Final Assignment

June 18, 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>
```

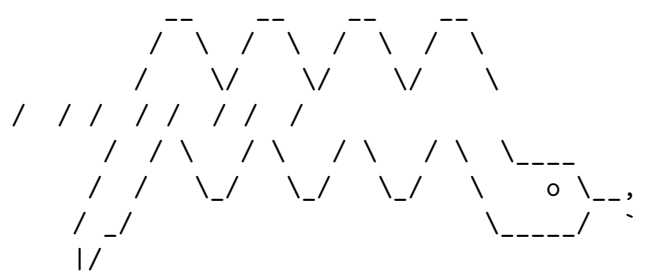
Estimated Time Needed: 30 min

[2]:
```
!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
```

```
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) (2.27.1)
Requirement already satisfied: lxml>=4.5.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (4.9.0)
Collecting multitasking>=0.0.7
  Downloading multitasking-0.0.10.tar.gz (8.2 kB)
```

```
  Preparing metadata (setup.py) … done
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) (2022.1)
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) (2022.5.18.1)
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) (1.26.9)
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) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (2.0.12)
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)
Building wheels for collected packages: multitasking
  Building wheel for multitasking (setup.py) … done
  Created wheel for multitasking: filename=multitasking-0.0.10-py3-none-
any.whl size=8498
sha256=522c648161e576eaf3f6ad722e8f79cd93805fe91c59b08bc34ef910ef21eb24
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/34/ba/79/c0260c6f1a03f
420ec7673eff9981778f293b9107974679e36
Successfully built multitasking
Installing collected packages: multitasking, yfinance
Successfully installed multitasking-0.0.10 yfinance-0.1.67
Collecting pandas==1.3.3
  Downloading
pandas-1.3.3-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.3 MB)
                              11.3/11.3 MB
69.1 MB/s eta 0:00:0000:0100:01
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.3) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.3) (2022.1)
Requirement already satisfied: numpy>=1.17.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
```

```
pandas==1.3.3) (1.21.6)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from python-
dateutil>=2.7.3->pandas==1.3.3) (1.16.0)
Installing collected packages: pandas
  Attempting uninstall: pandas
    Found existing installation: pandas 1.3.5
    Uninstalling pandas-1.3.5:
      Successfully uninstalled pandas-1.3.5
Successfully installed pandas-1.3.3
Collecting requests==2.26.0
  Downloading requests-2.26.0-py2.py3-none-any.whl (62 kB)
                           62.3/62.3 kB
8.1 MB/s eta 0:00:00
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (2022.5.18.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (1.26.9)
Requirement already satisfied: idna<4,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (2.0.12)
Installing collected packages: requests
  Attempting uninstall: requests
    Found existing installation: requests 2.27.1
    Uninstalling requests-2.27.1:
      Successfully uninstalled requests-2.27.1
Successfully installed requests-2.26.0
```



```
        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) 396  B / ?? (2.57 KB/s)
pkgs/main/noarch         [=>                ] (00m:00s) 396  B / ?? (2.57 KB/s)
pkgs/r/noarch            [>                  ] (--:--) Finalizing…
pkgs/main/noarch         [=>                ] (00m:00s) 396  B / ?? (2.57 KB/s)
pkgs/r/noarch            [>                  ] (--:--) Done
pkgs/r/noarch            [====================] (00m:00s) Done
pkgs/main/noarch         [=>                ] (00m:00s) 396  B / ?? (2.57 KB/s)
pkgs/main/noarch         [<=>               ] (00m:00s) 396  B / ?? (2.57 KB/s)
pkgs/main/noarch         [<=>               ] (00m:00s) 704 KB / ?? (2.28 MB/s)
pkgs/main/noarch         [<=>               ] (00m:00s) 704 KB / ?? (2.28 MB/s)
pkgs/r/linux-64          [<=>                ] (00m:00s)
pkgs/main/noarch         [<=>               ] (00m:00s) 704 KB / ?? (2.28 MB/s)
pkgs/r/linux-64          [=>                ] (00m:00s) 668 KB / ?? (2.16 MB/s)
pkgs/main/noarch         [<=>               ] (00m:00s) 704 KB / ?? (2.28 MB/s)
pkgs/r/linux-64          [=>                ] (00m:00s) 668 KB / ?? (2.16 MB/s)
pkgs/main/linux-64       [<=>                ] (00m:00s)
pkgs/main/noarch         [<=>               ] (00m:00s) 704 KB / ?? (2.28 MB/s)
pkgs/r/linux-64          [=>                ] (00m:00s) 668 KB / ?? (2.16 MB/s)
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/main/noarch         [ <=>               ] (00m:00s) Finalizing…
pkgs/r/linux-64          [=>                ] (00m:00s) 668 KB / ?? (2.16 MB/s)
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/main/noarch         [ <=>               ] (00m:00s) Done
pkgs/r/linux-64          [=>                ] (00m:00s) 668 KB / ?? (2.16 MB/s)
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/main/noarch         [====================] (00m:00s) Done
pkgs/r/linux-64          [=>                ] (00m:00s) 668 KB / ?? (2.16 MB/s)
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/r/linux-64          [<=>                ] (00m:00s) Finalizing…
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/r/linux-64          [<=>                ] (00m:00s) Done
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/r/linux-64          [====================] (00m:00s) Done
pkgs/main/linux-64       [=>                ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/main/linux-64       [<=>               ] (00m:00s) 660 KB / ?? (2.13 MB/s)
pkgs/main/linux-64       [ <=>               ] (00m:00s) 1 MB / ?? (2.91 MB/s)
pkgs/main/linux-64       [  <=>              ] (00m:00s) 1 MB / ?? (2.91 MB/s)
```

```
pkgs/main/linux-64       [  <=>               ] (00m:00s) 2 MB / ?? (3.30 MB/s)
pkgs/main/linux-64       [   <=>              ] (00m:00s) 2 MB / ?? (3.30 MB/s)
pkgs/main/linux-64       [   <=>              ] (00m:00s) 3 MB / ?? (3.52 MB/s)
pkgs/main/linux-64       [    <=>             ] (00m:00s) 3 MB / ?? (3.52 MB/s)
pkgs/main/linux-64       [    <=>             ] (00m:00s) 3 MB / ?? (3.71 MB/s)
pkgs/main/linux-64       [     <=>            ] (00m:00s) 3 MB / ?? (3.71 MB/s)
pkgs/main/linux-64       [     <=>            ] (00m:00s) 4 MB / ?? (3.84 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:


  + beautifulsoup4       4.10.0  pyh06a4308_0    pkgs/main/noarch
85 KB
  + bs4                  4.10.0  hd3eb1b0_0      pkgs/main/noarch
10 KB
  + soupsieve             2.3.1  pyhd3eb1b0_0    pkgs/main/noarch
34 KB

  Change:


  - certifi         2022.5.18.1  py37h89c1867_0  installed
  + certifi         2022.5.18.1  py37h06a4308_0  pkgs/main/linux-64
147 KB
  - openssl              1.1.1o  h166bdaf_0      installed
  + openssl              1.1.1o  h7f8727e_0      pkgs/main/linux-64
3 MB
```

```
  Summary:

  Install: 3 packages
  Change: 2 packages

  Total download: 3 MB


Downloading  [>                                          ] (00m:00s)    2.77 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)    2.77 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)   67.69 KB/s
Extracting   [>                                          ] (--:--)
Finished bs4                                 (00m:00s)              10
KB     65 KB/s
Downloading  [>                                          ] (00m:00s)   67.69 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)   67.69 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)   67.69 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)  285.99 KB/s
Extracting   [>                                          ] (--:--)
Finished soupsieve                           (00m:00s)              34
KB    219 KB/s
Downloading  [>                                          ] (00m:00s)  285.99 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)  285.99 KB/s
Extracting   [>                                          ] (--:--)
Downloading  [>                                          ] (00m:00s)  285.99 KB/s
Extracting   [========>                                  ] (00m:00s)        1 / 5
Downloading  [>                                          ] (00m:00s)  285.99 KB/s
Extracting   [========>                                  ] (00m:00s)        1 / 5
Downloading  [>                                          ] (00m:00s)  285.99 KB/s
Extracting   [================>                          ] (00m:00s)        2 / 5
Downloading  [=>                                         ] (00m:00s)  750.81 KB/s
Extracting   [================>                          ] (00m:00s)        2 / 5
Finished beautifulsoup4                      (00m:00s)              85
KB    491 KB/s
Downloading  [=>                                         ] (00m:00s)  750.81 KB/s
Extracting   [================>                          ] (00m:00s)        2 / 5
Downloading  [=======================================>  ] (00m:00s)   14.46 MB/s
Extracting   [================>                          ] (00m:00s)        2 / 5
Downloading  [=======================================>  ] (00m:00s)   14.46 MB/s
Extracting   [================>                          ] (00m:00s)        2 / 5
```

```
Downloading  [======================================>  ] (00m:00s)   14.46 MB/s
Extracting   [================>                        ] (00m:00s)        2 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [================>                        ] (00m:00s)        2 / 5
Finished certifi                          (00m:00s)             147
KB    736 KB/s
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [================>                        ] (00m:00s)        2 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [================>                        ] (00m:00s)        2 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [========================>                ] (00m:00s)        3 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [========================>                ] (00m:00s)        3 / 5
Finished openssl                          (00m:00s)              3
MB     14 MB/s
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [========================>                ] (00m:00s)        3 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [========================>                ] (00m:00s)        3 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [================================>        ] (00m:00s)        4 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [================================>        ] (00m:00s)        4 / 5
Downloading  [========================================] (00m:00s)   13.88 MB/s
Extracting   [========================================] (00m:00s)        5 / 5
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Collecting plotly==5.3.1
  Downloading plotly-5.3.1-py2.py3-none-any.whl (23.9 MB)
                           23.9/23.9 MB
50.0 MB/s eta 0:00:0000:0100:01
Requirement already satisfied: six in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
plotly==5.3.1) (1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
plotly==5.3.1) (8.0.1)
Installing collected packages: plotly
  Attempting uninstall: plotly
    Found existing installation: plotly 5.8.0
    Uninstalling plotly-5.8.0:
      Successfully uninstalled plotly-5.8.0
```

```
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.
dash 2.4.1 requires dash-core-components==2.0.0, which is not installed.
dash 2.4.1 requires dash-html-components==2.0.0, which is not installed.
dash 2.4.1 requires dash-table==5.0.0, which is not installed.
Successfully installed plotly-5.3.1
```

[3]:
```
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.

[6]:
```
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`.

[7]:
```
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.

[8]:
```
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.

[9]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head
```

[9]:
```
<bound method NDFrame.head of            Date        Open        High
Low       Close    Volume  \
0    2010-06-29    3.800000    5.000000    3.508000    4.778000  93831500
1    2010-06-30    5.158000    6.084000    4.660000    4.766000  85935500
2    2010-07-01    5.000000    5.184000    4.054000    4.392000  41094000
3    2010-07-02    4.600000    4.620000    3.742000    3.840000  25699000
4    2010-07-06    4.000000    4.000000    3.166000    3.222000  34334500
…           …           …           …           …           …         …
3010 2022-06-13  669.500000  679.900024  644.049988  647.210022  34255800
3011 2022-06-14  654.859985  678.989990  635.210022  662.669983  32662900
3012 2022-06-15  662.750000  706.989990  654.450012  699.000000  39710600
3013 2022-06-16  668.210022  675.500000  626.080017  639.299988  35796900
3014 2022-06-17  640.299988  662.909973  639.590027  650.280029  30810900

      Dividends  Stock Splits
0             0           0.0
1             0           0.0
2             0           0.0
3             0           0.0
4             0           0.0
…           …           …
3010          0           0.0
3011          0           0.0
3012          0           0.0
3013          0           0.0
3014          0           0.0

[3015 rows x 8 columns]>
```

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/reven
Save the text of the response as a variable named `html_data`.

[10]:
```
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"

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

Parse the html data using `beautiful_soup`.

[12]:
```
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
```

[22]:
```
tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])
for row in soup.find("tbody").find_all('tr'):
    col = row.find_all("td")
    date = col[0].text
    Revenue = col[1].text
    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.

[23]:
```
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.

[24]:
```
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.

[25]:
```
tesla_revenue.tail()
```

[25]:
```
    Date Revenue
8   2013    2013
9   2012     413
10  2011     204
11  2010     117
12  2009     112
```

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`.

[28]:
```
gme = 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.

[29]:
```
gme_data = gme.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.

[30]:
```
gme_data.reset_index(inplace=True)
gme_data.head
```

[30]:
```
<bound method NDFrame.head of            Date        Open        High
Low       Close    Volume  \
0    2002-02-13    6.480514    6.773400    6.413183    6.766666  19054000
1    2002-02-14    6.850829    6.864295    6.682504    6.733002   2755400
2    2002-02-15    6.733002    6.749834    6.632007    6.699337   2097400
3    2002-02-19    6.665670    6.665670    6.312188    6.430016   1852600
4    2002-02-20    6.463683    6.648840    6.413185    6.648840   1723200
…           …           …           …           …           …         …
5118 2022-06-13  120.510002  124.570000  114.300003  118.250000   3448200
5119 2022-06-14  117.570000  128.000000  116.099998  126.169998   3184800
5120 2022-06-15  124.949997  131.960007  123.639999  129.289993   2691100
5121 2022-06-16  124.940002  129.289993  120.580002  125.730003   2498800
5122 2022-06-17  126.860001  135.860001  126.320000  135.139999   3080200

      Dividends  Stock Splits
0           0.0           0.0
1           0.0           0.0
```

```
2           0.0           0.0
3           0.0           0.0
4           0.0           0.0
…           …             …
5118        0.0           0.0
5119        0.0           0.0
5120        0.0           0.0
5121        0.0           0.0
5122        0.0           0.0

[5123 rows x 8 columns]>
```

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`.

[31]:
```
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
```

Parse the html data using `beautiful_soup`.

[32]:
```
soup = BeautifulSoup(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
```

[33]:
```
gme_revenue = pd.DataFrame(columns=["Date", "Revenue"])
for row in soup.find("tbody").find_all('tr'):
    col = row.find_all("td")
    date = col[0].text
    Revenue = col[1].text
```

```
    gme_revenue = gme_revenue.append({"Date":date, "Revenue":Revenue},␣
↪ignore_index=True)
    gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")
```

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

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

[35]:
```
gme_revenue.tail()
```

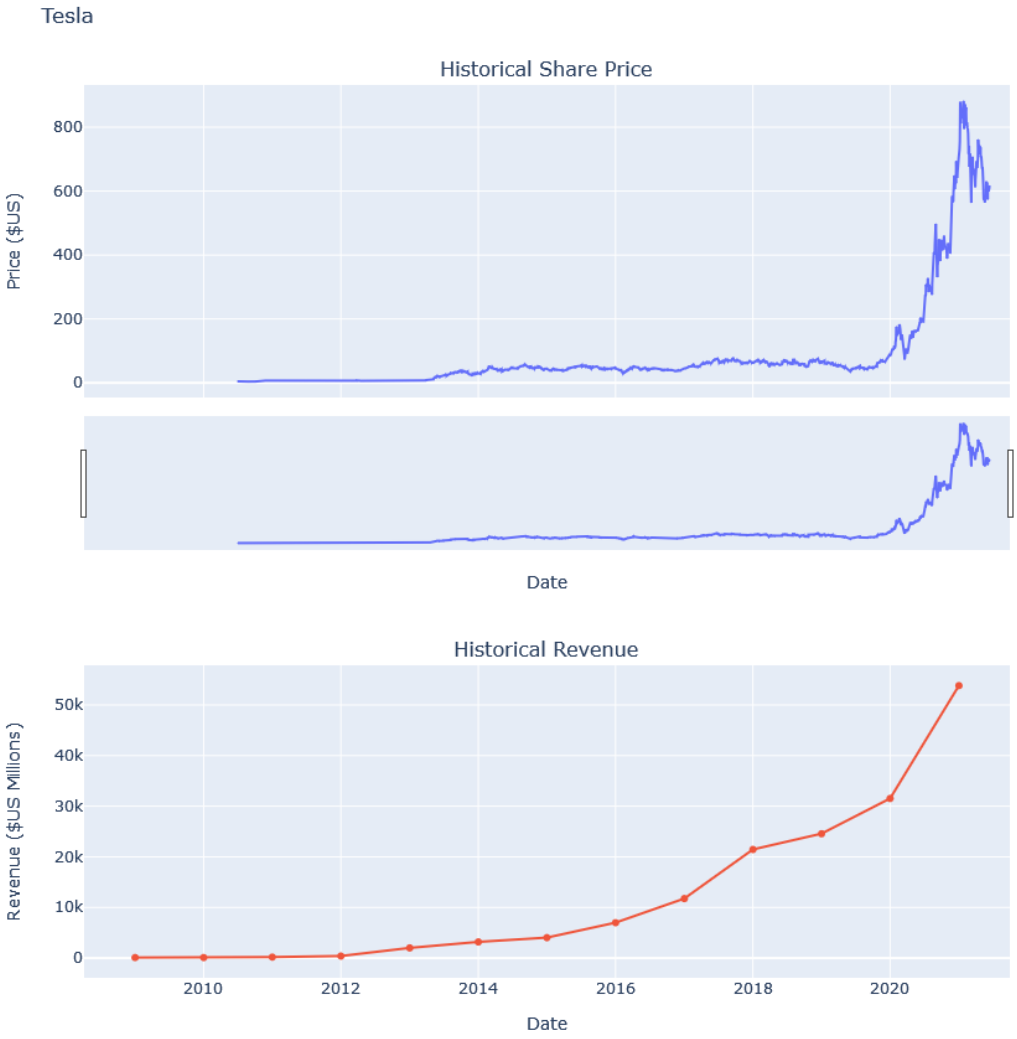
[35]:

	Date	Revenue
11	2009	8806
12	2008	7094
13	2007	5319
14	2006	3092
15	2005	1843

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.

[36]:
```
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.

[37]:
```
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

##