Python Notebook to source daily stock market data (Open, High, Low, Close and Volume) for each stock in S&P 500 Index

Ankush Garg · Follow
Published in Ai4Markets · 6 min read · Nov 28, 2020

Written by: Ankush Garg

Logo: Ai4Markets

Introduction

In this python notebook I have pulled daily end of day matket data from yahoo finance using yfinance API

Process

Get list of Stocks

1. I created separate python script to pull list of all 500 stocks in S&P500 Index. Link to notebook: Github link, Medium link, LinkedIn Post
2. List of S&P500 stocks are saved as csv file by name: 'S&P500Tickers.csv'
3. Kindly note that the file location of this python script and S&P500 stock list should be same

Data Pull

1. This notebook pulls daily closing price of each stock starting from the day of its listing on the Exchange
2. Daily run of this file will append data to existing csv file and update it to have latest data. Thus, to avoid unnecessary data loading and processing full data is downloaded only once when you first time run the scipt. Post that every run will append data to existing file.

Data Source

1. Data is sourced from Yahoo Finance using yfinance API: https://pypi.org/project/yfinance/
2. This API is free and there is no limitation on number of requests or number of scripts for which data can be pulled daily
3. Most of the other API offering FREE solutions for market data pull has limitation either on number of requests or number for scripts for which data can be pulled daily
4. API provides option to pull adjusted proces (i.e. adjustment post devidend, splits, etc) or non-adjusted prices. We will pull adjusted prices.

Please read API docs for more details.

Step 1: Import the required libraries

We will be using yfinance API to pull data from Yahoo Finance. To install and get API details visit: yfinance

```
import yfinance as yf
import pandas as pd
import csv
import datetime as dt
from datetime import datetime # To get the current date and time
from datetime import date, timedelta
import csv
import time
import os # To check if the file exists
```

Step 2: Load tickers for S&P 500 stocks from csv file

1. I have created seperate python script (S&P500Tickers.ipynb) to get S&P 500 stocks and store that in file name is 'S&P500Tickers.csv'
2. I created seperate python script to pull list of all 500 stocks in S&P500 Index. Link to notebook: Github link, Medium link, LinkedIn Post
3. Ensure that file 'S&P500Tickers.csv' is in same folder as this python notebook. You can download file 'S&P500Tickers.csv' from gibhub as well link

```
ticker_data = pd.read_csv("S&P500Tickers.csv") # Open csv file and
read data in pandas dataframe
tickers_list = ticker_data['Symbol'].tolist() # Get all the tickers
in a list
```

Step 3: Create a function that pull data for yfinance api for each ticker

To Pull daily Open High Low Close (OHLC) and Volume of a each stock, in the S&P500 list, on a given date, script needs start and end date else data for max period will be pulled. Max period for a stock is date from which it was listed to current date Start date and end date has to be in format YYYY-MM-DD

```
def get_OHLC_data(tickers, start_date = None, end_date = None):
 ‘’’
 This function gets daily OHLC data from Yahoo Finance API for the
provided number of days.
 In case days is not provided then data is pulled for maximum number
of days

 Input Parameters:
 tickers: List of tickers for which data needs to be extracted
 Start Date: Date from which data needs to be pulled
 End Date: Date until which data needs to be pulled
 If start and end date is null then data for maximum number of days
is pulled

 Returns: Dataframe of the extracted data
 ‘’’
 final_OHLC_df = pd.DataFrame() # Declare the final empty dataframe
 for ticker in tickers: # For each stock symbol in the list of
symbols
 OHLC_data = pd.DataFrame() # Declare intermediate data frame

 yf_ticker_obj = yf.Ticker(ticker) # Initiate object to get the data
from API

 # If start date and end date is provided then pull data for those
days
 if(start_date != None and end_date != None):
 OHLC_data = yf_ticker_obj.history(start = start_date, end =
end_date, interval = “1d”, auto_adjust = True)

 else: # Pull data for all the available days
 OHLC_data = yf_ticker_obj.history(period=”max”, interval = “1d”,
auto_adjust = True)
 # Note: In the above period = ‘max’ as we are pulling data for
maximum number of days.
 # interval = ‘1d’ as per pulling daily data for Open, High, Low,
```

```
Close and Volume
 # auto_adjust = 'True' as we are adjusting data for Dividends and
Splits

 OHLC_data.insert(0, 'Symbol', ticker) # Adding this data to
dataframe
 # Delete split and dividend columns as this is not required.

 OHLC_data = OHLC_data.drop(['Dividends','Stock Splits'], axis=1,
errors='ignore')

 # Apppending this data to final dataframe
 final_OHLC_df = final_OHLC_df.append(OHLC_data)
 time.sleep(.5)

 final_OHLC_df.reset_index(inplace=True) # Re-setting the index

 # Setting index to symbol and date
 final_OHLC_df.set_index(["Date", "Symbol"], inplace=True)
 return final_OHLC_df
```

Step 4: Get the last date and dataframe from the existing OHLC csv file

1. This function will open existing csv file. It will pull the date till which OHLC is already pulled for all the stocks and will also load data in the pandas dataframe.
2. This function returns last date for which data is present in the file and pandas dataframe
3. The reson why this function is created is to identify the date till which data is already loaded. So that we pull data only for the days it is not present in the file.

def get_last_date_data(file):
"”

This function open existing OHLC file, put that in pandas dataframe and extract the last date from the file

Input Parameters: Name of the existing csv file that needs to be read in dataframe

Returns: Last date and pandas dataframe containing the data in the file

”


```
# Convert Date column to datetime
 old_data_df['Date']= pd.to_datetime(old_data_df['Date'])

 # Get the maximum date which is the last date for which data is
present
 previous_date = max(old_data_df['Date'])

 # Converting datetime to only date format as for daily OHLC data we
don't need time
 old_data_df['Date'] = old_data_df['Date'].dt.date

 # Set dataframe index to Date and Symbol
 old_data_df.set_index(["Date", "Symbol"], inplace=True)
 return previous_date, old_data_df
```

The main function to run all the sub functions

1. This function will check if the OHLC csv file existing the root folder from where this script is ran
2. If file is present then the latest date for which data is present will be pulled
3. If file is not present script will be considered as first run and entrie data will be pulled by API
4. This is done to check if we are running this script for the first time or we have ran it before
5. In case script is ran for the first time then entire data will be pulled

6. In case script has ran before then data from the last business date in the file to current business date will be pulled

7. Final data will be written to csv file

```
if __name__ == “__main__”:

 # tickers_list = [‘AAPL’, ‘MSFT’] # Created for testing. Test the
script on 2 symbols instead of 500
 # Get the next date in YYYY-MM-DD format.
 # Yfinance API gives data from last day if today’s date is entered
as start date
 today = pd.Timestamp.now().normalize() # Today’s date
 next_day = today + timedelta(days=1) # Adding 1 to get next date

 # Name of the output file
 OHLC_data_file = “OHLC_yfinance_data.csv”

 # Check if the OHLC csv file existing the root folder from where
this script is ran
 if os.path.isfile(OHLC_data_file):

 # Get the latest date in the datafile and load data in pandas
dataframe
 previous_day, old_data_df = get_last_date_data(OHLC_data_file)

 # Get the data from latest date in the old file to todays date
 latest_data = get_OHLC_data(tickers_list, previous_day, next_day)

 # Append new data with old data
 final_df = old_data_df.append(latest_data)

 # Reset index and remove duplicates
 final_df.reset_index(inplace=True)
 # Drop duplicates that has same symbol and date
 final_df = final_df.drop_duplicates(subset=[‘Date’, ‘Symbol’],
keep=’first’)

 # Set index to symbol and date
 final_df.set_index([“Date”, “Symbol”], inplace=True)

 else: # This is done in case we are running script for the first
time or data file does not exist
 latest_data = get_OHLC_data(tickers_list) # Get the data for max
available time
 final_df = latest_data
```

```
 if (‘Adj Close’ in final_df.columns): # Drop adjusted close column
if exists
 final_df.drop(‘Adj Close’, axis=1, inplace=True)

 # Writing the data to the output csv file
 final_df.to_csv(OHLC_data_file, mode=’w’, index=True) #index is True
as we want it to be written in file
```

Full Code

Github repository link: https://github.com/gargankush/marketdata_yfinance

CONGRATULATIONS!!! You have successfully pulled data daily Open, High, Close, Low and Volume of all the stocks in S&P 500 list

AI for Financial Markets. Watch out for more articles from me on stock market data collection and machine learning for stock predictions.

Ankush Garg

Clap if you like!!!

Python For Finance

Stock Market Trading

Share Market Training

Ai In Finance

MI In Finance

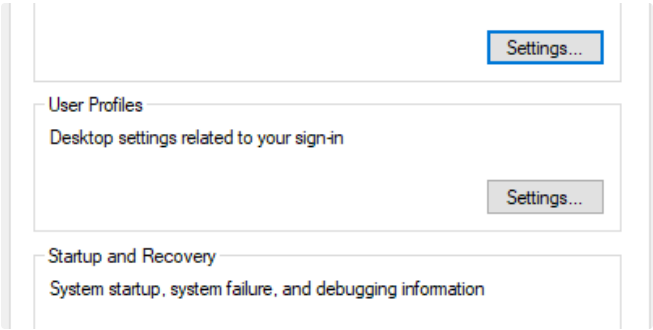
Written by Ankush Garg

Follow

18 Followers · Editor for Ai4Markets

15 years of professional work experience. I am passionate about latest technologies (Data Analytics, Machine Learning and Artificial Intelligence).

More from Ankush Garg and Ai4Markets



Ankush Garg

Storing API keys as Environmental Variable for Windows, Linux and...

In this article I will share simple steps to save your API key as environmental variable.

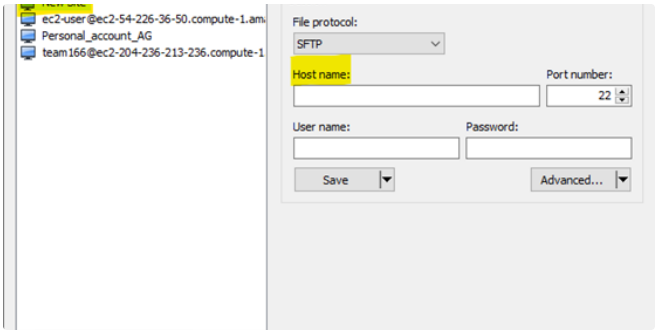
Jan 1, 2021 9

Ankush Garg in Ai4Markets

Python got Financial Markets—Simple script to pull list of S&P 50...

In this article I will cover how we can pull list of 500 S&P stocks from Wikipedia and save it i...

Nov 26, 2020 2



Ankush Garg in Ai4Markets

Connect and copy data to and from AWS EC2 instance—WinSCP

Do not have an EC2 instance? No worries! Setup new EC2 instance for FREE in 10...

Nov 25, 2020

Ankush Garg in Ai4Markets

Python Script to source stock symbols and their latest...

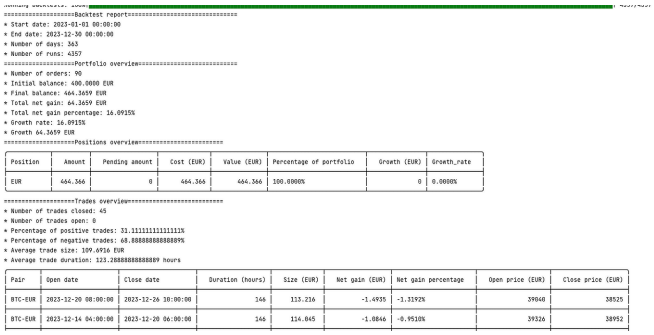
FMP API has data for 27,364 trading symbols covering various exchanges across the glob...

Jan 4, 2021 2

See all from Ankush Garg

See all from Ai4Markets

Recommended from Medium



Marc van Duyn

How to create a trading bot in 5 steps

Would you like to build your own trading bot but do not know where to start? You have...

Jan 1 46

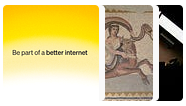
PulsePointFX

Simple Moving Average Crossover Strategy using Python 🐍

Create a Simple Moving Average Crossover Strategy using Python programming...

Jan 9 54

Lists

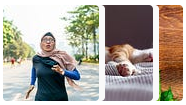


Staff Picks

663 stories · 1061 saves

Stories to Help You Level-Up at Work

19 stories · 652 saves



Self-Improvement 101

20 stories · 2102 saves

Productivity 101

20 stories · 1863 saves

Henrique Centieiro & Bee ... in Limitless Invest...

52% Returns in 30 Days: Your GPT-4o Quant Trading Bot Strategy

How to Profit in Any Market Condition with GPT-4o's Mean Regression Strategies

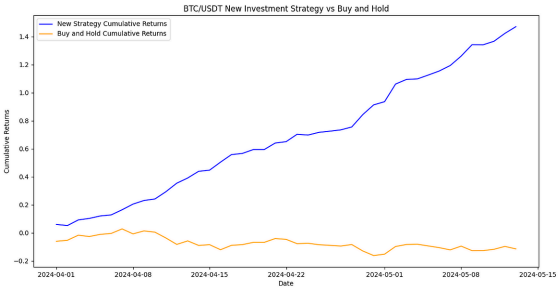
May 21 1.5K 25

Paul Lenosky

I have created an indicator that actually makes money, unlike any...

How the indicator works in a nutshell

Feb 9 2K 55



Robertorusso in BIP xTech

Stop using Moving Average to smooth your Time Series

Why Savitzky-Golay Filters often make a better job smoothing Time-Series

Apr 17 732 10

Javier Santiago Gaston de Iriarte Cabrera

Back-testing Cryptocurrencies: Astonishing Results from a Simpl...

In the rapidly fluctuating world of cryptocurrencies, back-testing plays a cruci...

May 14 112 7

See more recommendations