



Scraping S&P 500 index and extracting stock market data from Yahoo Finance API.

26.12.2020

---

Madeleine Moghadasi

Data Analysis| Breaker of the newgrounds

Toronto, ON

## Overview

Stock trading is among the most sophisticated processes in the world. There have been increased efforts to learn about stock trading systems in hopes of predicting the way stock prices behave.

Stock prices can be predicted better if one has access to historical data on stock prices. This can be done in two ways: through data service providers or through simple web scrapers. We can scrape websites that give us stock price data for this job. Stock data can be found on many websites, including one of the most popular, Yahoo Finance.

In this article, we will focus on getting the list of companies from the S&P 500 via scraping Wikipedia, and then extracting stock-related data from Yahoo. We will display the web scraping implementation step by step, so that you will be able to grasp it easily.

## Goals

1. getting the list of companies from the S&P 500 via scraping Wikipedia
2. extracting stock-related data for S&P 500 companies from Yahoo

## Specifications

The S&P 500 consists of 500 companies representing all the sectors of the economy. The index covers only large-cap companies listed on the U.S. market, either the New York Stock Exchange or the Nasdaq. Because the

S&P 500 represents the largest publicly traded companies in the U.S., it is considered one of the most widely quoted stock market indexes. Those companies in the S&P 500 account for three-quarters of all US stocks.

## Milestones

### I. Web Scraping

First we need to import the required libraries. We scraped the Wikipedia table containing the companies' tickers or symbols using BeautifulSoup. We retrieve the source code from wikipedia using requests, and then use Pickle to save it afterwards.

#1

Before we extract the list, we need to have a look at the source code. You can see it by right-clicking somewhere in your browser and selecting *inspect element* (press *F12* on Windows). The *html* coding can be viewed below the `<table>` tag.

The screenshot shows a web browser window with the title "S&P 500 component stocks [edit]". The main content is a table with the following columns: Symbol, Security, SEC filings, GICS Sector, GICS Sub-Industry, Headquarters Location, and Date first added. The table lists several companies, including MMM (3M Company), ABT (Abbott Laboratories), ABBV (AbbVie Inc.), ABMD (ABIOMED Inc.), and ACN (Accenture plc). To the right of the table, the browser's developer tools are open, showing the HTML source code for the table. The code includes the table's structure, including the header and body rows, and the class attributes for styling and sorting.

Symbol	Security	SEC filings	GICS Sector	GICS Sub-Industry	Headquarters Location	Date first added
MMM	3M Company	reports	Industrials	Industrial Conglomerates	St. Paul, Minnesota	1976-08-09
ABT	Abbott Laboratories	reports	Health Care	Health Care Equipment	North Chicago, Illinois	1964-03-31
ABBV	AbbVie Inc.	reports	Health Care	Pharmaceuticals	North Chicago, Illinois	2012-12-31
ABMD	ABIOMED Inc.	reports	Health Care	Health Care Equipment	Danvers, Massachusetts	2018-05-31
ACN	Accenture plc	reports	Information Technology	IT Consulting & Other Services	Dublin, Ireland	2011-07-06

This is then converted into a BeautifulSoup Object, that can be manipulated like a Python object. To put it more specifically, the script makes an *http* request, after which it parses the Wiki page with BeautifulSoup. BeautifulSoup allows us to extract data from *HTML* in a format that resembles a parse tree.

#2

Next, we loop through the table. *Findall* method in BeautifulSoup will extract the text from the *html* table. Each row of the table is denoted by the `<tr>` tag in the *html* code. The header row needed to be removed from the table (i.e. `[1:]`). We create an empty list and append the tickers to it. Each row includes the ticker which is the table data `<td>`. Note that in the process of extracting data from Wikipedia, it generates “\n” at the end of each string. Therefore, it should be deleted before being included in the list.

#3

In this stage, we can save the list using Pickle. Alternatively, we could convert it to a dataframe, but we go with the former.

#4

## II. Get stock price data

In this section, we are going to obtain the Financials of the S&P 500 companies we scraped in the previous section. We will use Yahoo Finance data with the help of Pandas, a package that is one of the best Python packages.


We add a few more libraries, including *datetime* to specify the date, os to create and check directories.

#5

We need to access the pickle file we saved previously. I would then store each individual stock information in its own separate directory. It is optional, one can either save all the data of 500 companies in one file or split the data into separate files.

#6

This is the same process as before, so we iterate through the ticker list and get Yahoo data. Occasionally, Wikipedia may use different



tickers to represent the same company, like Berkshire Hathaway Inc. (BRK-B), however, the abbreviation in Yahoo seems to be “BRK.B” so we need to replace these special characters before we go on to encounter any errors.

#7

All the information we need is within our grasp, so it is time to put on our thinking caps and begin creating some useful and exciting analyses of stock data.