Final Assignment

April 18, 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

```
<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

[64]:
```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0
```

```
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) (1.3.5)
Requirement already satisfied: requests>=2.20 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (2.28.1)
Requirement already satisfied: lxml>=4.5.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (4.9.1)
Requirement already satisfied: multitasking>=0.0.7 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
yfinance==0.1.67) (0.0.11)
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) (2022.6)
Requirement already satisfied: charset-normalizer<3,>=2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.20->yfinance==0.1.67) (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==0.1.67) (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==0.1.67) (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==0.1.67) (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==0.1.67) (1.16.0)

                  __    __    __    __
                 /  \  /  \  /  \  /  \
                /    \/    \/    \/    \
        /  / /  / /  / /  /
              /  / \   / \   / \   / \  \____
             /  /   \_/   \_/   \_/   \    o \__,
            / _/                       \_____/  `
            |/





        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/main/noarch         [>                    ] (--:--) No change
pkgs/main/noarch         [====================] (00m:00s) No change
pkgs/r/linux-64          [>                    ] (--:--) No change
pkgs/r/linux-64          [====================] (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: nbformat==4.2.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (4.2.0)
Requirement already satisfied: jupyter-core in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (4.12.0)
Requirement already satisfied: traitlets>=4.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (5.6.0)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (4.17.3)
Requirement already satisfied: ipython-genutils in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
nbformat==4.2.0) (0.2.0)
Requirement already satisfied: pkgutil-resolve-name>=1.3.10 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (1.3.10)
Requirement already satisfied: importlib-resources>=1.4.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (5.10.1)
Requirement already satisfied: attrs>=17.4.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (22.1.0)
Requirement already satisfied: typing-extensions in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (4.4.0)
Requirement already satisfied: importlib-metadata in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (4.11.4)
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!=2.5.0,>=2.4->nbformat==4.2.0) (0.19.2)
Requirement already satisfied: zipp>=3.1.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from importlib-
resources>=1.4.0->jsonschema!=2.5.0,>=2.4->nbformat==4.2.0) (3.11.0)
```

[65]:
```
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.

[66]:
```
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`.

[67]:
```
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.

[68]:
```
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.

[69]:
```
tesla_data.reset_index(inplace=True)

print(tesla_data.head())
```

```
        Date      Open      High       Low     Close     Volume  Dividends  \
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

   Stock Splits
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`.

[70]:
```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
  ↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm"
response = requests.get(url)
html_data = response.text
```

Parse the html data using `beautiful_soup`.

[71]:
```
soup = BeautifulSoup(html_data, 'html.parser')
```

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
```

[72]:
```
table = soup.find('table', {'class': 'table'})
tesla_revenue = pd.read_html(str(table))[0]
```

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

[73]:
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~/conda/envs/python/lib/python3.7/site-packages/pandas/core/indexes/base.py in␣
 ↪get_loc(self, key, method, tolerance)
   3360             try:
-> 3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:

~/conda/envs/python/lib/python3.7/site-packages/pandas/_libs/index.pyx in panda
 ↪_libs.index.IndexEngine.get_loc()

~/conda/envs/python/lib/python3.7/site-packages/pandas/_libs/index.pyx in panda
 ↪_libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

KeyError: 'Revenue'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
/tmp/ipykernel_69/349343550.py in <module>
----> 1 tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.
 ↪replace(',|\$',"")

~/conda/envs/python/lib/python3.7/site-packages/pandas/core/frame.py in␣
 ↪__getitem__(self, key)
```

```
   3456                 if self.columns.nlevels > 1:
   3457                     return self._getitem_multilevel(key)
-> 3458                 indexer = self.columns.get_loc(key)
   3459                 if is_integer(indexer):
   3460                     indexer = [indexer]

~/conda/envs/python/lib/python3.7/site-packages/pandas/core/indexes/base.py in␣
 ↪get_loc(self, key, method, tolerance)
   3361                     return self._engine.get_loc(casted_key)
   3362                 except KeyError as err:
-> 3363                     raise KeyError(key) from err
   3364
   3365             if is_scalar(key) and isna(key) and not self.hasnans:

KeyError: 'Revenue'
```

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

[ ]:
```
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.

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

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`.

[ ]:
```
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.

[ ]:
```
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.

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

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`.

```
[ ]: url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
      ↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"
     response = requests.get(url)
     html_data = response.text
```

Parse the html data using `beautiful_soup`.

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

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
```

```
[ ]: # Extract table with GameStop Quarterly Revenue
     table = soup.find_all("table")[1]

     # Convert table to dataframe
     gme_revenue = pd.read_html(str(table))[0]

     # Rename columns and remove dollar signs and commas from Revenue column
     gme_revenue.columns = ['Date', 'Revenue']
     gme_revenue['Revenue'] = gme_revenue['Revenue'].str.replace('$', '').str.
      ↪replace(',', '').astype(float)

     # Display dataframe
     print(gme_revenue.head())
```

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

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

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.

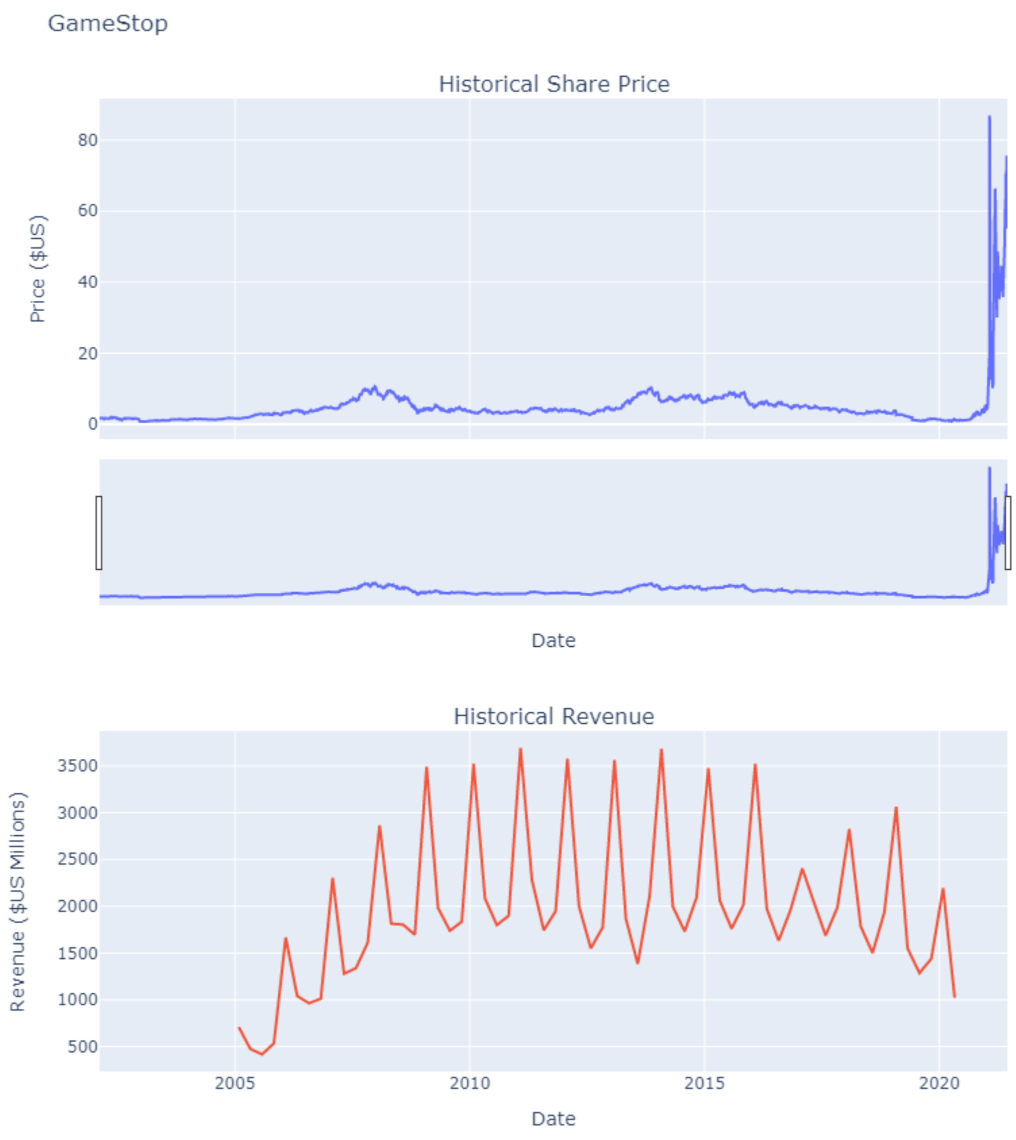
[75]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
/tmp/ipykernel_69/835219548.py in <module>
----> 1 make_graph(tesla_data, tesla_revenue, 'Tesla')
      2

/tmp/ipykernel_69/2068038883.py in make_graph(stock_data, revenue_data, stock)
      2     fig = make_subplots(rows=2, cols=1, shared_xaxes=True,␣
 ↪subplot_titles=("Historical Share Price", "Historical Revenue"),␣
 ↪vertical_spacing = .3)
      3     stock_data_specific = stock_data[stock_data.Date <= '2021--06-14']
----> 4     revenue_data_specific = revenue_data[revenue_data.Date <=␣
 ↪'2021-04-30']
      5     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)
      6     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)

~/conda/envs/python/lib/python3.7/site-packages/pandas/core/generic.py in␣
 ↪__getattr__(self, name)
   5485         ):
   5486             return self[name]
-> 5487         return object.__getattribute__(self, name)
   5488
   5489     def __setattr__(self, name: str, value) -> None:

AttributeError: 'DataFrame' object has no attribute 'Date'
```

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.

[74]:
```
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

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

##