Web Scraping with BeautifulSoup

Bety E. Rodriguez-Milla

Motivation

A not-for-profit organization is trying to reach the 194 Community Foundations of Canada (CFC) across the nation.

They want to mail them some materials. They need a *spreadsheet* with the name of the contact, title, mailing address, etc. (11 fields), for each of the CFCs.

Copy-paste each field? Web Scraping!

What is Web Scraping?

It is the practice of gathering data, through any means other than API.

For example, by writing an automated program that:

- queries a web server,
- requests and retrieves data,
- parses that data to extract information, and
- stores it.

Web Scraping with Python by Ryan Mitchell (O'Reilly)

Why Web Scraping?

- Web scrapers are excellent at gathering and processing large amounts of data, thousands of pages at once, from a collection of sites.
- Not all websites have an API - or an API that suits your needs.

If you can *see* it in your browser, you can *access* it via a Python script.

And, if you can *access* it, you can *store* it in a database to *retrieve* and *analyze*.

Inspect the Page

ABOUT CFC OUR WORK LEARNING INSTITUTE PARTNERS GET INVOLVED

AB

Wood Buffalo Community Foundation

Alberta

Airdrie and District Community Foundation

Banff Canmore Community Foundation

Battle River Community Foundation

```
254     <div class="wrapper">
255
256           <div class="has-columns prov prov__can">
257         <h2>AB</h2>
258
259     <h3><a href="https://www.communityfoundations.ca/cfc_locations/wood-buffalo-community-
    foundation/">Wood Buffalo Community Foundation</a></h3>
260
261     <br class="clearfix"></div>          <div class="has-columns prov prov__can prov__ab">
262           <h2>Alberta</h2>
263
264     <h3><a href="https://www.communityfoundations.ca/cfc_locations/airdrie-and-district-community-
    foundation/">Airdrie and District Community Foundation</a></h3>
265
266
267     <h3><a href="https://www.communityfoundations.ca/cfc_locations/the-banff-community-
    foundation/">Banff Canmore Community Foundation</a></h3>
```

Banff Canmore Community Foundation

214 Banff Avenue/Box 3100 | Banff | T1L 1C7

403-762-8549

www.banffcanmorecf.org

Rob Buffler, Executive Director

```
282         <h1>Banff Canmore Community Foundation</h1>
283
284
285         <div class="single-meta single-event">
286
287         <p class="meta-line location">214 Banff Avenue/Box 3100 | Banff | T1L 1C7</p>
288                 <p class="meta-line phone"><a href="tel:403-762-8549">403-762-8549</a></p>
289                         <p class="meta-line link"><a
    href="http://www.banffcanmorecf.org">www.banffcanmorecf.org</a></p>
290                         <p class="meta-line contact">Rob Buffler, Executive Director</p>
291
292         </div>
```

BeautifulSoup

- Python library for parsing HTML and XML documents - even for pages with malformed markup or poorly designed.
- Provides simple methods to navigate, search, and modify parse trees.
- Automatically converts incoming documents to Unicode and outgoing to UTF-8.

www.crummy.com/software/BeautifulSoup

Libraries Used

In [125]:
```
# import libraries
from requests import get
from bs4 import BeautifulSoup
import regex as re
import csv
import pandas as pd
import time
from genderize import Genderize
import nltk
from nltk.tokenize import word_tokenize
from nltk.tag import pos_tag
import spacy
from spacy import displacy
from collections import Counter
import en_core_web_sm
nlp = en_core_web_sm.load()
```

In [3]:
```
url = 'https://communityfoundations.ca/find-a-community-foundation/'
response = get(url)

#print 500 characters for show
print(response.text[:500])
```

```
<!doctype html>
<!--[if lt IE 7]> <html class="no-js ie6 oldie" lang="en-US" prefix="og:
 http://ogp.me/ns#"> <![endif]-->
```

In [4]:
```
html_soup = BeautifulSoup(response.text, 'html.parser')
type(html_soup)
```

Out[4]:
```
bs4.BeautifulSoup
```

In [59]:
```
info_containers = html_soup.find_all('h3')
print(type(info_containers))
print(len(info_containers))
```

```
<class 'bs4.element.ResultSet'>
194
```

In [64]:
```
first_cfc = info_containers[0]
first_cfc
```

Out[64]:
```
<h3><a href="https://communityfoundations.ca/cfc_locations/wood-buffalo-c
ommunity-foundation/">Wood Buffalo Community Foundation</a></h3>
```

In [123]:
```
location_title = first_cfc.text
location_title
```

Out[123]:
```
'Wood Buffalo Community Foundation'
```

In [211]:
```
info_containers_all = html_soup.find_all(["h2", "h3"],
                                         class_=lambda x: x != 'hidden')
print(type(info_containers_all))
print(len(info_containers_all))
```

```
<class 'bs4.element.ResultSet'>
207
```

In [268]:
```
for lines in info_containers_all:
    if lines.name == 'h2':
        province = lines.text
        print ('In Province', lines.text)
    if lines.name == 'h3':
        print('Foundation: ', lines.text)
        foundation = lines.text
        print ('Foundation url: ', lines.find_all("a",
                        href=re.compile("cfc_locations"))[0].get('href'))
```

```
In Province AB
Foundation:  Wood Buffalo Community Foundation
Foundation url:  https://communityfoundations.ca/cfc_locations/wood-buffalo-community-foundation/
In Province Alberta
Foundation:  Airdrie and District Community Foundation
Foundation url:  https://communityfoundations.ca/cfc_locations/airdrie-and-district-community-foundation/
Foundation:  Banff Canmore Community Foundation
Foundation url:  https://communityfoundations.ca/cfc_locations/the-banff-community-foundation/
```

In [ ]:
```
url = 'https://communityfoundations.ca/cfc_locations/the-banff-community-fou
subresponse = get(url)
html_subsoup = BeautifulSoup(subresponse.text, 'html.parser')
```

In [16]:
```
addr_containers = html_subsoup.find_all('div',
                                        class_ = 'single-meta single-event')
print(type(addr_containers))
print(len(addr_containers))
```

```
<class 'bs4.element.ResultSet'>
1
```

In [17]:
```
first_subcfc = addr_containers[0]
first_subcfc
```

Out[17]:
```
<div class="single-meta single-event">
<p class="meta-line location">214 Banff Avenue/Box 3100  | Banff | T1L 1C
7</p>
<p class="meta-line phone"><a href="tel:403-762-8549">403-762-8549</a></p
>
<p class="meta-line link"><a href="http://www.banffcanmorecf.org">www.ban
ffcanmorecf.org</a></p>
<p class="meta-line contact">Rob Buffler, Executive Director</p>
</div>
```

In [37]:
```
c_location = html_subsoup.find_all('p', class_ = 'meta-line location')
print(type(c_location))
print(len(c_location))
ctext_location = c_location[0]
ctext_location.text
```

```
<class 'bs4.element.ResultSet'>
1
```

Out[37]: '214 Banff Avenue/Box 3100 | Banff | T1L 1C7'

In [128]:
```
address_array = c_location[0].text
address_array
```

Out[128]: '214 Banff Avenue/Box 3100 | Banff | T1L 1C7'

How about using NLP for parsing the address?

NLTK

In [131]:
```
def preprocess(sent):
    sent = nltk.word_tokenize(sent)
    sent = nltk.pos_tag(sent)
    return sent
```

In [132]:
```
sent = preprocess(address_array)
sent
```

Out[132]:
```
[('214', 'CD'),
 ('Banff', 'NNP'),
 ('Avenue/Box', 'NNP'),
 ('3100', 'CD'),
 ('|', 'NNP'),
 ('Banff', 'NNP'),
 ('|', 'NNP'),
 ('T1L', 'NNP'),
 ('1C7', 'CD')]
```

Proper Nouns

Tokenization split zip code

Train model for geographical data?

spaCy

```
In [135]: article = nlp(address_array)
          len(article.ents)
```

```
Out[135]: 5
```

```
In [141]: sentences = [x for x in article.sents]
          print(sentences[0])
```

```
214 Banff Avenue/Box 3100  | Banff | T1L 1C7
```

```
In [142]: displacy.render(nlp(str(sentences[0])), jupyter=True, style='ent')
```

214 CARDINAL Banff Avenue/Box FAC 3100 PERCENT | Banff PERSON | T1L 1C7 CARDINAL

The Code

In [321]:
```
# Run only once, do not run again

# Get urls container
subresponse=[]

html_soup = BeautifulSoup(response.text, 'html.parser')
info_containers_all = html_soup.find_all(["h2", "h3"],
                                         class_=lambda x: x != 'hidden')
#print(len(info_containers_all))

for lines in info_containers_all:
    if lines.name == 'h3':
        url_fou = lines.find_all("a",
                                 href=re.compile("cfc_locations"))[0].get('href')
        print(url_fou)
        subresponse.append(get(url_fou))
        time.sleep(10)
```

```
https://communityfoundations.ca/cfc_locations/wood-buffalo-community-fou
ndation/
https://communityfoundations.ca/cfc_locations/airdrie-and-district-commu
nity-foundation/
https://communityfoundations.ca/cfc_locations/the-banff-community-founda
tion/
```

In [418]:
```
# Creating containers for the information that will be written to file.
organization = []
person = []
person_title = []
street = []
pobox = []
municipality = []
provinces = []
postalCode = []
phone = []
org_url = []
gender_title = []

# A dictionary so that we use the two letter abbreviation for the mailing
provincesDict = {"Alberta": 'AB',
                 "British Columbia": 'BC',
                 "Manitoba": 'MB',
                 "New Brunswick": 'NB',
                 "Newfoundland and Labrador": "NL",
                 "Northwest Territories": 'NT',
                 "Nova Scotia": 'NS',
                 "Ontario": 'ON',
                 "Prince Edward Island": 'PE',
                 "Québec": "QC",
                 "Saskatchewan": 'SK',
                 "Yukon": 'YT',
                 "Nunavut": 'NU',
                 "AB": 'AB'
                }

genderDict = {"male": 'Mr.',
              "female": 'Ms.'}
```

```
html_soup = BeautifulSoup(response.text, 'html.parser')
info_containers_all = html_soup.find_all(["h2", "h3"],
                                         class_=lambda x: x != 'hidden')
#print(type(info_containers_all))
#print(len(info_containers_all))

counter = 0

html_subsoup=[]

for lines in info_containers_all:
    if lines.name == 'h2':
        province = lines.text
        #print ('In Province', lines.text)

    if lines.name == 'h3':
        #print('Foundation: ', lines.text)
        foundation = lines.text
        organization.append(foundation)

        html_subsoup.append(BeautifulSoup(subresponse[counter].text,
                                          'html.parser'))

        # Get Address
        c_location = html_subsoup[counter].find_all('p',
                                           class_ = 'meta-line location')

        address_array = re.split(r' \| ', c_location[0].text)
        # If three pieces, it does not have P.O. Box
        #print("address_array length: ", len(address_array))
        for i in range(0,len(address_array)):
            address_array[i]=address_array[i].strip()
            #print(address_array[i])
        if len(address_array) == 3:
```

```
        if len(address_array) == 3:
            municipality.append(address_array[1])
            provinces.append(provincesDict.get(province, "None"))
            postalCode.append(address_array[2])

            if "box" in address_array[0].lower():
                #print(address_array[0], " has Box")
                if "," in address_array[0]:
                    #print(address_array[0], " is not only a po box ")
                    # Split by comma
                    # Find which one has the box, assign accordingly
                    sub_address = address_array[0].split(',', 1)
                    for i in range(0,len(sub_address)):
                        sub_address[i] = sub_address[i].strip()
                        #print(sub_address[i])
                        if "box" in sub_address[i].lower():
                            pobox.append(sub_address[i])
                        else:
                            street.append(sub_address[i])
                else:
                    street.append('')
                    pobox.append(address_array[0])

            else:
                street.append(address_array[0])
                pobox.append('')
        else:
            print("Something went wrong with address for foundation: ", fo
```

```
# Get person
c_contact = html_subsoup[counter].find_all('p',
                                           class_ = 'meta-line contact')
if len(c_contact) > 0:
    nameArray = re.split(r', ', c_contact[0].text) #Means name consists o

    if " - " in c_contact[0].text: #Means name consists of 'name - positi
        nameArray = re.split(r' - ', c_contact[0].text)

    for i in range(0,len(nameArray)):
        #print(len(nameArray))
        nameArray[i] = nameArray[i].strip()
        nameArray[i] = nameArray[i].strip(',')

    if len(nameArray) == 1:
        name = nameArray[0]
        person.append(name)
        person_title.append('')
    elif len(nameArray) == 2:
        name = nameArray[0]
        person.append(name)
        name.strip('\'')
        person_title.append(nameArray[1])
    else:
        print("Something went wrong with person's name for foundation: ",
```

```
    if len(nameArray) == 1 or len(nameArray) == 2:
        first_name = name.split(' ')
        if len(first_name) <=3:
            gen = Genderize().get([first_name[0]])[0]['gender']
            #print(gen)
            gender_title.append(genderDict.get(gen, ""))
            #print(genderDict.get(gen, "None")) # In case it is easier to
        elif "." in c_contact[0].text:
            gender_title.append(first_name[0])
        else:
            print("Something went wrong with person's gender for foundati
    else:
        gender_title.append('')

else:
    person.append('')
    person_title.append('')
    gender_title.append('')
```

```
# Get phone
c_phone = html_subsoup[counter].find_all('p', class_ = 'meta-line phone')
if len(c_phone) > 0:
    phone.append(c_phone[0].text)
else:
    phone.append('')
```

```
# Get website
c_org_url = html_subsoup[counter].find_all('p', class_ = 'meta-line link')
if len(c_org_url) > 0:
    org_url.append(c_org_url[0].text)
else:
    org_url.append('')
```

In [419]:
```
# Put the info into frame
test_df = pd.DataFrame({'Organization': organization,
        'Title': gender_title,
        'Addressee (First Name, Last Name)': person,
        'Additional Info (Addressee Job Title, Dept, Etc.)': person_title,
        'Civic Address 1 (Apt/Suite #, Building #, Street Name)': street,
        'Civic Address 2 (PO Box #/RR #, or GD (General Delivery) and \
                        STN ID)': pobox,
        'Municipality': municipality,
        'Province or Territory': provinces,
        'Postal Code': postalCode,
        'Phone': phone,
        'Website': org_url
                        })
print(test_df.info())
test_df
```

```
cols = ['Organization',
        'Title',
        'Addressee (First Name, Last Name)',
        'Additional Info (Addressee Job Title, Dept, Etc.)',
        'Civic Address 1 (Apt/Suite #, Building #, Street Name)',
        'Civic Address 2 (PO Box #/RR #, or GD (General Delivery) and STN
        'Municipality',
        'Province or Territory',
        'Postal Code',
        'Phone',
        'Website'
        ]

# Use pandas to write to csv
test_df.to_csv('data/cfcMailingAddresses.csv', encoding='utf-8',
                index=False, columns = cols)
```

cfcMailingAddresses

Organization	Title	Addressee (First	Additional Info (Ad	Civic Address 1 (A	Civic Address 2 (PO	Municipality	Provi	Postal Co	Phone	Website
Wood Buffalo Community Foundation				c/o Redpoll Centre at Shell Place, 1 C.A. K		Fort McMurra	AB	T9H 5C5		www.wbcfoundat
Airdrie and District Community	Mr.	Dale Rathgeber		1, 213 Main Street	PO Box 10249	Airdrie	AB	T4B 0R6	403-948-5	www.airdriefound
Banff Canmore Community Fo	Mr.	Rob Buffler	Executive Director		214 Banff Avenue/Bo	Banff	AB	T1L 1C7	403-762-8	www.banffcanmo
Battle River Community Found	Ms.	Dana Andreassen	Executive Director		Box 1122	Camrose	AB	T4V 4E7	780-679-0	www.brcf.ca
Community Foundation of Letl	Ms.	Charleen Davidso	Executive Director	1202 - 2 Avenue S, Unit 50		Lethbridge	AB	T1J 0E3	403-328-5	www.cflsa.ca
Community Foundation Of Nor	Ms.	Tracey Vavrek	Executive Director	11330-106 Street, Suite 200		Grande Prairi	AB	T8V 7X9	780-538-2	www.buildingtom
Community Foundation of Sou	Mr.	Chris Christie		104, 430 - 6th Avenue S.E.		Medicine Hat	AB	T1A 2S8	403-527-9	www.cfsea.ca
Drayton Valley Community Fou	Ms.	Charlene Jones	Executive Director		Box 6836	Drayton Valle	AB	T7A 1S2	780-514-2	www.dvcf.org
Edmonton Community Founda	Mr.	Martin Garber-Co	President and CEO	9910 - 103rd Street NW		Edmonton	AB	T5K 2V7	780-426-0	www.ecfoundatio
Red Deer & District Communit	Ms.	Kristine Bugayong	CEO	Suite 203, Mid City Plaza, 4805-48 Street		Red Deer	AB	T4N 1S6	403-341-6	www.rddcf.ca
St. Albert Community Foundat	Mr.	Dave Reidie	Executive Director		P.O. Box 65068	St. Albert	AB	T8N 5Y3	780-458-8	www.sacf.ca

Summary

- Use web scraping to gather and process information.
- Inspect the webpage, view the source.
- BeautifulSoup can help parse HTML and XML.
  - find( ), findAll( ), tag names and attributes, works with regEx, search by CSS class.
- Fortunately, CFC page was straightforward to process.
- Refining code: adjust headers, throw exceptions.