EX.No:8 DATE:	**Create an ARIMA model for time series forecasting**

AIM:

To build an ARIMA model for forecasting future values of a time series based on historical data.

ALGORITHM:

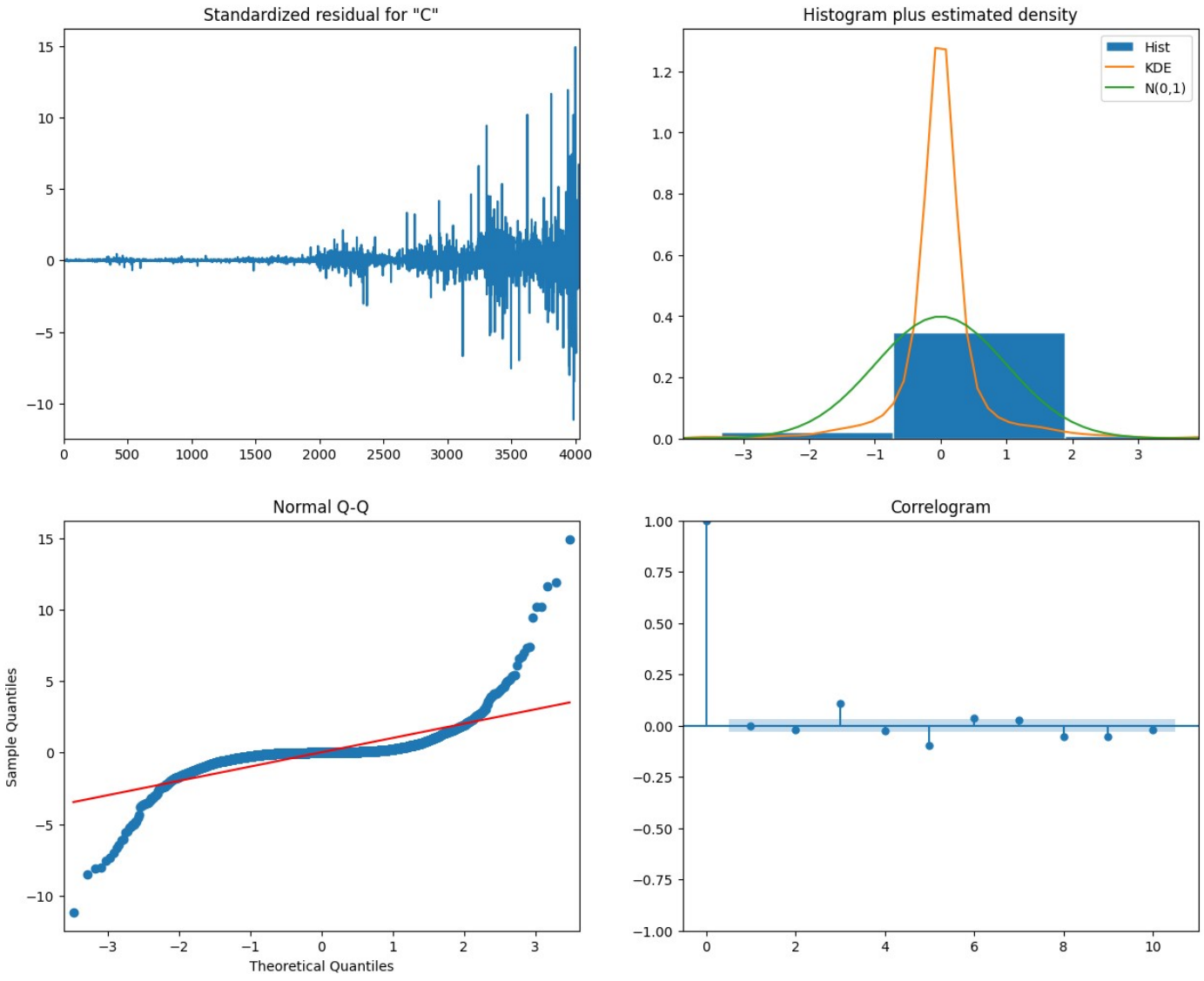
1. Load the time series data and set the date as the index.
2. Plot the data and check for stationarity.
3. Select ARIMA parameters (p, d, q).
4. Fit the ARIMA model to the data.
5. Forecast future values and visualize the results.

CODE:

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
from sklearn.model_selection import train_test_split
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np

# Load data
df = pd.read_csv('NFLX (1).csv', parse_dates=['Date'])

# Data preparation
df['Close_diff'] = df['Close'].diff()
df.dropna(subset=['Close_diff'], inplace=True)

# Data splitting
train_data, test_data = train_test_split(df, test_size=0.2, shuffle=False)

# Model training
p = 1
d = 1
q = 1
arima_model = ARIMA(train_data['Close_diff'], order=(p, d, q))
```

```
arima_result = arima_model.fit()

# Model evaluation
predictions = arima_result.predict(start=len(train_data), end=len(df)-1)
predictions.index = test_data.index
rmse = np.sqrt(mean_squared_error(test_data['Close_diff'], predictions))
mae = mean_absolute_error(test_data['Close_diff'], predictions)
mape = np.mean(np.abs((test_data['Close_diff'] - predictions) / (test_data['Close_diff'] + 1e-8))) * 100
print(f"RMSE: {rmse}")
print(f"MAE: {mae}")
print(f"MAPE: {mape}")
```

OUTPUT:



RESULT:

Thus the program has been completed and verified successfully.