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**

In [1]:
```
!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: numpy>=1.15 in /opt/conda/envs/Python-3.8-mai
n/lib/python3.8/site-packages (from yfinance==0.1.67) (1.19.2)
Requirement already satisfied: pandas>=0.24 in /opt/conda/envs/Python-3.8-mai
n/lib/python3.8/site-packages (from yfinance==0.1.67) (1.2.4)
Requirement already satisfied: requests>=2.20 in /opt/conda/envs/Python-3.8-m
ain/lib/python3.8/site-packages (from yfinance==0.1.67) (2.25.1)
Requirement already satisfied: lxml>=4.5.1 in /opt/conda/envs/Python-3.8-mai
n/lib/python3.8/site-packages (from yfinance==0.1.67) (4.6.3)
Collecting multitasking>=0.0.7
  Downloading multitasking-0.0.10.tar.gz (8.2 kB)
Requirement already satisfied: python-dateutil>=2.7.3 in /opt/conda/envs/Pyth
on-3.8-main/lib/python3.8/site-packages (from pandas>=0.24->yfinance==0.1.67)
(2.8.1)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.8-mai
n/lib/python3.8/site-packages (from pandas>=0.24->yfinance==0.1.67) (2021.1)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.8-main/li
b/python3.8/site-packages (from python-dateutil>=2.7.3->pandas>=0.24->yfinanc
e==0.1.67) (1.15.0)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Python-
3.8-main/lib/python3.8/site-packages (from requests>=2.20->yfinance==0.1.67)
(2021.10.8)
Requirement already satisfied: chardet<5,>=3.0.2 in /opt/conda/envs/Python-3.
8-main/lib/python3.8/site-packages (from requests>=2.20->yfinance==0.1.67)
(3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/envs/Python-3.8-mai
n/lib/python3.8/site-packages (from requests>=2.20->yfinance==0.1.67) (2.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/envs/Pytho
n-3.8-main/lib/python3.8/site-packages (from requests>=2.20->yfinance==0.1.6
7) (1.26.6)
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.w
hl size=8487 sha256=70655a632839a978ae81718afd16ee000631f7e8dd713e8756453712c
e0b3c15
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/21/c9/66/b41c847de65c798
5db52ec21d59996841598b8b0e93f2b9500
Successfully built multitasking
Installing collected packages: multitasking, yfinance
Successfully installed multitasking-0.0.10 yfinance-0.1.67
/usr/bin/sh: mamba: command not found
```

In [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
```

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.

In [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 <= '2021-06-30']
    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()
```

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`.

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

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.

In [5]:
```
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.

In [6]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head()
#tesla_data
```

Out[6]:

	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

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/revenue (https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0220ENSkillsNetwork23455606-2021-01-01). Save the text of the response as a variable named `html_data`.

In [7]:
```
import pandas as pd
import requests
from bs4 import BeautifulSoup
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"
html_data = requests.get(url).text
#html_data
```

Parse the html data using `beautiful_soup`.

In [8]:
```
beautiful_soup = BeautifulSoup(html_data,'html5lib')
#beautiful_soup
```

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

In [9]:
```
tesla_revenue = pd.read_html(url)
tesla_revenue = tesla_revenue[1]
tesla_revenue= tesla_revenue.rename(columns={'Tesla Quarterly Revenue(Millions of US $)':'Date', 'Tesla Quarterly Revenue(Millions of US $).1':'Revenue'})
#tesla_revenue
```

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

In [10]:
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

```
/tmp/wsuser/ipykernel_206/349343550.py:1: FutureWarning: The default value of
regex will change from True to False in a future version.
  tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

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

In [11]:
```
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.

In [12]:
```
tesla_revenue.tail()
```

Out[12]:

	Date	Revenue
44	2010-09-30	31
45	2010-06-30	28
46	2010-03-31	21
48	2009-09-30	46
49	2009-06-30	27

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` .

In [13]:
```
import yfinance as yf
import pandas as pd
import requests
from bs4 import BeautifulSoup
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.

In [14]:
```
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.

In [15]:
```
gme_data.reset_index(inplace=True)
gme_data.head()
```

Out[15]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13	6.480512	6.773398	6.413182	6.766665	19054000	0.0	0.0
1	2002-02-14	6.850830	6.864296	6.682505	6.733002	2755400	0.0	0.0
2	2002-02-15	6.733003	6.749835	6.632008	6.699338	2097400	0.0	0.0
3	2002-02-19	6.665671	6.665671	6.312188	6.430016	1852600	0.0	0.0
4	2002-02-20	6.463683	6.648840	6.413185	6.648840	1723200	0.0	0.0

Question 4: Use Webscraping to Extract GME Revenue Data

Use the `requests` library to download the webpage https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue (https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0220ENSkillsNetwork23455606-2021-01-01). Save the text of the response as a variable named `html_data`.

In [16]:
```
url = "https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue"
html_data = requests.get(url).text
```

Parse the html data using `beautiful_soup`.

In [17]:
```
beautiful_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

In [18]:
```
gme_revenue = pd.read_html(url)
gme_revenue = gme_revenue[1]
gme_revenue= gme_revenue.rename(columns={'GameStop Quarterly Revenue(Millions
 of US $)':'Date', 'GameStop Quarterly Revenue(Millions of US $).1':'Revenue'
})
gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")
gme_revenue.head()
```

```
/tmp/wsuser/ipykernel_206/2594000764.py:4: FutureWarning: The default value o
f regex will change from True to False in a future version.
  gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")
```

Out[18]:

	Date	Revenue
0	2021-10-31	1297
1	2021-07-31	1183
2	2021-04-30	1277
3	2021-01-31	2122
4	2020-10-31	1005

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

In [19]:
```
gme_revenue.tail()
```

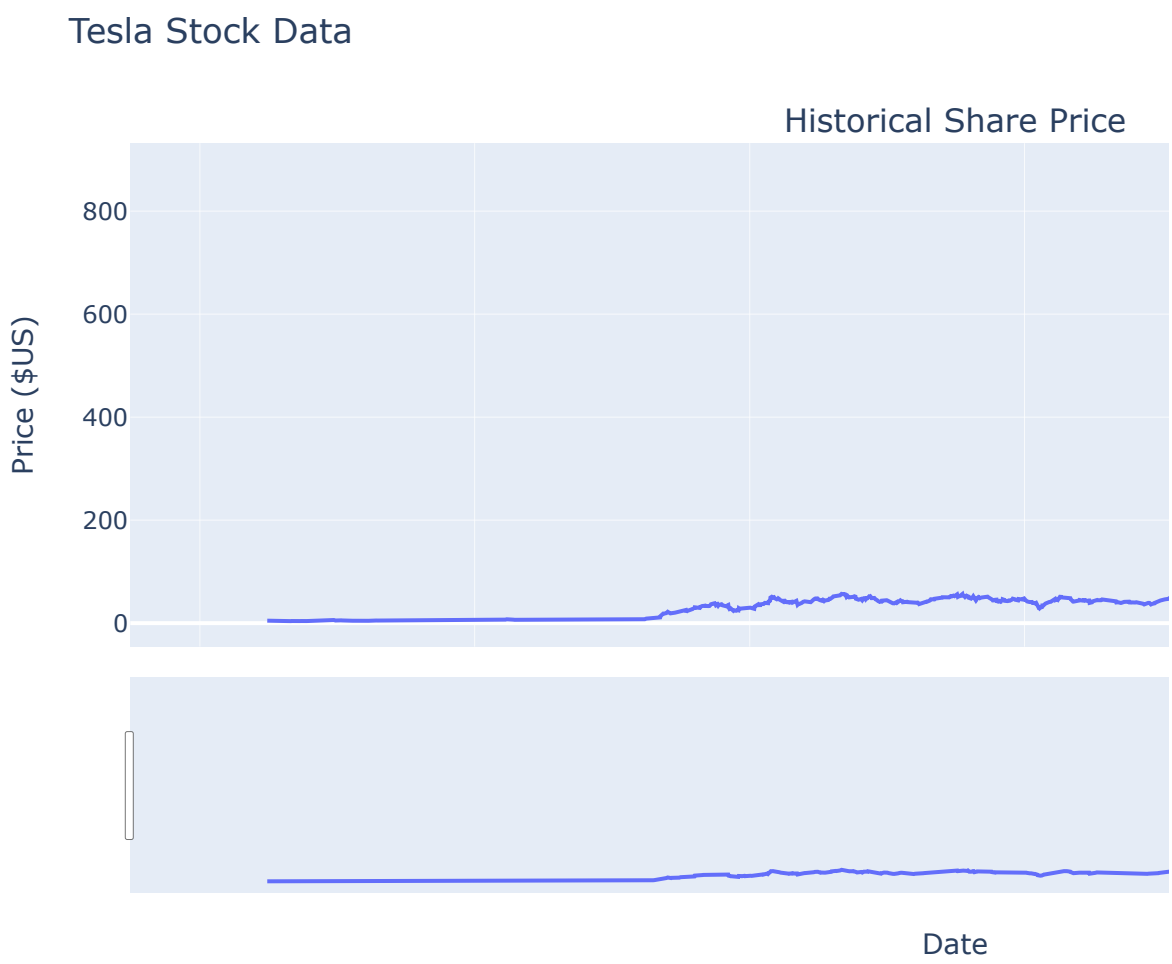
Out[19]:

	Date	Revenue
63	2006-01-31	1667
64	2005-10-31	534
65	2005-07-31	416
66	2005-04-30	475
67	2005-01-31	709

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.

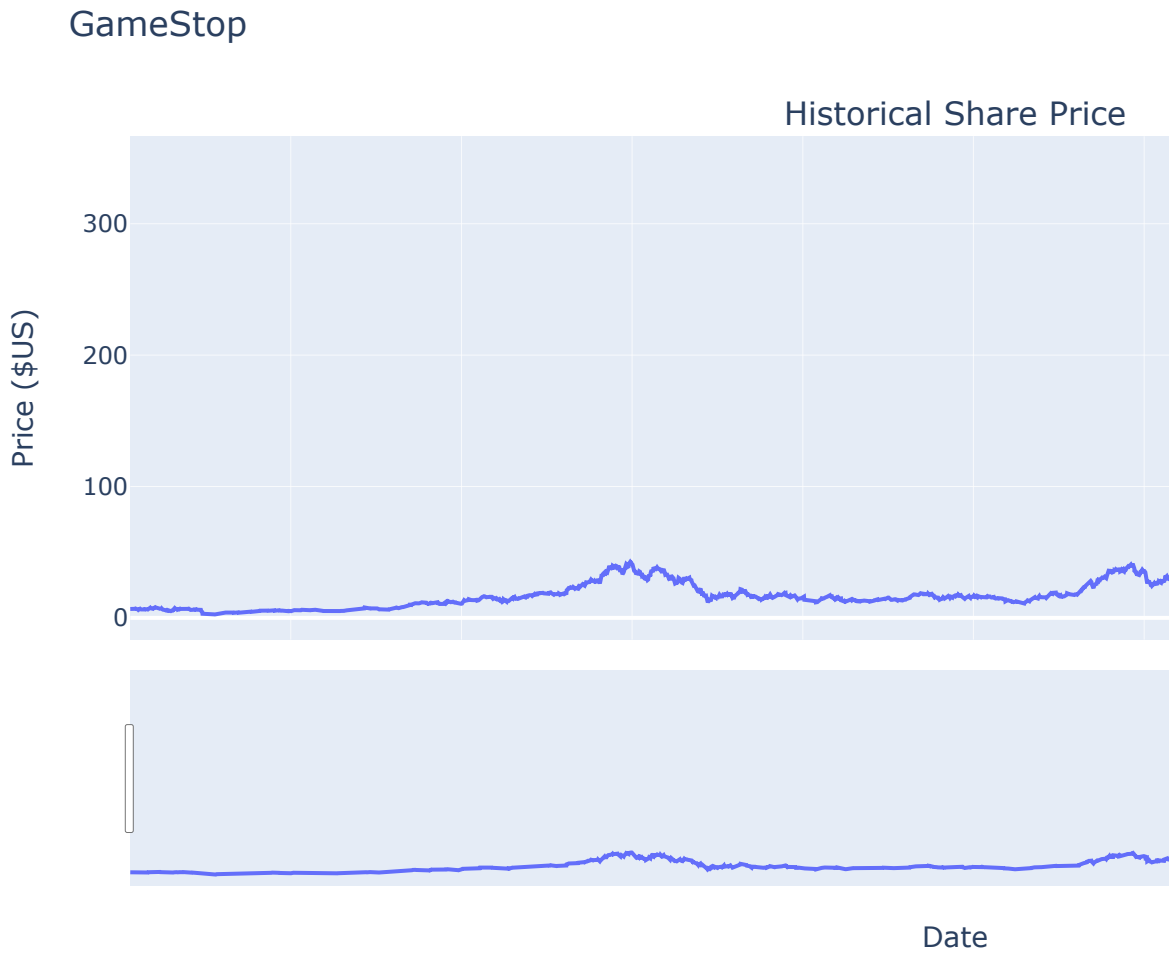
In [20]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla Stock Data')
```



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.

In [21]:
```
make_graph(gme_data, gme_revenue, 'GameStop')
```



About the Authors:

Joseph Santarcangelo (https://www.linkedin.com/in/joseph-s-50398b136/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0220ENSkillsNetwork23455606-2021-01-01) 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 Hiriani

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-11-10	1.1	Malika Singla	Deleted the Optional part
2020-08-27	1.0	Malika Singla	Added lab to GitLab