Indeed

June 3, 2021

1 Scraping Indeed Listings

The goal of this project is to scrape information off of a job listing on Indeed's website. This process would include the user inputting a city of their choice and returning a CSV file containing information of data science job listings in the first 10 pages of the search result. This will make the job searching process much easier for the user and the data scraped from the can also be used to create an awesome dataframe for some data science project. An example would be to find and compare the salaries and ratings quartiles of data scientist positions in different cities.

Through this project, I hope to gain exposure to web scraping through the use of BeautifulSoup.

1.1 Import Statements

[1]:
```
from bs4 import BeautifulSoup as BSoup
import requests
import pandas as pd
```

1.2 Creating our Response and BeautifulSoup objects

Let's first start by attempting to create a dataframe from just one Indeed page URL. Note this is a sample URL.

[2]:
```
URL = "https://www.indeed.com/jobs?q=Data+Scientist&l=New+York"
request = requests.get(URL)
print(request)
```

```
<Response [200]>
```

Awesome! We got a response code 200, meaning that our request to the webpage was successful! Let's now view the HTML of the webpage and use BeautifulSoup to make it look nicer. I will comment out the prettify statement, which prints out a nice version of the HTML, so it won't display the html code because it is very, very long :) If you are interested in what it looks like, feel free to copy paste the URL into your browser, right click the webpage, and click Inspect.

[3]:
```
page_html = BSoup(request.text, "html.parser")
# page_html.prettify()
```

Let's check how many job listings there are on this specific webpage (not counting all the other pages for the listings). We will call these each of these listings "cotainers" since they look like separate boxes when inspecting the page.

```
[5]: containers = page_html.findAll(name="div", attrs={"class": "row"})
     len(containers)
```

```
[5]: 15
```

It looks like there are 15 job from this sample URL! ## Extracting data by looking at the HTML tags from the BeautifulSoup object Let's start out by extracting the job title.

```
[6]: # Scrapes the job titles from the BeautifulSoup object
     def extract_job_title_from_result(soup):
         jobs = []
         for div in soup.find_all(name="div", attrs={"class":"row"}):
             for a in div.find_all(name="a", attrs={"data-tn-element":"jobTitle"}):
                 jobs.append(a["title"])
         return(jobs)

     extract_job_title_from_result(page_html)
```

```
[6]: ['Data Scientist',
      'Data Scientist',
      'Data Scientist',
      'Data Scientist',
      'Data Scientist, Podcasts',
      'Senior Data Scientist',
      'Data Scientist',
      'junior Data Scientist',
      'Data Scientist',
      'Data Scientist',
      'Junior Data Scientist',
      'Data Scientist',
      'Data Scientist, SEO Analytics',
      'Data Scientist',
      'Data Scientist']
```

Let's do the same for the company.

```
[7]: # Scrapes the company from the BeautifulSoup object
     def extract_company_from_result(soup):
         companies = []
         for container in containers:
             company = container.find_all(name="span", attrs={"class": "company"})
             if len(company) > 0:
                 for b in company:
                     companies.append(b.text.strip())
```

```
        else:
            test2 = div.find_all(name="span", attrs={"class":␣
↪"result-link-source"})
            for span in test2:
                companies.append(span.text.strip())
    return(companies)

extract_company_from_result(page_html)
```

[7]:
```
['PrimeNeuro',
 'Digital Republic Talent',
 'Robert Half',
 'Hinge',
 'Spotify',
 'Corning',
 'Carta',
 'CFSB',
 'Betterview',
 'Pluto TV',
 'Decode_M',
 'Boll & Branch',
 'Reddit',
 'Source Enterprises',
 'Gannett']
```

Let's do the same for the salary.

[8]:
```
# Scrapes the salary from the BeautifulSoup object
def extract_salary_from_result(soup):
    salaries = []
    for div in soup.find_all(name="div", attrs={"class": "row"}):
        div_two = div.find(name="span", attrs={'class': "salaryText"})
        if div_two == None:
            salaries.append("Not Available")
        else:
            salaries.append(div_two.text.strip())
    return salaries

extract_salary_from_result(page_html)
```

[8]:
```
['$95,000 - $105,000 a year',
 '$100,000 - $140,000 a year',
 '$170,000 - $175,000 a year',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 '$80,000 - $90,000 a year',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available']
```

Finally, let's do the same for ratings.

```
[9]: # Scrapes the ratings from the BeautifulSoup object
     def extract_ratings_from_result(soup):
         ratings = []
         for div in soup.find_all(name="div", attrs={"class": "row"}):
             div_two = div.find(name="span", attrs={'class': "ratingsContent"})
             if div_two == None:
                 ratings.append("Not Available")
             else:
                 ratings.append(div_two.text.strip())
         return ratings

     extract_ratings_from_result(page_html)
```

```
[9]: ['Not Available',
      'Not Available',
      '3.9',
      '4.8',
      '4.3',
      '3.8',
      '3.8',
      '4.4',
      'Not Available',
      'Not Available',
      'Not Available',
      'Not Available',
      '4.0',
      '4.3',
      '3.0']
```

Now let's build a dataframe by combining all the information we have so far!

```
[10]: example_df = pd.DataFrame(
          {"job_title": extract_job_title_from_result(page_html),
          "company": extract_company_from_result(page_html),
          "salary": extract_salary_from_result(page_html),
          "rating": extract_ratings_from_result(page_html)}
      )
      example_df
```

[10]:
```
                         job_title                  company  \
0                   Data Scientist               PrimeNeuro
1                   Data Scientist  Digital Republic Talent
2                   Data Scientist              Robert Half
3                   Data Scientist                    Hinge
4         Data Scientist, Podcasts                  Spotify
5            Senior Data Scientist                  Corning
6                   Data Scientist                    Carta
7            junior Data Scientist                     CFSB
8                   Data Scientist               Betterview
9                   Data Scientist                 Pluto TV
10           Junior Data Scientist                 Decode_M
11                  Data Scientist             Boll & Branch
12  Data Scientist, SEO Analytics                    Reddit
13                  Data Scientist       Source Enterprises
14                  Data Scientist                  Gannett

                         salary         rating
0    $95,000 - $105,000 a year  Not Available
1   $100,000 - $140,000 a year  Not Available
2   $170,000 - $175,000 a year            3.9
3                Not Available            4.8
4                Not Available            4.3
5                Not Available            3.8
6                Not Available            3.8
7     $80,000 - $90,000 a year            4.4
8                Not Available  Not Available
9                Not Available  Not Available
10               Not Available  Not Available
11               Not Available  Not Available
12               Not Available            4.0
13               Not Available            4.3
14               Not Available            3.0
```

Awesome! It looks good except that we need to clean the salary series since it is not consistent with units (years and hour) and fix the style of the salary text. We won't worry too much about that right now. Let's now try to get all listings from the first 10 pages of the Indeed searches

1.3 Looping through the first 5 webpages of a specific city.

[11]:
```
# Limit to first 10 pages counting from 0 in increments of 10 to match the URL␣
 ↪pattern
limit = 50

# Headers for the data we want to extract
columns = ["job_title", "company", "salary", "rating"]
```

[14]:
```
# We want the user to be able to input a city of their choice, so we will test␣
 ↪it using the input method
a = input()
city_selection = [a]
```

```
San Francisco
```

[15]:
```
# Creates a dataframe
sample_df = pd.DataFrame(columns=columns)

# Loops through the 10 webpages for the selected city and gets the information␣
 ↪similar to the functions we created above
for city in city_selection:
    for start in range(0, limit, 10):
        page = requests.get("https://www.indeed.com/jobs?q=Data+Scientist&l=" +␣
 ↪str(city) + "&start=" + str(start))
        soup = BSoup(page.text, "lxml", from_encoding="utf-8")

        for div in soup.find_all(name="div", attrs={"class": "row"}):
            num = (len(sample_df) + 1)
            job_post = []

            for a in div.find_all(name="a", attrs={"data-tn-element":␣
 ↪"jobTitle"}):
                job_post.append(a["title"])
            company = div.find_all(name="span", attrs={"class": "company"})
            if len(company) > 0:
                for b in company:
                    job_post.append(b.text.strip())
            else:
                test2 = div.find_all(name="span", attrs={"class":␣
 ↪"result-link-source"})
                for span in test2:
                    job_post.append(span.text)

            div_two = div.find(name="span", attrs={"class": "salaryText"})
            if div_two == None:
```

```
            job_post.append("Not Available")
        else:
            job_post.append(div_two.text.strip())

        div_three = div.find(name="span", attrs={"class": "ratingsContent"})
        if div_three == None:
            job_post.append("Not Available")
        else:
            job_post.append(div_three.text.strip())

        sample_df.loc[num] = job_post
```

Awesome, now that our dataframe is created, let's check it out! We will look at the first and last 5 rows to avoid displaying all of it.

Note: Some of the rows in the notebook file uploaded on GitHub may include some tags and information on GitHub, which is a problem of displaying the data on a new row (you can see this by the rows skipping). This problem is not see on the notebook itself. If you would like to see the normal dataframe, check out the PDF file of the notebook or the CSV file of the data.

[16]:
```
sample_df.head()
```

[16]:
```
                                             job_title                   company  \
1                                  Staff Data Scientist                 Metromile
2  Principal Data Scientist - Candidate Recommend...                     Indeed
3                                    Data Scientist III  Thermo Fisher - America
4                                  Staff Data Scientist  Thermo Fisher - America
5                                        Data Scientist                PrimeNeuro

                        salary         rating
1                Not Available            2.8
2  $187,000 - $231,000 a year            4.3
3                Not Available            3.5
4                Not Available            3.5
5   $95,000 - $105,000 a year  Not Available
```

[17]:
```
sample_df.tail()
```

[17]:
```
                                             job_title       company  \
45                                      Data Scientist  Zipcar, Inc.
46                                      Data Scientist   Commonstock
47  Data Scientist, Customer Intelligence - Opport...        VMware
48                                      Data Scientist          RAPP
49                              Data Analyst - Disqus   Zeta Global

           salary         rating
45  Not Available            3.5
```

```
46  Not Available  Not Available
47  Not Available            4.0
48  Not Available            3.5
49  Not Available            2.6
```

1.4 Data Cleaning

Let's first fix the salary series since there is a bolding issue with the Markdown concatenation of the '$', the first number of the range, and '-'.

[18]:
```
lst = []

for i in range(1, len(sample_df['salary']) + 1):
    # 6 figure salary a year with range
    if len(sample_df['salary'][i]) == 26:
        lst.append(sample_df['salary'][i][0:8] + ' - ' +␣
 ↪sample_df['salary'][i][12:19] + ' a year')

    # 5 figure salary a year with range
    elif len(sample_df['salary'][i]) == 25:
        lst.append(sample_df['salary'][i][0:7] + ' - ' +␣
 ↪sample_df['salary'][i][11:18] + ' a year')

    # 6 figure salary a year no range
    elif len(sample_df['salary'][i]) == 15:
        lst.append(sample_df['salary'][i][0:8] + ' a year')

    # 2 figure salary a hour no range
    elif len(sample_df['salary'][i]) == 17:
        lst.append(sample_df['salary'][i][0:3] + ' - ' +␣
 ↪sample_df['salary'][i][7:9] + ' an hour')

    # a 3 salary figure a hour no range
    elif len(sample_df['salary'][i]) == 18:
        lst.append(sample_df['salary'][i][0:3] + ' - ' +␣
 ↪sample_df['salary'][i][7:9] + ' an hour')

    # 4 figure salary a month no range
    elif len(sample_df['salary'][i]) == 14:
        lst.append(sample_df['salary'][i][0:6] + ' a month')

    else:
        lst.append(sample_df['salary'][i])

sample_df['salary'] = lst
```

[19]:
```
sample_df.head()
```

[19]:
```
                                            job_title                 company  \
1                                  Staff Data Scientist               Metromile
2  Principal Data Scientist - Candidate Recommend...                  Indeed
3                                    Data Scientist III  Thermo Fisher - America
4                                  Staff Data Scientist  Thermo Fisher - America
5                                        Data Scientist              PrimeNeuro

                      salary         rating
1              Not Available            2.8
2  $187,000 - 231,000 a year            4.3
3              Not Available            3.5
4              Not Available            3.5
5   $95,000 - 105,000 a year  Not Available
```

[20]:
```
sample_df.tail()
```

[20]:
```
                                             job_title       company  \
45                                      Data Scientist  Zipcar, Inc.
46                                      Data Scientist   Commonstock
47  Data Scientist, Customer Intelligence - Opport...        VMware
48                                      Data Scientist          RAPP
49                               Data Analyst - Disqus   Zeta Global

           salary         rating
45  Not Available            3.5
46  Not Available  Not Available
47  Not Available            4.0
48  Not Available            3.5
49  Not Available            2.6
```

Next, we are going to clean this data and then convert the dataframe into a CSV file. Let's start by cleaning the salary series to the correct rates. We will convert them into dollars a year. There are a lot of different cases involving different units and ranges, and some cases may not be covered. I will focus on covering cases displayed in San Francisco. We will also make the $ sign consistent among rows and different units.

Note: the string manipulation below assumes that hourly salaries are two digits and monthly to be in the thousands since salaries for these jobs. We can safely make this assumption for now as annual income for this position is usually 50k-200k.

[21]:
```
result = sample_df['salary']

for index, item in enumerate(sample_df["salary"]):
    # Ranges in hour -> convert to year assuming 8 hrs a day, 5 times a week
    if "hour" in item and '-' in item:
        lower = int(item[1:3])*8*365
        upper = int(item[6:8])*8*365
        result[index + 1] = "$" + "{:,}".format(lower) + " - " + "{:,}".
 ↪format(upper) + " a year"

    # No range in hour -> convert to year assuming 8 hrs a day, 5 times a week
    elif "hour" in item and '-' not in item:
        salary = int(item[1:3])*8*365
        result[index + 1] = "$" + "{:,}".format(salary) + " a year"

    # No range in month -> convert to year assuming 8 hrs a day, 5 times a week
    elif 'month' in item:
        no_range = int(item[1:2] + item[3:6])*12
        result[index + 1] = "$" + "{:,}".format(no_range) + " a year"

    # Already in year
    else:
        result[index + 1] = item

result
```

```
[21]: 1                  Not Available
      2     $187,000 - 231,000 a year
      3                  Not Available
      4                  Not Available
      5      $95,000 - 105,000 a year
      6     $130,000 - 156,000 a year
      7                  Not Available
      8                  Not Available
      9                  Not Available
      10                 Not Available
      11                 Not Available
      12                 Not Available
      13     $85,000 - 125,000 a year
      14                 Not Available
      15    $120,000 - 160,000 a year
      16                 Not Available
      17     $95,000 - 105,000 a year
      18                 Not Available
      19                 Not Available
      20                 Not Available
      21                 Not Available
      22                 Not Available
      23                 Not Available
      24                 Not Available
      25                 Not Available
      26                 Not Available
```

```
27                Not Available
28                Not Available
29                Not Available
30                Not Available
31                Not Available
32                Not Available
33                Not Available
34                Not Available
35                Not Available
36                Not Available
37                Not Available
38                Not Available
39                Not Available
40                Not Available
41                Not Available
42                Not Available
43                Not Available
44    $150,000 - 180,000 a year
45                Not Available
46                Not Available
47                Not Available
48                Not Available
49                Not Available
Name: salary, dtype: object
```

Let's check our dataframe now for our conversion rates, which should all be $ a year.

[22]:
```
sample_df
```

[22]:
```
                                            job_title  \
1                                 Staff Data Scientist
2   Principal Data Scientist - Candidate Recommend...
3                                   Data Scientist III
4                                 Staff Data Scientist
5                                       Data Scientist
6      Senior Data Scientist - Moderation Engineering
7            Senior Data Scientist - Machine Learning
8                                       Data Scientist
9                                       Data Scientist
10                                      Data Scientist
11                                      Data Scientist
12                     Data Scientist - Intermediate
13                                      Data Scientist
14                                      Data Scientist
15                       Abl Schools | Data Scientist
16                            Associate Data Scientist
17                                      Data Scientist
```

```
18  Artificial Intelligence/Machine Learning Data ...
19            Principal Data Scientist - Telecommute
20                    Staff Data Scientist - Product
21                   Machine Learning Data Scientist
22                  Data Scientist, Machine Learning
23                                    Data Scientist
24       Data Scientist, Analytics - Messaging Graph
25                                    Data Scientist
26                           Research Data Scientist
27                            Data Product Associate
28                              Staff Data Scientist
29                             Vitria Data Scientist
30                           Data Science Consultant
31             Data Scientist - Research & Economics
32                    Senior Data Analyst-Marketing
33  Data Scientist (Digital Analytics & Monetization)
34                                    Data Scientist
35                         Data Scientist, Analytics
36                                    Data Scientist
37                                    Data Scientist
38           Associate Software Development Engineer
39                                    Data Scientist
40                         Data Management Associate
41                                    Data Scientist
42                                    Data Scientist
43                           Data Scientist Director
44      Sr. Data Scientist/Machine Learning Engineer
45                                    Data Scientist
46                                    Data Scientist
47  Data Scientist, Customer Intelligence - Opport...
48                                    Data Scientist
49                              Data Analyst - Disqus

                                  company                      salary  \
1                               Metromile               Not Available
2                                  Indeed  $187,000 - 231,000 a year
3                   Thermo Fisher - America               Not Available
4                   Thermo Fisher - America               Not Available
5                              PrimeNeuro   $95,000 - 105,000 a year
6                                  Indeed  $130,000 - 156,000 a year
7                                Blue Owl               Not Available
8                                Gap Inc.               Not Available
9                                   Carta               Not Available
10                     First Republic Bank               Not Available
11                             VIZIO, Inc.               Not Available
12                                  Bayer               Not Available
13                              The Beans   $85,000 - 125,000 a year
```

```
14                           Grammarly, Inc.            Not Available
15                               Abl Schools  $120,000 - 160,000 a year
16                                  Gap Inc.            Not Available
17                                PrimeNeuro   $95,000 - 105,000 a year
18                        Mitre Corporation            Not Available
19                       UnitedHealth Group            Not Available
20                                   Twitter            Not Available
21                                      Oura            Not Available
22                                     Carta            Not Available
23                             Komodo Health            Not Available
24                                  Facebook            Not Available
25            Cognizant Technology Solutions            Not Available
26     University of California San Francisco            Not Available
27                       Levi Strauss & Co.            Not Available
28                                  Opendoor            Not Available
29                        Vitria Technology            Not Available
30                                 Accenture            Not Available
31                                      Uber            Not Available
32                         Kaiser Permanente            Not Available
33  Sony Interactive Entertainment PlayStation            Not Available
34                                      Tubi            Not Available
35                      First Place for Youth            Not Available
36                             a.k.a. Brands            Not Available
37                             Grid Dynamics            Not Available
38                             NextEra Energy            Not Available
39                                  Aquabyte            Not Available
40                     ClimateWorks Foundation            Not Available
41                             Joby Aviation            Not Available
42                             Second Genome            Not Available
43                                    Oracle            Not Available
44                             Perfect Minds  $150,000 - 180,000 a year
45                              Zipcar, Inc.            Not Available
46                               Commonstock            Not Available
47                                    VMware            Not Available
48                                      RAPP            Not Available
49                               Zeta Global            Not Available

          rating
1            2.8
2            4.3
3            3.5
4            3.5
5  Not Available
6            4.3
7  Not Available
8            3.8
9            3.8
```

```
10              3.9
11    Not Available
12              4.2
13    Not Available
14              4.6
15    Not Available
16              3.8
17    Not Available
18              4.0
19              3.7
20              4.1
21    Not Available
22              3.8
23              3.3
24              4.1
25              3.9
26              4.2
27              3.9
28              4.2
29              3.3
30              4.0
31              3.7
32              4.1
33              3.7
34    Not Available
35              3.4
36    Not Available
37    Not Available
38              3.9
39    Not Available
40    Not Available
41    Not Available
42              4.5
43              3.8
44    Not Available
45              3.5
46    Not Available
47              4.0
48              3.5
49              2.6
```

Awesome! It looks like it works properly. Let's now finally convert this Pandas dataframe into a CSV file and check it out!

```
[23]: sample_df.to_csv('indeed.csv', index=False)
```

```
[ ]:
```