EX:4 Implementation of Stationary checking

**AIM**

To implement the Implementation of Stationary checking using time series data

**Algorithm:**

Step 1: Load and Visualize the Data

1. Load the CAD/JPY exchange rate dataset.
2. Plot the original time series to observe trends and patterns.

Step 2: Check for Stationarity

1. Compute and plot rolling mean and standard deviation to assess variance stability.
2. Perform the Augmented Dickey-Fuller (ADF) test to check stationarity.
   * If p-value > 0.05, the data is non-stationary.

Step 3: Make the Data Stationary (If Needed)

1. Apply differencing (Price.diff()) to remove trends.
2. Re-run the ADF test to confirm stationarity.
   * If p-value ≤ 0.05, the data is now stationary.

Step 4: Visualize the Transformed Data

1. Plot the differenced time series to confirm the transformation.

**CODE**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.stattools import adfuller

cad\_jpy\_df = pd.read\_csv("cad\_jpy.csv", index\_col="Date", infer\_datetime\_format=True, parse\_dates=True)

plt.figure(figsize=(12, 6))

plt.plot(cad\_jpy\_df["Price"], label="CAD/JPY Price")

plt.title("CAD/JPY Exchange Rate Over Time")

plt.xlabel("Date")

plt.ylabel("Exchange Rate")

plt.legend()

plt.show()

# Check for stationarity using Rolling Mean & Standard Deviation

rolling\_mean = cad\_jpy\_df["Price"].rolling(window=30).mean()

rolling\_std = cad\_jpy\_df["Price"].rolling(window=30).std()

plt.figure(figsize=(12, 6))

plt.plot(cad\_jpy\_df["Price"], label="Original Price")

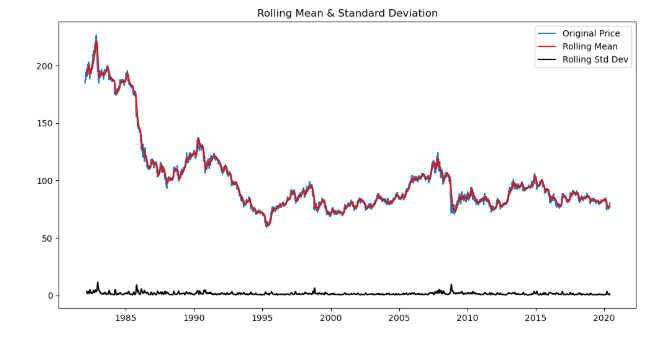
plt.plot(rolling\_mean, label="Rolling Mean", color='red')

plt.plot(rolling\_std, label="Rolling Std Dev", color='black')

plt.title("Rolling Mean & Standard Deviation")

plt.legend()

plt.show()



# Augmented Dickey-Fuller Test

def adf\_test(series):

result = adfuller(series.dropna())

labels = ['ADF Statistic', 'p-value', '# Lags Used', 'Number of Observations']

for label, value in zip(labels, result[:4]):

print(f"{label}: {value}")

print("Critical Values:")

for key, value in result[4].items():

print(f"{key}: {value}")

if result[1] <= 0.05:

print("\nData is stationary (reject H0).")

else:

print("\nData is non-stationary (fail to reject H0).")

print("ADF Test on Original Data:")

adf\_test(cad\_jpy\_df["Price"])

# If non-stationary, apply differencing

cad\_jpy\_df['Price\_Diff'] = cad\_jpy\_df['Price'].diff()

# Re-check stationarity

plt.figure(figsize=(12, 6))

plt.plot(cad\_jpy\_df['Price\_Diff'], label="Differenced Price")

plt.title("Differenced Time Series Data")

plt.xlabel("Date")

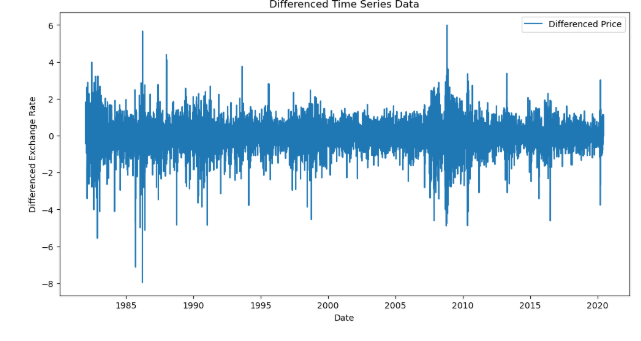
plt.ylabel("Differenced Exchange Rate")

plt.legend()

plt.show()

print("\nADF Test on Differenced Data:")

adf\_test(cad\_jpy\_df['Price\_Diff'])



**Result:**

Thus the program to implement the stationary checking is done successfully using the time series data