

# Final Assignment

January 23, 2023

## 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

- <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>

Estimated Time Needed: 30 min

```
[1]: !pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0
```

Collecting yfinance==0.1.67

Downloading yfinance-0.1.67-py2.py3-none-any.whl (25 kB)

Requirement already satisfied: pandas>=0.24 in

/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance==0.1.67) (1.3.5)

Requirement already satisfied: requests>=2.20 in

/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance==0.1.67) (2.28.1)

Requirement already satisfied: lxml>=4.5.1 in

/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance==0.1.67) (4.9.1)

Collecting multitasking>=0.0.7

Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)

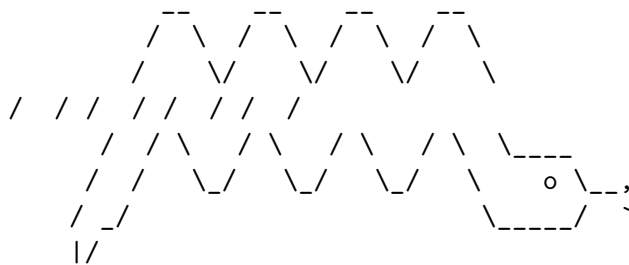
Requirement already satisfied: numpy>=1.15 in

/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from

```

yfinance==0.1.67) (1.21.6)
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas>=0.24->yfinance==0.1.67) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas>=0.24->yfinance==0.1.67) (2022.6)
Requirement already satisfied: charset-normalizer<3,>=2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (2.1.1)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (2022.9.24)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (1.26.13)
Requirement already satisfied: idna<4,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (3.4)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from python-
dateutil>=2.7.3->pandas>=0.24->yfinance==0.1.67) (1.16.0)
Installing collected packages: multitasking, yfinance
Successfully installed multitasking-0.0.11 yfinance-0.1.67

```



mamba (0.15.3) supported by @QuantStack

GitHub: <https://github.com/mamba-org/mamba>

Twitter: <https://twitter.com/QuantStack>

Looking for: ['bs4==4.10.0']

```
pkgs/main/noarch      [<=>          ] (00m:00s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/r/linux-64       [<=>          ] (00m:00s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [<=>          ] (00m:00s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [<=>          ] (00m:00s)
pkgs/main/noarch      [=>          ] (00m:00s) 799 KB / ?? (2.58 MB/s)
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/main/noarch      [<=>          ] (00m:00s) Finalizing...
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/main/noarch      [<=>          ] (00m:00s) Done
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/main/noarch      [=====] (00m:00s) Done
pkgs/r/linux-64       [=>          ] (00m:00s) 756 KB / ?? (2.44 MB/s)
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/r/linux-64       [<=>          ] (00m:00s) Finalizing...
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/r/linux-64       [<=>          ] (00m:00s) Done
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/r/linux-64       [=====] (00m:00s) Done
pkgs/r/noarch         [=>          ] (00m:00s) 668 KB / ?? (2.15 MB/s)
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/r/noarch         [<=>          ] (00m:00s) Finalizing...
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/r/noarch         [<=>          ] (00m:00s) Done
pkgs/main/linux-64    [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/r/noarch         [=====] (00m:00s) Done
```

```

pkgs/main/linux-64      [=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/main/linux-64      [<=>          ] (00m:00s) 716 KB / ?? (2.30 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 1 MB / ?? (2.93 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 1 MB / ?? (2.93 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 2 MB / ?? (3.64 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 2 MB / ?? (3.64 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 3 MB / ?? (3.85 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 3 MB / ?? (3.85 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 4 MB / ?? (4.04 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 4 MB / ?? (4.04 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) 4 MB / ?? (4.20 MB/s)
pkgs/main/linux-64      [ <=>          ] (00m:00s) Finalizing...
pkgs/main/linux-64      [ <=>          ] (00m:00s) Done
pkgs/main/linux-64      [=====] (00m:00s) Done

```

Pinned packages:

- python 3.7.\*

Transaction

Prefix: /home/jupyterlab/conda/envs/python

Updating specs:

- bs4==4.10.0
- ca-certificates
- certifi
- openssl

Package	Version	Build	Channel	Size
---------	---------	-------	---------	------

Install:

+ bs4	4.10.0	hd3eb1b0_0	pkgs/main/noarch	
10 KB				

Change:

- openssl	1.1.1s	h0b41bf4_1	installed	
+ openssl	1.1.1s	h7f8727e_0	pkgs/main/linux-64	
4 MB				

Upgrade:

```

- ca-certificates 2022.9.24 ha878542_0 installed
+ ca-certificates 2022.10.11 h06a4308_0 pkgs/main/linux-64
124 KB
- certifi 2022.9.24 pyhd8ed1ab_0 installed
+ certifi 2022.12.7 py37h06a4308_0 pkgs/main/linux-64
150 KB

```

Downgrade:

```

- beautifulsoup4 4.11.1 pyha770c72_0 installed
+ beautifulsoup4 4.10.0 pyh06a4308_0 pkgs/main/noarch
85 KB

```

Summary:

```

Install: 1 packages
Change: 1 packages
Upgrade: 2 packages
Downgrade: 1 packages

```

Total download: 4 MB

```

Downloading [> ] (00m:00s) 518.56 KB/s
Extracting [> ] (--:-- )
Finished beautifulsoup4 (00m:00s) 85
KB 518 KB/s
Downloading [> ] (00m:00s) 518.56 KB/s
Extracting [> ] (--:-- )
Downloading [> ] (00m:00s) 518.56 KB/s
Extracting [> ] (--:-- )
Downloading [> ] (00m:00s) 518.56 KB/s
Extracting [> ] (--:-- )
Downloading [> ] (00m:00s) 518.56 KB/s
Extracting [=====> ] (00m:00s) 1 / 5
Downloading [> ] (00m:00s) 546.43 KB/s
Extracting [=====> ] (00m:00s) 1 / 5
Finished bs4 (00m:00s) 10
KB 58 KB/s
Downloading [> ] (00m:00s) 546.43 KB/s
Extracting [=====> ] (00m:00s) 1 / 5
Downloading [> ] (00m:00s) 546.43 KB/s
Extracting [=====> ] (00m:00s) 1 / 5
Downloading [> ] (00m:00s) 546.43 KB/s
Extracting [=====> ] (00m:00s) 1 / 5

```

```

Downloading [==> ] (00m:00s) 1.21 MB/s
Extracting [=====> ] (00m:00s) 1 / 5
Downloading [==> ] (00m:00s) 1.21 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Finished ca-certificates (00m:00s) 124
KB 702 KB/s
Downloading [==> ] (00m:00s) 1.21 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Downloading [==> ] (00m:00s) 1.21 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Downloading [==> ] (00m:00s) 1.21 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Downloading [===> ] (00m:00s) 2.01 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Finished certifi (00m:00s) 150
KB 836 KB/s
Downloading [===> ] (00m:00s) 2.01 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Downloading [===> ] (00m:00s) 2.01 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Downloading [===> ] (00m:00s) 2.01 MB/s
Extracting [=====> ] (00m:00s) 2 / 5
Downloading [===> ] (00m:00s) 2.01 MB/s
Extracting [=====> ] (00m:00s) 3 / 5
Downloading [=====> ] (00m:00s) 20.16 MB/s
Extracting [=====> ] (00m:00s) 3 / 5
Downloading [=====> ] (00m:00s) 20.16 MB/s
Extracting [=====> ] (00m:00s) 4 / 5
Finished openssl (00m:00s) 4
MB 18 MB/s
Downloading [=====> ] (00m:00s) 20.16 MB/s
Extracting [=====> ] (00m:00s) 4 / 5
Downloading [=====> ] (00m:00s) 20.16 MB/s
Extracting [=====> ] (00m:00s) 4 / 5
Downloading [=====> ] (00m:00s) 20.16 MB/s
Extracting [=====> ] (00m:00s) 4 / 5
Downloading [=====> ] (00m:00s) 20.16 MB/s
Extracting [=====> ] (00m:00s) 5 / 5
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Collecting nbformat==4.2.0
  Downloading nbformat-4.2.0-py2.py3-none-any.whl (153 kB)
    153.3/153.3 kB
18.5 MB/s eta 0:00:00
Requirement already satisfied: jupyter-core in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (4.12.0)

```

```

Requirement already satisfied: traitlets>=4.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (5.6.0)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (4.17.3)
Requirement already satisfied: ipython-genutils in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (0.2.0)
Requirement already satisfied: importlib-resources>=1.4.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (5.10.1)
Requirement already satisfied: attrs>=17.4.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (22.1.0)
Requirement already satisfied: typing-extensions in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (4.4.0)
Requirement already satisfied: pkgutil-resolve-name>=1.3.10 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (1.3.10)
Requirement already satisfied: importlib-metadata in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (4.11.4)
Requirement already satisfied: pyparsing!=0.17.0,!0.17.1,!0.17.2,>=0.14.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (0.19.2)
Requirement already satisfied: zipp>=3.1.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from importlib-
resources>=1.4.0->jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (3.11.0)
Installing collected packages: nbformat
  Attempting uninstall: nbformat
    Found existing installation: nbformat 5.7.0
    Uninstalling nbformat-5.7.0:
      Successfully uninstalled nbformat-5.7.0

```

ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.

nbconvert 7.2.6 requires nbformat>=5.1, but you have nbformat 4.2.0 which is incompatible.

nbclient 0.7.2 requires nbformat>=5.1, but you have nbformat 4.2.0 which is incompatible.

jupyter-server 1.23.3 requires nbformat>=5.2.0, but you have nbformat 4.2.0 which is incompatible.

Successfully installed nbformat-4.2.0

```
[13]: !pip install html5lib
```

Collecting html5lib

Downloading html5lib-1.1-py2.py3-none-any.whl (112 kB)

112.2/112.2 kB

13.4 MB/s eta 0:00:00

Requirement already satisfied: webencodings in

/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from html5lib) (0.5.1)

Requirement already satisfied: six>=1.9 in

/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from html5lib) (1.16.0)

Installing collected packages: html5lib

Successfully installed html5lib-1.1

```
[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.

```
[4]:
```



```
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_rangeflider_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.

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

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")
```

**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.

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

```
[7]:
```

	Date	Open	High	Low	Close	Volume	Dividends	\
0	2010-06-29	1.266667	1.666667	1.169333	1.592667	281494500	0	
1	2010-06-30	1.719333	2.028000	1.553333	1.588667	257806500	0	
2	2010-07-01	1.666667	1.728000	1.351333	1.464000	123282000	0	
3	2010-07-02	1.533333	1.540000	1.247333	1.280000	77097000	0	
4	2010-07-06	1.333333	1.333333	1.055333	1.074000	103003500	0	

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

### 0.3 Question 2: Use Webscraping to Extract Tesla 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/revenue.htm> Save the text of the response as a variable named `html_data`.

```
[10]: tesla_url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
↳IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm"
html_data = requests.get(tesla_url).text
```

Parse the html data using `beautiful_soup`.

```
[14]: tesla_soup = BeautifulSoup(html_data, "html.parser")
```

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

```
[18]: tesla_tables = tesla_soup.find_all('table')

for index,table in enumerate(tesla_tables):
    if ("Tesla Quarterly Revenue" in str(table)):
        tesla_table_index = index

tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])

for row in tesla_tables[tesla_table_index].tbody.find_all("tr"):
    col = row.find_all("td")
    if (col != []):
        date = col[0].text
```

```
revenue = col[1].text.replace("$", "").replace(",", "")
tesla_revenue = tesla_revenue.append({"Date" : date, "Revenue" : revenue}, ignore_index=True)
```

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

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

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/ipykernel_launcher.py:1: FutureWarning: The default value of regex will
change from True to False in a future version.
    """Entry point for launching an IPython kernel.
```

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

```
[20]: 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.

```
[22]: tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
tesla_revenue.tail()
```

```
[22]:
```

	Date	Revenue
48	2010-09-30	31
49	2010-06-30	28
50	2010-03-31	21
52	2009-09-30	46
53	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.

```
[23]: gamestop = yf.Ticker("GME")
```

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.

```
[24]: gme_data = gamestop.history(period="max")
```

**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.

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

```
[25]:      Date      Open      High      Low      Close      Volume  Dividends  \
0 2002-02-13  1.620129  1.693350  1.603296  1.691667  76216000      0.0
1 2002-02-14  1.712707  1.716074  1.670626  1.683250  11021600      0.0
2 2002-02-15  1.683250  1.687458  1.658002  1.674834   8389600      0.0
3 2002-02-19  1.666417  1.666417  1.578047  1.607504   7410400      0.0
4 2002-02-20  1.615921  1.662210  1.603296  1.662210   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]: gme_url = "https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue"
      gme_html_data = requests.get(gme_url).text
```

Parse the html data using `beautiful_soup`.

```
[27]: gme_soup = BeautifulSoup(gme_html_data, "html.parser")
```

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

```
[28]: gme_tables = gme_soup.find_all('table')

      for index,table in enumerate(gme_tables):
```

```

    if ("GameStop Quarterly Revenue" in str(table)):
        gme_table_index = index

gme_revenue = pd.DataFrame(columns=["Date", "Revenue"])

for row in gme_tables[gme_table_index].tbody.find_all("tr"):
    col = row.find_all("td")
    if (col != []):
        date = col[0].text
        revenue = col[1].text.replace("$", "").replace(",", "")
        gme_revenue = gme_revenue.append({"Date" : date, "Revenue" : revenue},
        ignore_index=True)

```

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

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

```

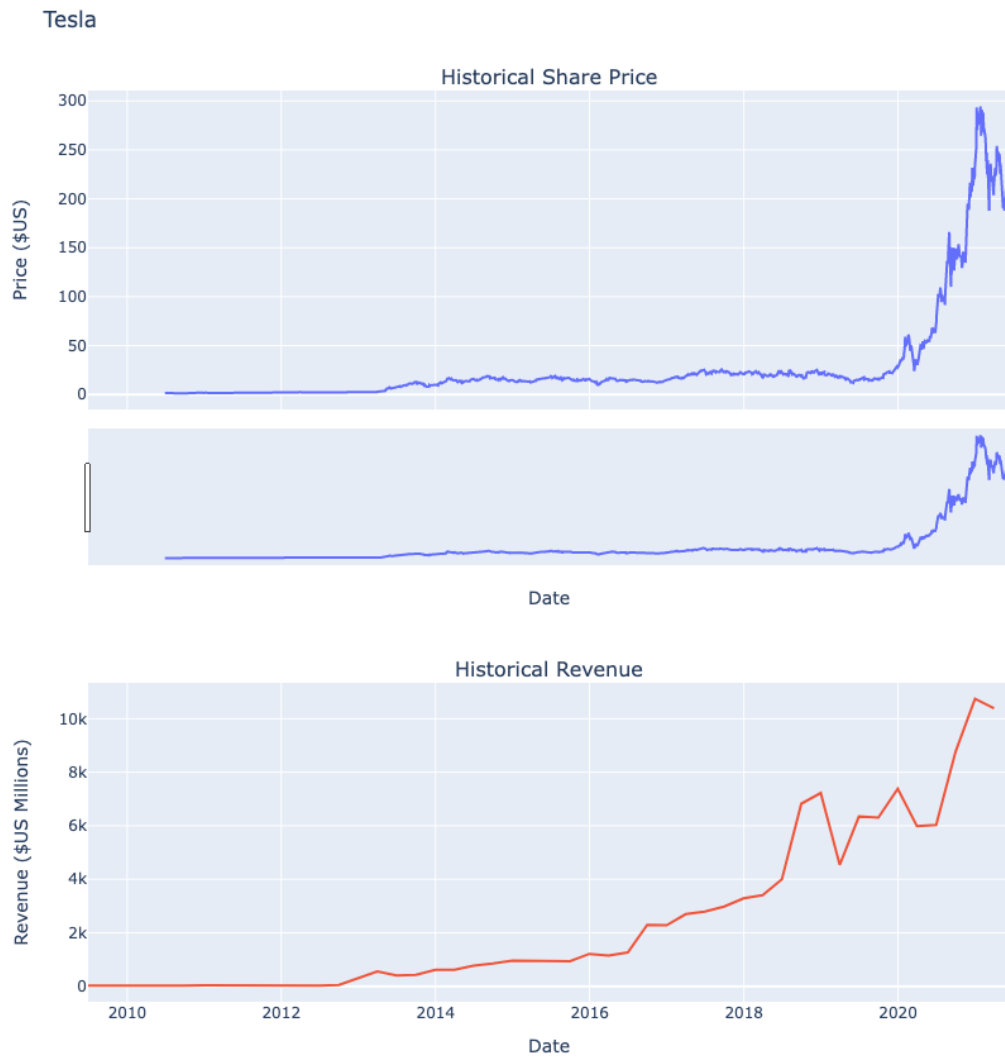
[29]:
      Date Revenue
51  2010-01-31   3524
52  2009-10-31   1835
53  2009-07-31   1739
54  2009-04-30   1981
55  2009-01-31   3492

```

## 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.

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

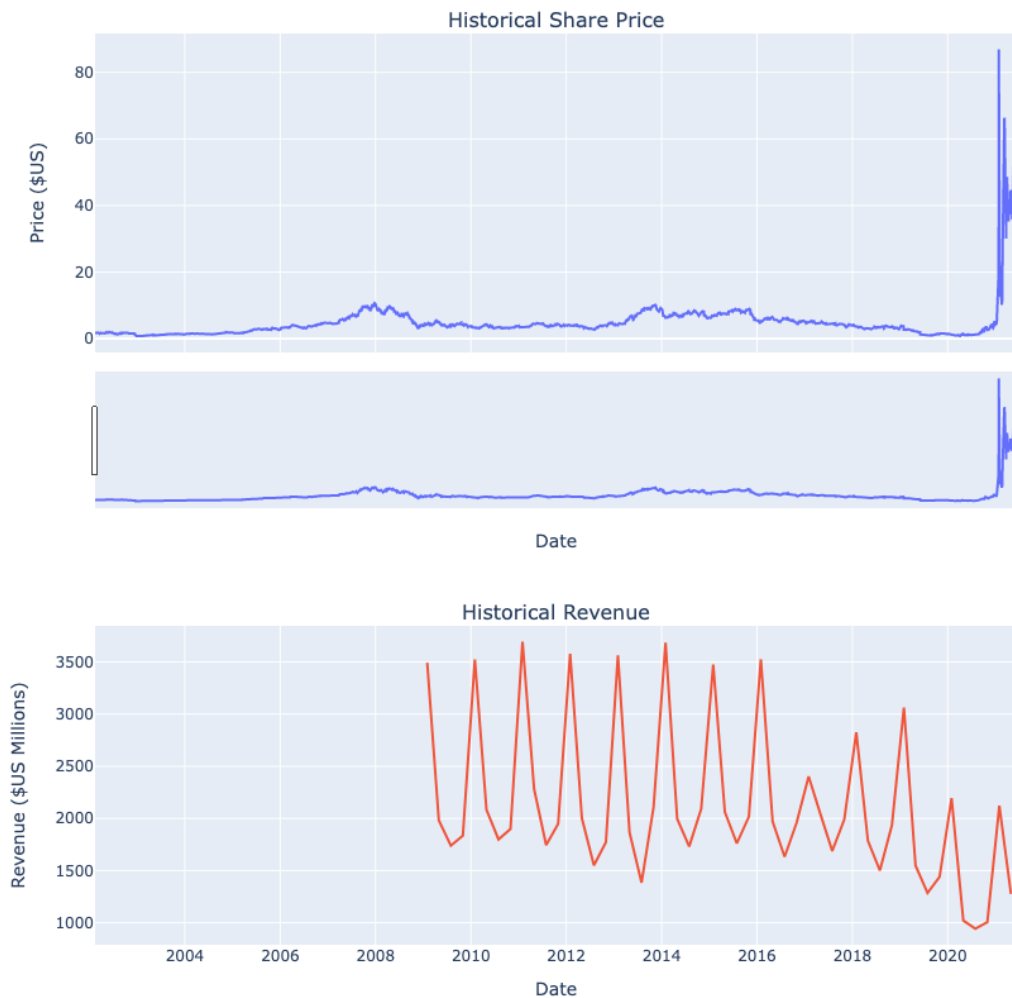


## 0.7 Question 6: Plot GameStop Stock Graph

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.

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

## 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

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-08-27	1.0	Malika Singla	Added lab to GitLab

##

© IBM Corporation 2020. All rights reserved.