Final Assignment

August 6, 2022

Extracting and Visualizing Stock Data

Description

Extracting essential data from a dataset and displaying it is a necessary part of data science; therefore individuals can make correct decisions based on the data. In this assignment, you will extract some stock data, you will then display this data in a graph.

Table of Contents

```
<ul>
    <li>Define a Function that Makes a Graph</li>
    <li>Question 1: Use yfinance to Extract Stock Data</li>
    <li>Question 2: Use Webscraping to Extract Tesla Revenue Data</li>
    <li>Question 3: Use yfinance to Extract Stock Data</li>
    <li>Question 4: Use Webscraping to Extract GME Revenue Data</li>
    <li>Question 5: Plot Tesla Stock Graph</li>
    <li>Question 6: Plot GameStop Stock Graph</li>
</ul>
```

Estimated Time Needed: 30 min

[1]:
```
# These are already installed on my system.
#!pip install yfinance==0.1.67
#!pip install pandas==1.3.3
#!pip install requests==2.26.0
#!mamba install bs4==4.10.0 -y
#!pip install plotly==5.3.1
```

[2]:
```
import yfinance as yf
import pandas as pd
import requests
from bs4 import BeautifulSoup
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

0.1 Define Graphing Function

In this section, we define the function `make_graph`. You don't have to know how the function works, you should only care about the inputs. It takes a dataframe with stock data (dataframe must contain Date and Close columns), a dataframe with revenue data (dataframe must contain Date and Revenue columns), and the name of the stock.

[3]:
```
def make_graph(stock_data, revenue_data, stock):
    fig = make_subplots(rows=2, cols=1, shared_xaxes=True,␣
 ↪subplot_titles=("Historical Share Price", "Historical Revenue"),␣
 ↪vertical_spacing = .3)
    stock_data_specific = stock_data[stock_data.Date <= '2021--06-14']
    revenue_data_specific = revenue_data[revenue_data.Date <= '2021-04-30']
    fig.add_trace(go.Scatter(x=pd.to_datetime(stock_data_specific.Date,␣
 ↪infer_datetime_format=True), y=stock_data_specific.Close.astype("float"),␣
 ↪name="Share Price"), row=1, col=1)
    fig.add_trace(go.Scatter(x=pd.to_datetime(revenue_data_specific.Date,␣
 ↪infer_datetime_format=True), y=revenue_data_specific.Revenue.
 ↪astype("float"), name="Revenue"), row=2, col=1)
    fig.update_xaxes(title_text="Date", row=1, col=1)
    fig.update_xaxes(title_text="Date", row=2, col=1)
    fig.update_yaxes(title_text="Price ($US)", row=1, col=1)
    fig.update_yaxes(title_text="Revenue ($US Millions)", row=2, col=1)
    fig.update_layout(showlegend=False,
    height=900,
    title=stock,
    xaxis_rangeslider_visible=True)
    fig.show()
```

0.2 Question 1: Use yfinance to Extract Stock Data

Using the `Ticker` function enter the ticker symbol of the stock we want to extract data on to create a ticker object. The stock is Tesla and its ticker symbol is `TSLA`.

[4]:
```
tesla = yf.Ticker("TSLA")
```

[5]:
```
tesla.info
```

[5]:
```
{'zip': '78725',
 'sector': 'Consumer Cyclical',
 'fullTimeEmployees': 99290,
 'longBusinessSummary': 'Tesla, Inc. designs, develops, manufactures, leases,
and sells electric vehicles, and energy generation and storage systems in the
United States, China, and internationally. The company operates in two segments,
Automotive, and Energy Generation and Storage. The Automotive segment offers
electric vehicles, as well as sells automotive regulatory credits. It provides
sedans and sport utility vehicles through direct and used vehicle sales, a
network of Tesla Superchargers, and in-app upgrades; and purchase financing and
leasing services. This segment is also involved in the provision of non-warranty
after-sales vehicle services, sale of used vehicles, retail merchandise, and
vehicle insurance, as well as sale of products to third party customers;
```

```
services for electric vehicles through its company-owned service locations, and
Tesla mobile service technicians; and vehicle limited warranties and extended
service plans. The Energy Generation and Storage segment engages in the design,
manufacture, installation, sale, and leasing of solar energy generation and
energy storage products, and related services to residential, commercial, and
industrial customers and utilities through its website, stores, and galleries,
as well as through a network of channel partners. This segment also offers
service and repairs to its energy product customers, including under warranty;
and various financing options to its solar customers. The company was formerly
known as Tesla Motors, Inc. and changed its name to Tesla, Inc. in February
2017. Tesla, Inc. was incorporated in 2003 and is headquartered in Austin,
Texas.',
 'city': 'Austin',
 'phone': '(512) 516-8177',
 'state': 'TX',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.tesla.com',
 'maxAge': 1,
 'address1': '13101 Tesla Road',
 'industry': 'Auto Manufacturers',
 'ebitdaMargins': 0.20889,
 'profitMargins': 0.14168,
 'grossMargins': 0.27099,
 'operatingCashflow': 14078000128,
 'revenueGrowth': 0.416,
 'operatingMargins': 0.16139,
 'ebitda': 14030000128,
 'targetLowPrice': 73,
 'recommendationKey': 'buy',
 'grossProfits': 13606000000,
 'freeCashflow': 5962749952,
 'targetMedianPrice': 950,
 'currentPrice': 864.51,
 'earningsGrowth': 0.907,
 'currentRatio': 1.431,
 'returnOnAssets': 0.10957,
 'numberOfAnalystOpinions': 39,
 'targetMeanPrice': 879.33,
 'debtToEquity': 17.699,
 'returnOnEquity': 0.2989,
 'targetHighPrice': 1300,
 'totalCash': 18915000320,
 'totalDebt': 6664999936,
 'totalRevenue': 67165999104,
 'totalCashPerShare': 18.109,
 'financialCurrency': 'USD',
'revenuePerShare': 65.785,
'quickRatio': 0.968,
'recommendationMean': 2.4,
'exchange': 'NMS',
'shortName': 'Tesla, Inc.',
'longName': 'Tesla, Inc.',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'TSLA',
'messageBoardId': 'finmb_27444752',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 13.281,
'beta3Year': None,
'enterpriseToEbitda': 63.578,
'52WeekChange': 0.21120548,
'morningStarRiskRating': None,
'forwardEps': 15.93,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 1044489984,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 34.943,
'sharesShort': 23491892,
'sharesPercentSharesOut': 0.0226,
'fundFamily': None,
'lastFiscalYearEnd': 1640908800,
'heldPercentInstitutions': 0.4284,
'netIncomeToCommon': 9521000448,
'trailingEps': 7.76,
'lastDividendValue': None,
'SandP52WeekChange': -0.06478733,
'priceToBook': 24.740578,
'heldPercentInsiders': 0.17188999,
'nextFiscalYearEnd': 1703980800,
'yield': None,
'mostRecentQuarter': 1656547200,
'shortRatio': 0.76,
'sharesShortPreviousMonthDate': 1655251200,
'floatShares': 865203304,
'beta': 2.176087,
'enterpriseValue': 892004073472,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 1598832000,
'lastSplitFactor': '5:1',
'legalType': None,
'lastDividendDate': None,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': 0.978,
'priceToSalesTrailing12Months': 13.443886,
'dateShortInterest': 1657843200,
'pegRatio': None,
'ytdReturn': None,
'forwardPE': 54.269302,
'lastCapGain': None,
'shortPercentOfFloat': 0.0276,
'sharesShortPriorMonth': 26882235,
'impliedSharesOutstanding': 0,
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 925.9,
'regularMarketOpen': 908.01,
'twoHundredDayAverage': 910.92194,
'trailingAnnualDividendYield': 0,
'payoutRatio': 0,
'volume24Hr': None,
'regularMarketDayHigh': 913.8199,
'navPrice': None,
'averageDailyVolume10Day': 29232460,
'regularMarketPreviousClose': 925.9,
'fiftyDayAverage': 744.3584,
'trailingAnnualDividendRate': 0,
'open': 908.01,
'toCurrency': None,
'averageVolume10days': 29232460,
'expireDate': None,
'algorithm': None,
'dividendRate': None,
'exDividendDate': None,
'circulatingSupply': None,
'startDate': None,
'regularMarketDayLow': 856.634,
'currency': 'USD',
'trailingPE': 111.40593,
'regularMarketVolume': 37724299,
'lastMarket': None,
'maxSupply': None,
'openInterest': None,
'marketCap': 902972047360,
 'volumeAllCurrencies': None,
 'strikePrice': None,
 'averageVolume': 31159732,
 'dayLow': 856.634,
 'ask': 862.2,
 'askSize': 1000,
 'volume': 37724299,
 'fiftyTwoWeekHigh': 1243.49,
 'fromCurrency': None,
 'fiveYearAvgDividendYield': None,
 'fiftyTwoWeekLow': 620.57,
 'bid': 862.3,
 'tradeable': False,
 'dividendYield': None,
 'bidSize': 2200,
 'dayHigh': 913.8199,
 'coinMarketCapLink': None,
 'regularMarketPrice': 864.51,
 'preMarketPrice': None,
 'logo_url': 'https://logo.clearbit.com/tesla.com',
 'trailingPegRatio': 2.1897}
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `tesla_data`. Set the `period` parameter to `max` so we get information for the maximum amount of time.

[6]:
```
tesla_data = tesla.history(period="max")
```

[7]:
```
tesla_data
```

[7]:
```
                  Open        High         Low       Close    Volume  \
Date
2010-06-29    3.800000    5.000000    3.508000    4.778000  93831500
2010-06-30    5.158000    6.084000    4.660000    4.766000  85935500
2010-07-01    5.000000    5.184000    4.054000    4.392000  41094000
2010-07-02    4.600000    4.620000    3.742000    3.840000  25699000
2010-07-06    4.000000    4.000000    3.166000    3.222000  34334500
…                  …         …         …          …        …
2022-08-01  903.830017  935.630005  885.000000  891.830017  39014300
2022-08-02  882.010010  923.500000  878.000000  901.760010  31859200
2022-08-03  915.000000  928.650024  903.450012  922.190002  26697000
2022-08-04  933.000000  940.820007  915.000000  925.900024  24085400
2022-08-05  908.010010  913.820007  856.630005  864.510010  37655300

            Dividends  Stock Splits
Date
2010-06-29          0           0.0
```

```
2010-06-30           0            0.0
2010-07-01           0            0.0
2010-07-02           0            0.0
2010-07-06           0            0.0
…                  …            …
2022-08-01           0            0.0
2022-08-02           0            0.0
2022-08-03           0            0.0
2022-08-04           0            0.0
2022-08-05           0            0.0

[3048 rows x 7 columns]
```

Reset the index using the `reset_index(inplace=True)` function on the tesla_data DataFrame and display the first five rows of the `tesla_data` dataframe using the `head` function. Take a screenshot of the results and code from the beginning of Question 1 to the results below.

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

[9]:
```
tesla_data.head()
```

[9]:
```
        Date   Open   High    Low  Close    Volume  Dividends  Stock Splits
0 2010-06-29  3.800  5.000  3.508  4.778  93831500          0           0.0
1 2010-06-30  5.158  6.084  4.660  4.766  85935500          0           0.0
2 2010-07-01  5.000  5.184  4.054  4.392  41094000          0           0.0
3 2010-07-02  4.600  4.620  3.742  3.840  25699000          0           0.0
4 2010-07-06  4.000  4.000  3.166  3.222  34334500          0           0.0
```

0.3 Question 2: Use Webscraping to Extract Tesla Revenue Data

Use the `requests` library to download the webpage https://www.macrotrends.net/stocks/charts/TSLA/tesla/reve
Save the text of the response as a variable named `html_data`.

[10]:
```
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue?
 ↪utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_

html_data = requests.get(url).text
```

[11]:
```
html_data[0:500]
```

[11]:
```
'\r\n<!DOCTYPE html>\r\n<!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8
lt-ie7"> <![endif]-->\r\n<!--[if IE 7]>         <html class="no-js lt-ie9 lt-
ie8"> <![endif]-->\r\n<!--[if IE 8]>         <html class="no-js lt-ie9">
<![endif]-->\r\n<!--[if gt IE 8]><!--> <html class="no-js"> <!--<![endif]-->\r\n
<head>\r\n        <meta charset="utf-8">\r\n        <meta http-equiv="X-UA-
Compatible" content="IE=edge,chrome=1">\r\n\t\t<link rel="canonical"
href="https://www.macrotrends.net/stocks/charts/TSLA/tesla/reve'
```

Parse the html data using `beautiful_soup`.

```
[12]: soup = BeautifulSoup(html_data, 'html5lib')
```

Using `BeautifulSoup` or the `read_html` function extract the table with `Tesla Quarterly Revenue` and store it into a dataframe named `tesla_revenue`. The dataframe should have columns `Date` and `Revenue`.

Click here if you need help locating the table

```
Below is the code to isolate the table, you will now need to loop through the rows and columns

soup.find_all("tbody")[1]

If you want to use the read_html function the table is located at index 1
```

```
[13]: read_html_tesla_revenue_data = pd.read_html(str(soup))
```

```
[14]: tesla_revenue = read_html_tesla_revenue_data[1]
      tesla_revenue.columns = ["Date", "Revenue"]
```

```
[15]: tesla_revenue.head()
```

```
[15]:          Date  Revenue
      0  2022-06-30  $16,934
      1  2022-03-31  $18,756
      2  2021-12-31  $17,719
      3  2021-09-30  $13,757
      4  2021-06-30  $11,958
```

Execute the following line to remove the comma and dollar sign from the `Revenue` column.

```
[16]: tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

```
[17]: tesla_revenue.head()
```

```
[17]:          Date Revenue
      0  2022-06-30   16934
      1  2022-03-31   18756
      2  2021-12-31   17719
      3  2021-09-30   13757
      4  2021-06-30   11958
```

Execute the following lines to remove an null or empty strings in the Revenue column.

```
[18]: tesla_revenue.dropna(inplace=True)

      tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
```

Display the last 5 row of the `tesla_revenue` dataframe using the `tail` function. Take a screenshot of the results.

[19]:
```
tesla_revenue.tail()
```

[19]:
```
          Date Revenue
47  2010-09-30      31
48  2010-06-30      28
49  2010-03-31      21
51  2009-09-30      46
52  2009-06-30      27
```

0.4 Question 3: Use yfinance to Extract Stock Data

Using the `Ticker` function enter the ticker symbol of the stock we want to extract data on to create a ticker object. The stock is GameStop and its ticker symbol is `GME`.

[20]:
```
gme = yf.Ticker("GME")
```

[21]:
```
gme_info = gme.info
gme_info
```

[21]:
```
{'zip': '76051',
 'sector': 'Consumer Cyclical',
 'fullTimeEmployees': 12000,
 'longBusinessSummary': 'GameStop Corp., a specialty retailer, provides games
and entertainment products through its e-commerce properties and various stores
in the United States, Canada, Australia, and Europe. The company sells new and
pre-owned gaming platforms; accessories, such as controllers, gaming headsets,
virtual reality products, and memory cards; new and pre-owned gaming software;
and in-game digital currency, digital downloadable content, and full-game
downloads. It also sells collectibles comprising licensed merchandise primarily
related to the gaming, television, and movie industries, as well as pop culture
themes. As of January 29, 2022, the company operated 4,573 stores and ecommerce
sites under the GameStop, EB Games, and Micromania brands; and 50 pop culture
themed stores that sell collectibles, apparel, gadgets, electronics, toys, and
other retail products under the Zing Pop Culture brand, as well as offers Game
Informer, a print and digital video game publication featuring reviews of new
releases, previews of the big titles on the horizon, and coverage of the latest
developments in the gaming industry. The company was formerly known as GSC
Holdings Corp. GameStop Corp. was founded in 1996 and is headquartered in
Grapevine, Texas.',
 'city': 'Grapevine',
 'phone': '817 424 2000',
 'state': 'TX',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.gamestop.com',
 'maxAge': 1,
 'address1': '625 Westport Parkway',
 'industry': 'Specialty Retail',
 'ebitdaMargins': -0.05618,
 'profitMargins': -0.07729,
 'grossMargins': 0.21534,
 'operatingCashflow': -719400000,
 'revenueGrowth': 0.08,
 'operatingMargins': -0.06855,
 'ebitda': -343400000,
 'targetLowPrice': 5.75,
 'recommendationKey': 'underperform',
 'grossProfits': 1347800000,
 'freeCashflow': -572687488,
 'targetMedianPrice': 7.5,
 'currentPrice': 40.02,
 'earningsGrowth': None,
 'currentRatio': 2.067,
 'returnOnAssets': -0.092080005,
 'numberOfAnalystOpinions': 3,
 'targetMeanPrice': 13.58,
 'debtToEquity': 42.531,
 'returnOnEquity': -0.40546,
 'targetHighPrice': 27.5,
 'totalCash': 1035000000,
 'totalDebt': 617000000,
 'totalRevenue': 6112300032,
 'totalCashPerShare': 3.416,
 'financialCurrency': 'USD',
 'revenuePerShare': 20.354,
 'quickRatio': 1.01,
 'recommendationMean': 4,
 'exchange': 'NYQ',
 'shortName': 'GameStop Corporation',
 'longName': 'GameStop Corp.',
 'exchangeTimezoneName': 'America/New_York',
 'exchangeTimezoneShortName': 'EDT',
 'isEsgPopulated': False,
 'gmtOffSetMilliseconds': '-14400000',
 'quoteType': 'EQUITY',
 'symbol': 'GME',
 'messageBoardId': 'finmb_1342560',
 'market': 'us_market',
 'annualHoldingsTurnover': None,
 'enterpriseToRevenue': 1.812,
 'beta3Year': None,
 'enterpriseToEbitda': -32.246,
 '52WeekChange': -0.0065164566,
 'morningStarRiskRating': None,
 'forwardEps': -1.02,
 'revenueQuarterlyGrowth': None,
 'sharesOutstanding': 304516000,
 'fundInceptionDate': None,
 'annualReportExpenseRatio': None,
 'totalAssets': None,
 'bookValue': 4.803,
 'sharesShort': 59621904,
 'sharesPercentSharesOut': 0.1958,
 'fundFamily': None,
 'lastFiscalYearEnd': 1643414400,
 'heldPercentInstitutions': 0.28042,
 'netIncomeToCommon': -472400000,
 'trailingEps': -6.292,
 'lastDividendValue': 0.095,
 'SandP52WeekChange': -0.06478733,
 'priceToBook': 8.332293,
 'heldPercentInsiders': 0.15626,
 'nextFiscalYearEnd': 1706486400,
 'yield': None,
 'mostRecentQuarter': 1651276800,
 'shortRatio': 5.17,
 'sharesShortPreviousMonthDate': 1655251200,
 'floatShares': 253523951,
 'beta': -0.777145,
 'enterpriseValue': 11073316864,
 'priceHint': 2,
 'threeYearAverageReturn': None,
 'lastSplitDate': 1658448000,
 'lastSplitFactor': '4:1',
 'legalType': None,
 'lastDividendDate': 1552521600,
 'morningStarOverallRating': None,
 'earningsQuarterlyGrowth': None,
 'priceToSalesTrailing12Months': 1.9938043,
 'dateShortInterest': 1657843200,
 'pegRatio': 0.76,
 'ytdReturn': None,
 'forwardPE': -39.235294,
 'lastCapGain': None,
 'shortPercentOfFloat': 0.2228,
 'sharesShortPriorMonth': 58045540,
 'impliedSharesOutstanding': 0,
 'category': None,
 'fiveYearAverageReturn': None,
 'previousClose': 38.36,
 'regularMarketOpen': 37.37,
 'twoHundredDayAverage': 35.03814,
 'trailingAnnualDividendYield': 0,
 'payoutRatio': 0,
 'volume24Hr': None,
 'regularMarketDayHigh': 40.4299,
 'navPrice': None,
 'averageDailyVolume10Day': 4957020,
 'regularMarketPreviousClose': 38.36,
 'fiftyDayAverage': 33.70245,
 'trailingAnnualDividendRate': 0,
 'open': 37.37,
 'toCurrency': None,
 'averageVolume10days': 4957020,
 'expireDate': None,
 'algorithm': None,
 'dividendRate': None,
 'exDividendDate': 1552521600,
 'circulatingSupply': None,
 'startDate': None,
 'regularMarketDayLow': 36.564,
 'currency': 'USD',
 'regularMarketVolume': 8124235,
 'lastMarket': None,
 'maxSupply': None,
 'openInterest': None,
 'marketCap': 12186730496,
 'volumeAllCurrencies': None,
 'strikePrice': None,
 'averageVolume': 13674754,
 'dayLow': 36.564,
 'ask': 40.4,
 'askSize': 800,
 'volume': 8124235,
 'fiftyTwoWeekHigh': 63.9225,
 'fromCurrency': None,
 'fiveYearAvgDividendYield': None,
 'fiftyTwoWeekLow': 19.395,
 'bid': 39.88,
 'tradeable': False,
 'dividendYield': None,
 'bidSize': 1000,
 'dayHigh': 40.4299,
 'coinMarketCapLink': None,
 'regularMarketPrice': 40.02,
 'preMarketPrice': None,
 'logo_url': 'https://logo.clearbit.com/gamestop.com'}
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `gme_data`. Set the `period` parameter to `max` so we get information for the maximum amount of time.

[22]:
```
gme_data = gme.history(period="max")
```

[23]:
```
gme_data.head()
```

[23]:
```
                Open      High       Low     Close    Volume  Dividends  \
Date
2002-02-13  1.620128  1.693350  1.603296  1.691666  76216000        0.0
2002-02-14  1.712707  1.716074  1.670626  1.683250  11021600        0.0
2002-02-15  1.683251  1.687459  1.658002  1.674834   8389600        0.0
2002-02-19  1.666418  1.666418  1.578047  1.607504   7410400        0.0
2002-02-20  1.615920  1.662209  1.603296  1.662209   6892800        0.0

            Stock Splits
Date
2002-02-13           0.0
2002-02-14           0.0
2002-02-15           0.0
2002-02-19           0.0
2002-02-20           0.0
```

Reset the index using the `reset_index(inplace=True)` function on the gme_data DataFrame and display the first five rows of the `gme_data` dataframe using the `head` function. Take a screenshot of the results and code from the beginning of Question 3 to the results below.

[24]:
```
gme_data.reset_index(inplace=True)
```

[25]:
```
gme_data.head()
```

[25]:
```
        Date      Open      High       Low     Close    Volume  Dividends  \
0 2002-02-13  1.620128  1.693350  1.603296  1.691666  76216000        0.0
1 2002-02-14  1.712707  1.716074  1.670626  1.683250  11021600        0.0
2 2002-02-15  1.683251  1.687459  1.658002  1.674834   8389600        0.0
3 2002-02-19  1.666418  1.666418  1.578047  1.607504   7410400        0.0
4 2002-02-20  1.615920  1.662209  1.603296  1.662209   6892800        0.0

   Stock Splits
0           0.0
1           0.0
2           0.0
3           0.0
4           0.0
```

0.5 Question 4: Use Webscraping to Extract GME Revenue Data

Use the `requests` library to download the webpage https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html. Save the text of the response as a variable named `html_data`.

[26]:
```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"

html_data = requests.get(url).text
```

[27]:
```
html_data[:500]
```

[27]:
```
'<!DOCTYPE html>\n<!-- saved from url=(0105)https://web.archive.org/web/20200814
131437/https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue -->\n<html
class=" js flexbox canvas canvastext webgl no-touch geolocation postmessage
websqldatabase indexeddb hashchange history draganddrop websockets rgba hsla
multiplebgs backgroundsize borderimage borderradius boxshadow textshadow opacity
cssanimations csscolumns cssgradients cssreflections csstransforms
csstransforms3d csstransitions fontface g'
```

Parse the html data using `beautiful_soup`.

[28]:
```
soup = BeautifulSoup(html_data, 'html5lib')
```

Using `BeautifulSoup` or the `read_html` function extract the table with `GameStop Quarterly Revenue` and store it into a dataframe named `gme_revenue`. The dataframe should have columns `Date` and `Revenue`. Make sure the comma and dollar sign is removed from the `Revenue` column using a method similar to what you did in Question 2.

Click here if you need help locating the table

```
Below is the code to isolate the table, you will now need to loop through the rows and columns

soup.find_all("tbody")[1]

If you want to use the read_html function the table is located at index 1
```

[29]:
```
read_html_gme_revenue_data = pd.read_html(str(soup))
```

[30]:
```
read_html_gme_revenue_data
```

[30]:
```
[    GameStop Annual Revenue(Millions of US $)  \
 0                                        2020
 1                                        2019
 2                                        2018
 3                                        2017
```

```
4                                       2016
5                                       2015
6                                       2014
7                                       2013
8                                       2012
9                                       2011
10                                      2010
11                                      2009
12                                      2008
13                                      2007
14                                      2006
15                                      2005

   GameStop Annual Revenue(Millions of US $).1
0                                      $6,466
1                                      $8,285
2                                      $8,547
3                                      $7,965
4                                      $9,364
5                                      $9,296
6                                      $9,040
7                                      $8,887
8                                      $9,551
9                                      $9,474
10                                     $9,078
11                                     $8,806
12                                     $7,094
13                                     $5,319
14                                     $3,092
15                                     $1,843  ,
   GameStop Quarterly Revenue(Millions of US $)  \
0                                    2020-04-30
1                                    2020-01-31
2                                    2019-10-31
3                                    2019-07-31
4                                    2019-04-30
..                                          …
57                                   2006-01-31
58                                   2005-10-31
59                                   2005-07-31
60                                   2005-04-30
61                                   2005-01-31

   GameStop Quarterly Revenue(Millions of US $).1
0                                         $1,021
1                                         $2,194
2                                         $1,439
```

```
3                                        $1,286
4                                        $1,548
..                                          …
57                                       $1,667
58                                         $534
59                                         $416
60                                         $475
61                                         $709

[62 rows x 2 columns],
                                             Sector  \
0                                 Retail/Wholesale
1  GameStop Corp. is the world's largest video ga…

                                           Industry  \
0                      Retail - Consumer Electronics
1  GameStop Corp. is the world's largest video ga…

                                         Market Cap  \
0                                           $0.293B
1  GameStop Corp. is the world's largest video ga…

                                            Revenue
0                                           $6.466B
1  GameStop Corp. is the world's largest video ga…  ,
                    Stock Name        Country Market Cap  PE Ratio
0               Best Buy (BBY)  United States   $27.033B     18.16
1               Aaron's, (AAN)  United States    $3.975B     15.14
2  GOME Retail Holdings (GMELY)          China    $1.684B      0.00
3               Systemax (SYX)  United States    $0.873B     18.34
4                Conn's (CONN)  United States    $0.325B      0.00
5   Taitron Components (TAIT)  United States    $0.016B     10.50,
                     Link Preview  HTML Code (Click to Copy)
0  GameStop Revenue 2006-2020 | GME                        NaN
1                       Macrotrends                        NaN
2                            Source                        NaN,
                     Link Preview  HTML Code (Click to Copy)
0  GameStop Revenue 2006-2020 | GME                        NaN
1                       Macrotrends                        NaN
2                            Source                        NaN]
```

[31]:
```
gme_revenue = read_html_gme_revenue_data[1]
gme_revenue.columns = ["Date", "Revenue"]
```

[32]:
```
gme_revenue.head()
```

```
[32]:          Date Revenue
       0  2020-04-30  $1,021
       1  2020-01-31  $2,194
       2  2019-10-31  $1,439
       3  2019-07-31  $1,286
       4  2019-04-30  $1,548
```

Execute the following line to remove the comma and dollar sign from the `Revenue` column.

```
[33]: gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")
```

```
[34]: gme_revenue.head()
```

```
[34]:          Date Revenue
       0  2020-04-30    1021
       1  2020-01-31    2194
       2  2019-10-31    1439
       3  2019-07-31    1286
       4  2019-04-30    1548
```

Execute the following lines to remove an null or empty strings in the Revenue column.

```
[35]: gme_revenue.dropna(inplace=True)

      gme_revenue = gme_revenue[gme_revenue['Revenue'] != ""]
```

Display the last five rows of the `gme_revenue` dataframe using the `tail` function. Take a screenshot of the results.

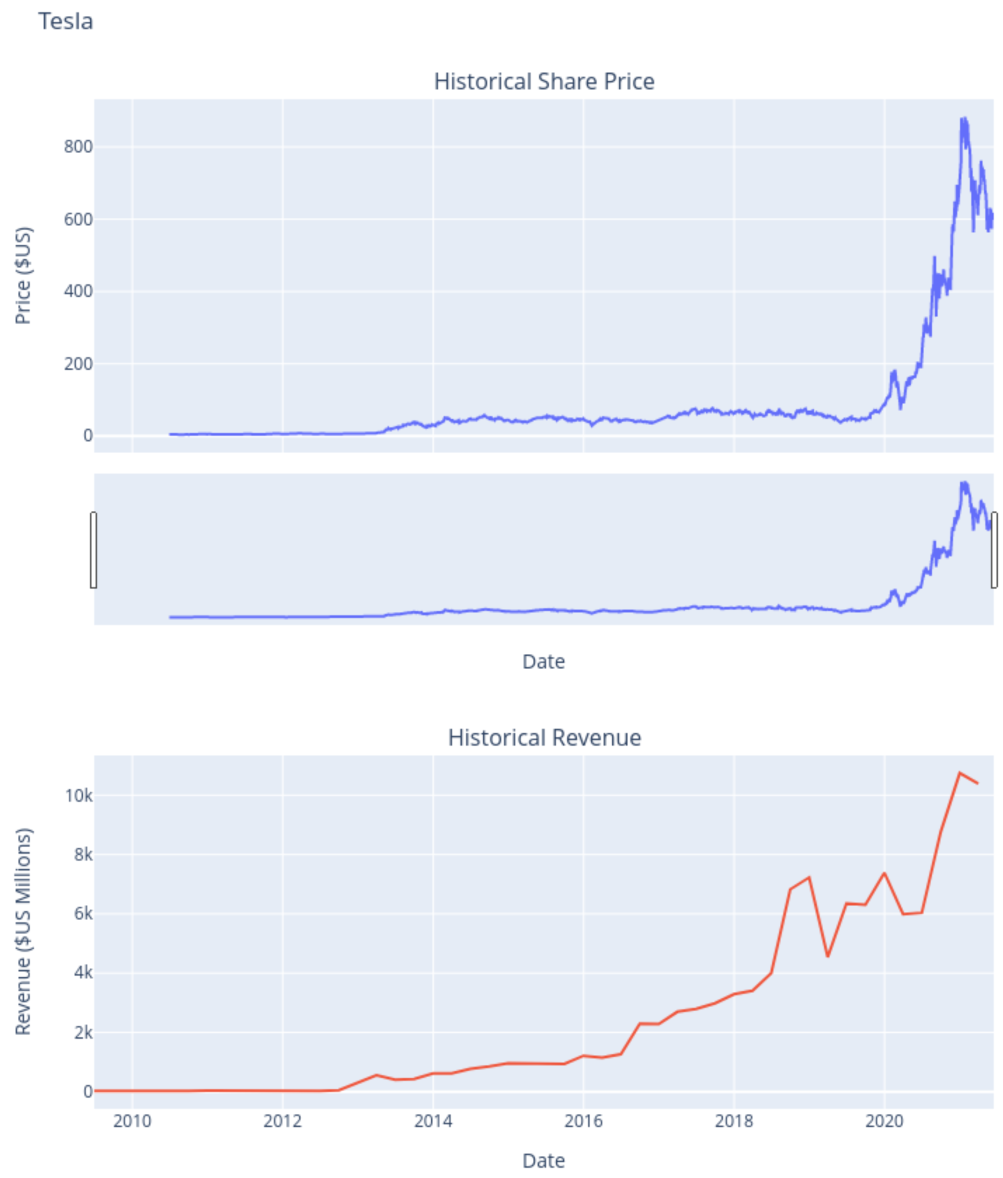
```
[36]: gme_revenue.tail()
```

```
[36]:           Date Revenue
       57  2006-01-31    1667
       58  2005-10-31     534
       59  2005-07-31     416
       60  2005-04-30     475
       61  2005-01-31     709
```

0.6 Question 5: Plot Tesla Stock Graph

Use the `make_graph` function to graph the Tesla Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(tesla_data, tesla_revenue, 'Tesla')`. Note the graph will only show data upto June 2021.

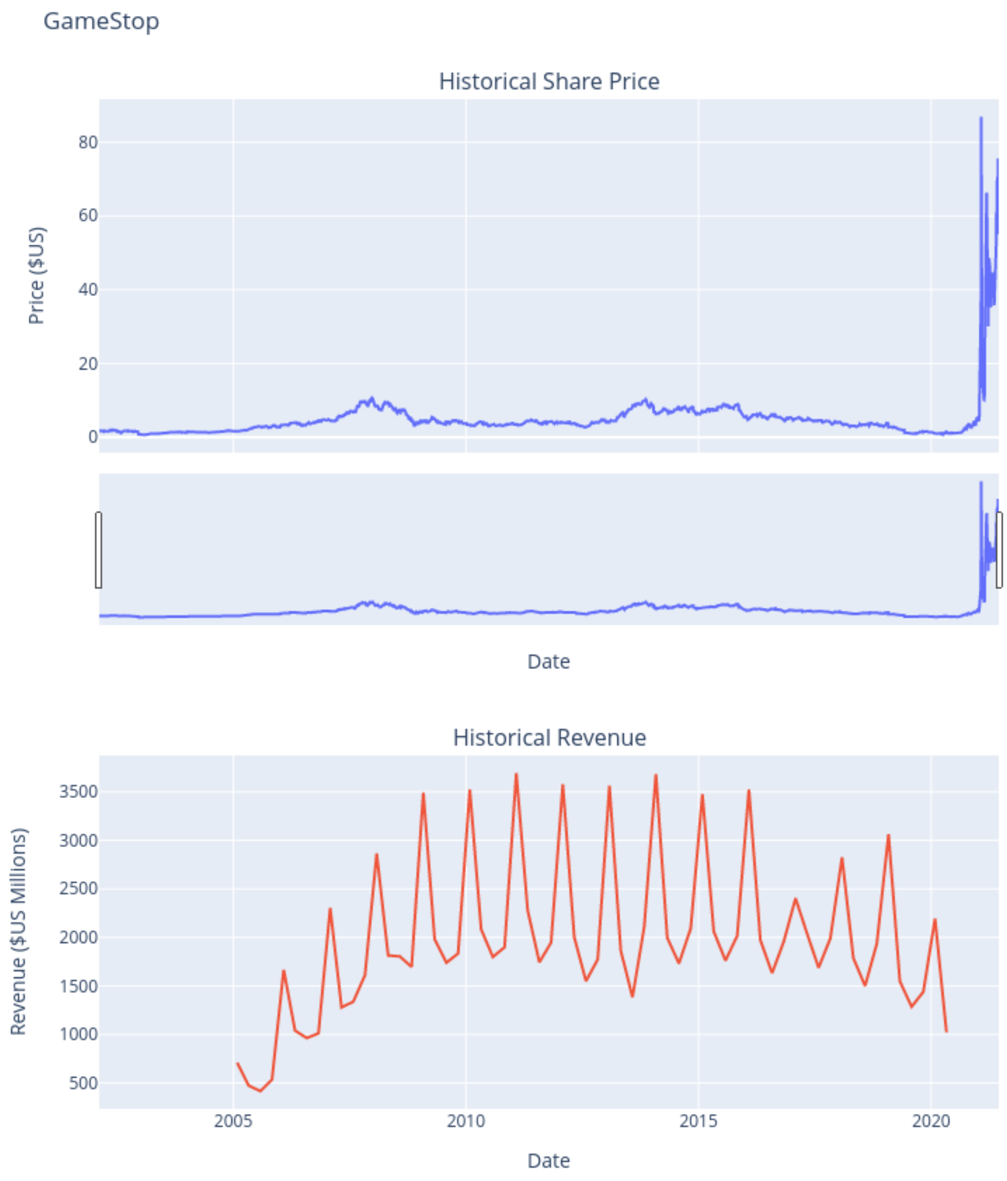
```
[37]: make_graph(tesla_data, tesla_revenue, 'Tesla')
```



0.7 Question 6: Plot GameStop Stock Graph

Use the `make_graph` function to graph the GameStop Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(gme_data, gme_revenue, 'GameStop')`. Note the graph will only show data upto June 2021.

[38]:
```
make_graph(gme_data, gme_revenue, 'GameStop')
```



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

0.8 Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2022-02-28	1.2	Lakshmi Holla	Changed the URL of GameStop
2020-11-10	1.1	Malika Singla	Deleted the Optional part
2020-08-27	1.0	Malika Singla	Added lab to GitLab

##