By Nitish Adhikari

Email id :nitishbuzzpro@gmail.com (mailto:nitishbuzzpro@gmail.com) , +91-9650740295

Linkedin : https://www.linkedin.com/in/nitish-adhikari-6b2350248 (https://www.linkedin.com/in/nitish-adhikari-6b2350248)

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

In [2]:
```
path='D:\\*************\\individual_stocks_5yr'
company_list=['AAPL_data.csv', 'GOOG_data.csv', 'MSFT_data.csv','AMZN_data.csv']
all_data = pd.DataFrame()
for file in company_list:
    current_df = pd.read_csv(path+'/'+ file )
    all_data = pd.concat([all_data,current_df])
all_data.shape
```

Out[2]:
```
(4752, 7)
```

In [3]:
```
all_data.head()
```

Out[3]:

	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL

In [4]:
```
all_data['Name'].unique() #List of name unique stocks
```

Out[4]:
```
array(['AAPL', 'GOOG', 'MSFT', 'AMZN'], dtype=object)
```

In [5]:
```
all_data.dtypes
```

Out[5]:
```
date       object
open      float64
high      float64
low       float64
close     float64
volume      int64
Name       object
dtype: object
```

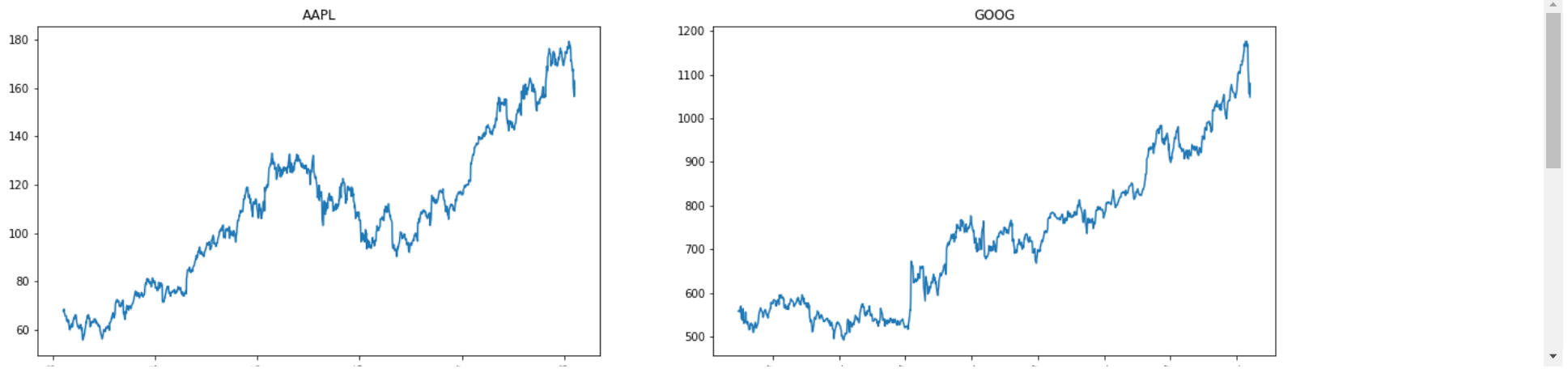
In [6]:
```
all_data['date'] = pd.to_datetime(all_data['date']) #convert date from string to datetime objects
```

In [7]:
```
all_data['date'].dtypes
```

Out[7]:
```
dtype('<M8[ns]')
```

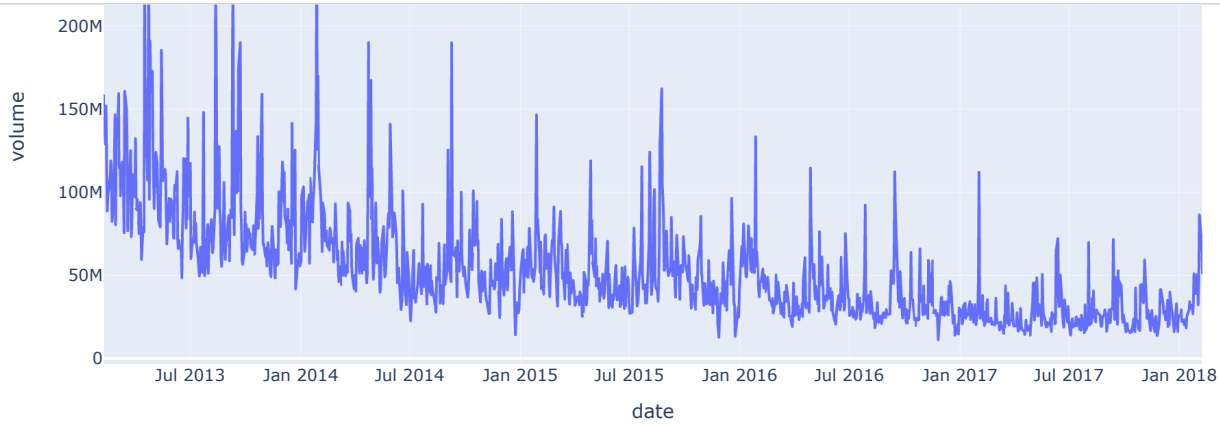
In [8]:
```
tech_list = ['AAPL', 'GOOG', 'MSFT', 'AMZN']
```

In [9]:
```
plt.figure(figsize=(20,12))
for i,company in enumerate(tech_list,1):
    plt.subplot(2,2,i)
    df=all_data[all_data['Name']==company]
    plt.plot(df['date'],df['close'])
    plt.xticks(rotation='vertical')
    plt.title(company)
```



In [10]:
```
import plotly.express as px
```

In [11]:
```
# Creating a plotly.express for volume vs date of stock
plt.figure(figsize=(15,10))
for company in tech_list:
    df = all_data[all_data['Name']==company]
    fig= px.line(df, x='date', y='volume', title=company)
    fig.show()
```



Analysing the daily price change in Apple stock

In [12]:
```
#Create Apple stock dataframe
df=pd.read_csv(r'D:\**********\individual_stocks_5yr/AAPL_data.csv')
df.head()
```

Out[12]:

	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL

In [13]:
```
#Adding Daily_Price_change feature to data frame
df['Daily_Price_change']=df['close']-df['open']
```

In [14]:
```
df.head()
```

Out[14]:

	date	open	high	low	close	volume	Name	Daily_Price_change
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL	0.1400
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL	0.4900
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL	-1.6586
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL	-0.0286
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL	0.2957

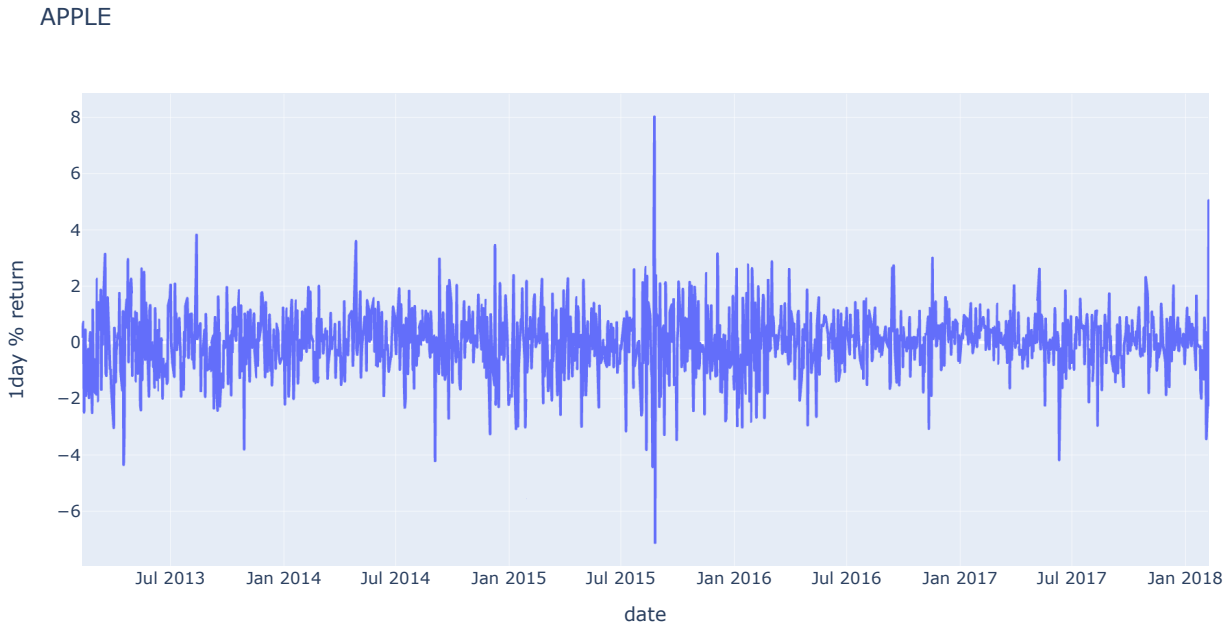
In [15]:
```
# adding '1day % return' feature to dataframe
df['1day % return']=((df['close']-df['open'])/df['close'])*100
```

In [16]:
```
df.head()
```

Out[16]:

	date	open	high	low	close	volume	Name	Daily_Price_change	1day % return
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL	0.1400	0.206325
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL	0.4900	0.714688
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL	-1.6586	-2.481344
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL	-0.0286	-0.042869
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL	0.2957	0.443624

In [17]:
```
# Creating a plotly.express for volume vs date for apple stock
fig= px.line(df, x='date', y='1day % return', title='APPLE')
fig.show()
```



ANALYSE MONTHLY MEAN OF CLOSE FEATURE

In [18]:
```
df2=df.copy() # creating a copy of df2
```

In [19]:
```
df2.dtypes
```

Out[19]:
```
date                 object
open                float64
high                float64
low                 float64
close               float64
volume                int64
Name                 object
Daily_Price_change  float64
1day % return       float64
dtype: object
```

In [20]:
```
#convert date to datetime object
df2['date']=pd.to_datetime(df2['date'])
```

In [21]:
```
df2.set_index('date',inplace=True)
```

In [22]:
```
df2.head()
```

Out[22]:

date	open	high	low	close	volume	Name	Daily_Price_change	1day % return
2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL	0.1400	0.206325
2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL	0.4900	0.714688
2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL	-1.6586	-2.481344
2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL	-0.0286	-0.042869
2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL	0.2957	0.443624

In [23]:
```
#Accessing date from  '2013-02-08'to'2013-02-14' by setting date as index
df2['2013-02-08':'2013-02-14']
```

Out[23]:

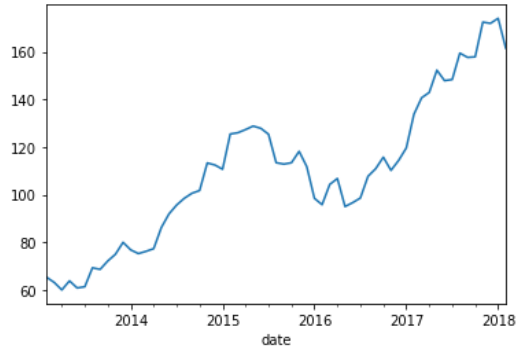
date	open	high	low	close	volume	Name	Daily_Price_change	1day % return
2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL	0.1400	0.206325
2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL	0.4900	0.714688
2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL	-1.6586	-2.481344
2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL	-0.0286	-0.042869
2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL	0.2957	0.443624

In [24]:
```
#resample 'close' by month and calculate mean
df2['close'].resample('M').mean()
```

Out[24]:
```
date
2013-02-28     65.306264
2013-03-31     63.120110
2013-04-30     59.966432
2013-05-31     63.778927
2013-06-30     60.791120
                  ...
2017-10-31    157.817273
2017-11-30    172.406190
2017-12-31    171.891500
2018-01-31    174.005238
2018-02-28    161.468000
Freq: M, Name: close, Length: 61, dtype: float64
```

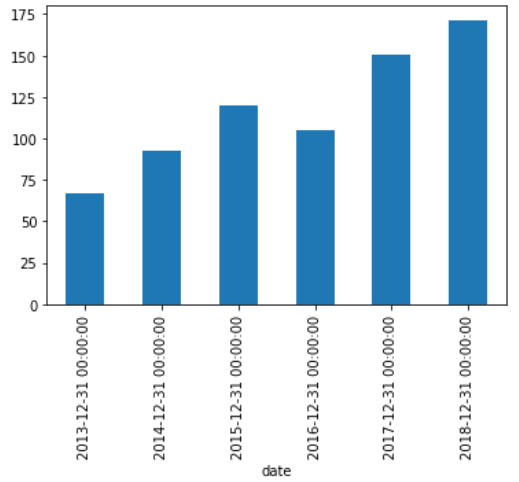
In [25]:
```
#plot - resample 'close' by month and calculate mean
df2['close'].resample('M').mean().plot()
#Conclusion from plot:- Monthly Mean Price is mostly increasing
```

Out[25]: <AxesSubplot:xlabel='date'>



In [26]:
```
#barplot - resample 'close' by year and calculate mean
df2['close'].resample('Y').mean().plot(kind='bar')
#Conclusion from plot:- Yearly Mean Price is mostly increasing
```

Out[26]: <AxesSubplot:xlabel='date'>



In [ ]:

Performing Multi-Variate Analysis

Analyse whether stock prices of these tech companies (Amazon,Apple,Google,Microsoft) are correlated or not

In [27]:
```
#Create 'aapl' dataframe from from 'AAPL_data.csv'
aapl=pd.read_csv(r'D:\*************\individual_stocks_5yr/AAPL_data.csv')
aapl.head()
```

Out[27]:

	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL

In [28]:
```
#Create 'amzn' dataframe from from 'AMZN_data.csv'
amzn=pd.read_csv(r'D:\*************\individual_stocks_5yr/AMZN_data.csv')
amzn.head()
```

Out[28]:

	date	open	high	low	close	volume	Name
0	2013-02-08	261.40	265.25	260.555	261.95	3879078	AMZN
1	2013-02-11	263.20	263.25	256.600	257.21	3403403	AMZN
2	2013-02-12	259.19	260.16	257.000	258.70	2938660	AMZN
3	2013-02-13	261.53	269.96	260.300	269.47	5292996	AMZN
4	2013-02-14	267.37	270.65	265.400	269.24	3462780	AMZN

In [29]:
```
#Create 'msft' dataframe from from 'MSFT_data.csv'
msft=pd.read_csv(r'D:\*************\individual_stocks_5yr/MSFT_data.csv')
msft.head()
```

Out[29]:

	date	open	high	low	close	volume	Name
0	2013-02-08	27.35	27.71	27.31	27.55	33318306	MSFT
1	2013-02-11	27.65	27.92	27.50	27.86	32247549	MSFT
2	2013-02-12	27.88	28.00	27.75	27.88	35990829	MSFT
3	2013-02-13	27.93	28.11	27.88	28.03	41715530	MSFT
4	2013-02-14	27.92	28.06	27.87	28.04	32663174	MSFT

In [30]:
```
#Create 'goog' dataframe from from 'GOOG_data.csv'
goog=pd.read_csv(r'D:\**********\individual_stocks_5yr/GOOG_data.csv')
goog.head()
```

Out[30]:

	date	open	high	low	close	volume	Name
0	2014-03-27	568.000	568.00	552.92	558.46	13052	GOOG
1	2014-03-28	561.200	566.43	558.67	559.99	41003	GOOG
2	2014-03-31	566.890	567.00	556.93	556.97	10772	GOOG
3	2014-04-01	558.710	568.45	558.71	567.16	7932	GOOG
4	2014-04-02	565.106	604.83	562.19	567.00	146697	GOOG

In [31]:
```
#creating a dataframe having 'close' price of (Amazon,Apple,Google,Microsoft) stocks
close = pd.DataFrame()
```

In [32]:
```
#Adding features to the 'close' dataframe from the stocks dataframes
close['aapl'] = aapl['close']
close['amzn'] = amzn['close']
close['msft'] = msft['close']
close['goog'] = goog['close']
```

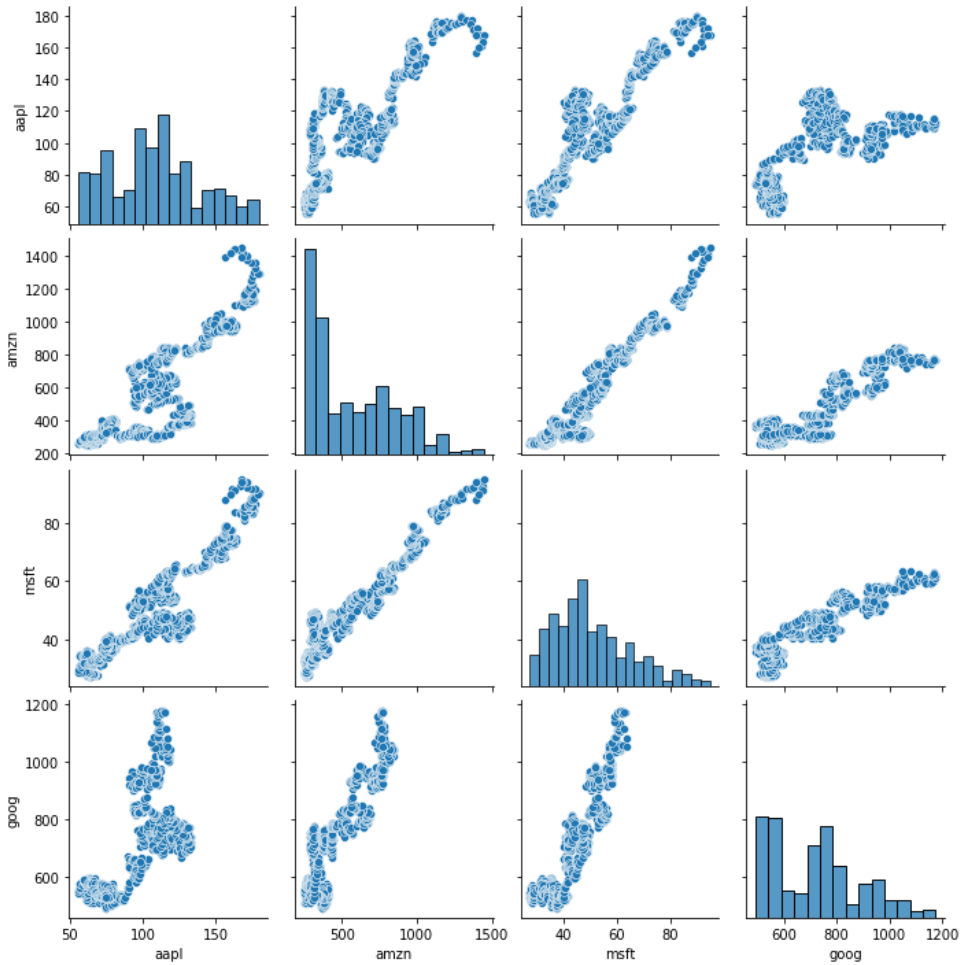
In [33]:
```
#check the dataframe
close.head()
```

Out[33]:

	aapl	amzn	msft	goog
0	67.8542	261.95	27.55	558.46
1	68.5614	257.21	27.86	559.99
2	66.8428	258.70	27.88	556.97
3	66.7156	269.47	28.03	567.16
4	66.6556	269.24	28.04	567.00

In [34]:
```
#create a pairplot of the 'close' dataframe
sns.pairplot(close)
```

Out[34]: <seaborn.axisgrid.PairGrid at 0x196b35ad570>



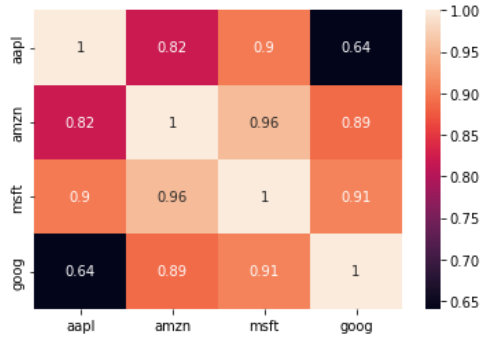
In [35]:
```
# Correlation in-between the close prices of the stock
close.corr()
```

Out[35]:

	aapl	amzn	msft	goog
aapl	1.000000	0.819078	0.899689	0.640522
amzn	0.819078	1.000000	0.955977	0.888456
msft	0.899689	0.955977	1.000000	0.907011
goog	0.640522	0.888456	0.907011	1.000000

In [36]:
```
#Creating heatmap of the correlation
sns.heatmap(close.corr(),annot=True)
#Conclusion :- Microsoft and Amazon close price seems to be highly corelated, Apple and google seems to least coorelated
```

Out[36]: <AxesSubplot:>



In [ ]:

Analyse daily returns of each stocks and how they are corelated

In [37]:
```
# create dataframe 'data' to store the % daily change of all stock prices
data =pd.DataFrame()
```

In [38]:
```
data['aapl_change']=((aapl['close']-aapl['open'])/aapl['close'])*100
data['amzn_change']=((amzn['close']-amzn['open'])/amzn['close'])*100
data['msft_change']=((msft['close']-msft['open'])/msft['close'])*100
data['goog_change']=((goog['close']-goog['open'])/goog['close'])*100
```

In [39]:
```
data.head()
```

Out[39]:

	aapl_change	amzn_change	msft_change	goog_change
0	0.206325	0.209964	0.725953	-1.708269
1	0.714688	-2.328836	0.753769	-0.216075
2	-2.481344	-0.189409	0.000000	-1.781065
3	-0.042869	2.946525	0.356761	1.489879
4	0.443624	0.694548	0.427960	0.334039

In [ ]:

Value at Risk Analysis for Tech Companies

#Create a distplot for appl_change feature sns.distplot(data['aapl_change'],kde=True) #conclusion from distplot:- It mostly follows a normal distribution

In [41]:
```
#Check for one standard deviation 'appl_change' feature
data['aapl_change'].std()
#conclusion:- one standard deviation 'appl_change' is as follows:-
```

Out[41]: 1.1871377131421237

68% of entire data lies within one standard deviation

In [42]:
```
#Check for two standard deviation 'appl_change' feature
data['aapl_change'].std()*2
#conclusion:- two standard deviation 'appl_change' is as follows:-
```

Out[42]: 2.3742754262842474

95% of entire data lies within two standard deviation

In [43]:
```
#Check for three standard deviation 'appl_change' feature
data['aapl_change'].std()*3
#conclusion:- three  standard deviation 'appl_change' is as follows:-
```

Out[43]: 3.561413139426371

99.7% of entire data lies within two standard deviation

In [ ]:

In [50]:
```
#Quantile
data['aapl_change'].quantile(0.1)
```

Out[50]: -1.4246644227944307

In [ ]:
```
### 90% of time the daily loss would not be more than above value
```

In [45]:
```
# checking fro all stocks
data.describe().T
```

Out[45]:

	count	mean	std	min	25%	50%	75%	max
aapl_change	1259.0	-0.000215	1.187138	-7.104299	-0.658021	0.042230	0.715427	8.000388
amzn_change	1259.0	-0.000398	1.358679	-9.363077	-0.738341	-0.002623	0.852568	5.640265
msft_change	1259.0	0.076404	1.059260	-5.177618	-0.509241	0.061069	0.703264	4.861491
goog_change	975.0	-0.012495	1.092560	-5.952266	-0.551963	0.024951	0.672649	4.943550

Complete!!