Stock Market Data Analysis and Prediction

Data Preparation:

For this study, historical stock price data was collected from **Yahoo Finance** for **five Indian stocks,** namely, Trent Ltd, Varun Beverages, HAL, Brigade enterprises and Zomato Ltd over a period spanning from **March 1, 2024, to March 1, 2025**. The original dataset included key financial indicators such as:

- **Date** – The trading date for each recorded entry.
- **Open Price (O)** – The price at which the stock started trading on a given day.
- **High Price (H)** – The highest price the stock reached during the trading session.
- **Low Price (L)** – The lowest price the stock dropped to during the session.
- **Close Price (C)** – The final price at which the stock traded before the market closed.
- **Volume (V)** – The total number of shares traded during the day.

Exploratory Data Analysis

Firstly, the 1-year graph was plotted and over the year if the return is in negative then the graph is plotted using a red line whereas if the return is positive then it is plotted using a green line.

1. Brigade Ent.

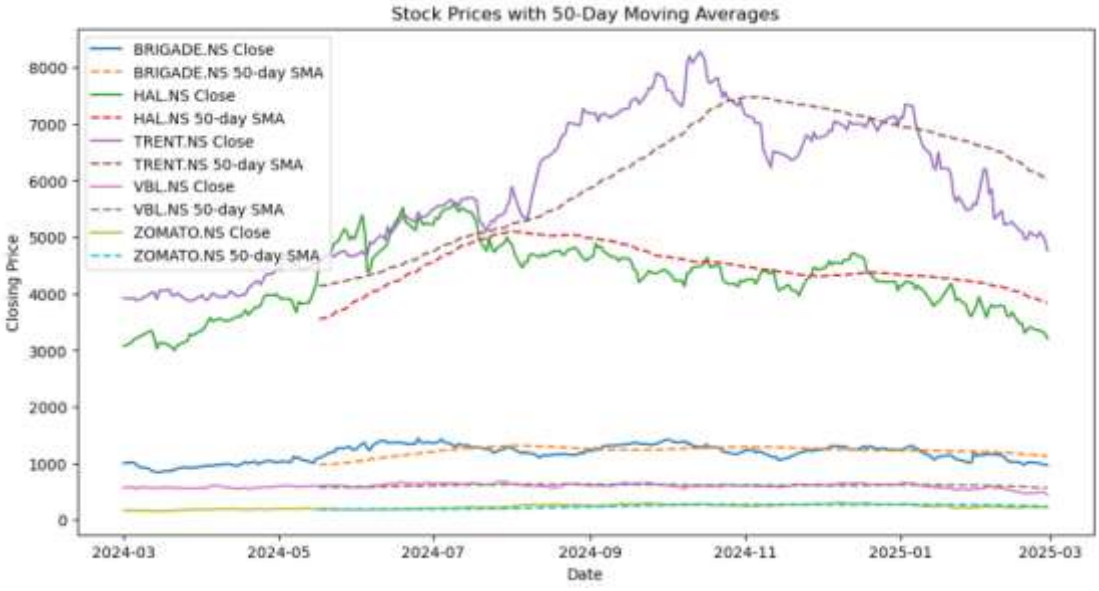
2. HAL Ltd

3. Trent Ltd.

4. Varun Beverages

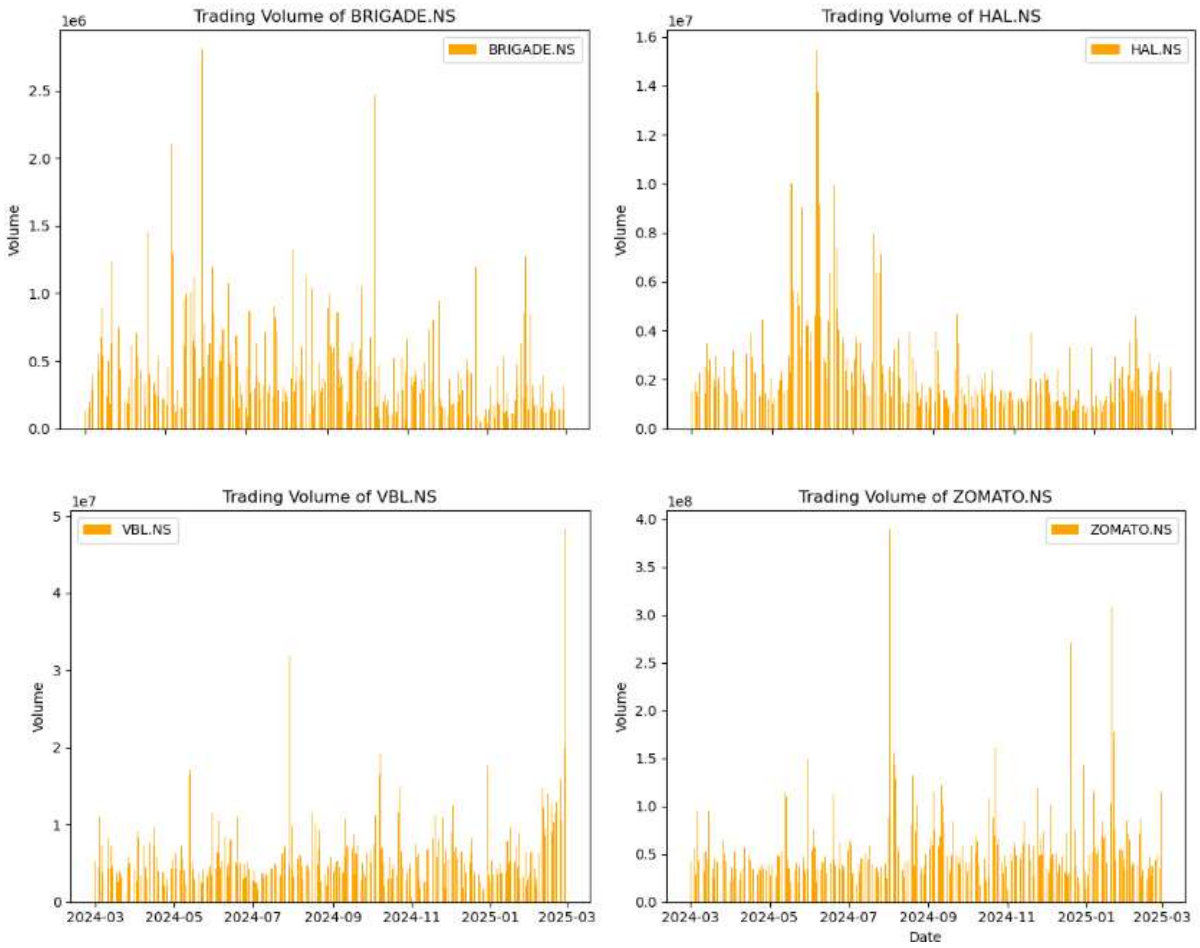
5. Zomato Ltd

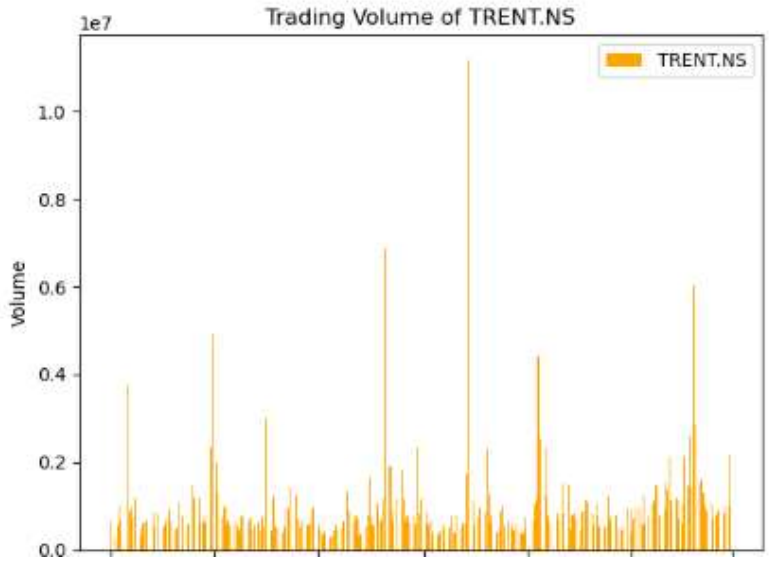
- After this for EDA, a graph with all the stocks was plotted along with their 50 DMA which is represented with dotted lines.



- To further analyze trading activity, **subplots were used to visualize the trading volumes** of all five stocks over the selected period. This graphical representation provided a comparative view of how trading volume fluctuated across different stocks over time.

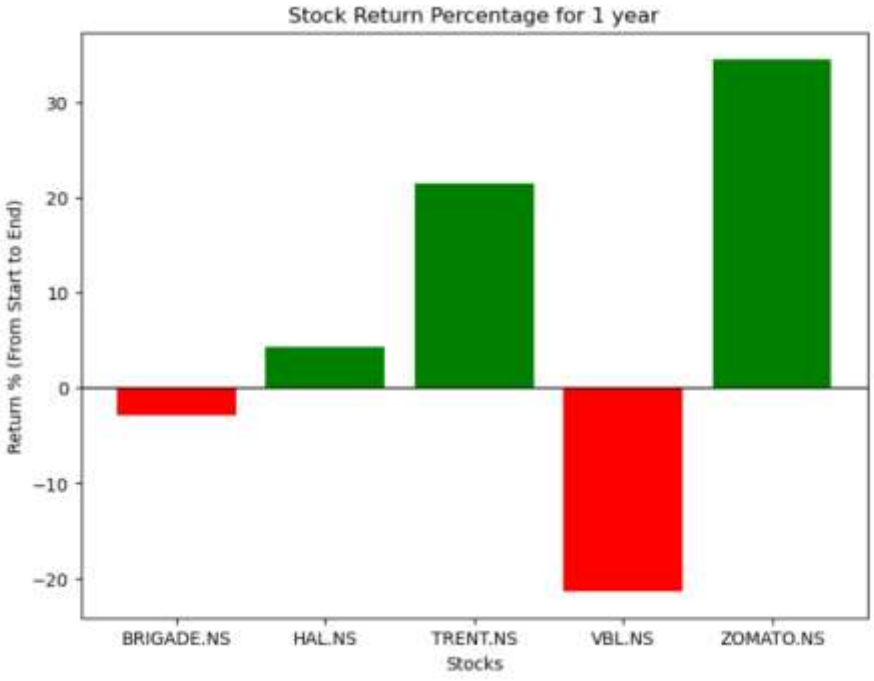
  From the plotted data, it was observed that **all stocks exhibited sharp spikes in trading volume around the same periods**. These synchronized surges indicate the presence of **market-wide trading patterns**, likely influenced by significant economic events, earnings announcements, macroeconomic developments, or broader investor sentiment shifts. One such instance could be the election results during June.





- To assess the overall performance of the selected stocks, a **simple comparison graph** was plotted, showcasing the **one-year return** of each stock. This visualization provides a clear understanding of how the five stocks, belonging to different sectors, performed over the year.

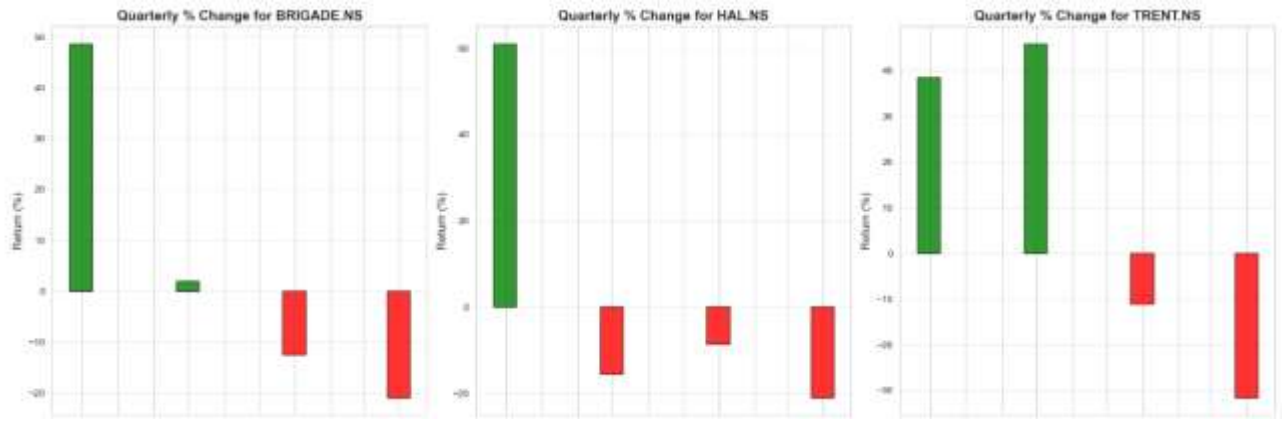
  By comparing the percentage returns, investors can gauge sector-wise performance and identify which industries experienced growth or decline. This analysis also helps in understanding market trends, sector resilience, and potential investment opportunities based on historical returns.

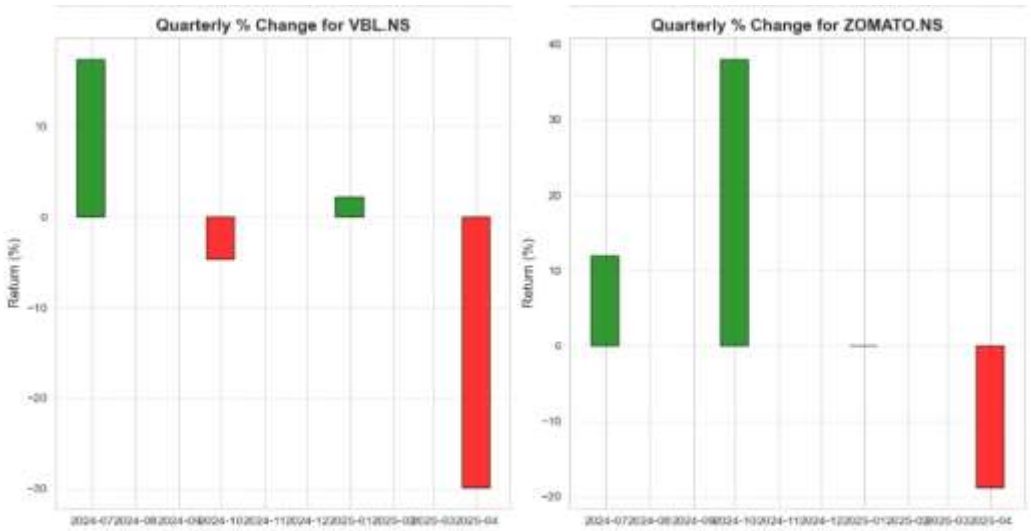


- To analyze **seasonal trends in stock performance**, **quarterly return graphs** were plotted. These graphs depict the return of each stock during specific three-month periods, helping to identify variations in performance over time.

  It is important to note that these quarters are **not based on the standard financial quarters** (April-June, July-September, etc.). Instead, they are derived directly from the dataset by dividing the **one-year period into four equal quarters**.

  By examining these quarterly trends, we can gain insights into **seasonal influences, market cycles, and sector-specific fluctuations** that impact stock returns. This helps in understanding whether certain periods tend to be more favorable for specific stocks or sectors, aiding in better investment decision-making.





Feature Engineering:

After conducting Exploratory Data Analysis (EDA), feature engineering was performed to enhance the dataset for modeling. This step is crucial as it helps in extracting meaningful patterns from raw data, making it more suitable for predictive analysis. The features were specifically designed to improve the performance of the two chosen models: Autoregressive Integrated Moving Average (ARIMA) and Gradient Boosting Regression (GBR).

- One of the essential steps in **feature engineering** for time-series forecasting is the calculation of **lag features**. A **lag feature** represents the value of a variable from a previous time step, allowing models to learn from historical patterns and dependencies. By incorporating lagged values, we can capture trends, momentum, and autocorrelations in stock prices.

  For this study, **lag features were created for 1-day, 3-day, and 7-day intervals**:

  1-Day Lag: Represents the stock price or volume from the previous trading day, helping capture immediate short-term changes.

  3-Day Lag: Incorporates data from three days prior, allowing the model to recognize patterns over a slightly extended period.

  7-Day Lag: Uses data from a week before, helping in identifying **weekly trends** and smoothing out short-term fluctuations.

- Another crucial feature engineered for the dataset was the **Simple Moving Average (SMA)**, a widely used indicator in stock price analysis. SMA helps in smoothing out short-term price fluctuations and identifying underlying trends over different time horizons. It is calculated by taking the average of stock prices over a specific period. The formula for SMA is:

  For this study, SMA was computed for **7-day, 30-day, and 50-day periods**:

  7-Day SMA: Captures short-term price trends and helps in detecting recent momentum shifts.

30-Day SMA: Represents medium-term trends, filtering out short-term noise while still being responsive to market movements.

50-Day SMA: Provides a long-term perspective, helping to identify sustained trends and potential reversals in stock prices.

- To capture the **daily price movement**, a new feature called **One-Day Change** was added to the dataset. This feature represents the difference between the **current day's closing price** and the **previous day's closing price**, helping to quantify the stock's daily fluctuations.

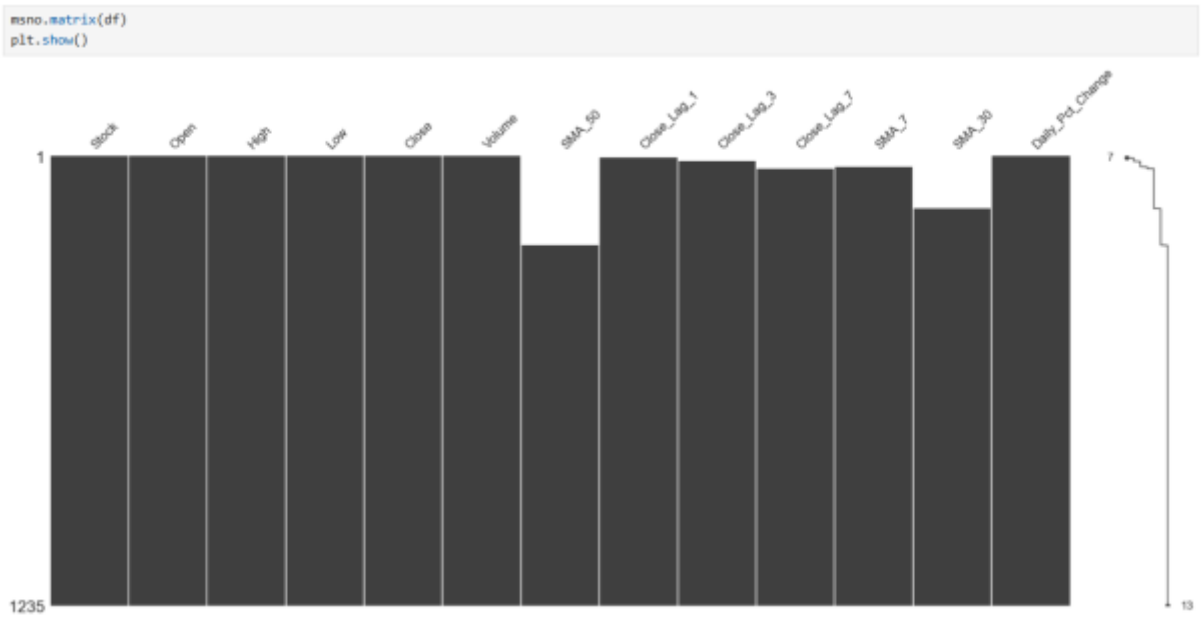
  This feature is useful for identifying **short-term volatility** and assessing the stock's momentum on a daily basis. It provides valuable insights for both **ARIMA**, which relies on past values to predict trends, and **Gradient Boosting Regression (GBR)**, which benefits from additional indicators reflecting price movement patterns.

Cleaning the dataset after Feature Engineering

After adding the new features—**lag values, moving averages, and one-day change**—the dataset contained some **NaN (Not a Number) values**. These missing values primarily appeared due to the nature of the calculations.

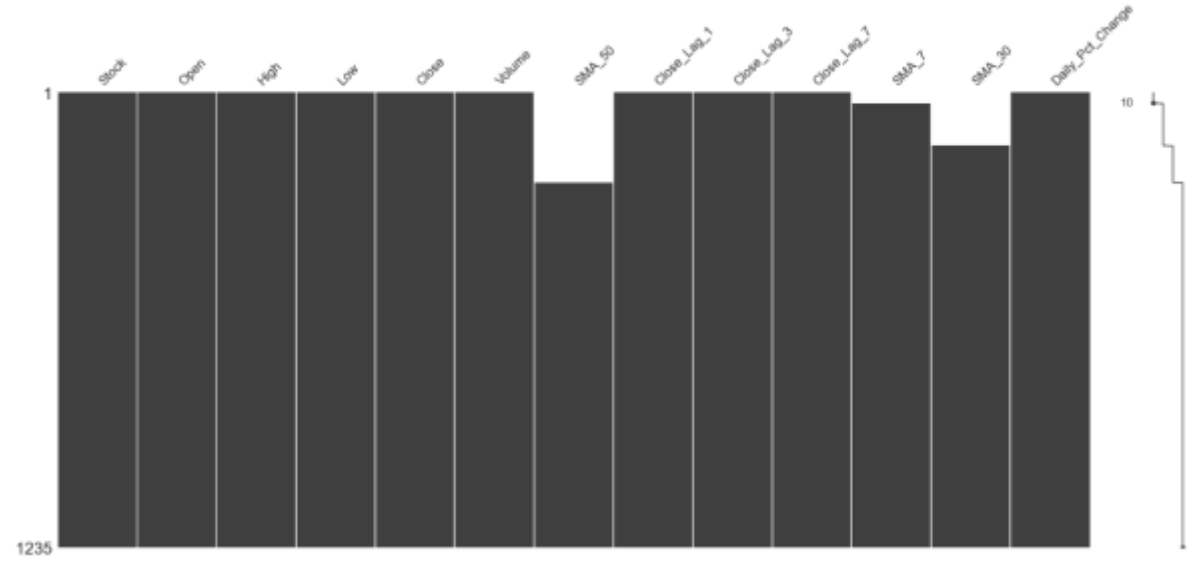
```
df.isnull().sum()
```

```
Stock               0
Open                0
High                0
Low                 0
Close               0
Volume              0
SMA_50            245
Close_Lag_1         5
Close_Lag_3        15
Close_Lag_7        35
SMA_7              30
SMA_30            145
Daily_Pct_Change    0
dtype: int64
```

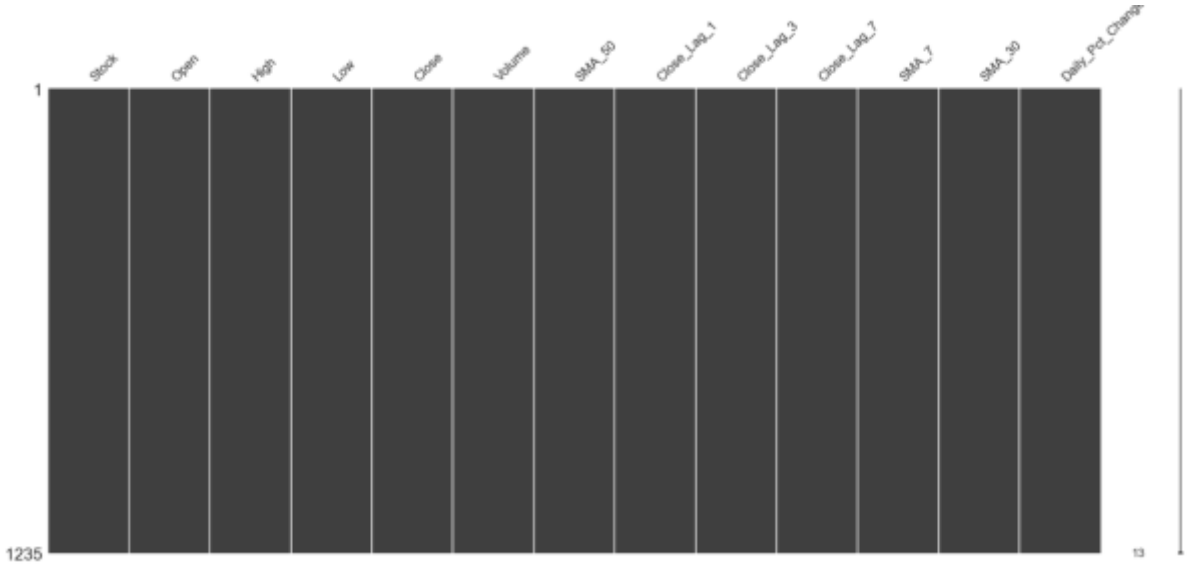


To ensure a clean and usable dataset for modeling, appropriate handling techniques were applied, such as forward-filling, backward-filling.

Lag Features: Since lag features represent past values, **forward-filling** was used, where each missing value was replaced with the most recent available value. This approach maintains the natural flow of stock price movements without introducing artificial trends.



Simple Moving Average (SMA): Since SMA relies on past values, missing values at the beginning of the dataset were filled using the **average of the available SMA values** for that stock. This ensures that early missing values do not distort trend analysis.




```
: df.isnull().sum()
```

```
: Stock              0
  Open               0
  High               0
  Low                0
  Close              0
  Volume             0
  SMA_50             0
  Close_Lag_1        0
  Close_Lag_3        0
  Close_Lag_7        0
  SMA_7              0
  SMA_30             0
  Daily_Pct_Change   0
  dtype: int64
```

Methodology:

After completing **data preparation**, including feature engineering and handling missing values, the next step was to build predictive models for stock price forecasting. Two different modeling approaches were implemented:

1. **Autoregressive Integrated Moving Average (ARIMA):** A statistical time-series model that captures trends, seasonality, and autocorrelations in stock prices. It was trained using historical price data to generate future forecasts.
2. **Gradient Boosting Regression (GBR):** A machine learning model that identifies complex patterns and relationships in stock price movements. It was trained on both raw and engineered features to improve prediction accuracy.

- ARIMA Modeling:

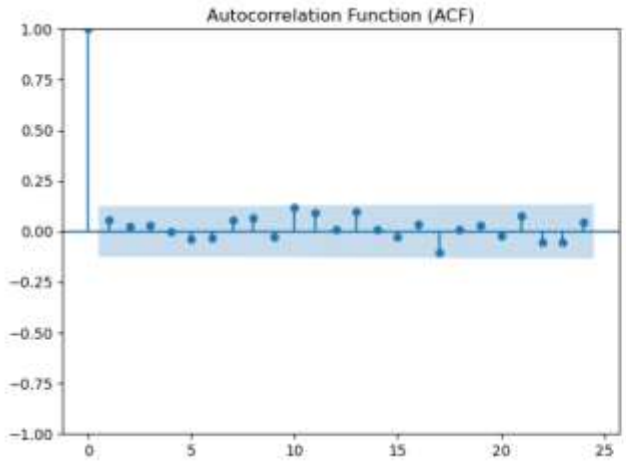
To apply **ARIMA** for stock price forecasting, the modeling process was specifically performed on **Trent Stock**. Essential Python libraries such as pandas, numpy, statsmodels.api, and matplotlib were imported to facilitate time-series analysis and model implementation. The dataset was structured for time-series modeling by setting the **Date** column as the index. The data was also sorted in ascending order to maintain the correct chronological sequence, ensuring the ARIMA model could learn from past trends effectively.

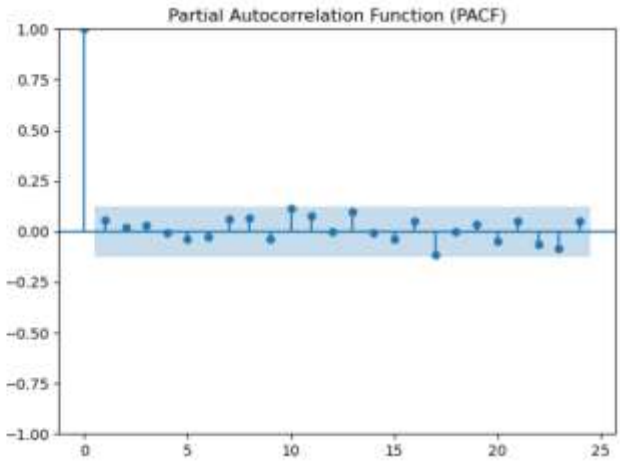
Before applying ARIMA, it was essential to check whether the time series was **stationary** since ARIMA models work best with stationary data. If the **p-value** from the test is below a chosen significance level (commonly **0.05**), the null hypothesis is rejected, indicating that the time series is stationary.

```
ADF Test for Close Prices:
ADF Statistic: -1.383568128604102
p-value: 0.5901279221048354
The series is NOT stationary
```

Since, the series was not stationary, first order differencing was applied. Then to calculate p and q the following methods were used:

1. **Plotting ACF (Autocorrelation Function)** – This helped determine the value of **q** by analyzing the number of significant lags in the autocorrelation plot.
2. **Plotting PACF (Partial Autocorrelation Function)** – This helped determine the value of **p** by examining how past lags influenced the current value while removing the effect of intermediate lags.
3. **Using Auto-ARIMA** – To automate parameter selection, the **Auto-ARIMA** function from the pmdarima library was used. This function tested multiple combinations of **p, d, and q** values and selected the best-fitting model based on statistical criteria such as **AIC (Akaike Information Criterion)** and **BIC (Bayesian Information Criterion)**.





```
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.95 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=3276.908, Time=0.03 sec
 ARIMA(1,2,0)(0,0,0)[0] intercept   : AIC=3214.462, Time=0.06 sec
 ARIMA(0,2,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.17 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=3274.914, Time=0.02 sec
 ARIMA(2,2,0)(0,0,0)[0] intercept   : AIC=3188.557, Time=0.07 sec
 ARIMA(3,2,0)(0,0,0)[0] intercept   : AIC=3177.957, Time=0.19 sec
 ARIMA(4,2,0)(0,0,0)[0] intercept   : AIC=3174.283, Time=0.21 sec
 ARIMA(5,2,0)(0,0,0)[0] intercept   : AIC=3171.264, Time=0.46 sec
 ARIMA(5,2,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.92 sec
 ARIMA(4,2,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.87 sec
 ARIMA(5,2,0)(0,0,0)[0]             : AIC=3169.284, Time=0.33 sec
 ARIMA(4,2,0)(0,0,0)[0]             : AIC=3172.302, Time=0.11 sec
 ARIMA(5,2,1)(0,0,0)[0]             : AIC=3133.088, Time=0.76 sec
 ARIMA(4,2,1)(0,0,0)[0]             : AIC=3132.092, Time=0.55 sec
 ARIMA(3,2,1)(0,0,0)[0]             : AIC=3130.226, Time=0.60 sec
 ARIMA(2,2,1)(0,0,0)[0]             : AIC=3128.237, Time=0.33 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=3126.239, Time=0.21 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=3124.483, Time=0.11 sec
 ARIMA(0,2,2)(0,0,0)[0]             : AIC=3126.238, Time=0.26 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=3212.473, Time=0.04 sec
 ARIMA(1,2,2)(0,0,0)[0]             : AIC=3128.234, Time=0.47 sec

Best model:  ARIMA(0,2,1)(0,0,0)[0]
Total fit time: 7.781 seconds

Optimal ARIMA Order: (0, 2, 1)
```

Taking reference to the above methods the values of p, d and q was decided as 0, 2, 1.

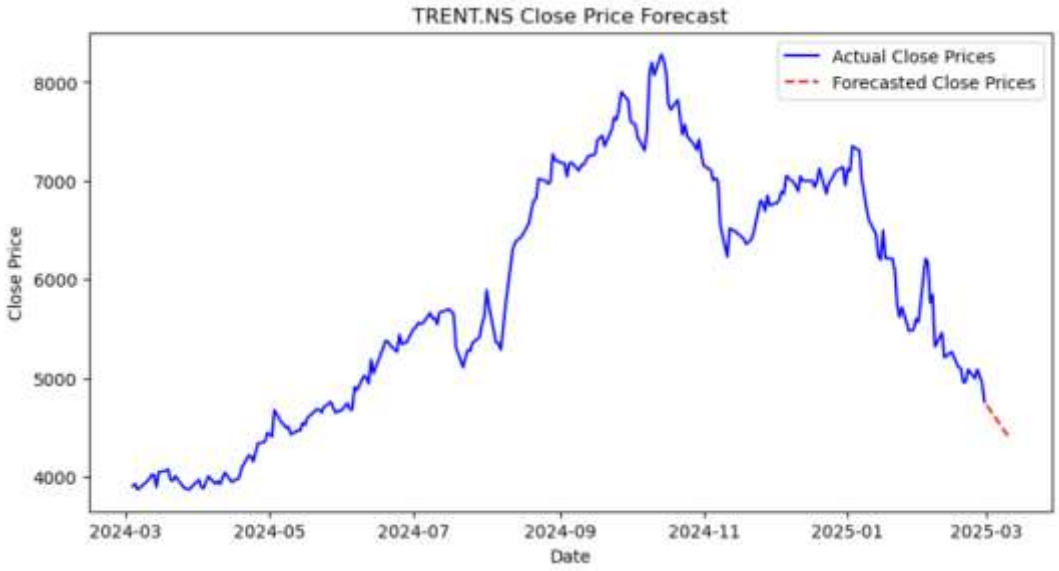
```
                               SARIMAX Results
==============================================================================
Dep. Variable:                  Close   No. Observations:                  246
Model:                 ARIMA(0, 2, 1)   Log Likelihood               -1560.242
Date:                Sat, 08 Mar 2025   AIC                           3124.483
Time:                        15:15:07   BIC                           3131.478
Sample:                             0   HQIC                          3127.300
                                  - 246
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.9734      0.016    -62.211      0.000      -1.004      -0.943
sigma2      2.068e+04   1277.650     16.184      0.000    1.82e+04    2.32e+04
===================================================================================
Ljung-Box (L1) (Q):                   0.13   Jarque-Bera (JB):                55.85
Prob(Q):                              0.71   Prob(JB):                         0.00
Heteroskedasticity (H):               3.87   Skew:                             0.14
Prob(H) (two-sided):                  0.00   Kurtosis:                         5.33
===================================================================================
```

To visually assess the model's performance, a **line graph** was plotted comparing the **actual stock prices** with the **predicted values**. This helped in understanding how well the ARIMA model captured historical trends and patterns.

Then the prices for the next 10 days was forecasted and the data was as follows:

```
Forecasted Prices:
            Forecasted_Close
Date
2025-03-01       4725.304378
2025-03-02       4690.608755
2025-03-03       4655.913133
2025-03-04       4621.217511
2025-03-05       4586.521888
2025-03-06       4551.826266
2025-03-07       4517.130644
2025-03-08       4482.435021
2025-03-09       4447.739399
2025-03-10       4413.043776
```

Once the ARIMA model was trained and fitted, it was used to **forecast future stock prices**. The forecasted values were then compared with the actual stock prices to evaluate the model's predictive power.



Upon plotting the **forecasted prices**, it was observed that the predicted values formed a **straight line**, indicating a downward trend. However, this behavior did not align with the actual stock movements.

```
RMSE: 437.59
MAE: 426.71
MAPE: 8.46%
```

This issue arises because **ARIMA primarily relies on past values and assumes linear patterns**, making it less effective in capturing complex price fluctuations in stock markets. As a result, the model **fails to adapt to sudden market changes**, leading to inaccurate predictions.

- # Gradient Boosting

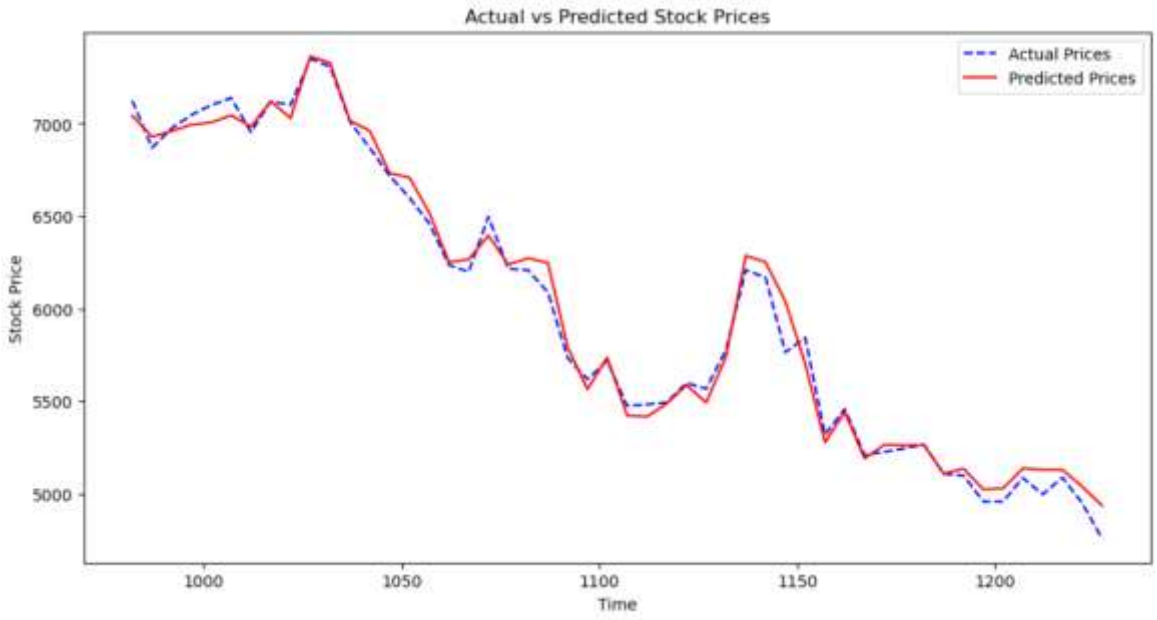
After analyzing stock price movements using ARIMA, the next approach explored was Gradient Boosting Regression (GBR). This machine learning technique is well-suited for capturing non-linear patterns and complex relationships in stock prices. For this study, GBR was applied on the same Trent Stock dataset used for ARIMA modeling, ensuring a fair comparison between the two approaches.

Before training the **Gradient Boosting Regression (GBR) model**, the dataset was split into **training and testing sets** to evaluate its predictive performance. The **following features** were selected based on their relevance in capturing stock price trends and patterns:

MA_50, SMA_7, SMA_30 – Moving Averages that smooth price fluctuations and capture trends.

Close_Lag_1, Close_Lag_3, Close_Lag_7 – Lag features that provide past stock price values to help the model understand price movement patterns.

Daily_Pct_Change – The percentage change in stock price, capturing day-to-day volatility.




```
MAE: 60.00082500992548, RMSE: 78.93079789962628
```

A hyperparameter tuning process using **GridSearchCV** optimizes key parameters (n_estimators, learning_rate, max_depth) to enhance model performance. The best-selected model is then trained and evaluated using **Mean Absolute Error (MAE)** and **Root Mean Squared Error (RMSE)**. Finally, a **visualization** is generated to compare actual vs predicted stock prices, providing insights into model accuracy and forecasting effectiveness.

- ## ARIMA vs Gradient Boosting

ARIMA, a linear time series model, struggles with non-linearity and external factors, leading to higher errors. It assumes stationarity and does not adapt well to sudden market fluctuations. In contrast, Gradient Boosting leverages multiple features like moving averages and trading volume, effectively capturing complex market patterns, leading to better accuracy. The models were evaluated using **RMSE**, where ARIMA resulted in **437**, while Gradient Boosting achieved **78**, indicating a significant performance difference.

Each model has its own strengths and limitations. **ARIMA** is advantageous due to its **interpretability**, as it decomposes time series data into distinct components such as **trend, seasonality, and residuals**. However, its reliance on **stationarity assumptions** and inability to **capture external market factors** limits its effectiveness in financial forecasting. **Gradient Boosting**, on the other hand, excels in identifying **non-linear dependencies** and utilizing **multiple features**, making it well-suited for stock price prediction. However, it is computationally **more expensive** and requires **extensive hyperparameter tuning** to avoid overfitting.

Based on the evaluation results, **Gradient Boosting proved to be a significantly more effective model** for predicting stock prices compared to ARIMA. Its ability to incorporate **multiple features, learn complex patterns, and handle market volatility** led to a much lower RMSE, indicating **higher accuracy and reliability**. While ARIMA remains useful for short-term forecasting in stationary time series, it is **less effective for dynamic and unpredictable financial markets**. Hence, **Gradient Boosting is recommended for stock price forecasting** due to its superior performance in capturing the complexities of financial data.