**EX:No.1 221501001**

**21/01/25**

**PROGRAM TO IMPLEMENT TIME SERIES DATA FOR IMPORT LIBRARY, LOAD DATA , PREPROCESSING AND VISUALISING**

**Aim:**

The aim of this experiment is to implement time series data processing by importing required libraries, loading stock data, preprocessing it to handle missing values and outliers, and visualizing the dataset to understand trends.

**Objectives:**

The primary objective of this analysis is to apply preprocessing techniques to stock market data and visualize trends. The experiment focuses on:

1. Importing necessary libraries for data handling and visualization.
2. Loading and processing stock data.
3. Handling missing values and outliers for data quality improvement.
4. Normalizing and splitting data for better analysis.
5. Visualizing trends using line charts to assess stock price movements.

**Background/Scope:**

Stock market data is a classic example of time series data, where prices fluctuate over time. Understanding and analyzing such data requires preprocessing techniques to clean the dataset, normalize values, and visualize trends effectively. This experiment helps in developing a foundational understanding of handling financial data for further predictive analysis and modeling.

**Code:**

Step 1: Importing libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

step2 : Load the stock data

file\_path = r'AAPL\_short\_volume.csv'

data = pd.read\_csv(file\_path)

close\_prices\_AAPL = data['Close']

**Step 3: Data Preprocessing**

1. **Handling Missing Values**

# Check for missing values in each column

print(data.isnull().sum())

# Drop rows with missing values

data.dropna(inplace=True)

# Fill missing values in 'Close' column with mean

data['Close'].fillna(data['Close'].mean(), inplace=True)

1. **Handling Outliers**

# Calculate IQR for outlier detection

Q1 = data['Close'].quantile(0.25)

Q3 = data['Close'].quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

# Remove outliers

data = data[(data['Close'] >= lower\_bound) & (data['Close'] <= upper\_bound)]

1. **Data Normalization & Splitting**

# Reverse the order of the data

close\_prices\_AAPL = data['Close'].iloc[::-1]

close\_prices\_AAPL.reset\_index(drop=True, inplace=True)

data = close\_prices\_AAPL.values.reshape(-1, 1)

data\_normalized = data / np.max(data)

# Split the data into training and testing sets

train\_size = int(len(data\_normalized) \* 0.8)

train\_data = data\_normalized[:train\_size]

test\_data = data\_normalized[train\_size:]

**step 4: visualizing**

plt.figure(figsize=(10, 6))

plt.plot(close\_prices\_AAPL, label='AAPL Close Prices')

plt.xlabel('Time')

plt.ylabel('Close Prices')

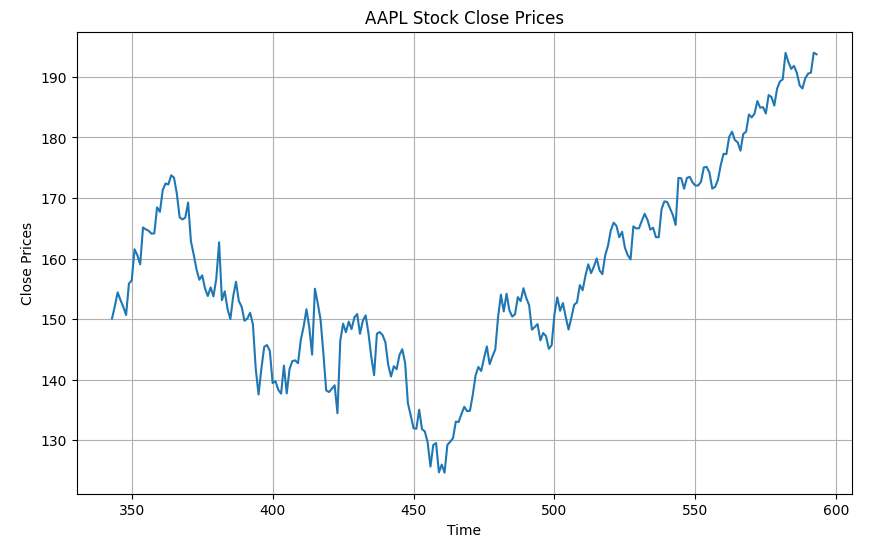
plt.title('AAPL Stock Close Prices')

plt.grid(True)

plt.legend()

plt.show()

**OUTPUT:**



**Result:**

Thus, the program using the time series data implementation has been done successfully.