* Using yfinance to extract stock data
  + Note also to install yfinance.
    - **!pip install yfinance==0.1.67**
  + To import
    - **Import yfinance as yf**
    - Also just import pandas
      * **Import pandas as pd**
  + To extract stock information, you need to know the ticker symbol
    - Ex: stocks for apple are **AAPL.**
    - Command:
      * **Apple = yf.Ticker(“AAPL”)**
    - Now we set object apple as the ticker.
    - We can now extract information by using .info on apple
      * **apple\_info=apple.info**
        + when printing apple\_info it should return a series of information in a dictionary.
    - We can also get a series of information such as ‘country’, ‘zip’, from the dictionary.
      * To return information from dictionary we can simply call
        + **Apple\_info[‘country’]**

This should return ‘United States’

* + - * Extracting share price information we use **history()**
        + The format is as follows

**apple\_share\_price\_data = apple.history**(**period=”max”)**

* + - * + Note **apple\_share\_price\_data** is just the object
        + **apple.history** #apple is the object that we are calling history on.)
        + **(period=”max”)** #period is a parameter within **history().** We are setting the parameter **“max”** as period.

There are multiple options for max.

**1 day (1d), 5d, 1 month (1mo) , 3mo, 6mo, 1 year (1y), 2y, 5y, 10y, ytd, and max.**

* + - * + To display with we simple use

**apple\_share\_price\_data.head()**

this function returns a dataframe in pandas showing all the columns and the first 5 rows

* + - * + We can reset the index of the Dataframe with

**apple\_share\_price\_data.reset\_index(inplace=True)**

* + - * + To plot using pandas (In this case we are plotting Data vs Open)

**apple\_share\_price\_data.plot(x="Date", y="Open")**

* + - * + To return dividends

**Use: apple.dividends**

* + - * + To plot

Use: **apple.dividends.plot()**

* + - * + To receive specific columns use this format

**apple\_share\_price\_data.head(1).Volume** #This .head(how many rows you want to see) the.Volume prints only the volume collumn.

* Packages to import
  + **!mamba install bs4==4.10.0 -y**
  + **!mamba install html5lib==1.1 -y**
  + **!pip install lxml==4.6.4**
* Import following
  + **import pandas as pd**
  + **import requests**
  + **from bs4 import BeautifulSoup**
* Using ‘Request’ library to download webpage info.
  + Set url to the link. \*Make sure you include the quotation marks
    - **url = “**[**https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/netflix\_data\_webpage.html**](https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/netflix_data_webpage.html)**”**
  + set the request to data \*note we insert url into get()
    - **data = requests.get(url).text**
* To parse text to html use beautiful\_soup \*note we already set the request to data and we are putting it inBeautifulSoup()
  + **soup = BeautifulSoup(data, 'html5lib')**
* Create a dataframe
  + **netflix\_data = pd.DataFrame(columns=["Date", "Open", "High", "Low", "Close", "Volume"])**
* Find all the column values for each row
  + **for row in soup.find("tbody").find\_all('tr'):**
  + **col = row.find\_all("td")**
  + **date = col[0].text**
  + **Open = col[1].text**
  + **high = col[2].text**
  + **low = col[3].text**
  + **close = col[4].text**
  + **adj\_close = col[5].text**
  + **volume = col[6].text**
* Appending all data to each row and column
  + **netflix\_data = netflix\_data.append({"Date":date, "Open":Open, "High":high, "Low":low, "Close":close, "Adj Close":adj\_close, "Volume":volume}, ignore\_index=True)**
* Using pandas’s read\_html using the url.
  + **read\_html\_pandas\_data = pd.read\_html(url)**
* Converting BeautifulSoup object to string
  + **read\_html\_pandas\_data = pd.read\_html(str(soup))**
* Printing the table
  + **netflix\_dataframe = read\_html\_pandas\_data[0]**
  + **netflix\_dataframe.head()**
* **Exercises**
  + Use the requests library to download the webpage <https://cf-courses-data.s3.us.cloud-objectstorage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/amazon_data_webpage.html>. Save the text of the response as a variable named html\_data.
    - **url\_1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/amazon\_data\_webpage.html"**
    - **html\_data\_! = requests.get(url).text**
  + Parse the html data using beautiful\_soup.
    - **parsedsoup = BeautifulSoup(html\_data\_1, 'html5lib')**
  + **Question 1** What is the content of the title attribute: \*Use soup.title
* <title>Amazon.com, Inc. (AMZN) Stock Historical Prices &amp; Data - Yahoo Finance</title>
  + Using beautiful soup extract the table with historical share prices and store it into a dataframe named amazon\_data. The dataframe should have columns Date, Open, High, Low, Close, Adj Close, and Volume. Fill in each variable with the correct data from the list col.
    - **amazon\_data = pd.DataFrame(columns=["Date", "Open", "High", "Low", "Close", "Volume"])**
    - **for row in parsedsoup.find("tbody").find\_all("tr"):**
    - **col = row.find\_all("td")**
    - **date = col[0].text**
    - **Open = col[1].text**
    - **high = col[2].text**
    - **low = col[3].text**
    - **close = col[4].text**
    - **adj\_close = col[5].text**
    - **volume = col[6].text**
    - **amazon\_data = amazon\_data.append({"Date":date, "Open":Open, "High":high, "Low":low, "Close":close, "Adj Close":adj\_close, "Volume":volume}, ignore\_index=True)**
  + Print out the first five rows of the amazon\_data dataframe you created.
    - **amazon\_data.head(5)**
  + **Question 2** What is the name of the columns of the dataframe
    - **Data, Open, High, Low, Close, Volume, Adj Close**
  + **Question 3** What is the Open of the last row of the amazon\_data dataframe?
    - **Amazon.data.tail()**
    - **amazon\_data.iloc[-1:]**
    - **109.35**
  + **Finding a specific row**
    - **amazon\_data['Open'][amazon\_data['Date']=="Jun 1, 2019"]**

**Final Assignment**

* **Question 1**
  + 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.
    - **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\_info = Tesla.info**
    - **Tesla\_data = Tesla.history(period="max")**
    - **Tesla\_data**
  + **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.head(5)**
* **Question 2**
  + Use the requests library to download the webpage [https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue](https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0220ENSkillsNetwork23455606-2021-01-01). Save the text of the response as a variable named html\_data.
    - **url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"**
    - **html\_data = requests.get(url).text**
    - **#html\_data**
  + Parse the html data using beautiful\_soup.
    - **tesla\_soup = BeautifulSoup(html\_data, "html5lib")**
    - **#tesla\_soup**
  + 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.
    - **read\_html\_pandas\_data = pd.read\_html(url)**
    - **read\_html\_pandas\_data = pd.read\_html(str(tesla\_soup))**
    - **tesla\_revenue = read\_html\_pandas\_data[1]**
    - **tesla\_revenue.rename(columns = {'Tesla Quarterly Revenue(Millions of US $)':'Date'}, inplace=True)**
    - **tesla\_revenue.rename(columns = {'Tesla Quarterly Revenue(Millions of US $).1':'Revenue'}, inplace=True)**
    - **tesla\_revenue.head(5)**
  + Execute the following line to remove the comma and dollar sign from the Revenue column.
    - **tesla\_revenue["Revenue"] = tesla\_revenue["Revenue"].str.replace(',|\$',"")**
    - **tesla\_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'] != ""]**
    - **tesla\_revenue**
* **Question 3**
  + 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\_info = gme.info**
    - **gme\_data = gme.history(period="max")**
    - **#gme\_data**
  + **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.head(5)**
* **Question 4**
  + Use the requests library to download the webpage [https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue](https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0220ENSkillsNetwork23455606-2021-01-01). Save the text of the response as a variable named html\_data.
    - **url\_gme = "https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue"**
    - **html\_data\_gme = requests.get(url\_gme).text**
    - **#html\_data\_gme**
  + Parse the html data using beautiful\_soup.
    - **gme\_soup = BeautifulSoup(html\_data\_gme, "html5lib")**
    - **#gme\_soup**
  + 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.
    - **gme\_read\_html\_pandas\_data = pd.read\_html(url\_gme)**
    - **gme\_read\_html\_pandas\_data = pd.read\_html(str(gme\_soup))**
    - **gme\_revenue = gme\_read\_html\_pandas\_data[1]**
    - **gme\_revenue.rename(columns = {'GameStop Quarterly Revenue(Millions of US $)':'Date'}, inplace=True)**
    - **gme\_revenue.rename(columns = {'GameStop Quarterly Revenue(Millions of US $).1':'Revenue'}, inplace=True)**
    - **gme\_revenue["Revenue"] = gme\_revenue["Revenue"].str.replace(',|\$',"")**
    - **gme\_revenue**