```
# pip install yfinance
```

```
Defaulting to user installation because normal site-packages is not
writeableNote: you may need to restart the kernel to use updated
packages.

Requirement already satisfied: yfinance in c:\users\prudh\appdata\
roaming\python\python311\site-packages (0.2.36)
Requirement already satisfied: pandas>=1.3.0 in c:\programdata\
anaconda3\lib\site-packages (from yfinance) (2.0.3)
Requirement already satisfied: numpy>=1.16.5 in c:\programdata\
anaconda3\lib\site-packages (from yfinance) (1.24.3)
Requirement already satisfied: requests>=2.31 in c:\programdata\
anaconda3\lib\site-packages (from yfinance) (2.31.0)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\prudh\
appdata\roaming\python\python311\site-packages (from yfinance)
(0.0.11)
Requirement already satisfied: lxml>=4.9.1 in c:\programdata\
anaconda3\lib\site-packages (from yfinance) (4.9.3)
Requirement already satisfied: appdirs>=1.4.4 in c:\programdata\
anaconda3\lib\site-packages (from yfinance) (1.4.4)
Requirement already satisfied: pytz>=2022.5 in c:\programdata\
anaconda3\lib\site-packages (from yfinance) (2023.3.post1)
Requirement already satisfied: frozendict>=2.3.4 in c:\users\prudh\
appdata\roaming\python\python311\site-packages (from yfinance) (2.4.0)
Requirement already satisfied: peewee>=3.16.2 in c:\users\prudh\
appdata\roaming\python\python311\site-packages (from yfinance)
(3.17.1)
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\
programdata\anaconda3\lib\site-packages (from yfinance) (4.12.2)
Requirement already satisfied: html5lib>=1.1 in c:\users\prudh\
appdata\roaming\python\python311\site-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in c:\programdata\
anaconda3\lib\site-packages (from beautifulsoup4>=4.11.1->yfinance)
(2.4)
Requirement already satisfied: six>=1.9 in c:\programdata\anaconda3\
lib\site-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in c:\programdata\
anaconda3\lib\site-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\
programdata\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance)
(2.8.2)
Requirement already satisfied: tzdata>=2022.1 in c:\programdata\
anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\
programdata\anaconda3\lib\site-packages (from requests>=2.31-
>yfinance) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\
anaconda3\lib\site-packages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\
anaconda3\lib\site-packages (from requests>=2.31->yfinance) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\
anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2024.2.2)
```

```python
import yfinance as yf
```

```python
from datetime import datetime
end = datetime.now()
start = datetime(end.year-20, end.month, end.day)
```

```python
stock = "GOOG"
google_data = yf.download(stock, start, end)
```

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

```python
google_data.head()
```

```
                Open      High       Low     Close  Adj Close
Volume
Date

2004-08-19  2.490664  2.591785  2.390042  2.499133   2.499133
897427216
2004-08-20  2.515820  2.716817  2.503118  2.697639   2.697639
458857488
2004-08-23  2.758411  2.826406  2.716070  2.724787   2.724787
366857939
2004-08-24  2.770615  2.779581  2.579581  2.611960   2.611960
306396159
2004-08-25  2.614201  2.689918  2.587302  2.640104   2.640104
184645512
```

```python
google_data.shape
```

```
(4908, 6)
```

```python
google_data.describe()
```

```
              Open         High          Low        Close    Adj Close
\
count  4908.000000  4908.000000  4908.000000  4908.000000  4908.000000

mean     42.507736    42.957677    42.079534    42.527346    42.527346

std      39.718836    40.168350    39.320105    39.751283    39.751283

min       2.470490     2.534002     2.390042     2.490913     2.490913

25%      12.876112    13.003946    12.766586    12.900458    12.900458

50%      26.682630    26.891829    26.459303    26.673268    26.673268
```

```
75%      58.500500    58.924312    57.883374    58.420749    58.420749

max     154.009995   155.199997   152.919998   154.839996   154.839996
```

```
             Volume
count  4.908000e+03
mean   1.179327e+08
std    1.507837e+08
min    1.584340e+05
25%    2.818250e+07
50%    5.959000e+07
75%    1.459008e+08
max    1.650833e+09
```

```
google_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 4908 entries, 2004-08-19 to 2024-02-16
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       4908 non-null   float64
 1   High       4908 non-null   float64
 2   Low        4908 non-null   float64
 3   Close      4908 non-null   float64
 4   Adj Close  4908 non-null   float64
 5   Volume     4908 non-null   int64
dtypes: float64(5), int64(1)
memory usage: 268.4 KB
```

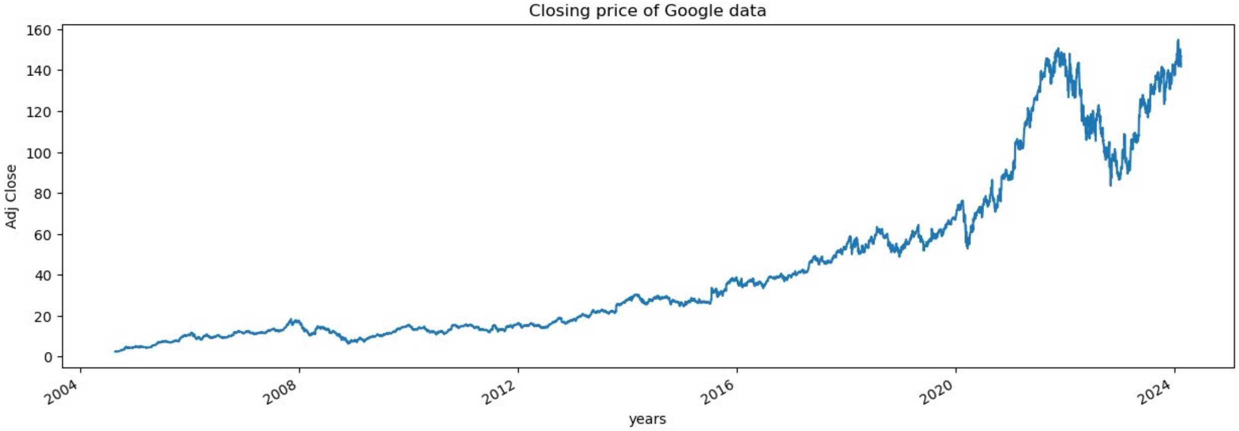
```
google_data.isna().sum()
```

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

```
import matplotlib.pyplot as plt
%matplotlib inline
```

```
plt.figure(figsize = (15,5))
google_data['Adj Close'].plot()
plt.xlabel("years")
plt.ylabel("Adj Close")
plt.title("Closing price of Google data")
```

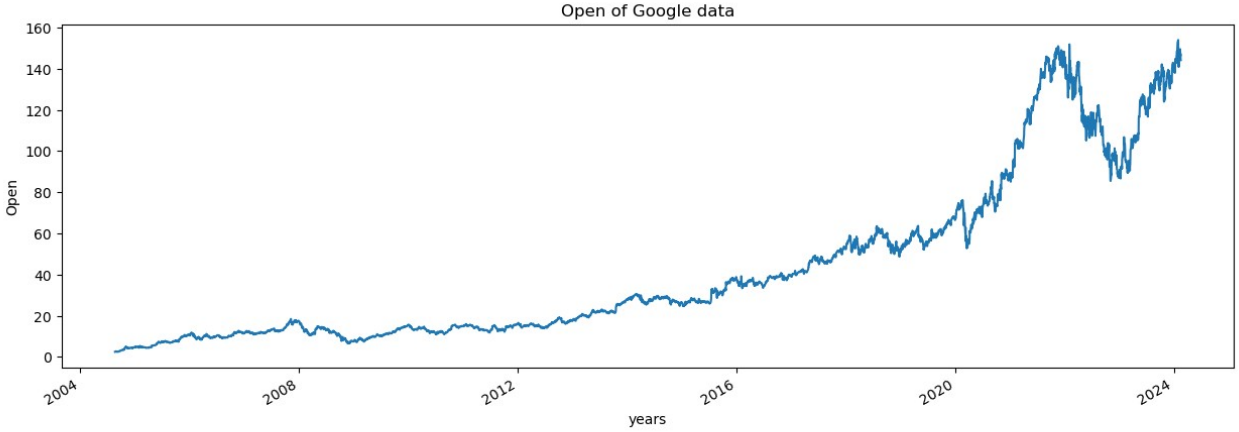
```
Text(0.5, 1.0, 'Closing price of Google data')
```

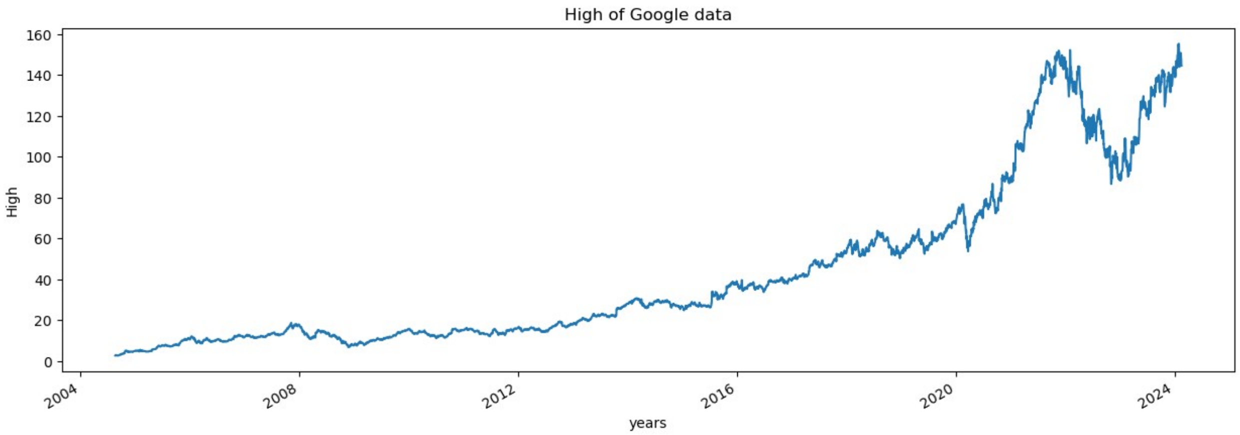


```
def plot_graph(figsize, values, column_name):
    plt.figure()
    values.plot(figsize = figsize)
    plt.xlabel("years")
    plt.ylabel(column_name)
    plt.title(f"{column_name} of Google data")


google_data.columns
```

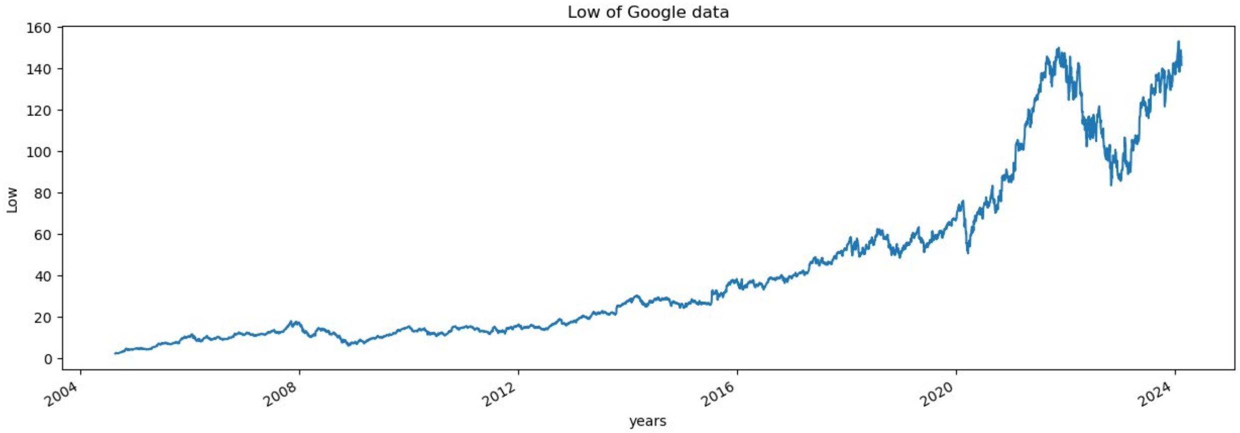
```
Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'],
dtype='object')
```

```
for column in google_data.columns:
    plot_graph((15,5),google_data[column], column)
```

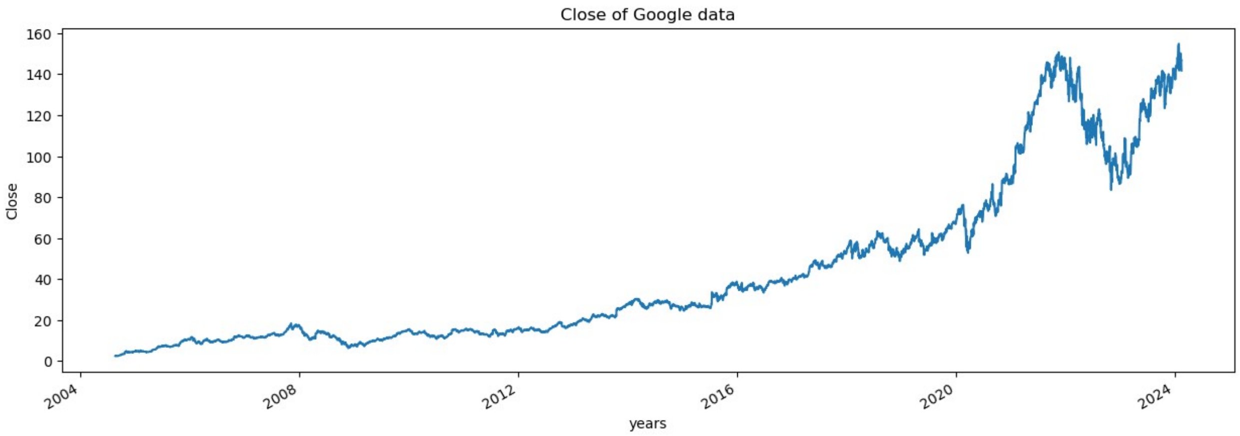




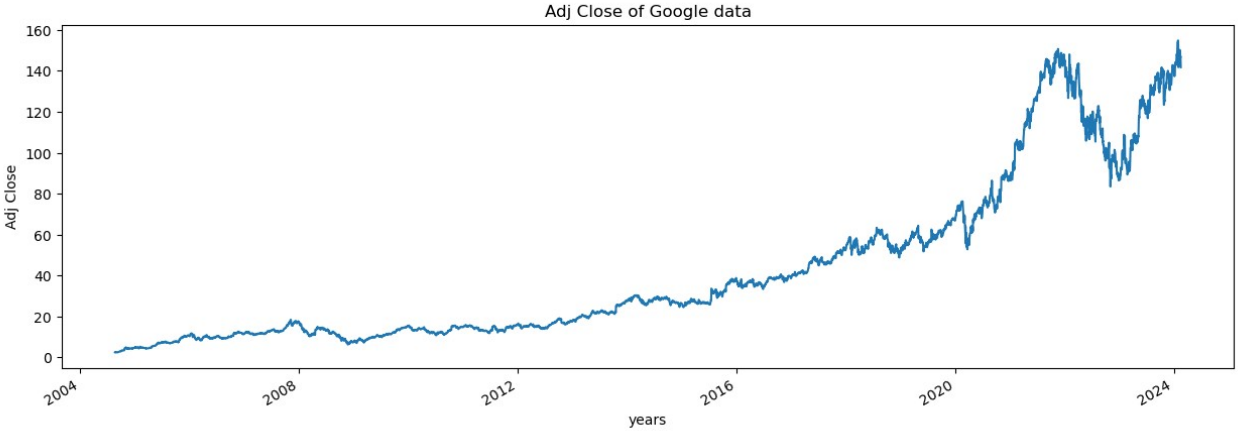
High of Google data

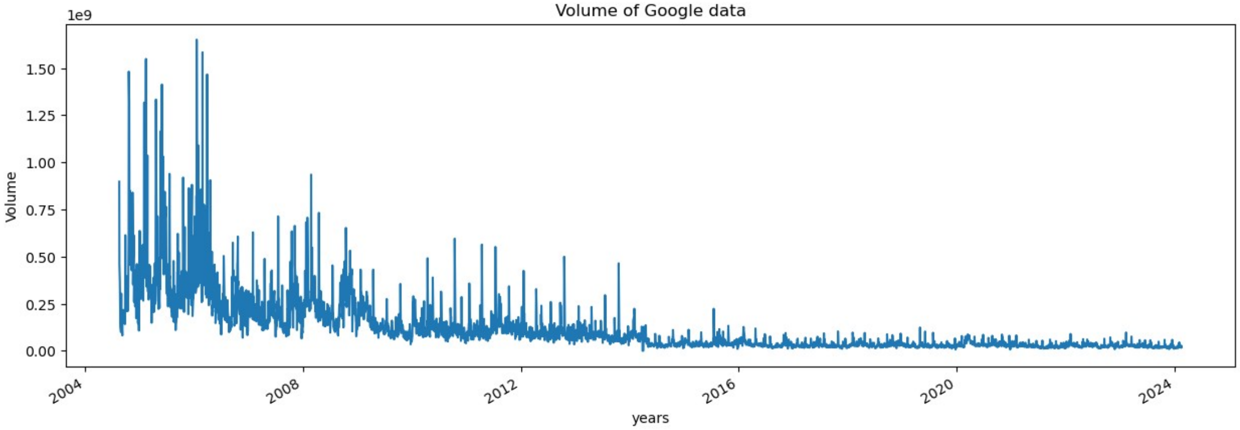


Low of Google data



Close of Google data





```
# 10, 20, 30, 40, 50, 60, 70, 80, 90, 100

# MA for 5 days ==> null null null null 30 40 50 60 70 80

temp_data = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
print(sum(temp_data[1:6])/5)
```

```
40.0
```

```
import pandas as pd
data = pd.DataFrame([10, 20, 30, 40, 50, 60, 70, 80, 90, 100])
data.head()
```

```
    0
0  10
1  20
2  30
3  40
4  50
```

```
data['MA'] = data.rolling(5).mean()
data
```

```
     0    MA
0   10   NaN
1   20   NaN
2   30   NaN
3   40   NaN
4   50  30.0
5   60  40.0
6   70  50.0
7   80  60.0
8   90  70.0
9  100  80.0
```

```
for i in range(2004,2025):
    print(i,list(google_data.index.year).count(i))
```

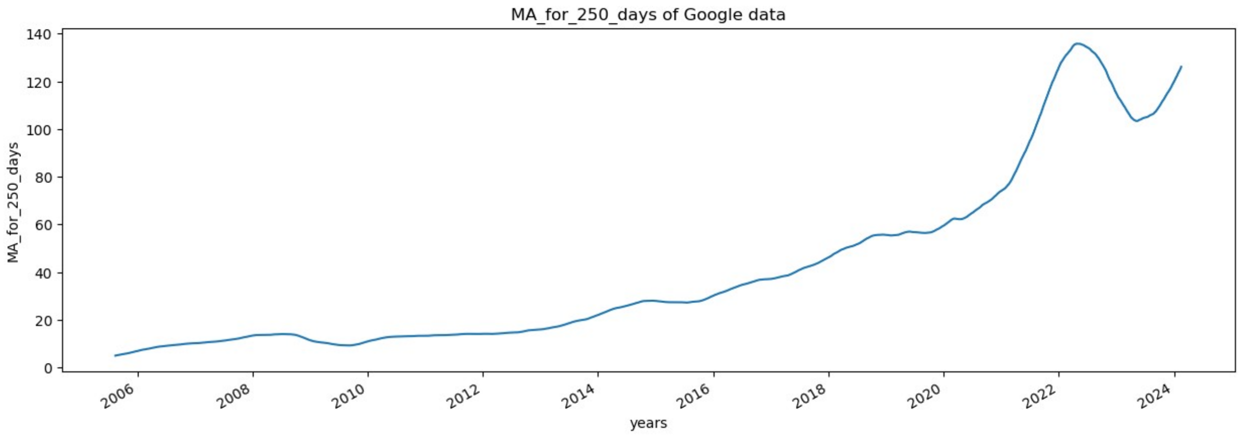
```
2004 94
2005 252
2006 251
2007 251
2008 253
2009 252
2010 252
2011 252
2012 250
2013 252
2014 252
2015 252
2016 252
2017 251
2018 251
2019 252
2020 253
2021 252
2022 251
2023 250
2024 33
```

```
google_data['MA_for_250_days'] = google_data['Adj
Close'].rolling(250).mean()
```

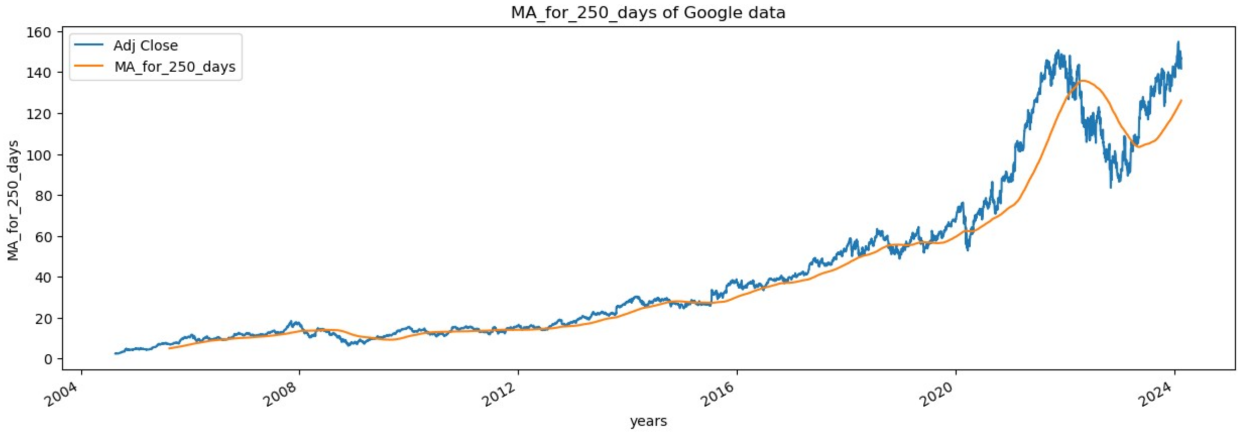
```
google_data['MA_for_250_days'][0:250].tail()
```

```
Date
2005-08-09         NaN
2005-08-10         NaN
2005-08-11         NaN
2005-08-12         NaN
2005-08-15    5.034039
Name: MA_for_250_days, dtype: float64
```

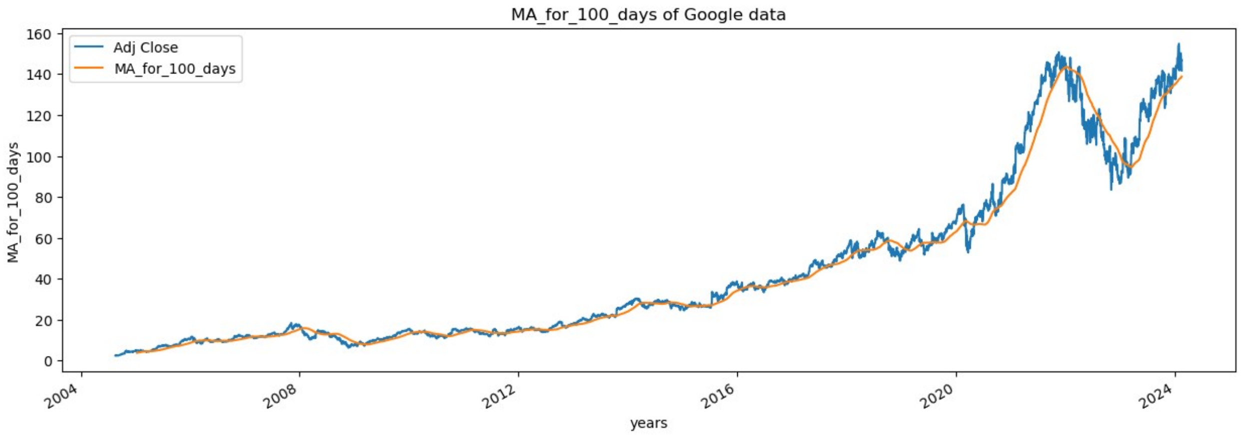
```
plot_graph((15,5), google_data['MA_for_250_days'], 'MA_for_250_days')
```



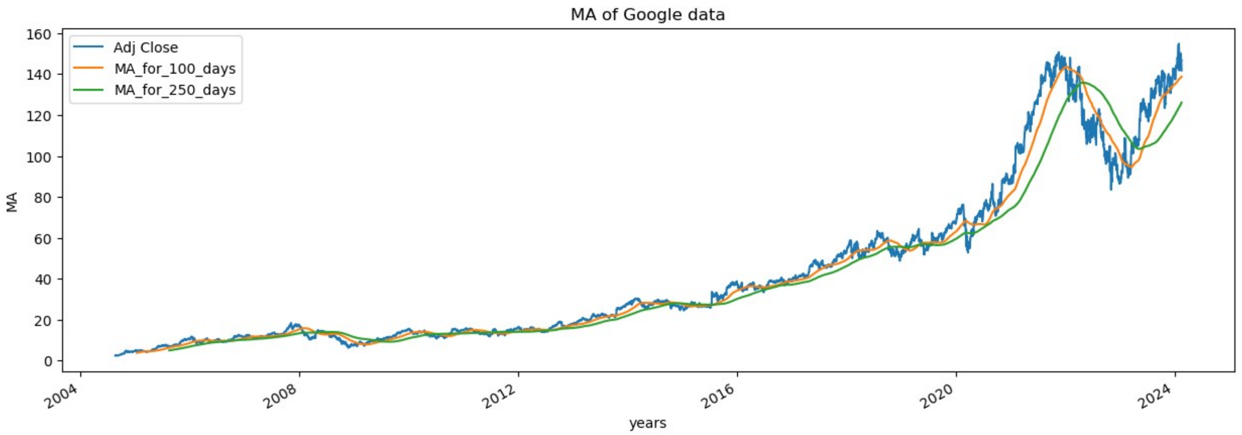
```
plot_graph((15,5), google_data[['Adj Close','MA_for_250_days']],
'MA_for_250_days')

<Figure size 640x480 with 0 Axes>
```



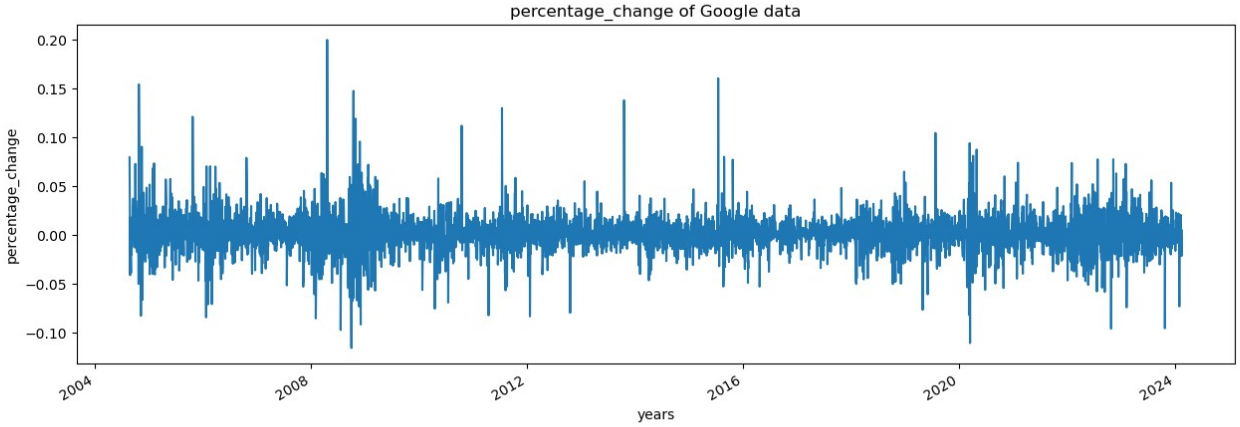
```
google_data['MA_for_100_days'] = google_data['Adj
Close'].rolling(100).mean()
plot_graph((15,5), google_data[['Adj Close','MA_for_100_days']],
'MA_for_100_days')

<Figure size 640x480 with 0 Axes>
```



```
plot_graph((15,5), google_data[['Adj Close','MA_for_100_days',
'MA_for_250_days']], 'MA')

<Figure size 640x480 with 0 Axes>
```



```
google_data['percentage_change_cp'] = google_data['Adj
Close'].pct_change()
google_data[['Adj Close','percentage_change_cp']].head()

            Adj Close  percentage_change_cp
Date
2004-08-19   2.499133                   NaN
2004-08-20   2.697639              0.079430
2004-08-23   2.724787              0.010064
2004-08-24   2.611960             -0.041408
2004-08-25   2.640104              0.010775

plot_graph((15,5), google_data['percentage_change_cp'],
'percentage_change')
```



```
Adj_close_price = google_data[['Adj Close']]

max(Adj_close_price.values),min(Adj_close_price.values)

(array([154.83999634]), array([2.49091291]))

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(Adj_close_price)
scaled_data

array([[5.39563192e-05],
       [1.35692365e-03],
       [1.53511972e-03],
       ...,
       [9.49458200e-01],
       [9.28453827e-01],
       [9.14144532e-01]])

len(scaled_data)

4908

x_data = []
y_data = []

for i in range(100, len(scaled_data)):
    x_data.append(scaled_data[i-100:i])
    y_data.append(scaled_data[i])

import numpy as np
x_data, y_data = np.array(x_data), np.array(y_data)

x_data[0],y_data[0]

(array([[5.39563192e-05],
        [1.35692365e-03],
        [1.53511972e-03],
        [7.94537238e-04],
        [9.79271669e-04],
        [1.29152755e-03],
        [1.00379442e-03],
        [3.26972665e-04],
        [3.85825710e-04],
        [3.92395367e-05],
        [2.45227543e-04],
        [0.00000000e+00],
        [2.56673581e-04],
        [3.74384367e-04],
        [3.76018174e-04],
        [8.69739308e-04],
        [1.22449765e-03],
        [1.87680291e-03],
        [1.96017714e-03],
        [2.28223900e-03],
        [2.85770682e-03],
        [3.16341965e-03],
        [2.91492449e-03],
        [3.00320171e-03],
        [3.40210883e-03],
        [3.24025709e-03],
        [2.98358977e-03],
        [4.38955110e-03],
        [5.07945427e-03],
        [4.83749591e-03],
        [5.32468025e-03],
        [5.73011634e-03],
        [6.27124917e-03],
        [6.06035818e-03],
        [6.34972042e-03],
        [6.16662136e-03],
        [5.76281752e-03],
        [6.11267287e-03],
        [6.68486525e-03],
        [6.86469513e-03],
        [7.20964750e-03],
        [8.03524428e-03],
        [7.83579464e-03],
        [6.61783535e-03],
        [8.07120838e-03],
        [1.18395200e-02],
        [1.42868721e-02],
        [1.33713643e-02],
        [1.40530938e-02],
        [1.52514286e-02],
        [1.48165636e-02],
        [1.56977396e-02],
        [1.55081022e-02],
        [1.49849490e-02],
        [1.38454657e-02],
        [1.13359851e-02],
        [1.18591320e-02],
        [1.12297250e-02],
        [1.10923945e-02],
        [1.35708140e-02],
        [1.34040592e-02],
        [1.38732592e-02],
        [1.18574982e-02],
        [1.18509598e-02],
        [1.10400814e-02],
        [1.13441573e-02],
        [1.06411789e-02],
        [1.10368106e-02],
        [1.22204349e-02],
        [1.29773664e-02],
        [1.32487505e-02],
        [1.34007916e-02],
        [1.30705560e-02],
        [1.29790034e-02],
        [1.31424874e-02],
        [1.24705639e-02],
        [1.16760329e-02],
        [1.14389807e-02],
        [1.20030009e-02],
        [1.17119986e-02],
        [1.15158165e-02],
        [1.28629280e-02],
        [1.30411225e-02],
        [1.24999912e-02],
        [1.30901680e-02],
        [1.38977788e-02],
        [1.36901570e-02],
        [1.41070438e-02],
        [1.43686126e-02],
        [1.50241854e-02],
        [1.51631498e-02],
        [1.51860387e-02],
        [1.59544132e-02],
        [1.51680543e-02],
        [1.67898175e-02],
        [1.54476106e-02],
        [1.52857636e-02],
        [1.44748820e-02],
        [1.53413474e-02],
        [1.55391633e-02]]),
 array([0.01529067]))
```

```
int(len(x_data)*0.7)
```

```
3365
```

```
4908-100-int(len(x_data)*0.7)
```

```
1443
```

```
splitting_len = int(len(x_data)*0.7)
x_train = x_data[:splitting_len]
y_train = y_data[:splitting_len]

x_test = x_data[splitting_len:]
y_test = y_data[splitting_len:]
```

```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(3365, 100, 1)
(3365, 1)
(1443, 100, 1)
(1443, 1)
```

```
from keras.models import Sequential
from keras.layers import Dense, LSTM
```

```
model = Sequential()
model.add(LSTM(128, return_sequences=True,
input_shape=(x_train.shape[1],1)))
model.add(LSTM(64,return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))
```

```
model.compile(optimizer='adam', loss='mean_squared_error')
```

```
WARNING:tensorflow:From C:\Users\prudh\AppData\Roaming\Python\
Python311\site-packages\keras\src\optimizers\__init__.py:309: The name
tf.train.Optimizer is deprecated. Please use
tf.compat.v1.train.Optimizer instead.
```

```
model.fit(x_train, y_train, batch_size=1, epochs = 2)
```

```
Epoch 1/2
WARNING:tensorflow:From C:\Users\prudh\AppData\Roaming\Python\
Python311\site-packages\keras\src\utils\tf_utils.py:492: The name
```

```
tf.ragged.RaggedTensorValue is deprecated. Please use
tf.compat.v1.ragged.RaggedTensorValue instead.

3365/3365 [==============================] - 341s 96ms/step - loss:
1.4831e-04
Epoch 2/2
3365/3365 [==============================] - 302s 90ms/step - loss:
6.2634e-05

<keras.src.callbacks.History at 0x272e9db6c10>
```

```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 lstm_2 (LSTM)               (None, 100, 128)          66560     
                                                                 
 lstm_3 (LSTM)               (None, 64)                49408     
                                                                 
 dense_1 (Dense)             (None, 25)                1625      
                                                                 
 dense_2 (Dense)             (None, 1)                 26        
                                                                 
=================================================================
Total params: 117619 (459.45 KB)
Trainable params: 117619 (459.45 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
predictions = model.predict(x_test)
```

```
46/46 [==============================] - 7s 79ms/step
```

```
predictions
```

```
array([[0.346481  ],
       [0.3471811 ],
       [0.34669545],
       ...,
       [0.9589185 ],
       [0.9565912 ],
       [0.94491225]], dtype=float32)
```

```
inv_predictions = scaler.inverse_transform(predictions)
inv_predictions
```

```
array([[ 55.276974],
       [ 55.383636],
       [ 55.309647],
       ...,
       [148.58127 ],
       [148.2267  ],
       [146.44743 ]], dtype=float32)
```

```
inv_y_test = scaler.inverse_transform(y_test)
inv_y_test
```

```
array([[ 53.9620018 ],
       [ 53.78300095],
       [ 53.01599884],
       ...,
       [147.13999939],
       [143.94000244],
       [141.75999451]])
```

```
rmse = np.sqrt(np.mean( (inv_predictions - inv_y_test)**2))
```

```
rmse
```

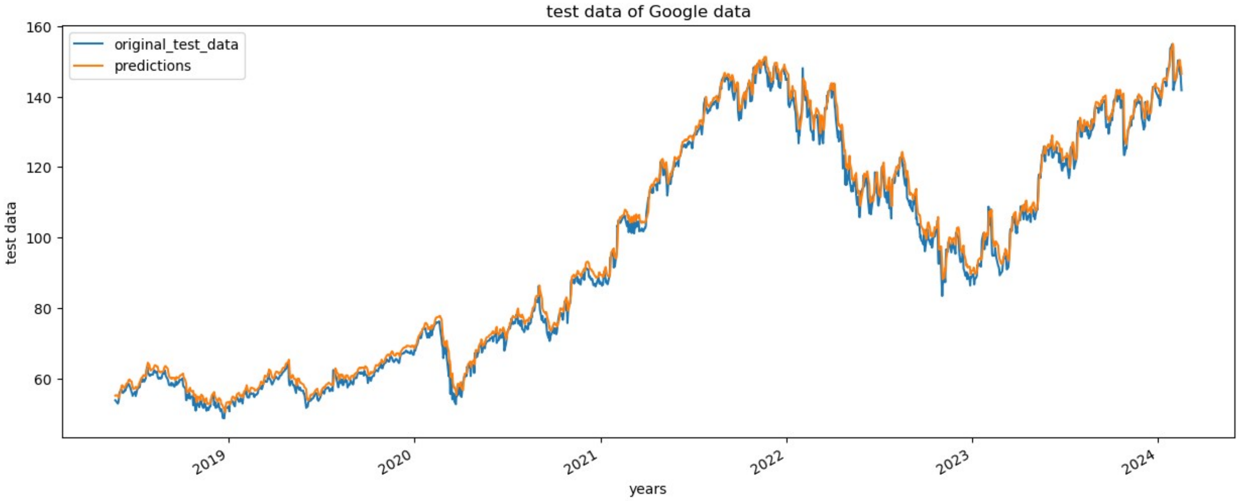
```
2.6333577251768654
```

```
ploting_data = pd.DataFrame(
 {
  'original_test_data': inv_y_test.reshape(-1),
    'predictions': inv_predictions.reshape(-1)
 } ,
    index = google_data.index[splitting_len+100:]
)
ploting_data.head()
```

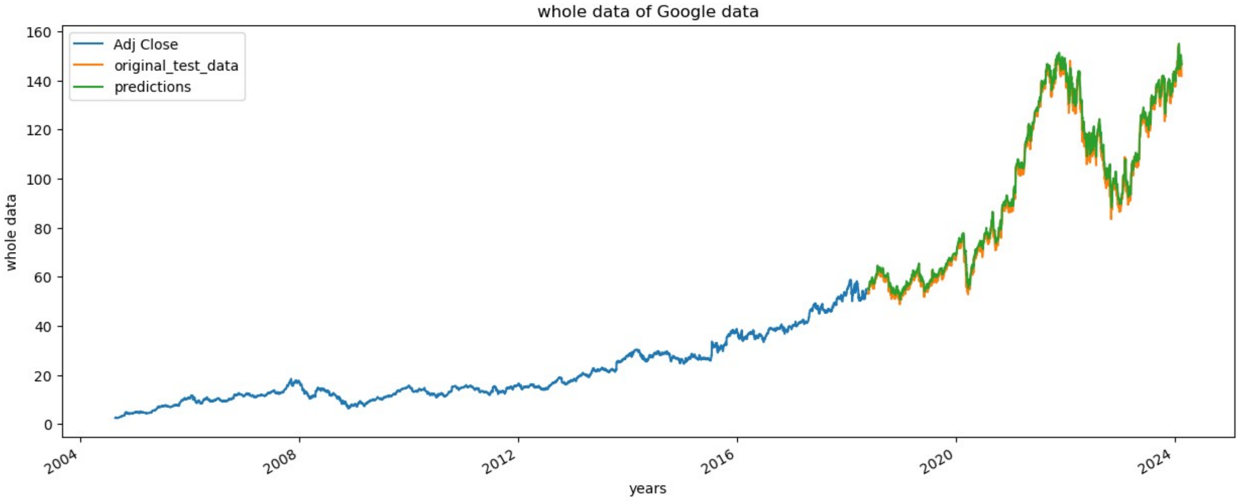
```
            original_test_data  predictions
Date
2018-05-24           53.962002    55.276974
2018-05-25           53.783001    55.383636
2018-05-29           53.015999    55.309647
2018-05-30           53.389999    54.820251
2018-05-31           54.249500    54.759434
```

```
plot_graph((15,6), ploting_data, 'test data')
```

```
<Figure size 640x480 with 0 Axes>
```



```
plot_graph((15,6),
pd.concat([Adj_close_price[:splitting_len+100],ploting_data], axis=0),
'whole data')

<Figure size 640x480 with 0 Axes>
```



```
model.save("Latest_stock_price_model.keras")
```