Module 7: Data Wrangling with Pandas

CPE311 Computational Thinking with Python

Submitted by: Cabacungan, John Martin C.

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.
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.
3. Append them together into a single dataframe.
4. Save the result in a CSV file called faang.csv.

```
# Read Each file in

import pandas as pd

am = pd.read_csv('/content/amzn.csv')
fb = pd.read_csv('/content/fb.csv')
go = pd.read_csv('/content/goog.csv')
nf = pd.read_csv('/content/nflx.csv')
ap = pd.read_csv('/content/aapl.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

```
am.loc[:, 'ticker'] = 'AMZN'
fb.loc[:, 'ticker'] = 'FCBK'
go.loc[:, 'ticker'] = 'GOGL'
nf.loc[:, 'ticker'] = 'NFLX'
ap.loc[:, 'ticker'] = 'AAPL'
```

```
# Append them together into a single dataframe.
faang = pd.concat([am,fb,go,nf,ap])
faang
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	1172.0000	1190.0000	1170.5100	1189.0100	2694494	AMZN
1	2018-01-03	1188.3000	1205.4900	1188.3000	1204.2000	3108793	AMZN
2	2018-01-04	1205.0000	1215.8700	1204.6600	1209.5900	3022089	AMZN
3	2018-01-05	1217.5100	1229.1400	1210.0000	1229.1400	3544743	AMZN
4	2018-01-08	1236.0000	1253.0800	1232.0300	1246.8700	4279475	AMZN
...	...	...	...	...	...	...	...
246	2018-12-24	147.5173	150.9027	145.9639	146.2029	37169232	AAPL
247	2018-12-26	147.6666	156.5585	146.0934	156.4987	58582544	AAPL
248	2018-12-27	155.1744	156.1004	149.4291	155.4831	53117065	AAPL
249	2018-12-28	156.8273	157.8430	153.8899	155.5627	42291424	AAPL
250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466	AAPL

1255 rows × 7 columns

Next steps: View recommended plots

```
# Save the result in a CSV file called faang.csv.
faang.to_csv('faang.csv', index=False)
faang
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	1172.0000	1190.0000	1170.5100	1189.0100	2694494	AMZN
1	2018-01-03	1188.3000	1205.4900	1188.3000	1204.2000	3108793	AMZN
2	2018-01-04	1205.0000	1215.8700	1204.6600	1209.5900	3022089	AMZN
3	2018-01-05	1217.5100	1229.1400	1210.0000	1229.1400	3544743	AMZN
4	2018-01-08	1236.0000	1253.0800	1232.0300	1246.8700	4279475	AMZN
...	...	...	...	...	...	...	...
246	2018-12-24	147.5173	150.9027	145.9639	146.2029	37169232	AAPL
247	2018-12-26	147.6666	156.5585	146.0934	156.4987	58582544	AAPL
248	2018-12-27	155.1744	156.1004	149.4291	155.4831	53117065	AAPL
249	2018-12-28	156.8273	157.8430	153.8899	155.5627	42291424	AAPL
250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466	AAPL

1255 rows × 7 columns

Next steps: View recommended plots

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 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 seperate columns for open, high, low, close, and volume.

1. 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.

```
faang.rename(
    columns = {
        'date' : 'datetime',
        'volume' : 'integers'
    }, inplace = False
)
```

	datetime	open	high	low	close	integers	ticker
0	2018-01-02	1172.0000	1190.0000	1170.5100	1189.0100	2694494	AMZN
1	2018-01-03	1188.3000	1205.4900	1188.3000	1204.2000	3108793	AMZN
2	2018-01-04	1205.0000	1215.8700	1204.6600	1209.5900	3022089	AMZN
3	2018-01-05	1217.5100	1229.1400	1210.0000	1229.1400	3544743	AMZN
4	2018-01-08	1236.0000	1253.0800	1232.0300	1246.8700	4279475	AMZN
...	...	...	...	...	...	...	...
246	2018-12-24	147.5173	150.9027	145.9639	146.2029	37169232	AAPL
247	2018-12-26	147.6666	156.5585	146.0934	156.4987	58582544	AAPL
248	2018-12-27	155.1744	156.1004	149.4291	155.4831	53117065	AAPL
249	2018-12-28	156.8273	157.8430	153.8899	155.5627	42291424	AAPL
250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466	AAPL

1255 rows × 7 columns

```
faang.sort_values(by = ['date', 'ticker'],inplace = True)
faang
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
0	2018-01-02	1172.0000	1190.0000	1170.5100	1189.0100	2694494	AMZN
0	2018-01-02	177.6800	181.5800	177.5500	181.4200	18151903	FCBK
0	2018-01-02	1048.3400	1066.9400	1045.2300	1065.0000	1237564	GOGL
0	2018-01-02	196.1000	201.6500	195.4200	201.0700	10966889	NFLX
...	...	...	...	...	...	...	...
250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466	AAPL
250	2018-12-31	1510.8000	1520.7600	1487.0000	1501.9700	6954507	AMZN
250	2018-12-31	134.4500	134.6400	129.9500	131.0900	24625308	FCBK
250	2018-12-31	1050.9600	1052.7000	1023.5900	1035.6100	1493722	GOGL
250	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX

1255 rows × 7 columns

Next steps: View recommended plots

```
faang.sort_values('volume')
```

	date	open	high	low	close	volume	ticker
126	2018-07-03	1135.8200	1135.8200	1100.0200	1102.8900	679034	GOGL
226	2018-11-23	1030.0000	1037.5900	1022.4000	1023.8800	691462	GOGL
99	2018-05-24	1079.0000	1080.4700	1066.1500	1079.2400	766773	GOGL
130	2018-07-10	1156.9800	1159.5900	1149.5900	1152.8400	798412	GOGL
152	2018-08-09	1249.9000	1255.5400	1246.0100	1249.1000	848601	GOGL
...	...	...	...	...	...	...	...
182	2018-09-21	219.0727	219.6482	215.6097	215.9768	96246748	AAPL
54	2018-03-21	164.8000	173.4000	163.3000	169.3900	106598834	FCBK
57	2018-03-26	160.8200	161.1000	149.0200	160.0600	126116634	FCBK
53	2018-03-20	167.4700	170.2000	161.9500	168.1500	129851768	FCBK
142	2018-07-26	174.8900	180.1300	173.7500	176.2600	169803668	FCBK

1255 rows × 7 columns

```
faang.sort_values('volume').tail(7)
```

	date	open	high	low	close	volume	ticker
212	2018-11-02	207.9295	211.9978	203.8414	205.8755	91328654	AAPL
245	2018-12-21	156.1901	157.4845	148.9909	150.0862	95744384	AAPL
182	2018-09-21	219.0727	219.6482	215.6097	215.9768	96246748	AAPL
54	2018-03-21	164.8000	173.4000	163.3000	169.3900	106598834	FCBK
57	2018-03-26	160.8200	161.1000	149.0200	160.0600	126116634	FCBK
53	2018-03-20	167.4700	170.2000	161.9500	168.1500	129851768	FCBK
142	2018-07-26	174.8900	180.1300	173.7500	176.2600	169803668	FCBK

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

```
faang.sort_values(by=['volume'], ascending = False).head(7)
```

	date	open	high	low	close	volume	ticker
142	2018-07-26	174.8900	180.1300	173.7500	176.2600	169803668	FCBK
53	2018-03-20	167.4700	170.2000	161.9500	168.1500	129851768	FCBK
57	2018-03-26	160.8200	161.1000	149.0200	160.0600	126116634	FCBK
54	2018-03-21	164.8000	173.4000	163.3000	169.3900	106598834	FCBK
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

·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.

```
faang.melt(id_vars = ['date','ticker'],
           value_vars = ['open','high','low','close', 'volume']
)
```

	date	ticker	variable	value
0	2018-01-02	AAPL	open	1.669271e+02