Stock_Market_Analysis_US_Stocks(Data Analysis Project)

May 21, 2024

[1]:
```
#!pip install --upgrade pandas
#!pip install --upgrade pandas-datareader
```

[2]:
```
#pip install yfinance
```

[3]:
```
import yfinance as yf
```

[4]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[5]:
```
#import pandas_datareader.data as web
```

[6]:
```
stock_list=['AAPL','MSFT', 'GOOG', 'AMZN']
```

[7]:
```
from datetime import datetime
end = datetime.now()
```

#1. Reading the Data from Yahoo finance website directly of each company

[8]:
```
end
```

[8]:
```
datetime.datetime(2024, 5, 4, 14, 53, 38, 897600)
```

[9]:
```
start_date = datetime(end.year-1,end.month,end.day)
```

[10]:
```
start_date
```

[10]:
```
datetime.datetime(2023, 5, 4, 0, 0)
```

[11]:
```
amzn_test = yf.download('AMZN',start=start_date,end=end,progress=False)
amzn_test
```

[11]:

Date	Open	High	Low	Close	Adj Close
2023-05-04	104.040001	105.389999	103.309998	104.000000	104.000000
2023-05-05	104.269997	105.760002	103.550003	105.660004	105.660004
2023-05-08	105.040001	106.099998	104.699997	105.830002	105.830002

```
2023-05-09  105.480003  106.790001  105.160004  106.620003  106.620003
2023-05-10  108.099998  110.669998  108.050003  110.190002  110.190002
…                 …           …           …           …           …
2024-04-29  182.750000  183.529999  179.389999  180.960007  180.960007
2024-04-30  181.089996  182.990005  174.800003  175.000000  175.000000
2024-05-01  181.639999  185.149994  176.559998  179.000000  179.000000
2024-05-02  180.850006  185.100006  179.910004  184.720001  184.720001
2024-05-03  186.815002  187.869995  185.429993  186.210007  186.210007

               Volume
Date
2023-05-04  45345500
2023-05-05  56912900
2023-05-08  49430900
2023-05-09  44089400
2023-05-10  78627600
…                 …
2024-04-29  54063900
2024-04-30  94639800
2024-05-01  94645100
2024-05-02  54303500
2024-05-03  38680749

[252 rows x 6 columns]
```

[12]:
```
aapl_test = yf.download('AAPL',start=start_date,end=end,progress=False)
aapl_test
```

[12]:
```
                 Open        High         Low       Close   Adj Close  \
Date
2023-05-04  164.889999  167.039993  164.309998  165.789993  164.909760
2023-05-05  170.979996  174.300003  170.759995  173.570007  172.648468
2023-05-08  172.479996  173.850006  172.110001  173.500000  172.578827
2023-05-09  173.050003  173.539993  171.600006  171.770004  170.858017
2023-05-10  173.020004  174.029999  171.899994  173.559998  172.638504
…                 …           …           …           …           …
2024-04-29  173.369995  176.029999  173.100006  173.500000  173.500000
2024-04-30  173.330002  174.990005  170.000000  170.330002  170.330002
2024-05-01  169.580002  172.710007  169.110001  169.300003  169.300003
2024-05-02  172.509995  173.419998  170.889999  173.029999  173.029999
2024-05-03  186.669998  187.000000  182.660004  183.380005  183.380005

                Volume
Date
2023-05-04   81235400
2023-05-05  113316400
2023-05-08   55962800
```

```
2023-05-09   45326900
2023-05-10   53724500
…               …
2024-04-29   68169400
2024-04-30   65934800
2024-05-01   50383100
2024-05-02   94214900
2024-05-03  157741757

[252 rows x 6 columns]
```

[13]:
```
msft_test = yf.download('MSFT',start=start_date,end=end,progress=False)
msft_test
```

[13]:
```
                 Open        High         Low       Close   Adj Close  \
Date
2023-05-04  306.239990  307.760010  303.399994  305.410004  302.923798
2023-05-05  305.720001  311.970001  304.269989  310.649994  308.121185
2023-05-08  310.130005  310.200012  306.089996  308.649994  306.137390
2023-05-09  308.000000  310.040009  306.309998  307.000000  304.500824
2023-05-10  308.619995  313.000000  307.670013  312.309998  309.767639
…                  …           …           …           …           …
2024-04-29  405.250000  406.320007  399.190002  402.250000  402.250000
2024-04-30  401.489990  402.160004  389.170013  389.329987  389.329987
2024-05-01  392.609985  401.720001  390.309998  394.940002  394.940002
2024-05-02  397.660004  399.929993  394.649994  397.839996  397.839996
2024-05-03  402.000000  407.149994  401.859985  406.660004  406.660004

               Volume
Date
2023-05-04  22519900
2023-05-05  28181200
2023-05-08  21318600
2023-05-09  21340800
2023-05-10  30078000
…               …
2024-04-29  19582100
2024-04-30  28781400
2024-05-01  23562500
2024-05-02  17709400
2024-05-03  17390522

[252 rows x 6 columns]
```

[14]:
```
goog_test = yf.download('GOOG',start=start_date,end=end,progress=False)
goog_test
```

[14]:

Date	Open	High	Low	Close	Adj Close	Volume
2023-05-04	106.160004	106.300003	104.699997	105.209999	105.209999	19780600
2023-05-05	105.320000	106.440002	104.738998	106.214996	106.214996	20705300
2023-05-08	105.794998	108.419998	105.790001	108.239998	108.239998	17266000
2023-05-09	108.779999	110.595001	107.724998	107.940002	107.940002	24782400
2023-05-10	108.550003	113.510002	108.480003	112.279999	112.279999	47533500
…	…	…	…	…	…	…
2024-04-29	170.770004	171.380005	167.059998	167.899994	167.899994	35914600
2024-04-30	167.380005	169.869995	164.500000	164.639999	164.639999	29420800
2024-05-01	166.179993	168.809998	164.899994	165.570007	165.570007	25223200
2024-05-02	166.669998	168.529999	165.690002	168.460007	168.460007	17041100
2024-05-03	169.526703	169.850006	164.979996	168.990005	168.990005	22716443

[252 rows x 6 columns]

#2. Add a column for each dataframe as company and add the company name

[15]:
```
amzn_test['Company'] = 'Amazon'
amzn_test
```

[15]:

Date	Open	High	Low	Close	Adj Close	Volume	Company
2023-05-04	104.040001	105.389999	103.309998	104.000000	104.000000	19780600	Amazon
2023-05-05	104.269997	105.760002	103.550003	105.660004	105.660004	20705300	Amazon
2023-05-08	105.040001	106.099998	104.699997	105.830002	105.830002	17266000	Amazon
2023-05-09	105.480003	106.790001	105.160004	106.620003	106.620003	24782400	Amazon
2023-05-10	108.099998	110.669998	108.050003	110.190002	110.190002	47533500	Amazon
…	…	…	…	…	…	…	…
2024-04-29	182.750000	183.529999	179.389999	180.960007	180.960007	35914600	Amazon
2024-04-30	181.089996	182.990005	174.800003	175.000000	175.000000	29420800	Amazon
2024-05-01	181.639999	185.149994	176.559998	179.000000	179.000000	25223200	Amazon
2024-05-02	180.850006	185.100006	179.910004	184.720001	184.720001	17041100	Amazon
2024-05-03	186.815002	187.869995	185.429993	186.210007	186.210007	22716443	Amazon

```
               Volume Company
Date
2023-05-04  45345500  Amazon
2023-05-05  56912900  Amazon
2023-05-08  49430900  Amazon
2023-05-09  44089400  Amazon
2023-05-10  78627600  Amazon
…              …      …
2024-04-29  54063900  Amazon
2024-04-30  94639800  Amazon
2024-05-01  94645100  Amazon
2024-05-02  54303500  Amazon
2024-05-03  38680749  Amazon

[252 rows x 7 columns]
```

[16]:
```
aapl_test['Company'] = 'Apple'
aapl_test
```

[16]:
```
                  Open        High         Low       Close   Adj Close  \
Date
2023-05-04  164.889999  167.039993  164.309998  165.789993  164.909760
2023-05-05  170.979996  174.300003  170.759995  173.570007  172.648468
2023-05-08  172.479996  173.850006  172.110001  173.500000  172.578827
2023-05-09  173.050003  173.539993  171.600006  171.770004  170.858017
2023-05-10  173.020004  174.029999  171.899994  173.559998  172.638504
…                 …           …           …           …           …
2024-04-29  173.369995  176.029999  173.100006  173.500000  173.500000
2024-04-30  173.330002  174.990005  170.000000  170.330002  170.330002
2024-05-01  169.580002  172.710007  169.110001  169.300003  169.300003
2024-05-02  172.509995  173.419998  170.889999  173.029999  173.029999
2024-05-03  186.669998  187.000000  182.660004  183.380005  183.380005

               Volume Company
Date
2023-05-04   81235400   Apple
2023-05-05  113316400   Apple
2023-05-08   55962800   Apple
2023-05-09   45326900   Apple
2023-05-10   53724500   Apple
…               …      …
2024-04-29   68169400   Apple
2024-04-30   65934800   Apple
2024-05-01   50383100   Apple
2024-05-02   94214900   Apple
2024-05-03  157741757   Apple
```

```
[252 rows x 7 columns]
```

[17]:
```
msft_test['Company'] = 'Microsoft'
msft_test
```

[17]:
```
                  Open        High         Low       Close   Adj Close  \
Date
2023-05-04  306.239990  307.760010  303.399994  305.410004  302.923798
2023-05-05  305.720001  311.970001  304.269989  310.649994  308.121185
2023-05-08  310.130005  310.200012  306.089996  308.649994  306.137390
2023-05-09  308.000000  310.040009  306.309998  307.000000  304.500824
2023-05-10  308.619995  313.000000  307.670013  312.309998  309.767639
…                 …           …           …           …           …
2024-04-29  405.250000  406.320007  399.190002  402.250000  402.250000
2024-04-30  401.489990  402.160004  389.170013  389.329987  389.329987
2024-05-01  392.609985  401.720001  390.309998  394.940002  394.940002
2024-05-02  397.660004  399.929993  394.649994  397.839996  397.839996
2024-05-03  402.000000  407.149994  401.859985  406.660004  406.660004

               Volume    Company
Date
2023-05-04  22519900  Microsoft
2023-05-05  28181200  Microsoft
2023-05-08  21318600  Microsoft
2023-05-09  21340800  Microsoft
2023-05-10  30078000  Microsoft
…                ...        …
2024-04-29  19582100  Microsoft
2024-04-30  28781400  Microsoft
2024-05-01  23562500  Microsoft
2024-05-02  17709400  Microsoft
2024-05-03  17390522  Microsoft

[252 rows x 7 columns]
```

[18]:
```
goog_test['Company'] = 'Google'
goog_test
```

[18]:
```
                  Open        High         Low       Close   Adj Close  \
Date
2023-05-04  106.160004  106.300003  104.699997  105.209999  105.209999
2023-05-05  105.320000  106.440002  104.738998  106.214996  106.214996
2023-05-08  105.794998  108.419998  105.790001  108.239998  108.239998
2023-05-09  108.779999  110.595001  107.724998  107.940002  107.940002
2023-05-10  108.550003  113.510002  108.480003  112.279999  112.279999
…                 …           …           …           …           …
```

```
2024-04-29  170.770004  171.380005  167.059998  167.899994  167.899994
2024-04-30  167.380005  169.869995  164.500000  164.639999  164.639999
2024-05-01  166.179993  168.809998  164.899994  165.570007  165.570007
2024-05-02  166.669998  168.529999  165.690002  168.460007  168.460007
2024-05-03  169.526703  169.850006  164.979996  168.990005  168.990005

              Volume Company
Date
2023-05-04  19780600  Google
2023-05-05  20705300  Google
2023-05-08  17266000  Google
2023-05-09  24782400  Google
2023-05-10  47533500  Google
…               …      …
2024-04-29  35914600  Google
2024-04-30  29420800  Google
2024-05-01  25223200  Google
2024-05-02  17041100  Google
2024-05-03  22716443  Google

[252 rows x 7 columns]
```

#3. Concatenate all the dataframes to convert into one dataframe

[19]:
```
conc=pd.concat([amzn_test,aapl_test,msft_test,goog_test])
```

[20]:
```
conc
```

[20]:
```
                  Open        High         Low       Close   Adj Close  \
Date
2023-05-04  104.040001  105.389999  103.309998  104.000000  104.000000
2023-05-05  104.269997  105.760002  103.550003  105.660004  105.660004
2023-05-08  105.040001  106.099998  104.699997  105.830002  105.830002
2023-05-09  105.480003  106.790001  105.160004  106.620003  106.620003
2023-05-10  108.099998  110.669998  108.050003  110.190002  110.190002
…                  …           …           …           …           …
2024-04-29  170.770004  171.380005  167.059998  167.899994  167.899994
2024-04-30  167.380005  169.869995  164.500000  164.639999  164.639999
2024-05-01  166.179993  168.809998  164.899994  165.570007  165.570007
2024-05-02  166.669998  168.529999  165.690002  168.460007  168.460007
2024-05-03  169.526703  169.850006  164.979996  168.990005  168.990005

              Volume Company
Date
2023-05-04  45345500  Amazon
2023-05-05  56912900  Amazon
2023-05-08  49430900  Amazon
2023-05-09  44089400  Amazon
```

```
2023-05-10  78627600  Amazon
…              …      …
2024-04-29  35914600  Google
2024-04-30  29420800  Google
2024-05-01  25223200  Google
2024-05-02  17041100  Google
2024-05-03  22716443  Google

[1008 rows x 7 columns]
```

#4. Save the dataframe as a csv file

[21]:
```
conc.to_csv("concat.csv")
```

#5. Check for null values, duplicate values

[22]:
```
conc.isnull().sum()
```

[22]:
```
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
Company      0
dtype: int64
```

[23]:
```
conc.duplicated().sum()
```

[23]:
```
0
```

#6a. Get the starting value & end value for each company

[24]:
```
print(amzn_test.head(1))
print(amzn_test.tail(1))
```

```
                Open        High         Low  Close  Adj Close    Volume  \
Date
2023-05-04  104.040001  105.389999  103.309998  104.0      104.0  45345500

           Company
Date
2023-05-04  Amazon
                Open        High         Low       Close   Adj Close  \
Date
2024-05-03  186.815002  187.869995  185.429993  186.210007  186.210007

              Volume Company
Date
2024-05-03  38680749  Amazon
```

[25]:
```
print(aapl_test.head(1))
print(aapl_test.tail(1))
```

```
                 Open        High         Low       Close  Adj Close  \
Date
2023-05-04  164.889999  167.039993  164.309998  165.789993  164.90976

              Volume Company
Date
2023-05-04  81235400   Apple
                 Open   High         Low       Close   Adj Close     Volume  \
Date
2024-05-03  186.669998  187.0  182.660004  183.380005  183.380005  157741757

           Company
Date
2024-05-03   Apple
```

[26]:
```
print(msft_test.head(1))
print(msft_test.tail(1))
```

```
                Open       High         Low       Close   Adj Close  \
Date
2023-05-04  306.23999  307.76001  303.399994  305.410004  302.923798

              Volume    Company
Date
2023-05-04  22519900  Microsoft
             Open        High         Low       Close   Adj Close    Volume  \
Date
2024-05-03  402.0  407.149994  401.859985  406.660004  406.660004  17390522

              Company
Date
2024-05-03  Microsoft
```

[27]:
```
print(goog_test.head(1))
print(goog_test.tail(1))
```

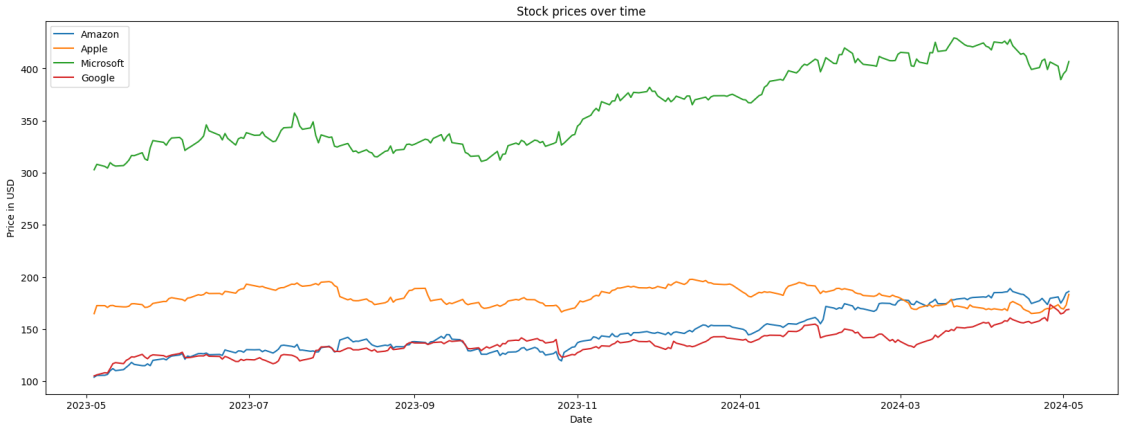
```
                 Open        High         Low       Close   Adj Close  \
Date
2023-05-04  106.160004  106.300003  104.699997  105.209999  105.209999

              Volume Company
Date
2023-05-04  19780600  Google
                 Open        High         Low       Close   Adj Close  \
Date
2024-05-03  169.526703  169.850006  164.979996  168.990005  168.990005
```

```
            Volume Company
Date
2024-05-03  22716443  Google
```

#6b. Create a line plot for each company - all the plots are in same figure region #Change in stock price over time

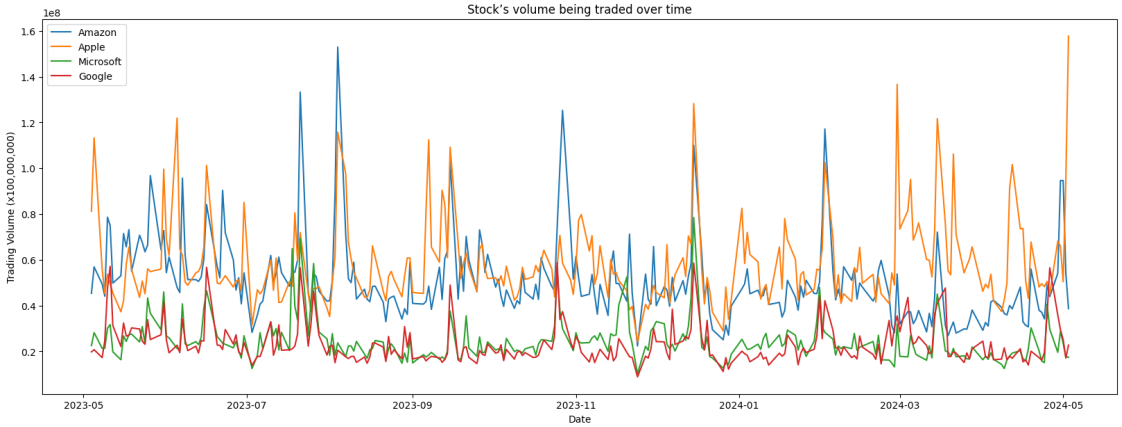
[28]:
```
from matplotlib.pyplot import figure
```

[29]:
```
plt.figure(figsize=(20, 7))
plt.plot(amzn_test['Adj Close'],label = "Amazon")
plt.plot(aapl_test['Adj Close'],label = "Apple")
plt.plot(msft_test['Adj Close'],label = "Microsoft")
plt.plot(goog_test['Adj Close'],label = "Google")
plt.xlabel("Date")
plt.ylabel("Price in USD")
plt.title("Stock prices over time")
plt.legend()
plt.show()
```



#7. Change in a stock's volume being traded over time

[30]:
```
plt.figure(figsize=(20, 7))
plt.plot(amzn_test['Volume'],label = "Amazon")
plt.plot(aapl_test['Volume'],label = "Apple")
plt.plot(msft_test['Volume'],label = "Microsoft")
plt.plot(goog_test['Volume'],label = "Google")
plt.xlabel("Date")
plt.ylabel("Trading Volume (x100,000,000)")
plt.title("Stock's volume being traded over time")
plt.legend()
plt.show()
```



#8. Moving average of various stocks

[31]:
```
amzn_test['MA for 20 Days'] = amzn_test['Adj Close'].rolling(20).mean()
amzn_test['MA for 50 Days'] = amzn_test['Adj Close'].rolling(50).mean()
amzn_test['MA for 100 Days'] = amzn_test['Adj Close'].rolling(100).mean()
```

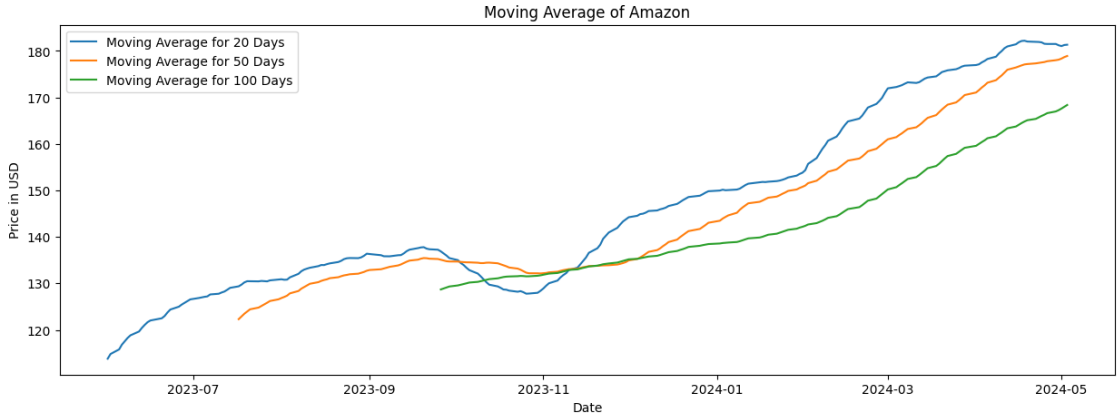
[32]:
```
amzn_test.tail(10)
```

[32]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days
2024-04-22	176.940002	178.869995	174.559998	177.229996	177.229996	37924900	Amazon	181.932502	177.3206	165.387900
2024-04-23	178.080002	179.929993	175.979996	179.539993	179.539993	37046500	Amazon	181.924001	177.4224	165.713000
2024-04-24	179.940002	180.320007	176.179993	176.589996	176.589996	34185100	Amazon	181.838501	177.5074	166.015699
2024-04-25	169.679993	173.919998	166.320007	173.669998	173.669998	49249400	Amazon	181.530501	177.6080	166.291499
2024-04-26	177.800003	180.820007	176.130005	179.619995	179.619995	43919800	Amazon	181.492500	177.7808	166.617399
2024-04-29	182.750000	183.529999	179.389999	180.960007	180.960007	54063900	Amazon	181.492001	178.0040	166.978600
2024-04-30	181.089996	182.990005	174.800003	175.000000	175.000000	94639800	Amazon	181.207500	178.1138	167.259799
2024-05-01	181.639999	185.149994	176.559998	179.000000	179.000000	94645100	Amazon	181.037000	178.3522	167.604599
2024-05-02	180.850006	185.100006	179.910004	184.720001	184.720001	54303500	Amazon	181.273000	178.6748	167.982999
2024-05-03	186.815002	187.869995	185.429993	186.210007	186.210007					

```
2024-05-03  38680749  Amazon      181.330000        178.9074       168.370900
```

[33]:
```
plt.figure(figsize=(15, 5))
plt.plot(amzn_test['MA for 20 Days'],label = "Moving Average for 20 Days")
plt.plot(amzn_test['MA for 50 Days'],label = "Moving Average for 50 Days")
plt.plot(amzn_test['MA for 100 Days'],label = "Moving Average for 100 Days")
plt.xlabel("Date")
plt.ylabel("Price in USD")
plt.title("Moving Average of Amazon")
plt.legend()
plt.show()
```



[34]:
```
aapl_test['MA for 20 Days'] = aapl_test['Adj Close'].rolling(20).mean()
aapl_test['MA for 50 Days'] = aapl_test['Adj Close'].rolling(50).mean()
aapl_test['MA for 100 Days'] = aapl_test['Adj Close'].rolling(100).mean()
```

[35]:
```
aapl_test.tail(10)
```

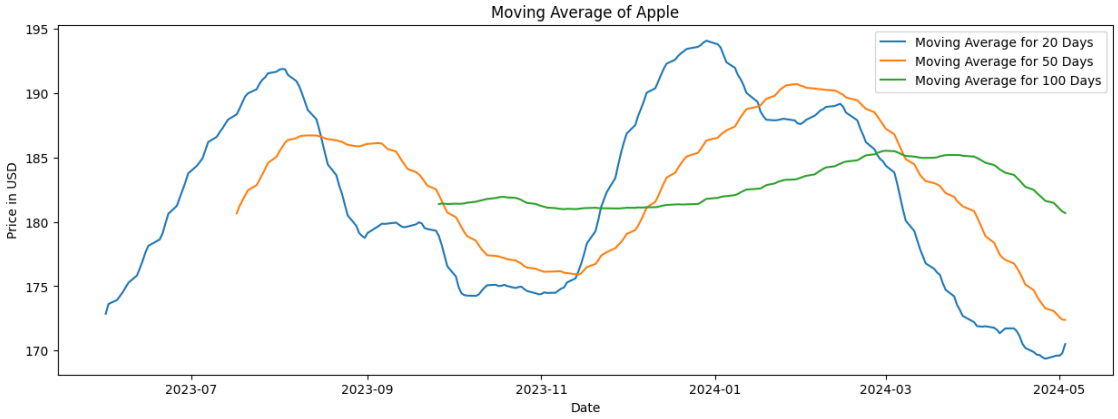
[35]:

Date	Open	High	Low	Close	Adj Close	\
2024-04-22	165.520004	167.259995	164.770004	165.839996	165.839996	
2024-04-23	165.350006	167.050003	164.919998	166.899994	166.899994	
2024-04-24	166.539993	169.300003	166.210007	169.020004	169.020004	
2024-04-25	169.529999	170.610001	168.149994	169.889999	169.889999	
2024-04-26	169.880005	171.339996	169.179993	169.300003	169.300003	
2024-04-29	173.369995	176.029999	173.100006	173.500000	173.500000	
2024-04-30	173.330002	174.990005	170.000000	170.330002	170.330002	
2024-05-01	169.580002	172.710007	169.110001	169.300003	169.300003	
2024-05-02	172.509995	173.419998	170.889999	173.029999	173.029999	
2024-05-03	186.669998	187.000000	182.660004	183.380005	183.380005	

Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days

Date					
2024-04-22	48116400	Apple	169.885500	174.687599	182.477606
2024-04-23	49537800	Apple	169.687999	174.248599	182.245033
2024-04-24	48251800	Apple	169.653499	173.885999	182.043946
2024-04-25	50558300	Apple	169.482499	173.583000	181.845767
2024-04-26	44838400	Apple	169.373499	173.286000	181.628804
2024-04-29	68169400	Apple	169.546999	173.078800	181.471918
2024-04-30	65934800	Apple	169.621500	172.839200	181.243483
2024-05-01	50383100	Apple	169.604000	172.594000	181.015734
2024-05-02	94214900	Apple	169.814500	172.408200	180.805809
2024-05-03	157741757	Apple	170.504500	172.388400	180.685004

[36]:
```
plt.figure(figsize=(15, 5))
plt.plot(aapl_test['MA for 20 Days'],label = "Moving Average for 20 Days")
plt.plot(aapl_test['MA for 50 Days'],label = "Moving Average for 50 Days")
plt.plot(aapl_test['MA for 100 Days'],label = "Moving Average for 100 Days")
plt.xlabel("Date")
plt.ylabel("Price in USD")
plt.title("Moving Average of Apple")
plt.legend()
plt.show()
```



[37]:
```
msft_test['MA for 20 Days'] = msft_test['Adj Close'].rolling(20).mean()
msft_test['MA for 50 Days'] = msft_test['Adj Close'].rolling(50).mean()
msft_test['MA for 100 Days'] = msft_test['Adj Close'].rolling(100).mean()
```

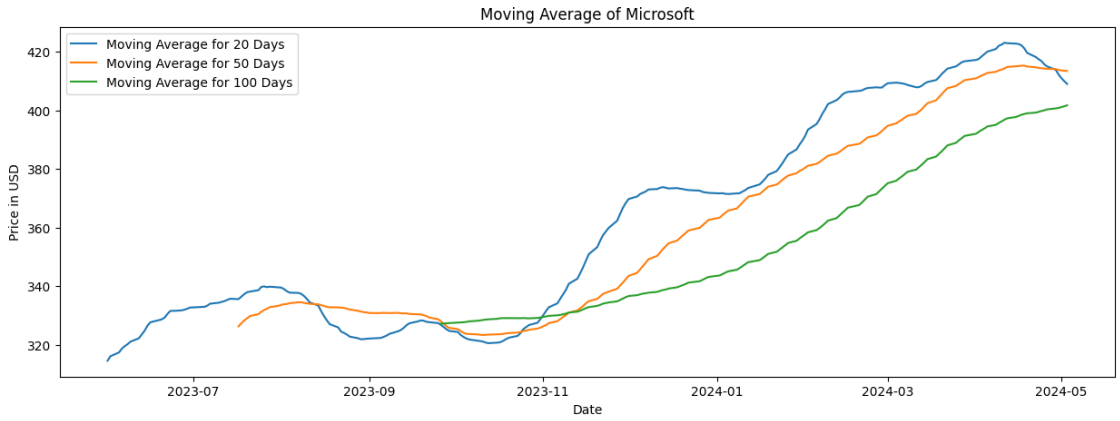
[38]:
```
msft_test.tail(10)
```

[38]:

Date	Open	High	Low	Close	Adj Close	\
2024-04-22	400.079987	402.850006	395.750000	400.959991	400.959991	
2024-04-23	404.239990	408.200012	403.059998	407.570007	407.570007	

```
2024-04-24  409.559998  412.470001  406.779999  409.059998  409.059998
2024-04-25  394.029999  399.890015  388.029999  399.040009  399.040009
2024-04-26  412.170013  413.000000  405.760010  406.320007  406.320007
2024-04-29  405.250000  406.320007  399.190002  402.250000  402.250000
2024-04-30  401.489990  402.160004  389.170013  389.329987  389.329987
2024-05-01  392.609985  401.720001  390.309998  394.940002  394.940002
2024-05-02  397.660004  399.929993  394.649994  397.839996  397.839996
2024-05-03  402.000000  407.149994  401.859985  406.660004  406.660004

              Volume    Company  MA for 20 Days  MA for 50 Days  \
Date
2024-04-22  20286900  Microsoft      418.244498      414.746144
2024-04-23  15734500  Microsoft      417.479999      414.502070
2024-04-24  15065300  Microsoft      416.850499      414.393400
2024-04-25  40586500  Microsoft      415.731000      414.262800
2024-04-26  29694700  Microsoft      415.011000      414.199401
2024-04-29  19582100  Microsoft      413.895000      414.113201
2024-04-30  28781400  Microsoft      412.289499      413.818600
2024-05-01  23562500  Microsoft      411.013998      413.661600
2024-05-02  17709400  Microsoft      410.011998      413.574800
2024-05-03  17390522  Microsoft      409.068999      413.475001

            MA for 100 Days
Date
2024-04-22       399.265914
2024-04-23       399.521678
2024-04-24       399.830771
2024-04-25       400.039065
2024-04-26       400.364078
2024-04-29       400.701992
2024-04-30       400.876968
2024-05-01       401.145175
2024-05-02       401.420923
2024-05-03       401.752130
```

[39]:
```
plt.figure(figsize=(15, 5))
plt.plot(msft_test['MA for 20 Days'],label = "Moving Average for 20 Days")
plt.plot(msft_test['MA for 50 Days'],label = "Moving Average for 50 Days")
plt.plot(msft_test['MA for 100 Days'],label = "Moving Average for 100 Days")
plt.xlabel("Date")
plt.ylabel("Price in USD")
plt.title("Moving Average of Microsoft")
plt.legend()
plt.show()
```



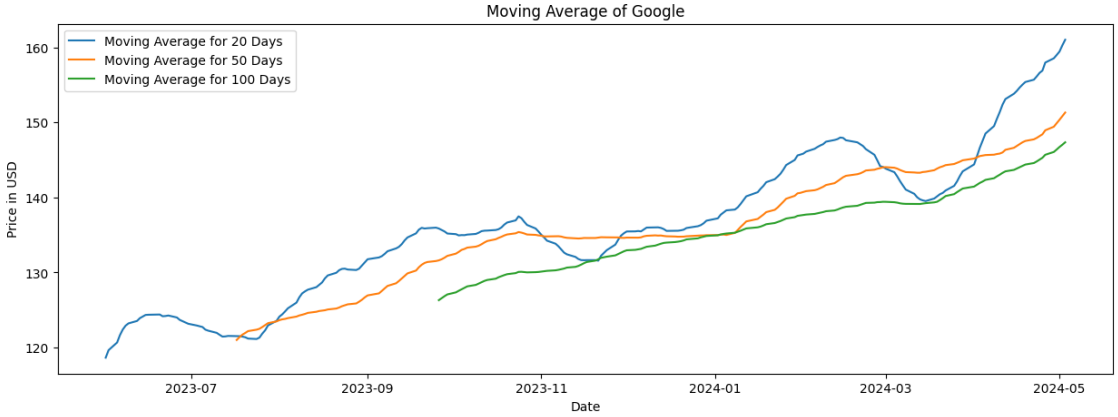
[40]:
```
goog_test['MA for 20 Days'] = goog_test['Adj Close'].rolling(20).mean()
goog_test['MA for 50 Days'] = goog_test['Adj Close'].rolling(50).mean()
goog_test['MA for 100 Days'] = goog_test['Adj Close'].rolling(100).mean()
```

[41]:
```
goog_test.tail(10)
```

[41]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days
2024-04-22	156.009995	159.184998	155.660004	157.949997	157.949997	17243900	Google	155.696500	147.747599	144.588999
2024-04-23	158.589996	160.479996	157.964996	159.919998	159.919998	16115400	Google	156.135000	147.941599	144.802000
2024-04-24	159.089996	161.389999	158.820007	161.100006	161.100006	19485700	Google	156.605000	148.188999	145.049000
2024-04-25	153.360001	158.279999	152.768005	157.949997	157.949997	36197800	Google	156.905500	148.420599	145.289300
2024-04-26	175.990005	176.419998	171.399994	173.690002	173.690002	56500800	Google	157.977000	148.951599	145.693000
2024-04-29	170.770004	171.380005	167.059998	167.899994	167.899994	35914600	Google	158.547000	149.430799	146.065699
2024-04-30	167.380005	169.869995	164.500000	164.639999	164.639999	29420800	Google	158.985500	149.888399	146.388199
2024-05-01	166.179993	168.809998	164.899994	165.570007	165.570007	25223200	Google	159.445501	150.355800	146.729600
2024-05-02	166.669998	168.529999	165.690002	168.460007	168.460007	17041100	Google	160.271501	150.848200	147.029700
2024-05-03	169.526703	169.850006	164.979996	168.990005	168.990005	22716443	Google	161.024001	151.321600	147.353200

[42]:
```
plt.figure(figsize=(15, 5))
plt.plot(goog_test['MA for 20 Days'],label = "Moving Average for 20 Days")
plt.plot(goog_test['MA for 50 Days'],label = "Moving Average for 50 Days")
plt.plot(goog_test['MA for 100 Days'],label = "Moving Average for 100 Days")
plt.xlabel("Date")
plt.ylabel("Price in USD")
plt.title("Moving Average of Google")
plt.legend()
plt.show()
```



#9.Daily return average of a stock

[43]:
```
amzn_test['Daily Return'] = amzn_test['Adj Close'].pct_change()
amzn_test.tail(10)
```

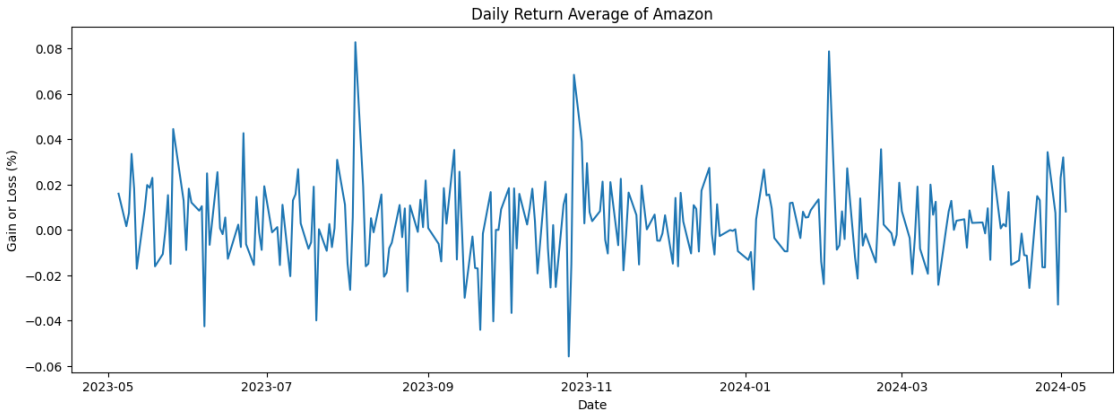
[43]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days
2024-04-22	176.940002	178.869995	174.559998	177.229996	177.229996	37924900	Amazon	181.932502	177.3206	165.387900
2024-04-23	178.080002	179.929993	175.979996	179.539993	179.539993	37046500	Amazon	181.924001	177.4224	165.713000
2024-04-24	179.940002	180.320007	176.179993	176.589996	176.589996					
2024-04-25	169.679993	173.919998	166.320007	173.669998	173.669998					
2024-04-26	177.800003	180.820007	176.130005	179.619995	179.619995					
2024-04-29	182.750000	183.529999	179.389999	180.960007	180.960007					
2024-04-30	181.089996	182.990005	174.800003	175.000000	175.000000					
2024-05-01	181.639999	185.149994	176.559998	179.000000	179.000000					
2024-05-02	180.850006	185.100006	179.910004	184.720001	184.720001					
2024-05-03	186.815002	187.869995	185.429993	186.210007	186.210007					

```
2024-04-24  34185100  Amazon      181.838501        177.5074        166.015699
2024-04-25  49249400  Amazon      181.530501        177.6080        166.291499
2024-04-26  43919800  Amazon      181.492500        177.7808        166.617399
2024-04-29  54063900  Amazon      181.492001        178.0040        166.978600
2024-04-30  94639800  Amazon      181.207500        178.1138        167.259799
2024-05-01  94645100  Amazon      181.037000        178.3522        167.604599
2024-05-02  54303500  Amazon      181.273000        178.6748        167.982999
2024-05-03  38680749  Amazon      181.330000        178.9074        168.370900
```

Date	Daily Return
2024-04-22	0.014889
2024-04-23	0.013034
2024-04-24	-0.016431
2024-04-25	-0.016535
2024-04-26	0.034260
2024-04-29	0.007460
2024-04-30	-0.032935
2024-05-01	0.022857
2024-05-02	0.031955
2024-05-03	0.008066

[44]:
```
plt.figure(figsize=(15, 5))
plt.plot(amzn_test['Daily Return'])
plt.xlabel("Date")
plt.ylabel("Gain or Loss (%)")
plt.title("Daily Return Average of Amazon")
plt.show()
```

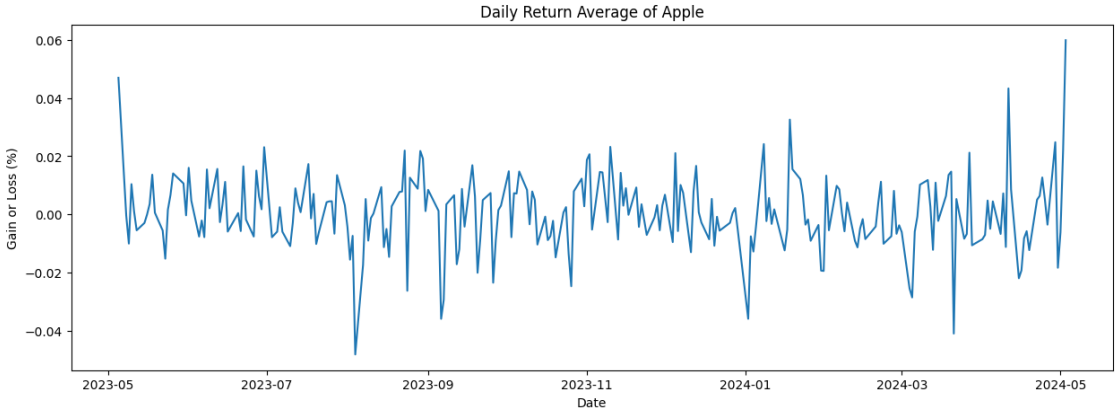


[45]:
```
aapl_test['Daily Return'] = aapl_test['Adj Close'].pct_change()
aapl_test.tail(10)
```

[45]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days	Daily Return
2024-04-22	165.520004	167.259995	164.770004	165.839996	165.839996	48116400	Apple	169.885500	174.687599	182.477606	0.005091
2024-04-23	165.350006	167.050003	164.919998	166.899994	166.899994	49537800	Apple	169.687999	174.248599	182.245033	0.006392
2024-04-24	166.539993	169.300003	166.210007	169.020004	169.020004	48251800	Apple	169.653499	173.885999	182.043946	0.012702
2024-04-25	169.529999	170.610001	168.149994	169.889999	169.889999	50558300	Apple	169.482499	173.583000	181.845767	0.005147
2024-04-26	169.880005	171.339996	169.179993	169.300003	169.300003	44838400	Apple	169.373499	173.286000	181.628804	-0.003473
2024-04-29	173.369995	176.029999	173.100006	173.500000	173.500000	68169400	Apple	169.546999	173.078800	181.471918	0.024808
2024-04-30	173.330002	174.990005	170.000000	170.330002	170.330002	65934800	Apple	169.621500	172.839200	181.243483	-0.018271
2024-05-01	169.580002	172.710007	169.110001	169.300003	169.300003	50383100	Apple	169.604000	172.594000	181.015734	-0.006047
2024-05-02	172.509995	173.419998	170.889999	173.029999	173.029999	94214900	Apple	169.814500	172.408200	180.805809	0.022032
2024-05-03	186.669998	187.000000	182.660004	183.380005	183.380005	157741757	Apple	170.504500	172.388400	180.685004	0.059816

[46]:
```
plt.figure(figsize=(15, 5))
plt.plot(aapl_test['Daily Return'])
plt.xlabel("Date")
plt.ylabel("Gain or Loss (%)")
plt.title("Daily Return Average of Apple")
plt.show()
```



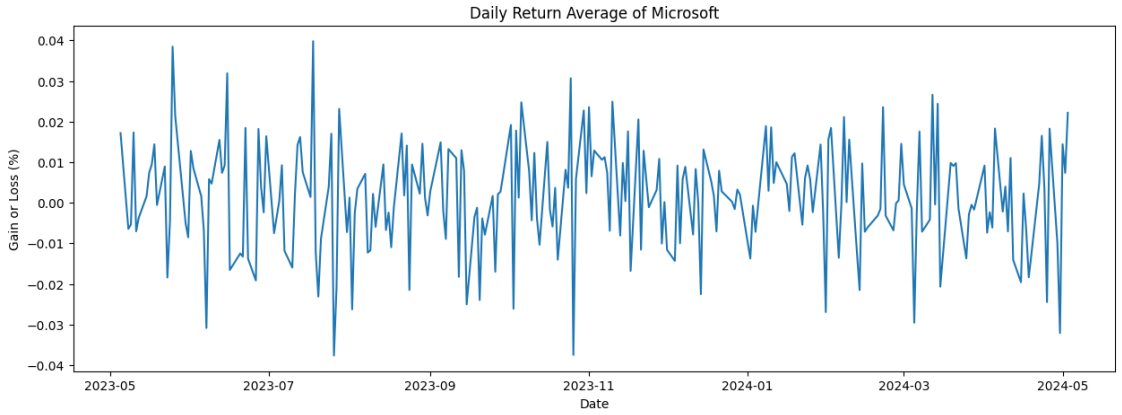
[47]:
```
msft_test['Daily Return'] = msft_test['Adj Close'].pct_change()
msft_test.tail(10)
```

[47]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days	Daily Return
2024-04-22	400.079987	402.850006	395.750000	400.959991	400.959991	20286900	Microsoft	418.244498	414.746144	399.265914	0.004610
2024-04-23	404.239990	408.200012	403.059998	407.570007	407.570007	15734500	Microsoft	417.479999	414.502070		
2024-04-24	409.559998	412.470001	406.779999	409.059998	409.059998	15065300	Microsoft	416.850499	414.393400		
2024-04-25	394.029999	399.890015	388.029999	399.040009	399.040009	40586500	Microsoft	415.731000	414.262800		
2024-04-26	412.170013	413.000000	405.760010	406.320007	406.320007	29694700	Microsoft	415.011000	414.199401		
2024-04-29	405.250000	406.320007	399.190002	402.250000	402.250000	19582100	Microsoft	413.895000	414.113201		
2024-04-30	401.489990	402.160004	389.170013	389.329987	389.329987	28781400	Microsoft	412.289499	413.818600		
2024-05-01	392.609985	401.720001	390.309998	394.940002	394.940002	23562500	Microsoft	411.013998	413.661600		
2024-05-02	397.660004	399.929993	394.649994	397.839996	397.839996	17709400	Microsoft	410.011998	413.574800		
2024-05-03	402.000000	407.149994	401.859985	406.660004	406.660004	17390522	Microsoft	409.068999	413.475001		

```
2024-04-23       399.521678      0.016485
2024-04-24       399.830771      0.003656
2024-04-25       400.039065     -0.024495
2024-04-26       400.364078      0.018244
2024-04-29       400.701992     -0.010017
2024-04-30       400.876968     -0.032119
2024-05-01       401.145175      0.014409
2024-05-02       401.420923      0.007343
2024-05-03       401.752130      0.022170
```

[48]:
```
plt.figure(figsize=(15, 5))
plt.plot(msft_test['Daily Return'])
plt.xlabel("Date")
plt.ylabel("Gain or Loss (%)")
plt.title("Daily Return Average of Microsoft")
plt.show()
```



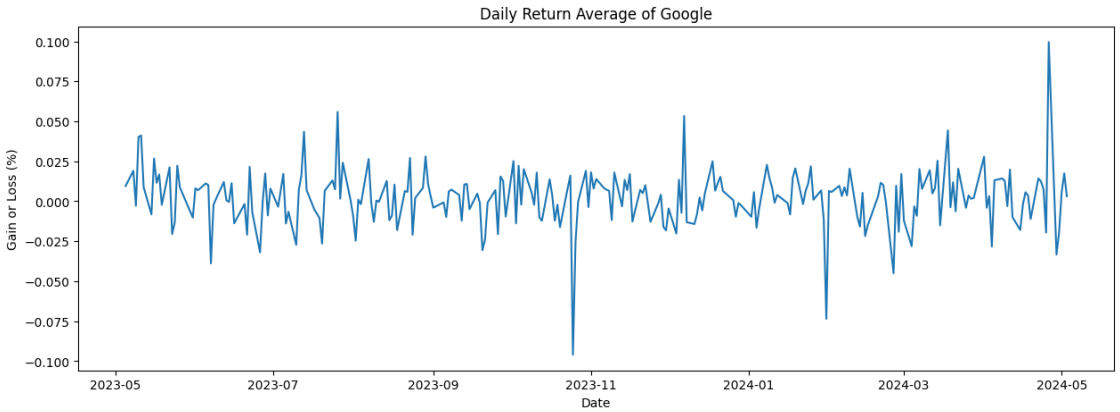
[49]:
```
goog_test['Daily Return'] = goog_test['Adj Close'].pct_change()
goog_test.tail(10)
```

[49]:

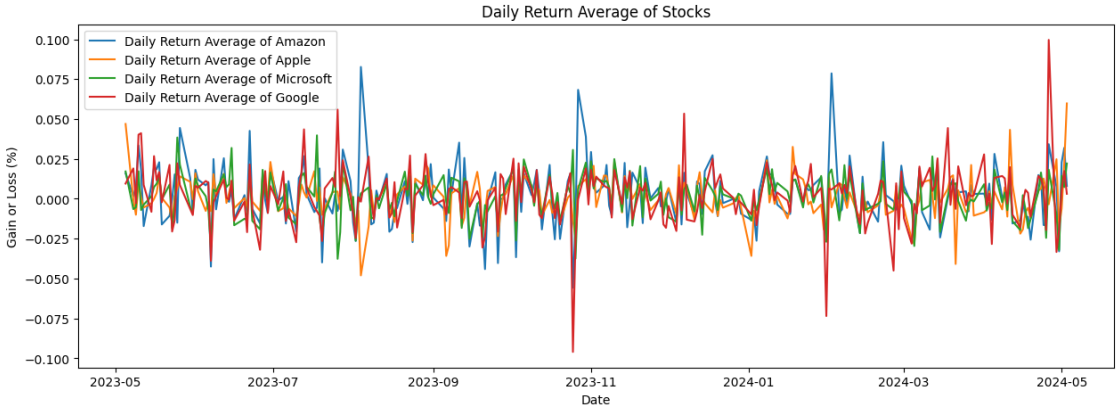
Date	Open	High	Low	Close	Adj Close	\
2024-04-22	156.009995	159.184998	155.660004	157.949997	157.949997	
2024-04-23	158.589996	160.479996	157.964996	159.919998	159.919998	
2024-04-24	159.089996	161.389999	158.820007	161.100006	161.100006	
2024-04-25	153.360001	158.279999	152.768005	157.949997	157.949997	
2024-04-26	175.990005	176.419998	171.399994	173.690002	173.690002	
2024-04-29	170.770004	171.380005	167.059998	167.899994	167.899994	
2024-04-30	167.380005	169.869995	164.500000	164.639999	164.639999	
2024-05-01	166.179993	168.809998	164.899994	165.570007	165.570007	
2024-05-02	166.669998	168.529999	165.690002	168.460007	168.460007	
2024-05-03	169.526703	169.850006	164.979996	168.990005	168.990005	

```
              Volume Company  MA for 20 Days  MA for 50 Days  MA for 100 Days  \
Date
2024-04-22  17243900  Google      155.696500       147.747599       144.588999
2024-04-23  16115400  Google      156.135000       147.941599       144.802000
2024-04-24  19485700  Google      156.605000       148.188999       145.049000
2024-04-25  36197800  Google      156.905500       148.420599       145.289300
2024-04-26  56500800  Google      157.977000       148.951599       145.693000
2024-04-29  35914600  Google      158.547000       149.430799       146.065699
2024-04-30  29420800  Google      158.985500       149.888399       146.388199
2024-05-01  25223200  Google      159.445501       150.355800       146.729600
2024-05-02  17041100  Google      160.271501       150.848200       147.029700
2024-05-03  22716443  Google      161.024001       151.321600       147.353200

            Daily Return
Date
2024-04-22      0.014321
2024-04-23      0.012472
2024-04-24      0.007379
2024-04-25     -0.019553
2024-04-26      0.099652
2024-04-29     -0.033335
2024-04-30     -0.019416
2024-05-01      0.005649
2024-05-02      0.017455
2024-05-03      0.003146
```

[50]:
```
plt.figure(figsize=(15, 5))
plt.plot(goog_test['Daily Return'])
plt.xlabel("Date")
plt.ylabel("Gain or Loss (%)")
plt.title("Daily Return Average of Google")
plt.show()
```



[51]:
```
plt.figure(figsize=(15, 5))
plt.plot(amzn_test['Daily Return'],label = "Daily Return Average of Amazon")
plt.plot(aapl_test['Daily Return'],label = "Daily Return Average of Apple")
plt.plot(msft_test['Daily Return'],label = "Daily Return Average of Microsoft")
plt.plot(goog_test['Daily Return'],label = "Daily Return Average of Google")
plt.xlabel("Date")
plt.ylabel("Gain or Loss (%)")
plt.title("Daily Return Average of Stocks")
plt.legend()
plt.show()
```



#10. Adding a new column 'Trend' whose values are based on the 'Daily Return'

[52]:
```
def trend(x):
  if x > -0.015 and x <= 0.015:
    return 'Slight or No change'
  elif x > 0.015 and x <= 0.04:
    return 'Slight Positive'
  elif x < -0.015 and x >= -0.4:
    return 'Slight Negative'
  elif x > 0.04 and x <= 0.06:
    return 'Positive'
  elif x < -0.04 and x >= -0.06:
    return 'Negative'
  elif x > 0.06 and x <= 0.07:
    return 'Among top gainers'
  elif x < -0.06 and x >= -0.07:
    return 'Among top losers'
  elif x > 0.07:
    return 'Bull run'
  elif x <= -0.07:
```

```
    return 'Bear drop'
```

[53]:
```
amzn_test['Trend'] = amzn_test['Daily Return'].apply(lambda x: trend(x))
amzn_test.tail(10)
```

[53]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days	Daily Return	Trend
2024-04-22	176.940002	178.869995	174.559998	177.229996	177.229996	37924900	Amazon	181.932502	177.3206	165.387900	0.014889	Slight or No change
2024-04-23	178.080002	179.929993	175.979996	179.539993	179.539993	37046500	Amazon	181.924001	177.4224	165.713000	0.013034	Slight or No change
2024-04-24	179.940002	180.320007	176.179993	176.589996	176.589996	34185100	Amazon	181.838501	177.5074	166.015699	-0.016431	Slight Negative
2024-04-25	169.679993	173.919998	166.320007	173.669998	173.669998	49249400	Amazon	181.530501	177.6080	166.291499	-0.016535	Slight Negative
2024-04-26	177.800003	180.820007	176.130005	179.619995	179.619995	43919800	Amazon	181.492500	177.7808	166.617399	0.034260	Slight Positive
2024-04-29	182.750000	183.529999	179.389999	180.960007	180.960007	54063900	Amazon	181.492001	178.0040	166.978600	0.007460	Slight or No change
2024-04-30	181.089996	182.990005	174.800003	175.000000	175.000000	94639800	Amazon	181.207500	178.1138	167.259799	-0.032935	Slight Negative
2024-05-01	181.639999	185.149994	176.559998	179.000000	179.000000	94645100	Amazon	181.037000	178.3522	167.604599	0.022857	Slight Positive
2024-05-02	180.850006	185.100006	179.910004	184.720001	184.720001	54303500	Amazon	181.273000	178.6748	167.982999	0.031955	Slight Positive
2024-05-03	186.815002	187.869995	185.429993	186.210007	186.210007	38680749	Amazon	181.330000	178.9074	168.370900	0.008066	Slight or No change

[54]:
```
aapl_test['Trend'] = aapl_test['Daily Return'].apply(lambda x: trend(x))
aapl_test.tail(10)
```

[54]:

Date	Open	High	Low	Close	Adj Close	Volume	Company	MA for 20 Days	MA for 50 Days	MA for 100 Days	Daily Return	Trend
2024-04-22	165.520004	167.259995	164.770004	165.839996	165.839996	48116400	Apple	169.885500	174.687599	182.477606	0.005091	Slight or No change
2024-04-23	165.350006	167.050003	164.919998	166.899994	166.899994	49537800	Apple	169.687999	174.248599	182.245033	0.006392	Slight or No change
2024-04-24	166.539993	169.300003	166.210007	169.020004	169.020004	48251800	Apple	169.653499	173.885999	182.043946	0.012702	Slight or No change
2024-04-25	169.529999	170.610001	168.149994	169.889999	169.889999	50558300	Apple	169.482499	173.583000	181.845767	0.005147	Slight or No change
2024-04-26	169.880005	171.339996	169.179993	169.300003	169.300003	44838400	Apple	169.373499	173.286000	181.628804	-0.003473	Slight or No change
2024-04-29	173.369995	176.029999	173.100006	173.500000	173.500000	68169400	Apple	169.546999	173.078800	181.471918	0.024808	Slight Positive
2024-04-30	173.330002	174.990005	170.000000	170.330002	170.330002	65934800	Apple	169.621500	172.839200	181.243483	-0.018271	Slight Negative
2024-05-01	169.580002	172.710007	169.110001	169.300003	169.300003	50383100	Apple	169.604000	172.594000	181.015734	-0.006047	Slight or No change
2024-05-02	172.509995	173.419998	170.889999	173.029999	173.029999	94214900	Apple	169.814500	172.408200	180.805809	0.022032	Slight Positive
2024-05-03	186.669998	187.000000	182.660004	183.380005	183.380005	157741757	Apple	170.504500	172.388400	180.685004	0.059816	Positive

[55]:
```
msft_test['Trend'] = msft_test['Daily Return'].apply(lambda x: trend(x))
msft_test.tail(10)
```

[55]:

Date	Open	High	Low	Close	Adj Close
2024-04-22	400.079987	402.850006	395.750000	400.959991	400.959991
2024-04-23	404.239990	408.200012	403.059998	407.570007	407.570007
2024-04-24	409.559998	412.470001	406.779999	409.059998	409.059998

```
2024-04-25  394.029999  399.890015  388.029999  399.040009  399.040009
2024-04-26  412.170013  413.000000  405.760010  406.320007  406.320007
2024-04-29  405.250000  406.320007  399.190002  402.250000  402.250000
2024-04-30  401.489990  402.160004  389.170013  389.329987  389.329987
2024-05-01  392.609985  401.720001  390.309998  394.940002  394.940002
2024-05-02  397.660004  399.929993  394.649994  397.839996  397.839996
2024-05-03  402.000000  407.149994  401.859985  406.660004  406.660004

              Volume    Company  MA for 20 Days  MA for 50 Days  \
Date
2024-04-22  20286900  Microsoft      418.244498      414.746144
2024-04-23  15734500  Microsoft      417.479999      414.502070
2024-04-24  15065300  Microsoft      416.850499      414.393400
2024-04-25  40586500  Microsoft      415.731000      414.262800
2024-04-26  29694700  Microsoft      415.011000      414.199401
2024-04-29  19582100  Microsoft      413.895000      414.113201
2024-04-30  28781400  Microsoft      412.289499      413.818600
2024-05-01  23562500  Microsoft      411.013998      413.661600
2024-05-02  17709400  Microsoft      410.011998      413.574800
2024-05-03  17390522  Microsoft      409.068999      413.475001

            MA for 100 Days  Daily Return                Trend
Date
2024-04-22       399.265914      0.004610  Slight or No change
2024-04-23       399.521678      0.016485      Slight Positive
2024-04-24       399.830771      0.003656  Slight or No change
2024-04-25       400.039065     -0.024495      Slight Negative
2024-04-26       400.364078      0.018244      Slight Positive
2024-04-29       400.701992     -0.010017  Slight or No change
2024-04-30       400.876968     -0.032119      Slight Negative
2024-05-01       401.145175      0.014409  Slight or No change
2024-05-02       401.420923      0.007343  Slight or No change
2024-05-03       401.752130      0.022170      Slight Positive
```

[56]:
```
goog_test['Trend'] = goog_test['Daily Return'].apply(lambda x: trend(x))
goog_test.tail(10)
```

[56]:
```
                 Open        High         Low       Close   Adj Close  \
Date
2024-04-22  156.009995  159.184998  155.660004  157.949997  157.949997
2024-04-23  158.589996  160.479996  157.964996  159.919998  159.919998
2024-04-24  159.089996  161.389999  158.820007  161.100006  161.100006
2024-04-25  153.360001  158.279999  152.768005  157.949997  157.949997
2024-04-26  175.990005  176.419998  171.399994  173.690002  173.690002
2024-04-29  170.770004  171.380005  167.059998  167.899994  167.899994
2024-04-30  167.380005  169.869995  164.500000  164.639999  164.639999
2024-05-01  166.179993  168.809998  164.899994  165.570007  165.570007
```

```
2024-05-02  166.669998  168.529999  165.690002  168.460007  168.460007
2024-05-03  169.526703  169.850006  164.979996  168.990005  168.990005

              Volume Company  MA for 20 Days  MA for 50 Days  MA for 100 Days  \
Date
2024-04-22  17243900  Google      155.696500      147.747599       144.588999
2024-04-23  16115400  Google      156.135000      147.941599       144.802000
2024-04-24  19485700  Google      156.605000      148.188999       145.049000
2024-04-25  36197800  Google      156.905500      148.420599       145.289300
2024-04-26  56500800  Google      157.977000      148.951599       145.693000
2024-04-29  35914600  Google      158.547000      149.430799       146.065699
2024-04-30  29420800  Google      158.985500      149.888399       146.388199
2024-05-01  25223200  Google      159.445501      150.355800       146.729600
2024-05-02  17041100  Google      160.271501      150.848200       147.029700
2024-05-03  22716443  Google      161.024001      151.321600       147.353200

            Daily Return                Trend
Date
2024-04-22      0.014321  Slight or No change
2024-04-23      0.012472  Slight or No change
2024-04-24      0.007379  Slight or No change
2024-04-25     -0.019553      Slight Negative
2024-04-26      0.099652             Bull run
2024-04-29     -0.033335      Slight Negative
2024-04-30     -0.019416      Slight Negative
2024-05-01      0.005649  Slight or No change
2024-05-02      0.017455      Slight Positive
2024-05-03      0.003146  Slight or No change
```

#11. Visualize trend frequency through a Pie Chart

[57]:
```
trendpie= amzn_test['Trend'].value_counts()
trendpie
```

[57]:
```
Trend
Slight or No change    158
Slight Positive         52
Slight Negative         36
Positive                 2
Bull run                 2
Among top gainers        1
Name: count, dtype: int64
```

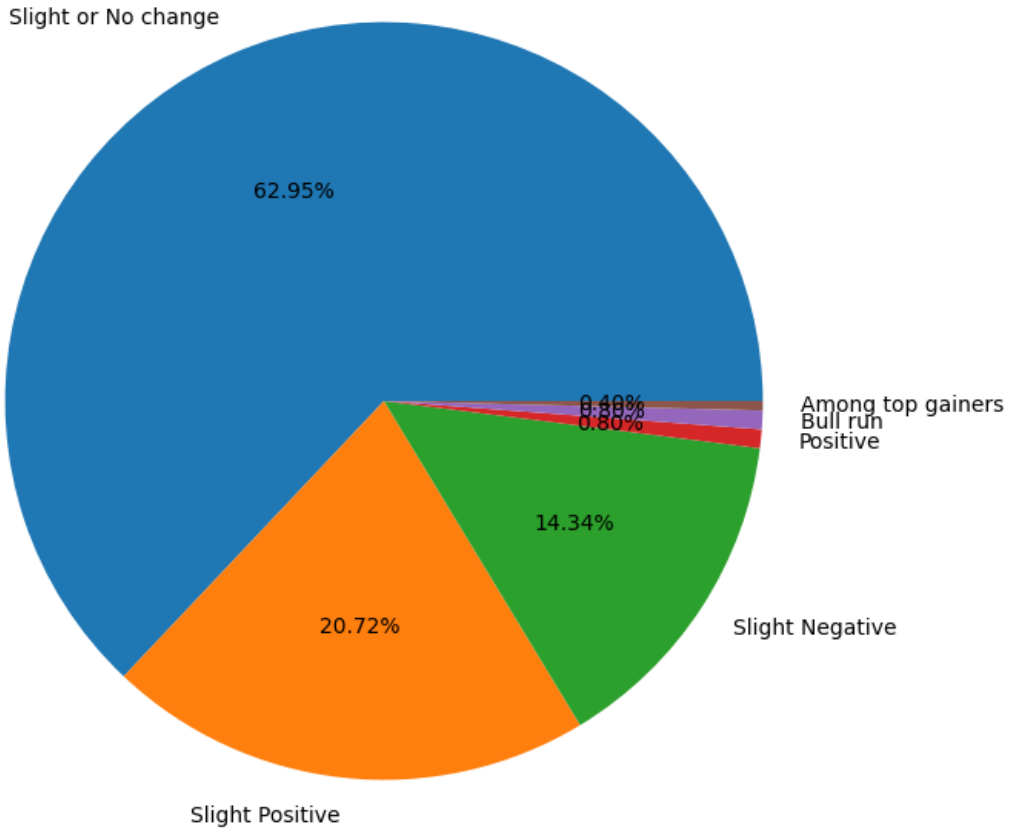
[58]:
```
trendpie.index
```

[58]:
```
Index(['Slight or No change', 'Slight Positive', 'Slight Negative', 'Positive',
       'Bull run', 'Among top gainers'],
      dtype='object', name='Trend')
```

[59]:
```
trendpie.values
```

[59]:
```
array([158,  52,  36,   2,   2,   1])
```

[60]:
```
plt.figure(figsize=(8, 8))
plt.pie(trendpie.values, labels = trendpie.index, autopct='%1.2f%%')
plt.show()
```



#12. What was the correlation between the daily returns of different stocks?

[61]:
```
amzn1 = amzn_test[['Adj Close']]
aapl1 = aapl_test[['Adj Close']]
msft1 = msft_test[['Adj Close']]
goog1 = goog_test[['Adj Close']]
```

[62]:
```
df = pd.DataFrame()
```

[63]:
```
df['Amazon']=amzn1
df['Apple']=aapl1
df['Misrosoft']=msft1
df['Google']=goog1
```

[64]:
```
df
```

[64]:

Date	Amazon	Apple	Misrosoft	Google
2023-05-04	104.000000	164.909760	302.923798	105.209999
2023-05-05	105.660004	172.648468	308.121185	106.214996
2023-05-08	105.830002	172.578827	306.137390	108.239998
2023-05-09	106.620003	170.858017	304.500824	107.940002
2023-05-10	110.190002	172.638504	309.767639	112.279999
…	…	…	…	…
2024-04-29	180.960007	173.500000	402.250000	167.899994
2024-04-30	175.000000	170.330002	389.329987	164.639999
2024-05-01	179.000000	169.300003	394.940002	165.570007
2024-05-02	184.720001	173.029999	397.839996	168.460007
2024-05-03	186.210007	183.380005	406.660004	168.990005

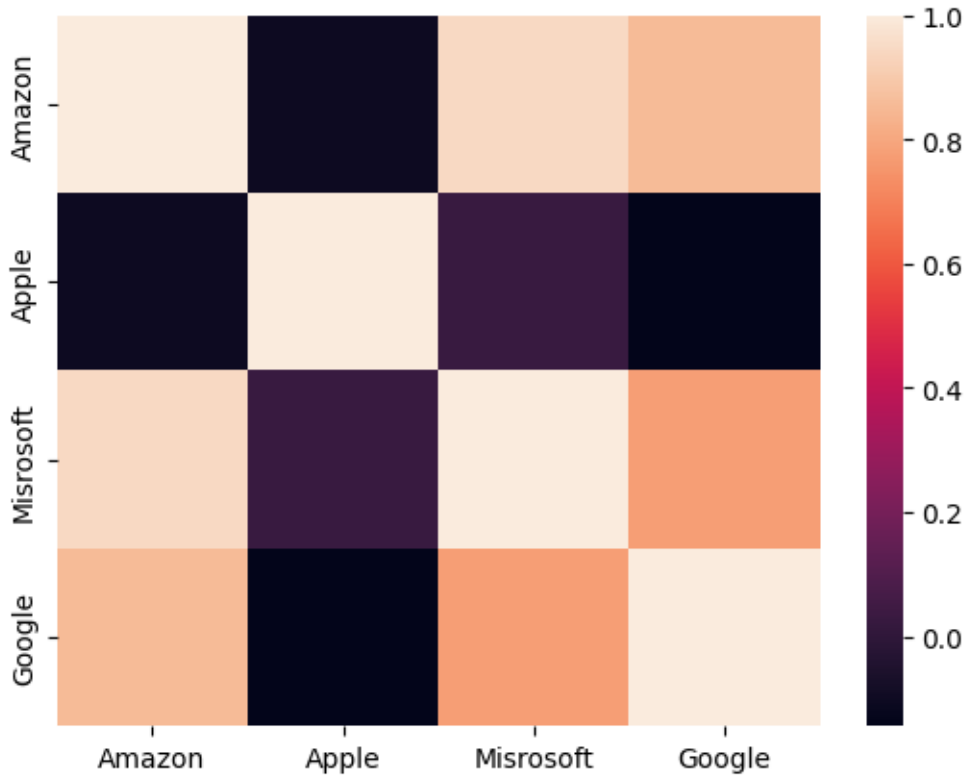
[252 rows x 4 columns]

[65]:
```
df.corr()
```

[65]:

	Amazon	Apple	Misrosoft	Google
Amazon	1.000000	-0.106709	0.945324	0.854489
Apple	-0.106709	1.000000	0.025577	-0.144629
Misrosoft	0.945324	0.025577	1.000000	0.775418
Google	0.854489	-0.144629	0.775418	1.000000

[66]:
```
sns.heatmap(df.corr())
plt.show()
```


Amazon
Apple
Misrosoft
Google
Amazon
Apple
Misrosoft
Google
1.0
0.8
0.6
0.4
0.2
0.0