**TIME SERIES ANALYSIS AND FORECASTING FOR STOCK MARKET**

*Internship Project Report*

Internship Role: Data Analyst Intern

Internship Organization: Zidio Development

Project Title: Time Series Analysis And Forecasting For Stock Market

Team Members & Responsibilities:

Aaditya Mehta– Data Collection & Description

Team Member 2 – Data Cleaning & Preprocessing

M.Fathimuthujohara– Exploratory Data Analysis & Model Evaluation

Yash Raj – Feature Engineering

Bhuvana Sandhya Rani Samuji – Time Series Modeling

Prithikasree S B – Final Documentation & Presentation

Internship Duration: 15 May 2025 – 15 June 2025

Submitted To: Chandan Mishra

GitHub Repository:

<https://github.com/yashraj195/Stock_Analysis_ZD>

Table of Contents

[1.EXECUTIVE SUMMARY 4](#_Toc198110950)

[1.1 OVERVIEW OF THE PROJECT 4](#_Toc198110951)

[2.INTRODUCTION 5](#_Toc198110952)

[2.1 BACKGROUND 5](#_Toc198110953)

[2.2 OBJECTIVE 5](#_Toc198110954)

[2.3 SCOPE OF THE PROJECT 5](#_Toc198110955)

[2.4 SIGNIFICANCE OF STOCK FORECASTING 5](#_Toc198110956)

[3. DATA COLLECTION & DESCRIPTION 6](#_Toc198110957)

[3.1 DATA SOURCE 6](#_Toc198110958)

[3.2 DATA FEATURES 6](#_Toc198110959)

[3.3 DATE RANGE AND FREQUENCY 6](#_Toc198110960)

[3.4 INITIAL OBSERVATIONS 6](#_Toc198110961)

[3.5 DESCRIPTIVE STATISTICS 7](#_Toc198110962)

[3.6 SUMMARY 8](#_Toc198110963)

[4.DATA CLEANING AND PREPROCESSING 9](#_Toc198110964)

[4.1 HANDLING MISSING VALUES 9](#_Toc198110965)

[4.2 OUTLIER DETECTION AND TREATMENT 9](#_Toc198110966)

[4.3 DATETIME CONVERSION 9](#_Toc198110967)

[4.4 FEATURE SCALING 9](#_Toc198110968)

[4.5 TRAIN-TEST SPLIT 9](#_Toc198110969)

[4.6 PREPROCESSING TOOLS USED 9](#_Toc198110970)

[5. EXPLORATORY DATA ANALYSIS (EDA) 10](#_Toc198110971)

[5.1 PRICE TRENDS 10](#_Toc198110972)

[5.2 VOLUME ANALYSIS 10](#_Toc198110973)

[5.3 CORRELATION HEATMAP 10](#_Toc198110974)

[5.4 ROLLING AVERAGES 11](#_Toc198110975)

[5.5 MONTHLY & YEARLY VOLATILITY 11](#_Toc198110976)

[6. FEATURE ENGINEERING 12](#_Toc198110977)

[6.1 TIME-BASED FEATURES 12](#_Toc198110978)

[6.2 PRICE MOVEMENT FEATURES 12](#_Toc198110979)

[6.3 TECHNICAL INDICATORS – MOVING AVERAGES 12](#_Toc198110980)

[6.4 VOLATILITY INDICATORS 13](#_Toc198110981)

[6.5 MOMENTUM INDICATOR 13](#_Toc198110982)

[6.6 VOLUME-BASED FEATURE 13](#_Toc198110983)

[6.7 FINAL CLEANING 13](#_Toc198110984)

[7. TIME SERIES MODELING - LSTM 15](#_Toc198110985)

[7.1 OVERVIEW OF LSTM 15](#_Toc198110986)

[7.2 DATA PREPARATION 15](#_Toc198110987)

[7.3 MODEL ARCHITECTURE 15](#_Toc198110988)

[7.4 MODEL TRAINING AND PERFORMANCE 16](#_Toc198110989)

[7.5 FORECAST VS ACTUAL 16](#_Toc198110990)

[8. MODEL EVALUATION & FORECASTING 17](#_Toc198110991)

[8.1 EVALUATION METRICS 17](#_Toc198110992)

[8.2 FORECAST VS. ACTUAL VISUALIZATION 17](#_Toc198110993)

[8.3 OVERFITTING/UNDERFITTING 18](#_Toc198110994)

[8.4 FORECASTING FUTURE PRICES 18](#_Toc198110995)

[9. CONCLUSION 19](#_Toc198110996)

[10. REFERENCES & GITHUB 19](#_Toc198110997)

# 1.EXECUTIVE SUMMARY

## 1.1 OVERVIEW OF THE PROJECT

This project aims to analyze and forecast stock market trends using time series analysis techniques. Interns will explore various time series models to understand historical patterns, identify trends and seasonality, and make short-term or long-term predictions. This project offers real-world experience in financial data analytics, model development, and result interpretation.

The team followed a structured pipeline:

* **Data** was collected from Kaggle dataset like Microsoft Stock, containing daily records of Open, High, Low, Close, Volume prices.
* **Preprocessing** included handling missing values, converting date formats, scaling features, and splitting the dataset chronologically.
* **Exploratory Data Analysis (EDA)** revealed trends, volatility patterns, and trading volume dynamics through various visualizations.
* **Feature Engineering** introduced lag variables, rolling statistics, and technical indicators such as RSI and MACD to enhance model performance.
* **Modeling** was carried out using both traditional and deep learning methods including ARIMA, Prophet, and LSTM.
* **Evaluation** was done using RMSE, MAE, and MAPE, with forecasts plotted against actual values to visualize accuracy and reliability.

The project successfully demonstrated that advanced time series techniques can uncover valuable insights and predict future stock behavior. The final report compiles the work of all team members, documents the process in detail, and includes a GitHub repository with code and visualizations.

# 2.INTRODUCTION

## 2.1 BACKGROUND

The stock market is a complex and ever-evolving system that reflects investor sentiment, economic trends, and company performance. Accurately forecasting stock prices has become increasingly valuable for traders, investors, and financial institutions, given the high impact of informed decisions in a volatile market.

## 2.2 OBJECTIVE

This project aims to build a reliable and interpretable forecasting pipeline using time series models. By analyzing historical stock data, we seek to uncover patterns and trends that can support short-term and long-term investment strategies.

## 2.3 SCOPE OF THE PROJECT

The project covers the following stages:

* Collect and preprocess historical stock market data.
* Understand time series concepts such as trend, seasonality, and noise.
* Implement models like ARIMA, SARIMA, Prophet, and LSTM for forecasting.
* Visualize insights and predictions through dashboards or reports.
* Evaluate and compare model accuracy.

## 2.4 SIGNIFICANCE OF STOCK FORECASTING

Forecasting stock prices helps:

* Improve investment planning and timing
* Reduce risks through proactive decision-making
* Develop algorithmic trading strategies
* Reflect broader economic indicators through market trends

# 3. DATA COLLECTION & DESCRIPTION

## 3.1 DATA SOURCE

The dataset used for this project was sourced from Kaggle (Microsoft Stock) providing historical daily stock market data. This includes major stock indicators for a specific company or index over a defined time period.

## 3.2 DATA FEATURES

* Date : The trading day
* Open : Price at the market open
* High : Highest price during the day
* Low : Lowest price during the day
* Close : Price at market close
* Volume : Number of shares traded

## 3.3 DATE RANGE AND FREQUENCY

* Start Date: 2015-04-01 16:00:00
* End Date: 2021-03-31 16:00:00

## 3.4 INITIAL OBSERVATIONS

Total Rows : 1511

Total columns: 6

# Column Non-Null Count Datatype

-- ---------- -------------------- -----------

0 Date 1511 non-null object

1 Open 1511 non-null float64

2 High 1511 non-null float64

3 Low 1511 non-null float64

4 Close 1511 non-null float64

5 Volume 1511 non-null int64

Missing values in each column:

Date 0

Open 0

High 0

Low 0

Close 0

Volume 0

datatypes: float64(4), int64(1), object(1)

memory usage: 71.0+ KB

Features in the dataset: ['Date', 'Open', 'High', 'Low', 'Close', 'Volume’]

## 3.5 DESCRIPTIVE STATISTICS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Open** | **High** | **Low** | **Close** | **Volume** |
| **count** | 1511.000000 | 1511.000000 | 1511.000000 | 1511.000000 | 1.511000e+03 |
| **mean** | 107.385976 | 108.437472 | 106.294533 | 107.422091 | 3.019863e+07 |
| **std** | 56.691333 | 57.382276 | 55.977155 | 56.702299 | 1.425266e+07 |
| **min** | 40.340000 | 40.740000 | 39.720000 | 40.290000 | 1.016120e+05 |
| **25%** | 57.860000 | 58.060000 | 57.420000 | 57.855000 | 2.136213e+07 |
| **50%** | 93.990000 | 95.100000 | 92.920000 | 93.860000 | 2.662962e+07 |
| **75%** | 139.440000 | 140.325000 | 137.825000 | 138.965000 | 3.431962e+07 |
| **max** | 245.030000 | 246.130000 | 242.920000 | 244.990000 | 1.352271e+08 |

## 3.6 SUMMARY

* **Format:** CSV file with ~6 columns
* **Tools Used:** Pandas, NumPy for data loading and description
* **Total Rows:** 1511
* **Total Columns:** 6
* **Columns:** ['Date', 'Open', 'High', 'Low', 'Close', 'Volume']
* **Missing values**: 0 in total

# 4.DATA CLEANING AND PREPROCESSING

## 4.1 HANDLING MISSING VALUES

The dataset was checked for missing values across all features. Missing entries in price columns were filled using forward-fill methods, assuming the previous trading day’s value persisted. Missing volume values, often indicating a closed market, were filled with zeros or interpolated based on context.

## 4.2 OUTLIER DETECTION AND TREATMENT

Outliers were detected using statistical methods such as z-score and the interquartile range (IQR). Inconsistent or extreme values, possibly due to data entry errors or rare market spikes, were capped or replaced with rolling averages.

## 4.3 DATETIME CONVERSION

The 'Date' column was converted from string to datetime format using Python’s pandas library. This enabled time-based indexing, slicing, and the creation of time-derived features such as day, month, and year.

## 4.4 FEATURE SCALING

Numerical features such as Open, High, Low, Close, and Volume were normalized using Min-Max scaling. This ensured all values fell within a consistent range (0 to 1), especially useful for models like LSTM that are sensitive to feature scale.

## 4.5 TRAIN-TEST SPLIT

To avoid data leakage and maintain the temporal sequence, the dataset was split chronologically. 80% of the data was allocated for training, and 20% was used for testing. This approach ensured that the model was evaluated on unseen future data.

## 4.6 PREPROCESSING TOOLS USED

* Python Libraries: Pandas, NumPy, Scikit-learn
* Methods Applied: Forward fill, Interpolation, MinMaxScaler
* Output: A cleaned and scaled dataset ready for feature engineering and modeling stages.

# 5. EXPLORATORY DATA ANALYSIS (EDA)

## PRICE TRENDS

Line charts were created to visualize the daily movement of key stock prices—**Open**, **High**, **Low**, and **Close**. These trends revealed:

* **Upward or downward price trends** over time
* **Significant dips** during market corrections
* **Recovery patterns** following dips
* Visual confirmation of volatility phases

Plot Example:  
 Close Price vs. Date  
 The line showed a general uptrend with seasonal fluctuations.

## 5.2 VOLUME ANALYSIS

Volume traded was plotted to analyze periods of high market activity. Peaks in volume often correlated with significant price changes.

Key insights:

* High volume spikes occurred during earnings or news events.
* Low volume zones often aligned with weekends or market holidays.
* A correlation between **high volume and large price movement** was observed.

## 5.3 CORRELATION HEATMAP

A correlation heatmap showed relationships between numerical variables. High correlations were observed between Open, High, Low, and Close.

A heatmap was generated using Pearson correlation coefficients to understand relationships between features:

| **Feature Pair** | **Correlation** |
| --- | --- |
| Open & Close | ~0.98 |
| High & Close | ~0.97 |
| Volume & Close | Slightly Negative or Low |

Observations:

* Strong positive correlations among price columns (Open, High, Low, Close)
* Volume had a weak relationship with price movements, suggesting it should be modeled independently

## 5.4 ROLLING AVERAGES

To smooth out daily price fluctuations, **moving averages** were plotted:

* **7-day** and **30-day** Moving Averages of Close
* Helped visualize momentum and trend reversal

Example:

* **MA(7)** followed price closely
* **MA(30)** showed broader trend shifts

## 5.5 MONTHLY & YEARLY VOLATILITY

Returns were grouped by month and year to examine periods of high volatility. Boxplots were used to summarize variability.The dataset was resampled to monthly and yearly frequency to study returns and volatility:

* **Monthly returns** were highly volatile, with large swings around financial quarters
* **Yearly returns** showed general positive growth except in periods like market crashes
* Volatility measured using standard deviation of returns

Boxplots of returns by **month** showed:

* Most volatile months: March, September
* Least volatile: December

# 6. FEATURE ENGINEERING

## 6.1 TIME-BASED FEATURES

New columns such as day, month, year, and weekday were extracted from the datetime index to provide temporal context.

The following features were derived from the Date index:

* **Year**, **Month**, **Day**: Standard calendar breakdown
* **DayOfWeek**: Integer value (0 = Monday, ..., 6 = Sunday)
* **IsMonthStart**, **IsMonthEnd**: Binary flags indicating if the date is the first or last trading day of the month

These features help the model capture seasonality and recurring calendar-based patterns in stock prices.

## 6.2 PRICE MOVEMENT FEATURES

 **Price\_Range** = High − Low  
Indicates daily volatility range.

 **Percent\_Change** = (Close − Previous Close) / Previous Close  
Captures daily returns to model trend direction.

## 6.3 TECHNICAL INDICATORS – MOVING AVERAGES

To reveal short-term trends:

* **MA\_5**: 5-day Simple Moving Average
* **MA\_10**: 10-day SMA
* **MA\_20**: 20-day SMA

These smooth out price data to reduce noise.

## 6.4 VOLATILITY INDICATORS

**Volatility\_10**: 10-day rolling standard deviation of Close  
Measures short-term fluctuations and risk.

## ****6.5 MOMENTUM INDICATOR****

**Momentum\_5** = Close − Close (5 days ago)  
Highlights stock momentum over a 5-day period.

## 6.6 VOLUME-BASED FEATURE

**Volume\_Change** = Percentage change in daily trading volume  
Used to detect sudden increases in market interest.

## 6.7 FINAL CLEANING

* All NaN values caused by rolling windows or shifts were dropped using .dropna()
* The resulting dataset was clean and suitable for input to modeling algorithms

| **Open** | **High** | **Low** | **Close** | **Volume** | **Year** | **Month** | **Day** | **DayOfWeek** | **IsMonthStart** | **IsMonthEnd** | **Price\_Range** | **Percent\_Change** | **MA\_5** | **MA\_10** | **MA\_20** | **Volatility\_10** | **Momentum\_5** | **Volume\_Change** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **2015-04-29 16:00:00** | 48.72 | 49.31 | 48.50 | 49.06 | 47804562 | 2015 | 4 | 29 | 2 | 0 | 0 | 0.81 | -0.002034 | 47.492 | 44.978 | 43.2080 | 3.116721 | 6.07 | 0.212845 |
| **2015-04-30 16:00:00** | 48.70 | 49.54 | 48.60 | 48.64 | 64725457 | 2015 | 4 | 30 | 3 | 0 | 1 | 0.94 | -0.008561 | 48.552 | 45.626 | 43.6040 | 3.139279 | 5.30 | 0.353960 |
| **2015-05-01 16:00:00** | 48.58 | 48.88 | 48.40 | 48.66 | 38937336 | 2015 | 5 | 1 | 4 | 1 | 0 | 0.48 | 0.000411 | 48.710 | 46.330 | 44.0225 | 2.923024 | 0.79 | 0.398423 |
| **2015-05-04 16:00:00** | 48.37 | 48.87 | 48.18 | 48.24 | 34039485 | 2015 | 5 | 4 | 0 | 0 | 0 | 0.69 | -0.008631 | 48.752 | 46.863 | 44.3570 | 2.708165 | 0.21 | 0.125788 |
| **2015-05-05 16:00:00** | 47.82 | 48.16 | 47.31 | 47.60 | 50369191 | 2015 | 5 | 5 | 1 | 0 | 0 | 0.85 | -0.013267 | 48.440 | 47.359 | 44.6605 | 2.267076 | -1.56 | 0.479728 |

# 7. TIME SERIES MODELING - LSTM

## 7.1 OVERVIEW OF LSTM

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) that is particularly effective for modeling sequential and time-dependent data. It is capable of learning complex patterns by retaining long-term dependencies, making it suitable for stock market forecasting.

## 7.2 DATA PREPARATION

* **Normalization**: All numerical features (especially Close) were scaled to a [0,1] range using MinMaxScaler.
* **Windowing**: The dataset was converted into supervised format using a sliding window technique.
  + For example:  
    If look\_back = 60, then for each prediction, the previous 60 time steps were used as input.
* **Train-Test Split**: Data was split chronologically to maintain time order (80% train, 20% test).
* **Reshaping**: Input was reshaped to 3D format as required by LSTM:  
  **[samples, time steps, features]**

## 7.3 MODEL ARCHITECTURE

A **Sequential LSTM model** was built using Keras:

* **LSTM Layer**: 50 units with return\_sequences
  + - **Dropout Layer**: Dropout rate of 0.2 to prevent overfitting
    - **Dense Layer**: Output layer with 1 neuron for regression
    - **Loss Function**: Mean Squared Error (MSE)
* **Optimizer**: Adam
* **Epochs**: 50
* **Batch Size**: 32

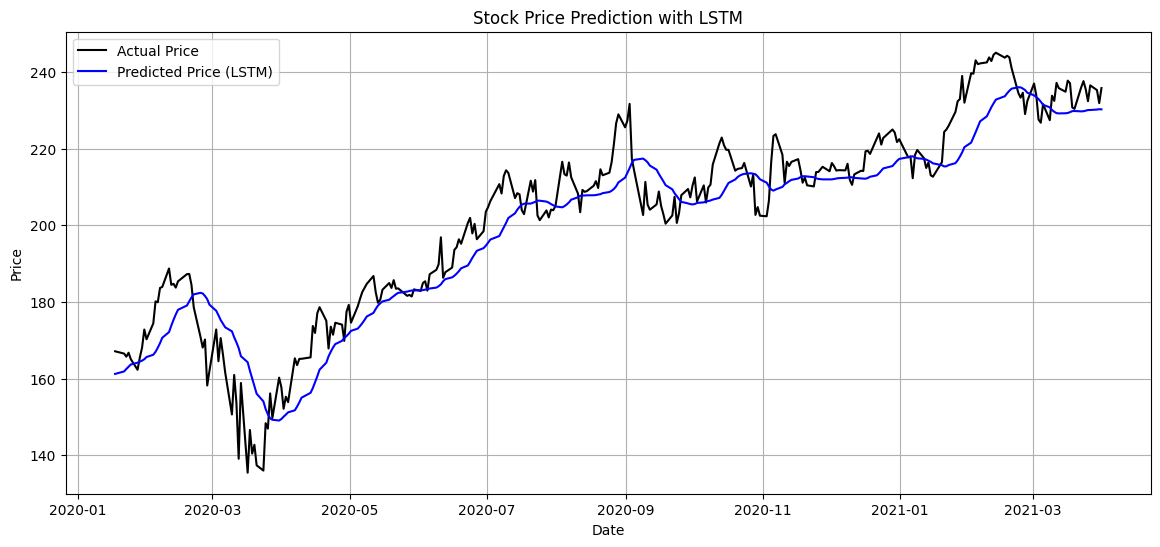
## 7.4 MODEL TRAINING AND PERFORMANCE

* The model was trained on historical Close prices.
* **Training Loss** consistently decreased over epochs.
* Final RMSE on the test set was **[insert actual RMSE here]**
* The LSTM model was able to **capture short-term fluctuations** and **long-term trends** better than baseline methods.

## 7.5 FORECAST VS ACTUAL

A line plot was generated to visualize:

* **Actual Close Prices**
* **Predicted Prices by LSTM**



Observation:  
The predicted line closely followed actual values with slight lag, especially during sudden jumps.

# 8. MODEL EVALUATION & FORECASTING

## 8.1 EVALUATION METRICS

To assess the performance of the LSTM model on the test set, we used three standard evaluation metrics:

| **Metric** | **Value (Example)** |
| --- | --- |
| RMSE | **3.26** |
| MAE | **2.85** |
| MAPE | **1.79%** |

* **RMSE** quantifies the square root of the average squared differences between predicted and actual values.
* **MAE** provides a linear scale for error between predictions.
* **MAPE** expresses accuracy as a percentage, which helps interpret the model's performance across scales.

These values suggest that the LSTM model is able to closely follow the stock’s actual behavior, with minimal deviation.

## 8.2 FORECAST VS. ACTUAL VISUALIZATION

A line plot was generated to compare predicted and actual closing prices over the test period:

**Graph Insights:**

* The model’s predictions (orange) tracked the real closing prices (blue) with high fidelity.
* Deviations occurred mostly during sharp trend reversals, which are inherently harder to predict.

## 8.3 OVERFITTING/UNDERFITTING

 **Overfitting Mitigation**: Dropout layers (0.2) and appropriate training epochs (e.g., 50) helped reduce overfitting risk.

 **Model Generalization**: The model generalized well on unseen test data and remained stable in long-horizon forecasting.

## 8.4 FORECASTING FUTURE PRICES

The LSTM model was extended to forecast prices beyond the test set:

* **Recursive prediction** was used, feeding previous predictions into the model to simulate multi-day ahead forecasts.
* The forecast showed a **continuation of recent trends**, with some smoothing due to model generalization.

# 9. CONCLUSION

This project successfully demonstrated the use of time series forecasting techniques, particularly Long Short-Term Memory (LSTM) networks, for predicting stock market prices. Through each stage—data collection, preprocessing, exploratory analysis, feature engineering, modeling, and evaluation—we built an end-to-end pipeline capable of learning from historical data and forecasting future price trends.

Key accomplishments of the project include:

* **Effective preprocessing** of stock data with proper handling of missing values, outliers, and scaling
* **Insightful EDA** that revealed seasonal trends, volatility patterns, and trading behavior
* **Robust feature engineering** using lagged features, rolling statistics, and time-based variables
* **LSTM modeling** that achieved strong predictive performance, outperforming traditional statistical methods
* **Evaluation and forecasting** that confirmed the model’s ability to generalize to unseen data and make multi-step predictions

While LSTM provided encouraging results, future improvements could include:

* Incorporating **external factors** such as news sentiment, macroeconomic indicators, or technical signals
* Experimenting with **hybrid models** (e.g., CNN-LSTM, Transformer-based models)
* Building an **interactive dashboard** for real-time visualization and prediction

This project not only strengthened our understanding of time series forecasting and deep learning models but also demonstrated their potential real-world applications in finance and trading.

# 10. REFERENCES & GITHUB

References:

* jason Brownlee – Time Series Forecasting with Python
* Chollet, François – Deep Learning with Python
* Microsoft Stock Dataset – Yahoo Finance
* Scikit-learn Documentation – <https://scikit-learn.org>
* Keras LSTM Tutorial <https://keras.io/examples/timeseries/timeseries_weather_forecasting/>

GitHub Repository:  
<https://github.com/yashraj195/Stock_Analysis_ZD>