EX.No:5 DATE: 25/01/2	**Develop A Linear Regression Model For The Time Series Data**

AIM:

To build a linear regression model for electricity production data, analyze trends over time, and visualize the results.

ALGORITHM:

1. Load the electricity production data from the CSV file.
2. Parse the DATE column and convert it to datetime format.
3. Create a numerical time index representing the number of days since the start.
4. Handle missing values by dropping or imputing them if necessary.
5. Fit a linear regression model using the time index as the independent variable and production values as the dependent variable.
6. Generate predictions from the trained model.
7. Plot the actual data points and the fitted linear regression trend to visualize the relationship.

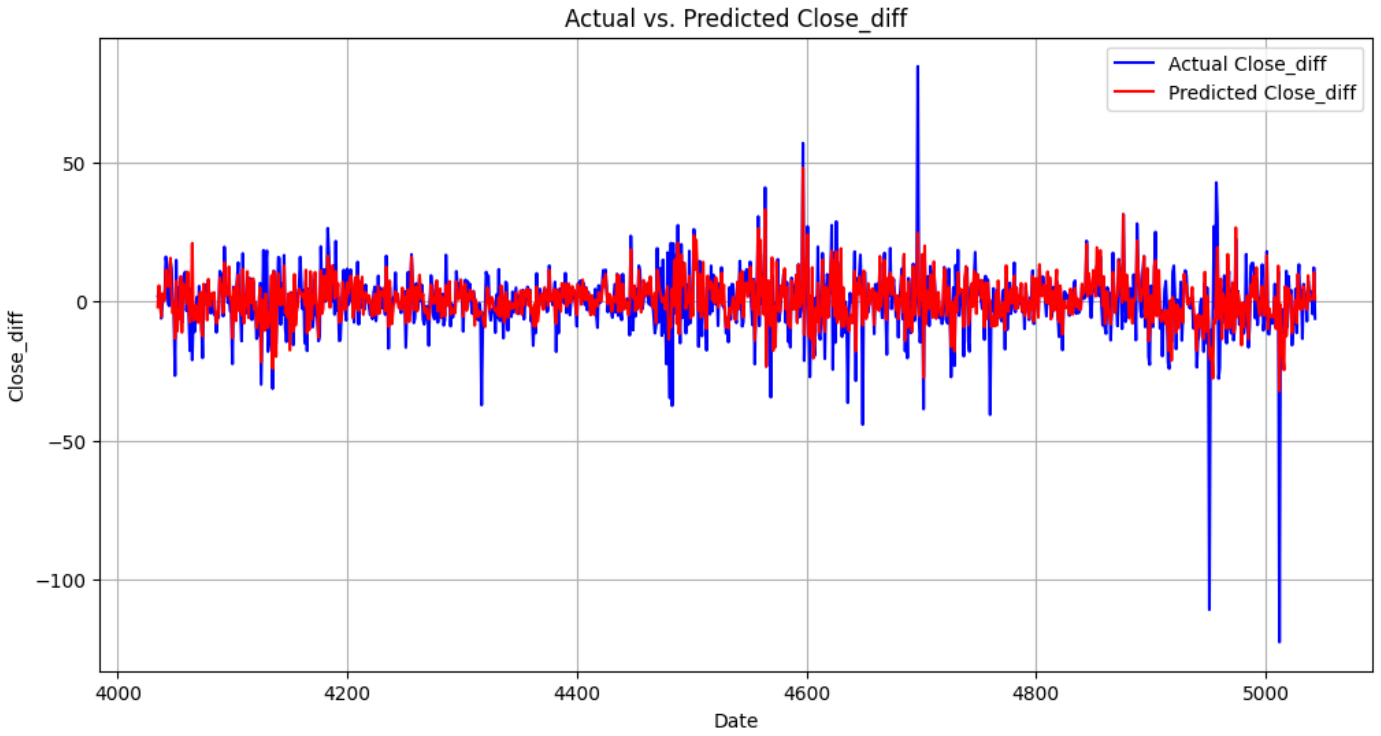
CODE:

```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.impute import SimpleImputer
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np
import matplotlib.pyplot as plt

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

# Feature Engineering
df['Day'] = df['Date'].dt.dayofweek
df['Month'] = df['Date'].dt.month
df['Year'] = df['Date'].dt.year
df['Quarter'] = df['Date'].dt.quarter
df['Close_diff'] = df['Close'].diff()
df.dropna(inplace=True)
```

```
# Data Splitting
train_size = int(len(df) * 0.8)
df_train = df.iloc[:train_size]
df_test = df.iloc[train_size:]

# More Feature Engineering
for df in [df_train, df_test]:
   df.loc[:, 'Close_diff_rolling_mean_7'] = df['Close_diff'].rolling(window=7,
min_periods=1).mean()
   df.loc[:, 'Close_diff_rolling_mean_30'] = df['Close_diff'].rolling(window=30,
min_periods=1).mean()
   df.loc[:, 'Close_diff_lag_1'] = df['Close_diff'].shift(1)
   df.loc[:, 'Close_diff_lag_5'] = df['Close_diff'].shift(5)
   df.ffill(inplace=True)

# Define features and target
features = ['Open', 'High', 'Low', 'Volume', 'Day', 'Month', 'Year', 'Quarter',
'Close_diff_rolling_mean_7', 'Close_diff_rolling_mean_30', 'Close_diff_lag_1',
'Close_diff_lag_5']
target = 'Close_diff'

X_train = df_train[features]
y_train = df_train[target]
X_test = df_test[features]
y_test = df_test[target]

# Impute missing values
imputer = SimpleImputer(strategy='mean')
X_train = imputer.fit_transform(X_train)
X_test = imputer.transform(X_test)

# Train the model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
predictions = model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, predictions)
rmse = np.sqrt(mean_squared_error(y_test, predictions))
r2 = r2_score(y_test, predictions)
```

```
mape = np.mean(np.abs((y_test - predictions) / y_test)) * 100

print(f"MAE: {mae}")
print(f"RMSE: {rmse}")
print(f"MAPE: {mape}")
print(f"R-squared: {r2}")

# (Optional) Visualize predictions
# ... (refer to the original code for visualization)
```

OUTPUT:



RESULT:

Thus the program has been completed and verified successfully.