Import libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import matplotlib.dates as mdates
from datetime import datetime
```

Import TCS Historical Data

```
import io
import os
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in browser session. Please rerun this cell to enable.
Saving TCS.NS.csv to TCS.NS (1).csv

```
df=pd.read_csv(io.BytesIO(uploaded['TCS.NS.csv']))
df.head()
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	04-04-16	1228.000000	1244.000000	1223.500000	1235.625000	1114.676147	1752352
1	05-04-16	1235.400024	1242.449951	1221.025024	1231.375000	1110.841919	1491794
2	06-04-16	1235.000000	1244.500000	1229.675049	1240.949951	1119.479370	1763150
3	07-04-16	1240.000000	1242.349976	1228.500000	1235.400024	1114.472900	2390942
4	08-04-16	1233.000000	1236.849976	1213.500000	1216.025024	1096.994385	1399500

```
df.tail()
```

	Date	Open	High	Low	Close	Adj Close	Volume

```
df.index
```

```
RangeIndex(start=0, stop=1234, step=1)
```

```
df.shape
```

```
(1234, 7)
```

```
df.describe()
```

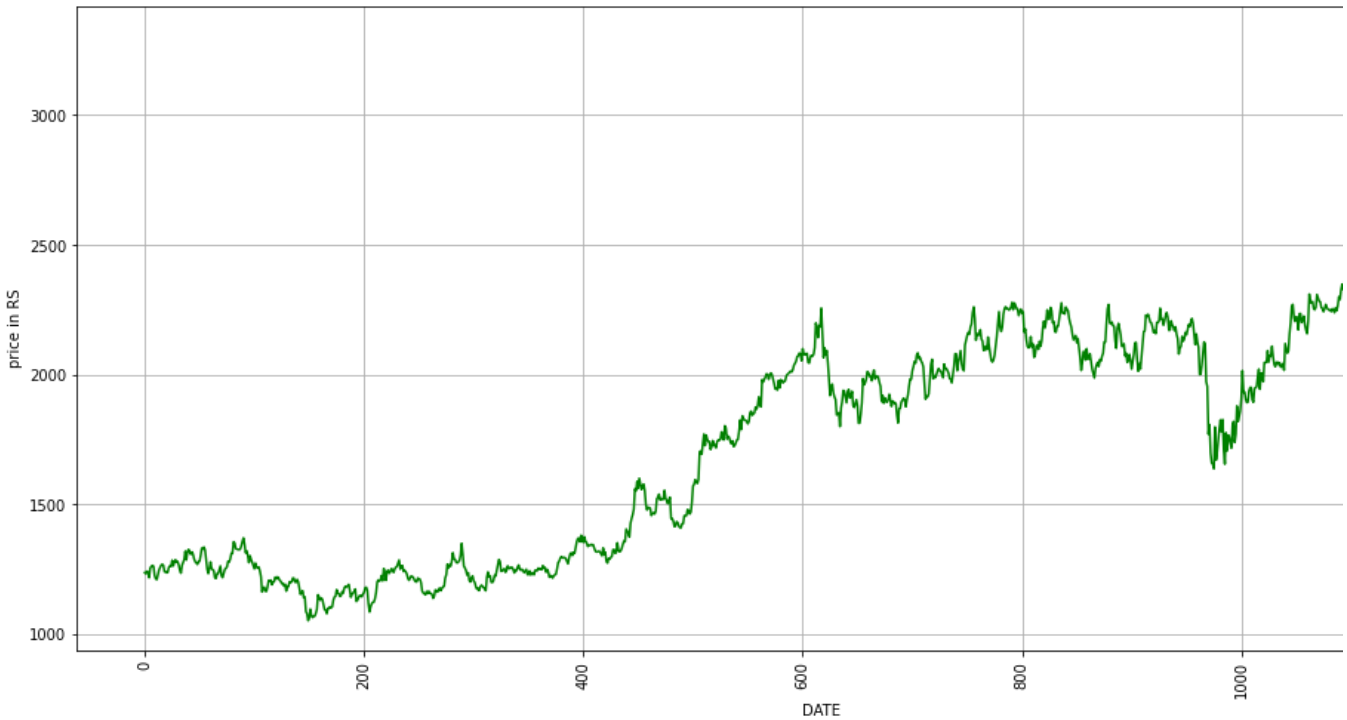
	Open	High	Low	Close	Adj Close	Volume
count	1234.000000	1234.000000	1234.000000	1234.000000	1234.000000	1.234000e+03
mean	1818.294710	1838.521531	1797.333169	1817.775164	1742.247399	3.060165e+06
std	547.341651	553.701620	540.114141	546.444437	573.711487	3.097732e+06
min	1058.449951	1070.000000	1025.949951	1050.574951	962.629150	8.682200e+04
25%	1267.456207	1281.699982	1255.000000	1267.506287	1164.788361	1.810728e+06
50%	1899.500000	1909.200012	1871.700012	1892.975036	1800.393738	2.542279e+06
75%	2154.837463	2179.699951	2125.000000	2154.450074	2081.041937	3.577867e+06
max	3308.949951	3339.800049	3278.649902	3308.800049	3308.800049	8.806715e+07

```
df.isnull().sum()
```

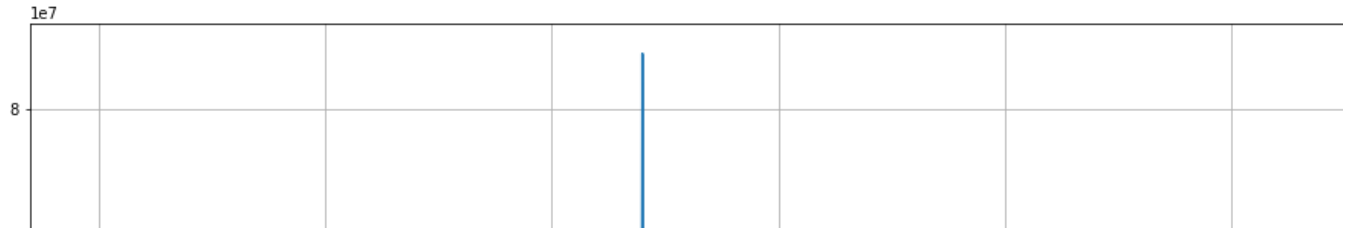
```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

```
plt.plot(df.index,df['Close'],color='g')
plt.xlabel('DATE')
plt.ylabel('price in RS')
plt.grid()
plt.rcParams["figure.figsize"] = (18,8)
plt.xticks(rotation=90)
```

```
plt.show()
```



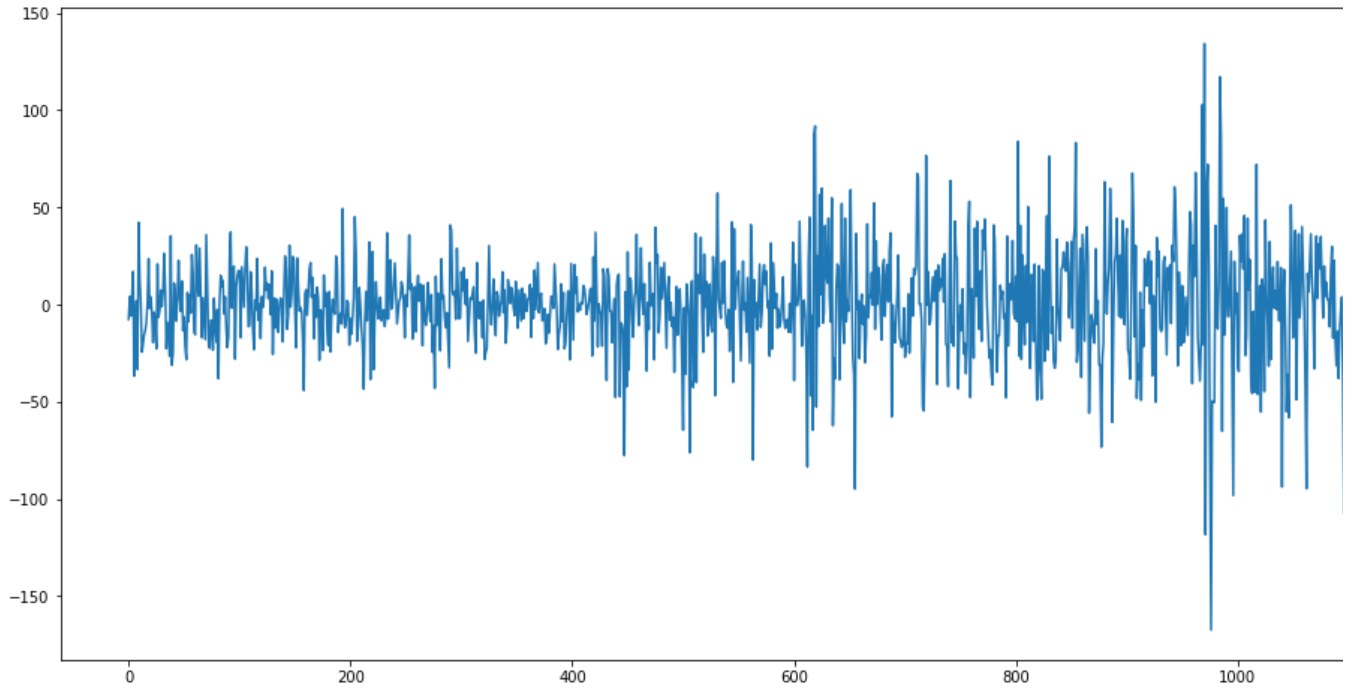

```
plt.plot(df.index,df['Volume'])
plt.rcParams["figure.figsize"] = (18,8)
plt.grid()
```




```
df['diff']=df['Open']-df['Close']
df['diff'].head()
```

```
0    -7.625000
1     4.025024
2    -5.949951
3     4.599976
4    16.974976
Name: diff, dtype: float64
```

```
plt.plot(df['diff'])
plt.rcParams["figure.figsize"] = (18,8)
```

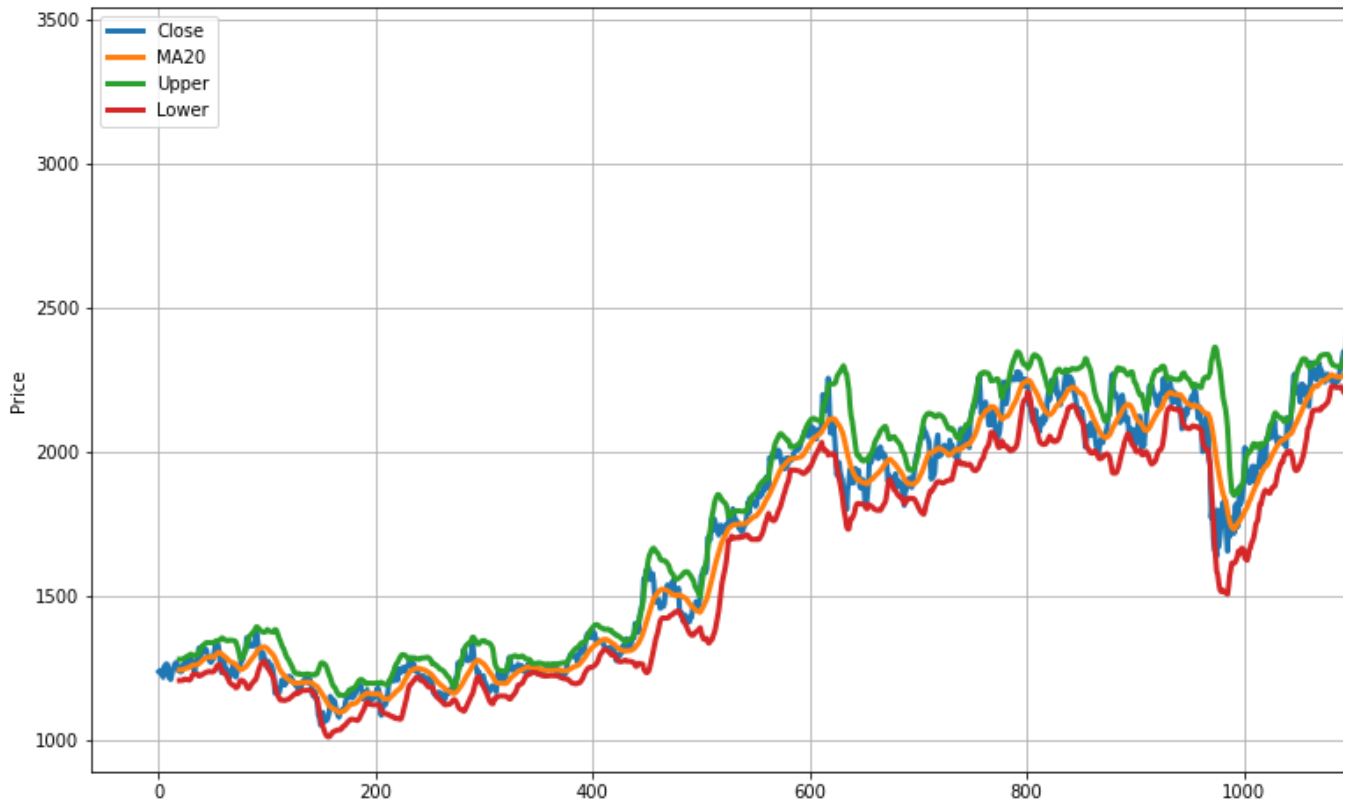



```
df['MA20'] = df['Close'].rolling(window=20).mean()
df['20dSTD'] = df['Close'].rolling(window=20).std()
```

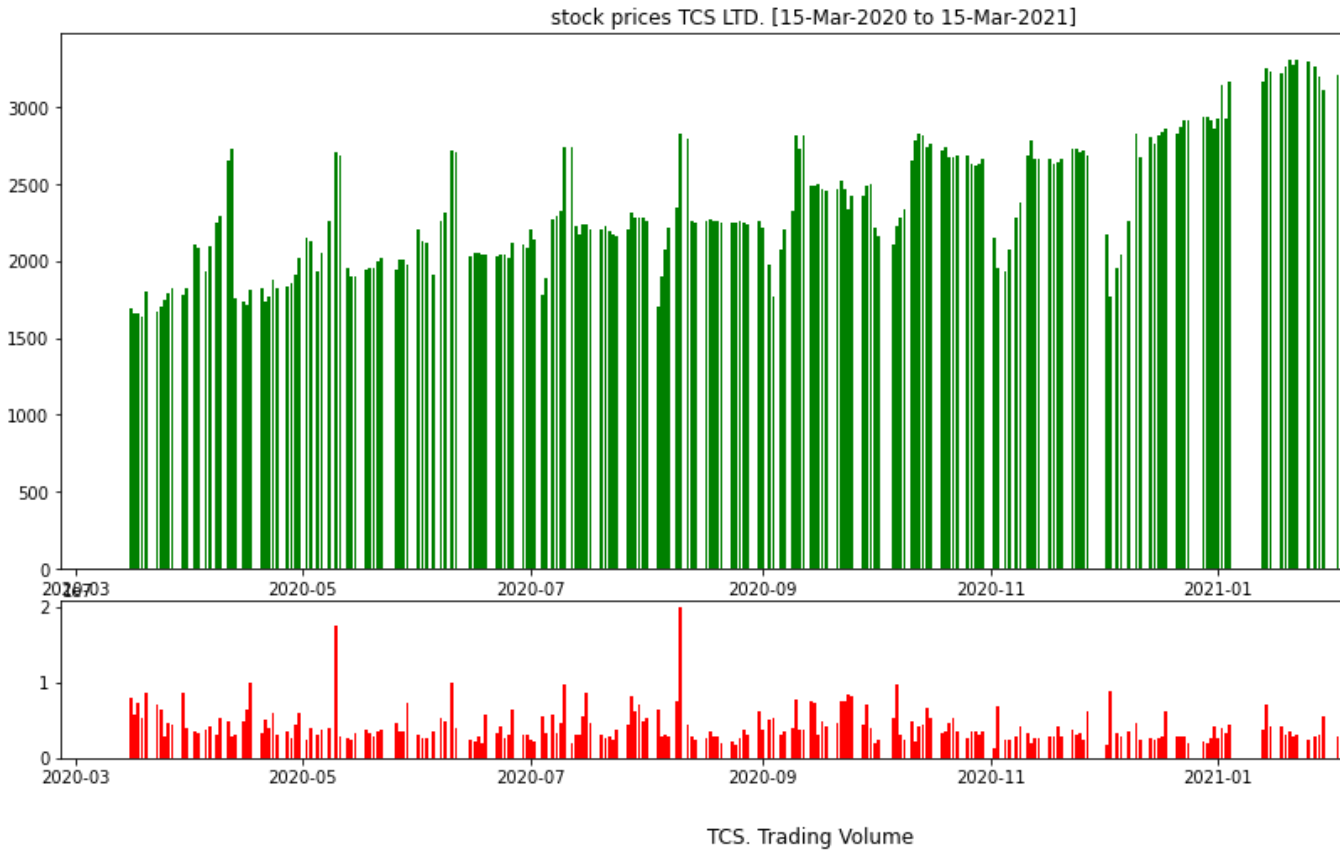
```
df['Upper'] = df['MA20'] + (df['20dSTD'] * 2)
df['Lower'] = df['MA20'] - (df['20dSTD'] * 2)
```

```
df[['Close','MA20','Upper','Lower']].plot(figsize=(15,8),linewidth=3)
plt.grid(True)
plt.axis('tight')
plt.ylabel('Price')
```

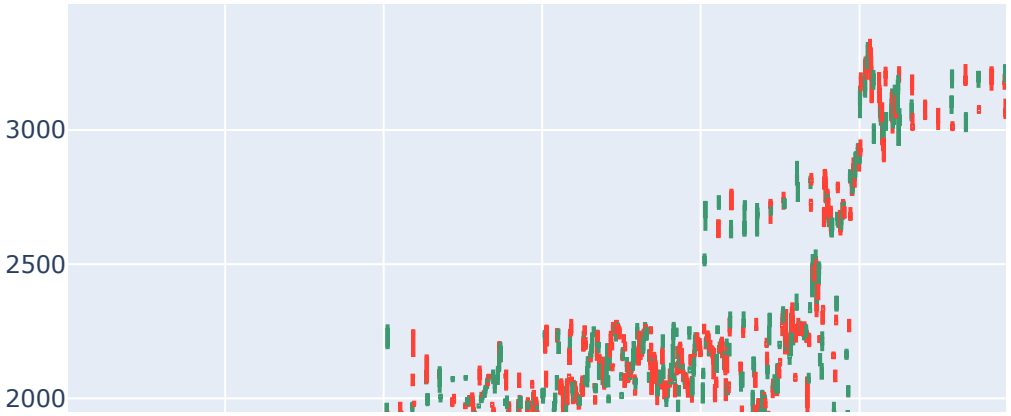
```
Text(0, 0.5, 'Price')
```




```
start_date = pd.to_datetime('2020-03-15')
end_date = pd.to_datetime('2021-03-15')

df['Date'] = pd.to_datetime(df['Date'])
new_df = (df['Date']>= start_date) & (df['Date']<= end_date)
df1 = df.loc[new_df]

stock_data = df1.set_index('Date')
top_plt = plt.subplot2grid((5,4), (0, 0), rowspan=3, colspan=4)
top_plt.bar(stock_data.index, stock_data["Close"], color='g')

plt.title(' stock prices TCS LTD. [15-Mar-2020 to 15-Mar-2021]')
bottom_plt = plt.subplot2grid((5,4), (3,0), rowspan=1, colspan=4)
bottom_plt.bar(stock_data.index, stock_data['Volume'], color='r')
plt.title('\nTCS. Trading Volume', y=-0.60)
plt.gcf().set_size_inches(16,10)
```




```
import plotly.graph_objects as go


fig = go.Figure(data=[go.Candlestick(x=df['Date'],
                open=df['Open'], high=df['High'],
                low=df['Low'], close=df['Close'])
                     ])

fig.update_layout(xaxis_rangeslider_visible=False)
fig.show()
```




```
def uptrend():
    print('\n\t\tUPTREND\t\n')
    print('Retracements\t\tExtensions\n')
    for r,e in zip(retracements,extensions):
        print(r,' = ',round((high - (diff * r/100)),2),'\t', e, ' = ', round((low + (diff * e
```

```
def downtrend():
    print('\n\t\tDOWNTREND\t\n')
    print('Retracements\t\tExtensions\n')
    for r,e in zip(retracements,extensions):
        print(r,' = ',round((low + (diff * r/100)),2),'\t',e, ' = ',round((high - (diff * e/1
```

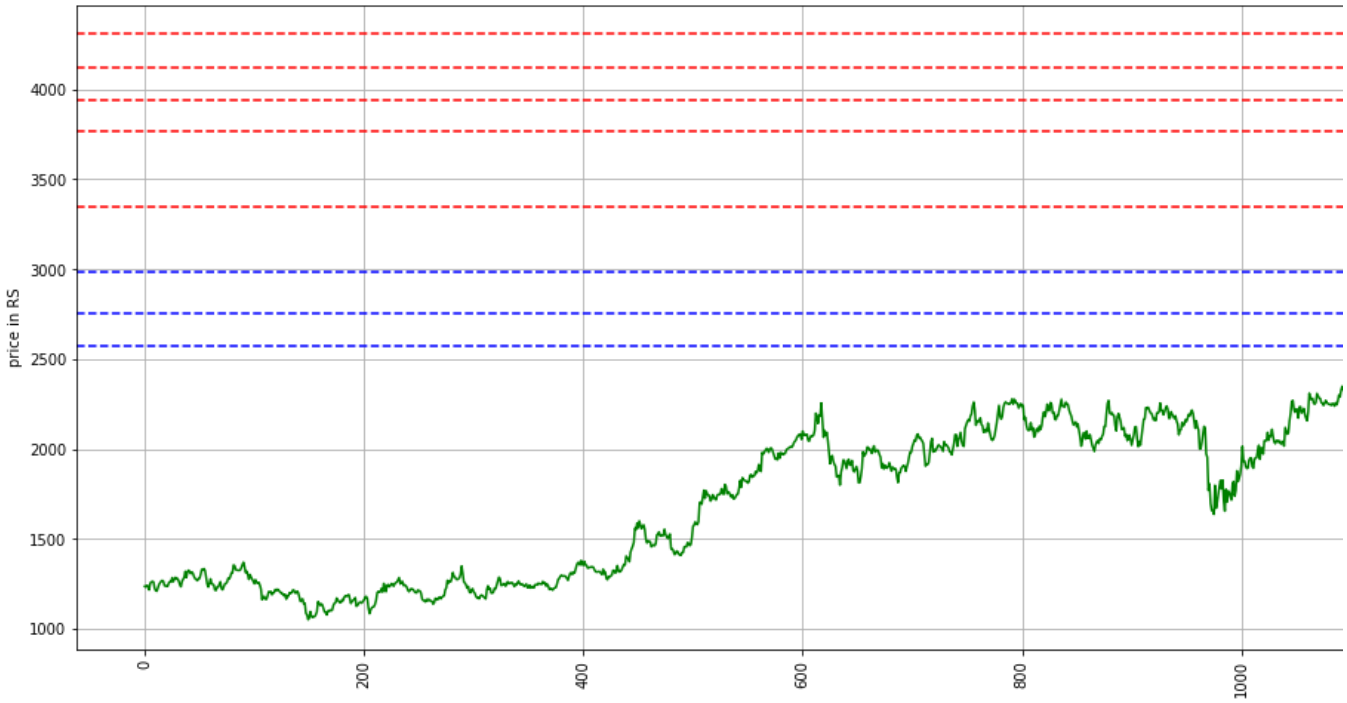
```
trend = input("What is the trend? up or down: ")
high = float(input('Enter High: '))
low = float(input('enter Low: '))
diff = high - low
print('The difference between High and Low is :',diff)

retracements = [23.6,38.2,50.00,61.8,76.4,78.6,85.40]
extensions = [127.2,138.2,150.00,161.8,176.4,261.8,423.6]

if high > low:
    if trend == 'up':
        uptrend()

    elif trend == 'down':
        downtrend()

    else:
        print("Enter 'up' or 'down'")
```

```
else:
    print('High entered must always be greater than Low entered!!!!!')
```

```
 What is the trend? up or down: up
 Enter High: 3350
 enter Low: 1800
 The difference between High and Low is : 1550.0

                 UPTREND

 Retracements            Extensions

 23.6  =  2984.2          127.2  =  3771.6

 38.2  =  2757.9          138.2  =  3942.1

 50.0  =  2575.0          150.0  =  4125.0

 61.8  =  2392.1          161.8  =  4307.9

 76.4  =  2165.8          176.4  =  4534.2

 78.6  =  2131.7          261.8  =  5857.9

 85.4  =  2026.3          423.6  =  8365.8
```

```
plt.plot(df.index,df['Close'],color='g')
plt.axhline(y = 3350, color = 'r', linestyle = '--')
plt.axhline(y = 3771.6, color = 'r', linestyle = '--')
plt.axhline(y = 3942.1, color = 'r', linestyle = '--')
plt.axhline(y = 4125.0, color = 'r', linestyle = '--')
plt.axhline(y = 4307.9 , color = 'r', linestyle = '--')
plt.axhline(y = 2984.2, color = 'b', linestyle = '--')
plt.axhline(y = 2757.9, color = 'b', linestyle = '--')
plt.axhline(y = 2575.0 , color = 'b', linestyle = '--')
plt.xlabel('DATE')
plt.ylabel('price in RS')
plt.grid()
plt.xticks(rotation=90)

plt.rcParams["figure.figsize"] = (18,8)
plt.show()
```

```
No handles with labels found to put in legend.
```




```
import fbprophet as fbp


df= df[["Date","Close"]]
df= df.rename(columns = {"Date":"ds","Close":"y"})
df.head()
```

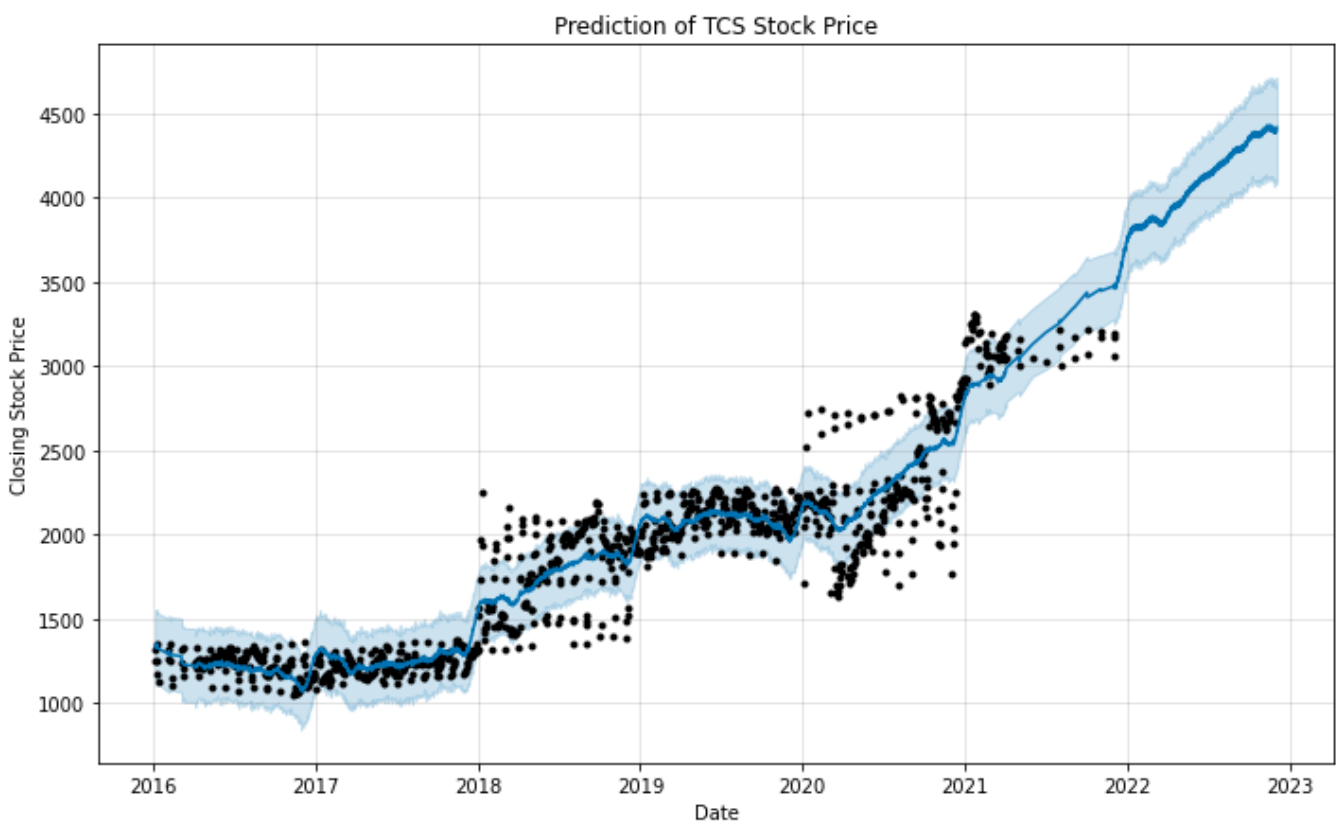
	ds	y
0	2016-04-04	1235.625000
1	2016-05-04	1231.375000
2	2016-06-04	1240.949951
3	2016-07-04	1235.400024
4	2016-08-04	1216.025024

```
from fbprophet import Prophet
m = Prophet(daily_seasonality=True)
m.fit(df)
```

```
<fbprophet.forecaster.Prophet at 0x7facf849cd50>
```

```
future = m.make_future_dataframe(periods=365)
predictions=m.predict(future)
m.plot(predictions)
plt.title("Prediction of TCS Stock Price")
plt.xlabel("Date")
```

```
plt.ylabel("Closing Stock Price")
plt.show()
```




```
import stocker

stocker.predict.tomorrow('TCS.NS')
```

```
[3069.3, 0.912, '2021-05-21']
```