Ex.no: 05 **221501008**

15/04/2025

Implement programs for estimating & eliminating trend in time series data- aggregation, smoothing.

AIM:

To Implement programs for estimating & eliminating trend in time series data- aggregation, smoothing.

PROCEDURE:

Step 1: Load the Dataset

1. Read the time series dataset containing **date** and **price** columns.
2. Ensure the dataset is sorted in chronological order.

Step 2: Preprocess the Data

1. Convert the **date** column to a proper **datetime format**.
2. Sort the dataset based on date to maintain proper sequence.

Step 3: Trend Estimation Using Aggregation

1. Extract the **Year-Month** from the date.
2. Compute the **monthly average price** by grouping data based on **Year-Month**.
3. Store the aggregated values for trend analysis.

Step 4: Trend Smoothing Using Moving Averages

1. Apply a **7-day moving average** to smooth short-term fluctuations.
2. Apply a **30-day moving average** to observe long-term trends.

Step 5: Visualize the Trends

1. Plot the **original price data** over time.
2. Overlay the **7-day moving average** to observe short-term trends.
3. Overlay the **30-day moving average** to highlight long-term trends.
4. Label axes, add a title, and use legends for better clarity.

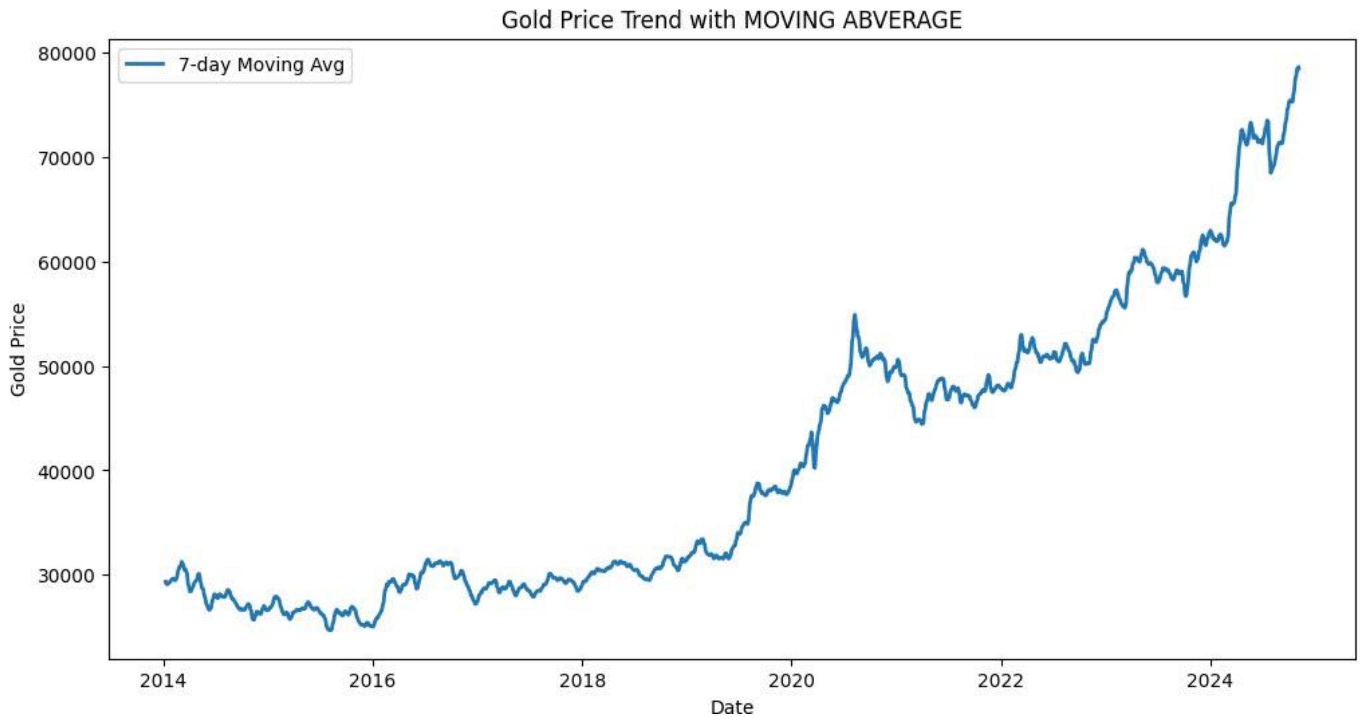
Step 6: Analyze the Results

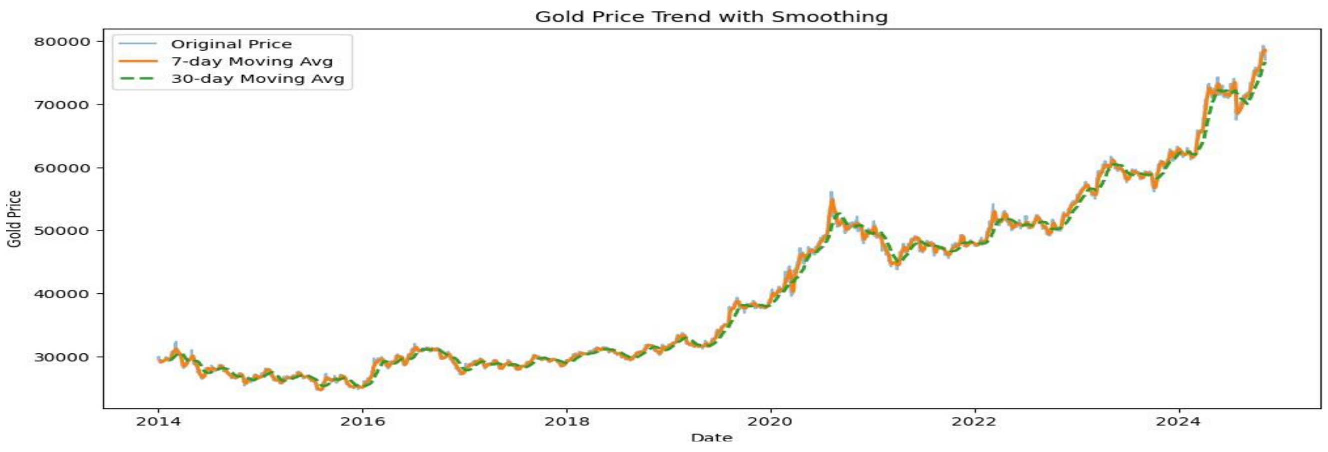
1. Identify if the trend is **increasing, decreasing, or stable**.
2. Compare short-term and long-term trends to understand market behavior.
3. Use insights for forecasting or decision-making.

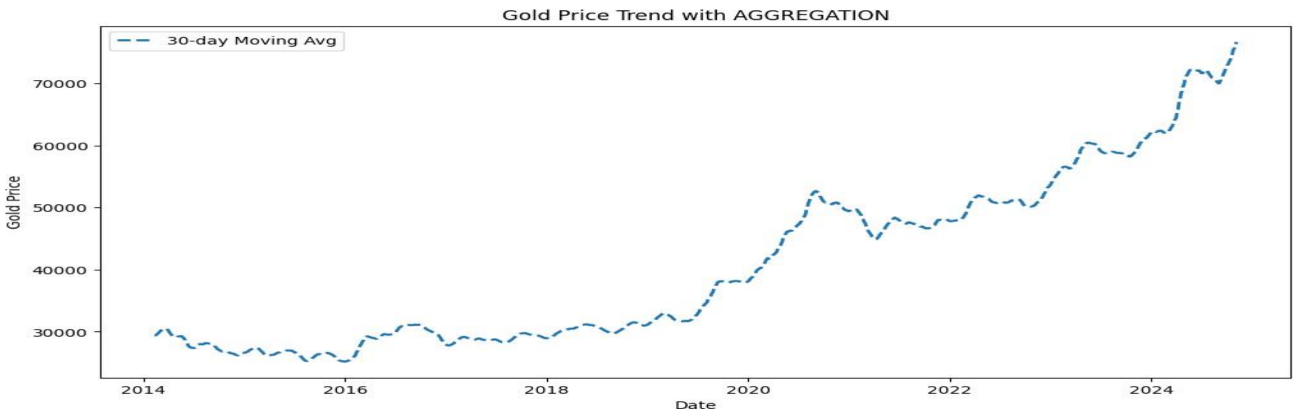
Code:

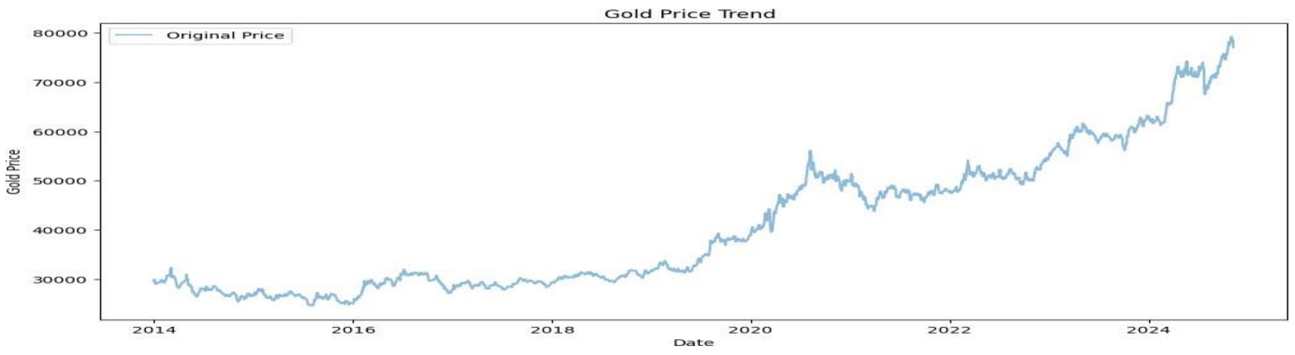
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
file_path = "/content/Gold_Price_DataSet.csv"
df = pd.read_csv(file_path)

# Convert 'Date' to datetime and sort by date
df['Date'] = pd.to_datetime(df['Date'])
df = df.sort_values('Date')

# Aggregation: Monthly Average Price
df['YearMonth'] = df['Date'].dt.to_period('M')
monthly_avg = df.groupby('YearMonth')['Price'].mean()

# Smoothing: Moving Average (7-day and 30-day)
df['Price_7MA'] = df['Price'].rolling(window=7).mean()
df['Price_30MA'] = df['Price'].rolling(window=30).mean()

# Plot original and smoothed data

plt.figure(figsize=(12, 6))
plt.plot(df['Date'], df['Price'], label="Original Price", alpha=0.5)
plt.plot(df['Date'], df['Price_7MA'], label="7-day Moving Avg", linewidth=2)
plt.plot(df['Date'], df['Price_30MA'], label="30-day Moving Avg", linewidth=2,
linestyle='dashed')
plt.xlabel("Date")
plt.ylabel("Gold Price")
plt.title("Gold Price Trend with Smoothing")
plt.legend()
plt.show()

plt.figure(1,figsize=(12, 6))
plt.plot(df['Date'], df['Price'], label="Original Price", alpha=0.5)
plt.xlabel("Date")
plt.ylabel("Gold Price")
plt.title("Gold Price Trend ")
plt.legend()
plt.show()

plt.figure(2,figsize=(12, 6))
plt.plot(df['Date'], df['Price_7MA'], label="7-day Moving Avg", linewidth=2)

plt.xlabel("Date")
plt.ylabel("Gold Price")
plt.title("Gold Price Trend with MOVING ABVERAGE")
plt.legend()
plt.show()
plt.figure(3,figsize=(12, 6))

plt.plot(df['Date'], df['Price_30MA'], label="30-day Moving Avg", linewidth=2,
linestyle='dashed')
plt.xlabel("Date")
plt.ylabel("Gold Price")
plt.title("Gold Price Trend with AGGREGATION")
plt.legend()
plt.show()
```

Output:









RESULT:

The program for implementing programs for estimating & eliminating trend in time series data-aggregation, smoothing has been completed.