Project final

June 1, 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

[1]:
```
!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
```

```
Collecting yfinance==0.1.67
  Downloading yfinance-0.1.67-py2.py3-none-any.whl (25 kB)
Collecting multitasking>=0.0.7
  Downloading multitasking-0.0.10.tar.gz (8.2 kB)
Requirement already satisfied: numpy>=1.15 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from yfinance==0.1.67)
(1.20.3)
Requirement already satisfied: requests>=2.20 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from yfinance==0.1.67)
(2.26.0)
Requirement already satisfied: pandas>=0.24 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from yfinance==0.1.67)
(1.3.4)
```

```
Requirement already satisfied: lxml>=4.5.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from yfinance==0.1.67)
(4.7.1)
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
pandas>=0.24->yfinance==0.1.67) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
pandas>=0.24->yfinance==0.1.67) (2021.3)
Requirement already satisfied: six>=1.5 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-
dateutil>=2.7.3->pandas>=0.24->yfinance==0.1.67) (1.15.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests>=2.20->yfinance==0.1.67) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests>=2.20->yfinance==0.1.67) (2021.10.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests>=2.20->yfinance==0.1.67) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests>=2.20->yfinance==0.1.67) (3.3)
Building wheels for collected packages: multitasking
  Building wheel for multitasking (setup.py) … done
  Created wheel for multitasking: filename=multitasking-0.0.10-py3-none-
any.whl size=8500
sha256=69ac65d3c31e6c88d90df1195310b71b1161d91e3244e401093ce382e1314854
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/f2/b5/2c/59ba95dcf854e54294
4c75fe3da584e4e3833b319735a0546c
Successfully built multitasking
Installing collected packages: multitasking, yfinance
Successfully installed multitasking-0.0.10 yfinance-0.1.67
Collecting pandas==1.3.3
  Downloading
pandas-1.3.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.5 MB)
     |                  | 11.5 MB 21.2 MB/s eta 0:00:01
Requirement already satisfied: pytz>=2017.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas==1.3.3)
(2021.3)
Requirement already satisfied: numpy>=1.17.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas==1.3.3)
(1.20.3)
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas==1.3.3)
(2.8.2)
Requirement already satisfied: six>=1.5 in
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-
dateutil>=2.7.3->pandas==1.3.3) (1.15.0)
Installing collected packages: pandas
  Attempting uninstall: pandas
    Found existing installation: pandas 1.3.4
    Uninstalling pandas-1.3.4:
      Successfully uninstalled pandas-1.3.4
Successfully installed pandas-1.3.3
Requirement already satisfied: requests==2.26.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (2.26.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests==2.26.0)
(2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests==2.26.0)
(1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests==2.26.0)
(2021.10.8)
Requirement already satisfied: idna<4,>=2.5 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests==2.26.0)
(3.3)
/usr/bin/sh: mamba: command not found
Collecting plotly==5.3.1
  Downloading plotly-5.3.1-py2.py3-none-any.whl (23.9 MB)
     |                  | 23.9 MB 12.6 MB/s eta 0:00:01
Requirement already satisfied: tenacity>=6.2.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from plotly==5.3.1)
(8.0.1)
Requirement already satisfied: six in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from plotly==5.3.1)
(1.15.0)
Installing collected packages: plotly
  Attempting uninstall: plotly
    Found existing installation: plotly 5.1.0
    Uninstalling plotly-5.1.0:
      Successfully uninstalled plotly-5.1.0
Successfully installed plotly-5.3.1
```

[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
```

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.

[45]:
```
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`.

[3]:
```
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.

[4]:
```
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.

[5]:
```
tesla_data.reset_index(inplace=True)
print(tesla_data.head())
```

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

0.3 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`.

[6]:
```
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"
html_data  = requests.get(url).text
```

Parse the html data using `beautiful_soup`.

[7]:
```
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
```

[29]:
```
tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])

# 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_all("tbody")[1].find_all('tr'):
    col = row.find_all("td")
#     print(col)
    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_index=True)
```

[31]:
```
tesla_revenue.head()
```

[31]:

	Date	Revenue
0	2022-03-31	$18,756
1	2021-12-31	$15,339
2	2021-09-30	$13,757
3	2021-06-30	$11,958
4	2021-03-31	$10,389

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

[32]:
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

```
/tmp/wsuser/ipykernel_154/349343550.py:1: FutureWarning: The default value of
regex will change from True to False in a future version.
  tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

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

[33]:
```
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.

[34]:
```
tesla_revenue.tail()
```

[34]:

	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

0.4 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`.

[35]:
```
gamestop = 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.

[36]:
```
gme_data = gamestop.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.

[37]:
```
gme_data.reset_index(inplace=True)
print(gme_data.head())
```

```
        Date      Open      High       Low     Close    Volume  Dividends  \
0 2002-02-13  6.480514  6.773400  6.413184  6.766667  19054000        0.0
1 2002-02-14  6.850828  6.864294  6.682503  6.733000   2755400        0.0
2 2002-02-15  6.733002  6.749834  6.632007  6.699337   2097400        0.0
3 2002-02-19  6.665672  6.665672  6.312189  6.430017   1852600        0.0
4 2002-02-20  6.463680  6.648838  6.413182  6.648838   1723200        0.0

   Stock Splits
0           0.0
1           0.0
2           0.0
3           0.0
4           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`.

[39]:
```
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
```

Parse the html data using `beautiful_soup`.

[40]:
```
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
```

[41]:
```
gme_revenue = pd.DataFrame(columns=["Date", "Revenue"])
```

```
# 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_all("tbody")[1].find_all('tr'):
    col = row.find_all("td")
#     print(col)
    date = col[0].text
    revenue = col[1].text

#     # Finally we append the data of each row to the table
    gme_revenue = gme_revenue.append({"Date":date, "Revenue":revenue},␣
 ↪ignore_index=True)
```

[42]:
```
gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")
```

```
/tmp/wsuser/ipykernel_154/401512746.py:1: FutureWarning: The default value of
regex will change from True to False in a future version.
  gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"")
```

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

[43]:
```
gme_revenue.tail()
```

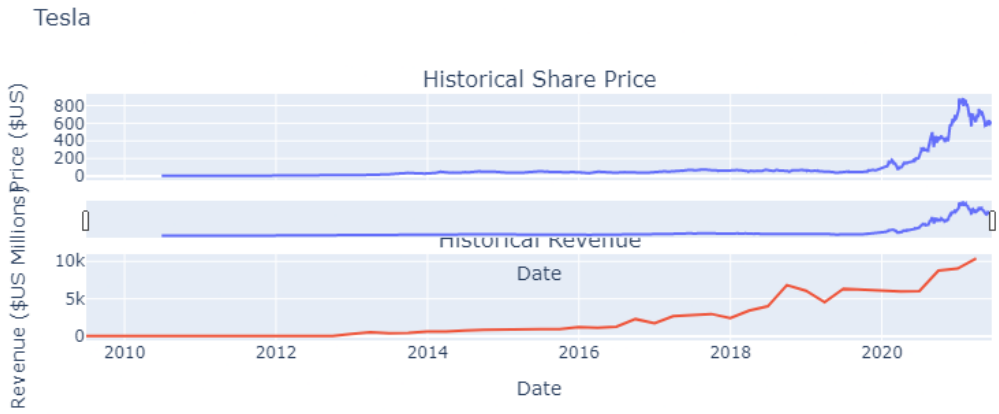
[43]:

	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

0.6 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.

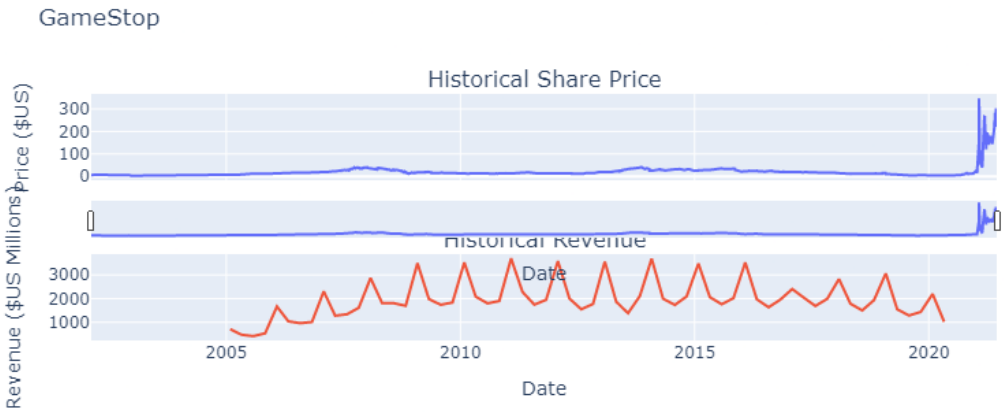
[46]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla')
```



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.

[47]:
```
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

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

##