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.

Note:- If you are working Locally using anaconda, please uncomment the following code and execute it.

```
#!pip install yfinance==0.2.38
#!pip install pandas==2.2.2
#!pip install nbformat

!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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.

```
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.

```
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()
```

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`.

!pip install yfinance==0.1.67 !mamba install bs4==4.10.0 -y !pip install nbformat==4.2.0

```
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

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.

```
tesla_data = tesla.history(period="max")
print (tesla_data)

                 Open      High       Low     Close     Volume
\
Date

2010-06-29   1.266667  1.666667  1.169333  1.592667  281494500

2010-06-30   1.719333  2.028000  1.553333  1.588667  257806500

2010-07-01   1.666667  1.728000  1.351333  1.464000  123282000
```

```
2010-07-02    1.533333    1.540000    1.247333    1.280000   77097000

2010-07-06    1.333333    1.333333    1.055333    1.074000  103003500

...                ...         ...         ...         ...        ...

2024-05-21  175.509995  186.880005  174.710007  186.600006  115266500

2024-05-22  182.850006  183.800003  178.119995  180.110001   88313500

2024-05-23  181.800003  181.899994  173.259995  173.740005   71975500

2024-05-24  174.839996  180.080002  173.729996  179.240005   65479700

2024-05-28  176.300995  176.539993  173.160004  175.630005   26626581


            Dividends  Stock Splits
Date
2010-06-29          0           0.0
2010-06-30          0           0.0
2010-07-01          0           0.0
2010-07-02          0           0.0
2010-07-06          0           0.0
...               ...           ...
2024-05-21          0           0.0
2024-05-22          0           0.0
2024-05-23          0           0.0
2024-05-24          0           0.0
2024-05-28          0           0.0

[3502 rows x 7 columns]
```

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.

```
tesla_data.reset_index(inplace=True)
tesla_data = tesla.history(period="5d")
tesla_data.head()

                  Open        High         Low       Close     Volume
\
Date

2024-05-21  175.509995  186.880005  174.710007  186.600006  115266500

2024-05-22  182.850006  183.800003  178.119995  180.110001   88313500
```

```
2024-05-23  181.800003  181.899994  173.259995  173.740005   71975500

2024-05-24  174.839996  180.080002  173.729996  179.240005   65479700

2024-05-28  176.300995  176.539993  173.160004  175.220001   27340585


            Dividends  Stock Splits
Date
2024-05-21          0             0
2024-05-22          0             0
2024-05-23          0             0
2024-05-24          0             0
2024-05-28          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`.

```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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

url='https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-
SkillsNetwork/labs/project/revenue.htm'
```

Parse the html data using `beautiful_soup`.

```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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

from bs4 import BeautifulSoup
import requests
url='https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-
SkillsNetwork/labs/project/revenue.htm'
html_data=requests.get(url)
soup=BeautifulSoup(html_data.text,'html')
```

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`.

```
soup.find_all('table')

soup.find_all('table',class_='historical_data_table table')[1]

import warnings
# Ignore all warnings
warnings.filterwarnings("ignore", category=FutureWarning)
import pandas as pd
tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])
for row in soup.find_all("tbody")[1].find_all('tr'):
    col = row.find_all("td")
    date = col[0].text
    revenue = col[1].text
    tesla_revenue = tesla_revenue.append({"Date":date,
"Revenue":revenue}, ignore_index=True)
    tesla_revenue["Revenue"] =
tesla_revenue['Revenue'].str.replace(',|\$',"")

tesla_revenue.tail()

          Date Revenue
49  2010-06-30      28
50  2010-03-31      21
51  2009-12-31
52  2009-09-30      46
53  2009-06-30      27
```

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

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

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`.

```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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

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")
print (gme_data)

                 Open       High        Low      Close     Volume  
Dividends  \
Date                                                                
                                                                
2002-02-13   1.620128   1.693350   1.603296   1.691666   76216000  
0.0
2002-02-14   1.712707   1.716074   1.670626   1.683250   11021600  
0.0
2002-02-15   1.683250   1.687458   1.658001   1.674834    8389600  
0.0
2002-02-19   1.666418   1.666418   1.578048   1.607504    7410400  
0.0
2002-02-20   1.615920   1.662210   1.603296   1.662210    6892800  
0.0
...               ...        ...        ...        ...        ...  
...
2024-05-22  21.559999  22.250000  20.760000  21.120001   43521400  
0.0
2024-05-23  21.400000  21.400000  18.260000  18.320000   30561100  
0.0
2024-05-24  18.420000  19.680000  17.700001  19.000000   41886700  
0.0
2024-05-28  23.100000  26.660000  21.150000  23.780001  104676900  
0.0
```

```
2024-05-29  22.000000  22.930000  21.799999  22.254299    9257035
0.0

            Stock Splits
Date
2002-02-13           0.0
2002-02-14           0.0
2002-02-15           0.0
2002-02-19           0.0
2002-02-20           0.0
...                  ...
2024-05-22           0.0
2024-05-23           0.0
2024-05-24           0.0
2024-05-28           0.0
2024-05-29           0.0

[5611 rows x 7 columns]
```

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)
gme_data = gme.history(period="5d")
gme_data.head()

                 Open   High        Low      Close     Volume
Dividends  \
Date

2024-05-22  21.559999  22.25  20.760000  21.120001   43521400
0
2024-05-23  21.400000  21.40  18.260000  18.320000   30561100
0
2024-05-24  18.420000  19.68  17.700001  19.000000   41886700
0
2024-05-28  23.100000  26.66  21.150000  23.780001  104676900
0
2024-05-29  22.000000  22.93  21.799999  22.200001    9362779
0

            Stock Splits
Date
2024-05-22             0
2024-05-23             0
2024-05-24             0
2024-05-28             0
2024-05-29             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`.

```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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

url="https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-
SkillsNetwork/labs/project/stock.html"
```

Parse the html data using `beautiful_soup`.

```
from bs4 import BeautifulSoup
import requests
url='https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-
SkillsNetwork/labs/project/revenue.htm'
data=requests.get(url)
soup=BeautifulSoup(data.text,'html')
```

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 using a method similar to what you did in Question 2.

```
soup.find_all('table')

soup.find_all('table',class_='historical_data_table table')[1]

import warnings
# Ignore all warnings
warnings.filterwarnings("ignore", category=FutureWarning)
import pandas as pd
gme_revenue = pd.DataFrame(columns=["Date", "Revenue"])
for row in soup.find_all("tbody")[1].find_all('tr'):
    col = row.find_all("td")
    date = col[0].text
    revenue = col[1].text
    gme_revenue = gme_revenue.append({"Date":date, "Revenue":revenue}, 
ignore_index=True)
    gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\
$',"")

gme_revenue.tail()
```

```
          Date Revenue
49  2010-06-30      28
50  2010-03-31      21
51  2009-12-31        
52  2009-09-30      46
53  2009-06-30      27
```

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

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.

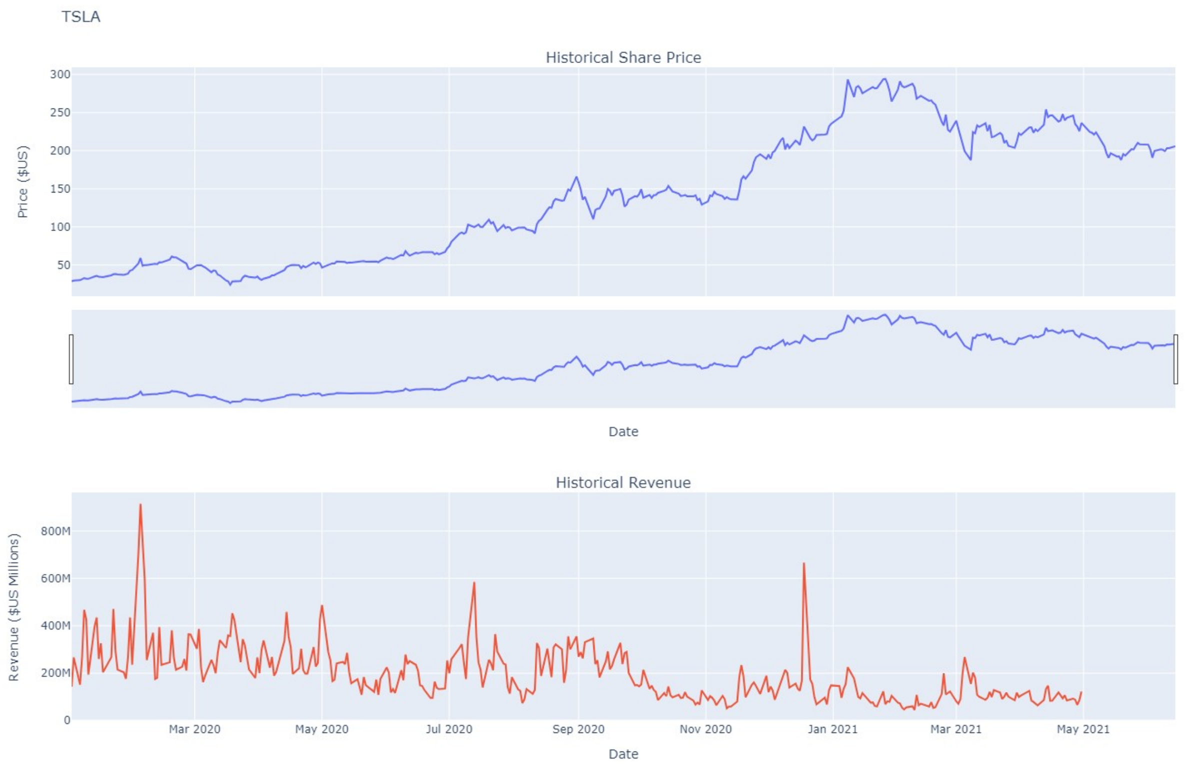
```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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

stock_data = yf.download("TSLA", start="2020-01-01", end="2021-09-30", 
progress=False)
revenue_data = yf.download("TSLA", start="2020-01-01", end="2021-09-
30", progress=False)
stock_data.reset_index(inplace=True)
revenue_data.reset_index(inplace=True)
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.Volume.astype("float"), name="Volume"), 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()
make_graph(stock_data, revenue_data, 'TSLA')
```



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.

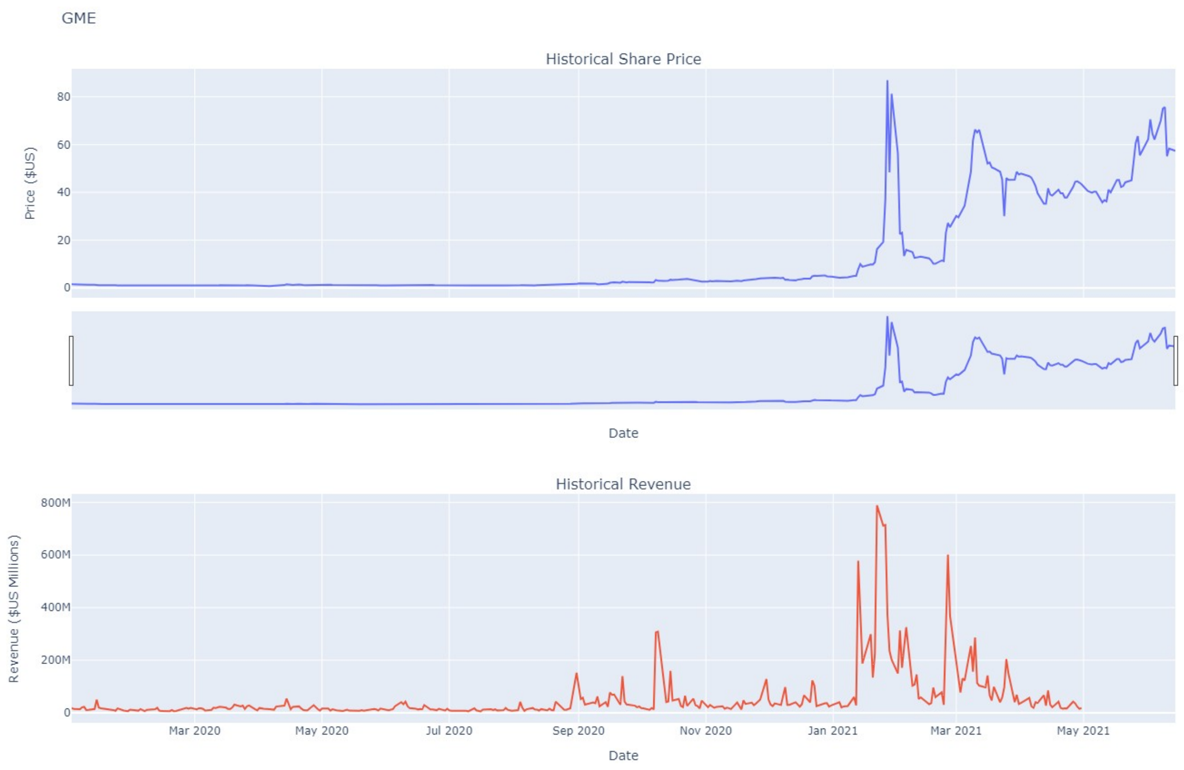
```
!pip install yfinance==0.1.67
!mamba install bs4==4.10.0 -y
!pip install nbformat==4.2.0

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

stock_data = yf.download("GME", start="2020-01-01", end="2021-09-30",
progress=False)
revenue_data = yf.download("GME", start="2020-01-01", end="2021-09-
30", progress=False)
stock_data.reset_index(inplace=True)
revenue_data.reset_index(inplace=True)
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.Volume.astype("float"), name="Volume"), 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()
make_graph(stock_data, revenue_data, 'GME')
```



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