towards
data science

Follow 593K Followers

A Simple Way to Get a Stock's Fundamental Data

How I used Python and API calls to retrieve a company's financial statements and earnings reports

Marco Santos 1 day ago · 5 min read★

Photo by Sigmund on Unsplash

Retrieving or scraping stock data is sometimes easier said than done. It can be fairly difficult to find the right resources or websites containing the appropriately, relevant data. In my data science projects, over the past few years, I have needed to use many different datasets and quite a few of them involved stock data and analysis.

In my experience, a stock's price history is one of the easiest pieces of data to find and retrieve. Usually, I would resort to using Yahoo Finance and the accompanying Python libraries to access this historical price data. However, I have found that the same cannot be said for a stock's fundamental data.

To find fundamental data, such as financial statements and earnings reports, you would potentially need to do some extensive Google searching and tiresome web-scraping. In a previous project of mine, I was fortunate to stumble upon a now defunct website called stockpup.com. This site contained thousands of rows of fundamental data for thousands of stocks. And they were all neatly organized into a simple CSV file for me to download and use for my machine learning projects.

Since that website was no longer available, I had to find that data elsewhere…

> *Sign up for a Medium Membership here to gain unlimited access and support content like mine! With your support I earn a small portion of the membership fee. Thanks!*

. . .

Financial Data APIs

There are numerous financial data APIs out there that can give you access to a stock's fundamental data. Personally, I have found and use one website which is able to provide more than just fundamental data and it is free to sign up called — **eodhistoricaldata.com**. *Disclosure: I earn a small commission from any purchases made through the link above*.

Using this financial data API, I was able to retrieve the fundamental data from thousands of companies. The data was ready and available but now I needed to organize and format it into a Pandas DataFrame that I'll be able to use. To do so, I utilized Python to get everything set up:

```
# Libraries
import pandas as pd
from eod import EodHistoricalData
from functools import reduce
from datetime import datetime, timedelta

# Importing and assigning the api key
with open("../eodHistoricalData-API.txt", "r") as f:
    api_key = f.read()

# EOD Historical Data client
client = EodHistoricalData(api_key)
```

With my provided API key and the code above, I was ready to begin retrieving and formatting the fundamental data.

. . .

Getting Fundamental Data

The API I used is able to provide me with quarterly fundamental data such as Balance Sheets, Income Statements, Cash Flow, and Earnings Reports but they are all neatly stored in their own sections within the API's return object. I needed to create function that is able to access, then consolidate all of this data into one large DataFrame.

So I created a function that is able to transform these separate pieces of data into their own DataFrame, then merge them together into one big DF:

In this function, I was able to retrieve fundamental data from the given stock ticker. As you can see, each financial statement is stored within a specific attribute I needed to access. Once I have stored them into their own respective DataFrames, I was then able to consolidate them by merging them together. I also removed any redundant columns such as the "*Date*" column which was being repeated throughout before I dropped them and any duplicate columns as well.

. . .

Getting Historical Price Data

One thing that I found that was missing from the returned fundamental data was the stock price at the time. I figured that would be an important piece of information to have. The API is able to retrieve historical prices fairly easy as well so that wasn't going to be a problem but I needed to combine those prices with the DataFrame I created above.

I created a function that was able to retrieve the daily historical prices and added them to the larger DataFrame:

A problem I have found to occur when I was initially adding the prices to the DataFrame was that some price data was missing for certain dates on the larger DF. I assume those dates were sometimes either holidays, weekends, or something similar. Regardless, I just wanted to know what the stock price was around the reported date from the financial statements. It didn't need to be exact, and in fact it could be just a day or two before or after the reported date.

As you can see in the function above, I was able to fill in weekends, holidays, etc. with the previous stock price. Afterwards, I added them to the larger DataFrame which was returned in the end.

. . .

Getting Both Fundamental and Price Data

Since I was able to create two functions to retrieve the fundamental and price data, I decided to condense them into one function:

In this one function, I incorporated the previous two functions. I also added the option for additional cleaning of the data by providing choice to remove null values within the larger DF. This option could be useful if you needed to train a machine learning model on this data and couldn't use NA values in the dataset.

. . .

Getting Data from Multiple Stocks

The previous functions are all useful in retrieving fundamental data from one given stock ticker but what if I wanted to retrieve data from more than just one stock? To do so, I created the following function which allows the retrieval of fundamental data from multiple stocks:

This is a relatively simple function that utilizes the functions above. The first thing I

needed to do was cross-reference the given tickers with the available tickers from the API. This is done in case a given ticker doesn't exist or was improperly given. Afterwards, if the given tickers were valid, then it will retrieve the fundamental data from all the given stocks and combine them into a bigger DF containing the fundamental data from multiple stocks.

. . .

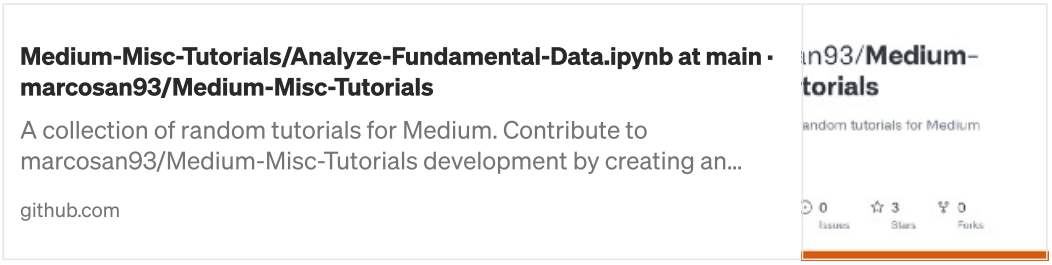
Closing Thoughts

By using Python and this Financial Data API, I was able to easily retrieve the fundamental data of nearly any stock. Besides coding the formatting process, the overall task was fairly simple.

With the fundamental data from multiple stocks ready to go, I can now apply this dataset to any future data science project. A classification or regression ML model could be used on this dataset. Or a simple analysis using Python's numerous visualization libraries. There are a never-ending assortment of projects that can use a dataset such as this.

. . .

Github



Thanks to Ben Huberman.

Finance | Data Science | Python | Machine Learning | Stock Market