Module 7: Data Wrangling with Pandas

Computational Thinking with Python

Submitted by: Jay-ann Alorro

Performed on: 03/20/2024

Submitted on: 03/20/2024

Submitted to: Engr. Roman M. Richard

7.1 Supplementary Activity

Exercise 1

We want to look at data for the Facebook, Apple, Amazon, Netflix, and Google (FAANG) stocks, but we were given each as separate CSV file. Combine them into a single file and store the dataframe of the FAANG data as faang for the rest of the exercises:

1. Read each file in.

In [14]:
```
import pandas as pd

# 1. Read each file in.
aapl_df = pd.read_csv('/content/module 7/aapl.csv')

amzn_df = pd.read_csv('/content/module 7/amzn.csv')

fb_df = pd.read_csv('/content/module 7/fb.csv')

goog_df = pd.read_csv('/content/module 7/goog.csv')

nflx_df = pd.read_csv('/content/module 7/nflx.csv')
```

2. Add a column to each dataframe, called ticker, indicating the ticker symbol it is for (Apple's is AAPL, for example). This is how you look up a stock. Each file's name is also the ticker symbol, so be sure to capitalize it.

In [15]:
```
# 2. Add a column to each dataframe, called ticker, indicating the ticker symbol
# This is how you look up a stock. Each file's name is also the ticker symbol, s
aapl_df['ticker'] = 'AAPL'
amzn_df['ticker'] = 'AMZN'
fb_df['ticker'] = 'FB'
goog_df['ticker'] = 'GOOG'
nflx_df['ticker'] = "NFLX"
```

In [16]:
```
print('Apple Dataframe:')
aapl_df.head()
```

```
Apple Dataframe:
```

Out[16]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL

In [17]:
```
print('Amazon Dataframe:')
amzn_df.head()
```

```
Amazon Dataframe:
```

Out[17]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	1172.00	1190.00	1170.51	1189.01	2694494	AMZN
1	2018-01-03	1188.30	1205.49	1188.30	1204.20	3108793	AMZN
2	2018-01-04	1205.00	1215.87	1204.66	1209.59	3022089	AMZN
3	2018-01-05	1217.51	1229.14	1210.00	1229.14	3544743	AMZN
4	2018-01-08	1236.00	1253.08	1232.03	1246.87	4279475	AMZN

In [18]:
```
print('Facebook Dataframe:')
fb_df.head()
```

```
Facebook Dataframe:
```

Out[18]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	177.68	181.58	177.5500	181.42	18151903	FB
1	2018-01-03	181.88	184.78	181.3300	184.67	16886563	FB
2	2018-01-04	184.90	186.21	184.0996	184.33	13880896	FB
3	2018-01-05	185.59	186.90	184.9300	186.85	13574535	FB
4	2018-01-08	187.20	188.90	186.3300	188.28	17994726	FB

In [19]:
```
print('Google Dataframe:')
goog_df.head()
```

```
Google Dataframe:
```

Out[19]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	1048.34	1066.94	1045.23	1065.00	1237564	GOOG
1	2018-01-03	1064.31	1086.29	1063.21	1082.48	1430170	GOOG
2	2018-01-04	1088.00	1093.57	1084.00	1086.40	1004605	GOOG
3	2018-01-05	1094.00	1104.25	1092.00	1102.23	1279123	GOOG
4	2018-01-08	1102.23	1111.27	1101.62	1106.94	1047603	GOOG

In [20]:
```
print('Netflix Dataframe:')
nflx_df.head()
```

```
Netflix Dataframe:
```

Out[20]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	196.10	201.65	195.4200	201.07	10966889	NFLX
1	2018-01-03	202.05	206.21	201.5000	205.05	8591369	NFLX
2	2018-01-04	206.20	207.05	204.0006	205.63	6029616	NFLX
3	2018-01-05	207.25	210.02	205.5900	209.99	7033240	NFLX
4	2018-01-08	210.02	212.50	208.4400	212.05	5580178	NFLX

3. Append them together into a single dataframe.

In [28]:
```
# 3. Append them together into a single dataframe.
faang_df = pd.concat([aapl_df, amzn_df, fb_df, goog_df, nflx_df], ignore_index=T

faang_df.head(10)
```

Out[28]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL
5	2018-01-09	171.2337	171.7340	170.1154	171.0179	21583997	AAPL
6	2018-01-10	169.8701	170.9884	169.7131	170.9786	23959895	AAPL
7	2018-01-11	171.2729	172.1545	171.1748	171.9498	18667729	AAPL
8	2018-01-12	172.8327	173.9903	172.3128	173.7254	25418080	AAPL
9	2018-01-16	174.5200	175.9817	172.7935	172.8425	29565947	AAPL

In [29]:
```
faang_df.tail(10)
```

Out[29]:

	date	open	high	low	close	volume	ticker
1245	2018-12-17	266.51	272.9800	261.075	262.800	9634734	NFLX
1246	2018-12-18	263.30	275.7500	263.290	270.940	10350079	NFLX
1247	2018-12-19	269.96	280.8700	263.770	266.770	13788448	NFLX
1248	2018-12-20	264.64	269.9000	251.880	260.580	16792928	NFLX
1249	2018-12-21	263.83	264.5000	241.290	246.390	21397595	NFLX
1250	2018-12-24	242.00	250.6500	233.680	233.880	9547616	NFLX
1251	2018-12-26	233.92	254.5000	231.230	253.670	14402735	NFLX
1252	2018-12-27	250.11	255.5900	240.100	255.565	12235217	NFLX
1253	2018-12-28	257.94	261.9144	249.800	256.080	10987286	NFLX
1254	2018-12-31	260.16	270.1001	260.000	267.660	13508920	NFLX

4. Save the result file called faang.csv.

In [30]:
```
# 4. Save the result file called faang.csv.
faang_df.to_csv('module 7/faang.csv', index=False)
```

Exercise 2

In [31]:
```
df = pd.read_csv('/content/module 7/faang.csv')
df.head()
```

Out[31]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL

- With faang, use type conversion to change the date column into a datetime and the volume column into integers. Then, sort by date and ticker.

In [32]:
```
# With faang, use type conversion to change the date column into a datetime and
# Then, sort by date and ticker.
df.dtypes
```

Out[32]:
```
date       object
open      float64
high      float64
low       float64
close     float64
volume      int64
ticker     object
dtype: object
```

In [36]:
```
df = df.astype({'date': 'datetime64[ns]', 'volume': 'int64'})
```

In [37]:
```
df.dtypes
```

Out[37]:
```
date      datetime64[ns]
open             float64
high             float64
low              float64
close            float64
volume             int64
ticker            object
dtype: object
```

In [38]:
```
df.head()
```

Out[38]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL

- Find the seven rows with the highest value for volume.

In [39]:
```
# Find the seven rows with the highest value for volume.
df.nlargest(n=7, columns='volume')
```

Out[39]:

	date	open	high	low	close	volume	ticker
644	2018-07-26	174.8900	180.1300	173.7500	176.2600	169803668	FB
555	2018-03-20	167.4700	170.2000	161.9500	168.1500	129851768	FB
559	2018-03-26	160.8200	161.1000	149.0200	160.0600	126116634	FB
556	2018-03-21	164.8000	173.4000	163.3000	169.3900	106598834	FB
182	2018-09-21	219.0727	219.6482	215.6097	215.9768	96246748	AAPL
245	2018-12-21	156.1901	157.4845	148.9909	150.0862	95744384	AAPL
212	2018-11-02	207.9295	211.9978	203.8414	205.8755	91328654	AAPL

In [41]:
```
df.head()
```

Out[41]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL

- Right now, the data is somewhere between long and wide format. Use melt() to make it completely long format. Hint: date and ticker are our ID variables (they uniquely identify each row). We need to melt the rest so that we don't have separate columns for open, high, low, close, and volume.

In [40]:
```
# Right now, the data is somewhere between long and wide format.
# Use melt() to make it completely long format.
# Hint: date and ticker are our ID variables (they uniquely identify each row).
# We need to melt the rest so that we don't have separate columns for open, high

melted_df = df.melt(id_vars=['date', 'ticker'], var_name='attribute', value_name
melted_df.head()
```

Out[40]:

	date	ticker	attribute	value
0	2018-01-02	AAPL	open	166.9271
1	2018-01-03	AAPL	open	169.2521
2	2018-01-04	AAPL	open	169.2619
3	2018-01-05	AAPL	open	170.1448
4	2018-01-08	AAPL	open	171.0375

Exercise 3

- Using web scraping, search for the list of the hospitals, their address and contact information. Save the list in a new csv file, hospitals.csv.

In [128]:
```
# Using web scraping, search for the list of the hospitals, their address and co
# Save the list in a new csv file, hospitals.csv.

# url used: https://sulit.ph/list-of-hospitals-in-metro-manila-with-contact-deta

import requests
from bs4 import BeautifulSoup
import pandas as pd

hosp_url = 'https://sulit.ph/list-of-hospitals-in-metro-manila-with-contact-deta

response = requests.get(hosp_url)
response.status_code
```

Out[128]: 200

In [129]:
```
# the html content is parsed
soup = BeautifulSoup(response.content, 'html.parser')

# table element
table = soup.find('table')

# extract headers of table
headers = [header.text.strip() for header in table.find_all('th')]

# table rows
rows = []
for row in table.find_all('tr'):
  rows.append([cell.text.strip() for cell in row.find_all('td')])
```

In [130]:
```
# create dataframe
df = pd.DataFrame(rows, columns=headers)
```

In [131]:
```
df.head()
```

Out[131]:

	CITY	NAME OF HOSPITAL	CONTACT NUMBER	WEBSITE / EMAIL	FACEBOOK LINK
0	None	None	None	None	None
1	LIST UPDATE				
2	15 SEPT 2021				
3	Caloocan	Caloocan City Medical Center	South 5310 7925, North 8282 3397, 0943 216 6963		https://www.facebook.com/Caloocan-City-Medical...
4	Caloocan	Dr. Jose N. Rodriguez Memorial Hospital and Sa...	0966 549 2697, 8294 2571 to 73	http://djnrmh.doh.gov.ph/	https://www.facebook.com/officialDJNRMHS

In [132]:
```
# converting to csv file
df.to_csv('module 7/hospitals.csv', index=False)
```

- Using the generated hospitals.csv, convert the csv file into pandas dataframe. Prepare the data using necessary preprocessing techniques.

In [134]:
```
# Using the generated hospitals.csv, convert the csv file into pandas dataframe.
# Prepare the data using necessary preprocessing techniques.

hosp_df = pd.read_csv('module 7/hospitals.csv')
hosp_df.head()
```

Out[134]:

	CITY	NAME OF HOSPITAL	CONTACT NUMBER	WEBSITE / EMAIL	FACEBOOK LINK
0	NaN	NaN	NaN	NaN	NaN
1	LIST UPDATE	NaN	NaN	NaN	NaN
2	15 SEPT 2021	NaN	NaN	NaN	NaN
3	Caloocan	Caloocan City Medical Center	South 5310 7925, North 8282 3397, 0943 216 6963	NaN	https://www.facebook.com/Caloocan-City-Medical...
4	Caloocan	Dr. Jose N. Rodriguez Memorial Hospital and Sa...	0966 549 2697, 8294 2571 to 73	http://djnrmh.doh.gov.ph/	https://www.facebook.com/officialDJNRMHS

In [137]:
```
# removing the first 2 rows
hosp_df = df.drop(df.index[:3])
hosp_df.reset_index(drop=True, inplace=True)
```

In [139]:
```
hosp_df.head(10)
```

Out[139]:

	CITY	NAME OF HOSPITAL	CONTACT NUMBER	WEBSITE / EMAIL	
0	Caloocan	Caloocan City Medical Center	South 5310 7925, North 8282 3397, 0943 216 6963		https://www.facebook.com/Caloc
1	Caloocan	Dr. Jose N. Rodriguez Memorial Hospital and Sa...	0966 549 2697, 8294 2571 to 73	http://djnrmh.doh.gov.ph/	https://www.facebook.cor
2	Caloocan	MCU – FDT Medical Foundations Hospital	8367 2031	https://www.mcuhospital.org/	
3	Caloocan	Metro Balayan Medical Center	(043) 740 1350	http://www.metrobalayanmc.com.ph/	https://www.facebook.
4	Las Pinas	Alabang Medical Center	8807 8189, 8850 8719		https://www.facebook.com/ala
5	Las Pinas	Las Pinas City Medical Center / City Med	8806 2288, 8800 5695, 8800 5678	http://www.citymed.com.ph/	https://www.faceb
6	Las Pinas	Las Pinas Doctors Hospital	0917 836 9466	lpdhinc.com	https://www.facebook.com/LasPi
7	Las Pinas	Las Pinas General Hospital	8872 0509, 8873 0556, 8824 9435	https://lpghstc.doh.gov.ph/	https://www.facebook.com/lp
8	Las Pinas	Perpetual Help Medical Center	8874 8515	https://phmc.com.ph/	https://www.facebook.com/perpe
9	Makati	Makati Medical Center	8651 7800 loc 1149 or 1150	https://www.makatimed.net.ph/	https://www.facebook.

7.2 Conclusion:

In conclusion, this activity was quite the challenge. The first two exercises were not that bad because we already did those things with the topics covered in module 7, come refreshers and quick skim through our modules got the job done. Exercise 3 was very tricky for me for I don't

have any knowledge with web scraping so I don't know where or how to start. The search for the list of hospitals I thought would be the easiest because there are a lot of resources in the internet. Still, converting those lists from various websites is something I'm not sure I'm capable of doing. I tried a bunch of things at first with the use of a python library called tabula that deals with table reading in websites that directs you to a PDF file, I also encountered this library called PyPDF2 which also reads a PDF file from a website. However, I found it hard to convert it into a dataframe, the contents are jumbled and messy when being converted and I don't know how to fix it.

With some more research, I discovered BeautifulSoup which is a python package used to get html elements so that helped and the code I did was short and pretty straight forward. I also got lucky and found a website that has a table element to it which contains all the information required in the hospital list which led to my final output in which I hope I did right. Nonetheless, I learned a lot of new things doind this activity and I would like to explore some of the things I encountered more and how to utilize them properly