**Day 10: Day 11: Time Series Basics**

* Parsing datetime data using pd.to\_datetime()
* Setting a datetime column as the index
* Resampling data (daily → monthly, weekly, etc.)
* Rolling window statistics (like moving average)
* Time-based filtering
* Plotting time series data

1. Parsing datetime data using pd.to\_datetime()
2. Setting a datetime column as the index
3. Resampling data (daily → monthly, weekly, etc.)
4. Rolling window statistics (like moving average)
5. Time-based filtering
6. Plotting time series data

**🔸 1. Import required libraries**

import pandas as pd

import matplotlib.pyplot as plt

**🔸 2. Load the sample time series CSV**

df = pd.read\_csv("time\_series\_day11.csv")

print(df.head())

**🔸 3. Parse the date column and set it as index**

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

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

print(df.info())

**🔸 4. Plot the original time series (e.g., Sales over time)**

df["Sales"].plot(figsize=(10, 4), title="Sales Over Time")

plt.ylabel("Sales")

plt.grid(True)

plt.show()

**🔸 5. Resample to Monthly Sales Total**

monthly\_sales = df["Sales"].resample("M").sum()

monthly\_sales.plot(title="Monthly Sales")

plt.ylabel("Sales")

plt.show()

**🔸 6. Calculate and plot a 7-day rolling average**

df["7\_day\_avg"] = df["Sales"].rolling(window=7).mean()

df[["Sales", "7\_day\_avg"]].plot(title="Sales with 7-Day Rolling Average")

plt.ylabel("Sales")

plt.grid(True)

plt.show()

**🔸 7. Filter data for a specific time range**

filtered = df.loc["2023-06":"2023-08"]

filtered["Sales"].plot(title="Sales (June–August 2023)")

plt.show()

**📝 Mini Task**

* Use the sample CSV.
* Plot daily sales and monthly total sales.
* Apply a 7-day rolling average and compare it with the original.
* Filter and plot sales for any 2-month range.
* Try saving one of the plots to a PNG file.