

Project focusing on Data Analysis and Visualization via Python Matplotlib & Third Party APIs

Data Bootcamp – Homework - Python APIs

kimberly GORDON

2018

Contents

Objectives

Requirements

Details

* + - Plot 1
    - Plot 2
    - Plot 3
    - Plot 4

Summary

Resources

Objectives

Build a series of scatter plots to showcase the relationship between

* Temperature (F) vs. Latitude
* Humidity (%) vs. Latitude
* Cloudiness (%) vs. Latitude
* Wind Speed (mph) vs. Latitude

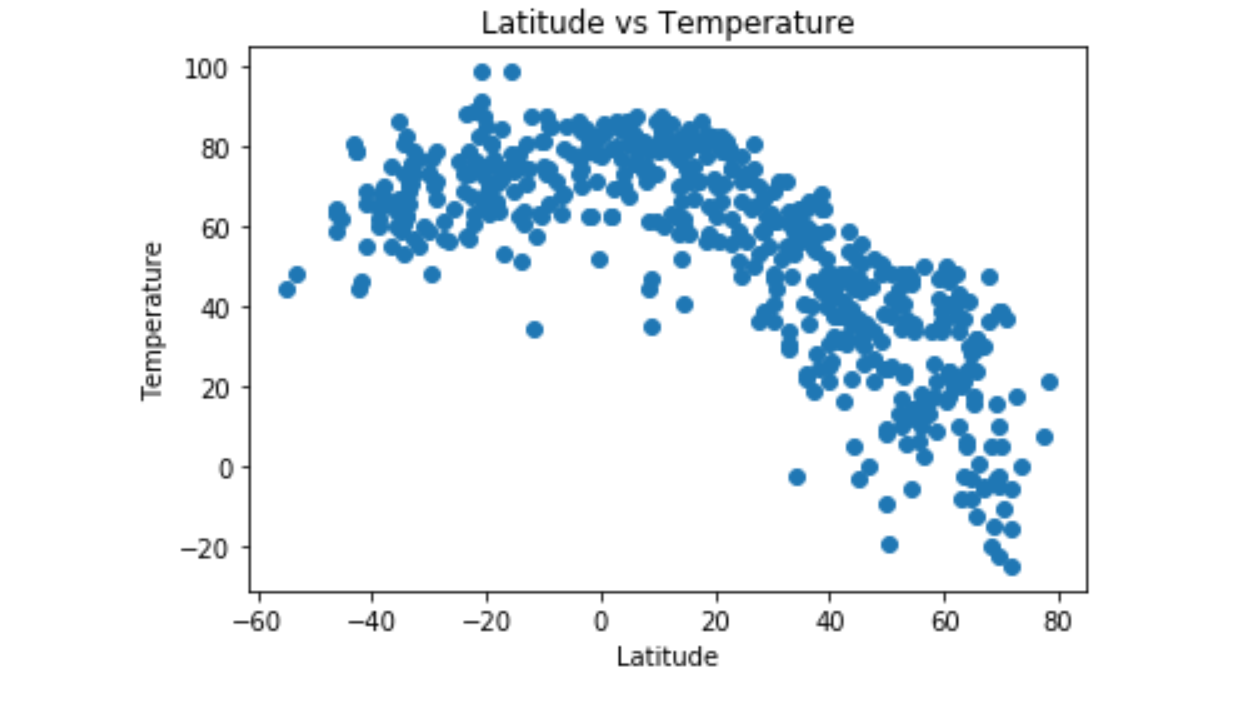
Include a written description of three observable trends based on the data.

Requirements (*Assumptions and Considerations)*

* Use the Pandas Requests, APIs, JSON traversals, Jupyter Notebook and Matplotlib library.
* Answer the question “What’s the weather like as we approach the equator?”
* Create Python script to visualize the weather of 500+ cities across the world of varying distances from the equator.
* Utilize Python library and OpenWeatherMap API
* Create a representative model of weather across the world cities
* See Example Solution (WeatherPy\_Example.pdf) for a reference on expected format.
* Randomly select at least 500 unique (non-repeat) cities based on latitude and longitude
* Perform a weather check on each of the cities using a series of successive API calls
* Include a print log of each city as it’s being process with the city number and city name
* Save a CSV of all data retrieved and png images for each scatter plot

Details

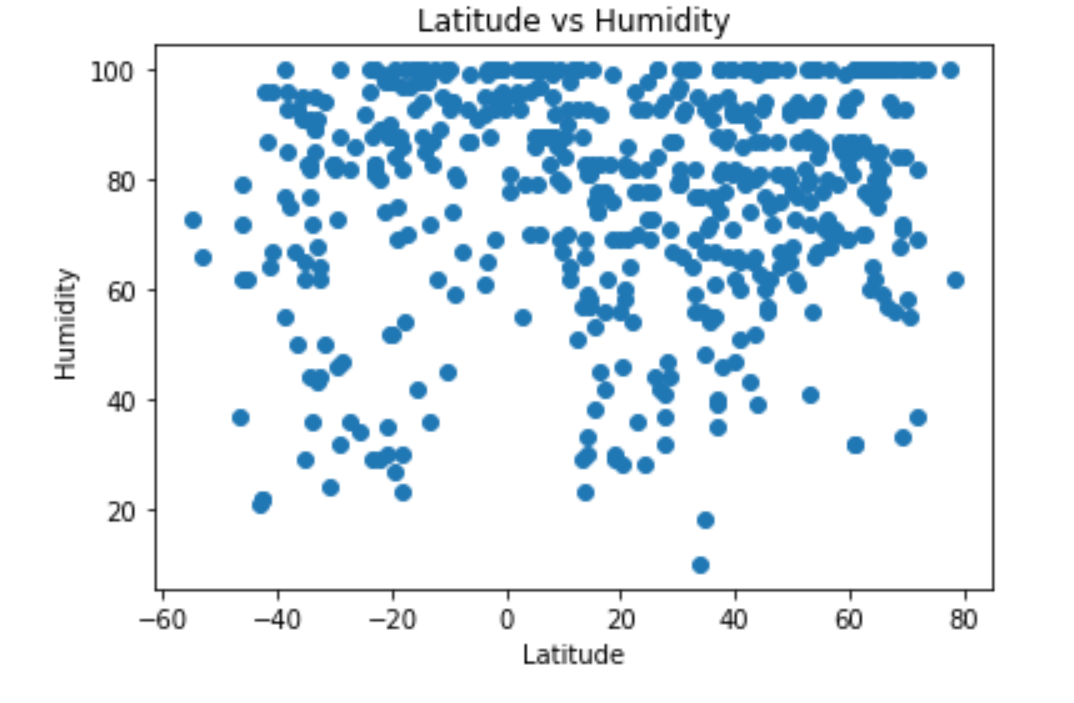
1. Temperature (F) vs. Latitude



The further away from the equator that a city resides, the less sunlight that city receives.

The relationship of temperature and latitude is as the temperature is warmer approaching the Equator and cooler approaching the Poles.

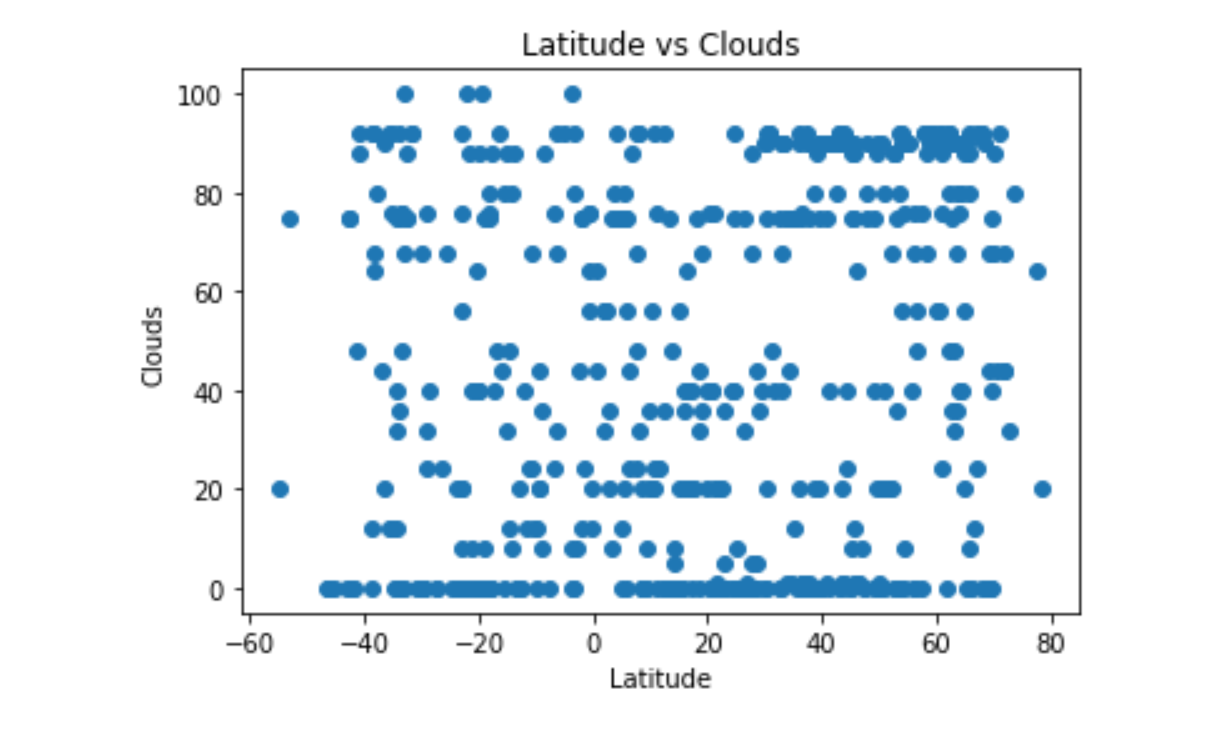
1. Humidity (%) vs. Latitude



When air temperature and dew point are close together relative humidity is high.

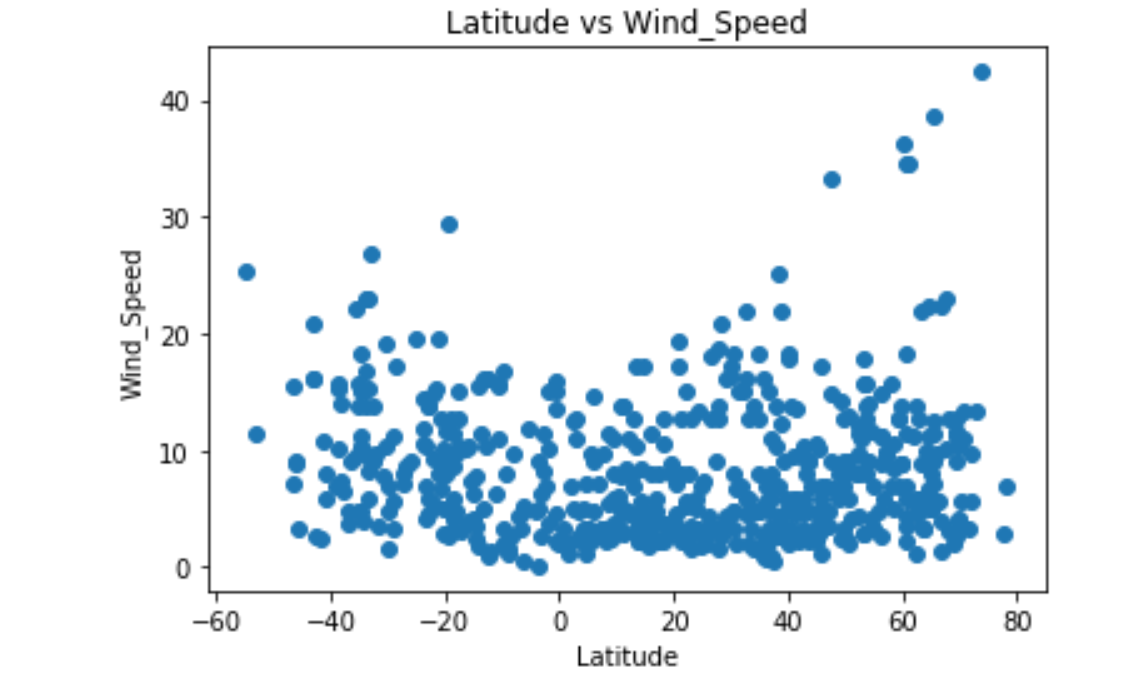
The relative humidity depends on both the water vapor content and the air temperature. The relative humidity is high near the equator because the actual vapor pressure is high. The relative humidity is high near the poles because the saturation vapor pressure is low.

1. Cloudiness (%) vs. Latitude



Cloudiness is suppressed at the equator due to equatorial ocean upwelling. At latitudes above 50 N, the annual average cloud cover is about 70%.

1. Wind Speed (mph) vs. Latitude



Wind speed is determined by the rate of the air pressure change, or gradient, between the two pressure areas. The greater the pressure difference, the faster the winds.

The trade winds, easterlies, blow near the equator between 30 degrees north latitude and 30 degrees south latitude. North of the equator, the trade winds blow from the northeast. Trade winds blow from the southeast south of the equator.

Summary

Citipy allows a user to have geolocation information on cities. Openweathermap API has information on the weather for cities based on their location. When using Citipy and Openweathermap API together, a user can quickly get the humidity, temperature, precipitation, cloud cover, wind speed, pressure quickly through a json call. See the “Details” section for scatter plots and interpretation of the results based on the latitude to the equator.

Terminology

Citipy

* Used to look for the nearest city with geo coordinates

Latitude

* Measurement of the distance of a location on the Earth from the equator
* One of the primary factos affecting the unequal heating of the Earth’s atmosphere

Openweathermap API

* Service that provides weather data, including current weather data, forecasts and historical data to developers of web services and mobile applications
* Provides an API with JSON, XML and HTML endpoints
* Users can request weather information, extended forecasts and graphical maps (showing cloud cover, wind speed, pressure and precipitation)

|  |
| --- |
| * # Python program to find current * # weather details of any city * # using openweathermap api * # import required modules * import requests, json * # Enter your API key here * api\_key = "Your\_API\_Key" * # base\_url variable to store url * base\_url = "<http://api.openweathermap.org/data/2.5/weather?>" * # Give city name * city\_name = input("Enter city name : ") * # complete\_url variable to store * # complete url address * complete\_url = base\_url + "appid=" + api\_key + "&q=" + city\_name * # get method of requests module * # return response object * response = requests.get(complete\_url) * # json method of response object * # convert json format data into * # python format data * x = response.json() * # Now x contains list of nested dictionaries * # Check the value of "cod" key is equal to * # "404", means city is found otherwise, * # city is not found * if x["cod"] != "404": * # store the value of "main" * # key in variable y * y = x["main"] * # store the value corresponding * # to the "temp" key of y * current\_temperature = y["temp"] * # store the value corresponding * # to the "pressure" key of y * current\_pressure = y["pressure"] * # store the value corresponding * # to the "humidity" key of y * current\_humidiy = y["humidity"] * # store the value of "weather" * # key in variable z * z = x["weather"] * # store the value corresponding * # to the "description" key at * # the 0th index of z * weather\_description = z[0]["description"] * # print following values * print(" Temperature (in kelvin unit) = " + * str(current\_temperature) + * "\n atmospheric pressure (in hPa unit) = " + * str(current\_pressure) + * "\n humidity (in percentage) = " + * str(current\_humidiy) + * "\n description = " + * str(weather\_description)) * else: * print(" City Not Found ") |

Copy CodeRun on IDE

**Output :**

Enter city name : Delhi

Temperature (in kelvin unit) = 312.15

atmospheric pressure (in hPa unit) = 996

humidity (in percentage) = 40

description = haze

Resources

Skills Developed

* Programmatically obtain and parse data from sources such as OpenWeatherMap, the US Census and OMDB.
* {lot Data Frames from the API data using Matplotlib.
* Make GET requests with `requests`.
* Convert JSON into a Python dictionary.
* Apply API specifications described in various API driven data bases, i.e. US Census.
* API work included:
  + signing up for and using an API keys
  + creating applications from scratch using knowledge of Python and an API documentation
  + loading JSON from API responses into a Pandas DataFrame
  + using `try` and `except` blocks to handle errors
  + using Google Maps and Places API to obtain information about geographic areas
  + using the Census API wrapper
  + understanding - implementing the concept of rate limits and the importance of creating "test cases" prior to running large scripts
  + understanding how to dissect new API documentation

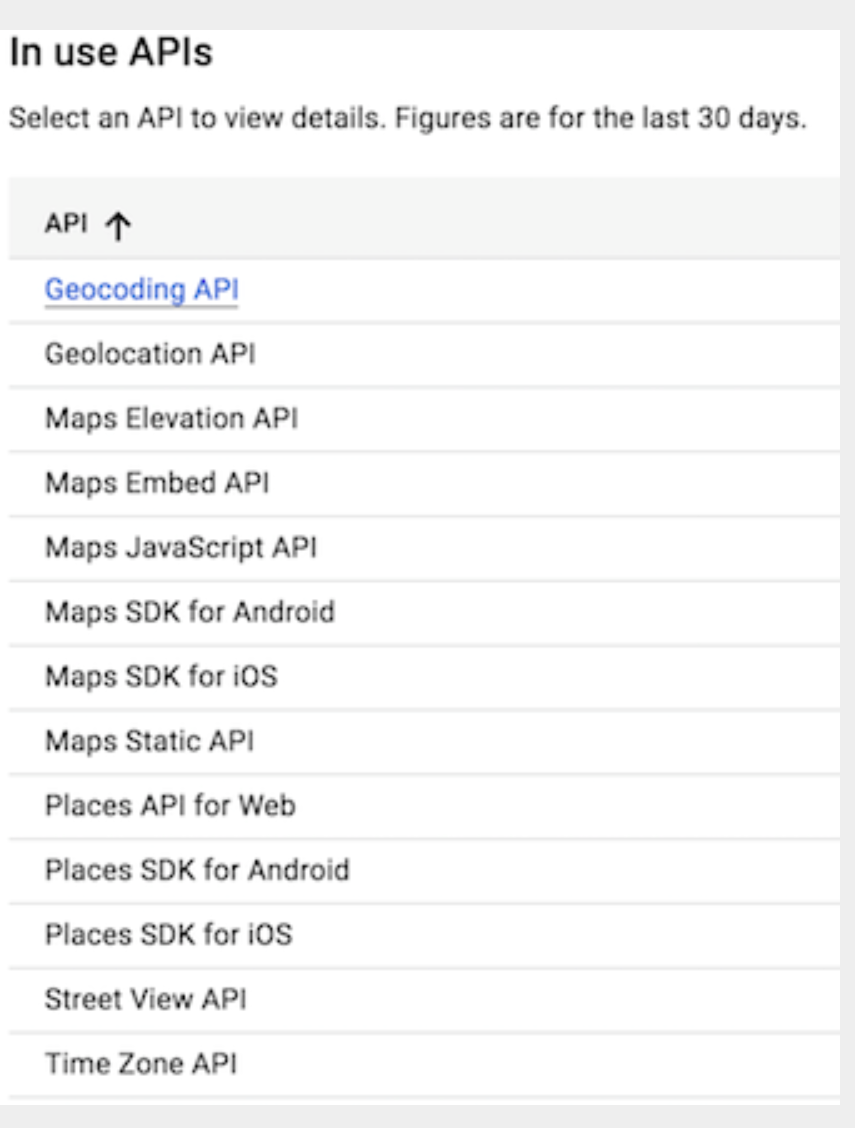
Links

* Geographic coordinate system (<http://desktop.arcgis.com/en/arcmap/10.3/guide-books/map-projections/about-geographiccoordinate-systems.htm>
* citipy Python library (https://pypi.python.org/pypi/citipy)
* [JSON Testing](https://jsonplaceholder.typicode.com/)
* [OMDb API](<http://www.omdbapi.com/>)
* [The New York Times API](<https://developer.nytimes.com/>)
* [Open Weather Map API](<http://openweathermap.org/api>)
* [The World Bank API](<http://api.worldbank.org/>)
* [Google Maps API](https://developers.google.com/maps/)
* Numpy <http://www.numpy.org/>
* Matplotlib Gallery <https://matplotlib.org/gallery.html>
* Pandas Plotting <https://pandas.pydata.org/pandas-docs/stable/visualization.html>
* Alternatives to Pie Charts <http://www.storytellingwithdata.com/blog/2014/06/alternatives-to-pies>
* Creating a Twitter Dev Account Supplemental/Dev\_Account\_Creation.pdf

Supporting Information

* Far Far Away Collect information from the Star Wars API. [Watch the Video](https://youtu.be/3I\_3dkO\_604)
* Bujumbura - Get the current temperature in Bujumbura. [Watch the Video](<https://youtu.be/dF56b0tM0Kg>)
* Bank Deserts Answer the question: "What is the relationship between poverty, age, and population with the number of banks in a given area?" [Watch the Video](<https://youtu.be/RH9Ca0-OUtc>)

APIs



Capping Queries [Setting Google API Daily Usage Caps]

As of June 16, 2018, Google is adjusting their pricing for the Google Maps Platform to require a credit card on file. They do charge for the platform, but provide a $200 credit for those services.

To avoid an unexpected increase in usage limits, the following walk-through will assist with setting up a maximum daily billable limit for your API calls.

1. Visit the [Google Dashboard](https://console.cloud.google.com/projectselector/apis/dashboard?\_ga=2.166576887.87610002.1528936935-349962923.1526573906&supportedpurview=project).

2. Select a project (if prompted).

![Select Project](Images/Select\_Project.png)

3. Once you have selected a project, select an API from your API list and navigate to the quotas tab.

![Select API](Images/API\_List.png)

![Quotas](Images/Quotas.png)

4. Finally, click the edit icon next to the "requests per day" quota to make the necessary changes.

![Edit Icon](Images/Edit\_Icon.png)

\* **\*\*Note\*\***: These steps will need to be repeated for each API in use.

- - -

Full documentation regarding the pricing change can be found within this [Google FAQ] (https://developers.google.com/maps/faq).

To set up a budget and/or billing alerts, visit the [Google Cloud Billing Documentation](https://cloud.google.com/billing/docs/how-to/budgets).

Setting up plots

1. How does the DataFrame need to be structured to have the right X and Y axis?

dataframe=pd.DataFrame(example, columns =[‘Height’, ‘Latitude’])

* + 1. In the series within the panda DataFrame, assign x = Height and y = Latitude.
       - X = dataframe.Height
       - Y = dataframe.Latitude
    2. Plot them as a scatter chart by adding:
       - plt .scatter(x,y)
    3. Show or save the chart with the following:
       - plt.show()
       - plt.savefig(“chart-name.png”)

1. How do I build a basic scatter plot?

Where s is the size of the dots.

Import matplotlib.pyplot as plt

plt.title(“Scatter plot”, fontsize=24)

x\_values = [2, 5, 8, 11, 14]

y\_values = [2, 6, 10, 14, 18]

plt.scatter(x\_values, y\_values, s=100)

plt.show()

1. How do I add a label to that scatter plot?

plt.xlabel(“Height (m)”, fontsize=20)

plt.ylabel(“Latitude”, fontsize=20)

Font size of the number labels on the axes

plt.xticks(fontsize=16)

plt.yticks(fontsize=16)

1. Where would the labels for that scatter plot originate?

The labels are identified by xlabel, ylabel, etc., in the code along with the fontsize

Python script to visualize the weather of 500+ cities across the world of varying distance from the equator. To accomplish this, you'll be utilizing a simple Python library, the OpenWeatherMap API, and a little common sense to create a representative model of weather across world cities.

Your objective is to build a series of scatter plots to showcase the following relationships:

Temperature (F) vs. Latitude

Humidity (%) vs. Latitude

Cloudiness (%) vs. Latitude

Wind Speed (mph) vs. Latitude

Your final notebook must:

Randomly select at least 500 unique (non-repeat) cities based on latitude and longitude.

Perform a weather check on each of the cities using a series of successive API calls.

Include a print log of each city as it's being processed with the city number and city name.

Save both a CSV of all data retrieved and png images for each scatter plot.

As final considerations:

You must complete your analysis using a Jupyter notebook.

You must use the Matplotlib or Pandas plotting libraries.

You must include a written description of three observable trends based on the data.

You must use proper labeling of your plots, including aspects like: Plot Titles (with date of analysis) and Axes Labels.

See Example Solution for a reference on expected format.

Hints and Considerations

You may want to start this assignment by refreshing yourself on the geographic coordinate system.

Next, spend the requisite time necessary to study the OpenWeatherMap API. Based on your initial study, you should be able to answer basic questions about the API: Where do you request the API key? Which Weather API in particular will you need? What URL endpoints does it expect? What JSON structure does it respond with? Before you write a line of code, you should be aiming to have a crystal clear understanding of your intended outcome.

A starter code for Citipy has been provided. However, if you're craving an extra challenge, push yourself to learn how it works: citipy Python library. Before you try to incorporate the library into your analysis, start by creating simple test cases outside your main script to confirm that you are using it correctly. Too often, when introduced to a new library, students get bogged down by the most minor of errors -- spending hours investigating their entire code -- when, in fact, a simple and focused test would have shown their basic utilization of the library was wrong from the start. Don't let this be you!

Part of our expectation in this challenge is that you will use critical thinking skills to understand how and why we're recommending the tools we are. What is Citipy for? Why would you use it in conjunction with the OpenWeatherMap API? How would you do so?

In building your script, pay attention to the cities you are using in your query pool. Are you getting coverage of the full gamut of latitudes and longitudes? Or are you simply choosing 500 cities concentrated in one region of the world? Even if you were a geographic genius, simply rattling 500 cities based on your human selection would create a biased dataset. Be thinking of how you should counter this. (Hint: Consider the full range of latitudes).

Lastly, remember -- this is a challenging activity. Push yourself! If you complete this task, then you can safely say that you've gained a strong mastery of the core foundations of data analytics and it will only go better from here. Good luck!