Module 7: Data Wrangling with Pandas

CPE311 Computational Thinking with Python

Submitted by: Belocora, John Rome A.

Performed on: 03/20/2024

Submitted on: 03/20/2024

Submitted to: Engr. Roman M. Richard

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.

```
import pandas as pd
# Reading each file for each dataset
facebook = pd.read_csv('data/fb.csv')
facebook
```

	date	open	high	low	close	volume
0	2018-01-02	177.68	181.58	177.5500	181.42	18151903
1	2018-01-03	181.88	184.78	181.3300	184.67	16886563
2	2018-01-04	184.90	186.21	184.0996	184.33	13880896
3	2018-01-05	185.59	186.90	184.9300	186.85	13574535
4	2018-01-08	187.20	188.90	186.3300	188.28	17994726
...	...	...	...	...	...	...
246	2018-12-24	123.10	129.74	123.0200	124.06	22066002
247	2018-12-26	126.00	134.24	125.8900	134.18	39723370
248	2018-12-27	132.44	134.99	129.6700	134.52	31202509
249	2018-12-28	135.34	135.92	132.2000	133.20	22627569
250	2018-12-31	134.45	134.64	129.9500	131.09	24625308

251 rows × 6 columns

Next steps: View recommended plots

```
apple = pd.read_csv('data/aapl.csv')
apple
```

	date	open	high	low	close	volume
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766
...	...	...	...	...	...	...
246	2018-12-24	147.5173	150.9027	145.9639	146.2029	37169232
247	2018-12-26	147.6666	156.5585	146.0934	156.4987	58582544
248	2018-12-27	155.1744	156.1004	149.4291	155.4831	53117065
249	2018-12-28	156.8273	157.8430	153.8899	155.5627	42291424
250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466

251 rows × 6 columns

Next steps: View recommended plots

```
amazon = pd.read_csv('data/amzn.csv')
amazon
```

	date	open	high	low	close	volume
0	2018-01-02	1172.00	1190.00	1170.51	1189.01	2694494
1	2018-01-03	1188.30	1205.49	1188.30	1204.20	3108793
2	2018-01-04	1205.00	1215.87	1204.66	1209.59	3022089
3	2018-01-05	1217.51	1229.14	1210.00	1229.14	3544743
4	2018-01-08	1236.00	1253.08	1232.03	1246.87	4279475
...	...	...	...	...	...	...
246	2018-12-24	1346.00	1396.03	1307.00	1343.96	7219996
247	2018-12-26	1368.89	1473.16	1363.01	1470.90	10411801
248	2018-12-27	1454.20	1469.00	1390.31	1461.64	9722034
249	2018-12-28	1473.35	1513.47	1449.00	1478.02	8828950
250	2018-12-31	1510.80	1520.76	1487.00	1501.97	6954507

251 rows × 6 columns

Next steps: View recommended plots

```
netflix = pd.read_csv('data/nflx.csv')
netflix
```

	date	open	high	low	close	volume
0	2018-01-02	196.10	201.6500	195.4200	201.070	10966889
1	2018-01-03	202.05	206.2100	201.5000	205.050	8591369
2	2018-01-04	206.20	207.0500	204.0006	205.630	6029616
3	2018-01-05	207.25	210.0200	205.5900	209.990	7033240
4	2018-01-08	210.02	212.5000	208.4400	212.050	5580178
...	...	...	...	...	...	...
246	2018-12-24	242.00	250.6500	233.6800	233.880	9547616
247	2018-12-26	233.92	254.5000	231.2300	253.670	14402735
248	2018-12-27	250.11	255.5900	240.1000	255.565	12235217
249	2018-12-28	257.94	261.9144	249.8000	256.080	10987286
250	2018-12-31	260.16	270.1001	260.0000	267.660	13508920

251 rows × 6 columns

Next steps: View recommended plots

```
google = pd.read_csv('data/goog.csv')
google
```

	date	open	high	low	close	volume
0	2018-01-02	1048.34	1066.94	1045.23	1065.00	1237564
1	2018-01-03	1064.31	1086.29	1063.21	1082.48	1430170
2	2018-01-04	1088.00	1093.57	1084.00	1086.40	1004605
3	2018-01-05	1094.00	1104.25	1092.00	1102.23	1279123
4	2018-01-08	1102.23	1111.27	1101.62	1106.94	1047603
...	...	...	...	...	...	...
246	2018-12-24	973.90	1003.54	970.11	976.22	1590328
247	2018-12-26	989.01	1040.00	983.00	1039.46	2373270
248	2018-12-27	1017.15	1043.89	997.00	1043.88	2109777
249	2018-12-28	1049.62	1055.56	1033.10	1037.08	1413772
250	2018-12-31	1050.96	1052.70	1023.59	1035.61	1493722

251 rows × 6 columns

Next steps: View recommended plots

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.

```
# Adding a new column named "ticker" that consist the ticker's symbol
facebook['ticker'] = 'FB'
facebook
```

	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
...	...	...	...	...	...	...	...
246	2018-12-24	123.10	129.74	123.0200	124.06	22066002	FB
247	2018-12-26	126.00	134.24	125.8900	134.18	39723370	FB
248	2018-12-27	132.44	134.99	129.6700	134.52	31202509	FB
249	2018-12-28	135.34	135.92	132.2000	133.20	22627569	FB
250	2018-12-31	134.45	134.64	129.9500	131.09	24625308	FB

251 rows × 7 columns

Next steps: View recommended plots

```
apple['ticker'] = 'AAPL'
apple
```

	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	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

251 rows × 7 columns

Next steps: View recommended plots

```
amazon['ticker'] = 'AMZN'
amazon
```

```
           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
...         ...      ...      ...      ...      ...       ...     ...
246  2018-12-24  1346.00  1396.03  1307.00  1343.96   7219996    AMZN
247  2018-12-26  1368.89  1473.16  1363.01  1470.90  10411801    AMZN
248  2018-12-27  1454.20  1469.00  1390.31  1461.64   9722034    AMZN
249  2018-12-28  1473.35  1513.47  1449.00  1478.02   8828950    AMZN
250  2018-12-31  1510.80  1520.76  1487.00  1501.97   6954507    AMZN

251 rows × 7 columns
```

Next steps: View recommended plots

```
netflix['ticker'] = 'NFLX'
netflix
```

```
           date    open      high       low    close    volume  ticker
0    2018-01-02  196.10  201.6500  195.4200  201.070  10966889    NFLX
1    2018-01-03  202.05  206.2100  201.5000  205.050   8591369    NFLX
2    2018-01-04  206.20  207.0500  204.0006  205.630   6029616    NFLX
3    2018-01-05  207.25  210.0200  205.5900  209.990   7033240    NFLX
4    2018-01-08  210.02  212.5000  208.4400  212.050   5580178    NFLX
...         ...     ...       ...       ...      ...       ...     ...
246  2018-12-24  242.00  250.6500  233.6800  233.880   9547616    NFLX
247  2018-12-26  233.92  254.5000  231.2300  253.670  14402735    NFLX
248  2018-12-27  250.11  255.5900  240.1000  255.565  12235217    NFLX
249  2018-12-28  257.94  261.9144  249.8000  256.080  10987286    NFLX
250  2018-12-31  260.16  270.1001  260.0000  267.660  13508920    NFLX

251 rows × 7 columns
```

Next steps: View recommended plots

```
google['ticker'] = 'GOOG'
google
```

	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
...	...	...	...	...	...	...	...
246	2018-12-24	973.90	1003.54	970.11	976.22	1590328	GOOG
247	2018-12-26	989.01	1040.00	983.00	1039.46	2373270	GOOG
248	2018-12-27	1017.15	1043.89	997.00	1043.88	2109777	GOOG
249	2018-12-28	1049.62	1055.56	1033.10	1037.08	1413772	GOOG
250	2018-12-31	1050.96	1052.70	1023.59	1035.61	1493722	GOOG

251 rows × 7 columns

Next steps: View recommended plots

```
# Using concat we can append all the dataframes together
faang = pd.concat([facebook, apple, amazon, netflix, google])
faang
```

	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
...	...	...	...	...	...	...	...
246	2018-12-24	973.90	1003.54	970.1100	976.22	1590328	GOOG
247	2018-12-26	989.01	1040.00	983.0000	1039.46	2373270	GOOG
248	2018-12-27	1017.15	1043.89	997.0000	1043.88	2109777	GOOG
249	2018-12-28	1049.62	1055.56	1033.1000	1037.08	1413772	GOOG
250	2018-12-31	1050.96	1052.70	1023.5900	1035.61	1493722	GOOG

1255 rows × 7 columns

Next steps: View recommended plots

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

```
# Converting the faang file into
faang.to_csv('faang.csv')
```

```
faang = pd.read_csv('data/faang.csv')
faang
```

	Unnamed: 0	date	open	high	low	close	volume	ticker
0	0	2018-01-02	177.68	181.58	177.5500	181.42	18151903	FB
1	1	2018-01-03	181.88	184.78	181.3300	184.67	16886563	FB
2	2	2018-01-04	184.90	186.21	184.0996	184.33	13880896	FB
3	3	2018-01-05	185.59	186.90	184.9300	186.85	13574535	FB
4	4	2018-01-08	187.20	188.90	186.3300	188.28	17994726	FB
...	...	...	...	...	...	...	...	...
1250	246	2018-12-24	973.90	1003.54	970.1100	976.22	1590328	GOOG
1251	247	2018-12-26	989.01	1040.00	983.0000	1039.46	2373270	GOOG
1252	248	2018-12-27	1017.15	1043.89	997.0000	1043.88	2109777	GOOG
1253	249	2018-12-28	1049.62	1055.56	1033.1000	1037.08	1413772	GOOG
1254	250	2018-12-31	1050.96	1052.70	1023.5900	1035.61	1493722	GOOG

1255 rows × 8 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.

```
# Converting the data type of date to datetime and volume to integer datatype
date = pd.to_datetime(faang.date),
volume=faang.volume.astype('int'),

faang.dtypes
```

```
Unnamed: 0              int64
date           datetime64[ns]
open                  float64
high                  float64
low                   float64
close                 float64
volume                  int64
ticker                 object
dtype: object
```

```
# Soring the date and ticker in the dataframe
faang.sort_values(by=['date','ticker'], inplace=True)
faang
```

	Unnamed: 0	date	open	high	low	close	volume	ticker
251	0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
502	0	2018-01-02	1172.0000	1190.0000	1170.5100	1189.0100	2694494	AMZN
0	0	2018-01-02	177.6800	181.5800	177.5500	181.4200	18151903	FB
1004	0	2018-01-02	1048.3400	1066.9400	1045.2300	1065.0000	1237564	GOOG
753	0	2018-01-02	196.1000	201.6500	195.4200	201.0700	10966889	NFLX
...	...	...	...	...	...	...	...	...
501	250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466	AAPL
752	250	2018-12-31	1510.8000	1520.7600	1487.0000	1501.9700	6954507	AMZN
250	250	2018-12-31	134.4500	134.6400	129.9500	131.0900	24625308	FB
1254	250	2018-12-31	1050.9600	1052.7000	1023.5900	1035.6100	1493722	GOOG
1003	250	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX

1255 rows × 8 columns

Next steps: View recommended plots

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

```
# Locating the first 7 rows of volume with the highest value
faang.sort_values(by='volume', ascending=False).head(7)
```

```
     Unnamed: 0        date      open      high       low     close     volume  ticker
142         142  2018-07-26  174.8900  180.1300  173.7500  176.2600  169803668      FB
53           53  2018-03-20  167.4700  170.2000  161.9500  168.1500  129851768      FB
57           57  2018-03-26  160.8200  161.1000  149.0200  160.0600  126116634      FB
54           54  2018-03-21  164.8000  173.4000  163.3000  169.3900  106598834      FB
433         182  2018-09-21  219.0727  219.6482  215.6097  215.9768   96246748    AAPL
496         245  2018-12-21  156.1901  157.4845  148.9909  150.0862   95744384    AAPL
463         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