Stock Price Movement Analysis Using Python

1. Introduction

Stock price analysis is crucial for traders and investors to make informed decisions. This report explores short-term stock price movements using Python and various libraries such as pandas, numpy, matplotlib, seaborn, and mplfinance. The focus is on visualizing trends, calculating indicators, and analyzing volatility.

2. Methodology

2.1 Data Collection

Stock data is loaded from a CSV file containing historical prices. Key columns include Close Price, Moving Averages, Daily Returns, Log Returns, and Volatility.

2.2 Data Processing & Analysis

Moving Averages (SMA_10 & SMA_20): Used to smooth out price fluctuations.
Daily & Log Returns: Measures stock price changes over time.
Volatility: Captures market fluctuations.
Candlestick Charts: Visual representation of stock price movements.

2.3 Libraries Used

pandas: Data handling & processing.
numpy: Numerical calculations.
matplotlib & seaborn: Data visualization.
mplfinance: Candlestick charting.

3. Code Implementation

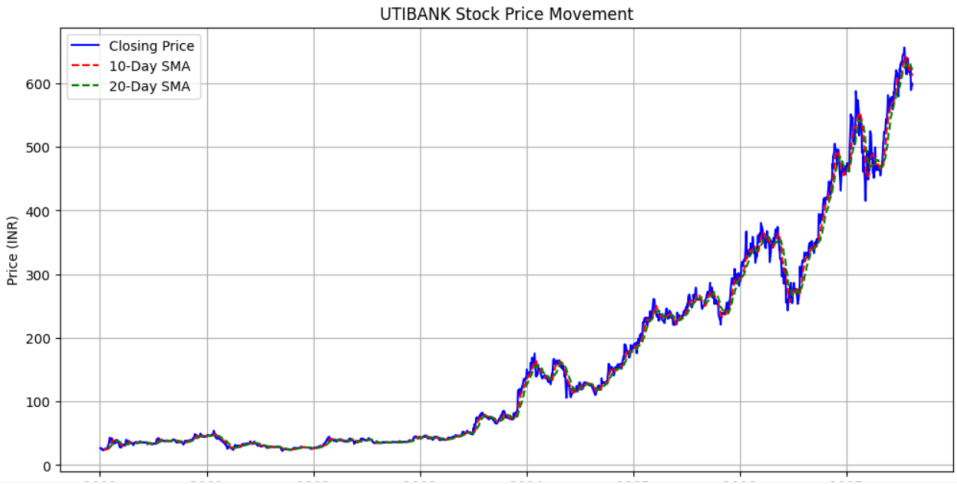
```
# Import necessary libraries
import pandas as pd  # Data handling and processing
import numpy as np  # Numerical operations
import matplotlib.pyplot as plt  # Plotting graphs
import seaborn as sns  # Statistical data visualization
import mplfinance as mpf  # Candlestick charting

# Load stock data from CSV file
stock_data = pd.read_csv("stock_data.csv", parse_dates=["Date"], index_col="Date")  # Load
```

and parse dates

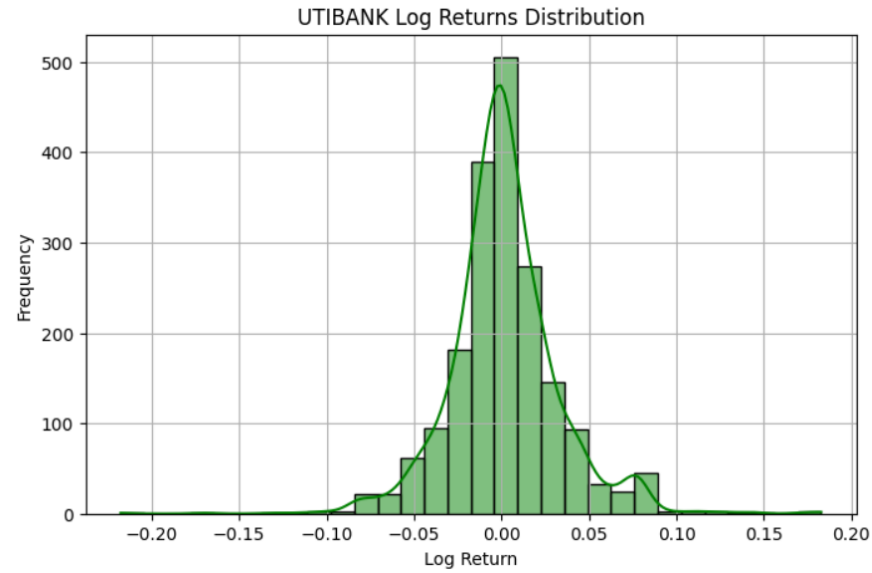
```
# Filter for a specific stock symbol (e.g., "UTIBANK")
symbol = "UTIBANK"
stock_data = stock_data[stock_data["Symbol"] == symbol]  # Filter the dataset for the chosen symbol

# --- STOCK PRICE MOVEMENT ---
plt.figure(figsize=(12, 6))  # Set figure size for better readability
plt.plot(stock_data["Close"], label="Closing Price", color="blue")  # Plot closing price in blue
plt.plot(stock_data["SMA_10"], label="10-Day SMA", color="red", linestyle="dashed")  # 10-day SMA in red dashed
plt.plot(stock_data["SMA_20"], label="20-Day SMA", color="green", linestyle="dashed")  # 20-day SMA in green dashed
plt.title(f"{symbol} Stock Price Movement")  # Title of the plot
plt.xlabel("Date")  # Label for x-axis
plt.ylabel("Price (INR)")  # Label for y-axis
plt.legend()  # Display legend
plt.grid()  # Show grid for better visualization
plt.show()  # Display the plot

# --- DAILY RETURNS ---
plt.figure(figsize=(12, 6))  # Set figure size
plt.plot(stock_data["Daily_Return"], label="Daily Return", color="purple")  # Plot daily returns in purple
plt.axhline(0, linestyle="--", color="black", linewidth=0.8)  # Zero-line reference for better comparison
plt.title(f"{symbol} Daily Returns")  # Title of the plot
plt.xlabel("Date")  # Label for x-axis
plt.ylabel("Return")  # Label for y-axis
plt.legend()  # Display legend
plt.grid()  # Show grid for better visualization
plt.show()  # Display the plot

# --- LOG RETURNS DISTRIBUTION ---
plt.figure(figsize=(8, 5))  # Set figure size for the histogram
sns.histplot(stock_data["Log_Return"].dropna(), bins=30, kde=True, color="green")  # Plot log return distribution in green
plt.title(f"{symbol} Log Returns Distribution")  # Title of the plot
plt.xlabel("Log Return")  # Label for x-axis
plt.ylabel("Frequency")  # Label for y-axis
plt.grid()  # Show grid for better visualization
plt.show()  # Display the plot
```

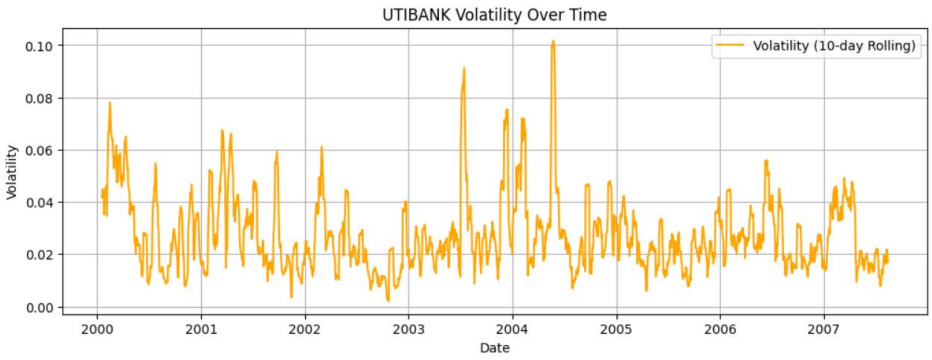
```
# --- VOLATILITY OVER TIME ---
plt.figure(figsize=(12, 4))  # Set figure size
plt.plot(stock_data["Volatility"], label="Volatility (10-day Rolling)", color="orange")  # Plot volatility in orange
plt.title(f"{symbol} Volatility Over Time")  # Title of the plot
plt.xlabel("Date")  # Label for x-axis
plt.ylabel("Volatility")  # Label for y-axis
plt.legend()  # Display legend
plt.grid()  # Show grid for better visualization
plt.show()  # Display the plot
# --- CANDLESTICK CHART ---
mpf.plot(stock_data[-50:], type="candle", style="charles",  # Plot candlestick chart for last 50 days
        title=f"{symbol} Candlestick Chart (Last 50 Days)",
        ylabel="Price (INR)", volume=True)  # Volume included in the chart
# Print last 10 rows of key indicators for quick inspection
print(stock_data[["Close", "SMA_10", "SMA_20", "Daily_Return", "Log_Return", "Volatility", "Signal"]].tail(10))
```

4. Output



UTIBANK Candlestick Chart (Last 50 Days)





5. Conclusion

This report presents a data-driven approach to analyzing short-term stock price movements. Key insights include:

- Moving averages help in trend identification.
- Daily & log returns measure price fluctuations.
- Volatility analysis highlights market uncertainty.
- Candlestick charts provide a visual representation of price action.

Such analyses assist traders and investors in making informed decisions. Further improvements can include machine learning models for predictive analysis.

6. References

Python documentation: https://docs.python.org/3/
Pandas: https://pandas.pydata.org/
Matplotlib: https://matplotlib.org/

Seaborn: https://seaborn.pydata.org/
mplfinance: https://github.com/matplotlib/mplfinance