**Exp no: 4 Implement programs for analyzing stationarity in time series data**

**Date: 20/03/25**

**Objectives**:

The primary objective of this experiment is to analyze and determine the stationarity of time series stock market data. By applying statistical tests such as the Augmented Dickey-Fuller (ADF) test, we aim to identify whether the dataset exhibits stationarity. If the data is non-stationary, we will apply seasonal differencing to stabilize the mean and remove trends.

**Background/Scope:**

Time series data is often non-stationary, meaning it has trends and seasonal variations. Stationarity is a key assumption for many time series forecasting models. The Augmented Dickey-Fuller (ADF) test is a widely used statistical test to check for stationarity. If the data is found to be non-stationary, techniques such as differencing are applied to make it stationary.

**Steps for Time Series Stationarity Analysis:**

Step 1: Load the Dataset

Load the dataset from a local CSV file and display the first few rows to understand its structure.

import pandas as pd

# Load the dataset

df = pd.read\_csv("market.csv")

# Display the first few rows

print(df.head())

Step 2: Data Cleaning and Preprocessing

Convert the date column to a datetime object and remove missing values.

# Convert 'Date' to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Remove missing values

df.dropna(subset=['Date', 'Close', 'Adj Close', 'Open', 'High', 'Low', 'Volume'], inplace=True)

Step 3: Visualizing Time Series Data

Plot the stock volume over time to observe trends and patterns.

import matplotlib.pyplot as plt

# Plot stock volume over time

plt.figure(figsize=(12, 5))

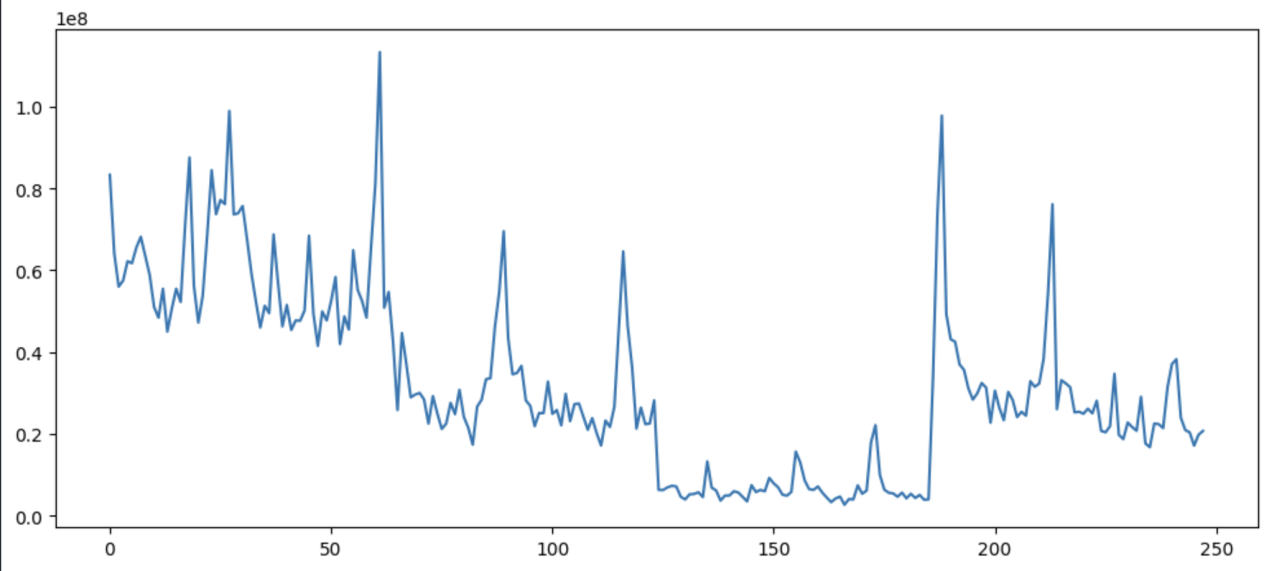
plt.plot(df['Volume'], color='blue')

plt.title('Stock Volume Over Time')

plt.xlabel('Time')

plt.ylabel('Volume')

plt.grid(True)

plt.show()

Step 4: Perform Augmented Dickey-Fuller (ADF) Test

The ADF test checks whether the time series data is stationary or not.

from statsmodels.tsa.stattools import adfuller

def adfuller\_test(series):

result = adfuller(series)

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

for value, label in zip(result, labels):

print(label+': '+str(value))

if result[1] <= 0.05:

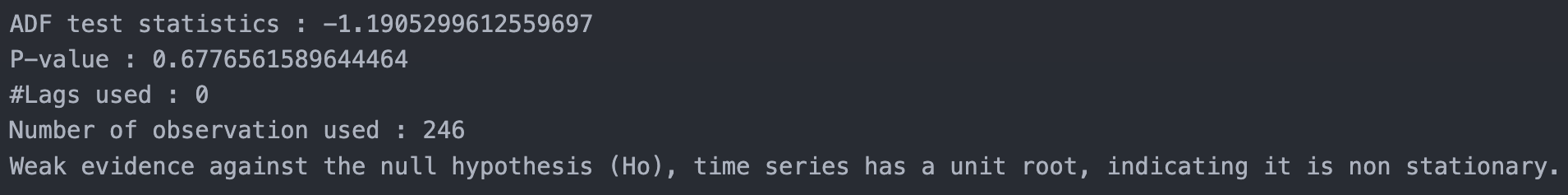
print("Strong evidence against the null hypothesis (H0), data is stationary.")

else:

print("Weak evidence against the null hypothesis (H0), data is non-stationary.")

# Perform ADF test on 'Close' column

adfuller\_test(df['Close'])



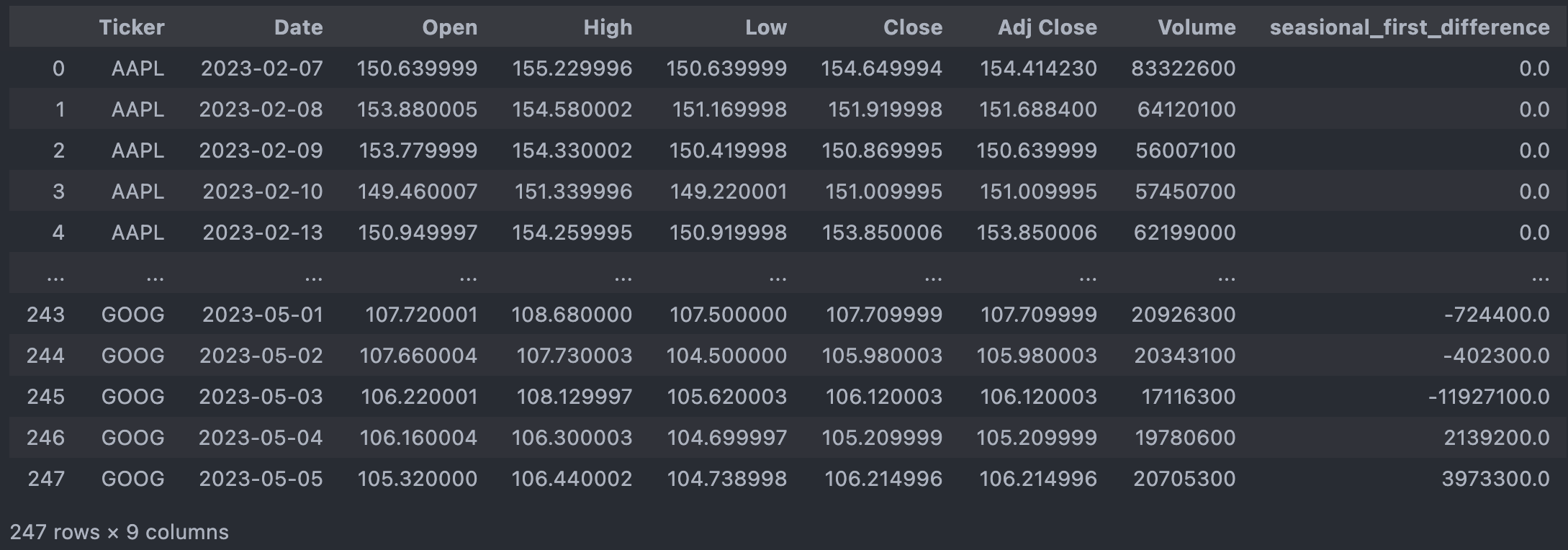
Step 5: Apply Seasonal Differencing

If the data is found to be non-stationary, apply seasonal differencing to remove trends.

# Apply seasonal differencing

df['seasonal\_first\_difference'] = df['Volume'] - df['Volume'].shift(12)

df['seasonal\_first\_difference'].fillna(0, inplace=True)



Step 6: Visualizing Differenced Data

Plot the differenced time series data to check for stationarity.

plt.figure(figsize=(12, 5))

plt.plot(df['seasonal\_first\_difference'], color='red')

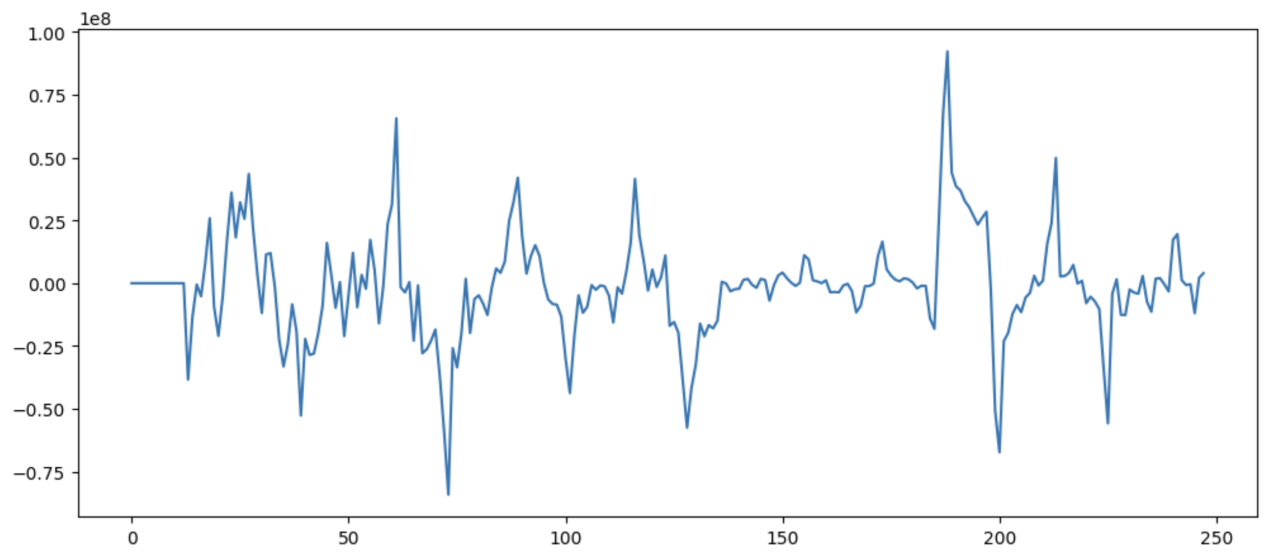
plt.title('Differenced Time Series Data')

plt.xlabel('Time')

plt.ylabel('Differenced Volume')

plt.grid(True)

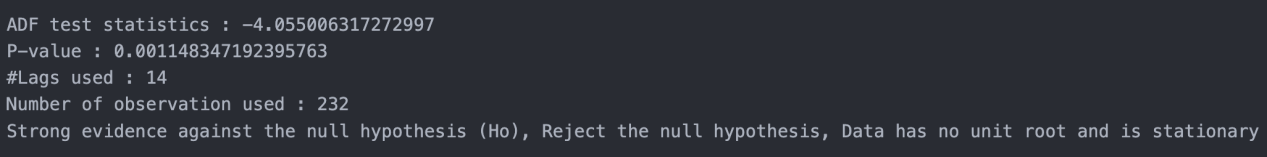
plt.show()



Step 7: Re-Test Stationarity with ADF Test

Perform the ADF test again on the differenced data to confirm stationarity.

adfuller\_test(df['seasonal\_first\_difference'].dropna())



**Result:**

Thus, the stationarity of the time series dataset is successfully analyzed. If required, transformations such as differencing are applied to make the data stationary for further forecasting applications.