Module 7: Data Wrangling with Pandas

CPE311 Computational Thinking With Python

Name: Borja, Angelo Louis C.
Section: CPE22S3
Performed on: 03/20/2024
Submitted on: 03/20/2024
Submitted to:Engr . Roman M. Richard

7.1 Supplementary Activity

Using the datasets provided, perform the following exercises:

Exercise 1

We want to look at data for the Facebook, Apple, Amazon, Netflix, and Google (FAANG) stocks, but we were given each as a 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.

```
from google.colab import drive
drive.mount('/content/drive')
import pandas as pd
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
apple_df = pd.read_csv('/content/drive/MyDrive/HOA7 files/aapl.csv')
amzn_df = pd.read_csv('/content/drive/MyDrive/HOA7 files/amzn.csv')
fb_df = pd.read_csv('/content/drive/MyDrive/HOA7 files/fb.csv')
google_df = pd.read_csv('/content/drive/MyDrive/HOA7 files/goog.csv')
netflix_df = pd.read_csv('/content/drive/MyDrive/HOA7 files/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.

```
apple_df ['ticker'] = ('AAPL')
amzn_df ['ticker'] = ('AMZN')
fb_df ['ticker'] = ('FB')
google_df ['ticker'] = ('GOOG')
netflix_df ['ticker'] = ('NFLX')
```

3.Append them together into a single dataframe.

```
merged_df = pd.concat([apple_df, amzn_df, fb_df, google_df, netflix_df])
merged_df
```

	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
...	...	...	...	...	...	...	...
246	2018-12-24	242.0000	250.6500	233.6800	233.8800	9547616	NFLX
247	2018-12-26	233.9200	254.5000	231.2300	253.6700	14402735	NFLX
248	2018-12-27	250.1100	255.5900	240.1000	255.5650	12235217	NFLX
249	2018-12-28	257.9400	261.9144	249.8000	256.0800	10987286	NFLX
250	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX

1255 rows × 7 columns

Next steps: View recommended plots

4.Save the result in a CSV file called faang.csv

```
merged_df.to_csv('/content/drive/MyDrive/HOA7 files/faang.csv', index = False)
```

Exercise 2

- 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.
- Find the seven rows with the highest value for volume.
- Right now, the data is somewehere between long and wide format. Use melt to make it completely long format. Hint: date and ticker are our ID Variables (the 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

```
faang_df = pd.read_csv('/content/drive/MyDrive/HOA7 files/faang.csv')
```

```
faang_df['date'] = pd.to_datetime(faang_df["date"])
faang_df.sort_values(['date', 'ticker'], ascending=(True, True))
faang_df.dtypes
```

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

```
faang_df
```

	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
...	...	...	...	...	...	...	...
1250	2018-12-24	242.0000	250.6500	233.6800	233.8800	9547616	NFLX
1251	2018-12-26	233.9200	254.5000	231.2300	253.6700	14402735	NFLX
1252	2018-12-27	250.1100	255.5900	240.1000	255.5650	12235217	NFLX
1253	2018-12-28	257.9400	261.9144	249.8000	256.0800	10987286	NFLX
1254	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX

1255 rows × 7 columns

Next steps: View recommended plots

```
faang_df.sort_values('volume', ascending=False).head(7)
```

	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

```
melted_faang = pd.melt(faang_df, id_vars=['date','ticker'], var_name='Type', value_name='value')
melted_faang
```

	date	ticker	Type	value
0	2018-01-02	AAPL	open	1.669271e+02
1	2018-01-03	AAPL	open	1.692521e+02
2	2018-01-04	AAPL	open	1.692619e+02
3	2018-01-05	AAPL	open	1.701448e+02
4	2018-01-08	AAPL	open	1.710375e+02
...	...	...	...	...
6270	2018-12-24	NFLX	volume	9.547616e+06
6271	2018-12-26	NFLX	volume	1.440274e+07
6272	2018-12-27	NFLX	volume	1.223522e+07
6273	2018-12-28	NFLX	volume	1.098729e+07
6274	2018-12-31	NFLX	volume	1.350892e+07

6275 rows × 4 columns

Next steps: View recommended plots

Exercise 3

- Using web scraping, search for the list of hospitals, their address and contact information. Save the list in a new csv file, hospitals.csv.
- convert the csv file into pandas dataframe. Prepare the data using the necessary preprocessing techniques.

```
from bs4 import BeautifulSoup
import requests as rq
req = rq.get("https://sulit.ph/list-of-hospitals-in-metro-manila-with-contact-details-website-and-social-media-accounts/")
bSoup = BeautifulSoup(req.content, 'html.parser')
rowheader = []
for x in bSoup.find_all('tr'):
  for y in x.find_all('th'):
    rowheader.append(y.text)
    hospitalTable = []

for x in bSoup.find_all('tr')[3:]:
  tdfind = x.find_all('td')
  tdval = [y.text for y in tdfind]
  hospitalTable.append(tdval)

hospitals = pd.DataFrame(hospitalTable, columns= rowheader).drop(["WEBSITE / EMAIL","FACEBOOK LINK"], axis = 1)
hospitals
```

	CITY	NAME OF HOSPITAL	CONTACT NUMBER
0	Caloocan	Caloocan City Medical Center	South 5310 7925, North 8282 3397, 0943 216 6963
1	Caloocan	Dr. Jose N. Rodriguez Memorial Hospital and Sa...	0966 549 2697, 8294 2571 to 73
2	Caloocan	MCU – FDT Medical Foundations Hospital	8367 2031
3	Caloocan	Metro Balayan Medical Center	(043) 740 1350
4	Las Pinas	Alabang Medical Center	8807 8189, 8850 8719
...	...	...	...
91	Taguig	Medical Center of Taguig	8888 6284
92	Valenzuela	Allied Care Experts (ACE) Medical Center	direct line to Admission 0917 844 3654, 8332 0...
93	Valenzuela	Fatima University Medical Center	8291 6538
94	Valenzuela	Valenzuela Citicare Medical Center	8860 9300, 8860 9300
95	Valenzuela	Valenzuela Medical Center	8294 6711

96 rows × 3 columns

Next steps: View recommended plots

7.2 Conclusion

In this hands-on activity, I learned how to apply data-wrangling techniques using the Pandas library. In more detail, these are the things I performed on the given four csv tables: I've added an additional column named 'ticker' to all four of the dataframes while also adding a value for that column to all of the entries, merged all of them into one dataframe, changed a column's data type, performed sorting, and reshaped a dataframe by using the melt() method. The most important thing I have learned is how to perform web scraping using the BeautifulSoup library.