Time-Series and Correlations with Stock Market Data using Python

I've recently created an account with IEX Cloud, a financial data service. As I've been learning the features of this new data source (new to me) and experimenting within my Jupyter Notebook, I thought the below may be helpful for others as well. Therefore, I'm creating my first Medium article and will focus it on financial time series data.

There are quite a few articles and sources on defining correlation, and the differences between correlation and causation; so what you will find below will primarily show some ways to test correlation and what the results mean. You may find this article beneficial if you're looking to use **IEX Cloud**, if you're looking to do Correlation tests in Python, and if you're interested in Time-Series data!

If you're following this and coding it yourself, go to https://iexcloud.io/ and get yourself an API key! You'll need it next! Also, don't forgot to install an IEX Python library: https://addisonlynch.github.io/iexfinance/ (Lynch, 2019). Install this library using:

$ pip3 install iexfinance

```
import config
import os

#os.environ['IEX_API_VERSION'] = "iexcloud-sandbox"
os.environ['IEX_TOKEN'] = config.iex_api_key  # Replace
"config.iex_api_key" with your API Key from IEX Cloud!
```

We'll load up some of the libraries we'll need next. Also, we'll be using data between January 1, 2017 and November 22, 2019.


```
from datetime import datetime
from iexfinance.stocks import get_historical_data
import matplotlib.pyplot as plt
%matplotlib inline

start = datetime(2017, 1, 1)
end = datetime(2019, 11, 22)
```

We'll **arbitrarily** choose 'close' for the sake of simplicity. From this API data response, you could also choose Open, High, Low, and Volume. We'll experiment with 'close.'

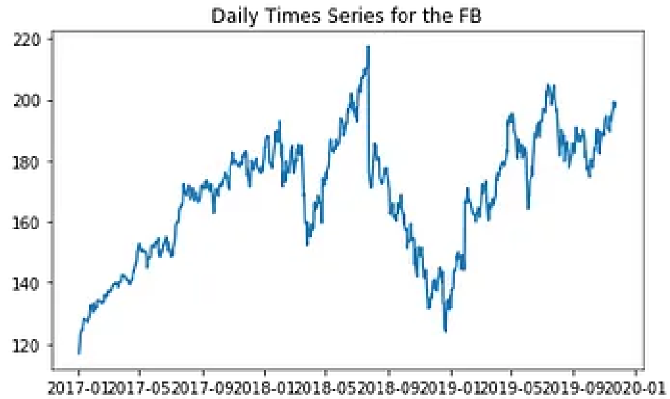
Now, let's make an API call and download more data, 'SPDR S&P 500 Trust ETF,' which tracks the S&P 500 (ticker: SPY). We'll be using this later in some correlation tests.

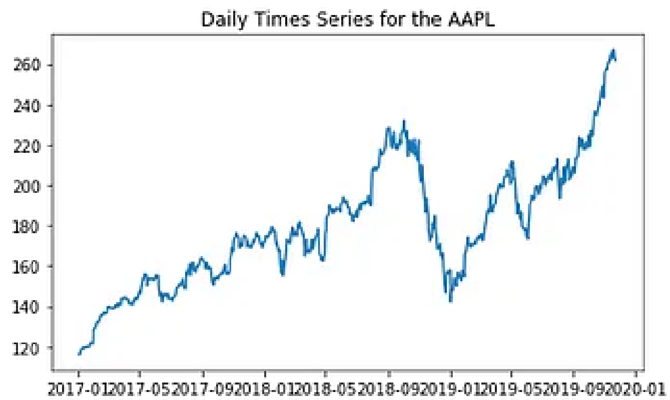
```
SPY = get_historical_data("SPY", start, end, output_format='pandas')

plt.figure(figsize=(10, 4))
plt.plot(SPY.index, SPY['close'])
plt.title('Daily Times Series for the SPY');
```

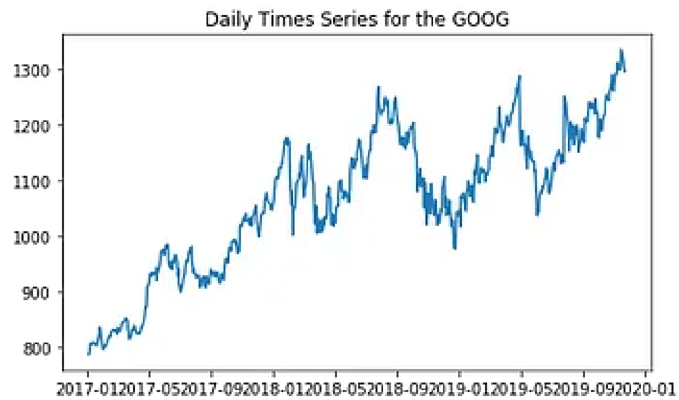
Now, let's continue to explore the API by downloading data for the FAANG stocks (Facebook, Amazon, Apple, Netflix, and Google) (Kenton, 2019), as well as add an interest of mine, Tesla. Also, chart these up. Pay careful attention to the charts and do a comparison of those with the SPY chart above. Some of these will look very similar to the SPY chart, some won't.

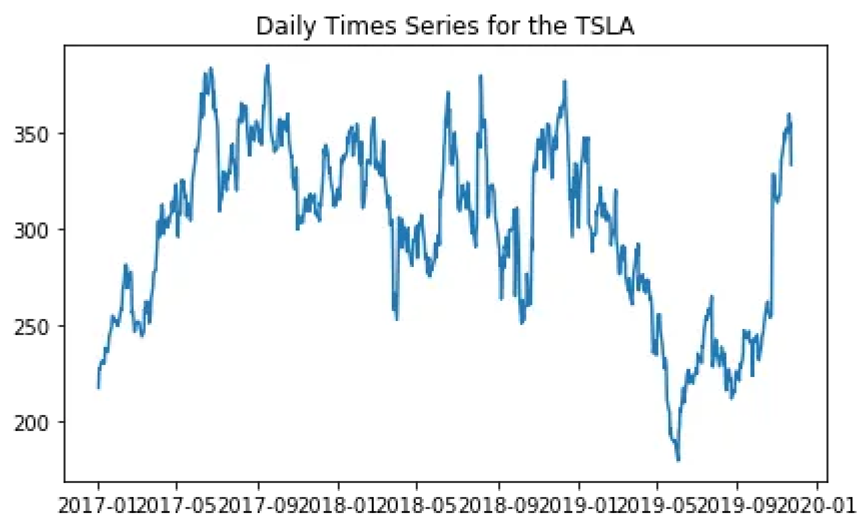
```
FAANGT =
get_historical_data(["FB","AMZN","AAPL","NFLX","GOOG","TSLA"], start,
end, output_format='pandas')
print(FAANGT.head())

FB                                              AMZN                \
              open    high     low   close    volume    open    high
low
date
2017-01-03  116.03  117.84  115.51  116.86  20663912  757.92  758.76
747.70
2017-01-04  117.55  119.66  117.29  118.69  19630932  758.39  759.68
754.20
2017-01-05  118.86  120.95  118.32  120.67  19492150  761.55  782.40
760.26
2017-01-06  120.98  123.88  120.03  123.41  28545263  782.36  799.44
778.48
2017-01-09  123.55  125.43  123.04  124.90  22880360  798.00  801.77
791.77

                               ...    GOOG
\
             close   volume   ...    open    high     low   close
volume
date                           ...
2017-01-03  753.67  3521066   ...  778.81  789.63  775.80  786.14
```

```
1657268
2017-01-04  757.18  2510526  ...  788.36  791.34  783.16  786.90
1072958
2017-01-05  780.45  5830068  ...  786.08  794.48  785.02  794.02
1335167
2017-01-06  795.99  5986234  ...  795.26  807.90  792.20  806.15
1640170
2017-01-09  796.92  3446109  ...  806.40  809.97  802.83  806.65
1274645

              TSLA
              open    high     low   close    volume
date
2017-01-03  214.86  220.33  210.96  216.99   5923254
2017-01-04  214.75  228.00  214.31  226.99  11213471
2017-01-05  226.42  227.48  221.95  226.75   5911695
2017-01-06  226.93  230.31  225.45  229.01   5527893
2017-01-09  228.97  231.92  228.00  231.28   3979484

[5 rows x 30 columns]

plt.figure(figsize=(15, 4))
#FB
plt.subplot(1, 2, 1)
plt.plot(FAANGT.index, FAANGT['FB']['close'])
plt.title('Daily Times Series for the FB')

#AMZN
plt.subplot(1, 2, 2)
plt.plot(FAANGT.index, FAANGT['AMZN']['close'])
plt.title('Daily Times Series for the AMZN');

plt.figure(figsize=(15, 4))
#AAPL
plt.subplot(1, 2, 1)
plt.plot(FAANGT.index, FAANGT['AAPL']['close'])
plt.title('Daily Times Series for the AAPL');

#GOOG
plt.subplot(1, 2, 2)
plt.plot(FAANGT.index, FAANGT['GOOG']['close'])
plt.title('Daily Times Series for the GOOG');

plt.figure(figsize=(15, 4))
#TSLA
plt.subplot(1, 2, 1)
plt.plot(FAANGT.index, FAANGT['TSLA']['close'])
plt.title('Daily Times Series for the TSLA');
```









Now that we have data for both the FAANG stocks (and TSLA) and the S&P 500, and we've plotted these so that we know what they look like; let's try an experiment! We're going to try a Pearson Correlation test, to test correlation on all of these equities and the S&P 500. What do you think? Based-on viewing the charts and going by intuition, will they correlate?

Correlation will show when the Pearson Correlation Coefficient is between -1 and +1. If closer to +1, we're seeing a positive correlation. If Pearson's correlation is closer to -1, a negative correlation (Cheong, 2019).

```
import pandas as pd
import scipy.stats as stats

# Slice this up to make it easier to work with.
indx = pd.IndexSlice
df1 = FAANGT.loc[:, (indx[:],'close')]

c, p = stats.pearsonr(df1['FB'].dropna()['close'], SPY.dropna()
['close'])
print(f"FB vs SPY Pearson Correlation: {c}\n")

c, p = stats.pearsonr(df1['AMZN'].dropna()['close'], SPY.dropna()
['close'])
print(f"AMZN vs SPY Pearson Correlation: {c}\n")

c, p = stats.pearsonr(df1['AAPL'].dropna()['close'], SPY.dropna()
['close'])
print(f"AAPL vs SPY Pearson Correlation: {c}\n")

c, p = stats.pearsonr(df1['GOOG'].dropna()['close'], SPY.dropna()
['close'])
print(f"GOOG vs SPY Pearson Correlation: {c}\n")

c, p = stats.pearsonr(df1['TSLA'].dropna()['close'], SPY.dropna()
['close'])
print(f"TSLA vs SPY Pearson Correlation: {c}")
FB vs SPY Pearson Correlation: 0.7325442525842248

AMZN vs SPY Pearson Correlation: 0.910899729798812

AAPL vs SPY Pearson Correlation: 0.9176098570966427

GOOG vs SPY Pearson Correlation: 0.9485878709468345

TSLA vs SPY Pearson Correlation: -0.26968006350226387
```

As of 11/22/2019, Google (GOOG) has the highest Pearson Correlation Coefficient out of all of these options. Also, Tesla (TSLA) has negative correlation to the S&P 500. Many of these you could find by looking in the charts above and comparing the charts with the S&P 500 chart; but now you have a quantitative approach for correlation!

Now, to explore the API a bit more, let's see how the **social sentiment** feature looks. We'll take a look at yesterday (11/22/2019), for Tesla.

```
from iexfinance.altdata import  get_social_sentiment,
get_ceo_compensation

period='minute'
specDay="20191122"

TSLA_Sent = get_social_sentiment("TSLA", period, specDay,
output_format='pandas')
print(TSLA_Sent.head())

minute  negative  positive  sentiment  totalScores
0   0000      0.12      0.88   0.084958           26
1   0001      0.12      0.88   0.160624           17
2   0002      0.11      0.89   0.061056           18
3   0003      0.29      0.71  -0.180071           17
4   0004      0.07      0.93   0.066293           15
```

For yesterday, what was the highest score for the most positive, and most negative, social sentiment of Tesla?

```
TSLA_Sent_Pos = TSLA_Sent['sentiment'].max()
TSLA_Sent_Neg = TSLA_Sent['sentiment'].min()

print("Highest Social Sentiment on 11/22/2019:", TSLA_Sent_Pos)
print("Lowest Social Sentiment on 11/22/2019:", TSLA_Sent_Neg)

Highest Social Sentiment on 11/22/2019: 0.9785
Lowest Social Sentiment on 11/22/2019: -0.9487
```

This API also has CEO information! Let's take a look at CEO information for the FAANG and TSLA stocks we researched earlier. We'll use df1 which was created and used to simplify when performing the correlation tests.

```
import pprint

for n, q in df1:
    pprint.pprint(get_ceo_compensation(n))
```

```
{'bonus': 0,
 'companyName': 'Facebook Inc. Class A',
 'location': 'Menlo Park, CA',
 'name': 'Mark Zuckerberg',
 'nonEquityIncentives': 0,
 'optionAwards': 0,
 'otherComp': 22554542,
 'pensionAndDeferred': 0,
 'salary': 1,
 'stockAwards': 0,
 'symbol': 'FB',
 'total': 22554543,
 'year': '2018'}
{'bonus': 0,
 'companyName': 'Amazon.com Inc.',
 'location': 'Seattle, WA',
 'name': 'Jeffrey Bezos',
 'nonEquityIncentives': 0,
 'optionAwards': 0,
 'otherComp': 1600000,
 'pensionAndDeferred': 0,
 'salary': 81840,
 'stockAwards': 0,
 'symbol': 'AMZN',
 'total': 1681840,
 'year': '2018'}
{'bonus': 0,
 'companyName': 'Apple Inc.',
 'location': 'Cupertino, CA',
 'name': 'Timothy Cook',
 'nonEquityIncentives': 12000000,
 'optionAwards': 0,
 'otherComp': 682219,
 'pensionAndDeferred': 0,
 'salary': 3000000,
 'stockAwards': 0,
 'symbol': 'AAPL',
 'total': 15682219,
 'year': '2018'}
{'bonus': 0,
 'companyName': 'Netflix Inc.',
 'location': 'Los Gatos, CA',
 'name': 'Reed Hastings',
 'nonEquityIncentives': 0,
 'optionAwards': 35380417,
 'otherComp': 0,
 'pensionAndDeferred': 0,
 'salary': 700000,
 'stockAwards': 0,
 'symbol': 'NFLX',
 'total': 36080417,
```

```
  'year': '2018'}
 {'bonus': 0,
  'companyName': 'Alphabet Inc. Class A',
  'location': 'Mountain View, CA',
  'name': 'Larry Page',
  'nonEquityIncentives': 0,
  'optionAwards': 0,
  'otherComp': 0,
  'pensionAndDeferred': 0,
  'salary': 1,
  'stockAwards': 0,
  'symbol': 'GOOG',
  'total': 1,
  'year': '2018'}
 {'bonus': 0,
  'companyName': 'Tesla Inc',
  'location': 'Palo Alto, CA',
  'name': 'Elon Musk',
  'nonEquityIncentives': 0,
  'optionAwards': 2283988504,
  'otherComp': 0,
  'pensionAndDeferred': 0,
  'salary': 56380,
  'stockAwards': 0,
  'symbol': 'TSLA',
  'total': 2284044884,
  'year': '2018'}
```


There's a lot more to explore and analyze. In the next article, I plan to explore some "real-time" data of IEX Cloud, such the books. Also, I may continue on with time-series analysis and move on to some basic forecasting. I hope you found this exploration useful!

References

- Kenton, W. (2019, November 18). What Are FAANG Stocks? Retrieved November 28, 2019, from https://www.investopedia.com/terms/f/faang-stocks.asp.

- Lynch, A. (2019, October 24). addisonlynch/iexfinance. Retrieved November 28, 2019, from https://github.com/addisonlynch/iexfinance.

- Cheong, J. H. (2019, May 13). Four ways to quantify synchrony between time series data. Retrieved November 28, 2019, from https://towardsdatascience.com/four-ways-to-quantify-synchrony-between-time-series-data-b99136c4a9c9.

Stock Market Finance Data Science Time Series Analysis Python

Sign up for The Variable

By Towards Data Science

Every Thursday, the Variable delivers the very best of Towards Data Science: from hands-on tutorials and cutting-edge research to original features you don't want to miss. Take a look.