AAM1 Task 1: Web Scraping

Sina Bozorgmehr

Western Governors University

AAM1 Task 1: Web Scraping

Introduction

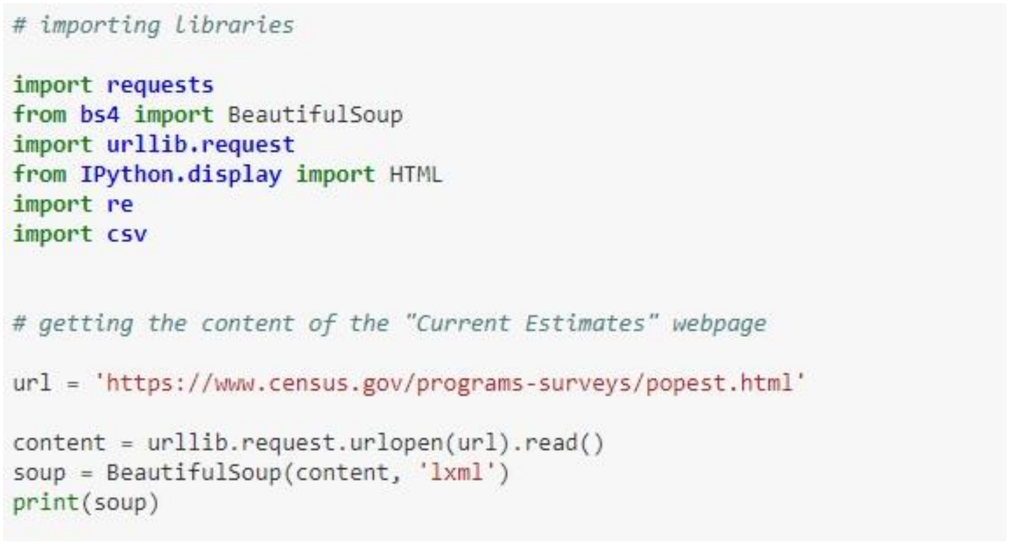
As Mitchell states, Web Scraping is the collection of data with any method other than using APIs or web browsers and most of the time is done by writing a script that automatically ask for data, gets it and then goes through it to separate the information of interest (2018). In this project, a simple script in python is used to parse a specific webpage and extract all the unique links that relocate to an HTML page, and then writes this links into a Comma Separated Value (CSV) file.

Script

The script, gets the HTML content of the Current Estimates webpage and parse it using *urllib.request* and *BeautifulSoup* libraries, respectively. First, *urlopen* and *read* methods, request and open and save the content of the page into a variable. Then, this variable is changed to a *BeautifulSoup* object and the content are parsed (Figure 1). You may refer to *Scraped_content.txt* for the html codes that were scraped.

Figure 1

Code to import libraries and retrieve the html content

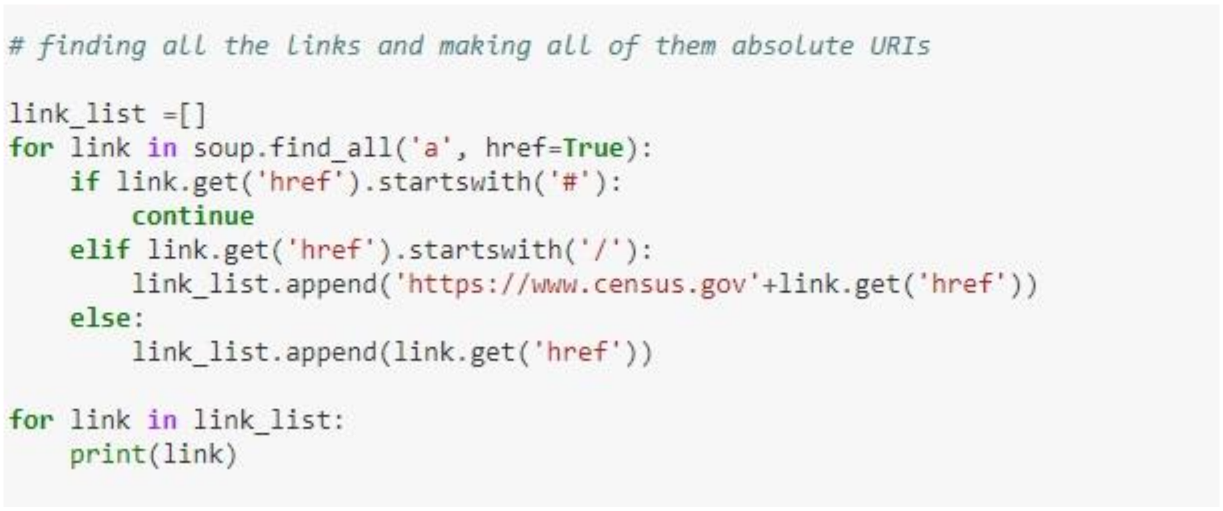


```
# importing libraries

import requests
from bs4 import BeautifulSoup
import urllib.request
from IPython.display import HTML
import re
import csv


# getting the content of the "Current Estimates" webpage

url = 'https://www.census.gov/programs-surveys/popest.html'

content = urllib.request.urlopen(url).read()
soup = BeautifulSoup(content, 'lxml')
print(soup)
```

Using a for loop and *find_all* method, all “*a*” tags with a value for “*href*” attribute are identified. Links that refer to another part of the same page are skipped using *startswith* method. Using the same method, links referring to another page inside the domain, those that start with a “/”, are marked and the main domain is added in the beginning to make them absolute links. All the extracted links are added into a list (Figure 2).

Figure 2

Codes to find links and make them absolute URIs



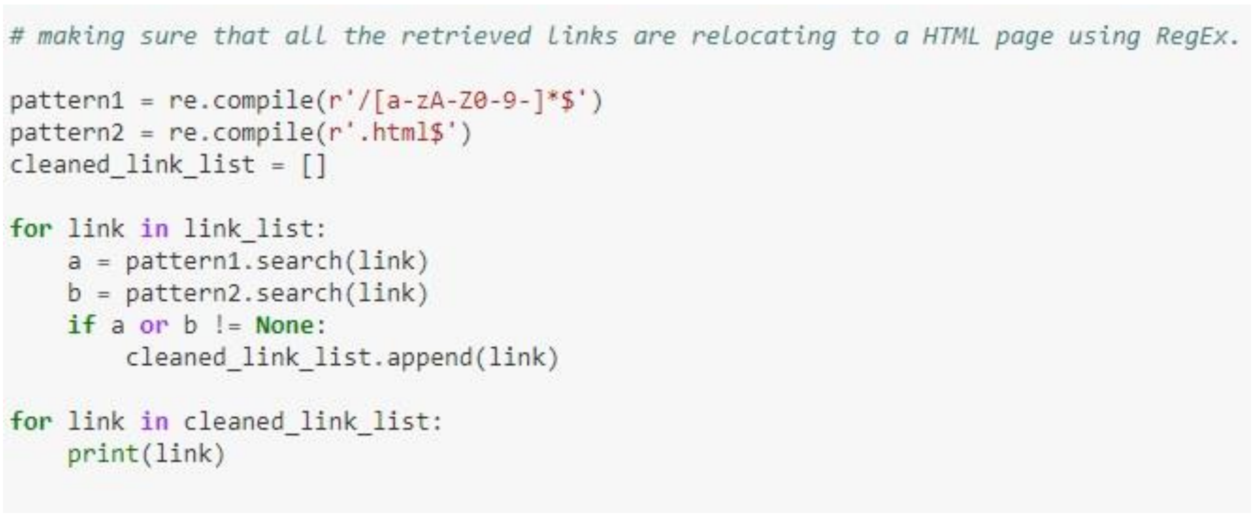
```
# finding all the links and making all of them absolute URIs

link_list =[]
for link in soup.find_all('a', href=True):
    if link.get('href').startswith('#'):
        continue
    elif link.get('href').startswith('/'):
        link_list.append('https://www.census.gov'+link.get('href'))
    else:
        link_list.append(link.get('href'))

for link in link_list:
    print(link)
```

Using regular expression, links that does not relocate to another HTML page are identified and removed from the list. Links that relocate to another HTML page end either in “.html” or are queries that end in “/”. All other links refer to a file. for example, “.pdf” relocate to a PDF file. Using a *for loop* and *search* method, HTML pages are separated using the string pattern of “*.html$*” and queries are marked using “*/[a-zA-Z0-9-]*$*” (Figure 3).

To remove the duplicate links, a set is created using the list of the link. A set automatically identifies duplicate items and keeps only one of them. This set is then changed back to a list to be used for writing into a CSV file. *Csv* package is used to make a CSV file that has one link per row (Figure 4).

Figure 3

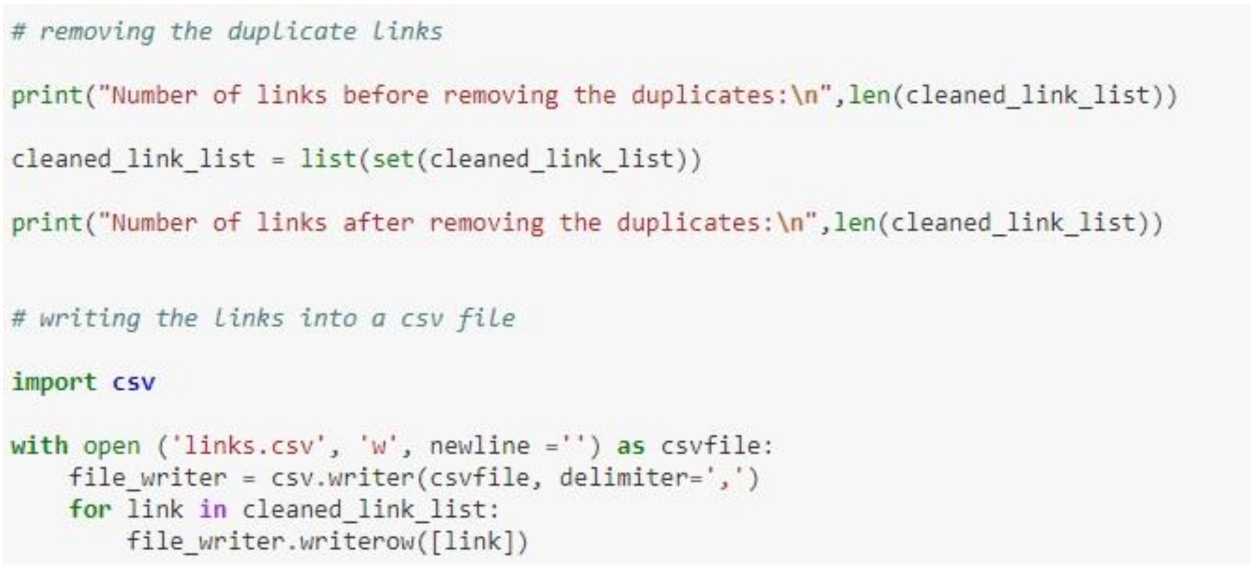
Codes to remove links that do not relocate to HTML pages.



```python
# making sure that all the retrieved links are relocating to a HTML page using RegEx.

pattern1 = re.compile(r'/[a-zA-Z0-9-]*$')
pattern2 = re.compile(r'.html$')
cleaned_link_list = []

for link in link_list:
    a = pattern1.search(link)
    b = pattern2.search(link)
    if a or b != None:
        cleaned_link_list.append(link)

for link in cleaned_link_list:
    print(link)
```

Figure 4

Codes to remove duplicate links and writing the unique links into a CSV file.



```python
# removing the duplicate links

print("Number of links before removing the duplicates:\n",len(cleaned_link_list))

cleaned_link_list = list(set(cleaned_link_list))

print("Number of links after removing the duplicates:\n",len(cleaned_link_list))


# writing the links into a csv file

import csv

with open ('links.csv', 'w', newline ='') as csvfile:
    file_writer = csv.writer(csvfile, delimiter=',')
    for link in cleaned_link_list:
        file_writer.writerow([link])
```

Results and Conclusion

Using this python script, refer to *link_scraper.txt*, a csv file was created, refer to *links.csv,* that contains 118 unique links in absolute format which relocate to HTML pages. A screenshot of the result of run scripts is available at *result_screenshot.jpg*. Each link has been stored in a separate row to make it easier to go through each one of them. World wide web is like an ocean of information which requires a specific method to capture this big data. This method is normally called Web Scraping (Oancea & Necula, 2019). After retrieving the HTML content of the web pages, the contents are parsed using regular expression methods and sometimes with the help of additional logics (Glez-Peña, Lourenço, López-Fernández, Reboiro-Jato, & Fdez-Riverola, 2013) to organize the content and extract the part of interest for additional analysis.

References

Glez-Peña, D., Lourenço, A., López-Fernández, H., Reboiro-Jato, M., & Fdez-Riverola, F. (2013). Web scraping technologies in an API world. *Briefings in Bioinformatics*, *15*(5), 788-797. doi:10.1093/bib/bbt026

Mitchell, R. (2018). *Web scraping with Python: Collecting data from the modern web*. Sebastopol, CA: O'Reilly.

Oancea, B., &amp; Necula, M. (2019). Web scraping techniques for price statistics – the Romanian experience. *Statistical Journal of the IAOS*, *35*(4), 657-667. doi:10.3233/sji-190529