M1 MSDA

Projet Web Scraping

Cheikh Yakhoub MAAS

Seydi Amadou DIALLO

Professeur: Mr BOUSSO

Sujet 14:

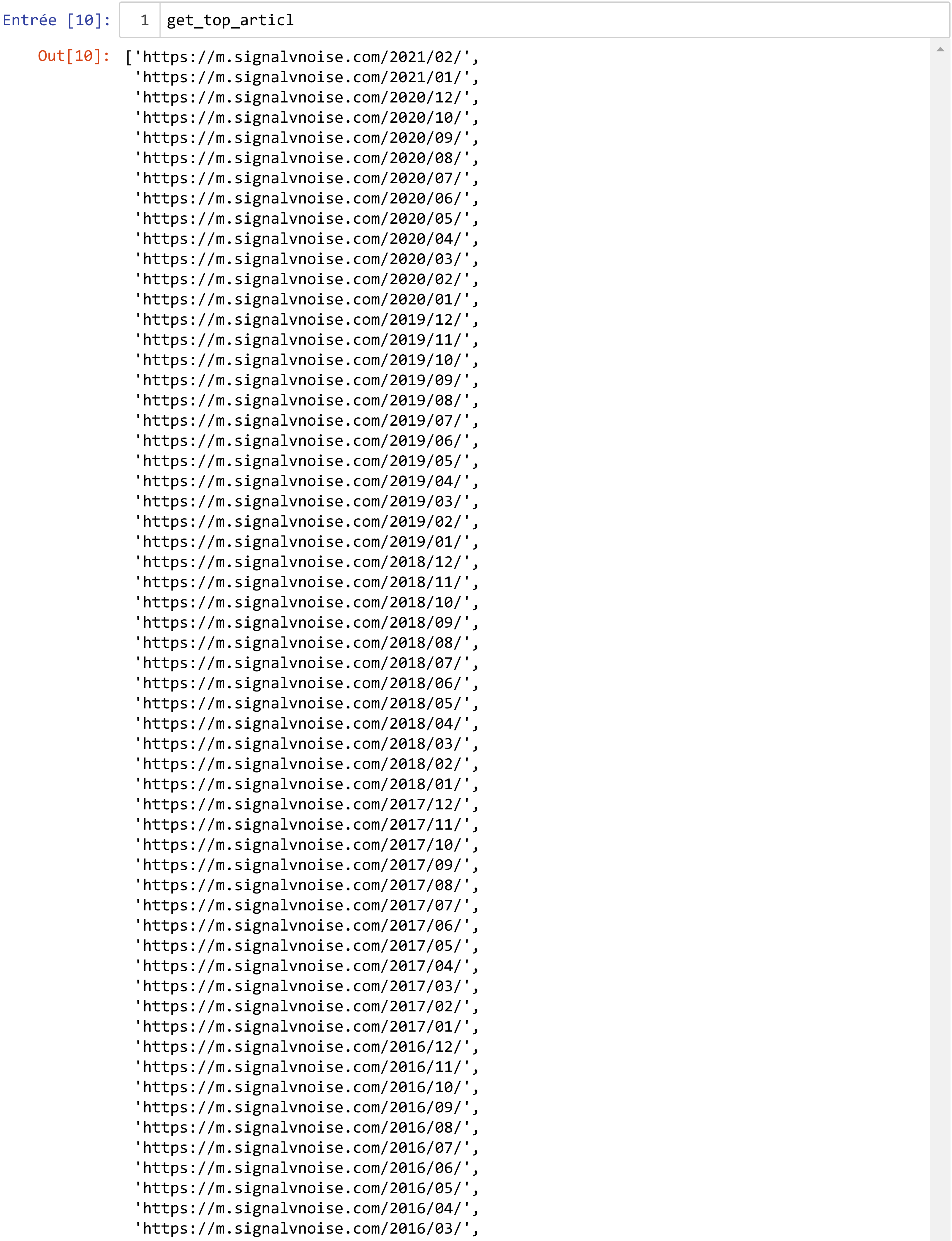
> *Créez un ensemble de données d'articles de blog sur un blog populaire, par ex. https://m.signalvnoise.com/search/ (https://m.signalvnoise.com/search/) . L'ensemble de données peut contenir des informations telles que le titre du blog, la date de publication, les balises, l'auteur, le lien vers un article de blog, etc.*

Entrée [6]:

```python
import requests
from bs4 import BeautifulSoup
```

Entrée [9]:

```
url='https://m.signalvnoise.com/search/'
response=requests.get(url)
doc = BeautifulSoup(response.text)
article_tags=doc.findAll('li')  # Permet de récupérer toutes les catégories d'articles
article_tag=article_tags[4] # Renvoie la catégorie d'articles en position 4(Septembre
'''
Cette fonction prends en parametre la variable qui renvoie la catégorie d'articles en
et qui retourne  l'url permettant d'acceder à ce groupe d'articles.
Nous avons en sommes 67 groupes d'articles. Dans la suite i varie entre 0 et 66
'''
def parse_articl(article_tag):
    a_tags = article_tag.find('a')

    url = a_tags['href']

    return url

'''
  Permet de récupérer sous forme de liste tous les urls permettant d'acceder aux diffé
'''

def get_top_articl(article_tags):
    all_article = [parse_articl(tag) for tag in article_tags]
    return all_article
get_top_articl=get_top_articl(article_tags)

```



```
Entrée [10]:  get_top_articl
```

```
Out[10]: ['https://m.signalvnoise.com/2021/02/',
          'https://m.signalvnoise.com/2021/01/',
          'https://m.signalvnoise.com/2020/12/',
          'https://m.signalvnoise.com/2020/10/',
          'https://m.signalvnoise.com/2020/09/',
          'https://m.signalvnoise.com/2020/08/',
          'https://m.signalvnoise.com/2020/07/',
          'https://m.signalvnoise.com/2020/06/',
          'https://m.signalvnoise.com/2020/05/',
          'https://m.signalvnoise.com/2020/04/',
          'https://m.signalvnoise.com/2020/03/',
          'https://m.signalvnoise.com/2020/02/',
          'https://m.signalvnoise.com/2020/01/',
          'https://m.signalvnoise.com/2019/12/',
          'https://m.signalvnoise.com/2019/11/',
          'https://m.signalvnoise.com/2019/10/',
          'https://m.signalvnoise.com/2019/09/',
          'https://m.signalvnoise.com/2019/08/',
          'https://m.signalvnoise.com/2019/07/',
          'https://m.signalvnoise.com/2019/06/',
          'https://m.signalvnoise.com/2019/05/',
          'https://m.signalvnoise.com/2019/04/',
          'https://m.signalvnoise.com/2019/03/',
          'https://m.signalvnoise.com/2019/02/',
          'https://m.signalvnoise.com/2019/01/',
          'https://m.signalvnoise.com/2018/12/',
          'https://m.signalvnoise.com/2018/11/',
          'https://m.signalvnoise.com/2018/10/',
          'https://m.signalvnoise.com/2018/09/',
          'https://m.signalvnoise.com/2018/08/',
          'https://m.signalvnoise.com/2018/07/',
          'https://m.signalvnoise.com/2018/06/',
          'https://m.signalvnoise.com/2018/05/',
          'https://m.signalvnoise.com/2018/04/',
          'https://m.signalvnoise.com/2018/03/',
          'https://m.signalvnoise.com/2018/02/',
          'https://m.signalvnoise.com/2018/01/',
          'https://m.signalvnoise.com/2017/12/',
          'https://m.signalvnoise.com/2017/11/',
          'https://m.signalvnoise.com/2017/10/',
          'https://m.signalvnoise.com/2017/09/',
          'https://m.signalvnoise.com/2017/08/',
          'https://m.signalvnoise.com/2017/07/',
          'https://m.signalvnoise.com/2017/06/',
          'https://m.signalvnoise.com/2017/05/',
          'https://m.signalvnoise.com/2017/04/',
          'https://m.signalvnoise.com/2017/03/',
          'https://m.signalvnoise.com/2017/02/',
          'https://m.signalvnoise.com/2017/01/',
          'https://m.signalvnoise.com/2016/12/',
          'https://m.signalvnoise.com/2016/11/',
          'https://m.signalvnoise.com/2016/10/',
          'https://m.signalvnoise.com/2016/09/',
          'https://m.signalvnoise.com/2016/08/',
          'https://m.signalvnoise.com/2016/07/',
          'https://m.signalvnoise.com/2016/06/',
          'https://m.signalvnoise.com/2016/05/',
          'https://m.signalvnoise.com/2016/04/',
          'https://m.signalvnoise.com/2016/03/',
          'https://m.signalvnoise.com/2016/02/',
          'https://m.signalvnoise.com/2016/01/',
          'https://m.signalvnoise.com/2015/12/',
          'https://m.signalvnoise.com/2015/11/',
          'https://m.signalvnoise.com/2015/10/',
          'https://m.signalvnoise.com/2015/09/',
          'https://m.signalvnoise.com/2014/06/',
          'https://m.signalvnoise.com/2013/11/']
```

Entrée [13]:

```
'''Prends en paramètre la position du groupe d'articles et retourne un dictionnaire ay
response qui récupère son url , soup pour analyser son contenu et article pour qui ren
Un groupe d'articles contient un ou plusieurs articles '''

def url_article(i):
    response=requests.get(get_top_articl[i])
    soup = BeautifulSoup(response.text,'html.parser')
    article=soup.findAll('article',class_="entry-summary grid__item grid__item--third"
    return {'response':response,
            'soup':soup,
            'article':article}

'''Prends en paramètre la position du groupe d'articles et l'indice de l'article conte
        et retourne ainsi cet article'''

def article_tag(i,j):
    article=url_article(i)['article']
    art_tag=article[j]
    return art_tag

```

Entrée [17]:

```
len(url_article(14)['article']) #Le nombre d'articles que contient le groupe 14(Novemb
```

Out[17]:

```
10
```

Entrée [14]:

```
article_tag(4,0)
```

Out[14]:

```
<article class="entry-summary grid__item grid__item--third">
<h2 class="entry-summary__title"><a href="https://m.signalvnoise.com/demand-side-sales-10
1-a-new-book-on-sales-by-bob-moesta/" rel="bookmark">Demand Side Sales 101, a new book on
sales by Bob Moesta.</a></h2>
<div class="entry-summary__meta">
<span class="byline"><a class="author url fn" href="https://m.signalvnoise.com/author/jas
on-fried/" rel="author" title="Posts by Jason Fried">Jason Fried</a></span> <span aria-hi
dden="true">/</span> <span class="posted-on"><span class="screen-reader-text">posted on
</span> <time class="entry-date published updated" datetime="2020-09-22T16:16:39-05:00">S
eptember 22, 2020</time></span> <span aria-hidden="true">/</span> <span class="comments-l
ink"><a href="https://m.signalvnoise.com/demand-side-sales-101-a-new-book-on-sales-by-bob
-moesta/#comments">9 Comments<span class="screen-reader-text"> on Demand Side Sales 101,
a new book on sales by Bob Moesta.</span></a></span> </div>
<p>Bob Moesta is a dear friend, mentor, and all around original thinker. He’s helped me s
ee around corners, shine lights on things I didn’t know were there, and approach product
development from unusual angles. Every time we talk, I come away inspired and full of opt
imism. So when he asked me to help him with… <a class="read-more" href="https://m.signalv
noise.com/demand-side-sales-101-a-new-book-on-sales-by-bob-moesta/">keep reading</a></p>
</article>
```

Entrée [18]:

```python

'''Retourne le nombre de commentaires que renferme un article contenu dans un groupe d

def comments(i,j):
    try:
        d=article_tag(i,j).find('div')
        c=d.find('span' ,class_="comments-link").text.strip()
    except (AttributeError):
        c='0'

    else:
        c= (c[0:2])
    return c

comments(14,9) #L'article numéro 9 contenu dans le groupe d'articles 14 (November 2019
```

Out[18]:

```
'92'
```

Entrée [19]:

```python
''' Fonction qui retourne le titre et l'url de de la catégorie,le titre, l'auteur,la d
commentaires de l'article.'''

def find_article(i,j):
    soup = url_article(i)['soup']
    a_tags = article_tag(i,j).findAll('div')
    t_tags = article_tag(i,j).find('time')
    title = soup.title.text
    url = get_top_articl[i]
    article_name=article_tag(i,j).find('a').text.strip()
    author=article_tag(i,j).find('a',class_="author").text.strip()
    date=t_tags.text.strip()
    comment_number=comments(i,j)

    return {
        'title':title,
        'url': url,
        'article_name':article_name,
        'author':author,
        'date':date,
        'comment_number':comment_number,

    }

find_article(4,0)
```

Out[19]:

```
{'article_name': 'Demand Side Sales 101, a new book on sales by Bob Moesta.',
 'author': 'Jason Fried',
 'comment_number': '9 ',
 'date': 'September 22, 2020',
 'title': 'September 2020 - Signal v. Noise',
 'url': 'https://m.signalvnoise.com/2020/09/'}
```

Entrée [20]:

```python
find_article(14,8)
```

Out[20]:

```
{'article_name': 'Rework Mailbag',
 'author': 'Wailin Wong',
 'comment_number': '0',
 'date': 'November 5, 2019',
 'title': 'November 2019 - Signal v. Noise',
 'url': 'https://m.signalvnoise.com/2019/11/'}
```

Entrée [21]:

```python
'''Retourne tous les articles concernant un groupe d'articles avec ses informations'''
def get_all_article(i):

    all_article = [find_article(i,j) for j in range(len(url_article(i)['article']))]
    return all_article
get_all_article(14)
```

Out[21]:

```
[{'article_name': 'Calm in the Political Storm',
  'author': 'Wailin Wong',
  'comment_number': '0',
  'date': 'November 26, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'The joy and power of being the independent underdog',
  'author': 'Jonas Downey',
  'comment_number': '8 ',
  'date': 'November 22, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Spending in the Clouds',
  'author': 'Wailin Wong',
  'comment_number': '17',
  'date': 'November 19, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': '7 leadership lessons over 2.5 years',
  'author': 'Claire Lew',
  'comment_number': '2 ',
  'date': 'November 18, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Breaking the Black Box',
  'author': 'Wailin Wong',
  'comment_number': '1 ',
  'date': 'November 15, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Launch: Basecamp Gets Personal',
  'author': 'Jason Fried',
  'comment_number': '38',
  'date': 'November 12, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Big Brother at the Office',
  'author': 'Wailin Wong',
  'comment_number': '0',
  'date': 'November 12, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Compounding time',
  'author': 'Jason Fried',
  'comment_number': '11',
  'date': 'November 5, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Rework Mailbag',
  'author': 'Wailin Wong',
  'comment_number': '0',
  'date': 'November 5, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'},
 {'article_name': 'Back to windows after twenty years',
  'author': 'DHH',
  'comment_number': '92',
  'date': 'November 4, 2019',
  'title': 'November 2019 - Signal v. Noise',
  'url': 'https://m.signalvnoise.com/2019/11/'}]
```

Entrée [22]:

```
#Une liste des noms des colonnes de notre dataframe
headers = list(get_all_article(0)[0].keys())
headers
```

Out[22]:

```
['title', 'url', 'article_name', 'author', 'date', 'comment_number']
```

Entrée [23]:

```
import csv
''' Prends en parametre la position du groupe d'articles et met toutes les informatio
le concernant dans un fichier csv nommé article.csv .'''

def csv_file(i):
    with open('article.csv', 'w') as output_file:
        dict_writer = csv.DictWriter(output_file, headers)
        dict_writer.writeheader()
        dict_writer.writerows(get_all_article(i))
csv_file(14)
```

Entrée [24]:

```
import pandas as pd
pd.read_csv('article.csv')
```

Out[24]:

	title	url	article_name	author	date	comment_number
0	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Calm in the Political Storm	Wailin Wong	November 26, 2019	0
1	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	The joy and power of being the independent und...	Jonas Downey	November 22, 2019	8
2	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Spending in the Clouds	Wailin Wong	November 19, 2019	17
3	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	7 leadership lessons over 2.5 years	Claire Lew	November 18, 2019	2
4	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Breaking the Black Box	Wailin Wong	November 15, 2019	1
5	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Launch: Basecamp Gets Personal	Jason Fried	November 12, 2019	38
6	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Big Brother at the Office	Wailin Wong	November 12, 2019	0
7	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Compounding time	Jason Fried	November 5, 2019	11
8	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Rework Mailbag	Wailin Wong	November 5, 2019	0
9	November 2019 - Signal v. Noise	https://m.signalvnoise.com/2019/11/	Back to windows after twenty years	DHH	November 4, 2019	92

Entrée [27]:

```python
''' Cette fonction retourne tous les articles des 67 groupes d'articles avec leur info
('title', 'url', 'article_name', 'author', 'date', 'comment_number')
'''

def all_csv_file():
 dataframe=[]
 for i in range(67):
   csv_file(i)
   df=pd.read_csv('article.csv')
   dataframe.append(df)
 resultat=pd.concat(dataframe)
 return resultat
```

Entrée [28]:

```python
all_csv_file()
```

Out[28]:

	title	url	article_name	author	date	comment_number
0	February 2021 - Signal v. Noise	https://m.signalvnoise.com/2021/02/	Testimony before the North Dakota Senate Indus...	DHH	February 9, 2021	0
0	January 2021 - Signal v. Noise	https://m.signalvnoise.com/2021/01/	Reiterating our Use Restrictions Policy	Jason Fried	January 18, 2021	0
0	December 2020 - Signal v. Noise	https://m.signalvnoise.com/2020/12/	HTML over the wire	DHH	December 23, 2020	0
1	December 2020 - Signal v. Noise	https://m.signalvnoise.com/2020/12/	Validation is a mirage	Jason Fried	December 22, 2020	7
2	December 2020 - Signal v. Noise	https://m.signalvnoise.com/2020/12/	The Making of a Dumpster Fire	Andy Didorosi	December 15, 2020	19
...	...	...	...	...	...	...
7	September 2015 - Signal v. Noise	https://m.signalvnoise.com/2015/09/	Disruption is better when it’s other people’s ...	DHH	September 22, 2015	0
8	September 2015 - Signal v. Noise	https://m.signalvnoise.com/2015/09/	Reminder: Design is still about words	Mig Reyes	September 22, 2015	0
9	September 2015 - Signal v. Noise	https://m.signalvnoise.com/2015/09/	It’s OK not to use tools	Jonas Downey	September 22, 2015	0
0	June 2014 - Signal v. Noise	https://m.signalvnoise.com/2014/06/	How I managed to get Tim Ferriss to advise me,...	Nathan Kontny	June 26, 2014	0
0	November 2013 - Signal v. Noise	https://m.signalvnoise.com/2013/11/	Business Failing? You Might Be Asking The Wron...	Nathan Kontny	November 14, 2013	0

543 rows × 6 columns

Nous avons donc en sommes 543 articles répartis dans les 67 groupes d'articles

RESUME DE CODE

Entrée [ ]:

```python
import requests
from bs4 import BeautifulSoupimport requests
from bs4 import BeautifulSoup
import csv
import pandas as pd
url='https://m.signalvnoise.com/search/'
response=requests.get(url)
doc = BeautifulSoup(response.text)
article_tags=doc.findAll('li')  # Permet de récupérer toutes les catégories d'article
article_tag=article_tags[4] # Renvoie la catégorie d'articles en position 4(Septembre

'''
Cette fonction prends en parametre la variable qui renvoie la catégorie d'articles en
et qui retourne  l'url permettant d'acceder à ce groupe d'articles.
Nous avons en sommes 67 groupes d'articles. Dans la suite i varie entre 0 et 66
'''
def parse_articl(article_tag):
    a_tags = article_tag.find('a')

    url = a_tags['href']

    return url

'''
  Permet de récupérer sous forme de liste tous les urls permettant d'acceder aux diffe
'''

def get_top_articl(article_tags):
    all_article = [parse_articl(tag) for tag in article_tags]
    return all_article
get_top_articl=get_top_articl(article_tags)

'''Prends en paramètre la position du groupe d'articles et retourne un dictionnaire ay
response qui récupère son url , soup pour analyser son contenu et article pour qui ren
Un groupe d'articles contient un ou plusieurs articles '''

def url_article(i):
    response=requests.get(get_top_articl[i])
    soup = BeautifulSoup(response.text,'html.parser')
    article=soup.findAll('article',class_="entry-summary grid__item grid__item--third'
    return {'response':response,
            'soup':soup,
            'article':article}

'''Prends en paramètre la position du groupe d'articles et l'indice de l'article conte
         et retourne ainsi cet article'''

def article_tag(i,j):
    article=url_article(i)['article']
    art_tag=article[j]
    return art_tag


'''Retourne le nombre de commentaires que renferme un article contenu dans un groupe

def comments(i,j):
    try:
        d=article_tag(i,j).find('div')
        c=d.find('span' ,class_="comments-link").text.strip()
    except (AttributeError):
        c='0'

    else:
        c= (c[0:2])
    return c

''' Fonction qui retourne le titre et l'url de de la catégorie,le titre, l'auteur,la
commentaires de l'article.'''

def find_article(i,j):
    soup = url_article(i)['soup']
    a_tags = article_tag(i,j).findAll('div')
    t_tags = article_tag(i,j).find('time')
    title = soup.title.text
    url = get_top_articl[i]
    article_name=article_tag(i,j).find('a').text.strip()
    author=article_tag(i,j).find('a',class_="author").text.strip()
    date=t_tags.text.strip()
    comment_number=comments(i,j)

    return {
        'title':title,
        'url': url,
        'article_name':article_name,
        'author':author,
        'date':date,
        'comment_number':comment_number,

    }

'''Retourne tous les articles concernant un groupe d'articles avec ses informations''
def get_all_article(i):

    all_article = [find_article(i,j) for j in range(len(url_article(i)['article']))]
    return all_article

#Une liste des noms des colonnes de notre dataframe
headers = list(get_all_article(0)[0].keys())


''' Prends en parametre la position du groupe d'articles et met toutes les informatio
le concernant dans un fichier csv nommé article.csv .'''

def csv_file(i):
    with open('article.csv', 'w') as output_file:
        dict_writer = csv.DictWriter(output_file, headers)
        dict_writer.writeheader()
        dict_writer.writerows(get_all_article(i))
csv_file(14)

pd.read_csv('article.csv')

''' Cette fonction retourne tous les articles des 67 groupes d'articles avec leur inf
('title', 'url', 'article_name', 'author', 'date', 'comment_number')
'''

def all_csv_file():
    dataframe=[]
    for i in range(67):
        csv_file(i)
        df=pd.read_csv('article.csv')
        dataframe.append(df)
    resultat=pd.concat(dataframe)
    return resultat

```