# Sentiment Analysis of Stock Market in Python (Part 1)- Web Scraping Financial News

Bee Guan Teo Follow
Oct 4, 2021 · 7 min read ★

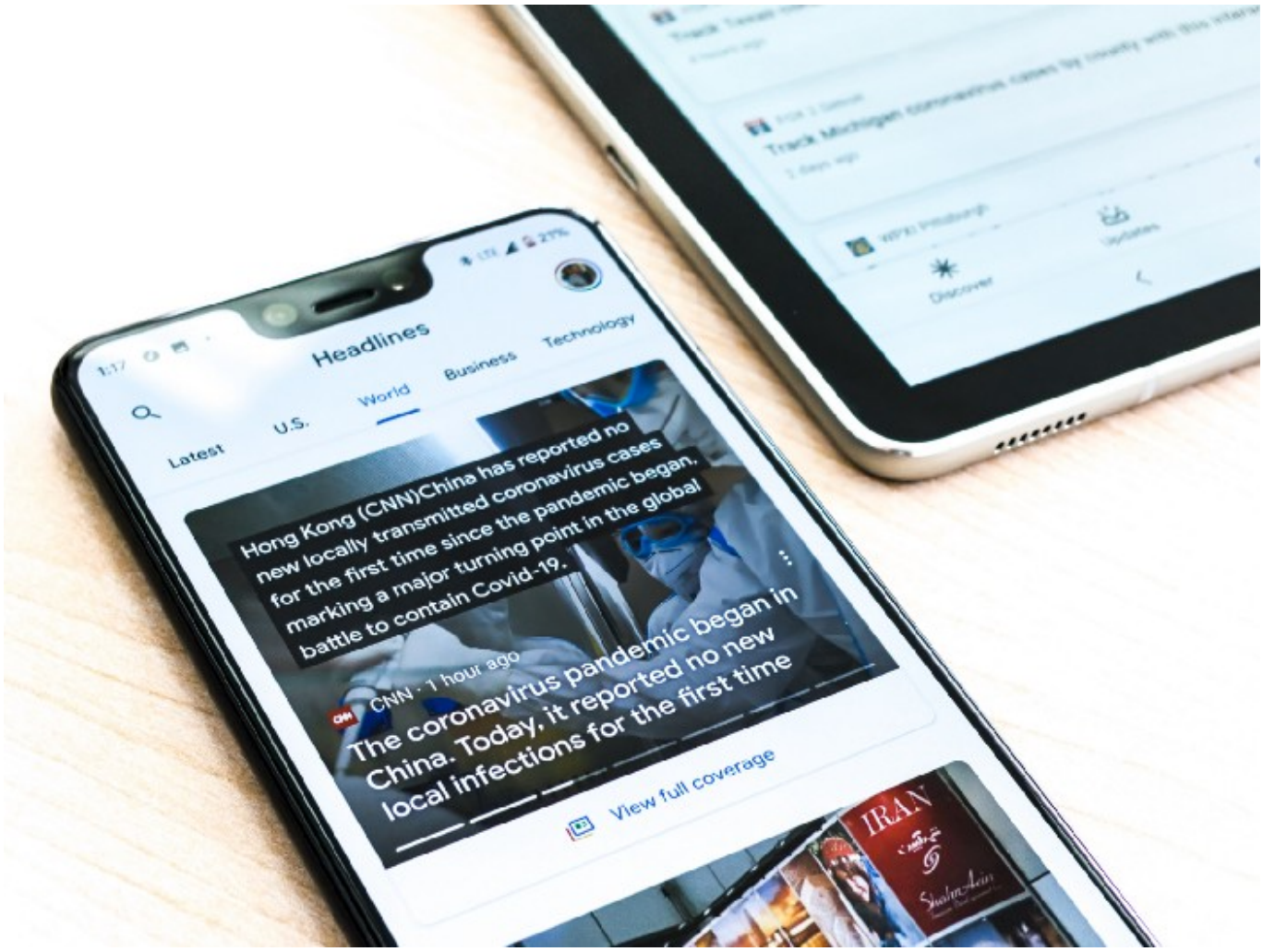



Photo by Obi Onyeador on Unsplash

Stock market sentiments can be valuable info that hints at future price action. Many often stock investors react to the market sentiments in making their decision to buy or

sell their assets. Hence, **stock sentiment analysis** has become a popular and useful technique to gauge the investors' opinions of a specific stock and plan for an investment…

One direct way to understand market sentiments is by following and reading the news on daily basis. However, this can be quite a tedious process. Here, we are going to explore how can we use **Python** to perform the stock sentiment analysis for us.

We will break this sentiment analysis process into two main parts:

1. **Web scraping financial news and preprocessing the text data**
2. **Calculating sentiment score and visualization (Presented in Part 2 Article)**

In this article, we will only focus on the **first part** and the second part will be presented in another article.

***Disclaimer**: The writing of this article is only aimed at demonstrating the steps to perform stock market sentiment analysis in Python. It doesn't serve any purpose of promoting any stock or giving any specific investment advice.*

## Prerequisite Python Packages

1. **BeautifulSoup** — https://pypi.org/project/beautifulsoup4/
2. **Pandas** — https://pandas.pydata.org/
3. **NLTK** — https://pypi.org/project/nltk/ (*Will be used in the Part 2 Article*)

## Github

The original full source codes presented in this article are available on my Github Repo. Feel free to download it (*SentimentAnalysis_part1.py*) if you wish to use it to follow my article.

## Web Scraping Financial News

### 1. Identifying sources of financial news

Firstly, we need to identify the source of the financial news where we would like to gather the sentiment data. There are many potential sources such as Google Finance, Yahoo Finance, FINVIZ, MarketWatch, etc.

In this article, we are going to gather our sentiment data from **Financial Modeling Prep (FMP)**.

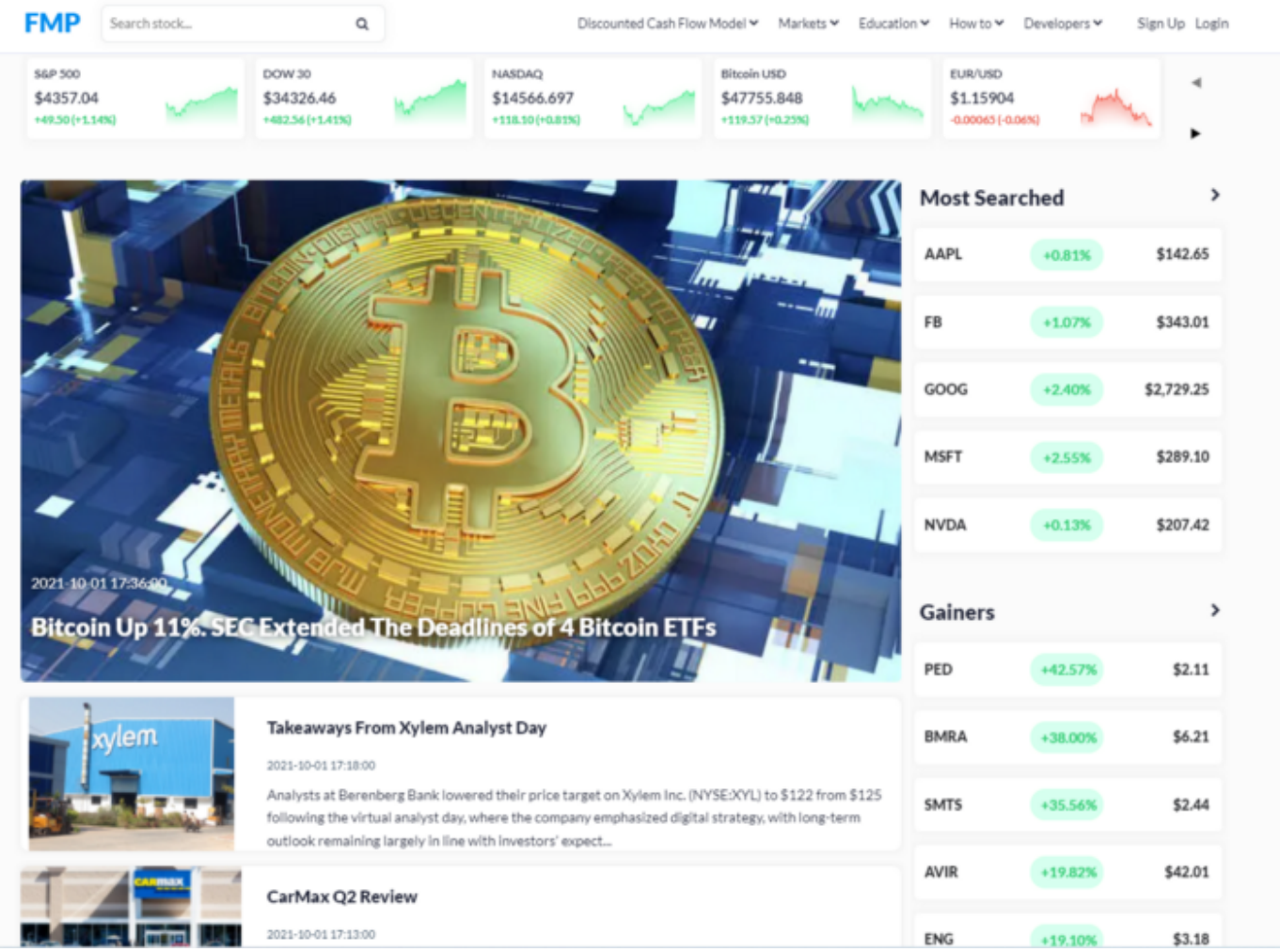



Image Prepared by the Author

FMP offers us clean and well structured financial information. We can simply type a ticker symbol "*AAPL*" in the search bar at the top-left corner to search for further details

of Apple stock.

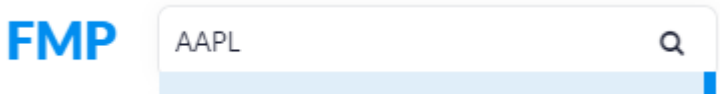



Image Prepared by the Author

The search will lead us to a financial summary page as below.

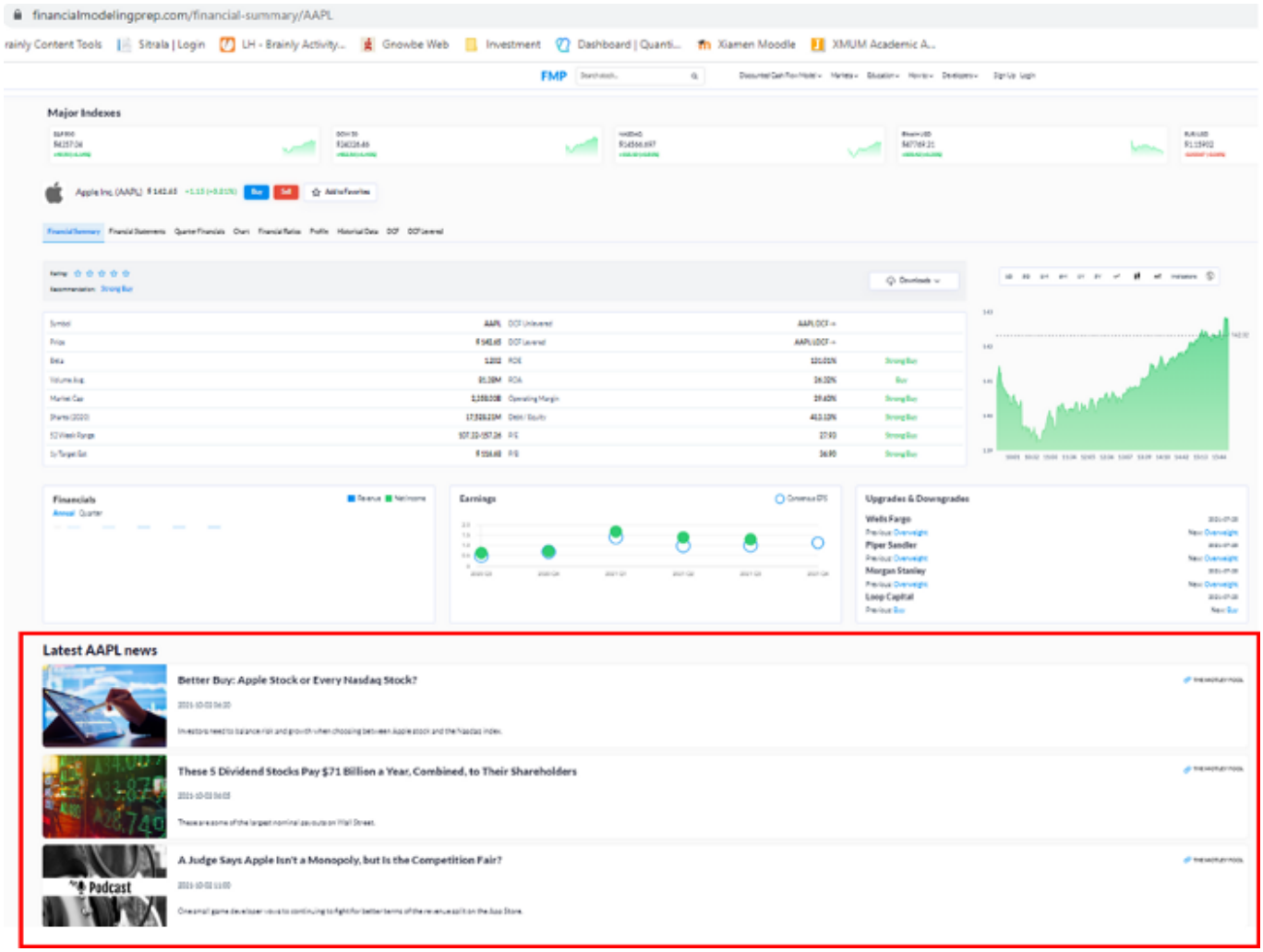
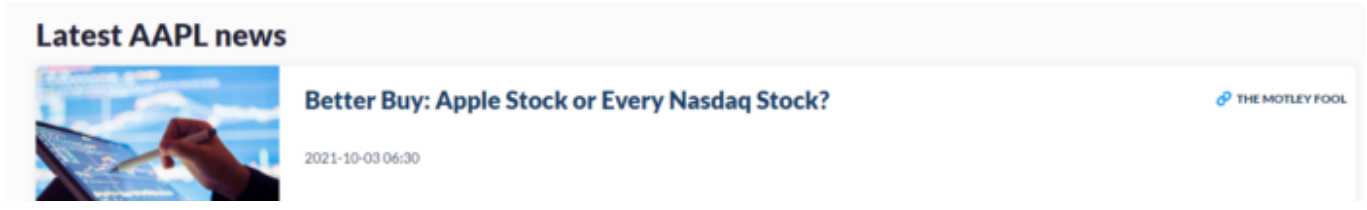



Image Prepared by the Author

If we look at the bottom part of the financial summary page of AAPL, there is a list of the latest AAPL news.



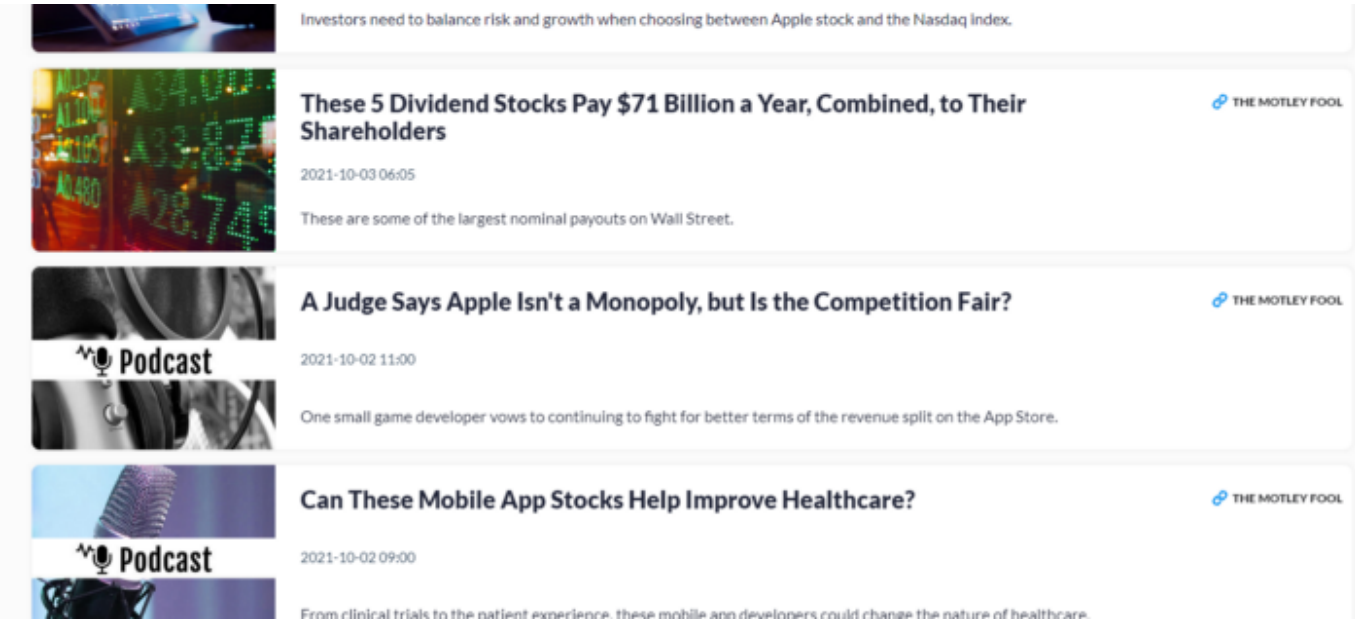



Image Prepared by the Author

The news is the sources of our sentiments that we will extract for sentiment analysis using Python.

## 2. Examining HTML Structure of Web Page

To extract the financial news, we will first need to examine the HTML structure of the page. **HTML** is a markup language that lays down the structure of a webpage.

We can right-click on the web page and click "**Inspect**" to view the HTML codes. *(This step is done by presuming we are using Google Chrome)*.

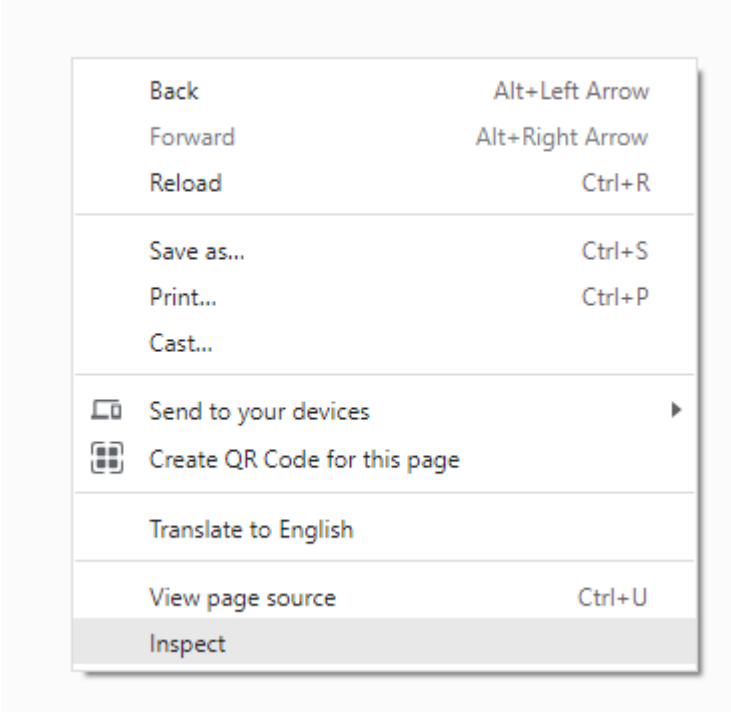
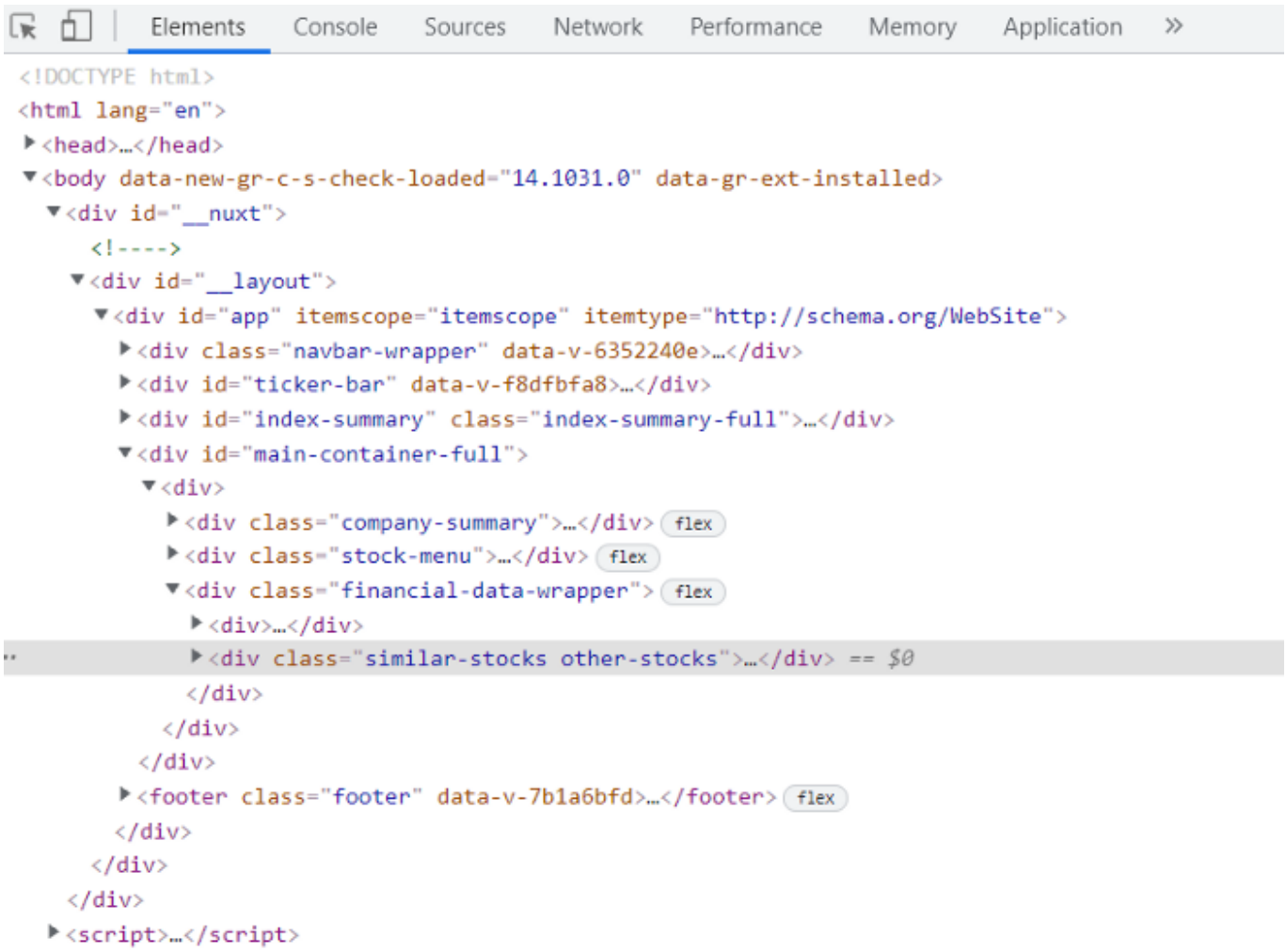



Image Prepared by the Author

We shall see the HTML codes that render our web page as below.



```
<script src="https://d14ccuzudddlhp.cloudfront.net/_nuxt/9efbc9b.js" defer></script>
<script src="https://d14ccuzudddlhp.cloudfront.net/_nuxt/a44303a.js" defer></script>
<script src="https://d14ccuzudddlhp.cloudfront.net/_nuxt/784cfc1.js" defer></script>
```

Image Prepared by the Author

We can traverse through the HTML tags to hunt for the tag that is responsible to render the news content. We do it by placing our mouse cursor on each of the tags (e.g. div) and examine the highlighted area of the webpage. Besides, we can also click on the "*triangle*" shape button to expand the HTML tags.

We will find that the news contents are rendered by a "div" with a class name "articles".

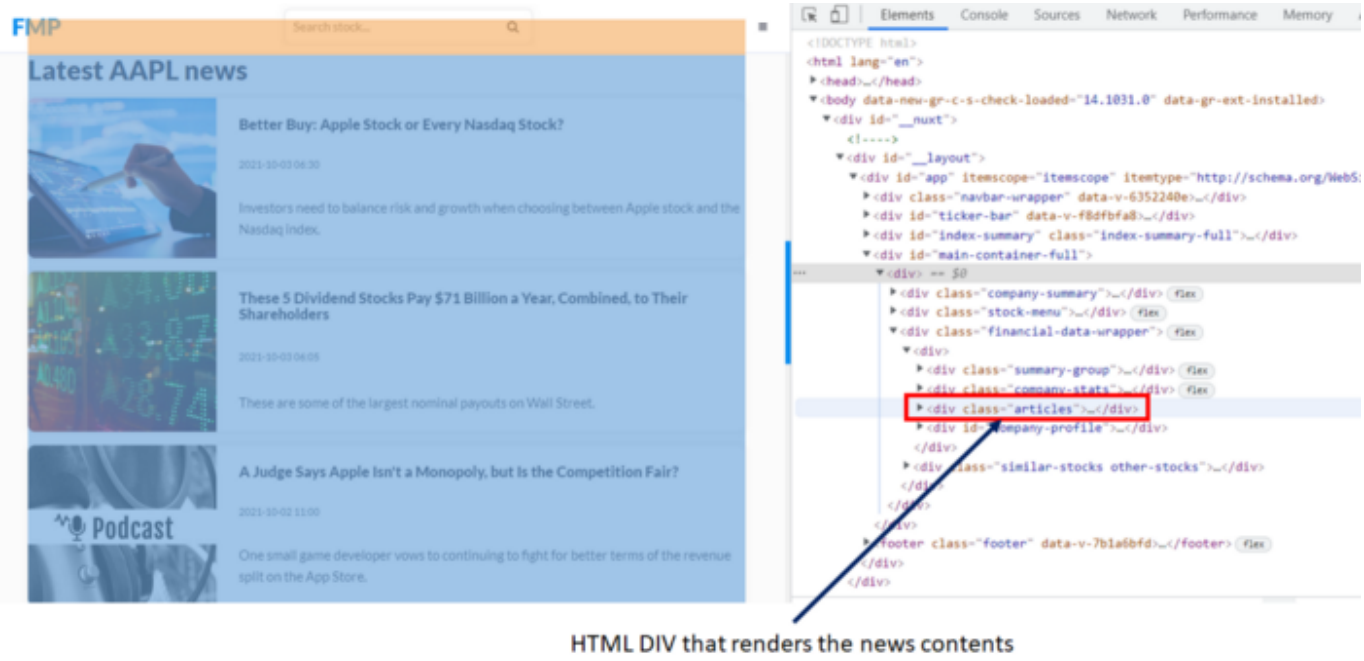



Image Prepared by the Author

If we try to expand the *div class= "article"* further, we shall see the news contents are wrapped inside an anchor tag *<a>* with a class name, *article-item*. Inside the anchor tag, the news' title, date and text are marked up by *h4, h5* and *p* tags, respectively.

```
▼<div class="articles">
    <h2>Latest AAPL news</h2>
  ▼<div>
    ▼<a target="_blank" rel="noopener" href="https://www.fool.com/investing/2021/10/03/better-buy-apple-stock-or-every-nasdaq-stock/" class="article-item"> flex
      ▶<div class="articles__img">…</div> flex
      ▼<div class="article-rightside"> flex
          <h4 class="article-title">Better Buy: Apple Stock or Every Nasdaq Stock?</h4>
          <h5 class="article-date">2021-10-03 06:30</h5>
```

```
<p class="article-text">
  "Investors need to balance risk and growth when choosing between Apple stock
  and the Nasdaq index."
</p>
</div>
<div class="article-source">…</div>
</a>
</div>
```

Image Prepared by the Author

A similar HTML structure, as shown above, is repetitive for all the news on the web page.

Our next task is to use Python to perform the web scraping on the financial news page.

## 3. Extracting HTML contents

Now, we are going to use Python to extract the content of the financial web page. To do so, let us examine the URL of the financial web page again. We can see the URL can be split into two components: **a static base URL**, and **a ticker**.


Image Prepared by the Author

Based on this observation, we can generate a dynamic link to the FMP financial page for different tickers.

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

ticker = "AAPL"
url = "https://financialmodelingprep.com/financial-summary/" + ticker
request = requests.get(url)
```

```
print(request.text)
```

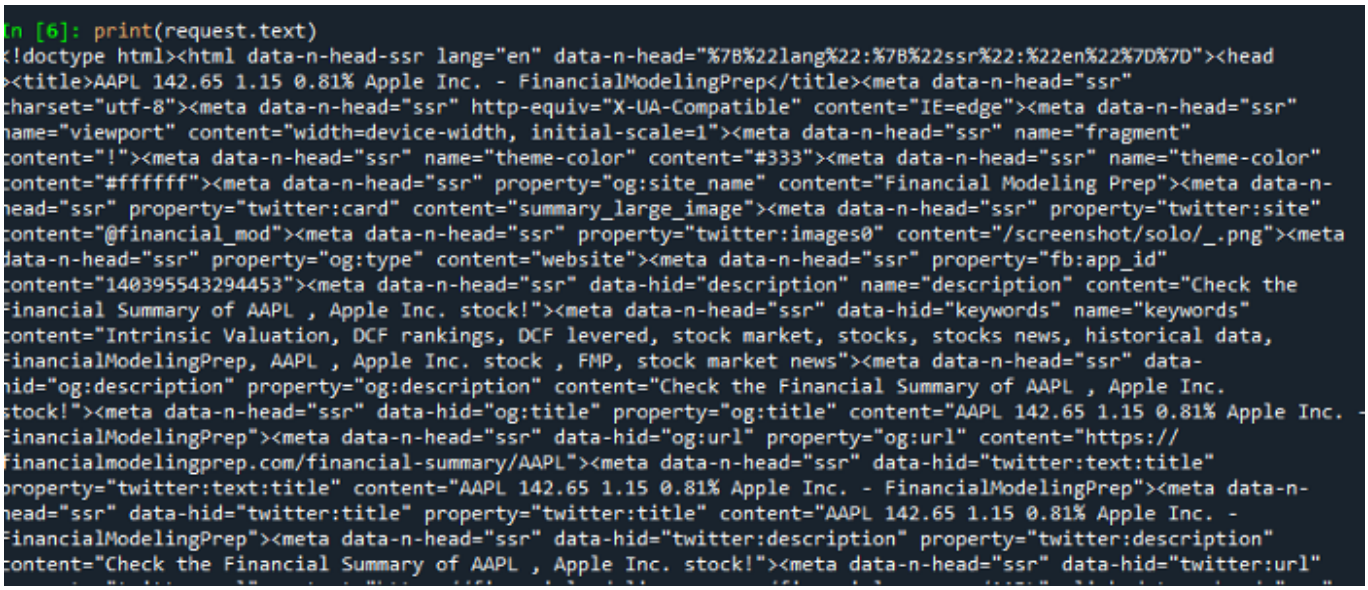
WebScraping_part1.py hosted with ♡ by GitHub view raw

Image Prepared by the Author

**Line 1–3:** Import all the required Python packages.

**Line 5–6:** Set a ticker (e.g. *AAPL*). Generate a URL to the FMP page for the ticker by joining the base URL with the ticker.

**Line 7:** We use the ***Python requests module's get method*** to start an HTTP request to the FMP website routed by the dynamic URL. This will return the web page content for us.



```
In [6]: print(request.text)
<!doctype html><html data-n-head-ssr lang="en" data-n-head="%7B%22lang%22:%7B%22ssr%22:%22en%22%7D%7D"><head
><title>AAPL 142.65 1.15 0.81% Apple Inc. - FinancialModelingPrep</title><meta data-n-head="ssr"
charset="utf-8"><meta data-n-head="ssr" http-equiv="X-UA-Compatible" content="IE=edge"><meta data-n-head="ssr"
name="viewport" content="width=device-width, initial-scale=1"><meta data-n-head="ssr" name="fragment"
content="!"><meta data-n-head="ssr" name="theme-color" content="#333"><meta data-n-head="ssr" name="theme-color"
content="#ffffff"><meta data-n-head="ssr" property="og:site_name" content="Financial Modeling Prep"><meta data-n-
head="ssr" property="twitter:card" content="summary_large_image"><meta data-n-head="ssr" property="twitter:site"
content="@financial_mod"><meta data-n-head="ssr" property="twitter:images0" content="/screenshot/solo/_.png"><meta
data-n-head="ssr" property="og:type" content="website"><meta data-n-head="ssr" property="fb:app_id"
content="140395543294453"><meta data-n-head="ssr" data-hid="description" name="description" content="Check the
Financial Summary of AAPL , Apple Inc. stock!"><meta data-n-head="ssr" data-hid="keywords" name="keywords"
content="Intrinsic Valuation, DCF rankings, DCF levered, stock market, stocks, stocks news, historical data,
FinancialModelingPrep, AAPL , Apple Inc. stock , FMP, stock market news"><meta data-n-head="ssr" data-
hid="og:description" property="og:description" content="Check the Financial Summary of AAPL , Apple Inc.
stock!"><meta data-n-head="ssr" data-hid="og:title" property="og:title" content="AAPL 142.65 1.15 0.81% Apple Inc. -
FinancialModelingPrep"><meta data-n-head="ssr" data-hid="og:url" property="og:url" content="https://
financialmodelingprep.com/financial-summary/AAPL"><meta data-n-head="ssr" data-hid="twitter:text:title"
property="twitter:text:title" content="AAPL 142.65 1.15 0.81% Apple Inc. - FinancialModelingPrep"><meta data-n-
head="ssr" data-hid="twitter:title" property="twitter:title" content="AAPL 142.65 1.15 0.81% Apple Inc. -
FinancialModelingPrep"><meta data-n-head="ssr" data-hid="twitter:description" property="twitter:description"
content="Check the Financial Summary of AAPL , Apple Inc. stock!"><meta data-n-head="ssr" data-hid="twitter:url"
```

Image Prepared by the Author

## 4. Parsing web content

The raw web content is not really useful for us as they look like some gibberish texts mixed with too much unnecessary info for our sentiment analysis. Here, we need to parse our extracted web contents and look only for the targeted financial news section.

We will use another Python module, ***BeautifulSoup*** to parse our web content.

```
parser = BeautifulSoup(request.text, "html.parser")
news_html = parser.find_all('a', {'class': 'article-item'})
print(news_html[0])

```

```
sentiments = []
for i in range(0, len(news_html)):
    sentiments.append(
            {
                'ticker': ticker,
                'date': news_html[i].find('h5', {'class': 'article-date'}).text,
                'title': news_html[i].find('h4', {'class': 'article-title'}).text,
                'text': news_html[i].find('p', {'class': 'article-text'}).text
            }
        )
```

WebScraping_part2.py hosted with ♡ by GitHub view raw

**Line 1:** Use the BeautifulSoup module to create a parser for our extracted web content.

**Line 2:** From the previous section, we have known our targetted news content are wrapped inside the anchor tag with a class name "*article-item*". Now, we use the parser's *find_all* method to look for all the anchor tags with a class name '*article-item*'. This will give us a list of news info wrapped inside the anchor tags and we store the parsed info in *news_html*.

**Line 3:** We print the first item of parsed info as a sample to visualize the news info we have managed to extract so far.

```
In [9]: print(news_html[0])
<a class="article-item" href="https://www.fool.com/investing/2021/10/03/better-buy-apple-stock-or-every-nasdaq-stock/" rel="noopener" target="_blank"><div class="articles__img"><img alt="Better Buy: Apple Stock or Every Nasdaq Stock?" src="https://cdn.snapi.dev/images/v1/e/9/the-3-worst-looking-dow-stocks-from-a-price-chart-perspective-1003611-1013437-1032443.jpg"/></div> <div class="article-rightside"><h4 class="article-title">Better Buy: Apple Stock or Every Nasdaq Stock?</h4> <h5 class="article-date">2021-10-03 06:30</h5> <p class="article-text">Investors need to balance risk and growth when choosing between Apple stock and the Nasdaq index.</p></div> <div class="article-source"><svg fill="none" height="15" stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="3" viewbox="0 0 24 24" width="15" xmlns="http://www.w3.org/2000/svg"><path d="M10 13a5 5 0 0 0 7.54.54l3-3a5 5 0 0 0-7.07-7.07l-1.72 1.71"></path><path d="M14 11a5 5 0 0 0-7.54-.54l-3 3a5 5 0 0 0 7.07 7.07l1.71-1.71"></path></svg> <h5>The Motley Fool</h5></div></a>
```

Image Prepared by the Author

From the result above, apart from our targetted **article's title, date and text,** we can still find a lot of unwanted info**.** In the following line of code, we will narrow down our search to only extract the three relevant pieces of info from each of the anchor tags stored in the *news_html*.

**Line 5–14:** Create a sentiments list to hold the target news info. Create a for-loop to traverse through the anchor tags and in every loop, we use the parser's *find* method to

search for the *h5, h4 & p* tags where the news date, title and text are placed. We also specify the associated class names (*article-date, article-title, article-text*) so that the parser can identify and return the correct info as below:

```
news_html[0].find('h5', {'class': 'article-date'}).text
'2021-10-03 06:30'

news_html[0].find('h4', {'class': 'article-title'}).text
('Better Buy: Apple Stock or Every Nasdaq Stock?',)

news_html[0].find('p', {'class': 'article-text'}).text
'Investors need to balance risk and growth when choosing between Apple stock and the Nasdaq index.'
```

Image Prepared by the Author

We encapsulate the news info such as the ticker, date, title and text into a Python dictionary and append the dictionary to the sentiments list in every round of loop.

At the end of the loop, our news info is captured in a list of Python dictionaries.

| Inde | Type | Size | Value |
|---|---|---|---|
| 0 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-04 07:40', 'title':'Hedge Fund Manag ... |
| 1 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-03 06:30', 'title':'Better Buy: Appl ... |
| 2 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-03 06:05', 'title':'These 5 Dividend ... |
| 3 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-02 11:00', 'title':'A Judge Says App ... |
| 4 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-02 09:00', 'title':'Can These Mobile ... |
| 5 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-01 17:53', 'title':'Apple, Microsoft ... |
| 6 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-01 14:09', 'title':'Apple Could Be T ... |
| 7 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-01 13:38', 'title':'Apple: Software ... |
| 8 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-01 12:10', 'title':'Nvidia Vs. Apple ... |
| 9 | dict | 4 | {'ticker':'AAPL', 'date':'2021-10-01 11:14', 'title':'Apple price pred ... |

Image Prepared by the Author

## 5. Converting Python List to Pandas Dataframe

To ease our task to perform sentiment analysis in the later stage, we can convert our Python list of news info into a Pandas Dataframe.

```
1  df = pd.DataFrame(sentiments)
2  df = df.set_index('date')
```

WebScraping_part3.py hosted with ♡ by GitHub view raw

**Line 1:** Use the *Pandas DataFrame* method to convert our Python list of news info into a dataframe.

**Line 2:** Use the *set_index* method to set the date as the index of the dataframe.

| date | ticker | title | text |
|---|---|---|---|
| 2021-10-04 07:40 | AAPL | Hedge Fund Managers Are Loading Up on Top Technology Stocks | To say that hedge fund and mutual fund mana… |
| 2021-10-03 06:30 | AAPL | Better Buy: Apple Stock or Every Nasdaq Stock? | Investors need to balance risk and growth w… |
| 2021-10-03 06:05 | AAPL | These 5 Dividend Stocks Pay $71 Billion a Year, Combined, to Their Shareholders | These are some of the largest nominal payouts on Wall Street. |
| 2021-10-02 11:00 | AAPL | A Judge Says Apple Isn't a Monopoly, but Is the Competition Fair? | One small game developer vows to continuing… |
| 2021-10-02 09:00 | AAPL | Can These Mobile App Stocks Help Improve Healthcare? | From clinical trials to the patient experie… |
| 2021-10-01 17:53 | AAPL | Apple, Microsoft, Disney And Amazon Lobby Against Climate Bill: Report | Apple Inc. (NASDAQ: AAPL), Amazon.com, Inc.… |
| 2021-10-01 14:09 | AAPL | Apple Could Be Targeting Facebook, Google Advertising Revenue Stream Via Latest Move | Apple Inc's (NASDAQ: AAPL) recent privacy c… |
| 2021-10-01 13:38 | AAPL | Apple: Software Tilt With Increasing Services Revenues | More than just the iPhone, the iMac, and th… |
| 2021-10-01 12:10 | AAPL | Nvidia Vs. Apple: How To Analyze These Two Stocks | Apple is the most prominent company of the … |
| 2021-10-01 11:14 | AAPL | Apple price prediction: Gene Munster sees upside to $200 | Shares of Apple Inc (NASDAQ: AAPL) are down… |

Image Prepared by the Author

## Conclusions

In this Part 1 Article, we have managed to web scrape the news info from the FMP website and also preprocess them into a dataframe format to be ready for the sentiment analysis later. The web scraping and HTML parsing are simple and straightforward and they are applicable to garner a variety of info from other web resources. Hence, you can

also treat this article as an independent article guide to web scrape online sources for analysis.

**In the Part 2 article, we will go through the process of sentiment analysis using the NLTK module.**

I wish you enjoy reading this article.

## Subscribe to Medium

***If you like my article and would like to read more similar articles from me or other authors, feel free to subscribe to Medium. Your subscription fee will partially go to me. This can be a great support for me to produce more articles that can benefit the community.***

## References

1. https://www.dailyfx.com/education/understanding-the-stock-market/stock-market-sentiment-analysis.html
2. https://www.investopedia.com/terms/m/marketsentiment.asp



Python Finance Programming Data Science Sentiment Analysis