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 Locally using anaconda, please uncomment the following code and execute it. Use the version as per your python version.

In [1]:
```
!pip install yfinance
!pip install bs4
!pip install nbformat
!pip install --upgrade plotly
```

```
Collecting yfinance
  Downloading yfinance-0.2.64-py2.py3-none-any.whl.metadata (5.8 kB)
Requirement already satisfied: pandas>=1.3.0 in /opt/conda/lib/python3.12/site-packa
ges (from yfinance) (2.3.0)
Requirement already satisfied: numpy>=1.16.5 in /opt/conda/lib/python3.12/site-packa
ges (from yfinance) (2.3.1)
Requirement already satisfied: requests>=2.31 in /opt/conda/lib/python3.12/site-pack
ages (from yfinance) (2.32.3)
Collecting multitasking>=0.0.7 (from yfinance)
  Downloading multitasking-0.0.11-py3-none-any.whl.metadata (5.5 kB)
Requirement already satisfied: platformdirs>=2.0.0 in /opt/conda/lib/python3.12/site
-packages (from yfinance) (4.3.6)
Requirement already satisfied: pytz>=2022.5 in /opt/conda/lib/python3.12/site-packag
es (from yfinance) (2024.2)
Requirement already satisfied: frozendict>=2.3.4 in /opt/conda/lib/python3.12/site-p
ackages (from yfinance) (2.4.6)
Collecting peewee>=3.16.2 (from yfinance)
  Downloading peewee-3.18.1.tar.gz (3.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.0/3.0 MB 89.2 MB/s eta 0:00:00
  Installing build dependencies ... one
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: beautifulsoup4>=4.11.1 in /opt/conda/lib/python3.12/s
ite-packages (from yfinance) (4.12.3)
Collecting curl_cffi>=0.7 (from yfinance)
  Downloading curl_cffi-0.11.4-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.
whl.metadata (14 kB)
Collecting protobuf>=3.19.0 (from yfinance)
  Downloading protobuf-6.31.1-cp39-abi3-manylinux2014_x86_64.whl.metadata (593 byte
s)
Collecting websockets>=13.0 (from yfinance)
  Downloading websockets-15.0.1-cp312-cp312-manylinux_2_5_x86_64.manylinux1_x86_64.m
anylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (6.8 kB)
Requirement already satisfied: soupsieve>1.2 in /opt/conda/lib/python3.12/site-packa
ges (from beautifulsoup4>=4.11.1->yfinance) (2.5)
Requirement already satisfied: cffi>=1.12.0 in /opt/conda/lib/python3.12/site-packag
es (from curl_cffi>=0.7->yfinance) (1.17.1)
Requirement already satisfied: certifi>=2024.2.2 in /opt/conda/lib/python3.12/site-p
ackages (from curl_cffi>=0.7->yfinance) (2024.12.14)
Requirement already satisfied: python-dateutil>=2.8.2 in /opt/conda/lib/python3.12/s
ite-packages (from pandas>=1.3.0->yfinance) (2.9.0.post0)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.12/site-pack
ages (from pandas>=1.3.0->yfinance) (2025.2)
Requirement already satisfied: charset_normalizer<4,>=2 in /opt/conda/lib/python3.1
2/site-packages (from requests>=2.31->yfinance) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.12/site-packag
es (from requests>=2.31->yfinance) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python3.12/site-
packages (from requests>=2.31->yfinance) (2.3.0)
Requirement already satisfied: pycparser in /opt/conda/lib/python3.12/site-packages
(from cffi>=1.12.0->curl_cffi>=0.7->yfinance) (2.22)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-packages
(from python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.17.0)
Downloading yfinance-0.2.64-py2.py3-none-any.whl (119 kB)
Downloading curl_cffi-0.11.4-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.wh
l (8.5 MB)
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.5/8.5 MB 148.4 MB/s eta 0:00:00
Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Downloading protobuf-6.31.1-cp39-abi3-manylinux2014_x86_64.whl (321 kB)
Downloading websockets-15.0.1-cp312-cp312-manylinux_2_5_x86_64.manylinux1_x86_64.man
ylinux_2_17_x86_64.manylinux2014_x86_64.whl (182 kB)
Building wheels for collected packages: peewee
  Building wheel for peewee (pyproject.toml) ... one
  Created wheel for peewee: filename=peewee-3.18.1-cp312-cp312-linux_x86_64.whl size
=303801 sha256=399539131df3bc7eeffd34472bb61dcb25b8ee4e04bd44aa049ce002c408789a
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/1a/57/6a/bb71346381d0d911c
d4ce3026f1fa720da76707e4f01cf27dd
Successfully built peewee
Installing collected packages: peewee, multitasking, websockets, protobuf, curl_cff
i, yfinance
Successfully installed curl_cffi-0.11.4 multitasking-0.0.11 peewee-3.18.1 protobuf-
6.31.1 websockets-15.0.1 yfinance-0.2.64
Collecting bs4
  Downloading bs4-0.0.2-py2.py3-none-any.whl.metadata (411 bytes)
Requirement already satisfied: beautifulsoup4 in /opt/conda/lib/python3.12/site-pack
ages (from bs4) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in /opt/conda/lib/python3.12/site-packa
ges (from beautifulsoup4->bs4) (2.5)
Downloading bs4-0.0.2-py2.py3-none-any.whl (1.2 kB)
Installing collected packages: bs4
Successfully installed bs4-0.0.2
Requirement already satisfied: nbformat in /opt/conda/lib/python3.12/site-packages
(5.10.4)
Requirement already satisfied: fastjsonschema>=2.15 in /opt/conda/lib/python3.12/sit
e-packages (from nbformat) (2.21.1)
Requirement already satisfied: jsonschema>=2.6 in /opt/conda/lib/python3.12/site-pac
kages (from nbformat) (4.23.0)
Requirement already satisfied: jupyter-core!=5.0.*,>=4.12 in /opt/conda/lib/python3.
12/site-packages (from nbformat) (5.7.2)
Requirement already satisfied: traitlets>=5.1 in /opt/conda/lib/python3.12/site-pack
ages (from nbformat) (5.14.3)
Requirement already satisfied: attrs>=22.2.0 in /opt/conda/lib/python3.12/site-packa
ges (from jsonschema>=2.6->nbformat) (25.1.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /opt/conda/li
b/python3.12/site-packages (from jsonschema>=2.6->nbformat) (2024.10.1)
Requirement already satisfied: referencing>=0.28.4 in /opt/conda/lib/python3.12/site
-packages (from jsonschema>=2.6->nbformat) (0.36.2)
Requirement already satisfied: rpds-py>=0.7.1 in /opt/conda/lib/python3.12/site-pack
ages (from jsonschema>=2.6->nbformat) (0.22.3)
Requirement already satisfied: platformdirs>=2.5 in /opt/conda/lib/python3.12/site-p
ackages (from jupyter-core!=5.0.*,>=4.12->nbformat) (4.3.6)
Requirement already satisfied: typing-extensions>=4.4.0 in /opt/conda/lib/python3.1
2/site-packages (from referencing>=0.28.4->jsonschema>=2.6->nbformat) (4.12.2)
Requirement already satisfied: plotly in /opt/conda/lib/python3.12/site-packages (5.
24.1)
Collecting plotly
  Downloading plotly-6.2.0-py3-none-any.whl.metadata (8.5 kB)
Collecting narwhals>=1.15.1 (from plotly)
  Downloading narwhals-1.45.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: packaging in /opt/conda/lib/python3.12/site-packages
(from plotly) (24.2)
Downloading plotly-6.2.0-py3-none-any.whl (9.6 MB)
```

```
                                        9.6/9.6 MB 132.9 MB/s eta 0:00:00
Downloading narwhals-1.45.0-py3-none-any.whl (371 kB)
Installing collected packages: narwhals, plotly
  Attempting uninstall: plotly
    Found existing installation: plotly 5.24.1
    Uninstalling plotly-5.24.1:
      Successfully uninstalled plotly-5.24.1
Successfully installed narwhals-1.45.0 plotly-6.2.0
```

In [2]:
```
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 [3]:
```
import plotly.io as pio
pio.renderers.default = "iframe"
```

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 [4]:
```
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 [52]:
```
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()
    from IPython.display import display, HTML
    fig_html = fig.to_html()
    display(HTML(fig_html))
```

Use the make_graph function that we've already defined. You'll need to invoke it in questions 5 and 6 to display the graphs and create the dashboard.

> **Note: You don't need to redefine the function for plotting graphs anywhere else in this notebook; just use the existing function.**

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 [53]:
```
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 [54]:
```
tesla_data=tsla.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.

In [55]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

Out[55]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2010-06-29 00:00:00-04:00	1.266667	1.666667	1.169333	1.592667	281494500	0.0	0.0
1	2010-06-30 00:00:00-04:00	1.719333	2.028000	1.553333	1.588667	257806500	0.0	0.0
2	2010-07-01 00:00:00-04:00	1.666667	1.728000	1.351333	1.464000	123282000	0.0	0.0
3	2010-07-02 00:00:00-04:00	1.533333	1.540000	1.247333	1.280000	77097000	0.0	0.0
4	2010-07-06 00:00:00-04:00	1.333333	1.333333	1.055333	1.074000	103003500	0.0	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 [56]:
```
import pandas as pd
import requests
from bs4 import BeautifulSoup

import warnings
# Ignore all warnings
warnings.filterwarnings("ignore", category=FutureWarning)

url=" https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevelop

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

Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`.

In [57]:
```
soup=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`.

▶ Step-by-step instructions

▶ Click here if you need help locating the table

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 = pd.concat([tesla_dataframe,pd.DataFrame({"Date":[date], "Revenue": [revenue]})]) # Finally we append the data of each row to the table

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

In [58]:
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"",regex=Tru
```

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

In [37]:
```
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 [59]:
```
tesla_revenue.tail()
```

Out[59]:

	Date	Revenue
0	2012	413
0	2011	204
0	2010	117
0	2009	112
0	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 [39]:
```
import yfinance as yf
import pandas as pd
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 [40]:
```
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 [41]:
```
gme_data.reset_index(inplace=True)
gme_data.head()
```

Out[41]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13 00:00:00-05:00	1.620128	1.693349	1.603295	1.691666	76216000	0.0	0.0
1	2002-02-14 00:00:00-05:00	1.712707	1.716074	1.670626	1.683251	11021600	0.0	0.0
2	2002-02-15 00:00:00-05:00	1.683250	1.687458	1.658002	1.674834	8389600	0.0	0.0
3	2002-02-19 00:00:00-05:00	1.666418	1.666418	1.578047	1.607504	7410400	0.0	0.0
4	2002-02-20 00:00:00-05:00	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_2`.

In [42]:
```
import pandas as pd
import requests
from bs4 import BeautifulSoup
warnings.filterwarnings("ignore",category=FutureWarning)
url=" https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevelop
html_data_2=requests.get(url).text
```

Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`.

In [43]:
```
soup=BeautifulSoup(html_data_2,'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.

> **Note: Use the method similar to what you did in question 2.**

▶ Click here if you need help locating the table

In [44]:
```
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 = pd.concat([gme_revenue,pd.DataFrame({"Date":[date], "Revenue":[re
gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"",regex=True)
```

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

In [45]:
```
gme_revenue.tail()
```

Out[45]:

	Date	Revenue
0	2009	8806
0	2008	7094
0	2007	5319
0	2006	3092
0	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. Note the graph will only show data upto June 2021.

▶ Hint

In [65]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla')
```

```
/tmp/ipykernel_1440/109047474.py:5: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future v
ersion. A strict version of it is now the default, see https://pandas.pydata.org/pde
ps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.

/tmp/ipykernel_1440/109047474.py:6: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future v
ersion. A strict version of it is now the default, see https://pandas.pydata.org/pde
ps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.
```

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.

▶ Hint

In [61]:
```
make_graph(gme_data, gme_revenue, 'GameStop')
```

```
/tmp/ipykernel_1440/109047474.py:5: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future v
ersion. A strict version of it is now the default, see https://pandas.pydata.org/pde
ps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.

/tmp/ipykernel_1440/109047474.py:6: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future v
ersion. A strict version of it is now the default, see https://pandas.pydata.org/pde
ps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.
```

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