Indeed

August 3, 2020

1 Indeed Data Scraping Project

The goal of this project is to automate the Indeed Job searching process by allowing the user to input a city of their choice and returning a CSV file containing the cleaned job listings in the first 10 pages on Indeed. This will make the job searching process much easier for the user. The data scraped from the Indeed website 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.

1.1 Import Statements

[52]:
```
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.

[53]:
```
URL = "https://www.indeed.com/q-Data-Scientist-l-San-Francisco,-CA-jobs.html?
 ↪vjk=bc7c0e642f6453f4"
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.

[54]:
```
page_html = BSoup(request.text, "html.parser")
# page_html.prettify()
```

Let's check how many job listings there are on this one 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.

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

```
[55]: 16
```

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.

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

      extract_job_title_from_result(page_html)
```

```
[56]: ['Senior Applied Scientist',
       'Data Scientist',
       'Data Scientist',
       'Personalization Artificial Intelligence / Machine Learning (AI/ML) Data
      Solutions Consultant - Data Management Consultant 3',
       'Enterprise Architect (Data) (2020-9124)',
       'Associate Data Scientist I',
       'Data Scientist',
       'Data Scientist Entry Level - Pathrise Recruiting Partners',
       'Research Data Scientist',
       'Data Scientist: Data Visualization',
       'Junior Data Scientist - Pathrise Recruiting Partners',
       'Data Scientist',
       'Data Scientist',
       'Data Scientist',
       'Senior Data Scientist',
       'Data Scientist']
```

Let's do the same for the company.

```
[57]: # 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)
```

```
[57]: ['Spiketrap',
 'Global Fishing Watch',
 'Blue Owl',
 'Wells Fargo Bank',
 'Fisher Investments',
 'Levi Strauss & Co.',
 'project AI',
 'Pathrise',
 'University of California San Francisco',
 'Kaiser Permanente',
 'Pathrise',
 'Applied Technology & Science (A-T-S)',
 'Common Networks',
 'Yelp',
 'MOLOCO',
 'Triplebyte']
```

Let's do the same for the salary.

```
[58]: # 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)
```

```
[58]: ['Not Available',
 '$45 - $65 an hour',
 '$250,000 - $375,000 a year',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 'Not Available',
 '$145,000 - $225,000 a year']
```

Finally, let's do the same for ratings.

```
[59]: # 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)
```

```
[59]: ['Not Available',
 'Not Available',
 'Not Available',
 '3.7',
 '3.6',
 '3.9',
 'Not Available',
 'Not Available',
 '4.2',
 '4.1',
 'Not Available',
 'Not Available',
 'Not Available',
 '3.5',
 'Not Available',
 '5.0']
```

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

```
[60]: 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
```

```
[60]:                                              job_title  \
      0                          Senior Applied Scientist
      1                                    Data Scientist
      2                                    Data Scientist
      3   Personalization Artificial Intelligence / Mach…
      4             Enterprise Architect (Data) (2020-9124)
      5                         Associate Data Scientist I
      6                                    Data Scientist
      7   Data Scientist Entry Level - Pathrise Recruiti…
      8                           Research Data Scientist
      9                     Data Scientist: Data Visualization
      10  Junior Data Scientist - Pathrise Recruiting Pa…
      11                                   Data Scientist
      12                                   Data Scientist
      13                                   Data Scientist
      14                            Senior Data Scientist
      15                                   Data Scientist

                                        company                      salary  \
      0                               Spiketrap              Not Available
      1                     Global Fishing Watch          $45 - $65 an hour
      2                                Blue Owl  $250,000 - $375,000 a year
      3                       Wells Fargo Bank              Not Available
      4                     Fisher Investments              Not Available
      5                      Levi Strauss & Co.              Not Available
      6                              project AI              Not Available
      7                                Pathrise              Not Available
      8   University of California San Francisco              Not Available
      9                       Kaiser Permanente              Not Available
      10                               Pathrise              Not Available
      11     Applied Technology & Science (A-T-S)              Not Available
      12                        Common Networks              Not Available
      13                                   Yelp              Not Available
      14                                 MOLOCO              Not Available
      15                             Triplebyte  $145,000 - $225,000 a year

                 rating
      0   Not Available
      1   Not Available
```

```
2    Not Available
3              3.7
4              3.6
5              3.9
6    Not Available
7    Not Available
8              4.2
9              4.1
10   Not Available
11   Not Available
12   Not Available
13             3.5
14   Not Available
15             5.0
```

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 10 webpages of a specific city.

[61]:
```
# Limit to first 10 pages counting from 0 in increments of 10 to match the URL␣
↪pattern
limit = 100

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

[62]:
```
# 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
```

[63]:
```
# 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("http://www.indeed.com/jobs?
↪q=data+scientist+%2420%2C000&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.

[64]:
```
sample_df.head()
```

[64]:
```
                                 job_title           company  \
1                 Senior Applied Scientist         Spiketrap
2                           Data Scientist        Triplebyte
3                           Data Scientist          Blue Owl
4                Machine Learning Engineer        Triplebyte
```

```
5  Data Solutions Sr. Consultant/Personalization …  Wells Fargo Bank

                          salary         rating
1              Not Available  Not Available
2  $145,000 - $225,000 a year            5.0
3  $250,000 - $375,000 a year  Not Available
4  $150,000 - $250,000 a year            5.0
5              Not Available            3.7
```

[65]:
```
sample_df.tail()
```

[65]:
```
                                          job_title    company  \
149                             Senior Data Analyst  Good Eggs
150                  Data and Evaluation Contractor     AI4ALL
151  Staff Data Scientist - Activision Blizzard Media  King.com
152                   Senior Data Scientist & Modeler       SoFi
153       Data Scientist - Global Business Operations     Splunk

            salary         rating
149  Not Available            3.0
150  Not Available  Not Available
151  Not Available  Not Available
152  Not Available            3.2
153  Not Available            4.2
```

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 '-'.

[66]:
```
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
```

[67]:
```
sample_df.head()
```

[67]:
```
                                            job_title           company  \
1                             Senior Applied Scientist         Spiketrap
2                                       Data Scientist        Triplebyte
3                                       Data Scientist          Blue Owl
4                            Machine Learning Engineer        Triplebyte
5  Data Solutions Sr. Consultant/Personalization …  Wells Fargo Bank

                      salary         rating
1              Not Available  Not Available
2  $145,000 - 225,000 a year            5.0
3  $250,000 - 375,000 a year  Not Available
4  $150,000 - 250,000 a year            5.0
5              Not Available            3.7
```

[68]:
```
sample_df.tail()
```

[68]:
```
                                            job_title    company  \
149                               Senior Data Analyst  Good Eggs
150                     Data and Evaluation Contractor     AI4ALL
151  Staff Data Scientist - Activision Blizzard Media   King.com
152                    Senior Data Scientist & Modeler       SoFi
153       Data Scientist - Global Business Operations     Splunk

            salary         rating
149  Not Available            3.0
```

```
150  Not Available  Not Available
151  Not Available  Not Available
152  Not Available            3.2
153  Not Available            4.2
```

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.

[69]:
```python
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
```

[69]:
```
1                    Not Available
2       $145,000 - 225,000 a year
3       $250,000 - 375,000 a year
4       $150,000 - 250,000 a year
5                    Not Available
6                    Not Available
```

```
7                  Not Available
8                  Not Available
9                  Not Available
10                 Not Available
11                 Not Available
12                 Not Available
13                 Not Available
14                 Not Available
15                 Not Available
16     $131,400 - 189,800 a year
17     $145,000 - 225,000 a year
18     $131,400 - 189,800 a year
19                $60,000 a year
20                 Not Available
21                 Not Available
22               $120,000 a year
23                 Not Available
24                 Not Available
25                 Not Available
26                 Not Available
27                 Not Available
28                 Not Available
29                 Not Available
30     $250,000 - 375,000 a year
                 …
124                Not Available
125                Not Available
126                Not Available
127                Not Available
128                Not Available
129                Not Available
130                Not Available
131                Not Available
132                Not Available
133                Not Available
134    $150,000 - 250,000 a year
135               $60,000 a year
136    $145,000 - 225,000 a year
137    $131,400 - 189,800 a year
138    $250,000 - 375,000 a year
139                Not Available
140                Not Available
141                Not Available
142    $132,476 - 163,000 a year
143                Not Available
144                Not Available
145                Not Available
```

```
146                Not Available
147                Not Available
148                Not Available
149                Not Available
150                Not Available
151                Not Available
152                Not Available
153                Not Available
Name: salary, Length: 153, dtype: object
```

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

[73]:
```
sample_df
```

[73]:
```
                                              job_title  \
1                              Senior Applied Scientist
2                                        Data Scientist
3                                        Data Scientist
4                             Machine Learning Engineer
5    Data Solutions Sr. Consultant/Personalization …
6                              Associate Data Scientist I
7                                        Data Scientist
8                               Research Data Scientist
9    Data Scientist Entry Level - Pathrise Recruiti…
10                  Data Scientist: Data Visualization
11   Junior Data Scientist - Pathrise Recruiting Pa…
12                                       Data Scientist
13              Data Scientist / Quantitative Research
14                                       Data Scientist
15                                       Data Scientist
16                                       Data Scientist
17                                       Data Scientist
18                                       Data Scientist
19   Financial Analyst Summer Intern - Ideal for a …
20             Enterprise Architect (Data) (2020-9124)
21                                Senior Data Scientist
22                                       Data Scientist
23                            Data Scientist, Marketing
24             Data Scientist, Legal Policy & Economics
25           Data Scientist, Machine Learning innovator
26                 Data Scientist - Shop Recommendations
27   Data Scientist - Experimentation (Contract Pos…
28                                       Data Scientist
29                              Data Science Intern - Remote
30                                       Data Scientist
..                                                  …
124  Machine Learning Engineer Entry Level - Pathri…
```

```
125                    Computer Vision Data Scientist
126                              Senior Data Scientist
127   Data Scientist - Regional Merchandising Strategy
128                  Data Scientist/Research Engineer
129    Data Science Instructional Designer and Analyst
130                                Senior Data Analyst
131                          Sr. Pharmacy Data Analyst
132                   Data Scientist, Medidata - Core
133            Sr. Data Analyst, Twitter Service Tech
134                          Machine Learning Engineer
135  Financial Analyst Summer Intern - Ideal for a …
136                                     Data Scientist
137                                     Data Scientist
138                                     Data Scientist
139                           Principal Data Scientist
140                                      Data Engineer
141     Investment Banking Analyst Intern, Summer 2021
142  Machine Learning Researcher - Multi View & Seg…
143         Marketing and Business Analytics Interns
144                  Senior Data Scientist - Marketing
145                  Data Engineer - Python Programmer
146  Staff Data Scientist - Global Payments & Fraud…
147  Machine Learning Engineers (Multiple Opportuni…
148                      Data Science Manager- Health
149                                Senior Data Analyst
150                       Data and Evaluation Contractor
151   Staff Data Scientist - Activision Blizzard Media
152                    Senior Data Scientist & Modeler
153        Data Scientist - Global Business Operations

                                             company  \
1                                          Spiketrap
2                                         Triplebyte
3                                           Blue Owl
4                                         Triplebyte
5                                   Wells Fargo Bank
6                                 Levi Strauss & Co.
7                                         project AI
8                University of California San Francisco
9                                           Pathrise
10                                 Kaiser Permanente
11                                          Pathrise
12              Applied Technology & Science (A-T-S)
13                                       PicnicHealth
14                                              Yelp
15                                    Common Networks
16                                Global Fishing Watch
```

```
17                                          Triplebyte
18                                  Global Fishing Watch
19                                          MPL Brands
20                                   Fisher Investments
21                                              MOLOCO
22                             GradTests (gradtests.com)
23                                              Twitch
24                                                Uber
25                                   Standard Chartered
26                                          Stitch Fix
27                                        Getty Images
28                                           Deep Labs
29                                     Interview Query
30                                            Blue Owl
..                                                 …
124                                           Pathrise
125                                            Enlitic
126                                            Landing
127                                               Wish
128                                             Apixio
129              University of California San Francisco
130                                            Life360
131 University of California San Francisco Medical… 
132                                 Medidata Solutions
133                                            Twitter
134                                         Triplebyte
135                                         MPL Brands
136                                         Triplebyte
137                               Global Fishing Watch
138                                           Blue Owl
139                                             Tapjoy
140                                        CommonStock
141                                                GCA
142                                            Fyusion
143                                      UpCounsel LLC
144                                           Opendoor
145                                              Mondo
146        Sony Interactive Entertainment PlayStation
147                                   Grelock Partners
148                                            Twitter
149                                          Good Eggs
150                                             AI4ALL
151                                           King.com
152                                               SoFi
153                                             Splunk

                      salary          rating
```

```
1                Not Available  Not Available
2    $145,000 - 225,000 a year            5.0
3    $250,000 - 375,000 a year  Not Available
4    $150,000 - 250,000 a year            5.0
5                Not Available            3.7
6                Not Available            3.9
7                Not Available  Not Available
8                Not Available            4.2
9                Not Available  Not Available
10               Not Available            4.1
11               Not Available  Not Available
12               Not Available  Not Available
13               Not Available  Not Available
14               Not Available            3.5
15               Not Available  Not Available
16   $131,400 - 189,800 a year  Not Available
17   $145,000 - 225,000 a year            5.0
18   $131,400 - 189,800 a year  Not Available
19              $60,000 a year  Not Available
20               Not Available            3.6
21               Not Available  Not Available
22             $120,000 a year  Not Available
23               Not Available            4.4
24               Not Available            3.7
25               Not Available            4.1
26               Not Available            3.2
27               Not Available            3.9
28               Not Available            3.7
29               Not Available  Not Available
30   $250,000 - 375,000 a year  Not Available
..                           …            …
124              Not Available  Not Available
125              Not Available  Not Available
126              Not Available            3.8
127              Not Available            3.8
128              Not Available            4.3
129              Not Available            4.2
130              Not Available            4.5
131              Not Available            4.2
132              Not Available            3.7
133              Not Available            4.1
134  $150,000 - 250,000 a year            5.0
135             $60,000 a year  Not Available
136  $145,000 - 225,000 a year            5.0
137  $131,400 - 189,800 a year  Not Available
138  $250,000 - 375,000 a year  Not Available
139              Not Available            3.3
```

```
140               Not Available  Not Available
141               Not Available            3.1
142  $132,476 - 163,000 a year  Not Available
143               Not Available  Not Available
144               Not Available            3.0
145               Not Available  Not Available
146               Not Available            3.7
147               Not Available  Not Available
148               Not Available            4.1
149               Not Available            3.0
150               Not Available  Not Available
151               Not Available  Not Available
152               Not Available            3.2
153               Not Available            4.2

[153 rows x 4 columns]
```

Awesome! It looks like it worked properly. For example, line 2 with company Global Fish Watch was converted from 45-65 to 131,400-189,800. Let's now finally convert this Pandas dataframe into a CSV file and check it out!

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

```
[ ]:
```