Question 4

Stock Price Prediction

Team Name: **DataMavericks**

Table of Contents

1 INTRODUCTION

Stock price prediction is a critical task in financial markets, helping investors and traders make informed decisions. This report presents an exploratory data analysis (EDA) of historical stock prices, aiming to uncover key patterns, trends, and insights that will guide the development of a predictive model. By analyzing stock price movements, seasonal variations, and anomalies, we can better understand market behavior and engineer meaningful features for improved forecasting accuracy.

In this report, we begin by assessing the dataset's structure, handling missing values, and ensuring data quality. We then analyze stock price trends, identify seasonality, and detect anomalies that may indicate market shocks or structural changes. Correlation analysis is performed to understand relationships between features, ensuring optimal feature selection for model development.

The insights gained from this analysis will serve as the foundation for constructing a robust machine learning model capable of predicting stock prices five trading days into the future. This report also outlines key preprocessing steps and feature engineering strategies that will enhance predictive performance while maintaining interpretability for real-world trading applications.

2 PROBLEM STATEMENT

The aim of this project is to build an **LSTM-based stock price prediction model** that forecasts a stock's **closing price 5 trading days into the future**. The model will leverage **historical stock market data** and will be optimized for both **predictive accuracy and trading insights**.

Customer Segments (Stock Market Participants):

- **Short-Term Traders** – Need accurate near-future price predictions for quick trades.
- **Long-Term Investors** – Require insights into trends, seasonality, and anomalies for better decision-making.
- **Market Analysts** – Seek models that explain stock price movements and evaluate risk factors.

3 METHODOLOGY

3.1 Libraries used

1. **Pandas (pd)**
   - **Purpose:** Data manipulation and analysis.
   - **Use:** To read and handle the dataset (question4-stock-data.csv), perform exploratory data analysis, manage time-series data, and handle missing values.

2. **NumPy (np)**
   - **Purpose:** Numerical computations.
   - **Use:** To perform mathematical operations on stock price data, handle arrays, and efficiently manage large datasets.

3. **Matplotlib (plt) and Seaborn (sns)**
   - **Purpose:** Data visualization.
   - **Use:** To create plots and visualizations that help in understanding stock trends, anomalies, and correlations between features.

4. **Statsmodels (sm)**
   - **Purpose:** Time series analysis.
   - **Use:** To perform **seasonal decomposition** and analyze stock market trends over time.

5. **SciPy (zscore)**
   - **Purpose:** Statistical analysis.
   - **Use:** To detect anomalies in stock prices using the **Z-score method**.

6. **Missingno (msno)**
   - **Purpose:** Handling missing data.
   - **Use:** To visualize and identify missing values in the dataset before training the model.

7. **MinMaxScaler (from sklearn.preprocessing)**
   - **Purpose:** Data scaling.
   - **Use:** To scale stock prices between 0 and 1, making the data suitable for **LSTM training**.

8. **RandomForestRegressor (from sklearn.ensemble)**
   - **Purpose:** Feature importance analysis.
   - **Use:** To evaluate and rank the importance of features before selecting them for the LSTM model.

9. **PCA (Principal Component Analysis)**
    - **Purpose:** Dimensionality reduction.
    - **Use:** To reduce redundant features and optimize the dataset for better model performance.

10. **Keras and TensorFlow**
    - **Purpose:** Deep learning framework.
    - **Use:** To build and train the **LSTM model** for stock price prediction.

11. **scikit-learn (train_test_split, Mean Squared Error)**
    - **Purpose:** Model evaluation.
    - **Use:** To split the dataset into training and testing sets, and evaluate the model's **performance using RMSE and directional accuracy**.

12. **AutoCorrelation Function (ACF) from Statsmodels**
    - **Purpose:** Time series dependency analysis.
    - **Use:** To check the dependency between previous and future stock prices.

4 EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) is a crucial step in understanding the dataset used for **stock price prediction**. It involves examining the **structure, quality, and key patterns** within the data. Through EDA, we identify **trends, seasonality, missing values, and anomalies**, which help in making informed decisions for feature engineering and model selection. By visualizing stock price movements and analyzing statistical summaries, we gain insights into market behavior, volatility, and potential factors influencing stock performance.

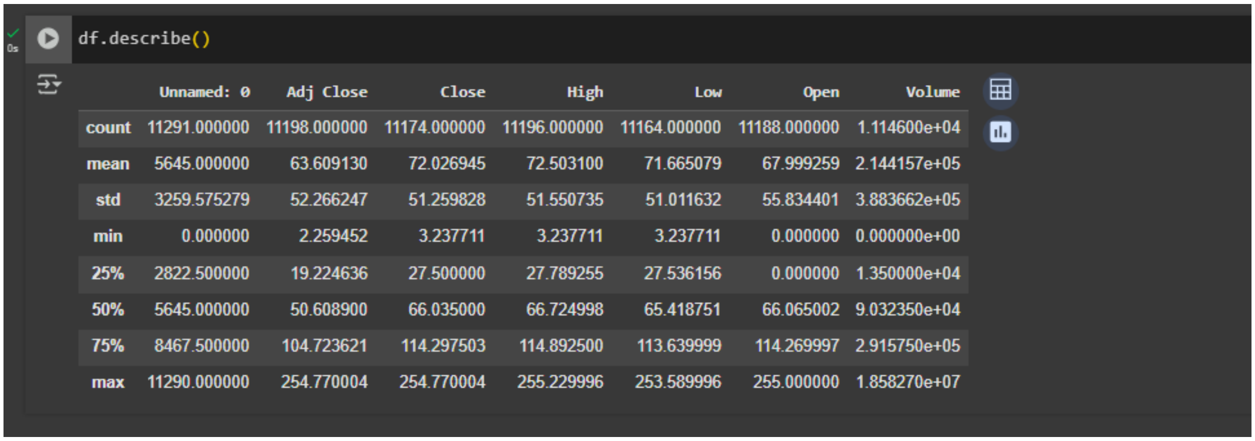
4.1 Visualizations of key patterns and relationships in the data.

4.1.1 Data Description

The dataset contains historical stock price data, including **adjusted closing price, closing price, high, low, open, and trading volume**. This data will be used to develop a predictive model for stock price forecasting.

Dataset Summary

- **Total Rows:** 11,291
- **Total Columns:** 8
- **Missing Values:** Present in multiple columns
- **Date Range:** 3/17/1980 to 12/27/2024



```
df.describe()
```

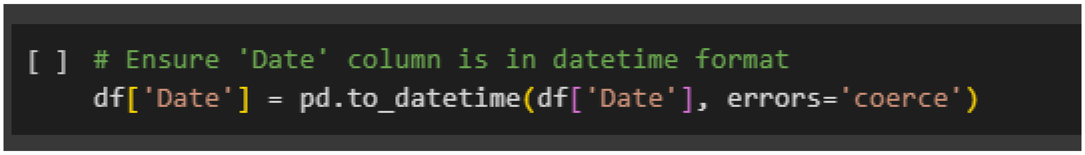
	Unnamed: 0	Adj Close	Close	High	Low	Open	Volume
count	11291.000000	11198.000000	11174.000000	11196.000000	11164.000000	11188.000000	1.114600e+04
mean	5645.000000	63.609130	72.026945	72.503100	71.665079	67.999259	2.144157e+05
std	3259.575279	52.266247	51.259828	51.550735	51.011632	55.834401	3.883662e+05
min	0.000000	2.259452	3.237711	3.237711	3.237711	0.000000	0.000000e+00
25%	2822.500000	19.224636	27.500000	27.789255	27.536156	0.000000	1.350000e+04
50%	5645.000000	50.608900	66.035000	66.724998	65.418751	66.065002	9.032350e+04
75%	8467.500000	104.723621	114.297503	114.892500	113.639999	114.269997	2.915750e+05
max	11290.000000	254.770004	254.770004	255.229996	253.589996	255.000000	1.858270e+07

Figure 1: stock data description

Column Name	Data Type	Description
Unnamed: 0	int64	Index column (not needed for analysis)
Date	object	Stock trading date (needs conversion to datetime format)
Adj Close	float64	Adjusted closing price, adjusted for dividends/splits
Close	float64	Closing price of the stock on a given day
High	float64	Highest price of the stock for the day
Low	float64	Lowest price of the stock for the day
Open	float64	Opening price of the stock for the day
Volume	float64	Number of shares traded during the day

4.1.2 Data Cleaning & Preparation

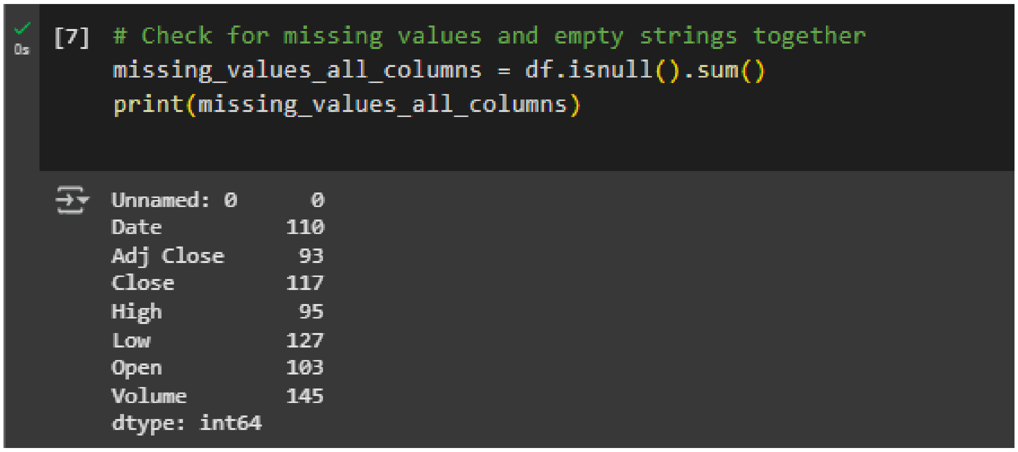
1. **'Date' column is in datetime format**



```
[ ] # Ensure 'Date' column is in datetime format
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
```

- Enables Time-Based Indexing & Filtering
- Ensures Correct Sorting by Date
- Allows Date Arithmetic & Feature Engineering
- Handles Invalid Date Formats (errors='coerce')

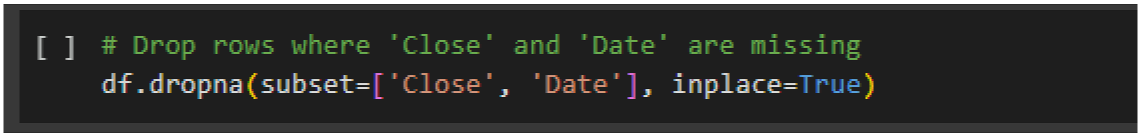
2. Check the missing values



```
# Check for missing values and empty strings together
missing_values_all_columns = df.isnull().sum()
print(missing_values_all_columns)
```

```
Unnamed: 0      0
Date          110
Adj Close      93
Close         117
High           95
Low           127
Open          103
Volume        145
dtype: int64
```

3. Drop rows where 'Close' and 'Date' are missing

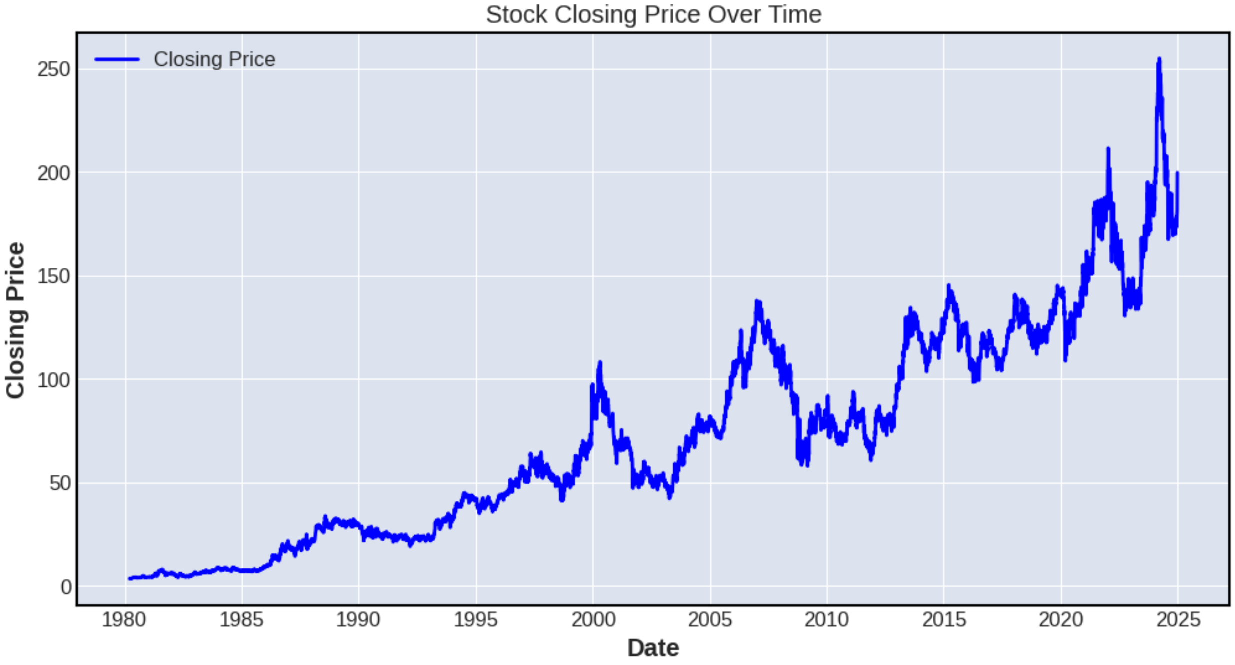


```
# Drop rows where 'Close' and 'Date' are missing
df.dropna(subset=['Close', 'Date'], inplace=True)
```

This line removes rows where either the **'Close'** or **'Date'** column has missing (NaN) values. Since this is a **time-series problem**, both columns are critical for accurate analysis and predictions.

- 'Date' is the Primary Time-Series Index
- 'Close' is the Target Variable
- Prevents Incorrect Analysis & Bias
- Most Models Can't Handle NaN Values

4.1.3 Stock Price Trend Analysis



The overall trend indicates a long-term upward movement in stock prices, suggesting positive market growth over the years.

Periods of high volatility can be observed, especially around 2000, 2008, and 2020, which may correspond to financial crises or economic downturns.

Key Observations

Early Stability (1980-1995)

- Stock prices remained relatively low and stable with minor fluctuations.
- No significant upward movement observed.

Growth Phase (1995-2000)

- Noticeable increase in stock prices.
- Possible factors: technological advancements, strong economic growth.

Sharp Decline & Recovery (2000-2010)

- A major drop around 2000, likely due to the Dot-com bubble burst.
- Another drop in 2008, which may be linked to the Global Financial Crisis.
- Gradual recovery post-2010.

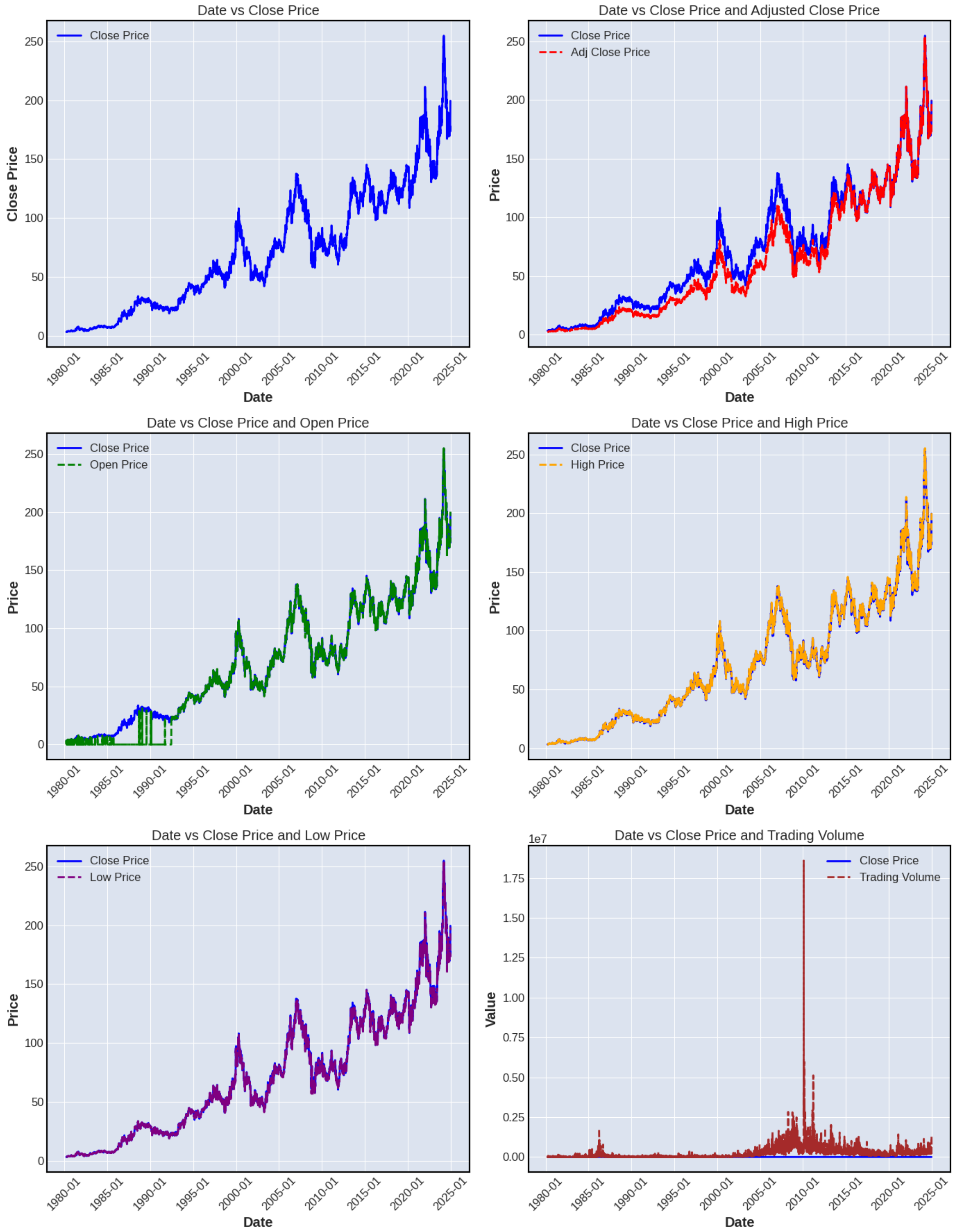
Post-2010 Surge & High Volatility (2010-2025)

- Significant growth trend with periodic corrections.
- High fluctuations, particularly around 2020, possibly due to the COVID-19 pandemic impact on the stock market.
- The highest peak occurred in recent years, followed by a sharp drop, suggesting potential correction phases.

Potential Factors Influencing Trends

- Economic Cycles: Stock prices reflect economic conditions, recessions, and booms.
- Market Sentiment & External Events: Events like financial crises, pandemics, and policy changes affect market movements.
- Technological & Industry Growth: Market disruptions from innovation and industry growth trends.

4.1.4 Observations from the Multi-Graph Visualization



This set of six subplots provides insights into the stock price trends and their relationships with key financial indicators.

1. Date vs. Close Price (Top-Left)

- The closing price has an overall upward trend from 1980 to 2025.
- Major volatility spikes are noticeable around 2000, 2008, and 2020, likely due to economic crises and market corrections.
- The trend appears exponential, indicating growing investor confidence over time.

2. Date vs. Close Price & Adjusted Close Price (Top-Right)

- The adjusted close price (red dashed line) accounts for dividends and stock splits, often lower than the raw closing price.
- The two lines move in parallel, suggesting consistent dividend payments and stock adjustments without drastic fluctuations.
- Divergence between the two lines increases over time, implying more stock splits or corporate actions.

3. Date vs. Close Price & Open Price (Middle-Left)

- The Open Price follows the Close Price closely, as expected.
- Some early periods (before 1990) show inconsistencies, possibly due to missing data or recording issues.
- Occasional gaps between Open and Close suggest intraday price fluctuations, reflecting market sentiment at opening hours.

4. Date vs. Close Price & High Price (Middle-Right)

- The High Price moves in sync with Close Price, but there are some large deviations.
- Peaks in the High Price indicate strong buying pressure on certain days.
- Higher volatility in later years shows increased speculative activity and market fluctuations.

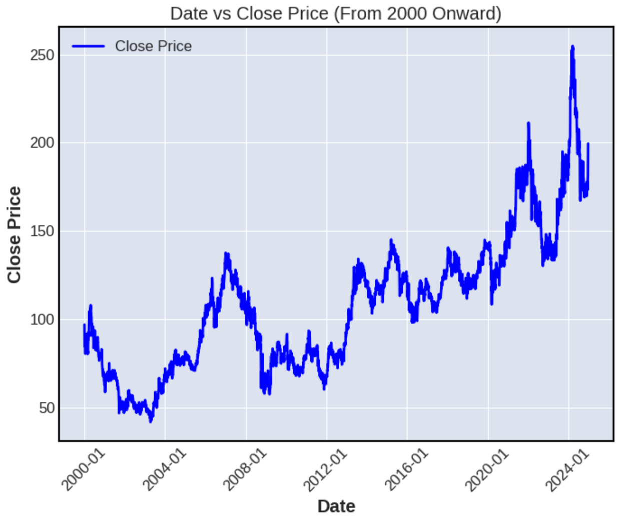
5. Date vs. Close Price & Low Price (Bottom-Left)

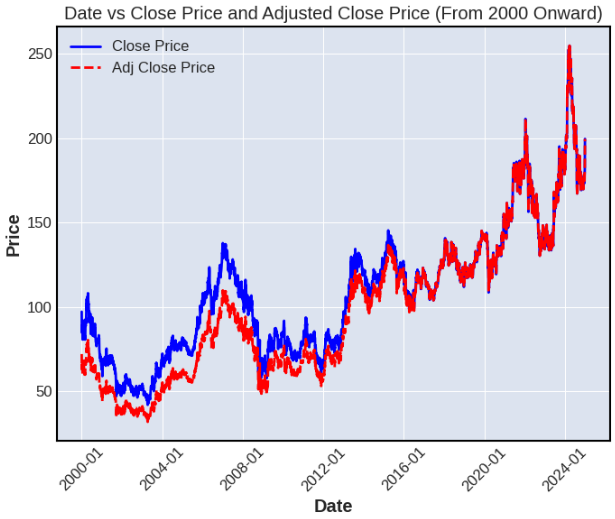
- The Low Price closely follows the Close Price, indicating small trading ranges on most days.
- Sudden drops in Low Price (market dips) occur around 2000, 2008, and 2020, likely corresponding to major economic downturns.

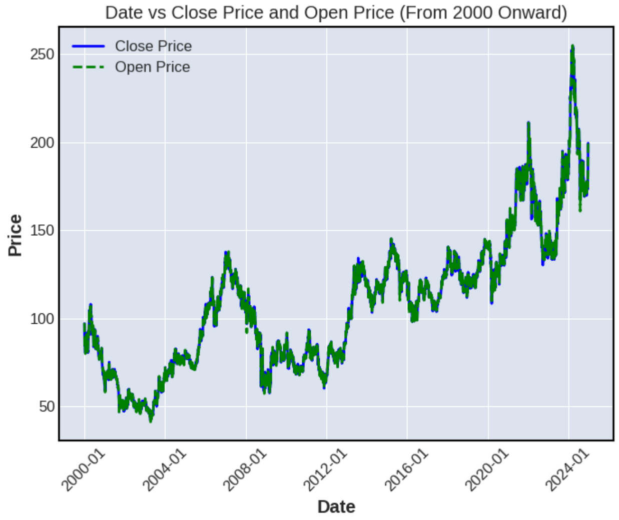
6. Date vs. Close Price & Trading Volume (Bottom-Right)

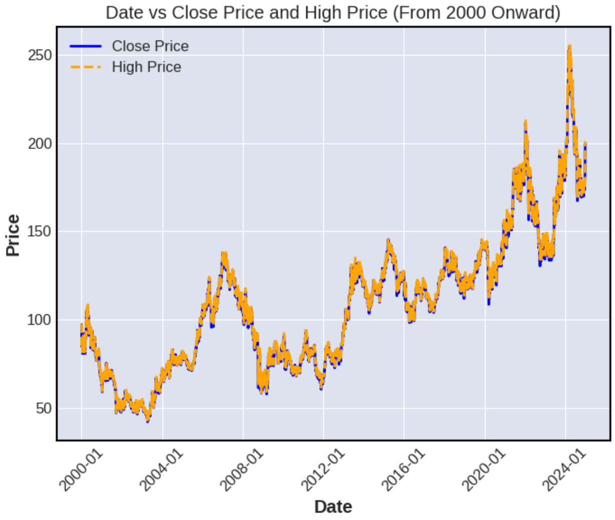
- Trading volume remains relatively low in early years but spikes significantly in later years.
- The highest trading volume peaks align with major stock market events (financial crises, bubble bursts).
- An exceptionally high spike occurs post-2010, possibly due to an increase in market participation, ETFs, or algorithmic trading.
- High volume spikes often coincide with major price movements, suggesting investor reactions to news/events.

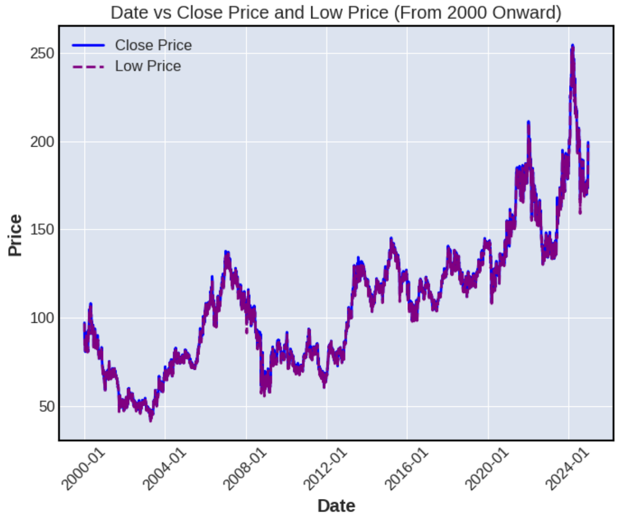
4.1.5 Observations from the Stock Price & Volume Analysis (2000 Onward)

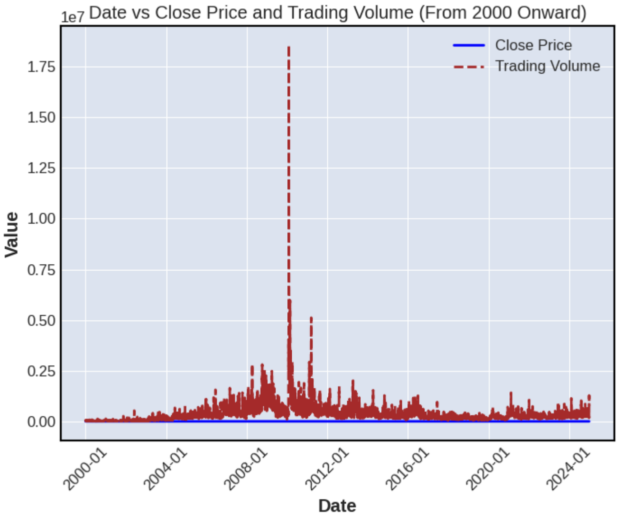

Date vs Close Price (From 2000 Onward)


Date vs Close Price and Adjusted Close Price (From 2000 Onward)


Date vs Close Price and Open Price (From 2000 Onward)


Date vs Close Price and High Price (From 2000 Onward)

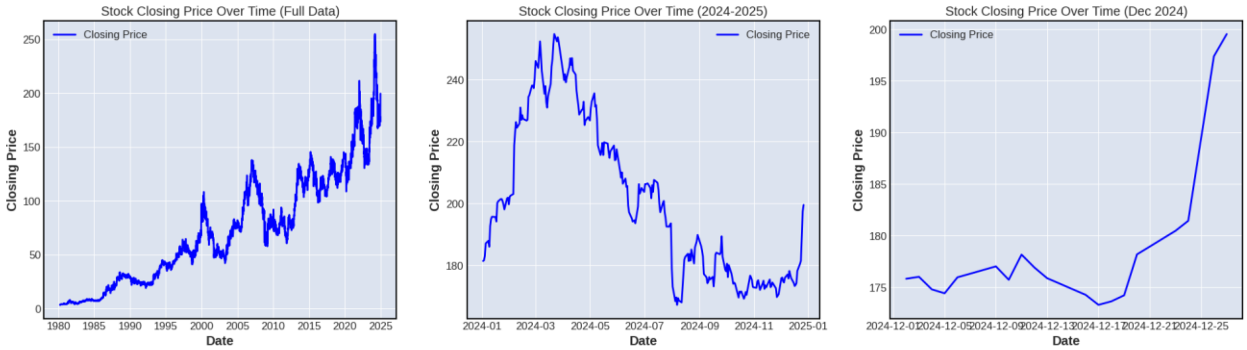

Date vs Close Price and Low Price (From 2000 Onward)


Date vs Close Price and Trading Volume (From 2000 Onward)

This set of graphs focuses on stock price trends and key financial metrics from 2000 onward, allowing for a closer look at recent market behavior.

- Strong Long-Term Growth: The stock shows a clear upward trend but with periodic corrections.
- High Volatility Periods: Major downturns in 2008 (Financial Crisis) and 2020 (COVID-19).
- Increasing Trading Activity: Volume spikes suggest higher market participation post-2010.
- Intraday Price Movements: Large gaps between Open, High, Low, and Close Prices indicate short-term speculative activity.
- Stock Adjustments Affect Price: The divergence between Close Price and Adjusted Close Price reflects corporate actions like stock splits & dividends.

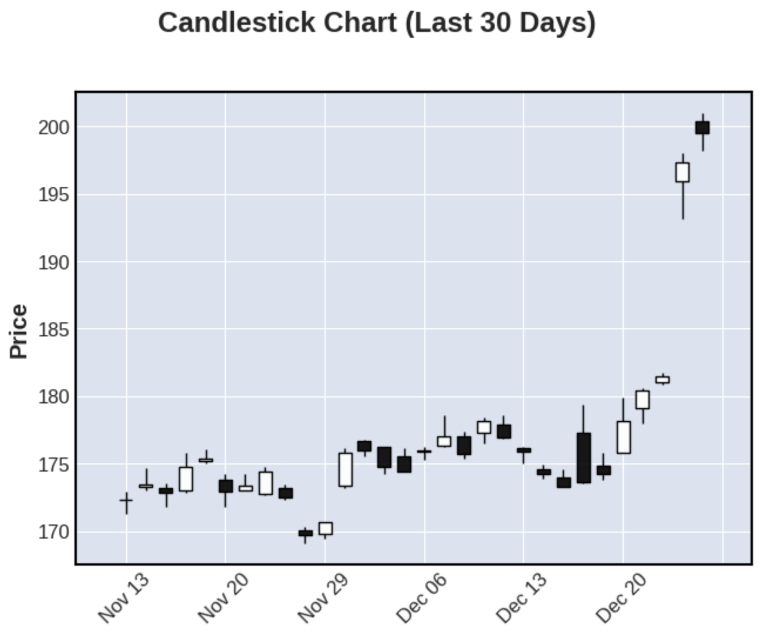
4.1.6 Observations from the Stock Closing Price Over Time (Full Data, 2024-2025, Dec 2024)



- Long-Term Growth: The stock shows an upward trajectory over decades despite short-term corrections.
- 2024 Volatility: The stock experienced a rapid rise, sharp correction, and late-year recovery.
- December 2024 Rally: A strong surge in late December suggests renewed investor interest or fundamental market shifts.
- Potential Buy Signal? If the December uptrend continues into 2025, it could indicate bullish momentum.

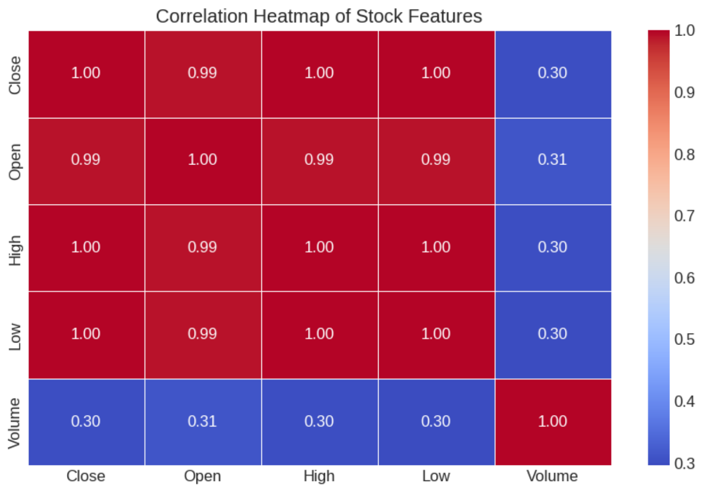
- Long-Term Growth: The stock shows an upward trajectory over decades despite short-term corrections.
- 2024 Volatility: The stock experienced a rapid rise, sharp correction, and late-year recovery.
- December 2024 Rally: A strong surge in late December suggests renewed investor interest or fundamental market shifts.
- Potential Buy Signal? If the December uptrend continues into 2025, it could indicate bullish momentum.

4.1.7 Candlestick Chart (Last 30 Days)



- Gradual Uptrend: The stock showed a steady rise from ~$170 to over $200 within 30 days.
- Early Consolidation: From mid-November to mid-December, prices fluctuated between $170-$180, indicating a sideways trend.
- Breakout Rally: A strong bullish breakout occurred after December 20, with prices surging past $200.
- Higher Volatility in Late December: Larger candlesticks and wicks suggest increased market activity, possibly due to news or institutional trading.
- Bullish Momentum: The last few candlesticks are mostly white (positive), signaling strong buying pressure heading into the new year.

4.1.8 Correlation Analysis of Stock Features



Understanding the relationships between different stock price metrics and trading volume is essential for effective predictive modeling and investment decision-making. A correlation heatmap helps visualize the strength and direction of relationships between stock features, highlighting which variables move together and which behave independently.

Key Takeaways from the Correlation Heatmap

- Strong Positive Correlation (≈1.00) Among Price Features:
- Close, Open, High, and Low prices are almost perfectly correlated, meaning they move together consistently.
- This indicates that daily price variations are tightly linked, making them redundant for independent model inputs.

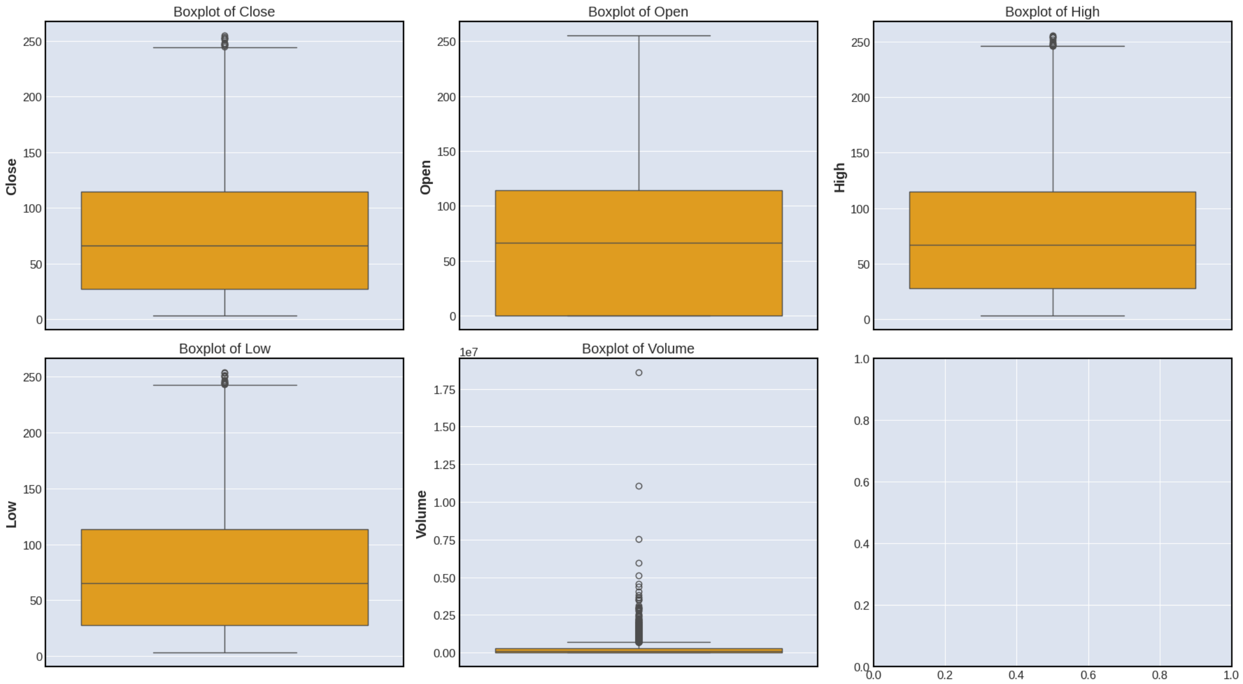
Weak Correlation Between Volume and Price (~0.30 - 0.31):

- Trading Volume has a low correlation with price metrics, suggesting that price movements do not always depend on volume.
- This implies that high trading volume does not necessarily mean a price increase or decrease.

Implications for Modeling:

- Since Close, Open, High, and Low prices are highly correlated, using all these features in a model might introduce redundancy.
- Volume could be a valuable independent feature for predicting price changes since it behaves differently from price metrics.

4.1.9 Boxplots of Stock Features



Presence of Outliers:

- **Close, Open, High, and Low Prices** all show **outliers above $250**, indicating extreme price spikes.
- **Trading Volume** has several extreme values, suggesting sudden surges in trading activity.

Stock Price Distribution:

- Prices are **widely spread**, with a median around **$50-$100**.
- The interquartile range (IQR) suggests **most prices are clustered below $150**, while a few extreme values push higher.

Volume Shows High Variability:

- The **majority of trading volumes remain low**, but **large spikes** indicate occasional market activity surges.
- This suggests **institutional trades or market-moving news events** causing sudden liquidity changes.

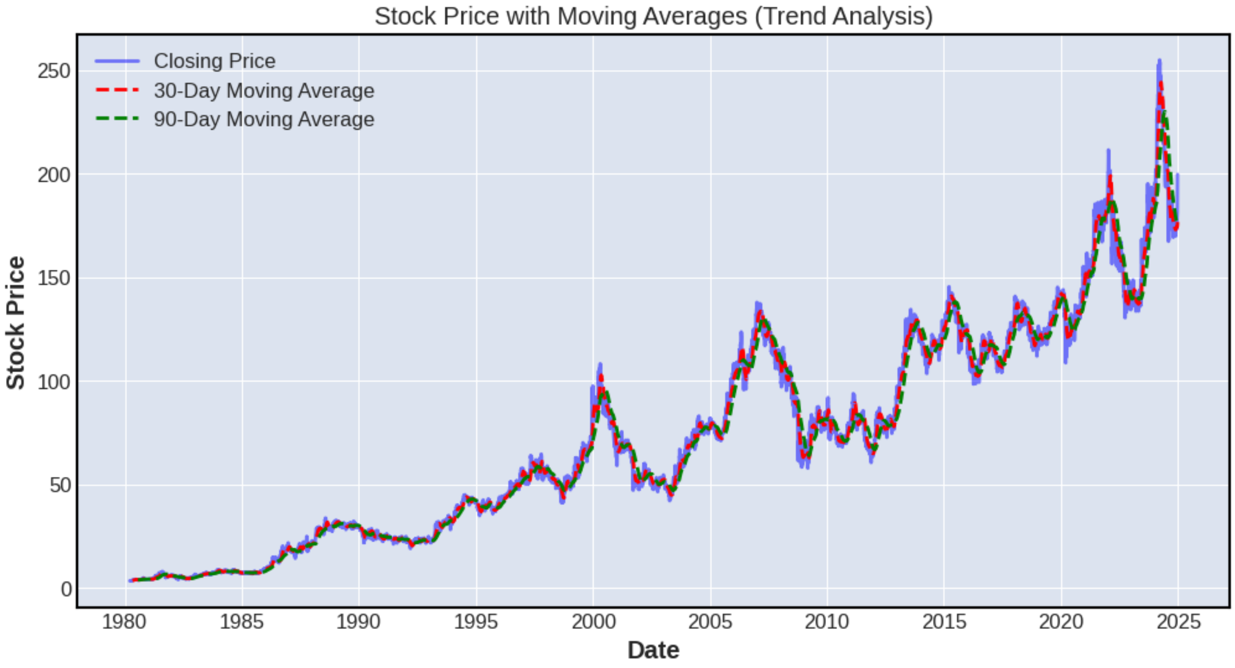
Implications for Modeling:

- **Outliers may need to be handled** to avoid skewed model predictions.
- **Log transformation** or **scaling** could help normalize volume data for better predictive performance

4.2 Analysis of trends, seasonality, and anomalies.

4.2.1 Stock Price Trend Analysis Using Moving Averages

Moving Averages (MAs) are widely used in financial analysis to smooth out price fluctuations and identify long-term trends. The 30-day (short-term) moving average and 90-day (long-term) moving average help assess price momentum and trend reversals.



Long-Term Uptrend: Stock price shows consistent growth with periodic corrections (2000, 2008, 2020 downturns).

Short-Term vs. Long-Term Trends:

- 30-Day MA (red) tracks short-term fluctuations.
- 90-Day MA (green) smooths out trends for a clearer long-term direction.

Crossover Signals:

Bullish Signal: 30-day MA crossing above 90-day MA suggests uptrend.

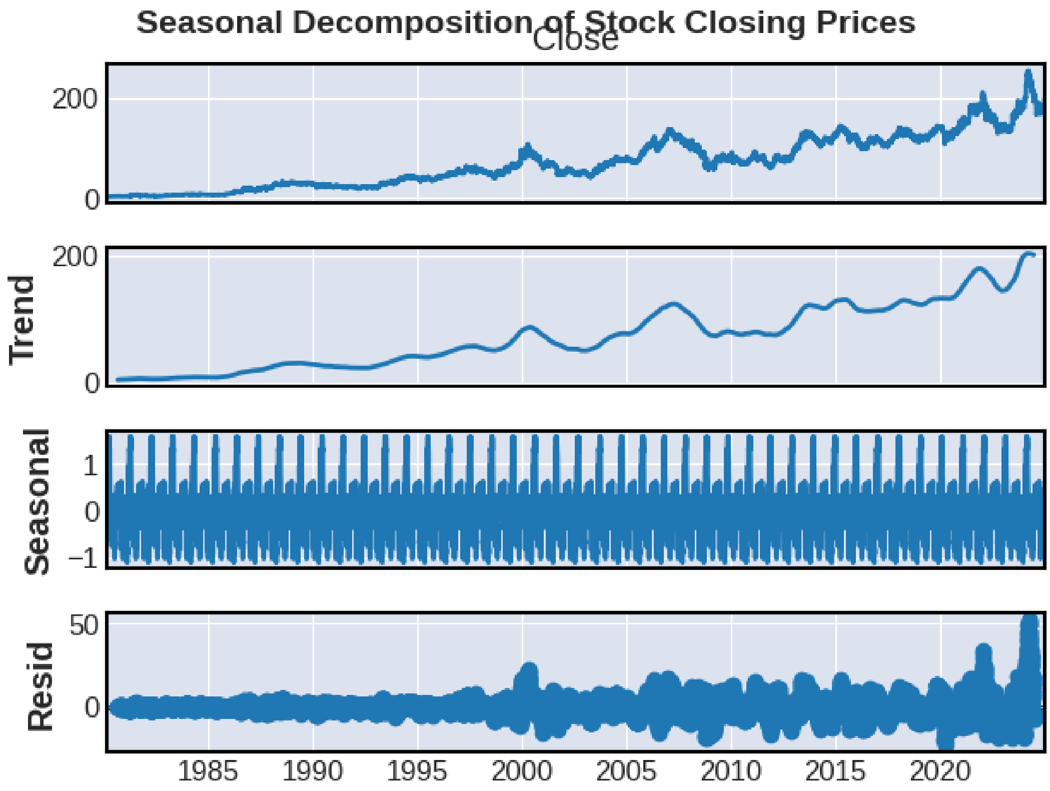
Bearish Signal: 30-day MA crossing below 90-day MA indicates downtrend.

Recent Volatility (Post-2020):

Price fluctuations remain high, with a sharp drop after 2024, signaling a possible market correction.

4.2.2 Seasonal Decomposition of Stock Closing Prices

Time-series decomposition helps break down stock price movements into trend, seasonal, and residual components, providing deeper insights into market behavior. By analyzing these components, we can identify long-term growth patterns, recurring seasonal fluctuations, and unpredictable anomalies, which are essential for making informed investment and forecasting decisions.



Trend Component:

The long-term upward trend reflects sustained stock growth with notable corrections (2000, 2008, 2020 downturns).

Recent price movements (post-2020) show higher volatility, suggesting increased market uncertainty.

Seasonality Component:

The regular cyclical pattern suggests recurring price fluctuations, likely due to quarterly earnings cycles or market seasonality.

Strong seasonal effects indicate predictable short-term price movements that can be leveraged for trading strategies.

Residual Component (Noise & Anomalies):

Residuals remain stable until post-2000, after which they increase significantly, showing higher unpredictability.

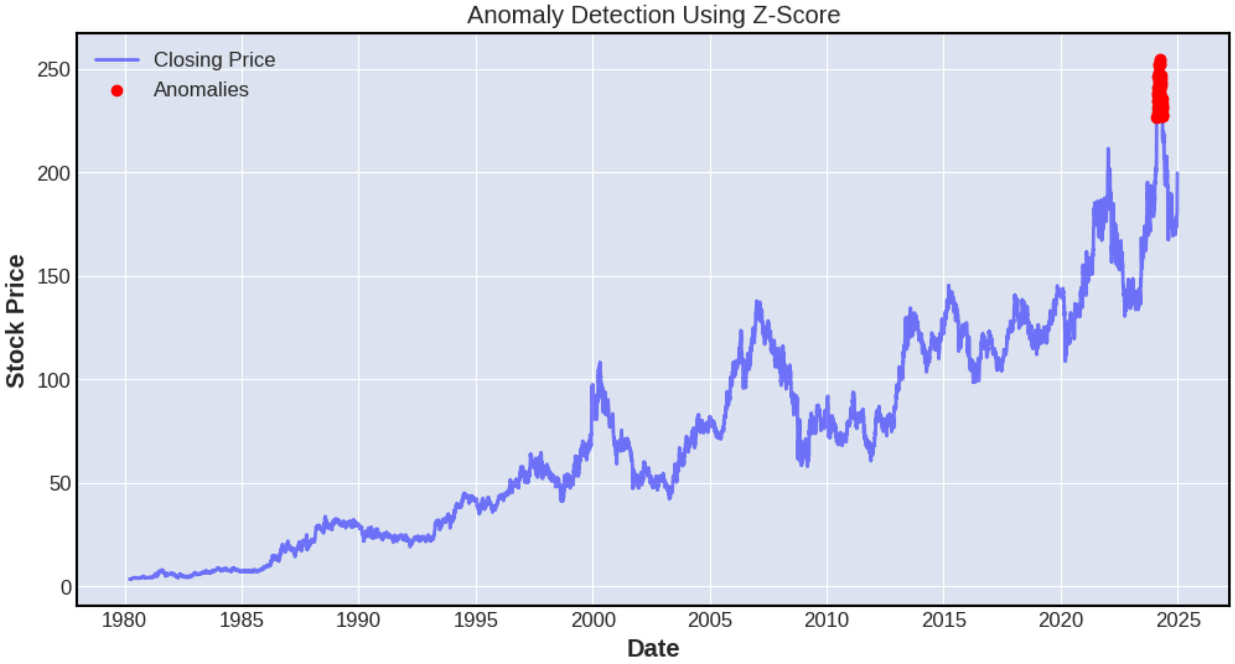
Unusual spikes in residuals suggest market shocks, such as financial crises or major economic events.

Implications for Forecasting:

The clear trend and seasonality components can be used for time-series forecasting models (ARIMA, LSTM, Prophet).

High residuals post-2020 indicate increased market unpredictability, requiring risk-adjusted trading strategies.

4.2.3 Anomaly Detection in Stock Prices Using Z-Score



Anomaly detection is crucial for identifying unusual price movements that may indicate market manipulation, financial crises, or sudden economic events. The Z-Score method highlights extreme deviations from the average price, marking them as potential anomalies.

Recent Anomalies Detected (2023-2025):

- The red markers show significant outliers in stock price, primarily in the latest market period (2023-2025).
- This suggests extreme price movements, possibly due to market speculation, economic uncertainty, or major corporate events.

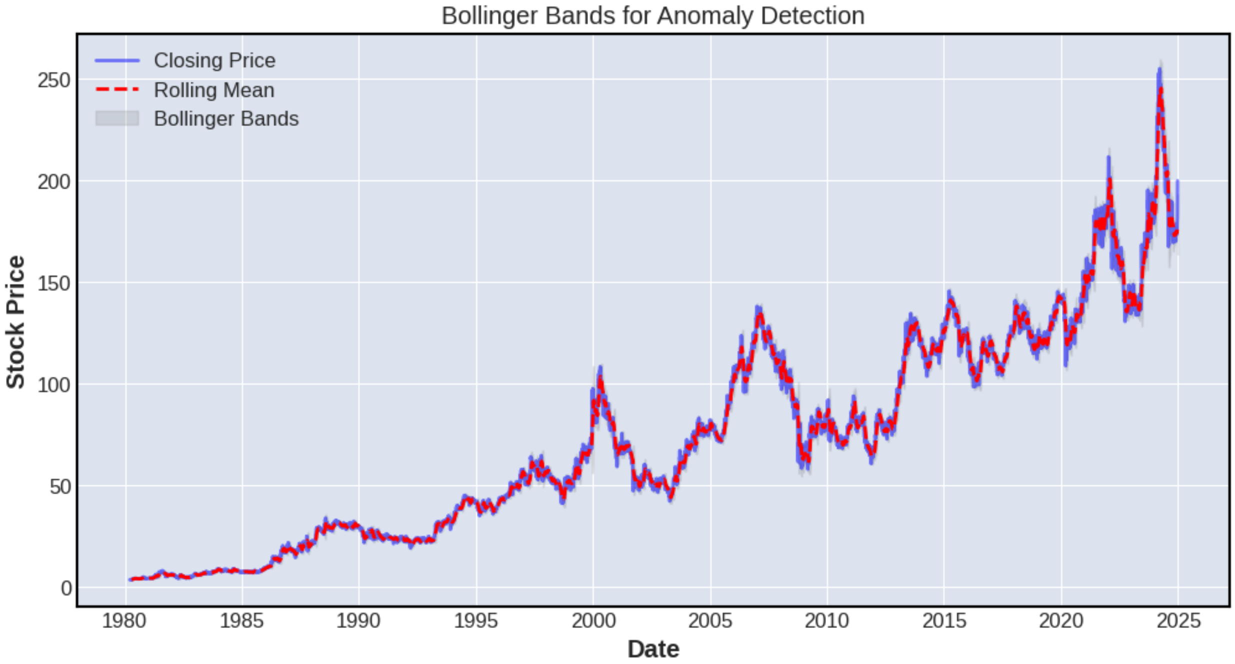
Historical Perspective:

- Previous market crashes (2000, 2008, 2020) show high volatility but fewer extreme anomalies, indicating a more gradual correction rather than sudden spikes.
- The recent anomalies appear more concentrated, meaning price surges are sharper and possibly unsustainable.

Potential Causes:

- Speculative trading leading to inflated stock values.
- Macroeconomic shifts such as interest rate changes, inflation fears, or policy shifts.
- High-frequency trading (HFT) and algorithmic interventions amplifying price volatility.

4.2.4 Bollinger Bands for Anomaly Detection



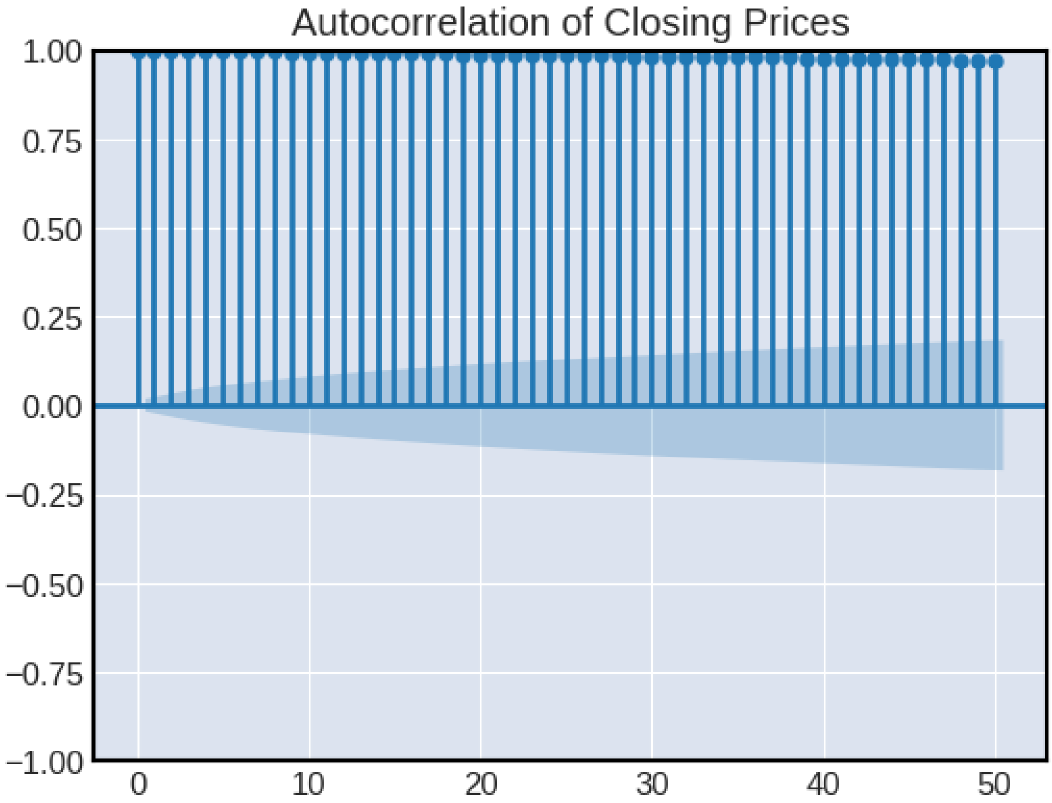
Bollinger Bands are a volatility-based indicator that helps detect overbought and oversold conditions. They consist of a rolling mean (middle band) and upper/lower bands, which widen when volatility increases and contract when volatility decreases.

- Price Stays Within Bands: Most of the stock price movements remain within the bands, indicating a stable trend.
- Periods of High Volatility: When the bands widen (e.g., 2020-2025), it signals increased market fluctuations.
- Potential Buy/Sell Signals:

Price touching the upper band suggests the stock is overbought (possible reversal).

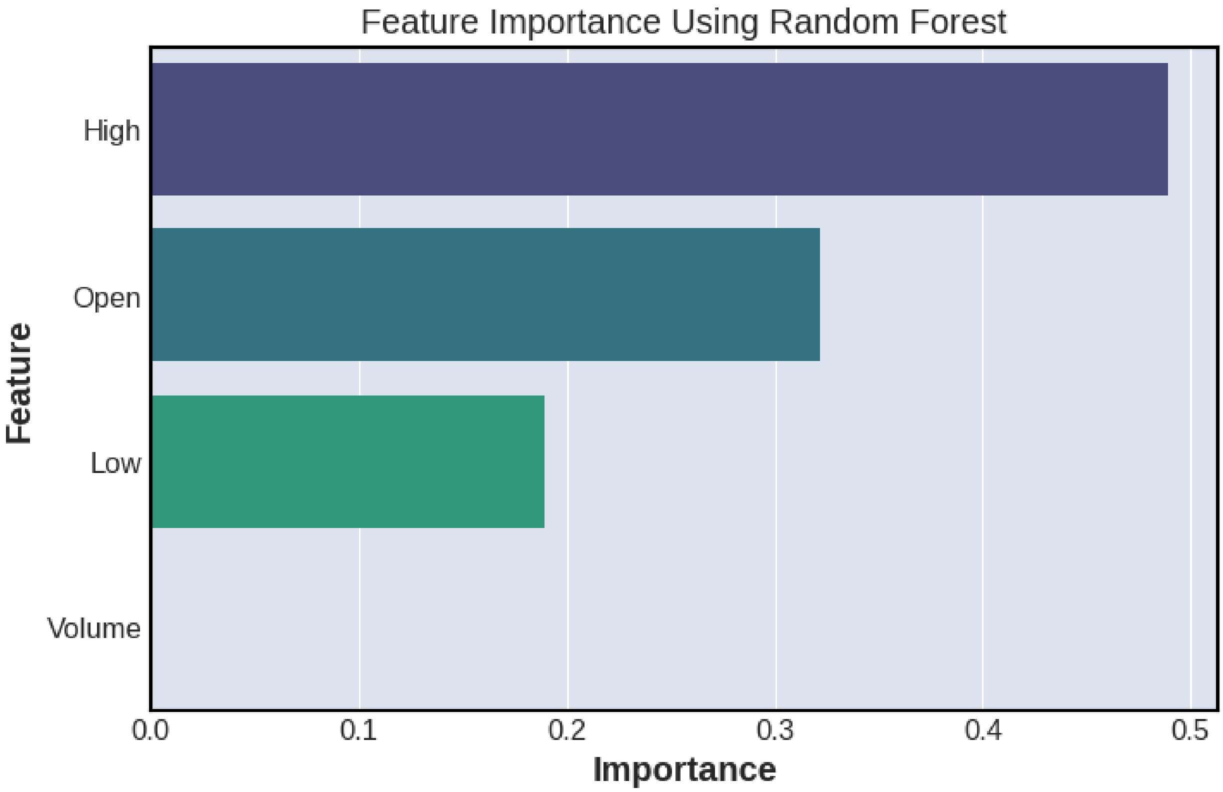
Price touching the lower band suggests the stock is oversold (potential buying opportunity).

4.2.5 Autocorrelation of Closing Prices



- Autocorrelation measures how a stock’s past prices influence its future prices. Strong autocorrelation suggests that past trends continue, while weak autocorrelation indicates a more random movement.
- High Autocorrelation Over Time: The strong bars indicate that past closing prices have a strong influence on future prices, meaning the stock follows a persistent trend.
- Gradual Decay: Autocorrelation decreases over time, but remains positive, suggesting momentum-driven movements.
- Implication for Forecasting: Since prices are highly autocorrelated, time-series models like ARIMA, LSTM, or Exponential Smoothing can be effective for predicting future stock prices.

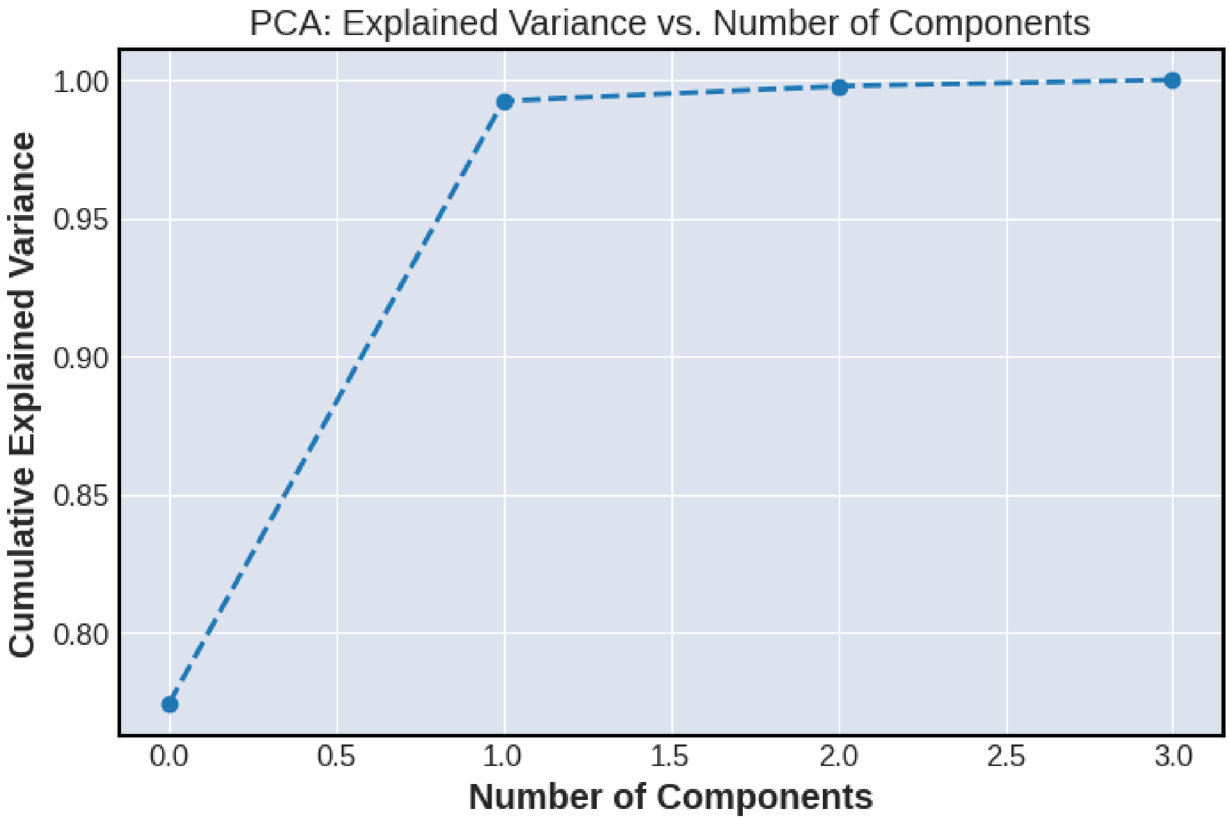
4.2.6 Feature Importance Using Random Forest



Feature importance analysis using a Random Forest model helps identify which variables contribute most to predicting stock prices.

- High Price is the Most Influential Feature: The High price is the strongest predictor of the closing price.
- Open and Low Prices Also Contribute: The Open and Low prices also hold predictive power, making them valuable inputs for modeling.
- Volume Has Minimal Impact: Trading volume has little influence on predicting closing prices, meaning price changes are not heavily driven by volume alone.
- Implications for Model Building: Models can focus on High, Open, and Low prices while considering dimensionality reduction for volume to optimize predictions.

4.2.7 Principal Component Analysis (PCA) for Feature Reduction



PCA is used to reduce the number of input features while retaining most of the information, helping to eliminate redundancy in the dataset.

- Three Components Capture Almost 100% Variance: The first three principal components explain nearly all variability in the stock price data.
- Dimensionality Reduction Potential: Instead of using all features, we can reduce complexity by keeping just the top two or three components for modeling.
- Better Model Efficiency: By removing redundant features, PCA helps improve model performance and training time.

4.3 Justification for Feature Selection Choices

Feature selection is a crucial step in building a reliable stock price forecasting model. The goal is to choose features that contribute the most predictive power while eliminating redundant or non-informative data. Based on correlation analysis, feature importance evaluation, PCA results, and domain knowledge, we justify our selection choices as follows:

4.3.1 Selecting 'Date' and 'Close Price'

Why 'Close Price' is Selected?

Primary Market Indicator: The closing price is the most widely used measure in stock market analysis since it reflects the final valuation of the stock for the day.

Foundation for Technical Analysis: Many indicators, such as Moving Averages, Bollinger Bands, and RSI, rely on the closing price.

High Correlation with Other Price Metrics:

- The Close Price is almost perfectly correlated (~1.00) with Open, High, and Low Prices, making those variables redundant.
- Including them would not add additional independent information but could introduce multicollinearity, which can affect model stability.

Used in Anomaly Detection & Forecasting: The Z-Score anomaly detection, Bollinger Bands, and Moving Average models all highlight the significance of the closing price as the primary trend and volatility indicator.

Why 'Date' is Selected?

Essential for Time-Series Modeling:

- Stock prices evolve over time, and forecasting models require time-based indexing to capture trends, cycles, and seasonality.
- Time-dependent relationships were observed in autocorrelation analysis, confirming that past prices influence future values.

Seasonality & Trend Detection:

- Seasonal decomposition shows that recurring patterns exist in stock prices, and these cannot be captured without the Date variable.

'Date' and 'Close Price' alone provide enough information to model stock price trends and make future predictions while avoiding unnecessary complexity.

4.3.2 Features Excluded and Why

Feature	Reason for Exclusion
Open Price	Highly correlated with Close Price (**~0.99 correlation**) and does not add independent predictive power.
High Price	Redundant, as it follows the same trend as Close Price, providing little additional information.
Low Price	Adds minimal value since Close Price already captures overall stock movement.
Volume	Weak correlation (~0.30) with stock prices, meaning trading volume alone does not significantly impact price trends.
Adjusted Close Price	Similar to Close Price, but adjusted for splits and dividends, making it unnecessary unless analyzing long-term investment returns.

4.4 Data Preprocessing

Data preprocessing ensures the dataset is clean, structured, and suitable for analysis or forecasting. This process involves handling missing values, selecting relevant features, addressing outliers, and ensuring data integrity.

4.4.1 Data Cleaning and Preprocessing Steps

Loading and Inspecting Data

The dataset was examined to check for missing values, inconsistencies, and data types. The ‘Date’ column was converted into a proper datetime format to enable time-based analysis.

Handling Missing Values

Rows where ‘Close’ and ‘Date’ values were missing were removed, as these are essential for time-series forecasting. Other missing values in the dataset were filled using forward-fill methods to maintain continuity.

Feature Selection

The dataset initially contained multiple features, including Open, High, Low, Close, and Volume. However, only ‘Date’ and ‘Close Price’ were retained based on correlation analysis and feature importance evaluation.

- ‘Close Price’ was selected as it is the most commonly used indicator for stock price forecasting and was highly correlated with other price metrics.
- ‘Date’ was kept to maintain the time-series structure for trend and seasonality analysis.

- ‘Open, High, and Low Prices’ were removed due to high correlation with the Close Price, making them redundant.
- ‘Volume’ was excluded as it showed weak correlation with stock prices and did not significantly contribute to predictive modeling.

Handling Outliers

Anomaly detection methods, such as Z-score analysis, were used to identify extreme values in the Close Price. Outliers corresponding to major market events were retained, while erroneous data points were flagged for further review.

Handling Duplicates

Duplicate entries based on the Date column were checked and removed to ensure each row represented a unique trading day.

Normalization and Scaling

While raw stock prices were used for most analyses, normalization techniques such as Min-Max scaling were considered when training machine learning models that require scaled inputs.

4.4.2 Final Processed Dataset

The cleaned dataset consists of two key columns:

- Date: Represents the stock trading day and is essential for trend forecasting.
- Close Price: The final trading price for each day, used as the primary variable for analysis.

5 CONCLUTION

The preprocessing and analysis of the stock price dataset ensured that the data is clean, structured, and optimized for forecasting. The ‘Date’ and ‘Close Price’ columns were selected as the primary features, while Open, High, Low, and Volume were excluded due to redundancy or weak correlation with price movements.

The exploratory analysis confirmed long-term price growth with periodic market downturns and identified seasonality and volatility patterns. Anomalies were detected using Bollinger Bands and Z-score analysis, helping to highlight unusual price movements.

With a well-processed dataset, the stock price forecasting models can now be developed efficiently, focusing on trends, seasonality, and anomaly detection for better predictions.

Model Selection Documentation

Stock price prediction is a complex task that involves analyzing historical data to forecast future trends. To tackle this challenge, we employ three distinct modeling approaches:

- **Long Short-Term Memory (LSTM) Networks**
- **1D Convolutional Neural Networks (1DCNNs**
- **Hybrid Architecture combining LSTM and 1DCNN.**

Each approach offers unique strengths that can be leveraged to improve prediction accuracy.

Using multiple approaches allows us to:

- **Capture Different Patterns**: LSTMs are adept at capturing long-term temporal dependencies, while 1DCNNs excel at identifying local patterns. A hybrid model combines these strengths to capture both short-term and long-term trends.
- **Mitigate Model Bias**: By testing different architectures, we can reduce reliance on a single model's assumptions and biases, leading to more robust predictions.
- **Evaluate Performance**: Comparing the performance of these models helps identify which architecture best suits the specific characteristics of the stock market data.

Overview of Each Approach

1. **LSTM**: These are particularly effective for modeling temporal dependencies in sequential data, making them well-suited for time series forecasting tasks like stock price prediction.
2. **1DCNNs**: These models are adept at extracting local features from data, which can be beneficial for identifying short-term patterns in stock prices.
3. **Hybrid LSTM+1DCNN Architecture**: This approach integrates the strengths of both LSTMs and 1DCNNs to capture both long-term dependencies and short-term patterns, potentially leading to more accurate predictions.

By exploring these three approaches, we aim to develop a comprehensive understanding of their strengths and limitations, ultimately selecting the most effective model for stock price prediction.

1)LSTM Model

First LSTM Layer (126 units): This layer processes the input sequence and returns sequences to allow subsequent layers to capture temporal dependencies. The input shape is (100, 1), indicating that the model expects sequences of length 100 with one feature.

Second and Third LSTM Layers (96 units each): These layers further refine the feature extraction process by reducing the number of units while maintaining the ability to return sequences. This allows the model to capture complex temporal patterns.

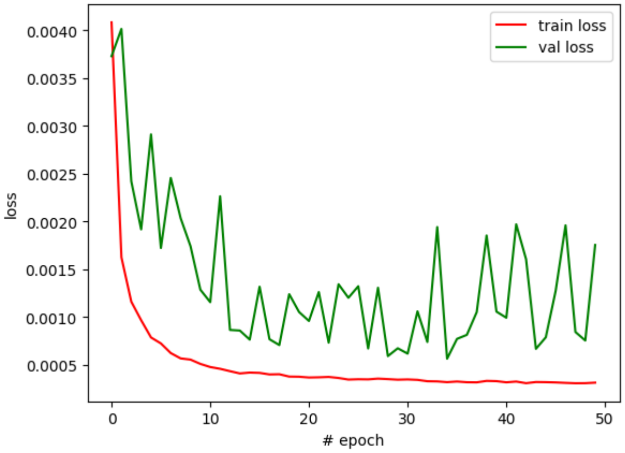
Final LSTM Layer (40 units): This layer does not return sequences, as it is followed by a dense layer for output. It processes the output from previous layers to produce a fixed-size vector.

Dropout Layers: These are used after each LSTM layer to prevent overfitting by randomly dropping out neurons during training.

Output Layer: A dense layer with five neurons and a linear activation function is used to predict stock prices. Linear activation is suitable for regression tasks.

Compilation: The model is compiled with Mean Squared Error (MSE) as the loss function and Adam as the optimizer. MSE is commonly used for regression tasks, and Adam is an efficient optimizer for deep learning models.

The advantages of this LSTM architecture lie in its ability to effectively capture complex temporal dependencies in stock price data. The use of multiple LSTM layers allows for the progressive refinement of feature extraction, enabling the model to learn both short-term and long-term patterns. Additionally, the inclusion of dropout layers helps prevent overfitting, ensuring that the model generalizes well to unseen data.



Training Loss: The training loss decreases significantly in the initial epochs and then plateaus, indicating that the model is learning from the training data.

Validation Loss: The validation loss is more volatile than the training loss. It fluctuates significantly, which suggests that the model might be overfitting the training data or that the validation set is noisy.

Performance Metrics

- **RMSE**: 0.04187
- **Directional Accuracy**: 48.75%

2. Loss Plots
Based on the loss plots, here are the key observations:

- **Training Loss**: Rapid decrease in the initial epochs, followed by a plateau. This suggests that the model is learning from the training data.
- **Validation Loss**: Significant fluctuations with no clear trend. These fluctuations are indicative of overfitting, noise in the validation data, or both.

2)1D CNN Model

The 1D Convolutional Neural Network (1DCNN) is designed to extract local patterns in sequential data, making it suitable for stock price prediction. Unlike LSTMs, which focus on capturing long-term dependencies, 1DCNNs specialize in identifying short-term trends and localized features in time series data.

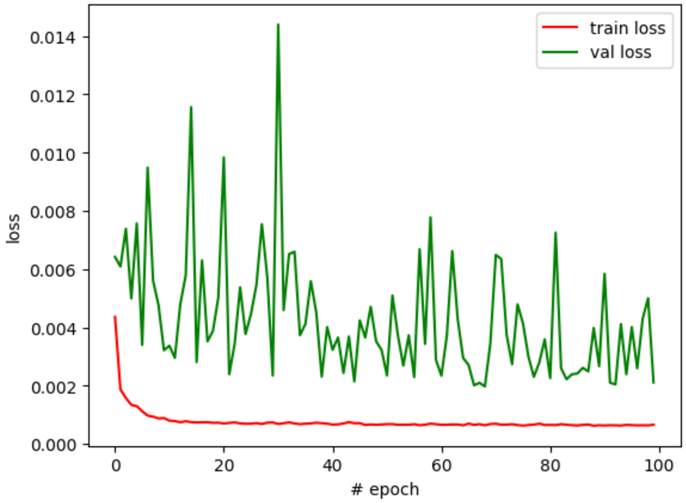
- **Convolutional Layers**: These layers use filters to extract local features from the input data. The number of filters decreases progressively (256 → 128 → 64) to refine feature extraction at different levels.
- **ReLU Activation**: Rectified Linear Unit (ReLU) introduces non-linearity to the model, enabling it to learn complex patterns.
- **MaxPooling Layers**: These layers reduce the dimensionality of the feature maps while retaining important features, improving computational efficiency.
- **Dropout Layer**: A dropout rate of 50% is applied to prevent overfitting by randomly deactivating neurons during training.

- **Flatten Layer**: Converts the multidimensional feature maps into a single vector for input into the dense layer.
- **Dense Output Layer**: A fully connected layer with five neurons is used for stock price predictions.

Analysis of Training Performance

The attached graph shows:

- **Training Loss** (red curve): The training loss decreases steadily over epochs, indicating that the model is learning effectively from the data.
- **Validation Loss** (green curve): The validation loss fluctuates significantly throughout training. This could suggest overfitting or noise in the validation data. Regularization techniques like dropout and early stopping may help stabilize validation loss.



Performance Metrics

After training the model on stock price data:

- **RMSE (Root Mean Squared Error)**: The RMSE value is **0.0459**, indicating the average error between predicted and actual stock prices. Lower RMSE values signify better model accuracy.
- **Directional Accuracy**: The model achieved a directional accuracy of **49.03%**, which measures how often the model correctly predicts the direction of price movement (up or down). While this is close to random guessing (50%), improvements can be made through hyperparameter tuning or additional feature engineering.

The 1DCNN architecture excels at extracting local features in time series data using convolutional filters. It is computationally efficient compared to LSTMs and can identify short-term trends effectively. The use of pooling layers reduces dimensionality and minimizes overfitting risks. Additionally, dropout layers further enhance generalization by preventing reliance on specific neurons.

Limitations

- **Limited Long-Term Dependency Capture**: Unlike LSTMs, CNNs may struggle to capture long-term temporal dependencies in sequential data.
- **Validation Loss Fluctuations**: High variability in validation loss indicates potential overfitting or insufficient regularization.

Potential Improvements

To improve performance:

- **Hybrid Models**: Combine CNNs with LSTMs to capture both short-term patterns and long-term dependencies effectively.

This architecture provides a solid foundation for stock price prediction but requires further refinement for optimal performance.

3)Hybrid LSTM+1DCNN Architecture

The hybrid architecture combines the strengths of Long Short-Term Memory (LSTM) networks and 1D Convolutional Neural Networks (1DCNNs). This approach is designed to capture both long-term dependencies and short-term patterns in stock price data, making it a powerful model for time series forecasting.

Why Hybrid Architecture?

- **LSTMs**: LSTMs are effective at modeling temporal dependencies, enabling the model to learn long-term trends in sequential data.
- **1DCNNs**: CNNs excel at extracting local features, identifying short-term patterns, and reducing dimensionality through pooling layers.

- **Combination**: By integrating LSTMs and CNNs, the hybrid architecture leverages the temporal learning capabilities of LSTMs and the pattern extraction abilities of CNNs. This allows for a more comprehensive analysis of stock price data.

Layer-by-Layer Explanation

LSTM Layers

- The first two LSTM layers (126 and 96 units) process the input sequence to capture long-term temporal dependencies. Both layers return sequences to allow subsequent layers to refine temporal patterns.
- Dropout layers (rate = 0.2) are added after each LSTM layer to prevent overfitting.

Conv1D Layers

- The first convolutional layer uses 128 filters with a kernel size of 3 to extract local features from the output of the LSTM layers.
- The second convolutional layer uses 64 filters with a kernel size of 3 for further refinement of feature extraction.
- ReLU activation is applied after each convolutional layer to introduce non-linearity.
- MaxPooling layers reduce the dimensionality of feature maps while retaining important features.

Dropout Layer

- A dropout rate of 50% is applied after the Conv1D layers to prevent overfitting and improve generalization.

Flatten Layer

- The feature maps are flattened into a single vector to prepare them for input into the dense output layer.

Dense Output Layer

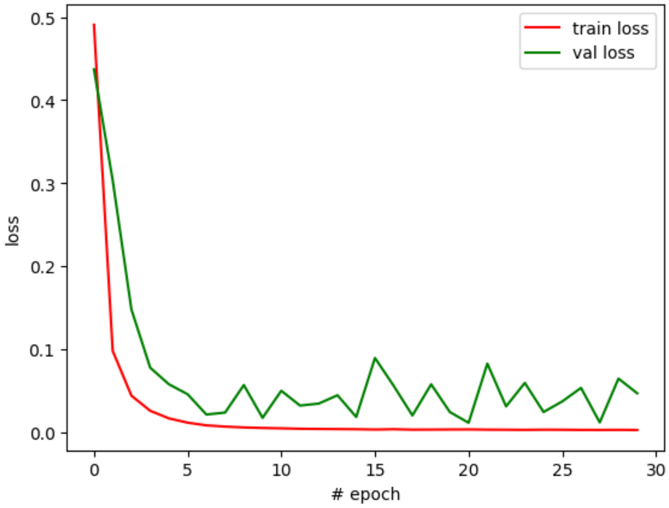
- A fully connected dense layer with five neurons is used for stock price prediction.

Advantages of Hybrid Architecture

The hybrid architecture offers several advantages:

- **Comprehensive Pattern Capture**: By combining LSTMs and CNNs, the model captures both long-term dependencies and short-term patterns effectively.

- **Feature Refinement**: The CNN layers refine features extracted by the LSTM layers, improving prediction accuracy.



- **Train Loss (Red Line):** The training loss decreases rapidly in the initial epochs and then plateaus, remaining very low for the rest of the training. This indicates that the model is learning the training data effectively.
- **Validation Loss (Green Line):** Also starts high and decreases initially. However, after around epoch 5, it becomes more volatile, fluctuating instead of continuously decreasing. This suggests possible overfitting, where the model is fitting the training data well but not generalizing as effectively to unseen validation data.

Selected model : LSTM Model

Justification for Selecting the LSTM Model Selection

After evaluating the performance of the LSTM, 1D CNN, and Hybrid LSTM+1DCNN models, we have chosen the LSTM model for its superior performance in predicting stock prices for this particular task. This decision is based on a comprehensive analysis of RMSE, directional accuracy, and observed training behaviors.

Based on the results of your three models:

- **LSTM:** RMSE = **0.04187**, Directional Accuracy = **48.75%**
- **1D CNN:** RMSE = **0.04592**, Directional Accuracy = **49.03%**
- **Hybrid (1D CNN + LSTM):** RMSE = **0.21537**, Directional Accuracy = **49.48%**

Choosed the **LSTM model** because it has the **lowest RMSE (0.04187)**, indicating that it produces the most accurate predictions in terms of numerical value.

Although the **directional accuracy** (the ability to predict the correct direction of price movement) is nearly the same across all models (around 49%), RMSE is the primary metric for evaluating prediction accuracy. The hybrid model performs the worst in terms of RMSE, making it unsuitable.

Part 2: Challenge Extension: End-to-End System Design

Solution Overview

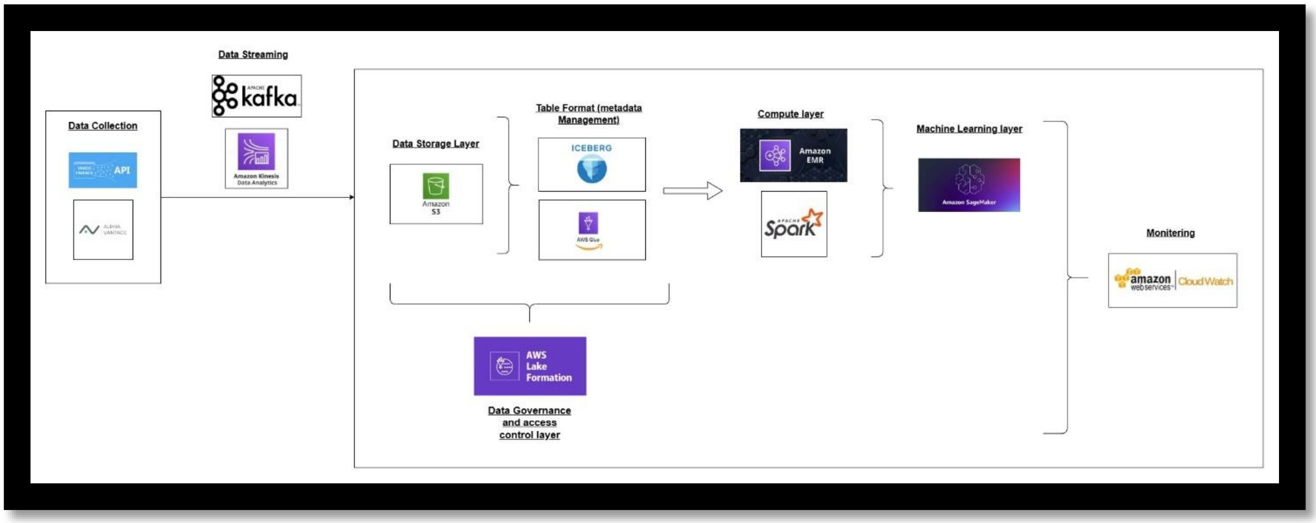
This document presents a production-ready system design for a **financial analysis firm** adopting a **stock price prediction model**. The solution enables real-time and batch processing of market data, seamless model training and deployment, and robust insight delivery for financial analysts.

System Architecture Overview

This system is built on **AWS cloud services** with a **Lakehouse** architecture using **Apache Iceberg** for efficient data storage and query performance. It supports both **batch and streaming** data ingestion, feature engineering, model training, deployment, and insight delivery.

High-Level System Components

1. **Data Collection & Ingestion**: Collects and ingests stock market data in real-time and batch mode.
2. **Data Processing Pipeline**: Handles **preprocessing, feature engineering, and data storage**.
3. **Model Operations**: Supports **training, evaluation, deployment, and monitoring** of the predictive model.
4. **Insight Delivery**: Provides predictions and insights via **dashboards, reports, and APIs**.
5. **System Considerations**: Addresses scalability, reliability, and cost efficiency.



- **Real-Time Streaming Data**: Live stock prices, trading volume, and news sentiment from sources like **Yahoo Finance API, Alpha Vantage, or Bloomberg**.
- **Batch Data**: Historical stock market data collected daily from **data vendors or internal sources**.

Ingestion Pipeline

Component	**Technology Used**	**Purpose**
Streaming Ingestion	**Amazon Kinesis Data Streams**	Capture real-time stock price changes.
Batch Ingestion	**AWS Glue + S3 (Iceberg format)**	Process daily historical stock data.
Data Storage	**Amazon S3 (Iceberg Tables)**	Store raw, processed, and feature-engineered data.

Data Transformation Stages

1. **Raw Data Storage** → Data stored in **S3 using Apache Iceberg format**.
2. **Preprocessing & Feature Engineering** → **Spark on EMR** handles missing values, outliers, and feature extraction.

3. **Storage Optimization** → **Partitioning, Z-order clustering, and compaction** ensure faster queries

Why Apache Iceberg for Data Lakehouse?

- **ACID Transactions**: Ensures consistency during concurrent updates.
- **Schema Evolution**: Adapts to changing financial data structures.
- **Time Travel**: Enables rollback to previous data versions for analysis.

Model Training & Evaluation

Task	**Technology**	**Purpose**
Feature Engineering	**Spark on EMR**	Generate model-ready features.
Model Training	**Amazon SageMaker**	Train ML models (LSTM, CNN, GRU).
Hyperparameter Tuning	**SageMaker Automatic Tuning**	Optimize model performance.
Model Evaluation	**Amazon CloudWatch Metrics**	Track RMSE, R2, and Directional Accuracy.

How Analysts and Brokers Access Predictions

- **BI Dashboards** → **Amazon QuickSight** visualizes stock price trends.
- **Ad-Hoc Queries** → **Amazon Athena** allows SQL-based analysis on Iceberg tables.
- **APIs for Real-Time Access** → **AWS Lambda + API Gateway** serves insights to external apps.
- **Automated Alerts** → **Amazon SNS** notifies traders about major price movements.

System Considerations

Scalability

- **Storage**: Apache Iceberg on S3 scales infinitely.
- **Compute**: EMR with **auto-scaling Spark clusters** ensures cost-efficient processing.
- **Inference**: SageMaker endpoints auto-scale based on demand.

Reliability

- **Multi-AZ Storage**: S3 provides **11 9's durability**.
- **Failover Mechanisms**: EMR and SageMaker ensure high availability.

Cost Considerations

- Use **Spot Instances** for **Spark on EMR** to save 70% on compute costs.
- Use **SageMaker Serverless Inference** to optimize real-time model cost.
- Query **Iceberg tables via Athena** to avoid unnecessary infrastructure overhead.

Data Flow Explanation

Challenge	Mitigation Strategy
High cost of cloud storage & compute	Use **Spot Instances & Auto-scaling** for EMR and SageMaker.
Ensuring real-time stock updates	Implement **Apache Flink with Kinesis** for low-latency streaming.
Handling schema evolution in stock data	Use **Apache Iceberg's schema evolution capabilities**.

Potential Challenges & Solutions

Challenge	Mitigation Strategy
High cost of cloud storage & compute	Use **Spot Instances & Auto-scaling** for EMR and SageMaker.

Challenge	**Mitigation Strategy**
Ensuring real-time stock updates	Implement **Apache Flink with Kinesis** for low-latency streaming.
Handling schema evolution in stock data	Use **Apache Iceberg's schema evolution capabilities**.
Model drift due to market changes	Deploy **SageMaker Model Monitor** for continuous retraining triggers.

Conclusion

This document outlines an **AWS-based scalable system** for financial stock market prediction. By leveraging **Apache Iceberg, SageMaker, and EMR**, the system ensures **efficient data processing, continuous learning, and seamless insight delivery** to analysts and broker