Astronomy to Astrology: Project Description

By Chaimi Lee

Motivation

As someone with a background in Physics, I wanted to work with data related to physical sciences. While researching the datasets that I could possibly use for my project, I got the idea to integrate my interests in astronomy with astrology, something that a lot of my friends follow. Although the two are very contrasting fields, I thought it would be interesting to find a common ground between hard science and the trendy pseudoscience. Therefore, I decided to write a program that can tell the user their horoscope based on astronomical data on the web as my final project.

Description of Data Sources

Data 1: worldcities.csv (source: https://www.kaggle.com/juanmah/world-cities)

The 'worldcities.csv' dataset is a csv file that contains information of 41,001 cities all over the world. This information includes the country and population, but more importantly, the latitude and longitude of these cities, which I need for my second dataset, a Visible Planets API.

Data 2: visible-planets-api (source: https://github.com/csymlstd/visible-planets-api)

The visible-planets-api uses the url https://visible-planets-api.herokuapp.com/v2?latitude=32&longitude=-98 that, by default, has a latitude value of 32 and a longitude value of -98. But, if I input a latitude and longitude value for a city that I would like to get information about, I can replace the values in the url with the latitude and longitude of said specific city and get information that way. The API returns a json of planets above the horizon (including the moon) at the location and time of querying (in GMT).

Data 3: Astrology.com (source: https://www.astrology.com)

The Astrology.com website's homepage has a category called "learn", which has a subpage on planets. The 'Planets' subpage (*https://www.astrology.com/planets*) has astrological information on all of the planets including the Sun, Moon, and Pluto. For my project, I have scrapped data from this specific page in order to provide astrological information to my program's user.

Analysis/Task Performed

For this project, I wanted to create a program that performs a task of interacting with its user and tells them a very simplified version of their horoscope based on their location, the time of querying, and the planets that are above the horizon in their location. The idea here is that if a planet is above the horizon, the user will feel the astrological impact of those planets. The user can interact with the program and tell them which planets they would like to get astrological information about. They can learn about as many planets as they wish.

When the user inputs a city name that they are located in (or want to get information about), the program accesses the pandas dataframe that contains information that it read from 'worldcities.csv', and uses it to

find the latitude and longitude of that city. The program then uses that information to access the visible-planets-api by using the requests library and the link that is given above. By doing so, the program can get a json of all of the planets that are above the horizon in that location at the time of querying in GMT. The program takes that json, extracts the name of the planets that are above the horizon, and puts them in a list. The program returns the list to the user and asks them which planet they would like to learn the astrological significance of. When the user inputs the name of a planet in the list, the program returns text about the planet scraped from the astrology.com website. The program then asks the user if they would like to know more about a different planet, which the user can answer 'yes' or 'no' to. The user can learn about as many of the planets as they wish that are present in the list of planets.

Issues I Ran Into

The first issue I ran into was working with bad inputs, such as spelling errors (for both cities and planets) or answering anything other than 'yes' or 'no' to my program's question when it asks if the user would like to know about additional planets. I worked around this problem by using try/except statements and a lot of while loops. For example, I have multiple while loops throughout the program that looks like this:

```
 while more_info.lower()!='yes' and more_info.lower()!='no':
   print('Please answer yes or no')
   more_info=input('\nWould you like to know more about a different
planet? ')
```

By having this loop, I can make sure the program tells the user to answer 'yes' or 'no' to my question and does not proceed or cause an error until the user gives an acceptable answer.

Another issue I ran into was when I tried to create a dictionary with my 'worldcities.csv'. I originally tried to create a dictionary with its keys as the name of the cities and their values as the city's latitude and longitude. However, I noticed that some of the values were getting messed up and random values were getting placed in the dictionary. I fixed this issue by, instead of working with `import csv`, working with pandas dataframe.

Things to Note

There are no special libraries that are necessary. Make sure you have these ones imported before running the program:

```
import pandas as pd
from bs4 import BeautifulSoup
import requests
import json
import sys
import sqlite3
from time import gmtime, strftime
```

```
import time
```

Also, as mentioned above, the program will require you to put in valid inputs. Please make sure to spell any planet's name correctly and also capitalize the first letter of each planet's name. The city names do not need to be capitalized (you can if you wish), but make sure you spell it correctly. Same goes for the 'yes' or 'no' inputs. The print statements that appear throughout the program will specify the input requirements. Make sure you read them!

Also note that I understand that Astrology doesn't just mean that if an object is present in the sky, that object determines a person's horoscope. I personally do not follow astrology, but I am aware that there are many layers to what certain celestial objects and phenomena mean in the field. As I have limited knowledge and resources, my idea of how astrology works for this project will be extremely rudimentary.

I have also created 4 static datasets, but use 3 of them for my final project. The three datasets are 'worldcities.csv' file, 'five_cities.json', and 'final_project.db'. I also had to create a json containing the information about each of the planets that I scrapped from the astrology website in order to create the 'final_project.db'. Make sure you have all 4 of these static datasets available.

Conclusion

In conclusion, I discovered that working with certain data frames works best for particular tasks or analysis. For example, it was extremely useful for me to work with a Pandas dataframe while pulling information from 'worldcities.csv' as opposed to the csv library. Another thing to note was that using a json for my static dataset was more useful and straightforward than any other data frames in terms of storing my data to use it during my static mode. This just has to do with how my program is built and how json does a good job of storing lists. I also came up with the conclusion that statements, such as try/except or if/else statements, as well as loops and booleans are extremely useful when building a user interface that includes inputs. This has helped me avoid numerous possible errors or code breaks.