

# 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

- Define a Function that Makes a Graph
- Question 1: Use yfinance to Extract Stock Data
- Question 2: Use Webscraping to Extract Tesla Revenue Data
- Question 3: Use yfinance to Extract Stock Data
- Question 4: Use Webscraping to Extract GME Revenue Data
- Question 5: Plot Tesla Stock Graph
- Question 6: Plot GameStop Stock Graph

Estimated Time Needed: **30 min**

**Note:-** If you are working Locally using anaconda, please uncomment the following code and execute it.

```
In [1]: #!pip install yfinance==0.2.38  
#!pip install pandas==2.2.2  
#!pip install nbformat
```

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

```
In [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
```

In Python, you can ignore warnings using the warnings module. You can use the filterwarnings function to filter or ignore specific warning messages or categories.

```
In [5]: import warnings
# Ignore all warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

## 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 [63]: def make_graph(stock_data, revenue_data, stock):
fig = make_subplots(rows=2, cols=1, shared_xaxes=True, subplot_titles=("Histori
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_datet
fig.add_trace(go.Scatter(x=pd.to_datetime(revenue_data_specific.Date, infer_dat
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 [6]: TSLA = 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.

```
In [ ]: tesla_data = TSLA.history(period = 'max')
tesla_data
```

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

```
Out[66]:
```

	level_0	index	Date	Open	High	Low	Close	Volume	Dividends	Str Sp
0	0	0	2010-06-29	1.266667	1.666667	1.169333	1.592667	281494500	0	
1	1	1	2010-06-30	1.719333	2.028000	1.553333	1.588667	257806500	0	
2	2	2	2010-07-01	1.666667	1.728000	1.351333	1.464000	123282000	0	
3	3	3	2010-07-02	1.533333	1.540000	1.247333	1.280000	77097000	0	
4	4	4	2010-07-06	1.333333	1.333333	1.055333	1.074000	103003500	0	

## 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` .

```
In [25]: url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevelo
r = requests.get(url)
html_data = r.text
```

Parse the html data using `beautiful_soup` .

```
In [ ]: html_content = BeautifulSoup(html_data, 'html.parser')
```

Using `BeautifulSoup` or the `read_html` function extract the table with `Tesla 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 [39]: table = html_content.find_all('tbody')[1]
dict_list = []
for row in table.find_all('tr'):
    col = row.find_all('td')
    date = col[0].text
    revenue = col[1].text
    dict1 = {'Date':date, 'Revenue':revenue}
    dict_list.append(dict1)

## make dataframe
tesla_revenue = pd.DataFrame(dict_list, columns = ['Date', 'Revenue'])
```

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

```
In [40]: tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$', "")
```

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

```
In [41]: tesla_revenue.dropna(inplace=True) ## null row

tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""] ## empty row
```

Display the last 5 row of the `tesla_revenue` dataframe using the `tail` function. Take a screenshot of the results.

```
In [42]: tesla_revenue.tail()
```

```
Out[42]:
```

	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

## 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 [44]: 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.

```
In [45]: 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.

```
In [46]: gme_data.reset_index(inplace=True)
gme_data.head()
```

```
Out[46]:
```

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13	1.620128	1.693350	1.603296	1.691667	76216000	0.0	0.0
1	2002-02-14	1.712707	1.716074	1.670626	1.683250	11021600	0.0	0.0
2	2002-02-15	1.683250	1.687458	1.658001	1.674834	8389600	0.0	0.0
3	2002-02-19	1.666417	1.666417	1.578047	1.607504	7410400	0.0	0.0
4	2002-02-20	1.615920	1.662210	1.603296	1.662210	6892800	0.0	0.0

## 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`.

```
In [48]: url2 = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevel
req = requests.get(url2)
html_data = req.text
```

Parse the html data using `beautiful_soup`.

```
In [74]: soup = BeautifulSoup(html_data, 'html.parser')
```

Using `BeautifulSoup` or the `read_html` function extract the table with `GameStop 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 [79]: table = soup.find_all('tbody')[1]
dict_list2 = []
for row in table.find_all('tr'):
    col = row.find_all('td')
    date = col[0].text
    revenue = col[1].text
    dict2 = {'Date':date, 'Revenue':revenue}
    dict_list2.append(dict2)

## make dataframe
gme_revenue = pd.DataFrame(dict_list2, columns = ['Date', 'Revenue'])
```

```
In [80]: gme_revenue['Revenue'] = gme_revenue['Revenue'].str.replace(',', '\$', '')
```

```
In [81]: gme_revenue = gme_revenue[gme_revenue['Revenue'] != ""]
```

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

```
In [82]: gme_revenue.tail()
```

```
Out[82]:
```

	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

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

**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 [83]: 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

## 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.

In [ ]: