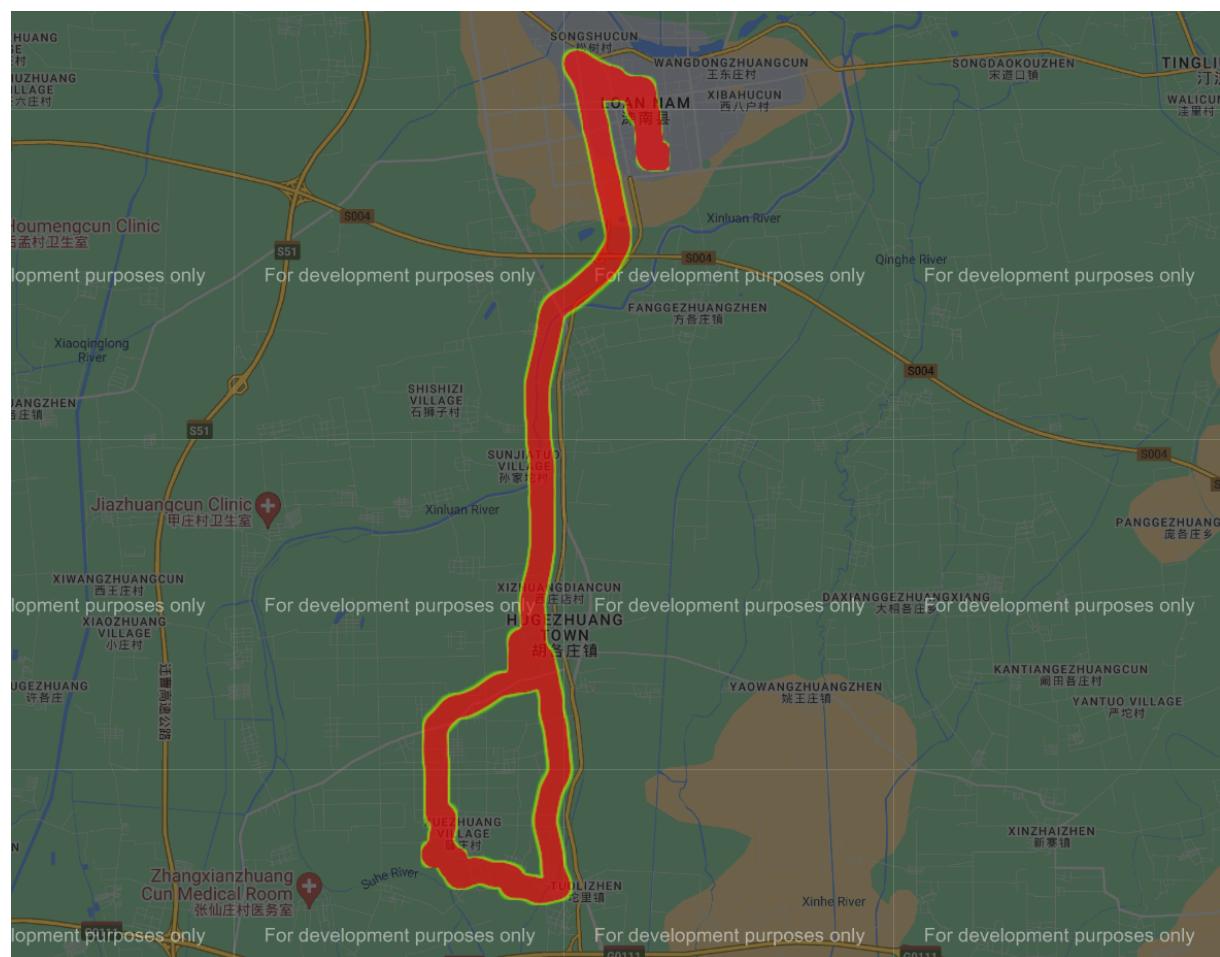
Dựa vào heatmap này có thể thấy người dùng 70 di chuyển từ Bắc Kinh đến Trường Sơn và ở lại Loan Nam



Hình 30: Heatmap của người dùng 70 tại Bắc Kinh



Hình 31: Heatmap của người dùng 70 tai Trường Sơn



Hình 32: Heatmap của người dùng 70 tai Loan Nam

5.5 Trên toàn bộ tệp dữ liệu - xác định các địa điểm được ghi nhận nhiều nhất

Mục tiêu nghiên cứu của vấn đề này là việc trích xuất xem địa điểm nào trên bản đồ có nhiều người đã đi qua nhất. Từ đó rút ra các kết luận về địa điểm tìm được.

Cách tiếp cận đơn giản nhất là đếm số lần xuất hiện của một tọa độ, xem những điểm nào trên bản đồ là có số lượt xuất hiện nhiều nhất được ghi nhận. Trong bài tập lớn này, nhóm chọn độ chia nhỏ nhất cho 1 khu vực là diện tích 10000m², tương đương hình vuông có chiều dài và rộng là 100m. Diện tích này tương đối bằng với diện tích của một toà nhà hoặc một trung tâm thương mại cỡ vừa. Sau khi xác định được số lần xuất hiện, nhóm thực hiện lọc ra khoảng 200 địa điểm có số lần xuất hiện nhiều nhất, sau đó sử dụng một API có tên là Nominatim để có thêm thông tin về điểm đó, khi truyền kinh độ và vĩ độ của điểm cần xem xét đến Nominatim, một file json sẽ trả về thông tin chi tiết của điểm đó. Ở đây ta quan tâm tới trường 'type' trong file này thể hiện loại (trường học, bệnh viện, điểm dừng xe bus,...) của địa điểm chúng ta đang xem xét. Sau đó trực quan hóa bằng PowerBI để ta có thể có cái nhìn tổng quát về phân bố của các điểm có nhiều lượt ghi nhận nhất.

Code hiện thực:

```
1 #Import library and read data
2 import pandas as pd
3 import numpy as np
4 col_list = ['User', "Longitude", "Latitude", "Date", "Time", "DateNumber"]
5 df = pd.read_csv("/content/drive/MyDrive/dataset/all_data.csv", usecols=col_list)
6
7 #Define API
8 from geopy.geocoders import Nominatim
9 from geopy.point import Point
10 def get_location_type(lat, lon):
11     geolocator = Nominatim(user_agent='test')
12     try:
13         location = geolocator.reverse(Point(lat, lon), timeout=None)
14         return location.raw['type']
15     except:
16         return None
17
18 #Drop Na and convert 'User' column to the same data type (float)
19 df = df.dropna()
20 df['User'] = df['User'].astype(float)
21 df = df.round(3)
22
23 #Location Frequency
24 MostFrequentLocation = (df.groupby(['Latitude',
25                                     'Longitude']).size().sort_values(ascending=False).reset_index(name='count'))
26 MostFrequentLocation = MostFrequentLocation[MostFrequentLocation['count'] > 10000]
27 #Get type of location from "nominatim" API
28 for i in range(len(MostFrequentLocation)):
29     MostFrequentLocation.loc[i, 'Location_type'] =
30         get_location_type(MostFrequentLocation.loc[i, 'Latitude'],
31                           MostFrequentLocation.loc[i, 'Longitude'])
32 MostFrequentLocation['Location_type'].unique()
33 MostFrequentLocation['Location_type'].nunique()
34 MostFrequentLocation.to_csv('/content/drive/MyDrive/dataset/MostFrequentLocation.csv')
```

	Latitude	Longitude	count
0	39.973	116.334	94096
1	40.008	116.319	88279
2	39.985	116.353	81809
3	39.975	116.331	72316
4	40.000	116.327	70939
...
673560	36.396	119.796	1
673561	36.396	119.798	1
673562	36.396	119.801	1
673563	36.396	119.802	1
673564	400.167	116.215	1

673565 rows × 3 columns

Hình 33: Dém số lần xuất hiện của các điểm

Ở đây nhóm trích xuất ra khoảng 200 địa điểm có số lần xuất hiện nhiều nhất và tìm loại địa điểm của điểm đó thông qua API Nominatim

	Latitude	Longitude	count	Location_type
0	39.973	116.334	94096	residential
1	40.008	116.319	88279	residential
2	39.985	116.353	81809	university
3	39.975	116.331	72316	yes
4	40.000	116.327	70939	unclassified
...
224	39.991	116.329	10112	cafe
225	39.975	116.303	10093	tertiary
226	39.995	116.326	10049	telecommunication
227	39.977	116.310	10028	mall
228	39.975	116.316	10020	apartments

229 rows × 4 columns

Hình 34: Thêm cột thể hiện loại địa điểm

```
MostFrequentLocation['Location_type'].unique()
```

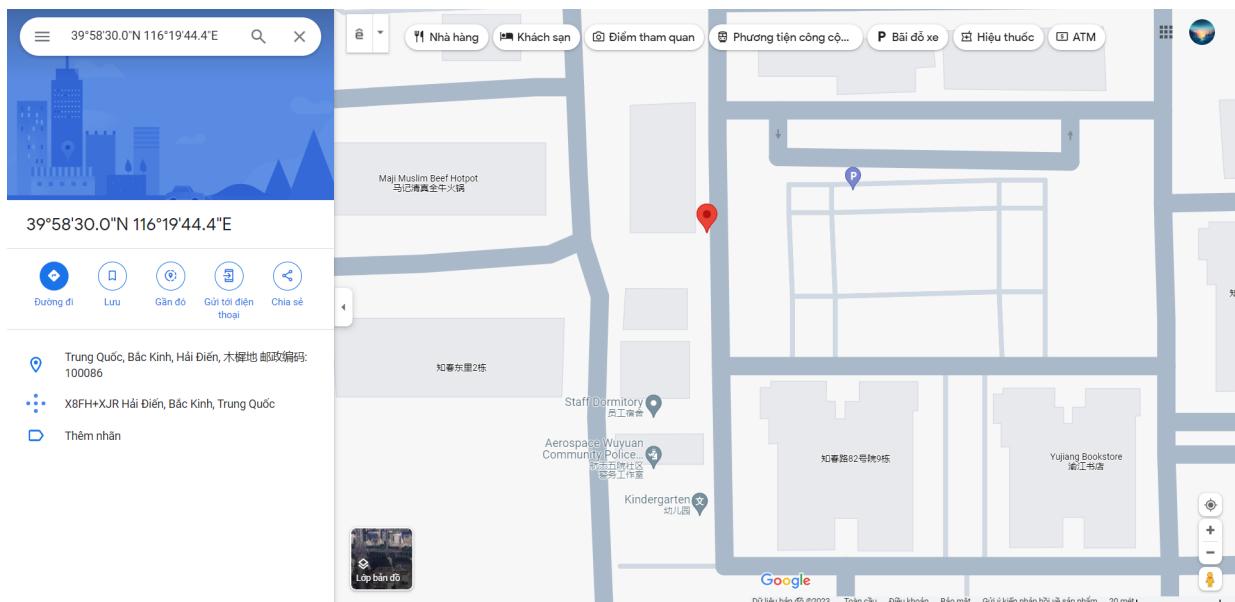
```
array(['residential', 'university', 'yes', 'unclassified', 'tertiary',
       'primary', 'house', 'it', 'kindergarten', 'parking', 'secondary',
       'massage', 'research_institute', 'school', 'hospital', 'police',
       'bureau_de_change', 'bicycle_parking', 'bakery', 'parcel_locker',
       'atm', 'government', 'cycleway', 'parking_entrance', 'bridge',
       'post_box', 'apartments', 'mall', 'hotel', 'research', 'pitch',
       'surveillance', 'bus_stop', 'cinema', 'cafe', 'artwork',
       'fast_food', 'vending_machine', 'stop', 'wastewater_plant',
       'commercial', 'restaurant', 'telecommunication'], dtype=object)
```

```
MostFrequentLocation['Location_type'].nunique()
```

43

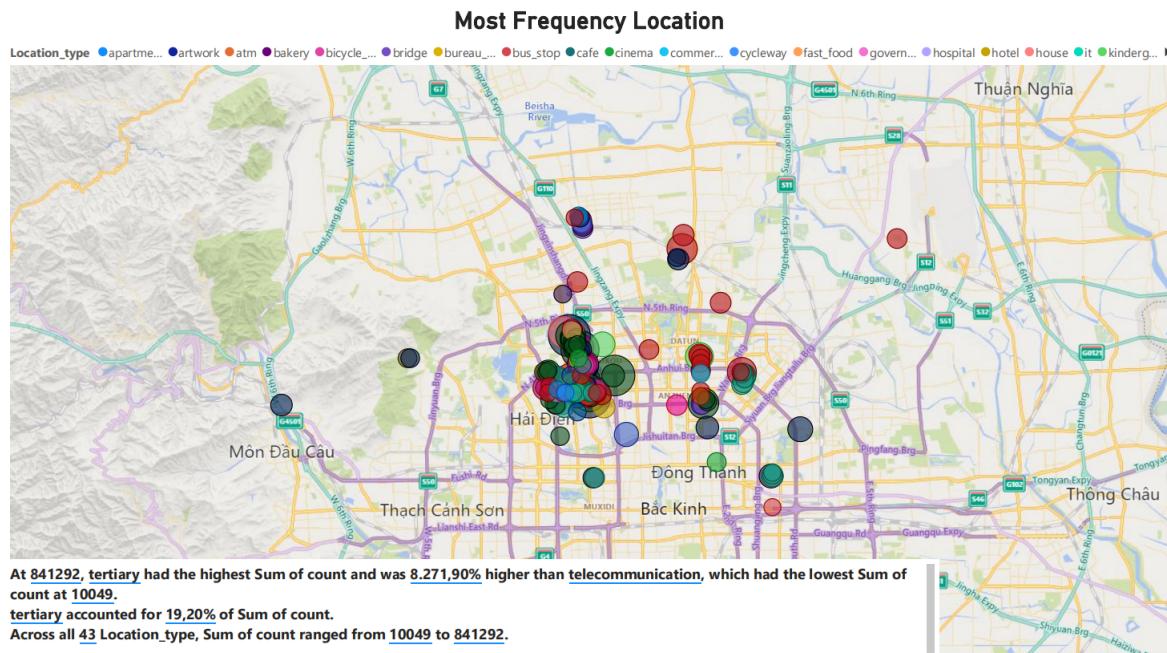
Hình 35: Danh sách các loại địa điểm xuất hiện nhiều nhất

Có thể thấy, các điểm xuất hiện nhiều quả thật là các điểm mà thường xuyên có người lui tới như trường học, khu dân cư, bệnh viện, tiệm bánh, cây ATM,... Một số điểm có loại là "yes" hoặc "primary", "secondary" nhóm có thử tìm kiếm trên Google Maps thì có thể là những con đường hoặc những điểm không được gán nhãn trên Google Maps



Hình 36: Một địa điểm có type là "yes"

Cuối cùng, khi trực quan hóa các điểm có số lần xuất hiện nhiều, ta có thể thấy, các điểm xuất hiện nhiều lần tập trung chủ yếu ở khu vực trung tâm thành phố Bắc Kinh, Trung Quốc với tần suất và mật độ khá lớn.



Hình 37: Các địa điểm có số lần xuất hiện nhiều nhất

5.6 Trên tập dữ liệu của mỗi người dùng - xác định các điểm được ghi nhận nhiều nhất

Tương tự như ý tưởng ở trên, khi xem xét từng người dùng, sở thích và thói quen đi lại của họ có thể dẫn đến mỗi người sẽ có cho mình những điểm đến ưa thích khác nhau, người dùng có thể đại diện cho nhiều nhóm đối tượng khác nhau, qua đó cũng có nhiều nhóm địa điểm khác nhau được thường xuyên chọn làm điểm đến. Ở bước này, ta sẽ thống kê xem đối với mỗi người dùng, đâu là điểm đến mà họ ưa thích nhất. Sau đó sử dụng API để tìm xem địa điểm ưa thích của họ là gì, cũng như xác định xem số lần địa điểm đó xuất hiện trong các ghi nhận của người dùng và tỉ lệ so với tổng số ghi nhận của mỗi người dùng.

Code hiện thực:

```

1 #Import library and read data
2 import pandas as pd
3 import numpy as np
4 col_list = ['User', "Longitude", "Latitude", "Date", "Time", "DateNumber"]
5 df = pd.read_csv("/content/drive/MyDrive/dataset/all_data.csv", usecols=col_list)
6
7 #Drop Na and convert 'User' column to the same data type (float)
8 df = df.dropna()
9 df['User'] = df['User'].astype(float)
10 df = df.round(3)
11
12 from geopy.geocoders import Nominatim
13 from geopy.point import Point
14 def get_location_type(lat, lon):
15     geolocator = Nominatim(user_agent='test')
16     try:
17         location = geolocator.reverse(Point(lat, lon), timeout=None)
18         return location.raw['type']
19     except:
20         return None
21
22
23 #LocationMostAppearOnUser

```



```
24 LocationAppearOnUser = df.groupby('User')[['Latitude', 'Longitude']].apply(lambda x:  
→ x.value_counts().index[0]).reset_index()  
25 LocationAppearOnUser = LocationAppearOnUser.rename(columns = {0 : 'Lat-Long'})  
26  
27 for i in range(len(LocationAppearOnUser)):  
28     LocationAppearOnUser.loc[i, 'Location_type'] =  
→         get_location_type(LocationAppearOnUser.loc[i, 'Lat-Long'][0],  
→         LocationAppearOnUser.loc[i, 'Lat-Long'][1])  
29  
30 LocationAppearOnUser['Location_type'].unique()  
31 LocationAppearOnUser['Location_type'].nunique()  
32  
33 def AppearanceFrequency(number):  
34     user = df[df['User'] == number]  
35     length = len(user)  
36     user = user.groupby(['Latitude',  
→     'Longitude']).size().sort_values(ascending=False).reset_index(name='count')  
37     count = user.loc[0, 'count']  
38     return (count, count/length)  
39  
40 for i in range(len(LocationAppearOnUser)):  
41     (a, b) = AppearanceFrequency(LocationAppearOnUser.loc[i, 'User'])  
42     LocationAppearOnUser.loc[i, 'Appearence'] = a  
43     LocationAppearOnUser.loc[i, 'Frequency'] = b  
44  
45 LocationAppearOnUser.to_csv('/content/drive/MyDrive/dataset/LocationAppearOnUser.csv')
```

Ở bước đầu tiên, ta sẽ tìm xem đâu là địa điểm mỗi người dùng thường tới nhất.

User	Lat-Long
0	0.0 (40.01, 116.315)
1	1.0 (40.014, 116.306)
2	2.0 (39.926, 116.337)
3	3.0 (40.008, 116.32)
4	4.0 (40.0, 116.327)
...	...
177	177.0 (39.938, 116.39)
178	178.0 (39.978, 116.332)
179	179.0 (40.008, 116.319)
180	180.0 (29.017, 115.807)
181	181.0 (39.99, 116.298)
182 rows × 2 columns	

Hình 38: Các địa điểm có số lần xuất hiện nhiều nhất của mỗi người

Kế đến ta sử dụng API để tìm loại địa điểm cho các điểm này



User	Lat-Long	Location_type
0	(40.01, 116.315)	house
1	(40.014, 116.306)	residential
2	(39.926, 116.337)	residential
3	(40.008, 116.32)	university
4	(40.0, 116.327)	unclassified
...
177	(39.938, 116.39)	bakery
178	(39.978, 116.332)	residential
179	(40.008, 116.319)	residential
180	(29.017, 115.807)	tertiary
181	(39.99, 116.298)	garden

182 rows × 3 columns

Hình 39: Loại địa điểm ưa thích nhất của mỗi người

```
LocationAppearOnUser['Location_type'].unique()
```

```
array(['house', 'residential', 'university', 'unclassified', 'parking',
       'kindergarten', 'pedestrian', 'research_institute', 'yes',
       'station', 'tertiary', 'primary', 'massage', 'school',
       'bicycle_parking', 'secondary', 'hospital', 'car', 'gift',
       'common', 'platform', 'hotel', 'attraction', 'research',
       'surveillance', 'trunk', 'cafe', 'post_box', 'garden', 'office',
       'motorway', 'bridge', 'administrative', 'parking_entrance',
       'construction', 'cycleway', 'books', 'playground', 'pitch',
       'restaurant', 'footway', 'living_street', 'theme_park', 'museum',
       'viewpoint', 'government', 'bakery'], dtype=object)
```

```
LocationAppearOnUser['Location_type'].nunique()
```

47

Hình 40: Những loại địa điểm ưa thích nhất của người dùng

Có thể thấy sự phân bố loại địa điểm ưa thích nhất của mỗi người dùng là tương đối lớn, tùy theo sở thích và thói quen của mỗi người. Ví dụ người nội trợ sẽ có xu hướng thích "house", sinh viên thì sẽ là "university", bác sĩ-y tá sẽ là "hospital",... Từ đây ta có thể có cái nhìn tổng quan về xu hướng chọn điểm đến của mỗi người trong khoảng thời gian được ghi nhận.

Kế đến, ta xem xét xem địa điểm ưa thích chiếm bao nhiêu % trong tất cả lô trình được ghi nhận của người dùng.

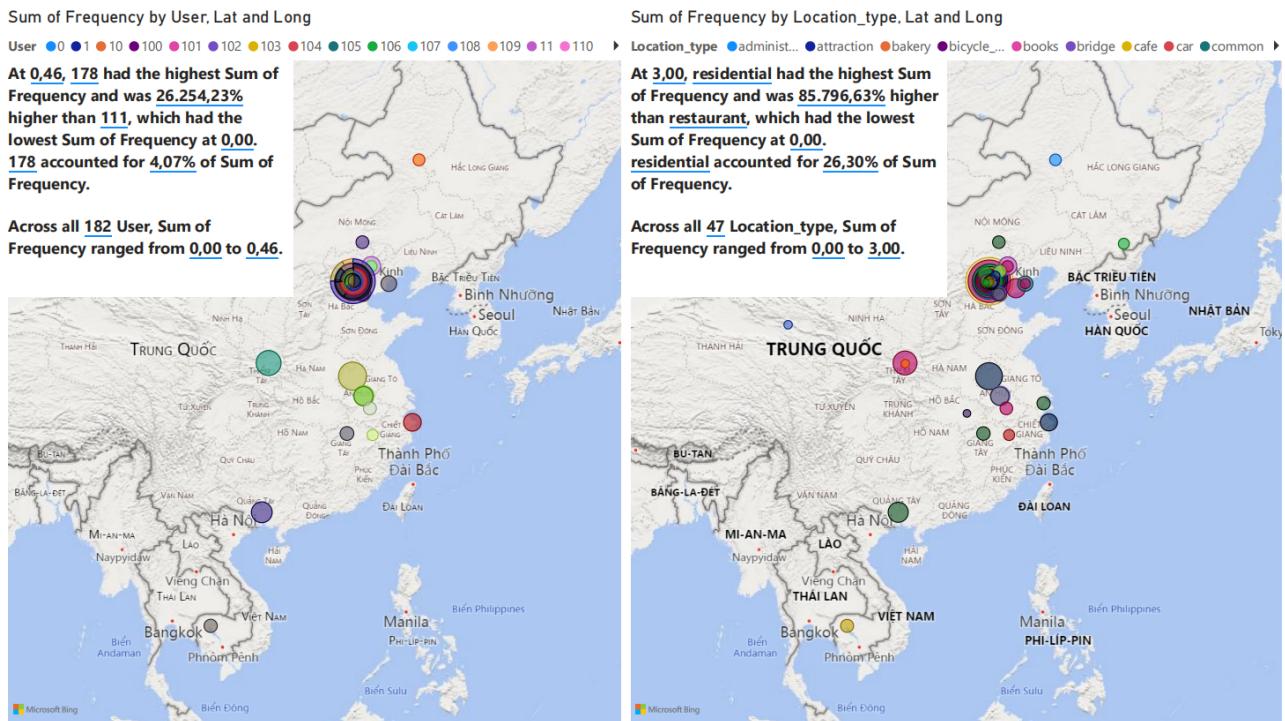
	User	Lat-Long	Location_type	Appearence	Frequency
0	0.0	(40.01, 116.315)	house	8595.0	0.049433
1	1.0	(40.014, 116.306)	residential	6947.0	0.063965
2	2.0	(39.926, 116.337)	residential	16156.0	0.065088
3	3.0	(40.008, 116.32)	university	21795.0	0.044917
4	4.0	(40.0, 116.327)	unclassified	16926.0	0.038521
...
177	177.0	(39.938, 116.39)	bakery	130.0	0.043993
178	178.0	(39.978, 116.332)	residential	39.0	0.464286
179	179.0	(40.008, 116.319)	residential	7832.0	0.046235
180	180.0	(29.017, 115.807)	tertiary	1543.0	0.032714
181	181.0	(39.99, 116.298)	garden	17.0	0.024745

182 rows × 5 columns

Hình 41: Tỉ lệ người dùng đến địa điểm ưa thích

Có thể thấy, đối với người dùng đi lại và được ghi nhận vị trí càng nhiều, tỉ lệ này xấp xỉ ở mức <10%, còn đối với những dữ liệu được ghi nhận ít, tỉ lệ này có thể lên đến 46.4% như user thứ 178.

Cuối cùng, trực quan hóa các địa điểm ưa thích của người dùng lên cùng một bản đồ, có thể thấy những người dùng ngoài tập trung ở khu vực Bắc Kinh còn có ở một số tỉnh khác của Trung Quốc, tuy nhiên hầu hết và với tần suất xuất hiện lớn, dữ liệu của người dùng vẫn được ghi nhận chủ yếu ở Bắc Kinh.



Hình 42: Trực quan hóa địa điểm ưa thích của từng người dùng

5.7 Trên toàn bộ tập dữ liệu - xác định thời gian nhiều users nhất cùng xuất hiện

Như đã trình bày ở phần mô tả tệp dữ liệu, bộ dữ liệu này được thu thập bởi 182 người dùng trong khoảng thời gian hơn 5 năm (từ tháng 4 năm 2007 đến tháng 8 năm 2012). Và sau khi đã khảo sát dữ liệu của từng user, nhóm nhận thấy thời gian xuất hiện GPS của các users là không trùng nhau, nghĩa là 182 users này không cùng lúc xuất hiện trên bản đồ. Từ đó nhóm đặt ra vấn đề là cần phải xác định ngày nào nhiều users nhất cùng xuất hiện và trực quan vị trí của các users đó lên bản đồ.

Khó khăn chính khi thực hiện vấn đề này là, trong tệp dữ liệu lớn, ngày xuất hiện nhiều nhất chưa hẳn đã là ngày có nhiều users nhất. Do nếu mỗi user xuất hiện rất nhiều lần trong 1 ngày thì ta vẫn đếm là 1 user hiện trong ngày.

Các bước tiến hành:

1. Nhóm các users lại thành các group dựa trên thuộc tính ngày để tìm ra những ngày có nhiều user được ghi nhận nhất
2. Chọn ra một số ngày để trực quan hóa địa điểm đầu tiên ghi nhận của từng user trong ngày đó
3. Dùng API để kiểm tra loại địa điểm xuất hiện đầu tiên của mỗi người dùng

Code thực thi:

```

1 #Import library and read data
2 import pandas as pd
3 import numpy as np
4 col_list = ['User', "Longitude", "Latitude", "Date", "Time", "DateNumber"]
5 df = pd.read_csv("/content/drive/MyDrive/dataset/all_data.csv", usecols=col_list)
6
7 #Drop Na and convert 'User' column to the same data type (float)
8 df = df.dropna()
9 df['User'] = df['User'].astype(float)
10 df = df.round(3)
11
12 from geopy.geocoders import Nominatim

```



```
13 from geopy.point import Point
14 def get_location_type(lat, lon):
15     geolocator = Nominatim(user_agent='test')
16     try:
17         location = geolocator.reverse(Point(lat, lon), timeout=None)
18         return location.raw['type']
19     except:
20         return None
21
22 UserAndDate=df.drop(['Latitude','Longitude', 'Time', 'Date'],axis=1)
23 UserAndDate['DateNumber'] = UserAndDate['DateNumber'].apply(np.floor)
24
25 UserDateGroupByDate =
26     ↪ UserAndDate.groupby('DateNumber')['User'].nunique().sort_values(ascending=False)
27 .reset_index(name='user_count')
28
29 from datetime import datetime, timedelta
30 base_date = datetime(1899, 12, 30)
31 for i in range(len(UserDateGroupByDate)):
32     UserDateGroupByDate.loc[i, 'Date'] = (base_date +
33     ↪ timedelta(days=UserDateGroupByDate.loc[i, 'DateNumber'])).date()
34
35 CrowdedDate = UserDateGroupByDate.loc[0, 'DateNumber']
36 LocationInCrowdedDate = df.loc[df['DateNumber'].apply(np.floor)==CrowdedDate]
37
38 rstLocation_OfEachUser_InCrowdedDate =
39     ↪ LocationInCrowdedDate.groupby('User').first().drop(['Date', 'Time',
40     ↪ 'DateNumber'],axis=1)
41
42 for i in FirstLocation_OfEachUser_InCrowdedDate.index:
43     FirstLocation_OfEachUser_InCrowdedDate.loc[i, 'Location_type'] =
44     ↪ get_location_type(FirstLocation_OfEachUser_InCrowdedDate.loc[i, 'Latitude'],
45     ↪ FirstLocation_OfEachUser_InCrowdedDate.loc[i, 'Longitude'])
46
47 FirstLocation_OfEachUser_InCrowdedDate['Location_type'].unique()
48 FirstLocation_OfEachUser_InCrowdedDate['Location_type'].nunique()
49
50 FirstLocation_OfEachUser_InCrowdedDate.to_csv('/content/drive/MyDrive/dataset/
51     ↪ FirstLocation_OfEachUser_InCrowdedDate1.csv')
```



	DateNumber	user_count	Date
0	39861.0	29	2009-02-17
1	39884.0	28	2009-03-12
2	39885.0	28	2009-03-13
3	39792.0	28	2008-12-10
4	39864.0	28	2009-02-20
...
1855	40498.0	1	2010-11-16
1856	40497.0	1	2010-11-15
1857	40475.0	1	2010-10-24
1858	40443.0	1	2010-09-22
1859	41117.0	1	2012-07-27

1860 rows × 3 columns

Hình 43: *Thống kê những ngày có số lượng người dùng được ghi nhận nhiều nhất*

Ở đây, nhóm chọn ra 2 ngày có số lượng người dùng ghi nhận nhiều nhất để trực quan hóa vị trí của những người dùng trong ngày đó, là ngày 17/02/2009 và 12/03/2009



User	Latitude	Longitude	Location_type
2.0	39.922	116.349	tertiary
3.0	40.011	116.315	house
4.0	39.990	116.332	bicycle_parking
13.0	39.975	116.363	designated
14.0	40.150	116.535	residential
17.0	40.010	116.346	primary
22.0	39.970	116.301	tertiary
24.0	39.991	116.331	bicycle_parking
25.0	39.987	116.302	university
28.0	39.975	116.325	it
29.0	39.798	116.517	primary
30.0	40.011	116.315	house
34.0	40.029	116.342	parking
35.0	39.982	116.322	swimming_pool
36.0	39.947	116.370	tertiary
37.0	39.977	116.252	bus_stop
38.0	39.968	116.302	university
39.0	39.975	116.363	designated
40.0	39.968	116.311	university
41.0	39.900	116.456	tertiary
42.0	39.909	116.453	hotel
44.0	39.975	116.334	primary
68.0	39.975	116.419	designated
126.0	40.003	116.331	secondary
128.0	40.077	116.327	residential
140.0	39.987	116.410	tertiary
144.0	40.043	116.427	tertiary
153.0	39.929	116.472	primary
167.0	40.003	116.331	secondary

Hình 44: Những người dùng xuất hiện trong ngày 17/02/2009 và địa điểm đầu tiên họ xuất hiện

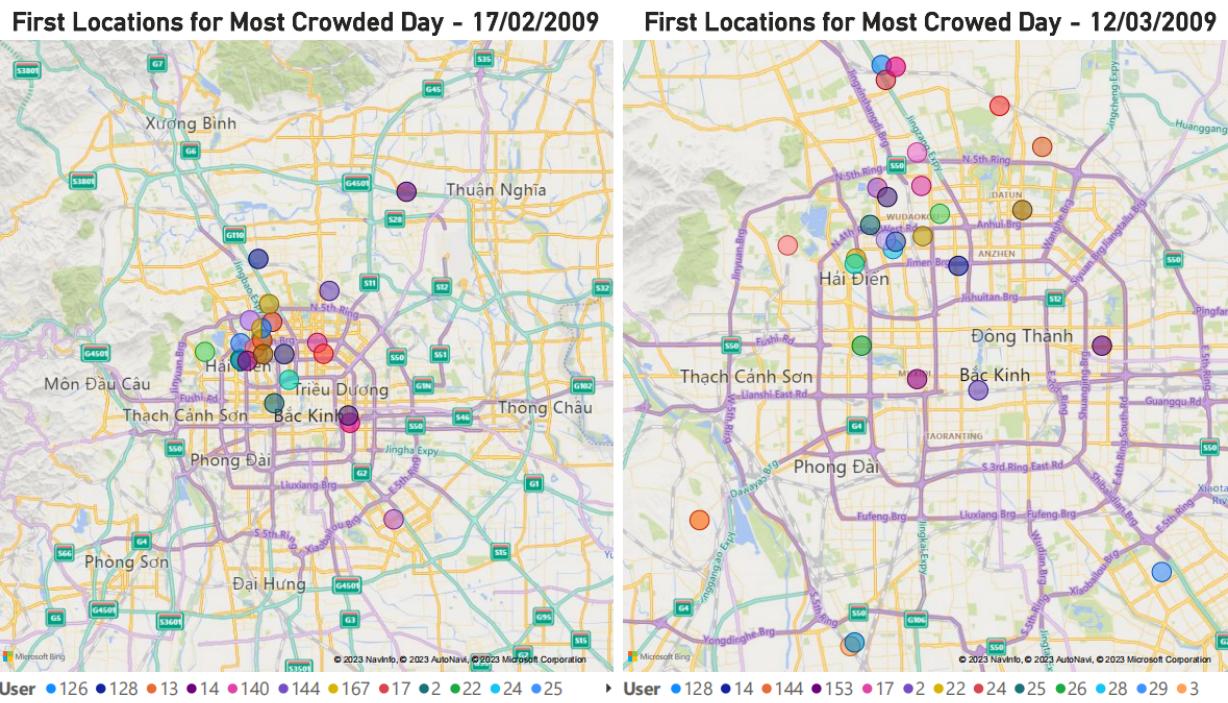
```
FirstLocation_OfEachUser_InCrowdedDate['Location_type'].unique()
```

```
array(['tertiary', 'house', 'bicycle_parking', 'designated',
       'residential', 'primary', 'university', 'it', 'parking',
       'swimming_pool', 'bus_stop', 'hotel', 'secondary'], dtype=object)
```

```
FirstLocation_OfEachUser_InCrowdedDate['Location_type'].nunique()
```

Hình 45: Những loại địa điểm xuất hiện đầu tiên của mỗi người dùng trong ngày 17/02/2009

Cuối cùng, trực quan hóa sự xuất hiện của người dùng qua hai ngày. Có thể thấy phần lớn người dùng trong khu vực này tập trung ở phía bắc, tây bắc của trung tâm thành phố Bắc Kinh vào những ngày này. Điều này có thể giúp nhiều cho việc phân phối nhân lực, nguồn lực phục vụ cho các dịch vụ, giải trí vào những ngày cao điểm đối với các doanh nghiệp. Đồng thời, mạng lưới giao thông cũng có thể được phân phối sao cho phù hợp hơn giúp việc di lại trong trung tâm thành phố trở nên thuận tiện.



Hình 46: Hai ngày có nhiều người dùng ghi nhận nhất

5.8 Trên toàn bộ tập dữ liệu - xác định vị trí nhiều user đi qua nhất trên bản đồ

Mục tiêu nghiên cứu của vấn đề này là việc trích xuất xem địa điểm nào trên bản đồ có nhiều người đã từ qua nhất. Từ đó rút ra các kết luận về địa điểm tìm được.

Khó khăn chính khi thực hiện vấn đề này là, trong tập dữ liệu lớn, vị trí xuất hiện nhiều nhất chưa hẳn đã là vị trí có nhiều users đi qua nhất. Do nếu mỗi user đi qua 1 điểm rất nhiều lần thì ta vẫn đếm là 1 user đã đi qua điểm đó. Đồng thời, vị trí GPS có sai số, và không nhất thiết khảo sát địa điểm mà users đứng trùng lấp lên nhau, ta nên tiếp cận 1 vùng hoặc 1 diện tích khu vực và user đã đến.

Các bước tiến hành như sau:

- Thống kê những địa điểm có số lượng user đi qua là nhiều nhất
- Sử dụng API để kiểm tra loại địa điểm của những địa điểm ghi nhận nhiều user đi qua nhất
- Trực quan hóa những điểm này thông qua PowerBI

```

1 #Import library and read data
2 import pandas as pd
3 import numpy as np
4 col_list = ['User', "Longitude", "Latitude", "Date", "Time", "DateNumber"]
5 df = pd.read_csv("/content/drive/MyDrive/dataset/all_data.csv", usecols=col_list)
6
7 #Drop Na and convert 'User' column to the same data type (float)
8 df = df.dropna()
9 df['User'] = df['User'].astype(float)
10 df = df.round(3)
11

```

```
12 from geopy.geocoders import Nominatim
13 from geopy.point import Point
14 def get_location_type(lat, lon):
15     geolocator = Nominatim(user_agent='test')
16     try:
17         location = geolocator.reverse(Point(lat, lon), timeout=None)
18         return location.raw['type']
19     except:
20         return None
21 AlluserLocation = df.drop(['Date', 'Time', 'DateNumber'], axis=1)
22 AlluserLocation = AlluserLocation.drop_duplicates()
23 MostappearlocationCount =
24     ↪ AlluserLocation.groupby(['Latitude', 'Longitude'])['User'].nunique().sort_values(ascending=False).re
25 MostappearlocationCount = MostappearlocationCount[MostappearlocationCount['user_count'] >
26     ↪ 70]
27 for i in range(len(MostappearlocationCount)):
28     MostappearlocationCount.loc[i, 'Location_type'] =
29         ↪ get_location_type(MostappearlocationCount.loc[i, 'Latitude'],
30         ↪ MostappearlocationCount.loc[i, 'Longitude'])
31 MostappearlocationCount['Location_type'].unique()
32 MostappearlocationCount['Location_type'].nunique()
```

Nhóm dữ liệu lại theo các địa điểm, ta có thống kê về những địa điểm ghi nhận nhiều người đi qua

	Latitude	Longitude	user_count
0	39.975	116.331	144
1	39.975	116.334	137
2	39.975	116.332	137
3	39.975	116.330	136
4	39.975	116.329	133
...
673560	34.774	113.569	1
673561	34.774	113.564	1
673562	34.774	113.552	1
673563	34.774	105.642	1
673564	400.167	116.215	1

673565 rows × 3 columns

Hình 47: Những địa điểm có số lượng user đi qua nhiều nhất

	Latitude	Longitude	user_count	Location_type
0	39.975	116.331	144	yes
1	39.975	116.334	137	primary
2	39.975	116.332	137	tertiary
3	39.975	116.330	136	yes
4	39.975	116.329	133	yes
...
170	39.986	116.361	71	trunk
171	39.985	116.343	71	tertiary
172	39.992	116.328	71	mall
173	39.974	116.328	71	yes
174	39.986	116.368	71	trunk

175 rows × 4 columns

Hình 48: Loại địa điểm của những điểm có số lượng user đi qua nhiều nhất

```
MostappearlocationCount['Location_type'].unique()
```

```
array(['yes', 'primary', 'tertiary', 'it', 'government', 'residential',
       'hospital', 'secondary', 'police', 'trunk', 'research_institute',
       'unclassified', 'pitch', 'bicycle_parking', 'electronics',
       'commercial', 'apartments', 'post_box', 'bridge', 'school',
       'cycleway', 'surveillance', 'university', 'parking', 'pedestrian',
       'bank', 'fast_food', 'cafe', 'restaurant', 'attraction',
       'bus_stop', 'mall'], dtype=object)
```

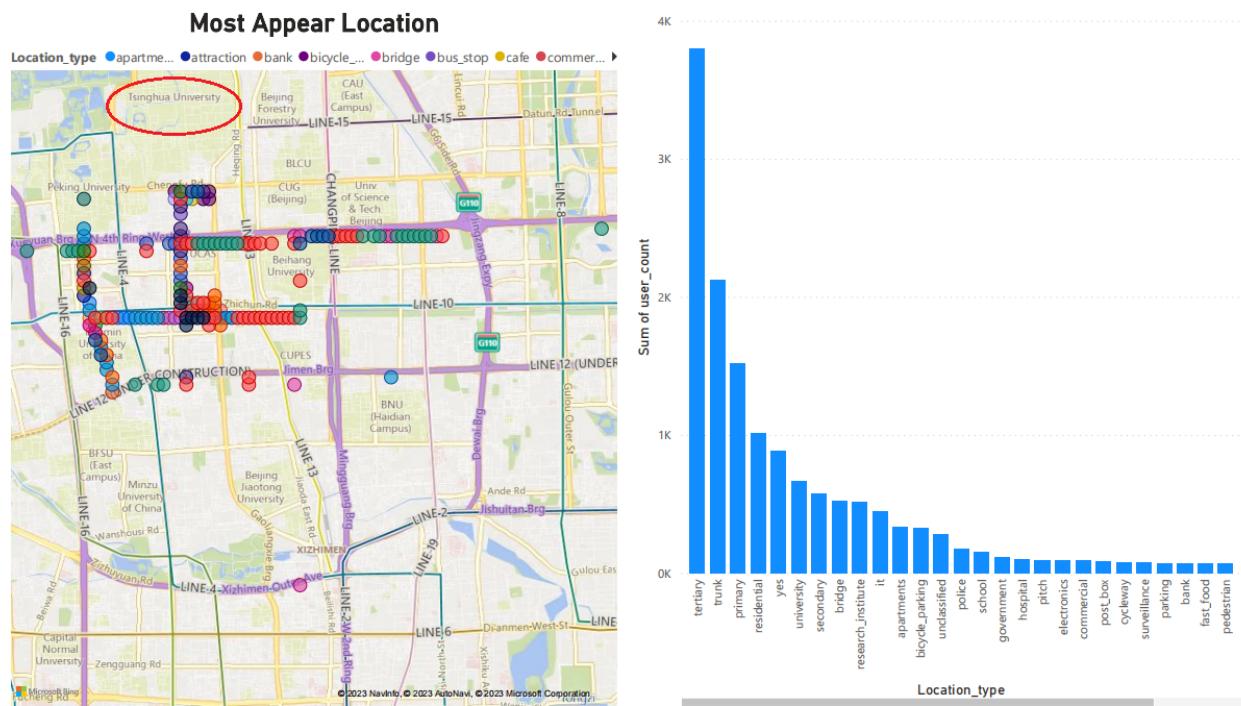
```
MostappearlocationCount['Location_type'].nunique()
```

32

Hình 49: Những loại địa điểm xuất hiện

Có thể thấy loại địa điểm xuất hiện nhiều user nhất lại có "type" là "yes" và "primary" có thể thấy đây có thể là những đoạn đường tấp nập thường xảy ra kẹt xe hoặc một địa điểm thu hút nào đó (quán ăn, nhà hàng, địa điểm du lịch-sống ảo,...)

Trực quan hóa các điểm này lên, có thể thấy khu vực tập trung nhiều user đặc biệt là ở khu vực tây bắc của trung tâm thành phố Bắc Kinh, gần đại học Thanh Hoa (khoanh tròn màu đỏ), điều này dễ hiểu vì đây là khu vực trung tâm thủ đô của Trung Quốc, cũng như tọa lạc trường đại học số 1 châu Á, chính vì vậy lượng người qua lại gồm sinh viên, khách du lịch,... ở khu vực này nhiều là điều đương đồi dẽ hiểu. So sánh với hình ảnh trực quan hóa ở phần 5.5, ta cũng thấy có nhiều nét tương đồng. Có thể thấy 2 cách tiếp cận cho ra kết quả về các điểm có nhiều người chọn làm điểm đến là tương đối giống nhau



Hình 50: Trực quan hóa các địa điểm ghi nhận nhiều user

5.9 Các loại hình giao thông hay sử dụng

5.9.1 Thời gian sử dụng phương tiện của mỗi user

Chuyển dữ liệu từ file txt sang file csv.

```

1     pathFile = []
2 def isExistFileLabels():
3     for x in os.listdir('F:/CaoHoc/Data engineer/Geolife Trajectories 1.3/Geolife
4         Trajectories 1.3/Data'):
4     for y in os.listdir('F:/CaoHoc/Data engineer/Geolife Trajectories 1.3/Geolife
5         Trajectories 1.3/Data/'+x):
5     if(y.endswith('.txt')):
6         pathFile.append('F:/CaoHoc/Data engineer/Geolife Trajectories 1.3/Geolife
7             Trajectories 1.3/Data/'+x)
8
8 def getTimeIntervalOfPoints(pi, pj):
9     t_i = time.mktime(time.strptime(pi, time_format))
10    t_j = time.mktime(time.strptime(pj, time_format))
11    return t_j - t_i
12 isExistFileLabels()
13 def convertToCsvFile():
14     for i in range(len(pathFile)):
15         f = pd.read_csv(pathFile[i]+'/labels.txt', delimiter = '\t')
16         f.to_csv(pathFile[i]+'/labels.csv', index=None)
17 totalResult = {}
18 def calculateDuration():
19     for j in range(len(pathFile)):
20         dataLabels = pd.read_csv(pathFile[j]+'/labels.csv')
21         start = dataLabels.iloc[:,0].values.tolist()
22         end = dataLabels.iloc[:,1].values.tolist()
23         vehicle = dataLabels.iloc[:,2].values.tolist()
24         result = {}
25         for i in range(len(vehicle)):

```

```

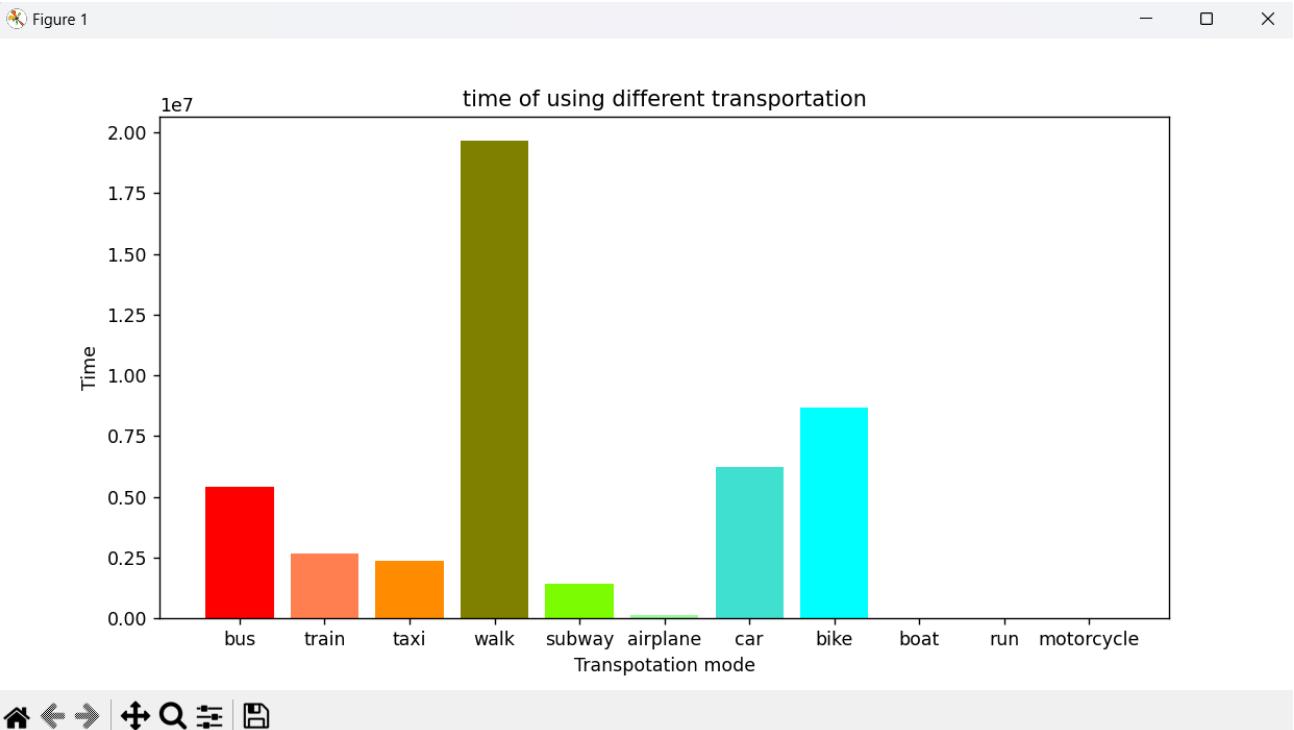
26         result[vehicle[i]] = 0;
27     for i in range(len(start)):
28         duration = getTimeIntervalOfPoints(start[i],end[i])
29         result[vehicle[i]] += duration
30     totalResult[pathFile[j]] = result
31 convertToCsvFile()
32 calculateDuration()
33 table = pd.DataFrame.from_dict(totalResult)
34 table = table.fillna(0)
35 table["total"] = table.sum(axis=1)
36 table = table.transpose()
37 color =
38     → ('red','coral','darkorange','olive','lawngreen','palegreen','turquoise','cyan','lavender','indigo',
39 table.to_csv('result.csv',index = None)
40 plt.figure(figsize=(10,5))
41 plt.bar(table.columns.values,table.iloc[-1],color = color)
42 plt.title('time of using different transportation')
43 plt.xlabel('Transportation mode')
44 plt.ylabel('Time')
45 plt.show()

```

User	bus	train	taxi	walk	subway	airplane	car	bike	boat	run	motorcycle
10	50605	1014012	85070	54213	32105	15259	2836	0	0	0	0
20	40399	4931	235	202230	39718	0	7729	135952	0	0	0
21	0	0	1161	56353	0	0	77317	0	0	0	0
52	444049	0	35482	788094	66025	23284	0	17266	0	0	0
53	2983	0	0	30096	0	0	8678	0	0	0	0
56	0	0	4953	22758	0	0	0	141459	0	0	0
58	7158	0	6082	9866	0	0	20511	0	0	0	0
59	0	0	0	13945	0	0	0	0	0	0	0
60	0	0	0	2890	0	0	0	0	0	0	0
62	1191405	110749	16360	390309	4821	0	61397	127327	1358	19	0
64	34333	0	0	48614	0	0	0	20404	0	580	0
65	30972	0	21686	242457	55732	0	5520	209754	0	0	0
67	20108	11121	0	271276	60779	0	568	0	0	0	0
68	655180	44186	6583	241765	15943	0	33753	839100	0	0	0
69	12670	0	0	57434	0	0	7224	10643	0	0	0
73	2313	0	0	43544	0	0	0	0	0	0	0
75	79560	1014012	85301	141754	84027	15259	2836	18826	0	580	0
76	0	0	0	0	0	0	525938	0	0	0	0
78	5516	0	8383	73331	23092	0	42815	0	0	0	0
80	42342	423	4177	11065	0	0	0	2110	0	0	0
81	56270	0	0	43581	7232	0	840	32244	0	0	0
82	789	0	5698	108050	104466	0	12430	16540	0	0	0
84	266963	3186	20002	656880	70551	0	101589	0	3854	630	0
85	702193	920	11997	861174	331127	0	54049	2846	0	0	0
86	0	0	0	3868	0	0	102682	0	0	0	0
87	0	0	0	27697	0	0	0	0	0	0	0
88	480	0	0	0	0	0	0	0	0	0	0
89	0	0	0	14210	0	0	238579	0	0	0	0
91	28123	1762	2868	133606	21482	0	0	2214	0	0	0

Hình 51: Thời gian sử dụng phương tiện của mỗi user

Nhóm sử dụng thư viện matplotlib để vẽ biểu đồ dạng cột cho tổng thời gian sử dụng 1 loại phương tiện của tất cả người dùng.



Hình 52: Thời gian sử dụng phương tiện của tất cả user

5.9.2 Nhận xét

Theo kết quả phân tích cho thấy, Motorcycle ít sử dụng nhất, người dùng dành nhiều thời gian nhất cho việc đi bộ. Thời gian sử dụng các phương tiện là khác nhau, trong đó boat, run và motorcycle gần như bằng 0 so với walk.

6 Kết luận

6.1 Kết quả

Có thể thấy, với nhiều hướng tiếp cận khác nhau ta đã có được những hiểu biết nhất định về thói quen và sở thích di chuyển của những người dùng trong tập dữ liệu từ đề bài. Hầu hết người dùng được ghi nhận tập trung ở khu vực trung tâm của thành phố Bắc Kinh, cụ thể hơn là phía Bắc và Tây Bắc, gần Đại học Thanh Hoa. Bên cạnh đó, do sự phân bố người dùng giữa các khu vực của Trung Quốc, một số người dùng được ghi nhận di lại nhiều ở các tỉnh khác như Quảng Tây, Thẩm Tây hay Hà Nam của Trung Quốc. Nếu đi chi tiết hơn, ta có thể sử dụng phân loại địa điểm để có thể có thêm nhiều hiểu biết hơn về thói quen và sở thích của người dùng, qua đó giúp đưa ra những chiến lược tối ưu để phát triển kinh tế - xã hội từ phía doanh nghiệp, nhà cung cấp dịch vụ và cả phía chính quyền

6.2 Khó khăn

6.2.1 Độ lớn của dữ liệu

Chúng ta đã biết, các dữ liệu đều có khả năng có ý nghĩa, do đó việc xem xét toàn bộ bộ dữ liệu là một điều cần thiết.

Như đã miêu tả trong phần mô tả dữ liệu, bộ dữ liệu có độ lớn là 1.8GB do đó việc xử lý bộ dữ liệu là không hề dễ dàng. Các thành viên của nhóm do giới hạn tài nguyên về phần cứng do đó cũng gặp rất nhiều khó khăn trong quá trình xử lý cũng như phân tích.

Bên cạnh đó, bộ dữ liệu cũng được chia ra thành các file nhỏ chứa trong nhiều thư mục khác nhau và không cùng cấp với nhau. Do đó việc đọc và tiền xử lý để lưu trữ về một hệ quản trị cơ sở dữ liệu cũng là một vấn đề cần được giải quyết.



6.2.2 Lựa chọn công cụ

Việc chọn công cụ lưu trữ và hỗ trợ cho bài toán dữ liệu là một thách thức lớn đối với các thành viên. Một số điểm mà nhóm lưu ý trong việc lựa chọn công cụ như sau:

- Quản lý dữ liệu: Công cụ lưu trữ phải hỗ trợ quản lý dữ liệu hiệu quả và cho phép truy xuất dữ liệu nhanh chóng và chính xác.
- Tính mở rộng: Với dữ liệu lớn, tính mở rộng là yếu tố quan trọng. Đặc điểm của bài toán là lượng dữ liệu GPS có thể gửi về liên tục do đó vấn đề mở rộng sẽ rất được lưu tâm.
- Tốc độ truy xuất: Công cụ lưu trữ phải hỗ trợ truy xuất dữ liệu nhanh chóng để đảm bảo hiệu suất của ứng dụng.
- Chi phí: Với dữ liệu lớn, chi phí là một yếu tố quan trọng. Công cụ lưu trữ phải có giá cả phù hợp và đảm bảo hiệu quả kinh tế.

Bên cạnh đó, phần lớn các thành viên trong nhóm cũng chưa có nhiều cơ hội được tiếp xúc với các công cụ hữu ích có thể giúp giải quyết bài toán, chính vì vậy việc làm quen sử dụng các công cụ hỗ trợ cũng là khó khăn nhất định trong quá trình thực hiện bài tập lớn.

6.2.3 Cách tiếp cận bài toán

Bản thân các thành viên trong nhóm cũng chưa có nhiều cơ hội được tiếp xúc với lượng dữ liệu lớn và phức tạp như vậy, cộng thêm với việc do sự phức tạp và đa dạng của dữ liệu dẫn đến tìm một hướng giải quyết cho bài toán là tương đối mất thời gian và cần cân nhắc kỹ.

Bên cạnh đó, do lượng dữ liệu quá lớn đòi hỏi các công cụ xử lý cần thời gian để thực hiện nên việc quyết định hướng tiếp cận cho bài toán cũng tương đối mất thời gian để thử nghiệm các cách khác nhau.

6.3 Ý nghĩa của đề tài

Bài toán phân tích vị trí GPS của người dùng là một bài toán quan trọng trong lĩnh vực xử lý dữ liệu và phân tích dữ liệu, có vai trò quan trọng trong việc đưa ra những hiểu biết về thói quen và sở thích của người dùng, qua đó:

- Cải thiện trải nghiệm người dùng: Bằng cách phân tích vị trí GPS của người dùng, các ứng dụng và dịch vụ có thể cung cấp các thông tin, dịch vụ và trải nghiệm phù hợp với vị trí của người dùng. Ví dụ: gợi ý địa điểm du lịch, khuyến mãi đặc biệt tại các cửa hàng và nhà hàng, cập nhật thông tin giao thông.
- Quản lý tài nguyên hiệu quả: Bằng cách phân tích vị trí GPS của người dùng, các công ty có thể quản lý tài nguyên của họ hiệu quả hơn. Ví dụ: phân bổ nhân lực theo khu vực khách hàng, lập kế hoạch điều phối xe giao hàng, quản lý lộ trình chuyến đi của tài xế.
- Nghiên cứu và phát triển sản phẩm mới: Bằng cách phân tích vị trí GPS của người dùng, các công ty có thể nghiên cứu và phát triển các sản phẩm và dịch vụ mới phù hợp với nhu cầu và thị hiếu của người dùng.
- Phân tích hành vi người dùng: Bằng cách phân tích vị trí GPS của người dùng, các công ty có thể phân tích hành vi và thói quen của người dùng, từ đó cung cấp các sản phẩm và dịch vụ phù hợp hơn, tăng khả năng tiếp cận khách hàng và cải thiện doanh số.
- An toàn và bảo mật: Dữ liệu GPS có thể được sử dụng để theo dõi và quản lý an toàn của người dùng, đặc biệt là trong lĩnh vực như an ninh giao thông hoặc du lịch, bên cạnh đó là cung cấp khả năng theo dõi và định vị thiết bị di động để bảo vệ khỏi mất mát hoặc trộm cắp.
- Dánh giá tác động của sự kiện địa lý: Phân tích cách mà sự kiện như hội chợ, concert, hay thậm chí thảm họa tự nhiên có thể ảnh hưởng đến di chuyển và tương tác của người dùng.
- Dự đoán và dự báo:
 - Dự đoán nhu cầu: Dữ liệu GPS có thể được sử dụng để dự đoán nhu cầu tiêu thụ một cách chính xác hơn, giúp doanh nghiệp chuẩn bị tốt hơn.



- Dự báo tình trạng giao thông: Phân tích dữ liệu GPS có thể giúp dự đoán và tránh tình trạng kẹt xe, cung cấp tuyến đường thay thế và giảm thời gian di chuyển.

Tổng quan, bài toán phân tích người dùng thông qua vị trí GPS của họ có tính cần thiết cao, nếu xử lý đưa ra những insight tốt về dữ liệu sẽ đem lại những lợi ích rất lớn về mặt kinh tế, đặc biệt là trong việc phân tích nhu cầu thói quen sinh hoạt và đi lại của thị trường, qua đó đưa ra chiến lược tối ưu để nâng cao chất lượng dịch vụ cho người dùng đồng thời mang lại hiệu quả cao về mặt kinh tế và xã hội.

Cuối cùng, nhóm xin cảm ơn giảng viên môn học - TS. Phan Trọng Nhân đã hỗ trợ nhóm hoàn thành bài tập lớn lần này. Qua bài tập lớn lần này, các thành viên trong nhóm đã có thêm kinh nghiệm và kiến thức trong việc lưu trữ và xử lý dữ liệu lớn bằng các công nghệ hỗ trợ như Hadoop, Spark, MySQL. Bên cạnh đó nhóm đã phân tích dữ liệu được cho dễ hiểu rõ hơn và có những nhận xét nhất định về lộ trình di chuyển của người dùng và ý nghĩa của nó.