

# Final Assignment

March 29, 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

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

```
[4]: 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.

```
[5]: 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.

```
[16]: 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.

```
[17]: 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.

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

```
[18]:
```

	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`.

```
[19]: url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
↳ IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm"

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

Parse the html data using `beautiful_soup`.

```
[20]: soup = BeautifulSoup(html_data)
```

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

```
[21]: 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.

```
[22]: 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.

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

```
[23]: 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.

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

```
[24]:      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`.

```
[11]: 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.

```
[12]: 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.

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

```
[13]:      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	1.683250	1.687458	1.658001	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	1.603295	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`.

```
[6]: 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`.

```
[7]: soup = BeautifulSoup(html_data)
```

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

```
[8]: 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:10: FutureWarning: The default value of regex will change from True to False in a future version.

# Remove the CWD from sys.path while we load stuff.

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

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

```
[9]:
```

	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.

```
[25]: 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.

```
[15]: 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

##

© IBM Corporation 2020. All rights reserved.