Submission Final Assignment(1)

January 22, 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

    Define a Function that Makes a Graph
    Question 1: Use yfinance to Extract Stock Data
    Question 2: Use Webscraping to Extract Tesla Revenue Data
    Question 3: Use yfinance to Extract Stock Data
    Question 4: Use Webscraping to Extract GME Revenue Data
    Question 5: Plot Tesla Stock Graph
    Question 6: Plot GameStop Stock Graph

Estimated Time Needed: 30 min

[1]:
```
#!pip install yfinance==0.1.67
#!mamba install bs4==4.10.0 -y
#!pip install nbformat
!pip install pandas==1.3.5.
```

```
Requirement already satisfied: pandas==1.3.5. in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (1.3.5)
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.5.) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.5.) (2022.6)
Requirement already satisfied: numpy>=1.17.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas==1.3.5.) (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==1.3.5.) (1.16.0)
```

[2]:
```
!pip install yfinance
!mamba install bs4
!pip install nbformat
#!pip install pandas==1.3.3
#!pip install requests==2.26.0
!mamba install bs4==4.10.0 -y
!mamba install html5lib==1.1 -y
!pip install lxml==4.6.4
#!pip install plotly==5.3.1
```

```
Collecting yfinance
  Downloading yfinance-0.2.4-py2.py3-none-any.whl (51 kB)
                              51.4/51.4 kB
5.3 MB/s eta 0:00:00
Requirement already satisfied: cryptography>=3.3.2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(38.0.2)
Requirement already satisfied: pytz>=2022.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(2022.6)
Collecting appdirs>=1.4.4
  Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Requirement already satisfied: html5lib>=1.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(1.1)
Collecting frozendict>=2.3.4
  Downloading
frozendict-2.3.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (99
kB)
                              99.5/99.5 kB
6.0 MB/s eta 0:00:00
Collecting multitasking>=0.0.7
  Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Collecting lxml>=4.9.1
  Downloading lxml-4.9.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.m
anylinux_2_24_x86_64.whl (6.6 MB)
                              6.6/6.6 MB
56.1 MB/s eta 0:00:0000:0100:01
Requirement already satisfied: numpy>=1.16.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(1.21.6)
Requirement already satisfied: pandas>=1.3.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(1.3.5)
Requirement already satisfied: requests>=2.26 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from yfinance)
(2.28.1)
```

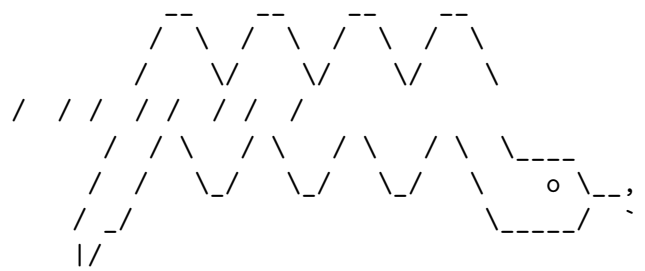
```
Collecting beautifulsoup4>=4.11.1
  Using cached beautifulsoup4-4.11.1-py3-none-any.whl (128 kB)
Requirement already satisfied: soupsieve>1.2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
beautifulsoup4>=4.11.1->yfinance) (2.3.2.post1)
Requirement already satisfied: cffi>=1.12 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
cryptography>=3.3.2->yfinance) (1.15.1)
Requirement already satisfied: webencodings in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: six>=1.9 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: charset-normalizer<3,>=2 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.26->yfinance) (2.1.1)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.26->yfinance) (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>=2.26->yfinance) (1.26.13)
Requirement already satisfied: idna<4,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
requests>=2.26->yfinance) (3.4)
Requirement already satisfied: pycparser in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
cffi>=1.12->cryptography>=3.3.2->yfinance) (2.21)
Installing collected packages: multitasking, appdirs, lxml, frozendict,
beautifulsoup4, yfinance
  Attempting uninstall: lxml
    Found existing installation: lxml 4.6.4
    Uninstalling lxml-4.6.4:
      Successfully uninstalled lxml-4.6.4
  Attempting uninstall: beautifulsoup4
    Found existing installation: beautifulsoup4 4.10.0
    Uninstalling beautifulsoup4-4.10.0:
      Successfully uninstalled beautifulsoup4-4.10.0
Successfully installed appdirs-1.4.4 beautifulsoup4-4.11.1 frozendict-2.3.4
lxml-4.9.2 multitasking-0.0.11 yfinance-0.2.4
```

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

```
        mamba (0.15.3) supported by @QuantStack

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

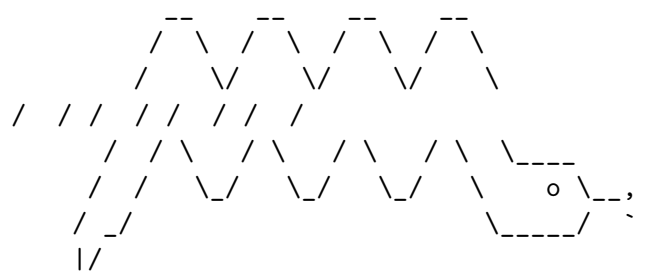
```
Looking for: ['bs4']

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

Pinned packages:
  - python 3.7.*


Transaction

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

  All requested packages already installed

Requirement already satisfied: nbformat in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (5.7.0)
Requirement already satisfied: jupyter-core in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from nbformat)
(4.12.0)
Requirement already satisfied: traitlets>=5.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from nbformat)
```

```
(5.6.0)
Requirement already satisfied: fastjsonschema in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from nbformat)
(2.16.2)
Requirement already satisfied: jsonschema>=2.6 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from nbformat)
(4.17.3)
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: typing-extensions>=3.6.4 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from importlib-
metadata>=3.6->nbformat) (4.4.0)
Requirement already satisfied: zipp>=0.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from importlib-
metadata>=3.6->nbformat) (3.11.0)
Requirement already satisfied: importlib-resources>=1.4.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema>=2.6->nbformat) (5.10.1)
Requirement already satisfied: attrs>=17.4.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
jsonschema>=2.6->nbformat) (22.1.0)
Requirement already satisfied: pkgutil-resolve-name>=1.3.10 in
/home/jupyterlab/conda/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) (0.19.2)
```



```
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       Using cache
pkgs/main/noarch         Using cache
pkgs/r/linux-64          Using cache
pkgs/r/noarch            Using cache

Pinned packages:
  - python 3.7.*


Transaction

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

  All requested packages already installed
```



```
        mamba (0.15.3) supported by @QuantStack

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


Looking for: ['html5lib==1.1']

pkgs/main/linux-64       Using cache
```

```
pkgs/main/noarch         Using cache
pkgs/r/linux-64          Using cache
pkgs/r/noarch            Using cache

Pinned packages:
  - python 3.7.*


Transaction

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

  All requested packages already installed

Collecting lxml==4.6.4
  Using cached lxml-4.6.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.
manylinux_2_24_x86_64.whl (6.3 MB)
Installing collected packages: lxml
  Attempting uninstall: lxml
    Found existing installation: lxml 4.9.2
    Uninstalling lxml-4.9.2:
      Successfully uninstalled lxml-4.9.2
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
yfinance 0.2.4 requires lxml>=4.9.1, but you have lxml 4.6.4 which is
incompatible.
Successfully installed lxml-4.6.4
```

[5]:
```
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.

[6]:
```
```

```
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`.

[7]:
```
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.

[8]:
```
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.

[9]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

[9]:

	Date	Open	High	Low	Close	\
0	2010-06-29 00:00:00-04:00	1.266667	1.666667	1.169333	1.592667	
1	2010-06-30 00:00:00-04:00	1.719333	2.028000	1.553333	1.588667	
2	2010-07-01 00:00:00-04:00	1.666667	1.728000	1.351333	1.464000	
3	2010-07-02 00:00:00-04:00	1.533333	1.540000	1.247333	1.280000	
4	2010-07-06 00:00:00-04:00	1.333333	1.333333	1.055333	1.074000	

```
      Volume  Dividends  Stock Splits
0  281494500        0.0           0.0
1  257806500        0.0           0.0
2  123282000        0.0           0.0
3   77097000        0.0           0.0
4  103003500        0.0           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`.

[10]:
```
url="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
  ↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm "
html_data  = requests.get(url).text
```

Parse the html data using `beautiful_soup`.

[11]:
```
soup = BeautifulSoup(html_data,'html5lib').text
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/bs4/builder/__init__.py:546: XMLParsedAsHTMLWarning: It looks like
you're parsing an XML document using an HTML parser. If this really is an HTML
document (maybe it's XHTML?), you can ignore or filter this warning. If it's
XML, you should know that using an XML parser will be more reliable. To parse
this document as XML, make sure you have the lxml package installed, and pass
the keyword argument `features="xml"` into the BeautifulSoup constructor.
  XMLParsedAsHTMLWarning.MESSAGE, XMLParsedAsHTMLWarning
```

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
```

[12]:
```
read_html_pandas_data = pd.read_html(url)
tesla_revenue=read_html_pandas_data [1]
```

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

[13]:
```
tesla_revenue["Tesla Quarterly Revenue(Millions of US $).1"] =␣
↪tesla_revenue['Tesla Quarterly Revenue(Millions of US $).1'].str.
↪replace(',|\$',"")
```

```
/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.
  """Entry point for launching an IPython kernel.
```

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

[14]:
```
tesla_revenue.dropna(inplace=True)

tesla_revenue = tesla_revenue[tesla_revenue['Tesla Quarterly Revenue(Millions␣
↪of US $).1'] != ""]
```

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

[15]:
```
tesla_revenue.tail()
```

[15]:
```
   Tesla Quarterly Revenue(Millions of US $)  \
48                                2010-09-30
49                                2010-06-30
50                                2010-03-31
52                                2009-09-30
53                                2009-06-30

   Tesla Quarterly Revenue(Millions of US $).1
48                                          31
49                                          28
50                                          21
52                                          46
53                                          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`.

[16]:
```
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.

[20]:
```
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.

[22]:
```
gme_data.reset_index(inplace=True)
gme_data.head()
```

[22]:
```
   index                       Date      Open      High       Low     Close  \
0      0 2002-02-13 00:00:00-05:00  1.620128  1.693350  1.603296  1.691666
1      1 2002-02-14 00:00:00-05:00  1.712707  1.716074  1.670626  1.683250
2      2 2002-02-15 00:00:00-05:00  1.683251  1.687459  1.658002  1.674834
3      3 2002-02-19 00:00:00-05:00  1.666418  1.666418  1.578047  1.607504
4      4 2002-02-20 00:00:00-05:00  1.615920  1.662210  1.603296  1.662210

     Volume  Dividends  Stock Splits
0  76216000        0.0           0.0
1  11021600        0.0           0.0
2   8389600        0.0           0.0
3   7410400        0.0           0.0
4   6892800        0.0           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`.

[23]:
```
url2="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
  ↪IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"
html_data2  = requests.get(url2).text
```

Parse the html data using `beautiful_soup`.

[24]:
```
soup2 = BeautifulSoup(html_data2)
GMEpandas = pd.read_html(url2)
GME_revenue=pd.DataFrame(GMEpandas[1])
```

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

[25]:
```
GME_revenue["GameStop Quarterly Revenue(Millions of US $).
  ↪1"]=GME_revenue["GameStop Quarterly Revenue(Millions of US $).1"].str.
  ↪replace(',|\$',"")
#GME_revenue.reset_index(inplace=True)
```

```
/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.
  """Entry point for launching an IPython kernel.
```

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

[27]:
```
GME_revenue.tail()
```

[27]:

	GameStop Quarterly Revenue(Millions of US $)	\
57	2006-01-31	
58	2005-10-31	
59	2005-07-31	
60	2005-04-30	
61	2005-01-31	

	GameStop Quarterly Revenue(Millions of US $).1
57	1667
58	534
59	416
60	475
61	709

0.6 Question 5: Plot Tesla Stock Graph

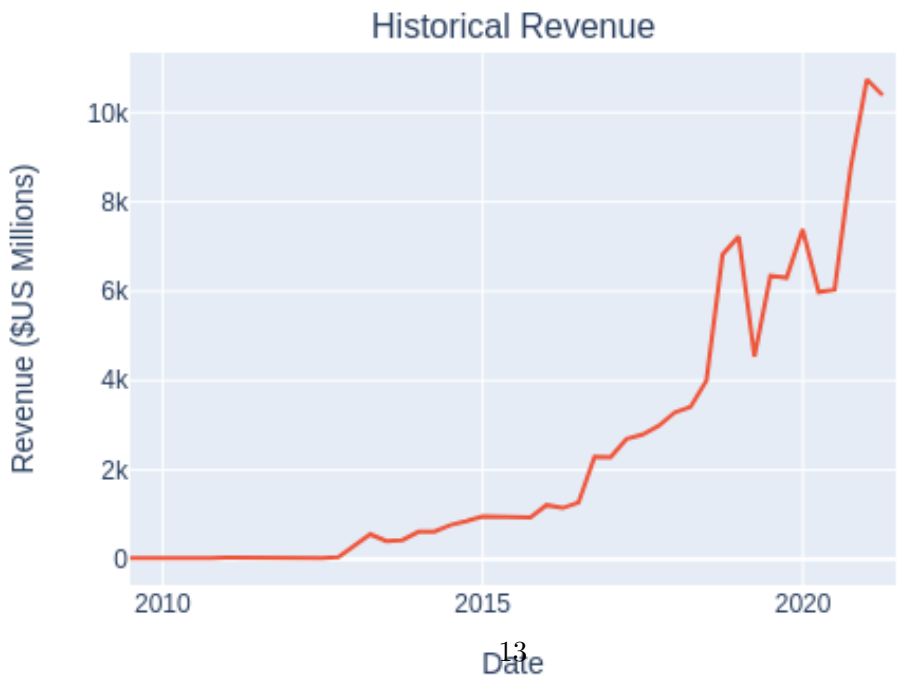
Use the `make_graph` function to graph the Tesla Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(tesla_data, tesla_revenue, 'Tesla')`. Note the graph will only show data upto June 2021.

[29]:
```
#tesla_revenue2=df.rename(Tesla Quarterly Revenue(Millions of US $)
tesla_revenue2=tesla_revenue.columns = ['Date', 'Revenue']
```

[30]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla')
```

Tesla


Historical Share Price
Price ($US)
300
250
200
150
100
50
0
Date

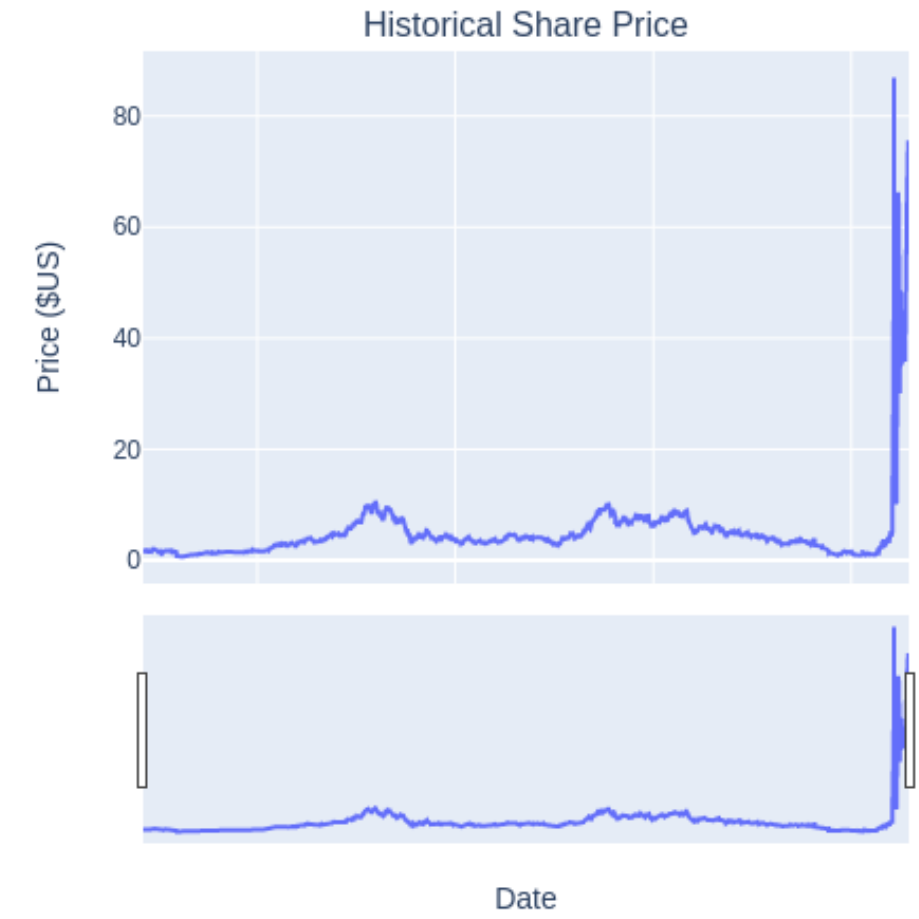

Historical Revenue
Revenue ($US Millions)
10k
8k
6k
4k
2k
0
2010
2015
2020
Date

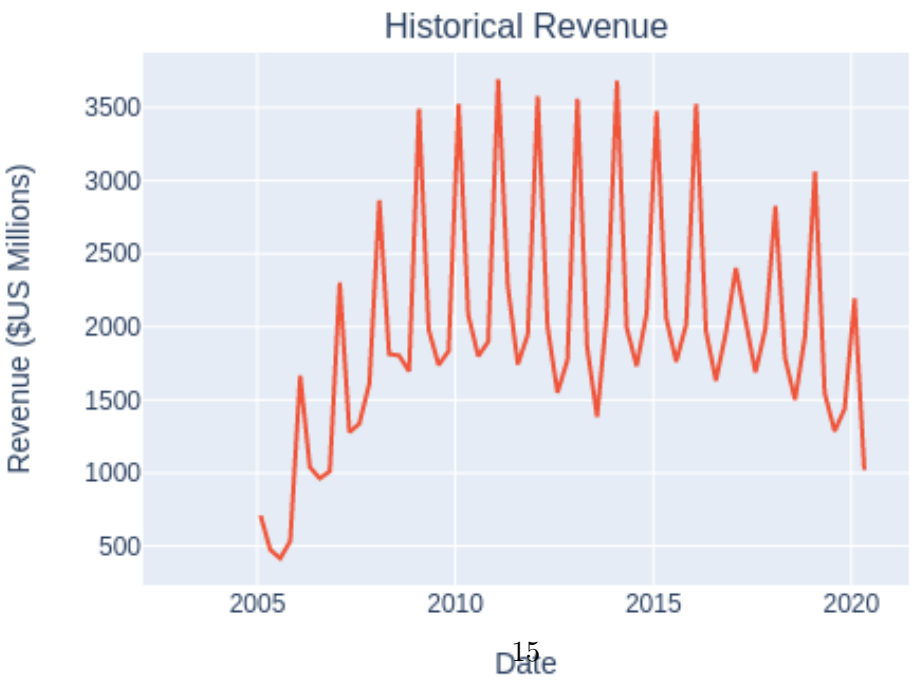
0.7 Question 6: Plot GameStop Stock Graph

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.

[32]:
```
GME_revenue.columns = ['Date', 'Revenue']
make_graph(GME_data, GME_revenue, 'GameStop')
```

GameStop


Historical Share Price
Price ($US)
80
60
40
20
0
Date


Historical Revenue
Revenue ($US Millions)
3500
3000
2500
2000
1500
1000
500
2005
2010
2015
2020
Date

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

##