Scrapers for Government COVID-19 Measures

Overview

This collection of web scrapers collects the information pertaining to government COVID-19-related emergency orders and stores them in CSV files.

Design Details

Implementation

This collection of web scrapers uses the BeautifulSoup 4 and requests libraries in order to collect the text from each state in the New England region of the United States. After collecting the text, there are generally 3 elements of information that are scraped. These elements are:

1. Date, or the day that the measure was passed by the government.
2. Description, or the supplied summary of the measure.
3. PDF Link, or the source to the primary text the order.

These elements of information were available and scraped for all 6 of the states in the New England region. The csv files contain these attributes in their columns and the emergency orders in their rows, sorted from most recent to least recent date. Sometimes, more information was available. For example, Vermont and Massachusetts offered Press Release article links and Guidance links. I included these in their respective csv files.

Pre-processing

In order to ensure consistency between files, the dates and descriptions had to be pre-processed. The dates were written in a variety of forms in the webpages. Using the datetime library, all of these dates were rewritten into mm/dd/yyyy format.

The descriptions generally needed more intricate processes to extract only the supplied text summaries for the government orders. A common theme across all of the government websites was inconsistent formatting, across both html tabs and the text within.

The inconsistent html tabs made me innovate solutions that rely less on positioning of tabs in the contents, and cater more to the relative structure of the contents. This approach is perhaps more robust to inconsistencies, but could still break if contents are rearranged.

The text was cleaned and pre-processed on a case-by-case basis. For example, the regex library was used for scraping Connecticut's website to eliminate excess spaces in the text. Many of the websites also needed their descriptions to be split to only include the description-associated text. For example, I was able to split off any text not related to the emergency order by looking for rare characters such as colons or parentheses, and breaking the text there. This will generally work, unless the website places one of these characters in the body of a summary.

Analysis

1. Connecticut featured a consistent use of html tabs and text formatting on its website. In effect, it this scraper was able to efficiently scrape the important information. The dates were converted from "Month Day, Year" format to mm/dd/yyyy format. The description text was stored in different elements, so it was necessary to grab text from each one and combine them into a single description. Some of the text contained extra whitespace, therefore regex was used to clean the text Sometimes periods were added to the end of the description in order to simulate grammatically correct sentences.
2. Massachusetts was easily the most data-intensive website. It contains not only the Dates, Descriptions, and Order PDF Links, but also Press Release Links and Guidance Links. I stored these as separate columns in the csv files in case they could be useful. The html tabs were predominantly consistent, however the text formatting featured inconsistent uses of colons and periods to separate the date text from the description text. The colons were exclusively used in these scenarios, making it convenient to convert them to periods then split the text on periods to obtain just the description. The dates were converted from "Month Day" format to mm/dd/yyyy format. Since there were different types of links being stored in the same list elements, it was necessary to check which type of link was being stored before adding it to the csv. This is done by checking if order, guidance, or press are contained in the link description before adding the link. In addition, one minor inconvenience about this website was that the first COVID order was listed at the top of the page, even though the rest were in time order throughout the body of the page, thus calling for extra code to scrape a single order.
3. Vermont followed Massachusetts in terms of its page layout. It similarly contains Dates, Descriptions, and Order PDF Links, Press Release Links and Guidance Links. The html tabs and text formatting were mostly consistent. Vertical bars were uniquely used to separate the date text from the description text, therefore it was easy to split the text on that unique character and save in the csv file. The dates were converted from "Month Day" format to mm/dd/yyyy format. Again, since there were different types of links being stored in the same list elements, it was necessary to check which type of link was being stored before adding it to the csv. This is done by checking if order, guidance, or press are contained in the link description before adding the link. The Order PDF Links were a bit more complicated than other scrapers because it was required to visit a second embedded webpage and scrape the link there, thus adding to the complexity of the scraper.
4. Rhode Island was written in very plain html format, thus making it easy to scrape. The description and text were located in the list item tabs on the page and separately extracted to the csv. The date was in the desired format, so there was no need for the datetime package. Since Rhode Island hosts all of their government orders on a single page, the scraper exits once it encounters an order before that order. On the other hand, the description text needed to be split on the hyphen to only extract text summarizing the order, and not the order name. It then needed to be split again by a

parenthesis to break off any of the text from the date. Overall, this was a relatively lightweight scraper for this plain html webpage.

5. New Hampshire had the most troubling webpage for the scraper. The reason for this is because it did not include the day that the government orders were issued, but only the month and year. For this reason, I think this file may not be as valuable as the others and needs to have the dates extracted from the accompanying PDFs before any data analysis or learning algorithms can be performed. I still included the scraper and output as a benchmark. Otherwise, it was similar to Rhode Island's html webpage and presented no major html issues. The description text needed to be split on a parenthesis in order to only grab the summary.
6. Maine has the least amount of information available in its summaries, which also raises the need for the accompanying PDFs to be scraped for additional insight. I will put this request into the future considerations section of the repository. As for the html, it was relatively simple like the previous scrapers. The description had to be split on a parenthesis to avoid the date and then the first three words were removed since they contained the order title. Since all of their government orders were on this page, the scraper exits once it encounters an order before the first order pertaining to COVID.

Testing

By visual inspection, I verified that the scripts are collecting most of the desired data from the websites. Link were checked to be working.

Requirements

The provided collection of scripts are written in Python 3 and uses the BeautifulSoup 4, requests, datetime, csv, and regex libraries.

How to Run

Run the scrapers by running the bash-script on the command-line as follows:

```bash run_scrapers.sh```

Look for the output in the /Outputs folder. I provided data up until Saturday, April 25, 2020 in there already.

Future Considerations

To anyone interested in going forward with this project, I would recommend the following tasks:

1. Implement a PDF to text translator for the PDFs linked in the csv outputs. Store the text in a new column in the csv files
2. Create web scraping scripts for government measures taken by the other 44 states.
```