KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN TIN HỌC ĐỊA CHẤT

========

BÁO CÁO SINH HOẠT HỌC THUẬT

BẢN ĐỒ HÓA CHO KHOA HỌC DỮ LIỆU KHÔNG GIAN ĐỊA LÝ BẰNG PYTHON

Người thực hiện: Nguyễn Thị Hải Yến

Hà Nội, tháng 12 năm 2021

MỤC LỤC

MỞ ĐẦU

Có rất nhiều gói Python để phân tích không gian địa lý, chẳng hạn như geopandas để phân tích dữ liệu vectơ và xarray để phân tích dữ liệu raster. Tuy nhiên, một số gói Python cung cấp GUI tương tác để tải dữ liệu không gian địa lý trong môi trường Jupyter. Có thể mất nhiều dòng để viết mã để tải và hiển thị dữ liệu không gian địa lý với các định dạng tệp khác nhau trên một bản đồ tương tác, đây có thể là một nhiệm vụ khó khăn đối với người dùng mới làm quen với kỹ năng viết mã hạn chế. Ngoài ra còn có một số gói Python đáng chú ý để trực quan hóa dữ liệu không gian địa lý trong môi trường Jupyter, chẳng hạn như plotly và kepler.gl. Tuy nhiên, cốt truyện được thiết kế để hiển thị dữ liệu tĩnh, thiếu giao tiếp hai chiều giữa front-end và backend. Kepler.gl cung cấp chức năng 3D độc đáo để hiển thị tập dữ liệu không gian địa lý quy mô lớn, nhưng nó thiếu các công cụ để thực hiện phân tích không gian địa lý, chẳng hạn như phân tích thủy văn và phân tích dữ liệu LiDAR. Ngược lại, leafmap cung cấp nhiều chức năng thuận tiện để tải và hiển thị trực quan các tập dữ liệu không gian địa lý chỉ với một dòng mã. Người dùng cũng có thể sử dụng GUI tương tác để tải tập dữ liệu không gian địa lý mà không cần mã hóa. Bản đồ lá dành cho bất kỳ ai muốn phân tích và trực quan hóa dữ liệu không gian địa lý một cách tương tác trong môi trường Jupyter. Nó đặc biệt thích hợp cho người dùng mới làm quen với kỹ năng lập trình hạn chế.

Chương 1

Giới thiệu gói Python để phân tích không gian địa lý và lập bản đồ tương tác

1.1 Leafmap

Leafmap là một gói Python để lập bản đồ tương tác và phân tích không gian địa lý với mã hóa tối thiểu trong môi trường Jupyter. Đây là một dự án phụ của gói Python bản đồ địa lý, được thiết kế đặc biệt để hoạt động với Google Earth Engine (GEE). Tuy nhiên, không phải tất cả mọi người trong cộng đồng không gian địa lý đều có quyền truy cập vào nền tảng điện toán đám mây GEE. Bản đồ lá được thiết kế để lấp đầy khoảng trống này cho những người không sử dụng GEE. Đây là một gói Python mã nguồn mở và miễn phí cho phép người dùng phân tích và trực quan hóa dữ liệu không gian địa lý với mã hóa tối thiểu trong môi trường Jupyter, chẳng hạn như Google Colab, Jupyter Notebook và JupyterLab. Leafmap được xây dựng dựa trên một số gói mã nguồn mở, chẳng hạn như folium và ipyleaflet (để tạo bản đồ tương tác), WhiteboxTools và whiteboxgui (để phân tích dữ liệu không gian địa lý) và ipywidgets (để thiết kế giao diện người dùng đồ họa tương tác [GUI]). Leafmap có một bộ công cụ với các công cụ tương tác khác nhau cho phép người dùng tải dữ liệu vectơ và raster lên bản đồ mà không cần mã hóa. Ngoài ra, người dùng có thể sử dụng chương trình phụ trợ phân tích mạnh mẽ (tức là WhiteboxTools) để thực hiện phân tích không gian địa lý trực tiếp trong giao diện người dùng bản đồ lá mà không cần viết một dòng mã. Thư viện WhiteboxTools hiện chứa 468 công cụ để phân tích không gian địa lý nâng cao, chẳng hạn như Phân tích GIS, Phân tích Địa mạo, Phân tích Thủy văn, Phân tích Dữ liệu LiDAR, Phân tích Toán học và Thống kê và Phân tích Mạng Dòng.

Có rất nhiều gói Python để phân tích không gian địa lý, chẳng hạn như geopandas để phân tích dữ liệu vectơ và xarray để phân tích dữ liệu raster. Tuy nhiên, một số gói Python cung cấp GUI tương tác để tải dữ liệu không gian địa lý trong môi trường Jupyter. Có thể mất nhiều dòng mã để tải và hiển thị dữ liệu không gian địa lý với nhiều định dạng tệp khác nhau trên một bản đồ tương tác, đây có thể là một nhiệm vụ khó khăn đối với người dùng mới làm quen với kỹ năng viết mã hạn chế. Ngoài ra còn có một số gói Python đáng chú ý để trực quan hóa dữ liệu không gian địa lý trong môi trường Jupyter, chẳng hạn như plotly và kepler.gl. Tuy nhiên, vấn đề được thiết kế để hiển thị dữ liệu tĩnh, thiếu giao tiếp hai chiều giữa front-end và backend. Kepler.gl cung cấp chức năng 3D độc đáo để hiển thị tập dữ liệu không gian địa lý quy mô lớn, nhưng nó thiếu các công cụ để thực hiện phân tích không gian địa lý, chẳng hạn như phân tích thủy văn và phân tích dữ liệu LiDAR. Ngược lại, leafmap cung cấp nhiều chức năng thuận tiện để tải và hiển thị trực quan bộ dữ liệu không

gian địa lý chỉ với một dòng mã. Người dùng cũng có thể sử dụng GUI tương tác để tải tập dữ liệu không gian địa lý mà không cần mã hóa. Bản đồ lá dành cho bất kỳ ai muốn phân tích và trực quan hóa dữ liệu không gian địa lý một cách tương tác trong môi trường Jupyter. Nó đặc biệt thích hợp cho người dùng mới làm quen với kỹ năng lập trình hạn chế. Các lập trình viên nâng cao cũng có thể tìm thấy bản đồ lá một công cụ hữu ích để phân tích dữ liệu không gian địa lý và xây dựng các ứng dụng web tương tác.

1.2 Các tính năng của leafmap

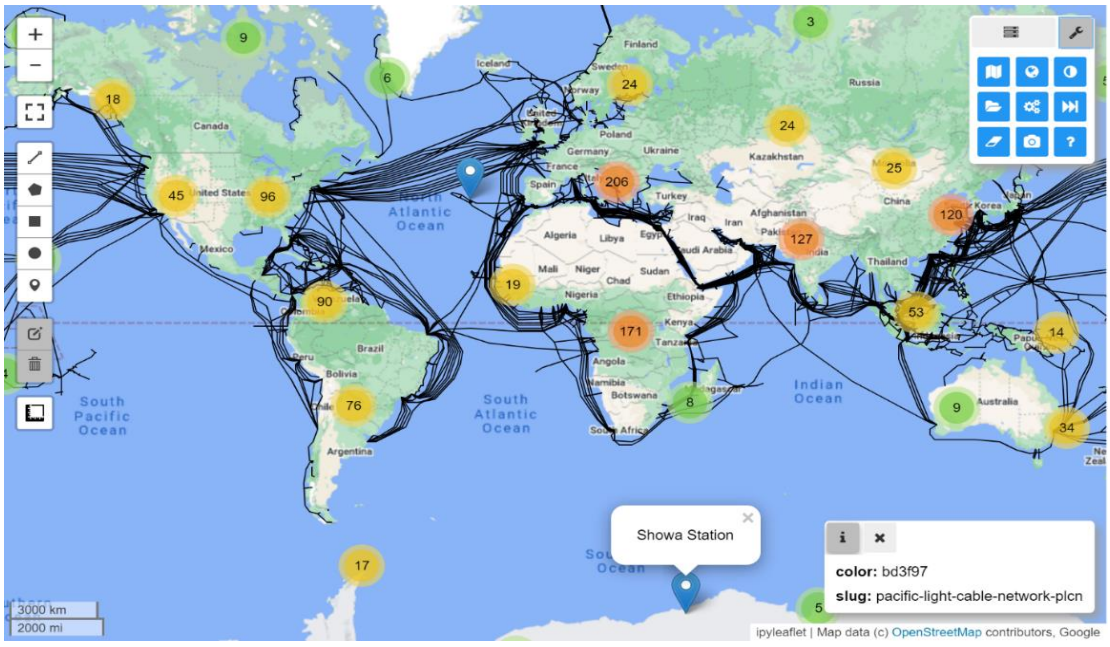
Dưới đây là danh sách một phần các tính năng có sẵn cho gói leafmap:

- Tạo một bản đồ tương tác chỉ với một dòng mã.
- Chọn từ nhiều bản đồ cơ sở khác nhau một cách tương tác mà không cần mã hóa.
- Thêm XYZ, WMS và các dịch vụ xếp hình vectơ vào bản đồ.
- Chuyển đổi CSV thành điểm và hiển thị điểm dưới dạng một cụm điểm đánh dấu.
- Thêm dữ liệu vectơ cục bộ (ví dụ: shapefile, GeoJSON, KML) vào bản đồ mà không cần mã hóa.
- Thêm dữ liệu raster cục bộ (ví dụ: GeoTIFF) vào bản đồ mà không cần mã hóa.
- Thêm GeoTIFF được tối ưu hóa trên đám mây (COG) và Danh mục tài sản theo thời gian (STAC) vào bản đồ.
- Thêm dữ liệu OpenStreetMap vào bản đồ bằng một dòng mã.
- Thêm GeoPandas GeoDataFrame vào bản đồ bằng một dòng mã.
- Thêm một lớp điểm với các thuộc tính bật lên vào bản đồ.
- Thêm dữ liệu từ cơ sở dữ liệu PostGIS vào bản đồ.
- Thêm truyền thuyết và thanh màu tùy chỉnh vào bản đồ.
- Thực hiện phân tích không gian địa lý bằng cách sử dụng WhiteboxTools và whiteboxgui.
- Tạo bản đồ bảng chia nhỏ và bản đồ được liên kết.
- Xuất bản bản đồ tương tác với một dòng mã.
- Tải xuống và hiển thị dữ liệu OpenStreetMap với một dòng mã.

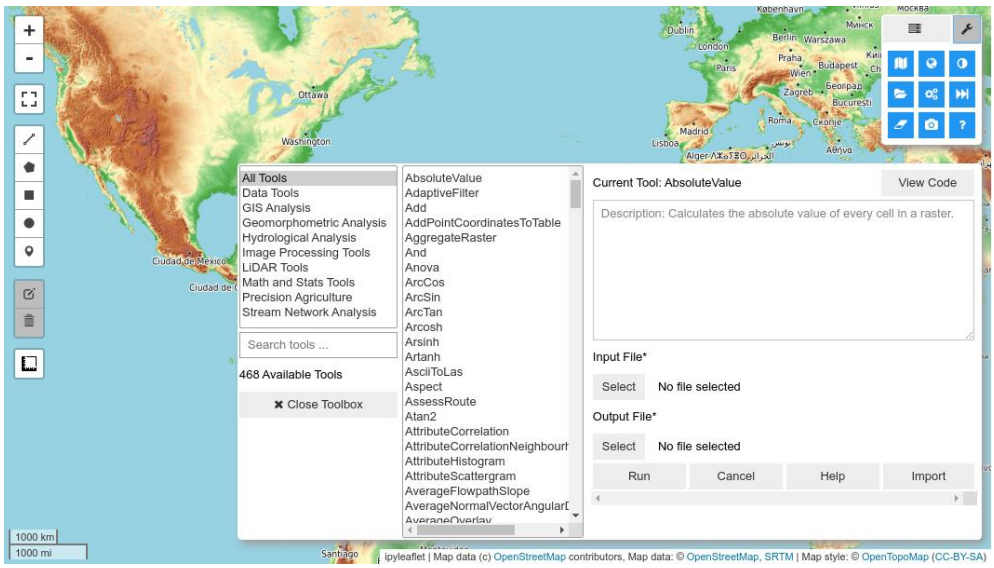
Leafmap có ba phụ trợ vẽ đồ thị, bao gồm folium, ipyleaflet và here-map-widget-for-jupyter. Một bản đồ tương tác được tạo bằng cách sử dụng một trong các phụ trợ vẽ đồ thị có thể được hiển thị trong môi trường Jupyter, chẳng hạn như Google Colab, Jupyter Notebook và JupyterLab. Theo mặc định, leafmap trong Jupyter Notebook và JupyterLab sẽ sử dụng phụ trợ vẽ sơ đồ ipyleaflet, trong khi nhập sơ đồ lá trong Google Colab sẽ sử dụng phụ trợ vẽ sơ đồ lá. Lưu ý rằng Google Colab chưa

hỗ trợ các tiện ích con tùy chỉnh, chẳng hạn như ipyleaflet và tiện ích bản đồ (nguồn). Do đó, các bản đồ tương tác được tạo bằng cách sử dụng phụ trợ tiện ích ipyleaflet và heremap sẽ không hiển thị trong Google Colab, mặc dù mã có thể chạy thành công mà không có bất kỳ lỗi nào.

Ba phần phụ trợ backend không cung cấp chức năng như nhau. Phần phụ trợ vẽ đồ thị ipyleaflet cung cấp chức năng tương tác phong phú nhất, bao gồm bộ công cụ tùy chỉnh để tải, phân tích và trực quan hóa dữ liệu không gian địa lý một cách tương tác mà không cần mã hóa. Ví dụ: người dùng có thể thêm dữ liệu vectơ (ví dụ: GeoJSON, Shapefile, KML, GeoDataFrame) và dữ liệu raster (ví dụ: GeoTIFF, Cloud Optimized GeoTIFF [COG]) vào bản đồ bằng một vài cú nhấp chuột (xem Hình 1). Người dùng cũng có thể thực hiện phân tích không gian địa lý bằng cách sử dụng WhiteboxTools GUI với 468 công cụ xử lý địa lý trực tiếp trong giao diện bản đồ (xem Hình 2). Chức năng tương tác khác (ví dụ: bản đồ bảng chia nhỏ, bản đồ được liên kết, thanh trượt thời gian, trình kiểm tra chuỗi thời gian) cũng có thể hữu ích để hiển thị dữ liệu không gian địa lý. Gói ipyleaflet được xây dựng dựa trên ipywidgets và cho phép giao tiếp hai chiều giữa front-end và backend cho phép sử dụng bản đồ để nắm bắt thông tin đầu vào của người dùng (nguồn). Ngược lại, folium có chức năng tương tác tương đối hạn chế. Nó chỉ dùng để hiển thị dữ liệu tĩnh. Phần phụ trợ vẽ sơ đồ trên lá cây được bao gồm trong gói này để hỗ trợ việc sử dụng bản đồ lá trong Google Colab. Lưu ý rằng bộ công cụ tùy chỉnh và chức năng tương tác nói trên không có sẵn cho chương trình phụ trợ vẽ đồ thị lá. So với ipyleaflet và folium, chương trình phụ trợ vẽ biểu đồ tiện ích con bản đồ cung cấp một số chức năng 3D độc đáo để trực quan hóa dữ liệu không gian địa lý. Cần có khóa API từ Cổng nhà phát triển tại đây để sử dụng bản đồ này.



Hình 1. Giao diện người dùng leafmap được xây dựng dựa trên ipyleaflet và ipywidgets.



Hình 2. Giao diện người dùng đồ họa của WhiteboxTools được tích hợp vào leafmap

1.3 Mô-đun leafmap

Chức năng chính của gói leafmap Python được tổ chức thành chín mô-đun như được hiển thị trong bảng bên dưới.

Mô đun	Mô tả
basemaps	bản đồ nền: Một tập hợp các lớp xếp XYZ và WMS được sử dụng làm bản đồ cơ sở
colormaps	bản đồ màu: Các bản đồ màu và bảng màu thường được sử dụng để hiển thị dữ liệu không gian địa lý
common	Các chức năng phổ biến đang được sử dụng bởi nhiều phần mềm phụ trợ vẽ đồ thị để xử lý dữ liệu không gian địa lý
foliumap	foliumap Một chương trình phụ trợ được xây dựng dựa trên gói folium Python
heremap	heremap Một chương trình phụ trợ được xây dựng dựa trên here-map-widget-for-jupyter
leafmap	Chương trình phụ trợ vẽ đồ thị mặc định được xây dựng dựa trên gói ipyleaflet Python
legends	tích hợp cho tập dữ liệu không gian địa lý thường được sử dụng
osm	Các chức năng để giải nén và tải xuống dữ liệu OpenStreetMap
toolbar	Một bộ công cụ tùy chỉnh với các công cụ tương tác được xây dựng dựa trên ipywidgets và ipyleaflet

Khởi chạy Jupyter notebook

```
1 conda activate env_name
```

2 jupyter notebook

Sử dụng ipyleaflet

```
import leafmap
m = leafmap.Map(center=(40, -100), zoom=4)
m
```

Sử dụng folium

```
import leafmap.foliumap as leafmap
m = leafmap.Map(center=(40, -100), zoom=4)
m
```

Sử dụng bản đồ heremap phụ trợ vẽ đồ thị

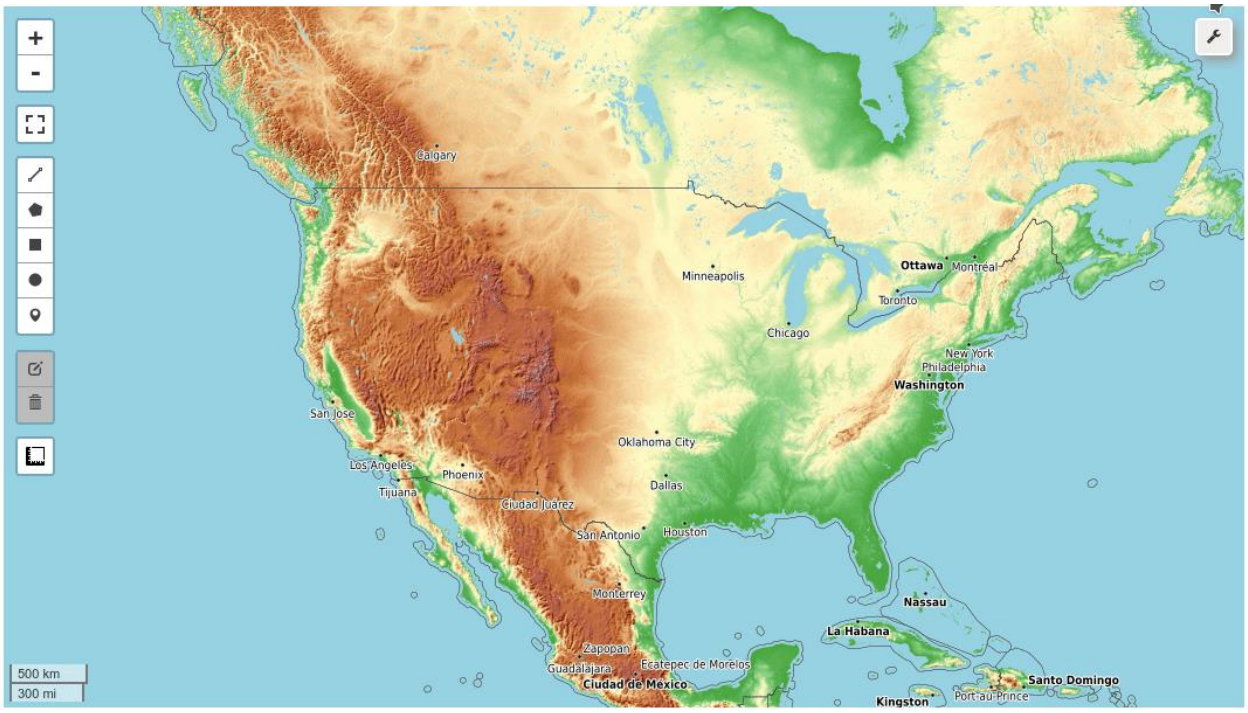
Điều kiện tiên quyết là Tài khoản nhà phát triển HERE, miễn phí và có sẵn trong Cổng thông tin nhà phát triển HERE. Khóa API từ Cổng nhà phát triển HERE.

Sau đó, xuất khóa API sang biến môi trường HEREMAPS_API_KEY bằng cú pháp:

export HEREMAPS_API_KEY=YOUR-ACTUAL-API-KEY

1.4. Tạo một bản đồ tương tác

```
import os
import leafmap.heremap as leafmap
os.environ["HEREMAPS_API_KEY"] = "YOUR_HEREMAPS_API_KEY"
api_key = os.environ.get("HEREMAPS_API_KEY") # read api_key from environment variable.
m = leafmap.Map(api_key=api_key, center=(40, -100), zoom=4)
m
```



Chương 2. Cài đặt và sử dụng leafmap phân tích dữ liệu địa không gian

2.1 Cài đặt leafmap

In [1]:

```
import os
import subprocess
```

In [2]:

```
try:
    import leafmap
except ImportError:
    print('Installing leafmap ...')
    subprocess.check_call(["python", '-m', 'pip', 'install', 'leafmap'])
```

2.2 Sử dụng ipyleaflet plotting backend

In [3]:

```
import leafmap.foliumap as leafmap
```

Sử dụng folium plotting backend

In [4]:

```
import leafmap
```

Tạo bản đồ tương tác

In [5]:

```
m = leafmap.Map(center=(40, -100), zoom=4)
m
```

Out[5]:

Tùy chỉnh chiều cao bản đồ

In [6]:

```
m = leafmap.Map(height="400px", width="800px")
m
```

Out[6]:

2.3 Đặt khả năng hiển thị điều khiển

In [7]:

```
m = leafmap.Map(draw_control=False, measure_control=False, fullscreen_control=False, attribution_control=True)
m
```

Out[7]:

2.4 Thay đổi bản đồ nền

In [8]:

```
m = leafmap.Map()
m.add_basemap("Esri.NatGeoWorldMap")
m
```

Out[8]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.5 Thêm lớp XYZ tile layer

In [9]:

```
m = leafmap.Map()
m.add_tile_layer(url="https://mt1.google.com/vt/lyrs=y&x={x}&y={y}&z={z}", name="Google Satellite", attribution="Google")
m
```

Out[9]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.6 Thêm lớp xếp WMS

In [10]:

```
m = leafmap.Map()
naip_url = 'https://services.nationalmap.gov/arcgis/services/USGSNAIPImagery/ImageServer/WMSServer?'
```

m.add_wms_layer(url=naip_url, layers='0', name='NAIP Imagery', format='image/png', shown=**True**)

m

Out[10]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.7 Thêm lớp COG layer

In [11]:

m = leafmap.Map()

url = 'https://opendata.digitalglobe.com/events/california-fire-2020/pre-event/2018-02-16/pine-gulch-fire20/1030010076004E00.tif'

m.add_cog_layer(url, name="Fire (pre-event)")

m

Out[11]:

2.8 Thêm lớp STAC

In [12]:

m = leafmap.Map()

url = 'https://canada-spot-ortho.s3.amazonaws.com/canada_spot_orthoimages/canada_spot5_orthoimages/S5_2007/S5_11055_6057_20070622/S5_11055_6057_20070622.json'

m.add_stac_layer(url, bands=['B3', 'B2', 'B1'], name='False color')

m

Out[12]:

2.9 Thêm ghi chú

In [13]:

m = leafmap.Map()

url = "https://www.mrlc.gov/geoserver/mrlc_display/NLCD_2016_Land_Cover_L48/wms?"

m.add_wms_layer(url, layers="NLCD_2016_Land_Cover_L48", name="NLCD 2016 CONUS Land Cover",format="image/png", transparent=**True**)

m.add_legend(builtin_legend='NLCD')

m

Out[13]:

2.10 Thêm thanh màu

In [14]:

m = leafmap.Map()

m.add_basemap('USGS 3DEP Elevation')

colors = ['006633', 'E5FFCC', '662A00', 'D8D8D8', 'F5F5F5']

vmin = 0

vmax = 4000

m.add_colorbar(colors=colors, vmin=vmin, vmax=vmax)

m

Out[14]:

2.11 Add GeoJSON

In [15]:

m = leafmap.Map(center=[0, 0], zoom=2)

in_geojson = 'https://raw.githubusercontent.com/giswqs/leafmap/master/examples/data/cable-geo.geojson'

m.add_geojson(in_geojson, layer_name="Cable lines")

m

Out[15]:

Add GeoJSON

In [15]:

m = leafmap.Map(center=[0, 0], zoom=2)

```
in_geojson = 'https://raw.githubusercontent.com/giswqs/leafmap/master/examples/data/cable-geo.geojson'

m.add_geojson(in_geojson, layer_name="Cable lines")

m
```

Out[15]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [16]:

```
# Add a GeoJSON with random filled color to the map.

m = leafmap.Map(center=[0, 0], zoom=2)

url = "https://raw.githubusercontent.com/giswqs/leafmap/master/examples/data/countries.geojson"

m.add_geojson(url, layer_name="Countries", fill_colors=['red', 'yellow', 'green', 'orange'])

m
```

Out[16]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [17]:

```
# Use custom style and hover_style functions.

m = leafmap.Map(center=[0, 0], zoom=2)

url = "https://raw.githubusercontent.com/giswqs/leafmap/master/examples/data/countries.geojson"

style = {
    "stroke": True,
    "color": "#0000ff",
    "weight": 2,
    "opacity": 1,
```

```
    "fill": True,
    "fillColor": "#0000ff",
    "fillOpacity": 0.1,
}
hover_style = {"fillOpacity": 0.7}
m.add_geojson(url, layer_name="Countries", style=style, hover_style=hover_style)
m
```

Out[17]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.12 Add shapefile

In [18]:

```
m = leafmap.Map(center=[0, 0], zoom=2)
in_shp = 'https://github.com/giswqs/leafmap/raw/master/examples/data/countries.zip'
m.add_shp(in_shp, layer_name="Countries")
m
```

Out[18]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.13 Add KML

In [19]:

```
try:
    import geopandas
except ImportError:
    print('Installing geopandas ...')
    subprocess.check_call(["python", '-m', 'pip', 'install', 'geopandas'])
```

In [20]:

```
m = leafmap.Map()
```

```
in_kml = 'https://raw.githubusercontent.com/giswqs/leafmap/master/examples/data/us-states.kml'

m.add_kml(in_kml,layer_name="US States KML")

m
```

Out[20]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.14 Thêm GeoDataFrame

In [21]:

```
import geopandas as gpd

m = leafmap.Map()

gdf = gpd.read_file("https://github.com/giswqs/leafmap/raw/master/examples/data/cable-geo.geojson")

m.add_gdf(gdf, layer_name="Cable lines")

m
```

Out[21]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.15 Tạo bản đồ nhiệt

In [22]:

```
m = leafmap.Map()

in_csv = "https://raw.githubusercontent.com/giswqs/leafmap/master/examples/data/world_cities.csv"

m.add_heatmap(in_csv, latitude="latitude", longitude='longitude', value="pop_max", name="Heat map", radius=20)
```

In [23]:

```
colors = ['blue', 'lime', 'red']

vmin = 0
```

```
vmax = 10000
m.add_colorbar(colors=colors, vmin=vmin, vmax=vmax)
m.add_title("World Population Heat Map", font_size="20px", align="center")
m
```

Out[23]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.16 Lưu bản đồ vào HTML

In [24]:

```
m = leafmap.Map()
m.add_basemap("Esri.NatGeoWorldMap")
m
```

Out[24]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [25]:

```
m.to_html("mymap.html")
```

2.17 Thêm hình ảnh vệ tinh

In [26]:

```
# os.environ["PLANET_API_KEY"] = "12345"
```

In [27]:

```
m = leafmap.Map()
m.add_planet_by_month(year=2020, month=8)
m.add_planet_by_quarter(year=2019, quarter=2)
m
```

Out[27]:

Make this Notebook Trusted to load map: File -> Trust Notebook

2.18 Thêm dữ liệu OpenStreetMap

Add OSM data of place(s) by name or ID to the map.

In [28]:

```
m = leafmap.Map(toolbar_control=False, layers_control=True)
m.add_osm_from_geocode("New York City", layer_name='NYC')
m
```

Out[28]:

Make this Notebook Trusted to load map: File -> Trust Notebook

Add OSM entities within boundaries of geocodable place(s) to the map.

In [29]:

```
m = leafmap.Map(toolbar_control=False, layers_control=True)
place = "Bunker Hill, Los Angeles, California"
tags = {"building": True}
m.add_osm_from_place(place, tags, layer_name="Los Angeles, CA")
m
```

2021-12-29 01:27:11 Configured OSMnx 1.1.2

2021-12-29 01:27:11 HTTP response caching is on

2021-12-29 01:27:11 Pausing 1 seconds before making HTTP GET request

2021-12-29 01:27:12 Get https://nominatim.openstreetmap.org/search?format=json&polygon_geojson=1&dedupe=0&limit=50&q=Bunker+Hill%2C+Los+Angeles%2C+California with timeout=180

2021-12-29 01:27:12 Resolved nominatim.openstreetmap.org to 140.211.167.100

2021-12-29 01:27:12 Downloaded 1.2kB from nominatim.openstreetmap.org

2021-12-29 01:27:12 Saved response to cache file "cache/8d8366ad6b96fa854a9d6d038c022b8bdcdb6fe6.json"

2021-12-29 01:27:12 Created GeoDataFrame with 1 rows from 1 queries

2021-12-29 01:27:12 Constructed place geometry polygon(s) to query API

2021-12-29 01:27:12 Projected GeoDataFrame to +proj=utm +zone=11 +ellps=WGS84 +datum=WGS84 +units=m +no_defs +type=crs

2021-12-29 01:27:12 Projected GeoDataFrame to epsg:4326

2021-12-29 01:27:12 Requesting data within polygon from API in 1 request(s)

2021-12-29 01:27:12 Resolved overpass-api.de to 178.63.48.217

2021-12-29 01:27:13 Pausing 0 seconds before making HTTP POST request

2021-12-29 01:27:13 Post https://overpass-api.de/api/interpreter?data=%5Bout%3Ajson%5D%5Btimeout%3A180%5D%3B%28%28node%5B%27building%27%5D%28poly%3A%2734.056122+-118.256052+34.055025+-118.254350+34.050990+-118.248100+34.053625+-118.245628+34.058656+-118.253434+34.057132+-118.255014+34.056122+-118.256052%27%29%3B%28._%3B%3E%3B%29%3B%29%3B%28way%5B%27building%27%5D%28poly%3A%2734.056122+-118.256052+34.055025+-118.254350+34.050990+-118.248100+34.053625+-118.245628+34.058656+-118.253434+34.057132+-118.255014+34.056122+-118.256052%27%29%3B%28._%3B%3E%3B%29%3B%29%3B%28relation%5B%27building%27%5D%28poly%3A%2734.056122+-118.256052+34.055025+-118.254350+34.050990+-118.248100+34.053625+-118.245628+34.058656+-118.253434+34.057132+-118.255014+34.056122+-118.256052%27%29%3B%28._%3B%3E%3B%29%3B%29%3B%29%3Bout%3B with timeout=180

2021-12-29 01:27:13 Resolved overpass-api.de to 178.63.48.217

2021-12-29 01:27:18 Downloaded 165.2kB from overpass-api.de

2021-12-29 01:27:18 Saved response to cache file "cache/95fea0c13cd89a78cbaa52ae4b13a8c9ebd9eb21.json"

2021-12-29 01:27:18 Got all geometries data within polygon from API in 1 request(s)

2021-12-29 01:27:18 Converting 1436 elements in JSON responses to geometries

2021-12-29 01:27:18 77 geometries created in the dict

2021-12-29 01:27:18 13 untagged geometries removed

2021-12-29 01:27:18 Created r-tree spatial index for 64 geometries

2021-12-29 01:27:18 Identified 64 geometries inside polygon

2021-12-29 01:27:18 0 geometries removed by the polygon filter

2021-12-29 01:27:18 8 geometries removed by the tag filter

2021-12-29 01:27:18 56 geometries in the final GeoDataFrame

Out[29]:

KẾT LUẬN

Leafmap là một gói Python để lập bản đồ tương tác và phân tích không gian địa lý với mã hóa tối thiểu trong môi trường Jupyter. Đây là một dự án spin-off của geemap Python gói, được thiết kế đặc biệt để làm việc với Google Earth Engine (GEE). Tuy nhiên, không phải tất cả mọi người trong cộng đồng không gian địa lý đều có quyền truy cập vào nền tảng điện toán đám mây GEE. Bản đồ lá được thiết kế để lấp đầy khoảng trống này cho những người không sử dụng GEE. Đây là một gói Python mã nguồn mở và miễn phí cho phép người dùng phân tích và trực quan hóa dữ liệu không gian địa lý với mã hóa tối thiểu trong môi trường Jupyter, chẳng hạn như Google Colab, Jupyter Notebook và JupyterLab. Bản đồ lá được xây dựng dựa trên một số gói mã nguồn mở, chẳng hạn như folium và ipyleaflet (để tạo bản đồ tương tác),WhiteboxTools và whiteboxgui (để phân tích dữ liệu không gian địa lý) và ipywidgets (để thiết kế giao diện người dùng đồ họa tương tác [GUI]). Leafmap có một bộ công cụ với các công cụ tương tác khác nhau cho phép người dùng tải dữ liệu vectơ và raster lên bản đồ mà không cần mã hóa. Ngoài ra, người dùng có thể sử dụng chương trình phụ trợ phân tích mạnh mẽ (tức là WhiteboxTools) để thực hiện phân tích không gian địa lý trực tiếp trong giao diện người dùng bản đồ lá mà không cần viết một dòng mã nào. Thư viện WhiteboxTools hiện chứa **468** công cụ để phân tích không gian địa lý nâng cao, chẳng hạn như Phân tích GIS , Phân tích Địa mạo , Phân tích Thủy văn , Phân tích Dữ liệu LiDAR ,Toán học và Phân tích thống kê , và Suối Phân tích mạng .

TÀI LIỆU THAM KHẢO

[1] https://leafmap.org/tutorials/

[2] https://github.com/giswqs/leafmap

[3] https://leafmap.org/notebooks/00_key_features/

[4] https://colab.research.google.com/github/giswqs/leafmap/blob/master/examples/workshops/SIGSPATIAL_2021.ipynb#scrollTo=EtgsP6wMvL_d