Skills
Network

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.

In [1]:
```
#!pip install yfinance==0.2.38
#!pip install pandas==2.2.2
#!pip install nbformat
```

In [2]:
```
!pip install yfinance
!pip install bs4
!pip install nbformat
```

```
Collecting yfinance
  Downloading yfinance-0.2.43-py2.py3-none-any.whl (84 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 84.6/84.6 kB 11.6 MB/s eta 0:00:00
Requirement already satisfied: pandas>=1.3.0 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from yfinance) (1.3.5)
Requirement already satisfied: numpy>=1.16.5 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from yfinance) (1.21.6)
Collecting requests>=2.31 (from yfinance)
  Downloading requests-2.31.0-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.6/62.6 kB 15.5 MB/s eta 0:00:00
Collecting multitasking>=0.0.7 (from yfinance)
  Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Requirement already satisfied: lxml>=4.9.1 in /home/jupyterlab/conda/envs/python/
lib/python3.7/site-packages (from yfinance) (4.9.2)
Collecting platformdirs>=2.0.0 (from yfinance)
  Downloading platformdirs-4.0.0-py3-none-any.whl (17 kB)
Requirement already satisfied: pytz>=2022.5 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from yfinance) (2023.3)
Collecting frozendict>=2.3.4 (from yfinance)
  Downloading frozendict-2.4.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86
_64.whl (103 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 103.7/103.7 kB 17.7 MB/s eta 0:00:00
Collecting peewee>=3.16.2 (from yfinance)
  Downloading peewee-3.17.6.tar.gz (3.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.0/3.0 MB 63.6 MB/s eta 0:00:00:0
0:01
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: beautifulsoup4>=4.11.1 in /home/jupyterlab/conda/e
nvs/python/lib/python3.7/site-packages (from yfinance) (4.11.1)
Collecting html5lib>=1.1 (from yfinance)
  Downloading html5lib-1.1-py2.py3-none-any.whl (112 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 112.2/112.2 kB 22.9 MB/s eta 0:00:00
Requirement already satisfied: soupsieve>1.2 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from beautifulsoup4>=4.11.1->yfinance) (2.3.2.post
1)
Requirement already satisfied: six>=1.9 in /home/jupyterlab/conda/envs/python/li
b/python3.7/site-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.7.3 in /home/jupyterlab/conda/e
nvs/python/lib/python3.7/site-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Collecting typing-extensions>=4.7.1 (from platformdirs>=2.0.0->yfinance)
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Requirement already satisfied: charset-normalizer<4,>=2 in /home/jupyterlab/cond
a/envs/python/lib/python3.7/site-packages (from requests>=2.31->yfinance) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /home/jupyterlab/conda/envs/
python/lib/python3.7/site-packages (from requests>=2.31->yfinance) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in /home/jupyterlab/conda/envs/
python/lib/python3.7/site-packages (from requests>=2.31->yfinance) (2023.5.7)
Building wheels for collected packages: peewee
  Building wheel for peewee (pyproject.toml) ... done
  Created wheel for peewee: filename=peewee-3.17.6-py3-none-any.whl size=138888 s
ha256=bb744422ae7a258389ca1abd5eeb767c97ba8986451cd36389a93df534baad63
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/dd/16/8f/bdde4dfda69996
dc9e226111ccfd4a4d247cb61b42a237c3cc
Successfully built peewee
```

```
Installing collected packages: peewee, multitasking, typing-extensions, requests,
html5lib, frozendict, platformdirs, yfinance
  Attempting uninstall: typing-extensions
    Found existing installation: typing_extensions 4.5.0
    Uninstalling typing_extensions-4.5.0:
      Successfully uninstalled typing_extensions-4.5.0
  Attempting uninstall: requests
    Found existing installation: requests 2.29.0
    Uninstalling requests-2.29.0:
      Successfully uninstalled requests-2.29.0
Successfully installed frozendict-2.4.4 html5lib-1.1 multitasking-0.0.11 peewee-
3.17.6 platformdirs-4.0.0 requests-2.31.0 typing-extensions-4.7.1 yfinance-0.2.43
Collecting bs4
  Downloading bs4-0.0.2-py2.py3-none-any.whl (1.2 kB)
Requirement already satisfied: beautifulsoup4 in /home/jupyterlab/conda/envs/pyth
on/lib/python3.7/site-packages (from bs4) (4.11.1)
Requirement already satisfied: soupsieve>1.2 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from beautifulsoup4->bs4) (2.3.2.post1)
Installing collected packages: bs4
Successfully installed bs4-0.0.2
Requirement already satisfied: nbformat in /home/jupyterlab/conda/envs/python/li
b/python3.7/site-packages (5.8.0)
Requirement already satisfied: fastjsonschema in /home/jupyterlab/conda/envs/pyth
on/lib/python3.7/site-packages (from nbformat) (2.16.3)
Requirement already satisfied: importlib-metadata>=3.6 in /home/jupyterlab/conda/
envs/python/lib/python3.7/site-packages (from nbformat) (4.11.4)
Requirement already satisfied: jsonschema>=2.6 in /home/jupyterlab/conda/envs/pyt
hon/lib/python3.7/site-packages (from nbformat) (4.17.3)
Requirement already satisfied: jupyter-core in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from nbformat) (4.12.0)
Requirement already satisfied: traitlets>=5.1 in /home/jupyterlab/conda/envs/pyth
on/lib/python3.7/site-packages (from nbformat) (5.9.0)
Requirement already satisfied: zipp>=0.5 in /home/jupyterlab/conda/envs/python/li
b/python3.7/site-packages (from importlib-metadata>=3.6->nbformat) (3.15.0)
Requirement already satisfied: typing-extensions>=3.6.4 in /home/jupyterlab/cond
a/envs/python/lib/python3.7/site-packages (from importlib-metadata>=3.6->nbforma
t) (4.7.1)
Requirement already satisfied: attrs>=17.4.0 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from jsonschema>=2.6->nbformat) (23.1.0)
Requirement already satisfied: importlib-resources>=1.4.0 in /home/jupyterlab/con
da/envs/python/lib/python3.7/site-packages (from jsonschema>=2.6->nbformat) (5.1
2.0)
Requirement already satisfied: pkgutil-resolve-name>=1.3.10 in /home/jupyterlab/c
onda/envs/python/lib/python3.7/site-packages (from jsonschema>=2.6->nbformat) (1.
3.10)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in /
home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from jsonschema>=
2.6->nbformat) (0.19.3)
```

In [4]:
```python
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 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 [6]:
```
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 [7]:
```
def make_graph(stock_data, revenue_data, stock):
    fig = make_subplots(rows=2, cols=1, shared_xaxes=True, subplot_titles=("Hist
    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), y=stock
    fig.add_trace(go.Scatter(x=pd.to_datetime(revenue_data_specific.Date), y=rev
    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()
```

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 Dataimport yfinance as yf

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 [8]:
```
import yfinance as yf
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 [9]:
```
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.

In [10]:
```
reset_index=(inplace=True)
tesla_data = tesla.history(period="5d")
tesla_data.head()
```

Out[10]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2024-09-20 00:00:00-04:00	241.520004	243.990005	235.919998	238.250000	99879100	0.0	0.0
2024-09-23 00:00:00-04:00	242.610001	250.000000	241.919998	250.000000	86927200	0.0	0.0
2024-09-24 00:00:00-04:00	254.080002	257.190002	249.050003	254.270004	88491000	0.0	0.0
2024-09-25 00:00:00-04:00	252.539993	257.049988	252.279999	257.019989	65034300	0.0	0.0
2024-09-26 00:00:00-04:00	260.600006	261.750000	251.529999	254.220001	66972600	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 [ ]:
```
import requests
from bs4 import BeautifulSoup
url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevel
response = requests.get(url)
html_data = response.text
soup = BeautifulSoup(html_data, 'html.parser')
```

'Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`. Make sure to use the `html_data` with the content parameter as follow `html_data.content` .

```
In [ ]: soup=BeautifulSoup(html_data,'html.parser')
        html_data.content
```

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`.

```
In [40]: %%html
         <h3>Tesla Revenue </h3>

         <p>
         <table class=('tesla_revenue')>
               <tr>
              <td>date</td>
            <td>Revenue</td>
               </tr>
```

Tesla Revenue

date   Revenue

▶ Step-by-step instructions

▶ Click here if you need help locating the table

```
In [ ]: soup.find_all("tbody")[1]
```

```
In [32]: pd.concat([df, pd.DataFrame],ignore_index=True)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[32], line 1
----> 1 pd.concat([df, pd.DataFrame],ignore_index=True)

NameError: name 'pd' is not defined
```

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

```
In [33]: tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"", regex
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[33], line 1
----> 1 tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|
\$',"", regex=True)

NameError: name 'tesla_revenue' is not defined
```

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

```
In [39]: tesla_revenue.dropna(inplace=True)
         tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[39], line 1
----> 1 tesla_revenue.dropna(inplace=True)
      2 tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]

NameError: name 'tesla_revenue' is not defined
```

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

```
In [38]: reset_index=True
         tesla_revenue = tesla.history(period="5d")
         tesla_revenue.tail()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[38], line 2
      1 reset_index=True
----> 2 tesla_revenue = tesla.history(period="5d")
      3 tesla_revenue.tail()

NameError: name 'tesla' is not defined
```

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 [ ]: import yfinance as yf
        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.

```
In [ ]: 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 [ ]: reset_index(inplace=True)
        gme_data = gme.history(period="5d")
        gme_data.head()
```

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 [ ]:
```
import requests
from bs4 import BeautifulSoup
url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDevel
response = requests.get(url)
html_data_2 = response.text
soup = BeautifulSoup(html_data, 'html.parser')
```

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

In [ ]:
```
soup=BeautifulSoup(html_data_2,'html.parser')
html_data_2.content
```

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 [ ]:
```
dataframe_list = pd.read_html(url, GME Revenue='bs4')
<table class='gme_revenue'>
  <tr>
     <td>date</td>
    <td>Revenue</td>
      </tr>
   dataframe_list
   dataframe_list

soup.find_all("tbody")[1]
```

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

In [ ]:
```
dataframe_list = pd.read_html(url, GME Revenue='bs4')
```

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 [ ]: make_graph(gme_data, gme_revenue, 'gme')
```

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

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1], line 1
----> 1 make_graph(gme_data, gme_revenue, 'GameStop')

NameError: name 'make_graph' is not defined
```

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.

```
toggle

toggle

toggle

toggle

toggle

toggle
```