Final Assignment

June 4, 2022

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

```
<ul>
    <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>
</ul>
```

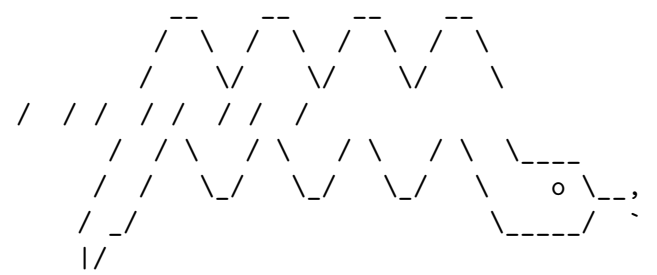
Estimated Time Needed: 30 min

[3]:
```
!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
!pip install html5lib
```

```
Requirement already satisfied: yfinance==0.1.67 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==0.1.67) (1.3.3)
Requirement already satisfied: requests>=2.20 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (2.26.0)
Requirement already satisfied: lxml>=4.5.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (4.8.0)
Requirement already satisfied: multitasking>=0.0.7 in
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (0.0.10)
Requirement already satisfied: numpy>=1.15 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (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==0.1.67) (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==0.1.67) (2022.1)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (2022.5.18.1)
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==0.1.67) (1.26.9)
Requirement already satisfied: idna<4,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (2.0.12)
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==0.1.67) (1.16.0)
Requirement already satisfied: pandas==1.3.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (1.3.3)
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.3) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.3) (2022.1)
Requirement already satisfied: numpy>=1.17.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.3) (1.21.6)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from python-
dateutil>=2.7.3->pandas==1.3.3) (1.16.0)
Requirement already satisfied: requests==2.26.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (2.26.0)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (2022.5.18.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (1.26.9)
```

```
Requirement already satisfied: idna<4,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests==2.26.0) (2.0.12)
```



```
        mamba (0.15.3) supported by @QuantStack

        GitHub:  https://github.com/mamba-org/mamba
        Twitter: https://twitter.com/QuantStack


Looking for: ['bs4==4.10.0']

pkgs/main/linux-64       [>                   ] (--:--) No change
pkgs/main/linux-64       [====================] (00m:00s) No change
pkgs/r/linux-64          [>                   ] (--:--) No change
pkgs/r/linux-64          [====================] (00m:00s) No change
pkgs/main/noarch         [>                   ] (--:--) No change
pkgs/main/noarch         [====================] (00m:00s) No change
pkgs/r/noarch            [>                   ] (--:--) No change
pkgs/r/noarch            [====================] (00m:00s) No change

Pinned packages:
  - python 3.7.*


Transaction

  Prefix: /home/jupyterlab/conda/envs/python
```

```
  All requested packages already installed

Requirement already satisfied: plotly==5.3.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (5.3.1)
Requirement already satisfied: six in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
plotly==5.3.1) (1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
plotly==5.3.1) (8.0.1)
Requirement already satisfied: html5lib in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (1.1)
Requirement already satisfied: webencodings in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from html5lib)
(0.5.1)
Requirement already satisfied: six>=1.9 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from html5lib)
(1.16.0)
```

[54]:
```
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
import html5lib
```

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.

[55]:
```
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_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`.

[101]:
```
tesla = yf.Ticker("TSLA") #copyright -- Mahmoud El Ahmad Matar
```

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.

[102]:
```
tesla_data = tesla.history(period="max") #copyright -- Mahmoud El Ahmad Matar
```

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.

[107]:
```
tesla_data.reset_index(inplace=True) #copyright -- Mahmoud El Ahmad Matar
tesla_data.head()
```

[107]:

	index	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	0	2010-06-29	3.800	5.000	3.508	4.778	93831500	0	0.0
1	1	2010-06-30	5.158	6.084	4.660	4.766	85935500	0	0.0
2	2	2010-07-01	5.000	5.184	4.054	4.392	41094000	0	0.0
3	3	2010-07-02	4.600	4.620	3.742	3.840	25699000	0	0.0
4	4	2010-07-06	4.000	4.000	3.166	3.222	34334500	0	0.0

[108]:
```
## 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/rever
Save the text of the response as a variable named **`html_data`**.

[109]:
```
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue?
↪utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_
html_data = requests.get(url).text #copyright -- Mahmoud El Ahmad Matar
```

Parse the html data using `beautiful_soup`.

[110]:
```
soup = BeautifulSoup(html_data,"html.parser") #copyright -- Mahmoud El Ahmad␣
↪Matar
```

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`.

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

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
```

[111]:
```
tesla_revenue = pd.read_html(url, match="Tesla Quarterly Revenue",␣
↪flavor='bs4')[0]
tesla_revenue.columns = ['Date','Revenue'] #copyright -- Mahmoud El Ahmad Matar
```

[112]:
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")␣
↪#copyright -- Mahmoud El Ahmad Matar
```

```
/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.

[113]:
```
tesla_revenue.dropna(inplace=True)

tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""] #copyright --␣
↪Mahmoud El Ahmad Matar
```

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

[114]:
```
tesla_revenue.tail() #copyright -- Mahmoud El Ahmad Matar
```

[114]:

	Date	Revenue
46	2010-09-30	31
47	2010-06-30	28
48	2010-03-31	21
50	2009-09-30	46
51	2009-06-30	27

0.3 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`.

[115]:
```
gme = yf.Ticker("GME") #copyright -- Mahmoud El Ahmad Matar
```

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.

[116]:
```
gme_data = gme.history(period="max") #copyright -- Mahmoud El Ahmad Matar
```

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.

[117]:
```
gme_data.reset_index(inplace=True)
gme_data.head() #copyright -- Mahmoud El Ahmad Matar
```

[117]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13	6.480514	6.773400	6.413183	6.766666	19054000	0.0	0.0
1	2002-02-14	6.850830	6.864296	6.682505	6.733003	2755400	0.0	0.0
2	2002-02-15	6.733001	6.749833	6.632006	6.699336	2097400	0.0	0.0
3	2002-02-19	6.665672	6.665672	6.312189	6.430017	1852600	0.0	0.0
4	2002-02-20	6.463680	6.648838	6.413182	6.648838	1723200	0.0	0.0

0.4 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`.

[118]:
```
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 #copyright -- Mahmoud El Ahmad Matar
```

Parse the html data using `beautiful_soup`.

[119]:
```
soup = BeautifulSoup(html_data,'html5lib') #copyright -- Mahmoud El Ahmad Matar
```

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
```

[120]:
```
gme_revenue = pd.read_html(url,match="GameStop Quarterly␣
↪Revenue",flavor="bs4")[0]
gme_revenue.columns = ['Date','Revenue']
gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")␣
↪#copyright -- Mahmoud El Ahmad Matar
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/ipykernel_launcher.py:3: FutureWarning:

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

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

[121]:
```
gme_revenue.tail() #copyright -- Mahmoud El Ahmad Matar
```

[121]:

	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

0.5 Question 5: Plot Tesla Stock Graph

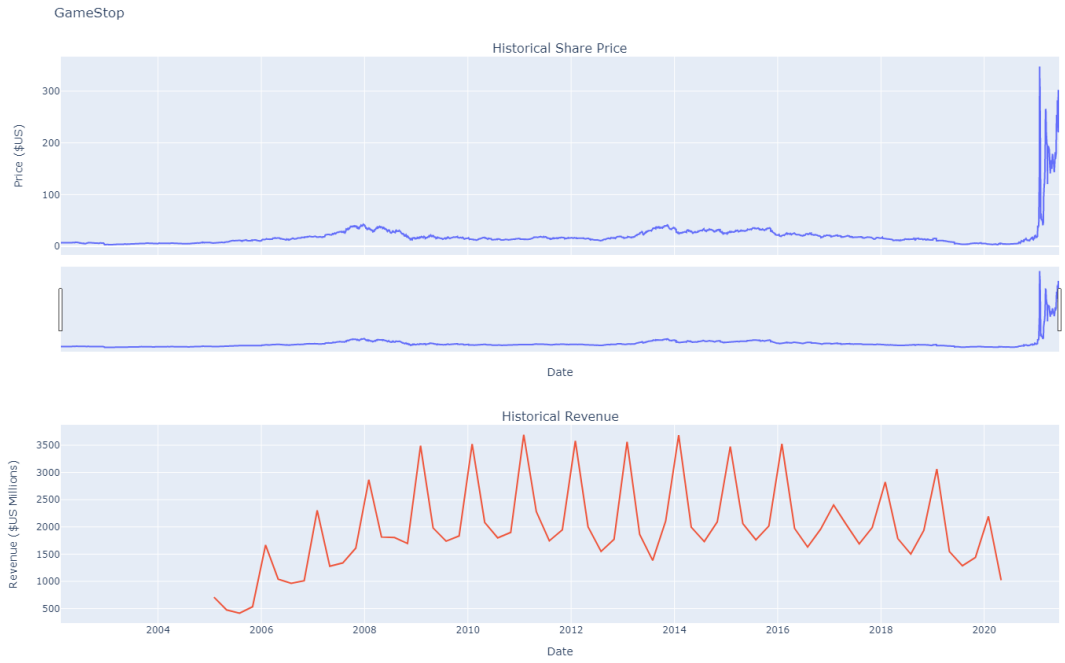
Use the `make_graph` function to graph the Tesla Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(tesla_data, tesla_revenue, 'Tesla')`. Note the graph will only show data upto June 2021.

[122]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla') #copyright -- Mahmoud El Ahmad␣
↪Matar
```

0.6 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.

[123]:
```
make_graph(gme_data, gme_revenue, 'GameStop') #copyright -- Mahmoud El Ahmad␣
↪Matar
```



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

0.7 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

##

[ ]: