In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
%matplotlib inline
```

In [2]:

```
df=pd.read_csv("AAPL - AAPL.csv")
```

In [3]:

```
df
```

Out[3]:

	Date	Open	High	Low	Close(t)	Volume	SD20	Upper_Band	Lower_Ban
0	2005-10-17	6.66	6.69	6.50	6.60	154208600	0.169237	6.827473	6.15052
1	2005-10-18	6.57	6.66	6.44	6.45	152397000	0.168339	6.819677	6.14632
2	2005-10-19	6.43	6.78	6.32	6.78	252170800	0.180306	6.861112	6.13988
3	2005-10-20	6.72	6.97	6.71	6.93	339440500	0.202674	6.931847	6.12115
4	2005-10-21	7.02	7.03	6.83	6.87	199181500	0.216680	6.974860	6.10814
...	...	...	...	...	...	...	...	...	
3727	2020-08-07	452.82	454.70	441.17	444.45	49453300	27.954399	455.316298	343.49870
3728	2020-08-10	450.40	455.10	440.00	450.91	53100900	29.847338	462.586675	343.19732
3729	2020-08-11	447.88	449.93	436.43	437.50	46975600	30.576290	466.543079	344.23792
3730	2020-08-12	441.99	453.10	441.19	452.04	41486200	32.050532	472.583564	344.38143
3731	2020-08-13	457.72	464.17	455.71	460.04	52520500	33.532634	479.279768	345.14923

3732 rows × 64 columns

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3732 entries, 0 to 3731
Data columns (total 64 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Date              3732 non-null   object 
 1   Open              3732 non-null   float64
 2   High              3732 non-null   float64
 3   Low               3732 non-null   float64
 4   Close(t)          3732 non-null   float64
 5   Volume            3732 non-null   int64  
 6   SD20              3732 non-null   float64
 7   Upper_Band        3732 non-null   float64
 8   Lower_Band        3732 non-null   float64
 9   S_Close(t-1)      3732 non-null   float64
 10  S_Close(t-2)      3732 non-null   float64
 11  S_Close(t-3)      3732 non-null   float64
 12  S_Close(t-5)      3732 non-null   float64
 13  S_Open(t-1)       3732 non-null   float64
 14  MA5               3732 non-null   float64
 15  MA10              3732 non-null   float64
 16  MA20              3732 non-null   float64
 17  MA50              3732 non-null   float64
 18  MA200             3732 non-null   float64
 19  EMA10             3732 non-null   float64
 20  EMA20             3732 non-null   float64
 21  EMA50             3732 non-null   float64
 22  EMA100            3732 non-null   float64
 23  EMA200            3732 non-null   float64
 24  MACD              3732 non-null   float64
 25  MACD_EMA          3732 non-null   float64
 26  ATR               3732 non-null   float64
 27  ADX               3732 non-null   float64
 28  CCI               3732 non-null   float64
 29  ROC               3732 non-null   float64
 30  RSI               3732 non-null   float64
 31  William%R         3732 non-null   float64
 32  SO%K              3732 non-null   float64
 33  STD5              3732 non-null   float64
 34  ForceIndex1       3732 non-null   int64  
 35  ForceIndex20      3732 non-null   int64  
 36  Date_col          3732 non-null   object 
 37  Day               3732 non-null   int64  
 38  DayofWeek         3732 non-null   int64  
 39  DayofYear         3732 non-null   int64  
 40  Week              3732 non-null   int64  
 41  Is_month_end      3732 non-null   int64  
 42  Is_month_start    3732 non-null   int64  
 43  Is_quarter_end    3732 non-null   int64  
 44  Is_quarter_start  3732 non-null   int64  
 45  Is_year_end       3732 non-null   int64  
 46  Is_year_start     3732 non-null   int64  
 47  Is_leap_year      3732 non-null   int64  
 48  Year              3732 non-null   int64  
 49  Month             3732 non-null   int64  
 50  QQQ_Close         3732 non-null   float64
 51  QQQ(t-1)          3732 non-null   float64
 52  QQQ(t-2)          3732 non-null   float64
 53  QQQ(t-5)          3732 non-null   float64
 54  QQQ_MA10          3732 non-null   float64
 55  QQQ_MA20          3732 non-null   float64
 56   QQQ_MA50          3732 non-null   float64
 57   SnP_Close         3732 non-null   float64
 58   SnP(t-1))         3732 non-null   float64
 59   SnP(t-5)          3732 non-null   float64
 60   DJIA_Close        3732 non-null   float64
 61   DJIA(t-1))        3732 non-null   float64
 62   DJIA(t-5)         3732 non-null   float64
 63   Close_forcast     3732 non-null   float64
dtypes: float64(46), int64(16), object(2)
memory usage: 1.8+ MB
```

In [5]:

```
df = df.rename(columns={'Close(t)':'Close'})
df.head()
```

Out[5]:

	Date	Open	High	Low	Close	Volume	SD20	Upper_Band	Lower_Band	S_Close
0	2005-10-17	6.66	6.69	6.50	6.60	154208600	0.169237	6.827473	6.150527	6.
1	2005-10-18	6.57	6.66	6.44	6.45	152397000	0.168339	6.819677	6.146323	6.
2	2005-10-19	6.43	6.78	6.32	6.78	252170800	0.180306	6.861112	6.139888	6.
3	2005-10-20	6.72	6.97	6.71	6.93	339440500	0.202674	6.931847	6.121153	6.
4	2005-10-21	7.02	7.03	6.83	6.87	199181500	0.216680	6.974860	6.108140	6.

5 rows × 64 columns

In [6]:

```
df.shape
```

Out[6]:

```
(3732, 64)
```

In [7]:

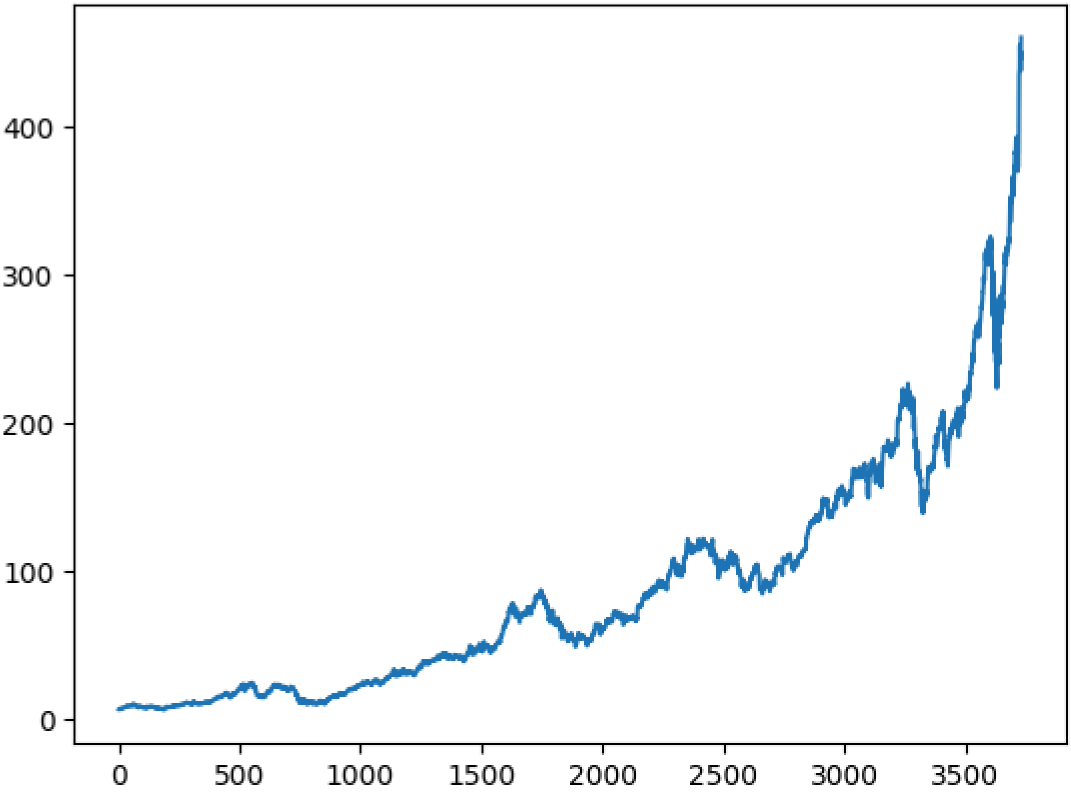
```
df.columns
```

Out[7]:

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Volume', 'SD20', 'Upper_Ba
nd',
       'Lower_Band', 'S_Close(t-1)', 'S_Close(t-2)', 'S_Close(t-3)',
       'S_Close(t-5)', 'S_Open(t-1)', 'MA5', 'MA10', 'MA20', 'MA50', 'MA20
0',
       'EMA10', 'EMA20', 'EMA50', 'EMA100', 'EMA200', 'MACD', 'MACD_EMA',
       'ATR', 'ADX', 'CCI', 'ROC', 'RSI', 'William%R', 'SO%K', 'STD5',
       'ForceIndex1', 'ForceIndex20', 'Date_col', 'Day', 'DayofWeek',
       'DayofYear', 'Week', 'Is_month_end', 'Is_month_start', 'Is_quarter_
end',
       'Is_quarter_start', 'Is_year_end', 'Is_year_start', 'Is_leap_year',
       'Year', 'Month', 'QQQ_Close', 'QQQ(t-1)', 'QQQ(t-2)', 'QQQ(t-5)',
       'QQQ_MA10', 'QQQ_MA20', 'QQQ_MA50', 'SnP_Close', 'SnP(t-1))',
       'SnP(t-5)', 'DJIA_Close', 'DJIA(t-1))', 'DJIA(t-5)', 'Close_forcas
t'],
      dtype='object')
```

In [9]:
```
df['Close'].plot()
```

Out[9]:
```
<Axes: >
```



In [27]:
```
df=df.drop(columns='Date')
```

In [28]:
```
df.isna()
```

Out[28]:

	Open	High	Low	Close	Volume	SD20	Upper_Band	Lower_Band	S_Close(t-1)	S_Cl
0	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	
3727	False	False	False	False	False	False	False	False	False	
3728	False	False	False	False	False	False	False	False	False	
3729	False	False	False	False	False	False	False	False	False	
3730	False	False	False	False	False	False	False	False	False	
3731	False	False	False	False	False	False	False	False	False	

3732 rows × 62 columns

In [29]:
```
df.isna().sum()
```

Out[29]:
```
Open              0
High              0
Low               0
Close             0
Volume            0
                 ..
SnP(t-5)          0
DJIA_Close        0
DJIA(t-1))        0
DJIA(t-5)         0
Close_forcast     0
Length: 62, dtype: int64
```

In [30]:
```
x=df.drop(columns='Close_forcast')
```

In [31]:
```
y=df['Close_forcast']
```

In [32]:
```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2)
```

In [33]:
```
lr=LinearRegression()
lr.fit(xtrain,ytrain)
```

Out[33]:
```
▾ LinearRegression

LinearRegression()
```

In [34]:
```
lr.coef_
```

Out[34]:
```
array([-8.02951549e-02,  1.41958892e-01,  2.58713797e-01, -1.38107297e+00,
        1.24160350e-09, -1.27127620e-02, -1.49770889e-01, -9.88768867e-02,
       -1.40576003e+00, -7.08865561e-01, -4.47781386e-01, -4.43556261e-01,
       -3.33762401e-02, -1.15586007e+00, -1.11997874e+00, -1.24321467e-01,
        1.51235905e-01,  2.82769344e-02,  1.51549189e+00,  1.51549189e+00,
        1.51549189e+00,  1.51549189e+00,  1.51549189e+00, -4.46336162e-01,
        8.87577147e-01,  1.87147289e-01,  1.66144853e-03, -1.02756692e-12,
       -1.89413945e-02,  2.11753316e-03,  4.42909211e-03,  4.42844138e-03,
       -1.14148485e+01, -1.10941389e-09, -2.54720689e-11, -1.41276944e-02,
       -1.81023783e-02,  1.01553173e-02, -2.63428660e-03,  4.26103903e-01,
       -2.12586360e-01, -6.97573709e-01, -8.04242246e-02,  2.21361179e+00,
       -1.08135723e-13, -1.04435500e-01, -4.68719705e-02, -3.02189742e-01,
       -1.70048633e-01,  1.01977536e-01, -1.81289986e-03, -2.96940462e-02,
       -2.74764154e-02,  1.03770778e-01,  3.82482039e-02,  4.61062191e-03,
        2.54734444e-04, -2.82898298e-03, -1.07284925e-03,  6.24776181e-04,
        5.76527133e-05])
```

In [35]:
```
lr.intercept_
```

Out[35]:
```
95.57516536347127
```

In [36]:
```
lr.score(xtrain,ytrain)
```

Out[36]:
```
0.9991247011916671
```

In [39]:
```
ypred=lr.predict(xtest)
```

In [40]:
```
metrics.mean_absolute_error(ytest,ypred)
```

Out[40]:
```
1.2437301966724754
```

In [41]:
```
metrics.r2_score(ytest,ypred)
```

Out[41]:
```
0.9989243375699739
```

In [42]:
```
metrics.mean_squared_error(ytest,ypred)
```

Out[42]:
```
6.812385609083958
```

In [46]:
```
df1=pd.DataFrame(ytest.values,columns=['actual'])
```

In [47]:
```
df1
```

Out[47]:
```
     actual
  0   55.55
  1  155.52
  2   11.23
  3   11.15
  4  163.13
...     ...
742   43.02
743   15.30
744   17.53
745   72.48
746   48.00

747 rows × 1 columns
```

In [48]:
```
df1['predicted']=ypred
```

In [49]:
```
df1
```

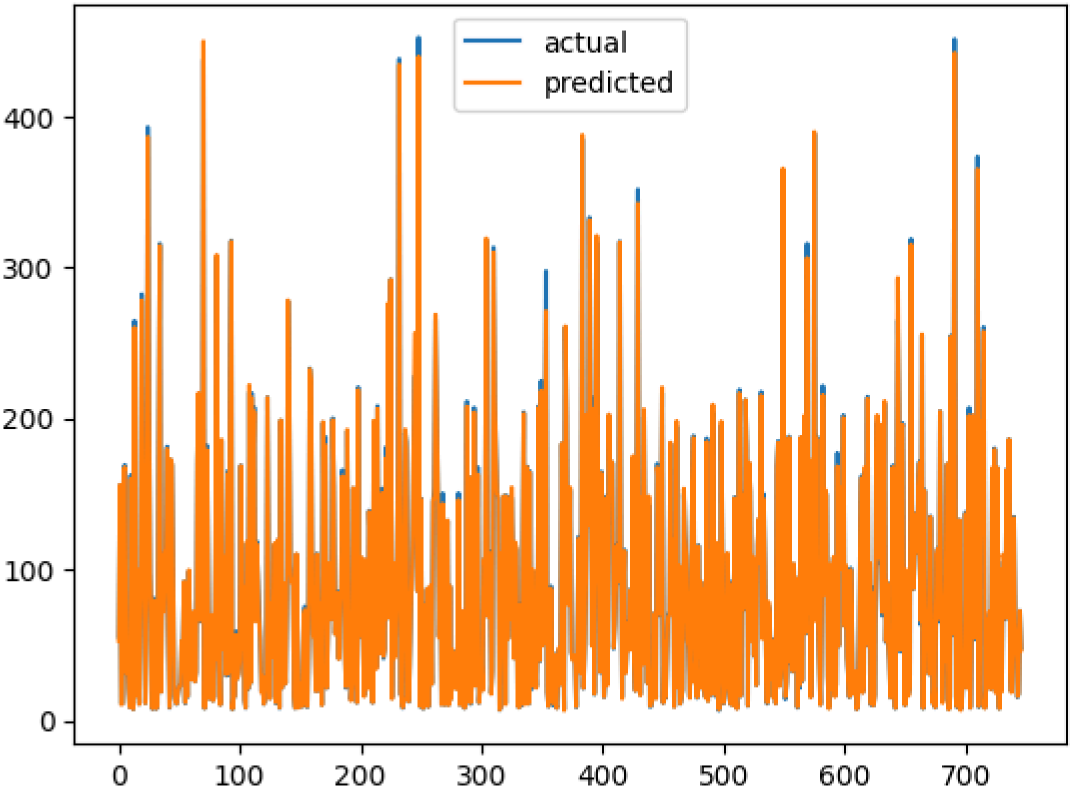
Out[49]:

	actual	predicted
0	55.55	52.730956
1	155.52	155.915093
2	11.23	10.598524
3	11.15	10.976693
4	163.13	164.460614
...	...	...
742	43.02	42.928119
743	15.30	15.977212
744	17.53	17.738846
745	72.48	72.423698
746	48.00	46.709091

747 rows × 2 columns

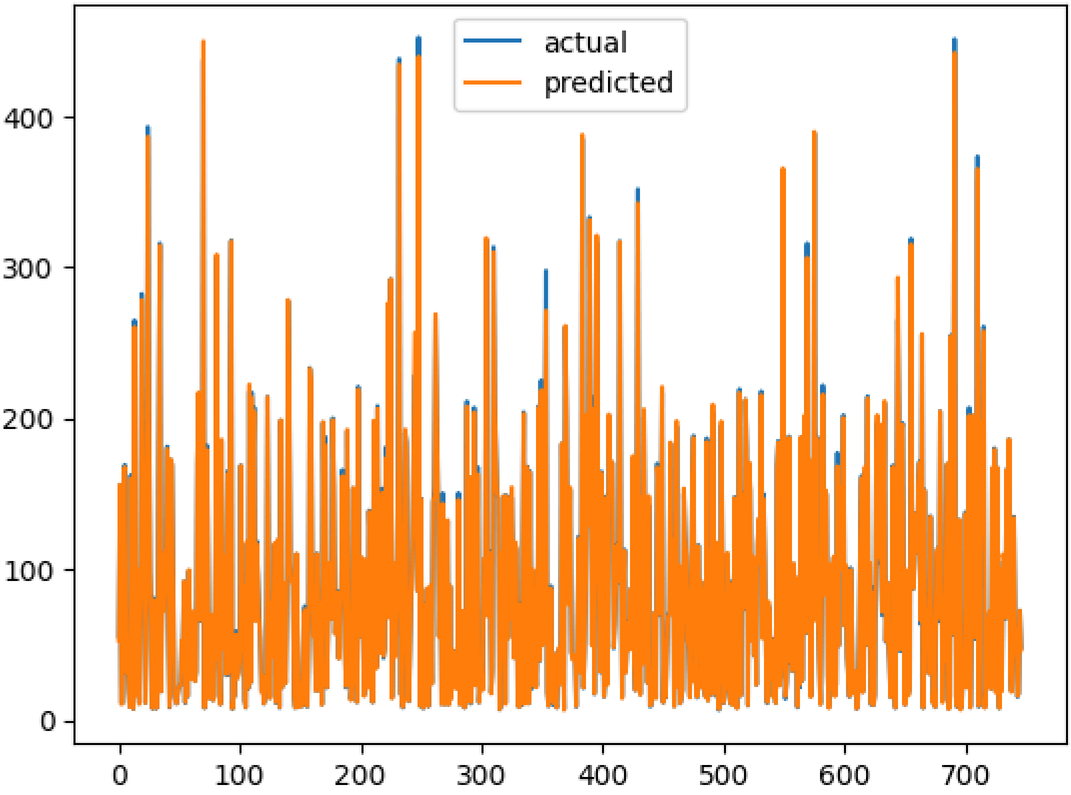
In [51]:
```
df1[['actual','predicted']].plot()
```

Out[51]:
```
<Axes: >
```



```
In [53]: df1[['actual', 'predicted']].plot()

Out[53]: <Axes: >
```



```
In [ ]:
```