**Exp no: 2 Implement programs for visualizing time series data**

**Date: 31/1/25**

**Objectives:**

The objective of visualizing Netflix time series data is to gain insights into viewing patterns, trends, and user behavior over time. This involves identifying trends, such as long-term growth or decline in viewership, popular genres, or specific titles; seasonality, recognizing recurring patterns related to time of year, holidays, or other cyclical events; user engagement, understanding how users interact with the platform (e.g., viewing frequency, binge-watching habits, device usage); content performance, evaluating the success of different content types, genres, or individual shows/movies; and forecasting, predicting future viewership trends to inform content acquisition, marketing strategies, and platform development.

**Background/Scope:**

### Netflix time series data, encompassing metrics like viewing hours, number of views, user demographics, and content metadata, offers valuable insights into user behavior and content performance. The scope of visualizing this data is to explore these patterns, understand user engagement, assess content success, and ultimately inform strategic decisions related to content creation, licensing, marketing, and platform development.

### ****Steps for Time Series Sales Data Preprocessing****:

**Step 1: Load the Dataset and libraries**

1. **Load** the dataset from a CSV file into a Pandas DataFrame.

# Step 1: Load the Dataset

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from statsmodels.graphics.tsaplots import plot\_acf

from statsmodels.tsa.stattools import adfuller

df=pd.read\_csv("/content/NFLX.csv")

**Step 2: Visualizing techniques**

**Autocorrelation Plot:** Create an autocorrelation plot to identify potential seasonality using the acf\_plot

if 'Date' not in df.columns:

    print("'Date' is already the index or not present in the DataFrame.")

else:

    df.set\_index('Date', inplace=True)

# Plot the ACF

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

plot\_acf(df['Volume'], lags=40) # You can adjust the number of lags as needed

plt.xlabel('Lag')

plt.ylabel('Autocorrelation')

plt.title('Autocorrelation Function (ACF) Plot')

plt.show()

**Lag Plot:** Create a lag plot (scatter plot of the time series against lagged values of itself) to visualize autocorrelation.

from pandas.plotting import lag\_plot

# lag scatter plot

lag\_plot(df)

plt.show()

**Line Plot:**

sns.set(style="whitegrid")  # Setting the style to whitegrid for a clean background

# Plotting the 'high' column with seaborn, setting x as the resampled 'Date'

plt.figure(figsize=(12, 6))  # Setting the figure size

sns.lineplot(data=df, x=df.index, y='High', label='Month Wise Average High Price', color='blue')

# Adding labels and title

plt.xlabel('Date (Monthly)')

plt.ylabel('High')

plt.title('Monthly Resampling Highest Price Over Time')

plt.show()

**Histogram Plot**:

plt.hist(df['Open'])

plt.title("Histogram plot for Opening price of the stock")

# Adding the legends

plt.show()

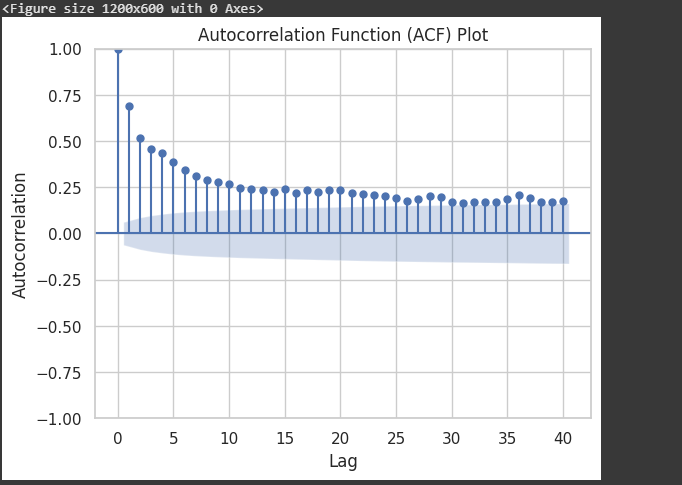
**Box Plot:**

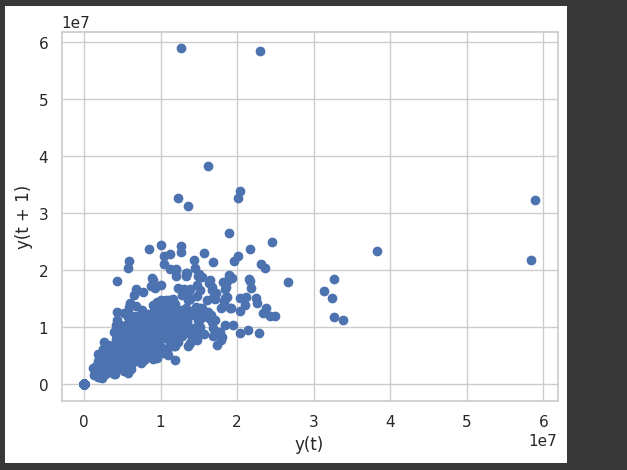
sns.boxplot(df, x="Date", y="High")

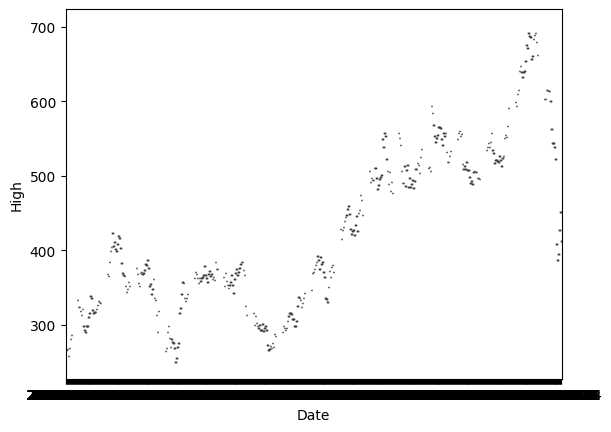
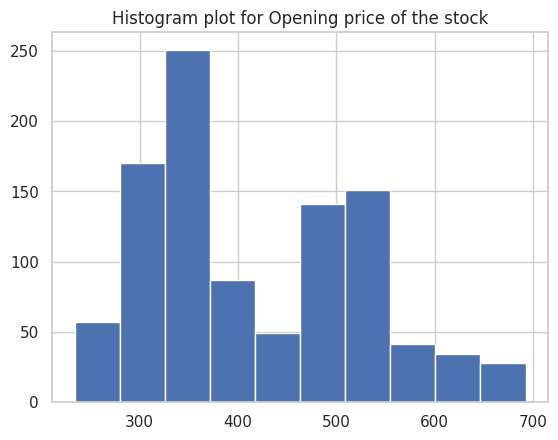
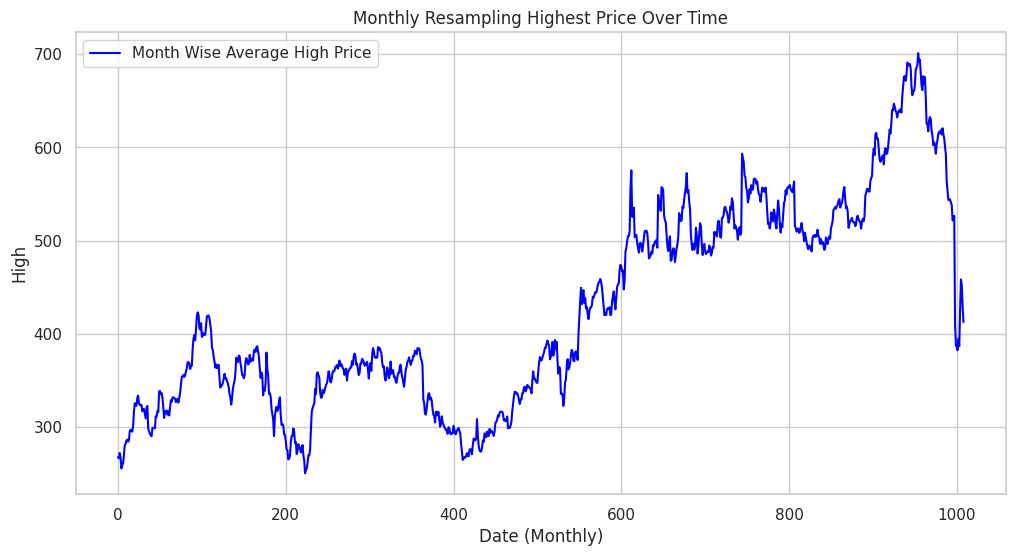
plt.figure(figsize=(25,25))

plt.show()

output :







Result:

Thus the time series dataset is visualized successfully.