UNIVERSITY OF MALAYA

MIDTERM EXAMINATION FOR THE DEGREE OF MASTER OF DATA SCIENCE

ACADEMIC SESSION 2019/2020 : SEMESTER II

WQD7005 : Data Mining

15 May 2020 TIME : 3 Hours

Name: Sidratul Muntaha
Matric Id : WQD180079/17199116

Step 1: Importing Libraries:

Pandas for data frame
Datetime for timestamp
Pandas_datareader for crawling
Numpy for dealing with numerical values and calculations
Matplotlib and seaborn is used for visualization and plotting.

```
##Importing Libraries:

import pandas as pd
import datetime as datetime
import pandas_datareader.data as web
from pandas import Series, DataFrame

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

Step 2: The below fragment of code is for crawling data from yahoo finance website, first a dictionary was created and as content the stock name of the companies were given. I selected a specific time period for collection.

```
# defining the company titles for scraping
companies_dict = {
 'Amazon': 'AMZN',
 'Apple': 'AAPL',}

companies = sorted(companies_dict.items(), key=lambda x: x[1])
```

```
#setting timeframe

start = datetime.datetime(2019, 1, 6)
end = datetime.datetime(2020,1,1)

#saving it in dataframe

df1 = web.get_data_yahoo(list(companies_dict.values()),start)
df1.to_csv('AMAZONAPPLE.csv',index=True,header= False) #saved in local s
```

Step 3: Checking for missing values

```
#Cleaning
print(df.isnull().sum())
```

Step 4: Merged two datasets, I crawled two datasets separately and then added them. Both of them contained companies and similar kind of data. I merged them by choosing the date as the common key column.

Data Integration

```
print(df1.columns)
print(df2.columns)
```

```
MultiIndex(levels=[['Adj Close', 'Close', 'High', 'Low', 'Open', 'Volum
e'], ['AAPL', 'AMZN']],
           codes=[[0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5], [0, 1, 0, 1, 0,
1, 0, 1, 0, 1, 0, 1]],
           names=['Attributes', 'Symbols'])
MultiIndex(levels=[['Adj Close', 'Close', 'High', 'Low', 'Open', 'Volum
e'], ['IBM', 'SNE']],
           codes=[[0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5], [0, 1, 0, 1, 0,
1, 0, 1, 0, 1, 0, 1]],
           names=['Attributes', 'Symbols'])
```

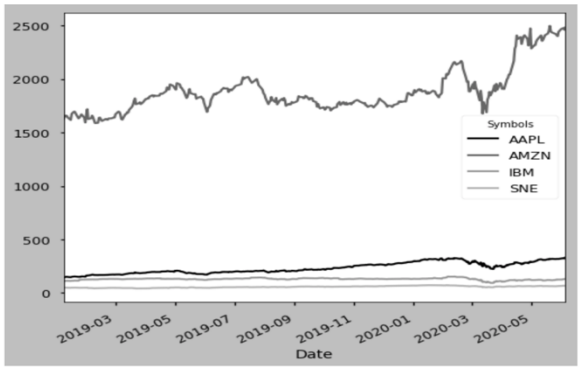
```
df = df1.merge(df2, left_index=True, right_on=['Date'])
```

Visualization :

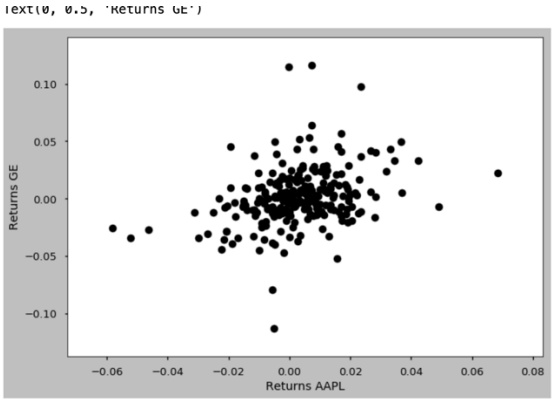
Comparing with other companies beyond the scraped companies :

```
plt.style.use( 'grayscale')

df["Adj Close"].plot(label='AAPL')
moving_avg.plot(label='moving average')
plt.legend()
```

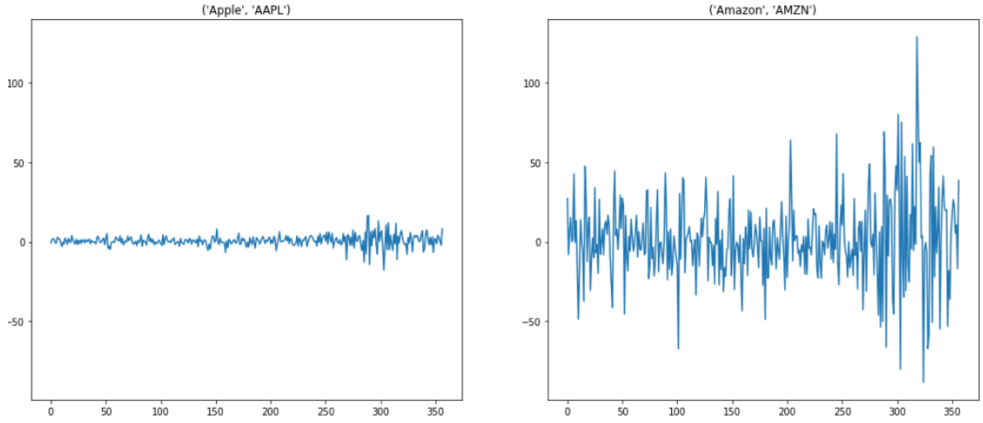
211]: <matplotlib.legend.Legend at 0x7fa19bb52940>



Step 5: This is the visualization of comparing two companies return values.The AAPL vs GE.



Step 6: Before normalizing the values of AMZN and AAPl gave visualization like below:



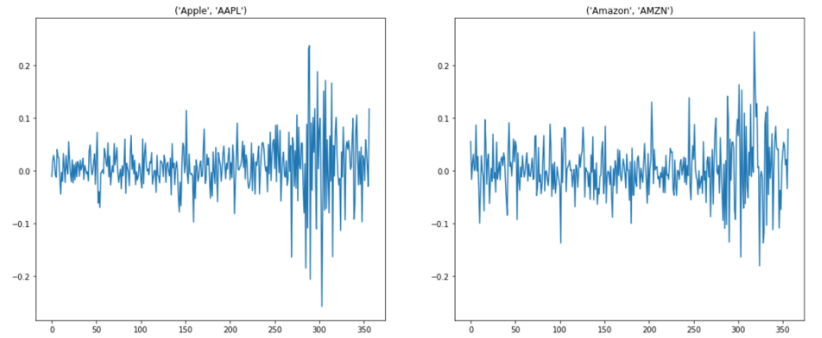
The normalizing code below :

```
# import Normalizer
from sklearn.preprocessing import Normalizer
# create the Normalizer
normalizer = Normalizer()

new = normalizer.fit_transform(movements)

print(new.max())
print(new.min())
print(new.mean())
```

```
0.3082122678663016
-0.32961590665128393
0.0032432759600199844
```

After normalizing :

```
plt.figure(figsize=(18,16))
ax1 = plt.subplot(221)
plt.plot(new[0][:])
plt.title(companies[0])

plt.subplot(222, sharey=ax1)
plt.plot(new[1][:])
plt.title(companies[1])
plt.show()
```



Applying Principle component Analysis:

```
# PCA
from sklearn.decomposition import PCA

# visualize the results
reduced_data = PCA(n_components = 2).fit_transform(new)

# run kmeans on reduced data
kmeans = KMeans(n_clusters=3)
kmeans.fit(reduced_data)
labels = kmeans.predict(reduced_data)

# create DataFrame aligning labels & companies
df = pd.DataFrame({'labels': labels, 'companies': companies})

# Display df sorted by cluster labels
print(df.sort_values('labels'))
```

```
   labels       companies
0       0   (Apple, AAPL)
1       0  (Amazon, AMZN)
3       1     (Sony, SNE)
2       2      (IBM, IBM)
```