Final Assignment

February 9, 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.

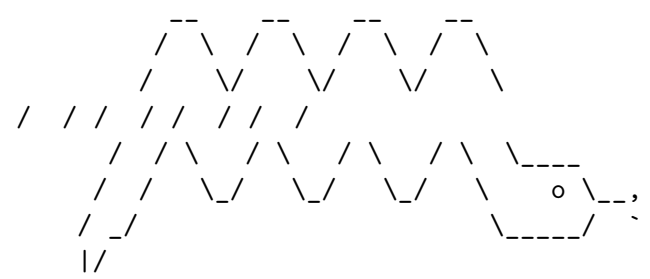
Table of Contents

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) (4.6.4)
Requirement already satisfied: multitasking>=0.0.7 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (0.0.11)
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) (2022.12.7)
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: yfinance
  Attempting uninstall: yfinance
    Found existing installation: yfinance 0.2.4
    Uninstalling yfinance-0.2.4:
      Successfully uninstalled yfinance-0.2.4
Successfully installed 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/r/linux-64          [>                   ] (--:--) No change
pkgs/r/linux-64          [====================] (00m:00s) No change
pkgs/main/linux-64       [>                   ] (--:--) No change
pkgs/main/linux-64       [====================] (00m:00s) No change
pkgs/r/noarch            [>                   ] (--:--) No change
pkgs/r/noarch            [====================] (00m:00s) No change
pkgs/main/noarch         [>                   ] (--:--) No change
pkgs/main/noarch         [====================] (00m:00s) No change

Pinned packages:
  - python 3.7.*


Transaction

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

  All requested packages already installed

Collecting nbformat==4.2.0
  Downloading nbformat-4.2.0-py2.py3-none-any.whl (153 kB)
                       153.3/153.3 kB
15.3 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: 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: 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: 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: pyrsistent!=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
```

[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`.

[10]:
```
ticker = 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.

[22]:
```
tesla_data = ticker.history(period="max")
tesla_data
```

[22]:

Date	Open	High	Low	Close	Volume	\
2010-06-29	1.266667	1.666667	1.169333	1.592667	281494500	
2010-06-30	1.719333	2.028000	1.553333	1.588667	257806500	
2010-07-01	1.666667	1.728000	1.351333	1.464000	123282000	
2010-07-02	1.533333	1.540000	1.247333	1.280000	77097000	
2010-07-06	1.333333	1.333333	1.055333	1.074000	103003500	
…	…	…	…	…	…	
2023-02-02	187.330002	196.750000	182.610001	188.270004	217448300	
2023-02-03	183.949997	199.000000	183.690002	189.979996	231684200	

```
2023-02-06  193.009995  198.169998  189.919998  194.759995  186188100
2023-02-07  196.429993  197.500000  189.550003  196.809998  186010300
2023-02-08  196.100006  203.000000  194.309998  201.289993  180154500

            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
…                   …           …
2023-02-02          0           0.0
2023-02-03          0           0.0
2023-02-06          0           0.0
2023-02-07          0           0.0
2023-02-08          0           0.0

[3176 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.

[24]:
```
#tesla_data.set_index('Date', inplace=True)
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

[24]:
```
   index       Date      Open      High       Low     Close     Volume  \
0      0 2010-06-29  1.266667  1.666667  1.169333  1.592667  281494500
1      1 2010-06-30  1.719333  2.028000  1.553333  1.588667  257806500
2      2 2010-07-01  1.666667  1.728000  1.351333  1.464000  123282000
3      3 2010-07-02  1.533333  1.540000  1.247333  1.280000   77097000
4      4 2010-07-06  1.333333  1.333333  1.055333  1.074000  103003500

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

[27]:
```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm"
response = requests.get(url)
html_data = response.text
```

Parse the html data using `beautiful_soup`.

[28]:
```
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
```

[31]:
```
table = soup.find_all('table')[1]
tesla_revenue = pd.read_html(str(table))[0]
tesla_revenue.columns = ['Date', 'Revenue']
tesla_revenue
```

[31]:

	Date	Revenue
0	2022-09-30	$21,454
1	2022-06-30	$16,934
2	2022-03-31	$18,756
3	2021-12-31	$17,719
4	2021-09-30	$13,757
5	2021-06-30	$11,958
6	2021-03-31	$10,389
7	2020-12-31	$10,744
8	2020-09-30	$8,771
9	2020-06-30	$6,036
10	2020-03-31	$5,985
11	2019-12-31	$7,384
12	2019-09-30	$6,303
13	2019-06-30	$6,350
14	2019-03-31	$4,541
15	2018-12-31	$7,226
16	2018-09-30	$6,824
17	2018-06-30	$4,002
18	2018-03-31	$3,409
19	2017-12-31	$3,288

```
20  2017-09-30   $2,985
21  2017-06-30   $2,790
22  2017-03-31   $2,696
23  2016-12-31   $2,285
24  2016-09-30   $2,298
25  2016-06-30   $1,270
26  2016-03-31   $1,147
27  2015-12-31   $1,214
28  2015-09-30     $937
29  2015-06-30     $955
30  2015-03-31     $940
31  2014-12-31     $957
32  2014-09-30     $852
33  2014-06-30     $769
34  2014-03-31     $621
35  2013-12-31     $615
36  2013-09-30     $431
37  2013-06-30     $405
38  2013-03-31     $562
39  2012-12-31     $306
40  2012-09-30      $50
41  2012-06-30      $27
42  2012-03-31      $30
43  2011-12-31      $39
44  2011-09-30      $58
45  2011-06-30      $58
46  2011-03-31      $49
47  2010-12-31      $36
48  2010-09-30      $31
49  2010-06-30      $28
50  2010-03-31      $21
51  2009-12-31      NaN
52  2009-09-30      $46
53  2009-06-30      $27
```

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

[32]:
```
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.

[33]:
```
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.

[35]:
```
tesla_revenue.tail(5)
```

[35]:
```
          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`.

[38]:
```
gme = yf.Ticker("GME")
gme
```

[38]:
```
yfinance.Ticker object <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.

[42]:
```
gme_data = gme.history(period="max")
gme_data
```

[42]:
```
                 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   1.683250   1.687458   1.658001   1.674834   8389600        0.0
2002-02-19   1.666418   1.666418   1.578047   1.607504   7410400        0.0
2002-02-20   1.615921   1.662210   1.603296   1.662210   6892800        0.0
…                 …          …          …          …         …          …
2023-02-02  22.440001  24.400000  22.219999  22.700001   7655700        0.0
2023-02-03  22.010000  23.930000  21.799999  22.250000   4930200        0.0
2023-02-06  21.879999  24.879999  21.770000  23.860001   8708400        0.0
2023-02-07  23.000000  23.049999  20.500000  21.219999   9093700        0.0
2023-02-08  21.430000  21.549999  20.610001  21.100000   2841800        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
…                    …
2023-02-02           0.0
2023-02-03           0.0
2023-02-06           0.0
2023-02-07           0.0
2023-02-08           0.0

[5284 rows x 7 columns]
```

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.

[47]:
```
gme_data.reset_index(inplace=True)
gme_data.head(5)
```

[47]:

	level_0	index	Date	Open	High	Low	Close	\
0	0	0	2002-02-13	1.620128	1.693350	1.603296	1.691666	
1	1	1	2002-02-14	1.712707	1.716074	1.670626	1.683250	
2	2	2	2002-02-15	1.683250	1.687458	1.658001	1.674834	
3	3	3	2002-02-19	1.666418	1.666418	1.578047	1.607504	
4	4	4	2002-02-20	1.615921	1.662210	1.603296	1.662210	

	Volume	Dividends	Stock Splits
0	76216000	0.0	0.0
1	11021600	0.0	0.0
2	8389600	0.0	0.0
3	7410400	0.0	0.0
4	6892800	0.0	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`.

[53]:
```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"
response = requests.get(url)
html_data = response.text
#html_data
```

Parse the html data using `beautiful_soup`.

[57]:
```
soup = BeautifulSoup(html_data, 'html.parser')
#soup
```

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
```

[64]:
```
table = soup.find_all("table")[1]
gme_revenue = pd.read_html(str(table))[0]
gme_revenue.columns = ['Date', 'Revenue']


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

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/ipykernel_launcher.py:6: FutureWarning: The default value of regex will
change from True to False in a future version.
```

[64]:

	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
..	…	…
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

[62 rows x 2 columns]

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

[66]:
```
gme_revenue.tail()
```

[66]:

	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

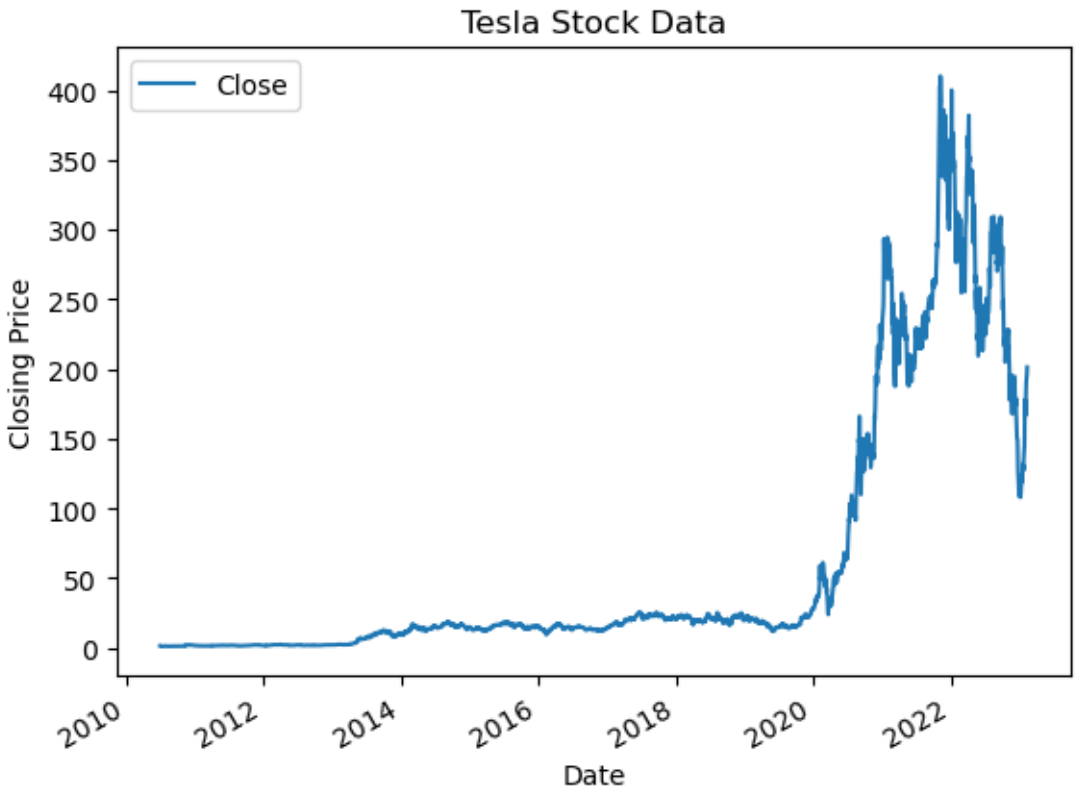
0.6 Question 5: Plot Tesla Stock Graph

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.

[69]:
```
import matplotlib.pyplot as plt

def make_graph(data,title):
    data.plot(x='Date', y='Close')    #, kind = 'Line')
    plt.title(title)
    plt.xlabel('Date')
    plt.ylabel('Closing Price')
    plt.show()

make_graph(tesla_data, 'Tesla Stock Data')
```



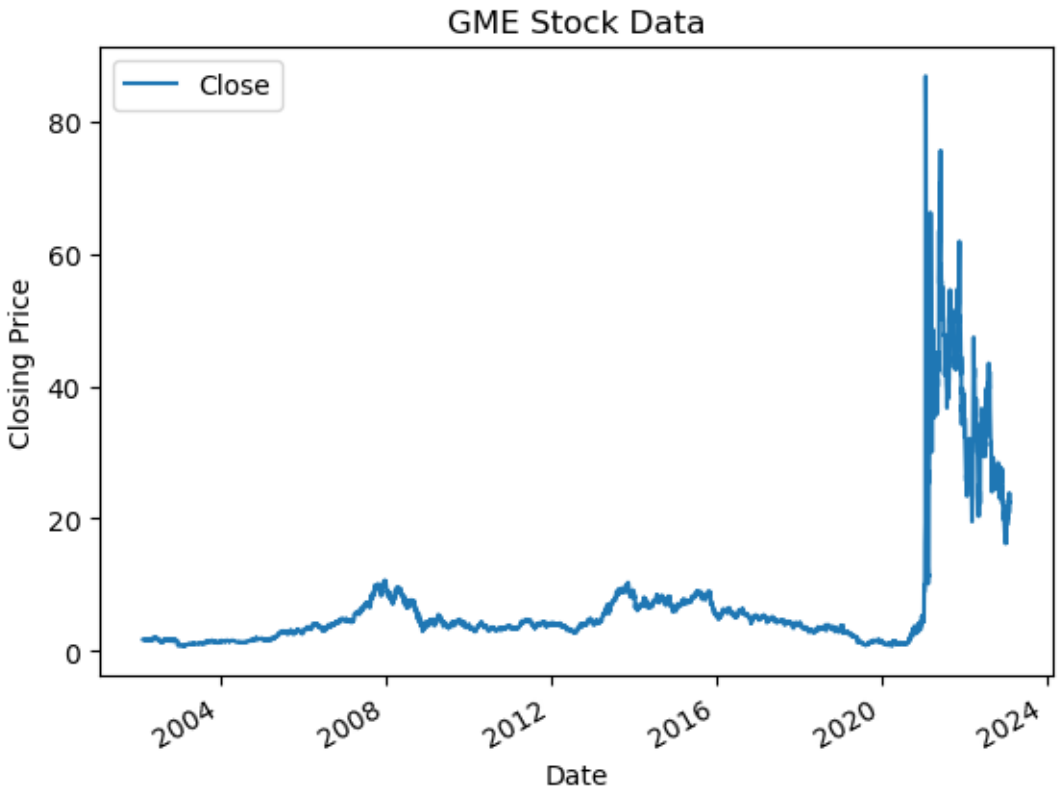
0.7 Question 6: Plot GameStop Stock Graph

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.

[71]:
```
#def make_graph(data, revenue, title):
#    fig, ax1 = plt.subplots()

def make_graph(data, revenue, title):
    data.plot(x='Date', y='Close')    #, kind = 'Line')
    plt.title(title)
    plt.xlabel('Date')
    plt.ylabel('Closing Price')
    plt.show()

make_graph(gme_data, gme_revenue,  'GME Stock Data')
```



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

##

© IBM Corporation 2020. All rights reserved.