Web-Scraping of Doctors Data

Prasad Kulkarni

September 22, 2021

Contents

1 Introduction **2**

1.0.1 Libraries used: 2

1.0.2 Extracting Pages of Site and Links from the Table 3

1.0.3 Extracting Profile Features 5

1.0.4 Saving file in.CSV and with DataFrame 7

Chapter 1

Introduction

- In this project, we are scrapping the doctors data from the https://www.drdata.in. The site consist of list of doctors in india across the different states and specialisation.
- The number of features extracted are Name, Specialization, Name of Hospital or Clinic, Phone number and State.
- Around 24000 doctors data is available however, few the features are not available for some doctors. For e.g, phone number or name of clinic etc. Thus the blank data has been replaced with Nan value.
- The Python code has been implemented for the scrapping is specific to the site https://www.drdata.in. To scrap from other sites, modification in the code needs to be done.

1.0.1 Libraries used:

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import requests
import pandas as pd
import numpy as np
```

1.0.2 Extracting Pages of Site and Links from the Table

```
def all_links(no_pages):

    def pages(no_pages):
        pages_=[]
        for i in range(1,no_pages):
            page_link='https://www.drdata.in/list-doctors.php?search=Doctor
&page='+str(i)
            pages_.append(page_link)
        return pages_

    a=[]
    for p in pages(no_pages):
        html = urlopen(p)
        bs = BeautifulSoup(html.read(), 'html.parser')
```

```
    name = bs.findAll('table')
    for i in name:
        link= i.findAll('a')


    for j  in link:
        link_name=j.get('href')
        a.append('https://www.drdata.in/'+link_name)

  return a
a=all_links(50)
```

- The pages on the site has following structure: 'https://www.drdata.in/list-doctors.php?search=Doctor&page='+str(i). The page number (i) changes. To see data on every page we have to scan each page individually.
- On each page we have table consist of doctors info however, the each doctor has their own profile link, where data is available. We have to scan each profile link to extract the data.
- The above code extracts all the profile links from each page. the function all_links() performs it. Only input we need to give to the function is no of pages to be extracted.
- 50 pages of site have been extracted.

1.0.3 Extracting Profile Features

```
def Profile():


  if 'Name' in list_:
    w=list_.index('Name')
    name_.append(list_[w+1])
  else:
    name_.append(np.nan)
  if 'Specialization' in list_:
     x=list_.index('Specialization')
     specialization_.append(list_[x+1])
  else:
    specialization_.append(np.nan)
  if 'Clinic/ Hospital Name' in list_:
    y=list_.index('Clinic/ Hospital Name')
    hospital_.append(list_[y+1])
  else:
    hospital_.append(np.nan)
  if 'Phone Number' in list_:
    z=list_.index('Phone Number')
    phone_.append(list_[z+1])
  else:
    phone_.append(np.nan)
```

```
    if 'State' in list_:
        q=list_.index('State')
        state_.append(list_[q+1])
    else:
        state_.append(np.nan)
```

```
dict_= {}
name_=[]
specialization_=[]
hospital_=[]
phone_=[]
state_=[]

for k in a:
    list_= []
    drlist = urlopen(k)
    bs1 = BeautifulSoup(drlist.read(), 'html.parser')
    for link in bs1.findAll('td'):
        list_.append(link.text)
    Profile()
```

- In this code function Profile(), extracts each feature, name address, phone number from the profile links and appends the data of each doctor.

- If the feature is not available then replaced the value with np.nan

1.0.4 Saving file in.CSV and with DataFrame

```
def save():
    dict_={'Name':name_,'Specialization':specialization_,'Hospital or
  Clinic':hospital_,'Phone Number': phone_,'State':state_}
    df=pd.DataFrame(dict_)
    df.to_csv('Dr_Data.csv')
    return df
```

- Data is saved with file name 'Dr_Data.csv' and return with pandas dataframe.