Final Assignment_Webscraping

July 21, 2021

Extracting Stock Data Using a Web Scraping

Not all stock data is available via API in this assignment; you will use web-scraping to obtain financial data. You will be quizzed on your results.
Using beautiful soup we will extract historical share data from a web-page.

Table of Contents

```
<ul>
    <li>Downloading the Webpage Using Requests Library</li>
    <li>Parsing Webpage HTML Using BeautifulSoup</li>
    <li>Extracting Data and Building DataFrame</li>
</ul>
```

Estimated Time Needed: 30 min

[1]:
```
#!pip install pandas
#!pip install requests
!pip install bs4
#!pip install plotly
```

```
Collecting bs4
  Downloading https://files.pythonhosted.org/packages/10/ed/7e8b97591f6f45617413
9ec089c769f89a94a1a4025fe967691de971f314/bs4-0.0.1.tar.gz
Collecting beautifulsoup4 (from bs4)
  Downloading https://files.pythonhosted.org/packages/d1/41/e6495bd7d3781c
ee623ce23ea6ac73282a373088fcd0ddc809a047b18eae/beautifulsoup4-4.9.3-py3-none-
any.whl (115kB)
     |                 | 122kB 43.6MB/s eta 0:00:01
Collecting soupsieve>1.2; python_version >= "3.0" (from
beautifulsoup4->bs4)
  Downloading https://files.pythonhosted.org/packages/36/69/d82d04022f02733bf9a7
2bc3b96332d360c0c5307096d76f6bb7489f7e57/soupsieve-2.2.1-py3-none-any.whl
Building wheels for collected packages: bs4
  Building wheel for bs4 (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/a0/b0/b2/4f80b94
56b87abedbc0bf2d52235414c3467d8889be38dd472
Successfully built bs4
Installing collected packages: soupsieve, beautifulsoup4, bs4
Successfully installed beautifulsoup4-4.9.3 bs4-0.0.1 soupsieve-2.2.1
```

[2]:

```
import pandas as pd
import requests
from bs4 import BeautifulSoup
```

0.1 Using Webscraping to Extract Stock Data Example

First we must use the request library to downlaod the webpage, and extract the text. We will extract Netflix stock data https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/netflix_data_webpage.html.

[3]:

```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/
 ↪netflix_data_webpage.html"

data  = requests.get(url).text
```

Next we must parse the text into html using beautiful_soup

[4]:

```
soup = BeautifulSoup(data, 'html5lib')
```

Now we can turn the html table into a pandas dataframe

[5]:

```
netflix_data = pd.DataFrame(columns=["Date", "Open", "High", "Low", "Close",␣
 ↪"Volume"])

# First we isolate the body of the table which contains all the information
# Then we loop through each row and find all the column values for each row
for row in soup.find("tbody").find_all('tr'):
    col = row.find_all("td")
    date = col[0].text
    Open = col[1].text
    high = col[2].text
    low = col[3].text
    close = col[4].text
    adj_close = col[5].text
    volume = col[6].text

    # Finally we append the data of each row to the table
    netflix_data = netflix_data.append({"Date":date, "Open":Open, "High":high,␣
 ↪"Low":low, "Close":close, "Adj Close":adj_close, "Volume":volume},␣
 ↪ignore_index=True)
```

We can now print out the dataframe

[6]:

```
netflix_data.head()
```

[6]:
```
           Date    Open    High     Low   Close       Volume Adj Close
0  Jun 01, 2021  504.01  536.13  482.14  528.21   78,560,600    528.21
1  May 01, 2021  512.65  518.95  478.54  502.81   66,927,600    502.81
2  Apr 01, 2021  529.93  563.56  499.00  513.47  111,573,300    513.47
3  Mar 01, 2021  545.57  556.99  492.85  521.66   90,183,900    521.66
4  Feb 01, 2021  536.79  566.65  518.28  538.85   61,902,300    538.85
```

We can also use the pandas `read_html` function

[7]:
```
read_html_pandas_data = pd.read_html(url)
```

Beacause there is only one table on the page, we just take the first table in the list returned

[8]:
```
netflix_dataframe = read_html_pandas_data[0]

netflix_dataframe.head()
```

[8]:
```
           Date    Open    High     Low  Close* Adj Close**     Volume
0  Jun 01, 2021  504.01  536.13  482.14  528.21       528.21   78560600
1  May 01, 2021  512.65  518.95  478.54  502.81       502.81   66927600
2  Apr 01, 2021  529.93  563.56  499.00  513.47       513.47  111573300
3  Mar 01, 2021  545.57  556.99  492.85  521.66       521.66   90183900
4  Feb 01, 2021  536.79  566.65  518.28  538.85       538.85   61902300
```

0.2 Using Webscraping to Extract Stock Data Exercise

Use the `requests` library to download the webpage https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/amazon_data_webpage.html. Save the text of the response as a variable named `html_data`.

[9]:
```
url1 = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/
 ↪amazon_data_webpage.html'
html_data = requests.get(url1).text
```

Parse the html data using `beautiful_soup`.

[10]:
```
beautiful_soup = BeautifulSoup(html_data, 'html5lib')
```

Question 1 What is the content of the title attribute:

[15]:
```
beautiful_soup.title
```

[15]:
```
<title>Amazon.com, Inc. (AMZN) Stock Historical Prices &amp; Data - Yahoo
Finance</title>
```

Using beautiful soup extract the table with historical share prices and store it into a dataframe named `amazon_data`. The dataframe should have columns Date, Open, High, Low, Close, Adj

Close, and Volume. Fill in each variable with the correct data from the list `col`.

[29]:
```
amazon_data = pd.DataFrame(columns=["Date", "Open", "High", "Low", "Close",␣
↪"Volume"])

for row in soup.find("tbody").find_all("tr"):
    col = row.find_all("td")
    date = col[0].text
    Open = col[1].text
    high = col[2].text
    low = col[3].text
    close = col[4].text
    adj_close = col[5].text
    volume = col[6].text

    amazon_data = amazon_data.append({"Date":date, "Open":Open, "High":high,␣
↪"Low":low, "Close":close, "Adj Close":adj_close, "Volume":volume},␣
↪ignore_index=True)
```

Print out the first five rows of the `amazon_data` dataframe you created.

[30]:
```
print(amazon_data)
```

```
            Date    Open    High     Low   Close       Volume Adj Close
0   Jun 01, 2021  504.01  536.13  482.14  528.21   78,560,600    528.21
1   May 01, 2021  512.65  518.95  478.54  502.81   66,927,600    502.81
2   Apr 01, 2021  529.93  563.56  499.00  513.47  111,573,300    513.47
3   Mar 01, 2021  545.57  556.99  492.85  521.66   90,183,900    521.66
4   Feb 01, 2021  536.79  566.65  518.28  538.85   61,902,300    538.85
..           …       …       …       …       …            …         …
65  Jan 01, 2016  109.00  122.18   90.11   91.84  488,193,200     91.84
66  Dec 01, 2015  124.47  133.27  113.85  114.38  319,939,200    114.38
67  Nov 01, 2015  109.20  126.60  101.86  123.33  320,321,800    123.33
68  Oct 01, 2015  102.91  115.83   96.26  108.38  446,204,400    108.38
69  Sep 01, 2015  109.35  111.24   93.55  103.26  497,401,200    103.26

[70 rows x 7 columns]
```

Question 2 What is the name of the columns of the dataframe

[31]:
```
amazon_data.columns
```

[31]:
```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close'],
dtype='object')
```

Question 3 What is the `Open` of the last row of the amazon_data dataframe?

[32]:
```
amazon_data.tail(1)['Open']
```

```
[32]: 69    109.35
      Name: Open, dtype: object
```

About the Authors:

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Azim Hirjani

0.3 Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2021-06-09	1.2	Lakshmi Holla	Added URL in question 3
2020-11-10	1.1	Malika Singla	Deleted the Optional part
2020-08-27	1.0	Malika Singla	Added lab to GitLab

##