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**

__Note__:- If you are working in IBM Cloud Watson Studio, please replace the command for installing nbformat from `!pip install nbformat==4.2.0` to simply `!pip install nbformat`

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

In [95]:
```
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 [96]:
```
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 [97]:
```
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.

In [98]:
```
tesla_data = tesla.history(period="max")

#tesla_data.head()
```

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 [99]:
```
tesla_data.reset_index(inplace=True)

tesla_data.head()
```

Out[99]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2010-06-29	1.266667	1.666667	1.169333	1.592667	281494500	0	0.0
1	2010-06-30	1.719333	2.028000	1.553333	1.588667	257806500	0	0.0
2	2010-07-01	1.666667	1.728000	1.351333	1.464000	123282000	0	0.0
3	2010-07-02	1.533333	1.540000	1.247333	1.280000	77097000	0	0.0
4	2010-07-06	1.333333	1.333333	1.055333	1.074000	103003500	0	0.0

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 [ ]:
```
url = " https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevel

html_data  = requests.get(url).text

print(html_data)
```

Parse the html data using `beautiful_soup`.

In [147…
```
soup = BeautifulSoup(html_data)
```

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 [148…
```
tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])

#tesla_revenue.head()

# First we isolate the body of the table which contains all the information
# Then we loop through each row and find all the column values for each row
for row in soup.find("tbody").find_all("tr"):
    col = row.find_all("td")
```

```
    date = col[0].text
    revenue = col[1].text

    # Finally we append the data of each row to the table
    tesla_revenue = tesla_revenue.append({"Date":date, "Revenue":revenue}, ignore_i
```

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

In [149…
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages/ipykernel_launcher.p
y: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.

In [150…
```
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 [151…
```
tesla_revenue.tail()
```

Out[151]:

	Date	Revenue
8	2013	2013
9	2012	413
10	2011	204
11	2010	117
12	2009	112

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 [106…
```
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 [107…
```
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 [108…
```
gme_data.reset_index(inplace=True)

gme_data.head()
```

Out[108]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13	1.620129	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	1.666418	1.666418	1.578048	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 [ ]:
```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevelo

html_data  = requests.get(url).text

print(html_data)
```

Parse the html data using `beautiful_soup`.

In [143…
```
soup = BeautifulSoup(html_data)
```

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 [144…
```
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

gme_revenue.head()
```

Out[144]:

	Date	Revenue
0	2020	$6,466
1	2019	$8,285
2	2018	$8,547
3	2017	$7,965
4	2016	$9,364

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

In [155…
```
gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")

gme_revenue.dropna(inplace=True)

gme_revenue = gme_revenue[gme_revenue['Revenue'] != ""]

gme_revenue.tail()
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages/ipykernel_launcher.p
y:1: FutureWarning:

The default value of regex will change from True to False in a future version.
```

Out[155]:

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

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 [152…

```
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 [156…

```
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
