Final Assignment

March 20, 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

    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

[2]:
```
!pip install yfinance
#!pip install pandas
#!pip install requests
!pip install bs4
#!pip install plotly
```

```
Requirement already satisfied: yfinance in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (0.1.67)
Requirement already satisfied: pandas>=0.24 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(1.3.5)
Requirement already satisfied: requests>=2.20 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(2.28.1)
Requirement already satisfied: lxml>=4.5.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(4.6.4)
Requirement already satisfied: multitasking>=0.0.7 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
```

```
(0.0.11)
Requirement already satisfied: numpy>=1.15 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(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) (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) (2022.6)
Requirement already satisfied: charset-normalizer<3,>=2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance) (2.1.1)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance) (2022.12.7)
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) (1.26.13)
Requirement already satisfied: idna<4,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance) (3.4)
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) (1.16.0)
Requirement already satisfied: bs4 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from bs4)
(4.10.0)
Requirement already satisfied: soupsieve>1.2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
beautifulsoup4->bs4) (2.3.2.post1)
```

[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.

[4]:
```
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)
    fig.add_trace(go.Scatter(x=pd.to_datetime(stock_data.Date,␣
 ↪infer_datetime_format=True), y=stock_data.Close.astype("float"), name="Share␣
 ↪Price"), row=1, col=1)
    fig.add_trace(go.Scatter(x=pd.to_datetime(revenue_data.Date,␣
 ↪infer_datetime_format=True), y=revenue_data.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`.

[5]:
```
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.

[6]:
```
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.

[7]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

[7]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
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	

```
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`.

[8]:
```
url = 'https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue'
html_data = requests.get(url).text
```

Parse the html data using `beautiful_soup`.

[9]:
```
soup = BeautifulSoup(html_data,"html5lib")
```

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
```

[10]:
```
tesla_revenue = pd.DataFrame(columns=['Date', 'Revenue'])

for table in soup.find_all('table'):

    if ('Tesla Quarterly Revenue' in table.find('th').text):
        rows = table.find_all('tr')

        for row in rows:
            col = row.find_all('td')

            if col != []:
                date = col[0].text
                revenue = col[1].text.replace(',','').replace('$','')
```

```
                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.

[12]:
```
tesla_revenue
```

[12]:

	Date	Revenue
0	2022-12-31	24318
1	2022-09-30	21454
2	2022-06-30	16934
3	2022-03-31	18756
4	2021-12-31	17719
5	2021-09-30	13757
6	2021-06-30	11958
7	2021-03-31	10389
8	2020-12-31	10744
9	2020-09-30	8771
10	2020-06-30	6036
11	2020-03-31	5985
12	2019-12-31	7384
13	2019-09-30	6303
14	2019-06-30	6350
15	2019-03-31	4541
16	2018-12-31	7226
17	2018-09-30	6824
18	2018-06-30	4002
19	2018-03-31	3409
20	2017-12-31	3288
21	2017-09-30	2985
22	2017-06-30	2790
23	2017-03-31	2696
24	2016-12-31	2285
25	2016-09-30	2298
26	2016-06-30	1270
27	2016-03-31	1147
28	2015-12-31	1214
29	2015-09-30	937
30	2015-06-30	955
31	2015-03-31	940
32	2014-12-31	957
33	2014-09-30	852
34	2014-06-30	769
35	2014-03-31	621
36	2013-12-31	615
37	2013-09-30	431
38	2013-06-30	405

39	2013-03-31	562
40	2012-12-31	306
41	2012-09-30	50
42	2012-06-30	27
43	2012-03-31	30
44	2011-12-31	39
45	2011-09-30	58
46	2011-06-30	58
47	2011-03-31	49
48	2010-12-31	36
49	2010-09-30	31
50	2010-06-30	28
51	2010-03-31	21
52	2009-12-31	
53	2009-09-30	46
54	2009-06-30	27

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

[13]:
```
tesla_revenue = tesla_revenue[tesla_revenue['Revenue'].astype(bool)]
```

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

[14]:
```
tesla_revenue.tail()
```

[14]:

	Date	Revenue
49	2010-09-30	31
50	2010-06-30	28
51	2010-03-31	21
53	2009-09-30	46
54	2009-06-30	27

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`.

[15]:
```
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.

[16]:
```
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.

[17]:
```
gme_data.reset_index(inplace=True)
gme_data.head()
```

[17]:

	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	1.658002	1.674834	8389600	0.0	0.0
3	2002-02-19	1.666418	1.666418	1.578047	1.607504	7410400	0.0	0.0
4	2002-02-20	1.615920	1.662209	1.603296	1.662209	6892800	0.0	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`.

[18]:
```
url = 'https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue'
html_data = requests.get(url).text
```

Parse the html data using `beautiful_soup`.

[19]:
```
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

```
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
```

[20]:
```
gme_revenue = pd.DataFrame(columns=['Date', 'Revenue'])

for table in soup.find_all('table'):
```

```
    if ('GameStop Quarterly Revenue' in table.find('th').text):
        rows = table.find_all('tr')

        for row in rows:
            col = row.find_all('td')

            if col != []:
                date = col[0].text
                revenue = col[1].text.replace(',','').replace('$','')

                gme_revenue = gme_revenue.append({"Date":date, "Revenue":
↪revenue}, ignore_index=True)
```

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

[21]:
```
gme_revenue.tail()
```

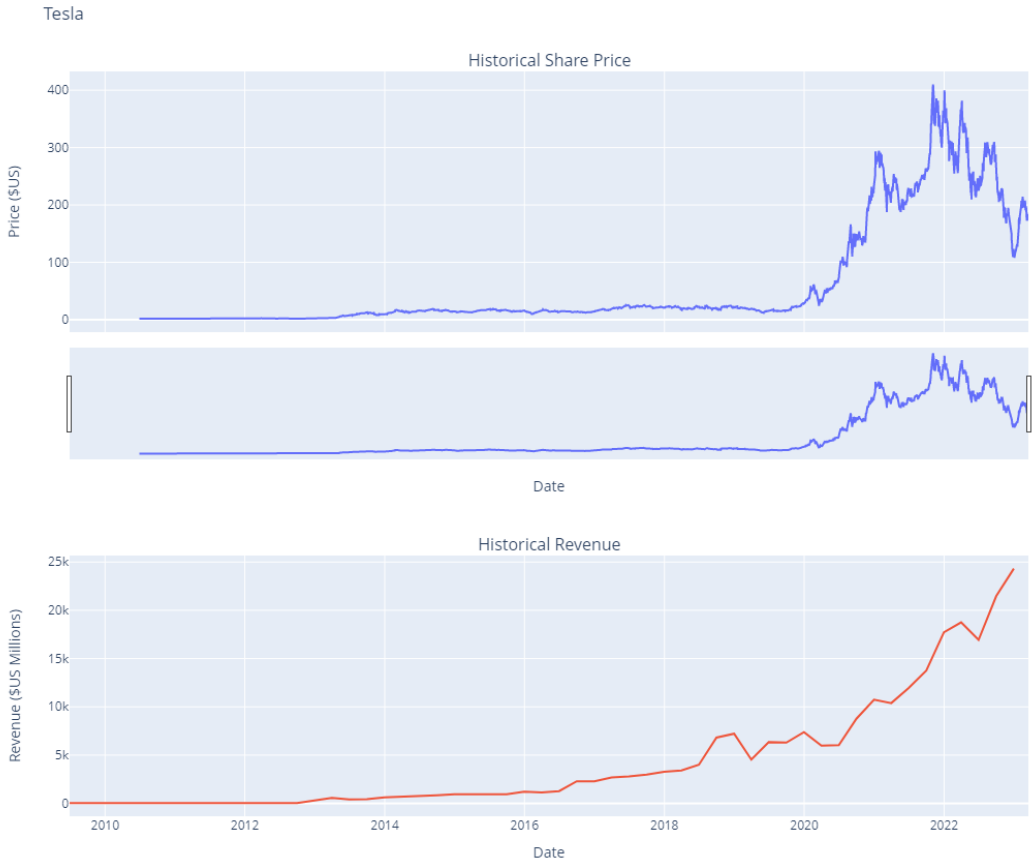
[21]:

	Date	Revenue
51	2010-01-31	3524
52	2009-10-31	1835
53	2009-07-31	1739
54	2009-04-30	1981
55	2009-01-31	3492

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.

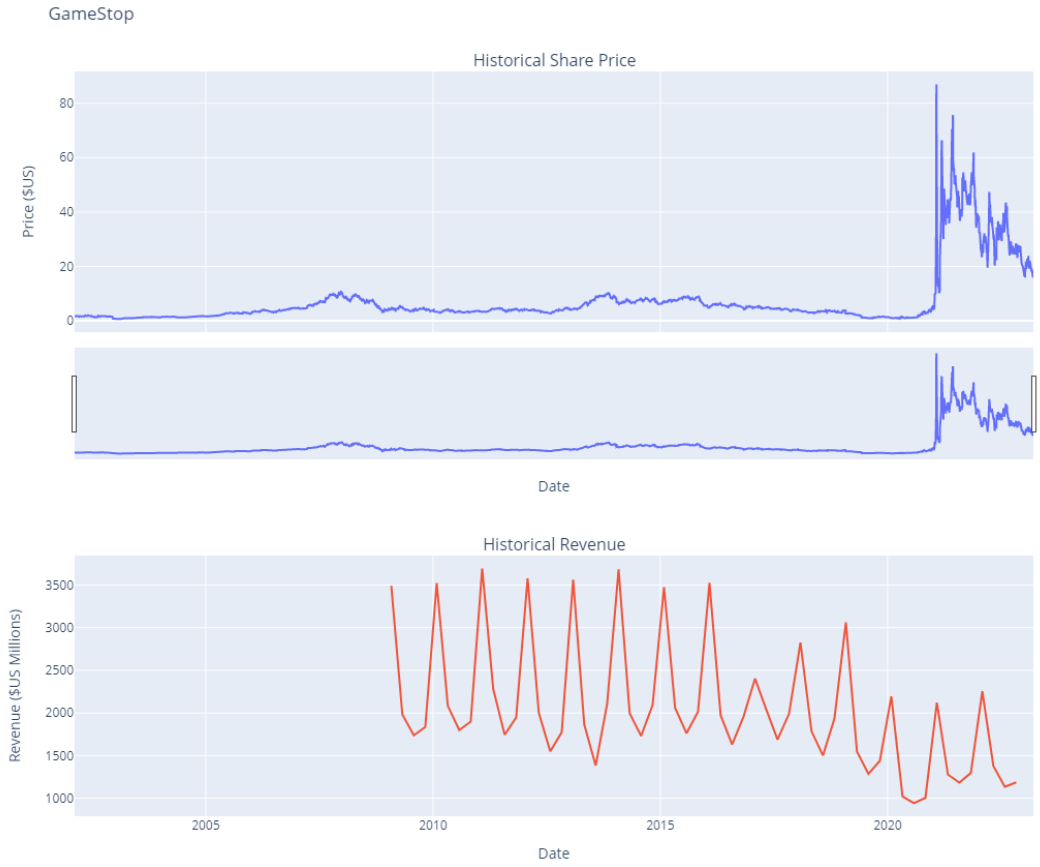
[22]:
```
make_graph(tesla_data[['Date','Close']], tesla_revenue, 'Tesla')
```



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.

[23]:
```
make_graph(gme_data[['Date','Close']], 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

##