Final_Assignment_Library

July 21, 2021

Extracting Stock Data Using a Python Library

A company's stock share is a piece of the company more precisely:

A stock (also known as equity) is a security that represents the ownership of a fraction of a corporation. This entitles the owner of the stock to a proportion of the corporation's assets and profits equal to how much stock they own. Units of stock are called "shares." [1]

An investor can buy a stock and sell it later. If the stock price increases, the investor profits, If it decreases,the investor with incur a loss. Determining the stock price is complex; it depends on the number of outstanding shares, the size of the company's future profits, and much more. People trade stocks throughout the day the stock ticker is a report of the price of a certain stock, updated continuously throughout the trading session by the various stock market exchanges.

You are a data scientist working for a hedge fund; it's your job to determine any suspicious stock activity. In this lab you will extract stock data using a Python library. We will use the yfinance library, it allows us to extract data for stocks returning data in a pandas dataframe. You will use the lab to extract.

Table of Contents

Estimated Time Needed: 30 min

[1]:
```
!pip install yfinance
#!pip install pandas
```

```
Collecting yfinance
  Downloading https://files.pythonhosted.org/packages/79/bd/d64719da8f5367f4d8b1
6e83507fa1d90942f433f748a4cf3ed7aa515d14/yfinance-0.1.63.tar.gz
Requirement already satisfied: pandas>=0.24 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from yfinance)
(1.1.5)
Requirement already satisfied: numpy>=1.15 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from yfinance)
(1.19.5)
```

```
Requirement already satisfied: requests>=2.20 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from yfinance)
(2.25.1)
Collecting multitasking>=0.0.7 (from yfinance)
  Downloading https://files.pythonhosted.org/packages/69/e7/e9f1661c28f7b87abfa0
8cb0e8f51dad2240a9f4f741f02ea839835e6d18/multitasking-0.0.9.tar.gz
Requirement already satisfied: lxml>=4.5.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from yfinance)
(4.6.3)
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
pandas>=0.24->yfinance) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
pandas>=0.24->yfinance) (2021.1)
Requirement already satisfied: idna<3,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.20->yfinance) (2.10)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.20->yfinance) (1.26.6)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.20->yfinance) (2021.5.30)
Requirement already satisfied: chardet<5,>=3.0.2 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.20->yfinance) (4.0.0)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from python-
dateutil>=2.7.3->pandas>=0.24->yfinance) (1.15.0)
Building wheels for collected packages: yfinance, multitasking
  Building wheel for yfinance (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/fe/a0/79/b73d4a0
c535b421b88fc7b393936b371fabbfeaf979eca4050
  Building wheel for multitasking (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/37/fa/73/d492849
e319038eb4d986f5152e4b19ffb1bc0639da84d2677
Successfully built yfinance multitasking
Installing collected packages: multitasking, yfinance
Successfully installed multitasking-0.0.9 yfinance-0.1.63
```

[2]:
```
import yfinance as yf
import pandas as pd
```

0.1 Using the yfinance Library to Extract Stock Data

Using the `Ticker` module we can create an object that will allow us to access functions to extract data. To do this we need to provide the ticker symbol for the stock, here the company is Apple and the ticker symbol is AAPL.

```
[3]: apple = yf.Ticker("AAPL")
```

Now we can access functions and variables to extract the type of data we need. You can view them and what they represent here https://aroussi.com/post/python-yahoo-finance.

0.1.1 Stock Info

Using the attribute info we can extract information about the stock as a Python dictionary.

```
[4]: apple_info=apple.info
     apple_info
```

```
[4]: {'zip': '95014',
      'sector': 'Technology',
      'fullTimeEmployees': 100000,
      'longBusinessSummary': 'Apple Inc. designs, manufactures, and markets
     smartphones, personal computers, tablets, wearables, and accessories worldwide.
     It also sells various related services. The company offers iPhone, a line of
     smartphones; Mac, a line of personal computers; iPad, a line of multi-purpose
     tablets; and wearables, home, and accessories comprising AirPods, Apple TV,
     Apple Watch, Beats products, HomePod, iPod touch, and other Apple-branded and
     third-party accessories. It also provides AppleCare support services; cloud
     services store services; and operates various platforms, including the App
     Store, that allow customers to discover and download applications and digital
     content, such as books, music, video, games, and podcasts. In addition, the
     company offers various services, such as Apple Arcade, a game subscription
     service; Apple Music, which offers users a curated listening experience with on-
     demand radio stations; Apple News+, a subscription news and magazine service;
     Apple TV+, which offers exclusive original content; Apple Card, a co-branded
     credit card; and Apple Pay, a cashless payment service, as well as licenses its
     intellectual property. The company serves consumers, and small and mid-sized
     businesses; and the education, enterprise, and government markets. It sells and
     delivers third-party applications for its products through the App Store. The
     company also sells its products through its retail and online stores, and direct
     sales force; and third-party cellular network carriers, wholesalers, retailers,
     and resellers. Apple Inc. was founded in 1977 and is headquartered in Cupertino,
     California.',
      'city': 'Cupertino',
      'phone': '408-996-1010',
      'state': 'CA',
      'country': 'United States',
      'companyOfficers': [],
      'website': 'http://www.apple.com',
      'maxAge': 1,
      'address1': 'One Apple Park Way',
      'industry': 'Consumer Electronics',
'ebitdaMargins': 0.30675,
'profitMargins': 0.23451,
'grossMargins': 0.39881,
'operatingCashflow': 99590995968,
'revenueGrowth': 0.536,
'operatingMargins': 0.27321,
'ebitda': 99820003328,
'targetLowPrice': 125,
'recommendationKey': 'buy',
'grossProfits': 104956000000,
'freeCashflow': 80121004032,
'targetMedianPrice': 160,
'currentPrice': 146.15,
'earningsGrowth': 1.188,
'currentRatio': 1.142,
'returnOnAssets': 0.169,
'numberOfAnalystOpinions': 39,
'targetMeanPrice': 159.34,
'debtToEquity': 194.78,
'returnOnEquity': 1.034,
'targetHighPrice': 185,
'totalCash': 69833998336,
'totalDebt': 134744997888,
'totalRevenue': 325405999104,
'totalCashPerShare': 4.185,
'financialCurrency': 'USD',
'revenuePerShare': 19.143,
'quickRatio': 0.967,
'recommendationMean': 2,
'exchange': 'NMS',
'shortName': 'Apple Inc.',
'longName': 'Apple Inc.',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'AAPL',
'messageBoardId': 'finmb_24937',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 7.707,
'beta3Year': None,
'enterpriseToEbitda': 25.123,
'52WeekChange': 0.46444273,
'morningStarRiskRating': None,
'forwardEps': 5.34,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 16687599616,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 4.146,
'sharesShort': 90213531,
'sharesPercentSharesOut': 0.0054,
'fundFamily': None,
'lastFiscalYearEnd': 1601078400,
'heldPercentInstitutions': 0.58561003,
'netIncomeToCommon': 76311003136,
'trailingEps': 4.449,
'lastDividendValue': 0.22,
'SandP52WeekChange': 0.29989743,
'priceToBook': 35.250843,
'heldPercentInsiders': 0.00066,
'nextFiscalYearEnd': 1664150400,
'yield': None,
'mostRecentQuarter': 1616803200,
'shortRatio': 1.24,
'sharesShortPreviousMonthDate': 1622160000,
'floatShares': 16670609616,
'beta': 1.20729,
'enterpriseValue': 2507813421056,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 1598832000,
'lastSplitFactor': '4:1',
'legalType': None,
'lastDividendDate': 1620345600,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': 1.101,
'priceToSalesTrailing12Months': 7.494922,
'dateShortInterest': 1625011200,
'pegRatio': 1.53,
'ytdReturn': None,
'forwardPE': 27.368912,
'lastCapGain': None,
'shortPercentOfFloat': 0.0054,
'sharesShortPriorMonth': 123121920,
'impliedSharesOutstanding': None,
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 142.45,
'regularMarketOpen': 143.46,
'twoHundredDayAverage': 130.30486,
 'trailingAnnualDividendYield': 0.0057564056,
 'payoutRatio': 0.1834,
 'volume24Hr': None,
 'regularMarketDayHigh': 147.0997,
 'navPrice': None,
 'averageDailyVolume10Day': 104232250,
 'regularMarketPreviousClose': 142.45,
 'fiftyDayAverage': 134.93735,
 'trailingAnnualDividendRate': 0.82,
 'open': 143.46,
 'toCurrency': None,
 'averageVolume10days': 104232250,
 'expireDate': None,
 'algorithm': None,
 'dividendRate': 0.88,
 'exDividendDate': 1620345600,
 'circulatingSupply': None,
 'startDate': None,
 'regularMarketDayLow': 142.96,
 'currency': 'USD',
 'trailingPE': 32.85008,
 'regularMarketVolume': 96350036,
 'lastMarket': None,
 'maxSupply': None,
 'openInterest': None,
 'marketCap': 2438892617728,
 'volumeAllCurrencies': None,
 'strikePrice': None,
 'averageVolume': 84313011,
 'dayLow': 142.96,
 'ask': 0,
 'askSize': 900,
 'volume': 96350036,
 'fiftyTwoWeekHigh': 150,
 'fromCurrency': None,
 'fiveYearAvgDividendYield': 1.32,
 'fiftyTwoWeekLow': 89.145,
 'bid': 0,
 'tradeable': False,
 'dividendYield': 0.006,
 'bidSize': 1200,
 'dayHigh': 147.0997,
 'regularMarketPrice': 146.15,
 'logo_url': 'https://logo.clearbit.com/apple.com'}
```

We can get the ‘country’ using the key country

```
[5]: apple_info['country']
```

```
[5]: 'United States'
```

0.1.2 Extracting Share Price

A share is the single smallest part of a company's stock that you can buy, the prices of these shares fluctuate over time. Using the history() method we can get the share price of the stock over a certain period of time. Using the `period` parameter we can set how far back from the present to get data. The options for `period` are 1 day (1d), 5d, 1 month (1mo) , 3mo, 6mo, 1 year (1y), 2y, 5y, 10y, ytd, and max.

```
[6]: apple_share_price_data = apple.history(period="max")
```

The format that the data is returned in is a Pandas DataFrame. With the `Date` as the index the share `Open`, `High`, `Low`, `Close`, `Volume`, and `Stock Splits` are given for each day.

```
[7]: apple_share_price_data.head()
```

```
[7]:                 Open      High       Low     Close     Volume  Dividends  \
     Date
     1980-12-12  0.100751  0.101189  0.100751  0.100751  469033600        0.0
     1980-12-15  0.095933  0.095933  0.095495  0.095495  175884800        0.0
     1980-12-16  0.088923  0.088923  0.088485  0.088485  105728000        0.0
     1980-12-17  0.090676  0.091114  0.090676  0.090676   86441600        0.0
     1980-12-18  0.093304  0.093742  0.093304  0.093304   73449600        0.0

                 Stock Splits
     Date
     1980-12-12           0.0
     1980-12-15           0.0
     1980-12-16           0.0
     1980-12-17           0.0
     1980-12-18           0.0
```

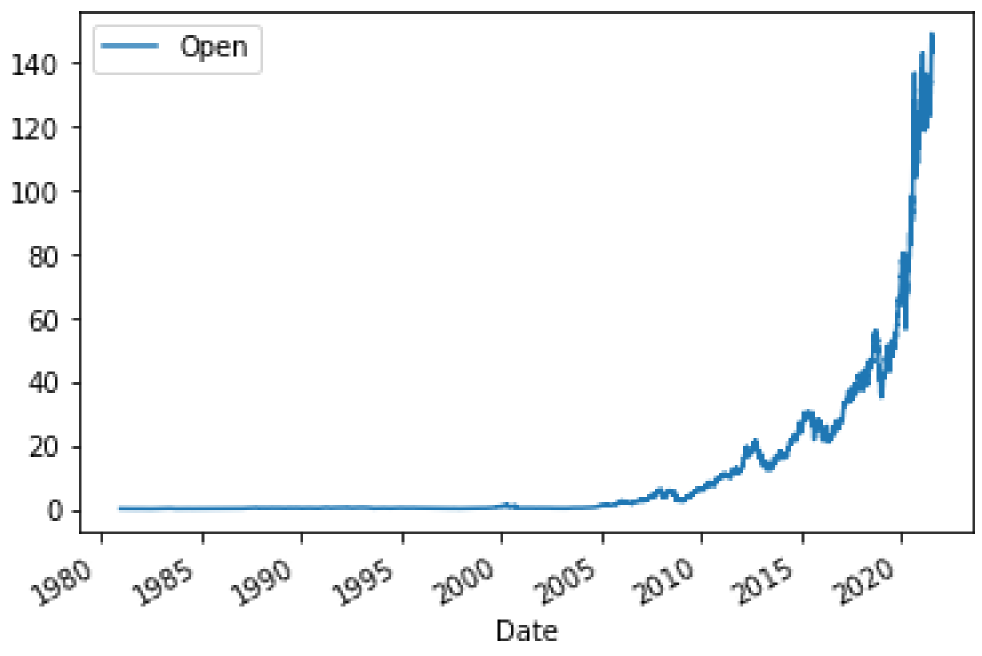
We can reset the index of the DataFrame with the `reset_index` function. We also set the `inplace` paramter to `True` so the change takes place to the DataFrame itself.

```
[8]: apple_share_price_data.reset_index(inplace=True)
```

We can plot the `Open` price against the `Date`:

```
[9]: apple_share_price_data.plot(x="Date", y="Open")
```

```
[9]: <AxesSubplot:xlabel='Date'>
```



0.1.3 Extracting Dividends

Dividends are the distribution of a companys profits to shareholders. In this case they are defined as an amount of money returned per share an investor owns. Using the variable `dividends` we can get a dataframe of the data. The period of the data is given by the period defined in the 'history' function.

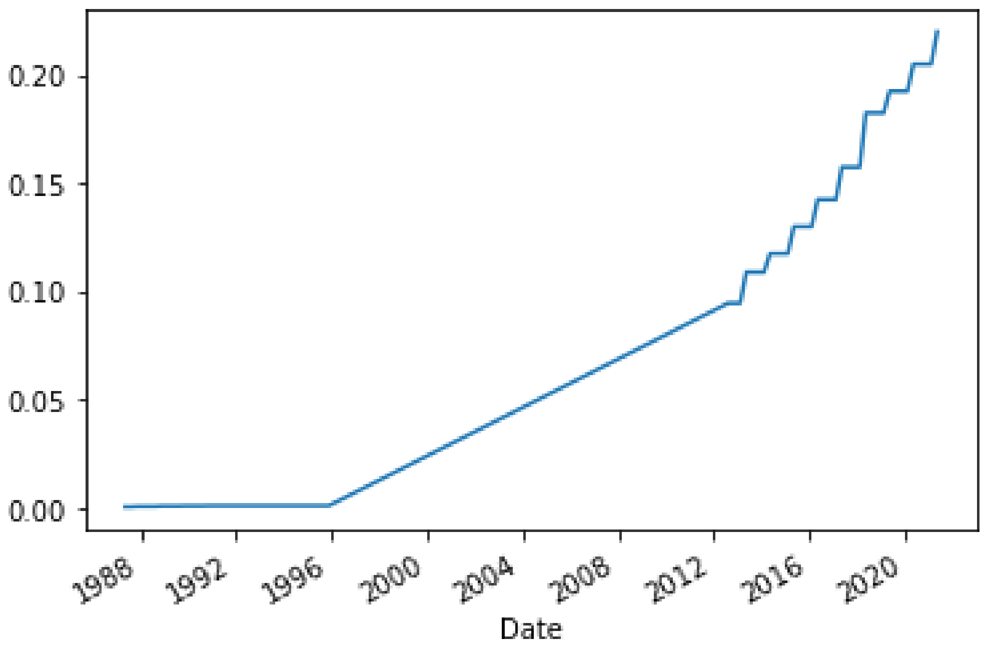
[10]:
```
apple.dividends
```

[10]:
```
Date
1987-05-11    0.000536
1987-08-10    0.000536
1987-11-17    0.000714
1988-02-12    0.000714
1988-05-16    0.000714
                ...
2020-05-08    0.205000
2020-08-07    0.205000
2020-11-06    0.205000
2021-02-05    0.205000
2021-05-07    0.220000
Name: Dividends, Length: 71, dtype: float64
```

We can plot the dividends overtime:

[11]:
```
apple.dividends.plot()
```

```
[11]: <AxesSubplot:xlabel='Date'>
```



0.2 Exercise

Now using the `Ticker` module create an object for AMD (Advanced Micro Devices) with the ticker symbol is `AMD` called; name the object amd.

```
[12]: amd = yf.Ticker('AMD')
```

Question 1 Use the key 'country' to find the country the stock belongs to, remember it as it will be a quiz question.

```
[14]: amd_info = amd.info
      amd_info['country']
```

```
[14]: 'United States'
```

Question 2 Use the key 'sector' to find the sector the stock belongs to, remember it as it will be a quiz question.

```
[15]: amd_info['sector']
```

```
[15]: 'Technology'
```

Question 3 Obtain stock data for AMD using the `history` function, set the `period` to max. Find the `Volume` traded on the first day (first row).

```
[26]: print('Date and Volume traded on the first day of AMD stock is :\n ',amd.
      ↪history(period = 'max')['Volume'].head(1))
```

```
Date and Volume traded on the first day of AMD stock is :
  Date
1980-03-17    219600
Name: Volume, dtype: int64
```

About the Authors:

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Azim Hirjani

0.3 Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-11-10	1.1	Malika Singla	Deleted the Optional part
2020-08-27	1.0	Malika Singla	Added lab to GitLab

##