Homework: Python and Web Scraper

Web-Scraping to Compare Pennsylvania Ski Resorts

Carol Moore, cm6ag@virginia.edu, March 14, 2021

Overview and Utility

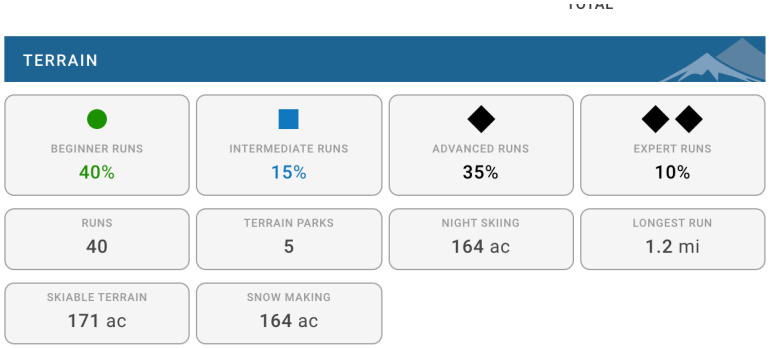
I collected and analyzed information about Pennsylvania ski resorts, with the aim of classifying and sorting resorts based on categories such as "most beginner friendly." Although many skiers and snow boarders stick to a home resort or prioritize destinations like Vail, others are routinely on the lookout for the next experience to safely challenge their skills. There are many websites that report data on resorts, but they often lack tools to sort resorts across some category of interest. My ultimate goal would be to classify resort terrain, and even particular runs, in a way that skiers might use to either plan trips or set goals about where to go in the future. However, this project really only exercises the idea at a very basic level.

Data

The website On the Snow (https://www.onthesnow.com/) contains detailed information on over 2,000 ski resorts around the world. After selecting a country or state, the user can then see current conditions at all the resorts in that region (a partial list is shown below for Pennsylvania).

Resort Name	Status	New Snow	Base Depth ▾ *Lower/Upper*	Lifts Open *Open/Total*	Open Acreage	Weather	Cams
Blue Mountain Resort Pennsylvania, USA Last Updated: 03/13 Compare	●	24 HR: 0" 72 HR: 0"	42" - 60" Full Report	7 / 16	171ac		
Elk Mountain Ski Resort Pennsylvania, USA Last Updated: 03/14 Compare	●	24 HR: 0" 72 HR: 0"	30" - 60" Full Report	7 / 7	N/A		
Shawnee Mountain Ski Area Pennsylvania, USA Last Updated: 03/10 Compare	●	24 HR: 0" 72 HR: 0"	36" - 60" Full Report	3 / 9	100ac		

Skiers and boarders can get more information about a specific resort by clicking on a link in the column titled *Resort Name*. The user is then sent to a child page that reports terrain and other features of the resort. In this project, I scraped the terrain data on the child pages of each of the 19 resorts listed for Pennsylvania. A screenshot of terrain characteristics I scraped, as rendered on the child page for Blue Mountain, is shown below.



Approach and Libraries

The Python code relies on the Beautiful Soup library to scrape the websites (child pages). I followed closely the procedure outlined in a Geeks for Geeks article on scraping https://www.passiton.com/inspirational-quotes [1]. The basic steps are

1. Request access to the child page and receive source code– requires importing the Request library.
2. Find the relevant data on the page. In addition to reviewing the structure of the page from *soup.prettify()*, I used Developer Tools in google locate the terrain data on each page.
3. Extract the terrain data using *terrain = soup.find('div', attrs = {'id':'resort_terrain'}),* where *terrain* is a bs4.element.Tag
4. Iterate through *terrain* to find the text on each row.
5. Write the page's data to a dictionary. The dictionary for the child page shown above is:

```
{'Name': 'blue-mountain-ski-area',
 'BeginnerRuns%': '40',
 'IntermediateRuns%': '15',
 'AdvancedRuns%': '35',
 'ExpertRuns%': '10',
 'Runs': '40',
 'TerrainParks': '5',
 'NightSkiingac': '164',
 'LongestRunmi': '12',
 'SkiableTerrainac': '171',
 'SnowMakingac': '164'}
```

I ran into two main problems executing these steps.

- I had intended to collect all of the URL for the child pages by scraping the parent page for Pennsylvania using methods to find links on a page, like those outlined in [2]. However, I was not able to find any of the child page links in the source code. Instead, I pasted each URL into a text file, and the Python code applies the steps above to each URL.
- I was not able to fully separate the strings on each row of *terrain* using Beautiful Soup. I relied on standard string manipulation tools (e.g., in [3]) to separate numbers from alpha characters and ensure that units of measurement, such as percentage signs, were represented. One field in the dictionaries, *Longest Run,* was consistently inaccurate if the data included a decimal point (in our example, the 12-mile run at Blue Mountain is actually 1.2 miles). I deleted this field from the final dataset.

The Python code applies steps 1-4 iteratively to each child page and builds a list (resortlist) made up of the dictionaries. I used the CSV library, which has a dictionary-writing capability, to write the list and dictionaries to a CSV file. Finally, I analyzed the data using Pandas. Pandas commands include checking for problematic data (percentage of beginner, intermediate, advanced, and expert runs not summing to 100; all of a certain type of run being greater than 90%, which is implausible). These checks resulted in dropping one observation, using methods presented in [4] the count of dropped observations was based on code in [5].

Some findings

The code produces average number of runs by ability level and starts to classify the resorts in terms consumers might look for: the most terrain parks; most Beginner's runs; and sorted rankings of these

attributes. An example print out from Pandas is shown below. Blue Mountain (which coincidentally has been our example throughout the report) has been deemed most "beginner-friendly". This is based on number of beginner's runs, calculated by multiplying scraped values for Beginner% and Runs (which gives the total number of runs). Although Eagle Rock has the greatest percentage of runs at the beginner level, Blue Mountain has the greatest number.

```
Most beginner friendly:  0    blue-mountain-ski-area
Name: Name, dtype: object
Number of beginner runs:   16.0
                          Name  BeginnerRuns%  Runs  num_beginner_runs
0       blue-mountain-ski-area             40    40               16.0
7    camelback-mountain-resort             39    39               15.0
6                seven-springs             36    33               12.0
17                sno-mountain             35    26                9.0
4   bear-creek-mountain-resort             30    23                7.0
13                     liberty             33    21                7.0
15                  eagle-rock             50    14                7.0
3         hidden-valley-resort             27    26                7.0
10            whitetail-resort             29    22                6.0
18                   big-bear              33    18                6.0
1      elk-mountain-ski-resort             22    27                6.0
2    shawnee-mountain-ski-area             26    23                6.0
12    roundtop-mountain-resort             26    19                5.0
8                  ski-sawmill             38    13                5.0
16             tussey-mountain             27    15                4.0
14                 big-boulder             38     8                3.0
5     spring-mountain-ski-area             37     9                3.0
9   mount-pleasant-of-edinboro             22    10                2.0
```

Ways to extend program beyond original features

The tool could be expanded to include regions other than Pennsylvania, allowing skiers in regions like DC, which have access to multiple states, to make better comparisons. More ambitiously, I would want to perform additional research on the true difficulty of each of the runs in the dataset. The ski industry does not have an absolute scale for measuring difficulty of a given terrain; it differs by geographic region and resort. User feedback on difficulty, or ideally, true geographic data, would be enhancements for skiers planning trips.

Coding References

[1] Geeks for Geeks, "Implementing Web Scraping in Python with Beautiful Soup," blogpost, Last Updated 20 Aug 2020. Last accessed 3/14/2021 at https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/

[2] Crummy.com, "Beautiful Soup Documentation." Last accessed 3/14/2021 at https://www.crummy.com/software/BeautifulSoup/bs4/doc/index.html?highlight=row

[3] Stack Overflow, "Removing Numbers from a String", various posts. Last accessed 3/14/2021 at https://stackoverflow.com/questions/12851791/removing-numbers-from-string

[4] Pandas Documentation, "pandas.DataFrame.drop", documentation. Last accessed 3/14/2021 at https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.drop.html#pandas-dataframe-drop

[5] Data Independent, "Pandas Number Of Rows – 6 Methods", blogpost, 14 Sept. 2020. Last accessed 3/14/2021 at https://www.dataindependent.com/pandas/pandas-number-of-rows/