Module 6 Notes

#### Deliverable 1: Retrieve Weather Data

For this deliverable, you'll generate a set of 2,000 random latitudes and longitudes, retrieve the nearest city, and perform an API call with OpenWeatherMap. In addition to the city weather data you gathered in this module, use your API skills to retrieve the current weather description for each city. Then, create a new DataFrame containing the updated weather data.

1. Create a folder called Weather\_Database to save all the files related with this deliverable.

GitHub Location: https://github.com/LouFoster/Weather\_Database.git

1. Save the Weather\_Database\_starter\_code.ipynb starter code to the Weather\_Database folder and rename it as Weather\_Database.ipynb.
2. Use the np.random.uniform function to generate a new set of 2,000 random latitudes and 2,000 longitudes.

Text

Description automatically generated

1. Use the citipy module to get the nearest city for each latitude and longitude combination.

Graphical user interface, text, application

Description automatically generated

1. Import your OpenWeatherMap's API key and assemble the API call URL as a string variable. Recall to edit the config.py file to add your API key; also, it's critical to avoid publishing your API key on your GitHub repository.

Graphical user interface, text, application

Description automatically generated

1. Retrieve the following information from the API call:
   * Latitude and longitude
   * Maximum temperature
   * Percent humidity
   * Percent cloudiness
   * Wind speed
   * Weather description (for example, clouds, fog, light rain, clear sky)

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

1. Add the weather data to a new DataFrame.
   * Before continue to the next step, take a moment to confirm that the DataFrame looks similar to the image below (note that cities and countries may vary):

Graphical user interface

Description automatically generated

1. Export the DataFrame as a CSV file, and save it as WeatherPy\_Database.csv in the Weather\_Database folder.

In this module, you'll practice your analysis, visualization, and statistical skills by retrieving and analyzing weather data for a hypothetical travel company, PlanMyTrip. Successfully completing the tasks will draw on your knowledge of Python, decision and repetition statements, data structures, Pandas, and Matplotlib.

**What You Will Learn**

By the end of this module, you will be able to:

* Perform tasks using new Python libraries and modules.
* Retrieve and use data from an API "get" request to a server.
* Retrieve and store values from a JSON array.
* Use try and except blocks to resolve errors.
* Write Python functions.
* Create scatter plots using the Matplotlib library, and apply styles and features to a plot.
* Perform linear regression, and add regression lines to scatter plots.
* Create maps by using GeoViews and the Geoapify API.

**PLANMYTRIP** is a top travel technology company that specializes in internet-related services in the hotel and lodging industry. Jack is the head of analysis for the user interface team. He's asked you to help him collect and present data for customers via the search page, which they will then filter based on their preferred travel criteria in order to find their ideal hotel, anywhere in the world.

To perform this task, you will be using a Jupyter Notebook and the CitiPy module to get the cities for more than 500 random latitudes and longitudes. Then, you will perform requests on the OpenWeatherMap API and retrieve the JSON weather data from these cities. The weather data will be added to a Pandas data frame, where you will use Matplotlib to create a series of scatter plots to show the relationship between the latitude and a variety of weather parameters for over 500 cities around the world.

As part of your analysis, you will need to perform statistical calculations on the data using linear regression on the weather parameters in the Northern and Southern hemispheres. This data will help your team predict the best time of year for people to plan their vacation. Finally, you will export the data, clean it, and use the weather data to choose the best cities for vacation based on certain weather criteria, and then map these cities using GeoViews and the Geoapify API. Are you ready to help travelers find their ideal vacation spot? If so, pack your suitcase and let's get started.

At the most fundamental level, Jack needs help answering a question: How might we provide real-time suggestions for our client's ideal hotels? Your first task was to define what you meant by "ideal." So, over the course of the conversation, you narrowed that to hotels that were (1) within a given range of latitude and longitude and that (2) provided the right kind of weather for the client.

**Basic Project Plan**

Here's an outline of your project plan:

* **Task:** Collect and analyze weather data across cities worldwide.
* **Purpose:** PlanMyTrip will use the data to recommend ideal hotels based on clients' weather preferences.
* **Method:** Create a Pandas DataFrame with 500 or more of the world's unique cities and their weather data in real time. This process will entail collecting, analyzing, and visualizing the data.

Your analysis of the data will be split into three main parts, or stages.

1. **Collect the Data**
   * Use the NumPy module to generate more than 1,500 random latitudes and longitudes.
   * Use the citipy module to list the nearest city to the latitudes and longitudes.
   * Use the OpenWeatherMap API to request the current weather data from each unique city in your list.
   * Parse the JSON data from the API request.
   * Collect the following data from the JSON file and add it to a DataFrame:
     + City, country, and date
     + Latitude and longitude
     + Maximum temperature
     + Humidity
     + Cloudiness
     + Wind speed
2. **Exploratory Analysis with Visualization**
   * Create scatter plots of the weather data for the following comparisons:
     + Latitude versus temperature
     + Latitude versus humidity
     + Latitude versus cloudiness
     + Latitude versus wind speed
   * Determine the correlations for the following weather data:
     + Latitude and temperature
     + Latitude and humidity
     + Latitude and cloudiness
     + Latitude and wind speed
   * Create a series of maps using GeoViews and the Geoapify API that showcases the following:
     + Latitude and temperature
     + Latitude and humidity
     + Latitude and cloudiness
     + Latitude and wind speed
3. **Visualize Travel Data**

Create a map can display information on specific cities based on a customer's travel preferences. Complete these steps:

* + Filter the Pandas DataFrame based on user inputs for a minimum and maximum temperature.
  + Create a map for the new DataFrame.
  + Find a hotel from the cities' coordinates using the Geoapify API.
  + Store the name of the first hotel in the DataFrame.
  + Add additional information to each point in the map that display information about the city, current maximum temperature, and a hotel in the city.

The table below reviews the functions' outputs and limitations:

|  |  |  |
| --- | --- | --- |
| Function | Output | Limitation |
| randint(-90, 89) | Returns an integer between the interval, -90 and up to 89. | Will not generate a floating-point decimal number. |
| random() | Returns a floating-point decimal number between 0 and 1. | Will not generate a whole integer. |
| randrange(-90, 90, step=1) | Returns a whole integer between the interval, -90 and 90 where the step is the difference between each number in the sequence. | Will not generate a floating-point decimal number. |
| uniform(-90, 90) | Returns a floating-point decimal number between the interval, -90 and 90. | Will not generate a whole integer. |

NumPy module is a numerical mathematics library that can be used to make arrays or matrices of numbers.

An API call is very similar to navigating to a website. An API points to a URL and collects some data from the webpage or server.

Using an API has its limitations because not all information from a server is accessible. Most APIs have tiered services, from free to paid.

e'll use the API setup to go out and get information when our clients ask us for it. So, now it's time to download the Python Requests Library and register for an API key.