Working with Python – Crawl and Scrape – Handout

Digital Management – November 2020

This is an introduction to web crawling and scraping. Please try to read it and test the code using Anaconda/Spyder before the Hands-on session.

1. Requesting information from the web

1.1. Python 'requests' module.

- This module provides functions to send a HTTP request and get the response from the server.
- Requests is a third-party module. If not installed, we will need to do "$pip install requests" in the mac terminal or in the command prompt of windows. (normally, you should have it already installed if you are using Anaconda / Spyder)
- For more details about requests, see: https://requests.readthedocs.io/en/v0.8.2/

1.2. Using 'requests' module.

- Use the requests module to make a HTTP request to http://www.tripadvisor.com
- Check the status of the request (should return 200)
- Display the response status information or a message to signal an error

Python code (test it with Anaconda/Spyder)

```
import requests
url = 'http://www.tripadvisor.com/'
response = requests.get(url)

if response.status_code == 200:
        print(response.status_code)
else:
        print('Failed to get a response from the url. Error code: ' , response.status_code )
```

2. Scraping websites

Sometimes, you may want a little bit of information-a movie rating, stock price, or product availability-but the information is available only in HTML pages, surrounded by ads and extraneous content.
To do this we build an automated web fetcher called a crawler or spider. After the HTML contents have been retrieved from the remote web servers, a scraper parses it to find the information that you need.

2.1. BeautifulSoup (BS) Module

The BS module can be used for searching a webpage (HTML file) and pulling required data from it. It does three things to make a HTML page searchable:

- First, converts the HTML page to Unicode string.

- Second, parses (analyses) the HTML page using the best available parser. It will use an HTML parser unless you specifically tell it to use an XML parser.
- Finally transforms a complex HTML document into a complex tree of Python objects.

This module takes the HTML page and creates four kinds of objects: Tag, NavigableString, BeautifulSoup, and Comment:

- The BeautifulSoup object itself represents the webpage as a whole.
- A Tag object corresponds to an XML or HTML tag in the webpage.
- The NavigableString contains the bit of text within a tag.

For detailed information about BeautifulSoup, go to
https://www.crummy.com/software/BeautifulSoup/bs4/doc/

2.2. Step 1: Making the soup

First, we need to use the BeautifulSoup module to parse the HTML data into Python readable Unicode Text format. Let us write the code to parse an html page. We will use the trip advisor URL for Le Jardin Napolitain:
https://www.tripadvisor.com/Restaurant_Review-g227613-d3531819-Reviews-Le_Jardin_Napolitain-Jouy_en_Josas_Versailles_Yvelines_Ile_de_France.html

Python code (test it with Anaconda/Spyder) – first delete the previous lines from 1b

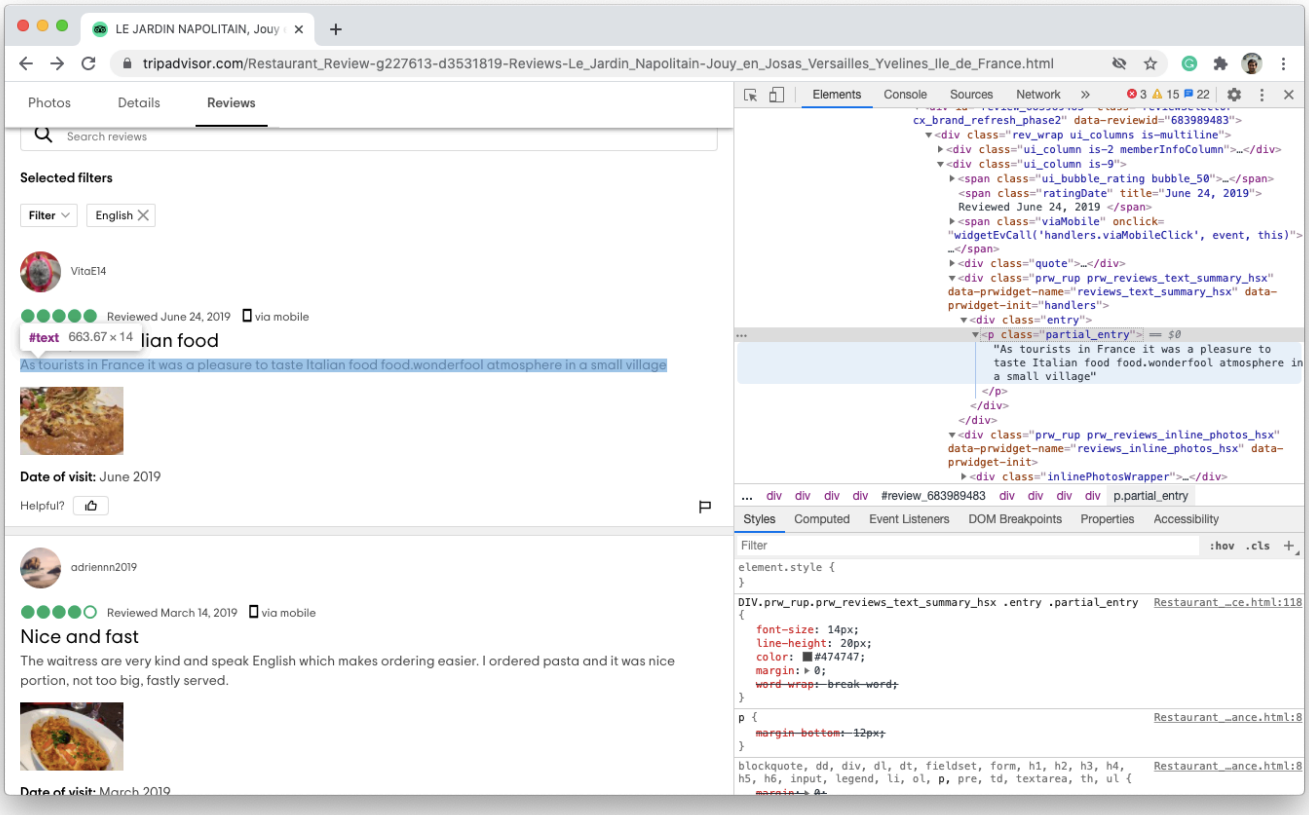
```
import requests
from bs4 import BeautifulSoup

scrape_url = 'https://www.tripadvisor.com/Restaurant_Review-g227613-d3531819-Reviews-
Le_Jardin_Napolitain-Jouy_en_Josas_Versailles_Yvelines_Ile_de_France.html'

response = requests.get(scrape_url)
print(response.status_code)

if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser') # Here you make the Soup!
        print(soup) # huge amount of data here, print once to see how it looks
```

2.3. Step 2: Inspecting the element you want to scrape

To inspect the elements in a TripAdvisor webpage, you can:

1. Go to the TripAdvisor html page for Le Jardin Napolitain, right-click on a review and select Inspect.
2. You will see a new window on the top right of the page with a blue-highlighted section. This is the HTML code of the page. Right-click on the highlighted section, select 'Copy'.

3. Then ‘Copy element’, and paste what you got in a word file in order to see it. You get:
   `<p class="partial_entry"> As tourists in France it was a pleasure to taste Italian food food.wonderfool atmosphere in a small village</p>.`
   We can see that the tag for this review is `<p>` (paragraph) with attribute “class” and attribute value “`partial_entry`”. We will use this information in the next section.



2.4. Step 3: Searching the soup for the data

Beautiful Soup defines a lot of methods for searching the parse tree (soup), the two most popular methods are: `find()` and `find_all()`.

The simplest filter is a tag. Pass a tag to a search method and BeautifulSoup will perform a match against that exact string. You can also be more specific in your search by including tag attributes like “class”.
In our example above we saw the tag for the reviews is the paragraph tag `<p>` and it has the “class” attribute with value “`partial_entry`”.
Let us try and find all the `<p>` (paragraph) tags in the soup with attribute ’class’ and attribute value “`partial_entry`”:

Working with Python – Crawl and Scrape – Handout

Digital Management – November 2020

Python code – continued (complete the if statement)

```
import requests
from bs4 import BeautifulSoup

scrape_url = 'https://www.tripadvisor.com/Restaurant_Review-g227613-d3531819-Reviews-
Le_Jardin_Napolitain-Jouy_en_Josas_Versailles_Yvelines_Ile_de_France.html'

response = requests.get(scrape_url)
print(response.status_code)
if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser') # Here you make the Soup
        count = 1 # we'll use this to show the review number

        # now we search tag <p> with attribute class 'partial_entry'
        for review in soup.find_all('p', class_='partial_entry'):
                print('review number: ',count)
                print(review.text) #We are interested only in the text data
                count += 1
```

You get something like this:

200 # this is the response code
review number: 1
As tourists in France it was a pleasure to taste Italian food food.wonderfool atmosphere in a small village
review number: 2
The waitress are very kind and speak English which makes ordering easier. I ordered pasta and it was nice portion, not too big, fastly served.
.
.
.
review number: 10
We (family of three) visited the place almost at the end of lunch time. We were offered very fast yet tasty food with smile. The staff is very helpful and courteous. They even offered complementary wine with the food. Overall a decent place in Jouy.

You'll see that as only 10 reviews are visible on the web page, you got only these 10 reviews. You can also notice that some reviews end with '…More'. If the user of the web page wants to see the full review, he/she has to click on 'More' to see it.
Basically, here you got an exact copy of what is displayed on the web page.

Food for thought: can you think of a way to get all the reviews for a restaurant in TripAdvisor?