IMF

By Gerard Utoware

Table of Contents

Overview of the tool used

The main advantage of python in comparison to other data analytic tools such as excel, would be Python's ability to handle large volumes of data without hindering productivity. Running scripts using the libraries on allows for more automation and ease of obtaining analysed data. With this task, the data was imported from python which ensures that data will not be lost or tampered with while performing data analysis. After defining the columns from the imported data, it was less tedious to run scripts than to perform individual analysis on excel.

Overview of the libraries used

Numpy, pandas and matplotlib were used in the script. All three python libraries allow different functions to be performed. Pandas allows data analysis scripts to be conducted on python. It utilises two other libraries, being numpy and matplotlib. Matplotlib allows for data visualisation through a variety of charts such as bar and pie charts. Numpy permits mathematical operations in python. The combination of these three libraries is what allowed the following data and charts to be formed and analysed. In addition to these, seaborn was installed in order for a heatmap to be printed and shown.

Overview of the dataset

The dataset imported onto python tallied the prices of an array of commodities from the years 1990 to the current day. Certain commodities were not recorded at the start of the dataset time period, such as 'food and beverage', 'industrial input and 'agriculture price index'. There were a few years where the index was 0 for different industries such as PTOMATO, which was the lowest industry in the dataset. By contrast, the industry with the highest is PCOBA. The dataset, however, does not take into consideration external factors that affect the supply and demand of these commodities throughout the years. For example, overall weather by the month and year can severely affect the commodity price of agriculture. Transportation and geopolitics are another major factor that has an effect on the price of commodity. With that being said, higher priced indexes are still more valuable and wanted more so than lower commodities.

Q1&2

```python
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# Written to import the csv file and to show the columns in the dataset
df = pd.read_excel(r'external-datanovember.xls')
print(df)
df.info()

columnN = list(df.columns.values)
highest = 0
lowest = 0
worstC = ''
bestC = ''


def highLow(Cname, highest, bestC, lowest, worstC):
    total = 0
    years = df['Commodity']
    curYear = years[3][0:4]
    for i in range(3, len(years)):
        if curYear in years[i]:
            total = total + df[Cname][i]
```

```
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO

Process finished with exit code 0
```

```python
    for i in range(3, len(years)):
        if curYear in years[i]:
            total = total + df[Cname][i]
    else:
        if total > highest:
            highest = total
            bestC = Cname
            if worstC=='':
                lowest = total
                worstC = Cname
        else:
            if total < lowest:
                lowest = total
                worstC = Cname
                total = df[Cname][i]
                curYear = years[i][0:4]

    return highest, bestC, lowest, worstC


columnN.remove('Commodity')
df = df.iloc[3:]
for x in columnN:
```

```
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO

Process finished with exit code 0
```

```python
                worstC = Cname
        else:
            if total < lowest:
                lowest = total
                worstC = Cname
                total = df[Cname][i]
                curYear = years[i][0:4]

    return highest, bestC, lowest, worstC


columnN.remove('Commodity')
df = df.iloc[3:]
for x in columnN:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)

print(highest, bestC)
print(lowest,worstC)
```

```
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO

Process finished with exit code 0
```

Highest index was the PCOBA at 1211.998893773785. The lowest index was PTOMATO with 0.

Q3,4&5

```
                worstC = Cname
            else:
                if total < lowest:
                    lowest = total
                    worstC = Cname
                    total = df[Cname][i]
                    curYear = years[i][0:4]

    return highest, bestC, lowest, worstC


columnN.remove('Commodity')
df = df.iloc[3:]
for x in columnN:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)

print(highest, bestC)
print(lowest,worstC)

print(df[['PCOAL', 'PCOPP', 'PRUBB']].mean())
```

```
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO
PCOAL     100.144245
PCOPP    4631.063670
PRUBB      75.988382
dtype: float64

Process finished with exit code 0
```

The average of the 3 respective columns to get the average price index for coal, copper and rubber. Average of coal, copper and rubber are 100.14, 4631.06 and 75.99.

Q6&7

```
            lowest = total
            worstC = Cname


    return highest, bestC, lowest, worstC


columnN.remove('Commodity')
df = df.iloc[3:]
namescoff=['PCOFFROB', 'PCOFFOTM']
for x in namescoff:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

namesoil=['POILAPSP', 'POILBRE', 'POILDUB', 'POILWTI']
highest = 0
lowest = 0
for x in namesoil:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PCOAL', 'PCOPP', 'PRUBB']].mean())
```

```
dtypes: object(95)
memory usage: 294.8+ KB
1865.4258743647658 PCOFFOTM
1692.2381265068614 POILAPSP
PCOAL     100.144245
PCOPP    4631.063670
PRUBB      75.988382
dtype: float64

Process finished with exit code 0
```

By defining two different lists for the columns containing different coffee and crude oil, the best type for each respective list. The best coffee for 2022 was PCOFFOTM with 1865.43(2dp). The best oil was POILAPSP with 1692.24(2dp).

Further Data

```
            worstC = Cname
    else:
        if total < lowest:
            lowest = total
            worstC = Cname


    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesalum=['PALUM']
for x in namesalum:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)
```

```
Run: SpeedyWork (1)
 90  Unnamed: 90  128 non-null    object
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
20820.727853002067 PALUM

Process finished with exit code 0
```

The best financial month for Aluminium had an index value of 20820.73(2dp).

```
    else:
        if total < lowest:
            lowest = total
            worstC = Cname


    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesalum=['PALUM']
for x in namesalum:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

namessawnwood=['PSAWMAL', 'PSAWORE']
highest = 0
lowest = 0
for x in namessawnwood:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)
```

```
Run: SpeedyWork (1)
 88  Unnamed: 88  128 non-null    object
 89  Unnamed: 89  128 non-null    object
 90  Unnamed: 90  128 non-null    object
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
20820.727853002067 PALUM
4904.8910000000005 PSAWMAL
```

When looking at sawnwood, Hard sawnwood had the largest value out of both hard and and soft sawnwood, with a value of 4904.89(2dp).

```
    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesalum=['PALUM']
for x in namesalum:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

namessawnwood=['PSAWMAL', 'PSAWORE']
highest = 0
lowest = 0
for x in namessawnwood:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PALUM', 'PSAWMAL', 'PSAWORE']].mean())
```

```
dtypes: object(95)
memory usage: 294.8+ KB
20820.727853002067 PALUM
4904.8910000000005 PSAWMAL
PALUM      1804.348573
PSAWMAL     707.637750
PSAWORE     307.313475
dtype: float64

Process finished with exit code 0
```

The average of all the values in their respective columns, this helps to confirm that hard sawnwood is more valuable than soft sawnwood.

```
    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PWOOLC', 'PWOOLF']].mean())
```

```
 87  PAPPLE       397 non-null    object
 88  Unnamed: 88  128 non-null    object
 89  Unnamed: 89  128 non-null    object
 90  Unnamed: 90  128 non-null    object
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
3367.109502740262 PNGAS
```

Highest value natural gas, being 3367.11(2dp).

```
        if total < lowest:
            lowest = total
            worstC = Cname


    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS','PWOOLC', 'PWOOLF']].mean())
```

```
 88  Unnamed: 88  128 non-null    object
 89  Unnamed: 89  128 non-null    object
 90  Unnamed: 90  128 non-null    object
 91  Unnamed: 91  128 non-null    object
 92  Unnamed: 92  128 non-null    object
 93  Unnamed: 93  128 non-null    object
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
3367.109502740262 PNGAS
8576.9954975 PWOOLF
```

Fine wool has the highest value in the spreadsheet when compared to coarse wool. Highest value is 8576.9954975

```
        worstC = Cname


    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS','PWOOLC', 'PWOOLF']].mean())
```

```
8576.9954975 PWOOLF
PNGAS      142.559089
PNGASEU      6.684325
PNGASJP      8.614802
PNGASUS      3.806763
PWOOLC     706.334417
PWOOLF     927.112009
dtype: float64

Process finished with exit code 0
```

Average of their respective columns. Fine wool still has a higher average than its coarse counterpart.

```
            lowest = total
            worstC = Cname


    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS','PWOOLC', 'PWOOLF']].median())
```

```
Run: SpeedyWork (1)
3367.109502740262 PNGAS
8576.9954975 PWOOLF
PNGAS      123.390647
PNGASEU      4.260000
PNGASJP      6.451739
PNGASUS      3.033382
PWOOLC     607.981099
PWOOLF     853.430829
dtype: float64

Process finished with exit code 0
```

Median of their respective columns.

```
            worstC = Cname


    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS','PWOOLC', 'PWOOLF']].mode())
```

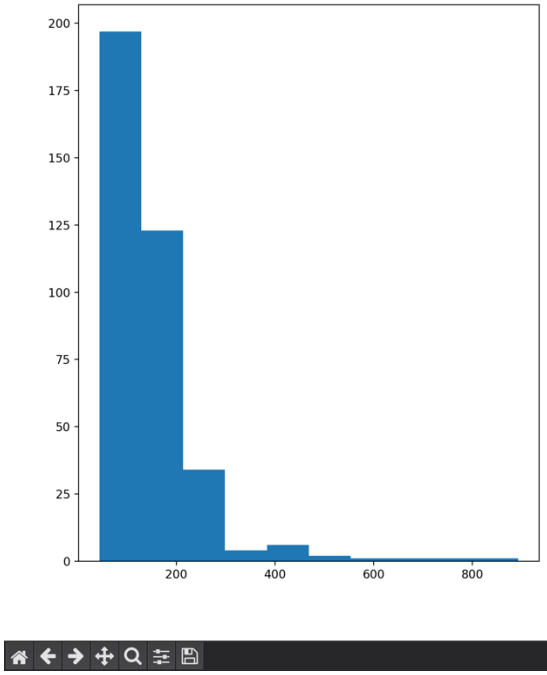
```
Run: SpeedyWork (1)
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
3367.109502740262 PNGAS
8576.9954975 PWOOLF
       PNGAS PNGASEU PNGASJP    PNGASUS      PWOOLC      PWOOLF
0  43.928331    2.69     3.5     1.1485   328.23938  596.018418
1  43.994589     NaN    3.82       1.18  418.442568         NaN
2  45.856918     NaN     NaN   1.251167         NaN         NaN
3  48.414884     NaN     NaN       1.26         NaN         NaN
4  48.514552     NaN     NaN     1.3028         NaN         NaN
```

Top 5 most common values in each column in the dataset. Only PNGAS has 5 different unique values. Other columns have ‘NaN’ which means that its Not A Number.

```
            worstC = Cname



    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS','PWOOLC', 'PWOOLF']].std())
```

```
Run: SpeedyWork (1)
3367.109502740262 PNGAS
8576.9954975 PWOOLF
PNGAS       98.802633
PNGASEU      7.071115
PNGASJP      6.956901
PNGASUS      2.174330
PWOOLC     350.159835
PWOOLF     328.399309
dtype: float64

Process finished with exit code 0
```

The standard deviation measures the spread of data in each column. The lower the standard deviation, the more the data points are to be found around the mean.




```
    return highest, bestC, lowest, worstC

#first column had to be removed as it was empty in the data sheet, then the
columnN.remove('Commodity')
df = df.iloc[3:]
namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
for x in namesgas:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, wors
print(highest,bestC)

nameswool=['PWOOLC', 'PWOOLF']
highest = 0
lowest = 0
for x in nameswool:
    highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, wors
print(highest,bestC)

print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS','PWOOLC', 'PWOOLF']].std
plt.hist(df['PNGAS'])
plt.show()
```

```
Run: SpeedyWork (1)
 94  Unnamed: 94  128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
3367.109502740262 PNGAS
8576.9954975 PWOOLF
PNGAS       98.802633
PNGASEU      7.071115
PNGASJP      6.956901
PNGASUS      2.174330
PWOOLC     350.159835
PWOOLF     328.399309
dtype: float64
```

Histogram of the PNGAS column to show where majority of the data points lay, being between 50-200.