Exp no: 3

Date: 23/2/25

### ****Objective****

### **To implement Python programs that check the stationarity of a time series using both visual and statistical methods. Stationarity ensures that the statistical properties of the time series, such as mean and variance, remain constant over time, which is essential for many forecasting models like ARIMA to work properly.**

### ****Background****

### **In time series analysis, stationarity refers to a property where the statistical characteristics of the series do not change over time. Non-stationary data can lead to misleading results when building predictive models. Therefore, testing for stationarity is a critical preprocessing step.**

### ****Types of stationarity:****

### **Strict Stationarity: The complete distribution of the data does not change over time.**

### **Weak Stationarity: The mean, variance, and autocovariance are time-invariant.**

### **Common methods to test stationarity:**

### **Rolling statistics (Visual Inspection)**

### **Augmented Dickey-Fuller Test (ADF)**

### **Kwiatkowski-Phillips-Schmidt-Shin Test (KPSS)**

### ****Scope****

### ****Load a time series dataset.****

### **Visualize the original series.**

### **Apply statistical tests (ADF and KPSS) to check for stationarity.**

### **Interpret the results.**

### **If non-stationary, transform the series using differencing and retest.**

### **\_**

### ****Step-by-Step Implementation with Explanation****

### ****Step 1: Install and Import Required Libraries****

### **import pandas as pd**

### **import numpy as np**

### **import matplotlib.pyplot as plt**

### **from statsmodels.tsa.stattools import adfuller, kpss**

### ****Step 2: Load and Plot the Time Series Data****

### **url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/airline-passengers.csv'**

### **df = pd.read\_csv(url, parse\_dates=['Month'], index\_col='Month')**

### **plt.figure(figsize=(10, 4))**

### **plt.plot(df, label='Monthly Passengers')**

### **plt.title('Monthly Air Passengers')**

### **plt.xlabel('Year')**

### **plt.ylabel('Number of Passengers')**

### **plt.legend()**

### **plt.grid()**

### **plt.show()**

### **Explanation: This visualization helps observe trend or seasonality in the data.**

### ****Step 3: Calculate and Plot Rolling Mean and Standard Deviation****

### **rolling\_mean = df.rolling(window=12).mean()**

### **rolling\_std = df.rolling(window=12).std()**

### **plt.figure(figsize=(10, 4))**

### **plt.plot(df, label='Original')**

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

### **plt.plot(rolling\_std, label='Rolling Std Dev', color='green')**

### **plt.legend()**

### **plt.title("Rolling Mean and Standard Deviation")**

### **plt.grid()**

### **plt.show()**

### **Explanation: If the rolling mean and standard deviation vary over time, the data is likely non-stationary.**

### ****Step 4: Augmented Dickey-Fuller (ADF) Test****

### **def adf\_test(series):**

### **result = adfuller(series.dropna())**

### **print("ADF Statistic:", result[0])**

### **print("p-value:", result[1])**

### **print("Critical Values:")**

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

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

### **if result[1] <= 0.05:**

### **print("=> The series is Stationary")**

### **else:**

### **print("=> The series is Non-Stationary")**

### **adf\_test(df['Passengers'])**

### **Explanation: The null hypothesis of ADF test is that the data is non-stationary. A p-value below 0.05 leads us to reject the null hypothesis.**

### ****Step 5: KPSS Test****

### **def kpss\_test(series):**

### **result = kpss(series.dropna(), regression='c')**

### **print("KPSS Statistic:", result[0])**

### **print("p-value:", result[1])**

### **print("Critical Values:")**

### **for key, value in result[3].items():**

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

### **if result[1] > 0.05:**

### **print("=> The series is Stationary")**

### **else:**

### **print("=> The series is Non-Stationary")**

### **kpss\_test(df['Passengers'])**

### **Explanation: KPSS test is the opposite of ADF. Here, the null hypothesis is that the data is stationary. If p-value > 0.05, the series is considered stationary.**

### ****Step 6: Differencing the Series (If Non-Stationary)****

### **df\_diff = df.diff().dropna()**

### **plt.figure(figsize=(10, 4))**

### **plt.plot(df\_diff, label='First Order Differenced Series')**

### **plt.title('First Differencing of Series')**

### **plt.grid()**

### **plt.show()**

### **# Retesting**

### **print("After Differencing:")**

### **adf\_test(df\_diff['Passengers'])**

### **kpss\_test(df\_diff['Passengers'])**

### **Explanation: Differencing removes the trend component and makes the series more stationary. After differencing, we retest using ADF and KPSS.**

### 

### ****Conclusion****

### **Stationarity is an essential assumption for many time series models.**

### **Visual inspection using rolling statistics provides an initial idea about stationarity.**

### **Statistical tests such as ADF and KPSS give more accurate and formal assessment.**

### **If a series is found non-stationary, it can be made stationary using transformations such as differencing.**

### **Once the series is stationary, it can be used for reliable time series modeling and forecasting.**