

STOCKS VISUALIZATION AND ANALYSIS USING DASH

A Projected Report

Submitted by

Harshit Shah (202218040)

Jitul Bakshi (202218059)

Nidhi Somaiya (202218060)



DEPARTMENT OF DATA SCIENCE

**DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION
TECHNOLOGY**

INDEX

<u>Sr.no</u>	<u>Topic</u>	<u>Pg.no</u>
<u>1</u>	<u>Introduction</u>	<u>1</u>
<u>2</u>	<u>Data pre-processing and cleaning</u>	<u>4</u>
<u>3</u>	<u>Visualization</u>	<u>7</u>
<u>4</u>	<u>Analyzation</u>	<u>17</u>
<u>5</u>	<u>GUI (Graphical User Interface)</u>	<u>33</u>
<u>6</u>	<u>Conclusion</u>	<u>37</u>
<u>7</u>	<u>Learning</u>	<u>37</u>

1.INTRODUCTION:

The stock market can be viewed as a platform where almost all major economic transactions in the world occur at a dynamic rate called the stock value which is based on the market equilibrium. For analysis of the stock market the basic requirement is the visualization of the various factors contributing to stock variation. Further comes the process of Analysis of the stocks.

1.1 TYPES OF ANALYSIS:

There are two types of Stocks Analysis:

Fundamental Stocks Analysis: The fundamental stock analysis method involves the evaluation of a business at a basic financial level. Investors use fundamental analysis to determine whether the current price of a company's stock reflects the future value of the company. Fundamental analysis uses different factors such as the current economic environment and finances of the company to estimate its stock value. Different key ratios are also used to determine the financial health and understand the true value of a company's stock.

Technical Stocks Analysis: The technical analysis method involves examining data generated through market activities, such as volume and prices. Analysts following such a type of stock analysis use technical indicators and tools like charts to identify patterns that can indicate price trends or direction. Technical analysts examine the historical trading data of a security and estimate the future move of the security.

1.2 PROPOSED ANALYSIS AND VISUALIZATION:

In this project we have done Technical Analysis. Here we have used the visualization tools to visualize the trends in the various fields of the stocks using python Visualization techniques. And made a GUI using python Dash components.

The key terms used in the project are as follows:

- OPEN- open price is the price at which the market opens for the stock.
- CLOSE- close price is the price at which the market close for the stock.

- **ADJ CLOSE-** The adjusted closing price factors in anything that might affect the stock price after the market closes.
- **VOLUME-** Volume is the number of shares of a security traded during a given period.
- **HIGH –** Highest value achieved in the complete day.
- **LOW-** Lowest value achieved in the complete day.

VISUALIZATION: In the Visualization Part , we have shown the visualization of all the individual fields viz open price, close price, adj close price, volume , high price and the low price.

ANALYSIS: 1. Individual Analysis: In individual analysis we have performed the individual analysis of the stocks and analysed the trend , percentage change in values of the closing price and analysis of trend using the key indicator called Simple Moving Average.

2. Combined Analysis: In Combined analysis we have performed the combined analysis of all the stocks and tried to analyse the risk , comparative trading in the market , comparative trend using the key indicator Simple Moving Average.

TECHNICAL PROFILE :

a. Library Used:

1. **Pandas:** It primarily used for data analysis, manipulation, cleaning, etc
2. **NumPy:** NumPy focuses on scientific computation. It features built-in mathematical functions for quick computation and supports big matrices and multidimensional data.
3. **Dash:** Dash is an open-source framework for building data visualization interfaces.
4. **Plotly.express :** Plotly Express is a new high -level Python visualization library. It's a wrapper for Plotly.py that exposes a simple syntax for complex charts.

5. **Plotly.graph_objects** :It used module contains the objects(Figure, layout,data,and the definition of the plots like scatter plot, line chart) that are responsible for creating the plots
6. **Matplotlib.pyplot**: Matplotlib is the plotting library for the python programming language and pyplot is a numerical mathematics extensions NumPy
7. **Dash Dependencies**: **a.** dcc : Dash Core Components all you to create interactive plots using Dropdown, graphs , date ranges etc. **b.** html: Dash html components allows you to access HTML tags. **c.** Input/Output: It is a dependency used to implement callback(Reactive Programming)

2. Data Pre-Processing and Cleaning

2.1. DATA FETCHING:

Data sets for this project are fetched from [Yahoo Finance](#). Four different market companies' data viz **Cipla Ltd, Hindustan Unilever Ltd, Infosys Ltd, Reliance Industries Ltd** is fetched. Time period of the data is of five financial years April 2018 to March 2022.

2.2. DATA CLEANING PROCESS:

- Individual file data cleaning

(for all the four csv files process is same)

- A. Importing csv files :- After importing csv we see that there are 7 columns as Date, High , Low , Open , Close , Adj Close and Volume.
- B. Adding company name as new column in each file.

```
1 ❶ PLA_df = pd.read_csv("D:\Study\Project\Stocks-Visualizing-and-Analysis\Datasets\CIPLA.NS .csv")
2  CIPLA_df["Name"] = "CIPLA"
3  CIPLA_df.head()
```

[54]

	Date	Open	High	Low	Close	Adj Close	Volume	Name
0	2018-04-02	548.000000	578.950012	545.900024	576.250000	562.826416	2823880	CIPLA
1	2018-04-03	572.000000	578.349976	568.799988	570.299988	557.014954	2493190	CIPLA
2	2018-04-04	572.349976	575.450012	564.500000	569.799988	556.526672	1852997	CIPLA
3	2018-04-05	574.000000	578.549988	558.349976	560.349976	547.296753	2396864	CIPLA
4	2018-04-06	560.299988	571.000000	557.250000	558.950012	545.929382	2420324	CIPLA

- C. Rounding off data up to two decimal places.

```
1  #as the data is too big. i am rounding of the data after decimals.
2  ❷ PLA_df = CIPLA_df.round(2)
3  CIPLA_df.head(2)
```

[55]

	Date	Open	High	Low	Close	Adj Close	Volume	Name
0	2018-04-02	548.0	578.95	545.9	576.25	562.83	2823880	CIPLA
1	2018-04-03	572.0	578.35	568.8	570.30	557.01	2493190	CIPLA

D. Checking for the null values.

```
1 CIPLA_df.isnull().sum()
[56]
... Date      0
    Open      0
    High      0
    Low       0
    Close     0
    Adj Close  0
    Volume    0
    Name      0
    dtype: int64
```

E. Finding the size of data.

```
1 Here is checked the size of data
2 CIPLA_df.shape
[58]
... (989, 8)
```

F. Checking the datatype of each column.

```
1 Lets check the datatypes of each the column
2 CIPLA_df.dtypes
[60]
... Date      object
    Open      float64
    High      float64
    Low       float64
    Close     float64
    Adj Close float64
    Volume    int64
    Name      object
    dtype: object
```

G. Converting the date (object) data type to date (Date Time)

```
1 Here we can see that data is in object format, pandas has inbuilt functions for data but the data should be in type 'datetime64[ns]'
2 CIPLA_df['Date'] = pd.to_datetime(CIPLA_df['Date'])
3 CIPLA_df.head(2)
```

[60]

	Date	Open	High	Low	Close	Adj Close	Volume	Name
0	2018-04-02	548.0	578.95	545.9	576.25	562.83	2023080	CIPLA
1	2018-04-03	572.0	578.35	568.8	570.30	557.01	2493190	CIPLA

H. Finding the total duration of analysis

```
1 Here is checked the size of data
2 CIPLA_df.shape
```

[58]

... (989, 8)

B.2- Creating a combined csv

- A. Creating a new list of all data frames
- B. Concatenating all the data frames in one
- C. Creating a new csv with name "CombinedALL.csv"

COMBINED ANALYSIS FILE

```
1 #Combining all the csvs
2 #creating the combined frame called "frames" consisting all the frames
3 frames=[INFY_df,HINDUNILVR_df,RELIANCE_df,CIPLA_df]
4 #concatinating the combined
5 data_all=pd.concat(frames)
6 Here is checked the size of all the data frames
7 data_all.to_csv('CombinedALL.csv')
8
```

[102]

3. VISUALIZATION

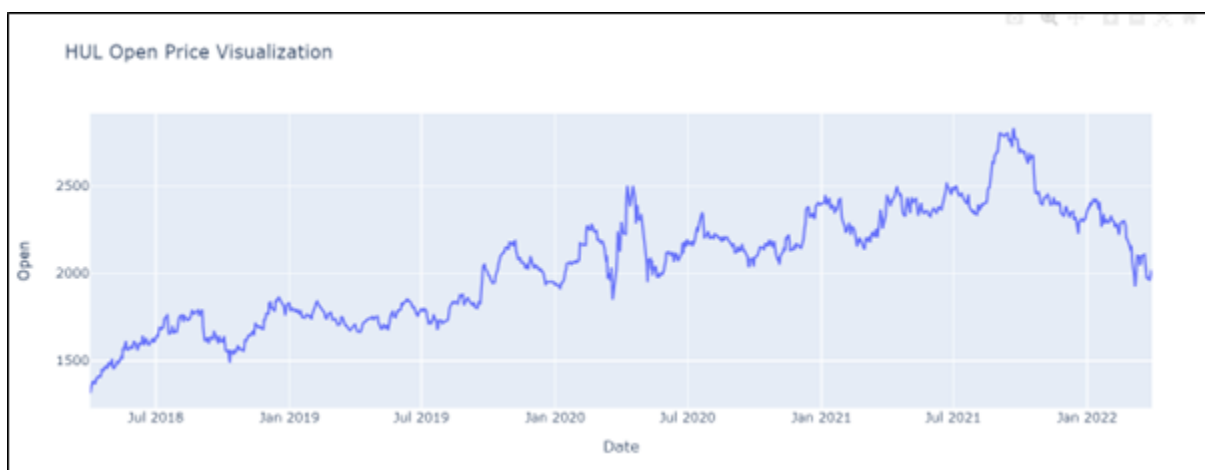
For the visualization of stocks, we have used line plot and candle chart to show individual data representation of all the fields of the data for all the files. Using Dash we have made an interactive GUI which on selection of a company name and the field gives the desired plot.

3.1 Open price of stocks Visualization

```
1 cipla_plot_line=px.line(x="Date",y="Open",data_frame=CIPLA_df,title="Cipla Open Price Visualization")
2 cipla_plot_line.show()
```

✓ 0.2s

(Similarly open prices of all the companies is plot.)



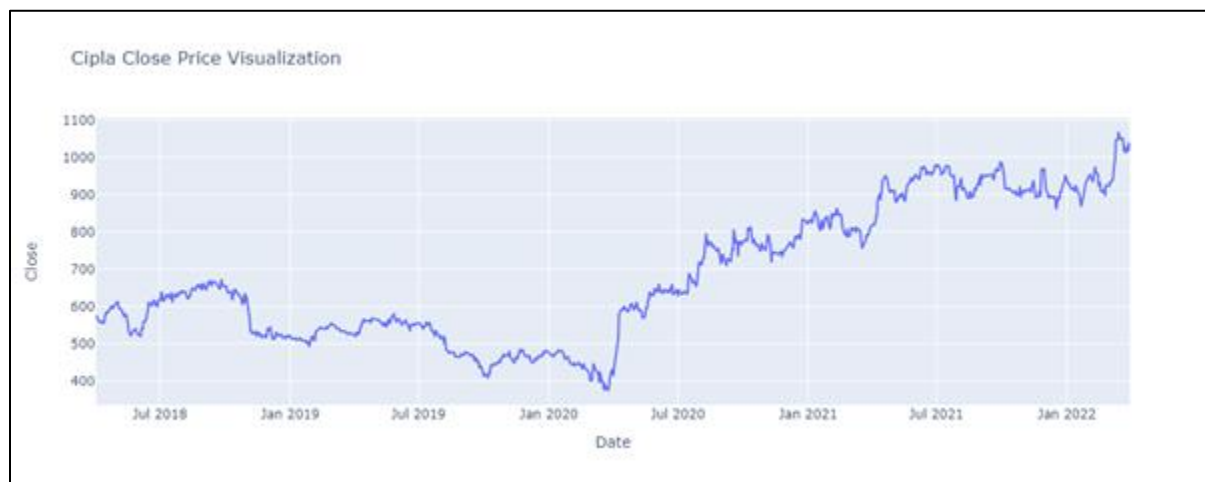


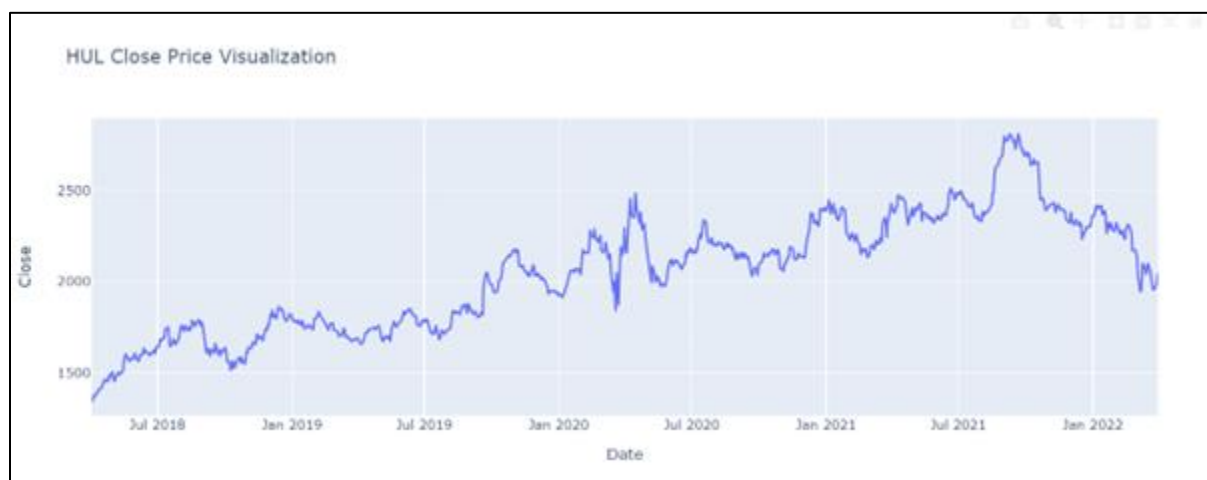
3.2 Close price of stocks Visualization

```
1 cipla_plot_line=px.line(x="Date",y="Close",data_frame=CIPLA_df,title="Cipla Close Price Visualization")
2 cipla_plot_line.show()
```

✓ 0.1s

(Similarly close prices of all the companies are plotted.)



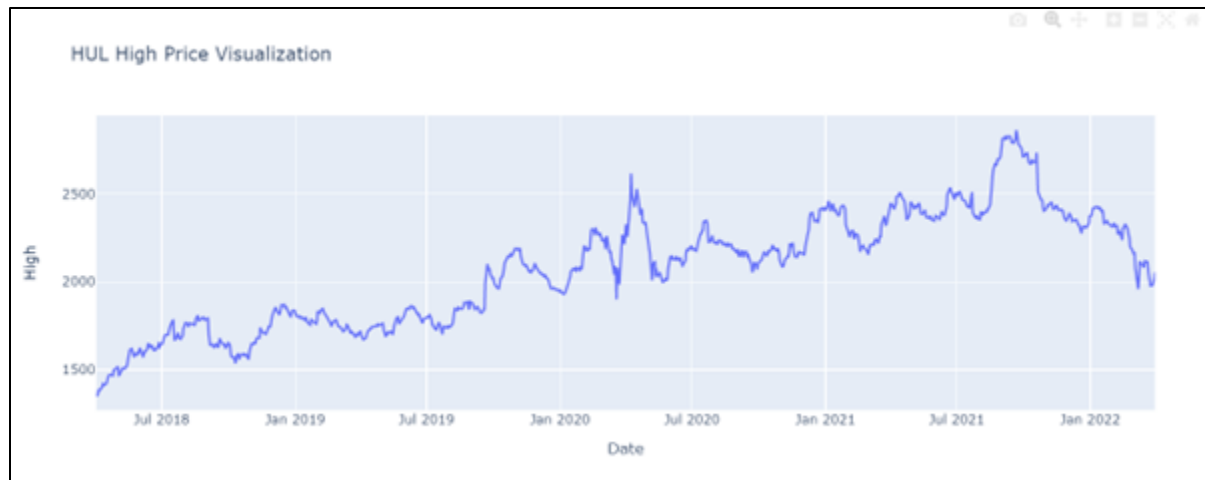


3.3 High price of stocks Visualization

```
1  cipla_plot_line=px.line(x="Date",y="High",data_frame=CIPLA_df,title="Cipla High Price Visualization")
2  cipla_plot_line.show()
✓ 0.4s
```

(Similarly high prices of all the companies are plotted.)

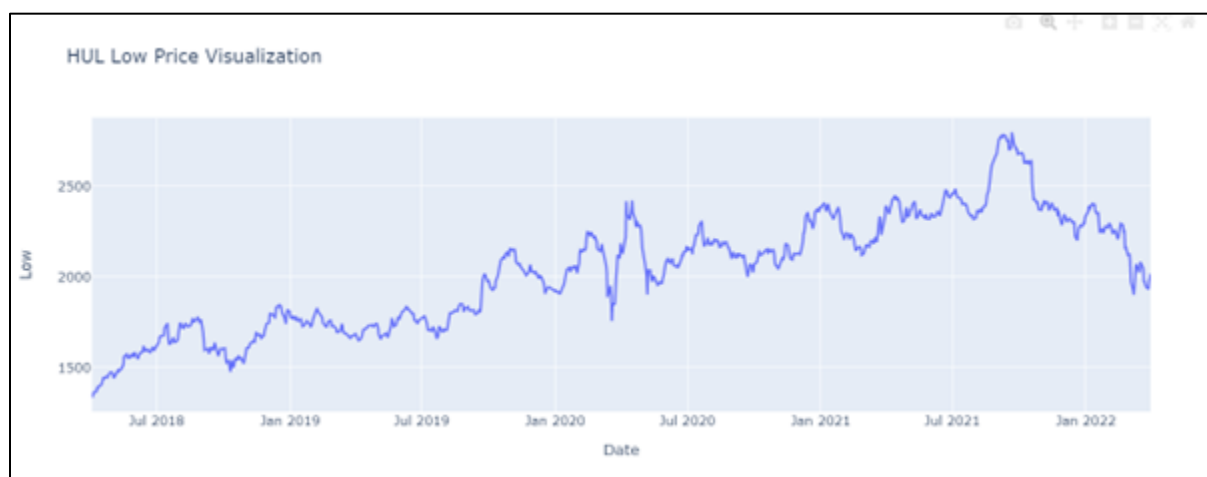
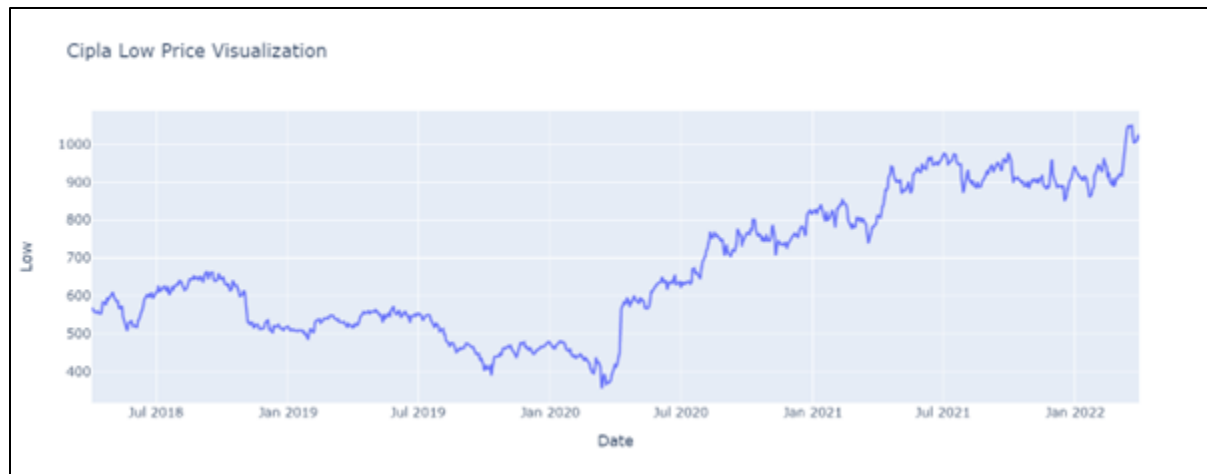




3.4 Low price of stocks Visualization

```
1 cipla_plot_line=px.line(x="Date",y="Low",data_frame=CIPLA_df,title="Cipla Low Price Visualization")
2 cipla_plot_line.show()
✓ 0.3s
```

(Similarly low prices of all the companies are plotted.)

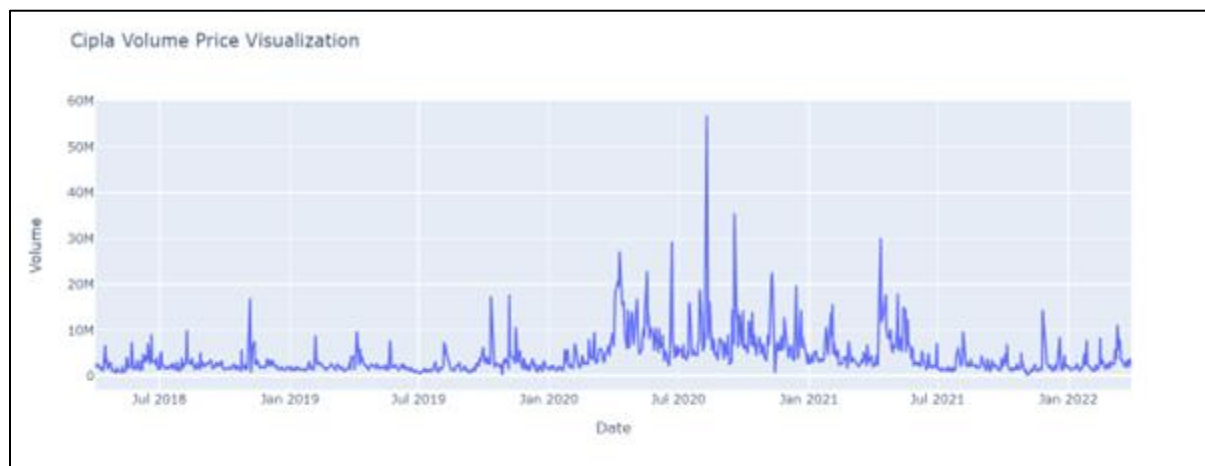


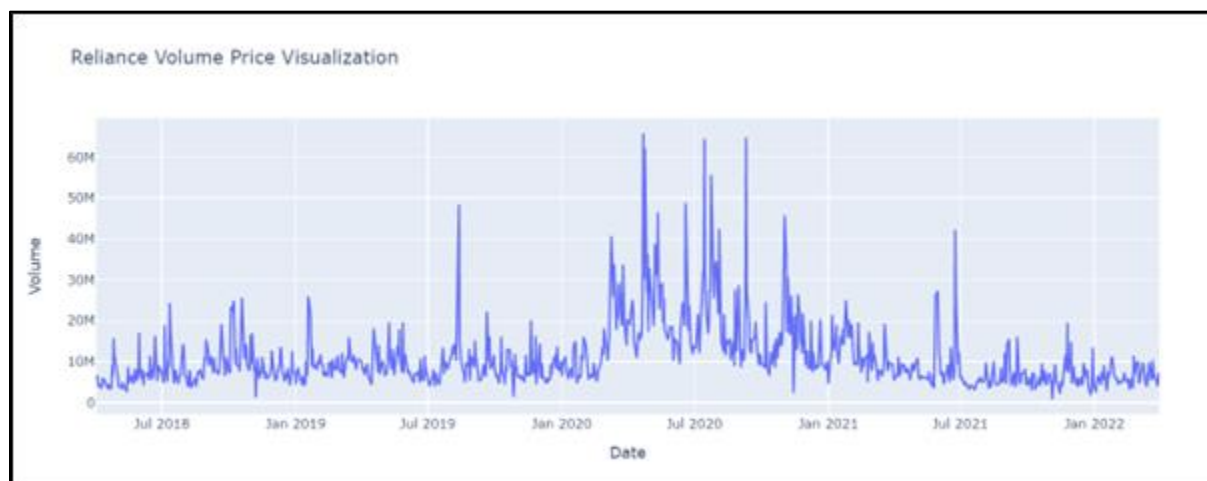
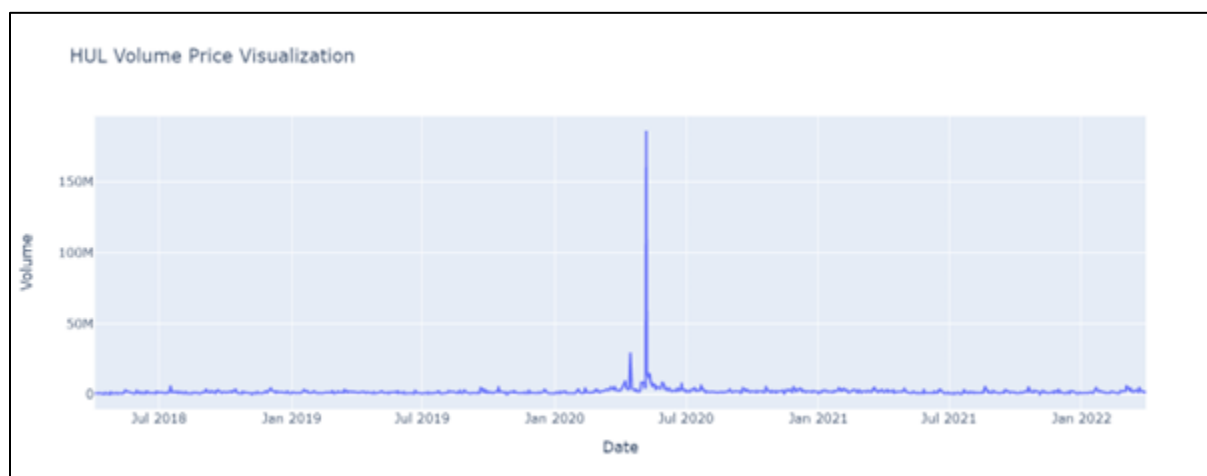
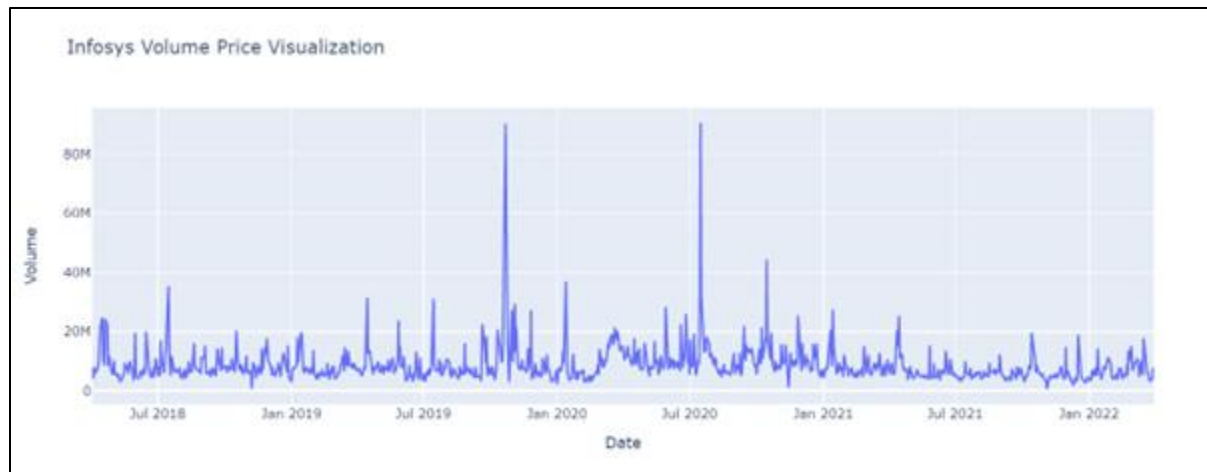


3.5 Volume of stocks Visualization

```
1 cipla_plot_line=px.line(x="Date",y="Volume",data_frame=CIPLA_df,title="Cipla Volume Price Visualization")  
2 cipla_plot_line.show()  
✓ 0.1s
```

(Similarly the volume of all the companies are plotted.)





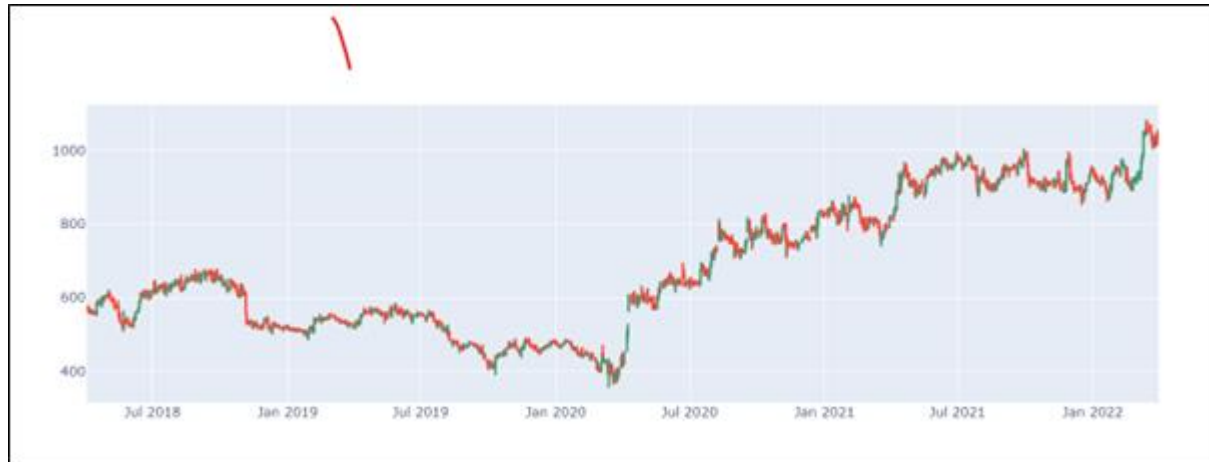
3.6 Candle chart

Cipla:

```
1 fig=go.Figure(data=[go.Candlestick(x=CIPLA_df.Date,open=CIPLA_df.Open,high=CIPLA_df.High,low=CIPLA_df.Low,close=CIPLA_df.Close)])
2 fig.update_layout(xaxis_rangeslider_visible=False)
3 fig.show()
```

✓ 0.6s

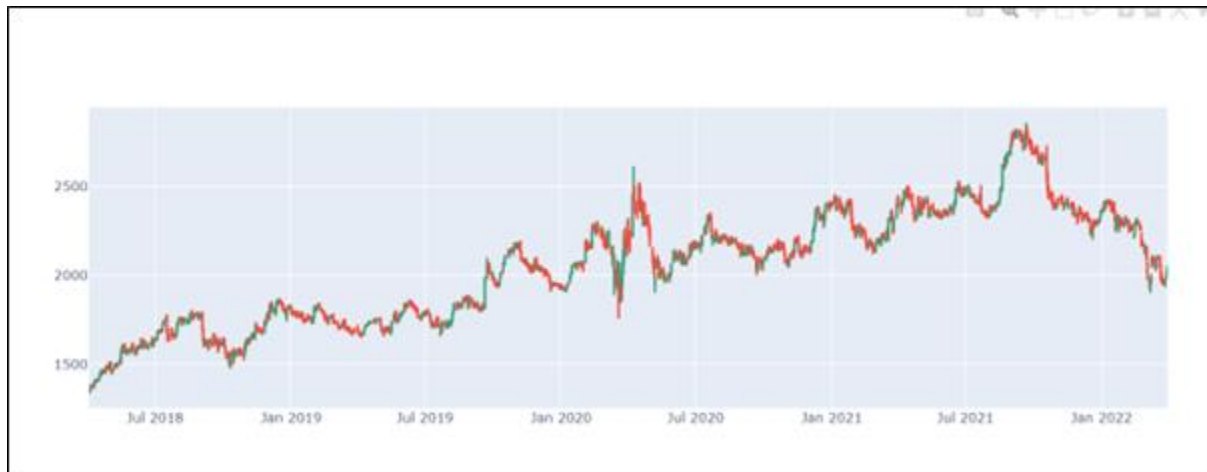
(**add the graphs of other companies by changing the dataframe.*)



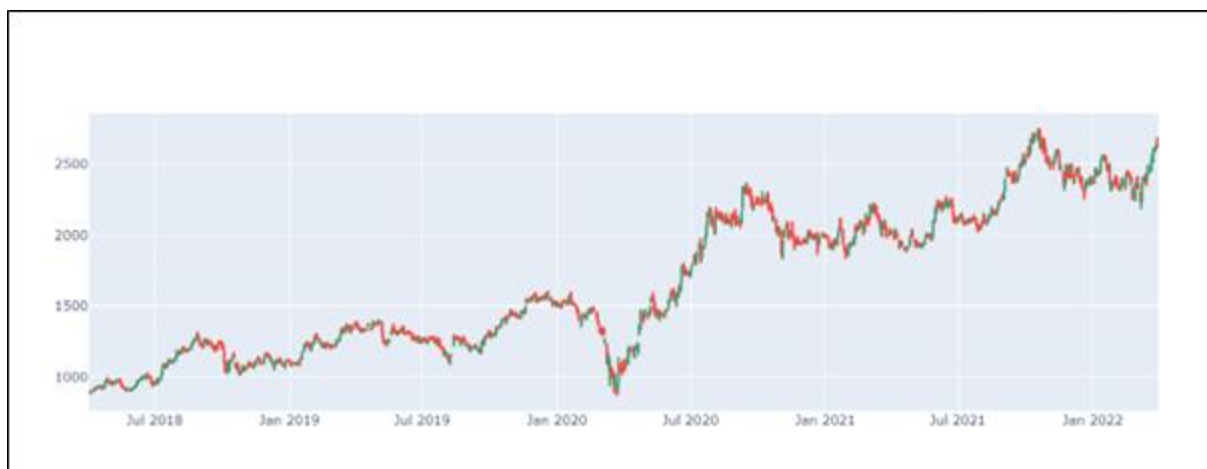
Infosys:



HUL:



Reliance:



4.ANALYSIS

For the analysis part, we have done analysis in two parts.

4.1. Individual Analysis

4.2. Combined Analysis

4.1 Individual Analysis of Stocks

(code snippet for only one company is shown , rest all can be plotted by changing the dataframe.)

Ø STOCK – CIPLA LTD

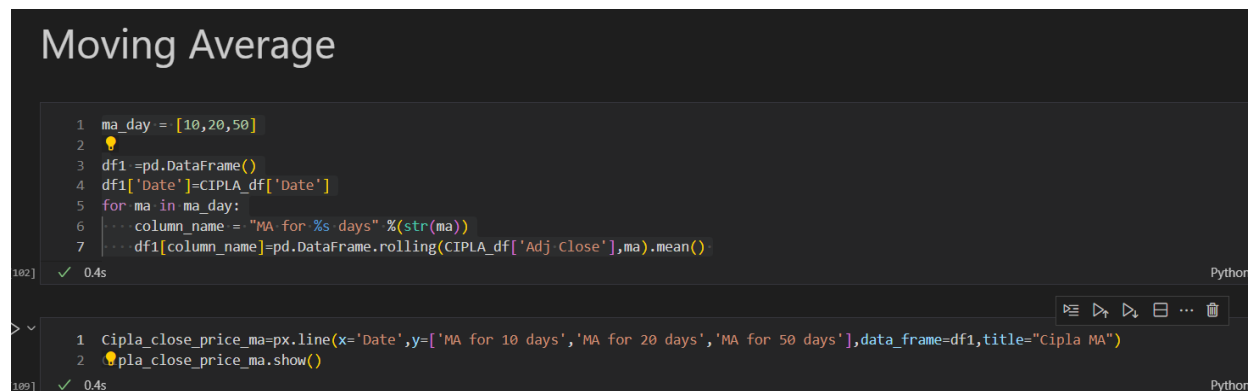
a. Trend Analysis Using SMA (Simple Moving Average)

Why Simple Moving Average?

As the stock price changes dynamically with time, it's difficult to find a trend with a big dataset. Hence, we use a key indicator – Simple Moving Average.

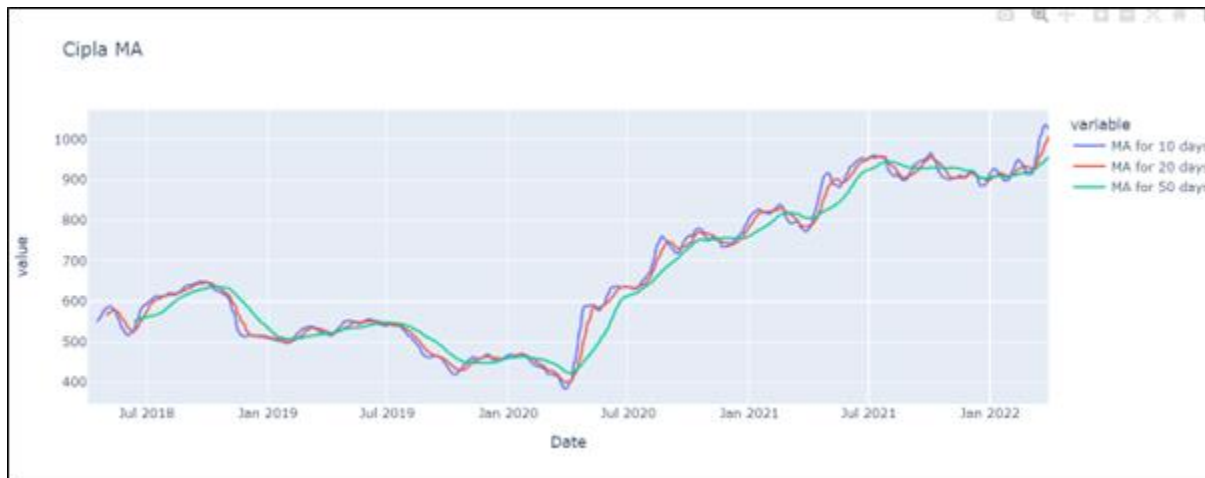
Analysis:

As you can see before March 2020, we see considerable ups and downs in the adjusted closing price , and in March 2020 there was a biggest depression in price but after March 2020 we see a bullish trend(upwards trend).



```
1 ma_day = [10,20,50]
2
3 df1 =pd.DataFrame()
4 df1['Date']=CIPLA_df['Date']
5 for ma in ma_day:
6     column_name = "MA for %s days" %(str(ma))
7     df1[column_name]=pd.DataFrame.rolling(CIPLA_df['Adj Close'],ma).mean()
102] ✓ 0.4s Python

1 Cipla_close_price_ma=px.line(x='Date',y=['MA for 10 days','MA for 20 days','MA for 50 days'],data_frame=df1,title="Cipla MA")
2 Cipla_close_price_ma.show()
109] ✓ 0.4s Python
```



b. Day to Day Analysis

Day to Day Analysis is the percent change in day-to-day price of Adj Closing Price.

Analysis:

It shows the minimum negative change in price was -7.2% on November 5, 2018. And the maximum positive change in price was +13% change on April 9, 2020.

```

1 CIPLA_df['Day_Perc_Change'] = CIPLA_df['Adj_Close'].pct_change()*100
2 CIPLA_df.head()
[95] ✓ 0.9s Python
...

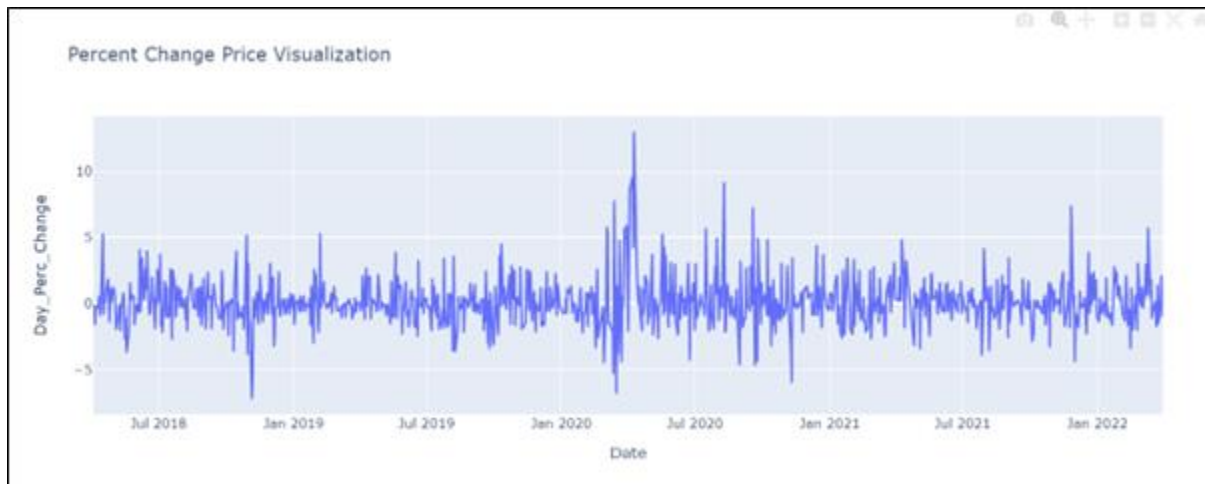
```

Unnamed: 0	Date	Open	High	Low	Close	Adj Close	Volume	Name	Day_Perc_Change
0	2018-04-02	548.00	578.95	545.90	576.25	562.83	2823880	CIPLA	NaN
1	2018-04-03	572.00	578.35	568.80	570.30	557.01	2493190	CIPLA	-1.034060
2	2018-04-04	572.35	575.45	564.50	569.80	556.53	1852997	CIPLA	-0.086174
3	2018-04-05	574.00	578.55	558.35	560.35	547.30	2396864	CIPLA	-1.658491
4	2018-04-06	560.30	571.00	557.25	558.95	545.93	2420324	CIPLA	-0.250320

```

1 CIPLA_df.dropna(axis = 0, inplace = True)
[96] ✓ 0.7s Python
...
1 CIPLA_plot_line=px.line(x="Date",y="Day_Perc_Change",data_frame=CIPLA_df,title="Percent Change Price Visualization")
2 CIPLA_plot_line.show()
[97] ✓ 0.1s Python

```



c. Trend Analysis

In this analysis, we categorize the trend based on percent change in closing price using a pivot table.

Analysis:

From the given plot we see that the adj closing price is mostly steady and does not see changes most of the times. This tells us the volatility in the price is low.

```

1 def trend(x):
2     if x > -0.5 and x <= 0.5:
3         return 'Slight or No change'
4     elif x > 0.5 and x <= 1:
5         return 'Slight Positive'
6     elif x > -1 and x <= -0.5:
7         return 'Slight Negative'
8     elif x > 1 and x <= 3:
9         return 'Positive'
10    elif x > -3 and x <= -1:
11        return 'Negative'
12    elif x > 3 and x <= 7:
13        return 'Among top gainers'
14    elif x > -7 and x <= -3:
15        return 'Among top losers'
16    elif x > 7:
17        return 'Bull run'
18    elif x <= -7:
19        return 'Bear drop'
20    PLA_df['Trend'] = np.zeros(CIPLA_df['Day_Perc_Change'].count())
21    CIPLA_df['Trend'] = CIPLA_df['Day_Perc_Change'].apply(lambda x:trend(x))
22    CIPLA_df.head()

```

✓ 0.1s

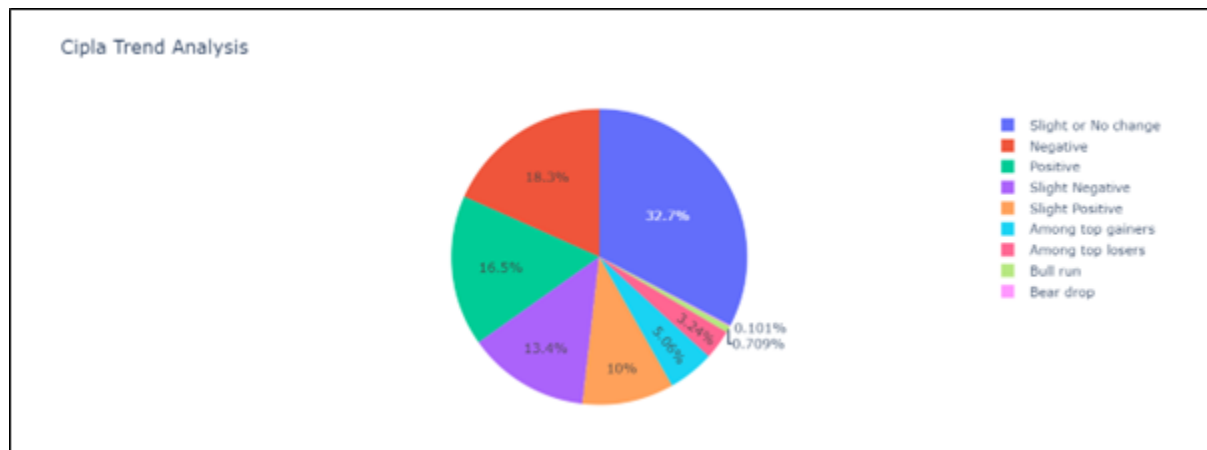
Unnamed: 0	Date	Open	High	Low	Close	Adj Close	Volume	Name	Day_Perc_Change	Trend
1	2018-04-03	572.00	578.35	568.80	570.30	557.01	2493190	CIPLA	-1.034060	Negative
2	2018-04-04	572.35	575.45	564.50	569.80	556.53	1852997	CIPLA	-0.086174	Slight or No change
3	2018-04-05	574.00	578.55	558.35	560.35	547.30	2396864	CIPLA	-1.658491	Negative
4	2018-04-06	560.30	571.00	557.25	558.95	545.93	2420324	CIPLA	-0.250320	Slight or No change
5	2018-04-09	565.90	565.90	555.50	556.75	543.78	1470765	CIPLA	-0.393823	Slight or No change

```

1 pie_plot=px.pie(CIPLA_df,names='Trend',title="Cipla Trend Analysis")
2 pie_plot.show()

```

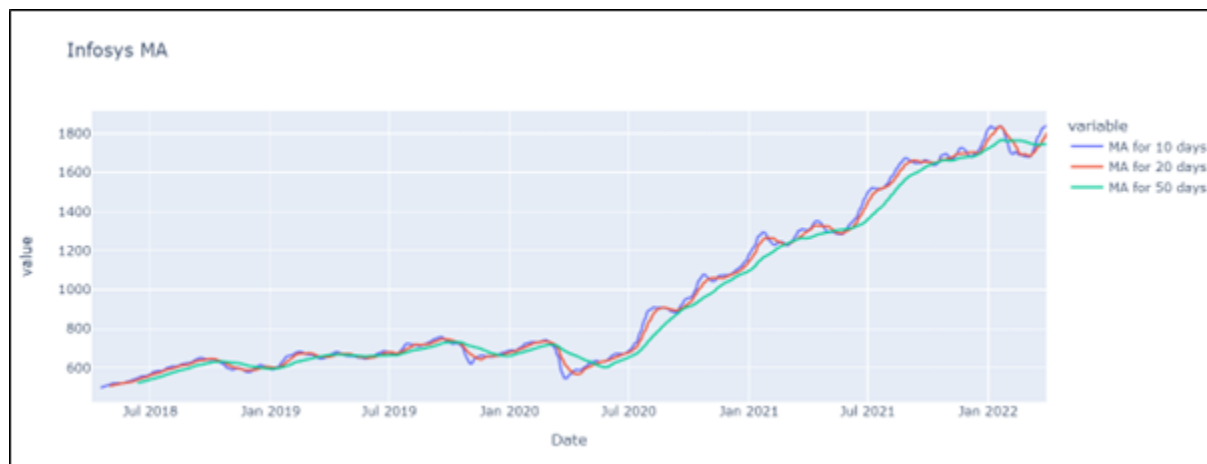
[108] ✓ 0.2s



Ø STOCK – INFOSYS LTD

a. Trend Analysis Using SMA (Simple Moving Average)

Analysis: As you can see before Covid in March 2020, the stock price was steady and then in March 2020 Infosys price stock dropped slightly and then the stock price saw bullish run.



b. Day to Day Analysis

Analysis: It shows the minimum negative change in price was -15% on October 22, 2019. And the maximum positive change in price was 12% change on March 24, 2020.

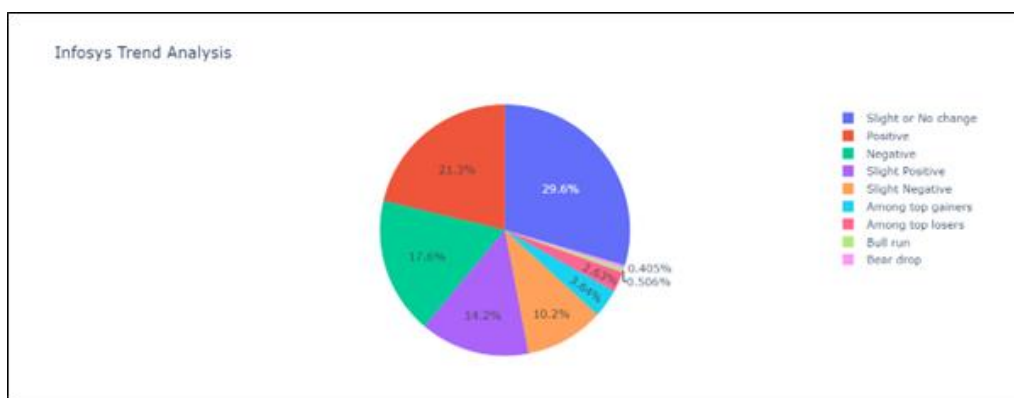


c. Trend Analysis

In this analysis, we categorize the trend based on percent change in closing price using a pivot table.

Analysis:

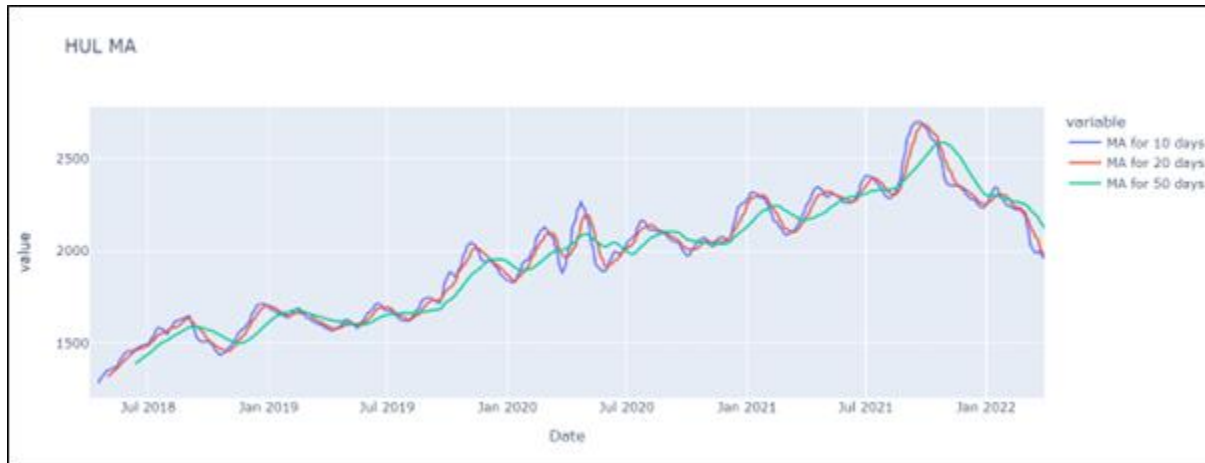
From the given plot we see that the change in adj closing price is mostly steady and positive. This tells us the price is either staying same or increasing.



Ø STOCK – HINDUSTAN UNILEVER LTD

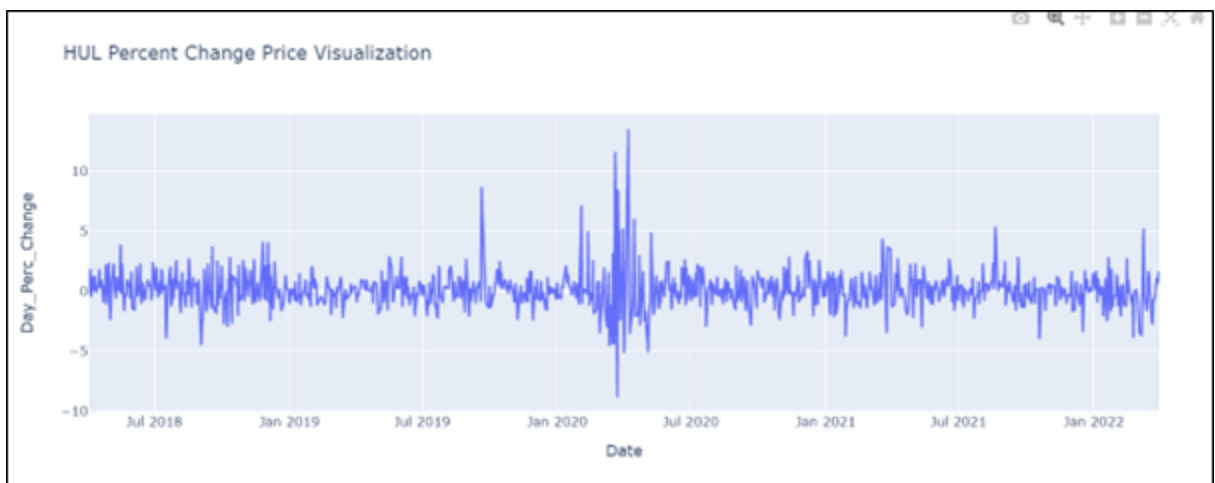
a. Trend Analysis Using SMA (Simple Moving Average)

Analysis: From the graph you can see that there is no bearish drop in the trend.



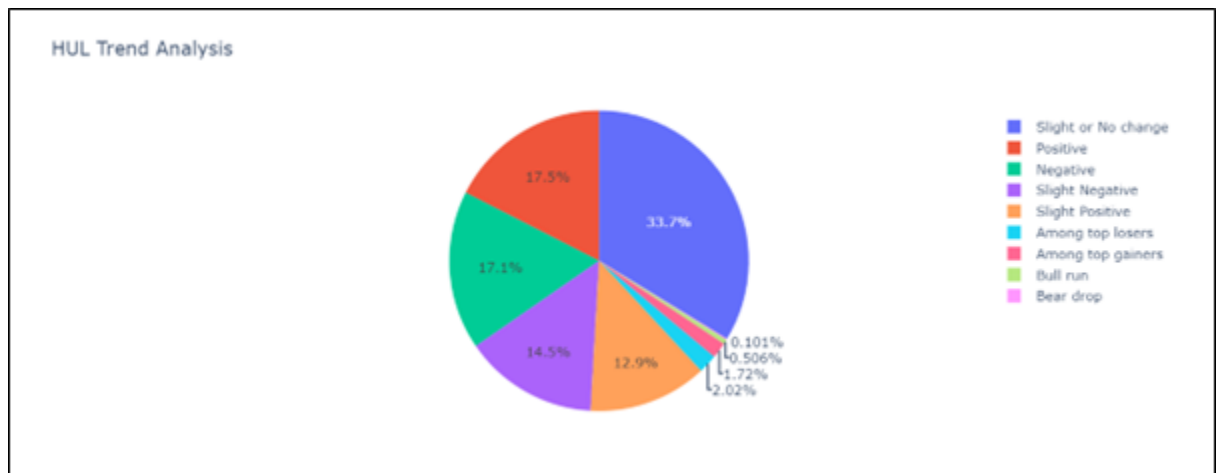
b. Day to Day Analysis

Analysis: It shows the minimum negative change in price was -8.8% on 22 March, 2020. And the maximum positive change in price was 13% change on 7 April, 2020.



c. Trend Analysis

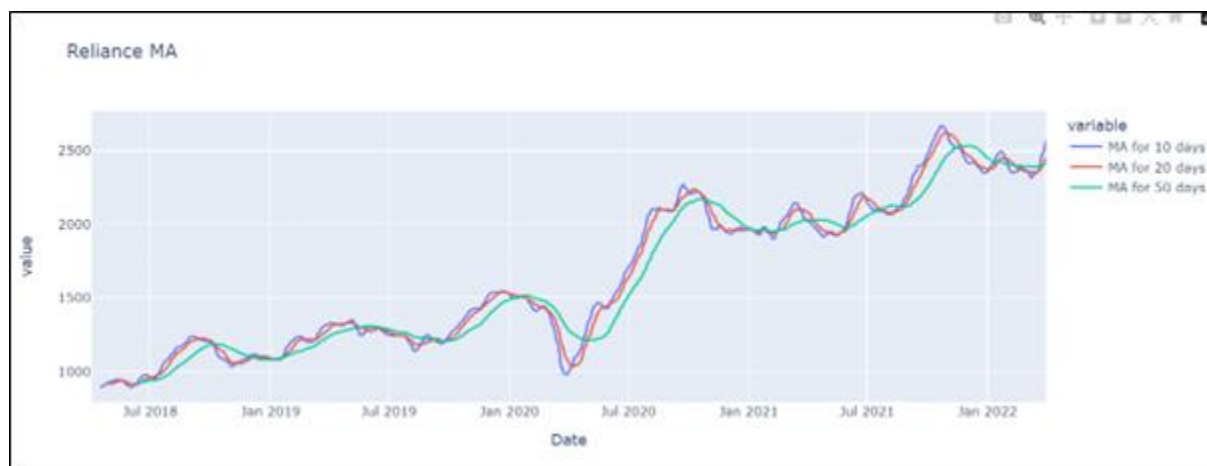
Analysis: The price is somewhat maintained as there are mostly positive or no change in the trend.



Ø STOCK – RELIANCE LTD

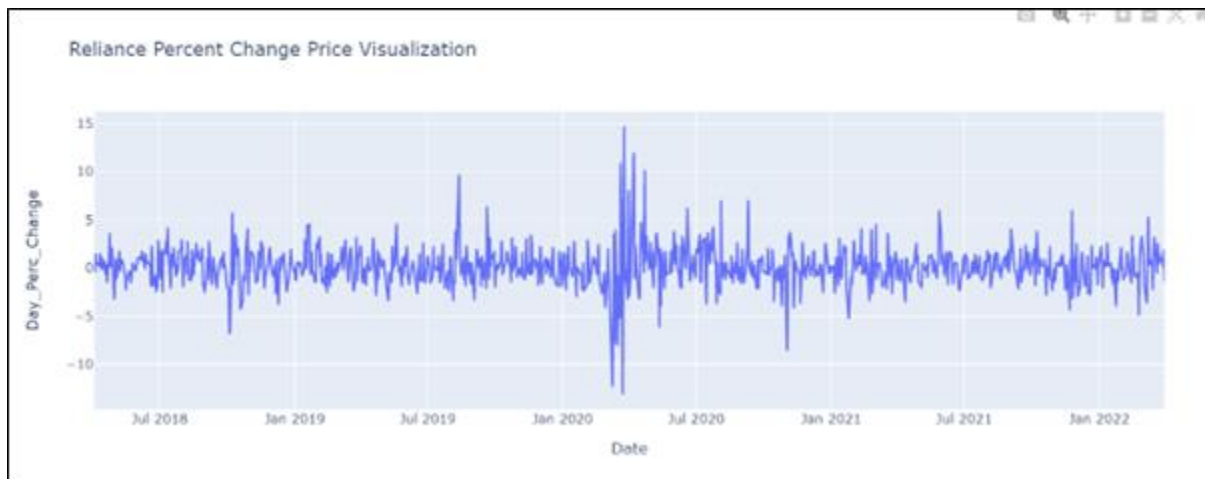
a. Trend Analysis Using SMA (Simple Moving Average)

Analysis: As you can see from the plot, Reliance stocks also faced a bear drop in the prices during covid period.



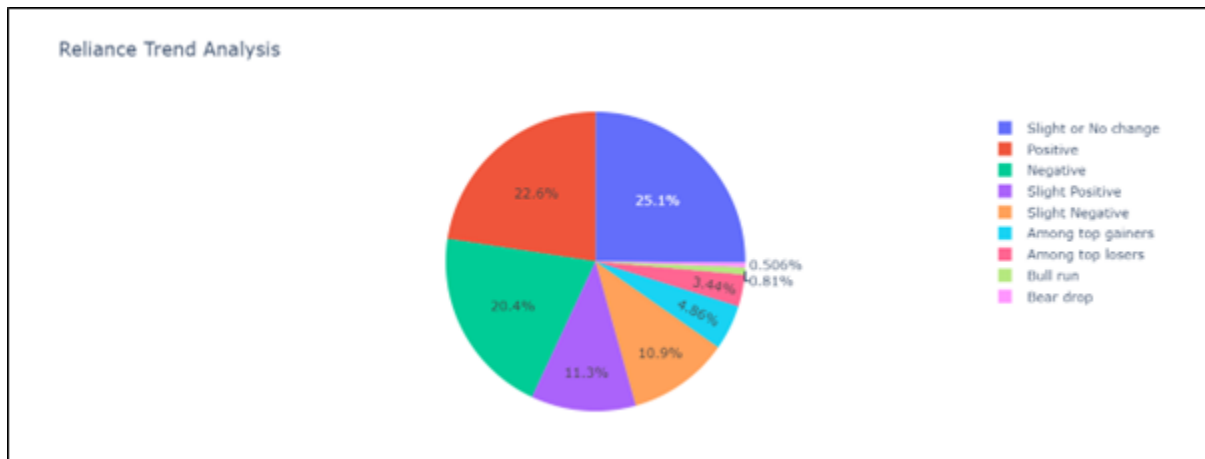
b. Day to Day Analysis

Analysis: It shows the minimum negative change in price was -13% on 23 March, 2020. And the maximum positive change in price was 14% change on 25 March, 2020.



c. Trend Analysis

Analysis: From the pie chart we can conclude that the price is more fluctuating. That is the volatility is quite high.



4.2. CombinedAnalysis

a. Trading Analysis (Using Volume)

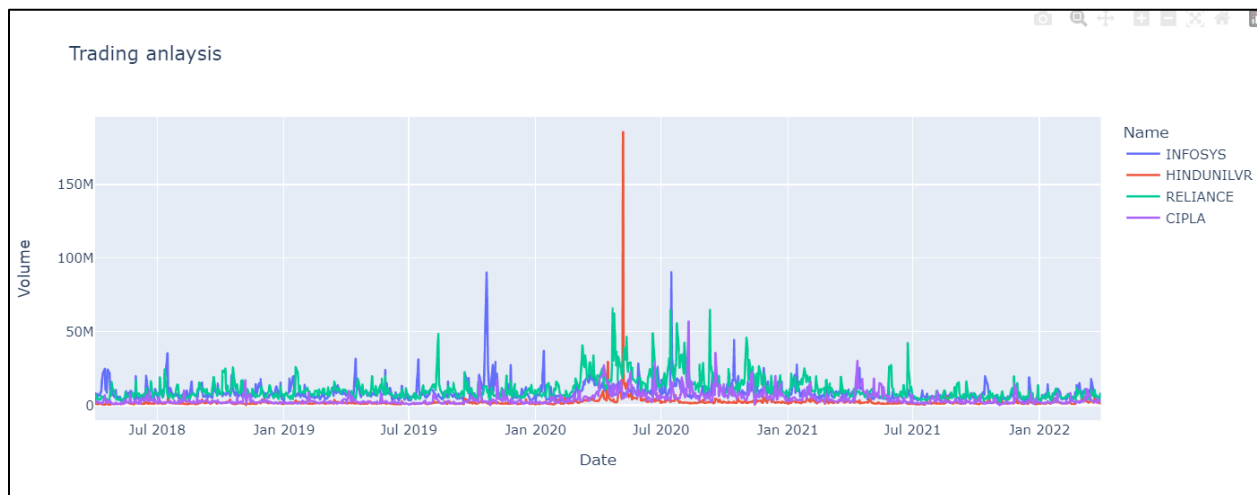
What Is Trading Analysis?

Trading Analysis uses the volume to see the trade comparisons of the different stocks.

Only volume or stock prices do not provide a comparison between companies. In this case, we have plotted a graph for Volume * Open price to better compare the companies.

```
1 data_all['MarketCap'] = data_all['Open']*data_all['Volume']
2 trade_plot=px.line(x='Date',y='Volume',data_frame=data_all,title="Trading anlysis",color="Name")
3 trade_plot.show()
```

✓ 1.6s



Analysis: As you can see from the plot Hindustan Unilever has the lowest trading in the market , with an abnormal hike on a particular day . whereas Reliance and Cipla does the comparatively similar amount of trading.

b. Combined Trend Analysis using SMA

In Combined Trend Analysis using SMA we calculate the simple moving average of open and close prices of 20 days for all the four companies.

Steps:

Open price SMA

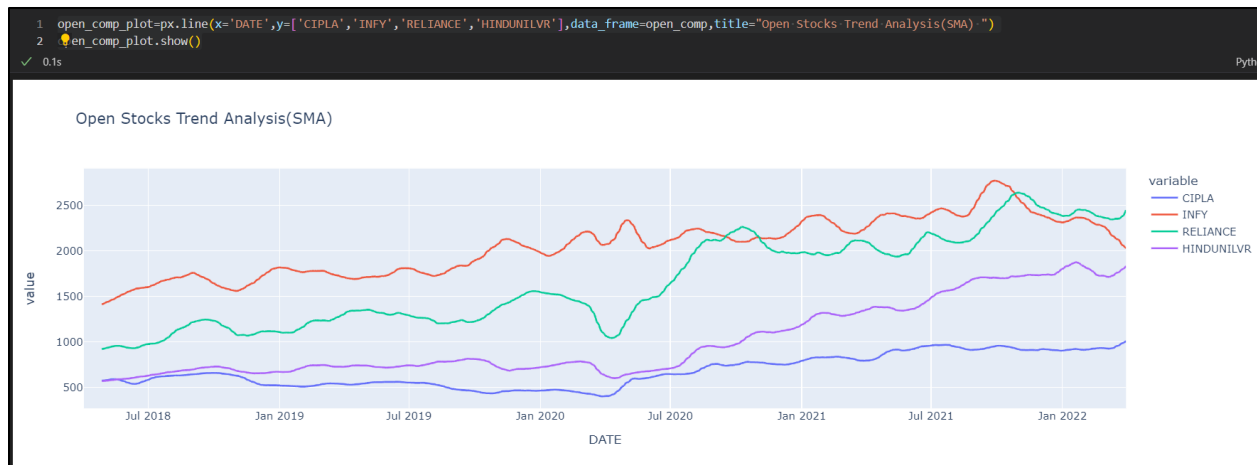
1. Creating a new column in every company as Open Price Rolling Mean (SMA20 OPEN) which calculates the SMA of open prices for 20 days.
2. Concatenating all the new columns in a data frame called open_comp
3. Dropping the null values.
4. Plotting a line graph using the open_comp

```
1 CIPLA_df['SMA20 OPEN'] = CIPLA_df['Open'].rolling(20).mean()
2 HINDUNILVR_df['SMA20 OPEN'] = HINDUNILVR_df['Open'].rolling(20).mean()
3 RELIANCE_df['SMA20 OPEN'] = RELIANCE_df['Open'].rolling(20).mean()
4 INFY_df['SMA20 OPEN'] = INFY_df['Open'].rolling(20).mean()
5
6
✓ 0.1s
```

```
1 open_comp=pd.concat([CIPLA_df['Date'],CIPLA_df['SMA20 OPEN'],HINDUNILVR_df['SMA20 OPEN'],RELIANCE_df['SMA20 OPEN'],INFY_df['SMA20 OPEN']],axis=1)
2 open_comp.columns=['DATE','CIPLA','INFY','RELIANCE','HINDUNILVR']
3 open_comp=pd.DataFrame(open_comp)
4 open_comp.dropna()
5
6
✓ 0.1s
```

	DATE	CIPLA	INFY	RELIANCE	HINDUNILVR
19	2018-04-27	574.4700	1413.6675	922.0660	570.8635
20	2018-04-30	577.1800	1421.6675	926.4745	571.8810
21	2018-05-02	579.0800	1429.9125	930.2390	573.4860
22	2018-05-03	581.1125	1435.7925	933.8650	574.8170
23	2018-05-04	583.0175	1439.5425	936.7255	575.9530
...
984	2022-03-25	985.5850	2067.8025	2378.3500	1797.1150
985	2022-03-28	991.6950	2055.8000	2393.6000	1805.9100
986	2022-03-29	997.5550	2045.0500	2411.4950	1813.8550
987	2022-03-30	1003.7025	2036.8500	2431.3400	1825.2450
988	2022-03-31	1009.6475	2031.3725	2447.8650	1834.7450

970 rows x 5 columns



Open Price Analysis:

Open Price of Infosys is all time high but the open prices drop in 2021. And Open price for Reliance sees a spike after a deep drop in prices in April 2020.

Close price SMA

1. Creating a new column in every company as Open Price Rolling Mean (SMA2 OPEN) which calculates the SMA of open prices for 20 days.
2. Concatenating all the new columns in a data frame called close_comp
3. Dropping the null values.
4. Plotting a line graph using the close_comp

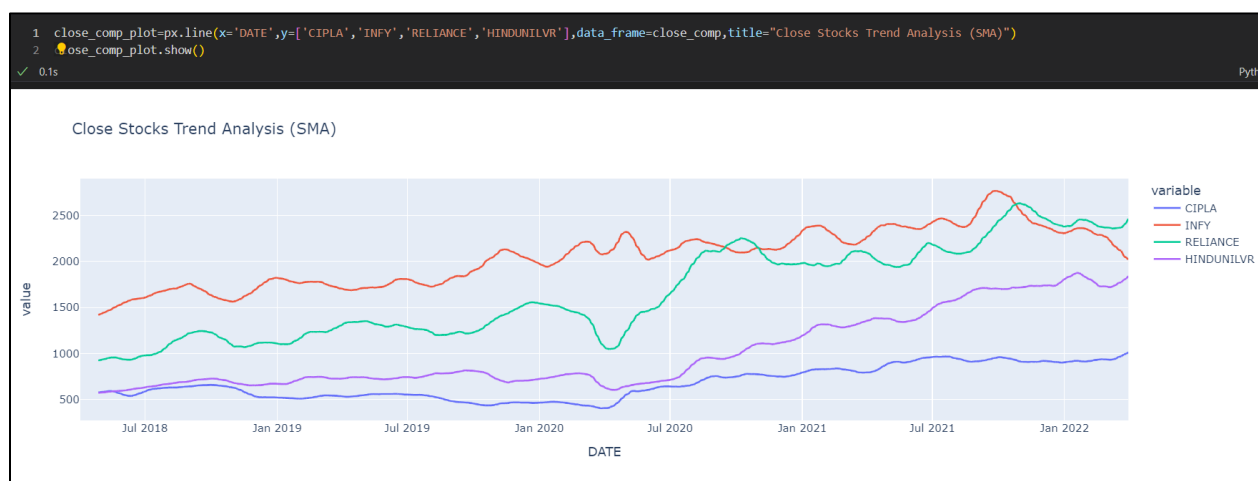
```

1 CIPLA_df['SMA20_CLOSE'] = CIPLA_df['Close'].rolling(20).mean()
2 HINDUNILVR_df['SMA20_CLOSE'] = HINDUNILVR_df['Close'].rolling(20).mean()
3 RELIANCE_df['SMA20_CLOSE'] = RELIANCE_df['Close'].rolling(20).mean()
4 INFY_df['SMA20_CLOSE'] = INFY_df['Close'].rolling(20).mean()
✓ 0.7s

1 close_comp=pd.concat([CIPLA_df['Date'],CIPLA_df['SMA20_CLOSE'],HINDUNILVR_df['SMA20_CLOSE'],RELIANCE_df['SMA20_CLOSE'],INFY_df['SMA20_CLOSE']],axis=1)
2 close_comp.columns=['DATE','CIPLA','INFY','RELIANCE','HINDUNILVR']
3 close_comp=pd.DataFrame(close_comp)
4 close_comp.dropna()
5
✓ 0.1s

```

	DATE	CIPLA	INFY	RELIANCE	HINDUNILVR
19	2018-04-27	577.0775	1420.2025	925.0190	572.9770
20	2018-04-30	578.6350	1428.0825	928.5030	574.5355
21	2018-05-02	580.7200	1434.2600	932.1265	575.9510
22	2018-05-03	582.6950	1438.9975	935.4300	577.4100
23	2018-05-04	584.5275	1443.0975	937.6960	578.0485
...
984	2022-03-25	990.8900	2053.7425	2400.0900	1803.2350
985	2022-03-28	996.9550	2043.9275	2418.4000	1813.0225
986	2022-03-29	1002.7150	2034.7450	2435.3300	1822.5925
987	2022-03-30	1007.9100	2026.9350	2451.0000	1832.0100
988	2022-03-31	1012.6575	2021.9125	2462.8100	1842.2125



Close Price Analysis: Close Price of Infosys is all time high but the close prices drop in 2021. And Close price for Reliance sees a spike after a deep drop in prices in April 2020.

Conclusion: The trend usually follows for both the open and the close prizes

c. Percentage Change in Price

```
1 CIPLA_df['percentage_change'] = (CIPLA_df['Close']/CIPLA_df['Close'].shift(1)) -1
2 INFY_df['percentage_change'] = (INFY_df['Close']/INFY_df['Close'].shift(1))-1
3 RELIANCE_df['percentage_change'] = (RELIANCE_df['Close']/RELIANCE_df['Close'].shift(1)) - 1
4 HINDUNILVR_df['percentage_change'] = (HINDUNILVR_df['Close']/HINDUNILVR_df['Close'].shift(1)) -1
```

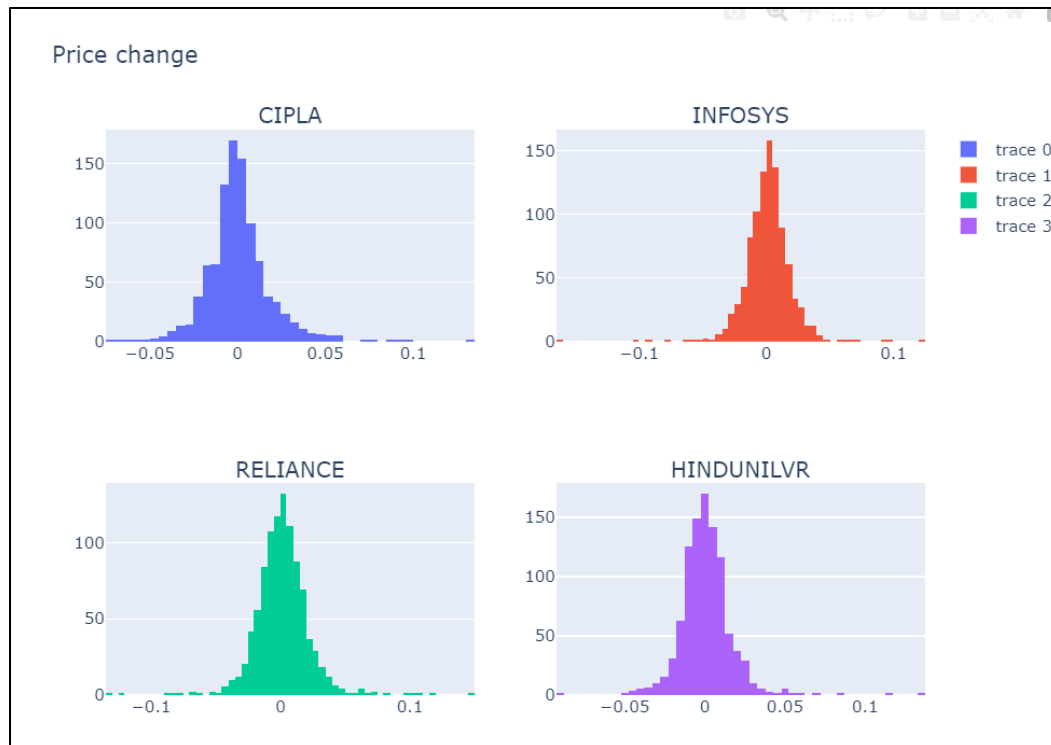
✓ 0.7s

Analysis : From all the histograms we can see the percentage change is most

```
1 from plotly.subplots import make_subplots
2 import plotly.graph_objects as go
3
4 fig = make_subplots(rows=2, cols=2, subplot_titles=( "CIPLA", "INFOSYS", "RELIANCE", "HINDUNILVR"))
5
6 fig.add_trace(
7     go.Histogram(x=CIPLA_df['percentage_change']),
8     row=1, col=1,
9
10 )
11
12 fig.add_trace(
13     go.Histogram(x=INFY_df['percentage_change']),
14     row=1, col=2
15 )
16 fig.add_trace(
17     go.Histogram(x=RELIANCE_df['percentage_change']),
18     row=2, col=1
19 )
20 fig.add_trace(
21     go.Histogram(x=HINDUNILVR_df['percentage_change']),
22     row=2, col=2
23 )
24
25 fig.update_layout(height=600, width=800, title_text="Price change")
26 fig.show()
27
```

✓ 0.1s

for Cipla.



d. Risk Analysis:

In Risk Analysis , we have computed the basic Risk analysis using the expected return and the volatility in the stocks as the risk.

Steps:

- 1.For daily return we computed the difference between Close price and Open Price
- 2.Found the indexes of every company with respect to name from the combined csv.
- 3.Using the location, we found the expected return and standard deviation for every company.
4. Made another Data frame called data having the expected return and standard deviation for all the companies.
5. Plotted the scatterplot between the expected Return and Risk.


```

1 #Calculating Daily Return
2 def diff(a, b):
3     return b - a
4 data_all['Daily Return'] = data_all.apply(
5     lambda x: diff(x['Adj Close'], x['Open']), axis=1)
6 data_all

```

✓ 0.1s

```

1 #calculating mean and standard deviation of the daily return
2 #mean and std is the mean of the daily return
3
4 imean=data_all['Daily Return'].iloc[0:989].mean()
5 print(imean)
6 isd=data_all['Daily Return'].iloc[0:989].std(axis=0)
7 print(isd)
8

```

✓ 0.7s

In such a way take out mean of other 3 companies.

```

1 data={'Name':['RELIANCE','HINDUNILVR','CIPLA','INFOSYS'],
2     'Expected Return':[rmean,hmean,cmean,imean], #mean of the daily return of each company
3     'Risk':[rsd,hsd,csd,isd], #standard deviation of the each company
4     }
5 Correlation_data=pd.DataFrame(data)
6 Correlation_data
7

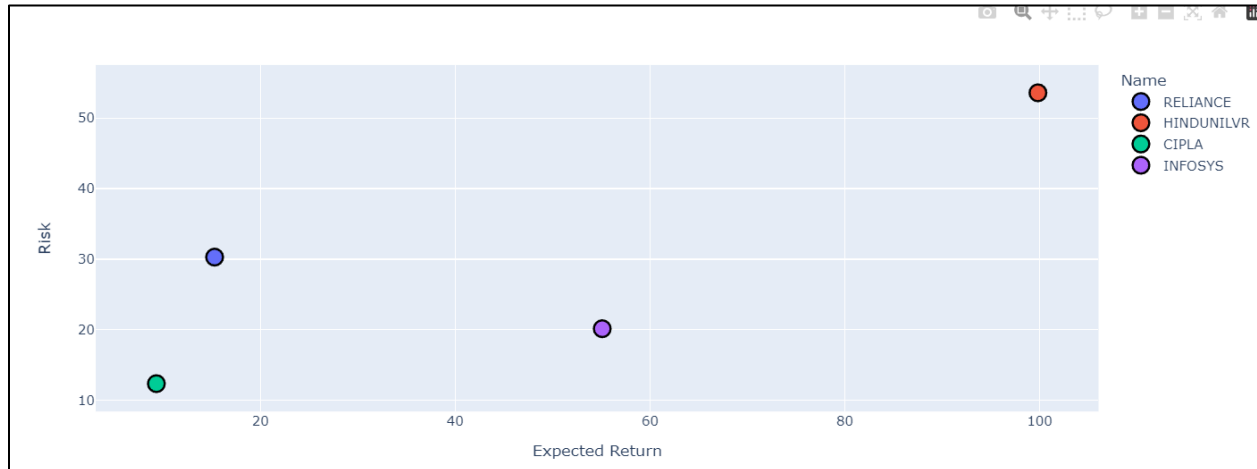
```

✓ 0.7s

	Name	Expected Return	Risk
0	RELIANCE	15.299606	30.284469
1	HINDUNILVR	99.827118	53.578675
2	CIPLA	9.321355	12.348913
3	INFOSYS	55.096936	20.134598

```
1 scatter_plot= px.scatter(Correlation_data, x="Expected Return", y="Risk", color="Name")
2 scatter_plot.update_traces(marker=dict(size=15, line=dict(width=2, color='Black')))
3 scatter_plot.show()
```

✓ 0.1s



Analysis :

From the scatter plot we conclude the following result:

Cipla Ltd: Low Risk Low Return

Reliance Ltd: Medium Risk Low Return

Infosys Ltd: Low Risk Medium Return

Hindustan Unilever Ltd: High Risk High Return

From the above analysis we can conclude that for steady profit investing in Infosys Ltd is advisable. But for high risky profit investing in Hindustan Unilever should be preferred.

5. GUI

Dash-

Dash is **an open source framework for building data visualization interfaces.**

Dash helps us to Build a UI Dashboard with the help of reactive programming. It helps us by implementing the lengthy UI Html code.

Three technologies constitute the core of Dash:

1. **Flask** supplies the web server functionality.
2. **React.js** renders the user interface of the web page.
3. **Plotly.js** generates the charts used in your application

5.1 Flow of using Dash

1. installation

- 1.1. pip install dash (on the cmd /PowerShell/terminal)

2. Layout

2.1. create a file named app.py , copy the code below into it, and then run it with python app.py

2.1.1. app = Dash(__name__)

2.2. Create a UI using DCC (Dash Core Components) and HTML attributes that are provide in dcc and html libraries.

3. Callbacks (reactive programing)

3.1. Callbacks add life to the static UI page created in Dash and make it reactive.

3.2. It takes 2 attributes as Arguments

3.2.1. `Output(component_id='my-output',
component_property='children'),`

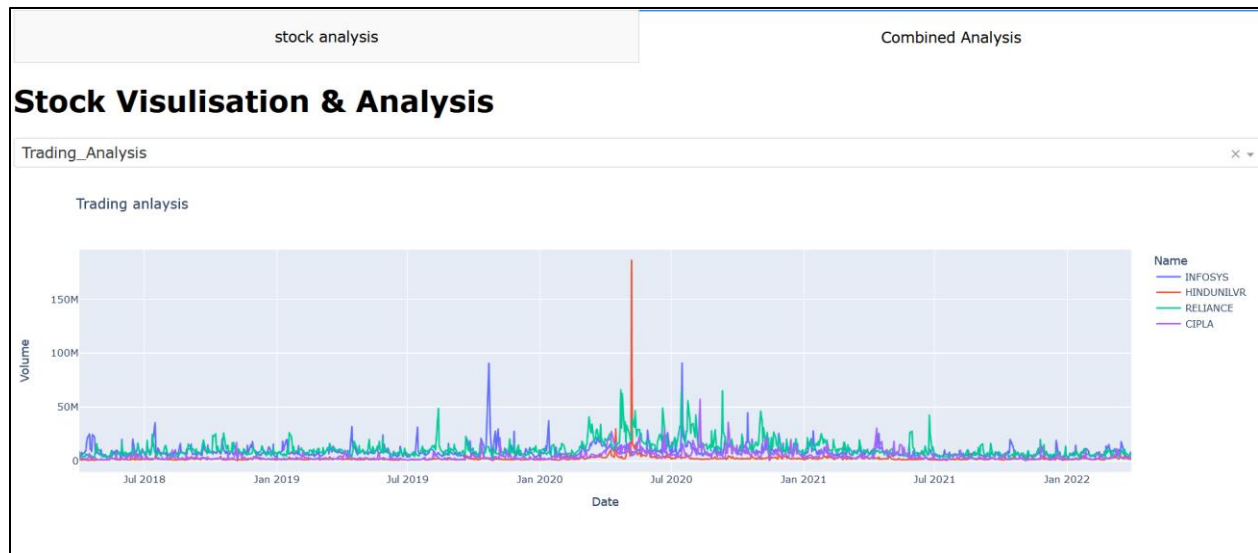
3.2.2. `Input(component_id='my-input',
component_property='value')`

5.2 Layout

Layout of the Dash app is designed with the basic logic of HTML. It uses dash core components to design input attributes like dropdown list, text box, radio button, check box buttons and much more. It is also used to display and format output containers like graphs, images or values.

At the same time it also uses Html tag attribute to print and format the appearance of the app. Both DCC and HTML tags support CSS parameters list as arguments for better appearance of the app.





5.3 Callback

Callbacks are used to make function dynamic and responsive in nature. It calls user defined functions with the input values from the DCC components to return something at the end as output from the function which is returned to the Dcc or Html components.

It uses the concept of Decorators to make program reactive. Decorators are used to call some specific function before and after a function (process is known as decorating a function).

Here update_output_div is function and @app.callback is the decorator.

Syntax:

```
@app.callback(
```

```
    Output(component_id='my-output', component_property='children'),
```

```
    Input(component_id='my-input', component_property='value')
```

```
)  
  
def update_output_div(input_value):  
    return f'Output: {input_value}'
```

6.CONCLUSION :

From all the analysis we come to the conclusion that in certain covid period there was a bearish run but then there was a significant price in growth after that .

7.LEARNINGS :

From the project we have learned various libraries mentioned above. The complete process of Data Analysis has been performed in this project including the steps of data importing and cleaning, Exploratory Data Analysis, Visualization , and creating Analytic Dashboards.