Guide Document to Use Folium in Python for Interactive Mapping

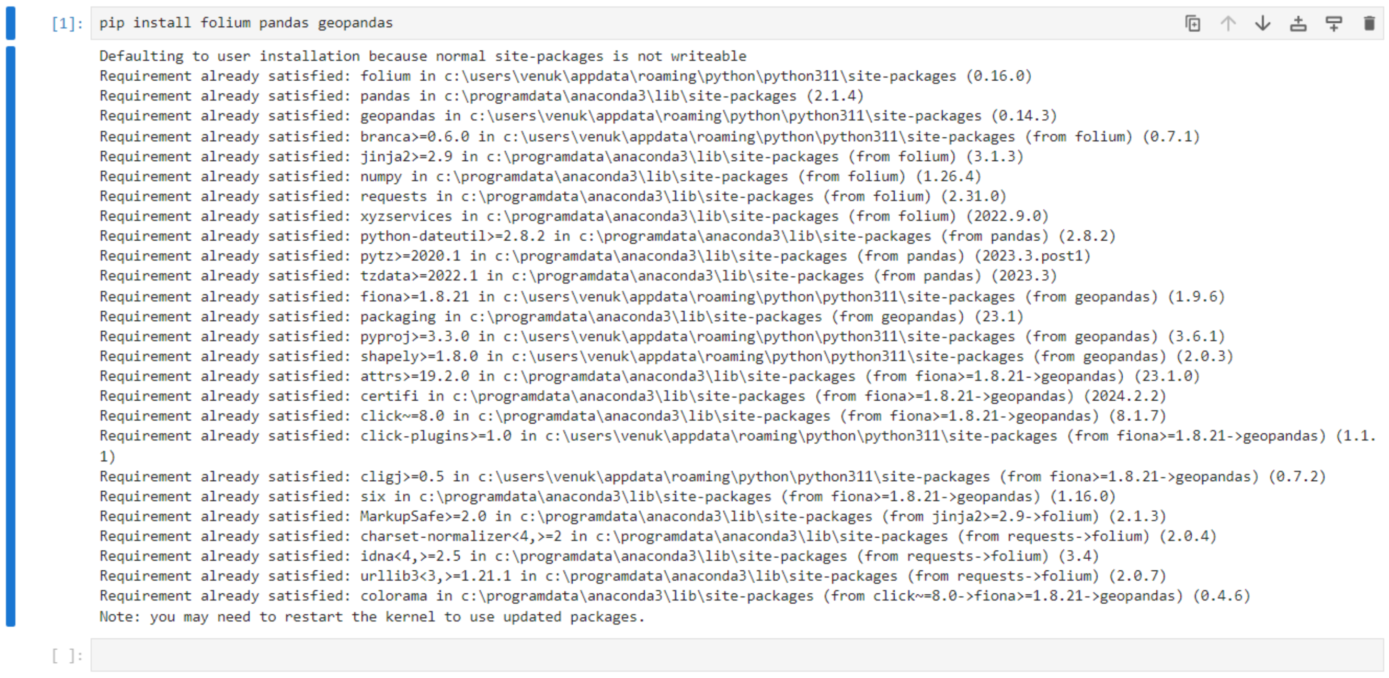
This document is designed to provide a comprehensive guide on using Folium, a Python library that enables the creation of interactive maps. Aimed at visualizing geospatial data effectively, Folium integrates with data handling libraries like Pandas and GeoPandas, making it a valuable tool for data scientists and geospatial analysts. By following the steps outlined, users will learn how to set up their environment, prepare data, and leverage Folium's capabilities to produce dynamic, interactive maps that can visually represent complex datasets.

1. Introduction to Folium

Folium is a Python library used for creating interactive maps. It leverages the mapping strengths of the Leaflet.js library, enabling the visual presentation of data overlaid on geographical maps. It's particularly useful for data scientists and geospatial data analysts who need to visualize complex data in an intuitive way.

2. Prerequisites

Ensure you have Python installed and the following packages: folium, pandas, geopandas. You can install them using pip if they are not already installed:



```
[1]: pip install folium pandas geopandas
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: folium in c:\users\venuk\appdata\roaming\python\python311\site-packages (0.16.0)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packages (2.1.4)
Requirement already satisfied: geopandas in c:\users\venuk\appdata\roaming\python\python311\site-packages (0.14.3)
Requirement already satisfied: branca>=0.6.0 in c:\users\venuk\appdata\roaming\python\python311\site-packages (from folium) (0.7.1)
Requirement already satisfied: jinja2>=2.9 in c:\programdata\anaconda3\lib\site-packages (from folium) (3.1.3)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from folium) (1.26.4)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (from folium) (2.31.0)
Requirement already satisfied: xyzservices in c:\programdata\anaconda3\lib\site-packages (from folium) (2022.9.0)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: fiona>=1.8.21 in c:\users\venuk\appdata\roaming\python\python311\site-packages (from geopandas) (1.9.6)
Requirement already satisfied: packaging in c:\programdata\anaconda3\lib\site-packages (from geopandas) (23.1)
Requirement already satisfied: pyproj>=3.3.0 in c:\users\venuk\appdata\roaming\python\python311\site-packages (from geopandas) (3.6.1)
Requirement already satisfied: shapely>=1.8.0 in c:\users\venuk\appdata\roaming\python\python311\site-packages (from geopandas) (2.0.3)
Requirement already satisfied: attrs>=19.2.0 in c:\programdata\anaconda3\lib\site-packages (from fiona>=1.8.21->geopandas) (23.1.0)
Requirement already satisfied: certifi in c:\programdata\anaconda3\lib\site-packages (from fiona>=1.8.21->geopandas) (2024.2.2)
Requirement already satisfied: click~=8.0 in c:\programdata\anaconda3\lib\site-packages (from fiona>=1.8.21->geopandas) (8.1.7)
Requirement already satisfied: click-plugins>=1.0 in c:\users\venuk\appdata\roaming\python\python311\site-packages (from fiona>=1.8.21->geopandas) (1.1.
1)
Requirement already satisfied: cligj>=0.5 in c:\users\venuk\appdata\roaming\python\python311\site-packages (from fiona>=1.8.21->geopandas) (0.7.2)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from fiona>=1.8.21->geopandas) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\programdata\anaconda3\lib\site-packages (from jinja2>=2.9->folium) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests->folium) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests->folium) (2.0.7)
Requirement already satisfied: colorama in c:\programdata\anaconda3\lib\site-packages (from click~=8.0->fiona>=1.8.21->geopandas) (0.4.6)
Note: you may need to restart the kernel to use updated packages.
```

```
[ ]:
```

3. Import Necessary Libraries

```
import folium
from folium.plugins import MarkerCluster, FeatureGroupSubGroup
import pandas as pd
import geopandas as gpd
```

Explanation:

- Folium: For creating interactive maps.
- MarkerCluster: To manage large numbers of markers and improve map performance.
- Pandas: For data manipulation and analysis.
- GeoPandas: For handling geospatial data.

4. Load and Prepare Data

```
# Load data into a DataFrame
data_path = 'C:/Users/venuk/Downloads/current_trees_renamed.csv'
current_trees_renamed = pd.read_csv(data_path)

# Convert DataFrame to GeoDataFrame
gdf = gpd.GeoDataFrame(
    current_trees_renamed,
    geometry=gpd.points_from_xy(current_trees_renamed.longitude, current_trees_renamed.latitude),
    crs="EPSG:4326"
)
```

Explanation:

- This step involves loading the dataset from a CSV file into a pandas DataFrame and converting it into a GeoDataFrame. This conversion is crucial for spatial operations, including mapping with Folium.

5. Initialize the Map

```
melbourne_map = folium.Map(location=[-37.8136, 144.9631], zoom_start=12)
```

Explanation:

- Initializes a map centered around Melbourne using Folium. The zoom_start parameter adjusts the initial zoom level to provide a suitable overview of the area.

6. Create Marker Clusters

```
marker_cluster = MarkerCluster().add_to(melbourne_map)
```

Explanation:

- Adds a MarkerCluster to the map, which is essential for handling a large number of markers efficiently. This cluster groups nearby markers together to improve the map's readability and performance.

7. Categorize and Add Markers

```python
# Create subgroups for parks and streets
park_group = FeatureGroupSubGroup(marker_cluster, 'Parks')
street_group = FeatureGroupSubGroup(marker_cluster, 'Streets')
park_group.add_to(melbourne_map)
street_group.add_to(melbourne_map)

# Add markers to the appropriate subgroup based on their location
for idx, row in gdf.iterrows():
    folium.CircleMarker(
        location=[row.geometry.y, row.geometry.x],
        radius=3,
        color='green' if row['located_in'] == 'Park' else 'red',
        fill=True,
        fill_color='green' if row['located_in'] == 'Park' else 'red',
        popup=f"Species: {row['species']}, Planted: {row['date_planted']}"
    ).add_to(park_group if row['located_in'] == 'Park' else street_group)
```

Explanation:

- Markers are categorized into two groups - Parks and Streets. Markers are added to these groups based on their specified locations. Each marker is interactive, displaying a popup with details on click.

8. Add Layer Control

```python
folium.LayerControl().add_to(melbourne_map)
```

Explanation:

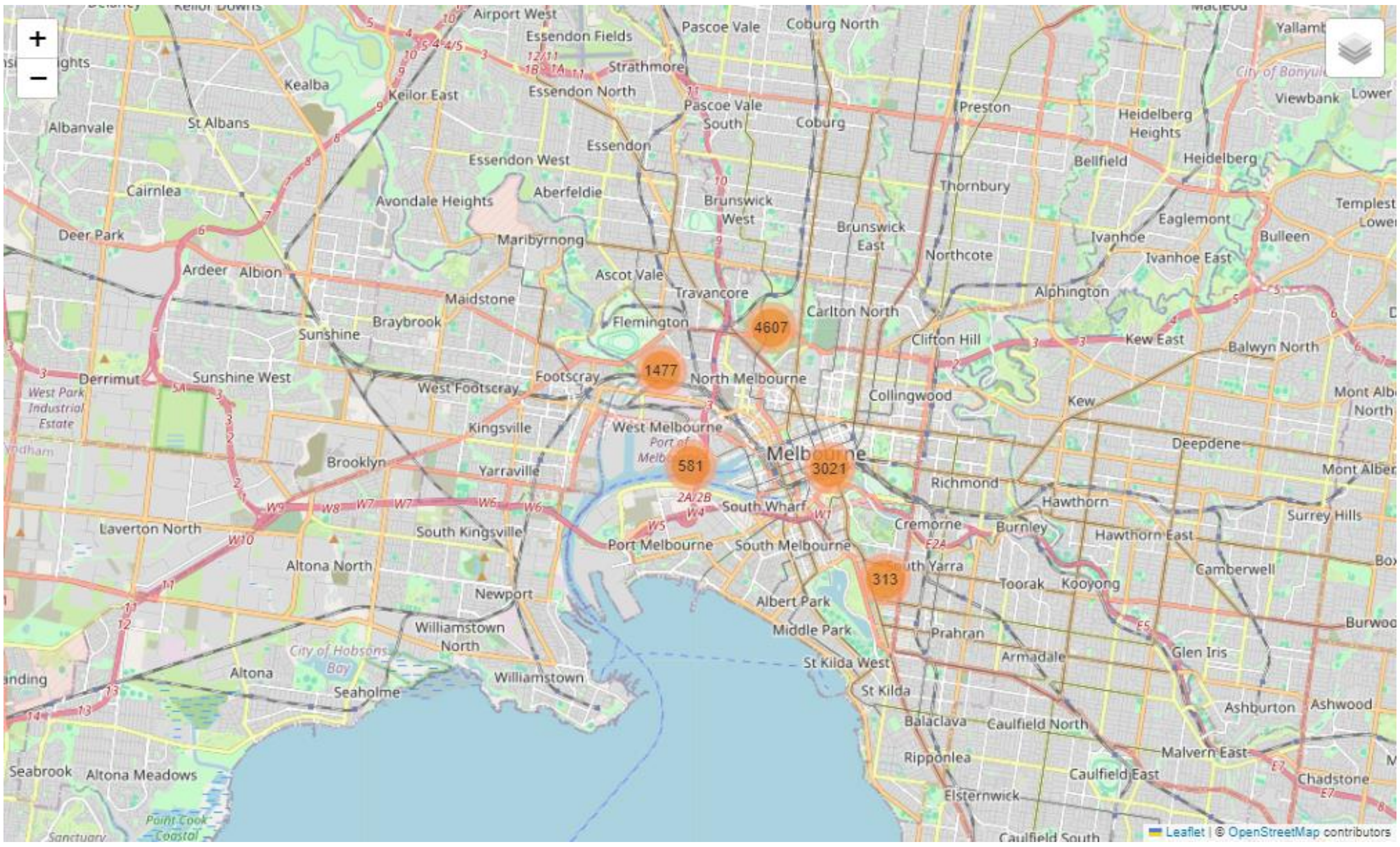
- Integrates a layer control tool into the map, allowing users to toggle visibility for each marker group. This enhances the interactivity, enabling users to customize their view.

9. Display the Map

```python
melbourne_map
```

Explanation:

- This final step renders the interactive map in the Jupyter Notebook. It showcases all the layers and markers, providing an interactive tool for exploring tree locations in Melbourne.



10. Conclusion

This tutorial offers a foundational approach to using Folium within a Jupyter Notebook to create dynamic and interactive maps. It is designed to help beginners understand and apply basic geospatial visualization techniques effectively. As users become more comfortable, they are encouraged to explore additional Folium features and customization options to further enhance their visualizations.

Author

Venuka 2024.v1