Est. 1841 | YORK ST JOHN UNIVERSITY

PERFORMANCE EVALUATION AND ANALYSIS OF MACHINE LEARNING ALGORITHMS IN STOCK PRICE PREDICTION

Gaurab Shrestha

230188112

Submitted in Accordance with the Requirements for the Degree of Master of Science (Level 7)

York St John University

Department of Computer Science

Sangita Pokharel

Date of Submission: 28th January 2025

DECLARATION

I certify that this dissertation is my original work and that any language from other sources is enclosed in quotation marks, with proper acknowledgement given for using others' language, ideas, expressions, or writings.

I affirm that I have not plagiarised from any source nor engaged in any form of copying, including outsourcing any part of this work (such as unauthorised proofreading), nor have I fabricated, altered, or exaggerated data in the completion of this dissertation.

This dissertation presents original research not submitted for any other degree at any educational institution.

I state that the inclusion of AI in this dissertation does not compromise the academic integrity of the work and adheres to the regulations established by York SJ University. Any help provided by Artificial Intelligence (AI) tools, including but not limited to research, composition, or editing, has been recognised and utilised following ethical standards.

Gaurab Shrestha

28th January 2025

ACKNOWLEDGEMENT

I would like to express my heartfelt appreciation to everyone who has contributed invaluable support and assistance in preparing this report. Their contributions have made this work possible.

I am truly thankful to my supervisor, Sangita Pokharel, from the Department of Computer Science, for her expert guidance, unwavering encouragement, and willingness to devote time from her busy schedule to reviewing the progress of my dissertation. Her constant inspiration has enabled me to complete this project on time and meet its objectives.

I also sincerely thank Prashant Bikram Shah, Department of Computer Science, for his ongoing motivation and support. Additionally, I would like to convey my heartfelt appreciation to all the faculty members for their encouragement and assistance.

My classmates' unwavering support has been crucial. They have motivated and assisted me directly or indirectly in completing this work. Their constructive feedback and encouragement have been essential to my success in this dissertation project.

Finally, I want to thank everyone who contributed to this report, either directly or indirectly. The help and encouragement I received from various individuals have been vital in bringing this project to fruition.

I appreciate all of you for your support and for being a part of this journey.

Gaurab Shrestha

230188112

gaurab.shrestha@yorksj.ac.uk

ABSTRACT

Forecasting stock price is a difficult yet highly sought-after endeavour in financial markets because of its potential for substantial financial rewards and influence on investment strategies and economic planning. The growing intricacy of financial markets and the abundance of large datasets have rendered machine learning a crucial tool for stock price predictions. This research explores the efficacy of various machine learning techniques in forecasting stock prices, concentrating on their precision and computational efficiency. Utilising two datasets—Google stock prices and Nabil Bank stock prices—the study assesses different machine learning models, including Linear Regression, Support Vector Machines, Random Forest, XGBoost, Long Short-Term Memory, Convolutional Neural Networks, a hybrid CNN-LSTM approach, alongside the statistical SARIMAX and FB Prophet models. Basic ML models like LR and SVM consistently outperformed more complex algorithms, especially on the volatile and noisy Nabil Bank dataset, where advanced machine learning techniques struggled with issues of overfitting and generalisation. The investigation also emphasises the effect of training methodologies, noting that the train_test_split() method from scikit-learn produced superior outcomes compared to training on the initial 80% of data, as the former offers a more balanced representation. Although deep learning models are proficient in capturing non-linear and temporal dynamics, their effectiveness is significantly influenced by the size, quality of the dataset, and meticulous tuning. This study highlights the necessity of matching model selection with the dataset's characteristics and training strategies. It concludes that simpler algorithms frequently yield better results in scenarios involving limited or noisy data and suggests that incorporating external elements like news sentiment or social media trends could improve prediction models in the future.

Keywords: Stock Price, machine learning, NEPSE, SVM, CNN, LSTM.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ML – Machine Learning

DL – Deep Learning

LR – Linear Regression

RF – Random Forest

SVM – Support Vector

XGB – Extended Gradient Boosting

ARIMA – AutoRegressive Integrated Moving Average

FB Prophet – Facebook Prophet

CNN – Convolutional Neural Network

LSTM – Long Short-Term Memory

MSE – Mean Squared Error

RMSE – Root Mean Squared Error

MAE – Mean Absolute Error

1. INTRODUCTION

1.1 Background:

According to (Statista, 2024), the total global market capitalisation of all stock markets exceeded $111 trillion by 2023. This figure represents the overall worth of publicly listed companies worldwide. The U.S. stock market, bolstered by exchanges such as the New York Stock Exchange and Nasdaq, makes up almost 46.2% of the total value of the world's stock markets. In contrast, China's Shanghai Stock Exchange, Shenzhen Stock Exchange, Tokyo Stock Exchange, and London Stock Exchange significantly contribute to global capitalisation.

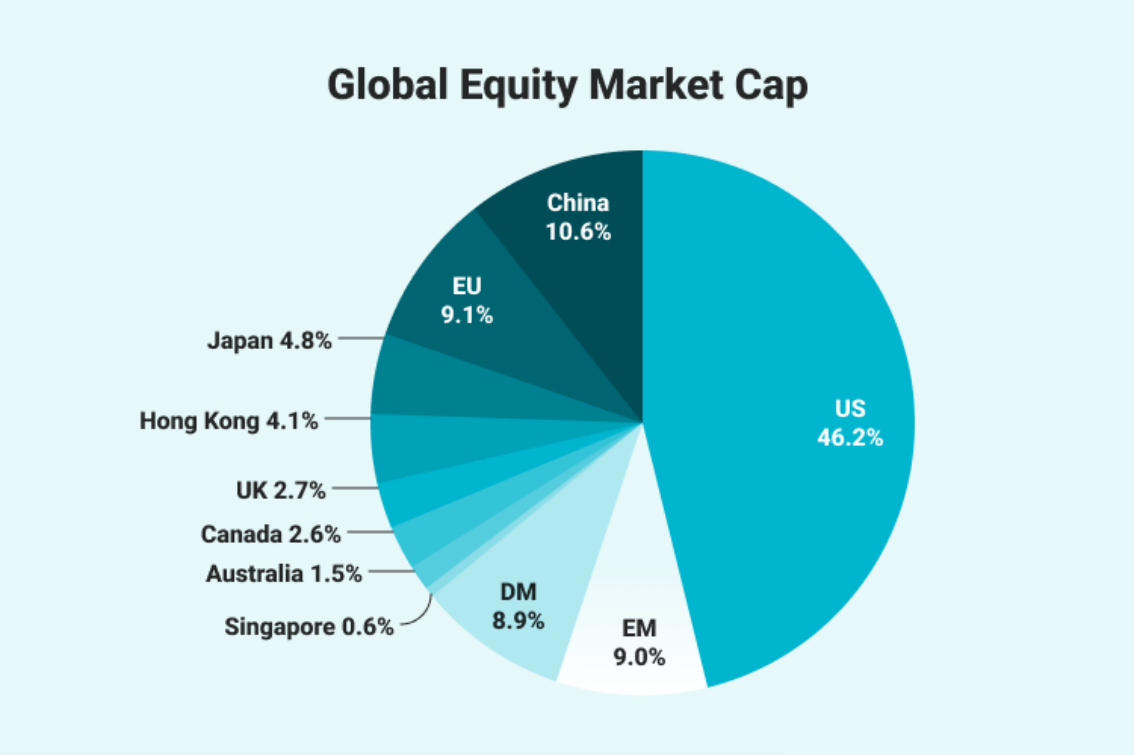


Figure 1: Global Equity Market Cap (Sahm Investor Education, 2023)

In an article (Bajracharya, 2019), he highlighted a growing fascination with stocks and shares, as people from various backgrounds repeatedly inquire, “Which shares do you suggest purchasing in the Nepalese Stock market at this moment?” In the late 1800s, Charles Dow, a co-founder of the Wall Street Journal and Dow Jones & Company, became one of the pioneers in forecasting stock prices. He developed the Dow Theory, which suggests that stock prices move in trends that analysing patterns of market highs and lows can predict. So basically, a stock is simply capital any company raises through the issue and subscription of shares whose value is dynamic throughout time. In contrast, the stock market is the combined platform of several markets and exchangers where various investors sell and purchase according to the availability of those stocks (Tretina, 2024). In simple terms, the stock market is where individuals can buy and trade shares, as well as currencies, equities, derivatives, and more. A nation's financial market plays a crucial role in determining its overall health, helping economists and analysts evaluate the country's economic condition. With the expansion of the broader financial market, people today are increasingly focused on buying and selling stocks,

trading forex, and more. However, many individuals lose money due to a lack of knowledge and understanding of the stock market. This leads us to the concept of stock price prediction. For years, stock market investors have devised strategies to forecast stock prices and maximise profits. The prediction of future stock price movements has intrigued researchers for many years.

Over time, individuals have relied on traditional methods, such as fundamental and technical analysis, to predict stock prices. Fundamental analysis examines factors like a company's intrinsic value in annual and quarterly reports, financial ratios, government policies, the effects of national and international events, and political and economic developments. Key intrinsic values often analysed include Earnings Per Share (EPS), Price-to-Book Value (PBV), Book Value (BV), Price-to-Earnings (P/E) Ratio, recent bonus percentages, and total market capitalisation. It considers the company's value and generally helps anticipate the worth of its assets and the profits it can generate, ultimately influencing the supply and demand of the stock. However, fundamental analysis cannot foresee what might occur in the short term. Technical analysis emphasises trading volume, patterns, charts, trends, daily price movements, and historical price data. Its primary focus is on price movements, providing insights into the supply and demand forces that drive stock prices. This method is primarily used for short-term predictions. One of the significant benefits of using technical analysis over fundamental analysis is the ability to analyse stocks swiftly and identify price targets. Some individuals also buy and sell stocks based on market sentiment, which includes herd behaviour, media stories, rumours, and recommendations. However, certain stocks may sometimes generate hype in the market, influencing prices over shorter or longer periods.

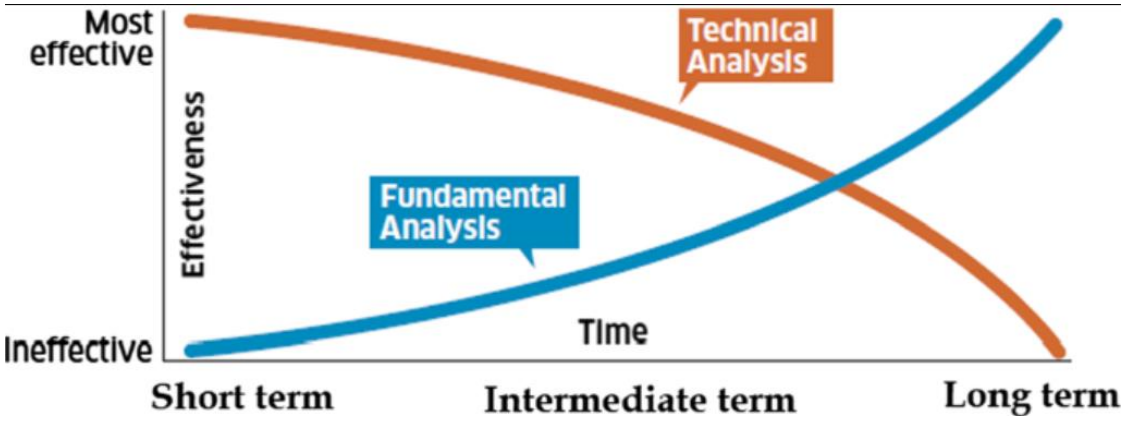


Figure 2: Stock Movement Prediction Methods Efficacy (Kumar, 2021)

With the progress in technology and computing, individuals are forecasting stock prices using machine learning and historical data, which can be seen as a sophisticated form of technical analysis. Machine learning involves the development of computer systems capable of learning and adapting without explicit instructions. It uses algorithms and statistical models to analyse data patterns and make predictions (Crabtree, 2023). Historical stock market data often reveals trends and patterns that can be identified and utilised by machine learning algorithms to predict future stock prices and their fluctuations.

1.2 Problem Statement and Motivation:

Data is rapidly becoming the most valuable resource in the world. Due to its significance, data is sometimes likened to gold. It contains numerous insights that can be derived from it. As society becomes more data literate and acknowledges the importance of data, curiosity continues to grow regarding how historical data can be used in stock market prediction to achieve a precise interpretation. Stock market predictions should be accurate, efficient, and robust, accounting for all variables affecting a stock's value and performance.

Over time, researchers have developed numerous machine learning and deep learning algorithms to address various problems and handle different kinds of data. People have modelled and conducted extensive experiments for stock price prediction, ranging from simple ML algorithms like linear regression algorithms to complex DL hybrid models. Researchers are continually refining the algorithms based on their predictive performance and computational efficiency in stock pricing. Simple linear regression is straightforward to implement and interpret and can predict stock prices close to their actual values, although not with high accuracy. Support vector machines can be effective in high-dimensional spaces and non-linear boundaries using kernel functions, but they require careful selection of kernel and regularisation parameters. The random forest algorithm helps reduce overfitting but is more challenging to interpret. Neural networks such as CNNs, LSTMs, and others can model highly complex and non-linear relationships, including time series, but demand significant computational power and extensive training data.

Furthermore, stock prices are influenced by time, making time series analysis a commonly employed technique for effectively analysing such time-dependent data. The Autoregressive Integrated Moving Average (ARIMA) model, introduced by Box et al. (2015), is one of the most well-known approaches for modelling time series data and is widely applied in real-world scenarios. However, as a linear model, ARIMA does not account for volatility or fluctuations in the underlying variance of stock prices. This limitation renders ARIMA ineffective for forecasting stock price data, particularly fluctuating volatilities. In recent times, LSTM models have gained popularity for predicting stock prices.

Stock market indices are highly volatile, which impacts investor confidence. Stock prices are regarded as dynamic and susceptible to rapid changes due to the nature of the financial sector, influenced by a combination of known factors (such as the previous day's closing price and P/E ratio) and unknown variables (like election results and rumours). ML methods used for stock price prediction analyse historical data to estimate the likelihood of future events or forecast trends. This process involves identifying data patterns, integrating current and past information, and applying predictive models that best align with the data. To achieve optimal performance, these models are trained using various algorithms and fine-tuned with parameters, including hyperparameters.

Many efforts have been made to predict stock prices using machine learning (ML), with numerous researchers developing different estimation models. However, ML models' predictions are generally less reliable than those made by humans. Despite this, these models can act as augmented intelligence, assisting investors in making informed decisions about

which stocks to invest in, potentially leading to higher returns and reduced losses (Kumar, 2021).

The emergence of ML and DL has significantly enhanced the accuracy of models for predicting stock prices. However, with the introduction of various machine learning and deep learning algorithms, their performance and predictive accuracy remain uncertain. Therefore, a comprehensive study is essential to evaluate and analyse the performance of different machine learning algorithms regarding prediction accuracy and computational efficiency. Additionally, it is crucial to examine how factors such as the characteristics of the stock dataset, data splitting methods, and stock trends impact the performance of these algorithms.

1.3 Objectives:

Despite significant progress, numerous obstacles remain in employing machine learning to forecast stock prices. These challenges include selecting appropriate algorithms, feature engineering, ensuring data quality, and evaluating model performance. Furthermore, detailed comparative studies are needed to assess the effectiveness of ML models under various market conditions and investment timeframes.

This dissertation explores the feasibility of predicting stock prices using historical data combined with machine learning and deep learning algorithms. It also evaluates and analyses these algorithms' prediction accuracy and computational efficiency.

The primary objective of this dissertation is to:

"**Evaluate and analyse the performance of ML algorithms in stock price prediction".**

Some of the objectives of the dissertation are listed below:

- Evaluate the effectiveness of various ML techniques in predicting stock prices.
- Compare the accuracy and reliability of ML models with traditional statistical methods.
- Identify the most effective machine learning model for stock price forecasting.
- Assess the impact of the data used for training and evaluating the machine learning models.

1.4 Research Questions and Hypothesis:

This dissertation seeks to answer:

i. How effectively and accurately can different ML algorithms and models predict future stock price trends?
ii. Which algorithm yields the highest accuracy and requires the least computational time for stock price prediction?
iii. What impact will the data utilised for training and evaluating the machine learning and deep learning models have?

Also, this dissertation seeks to test these three hypotheses:

a. **Prediction Accuracy Hypothesis:** Different machine learning algorithms exhibit varying levels of accuracy when predicting stock prices.
   - **Null Hypothesis (H0):** There is no significant difference in the prediction accuracy among different machine learning algorithms.
   - **Alternative Hypothesis (H1):** Different machine learning algorithms have significantly different prediction accuracy.

b. **Model Efficiency Hypothesis:** Some machine learning algorithms are more computationally efficient than others in stock price prediction.
   - **Null Hypothesis (H0):** All ML algorithms are similar in computational efficiency when used to predict stock prices.
   - **Alternative Hypothesis (H1):** The computational efficiency of different machine learning algorithms differs significantly.

c. **Data Effect Hypothesis:** The data used to train and test machine learning models for stock price prediction significantly impacts the results.
   - **Null Hypothesis (H0):** Data has no significant effect on the prediction accuracy of machine learning algorithms.
   - **Alternative Hypothesis (H1):** Data significantly affects the prediction accuracy of machine learning algorithms.

1.5 Dissertation Structure:

The remainder of this dissertation is organised as follows:

Chapter 1. Introduction - This section gives a general overview of the research, including background, problem statement, objectives, hypothesis, etc.

Chapter 2. Literature Review - provides a comprehensive review of recent research and practices on stock price prediction using different machine learning methodologies.

Chapter 3. Methodology - outlines the data collection and pre-processing procedures with detailed research methodology, including algorithm selection, model development, and evaluation metrics.

Chapter 4. Results. Analysis and Discussions - presents the comparative analysis's empirical results, findings, implications and limitations.

Chapter 5. Conclusions and Future Works - summarises the main contributions and proposes areas for future research.

2. LITERATURE REVIEW

Machine learning (ML) algorithms for predicting stock prices have garnered significant attention in academic and financial sectors. Forecasting stock prices is a complex endeavour shaped by market sentiment, economic conditions, and global events. Over the years, numerous researchers have proposed solutions to address these challenges. Although traditional methods such as fundamental and technical analysis have been widely employed for financial forecasting, they frequently fall short in capturing the non-linear patterns present in stock market data. Machine learning models, in contrast, provide the advantage of learning from extensive datasets and uncovering intricate patterns. This literature review explores existing research on the efficacy of various machine learning algorithms in stock price prediction and highlights the methodologies employed to assess and compare their performance.

2.1 Traditional Methods and Their Limitations

According to (Shrestha, 2021), there are lots of researches that found Fundamental Analysis as the most widely adopted approach for predicting the stock price where investor focuses on information such as the intrinsic value of the corporation published in the annual report, quarterly reports, financial ratios, governmental policies, national and international events, political and economic development. The history of fundamental analysis can be traced to the 1934 book Security Analysis by Benjamin Graham and David Dodd (Graham & Dodd, 1934) , which established the intellectual foundation for what would later be known as value investing. Many studies have been conducted to formalise and expand upon the stock selection principles outlined in this book.

Another approach, Technical Analysis, is almost different from the previous one. This involves using charts, technical indicators and historical price patterns to predict future price movements. According to (Kabir, et al., 2023)the authors, while traditional methods of forecasting stock prices, such as fundamental and technical analysis, can provide valuable insights, they may overlook short-term maker dynamics and external events, limiting their predictive capabilities. Also, his paper, which discusses theoretical and empirical literature on the *efficient market hypothesis,* found that all new information will be reflected in asset prices immediately without delay. So, changes in future prices have nothing to do with past and present information. From their perspective, predicting future asset prices is considered impossible. On the other hand, many studies have tried to prove adequat*e market hypothesis* experimentally, and empirical evidence shows that the stock market can be predictable in some ways.

2.2 Machine Learning Approach in Stock Price Prediction:

In recent years, the application of machine learning for stock price prediction has gained significant attention. Extensive research has been conducted in this area, with many studies demonstrating that machine learning methods can effectively predict stock prices using historical data.

2.2.1 Basic Machine Learning Approaches for Prediction

(Pathak & Pathak, 2020) conducted research on stock price prediction using fundamental machine learning algorithms and published a paper titled "Study of Machine Learning Algorithms for Stock Market Prediction." They used algorithms such as Random Forest, Support Vector Machine, KNN, and Logistic Regression to evaluate their performance with metrics like accuracy, recall, precision, and F-score. Their experiment revealed that Random Forest achieved the highest accuracy at 80.7%, followed by Logistic Regression with an accuracy of 78.6%.

Similarly, (Kabir, et al., 2023) presented a research paper on stock price prediction using machine learning. Basic algorithms such as linear regression, Support Vector Machine, Decision Tree, and Random Forest were utilised. They studied the behaviour of these algorithms and analysed the impact of different features on prediction accuracy. The results highlighted the potential of machine learning in the financial sector and emphasised the significance of feature selection and engineering.

The author (Shrestha, 2021) discussed stock price prediction for NEPSE (Nepal Stock Exchange) using machine learning algorithms such as Multi-Layer Perceptron, Support Vector Regressor, and Multiple Linear Regression. The research also identified several unpredicted factors, including foreign exchange rates, the base interest rate set by **Nepal Rastra Bank**, and the prices of commodities like gold, silver, and petroleum products, influencing stock market trends. The author concluded that while these unpredicted factors are correlated with stock prices, their significance is minimal and found that Multiple Linear Regression outperformed all other models in every area.

(Khanna, et al., 2022), in their paper, they studied four prediction models—LSTM, XG Boost, SVM, and Random Forest—and evaluated their accuracy and F1 scores. The research found that using continuous data with 10 technical indicators produced the best performance with the Random Forest classifier, achieving the highest accuracy of 84.89% and the highest F1 score of 89.33%. The experiment also provided insights into why the Naïve Bayes classification model is unsuitable for stock price prediction.

2.2.2 Deep Learning Approaches, along with other ML Algorithms

The research (Mehtab & Sen, 2019) proposed a hybrid approach for predicting stock price movement using machine learning, deep learning, and natural language processing. Various classification techniques were employed to predict price movement patterns, while regression models were used to predict the closing price. The study focused on the NIFTY 50 index values from India's National Stock Exchange (NSE), collecting daily price data over three years. LSTM-based deep learning models were developed to predict stock closing prices and compared with the prediction accuracies of other machine learning algorithms. Additionally, the model was enhanced by integrating a sentiment analysis module using Twitter data to link public sentiment to market trends. The proposed scheme was tested using cross-validation with Self-Organizing Fuzzy Neural Networks (SOFNN). In a follow-up study (Sen, et al., 2021), the same stock was analysed using four deep learning-based regression models built on Long Short-Term Memory (LSTM) networks, employing a novel walk-forward validation approach.

The results showed that the LSTM-based univariate model, which used one-week prior data to predict the next week's open value of the NIFTY index, was the most accurate.

In a study conducted (Rizvi & Khalid, 2024), the authors explored stock market analysis using computer models to predict stock price changes, focusing specifically on the AAPL dataset from Apple Inc. They compared various models—Multi-Layered Perceptron (MLP), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU)—to provide a comprehensive performance analysis of the deep learning approach. The study concluded that LSTM demonstrated the best predictive capability by capturing complex patterns due to its ability to handle long-term dependencies. In contrast, the MLP model struggled to capture the data's intricate patterns. The findings showed that the LSTM model achieved the highest accuracy when the number of epochs was set to 50, with lower MAE, MSE, and RMSE values at this epoch count.

The research (Saud & Shankya, 2019) examined two major commercial banks listed on the Nepal Stock Exchange (NEPSE) and compared the stock price prediction performance of the GRU model with widely used gradient descent optimisation techniques: Momentum, RMSProp, and ADAM. The study concluded that GRU with ADAM provided a more accurate and consistent approach to predicting stock prices. The authors used MAPE as the performance indicator, with the best MAPE achieved by GRU with ADAM at 1.15 for EBL and 2.13 for HBL stock data. GRU with RMSProp predicted the stock prices for EBL and HBL with MAPE values of 2.27 and 3.08, respectively, while GRU with Momentum resulted in MAPE values of 2.90 for EBL and 4.59 for HBL.

The paper by (K, et al., 2021) compared the effectiveness of various Artificial Neural Networks, such as LSTM, RNN, SVM, and other deep learning and machine learning algorithms, for stock price prediction, particularly in the context of long-term investing. It presents a comparative stock price prediction analysis using fundamental and technical analysis alongside machine learning techniques. The paper concludes that fundamental and technical analysis should be integrated for an effective market strategy in predicting stock prices, as this approach helps investors make informed long-term investment decisions.

In the research study (Huang, et al., 2021), the authors analysed 22 years of quarterly stock financial data and investigated three machine learning algorithms: Feed-forward Neural Network (FNN), Random Forest (RF), and Adaptive Neural Fuzzy Inference System (ANFIS) for stock prediction based on fundamental analysis. Additionally, RF-based feature selection and bootstrap aggregation were applied to enhance model performance and combine the predictions from different models. The study concluded that the RF model provided the best prediction results, and feature selection improved the performance of both FNN and ANFIS. The paper found that the aggregated model outperformed all baseline models, demonstrating that ML models can assist fundamental analysts in making informed stock investment decisions.

2.2.3 Hybrid Models and Ensemble Approaches

(Chen, 2020) in his research paper “Using Machine Learning Algorithms on Prediction of Stock Price”, three machine learning algorithms—LSTM, CNN, and SVR—were compared.

The paper also introduced a hybrid model combining CNN and LSTM (CNN-LSTM) and applied it to four stock datasets: Apple, Mastercard, Ford, and ExxonMobil. The study concluded that a machine learning model trained on one stock dataset could forecast other stock prices with slightly reduced accuracy. The paper summarised the model accuracies using mean absolute percentage error (MAPE), showing that combining CNN and LSTM improved accuracy compared to LSTM alone. Additionally, the SVR model was found to predict stock prices with the highest accuracy, likely due to the more comprehensive set of features used in the SVR model. The paper also included the kurtosis and skewness metrics of the financial data. Kurtosis measures the "tailedness" of the return distribution, indicating the presence of outliers and extreme returns (Kenton, 2024). In contrast, skewness measures the asymmetry of the return distribution around its mean, reflecting whether data points are concentrated on one side of the mean (Taylor, 2024).

The authors (Sonkavde, et al., 2023) discussed forecasting stock market prices using machine learning and deep learning models, presenting a systematic review, performance analysis, and implications. They provided an overview of the machine learning and deep learning models applied in the financial sector, along with a generic framework for stock price prediction and classification. Additionally, they implemented an ensemble hybrid (stacked ensemble model) combining Random Forest, XG-Boost, and LSTM to forecast the stock prices of TAINIWALCM and AGROPHOS. They performed a comparative analysis with other popular models. The authors evaluated the models using Root Mean Square Error (RMSE) and R-squared (R^2) scores. They found that the hybrid (stacked ensemble) model achieved the lowest RMSE and the highest R^2 score, making it the best model for stock price prediction. For the stacked ensemble model, they used Mean Squared Error (MSE) as the loss function, the ADAM optimiser, and a maximum of 50 epochs.

The paper presented by (A, et al., 2023), also illustrates that a CNN-LSTM Neural Network can effectively track the patterns and characteristics of stock data. It proposed that by extracting features from stock data and transforming them into tensors, it is possible to obtain these features and subsequently feed them into LSTM neural networks to identify patterns, thus enabling stock market predictions over a specified timeframe. The paper outlined the attributes of a custom CNN-LSTM model and tested its performance, revealing a high accuracy even when trained on real-time stock market data. The Mean Square Error (MSE) score was utilised to evaluate the performance of various machine learning algorithms, with the CNN-LSTM model achieving an MSE score of 0.035. In contrast, the LSTM and XG-Boost models recorded MSE scores of 0.045 and 0.047, respectively.

2.2.4 Performance Metrics for Evaluation

Multiple studies have emphasised the need for proper evaluation metrics to assess machine learning models in stock price prediction. (Guresen, et al., 2011) Examine various metrics such as Mean Absolute Error (MAE), mean squared error (MSE), root mean squared error (RMSE), and R-squared (R^2) to evaluate predictive accuracy. As mentioned in the previous section (Pathak & Pathak, 2020), recall, precision, and F1 scores were employed to assess predictive accuracy. (A, et al., 2023) utilised mean square error (MSE) as a performance metric while

(Khanna, et al., 2022) focusing on predictive accuracy and the F1 score. Other researchers also adopted MAE and RMSE, among others, as performance metrics to gauge predictive accuracy.

2.3 Research Gap:

Following the advent of machine learning for forecasting stock prices, most researchers concentrated on enhancing specific machine learning algorithms. They thoroughly explored various methods for predicting stock prices and proposed numerous advanced predictive models. The research papers studied for the literature review mainly focused on building models and calculating the prediction accuracy. Most of them compared a few of the algorithms in the same category based on the prediction accuracy, which is insufficient to understand the algorithm's behaviour in predicting the stock price in terms of accuracy and computational time. Various researchers utilised performance evaluation metrics, which were inadequate for comparing and analysing the effectiveness and prediction accuracy of different machine learning algorithms in forecasting stock prices.

To address the research gap mentioned, we plan to develop various machine learning models across several categories: basic machine learning algorithms, ensemble learning algorithms, time series forecasting algorithms, and deep learning algorithms, including hybrid models. For essential machine learning, we will use LR and SVM. In the ensemble learning category, we will implement the RF Regressor and the XGB algorithm. ARIMA will be chosen for time series forecasting, while the deep learning models will include CNN, LSTM, and a hybrid CNN-LSTM model. We intend to evaluate and compare these models based on prediction accuracy and computational time, using consistent performance evaluation metrics to assess their effectiveness.

3. Methodology

3.1 Project Overview

Forecasting stock prices is challenging due to financial markets' inherent volatility and complexity. However, with the continued advancement of machine learning (ML), there is increasing interest in using these algorithms to predict stock price trends by analysing historical data and identifying patterns that traditional methods might overlook. This project aims to systematically evaluate the performance of various ML algorithms in stock price prediction, comparing their accuracy, reliability, and practical application for both short-term and long-term forecasts.

The research will use stock market data to explore a range of algorithms—from basic models like linear regression and SVM to more advanced techniques such as CNN and LSTM networks. The project will assess how effectively each model can predict future stock movements by examining historical stock prices.

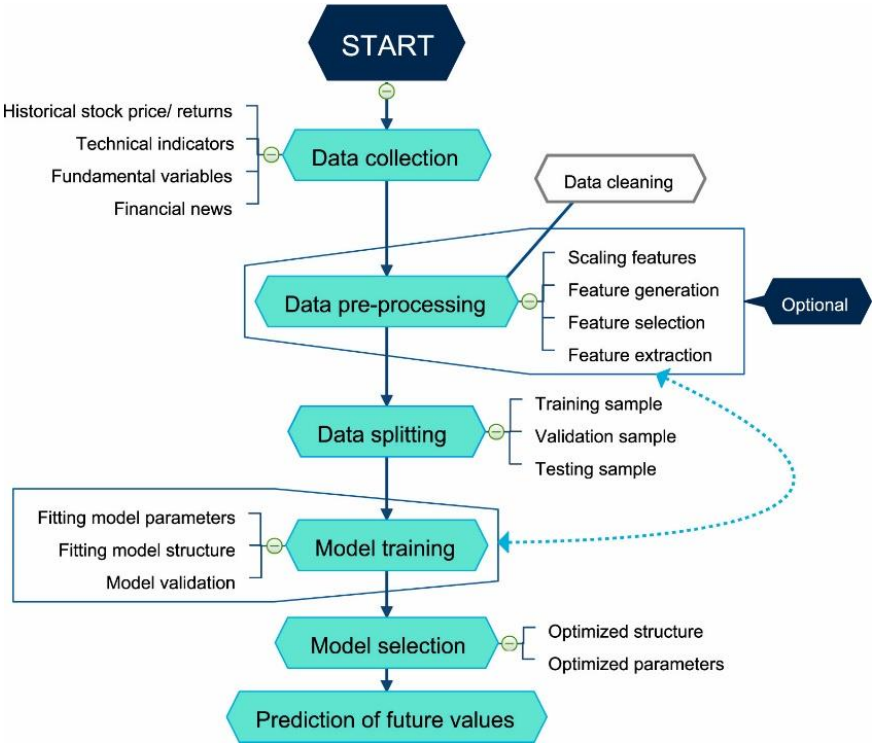


Figure 3: Workflow of Stock Price Prediction Model using Machine Learning (Kumbure, et al., 2022)

A thorough comparison is carried out using performance metrics such as Mean Absolute Error, Mean Squared Error, Root Mean Squared Error, and training time data to assess the predictive accuracy of these machine learning algorithms. Additionally, factors like computational efficiency, training duration, and model interpretability will be crucial in determining which machine learning algorithms are most effective for forecasting stock prices under various conditions. This performance evaluation provides valuable insights into the strengths and limitations of different ML models, contributing to the future improvement and application of predictive algorithms in financial markets. Ultimately, the results will provide important recommendations for traders, investors, and data scientists engaged in this domain.

3.2 System Design and Methodology Overview

The system design for this study, **"Performance Evaluation and Analysis of Machine Learning Algorithms in Stock Price Prediction",** emphasises developing a thorough framework for gathering data, training models, assessing their performance, and contrasting various algorithms. The objective is to forecast stock prices and evaluate the effectiveness of different machine learning algorithms based on prediction accuracy, performance metrics, and computational efficiency. This offers a comprehensive insight into their advantages and disadvantages in this field.

Designing a machine learning-based system for stock price prediction involves several key stages, including data acquisition, processing, model training, evaluation, and deployment. The following diagram illustrates the primary components of a typical system's architecture.

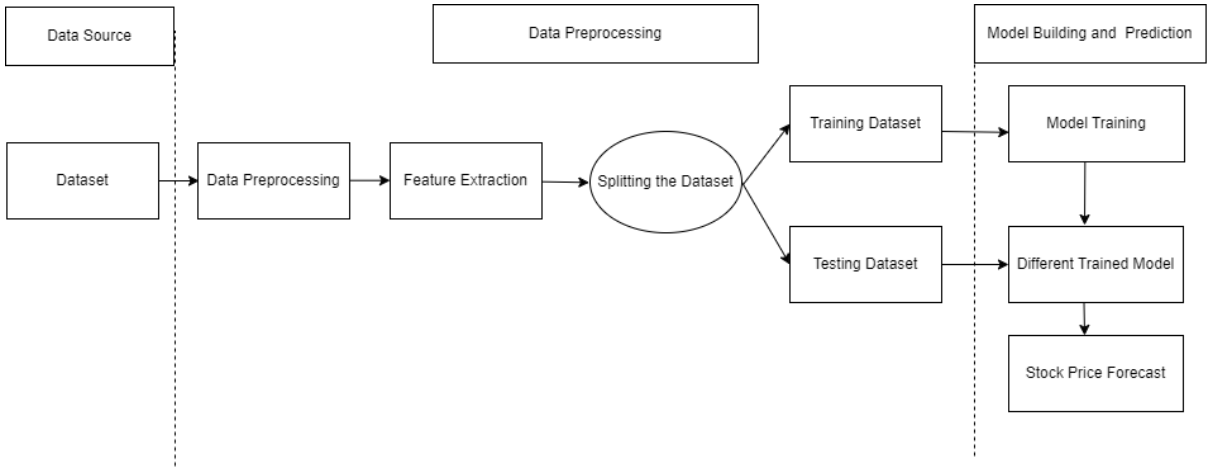


Figure 4: System Design of Stock Price Prediction

Our research project is developed and implemented following this system design. Overall, the system design of our project consists of data collection, data processing, model building, testing and predicting the future stock price.

3.2.1 Project Flowchart

The figure below shows the methodology and flow of our research project. As shown in the literature review section, our research started by understanding the problem by reviewing the system's existing architecture. We then collected the required data from different sources, performed the required data preprocessing, and extracted features. Finally, we trained the model with the training data to predict stock prices.

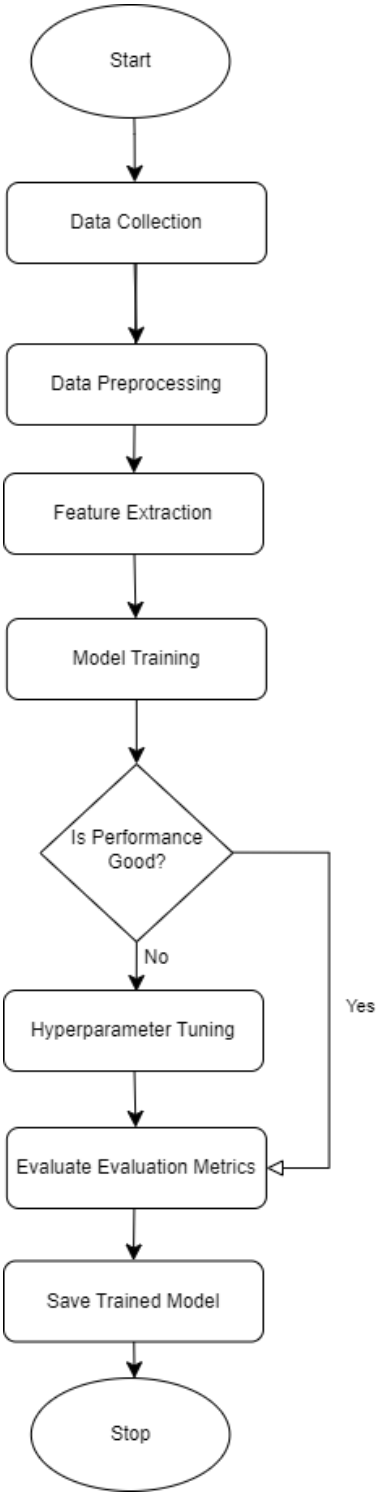


Figure 5: Model Training Flowchart of Stock Price Prediction

3.3 Data Understanding

According to (Ronaghan, 2019), quality data is fundamental to any machine learning and data science engagement to get actionable insights where appropriate data must be sourced and cleansed. Understanding data is a vital phase in the machine learning workflow that encompasses gathering, examining, and interpreting data to derive insights that guide the creation of predictive models. This stage is important since the data's quality, relevance, and

attributes greatly affect the effectiveness of ML algorithms. For our research project, we adhered to the subsequent steps for data comprehension:

3.3.1 Data Collection

For this research project, we used two datasets: google.csv (Google Stock Price Dataset) and nabil.csv (Nabil Bank Stock Price Dataset). These datasets are from different stock markets worldwide.

a. Google Stock Price Dataset

We took the Google Stock Price Dataset (google.csv) from the Yahoo Finance website, found on the link (Yahoo Finance). Google's parent company, Alphabet Inc., is registered on the NASDAQ stock exchange in the United States. Alphabet has two publicly traded shares: GOOGL (common shares with voting rights) and GOOG (shares with no voting rights). Both GOOGL and GOOG are listed on NASDAQ under their respective ticker symbols. NASDAQ is a major stock exchange known for listing technology companies, including other giants like Apple, Microsoft, and Amazon. Our research used the GOOGL share price, a class A share with voting rights. The share prices are in US dollars ($). It contains the Google Stock Price Data from 19th August 2004 to 25th September 2024. This dataset has 7 columns: **Date, Open, High, Low, Close, Adj. Close, Volume.** Due to the availability of high-quality data with a large historical dataset and a globally recognised company with strong financials, followed widely by investors, analysts and institutions, we chose this dataset for our research project. The dataset can be downloaded from the Yahoo Finance website or its Python library, '**yfinance**'.

```
1  import yfinance as yf
2  df = yf.download('GOOGL', start='2004-08-19', end=datetime.today())
3  df.head(2)
```

```
[**********100%**********]  1 of 1 completed
```

Date	Open	High	Low	Close	Adj Close	Volume
2004-08-19	2.502503	2.604104	2.401401	2.511011	2.504808	893181924
2004-08-20	2.527778	2.729730	2.515015	2.710460	2.703765	456686856

Figure 6: Code to Download Google Stock Price Data from Yahoo Finance

b. Nabil Bank Stock Price Dataset

Nabil Bank Stock Price Dataset (nabil.csv) is taken from the Share Sansar website, which can be found on the link (Share Sansar). Nabil Bank is one of the largest commercial banks in Nepal, and investors widely follow its stock due to its stable performance and regular dividends. Nabil Bank Stock is registered under the Nepal Stock Exchange (NEPSE), the only stock exchange in Nepal. The dataset contains the data from 20th March 2011 to 25th September 2024. There are 8 columns in the dataset: **Date, Open, High, Low, Ltp, Change, Qty, Turnover**. The share prices are in Nepalese currency (NPR).

The Nepalese stock dataset and the Nabil Bank Stock Price dataset are not readily available on the Internet, and we needed a dataset in the proper format. Since the Share Sansar website did not have a downloadable dataset file, we scrapped the dataset from the website. We got Share Sansar's approval to scrape the data from the website as part of ethical considerations.

For data scrapping, we used **Selenium** because it can extract dynamic information from websites that require user engagement, such as completing forms, pressing buttons, or moving between pages. The figure below shows the complete setup for web data scrapping using Selenium. The required libraries and Selenium web drivers were imported, and the driver instance, which will handle the browser and the requests, was created. We used VS code because it was easier to write, debug, and navigate the code.

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
import time
import pandas as pd

url = 'https://www.sharesansar.com/company/nabil'

driver = webdriver.Chrome()
driver.get(url)

price_history = driver.find_element(By.ID, 'btn_cpricehistory')
price_history.click()
```

Figure 7: Importing and Setting Up Selenium Driver

```python
while True:
    print(f"Scraping page {page}...")

    tbody = driver.find_elements(By.TAG_NAME, 'tbody')
    for table_rows in tbody:
        table_data = table_rows.find_elements(By.XPATH, './tr[@role="row"]')
        for data in table_data:
            SN.append(data.find_element(By.XPATH, './td[1]').text)
            Date.append(data.find_element(By.XPATH, './td[2]').text)
            Open.append(data.find_element(By.XPATH, './td[3]').text)
            h = (data.find_element(By.XPATH, './td[4]').text)
            High.append(h)
            Low.append(data.find_element(By.XPATH, './td[5]').text)
            Ltp.append(data.find_element(By.XPATH, './td[6]').text)
            Change.append(data.find_element(By.XPATH, './td[7]').text)
            Qty.append(data.find_element(By.XPATH, './td[8]').text)
            Turnover.append(data.find_element(By.XPATH, './td[9]').text)

    next_button = driver.find_element(By.ID, 'myTableCPriceHistory_next')
    next_button_class = next_button.get_attribute('class')

    if 'disabled' in next_button_class:
        print("Reached the last page.")
        break
    next_button.click()

    page += 1
    time.sleep(1)
```

Figure 8: Code for Scrapping Nabil Bank Stock Price Data

The scrapped data was stored in CSV format. The complete code for data scrapping can be found in the GitHub repository with the link (webscrapping-using-beautifulsoap-selenium).

The dataset comes with specific usage rights that impose restrictions on commercial use and public redistribution, and it is intended for development and research purposes.

3.3.2 Data Wrangling (Data Preprocessing and Feature Engineering)

As part of data preprocessing and feature engineering, we explored the dataset's foundational aspects, including the loading process, understanding its overall dimensions and gaining insights into variables at our disposal. We used Google Colaboratory for data preprocessing, feature engineering, EDA and model building.

3.3.2.1 Overview of the Datasets

This section contains the summary of the dataset. Regarding the data dimension of both datasets, there are seven variables (columns) and 5060 observations (data or rows) in the Google dataset. In contrast, the Nabil dataset has eight variables and 3083 observations.

The 7 variables of the Google dataset are:

a. **Date:** This column indicates the calendar date the stock data is recorded.
b. **Open:** This column indicates the stock's initial price for a trading session.
c. **High:** The high price refers to the maximum value at which the stock was traded during a particular session, reflecting the highest price achieved that day.
d. **Low:** The low price denotes the minimum value the stock reached during the session, marking the lowest price for that day.
e. **Close:** The closing price shows the final price at which the stock was traded at the end of the session, representing its last value before the market closed.
f. **Adj Close**: The adjusted closing price accounts for corporate actions such as dividends, stock splits, and new stock offerings, which may affect the stock's price but are not directly related to its performance. The adjusted close is often used to assess the stock's performance over time.
g. **Volume:** Volume reflects the total shares traded during a specific period. It indicates the level of market activity and liquidity for the stock. Higher volume often suggests greater investor interest, while lower volume may indicate less active trading.

For the Nabil dataset, the first four columns are the same as those for the Google dataset. The rest of the columns of the Nabil dataset are:

a. **Ltp:** It is the last transaction price at the end of the trading session, the same as the Close column of the Google dataset.
b. **% Change:** The percentage difference between the opening and closing prices.
c. **Qty:** It is the same as the volume column of the Google dataset, which shows the total number of shares traded in a specific period.
d. **Turnover:** It is the total value of shares traded rather than the number of shares.

3.3.2.2 Descriptive Statistics

We provided a comprehensive summary of each variable in the dataset, encompassing key statistical measures such as length, class, mode, minimum value, median, maximum value, mean and quartile values. This thorough examination allowed us to grasp the range of our

variables and identify potential outliers and missing values. Through this analysis, we could affirm the absence of any missing values in our dataset.

```
1  google.describe()
```

	Date	Open	High	Low	Close	Volume
count	5060	5060.000000	5060.000000	5060.000000	5060.000000	5.060000e+03
mean	2014-09-05 17:29:50.039525632	46.181969	46.668344	45.719767	46.201323	1.149868e+08
min	2004-08-19 00:00:00	2.470490	2.534002	2.390042	2.490913	1.584340e+05
25%	2009-08-26 18:00:00	13.075677	13.181156	12.924120	13.078167	2.708111e+07
50%	2014-09-06 12:00:00	27.155944	27.410246	26.969705	27.147716	5.426358e+07
75%	2019-09-16 06:00:00	61.206749	61.680125	60.516376	61.002500	1.415416e+08
max	2024-09-25 00:00:00	191.750000	193.309998	190.619995	192.660004	1.650833e+09
std	NaN	44.410807	44.898584	43.973791	44.438331	1.494489e+08

Figure 9: Descriptive Statistics of Google Dataset

```
1  nabil.describe()
```

	Date	Open	High	Low	Close	Volume
count	3083	3083.000000	3083.000000	3083.000000	3083.000000	3083.000000
mean	2018-01-07 23:14:41.608822528	1294.111288	1307.948362	1277.907168	1293.011482	31310.902692
min	2011-03-20 00:00:00	421.100000	425.000000	419.000000	422.000000	24.000000
25%	2014-08-16 00:00:00	830.000000	838.000000	820.000000	830.000000	2437.500000
50%	2018-01-17 00:00:00	1162.000000	1178.000000	1145.000000	1161.000000	7090.000000
75%	2021-06-16 12:00:00	1760.500000	1778.000000	1744.500000	1760.500000	41572.500000
max	2024-09-25 00:00:00	2720.000000	2800.000000	2700.000000	2750.000000	705430.000000
std	NaN	571.436265	579.153186	566.254316	573.467567	54132.495999

Figure 10: Descriptive Statistics of Nabil Bank

3.3.2.3 Data Cleaning

We transformed our data into structured formats or types that facilitate convenient data processing. This step is crucial for effective data cleaning and ensures optimal preparation for subsequent analyses.

A. Irrelevant Features

As we had to perform the exploratory data analysis on both datasets, we needed the same columns, so we dropped the ‘Adj Close’ column from the Google dataset as it was almost the same as the ‘Close’ column. Also, we dropped the ‘% Change’ and ‘Turnover’ columns from

the Nabil dataset. We also changed the column name of the Nabil dataset from 'Ltp' to 'Close' and 'Qty' to 'Volume'. The figure below shows the code to change the names and drop the unwanted columns.

```
[33]  1  google.drop('Adj Close', axis = 1, inplace=True)
      2  nabil.drop(['% Change', 'Turnover'], axis=1, inplace=True)
      3
      4  nabil = nabil.rename(columns = {
      5      'Ltp': 'Close',
      6      'Qty': 'Volume'
      7  })
```

Figure 11: Dropping and Renaming the Column Names

B. Missing values

Missing value refers to missing data in a specific variable or observation within a dataset. The overall overview of both the datasets is presented in the table:

Table 1: Overview of the Dataset

Columns	Description	Count		Missing Values	
Stock Price Dataset		Google	Nabil	Google	Nabil
Date	Date on which the stock is documented	5060	3083	0	0
Open	Initial listed price of the stock for a trading day	5060	3083	0	0
High	Peak price of the stock exchanged during a particular trading day	5060	3083	0	0
Low	Lowest price of the stock exchanged during a specific trading day	5060	3083	0	0
Close	Final trading price of the stock at the conclusion of a trading session	5060	3083	0	0
Volume	Total volume of shares exchanged within a certain timeframe	5060	3083	0	0

From Table 1 above, we were sure there were no missing values in both datasets.

C. Outliers

In this outlier subsection, we addressed unusual data points that deviate significantly from most of our dataset. Identifying and understanding these outliers is crucial, as they can impact statistical analyses and interpretations. While looking at the data with boxplots, we found some unusual points (outliers) that stand out in our time series analysis. Though we see these outliers,

our current knowledge does not cover specific methods for dealing with them. Therefore, for this report, we have decided to keep the outliers as they are.

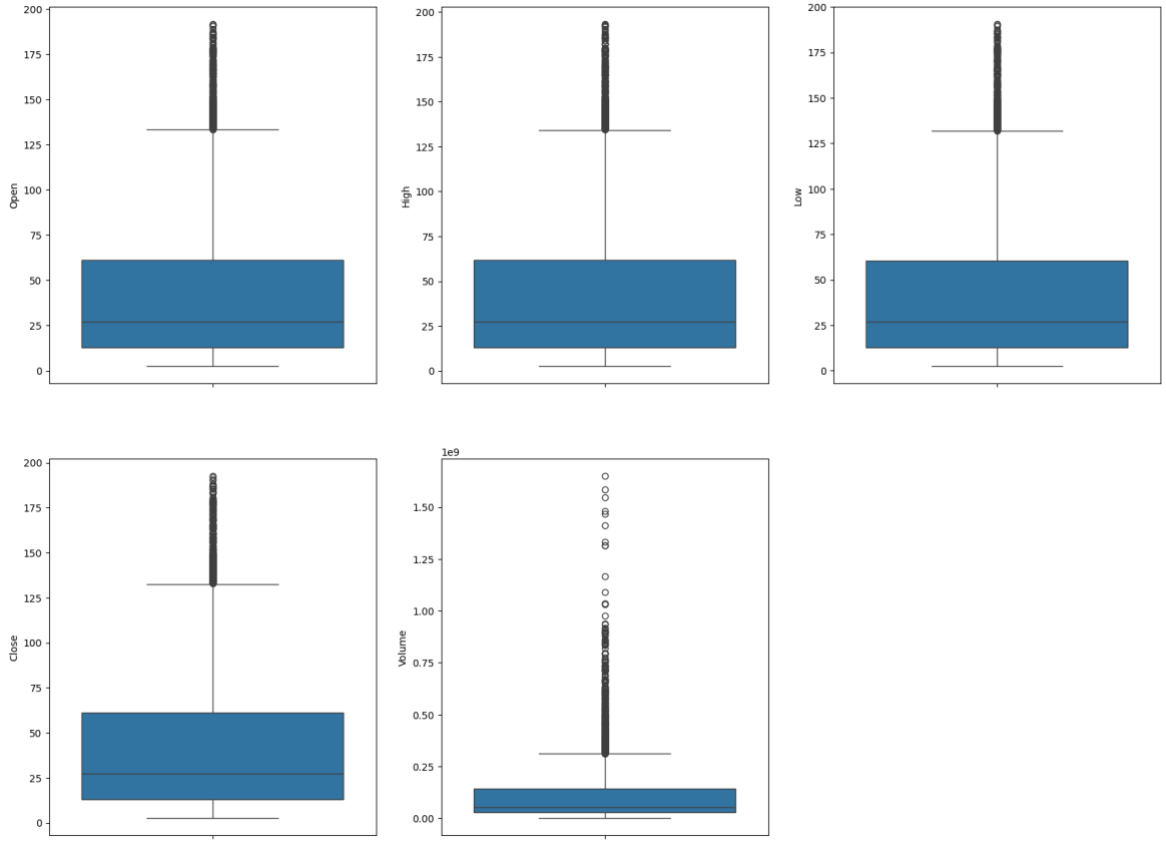


Figure 12: Boxplot of Google Stock Price Dataset for Outliers Detection

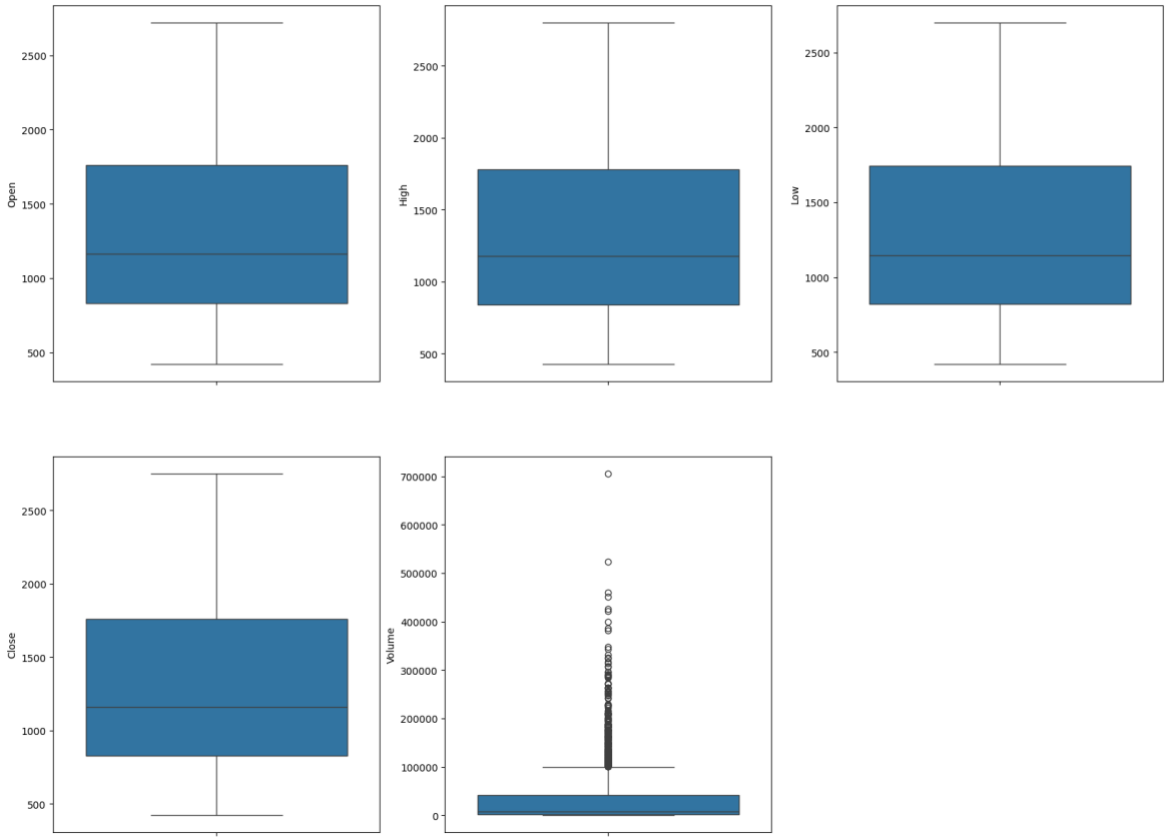


Figure 13: Boxplot of Nabil Stock Price Dataset for Outliers Detection

3.3.2.4 Data Transformation

Here we transformed our data into shapes or types that make data processing convenient. In the Nabil dataset, all the data types (dtypes) of the columns were objects, so we changed each column's data type (dtypes).

```
nabil['Date'] = pd.to_datetime(nabil['Date'])
nabil['Open'] = nabil['Open'].str.replace(',', '').astype(float)
nabil['High'] = nabil['High'].str.replace(',', '').astype(float)
nabil['Low'] = nabil['Low'].str.replace(',', '').astype(float)
nabil['Close'] = nabil['Close'].str.replace(',', '').astype(float)
nabil['Volume'] = nabil['Volume'].str.replace(',', '').astype(float).astype(int)
```

Figure 14: Changing dtypes of Columns of Nabil Dataset

3.3.3 Exploratory Data Analysis (EDA)

In this section, we employed visual representations and responses to address and uncover our dataset's significant patterns and insights, fostering a comprehensive understanding of its dynamics.

A. Stock Price over Time

Figure 15: Google Stock Price Trend over Time

Figure 16: Nabil Stock Price Trend over Time

From the above Google Stock Price line graph (Figure 15), a consistent upward trajectory over the years, with a notable peak in mid-2024, is observed, marking the highest closing price. However, a substantial decline is observed from 2022 until 2023. Interestingly, the stock demonstrates a renewed upward trend from early 2023, showcasing dynamic market movements and suggesting a resilient growth pattern amid fluctuations.

Figure 16 of the Nabil Stock Price trend graph shows that the stock price was initially high, which touched the highest closing price in late 2013. The price gradually declined, and its lowest closing price was observed in mid-2024.

B. Probability Distribution of Close Price of Stock

We utilised a probability distribution to illustrate the closing price data in our dataset. The graph above demonstrates the rising mean and standard deviation, suggesting that our series is not stationary.

The following graph presents the probability distribution of the closing price for both stock datasets.

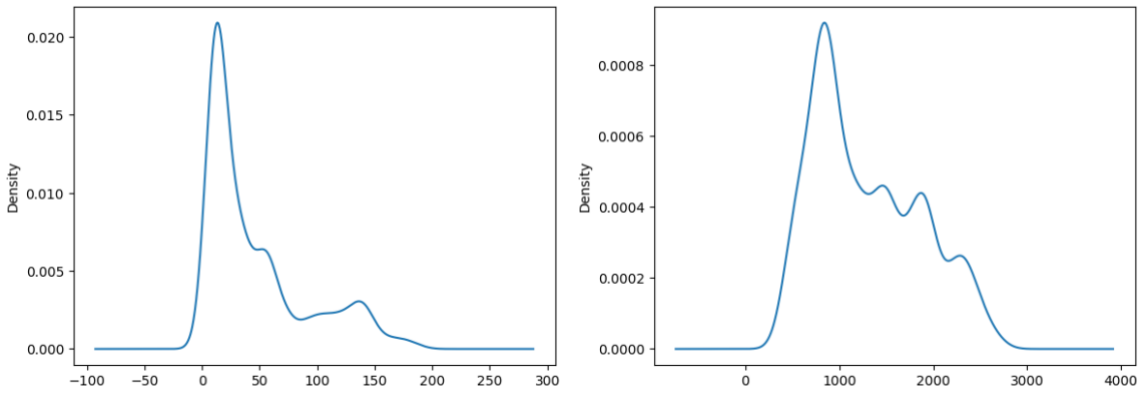


Figure 17: Probability Distribution of Close Prices of Google and Nabil, respectively

C. Historical Annual Price

The figures below provide a comprehensive overview of Google and Nabil's historical annual stock price data, featuring key metrics for each year, including the average stock price, stock open price, stock high price, stock low price, and stock close price, with annual changes.

Year	Average_Stock_Price	Year_Open	Year_High	Year_Low	Year_Close	Annual_Change (%)
2023	119.611860	89.830002	143.945007	85.570000	140.929993	56.885216
2022	115.193719	144.475494	152.100006	83.449997	88.730003	-38.584738
2021	125.530687	87.876999	151.850006	84.949997	144.679504	64.638650
2020	74.070191	67.077499	92.360001	50.676800	87.594002	30.586266
2019	59.419653	50.828499	68.250000	50.703499	66.850998	31.522668
2018	55.661257	52.417000	63.694500	48.505501	51.780499	-1.214303
2017	46.089042	38.940498	53.924500	38.790001	52.320000	34.358834
2016	37.174335	37.150002	40.834000	33.153000	38.591000	3.878864
2015	30.100284	26.378078	38.999001	24.311253	37.944000	43.846718
2014	27.990164	27.782366	30.607277	24.383057	26.247936	-5.523034
2013	22.023519	17.918339	27.920347	17.323069	27.913124	55.779642
2012	16.010401	16.262545	19.287207	13.861045	17.618462	8.337668
2011	14.171237	14.856315	16.108622	11.781341	16.087200	8.285268
2010	13.340574	15.615220	15.712356	10.800268	14.793799	-5.260385
2009	10.951161	7.686190	15.591310	7.042354	15.441621	100.900844
2008	11.577904	17.257067	17.369146	6.159413	7.662529	-55.597732
2007	13.418465	11.606496	18.611240	10.884203	17.222446	48.386271
2006	10.241246	10.523555	12.777108	8.257798	11.469011	8.984193
2005	6.918047	4.916571	11.113594	4.298140	10.332770	110.162124

Figure 18: Google Historical Annual Stock Price Data

Year	Average_Stock_Price	Year_Open	Year_High	Year_Low	Year_Close	Annual_Change (%)
2023	604.466520	859.0	865.0	493.0	506.0	-41.094296
2022	901.560744	1107.0	1213.0	726.0	843.0	-23.848238
2021	1357.630126	1100.0	1740.0	1042.0	1450.0	31.818182
2020	869.347826	683.0	1550.0	594.0	1100.0	61.054173
2019	782.918367	861.0	900.0	660.0	681.0	-20.905923
2018	965.150000	1020.0	1130.0	830.0	860.0	-15.686275
2017	1438.826087	1519.0	1810.0	1015.0	1020.0	-32.850560
2016	2045.556034	1860.0	2548.0	1421.0	1519.0	-18.333333
2015	2095.562791	1820.0	2794.0	1575.0	1860.0	2.197802
2014	2138.502203	2210.0	2680.0	1564.0	1820.0	-17.647059
2013	1899.513043	1693.0	2800.0	1540.0	2210.0	30.537507
2012	1234.938596	873.0	1701.0	814.0	1693.0	93.928981
2011	1038.686391	1200.0	1410.0	781.0	873.0	-27.250000

Figure 19: Nabil Historical Annual Stock Price Data

D. Stock Price High vs Low Comparison

The figures below show the high and low stock prices of each day of Google and Nabil stock. From the graph, we can understand the stock price range that had been in a day.

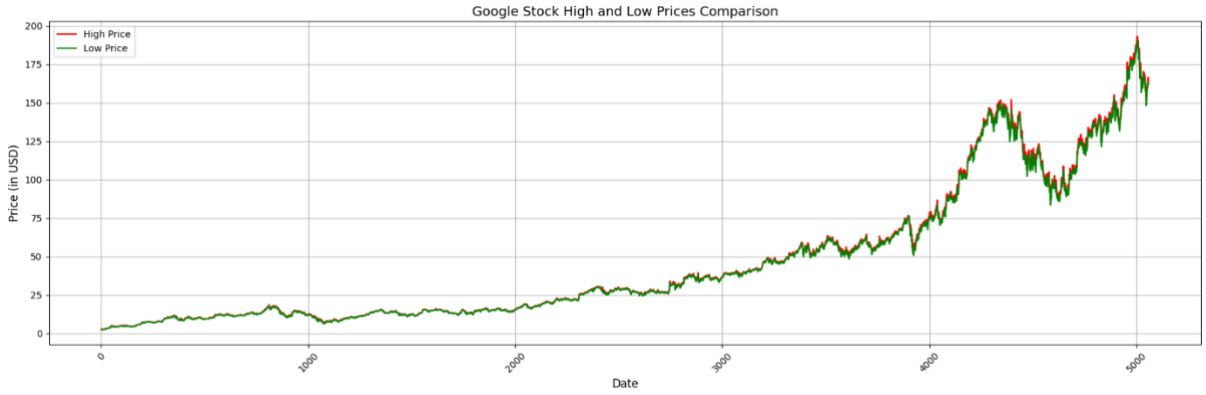


Figure 20: Google Stock High and Low-Price Comparison

Figure 21: Nabil Stock High and Low-Price Comparison

E. Top 20 Highest Share Volumes Traded in a Day

On January 20, 2006, Google traded its highest share volume in a day, around 1.6 billion shares, whereas on November 3, 2021, Nabil traded its highest share volume, 705 K shares. The graphs below show the top 20 days when both stocks traded their highest share volumes.

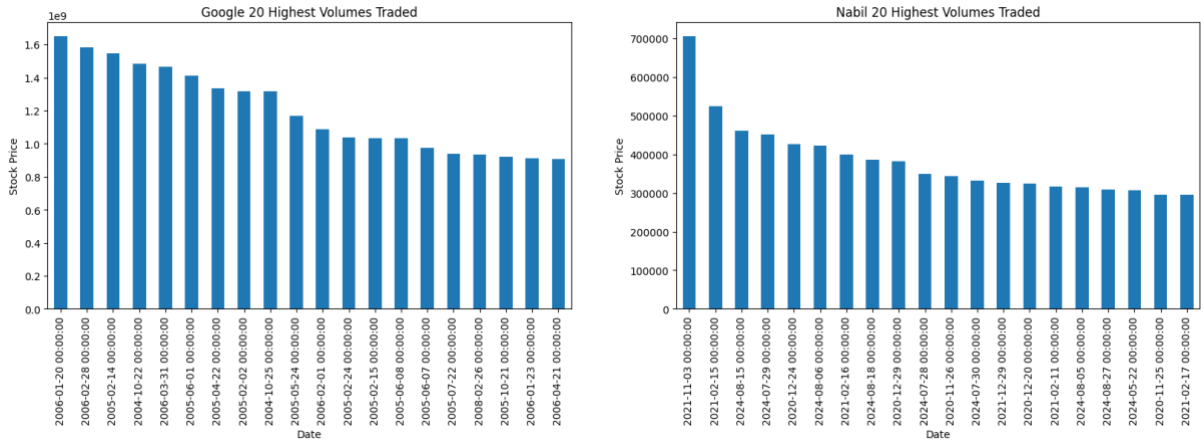


Figure 22: Top 20 Highest Share Volumes Traded in a Day

F. Daily Volume Average

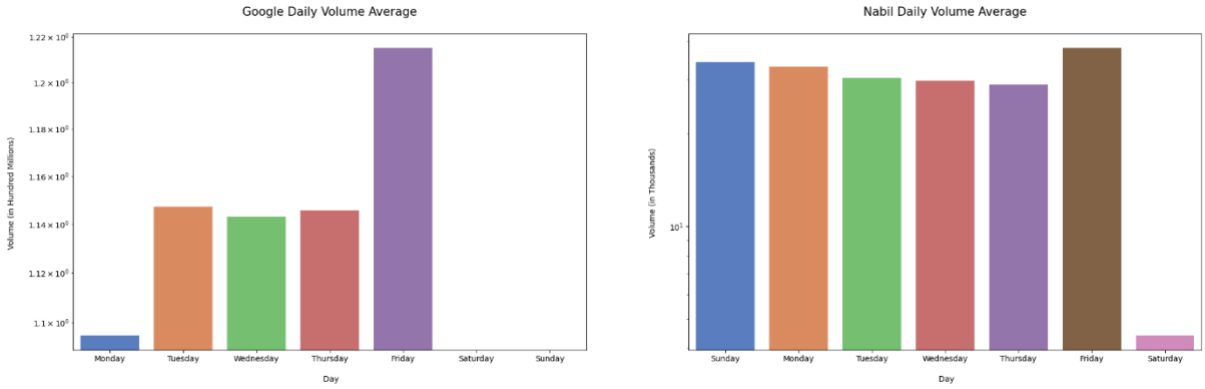


Figure 23: Daily Volume Average of the Stock

The plot analysis reveals that Friday has the highest volume average, indicating increased market activity, possibly influenced by end-of-week trading strategies. In contrast, Monday records the lowest volume average, suggesting a subdued start to the trading week. Google's remaining days exhibit relatively similar average volume levels. For Nabil, almost all weekdays exhibit similar average volume levels, but Friday records the highest volume average compared to other days.

G. Simple Moving Average Days

In finance, a moving average (MA) is a frequently used stock indicator in technical analysis. Its primary purpose is to smooth out price fluctuations by generating an average price that is continuously updated (Fernando, 2024).

The plots' analysis reveals that the moving average changes over 20, 50, and 100 days exhibit a strong similarity to the close price plot across all observed days. This suggests that the moving averages effectively capture the underlying trend and volatility of Google’s and Nabil’s stock prices over various time horizons.

Figure 24: Google Stock Price with 20, 50, 100-Day Moving Average

Figure 25: Nabil Stock Price with 20, 50, 100-Day Moving Average

H. Candlestick Representation

The images (Figure 26 & Figure 27) depict a candlestick chart representing Google's and Nabil's stock market performance over a period (from early 2023 to late 2023). The x-axis shows the date range, while the y-axis represents the stock price in USD and NPR.

Our first plot reveals a consistent upward trajectory, aligning with the broader trend observed in the stock market. Noteworthy instances include a substantial reduction in price range around February and the end of October, indicative of potential market stability or specific influencing factors during those periods. A cluster of green candles in late March and mid-May suggests bullish market sentiment, corresponding to periods of increased closing prices compared to openings. Conversely, pronounced occurrences of red candles at the beginning of February, the end of July, and the end of October signal bearish tendencies.

Similarly, in the second plot, the overall trend of Nabil Bank's stock price is downward. Starting at around 850 NPR in January 2023, the stock price has consistently decreased, reaching below 500 NPR by the end of the year. This chart provides an analysis of Nabil Bank's stock with a bearish pattern and decreasing value over the year, often used for technical analysis by traders to assess price movements and potential entry/exit points.

Figure 26: Google Stock Price: Candlestick Representation

Figure 27: Nabil Stock Price: Candlestick Representation

I. Variable Correlation

Analysing the relationships between stock market factors, such as opening, closing, highest and lowest prices, and trading volume, provides insight into the strength and direction of their interconnections. The correlation coefficient, which ranges from -1 to 1, can indicate a perfect positive correlation (where an increase in one variable leads to a corresponding increase in another), a perfect negative correlation (where an increase in one variable causes a decrease in another), or no linear correlation (suggesting a weak or non-existent relationship between the variables).

In the case of Google's stock market data, the correlation analysis shows a perfect positive relationship (correlation coefficient of 1) between the open, close, high, and low prices, demonstrating a strong linear association. On the other hand, volume demonstrates a moderate negative correlation (approximately -0.47) with the rest of the variables. The correlation analysis of Nabil Bank's stock prices shows a positive relationship between the open, high, low, and close prices, highlighting a strong connection between these variables. However, volume exhibits a negative correlation (around -0.28) with the other variables, suggesting that as the price-related metrics rise, trading volume tends to decline slightly.

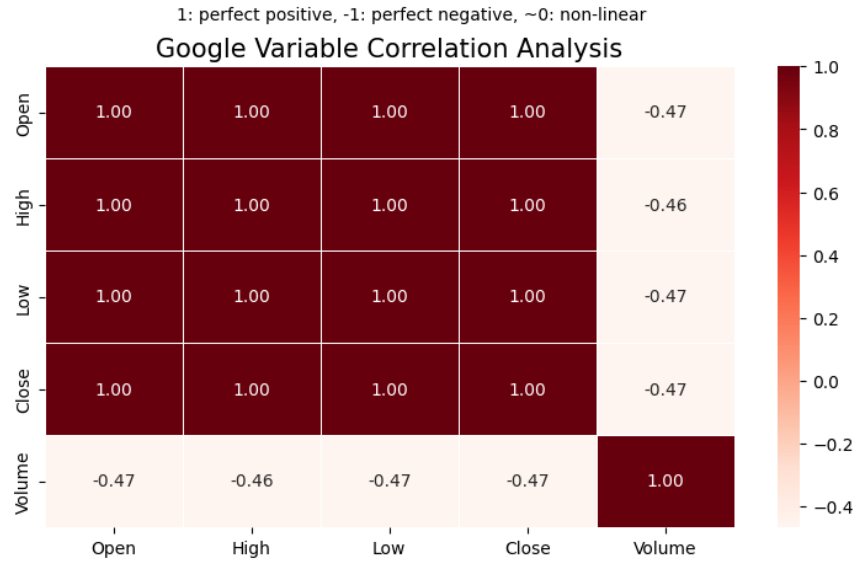


Figure 28: Google Stock Price Variable Correlation

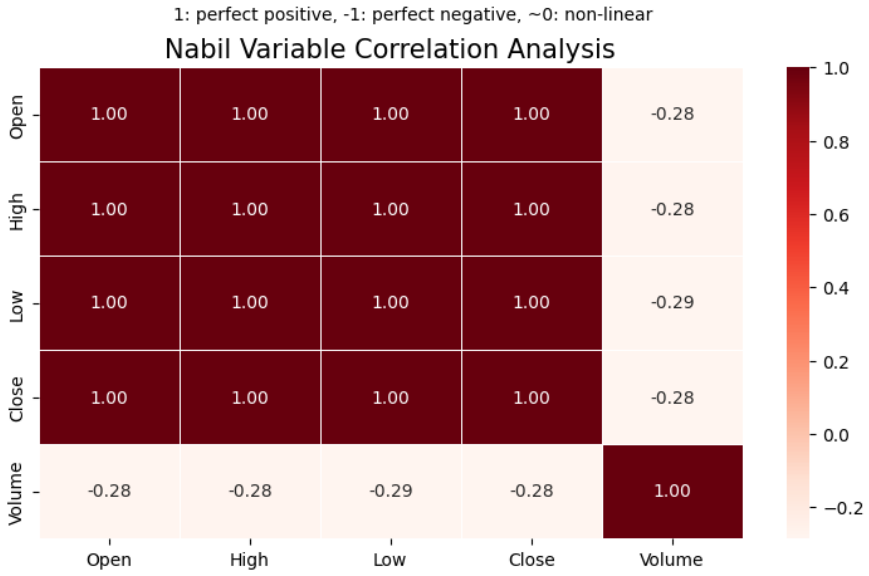


Figure 29: Nabil Stock Price Variable Correlation

The complete code for explanatory data analysis (EDA) can be found on the project's GitHub repository, Performance Evaluation and Analysis of Machine Learning Algorithms in Stock Price Prediction.

3.3.4 Data Splitting

To train and evaluate the machine learning models, it is important to split the dataset into training and testing subsets. In this case, 80% of the data was used for training the model, while the remaining 20% was reserved for testing the model's performance. The datasets are separated using two different methods: the first is from the **scikit learn library – (train_test_split), where we took a test size of 0.2, which is 20%, and a random state of 42**. This method violets the sequential pattern of the time series data but can cover the overall

pattern of the whole data. The second one we used was splitting the **first 80% of data as a train set and the last 20% as a test set.**

Splitting the data into train and test set

```
## First Method: Splitting data using scikit-learn train_test_split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

## Second Method: Splitting data as First 80% data as Train set and rest as Test set
X_train, X_test = X[:round(X.shape[0]*0.8)], X[round(X.shape[0]*0.8):]
y_train, y_test = y[:round(y.shape[0]*0.8)], y[round(y.shape[0]*0.8):]
```

Figure 30: Different Train-Test Dataset Splitting Methods

3.4 Model Building

After processing the data and engineering the features, the refined dataset was used to build, train, and evaluate the model. A range of machine learning and deep learning algorithms were applied in the model construction, which are detailed in the following sections:

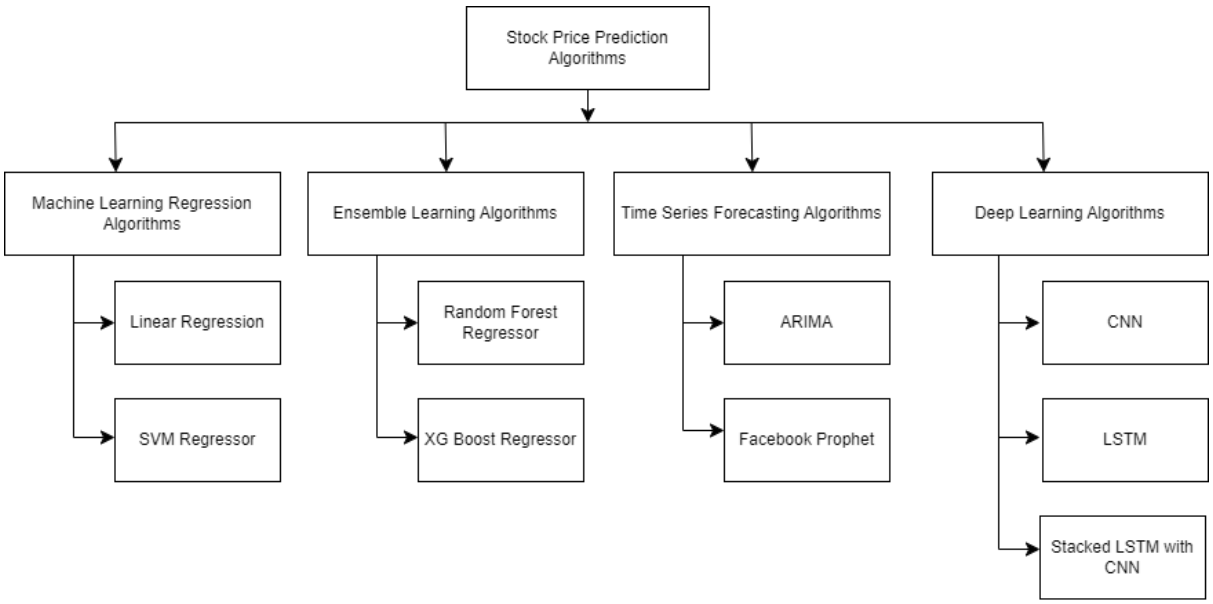


Figure 31: Algorithms Used for Predicting Stock Price

3.4.1 Basic Machine Learning Regression Algorithms

Machine learning regression algorithms predict continuous numerical values based on input features. We utilised two basic machine learning algorithms: linear regression and Support Vector Machine (SVM).

A. Linear Regression

Linear regression is one of the simplest and most widely used algorithms for predictive modelling in statistics and machine learning. It establishes a relationship between a dependent variable and one or more independent variables by fitting a linear equation to the data (Kavita, 2024). In this project, we implemented multiple linear regression, an extension of simple linear regression that allows for modelling a dependent variable influenced by multiple independent variables (features). Its objective is to analyse the relationship between a single continuous outcome variable and two or more predictor variables.

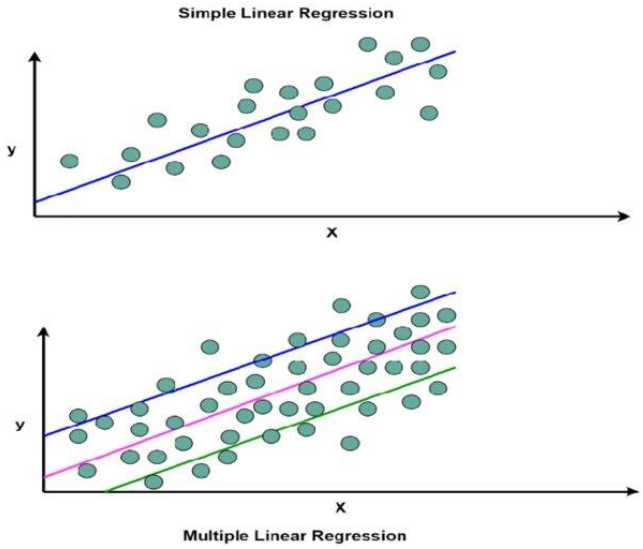


Figure 32: Graph Showing Simple Linear Regression and Multiple Linear Regression (Uddin, 2023)

The mathematical expression of Multiple Linear Regression is as follows:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots\ldots\ldots\ldots\ldots\ldots + \beta_n X_n + \epsilon$$

Where:

- Y is the dependent variable that we are trying to predict.
- X represents the independent variable (feature).
- β_0 is the y-intercept, which indicates the predicted value of Y when X = 0.
- B_1 is the slope of the line, representing how much Y changes for a one-unit increase in X.
- ϵ is the error term, which accounts for the difference between the predicted and actual values.
- n is the total number of observations.

B. Support Vector Machine (SVM)

Support Vector Machines (SVMs) were introduced by Vladimir Vapnik (Vapnik, 1999) and other researchers in the 1960s. The core concept of the SVM model is to create a linear classifier that identifies the optimal decision boundary among multiple possibilities. SVMs have gained popularity due to their strong performance, efficiency with medium-sized datasets, and ability to work well with CPU-based systems (Alakh, 2024). "Support vectors" refer to the data points closest to the decision boundary. SVMs are widely used in classification, recognition, regression, and time series analysis tasks. The main goal of SVM is to construct a hyperplane as a decision surface that maximises the margin between positive and negative data points.

Support Vector Regression (SVR) is a variant of the Support Vector Machine (SVM) algorithm tailored for regression problems. While SVM mainly focuses on classification tasks, SVR utilises comparable concepts to forecast continuous values, such as stock prices (Otten, 2024). The objective of SVR is to identify a function that diverges from the actual observed values by a minimal margin (regulated by a parameter) while remaining as flat as feasible.

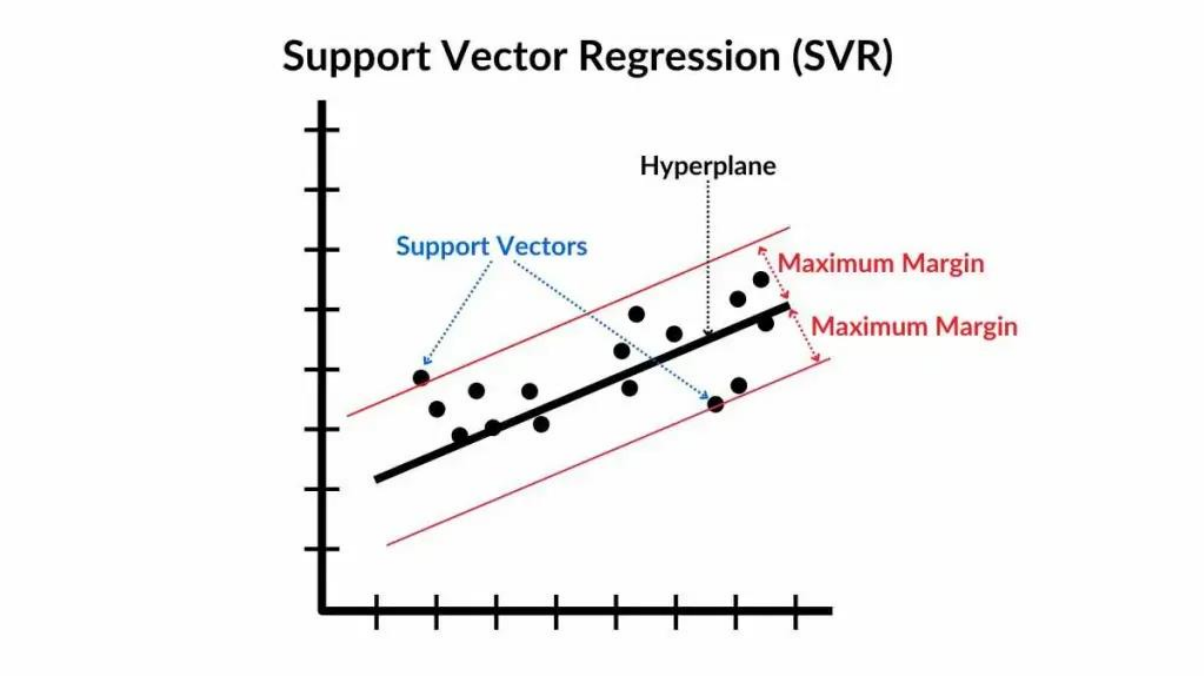


Figure 33: Support Vector Regressor Graph (Otten, 2024)

When the data exhibits non-linearity (which frequently occurs in stock price prediction), Support Vector Regression (SVR) can manage this by converting the input data into a higher-dimensional space using a kernel trick. This kernel trick enables SVR to execute non-linear regression by applying a kernel function to the input data.

The common kernels include:

- **Linear kernel:** For linear relationships.
- **Polynomial kernel:** For polynomial relationships.
- **Radial Basis Function (RBF) kernel:** For more complex, non-linear relationships.

The transformation is performed implicitly using the kernel function, avoiding the need to compute the mapping explicitly. The regression model in the kernel space becomes:

$$F(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(x_i, x) + b$$

Where:

- $i = n$ is the number of observations.
- $K(x_i, x)$ is the kernel function that computes the similarity between the points x_i and x.
- α_i and α_i^* are Lagrange multipliers from the dual formulation of the optimisation problem.

Support Vector Regression aims to fit a regression line within a defined margin (epsilon tube), minimising the errors beyond this margin and creating a smooth predictive model. Because it can use kernel functions, SVR is flexible enough to handle both linear and non-linear relationships, making it a powerful tool for tasks like stock price prediction, where non-linear patterns are often observed.

3.4.2 Ensemble Learning Algorithms

These algorithms are machine learning techniques that combine the predictions of multiple individual models to achieve more accurate and reliable outcomes than any single model could offer (Brownlee, 2021). The core idea is that by merging the results of different models, the limitations of one model can be compensated for by the strengths of others, leading to improved generalisation and performance. All ensemble methods are grounded in the Weak Law of Large Numbers (WLLN), which states that if each model in the ensemble has slightly above 50% accuracy and operates independently, increasing the number of models will make the overall accuracy of the combined prediction approach 100% through majority voting. This discussion will focus on one bagging method, the Random Forest Regressor, and one boosting method, the XG-Boost Regressor.

A. Random Forest Regression

Random Forest Regression is an ensemble learning method that employs decision trees to tackle regression tasks. It builds multiple decision trees and combines their predictions (Baheshti, 2022). The fundamental concept behind Random Forest is that a collection of diverse, weak learners (decision trees) can collaborate to produce a strong and accurate predictive model. This approach involves averaging individual decision tree outputs, reducing overfitting and improving the model's ability to generalise.

It is an ensemble learning approach known as bagging. In this approach, multiple models (typically of the same kind) are trained on different random samples of the dataset. The outcomes from all these models are then aggregated, usually by averaging (for regression) or voting (for classification), to enhance overall accuracy and lessen overfitting.
So, for a given input x, the predicted value from each tree $T_i(x)$ is averaged to get the final prediction:

$$\hat{y} = \frac{1}{N}\sum_{i=1}^{N} T_i(x)$$

Where:

- $\hat{y}$ is the final prediction
- $T_i(x)$ is the prediction from the ***i-th*** tree
- N is the total number of trees in the forest

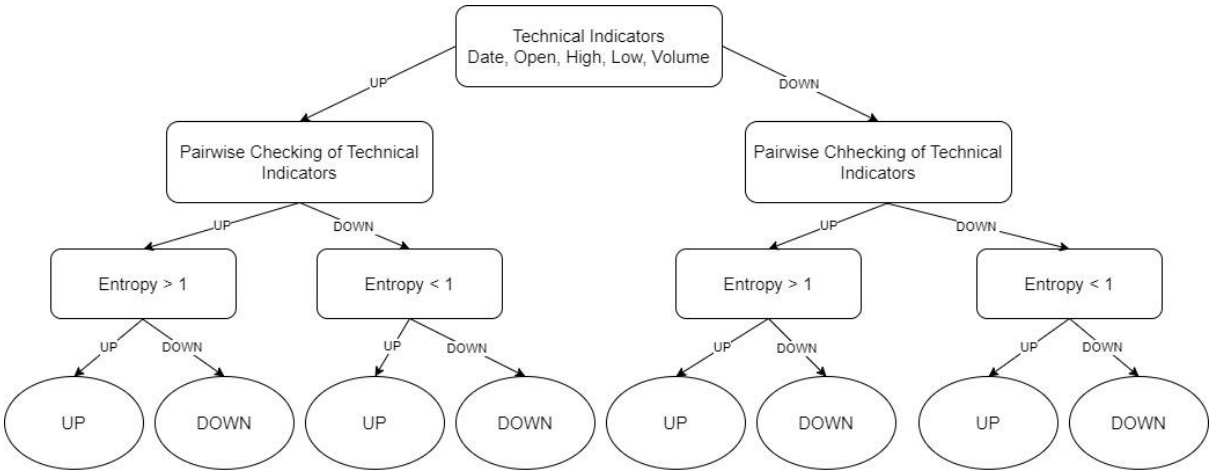


Figure 34: Single Decision Tree of Random Forest

B. XGBoost Regression

XGBoost (Extreme Gradient Boosting) is an advanced and highly efficient variant of gradient boosting designed for improved speed and performance. It builds a sequence of decision trees, where each subsequent tree is trained to correct the errors made by the previous ones. For regression tasks, XGBoost minimises the difference between predicted and actual values (the residuals) by iteratively fitting additional trees to these residuals (Brownlee, 2021).

XGBoost operates under the gradient boosting framework, which creates models sequentially by introducing new trees that address the errors from earlier trees. The model's performance is enhanced with each iteration by minimising a loss function (such as Mean Squared Error for regression) through gradient descent methods. XGBoost constructs the model as an additive sequence of decision trees. If the prediction at the ***t-th*** step is denoted by $\hat{y}^t$, the model at step ***t+1*** adds a new decision tree $f_{t+1}(x)$ to improve the previous predictions:

$$\hat{y}^{t+1}(x) = \hat{y}^t(x) + f_{t+1}(x)$$

The idea is that each new tree $f_{t+1}(x)$ is trained to minimise the residuals from the previous prediction.
The figure below demonstrates the algorithm:

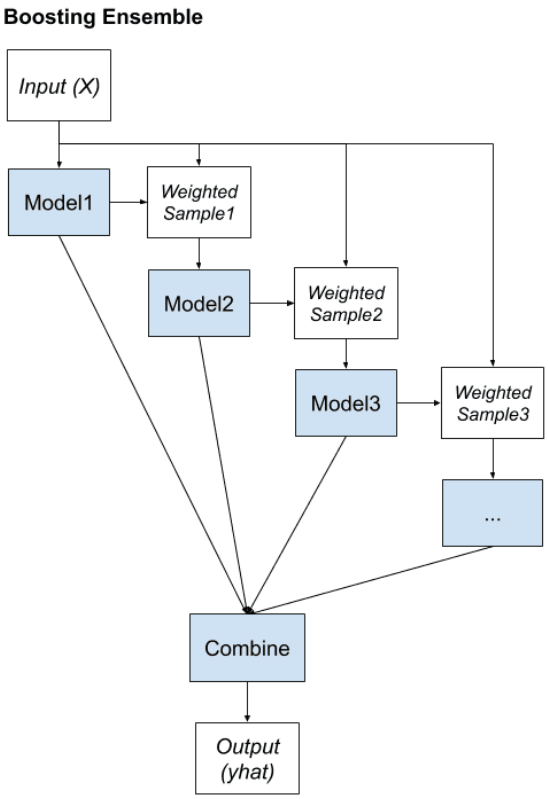


Figure 35: Flowchart showing Boosting Ensemble Learning (XGBoost)

The model is refined through gradient descent applied to a loss function, incorporating regularisation to mitigate overfitting. By integrating numerous trees and adding randomness, XGBoost attains excellent predictive performance while maintaining resilience against overfitting, which makes it ideal for intricate tasks such as predicting stock prices.

3.4.3 Time Series Forecasting Algorithms

Time series forecasting methods estimate future outcomes based on historical data points, considering the temporal relationships within the data. These methods are frequently employed in finance, economics, meteorology, and supply chain management. ARIMA and FB Prophet were used as time series forecasting techniques to develop the model.

A. ARIMA

ARIMA (Auto-Regressive Integrated Moving Average) is a widely used statistical method for forecasting time series data, combining three core components: autoregression (AR), integration (I), and moving average (MA) (Hayes, 2024). This technique is a form of regression analysis designed to explore the relationship between a dependent variable and various influencing factors. The primary objective of ARIMA is to predict future trends in securities or financial markets by analysing the fluctuations in a time series' values rather than focusing on the exact values themselves. However, ARIMA can only handle a single feature, so incorporating SARIMAX (Seasonal Auto-Regressive Integrated Moving Average with exogenous variables) allows for the inclusion of additional features, such as Open, High, Low, and Volume, which enhance the stock price prediction model.

SARIMAX models are particularly useful when seasonal patterns and external factors affect the target variable, such as the Close price in this context. Different components in ARIMA are:

1. **Auto-Regressive (AR) component:**

The model's autoregressive (AR) component represents how a current observation depends on several preceding ones (previous time points). The parameter p defines how many past observations are included in the model. The AR model can be represented as:

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots\ldots\ldots + \phi_p Y_{t-p} + \epsilon_t$$

Where:

- Y_t is the observed value at time t.
- $\Phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive coefficients.
- ϵ_t is a white noise error term.

This equation shows that the current value Y_t is influenced by the previous p values in the series.

2. **Integrated (I) component:**

The integrated component of the model is concerned with converting the time series into a stationary form, which is crucial for ARIMA modelling. A stationary time series has a constant mean and variance over time. The integration process involves differencing the data, represented by the parameter d, which indicates how often the data must be differenced to achieve stationarity. Differencing a time series involves subtracting the previous value from the current one:

$$Y'_t = Y_t - Y_{t-1}$$

If the series remains non-stationary after the first difference, further differencing (e.g., second differencing) may be applied until it becomes stationary.

3. **Moving Average (MA) component:**

The moving average component of the model captures the relationship between an observation and the residuals from a moving average model applied to prior observations (Saadeddin, 2024). The MA part is characterised by the parameter q, indicating the count of lagged forecast errors incorporated in the model. The MA model can be represented as:

$$Y_t = \theta_0 + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots\ldots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Where:

- θ_0 is a constant term.
- $\Theta_1, \theta_2, \theta_{t-q}$ are the moving average coefficients.
- $\epsilon_{t-1}, \epsilon_{t-2}, \epsilon_{t-q}$ are the white noise error terms from previous time points.

So, combining these terms, the general form of the ARIMA model can be represented as:

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots\ldots\ldots + \phi_p Y_{t-p} + \theta_0 + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots\ldots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Where:

- p is the number of autoregression terms.
- d is the number of times the series has been differenced.
- q is the number of moving average terms.

B. FB Prophet

FB Prophet is an open-source forecasting tool that Facebook's Core Data Science team created. It aims to analyse time series data with significant seasonal trends and multiple years of historical data. It's especially effective for business predictions because of its user-friendly nature and resilience to missing values and anomalies. Prophet offers a straightforward method for modelling seasonal patterns and includes the impact of holidays, making it applicable for various uses such as predicting stock prices, forecasting sales, and analysing website traffic (Melanie, 2024).

Figure 36: FB Prophet Logo (Ankit, 2022)

FB Prophet operates on an additive framework that integrates three essential elements:

1. **Trend component:**

The trend component represents the general direction of the time series as it progresses over time. Prophet depicts trends using piecewise linear or logistic growth curves.
The linear trend can be expressed as:

$$g(t) = g_0 + kt$$

Where:

- $g(t)$ is the value of the trend at time t.
- g_0 is the initial value of the trend at $t = 0$.
- K is the growth rate (slope) of the trend.

The logistic trend i.e., data is bounded, can be represented as:

$$g(t) = \frac{C}{1+e^{-k(t-m)}}$$

Where:

- C is the carrying capacity (the maximum value the time series can approach).
- m is the reflection point where the growth rate changes.

2. **Seasonal component:**
The seasonal component captures periodic fluctuations in the data. Prophet models' seasonality using Fourier series to approximate seasonal effects over different periods (e.g., daily, weekly, yearly).
The seasonal effect is modelled as follows:

$$s(t) = \sum_{n=1}^{N}(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right))$$

Where:

- ***s(t)*** is the seasonal effect at time ***t.***
- a_n ***and*** b_n are the Fourier coefficients for each frequency component ***n.***
- ***P*** is the period of seasonality.
- ***N*** is the number of Fourier terms included in the model to capture the seasonal pattern's complexity.

3. **Holiday component:**
FB Prophet enables users to incorporate significant events or holidays that may influence the time series, enhancing the model further. Users specify the holiday impacts, and the model integrates extra components to account for these effects.
The holiday effect can be modelled as:

$$h(t) = \sum_{j=1}^{J} I_j(t) \cdot \delta_j$$

Where:

- ***h(t)*** is the holiday effect at time ***t.***
- $I_j(t)$ is an indicator function of 1 if holiday ***j*** occurs at time ***t*** and ***0*** otherwise.
- δ_j is the impact of holiday ***j.***

Combining these components, the overall model can be expressed as:

$$Y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

Where:

- ***Y(t)*** is the predicted value at time t
- ***g(t)*** is the trend component.
- ***s(t)*** is the seasonal component.
- ***h(t)*** is the holiday effect.
- ϵ_t is the error term, assumed to be normally distributed with mean zero.

FB Prophet enables users to incorporate special events or holidays that could impact the time series, which enhances the model. Users specify the holiday effects, and the model integrates additional components for these effects.

3.4.4 Deep Learning Algorithms

Deep learning algorithms belong to the machine learning category and utilize neural networks with multiple layers (hence the term "deep") to capture intricate patterns within large datasets.

To create the predictive model, we employed CNN, LSTM, and a combination of CNN and LSTM (stacked CNN-LSTM).

A. CNN

Convolutional Neural Networks (CNNs), originally designed for computer vision tasks, have also proven effective for processing sequential data, such as stock prices. CNNs' fundamental idea is to identify local patterns within the data using convolutional processes. They primarily target the detection of temporal patterns essential for forecasting time series values while providing strong generalisation and resilience to noise. Their capability to extract abstract features and reveal hidden non-linear relationships in the data without human input makes CNNs a compelling alternative to conventional feature-based methods (Gunduz, et al., 2017).

Stock prices are time series data that exhibit patterns over time, such as trends, cycles, and volatility shifts. CNNs can capture these local patterns (such as a short-term trend or specific behaviour over a period) by using filters or kernels in the convolution layer. This capability makes CNNs suitable for identifying useful features that can help predict future price movements.

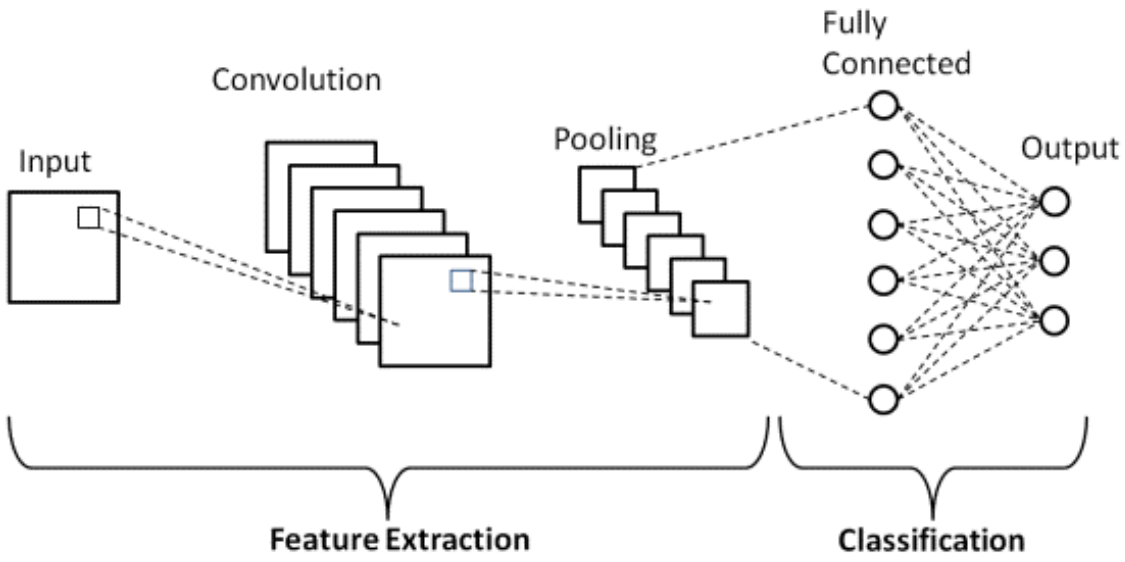


Figure 37: Layers of CNN

The CNN comprises five layers: the Input layer, Convolutional layer, Pooling layer, Fully Connected layer, and Output layer. The crucial layers within a CNN are the hidden layers, which are situated between the input and output layers.

1. Convolutional Layer:

The convolutional layer employs several filters (or kernels) to detect local features from the input data. Each filter is a small matrix (such as one-dimensional for time series) that moves along the time axis to identify a pattern. The main mathematical operation involved is convolution:

$$Z_{i,j} = \sum_{n=1}^{k} W_m X_{i+m-1,j} + b$$

Where:

- W_m is the weight of the filter at position m.
- $X_{i+m-1, j}$ is the input at position $i + m - 1$ for feature j.
- b is the bias term.
- $Z_{i, j}$ is the result after applying the filter at position i.

2. **Pooling Layer:**
A pooling layer is commonly used after the convolutional layer to reduce the size of the feature maps, improving the network's efficiency. Max pooling is a commonly used pooling technique that selects the highest value from a defined window of values:

$$P_{i, j} = max\ (0, Z_{i: i + f - 1, j})$$

Where:

- F is the pooling window size.
- $Z_{i, j}$ is the output from the convolution layer.
- $P_{i, j}$ is the pooled layer.

3. **Fully Connected Layer:**
In the fully connected layer, each neuron is connected to every element of the flattened vector. This layer combines the features detected by the convolutional layers to produce the final prediction. The output is calculated as:

$$y = \sigma(W_{fc} \cdot P + b_{fc})$$

Where:

- W_{fc} is the weight matrix of the fully connected layer.
- P is the pooled output.
- B_{fc} is the bias term.
- Σ is the activation function.

CNNs are useful for stock price prediction because they can capture localized patterns over time, such as trends or volatility patterns. They are computationally efficient due to weight sharing and parameter reduction in the convolution layers. When used with other models (e.g., LSTM or GRU), CNNs can help extract meaningful features from time series data for more accurate predictions.

B. LSTM

LSTM (Long Short-Term Memory) is a Recurrent Neural Network (RNN) type designed to handle and learn long-term dependencies in sequential data. Unlike traditional RNNs, which face challenges in retaining information over long sequences due to problems like vanishing or exploding gradients, LSTMs feature a specialised structure that allows them to selectively retain or discard information over time. This capability makes LSTMs particularly effective in time series prediction tasks, such as stock price forecasting, as they excel at identifying short-term and long-term patterns in sequential data.

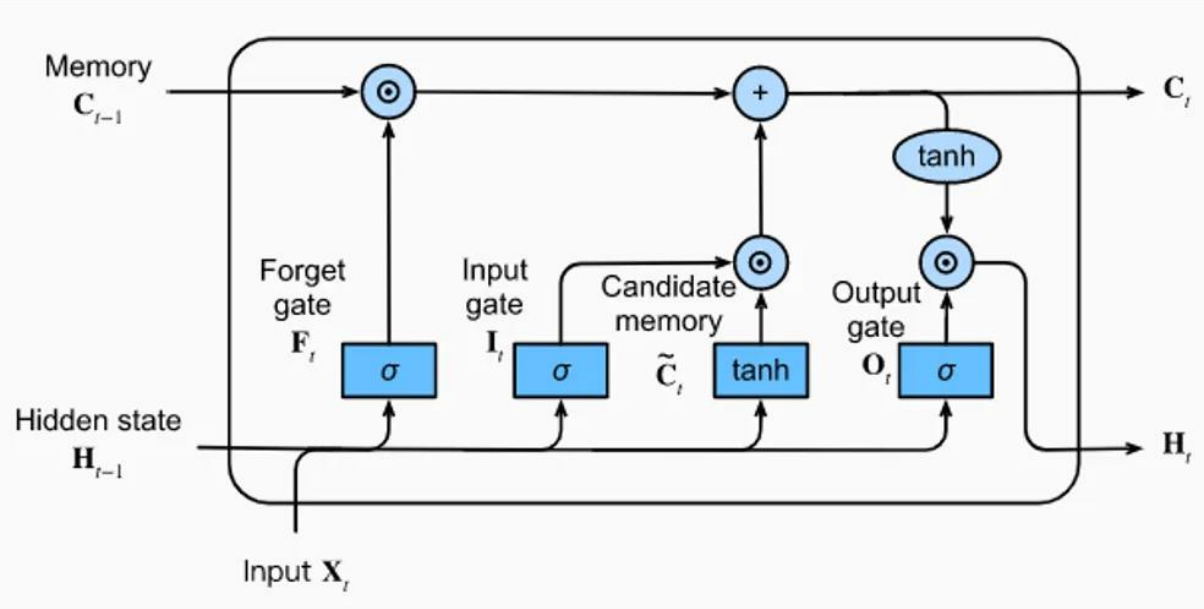


Figure 38: Components of LSTM (Calzone, 2022)

The LSTM structure has mainly four components:

1. **Forget Gate f_t**

   The forget gate controls which parts of the previous cell state (Ct-1) should be kept or discarded. Using a sigmoid activation function generates an output between 0 and 1, indicating the degree to which each element of the previous cell state should be remembered or forgotten.

$$f_t = \sigma\,(W_f \cdot [h_{t-1}, x_t] + b_f)$$

   Where:

   - F_t is the forget gate,
   - W_f is the weight.
   - h_{t-1} is the previous hidden state.
   - x_t is the input at the current time step.
   - σ is the sigmoid activation function.
   - b_f is the bias term.

2. **Input Gate i_f**

   The input gate controls the amount of new information to be incorporated into the cell state. It uses a sigmoid activation function to determine which values should be updated, allowing the network to adjust its memory based on the current input.

$$i_t = \sigma\,(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

   Where:

   - i_t
   - W_i
   - $\tilde{C}_t$
   - W_c is the weight matrix used to generate the candidate cell state.

- ***tanh*** is the hyperbolic tangent function that ensures the values of $\tilde{C}_t$ lie between -1 and 1
- b_i ***and*** b_c are biased.

3. **Update Cell State C_t**

The cell state C_t is updated by combining the previous state C_{t-1}, controlled by the forget gate, with the new candidate state $\tilde{C}t$, controlled by the input gate.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

Where:

- $f_t \cdot C_{t-1}$ represents the retained old cell state.
- $i_t \cdot \tilde{C}_t$ represents the added new information.

4. **Output Gate o_t**

The output gate determines the hidden state h_t, which is used for the next time step and serves as the output of the current cell.

$$o_t = \sigma\,(W_0 \cdot [h_{t-1}, x_t] + b_0)$$
$$h_t = o_t \cdot tanh(C_t)$$

Where:

- o_t is the output gate.
- h_t is the hidden state, which represents the output for the current time step.
- W_0 is the weight matrix for the output gate.
- $Tanh(C_t)$ is the non-linear transformation of the updated cell state

C. Stacked CNN-LSTM

A Stacked CNN-LSTM architecture sequentially integrates Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks to derive spatial and temporal characteristics from stock price information. This model utilizes the CNN's capacity to identify local patterns (such as short-term price fluctuations or trends) and the LSTM's capability to model long-term dependencies within time series data.

Stock prices are sequential data exhibiting localised patterns, such as short-term price movements or volatility spikes. CNNs can efficiently capture these localised patterns, while LSTMs capture long-term temporal dependencies. By stacking CNN layers on top of LSTM layers, the model can extract features using CNNs and then learn long-term dependencies with LSTMs. This hybrid approach effectively predicts stock prices because it incorporates local feature extraction and temporal sequence learning.

3.5 Hyperparameter Tuning

Hyperparameter tuning refers to selecting the optimal hyperparameters for an ML model (Jordan, 2017). A model may not perform at its best without tuning, as poor hyperparameters can lead to suboptimal outcomes, underfitting, or overfitting. Hyperparameter tuning aims to strike a balance between model complexity and generalisation capability. Common methods for hyperparameter tuning include Grid Search and Random Search.

For our research project, we used Grid Search for basic machine learning algorithms. This method exhaustively tries every possible combination of hyperparameters from a specified grid and is guaranteed to find the best combination. However, it is computationally expensive and scales poorly with large grids. For deep learning algorithms, we use random search to try a random combination of hyperparameters from the specified grid.

As for different machine learning, different parameters like regularization strength, kernel, gamma, n_estimators, max_iter, max_depth, etc., are tuned as per the algorithms. For deep learning algorithms, the filter number, kernel size, dropout rate, dense layer units, learning rate, pool size, etc, are tuned according to the algorithms.

3.6 Performance Evaluation Metrics

In this research project, we evaluated and analysed the performance of machine learning models in predicting stock prices. To accomplish this, we calculated performance evaluation metrics for each machine learning model. These metrics are essential in assessing the effectiveness and accuracy of ML and DL models. Our study's key performance evaluation metrics for assessing the machine learning models are summarised below.

3.6.1 Mean Squared Error (MSE)

The Mean Squared Error (MSE) measures the average of the squared differences between predicted values and actual results. By squaring the differences, it applies a greater penalty to larger mistakes than to smaller ones, making this metric responsive to significant prediction errors (Practicus AI, 2019). It is the most straightforward and widely used loss function.

The formula for **MSE**:

$$\boldsymbol{MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

Where:

- N is the number of data points,
- y_i is the actual value
- $\hat{y}_i$ is the predicted value

If we have lower MSE, it is said to be better model performance. Squaring the error makes MSE more sensitive to large errors (outliers), making it suitable when large deviations are undesirable. MSE penalises large errors more than small errors, helping to avoid models that make significant mistakes. The disadvantage is that if our model makes a particularly poor prediction, the squaring in the function amplifies the error. However, in many real-world scenarios, we aren't overly concerned with these outliers and are more focused on developing a balanced model that performs adequately for most cases. Also, it is difficult to interpret because it's not in the same units as the target variable.

3.6.2 Root Mean Square Error (RMSE)

The Root Mean Squared Error (RMSE) is the square root of the MSE, allowing the error metric to be interpreted in the same units as the target variable. It retains the benefits of MSE (penalising large errors) but provides easier-to-interpret results (Olumide, 2023).

To calculate the RMSE, we can root the MSE or use the formula,

$$\boldsymbol{RMSE} = \sqrt{\frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}(\boldsymbol{y_i} - \boldsymbol{\hat{y}_i})^2}$$

Where:

- N is the number of data points,
- y_i is the actual value
- $\hat{y}_i$ is the predicted value

The model is said to perform better if the RMSE value is lower. It is easier to understand and preferred than MSE because it is in the same units as the output. It is also sensitive to outliers. Similar to MSE, it penalises large errors with an interpretable scale.

3.6.3 Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) measures the average absolute difference between predicted and actual outcomes. It provides a simple metric in the same unit as the target variable, making it easy to understand.

The formula for MAE,

$$\boldsymbol{MAE} = \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}|\boldsymbol{y_i} - \boldsymbol{\hat{y}_i}|$$

Where:

- N is the number of data points,
- y_i is the actual value
- $\hat{y}_i$ is the predicted value

Lower MAE indicates better model performance. It is simple to calculate and understand. It treats all errors equally since it doesn't square the differences, unlike MSE. MAE is more robust to outliers than MSE because outliers aren't squared. It is directly interpretable in terms of the unit of the target variable, but it lacks large errors so it might underestimate models with a few significant errors.

3.6.4 Computational Time (Training Time)

Computational Time refers to the total time an algorithm or model takes to perform a specific task, which can include training, validation, testing, and inference (prediction). Understanding computational time is essential for evaluating the efficiency and scalability of machine learning models, especially in production environments where speed and resource management are critical (Qayyum, 2022).

The time it takes to compute, especially during training, is a vital metric for evaluating performance in machine learning, especially as models grow more complex and datasets expand. It indicates the duration needed to train a model with a specific dataset. Grasping training time is important for various reasons, such as managing resources, choosing models, and the practicality of implementing models in real-life scenarios. Although performance metrics and accuracy are critical, understanding the duration required to train a model offers valuable insights into its practicality and feasibility.

3.7 Project Implementation

Our research project, "**Performance Evaluation and Analysis of Machine Learning Algorithms in Stock Price Prediction,"** is focused on predicting stock prices using different machine learning algorithms and comparing, evaluating, and analysing their performance. So, when implementing a machine learning project for stock price prediction, it's essential to follow a structured approach that ensures accurate performance evaluation.

3.7.1 Implementation Details

After all the project requirements were analysed, the project was set to develop with the system designed, along with developing project methodology and flowchart. Firstly, the dataset for the project was collected from different websites, or the data was scrapped from the different websites. The datasets were imported into the integrated development environment (IDE) using the Pandas library. The data were then pre-processed and feature-engineered, removing irrelevant features, removing outlier data, fulfilling missing values, and transforming the data into required formats. The data were scaled with a min-max scaler, which helps ensure that machine learning models perform optimally by eliminating issues arising from differences in feature scales. It improves model convergence, provides regularisation, supports distance-based metrics, and enhances the interpretability of results, making it an essential step in numerous ML workflows. Following data pre-processing and feature engineering, the dataset is divided into training and testing sets for training and evaluating the machine learning model. We utilised 80% of the dataset for model training and reserved 20% for testing the machine learning model.

After processing the data, various machine learning and deep learning techniques, such as Linear Regression, SVM, Random Forest, XG Boost, LSTM, and CNN, were utilised to create predictive models for future data. We implemented these algorithms using the scikit-learn and TensorFlow libraries in Python. To enhance performance, we fine-tuned the hyperparameters of each algorithm, which aids in achieving optimal prediction accuracy. For the assessment and

analysis of the machine learning and deep learning models, we measured MSE, RMSE, MAE, and computational time, particularly focusing on training duration.

The complete code, setup, and implementation can be found on the project's GitHub repository, Performance Evaluation and Analysis of Machine Learning Algorithms in Stock Price Prediction.

3.7.2 Implementation Challenges

Implementing machine learning models for stock price prediction presents several challenges, from data collection to the model's performance evaluation. We used two stock price datasets: the **Google** and **Nabil Bank Stock Price datasets.** Google's stock price dataset can be found anywhere on websites such as Kaggle, Yahoo Finance, etc. Still, the Nabil stock price dataset is hard to find in downloadable form on any website in a proper format. So, we had to scrape the data from the dedicated website, which required legal approval. Also, the data scrapping process was too lengthy due to the time required for the data to be scrapped. Stock prices are inherently volatile, non-stationary, and influenced by many external factors, making it difficult for models to generalise well from historical data to future events. Common issues include data leakage, overfitting to noisy historical data, and dealing with limited historical information, which can result in misleading performance metrics. Additionally, selecting appropriate evaluation metrics is crucial. Standard metrics like MSE or RMSE might not reflect true performance in financial applications, where predicting trends or direction is more critical.

Another significant challenge is the availability of computational resources, particularly when working with complex algorithms such as CNN, LSTM and stacked models (CNN-LSTM) of these algorithms. These algorithms require substantial computing power and long training times due to their high computational demands. As the dataset grows and the number of parameters increases, the complexity of these models may become unmanageable with limited computing resources.

4. Results, Analysis and Discussion

This section outlines the experiments, their outcomes, and the results produced by the model developed for our research. We employed two datasets for our analysis: 1. Google Stock Price dataset 2. Nabil Bank Stock Price dataset. These datasets are utilised to create prediction models by applying various machine learning and deep learning techniques. A range of performance evaluation metrics is implemented to evaluate the effectiveness of these algorithms.

4.1 Results

For the study, all machine learning and deep learning algorithms except FB Prophet were trained on a dataset with the same features, the same scaling process (standard scalar), and the same amount of data. The FB Prophet algorithm takes two columns from the dataset: the Date and Close price columns. To train the model with **FB Prophet**, we used all price data except the last 30 days' price data, which was used to test the model.

In the time series algorithm, **ARIMA** can only take one feature for training, so we took the extended form of ARIMA- **SARIMAX**. SARIMAX can take multiple features, which take order parameters that contain three values as **p** (Order of Auto-Regressive), **d** (number of times the series is differentiated), **q** (order of moving average) and seasonal parameters, which are similar to order parameters but have a seasonal component. This order is to be determined to forecast values with SARIMAX. We used '**pmdarima',** a Python library for automating hyperparameter tuning and simplifying the forecasting pipeline. We employed a standard batch size of **32** for the deep learning models and ran for **50 epochs**. The batch size indicates the data points evaluated before the model adjusts its weights. At the same time, epochs signify a full iteration over the whole training dataset throughout the training period.

After conducting experiments on two datasets using nine different types of machine learning, deep learning, and time series algorithms, it was discovered that nearly all models achieved a prediction accuracy score of 99%. While accuracy is a clear and intuitive metric, it is often insufficient, particularly in complex, imbalanced, or domain-specific scenarios. Performance evaluation metrics offer a more comprehensive and detailed insight into model performance, addressing subtleties that accuracy may overlook (Jain, 2024). Consequently, MSE, RMSE, and MAE were utilised in this study to assess the performance of the machine learning and deep learning algorithms. The results of the experiments were organised to compare the performance evaluation metrics and computational efficiency (training time) of various algorithms.

Two methods were employed to split the data into a train set and a test set. We calculated performance evaluation metrics for each, but we calculated training time only for the model trained with data split with **scikit's model_selection** library **train_test_split()** method. The results and graphs from the experiment with two datasets are discussed below.

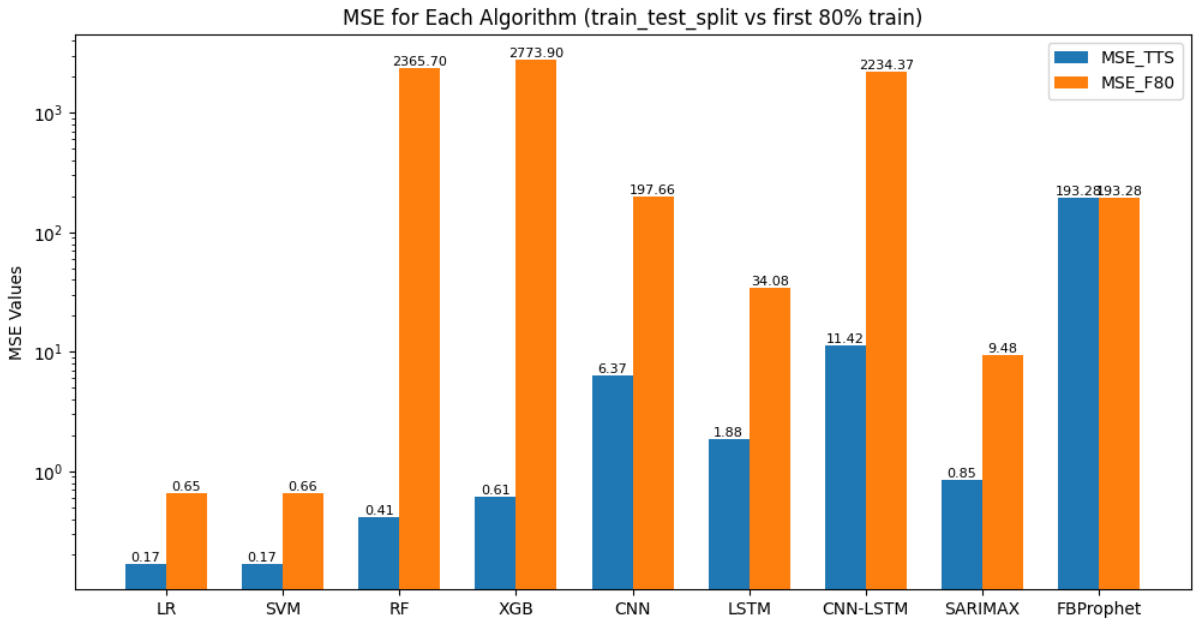
4.1.1 Result from an experiment on Google (GOOGL) Stock Price Dataset

After experimenting with the Google stock price dataset, we got the following performance from different algorithms presented in Table 2.

Table 2: Performance Metrics of Different Algorithms in Google Stock Price Dataset

Algorithm	Mean Squared Error (MSE)		Root Mean Squared Error (RMSE)		Mean Absolute Error (MAE)		Training Time (sec)
	Train Test Split	First 80% Train	Train Test Split	First 80% Train	Train Test Split	First 80% Train	Train Test Split
LR	0.168	0.653	0.410	0.808	0.218	0.626	0.023
SVM	0.167	0.660	0.409	0.812	0.217	0.630	1.283
RF	0.415	2365.704	0.644	48.638	0.325	41.600	2.320
XGB	0.614	2773.901	0.784	52.667	0.374	46.262	2.679
CNN	6.372	197.659	2.524	14.059	1.747	13.114	131.116
LSTM	1.879	34.076	1.371	5.837	0.741	4.745	1934.98
CNN-LSTM	11.416	2234.370	3.378	47.269	2.853	40.717	201.006
SARIMAX	0.846	9.485	0.920	3.079	0.768	2.689	30.902
FB-Prophet	193.277		13.902		10.923		6.552

From the table, the performance of different machine learning, deep learning and time series forecasting algorithms based on their MSE, RMSE, MAE and training time has been observed.



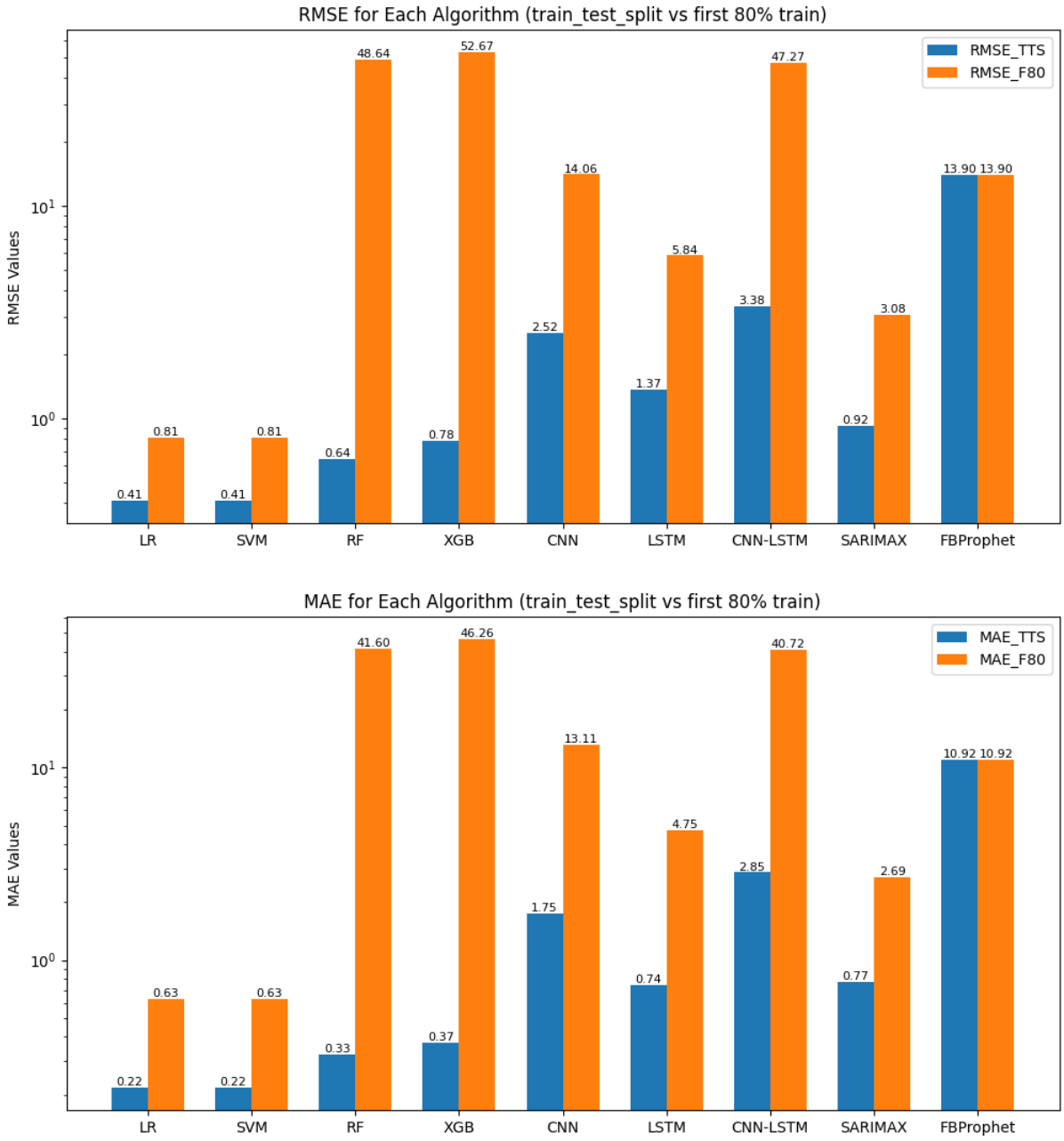


Figure 39: Performance Comparison of Different Model in Google Stock Price Data (train_test_split vs first 80% train)

In the above bar charts, the bar with blue colour is for the machine learning models trained with training data from the **train_test_split()** method, and the bar with orange color is for the machine learning models trained with the first 80% of the dataset. Based on their MSE, RMSE, and MAE, in basic machine learning algorithms, Linear Regression and Support Vector Machines are performing well with the lowest MSE (0.17), RMSE (0.41) and MAE (0.22) trained with training data from **train_test_split0**. In deep learning models, LSTM outperformed all other algorithms with MSE 1.88, RMSE 1.37 and MAE 0.74 with training data from **train_test_split0**. Looking at the models trained with the **first 80% of data**, all performance evaluation metrics are slightly higher than those trained with training data from **train_test_split0**. However, Linear Regression and Support Vector Machine performed better

than other algorithms. Looking at the ensemble learning algorithms, they are performing well when trained with training data from the **train_test_split()** method, but when trained with the first 80% of data, they are not performing well as they have the highest MSE, RMSE and MAE. In time series forecasting algorithms, SARIMAX performs better than FB Prophet with MSE 0.85, RMSE 0.92 and MAE 0.77 trained with training data from **train_test_split().**

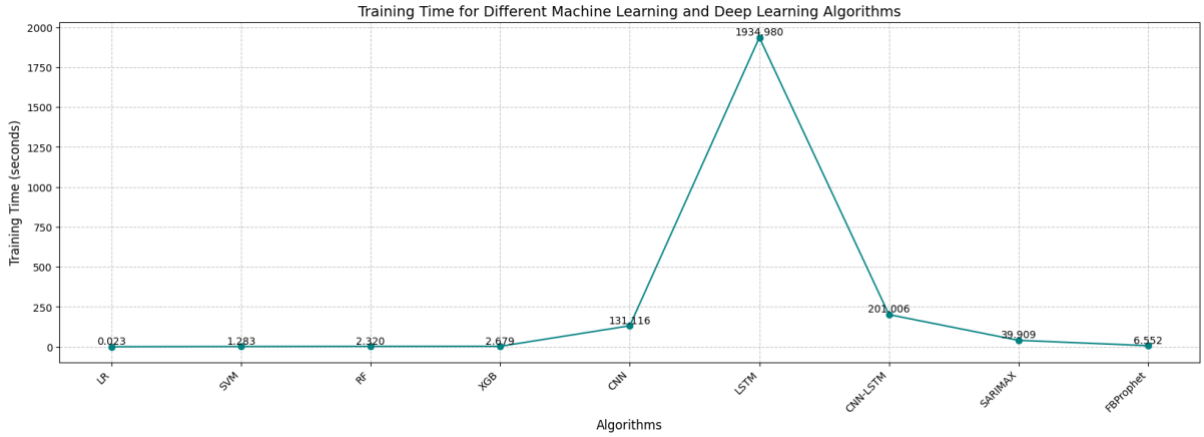


Figure 40: Training Time for Different Machine Learning Models in Google Stock Price Dataset

The figure above illustrates the training times required for each algorithm, highlighting the relative computational efficiency of various methods. The line graph represents different models along the horizontal axis, while the vertical axis indicates the training time in seconds. Among the algorithms, linear regression, being the simplest, has the shortest training duration at just 0.023 seconds. In contrast, LSTM shows the longest training time at 1934.98 seconds, followed closely by the stacked hybrid model CNN-LSTM, which takes 201.006 seconds to train. This extended training time indicates the presence of larger parameter spaces and more complex computations that these models necessitate, often resulting in improved accuracy but also requiring more computational resources. Basic machine learning and ensemble learning algorithms exhibit significantly shorter training durations than deep learning algorithms, which tend to be more efficient in training time due to their simpler nature and lower processing demands.

The line graphs below compare Google's actual closing stock prices to the predicted closing prices generated by machine learning, deep learning, and time series forecasting algorithms. For each algorithm, the first graph displays the results from models trained using the **train_test_split()** method, while the second graph presents outcomes from models trained on the **first 80% of the data**.

1. Linear Regression

Figure 41: Result from Linear Regression (train_test_split() vs first 80% train)

2. Support Vector Machine

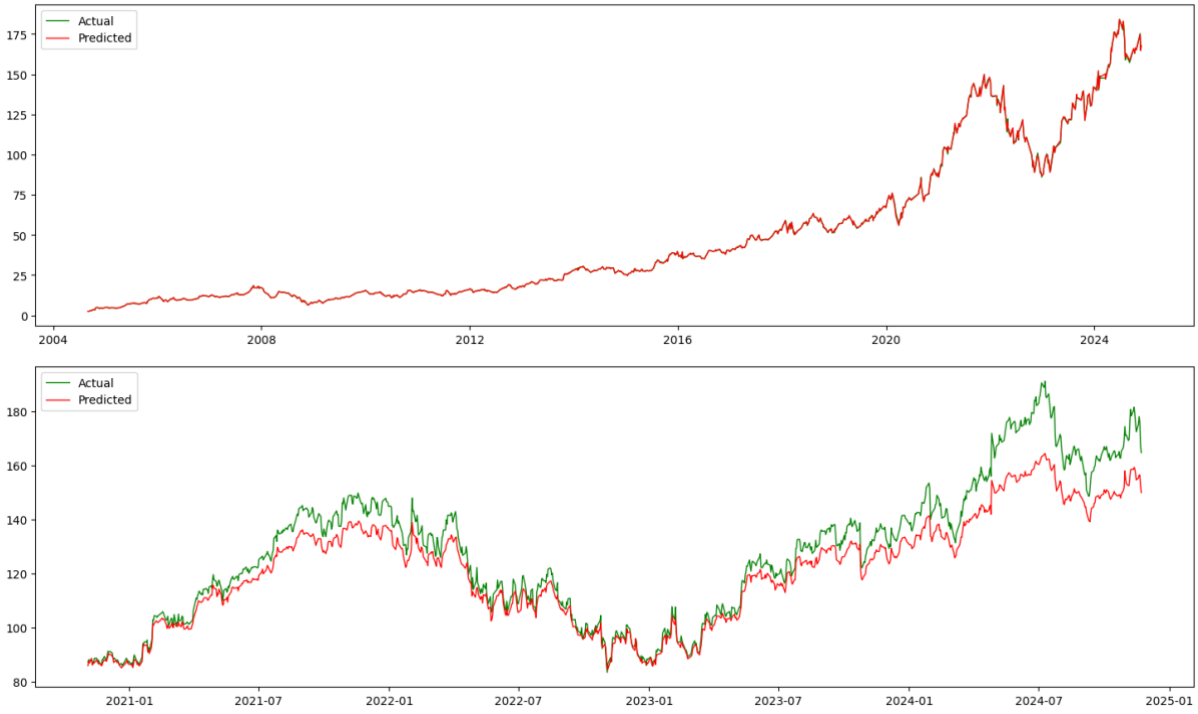


Figure 42: Result from SVM (train_test_split() vs first 80% train)

3. Random Forest

Figure 43: Result from Random Forest (train_test_split() vs first 80% train)

4. XG Boost

Figure 44: Result from XGBoost (train_test_split() vs first 80% train)

5. SARIMAX

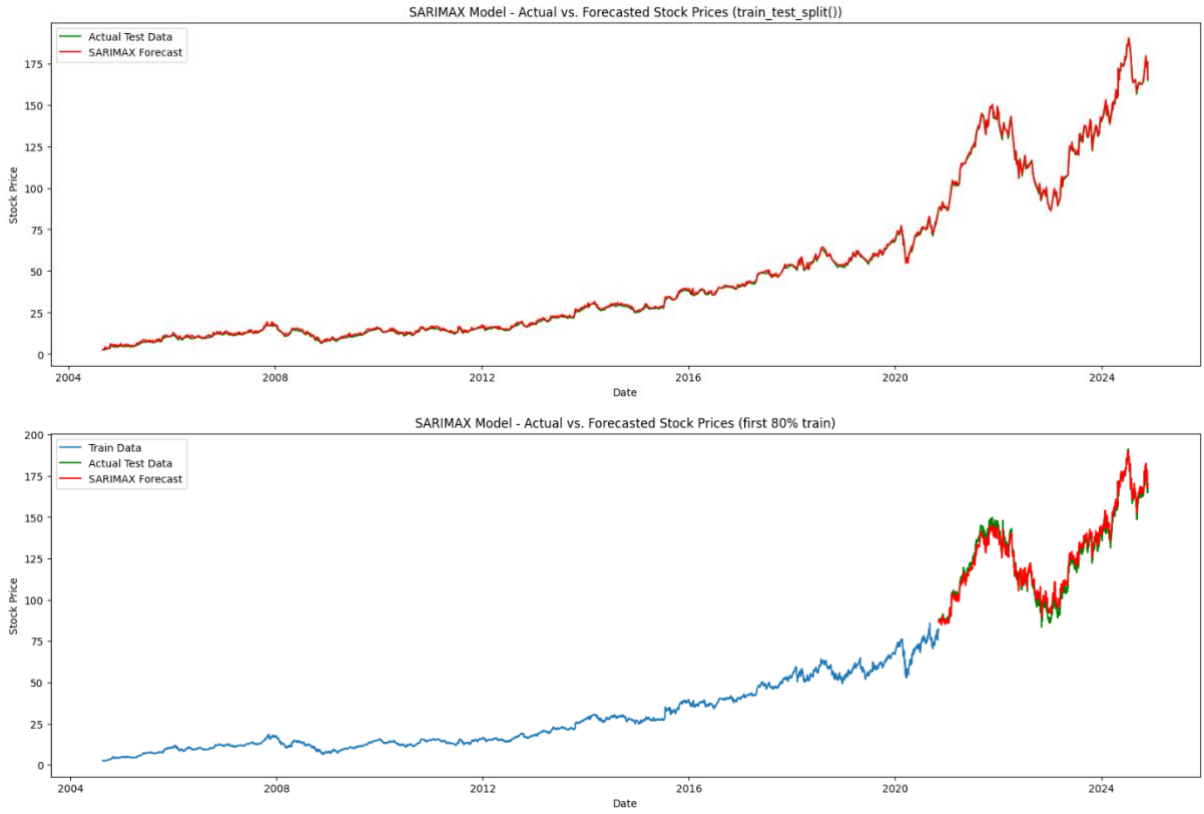


Figure 45: Result from SARIMAX (train_test_split() vs first 80% train)

6. FB Prophet

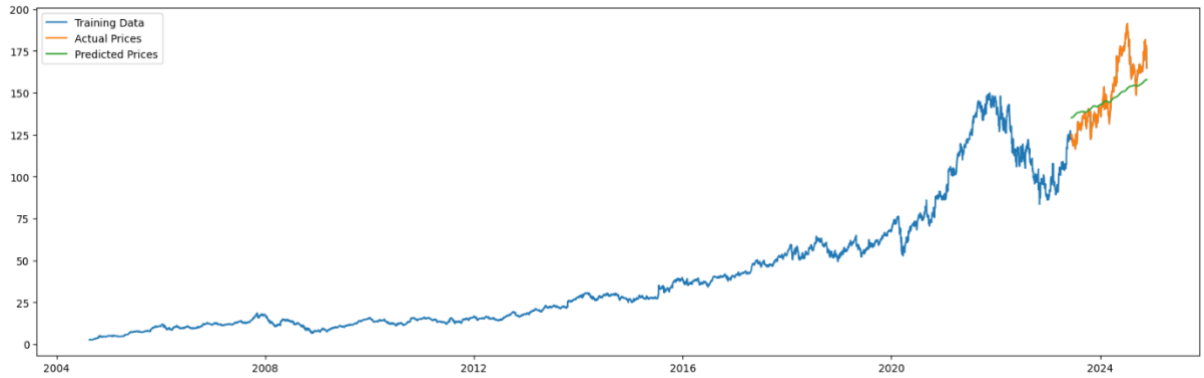


Figure 46: Result from FB Prophet

7. CNN

Figure 47: Result from CNN (train_test_split() vs first 80% train)

8. LSTM

Figure 48: Result from LSTM (train_test_split() vs first 80% train)

9. CNN-LSTM

Figure 49: Result from CNN-LSTM (train_test_split() vs first 80% train)

4.1.2 Result from an experiment on Nabil Bank (NABIL) Stock Price Dataset

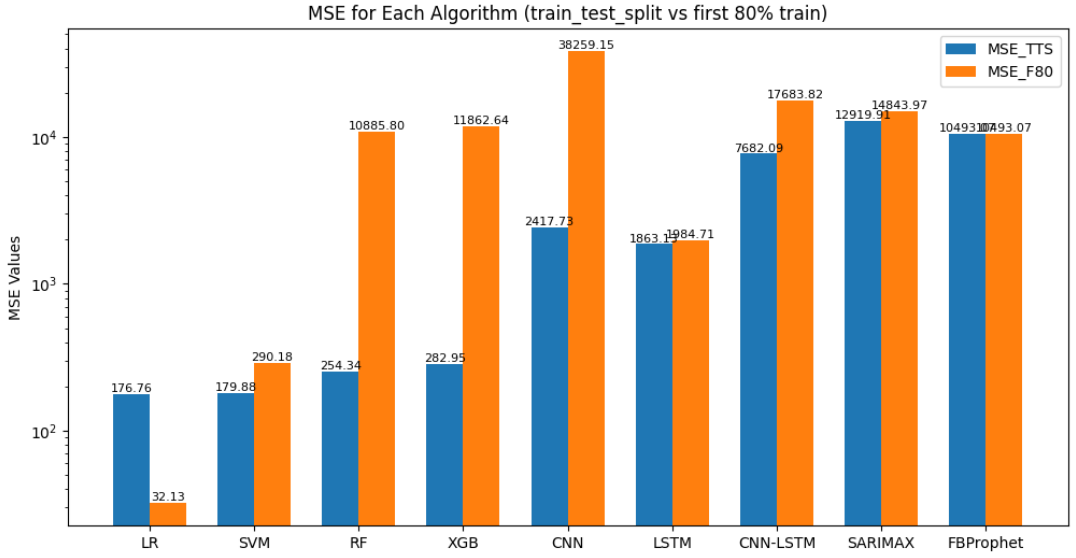
The following performance results are obtained from an experiment performed on Nabil Bank (NABIL) stock price dataset with different machine learning, deep learning and time series forecasting.

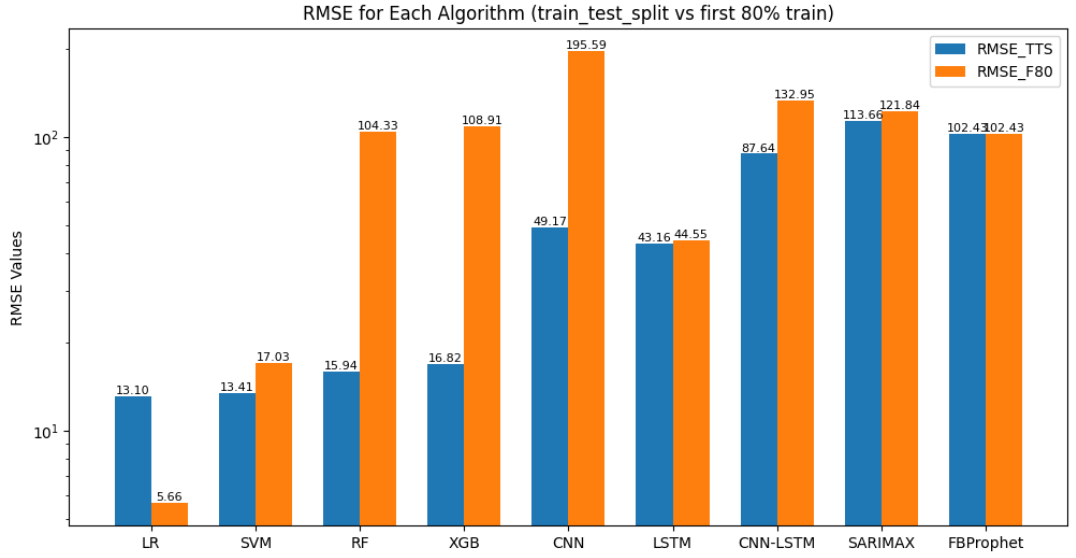
Table 3: Performance Metrics of Different Algorithms in Nabil Bank Stock Price Dataset

Algorithm	Mean Squared Error (MSE)		Root Mean Squared Error (RMSE)		Mean Absolute Error (MAE)		Training Time (sec)
	Train Test Split	First 80% Train	Train Test Split	First 80% Train	Train Test Split	First 80% Train	Train Test Split
LR	176.76	32.13	13.10	5.66	8.76	4.17	0.031
SVM	179.88	290.18	13.411	17.03	8.89	12.22	1.001
RF	254.34	10885.80	15.94	104.33	9.84	71.79	2.255
XGB	282.95	11862.64	16.82	108.91	10.66	76.59	1.234
CNN	2417.73	38259.15	49.17	195.59	34.01	150.12	91.938

LSTM	1863.13	1984.71	43.16	44.55	24.66	39.87	966.430
CNN-LSTM	7682.09	17683.82	87.64	132.95	58.81	111.23	135.355
SARIMAX	12919.91	14843.97	113.66	121.84	99.69	104.39	93.043
FB-Prophet	10493.07		102.43		86.13		6.603

The above tabular data is presented in below graphs where in each bar graphs, performance evaluation metrics MSE, RMSE and MAE of different machine learning models trained with training data from **train_test_split()** method and first 80% data are compared.





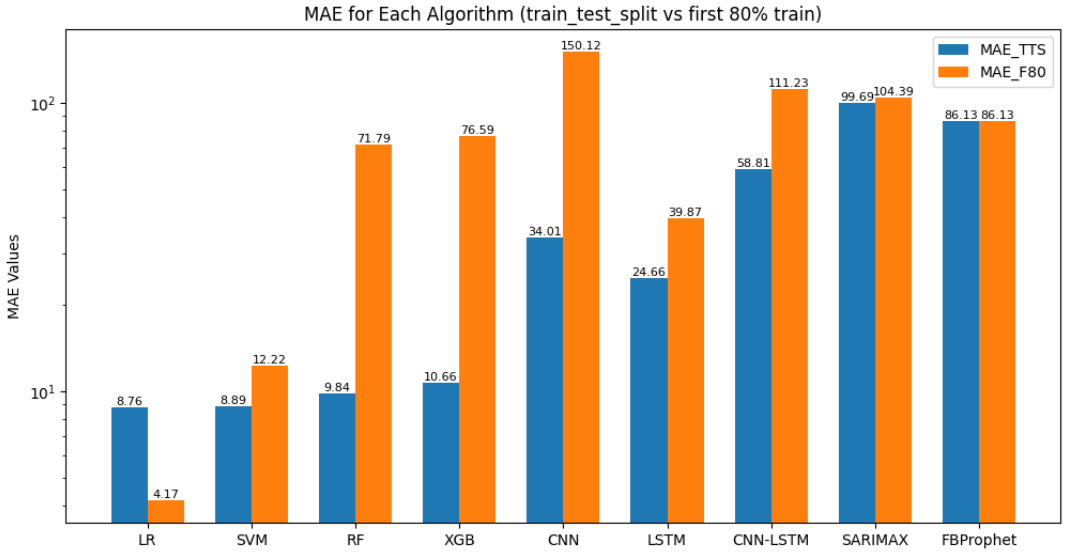


Figure 50: Performance Comparison of Different Model in Nabil Stock Price Data (train_test_split vs first 80% train)

Like in the Google stock price prediction model, a model trained with training data from the **train_test_split()** method outperformed the model trained with the first 80% of data, except for the linear regression model, which had lower MSE, RMSE, and MAE with the model trained with the first 80% of data.

However, if we observe all the performance evaluation metrics of the different machine learning models trained with Nabil Bank stock price, the models do not perform well in this dataset. Although basic machine learning algorithms have outperformed all other algorithms, LSTM performed better than others in deep learning algorithms.

The machine learning models trained with Nabil Bank stock data take the same training time as the models trained with Google stock price data. LSTM, the most basic algorithm among the nine, took the longest time to train, whereas linear regression took the shortest time.

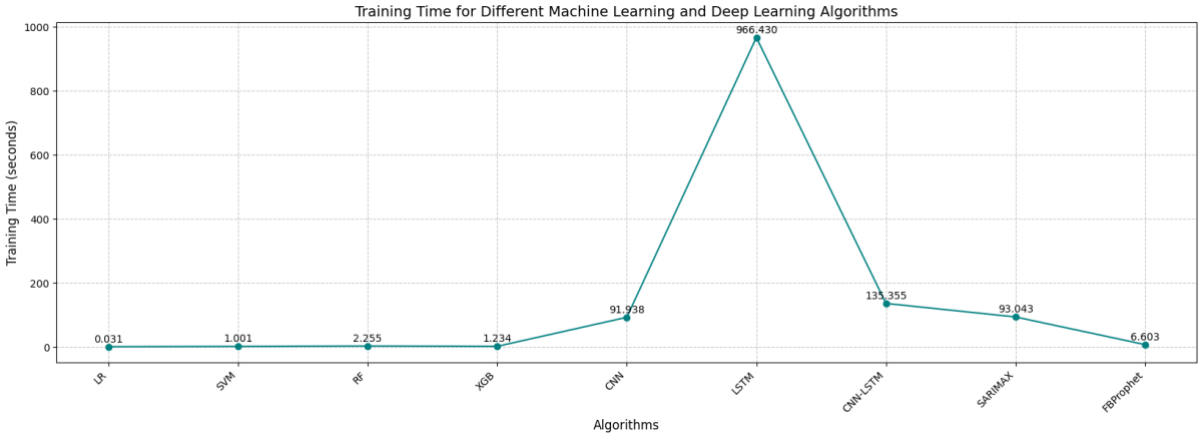


Figure 51: Training Time for Different Machine Learning Models in Nabil Bank Stock Price Dataset

The line graphs below show Google's Stock Actual Close Price vs. Predicted Close Price obtained from different machine learning, deep learning, and time series forecasting algorithms. For each algorithm, the first graph is obtained from the model trained with the

training data from **train_test_split(),** and the second graph is obtained from the model trained with the first 80% of the data as a training dataset.

1. Linear Regression

Figure 52: Result from Linear Regression (train_test_split() vs first 80% train)

2. Support Vector Machine

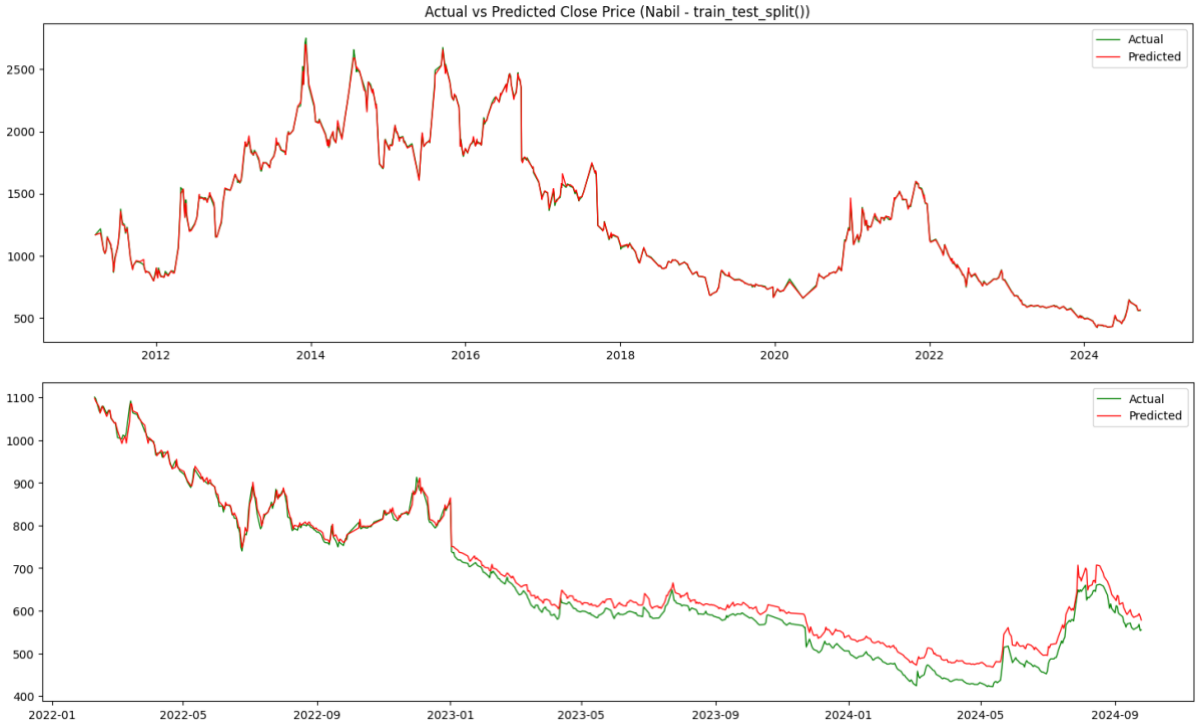


Figure 53: Result from SVM (train_test_split() vs first 80% train)

3. Random Forest

Figure 54: Result from Random Forest (train_test_split() vs first 80% train)

4. XG Boost

Figure 55: Result from XGBoost (train_test_split() vs first 80% train)

5. SARIMAX

Figure 56: Result from SARIMAX (train_test_split() vs first 80% train)

6. FB Prophet

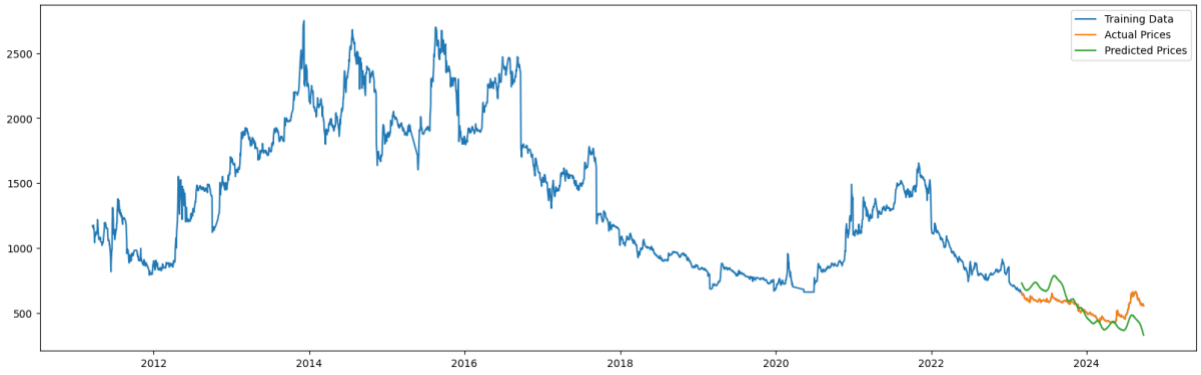


Figure 57: Result from FB Prophet

7. CNN

Figure 58: Result from CNN (train_test_split() vs first 80% train)

8. LSTM

Figure 59: Result from LSTM (train_test_split() vs first 80% train)

9. CNN-LSTM

Figure 60: Result from CNN-LSTM (train_test_split() vs first 80% train)

4.2 Analysis

The findings and the accompanying graph show that the research analysis underscores several key aspects regarding the efficacy of machine learning algorithms in predicting stock prices using two distinct datasets: Google stock prices, which reflect international markets, and Nabil Bank stock prices, sourced from Nepal. The outcomes provide intriguing insights into how machine learning and deep learning algorithms function when utilised on diverse datasets and training approaches.

4.2.1 Performance of Different Models

Fundamental machine learning methods, such as Linear Regression and Support Vector Machines, consistently outperformed more complex models like Random Forest, XGBoost, Convolutional Neural Networks, and Long Short-Term Memory networks. Both datasets showed that LR and SVM achieved lower values for Mean Squared Error, Root Mean Squared Error, and Mean Absolute Error while demonstrating faster training times. On the other hand, ensemble techniques (RF and XGB) and deep learning models (CNN, LSTM, and CNN-LSTM) encountered difficulties, exceptionally when trained with the first 80% of the dataset.

This disparity arises because simpler models like LR and SVM can effectively handle small, noisy datasets by capturing the essential linear or nonlinear relationships without overfitting. In contrast, deep learning models and ensemble methods are sensitive to data quality, requiring large amounts of well-structured and representative data to leverage their strengths fully. Moreover, the results indicate that ensemble learning methods, when trained with the first 80% of the data, fail to generalise effectively to the test set, likely due to their inability to capture

temporal dependencies inherent in stock prices. These methods excel in datasets with richer features or structured patterns, which are not as evident in the Nabil Bank dataset.

4.2.2 Impact of Dataset Characteristics

The fundamental differences in the characteristics of the Google and Nabil Bank datasets significantly influence model performance. The Google stock prices exhibit a more stable and upward trend over time, providing a more precise pattern for the models to learn. This stability likely allows simpler models to fit the data well without overfitting or underfitting. On the contrary, the Nabil Bank stock prices are highly volatile, with frequent and sharp fluctuations. This volatility introduces noise that confuses models, intense learning models, which rely on finding consistent temporal or spatial patterns. As a result, even advanced models struggle to perform well on this dataset, producing high error metrics.

4.2.3 Effect of Training Methodology

The methodology used for training is crucial in influencing the performance of a model. The findings indicate that the train-test split method produced superior results compared to using the initial 80% of the dataset for training. This is especially apparent for deep learning models and ensemble techniques. Utilising the train-test split method allows for random data allocation, which likely provides a more representative sampling for training and testing, thereby minimising overfitting and enhancing the model's generalisation capabilities. Conversely, training on the first 80% of the data adheres to a rigid chronological structure, which can cause models to have difficulty generalising if the test dataset includes patterns or trends not present in the training set.

The train-test split method performed particularly well with deep learning models like LSTM and CNN, which are designed to capture sequential dependencies. This approach better preserves the temporal aspect of stock prices, enabling these models to learn effectively. However, training with the first 80% of the data exposed the limitations of ensemble learning methods (e.g., RF, and XGB), as these models are not inherently designed to handle time-series data.

4.2.4 Why Deep Learning Underperformed

While deep learning models such as LSTM are theoretically ideal for predicting time-series data due to their capability to capture temporal relationships, they underperformed in this analysis. Several reasons played a role in this result. Firstly, deep learning models typically need a large amount of training data to learn effectively, and the datasets utilised in this study might not have been sufficiently large to leverage their full potential. Secondly, these models tend to overfit, mainly when dealing with noisy or highly volatile data like Nabil Bank stock prices. This overfitting is apparent from the elevated error metrics observed in the CNN and LSTM models when applied to this dataset. Lastly, the intricate nature of these models requires meticulous tuning of hyperparameters and preprocessing; if not properly optimised, this can result in subpar performance.

4.2.5 Effect of Stock Price Trends

The trends in stock prices have a profound impact on model building and training. The Google dataset's steady upward trend provided a smoother and more predictable pattern for models to learn, allowing even simpler algorithms to perform well. In contrast, the erratic fluctuations in the Nabil Bank dataset added complexity, requiring models to balance fitting to the observed data while avoiding overfitting to noise. Deep learning models, which rely heavily on-trend patterns, struggled with the inconsistencies in the Nabil Bank dataset, leading to significantly poorer performance than simpler algorithms.

4.3 Discussion

The findings of this research provide several important insights into the effectiveness of machine learning algorithms for stock price prediction. One significant observation is that more straightforward approaches, including Linear Regression and Support Vector Machines, achieved better results than more complex methods like Random Forest, XGBoost, and deep learning techniques (CNN, LSTM, CNN-LSTM) in both datasets analysed. This result is unexpected, given the theoretical advantages of deep learning and ensemble techniques for challenging tasks like stock price forecasting. The better performance of LR and SVM can be linked to their straightforward nature and capacity to generalise well in situations where the data is limited and noisy. In contrast, more sophisticated models are more likely to overfit smaller or highly volatile datasets, negatively affecting their predictive accuracy.

One of the key factors influencing the model performances is the difference in characteristics between the two datasets. The Google dataset exhibited a relatively smooth and upward trend, making it easier for models to learn and predict. Conversely, the Nabil Bank dataset displayed significant volatility and irregular fluctuations, introducing complexity that even advanced models struggled to handle. While designed to capture sequential dependencies, deep learning models like LSTM and CNN failed to perform effectively on the Nabil Bank dataset due to its noise and lack of consistent trends. Additionally, ensemble methods like RF and XGB showed poor performance when trained on the first 80% of data, likely because these methods are not inherently suited for time-series forecasting and could not effectively capture temporal dependencies.

Another important observation relates to the impact of the training methodology. The train-test split method consistently outperformed the first 80% training approach across most models, particularly for deep learning techniques. The randomness in the train-test split method provided a more balanced and representative dataset sample, enabling models to generalise unseen data better. However, training on the first 80% of the data posed challenges for most models, as it exposed them to limited patterns and trends, reducing their ability to predict future fluctuations effectively. These findings underscore the importance of selecting a training methodology and model that aligns with the characteristics and challenges of the dataset. While advanced algorithms have the potential to excel in stock price prediction, they require extensive tuning, sufficient data, and a structured approach to leverage their capabilities thoroughly.

5. Conclusion

5.1 Summary of the Work

5.1.1 Overview

In stock price forecasting, assessing and analysing machine learning algorithms is vital for obtaining reliable and actionable outcomes. This research examined how effective various algorithms, such as Linear Regression, Support Vector Machines, Random Forest, XGBoost, Long Short-Term Memory, Convolutional Neural Networks, and hybrid models like CNN-LSTM, are at predicting stock prices using historical data. The findings indicated that simpler models, like LR and SVM, outperformed more intricate models, including RF, XGB, and deep learning techniques (CNN, LSTM), particularly on the volatile Nabil Bank dataset. These straightforward models exhibited superior generalisation and required fewer computational resources, while more complex models faced issues with overfitting and struggled to identify non-linear patterns in the dataset effectively. Significantly, the train-test split method performed better than the first 80% training approach, especially for deep learning models, as it created a more representative sample for training purposes. While conventional machine learning techniques provided interpretability and efficiency, advanced models like LSTMs and CNNs demonstrated promise but necessitated larger datasets and meticulous tuning to prevent overfitting. This study underscores the significance of aligning model choice with the characteristics of the dataset and training methods, highlighting that simpler models such as LR and SVM can be more effective in situations involving noisy, limited data. Additionally, it points out that stock market prediction is inherently intricate and affected by variables beyond historical data, suggesting that subsequent research could investigate incorporating external elements, such as news sentiment and social media trends, to improve prediction precision.

5.1.2 Linkage to Objectives

This research aimed to evaluate and analyse the effectiveness of various machine learning algorithms in predicting stock prices. The findings directly address this objective by comparing the performance of different models, including Linear Regression, Support Vector Machine, Random Forest, XGBoost, and deep learning models (LSTM, CNN, CNN-LSTM), using two datasets: Google stock prices and Nabil Bank stock prices. The study provided valuable insights into the predictive accuracy, reliability, and computational efficiency of these algorithms, successfully achieving the goal of assessing the performance of different ML models.

Additionally, the study assessed the effectiveness of machine learning models compared to traditional statistical methods, such as SARIMAX, highlighting the strengths and weaknesses of each approach. The findings indicated that simpler models like Linear Regression and Support Vector Machines outperformed more complex algorithms, such as deep learning, in specific scenarios, thereby fulfilling the goal of comparing machine learning techniques with traditional statistical methods. Furthermore, the research identified the most effective machine learning model for predicting stock prices, with LR and SVM demonstrating better

generalisation, especially on datasets with high volatility. This outcome achieved the goal of determining the most efficient machine learning model for stock price forecasting.

Finally, the study examined the effect of different data training strategies, such as the train-test split method versus using the first 80% of the data for training, and how these strategies influenced model performance. The research concluded that the train-test split method provided more balanced and representative training data, which led to better generalisation and predictive accuracy. Thus, the study fulfilled its objective of identifying the effect of data used for training and testing machine learning models.

5.2 Reflection

This study has delivered essential insights regarding using machine learning algorithms for predicting stock prices, outperforming conventional methods in terms of adaptability and effectiveness in fluctuating market conditions. The study emphasised the importance of strong evaluation frameworks and high-quality data for assessing the dependability of predictive models. Although LSTM networks proved to be a promising method, it is crucial to recognise financial markets' complex and uncertain nature. The precision of predictions can be significantly affected by elements such as economic indicators, geopolitical situations, and investor sentiment, which may not be adequately represented by historical data alone.

A constraint of this research is the dependence on historical records, which might not sufficiently account for unexpected occurrences or abrupt changes in the market. Subsequent investigations could examine the merging of real-time news sentiment evaluation and social media information to enhance predictive precision. Furthermore, utilising more advanced deep learning methods, including attention mechanisms and transformer architectures, could significantly improve the effectiveness of predictive models. It's crucial to emphasise that although machine learning provides powerful tools for forecasting stock prices, it should not be the sole basis for making investment choices. A cautious strategy integrates quantitative analysis with qualitative insights from fundamental and technical assessments. Investors can arrive at better-informed choices by grasping the basic factors influencing stock price fluctuations and recognising the constraints of machine learning models.

5.3 Future Work

5.3.1 Research Limitations

Although research has advanced considerably in investigating machine learning algorithms for predicting stock prices, it is crucial to recognise various limitations that can influence the models' accuracy and reliability:

1. **Market Volatility and Uncertainty:**
   - **Unpredictable Events:** Sudden market shocks, economic crises, or geopolitical events can significantly impact stock prices, making accurate predictions challenging.
   - **Time-Varying Patterns:** Stock market trends and patterns can change over time, requiring continuous model adaptation.

2. **Model Complexity and Overfitting**
   - **Overfitting Risk:** Complex models with numerous parameters are more susceptible to overfitting the training data, which results in poor generalisation performance on unseen data.
   - **Computational Cost:** Advanced deep learning models can be resource-intensive to train and deploy, requiring significant computational power and time.

3. **Ethical Considerations and Responsible AI:**
   - **Fairness and Bias:** Models may unintentionally reinforce biases embedded in the training data, resulting in biased or unfair outcomes.
   - **Transparency and Interpretability:** Black-box models can be difficult to interpret, making it challenging to understand their decision-making process.

4. **Real-world Application and Practical Constraints:**
   - **Transaction Costs and Market Impact:** Real-world trading involves transaction costs, slippage, and market impact, which can affect the profitability of trading strategies.
   - **Regulatory and Compliance:** Adhering to regulatory requirements and compliance standards can impose additional constraints on investment strategies.

5.3.2 Recommendations for Future Research and Improvements

1. **Incorporating External Factors:**
   - **News Sentiment Analysis:** Analyze news articles and social media sentiment to capture market sentiment and its impact on stock prices.
   - **Economic Indicators:** To enhance predictive models, incorporate macroeconomic indicators such as GDP growth, interest rates, and inflation rates.

2. **Advanced Deep Learning Techniques:**
   - **Attention Mechanisms:** Attention mechanisms help the model focus on important parts of the input sequence, improving its ability to capture long-term dependencies.
   - **Transformer Models:** Investigate the application of transformer models, which are known for their success in natural language processing and computer vision tasks, to identifying intricate patterns within time series data.

3. **Hybrid Models:**
   - **Combining Technical and Fundamental Analysis: To develop hybrid models, i**ntegrate technical indicators (such as moving averages and RSI) with fundamental aspects (like earnings and dividends).

4. **Real-time Prediction and Adaption:**
   - **Online Learning:** Develop models that can adjust to evolving market conditions by continuously updating and learning from new data.
   - **Real-time Data Streams:** Process real-time data streams to make timely predictions and adjust strategies accordingly.

References

A, A., R, R., S, V. R. & Bagde, A. M., 2023. Predicting Stock Market Time-Series Data using CNN-LSTM Neural Network Model. *TechRxiv.*

Agarwal, N. B. & Yadav, D. K., 2024. A Comprehensive Analysis of Cassical Machine Learning and Modern Deep Learning Methodologies. *International Journal of Engineering Research & Technology (IJERT),* 13(5).

Alakh, 2024. *Support Vector Regression Tutorial for Machine Learning, Analytics Vidhya.* [Online]
Available at: https://analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/
[Accessed 2 October 2024].

Ankit, 2022. *Generate Quick and Accurate Time Series Forecasts using Facebook's Prophet (with Python & R codes), Analytics Vidhya.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2018/05/generate-accurate-forecasts-facebook-prophet-python-r/
[Accessed 2 October 2024].

Baheshti, N., 2022. *Random Forest Regression: A Basic Explanation and Use Case in 7 Minutes, Medium.* [Online]
Available at: https://towardsdatascience.com/random-forest-regression-5f605132d19d
[Accessed 2 October 2024].

Bajracharya, B. B., 2019. *Nepal's Safest Stock in a Volatile Share Market.* [Online]
Available at: https://www.smartfamily.com.np/finance/nepals-safest-stock-in-a-volatile-share-market
[Accessed 20 July 2027].

Biswal, A., 2024. *Stock Market Prediction using Machine Learning in 2024, Simplilearn.* [Online]
Available at: https://www.simplilearn.com/tutorials/machine-learning-tutorial/stock-price-prediction-using-machine-learning
[Accessed 23 September 2024].

Box, G. E. P., Jenkins, G. M., Reinsel, G. C. & Ljung, G. M., 2015. *Time Series Analysis: Forecasting and Control.* 5th ed. New York: Wiley.

Brownlee, J., 2021. *A Gentle Introduction to Ensemble Learning Algorithms, Machine Learning Mastery.* [Online]
Available at: https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/
[Accessed 1 October 2024].

Calzone, O., 2022. *An Intuitive Explanation of LSTM.* [Online]
Available at: https://medium.com/@ottaviocalzone/an-intuitive-explanation-of-lstm-

a035eb6ab42c
[Accessed 3 October 2024].

Chen, L.-P., 2020. Using Machine Learning Algorithms on Prediction of Stock Price. *Journal of Modeling and Optimization,* 12(2), pp. 84-99.

Codecademy Team, 2024. *What is Scikit-Learn?*. [Online] Available at: https://www.codecademy.com/article/scikit-learn [Accessed 24 September 2024].

Crabtree, M., 2023. *What is Machine Learning? Definition, Types, Tools & More.* [Online] Available at: https://www.datacamp.com/blog/what-is-machine-learning?dc_referrer=https%3A%2F%2Fwww.google.com%2F [Accessed 25 July 2024].

Fama, E. F., 1970. Effective Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance,* 25(2), pp. 383-417.

Fernando, J., 2024. *Moving Average (MA): Purpose, Uses, Formula, and Examples.* [Online] Available at: https://www.investopedia.com/terms/m/movingaverage.asp [Accessed 1 October 2024].

GeeksforGeeks, 2024. *Introduction to Matplotlib.* [Online] Available at: https://www.geeksforgeeks.org/python-introduction-matplotlib/ [Accessed 24 September 2024].

Graham, B. & Dodd, D., 1934. *Security Analysis.* 1st ed. s.l.:McGraw-Hill Book Co..

Gunduz, H., Yaslan, Y. & Cataltepe, Z., 2017. Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations. *Knowledge-Based Systems,* Volume 137, pp. 138-148.

Guresen, E., Kayakutlu, G. & Daim, T. U., 2011. Using artificial neural network models in stock market index prediction. *Expert Systems with Applications,* 38(8), pp. 10389-10397.

Hayes, A., 2024. *Autoregressive Integrated Moving Average (ARIMA) Prediction Model.* [Online]
Available at: https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp
[Accessed 2 October 2024].

Hopman, M., 2021. *Stock and Crypto Market Values, Unsplash.* [Online] Available at: https://unsplash.com/photos/red-and-blue-light-streaks-fiXLQXAhCfk [Accessed 2 October 2024].

Huang, Y., Capretz, L. F. & Ho, D., 2021. Machine Learning for Stock Prediction Based on Fundamental Analysis. *IEEE Symposium Series on Computational Intelligenc,* pp. 01-10.

Jain, A., 2024. *A Comprehensive Guide to Performance Metrics in Machine Learning.* [Online] Available at: https://medium.com/@abhishekjainindore24/a-comprehensive-guide-to-

performance-metrics-in-machine-learning-4ae5bd8208ce#:~:text=Performance%20metrics%20play%20a%20crucial,its%20performance%20across%20various%20tasks.
[Accessed 6 November 2024].

Jordan, J., 2017. *Hyperparameter tuning for machine learning models..* [Online] Available at: https://www.jeremyjordan.me/hyperparameter-tuning/
[Accessed 2 October 2024].

Kabir, M. H., Sobur, A. & Amin, M. R., 2023. Stock Price Prediction Using The Machine Learning. *International Journal of Creative Research Thoughts (IJCRT),* 11(7), pp. 946-950.

Kavita, 2024. *Linear Regression: A Comprehensive Guide, Analytics Vidhya.* [Online] Available at: https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/
[Accessed 1 October 2024].

Kenton, W., 2024. *Excess Kurtosis: Definition, Types, Example: Investopedia.* [Online] Available at: https://www.investopedia.com/terms/e/excesskurtosis.asp
[Accessed 25 July 2024].

Khanna, M. et al., 2022. Performance Evaluation of Machine Learning Algorithms for Stock Price and Stock Index Movement Prediction Using Trend Deterministic Data Prediction. *International Journal of Applied Metaheuristic Computing,* 13(1), pp. 1-30.

K, P. et al., 2021. Comparative Study: Stock Price Prediction Using Fundamental and Technical Analysis. *2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC),* pp. 1-4.

Kumar, A., 2021. *Stock Price Prediction using Machine Learning Techniques, Analytics Yogi.* [Online]
Available at: https://vitalflux.com/popular-machine-learning-techniques-for-stock-price-movement-prediction/
[Accessed 23 September 2024].

Kumar, R., 2022. *What is Numpy and How it works? An Overview and Its Use Cases? - DevOps School.* [Online]
Available at: https://www.devopsschool.com/blog/what-is-numpy-and-how-it-works-an-overview-and-its-use-cases-2/
[Accessed 24 September 2024].

Kumbure, M. M., Lohrmann, C., Luukka, P. & Porras, J., 2022. Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications,* 197(2022), pp. 1-41.

Lawrence, S. J., 2023. *What is LSTM? - Introduction to Long Short-Term Memory.* [Online] Available at: https://www.scaler.com/topics/deep-learning/lstm/
[Accessed 03 October 2024].

Matsuo, K., 2022. *Selenium Automation Testing - A Complete Guide for 2024, headspin.* [Online]
Available at: https://www.headspin.io/blog/selenium-testing-a-complete-guide
[Accessed 26 September 2024].

Mehtab, S. & Sen, J., 2019. A Robust Predictive Model for Stock Price Prediction Using Deep Learning and Natural Language Processing. *Proceedings of the 2019 International Conference on Business Analytics and Intelligence (ICBAI 2019),* 1 December.

Melanie, 2024. *Facebook Prophet : All you need to know, DataScientest.* [Online]
Available at: https://datascientest.com/en/facebook-prophet-all-you-need-to-know
[Accessed 2 October 2024].

Olumide, S., 2023. *Root Mean Square Error (RMSE) In AI: What You Need To Know, Arize.* [Online]
Available at: https://arize.com/blog-course/root-mean-square-error-rmse-what-you-need-to-know/
[Accessed 1 October 2024].

Otten, N. V., 2024. *Support Vector Regression (SVR) Simplified & How To Tutorial In Python, Spot Intelligence.* [Online]
Available at: https://spotintelligence.com/2024/05/08/support-vector-regression-svr/
[Accessed 2 October 2024].

Pathak, A. & Pathak, S., 2020. Study of Machine Learning Algorithms for Stock Market Prediction. *International Journal of Engineering Research & Technology (IJERT),* 9(06), pp. 295-300.

Practicus AI, 2019. *Understanding the 3 most common loss functions for Machine Learning Regression, Medium.* [Online]
Available at: https://towardsdatascience.com/understanding-the-3-most-common-loss-functions-for-machine-learning-regression-23e0ef3e14d3
[Accessed 01 October 2024].

Qayyum, R., 2022. *Computational Complexity of Machine Learning Algorithms, Medium.* [Online]
Available at: https://rafayqayyum.medium.com/computational-complexity-of-machine-learning-algorithms-254c275de84
[Accessed 1 October 2024].

Raju, S. S. et al., 2023. A Three-Dimesional Approach for Stock Prediction Using AI/ML Algorithms: A Review & Comparision. *2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT),* pp. 1-6.

Rizvi, D. R. & Khalid, M., 2024. Performance Analysis of Stocks Using Deep Learning Models. *5th International Conference on Innovative Data Communication Technologies and Application - ICIDCA,* 233(2024), pp. 753-762.

Ronaghan, S., 2019. *Data Understanding for Machine Learning: Assessment & Exploration, Medium.* [Online]
Available at: https://towardsdatascience.com/data-understanding-for-machine-learning-assessment-exploration-aca1aadc1cb6
[Accessed 24 September 2024].

Saadeddin, Z., 2024. *ARIMA for Time Series Forecasting: A Complete Guide, DataCamp.* [Online]
Available at: https://www.datacamp.com/tutorial/arima
[Accessed 2 October 2024].

Saud, A. S. & Shankya, S., 2019. Analysis of Gradient Descent Optimization Techniques with Gated Recurrent Unit for Stock Price Prediction: A Case Study on Banking Sector of Nepal Stock Exchange. *Journal of Institute of Science and Technology,* 24(2), pp. 17-21.

Sen, J., Mehtab, S. & Dutta, A., 2021. Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models. *TechRxiv (Powered by: IEEE).*

Shrestha, P., 2021. *Application of Machine Learning and Deep Learning Techniques for Nepal Stock Market Prediction.* [Online]
Available at: https://www.researchgate.net/publication/353634645_Application_of_Machine_Learning_and_Deep_Learning_Techniques_for_Nepal_stock_market_price_prediction
[Accessed 20 July 2024].

Sonkavde, G. et al., 2023. Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systemic Review, Performance Analysis and Discussion of Implications. *International Journal of Financial Studies,* 11(3), p. 22.

Statista, 2024. *Total market capitalizations of companies listed on stock exchanges worldwide from 2013 to 2023.* [Online]
Available at: https://www.statista.com/statistics/274490/global-value-of-share-holdings-since-2000/
[Accessed 15 August 2024].

Taylor, S., 2024. *Skewness, Corporate Finance Institute.* [Online]
Available at: https://corporatefinanceinstitute.com/resources/data-science/skewness/#:~:text=However%2C%20skewed%20data%20will%20increase,large%20losses%20on%20the%20investment.
[Accessed 25 July 2024].

Tretina, K., 2024. *What Is The Stock Market? How Does It Work?.* [Online]
Available at: https://www.forbes.com/advisor/in/investing/what-is-stock-market/
[Accessed 24 July 2024].

Uddin, M. R., 2023. *COVID-19 Pandemic Data Analysis and Prediction Using Machine-Learning Algorithms, Research Gate.* [Online]
Available at: https://www.researchgate.net/publication/374010617_COVID-

19_Pandemic_Data_Analysis_and_Prediction_Using_Machine-Learning_Algorithms [Accessed 01 October 2024].

Vapnik, V. N., 1999. *The Nature of Statistical Learning Theory.* s.l.:Springer.

Willems, K., 2022. *Pandas Tutorial: DataFrames in Python, datacamp.* [Online] Available at: https://www.datacamp.com/tutorial/pandas-tutorial-dataframe-python [Accessed 24 September 2024].