Contents

1 Making Plots with matplotlib *3*

2 Geographical Data *5*

3 Longitude and Latitude *7*

3.1 Nautical Mile *7*

3.2 Haversine Formula *8*

4 Geocoding and Reverse Geocoding *9*

4.1 Geocoding *9*

4.2 Reverse Geocoding *10*

5 Making a Map *13*

A Answers to Exercises *15*

Index *17*

CHAPTER 1

Making Plots with matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It's highly useful for presenting data in a more intuitive and easy-to-understand manner.

In order to use Matplotlib, you must first import it, typically using the following line of code:

```
import matplotlib.pyplot as plt
```

Let's create a simple line plot. Suppose we have a list of numbers and we want to visualize their distribution:

```
x = [1, 2, 3, 4, 5]
y = [1, 4, 9, 16, 25]

plt.plot(x, y)
plt.show()
```

Here, 'x' and 'y' are the coordinates of the points. The 'plt.plot' function plots y versus x as lines and/or markers. The 'plt.show' function then displays the figure.

Creating a bar plot follows a similar approach:

```
labels = ['A', 'B', 'C', 'D', 'E']
values = [5, 7, 9, 11, 13]

plt.bar(labels, values)
plt.show()
```

Here, 'labels' are the categories we are plotting, and 'values' are the respective sizes of those categories. The 'plt.bar' function creates a bar plot.

Matplotlib provides a variety of other plot types and customization options - everything from scatter plots and histograms to custom line styles and colors. Explore the official Matplotlib documentation to learn more about what this powerful library can offer.

Chapter 2

Geographical Data

Chapter 3

Longitude and Latitude

The Earth can be represented as a sphere, and the position of a point on its surface can be described using two coordinates: latitude and longitude.

Latitude is a measure of a point's distance north or south of the Equator, expressed in degrees. It ranges from $-90°$ at the South Pole to $+90°$ at the North Pole, with $0°$ representing the Equator.

Longitude, on the other hand, measures a point's distance east or west of the Prime Meridian (which passes through Greenwich, England). It ranges from $-180°$ to $+180°$, with the Prime Meridian represented as $0°$.

3.1 Nautical Mile

A nautical mile is a unit of measurement used primarily in aviation and maritime contexts. It is based on the circumference of the Earth and is defined as one minute ($1/60°$) of latitude. This makes it directly related to the Earth's geometry, unlike a kilometer or a mile, which are arbitrary in nature. The exact value of a nautical mile can vary slightly

depending on which type of latitude you use (e.g., geodetic, geocentric, etc.), but for practical purposes, it's often approximated as 1.852 kilometers or 1.15078 statute miles.

3.2 Haversine Formula

The haversine formula is an equation important in navigation for giving great-circle distances between two points on a sphere from their longitudes and latitudes. It's especially useful when it comes to calculating distances between points on the surface of the Earth, which we represent as a sphere for simplicity.

In its basic form, the haversine formula is as follows:

$$a = \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right)$$

$$c = 2 \cdot \operatorname{atan2}\left(\sqrt{a}, \sqrt{1-a}\right)$$

$$d = R \cdot c$$

Here, ϕ represents the latitudes of the two points (in radians), $\Delta\phi$ and $\Delta\lambda$ represent the differences in latitude and longitude (also in radians), and R is the radius of the Earth. The result, d, is the distance between the two points along the surface of the sphere.

Chapter 4

Geocoding and Reverse Geocoding

Geocoding and reverse geocoding are essential processes in geographic information systems (GIS) that are used to convert between addresses and spatial data.

4.1 Geocoding

Geocoding is the process of converting addresses (like ”1600 Amphitheatre Parkway, Mountain View, CA”) into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers on a map, or position the map. The resulting latitude and longitude are often used as a key index in merging datasets based on location.

Here is an example of using Google’s Geocoding service to get the longitude and latitude of the Dallas County Administration Building:

```
import requests
```

```
import json

# Encode the parameters
parameters = {"address": "411 Elm St, Dallas, TX 75202"  , "key": "YOUR_API_KEY"}
base_url = "https://maps.googleapis.com/maps/api/geocode/json?"

# Send the GET request
response = requests.get(base_url,  params=parameters)

# Convert the response to json
data = response.json()

# Extract the latitude and longitude
if len(data["results"]) > 0:
    latitude = data["results"][0]["geometry"]["location"]["lat"]
    longitude = data["results"][0]["geometry"]["location"]["lng"]
else:
    print(f"Could not find the latitude and longitude.")
```

4.2 Reverse Geocoding

Reverse geocoding, as the name implies, is the opposite process of geocoding. It involves converting geographic coordinates into a human-readable address. This can be useful in applications where you need to display an actual address to a user instead of latitude and longitude coordinates.

Here is an example of using Google's reverse geocoding API to find the address for

```
import requests
import json

api_key = "YOUR_API_KEY"
latitude = 33.9474096
longitude = -118.1179069

 # Encode the parameters
parameters = {"latlng": f"{latitude},{longitude}", "key": api_key}
base_url = "https://maps.googleapis.com/maps/api/geocode/json?"

# Send the GET request
response = requests.get(base_url, params=parameters)

# Convert the response to json
data = response.json()
```

```
# Extract the address
if len(data["results"]) > 0:
    address = data["results"][0]["formatted_address"]
else:
    print(f"Could not find the address")
```

Chapter 5

Making a Map

Plotly is an open-source data visualization library for Python, R, and JavaScript. It allows for interactive plots, including geographical maps. In this brief example, we will learn how to create a simple annotated map using Plotly in Python.

To start, you need to install Plotly. In Python, you can do this via pip:

```
pip install plotly
```

Once installed, you can create a map with annotations as follows:

```
import plotly.graph_objects as go

fig = go.Figure(data=go.Scattergeo(
    lon = [-75.789],
    lat = [45.4215],
    text = ['Ottawa'],
    mode = 'text',
))

fig.update_layout(
```

```
    title_text = 'Annotated␣Map␣with␣Plotly',
    showlegend = False,
    geo = dict(
        scope = 'world',
        projection_type = 'azimuthal␣equal␣area',
        showland = True,
        landcolor = 'rgb(243,␣243,␣243)',
        countrycolor = 'rgb(204,␣204,␣204)',
    ),
)

fig.show()
```

This code creates a world map and places a text annotation at the geographic coordinates for Ottawa. The 'go.Scattergeo' function is used to define the geographical scatter plot (i.e., the annotation), while the 'update_layout' function is used to define the appearance and the properties of the map itself.

In this example, you can replace the latitude, longitude, and text with the values corresponding to your desired location.

Appendix A

Answers to Exercises

Index

geocoding, 9

Haversine formula, 8

latitude, 7
longitude, 7

matplotlib, 3

nautical mile, 8

plotly, 13

reverse geocoding, 10