aflevering_2

March 8, 2024

1 Initialisation

[1]:
```
import requests

from bs4 import BeautifulSoup

import pandas as pd

import json
```

2 Part 2: Gathering Links

In order to extract article links across multiple pages and regions, I have defined three functions: **article_links** which extract all article links on a paticular page, **pages_in_section** which determines the number of pages in a section, and **article_links_in_section** which uses **article_links** and **pages_in_section** to extract all article links in a section. **article_links** and **pages_in_section** takes a url as a absolute path, but **article_links_in_section** takes the name of a section as a string, that is, a relative path.

[2]:
```
def article_links(url):
    # Extract HTML content
    contents = requests.get(url).text
    # Parse HTML content
    soup = BeautifulSoup(contents, 'html.parser')
    # Extract tags containing the attribute 'type' with value 'article'
    articles = soup.find_all(type='article')
    # Extract link for each article
    links = []
    for article in articles:
        links.append(article.find(href=True)['href'])

    return links

# print(article_links('https://www.bbc.com/news/world/europe'))

def pages_in_section(url):
    # Initialisation:
```

```
    # Make a request for first page
    n = 1
    r = requests.get(url + '?page=' + str(n))
    # Get first response status code
    s = r.status_code

    while s == 200:
        # Make a request for next page
        n = n + 1
        r = requests.get(url + '?page=' + str(n))
        # Get new response status code
        s = r.status_code

    return n-1

# print(pages_in_section('https://www.bbc.com/news/world/europe'))

def article_links_in_section(section):
    # Get URL of section
    url = 'https://www.bbc.com/news/world/' + section
    # Get number of pages in section
    num = pages_in_section(url)
    # Get all articles on each page
    links = []
    for i in range(1, num+1):
        links = links + article_links(url + '?page=' + str(i))

    return links

# print(article_links_in_section('europe'))
```

Now, let's write a little piece of code which extract all article links across all sections:

[3]:
```
sections = [
    'africa',
    'asia',
    'australia',
    'europe',
    'latin_america',
    'middle_east'
    ]

links = []
for section in sections:
    links += article_links_in_section(section)

print(len(links))
```

```
4796
```

As can be seen from the last line of the code, we've extracted around 5000 article links.

Now, let's export the links to a csv file:

[4]:
```
dic = {'source': links}

dataframe = pd.DataFrame(dic)

dataframe.to_csv('all_article_links.csv')
```

3 Part 3: Scraping Article Text

I have defined three functions: The first function is **meta_content** which tries to extract date, headline and author. It uses a scheme which works for most links, however, there are a few where it isn't able extract date, headline or author. The second function is **text_content** which will extract the relevant paragraphs that make up the main content of the article. Lastly, the third function is **article** which uses uses **meta_content** and **text_content** to extract date, headline, author (if it can) and the text.

[5]:
```
def meta_content(soup):
    # Extract meta content of article, if it exist according to scheme
    try:
        meta = soup.find('script', type='application/ld+json').text
    except Exception:
        return ['None', 'None', 'None']
    # Convert string to dictionary
    meta = json.loads(meta)
    # Extract date, headline and author, if they exist according to scheme
    lst = []
    try:
        lst.append(meta['datePublished'][:10])
    except Exception:
        lst.append('None')
    try:
        lst.append(meta['headline'])
    except Exception:
        lst.append('None')
    try:
        lst.append(meta['author'][0]['name'][3:])
    except Exception:
        lst.append('None')
    return lst

def text_content(soup):
    # Extract tags containing relevant paragraphs from article
    paragraphs = soup.find_all(attrs = {'data-component': 'text-block'})
```

```python
    # Extract text from paragraphs
    text = ''
    for paragraph in paragraphs:
        text = text + paragraph.text + ' '
    return text

def article(url):
    article = {'date': '', 'headline': '', 'author': '', 'text': ''}
    # Extract HTML content
    contents = requests.get(url).text
    # Parse HTML content
    soup = BeautifulSoup(contents, 'html.parser')

    # Add date, headline and author to dictionary
    meta = meta_content(soup)
    article['date']     = meta[0]
    article['headline'] = meta[1]
    article['author']   = meta[2]

    # Add text to dictionary
    text = text_content(soup)
    article['text'] = text

    return article
```

Let's try to scrape the first article link:

[6]:
```python
url = 'https://www.bbc.com'

print(article(url + links[0]))
```

```
{'date': '2024-03-07', 'headline': 'Kuriga kidnap: More than 280 Nigerian pupils
abducted', 'author': 'Mansur Abubakar', 'text': 'More than 280 Nigerian school
pupils have been abducted in the north-western town of Kuriga, officials say.
The pupils were in the assembly ground around 08:30 (07:30 GMT) when dozens of
gunmen on motorcycles rode through the school, one witness said. The students,
between the ages of eight and 15, were taken away, along with a teacher, they
added. Kidnap gangs, known as bandits, have seized hundreds of people in recent
years, especially the north-west. However, there had been a reduction in the
mass abduction of children over the past year until this week. Those kidnapped
are usually freed after a ransom is paid. The mass abduction was confirmed by
Uba Sani, the governor of Kaduna state, which includes Kuriga. He said 187
students had gone missing from a secondary school and 125 from the local primary
school but that 25 had since returned. The eyewitness said that one girl had
been shot by the gunmen and was receiving medical attention at the Birnin Gwari
hospital. A teacher who managed to escape said local people had tried to rescue
the children, but they were repelled by the gunmen and one person was killed.
Zakariyya Nasiru, who had a brother and sister taken hostage, told the BBC the
family were unable to sleep on Thursday night."All of us couldn\'t sleep as we
keep thinking about them. We are here praying for their safe return."Mr Nasiru
said one boy had escaped last night and had brought back harrowing reports of
their conditions, including a lack of food. Almost every family in the town is
thought to have a child among those kidnapped. The armed forces have launched an
operation to find them. "No child will be left behind," vowed the governor. In
January, bandits killed a school principal in the area and abducted his wife.
The kidnapping comes days after dozens of women and children were feared
kidnapped by the Boko Haram Islamist group while they were collecting firewood
in north-eastern Nigeria. However, the two cases of mass abductions are not
thought to be related. The criminal kidnap gangs that bring fear to north-
western Nigeria are separate to the militant Islamist group Boko Haram in the
north-east, although there have been reports that they may have worked together
on occasion. Thursday\'s attack happened in an area controlled by Ansaru, a
breakaway faction of Boko Haram, which kidnapped more than 200 schoolgirls from
the town of Chibok in 2014. In an attempt to curb Nigeria\'s spiralling and
lucrative kidnapping industry, a controversial law that has made it a crime to
make ransom payments was passed in 2022. It carries a jail sentence of at least
15 years, however no-one has ever been arrested. Earlier this year, the family
of a group of sisters kidnapped in the capital, Abuja, denied a police statement
that the security forces had rescued the girls, saying that they had no choice
but to pay the ransom. '}
```

… it seems to do the job in this case.

One could scrape all the articles with the following piece of code:

```
[ ]: '''
     articles = []
     for link in links:
         articles.append(article(url + link))
     '''
```

Due to time constraints, let's slice the list of links…

```
[7]: links = links[0:3]

     articles = []
     for link in links:
         articles.append(article(url + link))
```

And we can easily export it to a csv file as we did with the article links:

```
[9]: dic = {'articles': articles}

     dataframe = pd.DataFrame(dic)

     dataframe.to_csv('all_articles.csv')
```