Time Series Project

Author : THOMAS MORUGA OMBURA

Date: 17/3/2022

Resources

1. #### Clean Dataset

2. #### Submission Portal

If you are having problems please refer to this document:

1. #### Time Series Notebook

Instructions

Import all the libraries listed in the first cell. Make sure all modules are installed.

Use the provided data set to answer the following:

1. a) What is the lowest price for Safaricom (*SCOM*) b) What was the date when Safaricom had the lowest price?

2. a) What is the highest price Safaricom stock reached in the data b) What was the date when Safaricom stock recorded the highest price?

3. Create a line plot for Safaricom stock and verify if the information provided above is indeed correct.

4. Select **one** of the sectors provided (agric, comm, bank, const, energy, insur, invest, manu)

   a) Use **pandas** to create a subset containing all the rows of the dataframe and only companies in your selected sector. Rename this dataframe to the **sector_name_df**

   b) Using the subset for the sector, use **matplotlib** subplot to create subplots to fit all the sector stocks in one plot. One row can have a maximum of 3 charts.

   c) Using your sector DataFrame use the `corr()` DataFrame method to come up with a correlogram. Create a DataFrame for these correlations

   d) Use **Seaborn** to plot the **correlation plot** for your sector stocks.

Key performance Metrics:

- Go an extra step to produce charts that are visually appealing
- Ensure all the plots have a Title
- Ensure all plots have x labels and y labels where applicable
- Your plots should be clearly visible. Change the size of your plot to a comfortable width and height.

- Save all your plots

In [1]:
```
import os
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

Ensure that you have the ***clean_stock_prices.csv*** file in your working directory

In [2]:
```
os.listdir()
```

Out[2]:
```
['.ipynb_checkpoints',
 'Anaconda3-2021.11-Windows-x86_64.exe',
 'clean_stock_prices.csv',
 'desktop.ini',
 'MEASURING_CRIM1 (1).docx',
 'MEASURING_CRIM1.docx',
 'notebook-as-pdf-tutorial (1).pdf',
 'notebook-as-pdf-tutorial.pdf',
 'project-time-series-workbook (1).ipynb',
 'project-time-series-workbook (2).ipynb',
 'project-time-series-workbook (2).zip',
 'project-time-series-workbook (3).ipynb',
 'project-time-series-workbook (4).ipynb',
 'project-time-series-workbook (5).ipynb',
 'project-time-series-workbook (6).ipynb',
 'project-time-series-workbook (7).ipynb',
 'project-time-series-workbook.ipynb',
 'project2 - Copy (2) (1).py',
 'project2 - Copy (2).html',
 'project2 - Copy (2).pdf',
 'project2 - Copy (2).py',
 'pycharm-professional-2021.3.2.exe',
 'python-3.10.2-amd64.exe',
 'python-3.10.4-amd64.exe',
 'python-for-data-science-sample-submission.pdf',
 'python-installation (1) (1).pptx',
 'python-installation (1).pptx',
 'R Lesson 2.pptx',
 'R-and-R-Studio-Installation (1) (1).pdf',
 'R-and-R-Studio-Installation (1).pdf',
 'rrrrrr.pdf',
 'student_copy_pandas_workbook.ipynb',
 'th.jpg',
 'top-10-brands.png',
 'untitled',
 'Untitled.ipynb',
 'Untitled1.ipynb',
 'Untitled2.ipynb',
 'Untitled3.ipynb',
 'Untitled4.ipynb',
 'Untitled5.ipynb',
 'vehicle_data (1).csv',
 'vehicle_data.csv',
 'vehicle_dataset_project (1).ipynb',
 'vehicle_dataset_project (2).ipynb',
 'vehicle_dataset_project.ipynb',
 'vlc-3.0.16-win32.exe']
```

If you can see the *clean_stock_prices.csv* as an output in the above cell, read the data into a DataFrame using pandas

In [3]:
```
# read in the necessary file ('clean_stock_prices.csv')
df = pd.read_csv('clean_stock_prices.csv',index_col=0)
df.head()
```

Out[3]:

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2022-01-13	12.90	385.0	320.0	22.20	130.00	54.00	11.80	30.00	59.00	49.55	...	440.0	10.80	151.50	0.96	1.34
2022-01-11	13.80	385.0	320.0	20.55	134.75	44.75	11.90	30.75	59.50	52.00	...	445.0	10.85	161.00	0.88	1.31
2022-01-07	13.80	420.0	320.0	21.25	132.00	37.05	11.80	29.05	60.00	53.00	...	442.0	10.90	164.75	0.94	1.30
2022-01-06	13.80	420.0	320.0	20.25	130.75	33.70	11.80	29.30	60.00	53.00	...	442.0	10.90	160.75	0.99	1.29
2022-01-05	12.85	420.0	320.0	19.95	130.75	30.60	11.75	29.50	59.75	53.00	...	442.0	10.90	163.75	0.99	1.26

5 rows × 60 columns

In [4]:
```
df.tail()
```

Out[4]:

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2021-08-09	12.15	415.0	300.00	19.50	134.5	35.0	9.80	32.40	65.75	50.25	...	445.5	12.25	179.25	0.96	1.32
2021-08-06	12.15	415.0	300.00	20.00	134.5	35.0	9.80	32.40	65.75	50.00	...	454.0	12.25	179.00	0.98	1.32
2021-08-05	12.30	415.0	320.00	20.00	134.5	35.0	9.82	31.85	65.00	49.40	...	450.0	12.20	178.50	0.98	1.31
2021-08-04	12.00	415.0	320.00	19.95	135.0	35.0	9.76	29.75	64.00	49.10	...	455.0	12.00	179.75	0.98	1.30
2021-08-03	11.80	415.0	304.75	19.95	134.5	35.0	9.82	29.50	65.00	49.00	...	450.0	12.00	180.00	0.98	1.31

5 rows × 60 columns

Use this part to answer questions 1, 2 and 3

In [15]:
```
# lowest price for Safaricom
df.sort_values("SCOM")
```

Out[15]:

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2021-12-07	13.80	423.5	300.0	18.40	126.00	35.0	10.85	24.30	57.00	45.15	...	421.5	10.80	150.00	1.02	1.35
2021-12-08	13.80	423.5	300.0	18.40	126.00	35.0	10.95	26.35	56.50	46.95	...	421.5	10.80	150.50	1.02	1.35

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2021-12-09	13.80	423.5	300.0	18.85	126.00	31.5	10.95	26.35	58.00	47.05	...	430.0	10.80	149.50	1.02	1.35
2021-11-24	14.00	423.5	300.0	18.00	131.75	35.5	10.90	27.00	55.75	51.50	...	424.0	13.10	155.00	0.85	1.28
2021-11-25	14.00	423.5	300.0	18.00	131.75	35.5	10.90	27.00	56.25	51.50	...	424.0	12.90	153.50	0.85	1.30
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
2021-08-20	14.40	382.0	300.0	20.00	131.00	35.0	10.15	38.55	65.00	52.50	...	469.0	11.70	180.00	0.98	1.32
2021-08-25	13.25	415.0	300.0	19.85	133.50	35.0	10.55	38.55	65.25	53.00	...	467.0	11.85	175.75	0.95	1.32
2021-08-17	12.40	415.0	300.0	19.90	135.00	34.5	10.00	38.55	64.00	53.00	...	457.0	12.45	180.00	0.93	1.35
2021-08-24	13.25	415.0	300.0	20.00	132.75	35.0	10.55	38.55	66.75	53.00	...	466.0	11.60	178.50	0.99	1.32
2021-08-23	13.25	415.0	300.0	19.95	132.50	35.0	10.30	38.55	65.00	53.00	...	469.0	11.75	180.00	0.97	1.30

102 rows × 60 columns

In [16]:
```
df.sort_values("SCOM")
```

Out[16]:

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2021-12-07	13.80	423.5	300.0	18.40	126.00	35.0	10.85	24.30	57.00	45.15	...	421.5	10.80	150.00	1.02	1.35
2021-12-08	13.80	423.5	300.0	18.40	126.00	35.0	10.95	26.35	56.50	46.95	...	421.5	10.80	150.50	1.02	1.35
2021-12-09	13.80	423.5	300.0	18.85	126.00	31.5	10.95	26.35	58.00	47.05	...	430.0	10.80	149.50	1.02	1.35
2021-11-24	14.00	423.5	300.0	18.00	131.75	35.5	10.90	27.00	55.75	51.50	...	424.0	13.10	155.00	0.85	1.28
2021-11-25	14.00	423.5	300.0	18.00	131.75	35.5	10.90	27.00	56.25	51.50	...	424.0	12.90	153.50	0.85	1.30
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
2021-08-20	14.40	382.0	300.0	20.00	131.00	35.0	10.15	38.55	65.00	52.50	...	469.0	11.70	180.00	0.98	1.32
2021-08-25	13.25	415.0	300.0	19.85	133.50	35.0	10.55	38.55	65.25	53.00	...	467.0	11.85	175.75	0.95	1.32
2021-08-17	12.40	415.0	300.0	19.90	135.00	34.5	10.00	38.55	64.00	53.00	...	457.0	12.45	180.00	0.93	1.35
2021-08-24	13.25	415.0	300.0	20.00	132.75	35.0	10.55	38.55	66.75	53.00	...	466.0	11.60	178.50	0.99	1.32

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2021-08-23	13.25	415.0	300.0	19.95	132.50	35.0	10.30	38.55	65.00	53.00	...	469.0	11.75	180.00	0.97	1.30

102 rows × 60 columns

In [17]:
```
# highest price for Safaricom
df.sort_values(
    by="SCOM",
    ascending=False,
    kind="mergesort"
)
```

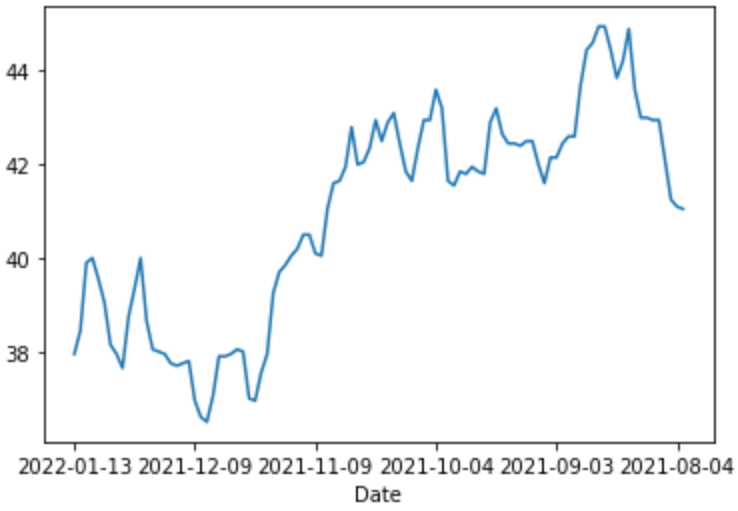
Out[17]:

Date	EGAD	KUKZ	LIMT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	EABL	EVRD	FTGH
2021-08-24	13.25	415.0	300.0	20.00	132.75	35.0	10.55	38.55	66.75	53.00	...	466.0	11.60	178.50	0.99	1.32
2021-08-23	13.25	415.0	300.0	19.95	132.50	35.0	10.30	38.55	65.00	53.00	...	469.0	11.75	180.00	0.97	1.30
2021-08-17	12.40	415.0	300.0	19.90	135.00	34.5	10.00	38.55	64.00	53.00	...	457.0	12.45	180.00	0.93	1.35
2021-08-25	13.25	415.0	300.0	19.85	133.50	35.0	10.55	38.55	65.25	53.00	...	467.0	11.85	175.75	0.95	1.32
2021-08-26	13.00	415.0	300.0	19.00	133.50	35.0	10.95	38.55	65.50	54.00	...	466.0	11.55	178.00	0.91	1.31
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2021-11-25	14.00	423.5	300.0	18.00	131.75	35.5	10.90	27.00	56.25	51.50	...	424.0	12.90	153.50	0.85	1.30
2021-12-09	13.80	423.5	300.0	18.85	126.00	31.5	10.95	26.35	58.00	47.05	...	430.0	10.80	149.50	1.02	1.35
2021-11-24	14.00	423.5	300.0	18.00	131.75	35.5	10.90	27.00	55.75	51.50	...	424.0	13.10	155.00	0.85	1.28
2021-12-08	13.80	423.5	300.0	18.40	126.00	35.0	10.95	26.35	56.50	46.95	...	421.5	10.80	150.50	1.02	1.35
2021-12-07	13.80	423.5	300.0	18.40	126.00	35.0	10.85	24.30	57.00	45.15	...	421.5	10.80	150.00	1.02	1.35

102 rows × 60 columns

In [18]:
```
# Plot SCOM to confirm above observations
df['SCOM'].plot()
```

Out[18]:
```
<AxesSubplot:xlabel='Date'>
```



Use this part to answer question 4

In [19]:

```
# agriculatural companies
agric = ['EGAD','KUKZ','LIMT','SASN','WTK']

# commercial companies
comm = ['XPRS','KQ','LKL','NBV', 'NMG','SMER','SCAN','SGL','TPSE','UCHM']

# banking companies
bank = ['ABSA','BKG','DTK','EQTY','HFCK','IMH', 'KCB','NBK', 'NCBA','SBIC','SCBK','COOP']

# construction sector
const = ['ARM','BAMB','CRWN','CABL','PORT']

# energy sector
energy = ['KEGN','KPLC','TOTL','UMME']

# insurance sector
insur = ['BRIT','CIC','JUB','KNRE','LBTY', 'SLAM']

# investement sector
invest = ['CTUM','HAFR','KURV','OCH','TCL','NSE']


# manufacturing sector
manu = ['BOC','BAT','CARB','EABL','EVRD','FTGH','ORCH', 'MSC','UNGA']
```

To subset a sector simply use the **slice** notation. For example if I choose the Insurance sector, i will use the **insur** list

In [23]:

```
energy_df = df.loc[:,'KEGN':'UMME'].copy()
energy_df.head()
```

Out[23]:

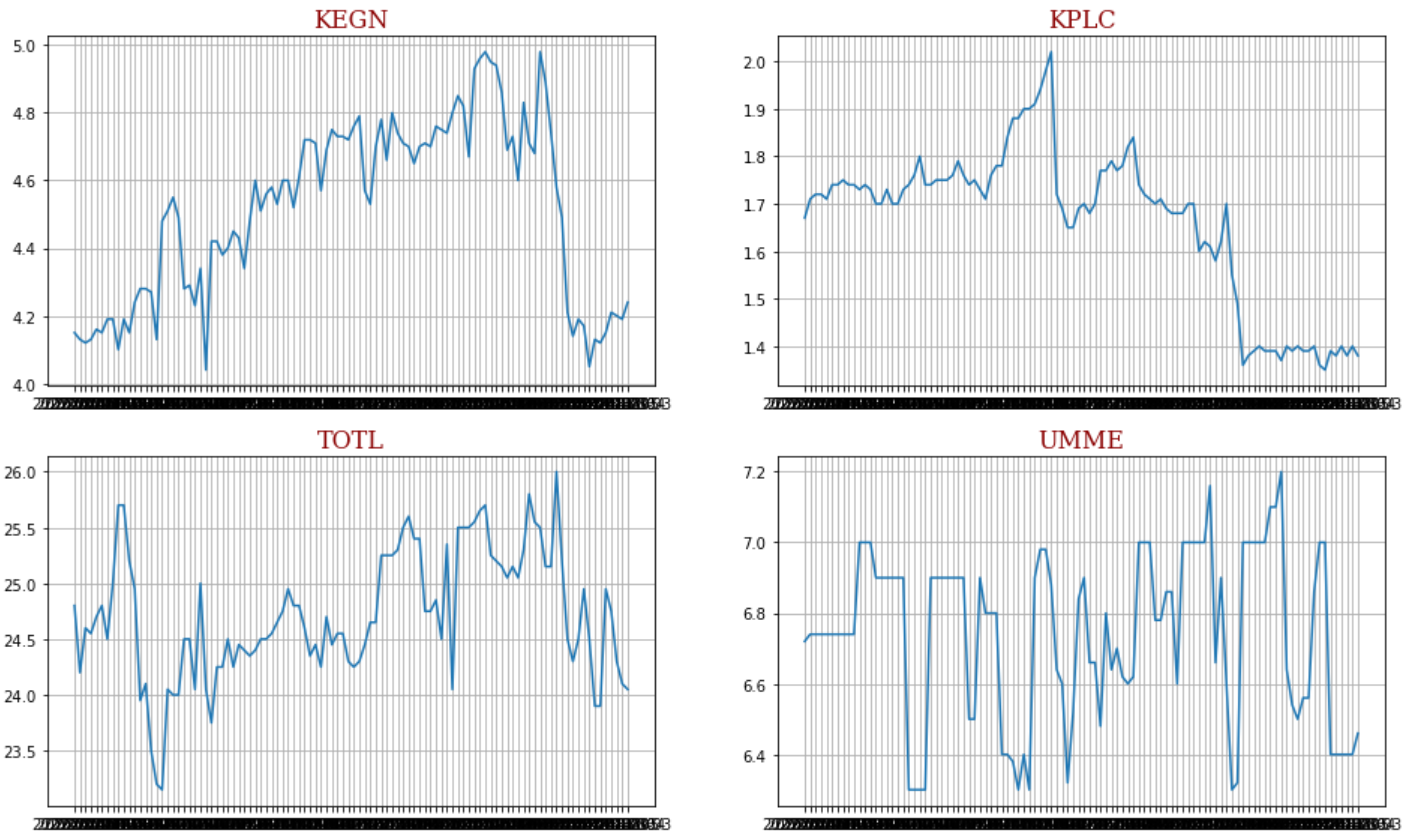
Date	KEGN	KPLC	TOTL	UMME
2022-01-13	4.15	1.67	24.80	6.72
2022-01-11	4.13	1.71	24.20	6.74
2022-01-07	4.12	1.72	24.60	6.74
2022-01-06	4.13	1.72	24.55	6.74
2022-01-05	4.16	1.71	24.70	6.74

In [36]:
```
energy_cols = energy_df.columns

font = {'family': 'serif',
        'color':  'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx,energy in enumerate(energy_cols,start=1):
    plt.subplot(6,2,idx)
    plt.title(energy,fontdict=font)
    plt.grid()
    plt.plot(energy,data=df)

fig = plt.gcf()
fig.set_size_inches(16,30)
plt.show()
```



In [37]:
```
energy_df.corr(method='pearson')
```

Out[37]:

	KEGN	KPLC	TOTL	UMME
KEGN	1.000000	0.104530	0.450142	0.206669
KPLC	0.104530	1.000000	-0.161972	-0.062976
TOTL	0.450142	-0.161972	1.000000	0.130735
UMME	0.206669	-0.062976	0.130735	1.000000

In [38]:
```
corr_df = energy_df.corr(method="pearson")
```

In [39]:
```
import seaborn as sns
```

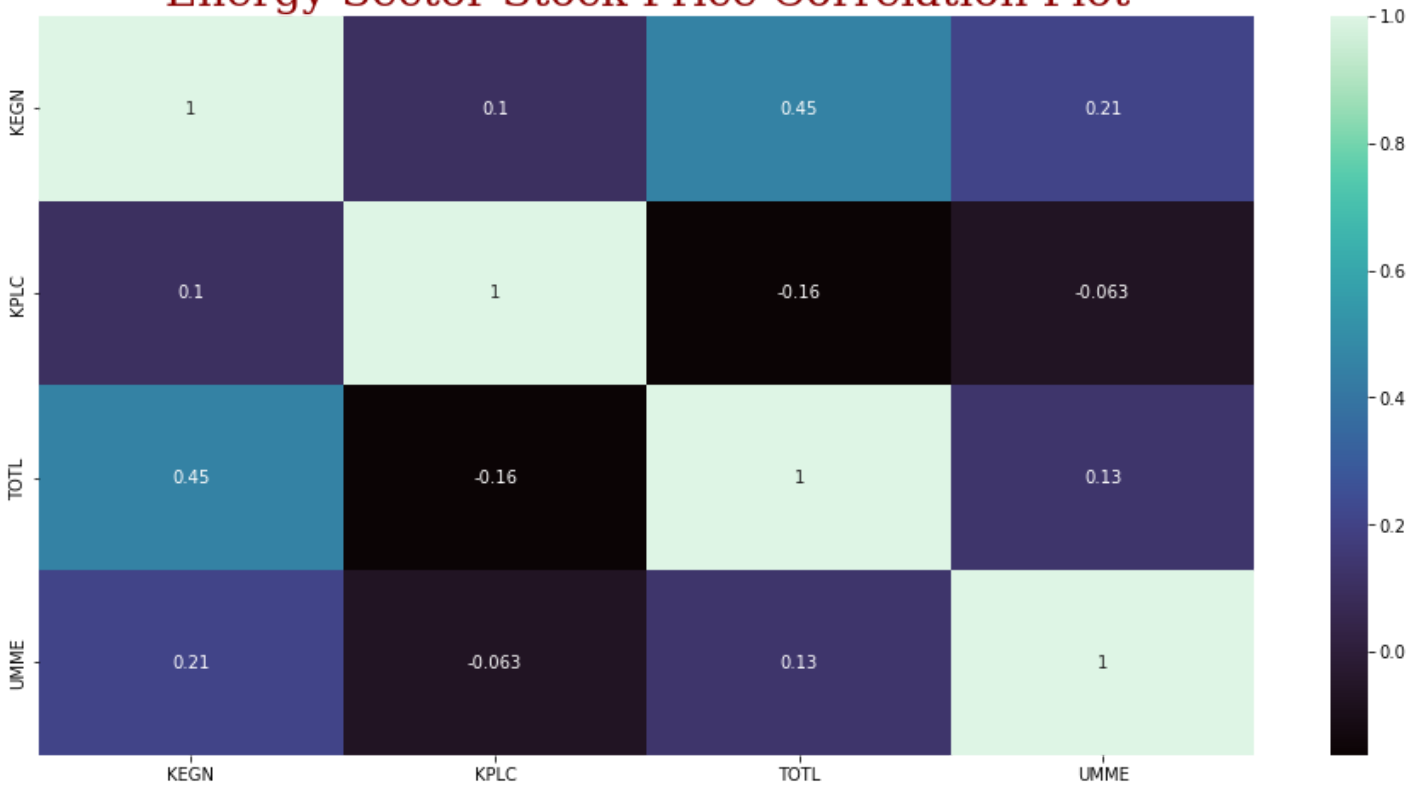
In [40]:
```
plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

Out[40]:
```
<Figure size 432x288 with 0 Axes>
```


```
<Figure size 432x288 with 0 Axes>
```

In [41]:
```
font = {'family': 'serif',
        'color':  'darkred',
        'weight': 'normal',
        'size': 26,
        }

plt.figure(figsize=(16,8))
plt.title("Energy Sector Stock Price Correlation Plot", fontdict=font)
cmap = ["mako","PiYG","YlGnBu","Blues"]
sns.heatmap(corr_df, annot=True, cmap=cmap[np.random.randint(len(cmap))])
plt.figure()
plt.show()
```




```
<Figure size 432x288 with 0 Axes>
```

In [ ]: