Assignment No-9

Name : Shubham Khalate

Class :S.Y-A

Roll No.: 23107062

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("TCS.csv")

df
```

```
            Date         Open         High          Low
Close  \
0     2004-08-27   122.800003   122.800003   119.820000   120.332497

1     2004-08-30   121.237503   123.750000   120.625000   123.345001

2     2004-08-31   123.312500   123.750000   122.000000   123.512497

3     2004-09-01   123.750000   124.375000   122.949997   123.487503

4     2004-09-02   123.737503   125.574997   123.250000   124.207497

...          ...          ...          ...          ...          ...

4489  2022-10-18  3150.000000  3155.350098  3128.550049  3144.699951

4490  2022-10-19  3159.000000  3159.000000  3112.000000  3121.850098

4491  2022-10-20  3105.000000  3160.000000  3105.000000  3157.300049

4492  2022-10-21  3157.800049  3160.399902  3127.000000  3137.399902

4493  2022-10-24  3170.100098  3178.000000  3155.000000  3161.699951


        Adj Close      Volume
0       88.088272  30646000.0
1       90.293549  24465208.0
2       90.416122  21194656.0
3       90.397820  19935544.0
```

```
4       90.924896  21356352.0
...           ...         ...
4489  3144.699951   1793722.0
4490  3121.850098   1194289.0
4491  3157.300049   1587601.0
4492  3137.399902   1021913.0
4493  3161.699951    260949.0

[4494 rows x 7 columns]
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4494 entries, 0 to 4493
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       4494 non-null   object 
 1   Open       4486 non-null   float64
 2   High       4486 non-null   float64
 3   Low        4486 non-null   float64
 4   Close      4486 non-null   float64
 5   Adj Close  4486 non-null   float64
 6   Volume     4486 non-null   float64
dtypes: float64(6), object(1)
memory usage: 245.9+ KB
```

```
df.describe()
```

```
               Open         High          Low        Close    Adj Close
\
count  4486.000000  4486.000000  4486.000000  4486.000000  4486.000000

mean   1146.182768  1158.538059  1132.825428  1145.521462  1049.456965

std     994.070086  1003.010607   984.043404   993.346465   992.062558

min     112.000000   116.112503   103.837502   111.550003    86.565590

25%     290.693748   295.300010   285.931259   290.275009   219.902748

50%     977.450012   995.000000   970.250000   981.337524   820.627839

75%    1564.774963  1597.287476  1548.662537  1576.781219  1443.631561

max    4033.949951  4043.000000  3980.000000  4019.149902  3964.502686


              Volume
count  4.486000e+03
mean   3.620596e+06
```

```
std    3.162368e+06
min    0.000000e+00
25%    1.942489e+06
50%    2.804749e+06
75%    4.297410e+06
max    8.806715e+07
```

```
df.ndim
```

```
2
```

```
df.dtypes
```

```
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close    float64
Volume       float64
dtype: object
```

```
df.size
```

```
31458
```

```
df.shape
```

```
(4494, 7)
```

```
df.head()
```

```
         Date        Open        High         Low       Close  Adj
Close  \
0  2004-08-27  122.800003  122.800003  119.820000  120.332497
88.088272
1  2004-08-30  121.237503  123.750000  120.625000  123.345001
90.293549
2  2004-08-31  123.312500  123.750000  122.000000  123.512497
90.416122
3  2004-09-01  123.750000  124.375000  122.949997  123.487503
90.397820
4  2004-09-02  123.737503  125.574997  123.250000  124.207497
90.924896

       Volume
0  30646000.0
1  24465208.0
2  21194656.0
3  19935544.0
4  21356352.0
```

```
df.tail()
```

```
            Date         Open         High          Low
Close  \
4489  2022-10-18  3150.000000  3155.350098  3128.550049  3144.699951

4490  2022-10-19  3159.000000  3159.000000  3112.000000  3121.850098

4491  2022-10-20  3105.000000  3160.000000  3105.000000  3157.300049

4492  2022-10-21  3157.800049  3160.399902  3127.000000  3137.399902

4493  2022-10-24  3170.100098  3178.000000  3155.000000  3161.699951


        Adj Close     Volume
4489  3144.699951  1793722.0
4490  3121.850098  1194289.0
4491  3157.300049  1587601.0
4492  3137.399902  1021913.0
4493  3161.699951   260949.0
```

```
df.isnull()
```

```
       Date   Open   High    Low  Close  Adj Close  Volume
0     False  False  False  False  False      False   False
1     False  False  False  False  False      False   False
2     False  False  False  False  False      False   False
3     False  False  False  False  False      False   False
4     False  False  False  False  False      False   False
...     ...    ...    ...    ...    ...        ...     ...
4489  False  False  False  False  False      False   False
4490  False  False  False  False  False      False   False
4491  False  False  False  False  False      False   False
4492  False  False  False  False  False      False   False
4493  False  False  False  False  False      False   False

[4494 rows x 7 columns]
```

```
df.isnull().value_counts()
```

```
Date   Open   High   Low    Close  Adj Close  Volume
False  False  False  False  False  False      False     4486
       True   True   True   True   True       True         8
Name: count, dtype: int64
```

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

```
Date         0
Open         8
High         8
```

```
Low          8
Close        8
Adj Close    8
Volume       8
dtype: int64
```

```
df.notnull()
```

```
      Date  Open  High   Low  Close  Adj Close  Volume
0     True  True  True  True   True       True    True
1     True  True  True  True   True       True    True
2     True  True  True  True   True       True    True
3     True  True  True  True   True       True    True
4     True  True  True  True   True       True    True
...    ...   ...   ...   ...    ...        ...     ...
4489  True  True  True  True   True       True    True
4490  True  True  True  True   True       True    True
4491  True  True  True  True   True       True    True
4492  True  True  True  True   True       True    True
4493  True  True  True  True   True       True    True

[4494 rows x 7 columns]
```

```
df.notnull().value_counts()
```

```
Date  Open   High   Low    Close  Adj Close  Volume
True  True   True   True   True   True       True      4486
      False  False  False  False  False      False        8
Name: count, dtype: int64
```

```
df.notnull().sum()
```

```
Date         4494
Open         4486
High         4486
Low          4486
Close        4486
Adj Close    4486
Volume       4486
dtype: int64
```

```
df.dropna(inplace=True)
```

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

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
```

```
Volume       0
dtype: int64
```

```
df["Date"] = pd.to_datetime(df["Date"])
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 4486 entries, 2004-08-27 to 2022-10-24
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Open       4486 non-null   float64
 1   High       4486 non-null   float64
 2   Low        4486 non-null   float64
 3   Close      4486 non-null   float64
 4   Adj Close  4486 non-null   float64
 5   Volume     4486 non-null   float64
dtypes: float64(6)
memory usage: 245.3 KB
```

```
df.set_index('Date', inplace=True)
```

```
df.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2004-08-27	122.800003	122.800003	119.820000	120.332497	88.088272	30646000.0
2004-08-30	121.237503	123.750000	120.625000	123.345001	90.293549	24465208.0
2004-08-31	123.312500	123.750000	122.000000	123.512497	90.416122	21194656.0
2004-09-01	123.750000	124.375000	122.949997	123.487503	90.397820	19935544.0
2004-09-02	123.737503	125.574997	123.250000	124.207497	90.924896	21356352.0

```
plt.figure(figsize=(12, 5))
plt.plot(df.index,df["Close"],label='Close Price')
plt.title('Close Price Over Time')
```

```
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.grid()
plt.show()
```


```
from statsmodels.tsa.stattools import adfuller

def adf_test(series):
    result = adfuller(series)
    labels = ["ADF Statistics:","P-Value","Lags_used","Number Of
Observation"]
    for value,label in zip(result,labels):
        print(label, ' : ' ,str(value))
        if(result[1]<= 0.05):
            print("Strong Evidences against the Null hypothesis,
reject the null hypothesis,and it stationary also has no unit root")
        else:
            print("Weak evidences against the null hypothesis ,Time
series has a root, indicates it is not stationary")

adf_test(df['Close'])
```

```
ADF Statistics:  :  0.8059405233011762
Weak evidences against the null hypothesis ,Time series has a root,
indicates it is not stationary
P-Value  :  0.9917314565589413
Weak evidences against the null hypothesis ,Time series has a root,
indicates it is not stationary
Lags_used  :  32
Weak evidences against the null hypothesis ,Time series has a root,
indicates it is not stationary
Number Of Observation  :  4453
```

```
Weak evidences against the null hypothesis ,Time series has a root,
indicates it is not stationary
```

```
df['Diff of Close'] = df['Close']-df['Close'].shift(1)
```

```
df.head()
```

```
                  Open        High         Low       Close  Adj Close
\
Date

2004-08-27  122.800003  122.800003  119.820000  120.332497  88.088272

2004-08-30  121.237503  123.750000  120.625000  123.345001  90.293549

2004-08-31  123.312500  123.750000  122.000000  123.512497  90.416122

2004-09-01  123.750000  124.375000  122.949997  123.487503  90.397820

2004-09-02  123.737503  125.574997  123.250000  124.207497  90.924896


                Volume  Diff of Close  12 Diff
Date
2004-08-27  30646000.0            NaN      NaN
2004-08-30  24465208.0       3.012504      NaN
2004-08-31  21194656.0       0.167496      NaN
2004-09-01  19935544.0      -0.024994      NaN
2004-09-02  21356352.0       0.719994      NaN
```

```
df['12 Diff'] = df['Close']-df['Close'].shift(12)
```

```
adf_test(df['12 Diff'].dropna())
```

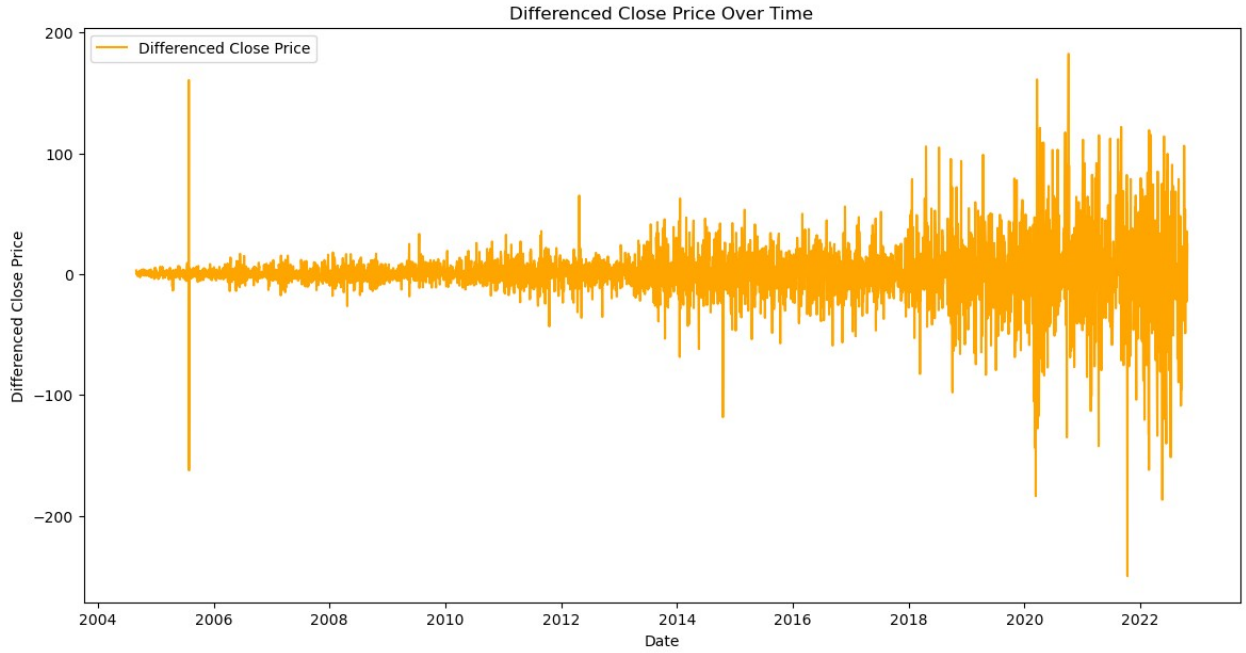
```
ADF Statistics:  :  -12.175333024797796
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
P-Value  :  1.3904621898083073e-22
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
Lags_used  :  31
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
Number Of Observation  :  4442
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
```

```
adf_test(df['Diff of Close'].dropna())
```

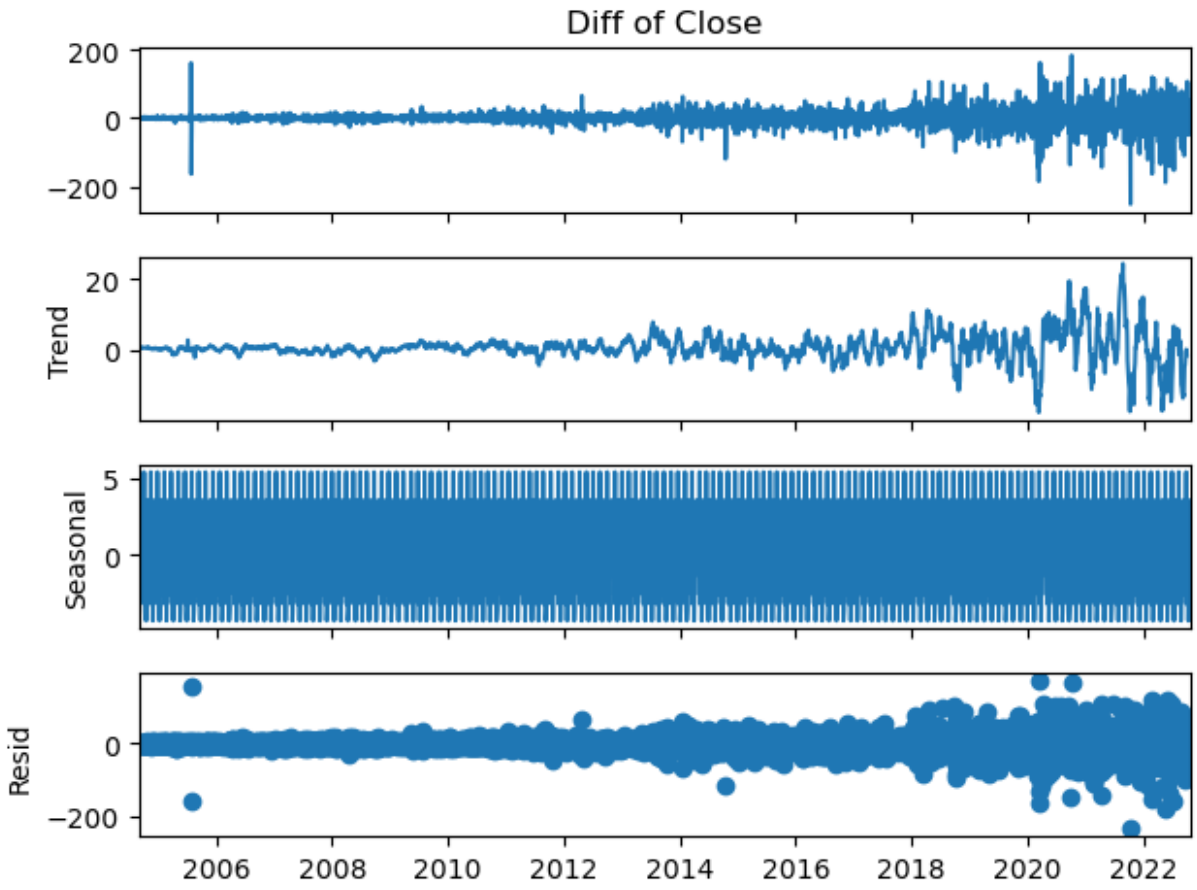
```
ADF Statistics:  :  -14.990218644441317
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
```

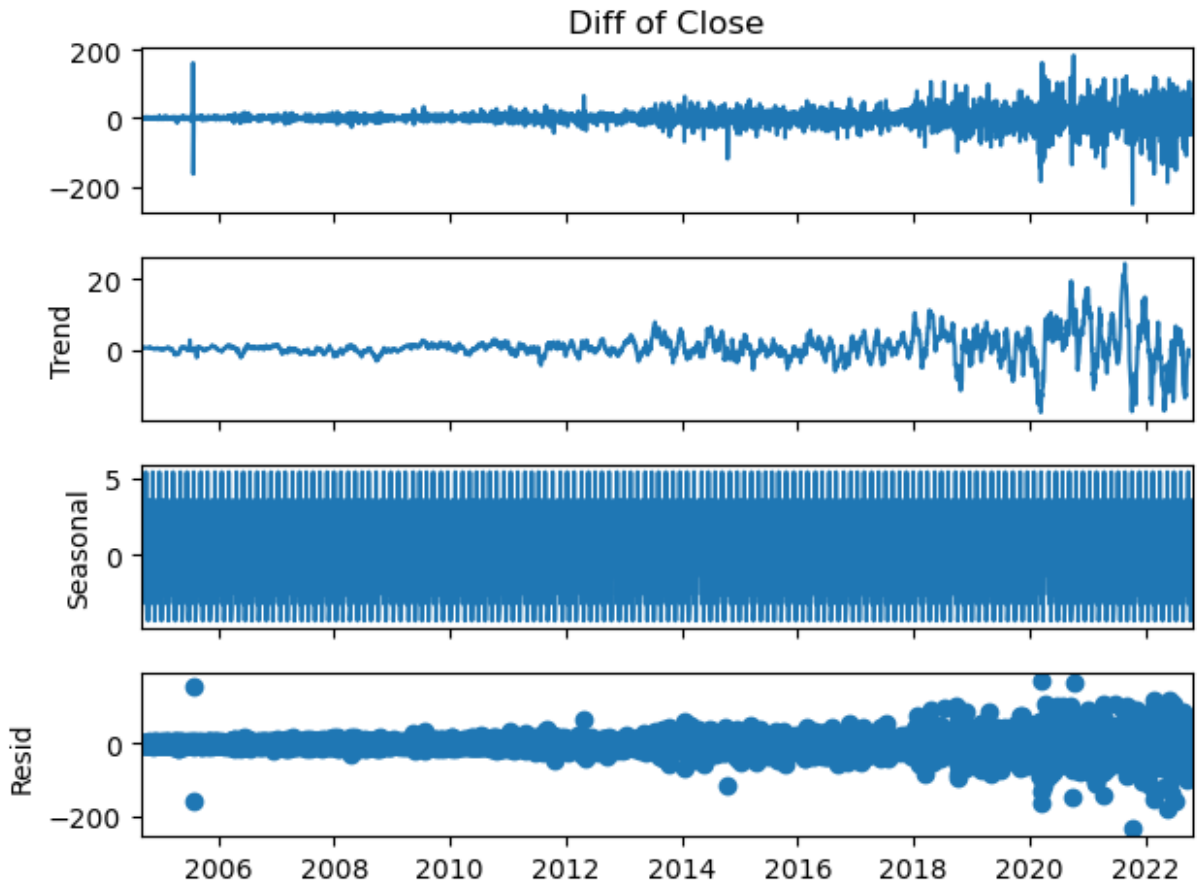
```
P-Value  :  1.1282227743588002e-27
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
Lags_used  :  31
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
Number Of Observation  :  4453
Strong Evidences against the Null hypothesis, reject the null
hypothesis,and it stationary also has no unit root
```

```
plt.figure(figsize=(14, 7))
plt.plot(df.index, df['Diff of Close'], label='Differenced Close
Price',color='orange')
plt.title('Differenced Close Price Over Time')
plt.xlabel('Date')
plt.ylabel('Differenced Close Price')
plt.legend()
plt.show()
```



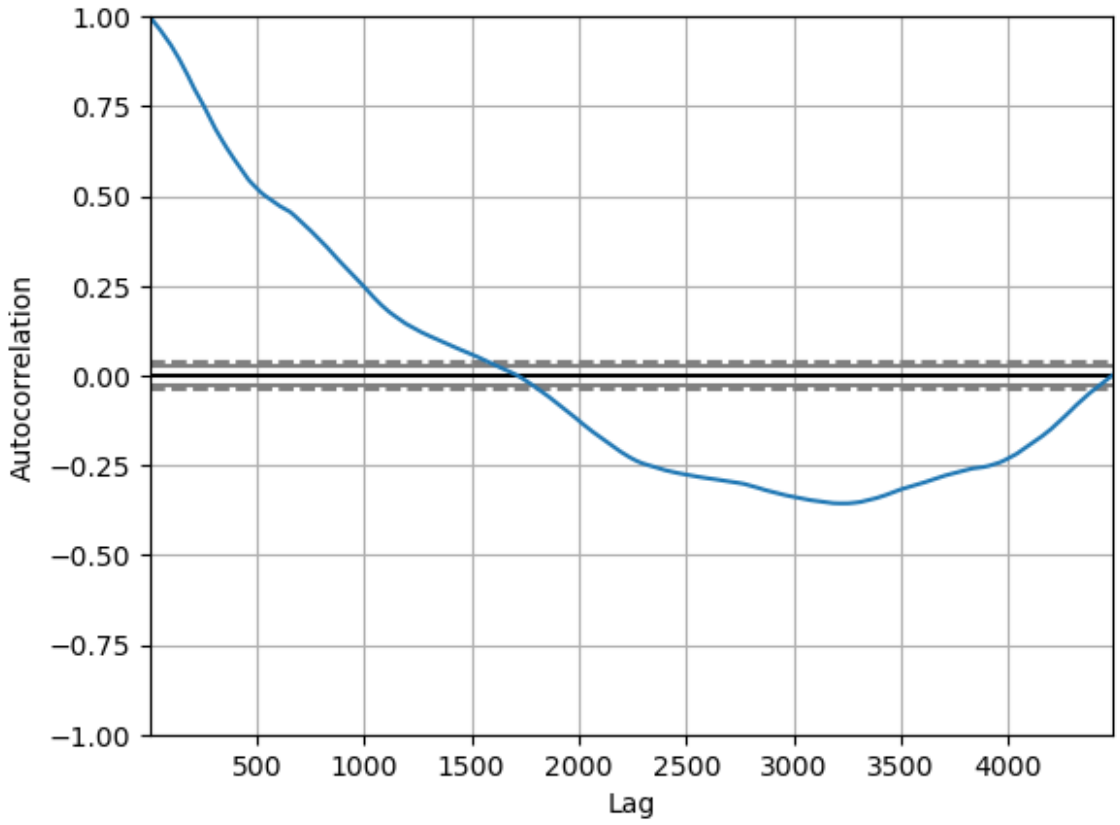

```
from statsmodels.tsa.seasonal import seasonal_decompose
decompose = seasonal_decompose(df['Diff of
Close'].dropna(),model='addictive',period=30)
decompose.plot()
```


Diff of Close
200
0
−200
Trend
20
0
Seasonal
5
0
Resid
0
−200
2006
2008
2010
2012
2014
2016
2018
2020
2022



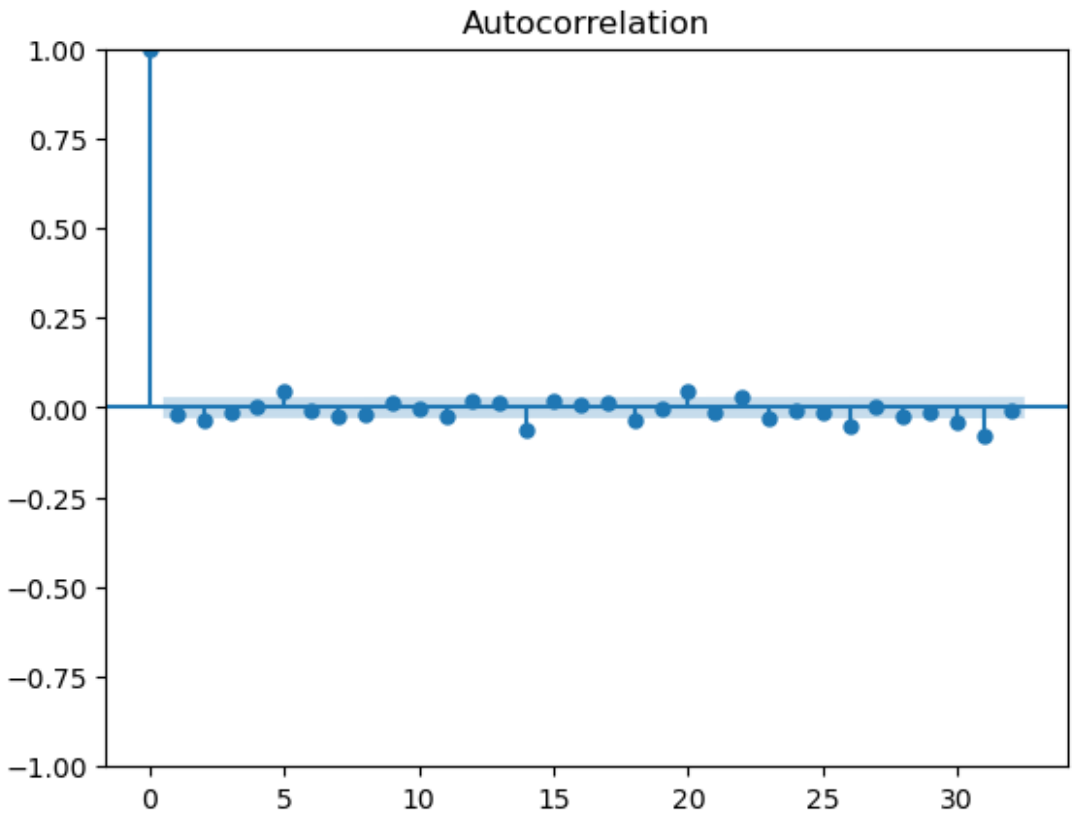

```
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
pd.plotting.autocorrelation_plot(df['Close'])
```

```
<Axes: xlabel='Lag', ylabel='Autocorrelation'>
```




```
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
from statsmodels.tsa.stattools import acf,pacf

acf1 = plot_acf(df['Diff of Close'].dropna(),alpha = 0.05,lags = 32)
"""
Correlation with each month and there is Pearson Correlation
From Acf we find the parameter q
Q < blue area(error)
"""
```

```
'\nCorrelation with each month and there is Pearson Correlation\nFrom
Acf we find the parameter q\nQ < blue area(error)\n'
```

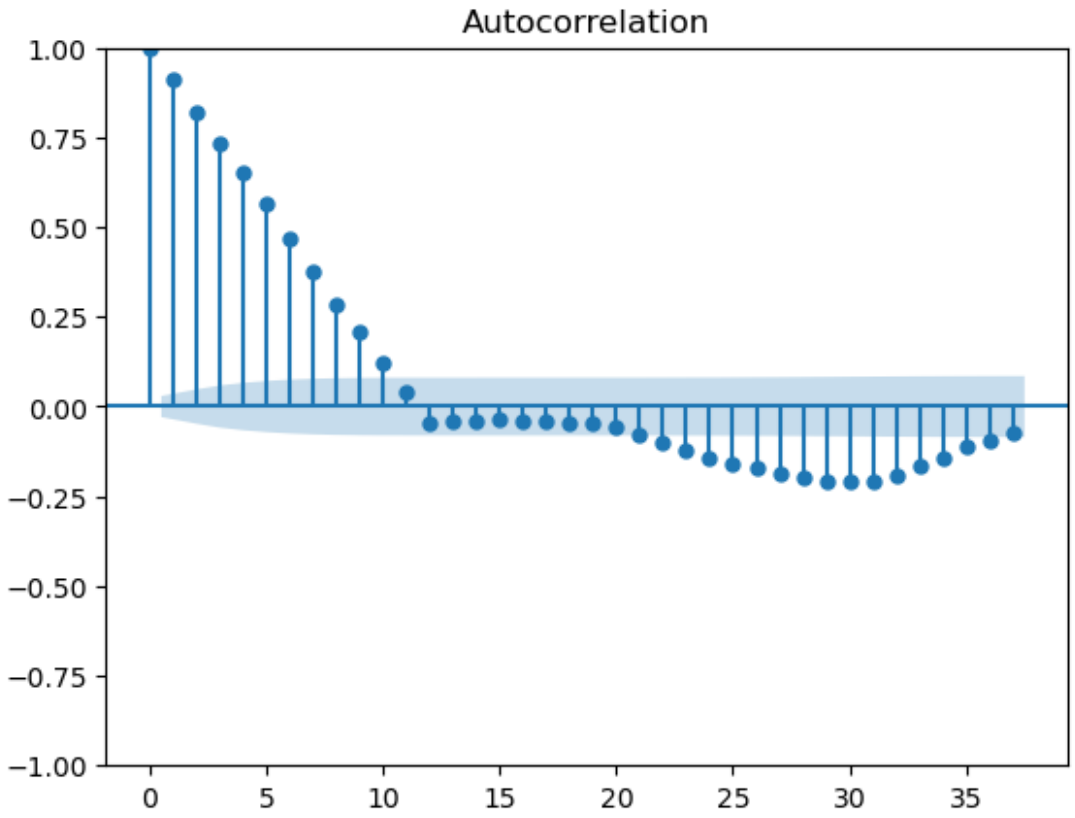



```
xacf = pd.DataFrame(acf(df['Diff of Close'].dropna()))
print(xacf)
```

```
           0
0   1.000000
1  -0.016574
2  -0.036398
3  -0.016025
4   0.004304
5   0.048556
6  -0.010082
7  -0.026128
8  -0.021521
9   0.011150
10 -0.002779
11 -0.024594
12  0.016758
13  0.014371
14 -0.060926
15  0.017095
16  0.009561
17  0.011891
18 -0.034102
19 -0.003790
```

```
20  0.046849
21 -0.013866
22  0.029597
23 -0.028894
24 -0.007285
25 -0.011749
26 -0.051880
27  0.004981
28 -0.024287
29 -0.014058
30 -0.039046
31 -0.078959
32 -0.010349
33  0.012727
34 -0.005207
35  0.027834
36 -0.010172
```

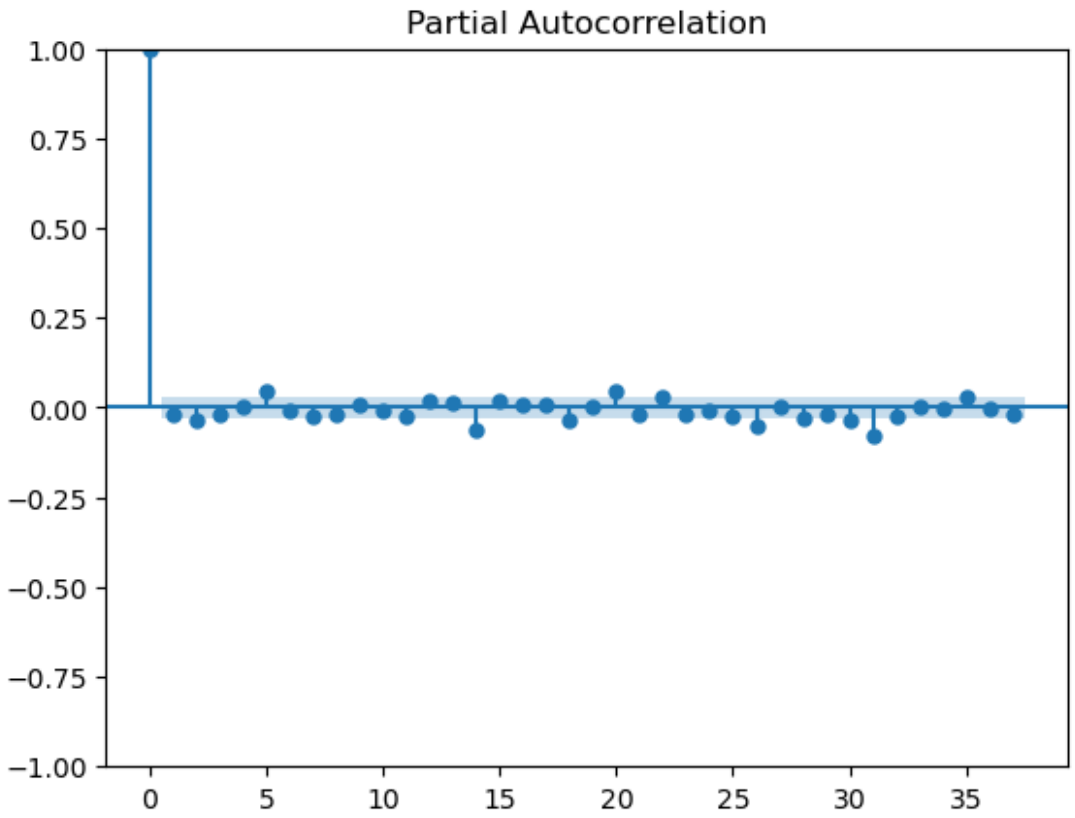
```
acf12 = plot_acf(df['12 Diff'].dropna())
```



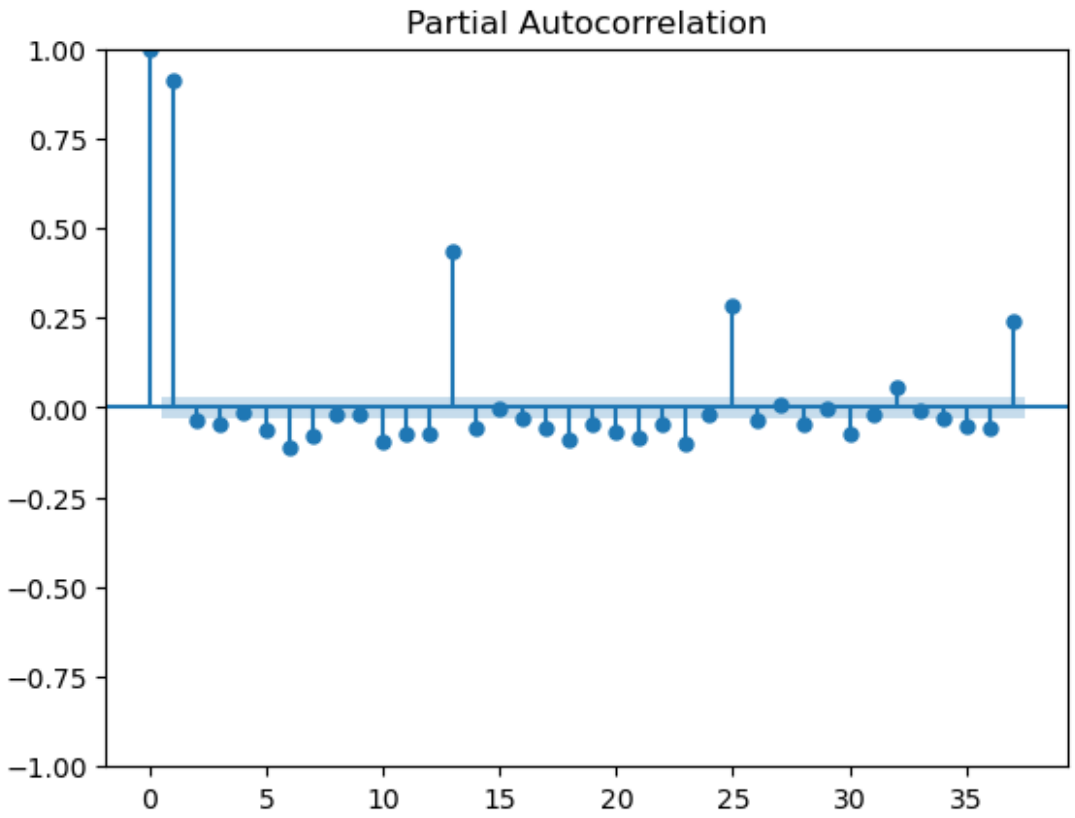

```
xacf = pd.DataFrame(acf(df['12 Diff'].dropna()))
print(xacf)
```

```
           0
0   1.000000
1   0.909188
2   0.820225
3   0.731854
4   0.649476
5   0.564754
6   0.470228
7   0.373339
8   0.285304
9   0.206131
10  0.122287
11  0.037955
12 -0.045308
13 -0.040620
14 -0.040647
15 -0.037193
16 -0.038258
17 -0.042232
18 -0.045696
19 -0.047362
20 -0.057994
21 -0.080545
22 -0.098546
23 -0.122595
24 -0.141596
25 -0.158140
26 -0.172177
27 -0.185589
28 -0.199821
29 -0.205880
30 -0.209829
31 -0.209777
32 -0.192381
33 -0.165422
34 -0.141144
35 -0.112478
36 -0.094657
```

```
pacf = plot_pacf(df['Diff of Close'].dropna())
```




```
pacf12= plot_pacf(df['12 Diff'].dropna())
```




```
df
```

Date	Open	High	Low	Close	Adj Close
2004-08-27	122.800003	122.800003	119.820000	120.332497	88.088272
2004-08-30	121.237503	123.750000	120.625000	123.345001	90.293549
2004-08-31	123.312500	123.750000	122.000000	123.512497	90.416122
2004-09-01	123.750000	124.375000	122.949997	123.487503	90.397820
2004-09-02	123.737503	125.574997	123.250000	124.207497	90.924896
...	...	...	...	...	...
2022-10-18	3150.000000	3155.350098	3128.550049	3144.699951	3144.699951
2022-10-19	3159.000000	3159.000000	3112.000000	3121.850098	3121.850098
2022-10-20	3105.000000	3160.000000	3105.000000	3157.300049	3157.300049

```
2022-10-21  3157.800049  3160.399902  3127.000000  3137.399902
3137.399902
2022-10-24  3170.100098  3178.000000  3155.000000  3161.699951
3161.699951

                Volume  Diff of Close     12 Diff
Date
2004-08-27  30646000.0            NaN         NaN
2004-08-30  24465208.0       3.012504         NaN
2004-08-31  21194656.0       0.167496         NaN
2004-09-01  19935544.0      -0.024994         NaN
2004-09-02  21356352.0       0.719994         NaN
...                ...            ...         ...
2022-10-18   1793722.0      32.949951  147.399902
2022-10-19   1194289.0     -22.849853  117.300049
2022-10-20   1587601.0      35.449951  172.350098
2022-10-21   1021913.0     -19.900147   46.250000
2022-10-24    260949.0      24.300049   59.750000

[4486 rows x 8 columns]
```

```python
from statsmodels.tsa.arima.model import ARIMA
train_size = int(len(df) * 0.8)
train, test = df.iloc[:train_size], df.iloc[train_size:]

model = ARIMA(train["Close"].dropna(), order=(1,1,1))
model_fit = model.fit()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:473: ValueWarning: A date index has been provided, but it
has no associated frequency information and so will be ignored when
e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:473: ValueWarning: A date index has been provided, but it
has no associated frequency information and so will be ignored when
e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:473: ValueWarning: A date index has been provided, but it
has no associated frequency information and so will be ignored when
e.g. forecasting.
  self._init_dates(dates, freq)
```

```python
forecast = model_fit.forecast(steps=len(test))
plt.figure(figsize=(14,7))
plt.plot(train.index, train["Close"], label='Train', color='#203147')
plt.plot(test.index, test["Close"], label='Test', color='#01ef63')
plt.plot(test.index, forecast, label='Forecast', color='orange')
plt.title('Close Price Forecast')
```

```
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:836: ValueWarning: No supported index is available.
Prediction results will be given with an integer index beginning at
`start`.
  return get_prediction_index(
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:836: FutureWarning: No supported index is available. In
the next version, calling this method in a model without a supported
index will result in an exception.
  return get_prediction_index(
```


```
print(f"AIC: {model_fit.aic}")
print(f"BIC: {model_fit.bic}")
```

```
AIC: 29303.565323136136
BIC: 29322.12053756873
```

```
 print("Mean Close Price:", df['Close'].mean())
print("Max Close Price:", df['Close'].max())
print("Min Close Price:", df['Close'].min())
```

```
Mean Close Price: 1145.5214623165402
Max Close Price: 4019.149902
Min Close Price: 111.550003
```

```
 from sklearn.metrics import mean_squared_error
import numpy as np

rmse = np.sqrt(mean_squared_error(test["Close"], forecast))
print("RMSE:", rmse)
```

```
RMSE: 1027.441327038619
```

SARIMA

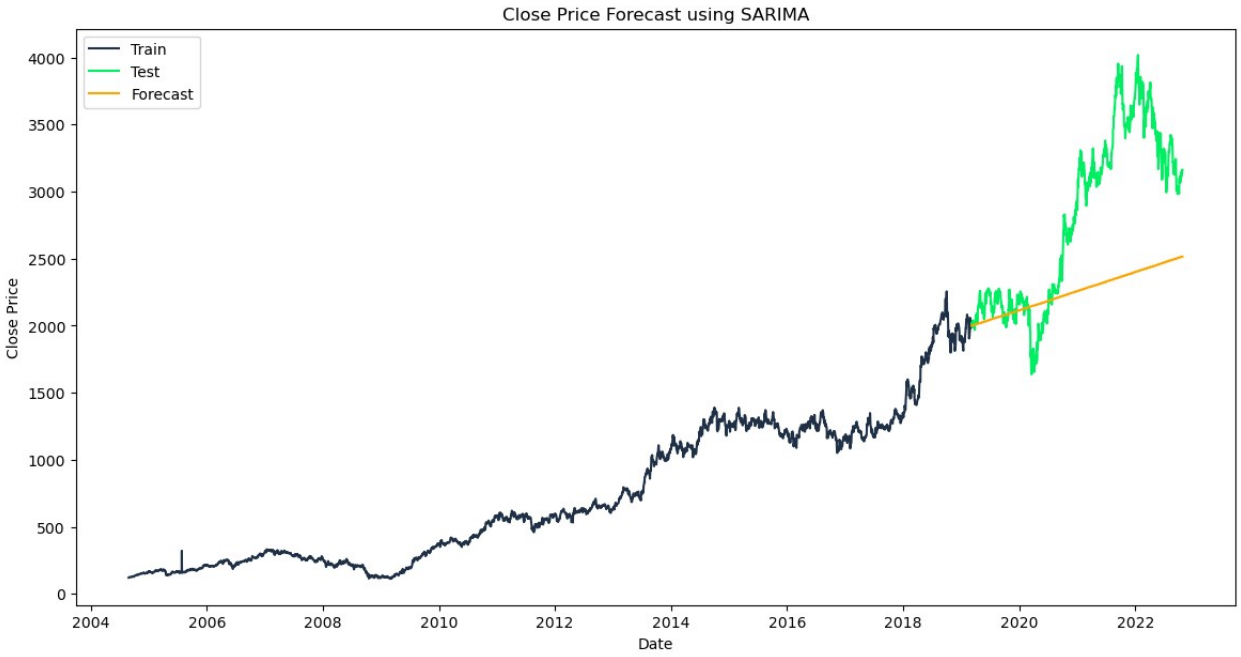
```
from statsmodels.tsa.statespace.sarimax import SARIMAX
import matplotlib.pyplot as plt

model = SARIMAX(train["Close"].dropna(),order=(1,1,1),
seasonal_order=(1,1,1,12))
# 12 = yearly seasonality for monthly data
model_fit = model.fit()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:473: ValueWarning: A date index has been provided, but it
has no associated frequency information and so will be ignored when
e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:473: ValueWarning: A date index has been provided, but it
has no associated frequency information and so will be ignored when
e.g. forecasting.
  self._init_dates(dates, freq)
```

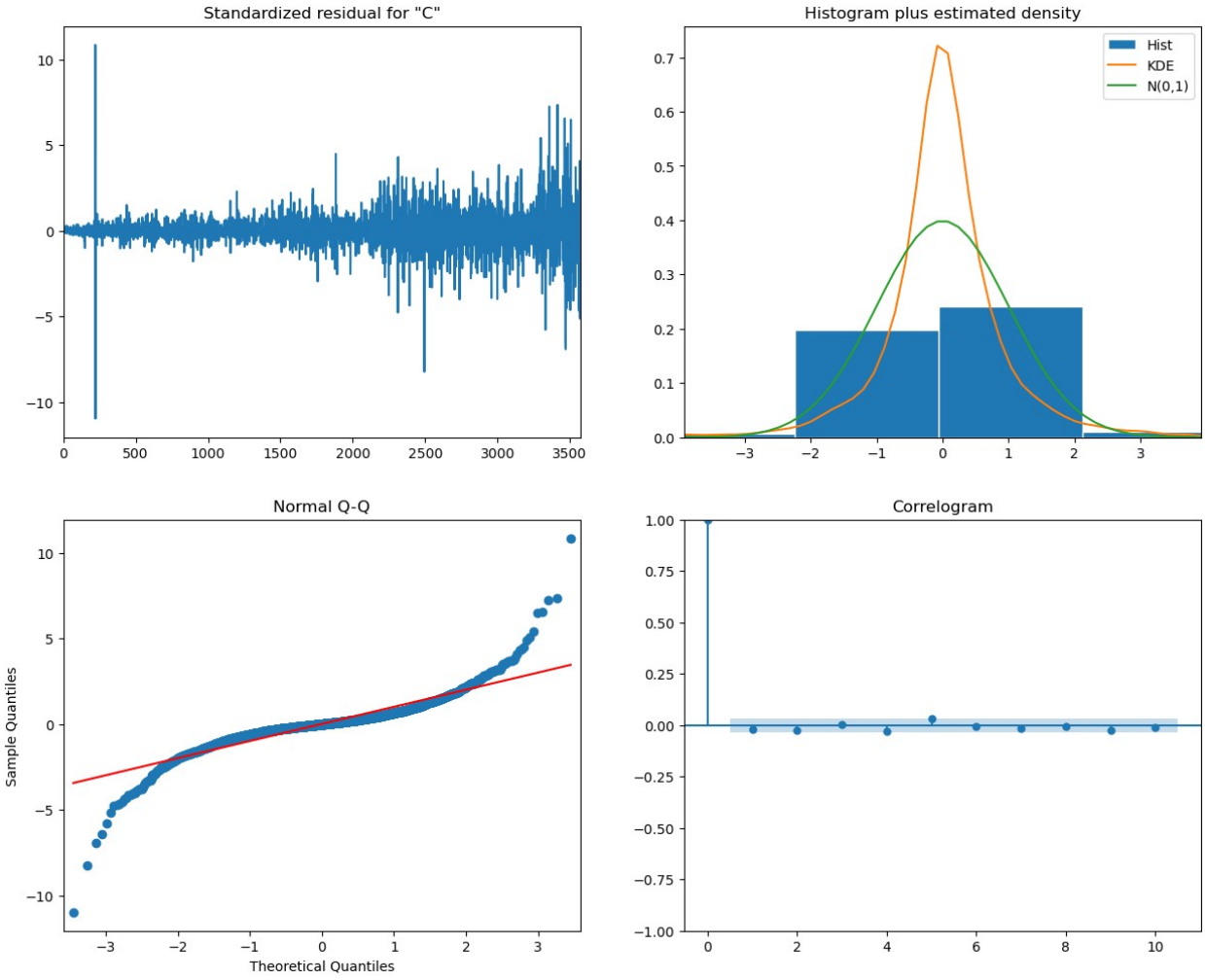
```
forecast = model_fit.forecast(steps=len(test))
```

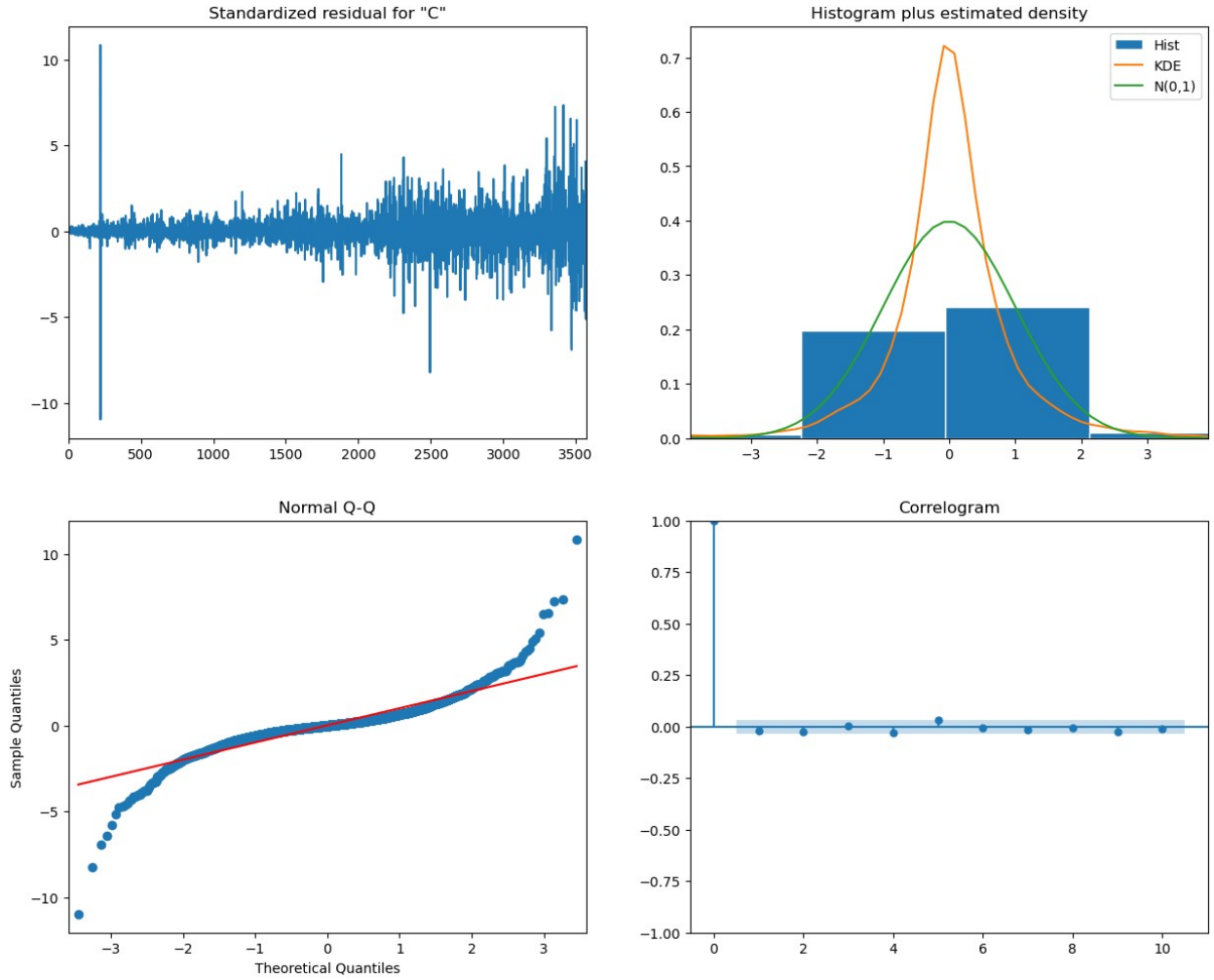
```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:836: ValueWarning: No supported index is available.
Prediction results will be given with an integer index beginning at
`start`.
  return get_prediction_index(
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\
tsa_model.py:836: FutureWarning: No supported index is available. In
the next version, calling this method in a model without a supported
index will result in an exception.
  return get_prediction_index(
```

```
plt.figure(figsize=(14,7))
plt.plot(train.index, train["Close"], label='Train', color='#203147')
plt.plot(test.index, test["Close"], label='Test', color='#01ef63')
plt.plot(test.index, forecast, label='Forecast', color='orange')
plt.title('Close Price Forecast using SARIMA')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
plt.show()
```




```
model_fit.plot_diagnostics(figsize=(15, 12))
```


Standardized residual for "C"
Histogram plus estimated density
Hist
KDE
N(0,1)
Normal Q-Q
Sample Quantiles
Theoretical Quantiles
Correlogram




```
from sklearn.metrics import mean_squared_error
import numpy as np

rmse = np.sqrt(mean_squared_error(test["Close"], forecast))
print("RMSE:", rmse)
```

```
RMSE: 749.4810675385451
```

Prophet

```
df = pd.read_csv('TCS.CSV')

df['Date'] = pd.to_datetime(df['Date'])
df = df[['Date', 'Close']]

!pip install Prophet
```

```
Collecting Prophet
  Downloading prophet-1.1.6-py3-none-win_amd64.whl.metadata (3.6 kB)
Collecting cmdstanpy>=1.0.4 (from Prophet)
```

```
  Downloading cmdstanpy-1.2.5-py3-none-any.whl.metadata (4.0 kB)
Requirement already satisfied: numpy>=1.15.4 in c:\programdata\
anaconda3\lib\site-packages (from Prophet) (1.26.4)
Requirement already satisfied: matplotlib>=2.0.0 in c:\programdata\
anaconda3\lib\site-packages (from Prophet) (3.9.2)
Requirement already satisfied: pandas>=1.0.4 in c:\programdata\
anaconda3\lib\site-packages (from Prophet) (2.2.2)
Collecting holidays<1,>=0.25 (from Prophet)
  Downloading holidays-0.70-py3-none-any.whl.metadata (34 kB)
Requirement already satisfied: tqdm>=4.36.1 in c:\programdata\
anaconda3\lib\site-packages (from Prophet) (4.66.5)
Collecting importlib-resources (from Prophet)
  Downloading importlib_resources-6.5.2-py3-none-any.whl.metadata (3.9
kB)
Collecting stanio<2.0.0,>=0.4.0 (from cmdstanpy>=1.0.4->Prophet)
  Downloading stanio-0.5.1-py3-none-any.whl.metadata (1.6 kB)
Requirement already satisfied: python-dateutil in c:\programdata\
anaconda3\lib\site-packages (from holidays<1,>=0.25->Prophet)
(2.9.0.post0)
Requirement already satisfied: contourpy>=1.0.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib>=2.0.0->Prophet) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib>=2.0.0->Prophet) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib>=2.0.0->Prophet) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib>=2.0.0->Prophet) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib>=2.0.0->Prophet) (24.1)
Requirement already satisfied: pillow>=8 in c:\programdata\anaconda3\
lib\site-packages (from matplotlib>=2.0.0->Prophet) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib>=2.0.0->Prophet) (3.1.2)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\
anaconda3\lib\site-packages (from pandas>=1.0.4->Prophet) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\
anaconda3\lib\site-packages (from pandas>=1.0.4->Prophet) (2023.3)
Requirement already satisfied: colorama in c:\programdata\anaconda3\
lib\site-packages (from tqdm>=4.36.1->Prophet) (0.4.6)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\
lib\site-packages (from python-dateutil->holidays<1,>=0.25->Prophet)
(1.16.0)
Downloading prophet-1.1.6-py3-none-win_amd64.whl (13.3 MB)
   ---------------------------------------- 0.0/13.3 MB ? eta -:--:--
   - -------------------------------------- 0.5/13.3 MB 2.8 MB/s eta
0:00:05
   ----- ---------------------------------- 1.8/13.3 MB 4.2 MB/s eta
0:00:03
   -------- ------------------------------- 2.9/13.3 MB 4.8 MB/s eta
```

```
0:00:03
   ---------- ---------------------------- 3.9/13.3 MB 4.5 MB/s eta
0:00:03
   ---------- ---------------------------- 5.2/13.3 MB 4.7 MB/s eta
0:00:02
   ---------- ---------------------------- 6.6/13.3 MB 4.9 MB/s eta
0:00:02
   ---------- ---------------------------- 7.9/13.3 MB 4.8 MB/s eta
0:00:02
   ---------- ---------------------------- 9.2/13.3 MB 4.9 MB/s eta
0:00:01
   ---------- ---------------------------- 10.0/13.3 MB 4.8 MB/s eta
0:00:01
   ---------- ---------------------------- 10.5/13.3 MB 4.6 MB/s eta
0:00:01
   ---------- ---------------------------- 11.0/13.3 MB 4.5 MB/s eta
0:00:01
   ---------- ---------------------------- 12.1/13.3 MB 4.4 MB/s eta
0:00:01
   ---------- ---------------------------- 13.1/13.3 MB 4.5 MB/s eta
0:00:01
   ---------- ---------------------------- 13.3/13.3 MB 4.2 MB/s eta
0:00:00
Downloading cmdstanpy-1.2.5-py3-none-any.whl (94 kB)
Downloading holidays-0.70-py3-none-any.whl (903 kB)
   ---------- ---------------------------- 0.0/903.1 kB ? eta -:--:--
   ---------- ---------------------------- 903.1/903.1 kB 4.5 MB/s
eta 0:00:00
Downloading importlib_resources-6.5.2-py3-none-any.whl (37 kB)
Downloading stanio-0.5.1-py3-none-any.whl (8.1 kB)
Installing collected packages: stanio, importlib-resources, holidays,
cmdstanpy, Prophet
Successfully installed Prophet-1.1.6 cmdstanpy-1.2.5 holidays-0.70
importlib-resources-6.5.2 stanio-0.5.1
```

```python
from prophet import Prophet

df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
df.dropna(inplace=True)

train_size = int(len(df) * 0.8)
train = df.iloc[:train_size]
test = df.iloc[train_size:]

model = Prophet()
model.fit(train)
```

```
01:05:05 - cmdstanpy - INFO - Chain [1] start processing
01:05:10 - cmdstanpy - INFO - Chain [1] done processing

<prophet.forecaster.Prophet at 0x222675295e0>
```

```
future = model.make_future_dataframe(periods=len(test))
```