India Market Performance Analysis using Python

Stock Market Performance Analysis involves calculating moving averages, measuring volatility, conducting correlation analysis and analyzing various aspects of the stock market to gain a deeper understanding of the factors that affect stock prices and the relationships between the stock prices of different companies.

```
import pandas as pd
import yfinance as yf
from datetime import datetime


start_date = datetime.now() - pd.DateOffset(months=3)
end_date = datetime.now()


tickers = ['HDFCBANK.NS', 'RELIANCE.NS', 'TCS.NS', 'HINDUNILVR.NS']



df_list = []


for ticker in tickers:
    data = yf.download(ticker, start=start_date, end=end_date)
    df_list.append(data)
```

```
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
```

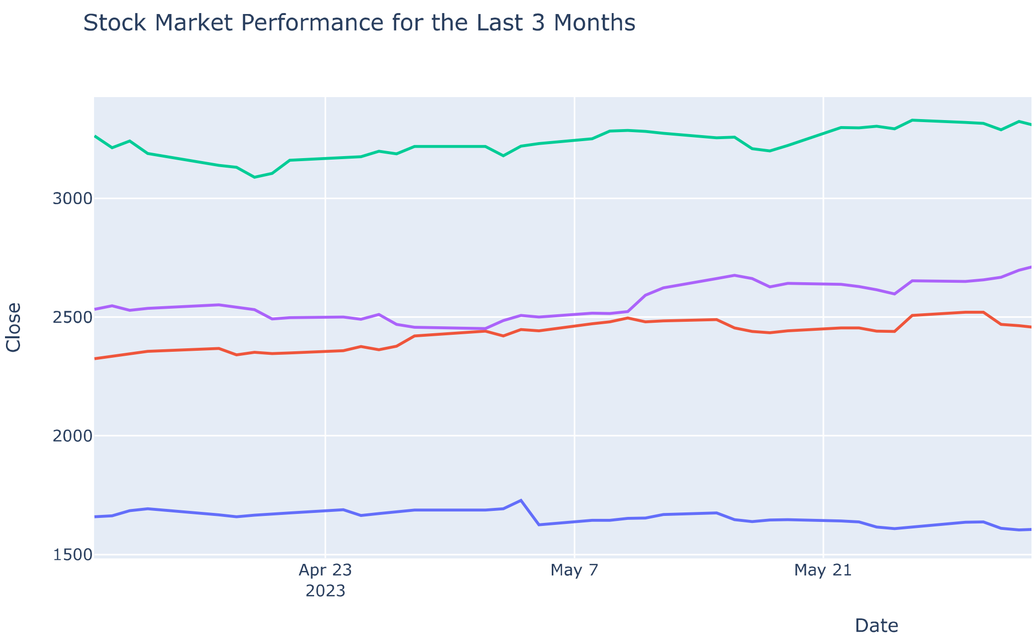
```
df = pd.concat(df_list, keys=tickers, names=['Ticker', 'Date'])
print(df.head())
```

```
                               Open         High          Low        Close  \
Ticker      Date
HDFCBANK.NS 2023-04-10  1663.250000  1671.000000  1654.000000  1658.449951
            2023-04-11  1659.000000  1669.400024  1651.099976  1663.300049
            2023-04-12  1668.000000  1688.199951  1667.800049  1684.900024
            2023-04-13  1688.300049  1697.050049  1678.150024  1692.449951
            2023-04-17  1720.000000  1720.000000  1653.300049  1666.650024

                          Adj Close    Volume
Ticker      Date
HDFCBANK.NS 2023-04-10  1639.646606  15906643
            2023-04-11  1644.441772  21180771
            2023-04-12  1665.796875  13477752
            2023-04-13  1673.261108  17665043
            2023-04-17  1647.753784  16202574
```

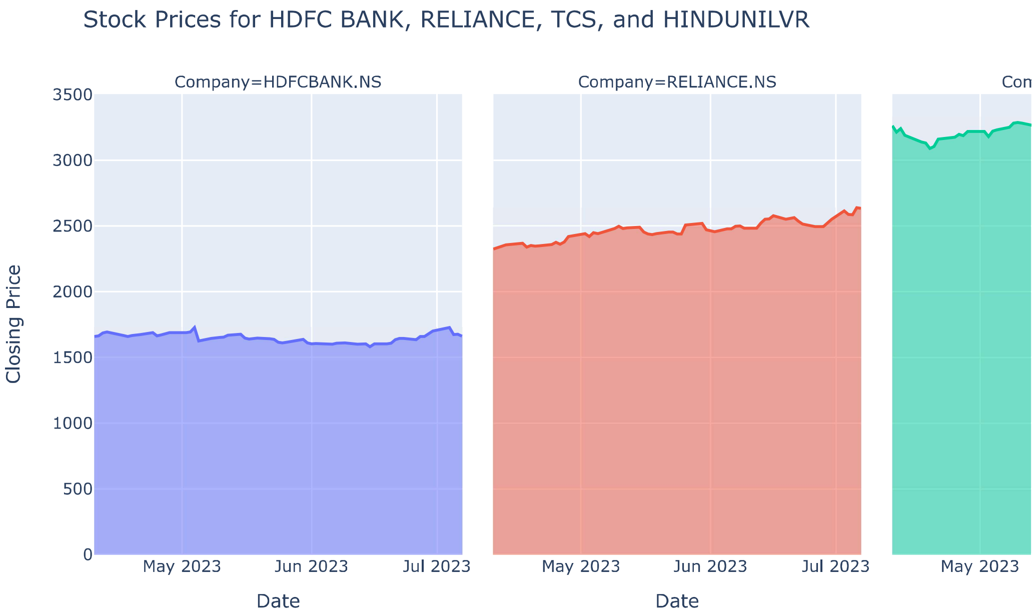
```
df = df.reset_index()
print(df.head())
```

```
        Ticker       Date         Open         High          Low        Close  \
0  HDFCBANK.NS 2023-04-10  1663.250000  1671.000000  1654.000000  1658.449951
1  HDFCBANK.NS 2023-04-11  1659.000000  1669.400024  1651.099976  1663.300049
2  HDFCBANK.NS 2023-04-12  1668.000000  1688.199951  1667.800049  1684.900024
3  HDFCBANK.NS 2023-04-13  1688.300049  1697.050049  1678.150024  1692.449951
4  HDFCBANK.NS 2023-04-17  1720.000000  1720.000000  1653.300049  1666.650024

     Adj Close    Volume
0  1639.646606  15906643
1  1644.441772  21180771
2  1665.796875  13477752
3  1673.261108  17665043
4  1647.753784  16202574
```

```
import plotly.express as px
fig = px.line(df, x='Date',
              y='Close',
              color='Ticker',
              title="Stock Market Performance for the Last 3 Months")
fig.show()
```



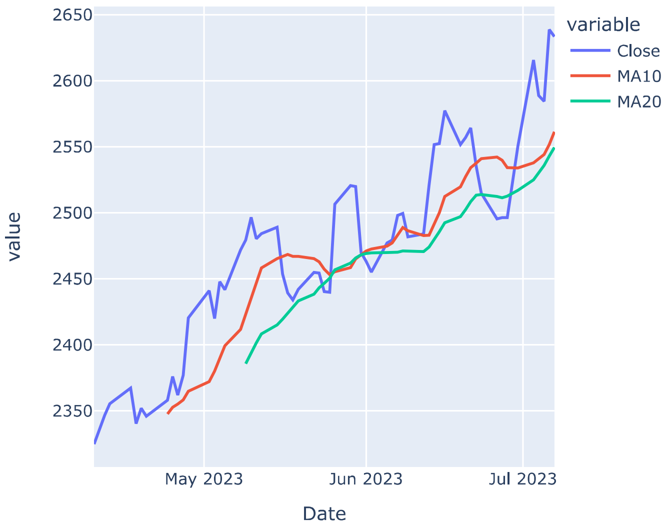
```
fig = px.area(df, x='Date', y='Close', color='Ticker',
              facet_col='Ticker',
              labels={'Date':'Date', 'Close':'Closing Price', 'Ticker':'Company'},
              title='Stock Prices for HDFC BANK, RELIANCE, TCS, and HINDUNILVR')
fig.show()
```



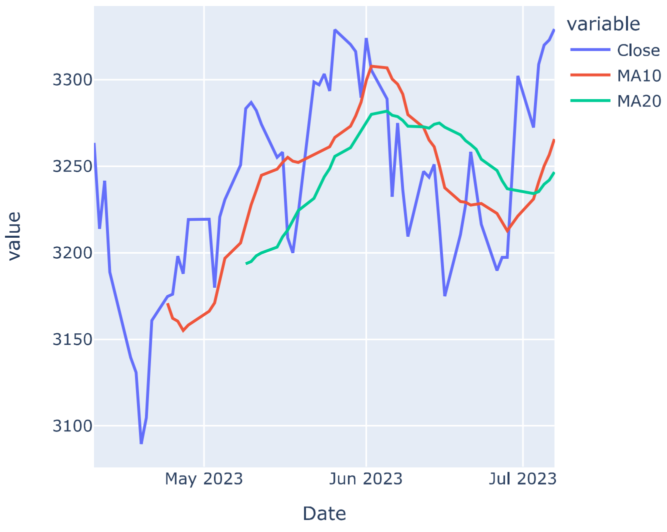
```
df['MA10'] = df.groupby('Ticker')['Close'].rolling(window=10).mean().reset_index(0, drop=True)
df['MA20'] = df.groupby('Ticker')['Close'].rolling(window=20).mean().reset_index(0, drop=True)

for ticker, group in df.groupby('Ticker'):
    print(f'Moving Averages for {ticker}')
    print(group[['MA10', 'MA20']])
```

```
0           NaN          NaN
1           NaN          NaN
2           NaN          NaN
3           NaN          NaN
4           NaN          NaN
..          ...          ...
57  1650.830005  1626.497504
58  1663.234998  1632.687500
59  1669.815002  1636.382501
60  1673.755005  1639.747504
61  1675.435010  1642.332507

[62 rows x 2 columns]
Moving Averages for HINDUNILVR.NS
            MA10         MA20
186          NaN          NaN
187          NaN          NaN
188          NaN          NaN
189          NaN          NaN
190          NaN          NaN
..           ...          ...
243  2666.590063  2675.430005
244  2668.265063  2675.640002
245  2676.240063  2678.875012
246  2684.290063  2680.920020
247  2688.665063  2681.830029

[62 rows x 2 columns]
Moving Averages for RELIANCE.NS
            MA10         MA20
62           NaN          NaN
63           NaN          NaN
64           NaN          NaN
65           NaN          NaN
66           NaN          NaN
..           ...          ...
119  2537.785010  2525.050000
120  2541.480005  2530.625000
121  2544.219995  2535.882495
122  2551.664990  2542.912500
123  2561.475000  2549.610010

[62 rows x 2 columns]
Moving Averages for TCS.NS
            MA10         MA20
124          NaN          NaN
125          NaN          NaN
126          NaN          NaN
127          NaN          NaN
128          NaN          NaN
..           ...          ...
181  3231.005005  3234.237488
182  3240.850024  3235.234998
183  3250.075024  3239.617493
184  3256.545020  3242.017493
185  3265.620020  3246.657495

[62 rows x 2 columns]
```

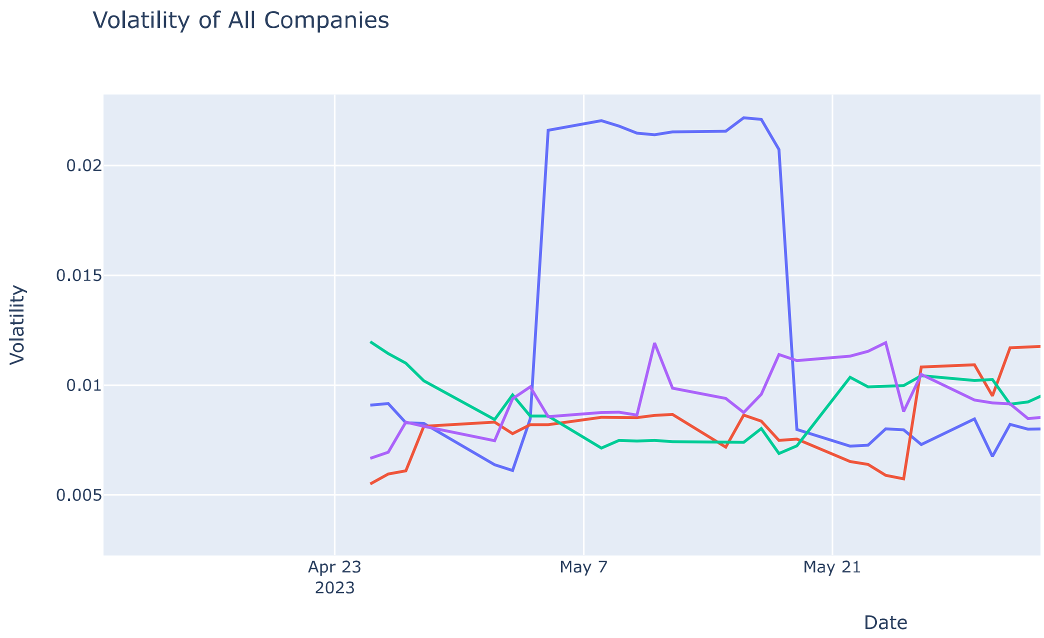
```
for ticker, group in df.groupby('Ticker'):
    fig = px.line(group, x='Date', y=['Close', 'MA10', 'MA20'],
                  title=f"{ticker} Moving Averages")
    fig.show()
```



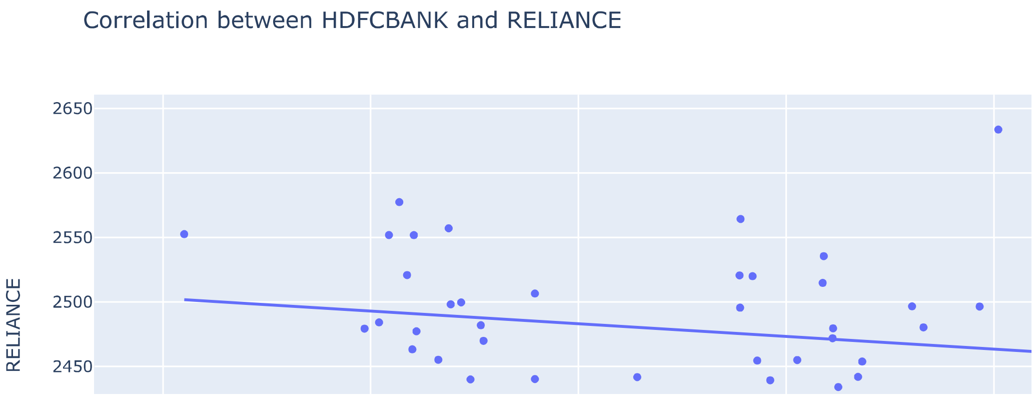
TCS.NS Moving Averages



```
df['Volatility'] = df.groupby('Ticker')['Close'].pct_change().rolling(window=10).std().reset_index(0, drop=True)
fig = px.line(df, x='Date', y='Volatility',
              color='Ticker',
              title='Volatility of All Companies')
fig.show()
```



```
# create a DataFrame with the stock prices of HDFC_BANK and RELIANCE
HDFC_BANK = df.loc[df['Ticker'] == 'HDFCBANK.NS', ['Date', 'Close']].rename(columns={'Close': 'HDFCBANK.NS'})
RELIANCE= df.loc[df['Ticker'] == 'RELIANCE.NS', ['Date', 'Close']].rename(columns={'Close': 'RELIANCE'})
df_corr = pd.merge(HDFC_BANK, RELIANCE, on='Date')

# create a scatter plot to visualize the correlation
fig = px.scatter(df_corr, x='HDFCBANK.NS', y='RELIANCE',
                 trendline='ols',
                 title='Correlation between HDFCBANK and RELIANCE')
fig.show()
```



Stock Market Performance Analysis: Overview and Tasks Performed

1. Stock market performance analysis is used to analyze historical stock price data, identify opportunities and risks, and make informed investment decisions.
2. The analysis involves collecting real-time stock market data using the Yahoo Finance API.
3. The dataset consists of historical stock price data for companies like Hdfc bank, Reliance, T.C.S, and Hindunilvr.
4. Data visualization is performed using line and area charts to visualize the stock market performance of the companies over the last three months.
5. Moving averages are calculated to identify trends and patterns in each company's stock price movements.
6. The moving averages are visualized to observe the crossover points and identify bullish or bearish signals.
7. Volatility is measured by calculating the percentage change in stock prices and rolling standard deviation.
8. The volatility of all companies is visualized using a line chart to observe the fluctuations in stock prices.
9. Correlation analysis is conducted between the stock prices of Hdfc Bank and Reliance to identify any relationship or similarity between them.
10. The correlation is visualized using a scatter plot with a trendline to show the strength and direction of the relationship.

Tasks performed:

- Importing necessary Python libraries and downloading historical stock price data.
- Resetting the index of the dataset for further analysis.
- Visualizing stock market performance using line and area charts.
- Calculating and visualizing moving averages for each company.
- Measuring and visualizing volatility of stock prices.
- Conducting correlation analysis between Apple and Microsoft stock prices.

✓ 0s completed at 10:47 PM ● ×