PREDICTING INDIAN AND GLOBAL STOCK INDICES BY APPLYING MACHINE LEARNING TECHNIQUES

THESIS
submitted in partial fulfillment of the requirements
for the award of the degree of

DOCTOR OF PHILOSOPHY
in MANAGEMENT
By

MOHIT BENIWAL

2K18/PhD/DSM/07
Delhi School of Management

Under the supervision of

Dr. ARCHANA SINGH
Delhi School of Management
Delhi Technological University

Prof. NAND KUMAR
Department of Humanities
Delhi Technological University

DELHI SCHOOL OF MANAGEMENT
DELHI TECHNOLOGICAL UNIVERSITY
MAIN BAWANA ROAD,
SHAHABAD DAULATPUR
DELHI 110042, INDIA 2022

Candidate's Declaration

I, hereby, certify that the thesis titled "**PREDICTING INDIAN AND GLOBAL STOCK INDICES BY APPLYING MACHINE LEARNING TECHNIQUES**" and submitted in fulfillment of the requirements for the award of the degree of Doctor of Philosophy is an authentic record of my research work carried out under the guidance of Dr. Archana Singh and Dr. Nand Kumar.

The matter presented in this thesis has not been submitted elsewhere in part or fully to any other university or institute for the award of any degree.

Mohit Beniwal
2K18/PhD/DSM/07
Delhi School of Management
Delhi Technological University

DELHI TECHNOLOGICAL UNIVERSITY
(Govt. of National Capital Territory of Delhi)
MAIN BAWANA ROAD, SHAHABAD DAULATPUR, DELHI
110042, INDIA

Certificate

This is to certify that the thesis titled "**PREDICTING INDIAN AND GLOBAL STOCK INDICES BY APPLYING MACHINE LEARNING TECHNIQUES**" submitted by Mohit Beniwal **(2K18/PhD/DSM/07)** in fulfillment of the requirements for the award of the degree of Doctor of Philosophy is an authentic research work carried out by him under our guidance and supervision. The contents embodied in this thesis had not been submitted by her earlier to any institution or organization for any degree or diploma to the best of our knowledge and belief.

Dr. Archana Singh
Delhi School of Management
Delhi Technological University

Dr. Nand Kumar
Department of Humanities
Delhi Technological University

Acknowledgment

I am humbled and deeply grateful to the divine and benevolent supreme goddess of fortune, Radha Rani, and the supreme lord Krishna, for bestowing upon me the ability to undertake my Ph.D. research.

I take immense pleasure in expressing my deepest gratitude to my research supervisor, Dr. Archana Singh, whose unwavering support and guidance have been invaluable throughout my research journey. Her constant motivation and dynamic mentoring approach continue to inspire me. I would also like to extend my heartfelt appreciation to my joint supervisor, Dr. Nand Kumar, for his continuous motivation, valuable feedback, and unwavering support in my research endeavors. His mentoring and encouragement have played a pivotal role in my academic pursuits.

I am indebted to my beloved mother, Smt. Rajwati Devi, and father, Shri Jai Bhagwan, whose blessings and unwavering belief in me have propelled me forward in life and my career. A special mention of gratitude goes to my loving wife, Smt. Shweta, whose unwavering support and understanding have been instrumental in enabling me to dedicate the necessary time and effort to pursue my Ph.D. I am thankful to my younger brother, Dr. Rohit Beniwal, and his wife, Smt. Meenu, for supporting me throughout the journey. I would also like to thank the dear children in our family, Devesh, Gauravi, and Vedant, whose presence and love have brought immense joy and fortune into my life. It was their precious time that I occasionally borrowed to fulfil my academic pursuits.

I am also thankful to all my friends and well-wishers who have stood by me during this challenging yet rewarding journey. Their encouragement and support have been an immense source of strength. Lastly, I express my profound gratitude to all the researchers, scholars, and authors whose work and contributions have served as a guiding light throughout my Ph.D. endeavor. May the blessings of the divine continue to illuminate my path in all future endeavors.

Mohit Beniwal

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Full Form
ADF	Augmented Dickey-Fuller
AI	Artificial Intelligence
AIC	Akaike Information Criterion
a-m-LSTM-o	Average True range Momentum Long Short-Term Memory online
ANN	Artificial Neural Network
ARCH	Auto-Regressive Conditional Heteroskedasticity
ARIMA	Auto-Regressive Integrated Moving Average
ATR	Average True Range
atr-LSTM-o	Average True range Long Short-Term Memory online
AutoML	Automated Machine Learning
B&H	Buy-and-Hold
Bi-LSTM	Bi-Directional Long Short-Term Memory
BP	Back Propagation
CNN	Convolutional Neural Network
DAX	Deutscher Aktienindex
DJIA	Dow Jones Industrial Average
DL	Deep Learning
DNN	Deep Neural Network
DT	Decision Trees
EMH	Efficient Market Hypothesis
fma-LSTM-o	Four Moving Averages Long Short-Term Memory Online
GA	Genetic Algorithm
GARCH	Generalized Auto-Regressive Conditional Heteroskedasticity
GA-SVR	Genetic Algorithm Support Vector Regression
GDP	Gross Domestic Product
GRU	Gated Recurrent Unit
GS	Grid-Search
GS-SVR	Grid Search Support Vector Regression
*h*MoM	Historical Momentum
TSE	Tokyo Stock Exchange
KNN	K-Nearest Neighbors
LightGBM	Light Gradient Boosting Machine
LSE	London Stock Exchange
LSTM	Long Short-Term Memory
LSTM-o	Long Short-Term Memory Online
MA	Moving Averages
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MLP	Multilayer Perceptron
MOM	Momentum
MSE	Mean Squared Error
NI225	Nikkei 225

NASDAQ	National Association of Securities Dealers Automated Quotations
NB	Naive Bayes
NIFTY	National Stock Exchange Fifty
NSE	National Stock Exchange
NYSE	New York Stock Exchange
OGA-SVR	Optimized Genetic Algorithm Support Vector Regression
PSO	Particle Swarm Optimization
RF	Random Forests
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
RWH	Random Walk Hypothesis
SSE	Shanghai Stock Exchange
SVM	Support Vector Machine
SVR	Support Vector Regression
XGBoost	eXtreme Gradient Boosting

1 Introduction

With the widespread availability of mobile phones and internet access, there has been a significant increase in stock market participation across the globe. In India alone, the number of demat accounts has reached a staggering 110 million (TimesofIndia, 2023). As individuals seek opportunities for excessive returns, many are turning to technical analysis as a means to navigate the complexities of the stock market. Moreover, professional traders have long relied on technical trading strategies. The emergence of Artificial Intelligence (AI) and Machine Learning (ML) has made a substantial impact on the financial landscape, offering the ability to analyze vast amounts of data and uncover new insights and patterns that can unlock opportunities for financial enthusiasts.

This chapter aims to provide an introductory background on the stock market and explore the methods of analyzing and forecasting stock market movements, prices, and trends. It also discusses the motivation behind the study, outlines the challenges in the field, and presents the problem statement that this research seeks to address. Furthermore, it outlines the organization and structure of the thesis, providing a roadmap for the subsequent chapters.

1.1 A Brief History of the Stock Market

Stock trading has its roots in ancient civilizations. The earliest example of relatively organized equities can be traced back to the 2nd century BC (B. M. Smith, 2004). The modern stock market can trace its origins back to the 17th century. The first formal stock exchange, the Amsterdam Stock Exchange, was established in 1602 when the Dutch East India Company as a joint stock company (C. F. Smith, 1929) issued shares to raise capital

(Petram, 2011). The stockholders of the company were guaranteed a share of the profit from the earnings (Michie, 2001). The London Stock Exchange (LSE) was established in 1698 and quickly became a prominent center for stock trading (Johannessen & Johannessen, 2017). The LSE played a crucial role in financing the Industrial Revolution and acted as a model for subsequent stock exchanges worldwide.

The United States played a significant role in the development of modern stock markets. In 1792, the New York Stock Exchange (NYSE) was founded by a group of brokers (Mexmonov, 2020) who regularly met under a buttonwood tree on Wall Street (Eames, 1894). The NYSE grew in prominence, facilitating the trading of stocks and bonds. The latter half of the 20th century witnessed significant technological advancements that revolutionized stock trading. The first electronic trading systems were introduced in the National Association of Securities Dealers Automated Quotations (NASDAQ) stock exchange in 1968, which enabled faster and more efficient transactions in the 1970s and 1980s (Heckman, 2013). The advent of the internet further transformed the stock market, allowing individuals to trade stocks online and opening new opportunities for retail investors.

Stock markets became increasingly interconnected as globalization progressed. Cross-border investments and the growth of international stock exchanges allowed investors to diversify their portfolios and participate in global markets. Emerging markets have also gained prominence, with countries like China and India experiencing rapid stock market growth. Today, the stock market is a complex and dynamic system that plays a critical role in global economies. Stock exchanges operate worldwide, providing a platform for

companies to raise capital and investors to trade securities. Every day, stock exchanges witness the trading of assets valued at billions of dollars (Hoseinzade & Haratizadeh, 2019a). The stock market's influence extends beyond economic indicators, with market movements and investor sentiment impacting various aspects of society. It is important for investors and traders to gain insight through a forecasting system that accurately predicts stock indices.

Stock forecasting is a complex task as many factors influence stock prices, such as market trends, company financials, economic indicators, investor sentiment, geopolitical events, regulatory changes, and technological advancements. In academia, the predictability of stock prices remains a subject of debate. The Efficient Market Hypothesis (Fama, 1970) and Random Walk Hypothesis (Malkiel, 1973) propose that stock prices cannot be predicted, as they contend that prices incorporate all relevant information and follow a random trajectory. Conversely, alternative theories, like Technical Analysis (Achelis & Steven, 2013), argue that stock prices do display discernible patterns and trends based on historical data, implying that past patterns can offer insights into future price movements. This ongoing discourse on stock predictability has spurred extensive research and the creation of diverse forecasting models and techniques to investigate the potential predictability of stock prices. Other methods that are utilized to analyze securities are fundamental analysis, statistical analysis, and machine learning.

1.2 Efficient Market Hypothesis

According to the Efficient Market Hypothesis (EMH), the price of an asset incorporates all available information regarding its intrinsic value. Share prices accurately and promptly reflect all available information (R. Kumar, 2016). This theory suggests that it

is difficult for investors to consistently outperform the market and achieve abnormal or excessive risk-adjusted returns. The concept of market efficiency implies that it is challenging to identify mispriced assets or predict future price movements based on publicly available information. Investors cannot consistently exploit market inefficiencies to achieve sustained above-average returns. The EMH implies that any attempts to outperform the market are more likely to be the result of luck than skill.

The EMH has been the subject of extensive academic research and debate. Proponents argue that the efficiency of the market is driven by the collective actions of numerous market participants, who quickly process and incorporate new information into the price. Critics of the EMH, on the other hand, argue that certain market anomalies, behavioral biases, or insider trading can lead to temporary inefficiencies that can be exploited for profit. Factors such as market sentiment, insider trading, irrational behavior, and informational asymmetry can still impact price movements and lead to temporary deviations from fundamental value. The studies (Chowdhury et al., 1993; Pettit & Venkatesh, 1995) examined the returns achieved by insiders at companies. In both studies, it was found that insiders consistently and significantly obtained abnormal returns, indicating a departure from the EMH.

1.3 Random Walk Hypothesis

The Random Walk Hypothesis (RWH) is closely associated with the Efficient Market Hypothesis (EMH). A “Random Walk” refers to a price series where each subsequent price change is considered a random deviation from previous prices (Malkiel, 2003b). In an efficient market, new information is quickly and accurately incorporated into asset prices.

Consequently, the price change in the market tomorrow will solely reflect the news that emerges tomorrow and will be independent of today's price fluctuations. This implies that future price movements are not influenced by past prices, and the unpredictable nature of news further adds to the uncertainty. As a result, price changes resulting from unforeseeable news events are expected to follow a random pattern and cannot be reliably predicted.

While the Random Walk Hypothesis has its critics and is not universally accepted, it has contributed to the development of index investing and passive investment strategies. It has also prompted research and debate in the field of finance regarding the efficiency and predictability of financial markets. According to (Lo & MacKinlay, 1998), stock prices do not adhere to a random walk pattern. Their study challenged the assumptions of the Random Walk Hypothesis by examining the empirical behavior of stock prices. They conducted extensive analysis of the random walk model, providing compelling evidence for its conclusive rejection throughout the entire sample period from 1962 to 1985. Moreover, this rejection holds true across multiple subperiods and encompasses a wide range of aggregate return indices as well as portfolios sorted by size.

1.4 Fundamental Analysis

Fundamental analysis refers to determining the intrinsic value of shares in stock markets. This involves employing a general framework to analyze expected economic forecasts, particularly focusing on sectors that are expected to generate growth in sales and profits (Wafi et al., 2015). By assessing the financial strength of companies, the efficiency of their management, and current market conditions, one can measure the potential for business opportunities. To determine the fair value of a stock, historical financial statements

are analyzed alongside with current market conditions. This analysis helps assess the intrinsic value of the stock. Additionally, fundamental analysts may consider macroeconomic factors, such as interest rates, inflation, and industry-specific trends, to gauge the overall health of the market and the potential opportunities or risks associated with an investment.

The goal of fundamental analysis is to identify assets that are either undervalued or overvalued relative to their intrinsic worth. By comparing the estimated intrinsic value of an asset with its current market price, fundamental analysts can make investment decisions, such as buying, selling, or holding an asset. Fundamental analysis is often used by long-term investors who seek to assess the underlying fundamentals of an asset and make investment choices based on their analysis of its potential future performance. Lev & Thiagarajan (1993) suggested that information obtained from analyzing the fundamentals of a company has an influence on how the market responds to earnings announcements.

1.5 Technical Analysis

Technical analysis involves the study of price and volume primarily using charts, with the objective of predicting future price trends (Lohrmann & Luukka, 2019; Murphy, 1999). Charts display price movements over time, allowing analysts to visually identify patterns and trends. Many accomplished traders highly value the use of charts in their trading activities (Schwager, 1995). Hardly anyone in the financial world ignores technical analysis (Roberts, 1959). By examining patterns and trends in price charts, technical analysts aim to uncover opportunities for buying or selling assets at advantageous moments. Charles Dow

is credited with developing the Dow Theory around 1900, which forms the foundation of modern technical analysis (Achelis, 2001).

The underlying principle of technical analysis is that market prices reflect all available information, including fundamental factors and market sentiment. It is important to note that technical analysis is based on the assumption that historical price patterns tend to repeat themselves, indicating the existence of identifiable market trends. However, technical analysis is highly subjective. Different individuals analyzing the same charts may perform different technical analyses and draw distinct trend lines. In fact, even the same person, when presented with the same chart at different times, may draw varying trend lines (DeMark, 1994). In terms of expected returns, technical analysis cannot outperform the performance of a buy-and-hold strategy (Fama, 1965; Kumbure et al., 2022; Leigh et al., 2002). Despite its shortcomings, technical analysis remains widely used by traders and investors to complement their decision-making process. It offers a systematic approach to analyzing price data and can provide valuable insights into market behavior and potential trading opportunities.

1.6 Statistical Analysis

Statistical analysis plays a crucial role in the field of finance, particularly when it comes to analyzing and understanding the behavior of financial time series. Financial time series data, such as stock prices, trading volumes, and other market variables, contain valuable information. In this context, statistical methods and techniques are employed to analyze and interpret these data, uncover patterns and trends to gain insights into the underlying dynamics of the financial markets.

Statistical analysis in the stock market involves the utilization of various statistical tools and measures. The descriptive analysis provides a summary of the data, offering key statistics such as mean, median, standard deviation, and skewness. Correlation analysis helps identify relationships between different variables, revealing the degree of association between them. Regression analysis, on the other hand, explores the causal relationships between variables, allowing for the prediction of one variable based on others. Hypothesis testing provides a framework for making inferences about population parameters and drawing conclusions from sample data.

Within the realm of financial time series analysis, specific statistical methods are employed to gain deeper insights. Autocorrelation analysis examines the presence of the dependence of current values on past values, which can help identify predictable patterns. Time series decomposition separates a time series into its various components, such as trend, seasonality, and residual elements, providing a clearer understanding of the underlying patterns and dynamics. Auto-Regressive Integrated Moving Average (ARIMA) models capture the linear relationships between past and current values, incorporating autoregressive, moving average, and differencing components. Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models are used to model and forecast volatility in financial time series, taking into account the time-varying nature of volatility.

1.7 Machine Learning

Artificial intelligence has ushered in a transformative revolution in various industries, and the field of finance is no exception. Especially after the advent of AI tools like ChatGPT, the focus on artificial intelligence has taken center stage. The ability to

analyze large volumes of data, uncover hidden patterns, and generate insights at a rapid pace has not only increased efficiency and productivity but has also opened up new possibilities for innovation and discovery. Machine learning is an interdisciplinary field that focuses on developing computational algorithms capable of automatically extracting meaningful patterns and insights from empirical data. It has emerged from the intersection of traditional statistical analysis and artificial intelligence research (Edgar & Manz, 2017).

The field of finance has witnessed a notable rise in publication trends, reflecting the extensive utilization of AI and ML across diverse domains. Within this context, AI and ML have found widespread applications in areas such as bankruptcy prediction, stock price forecasting, portfolio management, oil price prediction, anti-money laundering measures, behavioral finance analysis, big data analytics, and blockchain technology. These advancements highlight the growing significance of AI and ML in shaping the future of finance and driving innovation within the industry (Ahmed et al., 2022).

Machine learning models can analyze historical stock data, market trends, and other relevant factors to forecast future price movements. The integration of machine learning algorithms in finance has the potential to enhance decision-making processes and improve risk management for financial analysts, traders, and investors. Machine learning algorithms can be broadly categorized into two types: supervised and unsupervised learning algorithms (Madge & Bhatt, 2015). Supervised learning algorithms are machine learning algorithms that learn from labeled data, where each data point is associated with a corresponding label or target variable. The algorithm learns by mapping input features to their corresponding output labels based on the provided training examples. It generalizes from the labeled data

to make predictions on new, unlabeled data. On the other hand, unsupervised learning algorithms operate on unlabeled data, where the objective is to discover patterns, structures, or relationships within the data. These algorithms do not have access to explicit target variables or labels. Unsupervised learning can be used for tasks such as clustering, dimensionality reduction, and anomaly detection.

Artificial Neural Networks (ANN) and Support Vector Machines (SVM) have emerged as widely used supervised ML algorithms extensively employed to predict and analyze the stock market and its future movements (Chhajer et al., 2022; Selvamuthu et al., 2019). LSTM, which stands for Long Short-Term Memory, is a specialized type of artificial neural network (ANN) architecture, specifically designed to model and capture long-term dependencies and patterns in sequential data. Similarly, Support Vector Regression (SVR) is a variant of Support Vector Machines (SVM) that is specifically designed for regression analysis. However, while SVM is primarily used for classification tasks, SVR focuses on predicting continuous numeric values.

In recent years, the Deep Learning (DL) computing paradigm has attained significant recognition as the leading approach in the ML community. Over time, it has progressively established itself as the predominant computational approach within the field of machine learning (Alzubaidi et al., 2021). Deep learning technology, which originated from artificial neural networks (ANN), falls under the umbrella of machine learning (ML) and artificial intelligence (AI) and is now widely recognized as a foundational technology (Sarker, 2021a). Among the prominent deep learning architectures, this study uses the Deep Neural Network (DNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM),

Bidirectional LSTM (Bi-LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN).

DNN is an artificial neural network characterized by multiple hidden layers situated between the input and output layers. It follows a feed-forward architecture, where information flows through the network in a unidirectional manner from the input layer to the output layer (Y. Qian et al., 2014). DNNs are known for their ability to learn hierarchical representations of complex data. RNNs, on the other hand, consist of neurons with recurrent connections that serve as memory, allowing them to learn the temporal dynamics from sequential data (Farsi, 2021). The primary challenge with standard RNNs lies in the problems of exploding and vanishing gradient (Y. Wang et al., 2021). LSTM is a type of RNN that addresses this issue of vanishing or exploding gradients, enabling it to capture long-term dependencies in sequential data. It achieves this through memory cells and gates that control the flow of information. Bi-LSTMs enhance the capability of LSTMs by incorporating information from both past and future contexts, enabling a richer understanding of the data.

GRU is similar to LSTM, and both address the vanishing gradient problem (J. Zhang et al., 2021). They employ reset and update gates to control the flow of information, making them more computationally efficient than LSTMs while maintaining good performance. CNNs, on the other hand, are primarily used for image classification tasks (Yadav & Jadhav, 2019). They leverage convolutional layers to extract local patterns and features from the input data, enabling them to capture spatial relationships effectively. However, CNN can also be used for stock time series forecasting. Figure shows the schematic diagram of AI.

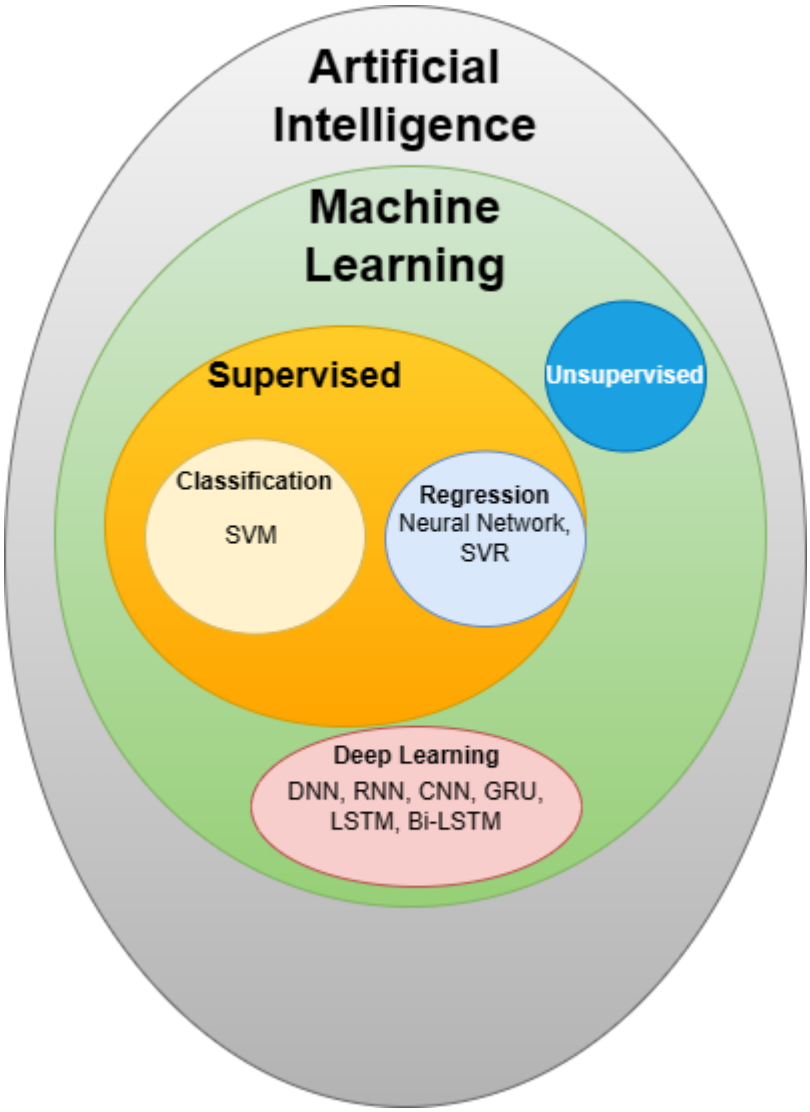

Figure 1.1 Schematic Diagram of AI

1.8 Motivation

The task of predicting the stock market is widely recognized for its complexity, which presents significant challenges to achieving accurate forecasts. With the advent of digitization and globalization, investors and institutions now have easier access to investing in global stock indices. This necessitates the development of a robust and data-driven

forecasting system that can cater to the needs of investors and traders seeking informed investment decisions. Notably, emerging economies like India and China have demonstrated attractive returns on assets, with several Fortune 500 companies originating from these regions. Additionally, developed economies such as the USA, Germany, and Japan boast established stock exchanges, further emphasizing the importance of accurate global forecasting. These counties are top five GDPs of the world (WorldData.info, 2021).

The figure shows the cumulative stock returns of top 5 GDPs. It can be noted that Nifty from India outperforming other world indices in recent times.

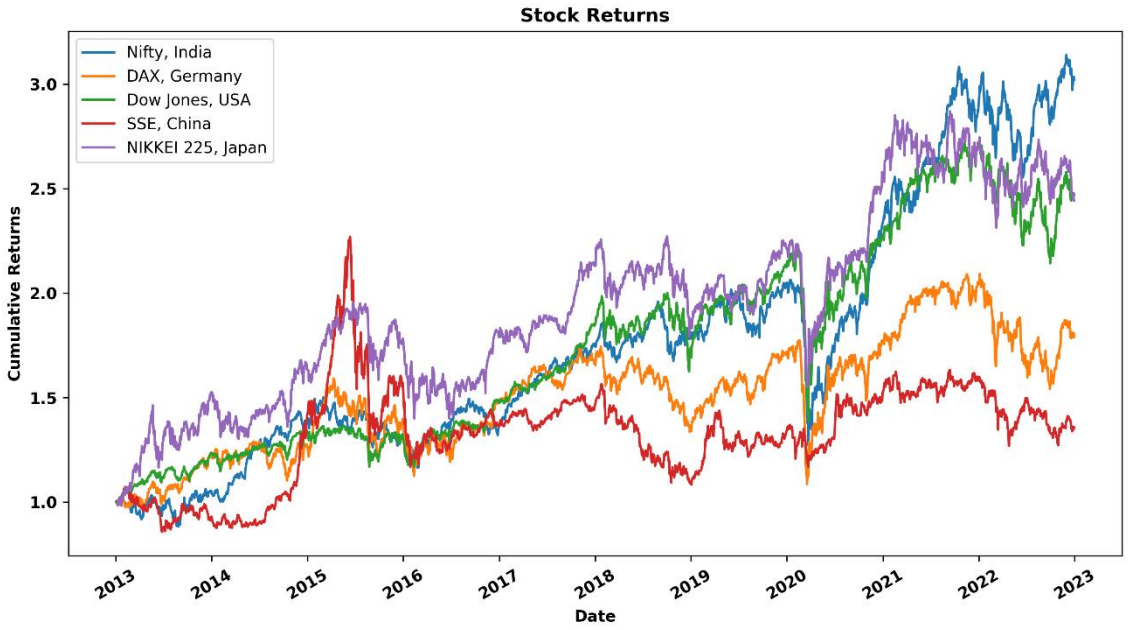

Figure 1.2 Cumulative Stock Return of Top 5 GDPs

The academic community holds varying opinions regarding the effectiveness of stock market prediction. The Efficient Market Hypothesis and Random Market Hypothesis propose that obtaining excessive returns through stock market prediction is impossible.

Statistical tests indicate that future stock price movements are independent of past prices, undermining the assumptions of technical analysis, which relies on the repetition of past patterns. However, the emergence of machine learning has revolutionized data analysis and opened new possibilities for identifying underlying patterns in the financial domain. This study aims to develop a forecasting system capable of achieving excess returns over the market return. The dynamics of emerging and developing economies are different. The stock market is known as a barometer of the national economy. The study further aims to compare machine learning techniques with statistical techniques in emerging and developed economies.

Accurate forecasting plays a crucial role in informed decision-making and risk mitigation. In machine learning, predictions are typically focused on the immediate next steps, which may not be sufficient for long-term planning. Additionally, machine learning algorithms have numerous hyperparameters that require careful tuning, as their performance is sensitive to these settings. SVM, SVR, DNN, RNN, LSTM, GRU, and other algorithms all possess multiple hyperparameters. Therefore, a multistep forecasting approach is essential and beneficial for decision-making purposes, especially for the investor community. Machine learning and deep learning algorithms have demonstrated their ability to tackle complex problems. In this research, using machine learning, the study aims to develop a system capable of multistep long-term forecasting where hyperparameters are automatically tuned using evolutionary algorithms like genetic algorithms or grid search. The study further aims to evaluate the long-term forecasting abilities of deep learning models.

1.9 Challenges

The challenges encountered in this thesis revolve around the inherent complexities and uncertainties associated with financial time series. The noise and complexity inherent in these time series make accurate forecasting a daunting task. Stock prices are influenced by numerous factors, and trends can be volatile and rapidly changing. Additionally, the presence of randomness in stock prices cannot be ignored. Furthermore, the availability of public information for all market participants makes it challenging to consistently achieve excess returns. The evolving nature of national economies adds another layer of complexity to the analysis. Therefore, a forecasting system is needed that can adapt to the dynamic nature of financial time series.

The noisy nature of financial time series also poses difficulties in accurately predicting their behavior. Machine learning algorithms require careful tuning of hyperparameters, and manual parameter selection may introduce biases into the research. To address this, an automated system is required to select the optimal parameters without human intervention. Moreover, the performance of a forecasting system can vary significantly across different financial time series. A model that performs well in one series may not perform effectively in another series. Developing a robust and generalized system capable of handling diverse financial time series from different economies presents a significant modeling challenge.

1.10 Problem Statement

The challenges discussed above motivated us to develop a machine-learning forecasting system that is robust to handle different financial time series stock indices from the top five GDPs of the world. The top five GDPs of the world are the USA, China, Japan, Germany, and India (WorldData.info, 2021). Following are the problem statements to overcome the above challenges:

- There is a need to develop a system that leverages Neural Networks and technical analysis methods to generate higher returns for investors over the long term compared to the buy-and-hold strategy.
- To make informed investment decisions, it is crucial to compare statistical techniques and machine learning in forecasting stock prices for both short and long terms, considering the distinct dynamics of emerging and developed countries.
- A multistep prediction system is required to address the needs of investors who are seeking long-term price and pattern information, as machine learning predictions typically focus on the next-step predictions.
- Clarity is needed to determine which deep learning algorithm is most suitable for long-term multistep forecasting of financial time series, considering that deep learning can solve complex tasks and there are numerous algorithms available for forecasting financial time series.

1.11 Significance of the Study

The significance of this study lies in addressing the challenges associated with predicting stock market movements and developing a robust forecasting system that can

generate higher returns for investors over the long term. The study also aims to contribute to the ongoing debate within the academic community regarding the effectiveness of stock market prediction. The study shows a clear comparison of machine learning techniques with statistical techniques in predicting global stock indices.

Furthermore, the study recognizes the importance of accurate forecasting in facilitating informed decision-making and risk mitigation. By focusing on multistep forecasting, the research aims to provide investors with long-term price and pattern information, overcoming the limitations of machine learning models that primarily focus on immediate next steps. The research also acknowledges the challenges posed by hyperparameter tuning in machine learning algorithms and aims to develop an automated system that optimizes these parameters using evolutionary algorithms. It also recognizes the challenge of developing a robust and generalized system capable of handling diverse financial time series from different economies and locations.

1.12 Organization of the Thesis

Chapter 1: Introduction

This chapter provides an overview of the stock market, including its brief history and key concepts. It also discusses the methods of analysis and prediction used in the field, along with the motivation behind the study and the challenges encountered. Additionally, it presents the problem statement and the significance of the study.

Chapter 2: Literature Review

This chapter conducts a comprehensive review of the existing literature on forecasting financial time series. The gaps and limitations in the literature are identified, and

the research questions and objectives of the study are defined. This chapter sets the foundation for the research and establishes its relevance within the existing body of knowledge.

Chapter 3: Trading Framework using LSTM and Technical Analysis

This chapter proposes a trading framework that utilizes LSTM and technical analysis techniques to outperform the buy-and-hold strategy. The framework is designed to capture patterns and trends in financial time series data, enabling more effective decision-making for investors.

Chapter 4: Performance Comparison of ARIMA and SVR

The focus of this chapter is to compare the performance of iterative and static machine learning techniques such as Support Vector Regression with statistical methods such as ARIMA in forecasting stock prices in both emerging and developed economies. By evaluating and analyzing the results, insights into the strengths and limitations of each approach are gained.

Chapter 5: Long-term Price Forecasting using Optimized GA and SVR

This chapter addresses the need for long-term price forecasting by proposing a model based on optimized Genetic Algorithms and Support Vector Regression (SVR). The model aims to forecast the multistep prices of global stock indices, taking into account the dynamic nature of the financial markets and the complexities involved.

Chapter 6: Deep Learning Models for Long-term Price Forecasting

This chapter compares the performance of various deep learning models, such as DNN, RNN, LSTM, Bi-LSTM, GRU, and CNN, for multistep long-term price forecasting

of global stock indices. The analysis provides insights into the suitability and effectiveness of these models for capturing complex patterns and trends.

Chapter 7: Conclusion and Future Work

The final chapter concludes the study by summarizing the key findings and contributions. It also discusses the limitations of the research and suggests areas for future exploration and improvement. This chapter provides a comprehensive closure to the study and outlines the potential for further advancements in stock market forecasting.

2 Literature Review

In this chapter, an extensive literature review is conducted on various machine learning algorithms used for forecasting stock movement and prices. The field of machine learning offers a wide range of algorithms, including but not limited to Logistic Regression, Decision Trees, Random Forests, Gradient Boosting (e.g., XGBoost, LightGBM), Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Naive Bayes, Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU). Each algorithm possesses its own unique advantages and disadvantages, making it important to understand its characteristics and suitability for specific forecasting tasks. By examining the existing literature, this review aims to provide insights into the performance and applicability of these algorithms in the context of predicting stock market behavior.

2.1 Predictability of Stock Market

The efficient market and random walk have long been cherished in academia. It is well-accepted that stock returns are essentially unpredictable. In the words of Barber, "Trading is hazardous to your wealth (Barber & Odean, 2000)." However, investors and traders kept using fundamental and technical analysis to invest and trade in the stock market. Benjamin Graham once said that a stock market is a voting machine in the short run, but in the long run, it is a weighing machine (Graham & David, 1965). It is also observed that stock index prices tend to exhibit an upward bias over time.

The efficient market hypothesis does not affiliate fundamental, technical, or sentimental analysis. Malkiel (2003a) argues that it is impossible to beat market returns in the long term without accepting more significant risks. He suggests that there are no predictable patterns, and if there are, they are self-destructive and won't generate extra returns after transaction costs in the long term. However, many studies disagree with Malkiel's claims and suggest that the market is predictable to some extent (Poterba & Summers, 1988). Pesaran & Timmermann (1995) examined the predictability of the US stock market and concluded that in a volatile market, the predictability increased. Christoffersen (2006) conducted a simulation experiment on various data frequencies and horizons and concluded that directional predictability exists. Nyberg (2011) forecasted the direction of the US stock market using the binary probit model. He found that returns are better than buy-and-hold for in-sample data. However, for out-of-sample data, the results were weak. The model predicted the signs of returns better than the Autoregressive moving average and volatility models in out-of-sample data. Gu & Peng (2019) developed a forecasting model based on probability density to predict the direction of market returns. They applied their model to the Shanghai Composite Index and CITIC securities. Their model gave a higher return for out-of-sample data than the buy-and-hold strategy.

Cochrane H., (1999), in his paper "New Facts in Finance," recognized that stock returns do have predictability over a long-term horizon. Lo & MacKinlay (1998) examined the random walk hypothesis for weekly stock market returns by comparing variance estimators derived from data sampled at different frequencies. The researchers found that the random walk model was strongly rejected for the entire sample period (1962-1985) and

for all subperiods, encompassing various indexes of aggregate returns and portfolios sorted by size. The rejections were primarily driven by the behavior of small stocks, but they could not be solely attributed to the effects of infrequent trading or time-varying volatilities. Furthermore, the rejection of the random walk for weekly returns did not support a mean-reverting model of asset prices.

The increasing application of soft computing techniques has sparked significant interest in academia regarding the prediction of stock market movements using Artificial Intelligence (AI) and Machine Learning (ML) methodologies. While the stock market is widely recognized for its inherent risks, the potential for accurate financial time series prediction presents lucrative opportunities for investors. By harnessing the power of AI and ML, researchers and practitioners aim to uncover patterns and relationships within stock market data that can lead to more informed investment decisions and potentially yield substantial returns.

2.2 Long Short-Term Memory (LSTM)

Recently, the application of ML algorithms in predicting stock market prices has been a focal point in academic research within the field of time series analysis. Artificial Neural Networks (ANNs) have gained popularity in stock prediction due to their ability to learn complex patterns and make generalizations. Teixeira Zavadzki de Pauli et al. (2020) compared the different architectures of ANNs to predict Brazilian stocks. They compared the performance of five Neural Network (NN) architectures to predict the close prices of six major stocks in the Brazilian stock exchange. The models were trained on historical stock prices to predict the next day's closing price. The models were evaluated based on the Root

Mean Square Error (RMSE). The results showed that all architectures performed well except the radial basis function (RBF) based model. Recurrent Neural Networks (RNNs) are a type of ANN specially developed to handle sequential data, such as time-series data. Yang et al. (2021) proposed a hybrid model with an improved Particle Swarm Optimization (PSO) algorithm and RNN to predict stock prices with better accuracy. The proposed algorithm, a combination of a NN and an adaptive adjustment of inertial weight, enhanced its global search ability in the early stage and its local search ability in the later stage. The results indicated the practical effectiveness of the system. However, RNNs suffer from a vanishing gradient problem (Kolen & Kremer, 2001).

LSTM, a variant of RNN, is specifically designed to overcome the challenge of vanishing gradient that RNNs suffer. Moreover, the most frequently utilized ML algorithm for predicting financial time series is LSTM (Henrique et al., 2018). Mehtab et al. (2021) proposed a hybrid modeling approach to predict stock prices using both ML and deep learning-based models. In this study, they used the NIFTY 50 index prices to construct eight regression models to predict the multi-step future open values of NIFTY 50 for the upcoming week. Four deep learning-based regression models were also built using LSTM networks with walk-forward validation and optimized hyperparameters. Their results showed that the LSTM-based univariate model that used one-week prior data as input was the most accurate model among all. Zhang et al. (2021) applied a neural network and Back Propagation (BP) algorithm to forecast intraday stock prices by utilizing transaction data for five consecutive days. The BP algorithm neural network showed a prediction accuracy of 73.29%, whereas the deep learning fuzzy algorithm had an accuracy of 62.12%. They found that the best

prediction range was 15 days, and the BP algorithm NN outperformed the deep learning fuzzy algorithm in terms of accuracy. Lu et al. (2021) proposed a CNN-BiLSTM-AM model for predicting stock prices by combining CNN, bi-directional LSTM (Bi-LSTM), and attention mechanisms (AM). Their model predicted the stock closing prices of the next day for the Shanghai Composite Index and compared the performance with seven other models as the benchmark. The results demonstrated that the CNN-BiLSTM-AM model was superior to the other models, with the smallest Mean Absolute Error (MAE), RMSE, and the largest R^2.

Some studies prefer predicting direction instead of price. Zhang et al. (2022) predicted the direction and developed a trading strategy. Further, they compared the performance of multiple models. They used Logistic Regression, SVM, Gradient-Boosting Decision Trees, RF, and LSTM. However, they experimented with the models on a single index, namely the Shanghai Stock Exchange (SSE) index. They proposed a methodology that considered both intraday and interday dynamics of the stock market during the 2015 stock market crisis in China. The empirical results demonstrated that the market can be predicted. Kumar and Haider (2021a) developed a hybrid mechanism that was a fusion of RNN-LSTM and combined it with metaheuristic optimization techniques to predict intra-day stock market trends. They introduced two hybrid approaches in their study. The proposed models provided a systematic method for automatically generating an optimized network, resulting in a more precise learning process with minimized error rates and improved accuracy. They evaluated the efficacy of the optimized RNN-LSTM network using data from

six different stock exchanges, and the results showed that the metaheuristic approach increased the forecasting accuracy by approximately 4-6%.

Liu & Lee (1997) presented an intelligent expert system based on technical analysis. Their system recommended signals based on rules defined in the system. The net return after transaction costs was 22%, which was better than the inflation rate and bank rate at that time. Neural Networks overcome the shortcoming of linear methods such as Linear Regression (LR) or Multiple Linear Regression (MLR). They can solve non-linear and complex problems, which are difficult to solve with linear models. The neural networks can be combined with other statistical, evolutionary, or forecasting techniques to make them more robust. Bao et al. (2008) also developed an intelligent stock trading system based on turning points and probabilistic model. They experimented with their system on S&P 500 components for around 3000 daily data. They achieved great returns and concluded that their system was universally profitable for most stocks. Nayak et al. (2017) proposed a Neural Network model optimized using an Artificial Chemical Reaction (ACRNN). They compared their model with the Multilayer Perceptron (MLP), Radial Bias Function Neural Network (RBFNN), and Multiple Linear Regression (MLP). They concluded that ACRNN shows greater accuracy than the other model considered for all seven indices.

Chung & Shin (2018) investigated the performance of LSTM and the Genetic Algorithm (GA) on the Korean Stock Price Index (KOSPI). They predicted the index price for the next day and compared it with a benchmark model, which predicts no change in the price. They found that the hybrid model of LSTM-GA outperformed their benchmark model based on the evaluation parameters MAE, MSE, and Mean Absolute Percentage Error

(MAPE). Chen & Ge (2019) explored the attention mechanism in the Long–Short-Term Memory (am-LSTM) network in the Hong Kong stock market. They concluded that it significantly enhances prediction power when compared to LSTM. Nikou et al. (2019a) analyzed the recurrent network method with an LSTM block function. They determined that the model predicts the close price of iShares MSCI United Kingdom better than the other models, such as Support Vector Regression (SVR) and Random Forest (RF). Lu et al. (2020) proposed a novel CNN-LSTM-based model. They compared the CNN-LSTM model with the CNN-RNN, LSTM, RNN, and MLP models. They concluded that CNN-LSTM could be trusted to predict the stock market and perform better than other compared models.

Optimization problems are computationally intensive. Researchers are focusing on nature-inspired metaheuristic optimization algorithms that do not require the computation of the gradient of the objective function. Khan et al. (2020) applied a nature-inspired metaheuristic optimizer known as Beetle Antennae Search (BAS) to the portfolio selection problem. Their portfolio selection included transaction costs and cardinality constraints. They compared the results with those of other optimizers such as Particle Swarm Optimizer (PSO), Pattern Search (PS), Interior Point Search, and Genetic Algorithm (GA). While achieving the same level of performance, their algorithm was six times faster in the worst case and twenty-five times faster in the best case, compared to other algorithms.

In summary, the available literature is divided into opinions about whether the stock market can be predicted. One school favors the efficient market hypothesis and the random walk theory, while another favors some predictability. The first school suggests passively investing in index-traded funds, whereas the other school advocates actively predicting the

market for excess returns. Even if the stock market has a certain predictability, it remains a highly challenging and complex task to predict the stock market efficiently. Many studies focused on a single index, such as the S&P, NASDAQ, etc., or a basket of stocks. Further, most studies limit themselves only to developed economies, limiting the forecasting model's ability to efficiently predict emerging market indices. Emerging economies may have different dynamics.

Further, some studies prefer predicting stock prices instead of direction. Leung et al. (2000) compared classification models that predicted directions to level estimation models that predicted returns. They found that the performance of classification models was superior to level estimation models. Moreover, those studies that predicted the direction of stock returns limited themselves to improving accuracy. They did not consider whether it would increase the stock's overall returns. Very few studies have had the benchmark as the buy-and-hold. Many of them did not consider the transaction cost. These gaps provide motivation for the present research. This study proposed a hybrid LSTM model utilizing technical indicators in an innovative and intuitive way. It predicts multiple global indices, which are top GDPs and have a good mix of emerging and developed economies. The study suggests a model to improve returns while reducing risk and compares the model with standard neural network models and the passive buy-and-hold strategy.

2.3 Auto-Regressive Integrated Moving Average (ARIMA)

In statistical methods, autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA), autoregressive conditional heteroskedasticity (ARCH), and generalized autoregressive conditional heteroskedasticity (GARCH) are used

to predict stock prices. Out of these methods, ARIMA is one of the most frequently used statistical methods for time series analysis (Hiransha et al., 2018). The close prices may have an autoregressive component in stock time series data (Paper et al., 2011a). Consequently, ARIMA may have the potential for short-term prediction and can compete with other techniques (Adebiyi et al., 2014). However, ARIMA is generally not preferred in predicting financial time series. Due to the non-linear and non-stationary nature of stock prices, it is difficult to model the stock time series with these models (Kazem et al., 2013).

Hence, machine learning methods such as ANN and SVM are gaining importance due to their capabilities to solve non-linear patterns. Pai & Lin (2005) compared the ARIMA, SVM, and hybrid ARIMA-SVM models. They evaluated these models based on Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE). They concluded that the hybrid ARIMA-SVM model is more promising than the individual ARIMA and SVM models. McNally et al. (2018) experimented with LSTM and ARIMA using the Bitcoin time series data. They determined that LSTM achieved an accuracy of 52% and an RMSE of 8%. They also compared their results with ARIMA's and discovered that ARIMA performed poorly.

For time series data, parametric models such as ARIMA are often robust. However, these models do not perform well in the case of highly dynamic and noisy financial time series. Qian and Gao (2017) examined the performance of ARIMA and machine learning models such as SVM, Logistic Regression, Multi-Layer Perceptron, and denoising autoencoder. He utilized three index prices: the DOW 30, the S&P 500, and the Nasdaq. He discovered that machine learning models are far superior to ARIMA.

Wijaya et al. (2010) analyzed the models of ANN with ARIMA on financial time series. They took a daily closing price of 2 months for the prediction. RMSE and MAE were used to evaluate the models. They found that ANN has a smaller error than ARIMA. Du (2018) compared a hybrid model of ARIMA and Back Propagation Neural Network (ARIMA-BPNN) with an individual model of ARIMA and BPNN. He chose the Shanghai Securities composition index for prediction and took the data for around one year. He found that ARIMA-BPNN accuracy was better than BPNN. Similarly, BPNN outperformed ARIMA in terms of accuracy.

Paper et al. (2011b) demonstrated the effectiveness of the ARIMA intervention model in financial time series analysis and forecasting. They asserted that the ARIMA intervention model could predict stock index prices. Khashei and Hajirahimi (2019) compared the hybrid models ARIMA-MLP and MLP-ARIMA with the individual component models ARIMA and MLP (Multiple Layer Perceptron). They predicted the stock index prices of the Dow Jones Industrial Average (DJIA), Shenzhen Integrated Index (SZII), and Nikkei 225 (NI225). The performance of models was assessed using MAE, MSE, MAPE, and RMSE. They concluded that one of the hybrid models always performed better than the individual component models, i.e., ARIMA and MLP.

Ince and Trafalis (2008) examined the performance of SVR, MPL, and ARIMA. They designed two trading techniques after selecting ten stocks from the National Association of Securities Dealers Automated Quotations (NASDAQ). These trading strategies were based on technical indicators such as Exponential Moving Averages (EMA), Moving Average Convergence and Divergence (MACD), Relative Strength Indicator (RSI),

Bollinger Bands (BB), and Chaikin Money Flow (CMF). Based on RSME, they determined that the performance of SVR was superior to MLP and ARIMA. However, different trading strategies produced different results.

Kazem et al. (2013) proposed a hybrid SVR model employing the firefly algorithm based on chaos mapping and metaheuristic optimization. The authors predicted the stock prices of three stocks on the NASDAQ: Microsoft, Intel, and National Bank. They compared the proposed model with hybrid genetic algorithm-based SVR (SVR-GA), firefly algorithm-based SVR (SVR-FA), Artificial Neural Networks (ANN), chaotic genetic algorithm-based SVR (SVR-CGA), and adaptive neuro-fuzzy inference systems (ANFIS). Based on evaluation parameters such as MSE and MAPE, the authors concluded that the proposed hybrid model is superior to other compared models.

Chen et al. (2022) tested the performance of their LSTM model on China's commercial bank stocks. They predicted the price of stocks for the mid and long-term. They compared their model with MLP, SVM, and Generalized Auto-Regressive Conditional Heteroskedasticity (GARCH). Based on evaluation using R-Square, MAE, and MSE, they concluded that their model had superior generalization ability compared to other models. Yeh et al. (2011) presented a Multiple Kernel Support Regression (MKSVR). This model did not require the user to configure the free parameter of MKSVR manually. Using multiple kernels, the model itself optimized the parameter. They compared the model with Single Kernel Support Vector Regression (SKSVR) and ARIMA. Based on RSME, they concluded that their proposed model outperformed other models. Al Galib et al. (2014) used the hidden

Markov model and nearest neighbor algorithm and concluded that their algorithm could predict price fluctuations on a given day.

Hence, most of the literature focuses on ARIMA and neural network models, and very few studies compare ARIMA with SVM or SVR. Further, most of the studies focus on predicting next-day prices (Zhang et al., 2019; Wu et al., 2021; al Galib et al., 2014). A study on the comparison of ARIMA and SVR to predict stock index prices in developed and emerging economies has not been observed. The study evaluates the performance of static and iterative models of ARIMA and SVR. Static models forecast the long-term price at once for more than 1 year. In contrast, iterative models predict the next day's price and retrain the models before predicting the next period's price. Furthermore, to evaluate the prediction power, the static and iterative models of ARIMA and SVR are evaluated using the baseline Naïve models of prediction for NIFTY, Dow Jones Industrial Average (DJIA), DAX performance index (DAX), Nikkei 225 (NI225), and Shanghai Stock Exchange (SSE) composite index.

2.4 Support Vector Regression (SVR)

Another common machine-learning algorithm for stock prediction is SVM (Kurani et al., 2023). LSTM and SVM are highly effective for predicting stock price movements (Kara et al., 2011). While LSTM utilizes memory cells and gates to manage time-based connections between data points, SVM is a kernel-based algorithm that identifies the optimal hyperplane for class separation. SVR, a variation of SVM, is used for regression tasks and aims to find a hyperplane that fits the data closely with an acceptable degree of error rather than dividing it into distinct classes like SVM.

Zou et al. (2022) proposed a Twin Support Vector Machines (TWSVM) prediction model. They employed thirteen indicators, extracted from historical data, as input features to predict the next day's stock direction. A comparison was made between the TWSVM predicting model and five other models, including decision tree, Naive-Bayes, RF, probabilistic neural network (PNN), and SVM. The results indicated that the TWSVM prediction model surpassed the other models in terms of predicting stock price and index daily movement. Doroudyan and Niaki (2021) proposed a novel method based on an SVM for detecting upward and downward shifts in auto-correlated financial processes modeled by the Autoregressive Moving Average-Generalized Autoregressive Conditional Heteroskedasticity time series model. The authors identified specific features that can capture various characteristics of the patterns, which helps detect shifts in financial processes.

The SVM and SVR are frequently used with hyperparameter optimization techniques such as grid search (GS), randomized search, or Bayesian optimization to find the best combination of hyperparameters for the given dataset and problem. These techniques help to efficiently search through the large hyperparameter space and find the best model configuration that can lead to better performance and reduce overfitting. Dash et al. (2021) introduced a novel ML approach for time series stock forecasting using a fine-tuned version of SVR (FT-SVR). The authors used the grid search technique to find the fine tuned hyperparameters of SVR from the training set and validated them on separate data. The proposed method, FT-SVR, predicted stock prices from eight large-sized datasets from various domains. The authors compared the FT-SVR with similar methods using RMSE and

MAPE, which demonstrated that the proposed approach was more accurate in predicting stock performance for the utilized datasets and required less time compared to other methods. Mahmoodi et al. (2022) utilized SVM with PSO to improve the classification. They compared the performance of SVM-PSO with two other meta-heuristic algorithms, namely the NN and the SVM-Cuckoo Search (SVM-CS) algorithm. The results indicated that SVM-PSO outperformed SVM-CS and NN algorithms.

The genetic algorithm (GA) possesses the capability to handle large-scale optimization problems and maintain diversity in the population, which helps prevent premature convergence. Furthermore, genetic algorithms are computationally efficient and can rapidly converge to the optimal solution. Therefore, Genetic algorithm is also frequently utilized to search for the best combination of hyperparameters of SVM or SVR, such as the kernel type, regularization parameter, and kernel coefficient, that minimize the error of the SVR model on a validation set. Li and Sun (2020) presented a predictive model combining kernel parameters and optimization with the SVM model. The authors employed mesh search, genetic algorithm, and PSO to optimize SVM parameters. The results indicated that optimizing SVM parameters enhanced prediction accuracy. Among the different approaches, the genetic algorithm (GA) with the radial basis kernel (RBF) function demonstrated the highest performance for stock market forecasting. The PSO was slightly less effective than the GS method. Comparison experiments demonstrated that the BP neural network was less accurate than the SVM in predicting the stock market.

Similarly, for deep learning models, a genetic algorithms can also be used to optimize parameters such as the number of hidden layers, the number of nodes per layer, the activation

function, and the learning rate, among others. Gao et al. (2022) proposed a deep learning approach that combined genetic algorithms to predict the overnight return direction of the stock market indices. The method utilized global stock market indices as an information source and employed multiple convolutional units to extract features from all regions. The model was optimized and applied to forecast the overnight return directions of nine global stock indices. The results showed that the proposed model performed better than other competing methods in terms of accuracy, F-measure, and Sharpe ratio.

Solares et al. (2022) presented a comprehensive decision support system that was proposed for the management of stock portfolios that addressed the forecasting of price, the selection of stock, and the optimization of the stock portfolios. The system utilized ANNs, differential evolution, and GA in different stages. The authors evaluated the system using historical data from the S&P 500 index and compared it to multiple benchmarks. The results demonstrated superior performance compared to the benchmarks. Thakkar and Chaudhari (2022) proposed a method based on GA and information fusion, called inter-intra crossover and adaptive mutation (ICAN), for predicting stock prices and trends. The study proposed a method to optimize the parameters and feature selection of an LSTM model using a genetic algorithm with ICAN. Their method outperformed existing GA-based optimization approaches in terms of MSE, MAE, MAPE, and R^2 score. Another proposed algorithm used hybridized genetic algorithm-machine learning regressions for feature selection, achieved a parsimonious feature subset for interpretability, and improved the average forecasting RMSE.

Lv et al. (2022) predicted the price of multiple stock indices. They proposed a new hybrid model called CEEMDAN-DAE-LSTM for stock index prediction. They conducted experiments using a dataset consisting of two emerging market indices and four developed market indices. The model showed superior performance in both prediction accuracy and stock index trends, particularly for stock indices with higher volatility, compared to other reference models. Table 2.1 shows some recent studies with prediction durations.

Table 2.1 Prediction durations of recent studies

Study	Dataset	Input Features	Prediction Methods	Evaluation Metrics	Prediction Duration
(Kanwal et al., 2022a)	Crude Oil, German stock index (DAX)	Historical data	Hybrid DNN, LSTM, CNN	RSME, MAE	Daily
(Chaudhari & Thakkar, 2023)	Korean Index and Individual Stocks	Historical data, Technical Indicator, Google trend	BPNN, LSTM, GRU, CNN	MAE, MSE, Accuracy, R^2	Daily
(Yun et al., 2021)	Korean Index	Historical data, Technical Indicators	Hybrid GA-XGBoost	Accuracy	Daily
(Dash et al., 2021)	State Bank of India	Historical data, Technical Indicators	SVR	RMSE, MAPE	Daily, Monthly
(Kaczmarek & Perez, 2022)	Portfolio	Macro-economic	Random Forest	Sharpe and Calmer ratio	Monthly
(Lu et al., 2021)	SSE Composite Index	Historical data	Hybrid CNN, BiLSTM	RMSE, MAE, R^2	Daily

Thus, most of the study focuses on predicting next-day prices or directions (Nazareth & Reddy, 2023). Some studies focus on short-term prediction but there is a lack of emphasis

on long-term prediction which is an important area for financial time series forecasting. To address this gap, the study proposes a method that utilizes SVR and GA, which can benefit investors and analysts seeking a longer-term view. To enhance the accuracy of our method, the study employs rolling forward validating GA fitness function. As few studies have focused on experimenting with their models on multiple global indices, this study experiments with the proposed method on multiple global indices. Additionally, the study compares the model with grid search-based SVR, genetic algorithm-based SVR, and LSTM.

2.5 Deep Learning (DL)

Deep learning is emerging field, which is capable of solving complex problem. Deep Neural Networks (DNNs) are feedforward neural networks, meaning they process input data in one direction, from the input layer through one or more hidden layers, to the output layer. Singh et al. (2017) aimed to demonstrate that deep learning can enhance the accuracy of stock market forecasting. The study compared the performance of (2D)2PCA + Deep Neural Network (DNN) with 2-Directional 2-Dimensional Principal Component Analysis (2D)2PCA + Radial Basis Function Neural Network (RBFNN) and found that the proposed method outperformed the RBFNN method, with an improved accuracy of 4.8% for hit rate with a window size of 20. The proposed model was also compared with Recurrent Neural Network (RNN) and showed an improved accuracy of 15.6% for hit rate. Additionally, the correlation coefficient between actual and predicted returns for DNN was found to be 17.1% higher than RBFNN and 43.4% better than RNN.

Kanwal et al. (2022b) proposed a hybrid deep learning (DL) model for the timely and efficient prediction of stock prices. The proposed model, BiCuDNNLSTM-1dCNN,

combined Bidirectional Cuda Deep Neural Network Long Short-Term Memory and a one-dimensional Convolutional Neural Network. The model was compared with other hybrid DL-based models and state-of-the-art models using five stock price datasets. The results indicated that the proposed hybrid model was accurate and reliable for supporting informed investment decisions.

RNNs are designed to handle sequential data that has temporal dependencies, such as time series data or natural language processing tasks. Liu et al. (2020) proposed two attention-based RNN models, DSTP-RNN and DSTP-RNN-II, for long-term and multivariate time series prediction. These models were found to outperform state-of-the-art methods and provided insights for further exploration of attention-based methods in time series prediction. Ranjan and Mahadani (2022) compared LSTM, bi-directional LSTM, and RNN models with univariate and multivariate features to predict stock prices. The study found that the recurrent neural network approach had the highest accuracy with both univariate and multivariate features. The performance was evaluated using root mean square error and mean square error as criteria. Naik et al. (2019) proposed an RNN with recurrent dropout model to avoid overfitting and used stock returns based on closing prices as input to the model. Data was collected from NSE India, and the proposed model outperformed a feed-forward artificial neural network in terms of error minimization.

RNNs suffer from the problem of vanishing and exploding gradients, which can make it difficult to train the model effectively. LSTMs overcome the vanishing gradient problem in RNNs. Gülmez (2023) developed a deep LSTM network with the Artificial Rabbits Optimization (ARO) model (LSTM-ARO) to predict stock prices using DJIA index

stocks. Four other models, one ANN and three LSTM, including one optimized by Genetic Algorithm (GA), were compared with LSTM-ARO using MSE, MAE, MAPE, and R^2 evaluation criteria. The results indicate that LSTM-ARO outperformed the other models. Budiharto (2021) experimented with LSTM and found it to be a reliable predictor for short-term data with an accuracy of 94.57%. It was observed that using a shorter training period of 1 year with high epochs produced better results than using 3-year training data. Rather (2021) implemented a new regression scheme on an LSTM-based deep neural network to construct a predicted portfolio. The author conducted a large set of experiments using stock data of NIFTY-50 obtained from the National Stock Exchange of India. The results indicated that the proposed model outperformed various standard predictive and portfolio optimization models.

Bi-LSTM has an additional LSTM layer that processes the input data in reverse order. Lee et al. (2022) proposed an attention-based BiLSTM (AttBiLSTM) model and applied it to trading strategy design. The model is evaluated with various technical indicators (TIs), including a stochastic oscillator, RSI, BIAS, W%R, and MACD. Two trading strategies suitable for deep neural networks (DNNs) are also proposed and verified for their effectiveness. The study introduces five well-known TIs and demonstrates the highest accuracy of 68.83% in predicting stock trends. Additionally, the concept of exporting the probability of the deep model to the trading strategy is introduced, resulting in the highest return on investment of 42.74% on the back test of Yuanta/P-shares Taiwan Top 50 ETF (TPE0050).

GRU has fewer parameters to train compared to LSTM. Hamayel and Owda (2021) proposed and compared three recurrent networks (RNN) models for predicting cryptocurrency prices: LSTM, bi-LSTM, and GRU. The models were evaluated using the Mean Absolute Percentage Error (MAPE). Results showed that the GRU model outperformed LSTM and bi-LSTM for all three cryptocurrencies (Bitcoin, Ethereum, and Litecoin), with the lowest MAPE percentages. The bi-LSTM model had the highest prediction error. Overall, the proposed models showed accurate predictions of cryptocurrency prices.

CNN has given less emphasis to forecast stock prices. Generally, CNN is used for computer vision, but recently it has been applied for stock time series forecasting as well. Khodaee et al. (2022) developed a hybrid model consisting of a CNN and LSTM to forecast Turning Points (TPs) in stock prices. The model first classified each day in the time series as a TP or Ordinary Point (OP) and used a balancing approach to have a balanced number of TPs and OPs. Technical indicators were then converted into 2D images to consider the relationship between them, and the Fuzzy C-Means algorithm was applied to segment the inputs and aid training efficiency. A classification hybrid CNN-LSTM-ResNet model was proposed to forecast TPs and OPs, and augmentation techniques, including Residual Networks (ResNet), were employed. The proposed model outperformed other benchmarks with an average accuracy of 60.19% in the Dow-30 and 63.62% in ETFs, achieving a profit of up to three times in the Dow-30 and up to four times more than the Buy and Hold strategy in ETFs.

Using deep learning for financial time series stock forecasting, the majority of research focuses on short-term prediction. Specifically, the studies have primarily focused on predicting prices for the next day (Nazareth & Reddy, 2023). This short-term emphasis on prediction has created a gap in the literature. In this study, this gap is addressed by predicting long-term prices. Additionally, most studies have experimented with a single index. However, this study used the historical daily price of the top five global economies' indices and predicted the daily price for the next year at once, providing investors and traders with a long-term market outlook to make informed investment decisions and improve risk management. Furthermore, this study exhaustively experimented with six deep learning models, namely DNN, RNN, LSTM, Bi-LSTM, GRU, and CNN, to forecast long-term stock prices. This extensive research differentiates this study from other studies.

2.6 Research Gap

The existing literature in stock market forecasting has primarily focused on specific indices or baskets of stocks, mainly in developed economies, limiting their effectiveness in predicting multiple global market indices with different dynamics. Furthermore, studies have prioritized accuracy improvement in predicting stock prices without considering overall returns. To address these gaps, this study introduces a novel hybrid LSTM model that incorporates technical indicators to predict multiple indices globally with a focus on improving returns and reducing risk. The model is compared against other LSTM models and a passive buy-and-hold strategy. There is also less focus on comparing the model's performance in emerging and developed economies. Hence, this study evaluates the performance of static and iterative models of ARIMA and SVR in emerging and developed

economies, where static models forecast long-term prices and iterative models predict next-day prices.

The focus of most of the studies is on the short term, varying from a day to a month. The study also addresses the limited emphasis on long-term prediction using machine learning techniques by proposing an approach that incorporates SVR and GA. Additionally, there is a lack of clarity on the suitability of deep learning algorithms for long-term prediction. This study compares the forecasting abilities of six deep learning models, namely DNN, RNN, LSTM, Bi-LSTM, GRU, and CNN, in predicting long-term prices, establishing the study's distinctiveness through comprehensive research methodology.

2.7 Research Questions

Inspired by the identified gaps in the existing literature, this study aims to address these shortcomings and provide valuable insights by answering the following research questions:

1) Can an LSTM and Technical Analysis-based model improve the returns of global stock indices compared to the buy-and-hold strategy?
2) What is the performance difference between statistical techniques, such as ARIMA, and machine learning models, namely Support Vector Regression, in predicting global stock indices in both emerging and developed economies for the long and short term?
3) Can an evolutionary algorithm and a Support Vector Regression-based model improve the prediction of long-term prices of global stock indices over state-of-the-art machine learning algorithms?

4) What is the level of performance of different deep learning models in predicting global stock indices over the long-term?

2.8 Research Objectives

The research objectives are designed to investigate and accomplish the following key goals:

- **RO1**: To develop a model based on LSTM and Technical Analysis to improve the returns of global stock indices
- **RO2**: To compare the performance of statistical techniques such as ARIMA and machine learning models such as Support Vector Regression in predicting global stock indices in emerging and developed economies
- **RO3**: To develop a model to predict long-term prices of global stock indices using a Genetic Algorithm and Support Vector Regression
- **RO4**: To compare the performance of the deep learning models in predicting global stock indices over the long-term.

2.9 Publications

Table 2.2 presents a mapping of the publications with their respective research objectives.

Table 2.2. Publications

RO	Publication	Status	Index
RO1	Beniwal, M., Singh, A., & Kumar, N. "Alternative to Buy-and-Hold: Predicting Indices Direction and Improving Returns Using a Novel Hybrid LSTM Model." International Journal on Artificial Intelligence Tools (2023).	Accepted	SCIE
RO2	Beniwal, M., Singh, A., & Kumar, N. " A comparative study of static and iterative models of ARIMA and SVR to predict stock indices prices in developed and emerging economies." International Journal of Applied Management Science (2023)	Accepted	Scopus and ESCI
RO3	Beniwal, M., Singh, A., & Kumar, N. (2023). Forecasting long-term stock prices of global indices: A forward-validating Genetic Algorithm optimization approach for Support Vector Regression. Applied Soft Computing, 110566.	Published	SCIE
RO4	Beniwal, M., Singh, A., & Kumar, N. "Forecasting Multistep Daily Stock Prices for Long-Term Investment Decisions: A study of Deep Learning models on Global Indices."	Communicated	

3. Trading Framework using LSTM and Technical Analysis

3.1 Overview

Predicting stock direction is a complex and challenging task and stock prices time series are extremely noisy. Moreover, the widely accepted academic theories, such as the efficient market hypothesis and random walk theory, state that it is impossible to consistently generate excess returns over a long-term horizon than the market. However, some researchers suggest that the market does have a predictable component. Neural networks have a remarkable ability to generalize and predict non-linear and complex data. In this study, daily indices data, namely the Dow Jones Industrial Average (DJIA), Nifty Index (NIFTY), DAX performance index (DAX), Nikkei 225 (NI225), and SSE Composite Index (SSE), of the top five economies of the world, is used for analysis.

This study experimented with multiple LSTM models, two standards (LSTM and LSTM-o), and three newly constructed (*fma*-LSTM-o, *atr*-LSTM-o, and *a-m*-LSTM-o), to predict the next-day direction of the indices. Based on the predicted directions, the standard models LSTM and LSTM-o returned -7% and 70%, respectively, for the testing period of around 15 years. On the other hand, the new models *fma*-LSTM-o, *atr*-LSTM-o, and *a-m*-LSTM-o yielded returns of 76%, 173%, and 172%, respectively. The study also proposes a hybrid LSTM model (*a-m*-LSTM-o), which improves the returns over the standard LSTM models, reduces the number of transactions, and beats the buy-and-hold return. The number of transactions for standard models LSTM and LSTM-o is around 90 and 98 per year,

respectively; on the other hand, the proposed model has only around 1 transaction per year. The buy-and-hold return for the portfolio of five indices is 139%, while the proposed model return is 172%. Moreover, the drawdown of the proposed model (-49%) is also better than the buy-and-hold (-51%). Hence, the proposed model also reduces the risk. Therefore, the proposed model is a good alternative to the conventional approach of an investor.

3.2 Background

There is a famous story to explain the efficient market hypothesis. Once, a professor was walking down the road with a student. The student saw a $100 bill. When he moved ahead to pick it up, the professor told him not to worry about that $100 bill because if it were real, then it wouldn't be here. Similarly, the efficient market hypothesis argues that no excess returns are available for a novice or a professional by predicting the market. In the words of Ellis (1975) actively trying to beat the market is "The Loser's Game." The only way to earn higher returns is to accept higher risk (Malkiel, 2003a). The efficient market hypothesis does not rule out the possibility of someone making higher returns. It says that someone might be lucky to make higher returns, just as someone might be unlucky to make lower returns in the market. However, the simple story cannot describe the complex dynamics of the stock market. The story assumes that a person's behavior in normal and panic conditions is rational. It implies that no one would leave a $100 bill, even in a panic situation. Similarly, the efficient market hypothesis assumes that the whole market is perfectly efficient and leaves no scope for excessive returns. However, it might not always be the case. The flash crash in the Dow Jones in 2010 led Accenture to trade at nearly zero valuation for a $0.01 price (Mcinish et al., 2014). In 2020, the world market crashed due to fear of COVID-19.

These events create doubt that the stock market is always efficient. Sometimes there is panic in the stock market, which might offer an opportunity to buy stocks at lower prices. To perform well in the stock market, one needs to develop an algorithm that, with low risk, is able to achieve high returns (Nobre & Neves, 2019). This study aims to experiment with machine learning techniques and technical analysis to create a model that conservatively shorts the indices when the market might panic and aggressively looks to buy back the indices at a lower price when the market might rise again.

Stock markets are affected by many interlinked factors, such as macroeconomic, geopolitical, exchange rate, commodity price, global stock markets, and the psychological behavior of traders. In addition, unseen events such as war breakouts, tsunamis, earthquakes, etc. may affect stock markets. These events impact a country's economy and may reduce the profitability of certain companies. The stock market is one of the riskiest investments. At the same time, it can generate significant returns. Investors, traders, and economists have immense interest in stock markets. They desire maximum returns with minimum risk. Predicting the stock market is an arduous task. Additionally, a time series of stock prices is extremely noisy. An accurate prediction of stock prices could be highly fruitful. The efficient market hypothesis and random walk theory indicate that predicting the stock market is futile. It is impossible to time the market because all the information is already reflected in the stock price. However, some studies also favored some predictability and provided evidence that the stock market may not necessarily obey the random walk hypothesis and the efficient market hypothesis (Fama & French, 1988; Gregory Mankiw et al., 1991; Jegadeesh & Titman, 1993; Lo & MacKinlay, 1998; Poterba & Summers, 1988).

Stock market analysis can be categorized into four groups: fundamental analysis, technical analysis, statistical analysis, and soft computing techniques. Fundamental analysis focuses on macroeconomic data, the Industrial outlook, a company's financial position, management, corporate governance, etc. A fundamental investor aims to arrive at the stock's intrinsic price. This intrinsic price is then compared with the security's current price to evaluate whether the security is undervalued or overvalued. A fundamental investor buys an undervalued stock with a strong potential to grow multi-fold in the long-term horizon. In contrast to fundamental analysis, technical analysis uses historical price and volume data to determine future returns.

Even though technical analysis is applied to all assets these days, it was originally developed in the context of the stock market (Neely et al., 1997). Technical analysis assumes that historical patterns repeat themselves. It is a study of past price patterns to predict future prices (Achelis & Steven, 2013). Traders who rely on technical analysis use technical indicators such as moving averages (MA), relative strength indicators (RSI), stochastic oscillators, average directional index (ADX), trendlines, etc., to predict whether the price of the stock will increase or decrease. Accordingly, they take a long or short position in the stocks. Brock et al. (1992) analyzed 90 years of historical data on American stock prices. They found that investment returns are higher when investors use technical analysis than the buy-and-hold (B&H) strategy. Although many indicators are available, it is difficult to find the daily or weekly trend of the stock market (Ticknor, 2013). Moreover, these technical analyses are self-destructive (B. Qian & Rasheed, 2007). Once a profitable strategy is known publicly, the opportunity soon vanishes.

In recent years, machine learning algorithms have been explored to predict the stock market (Gerlein et al., 2016). Although stock prices are inherently very noisy and extremely difficult to predict, many prediction algorithms claim that stock price movements can be predicted to some extent. Some of the well-known algorithms of machine learning are Artificial Neural Networks (ANNs), Long Short-Term Memory (LSTM), and Support Vector Machines (SVM). Other algorithms are K-Nearest Neighbor, Random Forest, Decision Tree, Logistic Regression, etc. ANNs are able to solve many problems due to their robustness, non-linear mapping, and memory ability.

Some studies are able to obtain higher returns than the buy-and-hold benchmark (A.-S. Chen et al., 2003; Gu & Peng, 2019). Considering the advancement in machine learning capabilities, asset management companies and investment banks are increasing their research grants as there is an opportunity for potential profit powered by capable deep learning models (Jiang, 2020). CNN is used to analyze visual and speech-type data. Traditional neural networks are feed-forward neural networks. RNN is a feedback neural network; it stores prior data in memory, making it ideal for sequential data but it suffer from vanishing and exploding gradient problems. However, long-term processing memory is irrelevant in some cases. LSTMs are a form of RNN that forgets no longer relevant memories.

This study experiments with traditional LSTM models and a hybrid LSTM model. These models combine LSTM with technical analysis. Technical analysis helps smooth prices and reduce noise. It also generates buy or sell signals. An LSTM learns these patterns of buy and sell signals and predicts the future direction of the indices. Overall, this study

experiments with five LSTM models. Out of the five models, two are standard LSTM models, and three are hybrid models using technical indicators such as Moving Averages (MA), Average True Range (ATR), or Momentum (MOM). Five indices are taken from different economies, such as the USA, Germany, Japan, India, and China. The USA, Germany, and Japan have developed economies, whereas India and China are emerging markets.

The contributions of the study are as follows:

(1) The performance of multiple LSTM models is compared using nearly 15 years of data from different economies.

(2) Online LSTM models are combined with technical analysis to improve returns and reduce transactions compared to traditional LSTM models.

(3) The proposed hybrid model is tested to demonstrate that it outperforms traditional LSTM models.

(4) The performance of the proposed hybrid model is also compared to the buy-and-hold approach. It shows better returns for out-of-sample data and provides an alternative to the traditional "buy-and-hold" investor's approach.

3.3 Proposed Methodology

3.3.1 Models

The study experimented with five models, as described in Table 3.1. The models are primarily designed to improve returns, while improving classification accuracy is a secondary goal. LSTM and LSTM-o are two traditional machine learning models. In this

study, they are trained to find a pattern in close prices and predict future movement. Three new models, namely *fma*-LSTM-o, *atr*-LSTM-o, and *a*-*m*-LSTM-o, were created with the help of technical indicators. The buy-and-hold model is the sixth model, one of our baseline models. Along with the buy-and-hold model, the LSTM and LSTM-o models are also our baseline models.

Table 3.1. Model's Descriptions

S. no.	Model Abbreviation	Model Name	Model Description
1	LSTM	Long short-term memory	Long short-term memory neural network. Close price directions are used as the input.
2	LSTM-o	Long short-term memory online	The data is added as it arrives, and the model is retrained with new data before predicting the future direction.
3	*fma*-LSTM-o	Four moving averages LSTM-online	Four moving averages of close prices MA^5, MA^{10}, MA^{20}, and MA^{50} are used to calculate direction. These are then used to train LSTM-o
4	*atr*-LSTM-o	Average True range LSTM-online	The Average True Ranges (ATR) of close prices are used to calculate directions that train LSTM-o.
5	*a*-*m*-LSTM-o	Average True range and Momentum LSTM online	The proposed novel model. Directions are derived from combined historical momentum (*h*MoM) and Average True Range (ATR) that train LSTM-o
6	B&H	Buy-and-Hold	Traditional investment approach, where an investor buys a stock and holds it for the long term. It is treated as a special case where the prediction model always gives a buy signal.

Usually, in the standard LSTM model, data is split into train and test data. The LSTM model is trained on a fixed training data set. The prediction is made at once for the rest of the test data and evaluated by comparing the prediction and test data. However, this approach is static. New data, when it arrives, does not play any role in updating the weights of LSTM. Hence, new information coming with new data is lost and does not contribute to predictions.

Consequently, in dynamic conditions such as the stock market, the LSTM model may quickly become obsolete.

In LSTM-o, an online LSTM, when new data arrives in this model, the model can retrain itself to update the weights dynamically. The model adds the new data iteratively in a Last Come, First Out (LIFO) manner and predicts the next day's direction. Once the LSTM-o model predicts the next day's direction, the original data of the next day is available to the model for retraining. This way, the model is still using historical data to predict the future, but with the advantage of updated information from new data.

3.3.2 LSTM

Long Short-Term Memory (LSTM), a type of RNN, overcomes the problem of the vanishing gradient that is common in traditional RNNs. Hochreiter and Schmidhuber (1997) introduced LSTM to address this problem of RNNs. LSTMs are specifically designed to acquire and preserve sequential relationships in time-series data over an extended period of time. An LSTM unit comprises two states and three gates, with the gates regulating the information flow into and out of the cell state while the states preserve the internal memory of the LSTM. Figure 3.1 shows the LSTM unit.

Input Gate: The input gate uses a sigmoid function to decide the weights of each value in the current input and adds the weights to the cell state. The sigmoid function produces an output range of values between 0 and 1.

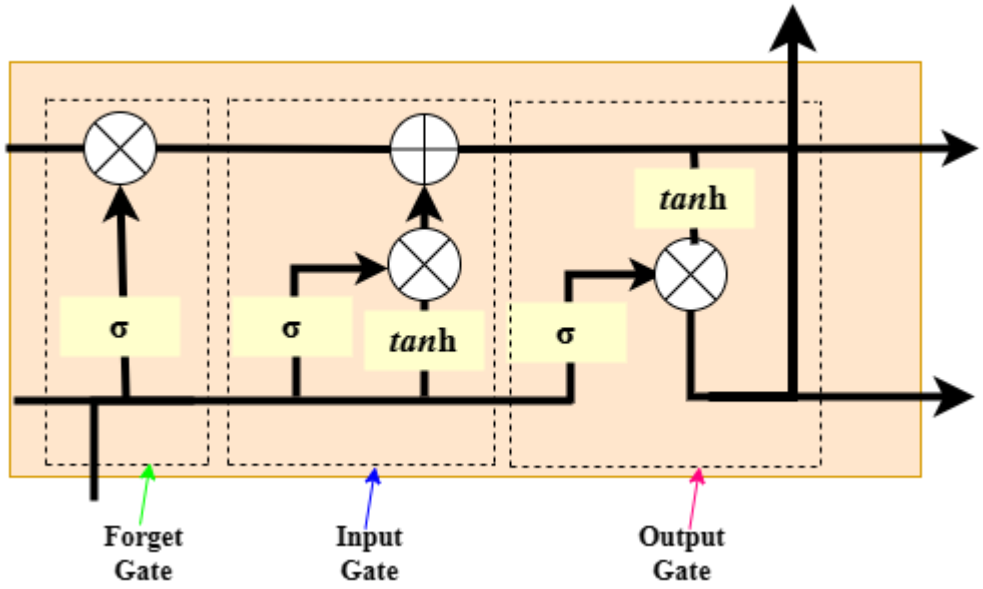

Figure 3.1 LSTM Unit

Forget Gate: The forget gate assesses which elements of the previous cell state should be retained or discarded, and the amount to forget is determined by a sigmoid function, which produces an output range of values between 0 and 1.

Output Gate: The output gate determines which weights from the cell state should be output to compute the next hidden state. The gate uses a sigmoid function and a hyperbolic tangent (tan*h*) function to produce an output value in the range between -1 and 1, indicating the importance and strength of each cell state value.

Cell State: The cell state serves as the internal memory of the LSTM, storing data that has been accumulated over time and updated by the input gate and forget gate.

Hidden State: The hidden state is the output of the LSTM unit and is utilized for forecasting purposes. It is produced by the output gate and is dependent on the present input, the prior hidden state, and the current cell state.

The gates and states of an LSTM unit function in tandem to enable the network to selectively preserve or discard data over time, resulting in an efficient tool for analyzing sequential data. The hidden state at time t, h_t, in an LSTM network is calculated using the input at time t, x_t, the previous hidden state at time t-1, h_{t-1}, and the cell state at time t-1, c_{t-1}. The cell state is updated based on x_t and h_{t-1}, and the updated cell state, c_t, is then passed through gates that decide which information to keep and which to discard. The forget gate, f_t, decides which information to discard from the previous cell state, c_{t-1}, by applying a sigmoid function to a linear transformation of x_t and h_{t-1}. The forget gate can be expressed as in Eq. (3.1):

$$f_t = \sigma\left(W_f\, x_t + U_f\, h_{t-1} + b_f\right) \tag{3.1}$$

where W_f, U_f, and b_f are the weight matrix, recurrent weight matrix, and bias vector for the forget gate, respectively. The input gate, i_t, determines which new information should be added to the cell state by applying sigmoid and tan*h* functions to a linear transformation of x_t and h_{t-1}. The sigmoid function controls which elements of the input should be updated, while the tan*h* function creates a vector of new candidate values to be added to the cell state. The input gate can be expressed as in Eq. (3.2):

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$
$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3.2)$$

where W_i, U_i, b_i, W_c, U_c, and b_c are the weight matrix, recurrent weight matrix, and bias vector for the input gate and the candidate values, respectively. The cell state, c_t, is updated by forgetting old information and adding new information, as in Eq. (3.3):

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (3.3)$$

Where $\odot$ denotes element-wise multiplication, the input gate, i_t, determines which new information should be added to the cell state by applying sigmoid and tan*h* functions to a linear transformation of x_t and h_{t-1}. The sigmoid function controls which elements of the input should be updated, while the *tanh* function creates a vector of new candidate values to be added to the cell state. The output gate can be expressed as in Eq. (3.4):

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$
$$h_t = o_t \odot \tanh(c_t) \quad (3.4)$$

where W_o, U_o, and b_o are the weight matrix, recurrent weight matrix, and bias vector for the output gate, respectively. By using these gates, LSTMs are able to selectively retain important information from the previous cell state, add new information based on the current input, and output relevant information as the hidden state. This enables LSTMs to capture long-term dependencies and overcome the vanishing gradient problem of traditional RNNs.

3.3.3 Model's framework

Figure 3.2 describes the framework of the models *fma*-LSTM-o, *atr*-LSTM-o, and *a-m*-LSTM-o. The difference between the *fma*-LSTM-o, *atr*-LSTM-o, and *a-m*-LSTM-o is price transformation using technical indicators such as moving averages, average true ranges, or a combination of average true range and historical momentum. The framework of the models has three stages, namely data preparation, prediction, and evaluation.

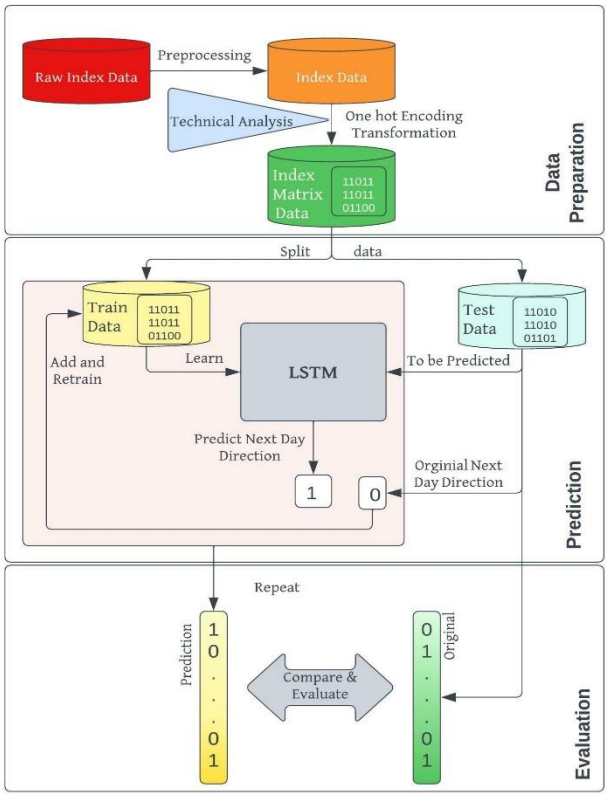

Figure 3.2 Prediction Framework

The first stage is data preparation, which includes downloading, pre-processing, and transforming the data. The time series data consists of close prices only. The pre-processing

step uses the forward-fill method to deal with missing values. Close prices are noisy and complex, so it is necessary to smooth them for modeling by appropriate transformation. The transformation using technical indicators is appropriate to reduce the noise and generate buy-and-sell signals. Table 3.2 shows pseudo code of the model's algorithm.

Table 3.2 Pseudo Code of Models' Algorithm

Algorithm
Input: One hot encoded technical indicator value
Output: Next day direction

Procedure
Forward Fill the Close price for missing data
Calculate technical indicator value (MAs, ATRs, MoMs)
One hot encoding {1,0} of the value obtained from the previous step
Reshape the input data (None, 15) and label data (none, 1)
Compile LSTM models
Model. Sequential()
LSTM(512, stateful=True, batch_input_shape(batch_size=20, window_size=15, features=1))
Dense(1, activation=sigmoid)
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
While (test_data)
Train Model
Predict Next Day
Append original Label to Train data
Calculate
Accuracy, Weighted Precision, Weighted Recall, Weighted F1-Score, Prediction return Prediction drawdown, and the number of transactions
Evaluate Models

This study further transforms the close prices to represent long or short positions. Machine learning algorithms are sensitive to data representation. The prediction performance is not only dependent on the algorithm but also on the representation of the input (Patel et al., 2015). Hence, this transformation impacts the model's training, which further influences predictions of the directions. Ultimately, price transformation affects returns from the index. Using one hot encoding and technical analysis, the final output of

the transformation is a matrix of data containing ones and zeros. The ones and zeros represent the direction of the close prices. One represents the close price at the time t_n is increased from the close price at the time t_{n-1}. Similarly, zero represents the close price at t_n is decreased from the close price at the time t_{n-1}.

The second stage is the prediction stage. For supervised learning, the matrix data is split at this stage into training and testing data. Next, the training data is fed into the five LSTM models. While training, the model tries to minimize the binary cross-entropy loss function using the sigmoid activation function. The input parameter values of LSTM models are kept the same for all models. Apart from standard LSTM, all other models are online models. Only the first 100 days' data is used for initial supervised training. After the initial training, the next day's indices' price direction is predicted. If the prediction is one, then a long position is taken; otherwise, a short position is taken. The original next day's direction is then added to training data in the Last Come-First Out (LIFO) manner. To update the model with the changing stock market conditions, the algorithm fed the data into the models in a fixed moving window in which the latest day data was added and the first-day data in the moving window was removed. Finally, the online models are then retrained with the latest data before the next prediction. This cycle continues until all the predictions are made for the testing period.

Test and prediction data are compared in the final stage. Both have one dimension of data containing stock price directions. For evaluation purposes, seven factors are captured: accuracy, weighted precision, weighted recall, weighted F1-Score, prediction return, prediction drawdown, and the number of transactions (TXNs). Weighted precision, recall,

and F-score provide a better view than accuracy in cases of class imbalance. This study focuses on increasing returns and reducing drawdown (risk) with a minimum number of transactions.

3.3.4 The Data

The study analyzes five major indices using data from the top five GDPs (StatisticsTimes.com, 2020). One index from each country, namely the US, India, Germany, Japan, and China. The data is downloaded from Yahoo Finance. Table 3.3 summarizes the data.

Table 3.3 Index Data Summary

Country Name	Stock Exchange	Index Name	Index Symbol	Positive Closings	Negative Closings
USA	New York Stock Exchange (NYSE) and National Association of Securities Dealers Automated Quotations (NASDAQ)	Dow Jones Industrial Average	DJIA	54%	46%
INDIA	National Stock Exchange (NSE)	NIFTY 50	NIFTY	53%	47%
GERMANY	Frankfurt Stock Exchange	Deutscher Aktienindex	DAX	53%	47%
JAPAN	Tokyo Stock Exchange (TSE)	Nikkei 225	NI225	52%	48%
CHINA	Shanghai Stock Exchange	SSE Composite Index	SSE	52%	48%

If the close price of a day is higher than the previous day's close price, it is a positive closing; otherwise, it is a negative closing. For all indices, Positive Closing ranges from 52% to 54%, and Negative Closing ranges from 46% to 48%. The number of trading days is kept

common for all indices. Hence, it is daily data for around 15 years, from 2007 to 2021, consisting of 3480 trading days. Figure 3.3 shows the close price movement of all indices.

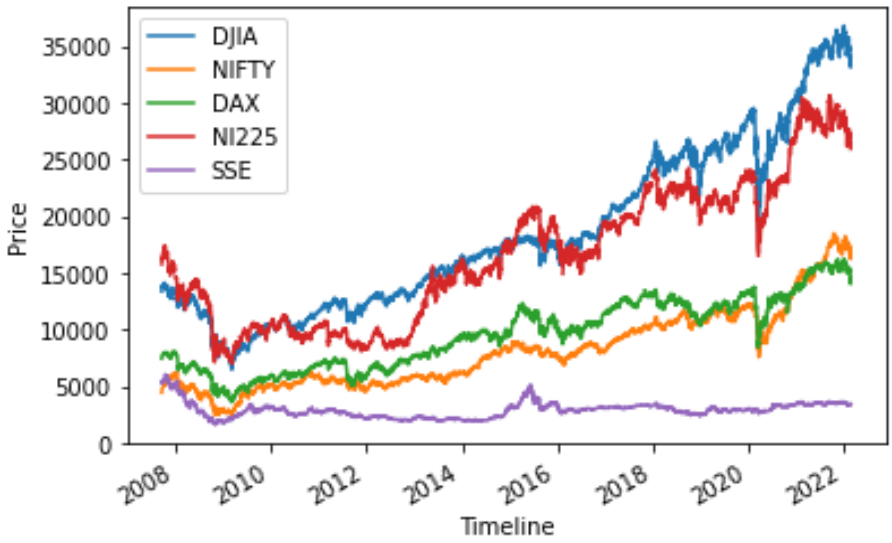

Figure 3.3 Close Price Timeline

The data from these five countries is very diverse, originating from different continents, cultures, and economies. The New York Stock Exchange is the world's largest stock exchange, and its index, the Dow Jones Industrial Average, is one of the oldest indices. The USA has a developed economy, and its stock markets are considered highly efficient. Japan and China are East Asian countries. Japan is a developed economy, whereas China is considered a developing economy.

Japan's Tokyo Stock Exchange and China's Shanghai Stock Exchange are among the top exchanges in the world. Germany is a western European developed economy. India is a South Asian country. Interestingly, China and India contribute more than 30% of the world's

population, and India is the most populated country and an emerging economy. The National Stock Exchange is India's largest electronically traded exchange.

3.3.5 Technical Indicators

The raw data is pre-processed for missing or invalid values. Specifically, the forward-filling method is used, which fills in the missing or invalid values from the previous day's data. The close price is used as input among the available features, such as Open, High, Low, and Close prices. For LSTM and LSTM-o, close prices are one-hot encoded. For the novel hybrid models, technical indicators such as Moving averages, Average True Range, and Historical Momentum or combinations thereof are computed from close prices. One-hot encoding is applied after the transformation of close prices from technical indicators. Historical momentum is a new and uniquely modified version of the momentum indicator. Table 3.4 summarizes technical indicators derived from close prices to create additional features.

Table 3.4 Technical Indicators included in the study

S. No.	Technical Indicator Name	Abbreviation
1	Simple Moving Averages	SMA
2	Average True Range	ATR
3	Historical Momentum	*h*MoM

Moving averages (MA) are the averages of the closing price over a specific time period. This study uses a simple moving average to smooth the closing prices in a rolling window fashion. In the rolling window simple moving average, the simple moving average SMA_N^n of the N_{th}day is updated with the close prices of the latest n days. For example,

SMA_N^{15} refers to 15 day's simple moving average on the N_{th} day. Eq. (3.5) shows the calculation over n days.

$$SMA_N^n = \frac{Close_{N-1} + Close_{N-2}..+..Close_{N-n}}{\mathrm{N}} \tag{3.5}$$

$$SMA_N^5 = \frac{Close_{N-1} + Close_{N-2} + Close_{N-3} + Close_{N-4} + Close_{N-5}}{5} \tag{3.6}$$

$$MoM_N = \frac{SMA_N^5 - SMA_{N-5}^5}{5} \tag{3.7}$$

$$MoM\_Min_N = |Min\{MoM_N, MoM_{N-1}, \ldots, MoM_1\}| \tag{3.8}$$

$$Thank\ you\ for\ reaching\ out.\ hMOM_N = \frac{price\_change_N * 100}{MoM\_Min_N} \tag{3.9}$$

$Close_N$ stands for the close price of the N_{th} day. Naughton et al. (2008) found substantial profits using momentum strategies in Chinese stocks. They claimed that momentum is a pervasive feature of stock returns. The study also uses a modified version of momentum, historical momentum (*h*MoM). Generally, in the stock market, momentum is calculated as rolling momentum over a period of time. However, in this study, rolling momentum is not calculated. Instead, historical momentum is used. Historical momentum has the memory of the momentum of all past days. For example, MoM_Min_{100}, in Eq. (3.8),

has the absolute minimum momentum over 100 days. This intuitive and innovative transformation of momentum helps the models extract important historical information.

The average true range (ATR) is a volatility-based technical indicator. To determine the average true range, first the true range of the indices is determined. True range is the maximum of the absolute difference between the High price and Low price, the absolute difference between the High price and the previous Close price, and the absolute difference between the Low price and the previous Close. The average true range is the moving average of the true ranges. Eq. (3.10) and Eq. (3.11) calculate True Range and Average True range, respectively.

$$TR_N = MAX\begin{Bmatrix} |High_N - Low_N|, |High_N - Close_{N-1}|, \\ |Low_N - Close_{N-1}| \end{Bmatrix} \tag{3.10}$$

$$ATR_N^n = \frac{TR_{N-1} + TR_{N-2} + \cdots + TR_{N-n}}{n} \tag{3.11}$$

Trailing ATR ($Trailing\_ATR_N^n$) is utilized to generate signals of buy and sell indices. A Trailing ATR follows the close price if it rises or falls in a trend. First, the average true range is multiplied by a factor n , and then it is subtracted from the current close price. In this study, the maximum trailing price from the last 15 days is the trailing ATR. Eq. (3.12) calculates the Trailing ATR.

$$\begin{aligned} Trailing\_ATR_N^n = Max\{&(Close_N - n * \ ATR_N^{15}), (Close_{N-1} - \\ &n \ ATR_{N-1}^{15}), \ldots, (Close_{N-15} - n * \ ATR_{N-14}^{15})\} \end{aligned} \tag{3.12}$$

3.3.6 Data Transformation

Technical indicators help smooth prices. Without smoothing, prices values and direction tend to be noisy. Technical indicators transform prices into bigger patterns. There are many technical indicators, and it is not feasible to study them all. Hence, selective indicators such as Moving averages, Average True ranges, and momentum are taken into account for the study. There is no fixed rule on how technical indicators should be used. Generally, technical indicators come with some default parameters. Traders or investors can change these to adjust according to market conditions. The purpose of most investors or traders is to maximize returns with minimum risk. This study uses these technical indicators intuitively and innovatively to fulfill these goals. Typically, traders also focus on accuracy. This study does not primarily aim to improve accuracy. Hence, in this study, the use of technical indicators differs from how they are generally used.

Both standard models, LSTM and LSTM-o, use close price directions. Close prices are one-hot encoded based on Eq. (3.13). Long positions are represented by 1, whereas short positions are represented by 0. The ones and zeros matrix is used as input to the LSTM and LSTM-o.

$$Position = \begin{cases} 0, & (Close_{nth} - Close_{(nth-1)}) \leq 0 \\ 1, & Elsewhere \end{cases} \quad (3.13)$$

One of the critical components of a machine-learning experiment is data representation. This study proposes an approach to transform the data to improve the returns over the buy-and-hold strategy. In this study, there are three essential aspects of the

transformed data. First, the model should learn to decide where to buy and sell the indices based on the historical pattern. Second, the model should also learn to take a short position very conservatively with a higher probability of accuracy and maintain a long position aggressively. A model must accurately predict the short positions to beat the buy-and-hold returns. This study hypothesizes that the stock market is weakly efficient during unprecedented events such as Tsunamis, COVID-19, the Russia-Ukraine war etc. These events may create panic among some traders and investors, creating selling pressure in the stock market. The market falls sharply during such events. Hence, it might be possible to predict the downfall of the stock market from historical patterns. Third, along with improving returns, investors also aspire to reduce risk. Hence, in this study, the representation of the transformed data is such that the models learn to reduce the drawdown (risk). However, this happens concurrently when short positions are accurately predicted.

For *fma*-LSTM, *atr*-LSTM-o, and *a-m*-LSTM-o, technical indicators transform the close price direction so that the short position is taken very conservatively and the long position is maintained aggressively. Moving averages have been the basic indicators used by technical analysts for a long time. Normally, one or two moving averages are used by traders. However, this study takes a different approach. *Fma*-LSTM uses four moving averages, namely MA^5, MA^{10}, MA^{20}, and MA^{50}. Long and short positions were taken according to Eq. (3.14).

$$Position = \{ \quad 0, if\,(MA^5 < MA^{10} < MA^{20} < MA^{50}) = True \quad (3.14)$$
$$1, Elsewhere\}$$

As shown by Eq. (3.14), *fma*-LSTM is highly conservative in taking a short position. All moving averages MA^5, MA^{10}, M^{20}, and MA^{50} must be sequentially less than the bigger moving average; only then can a short position be taken. On the other hand, when a fast-moving average such as MA^5 crosses above MA^{10}, the long position is taken again. The intuition of being conservatively short and aggressively long is to sell in a panic and buy at a low price to increase returns. Similarly, in *atr*-LSTM-o, positions are taken according to Eq. (3.15)

$$\begin{aligned} Position = \ \{ 0, if\{(Close_N \ < \ Trailing\_ATR_N^1) \text{ and} \\ (Close_N < Trailing\_ATR_N^4)\} \\ 1, \text{Elsewhere}\} \end{aligned} \quad (3.15)$$

When the close price decreases rapidly and goes below both $Trailing\_ATR_N^1$ and $Trailing\_ATR_N^4$, then a short position is taken. If any condition becomes false, the position is reversed to long. In the last model, *a-m*-LSTM-o, average true range, and historical momentum were used. Eq. (3.16) describes transformation.

$$Position = \{\ 0, if(hMOM_N\ < (-50))$$
$$1,\ \text{Elsewhere}\} \qquad (3.16)$$

Historical momentum $hMOM_N$ is negative when the close price decreases compared to the previous timeframe. Similarly, it is positive when close prices increase. Negative momentum varies from -100 to 0.

3.3.7 Trading Strategy

All models are trained using common parameters of LSTM on five index datasets: DJIA, NIFTY, DAX, NI225, and SSE. A single row in the matrix has 16 days of close price direction. The column of the matrix is continuous data of directions. The first 15 days' close price direction is used as training input to the LSTM, and the 16^{th} day is the target. The data of each index is split into a matrix of train and test sets. The train data set has 100 rows and 16 columns, and the test data set has 3350 rows and 16 columns. The LSTM model is trained on fixed 100 rows for 100 epochs. This model makes predictions at once for all the test data of 3350 days. The predictions are an array containing 1 and 0, representing long and short positions. Table 3.5 shows the parameters' values.

In the second model, LSTM-o is also trained for 100 rows using ten epochs initially. Further, once the prediction is made for the next day, the next day's original direction is appended to the moving window training set in Last-In-First-Out fashion (LIFO).

Table 3.5 LSTM models parameter values

Parameter	Values
Activation	Sigmoid
Optimizer	Adam
Loss	binary_crossentropy
Metrics	Accuracy
Stateful	True
Layers	2
Shuffle	False
Batch Size	20
Input Shape	(15, 1)
LSTM layer Shape	(None,512)
Dense layer Shape	(None, 1)

The moving window has 20 rows and 16 columns of data. A total of 35 days of data are present in a moving window. LSTM-o is again trained on the moving window containing 20 rows of 15 columns using one epoch only. Importantly, this retrains the model quickly and updates the weights by identifying the dynamically changing conditions. This process continues until all predictions are made for 3350 days. Similarly, *fma*-LSTM-o, *atr*-LSTM-o, and *a-m*-LSTM-o are trained like LSTM-o. The only difference from LSTM-o is price transformation using technical indicators using Eq. from (3.14) to (3.16). The models' parameters are kept the same to ensure a fair comparison.

3.3.8 Evaluation Criteria

Seven performance indicators are captured for comparison and evaluation purposes: accuracy, weighted precision, weighted recall, weighted F1-score, prediction return, prediction drawdown, and the number of transactions (TXNs). Weighted precision, weighted recall, and weighted F1-score are components of a matrix known as the confusion matrix. Although improving accuracy is not the primary purpose of the study; still, it provides insight

into the comparison of the models. As the close prices are one-hot-encoded into ones and zeros, the model's framework predicts the next day's direction as a classification problem. Accuracy is calculated using Eq. (3.17). However, it gives a biased evaluation in case where one class in the classification is in the majority. For example, if 80 cases are positive and 20 cases are negative, then if the model predicts all cases are positive, it still gives 80% accuracy. This accuracy of 80% could lead to a biased evaluation.

$$Accuracy \; = \; correct\, prediction * 100 \; / \; total\, predictions \tag{3.17}$$

Hence, accuracy can be misleading if there is an imbalance between classification classes. The confusion matrix's components, such as precision, recall, and F1-score, take class imbalance into account and provide a better evaluation than accuracy. Precision, recall, and F1-score are calculated from Eq. from (3.18) to (3.20).

$$P_{c1} = \frac{TP_{c1}}{TP_{c1} + FP_{c1}} \tag{3.18}$$

$$R_{c1} = \frac{TP_{c1}}{TP_{c1} + FN_{c1}} \tag{3.19}$$

$$F1_{c1} = \frac{2}{\frac{1}{Precision_{c1}} + \frac{1}{Recall_{c1}}} \tag{3.20}$$

For positive class: P_{c1} denotes precision, TP_{c1} denotes True Positive, FP_{c1} denotes False Positives, R_{c1} denotes recall, FP_{c1} denotes False Positives, and $F1_{c1}$ is F1 score

computed as harmonic means of precision and recall. Weighted precision (W_p), weighted recall (W_r), and weighted F1-score (W_{F1}) assign weights according to the ratio of classes present in the dataset. They are calculated using Eq. from (3.21) to (3.23).

$$W_p = \frac{W_{c1} * P_{c1} + W_{c2} * P_{c2}}{W_{c1} + W_{c2}} \tag{3.21}$$

$$W_r = \frac{W_{c1} * R_{c1} + W_{c2} * R_{c2}}{W_{c1} + W_{c2}} \tag{3.22}$$

$$W_{F1} = \frac{W_{c1} * F1_{c1} + W_{c2} * F1_{c2}}{W_{c1} + W_{c2}} \tag{3.23}$$

$W_{c1}\ and\ W_{c2}$ denotes weights of the first and second classes, respectively. To calculate the prediction return, the study compares the original direction of the index with the prediction direction in the test data set. For example, suppose the next day's original direction of index price is "1", and the prediction direction for the next day is also "1". In that case, the next day's return is added to the cumulative predicted return. Similarly, if the prediction direction is opposite to the original direction of the index price, then the actual return is subtracted from the cumulative predicted return. Eq. (3.24) and Eq. (3.25) summarize cumulative return calculations.

$$Return_N = \frac{Close_N - Close_{N-1}}{Close_{N-1}} \tag{3.24}$$

$$if\{(Original\_Direction_N = Predicted\_Return_N)$$

$$Then\ Predicted\_Return_N = Predicted\_Return_{N-1} + Return_N \quad (3.25)$$

$$Else\ Predicted\_Return_N = Predicted\_Return_{N-1} - Return_N\ \}$$

Transactions (TXNs) hold a lot of weight for investors or traders to choose between alternative strategies. Please note that transaction costs are not calculated in the study. Transaction costs vary with time, technology, and government policies, making it difficult to accommodate returns accurately.

3.4 Findings

3.4.1 Dow Jones Industrial Average (DJIA)

The Dow Jones Industrial Average (DJIA) is one of the oldest and most studied indexes, with a market cap of around $11 trillion. Moreover, some of the world's largest companies are listed on DJIA. This makes the DJIA index an essential index to track for traders and investors. In buy-and-hold (B&H), accuracy is treated as a special case of prediction, where investors only predict the market going up. Investors hold the investment forever after buying it at once. Hence, the number of transactions in the buy-and-hold is always 1. Historically, for DJIA, the market went up 54% of the time in the trading duration. As the buy-and-hold strategy never predicts the direction downward, the weighted precision is drastically low at 30%. Further, the F1 score of the buy-and-hold is 38%. All other models show a better F1 score. The LSTM-o model has the best F1 score. However, the return is

merely 18%, whereas the returns of B&H are 178%. The results of all models are summarized in Table 3.6.

Table 3.6 Index Name: DJIA, Country: USA

Model Name	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-Score	Prediction Return	Prediction Drawdown	TXNs
LSTM	52%	50%	50%	49%	-30%	-95%	1525
LSTM-o	50%	50%	50%	50%	18%	-50%	1602
fma-LSTM-o	54%	51%	54%	45%	152%	-49%	170
atr-LSTM-o	54%	51%	54%	44%	204%	-63%	129
a-m-LSTM-o	54%	54%	54%	41%	198%	-44%	33
B&H	54%	30%	54%	38%	178%	-50%	1

The LSTM model predictions are close to random predictions in multiple runs of the algorithm. It has the lowest return of "-30%", and the return varies a lot in every run. The return from the model *fma*-LSTM-o is less than the buy-and-hold. The *atr*-LSTM-o gives the best return of 204%. In terms of return, the *a-m*-LSTM-o is the second-best with 198%. However, the weighted precision score of *a-m*-LSTM-o is better than atr-LSTM-o. Both atr-LSTM-o and a-m-LSTM-o beat the buy-and-hold. The return from the *a-m*-LSTM-o is very close to the buy-and-hold. Moreover, *a-m*-LSTM-o has a lower drawdown of "-44%" compared to "-50%" in the buy-and-hold. LSTM and LSTM-o are not viable strategies to invest in or trade because of the high number of transactions. Moreover, the returns from LSTM and LSTM-o are also significantly lower than buy-and-hold.

3.4.2 Nifty

Nifty's weighted precision for buy-and-hold is 28%. Interestingly, LSTM has nearly double the drawdown of buy-and-hold. A drawdown of "-100%" or more is possible in these models as they take short positions too. For example, if the model shorts the stock at a price of 100 and the price goes to 250, it is a "-150%" drawdown. Similar to the performance of LSTM in DJIA, the number of transactions is extremely high, making the model hard to implement. The return is merely 15%. After transaction costs, it will reduce a large percentage of return. Apart from LSTM, all models give better returns than the buy-and-hold. *atr*-LSTM-o has 98% more return than the buy-and-hold. The number of transactions is moderate, less than 10 per year for 15 years duration. Similarly, *a-m*-LSTM-o return is 66% more than the buy-and-hold. However, it has only three transactions in 15 years. Moreover, the weighted precision of the *a-m*-LSTM-o is 53%, whereas the weighted precision of *atr*-LSTM-o is 51%. The results of all six models for NIFTY are summarized in Table 3.7.

Table 3.7 Index Name: NIFTY 50, Country: India

Model Name	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-Score	Prediction Return	Prediction Drawdown	TXNs
LSTM	50%	51%	50%	49%	15%	-100%	1246
LSTM-o	52%	52%	52%	52%	247%	-29%	1480
fma-LSTM-o	52%	49%	52%	43%	252%	-58%	177
atr-LSTM-o	53%	51%	53%	43%	344%	-40%	139
a-m-LSTM-o	53%	53%	53%	42%	312%	-52%	3
B&H	53%	28%	53%	37%	246%	-52%	1

3.4.3 DAX Performance-Index (DAX)

In DAX, the weighted F1-scores of the models LSTM and LSTM-o are the highest, whereas weighted recall is the lowest. Further, like in DJIA and NIFTY, these models have a high number of transactions and lower returns than buy-and-hold. The results of all six models for DAX are summarized in Table 3.8.

Table 3.8 Index Name: DAX, Country: Germany

Model Name	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-Score	Prediction Return	Prediction Drawdown	TXNs
LSTM	50%	51%	51%	50%	76%	-47%	1132
LSTM-o	50%	50%	51%	50%	-13%	-103%	1351
fma-LSTM-o	53%	50%	53%	44%	66%	-69%	213
atr-LSTM-o	53%	50%	53%	42%	135%	-57%	138
a-m-LSTM-o	54%	58%	54%	38%	220%	-38%	22
B&H	53%	28%	53%	37%	140%	-49%	1

A weighted precision of 58% is the highest for *a-m*-LSTM-o. Except for *a-m*-LSTM-o, all other models have lower returns than buy-and-hold. Further, the *a-m*-LSTM-o model has 80% more return and 11% less drawdown in the DAX. Hence, it yields a higher return with lower risk. Moreover, the number of transactions is also low, nearly 1.5 per year. The *atr*-LSTM-o is the second best.

The return is closer to B&H, but the number of transactions is moderately high. The risk is also an essential factor to consider before investing in a model. Drawdowns and transactions represent the risk to the investors and traders. Only *a-m*-LSTM-o and LSTM have a lower drawdown than the buy-and-hold. However, *a-m*-LSTM-o has the least number of transactions among all the Neural Network models.

3.4.4 Nikkei 225 (NI225)

In NI225, the weighted F1 score is the highest for LSTM-o, but the return is merely 28% compared to the buy-and-hold return of 124%. This fact is interesting as it indicates that a high F1 score does not guarantee high returns. The results of all six models for NI225 are summarized in Table 3.9.

Table 3.9 Index Name: NIKKEI 225, Country: JAPAN

Model Name	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-Score	Prediction Return	Prediction Drawdown	TXNs
LSTM	49%	52%	47%	39%	-126%	-129%	1194
LSTM-o	51%	51%	51%	51%	28%	-57%	1519
fma-LSTM-o	51%	49%	51%	44%	20%	-82%	209
atr-LSTM-o	53%	52%	52%	43%	167%	-32%	165
a-m-LSTM-o	53%	47%	53%	37%	141%	-48%	23
B&H	53%	28%	53%	36%	124%	-51%	1

Further, the number of transactions is also a lot higher than that of other models. The LSTM performed the worst in NI225. Its predicted return is negative at "-126%". In contrast, the *atr*-LSTM-o return is the highest, and the *a-m*-LSTM-o return is the second highest. Further, *atr*-LSTM-o has the lowest predicted drawdown. However, the number of transactions in *atr*-LSTM-o is nearly seven times that of *a-m*-LSTM-o. *a-m*-LSTM-o has nearly 1.5 transactions per year, whereas *atr*-LSTM-o has nearly 11 transactions per year. Moreover, the return of *a-m*-LSTM-o is also higher than the buy-and-hold.

3.4.5 SSE Composite Index (SSE)

China's SSE Composite index return for the buy-and-hold approach over the duration is just 8%. The data shows that the index price rises quickly and falls sharply. The results of all six models for SSE are summarized in Table 3.10.

Table 3.10 Index Name: SSE Country: China

Model Name	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-Score	Prediction Return	Prediction Drawdown	TXNs
LSTM	50%	50%	50%	50%	32%	-32%	1683
LSTM-o	52%	52%	52%	52%	72%	-33%	1427
fma-LSTM-o	52%	51%	52%	46%	-110%	-124%	271
atr-LSTM-o	51%	49%	52%	42%	14%	-53%	172
a-m-LSTM-o	53%	46%	53%	36%	-11%	-64%	6
B&H	53%	28%	53%	36%	8%	-55%	1

The prediction drawdowns of LSTM and LSTM are lower than those of other models and buy-and-hold. The *fma*-LSTM-o prediction return is negative at 110%. The *atr*-LSTM-o has returned closer to buy-and-hold. Moreover, it also has a similar drawdown. However, transactions again are nearly 11 per year, whereas the number of transactions in *a-m*-LSTM-o is only 0.4 per year. Interestingly, *a-m*-LSTM-o gives a small negative return but a bigger drawdown than buy-and-hold strategy. The model fails to beat the buy-and-hold strategy only in this index. This result indicates that *a-m*-LSTM-o cannot always guarantee greater returns than the buy-and-hold strategy in an index. A portfolio of indices is a better approach to diversifying risk and making the model robust.

3.4.6 Consolidated

To stabilize the returns from the models, the study made a portfolio to invest in all indices with equal weightage. Hence, the respective average of all evaluation parameters is computed. Table 3.11 has consolidated numbers. For example, the accuracy in the table refers to the average accuracy of all five indices. Similarly, weighted precision, weighted recall, and weighted F1-score are determined. The accuracy of *atr*-LSTM-o and *a-m*-LSTM-o is equal to buy-and-hold. However, the weighted precision of these models is better than the buy-and-hold strategy. Similarly, the weighted F1 score of LSTM, LSTM-o, and *fma*-LSTM-o is better than the buy-and-hold. Interestingly, the accuracies of these three models are less than the hypothetical accuracy of the buy-and-hold. Table 3.11 shows the consolidated result.

Table 3.11 Consolidated Returns from all five indices

Model Name	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-Score	Prediction Return	Prediction Drawdown	TXNs
LSTM	50%	51%	49%	47%	-7%	-80%	1356
LSTM-o	51%	51%	51%	51%	70%	-54%	1476
fma-LSTM-o	52%	50%	52%	44%	76%	-76%	208
atr-LSTM-o	53%	50%	53%	43%	173%	-49%	148
a-m-LSTM-o	53%	51%	53%	39%	172%	-49%	17
B&H	53%	28%	53%	37%	139%	-51%	1

The average prediction returns from all indices for the models LSTM, LSTM-o, and *fma*-LSTM-o are less than the average return of the buy-and-hold. Moreover, the drawdowns of these three models are also greater than the buy-and-hold. Further, the numbers of transactions are extremely high for LSTM and LSTM-o. *Fma*-LSTM-o and *a-m*-LSTM-o

have a moderate number of transactions. Only *atr*-LSTM-o and *a-m*-LSTM-o returns are higher than buy-and-hold. Additionally, the drawdowns of these two models are less than the buy-and-hold. The number of transactions in *a-m*-LSTM-o is on the lower side.

3.5 Significant Outcomes

Significant outcomes from the analysis are as follows:

- The accuracy of the *atr*-LSTM-o and *a-m*-LSTM-o models is equal to the buy-and-hold strategy.
- The average prediction returns of the LSTM, LSTM-o, and *fma*-LSTM-o models are lower than the average return of the buy-and-hold strategy.
- The drawdowns of the LSTM, LSTM-o, and *fma*-LSTM-o models are higher compared to the buy-and-hold strategy.
- LSTM and LSTM-o models have a high number of transactions, while *fma*-LSTM-o and *a-m*-LSTM-o models have a moderate number of transactions.
- *Atr*-LSTM-o and *a-m*-LSTM-o models have higher returns than the buy-and-hold strategy.
- The drawdowns of the *atr*-LSTM-o and *a-m*-LSTM-o models are lower than those of the buy-and-hold strategy.
- The number of transactions in the a-m-LSTM-o model is relatively lower compared to other models.

In summary, the study suggests that the proposed model *a-m*-LSTM-o is a good alternative for active investors compared to passive buy-and-hold strategy.

This chapter is based on the study:

Beniwal, M., Singh, A., & Kumar, N. (2023). *Alternative to Buy-and-Hold: Predicting Indices Direction and Improving Returns Using a Novel Hybrid LSTM Model.* **International Journal on Artificial Intelligence Tools**.

(Index: SCIE)

4 Performance Comparison of ARIMA and SVR

4.1 Overview

Stock market forecasting is a complicated and strenuous task. Moreover, the stock market time series is non-linear, volatile, dynamic, and chaotic. Auto-Regressive Integrated Moving Average (ARIMA) and Support Vector Regression (SVR) are popular methods in time series forecasting. This study empirically compares static and iterative models of ARIMA's and SVR's ability to predict stock market indices in developed and emerging economies. Five global stock indices, two from emerging and three from developing economies, are predicted. In the long term, in contrast to the Efficient Market Hypothesis and Random Walk Hypothesis, the results show that the SVR has some predictable power. Further, the SVR has better predictability in emerging economies than in developed ones in long-term forecasting. However, the market shows efficient behavior in daily prediction, and ARIMA and SVR fail to forecast better than the Naïve model.

4.2 Background

The future is mostly uncertain, and so is the prediction. The field of prediction is a challenging area. Especially financial time series prediction is an immensely complex and arduous task. The stock market is considered extremely risky, especially when predicting the short term. In this study, the "short-term" means daily prediction.

The Box-Jenkins model or methodology, often known as Auto-Regressive Integrated Moving Average (ARIMA), is a classical statistical model helpful in analyzing and forecasting time series data. It was developed by George Box and Gwilym Jenkins in 1970

(Box et al., 1970). ARIMA is one of the most widely utilized methodologies in time series modeling (Hiransha et al., 2018). Recently, there has been an increase in academic interest in forecasting the stock market using artificial intelligence and machine learning. Jeff Hawkins once said, "The key to artificial intelligence has always been the representation" (Hardy, 2012). There are several ways that data may be represented, and machine learning models can be trained.

In machines, artificial intelligence mimics human or animal intelligence. There are three types of machine learning algorithms: supervised, unsupervised, and reinforcement learning. This study focuses on supervised learning for stock market prediction. Some prominent supervised machine learning algorithms are Linear regression, the SVM algorithm, the Naive Bayes algorithm, Artificial Neural Networks (ANN), Random Forests, etc. Vapnik (1995) developed the Support Vector Machine (SVM), a supervised machine learning algorithm used for classification. SVM distinguishes between classes by locating a hyperplane in the N dimensions. In contrast, Support Vector Regression (SVR) is a regression algorithm that provides continuous real values output. SVR has high generalization and prediction accuracy (Awad & Khanna, 2015b).

The objectives of the study are to compare the static and iterative models of ARIMA and SVR to predict stock indices' prices in developed and emerging economies. The study experiments with multiple static and iterative ARIMA and SVR models, and their performance on the NIFTY, Dow Jones Industrial Average (DJIA), DAX performance index (DAX), Nikkei 225 (NI225), and Shanghai Stock Exchange (SSE) composite index is assessed. From 2017 through 2022, daily data for these two indexes is obtained from Yahoo

Finance. This time period is marked by substantial stock market cycles. The period from 2017 through 2020 was characterized by a bull market. The coronavirus pandemic came in March 2020, causing the global market to tumble in unprecedented ways. After a few months of the pandemic, the world market started recovering and started a very strong bull market again. Hence, this period from 2017 to 2022 covers all cycles of the market.

The study examines six models, namely Naïve static, ARIMA static, SVR static, Naïve iterative, ARIMA iterative, and SVR iterative. The stock indices data from five countries, namely the US, Germany, Japan, India, and China, are analyzed. India and China are emerging economies, while the US, Japan, and Germany are considered developed countries. The study evaluates the model's predictive powers in emerging and developed economies. Static and iterative Naïve models are baseline models used for comparison. Naïve models assume the future prices to be the last known price of the training period. This model supports the random walk theory that prices cannot be predicted in an efficient market. Static models forecast the prices for the entire test duration at once. These models do not retrain themselves even if new data is available. To address this issue, iterative models only predict the price for a day ahead. After the next day's prediction, the original data is appended to the training set for retraining. This process updates the models and makes them dynamic. The iterative models then predict the prices for the upcoming day. The same cycle is repeated until the last day price is forecasted. All six models are evaluative using mean absolute error, mean squared error, mean absolute percentage error, and root mean square error.

4.3 Proposed Methodology

4.3.1 The Data

The data for this study is obtained from Yahoo Finance, which contains daily prices between 2017 and 2022. The indices of three developed economies, the Dow Jones Industrial Average (DJIA), the DAX performance index (DAX), and the Nikkei 225 (NI225), and two emerging economies, the Nifty 50 and the SSE composite index (SSE), are used for analysis. The economies of the US, Germany, Japan, India, and China represent different continents and hence a good representation of the world economy. The diversity of time series data will test the robustness of models. These countries are also in the top five GDP ("World GDP Ranking," 2020). DJIA is one of the most popular and followed indices. The USA is a developed country, and its stock market is considered efficient. Similarly, the Nikkei 225 and DAX indices belong to developed countries, and their stock markets are also considered efficient. On the other hand, India and China are emerging economies considered developing, although China's GDP is the second-largest. The stock market is a barometer of a country's economy. The stock market dynamics of developed and developing countries can be quite different. As a result, a comparison study is required to understand the difference in performance, if any, between ARIMA and SVR in two distinct types of economies.

4.3.2 ARIMA

The ARIMA model is divided into three stages: identification, estimation, and diagnostic checking. ARIMA (p, d, q) is composed of three parts Auto-Regressive AR(p), Integrated I(d), and Moving Average MA(q). The parameters (p, d, and q) of ARIMA

indicates the specific model best fits the time series. Following is a description of the components:

Auto-Regressive AR(p): The dependent variables are lagged observations in this model. The significant cutoff number of lagged observations is p.

Integrated I(d): The observations are subtracted from their predecessors to make the time series stationary. The number of times the observations are subtracted is a whole number, denoted by d.

Moving Average MA(q): This model regresses over past forecasted errors to predict future values. "q" represents the order of the moving average, which is the number of past errors significant for regression and forecast.

The ARIMA model is described in Eq. (4.1) as follows:

$$\widehat{Y_t} = c + \theta_1 Y_{t-1} + \theta_2 Y_{t-2} + .. + \theta_p Y_{t-p} - \phi_1 e_{t-1} - \phi_2 e_{t-2} - .. - \phi_q e_{t-q} \quad (4.1)$$

where $\widehat{Y_t}$ denotes the predicted target, c is a constant, e is an error term, and Y_t is past observation at time t after differencing d times.

4.3.3 Support Vector Regression (SVR)

The present form of Support Vector Machine (SVM) is largely attributed to the contributions of Vapnik et al. (Boser et al., 1992a, 1992b; Cortes et al., 1995; Guyon et al., 1993; Schslkopf et al., 1996; Vapnik et al., 1997). SVM is a type of supervised ML algorithm

used for the classification of data. If in a dataset, data points are given (x_1, y_1), (x_2, y_2), ..., (x_n, y_n), where x_i is the input vector of features and y_i is the corresponding output, the SVM algorithm detects the weight vector w and the bias term b that define the hyperplane in Eq. (4.2):

$$y = w * x + b \tag{4.2}$$

The weight vector *w* determines the orientation of the hyperplane, while the bias term *b* sets its position. The hyperplane segregates the feature space into two regions, one for each class. SVM seeks to find the optimal hyperplane that best separates the two classes. The optimal hyperplane is the one that maximizes the margin, which is the distance between the hyperplane and the closest data points of each class. The margin is defined as in Eq. (4.3):

$$\gamma = \frac{2}{||w||} \tag{4.3}$$

where γ denotes the margin and $||w||$ represents the Euclidean norm of the weight vector. To ensure that outliers do not influence the optimal hyperplane, SVM introduces the concept of "slack variables." The slack variables ξ_i and ξ_i^* enable certain data points to exist on the incorrect side of the hyperplane while still penalizing them for being misclassified. The constraints on the slack variables are defined in Eq. (4.4):

$$\begin{gathered} y_i - w * x_i - b \leq \varepsilon + \xi_i \\ w * x_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{gathered} \tag{4.4}$$

where ε is a small positive constant that defines the width of the margin, and y_i is the output or class label of the data points. The optimization problem for SVM can be formulated as in Eq. (4.5):

$$minimize \frac{1}{2}\left|\left|w\right|\right|^2 + C * \sum(\xi_i + \xi_i^*) \tag{4.5}$$

$$subject\ to: y_i - w * x_i - b \leq \varepsilon + \xi_i$$

$$w * x_i + b - y_i \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

where C is a positive constant that controls the trade-off between maximizing the margin and minimizing the slack variables. The optimization problem can be solved using various methods, such as the quadratic programming method or the gradient descent method. Figure 4.1 shows the SVM.

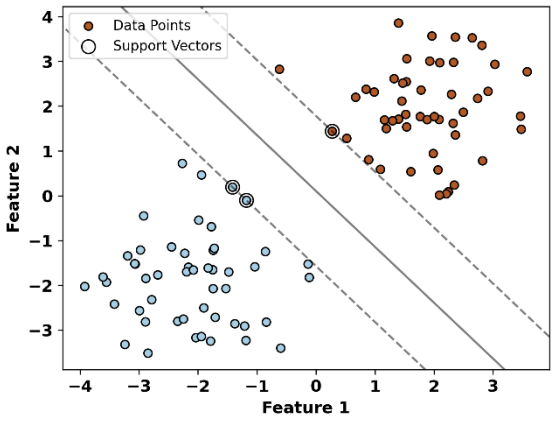

Figure 4.1 Support Vector machine

Support Vector Regression (SVR) is a variant of the SVM algorithm that is widely employed for regression analysis. The SVR algorithm finds the function f(x) that approximates the relationship between the input features and the output values. The function f(x) is defined as in Eq. (4.6):

$$f(x) = w * x + b \tag{4.6}$$

where f(x) is the predicted output value, x is the input vector of features, w is the weight vector, and b is the bias term. The goal of SVR is to find the optimal values of w and b that minimize the error between the predicted output values and the actual output values. The optimization problem for SVR can be formulated as in Eq. (4.7):

$$minimize \frac{1}{2}||w||^2 + C * \sum(\xi_i + \xi_i^*) \tag{4.7}$$

$$subject\ to: y - f(x) \leq \varepsilon + \xi_i$$

$$f(x) - y \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

where C is a positive constant, same as in SVM, that controls the trade-off between minimizing the error and minimizing the slack variables, ε is the width of the insensitive zone, and y is the actual output value. The optimization problem in SVM and SVR has some similarities, but they have different constraints. The constraints of SVM aim to maximize the margin between the classes, while the constraints of SVR focus on minimizing the error margin of the function f(x). Additionally, the loss function used to measure the error in SVR and SVR is different. Figure 4.2 shows SVR.

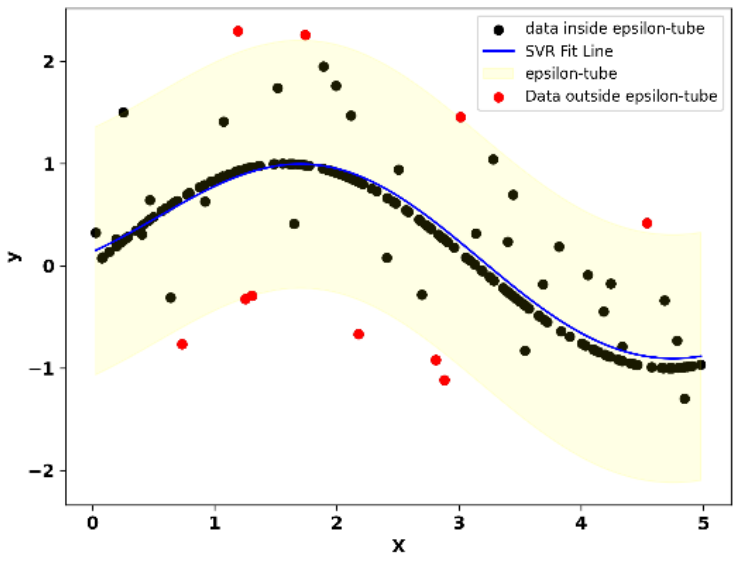

Figure 4.2 Support Vector Regression

4.3.4 The Experiment

This study compares the ARIMA model to SVR for five indices: (DJIA), NIFTY 50 (NSEI), DAX, Nikkei 225 (NI225), and SSE Composite Index (SSE). The data is gathered from Yahoo Finance which is daily data for around five years, from January 2017 to July 2022. Two types of models, namely static and iterative, are designed and evaluated. Static models of ARIMA and SVR are trained on 75% of the data and predict prices for the remaining 25% of the days. On the other hand, iterative models of ARIMA and SVR are first trained on 75% of the data and then predict the next day's price. Once the price is predicted for the next day, the original price is used to retrain the iterative models before predicting the price of the upcoming day. This process continues until all prices for the test duration are predicted in succession. The models are compared using RMSE, MAPE, MSE, and MAE.

The hyperparameters of the SVR models are selected automatically using grid search to eliminate bias in the experiments. Figure 4.3 shows the flow of the SVR model implementation.

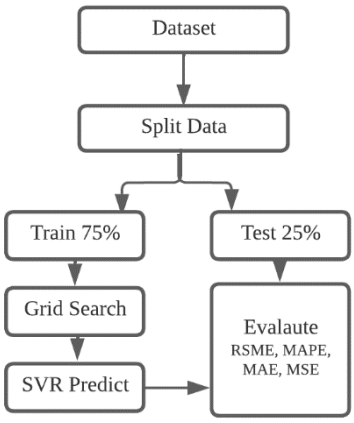

Figure 4.3 SVR algorithm implementation

4.4 Findings

All time series of five indices are tested for stationarity using the Augmented Dickey-Fuller (ADF) test. Subsequently, the ARIMA model is selected using the Akaike Information Criterion (AIC). Table 4.1 shows the output of the ADF test on the original time series.

Table 4.1 ADF Test

	NSEI	DJI	DAX	NI225	SSE
ADF Statistic:	-0.84	-1.6	-2.3	-1.7	-2.1
p-value:	0.8	0.48	0.17	0.42	0.24
usedlag:	7	10	7	4	0
The number of observations:	1348	1372	1383	1336	1332
Critical Values: 1%	-3.43	-3.43	-3.43	-3.43	-3.43
Critical Values: 5%	-2.86	-2.86	-2.86	-2.86	-2.86
Critical Values: 10%	-2.56	-2.56	-2.56	-2.56	-2.56

The ADF statistics of all time series are greater than critical values at 1%, 5%, and 10% levels. Hence, all stock indices' time series are non-stationary. The auto-Arima function is utilized to find the best parameters of ARIMA (p, d, q). Table 4.2 shows the list of parameters experimented with and their AIC values. The best parameters with the lowest AIC values are in bold font.

Table 4.2 ARIMA(p,d,q) and AIC of each index

NSEI		DJI		DAX		NI225		SSE	
(p,d,q)	**AIC**	**(p,d,q)**	**AIC**	**(p,d,q)**	**AIC**	**(p,d,q)**	**AIC**	**(p,d,q)**	**AIC**
(0,1,0)	17206	(0,1,0)	19941	(0,1,0)	17991	(0,1,0)	18904	(0,1,0)	13157
(0,1,1)	17208	(0,1,1)	19921	(0,1,1)	17993	(0,1,1)	18906	(0,1,1)	13159
(0,1,2)	17210	(0,1,2)	19904	(0,1,2)	17994	(0,1,2)	18903	(0,1,2)	13160
(1,1,0)	17208	(1,1,0)	19917	(1,1,0)	17993	(1,1,0)	18906	(1,1,0)	13159
(1,1,1)	17210	(1,1,1)	19911	(1,1,1)	17995	(1,1,1)	18903	(1,1,1)	13161
(1,1,2)	17212	(1,1,2)	19906	(1,1,2)	17995	(1,1,2)	18901	(1,1,2)	13162
(2,1,0)	17210	(2,1,0)	19905	(2,1,0)	17994	(2,1,0)	18903	(2,1,0)	13160
(2,1,1)	17212	(2,1,1)	19907	(2,1,1)	17996	**(2,1,1)**	**18901**	(2,1,1)	inf
(2,1,2)	17214	(2,1,2)	19907	(2,1,2)	17998	(2,1,2)	18903	**(2,1,2)**	**13157**
(3,1,0)	17211	(3,1,0)	19907	(3,1,0)	17996	(3,1,0)	18903	(3,1,0)	13161
(3,1,1)	17213	(3,1,1)	19862	(3,1,1)	17998	(3,1,1)	18903	(3,1,1)	13159
(3,1,2)	inf	(3,1,2)	19838	**(3,1,2)**	**17985**	(3,1,2)	18905	(3,1,2)	13158
(4,1,0)	17211	(4,1,0)	19904	(4,1,0)	17998	(4,1,0)	18903	(4,1,0)	13157
(4,1,1)	17208	(4,1,1)	19855	(4,1,1)	17999	(4,1,1)	18904	(4,1,1)	13157
(4,1,2)	17208	(4,1,2)	19841	(4,1,2)	17997	(4,1,2)	18906	(4,1,2)	13158
(5,1,0)	17198	(5,1,0)	19904	(5,1,0)	17994	(5,1,0)	18904	(5,1,0)	13158
(5,1,1)	17193	(5,1,1)	19854	(5,1,1)	17986	(5,1,1)	18903	(5,1,1)	13159
(5,1,2)	17186	(5,1,2)	19840	(5,1,2)	17987	(5,1,2)	18904	(5,1,2)	inf
(6,1,0)	17193	(6,1,0)	19887	(6,1,0)	17990	(6,1,0)	18906	(6,1,0)	13157
(6,1,1)	17193	(6,1,1)	19838	(6,1,1)	17987	(6,1,1)	18905	(6,1,1)	13158
(6,1,2)	17184	**(6,1,2)**	**19830**	(6,1,2)	17988	(6,1,2)	18908	(6,1,2)	13159
(7,1,0)	17192	(7,1,0)	19860	(7,1,0)	17989	(7,1,0)	18906	(7,1,0)	13158
(7,1,1)	17194	(7,1,1)	19834	(7,1,1)	17989	(7,1,1)	18908	(7,1,1)	13160
(7,1,2)	**17183**	(7,1,2)	19832	(7,1,2)	17991	(7,1,2)	18909	(7,1,2)	13161
(8,1,0)	17194	(8,1,0)	19847	(8,1,0)	17989	(8,1,0)	18908	(8,1,0)	13159
(8,1,1)	17196	(8,1,1)	19835	(8,1,1)	17991	(8,1,1)	18910	(8,1,1)	13161
(8,1,2)	17197	(8,1,2)	19834	(8,1,2)	17993	(8,1,2)	18911	(8,1,2)	13160

(9,1,0)	17194	(9,1,0)	19835	(9,1,0)	17991	(9,1,0)	18909	(9,1,0)	13161
(9,1,1)	17193	(9,1,1)	19834	(9,1,1)	17993	(9,1,1)	18911	(9,1,1)	13162
(9,1,2)	17193	(9,1,2)	19834	(9,1,2)	17995	(9,1,2)	18908	(9,1,2)	13162
(10,1,0)	17196	(10,1,0)	19832	(10,1,0)	17993	(10,1,0)	18909	(10,1,0)	13162
(10,1,1)	17195	(10,1,1)	19834	(10,1,1)	17995	(10,1,1)	18911	(10,1,1)	13163
(10,1,2)	17197	(10,1,2)	19836	(10,1,2)	17990	(10,1,2)	18904	(10,1,2)	13162

Support Vector Regression (SVR) is a non-parametric model. The time series is split into a 75:25 ratio in both static and iterative models. 75% of the days are used to train the SVR model, and 25% of the days are kept as test data. The prices are predicted for 25% of the days. In this study, the RBF kernel is applied to the model. Further, the dates are converted to integers and used as input. Before feeding the Close Prices to SVR, they are scaled and transformed. The close prices range from 0 to 1 while training. Once the SVR model predicts the close prices, they are again inverse-transformed to the original scale. Parameters C and epsilon determine a trade-off between model complexity and training error. This study selects the hyperparameter of SVR using grid search. The grid search selects the best parameter from Table 4.3. The grid search is designed in such a way that the values of C and epsilon avoid overfitting and underfitting.

Table 4.3 Grid Search Parameters

Parameter	Values
C	0.1, 1, 10
Epsilon	0.05, 0.1, 0.2

Six models are developed, three for static and three for iterative. Further, all six models are tested on five indices. RMSE, MAPE, MAE, and MSE are utilized to evaluate the efficiency of these models. In three static models, namely Naïve, ARIMA, and SVR, prices

are predicted for 25% of the days at once and compared with the original prices of 25% of the test data. The Naïve static model assumes that the prices for the rest of the 25% of days remain the same as the last day of the 75% of train data. In the other three iterative models, namely Naïve iterative, ARIMA iterative, and SVR iterative, price is predicted for the next day after training on 75% of the data. Once the price is predicted for the next day, the model is retrained again with newly available data, and the next-to-next-day price is predicted iteratively. Naïve iterative assumes the next day's price is the same as the previous day's close price.

4.4.1 NIFTY

Table 4.4 shows the performance of all six models in NIFTY. The RMSE, MAPE, MAE, and MSE of ARIMA static are the highest. Notably, the ARIMA static model performs worse than the Naïve static model, indicating that it is difficult for the ARIMA model to handle the complexity of financial time series.

Table 4.4 NIFTY Evaluation

Model	RMSE	MAPE	MAE	MSE
Naïve static	1728.89	8.67	1481.9	2989059
ARIMA static	1765.3	8.86	1515.63	3116274
SVR static	842.23	4.37	703.88	709349
Naïve iterative	172.33	0.79	130.97	29697
ARIMA iterative	175.75	0.81	133.08	30889
SVR iterative	556.71	2.77	452.97	309930

Among static models, Support Vector Regression (SVR) performs the best. However, SVR iterative performs worst compared to ARIMA iterative and Naïve iterative. So, the predicted prices of SVR are far from the original prices compared to the ARIMA and Naïve

predicted prices. For Nifty, Figure 4.4 shows the predicted price of NIFTY 50 for SVR and ARIMA models.

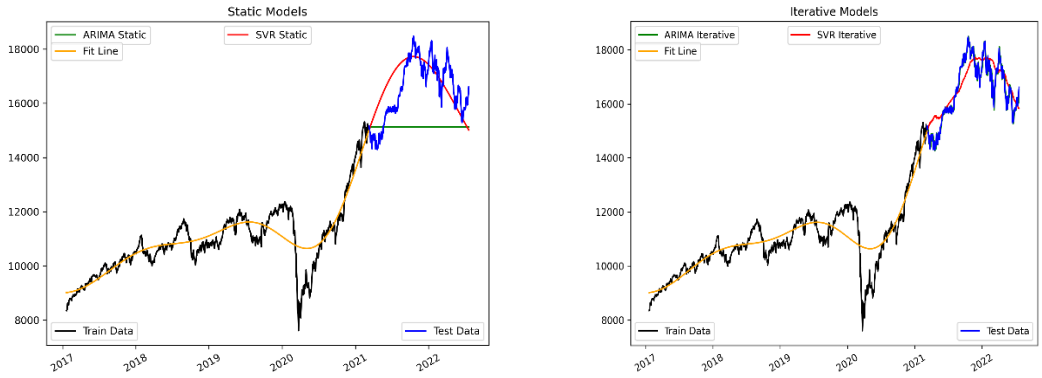

Figure 4.4 Static and iterative model prediction for NIFTY

4.4.2 DJIA

Table 5 shows the performance of all models in the Dow Jones Industrial Average (DJIA). Results vary a little compared to NIFTY in the case of DJIA.

Table 4.5 DJIA Evaluation

Model	RMSE	MAPE	MAE	MSE
Naïve static	2751.42	7.19	2499.18	7570314
ARIMA static	2730.61	7.13	2478.92	7456205
SVR static	3376.69	9.47	3258.63	11402052
Naïve iterative	329.82	0.73	247.56	108779
ARIMA iterative	335.97	0.76	256.51	112878
SVR iterative	1222.93	3.06	1029.75	1495556

In contrast to the SVR static prediction for the NIFTY, the prediction of SVR static for DJIA is the worst among the static models in terms of RMSE, MAPE, MAE, and MSE. ARIMA static performs the best in predicting prices. Although the SVR model performed poorly in terms of evaluation parameters, it is able to predict the pattern correctly. This

prediction can be important to investors. In iterative models, SVR iterative performed worst, and Naïve iterative the best. This result also indicates that the performance of the models can differ in emerging and developed economies. For DJIA, Figure 4.5 shows the predicted price of DJIA for SVR and ARIMA models.

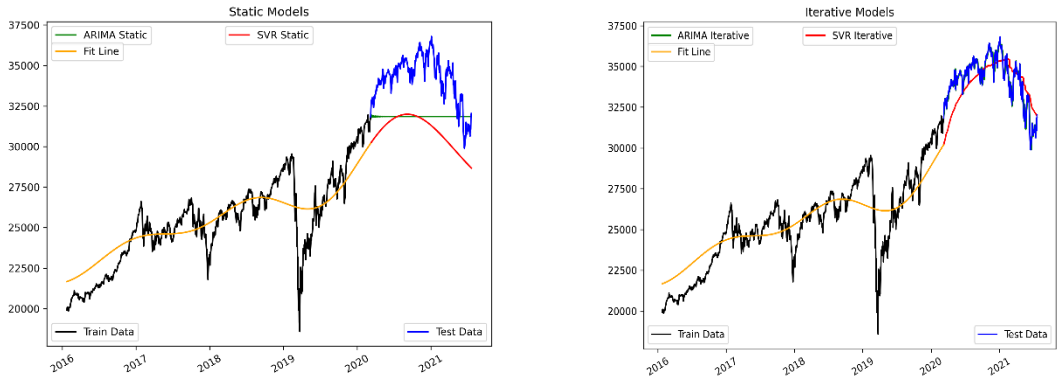

Figure 4.5 Static and iterative model prediction for DJIA

4.4.3 DAX

Table 4.6 shows the RMSE, MAPE, MAE, and MSE for the DAX index. ARIMA marginally outperformed the SVR static model. Similarly, in iterative models, naïve outperformed the ARIMA marginally.

Table 4.6 DAX Evaluation

Model	RMSE	MAPE	MAE	MSE
Naïve static	1031.86	6.15	923.67	1064732
ARIMA static	1000.75	5.98	894.62	1001507
SVR static	1001.87	5.97	905.09	1003738
Naïve iterative	182.22	0.89	131.16	33205
ARIMA iterative	185.45	0.91	133.58	34393
SVR iterative	657.85	3.69	535.82	432768

Figure 4.6 shows the prediction chart for the ARIMA and SVR models. The result hints that model performance can vary within developed countries as well.

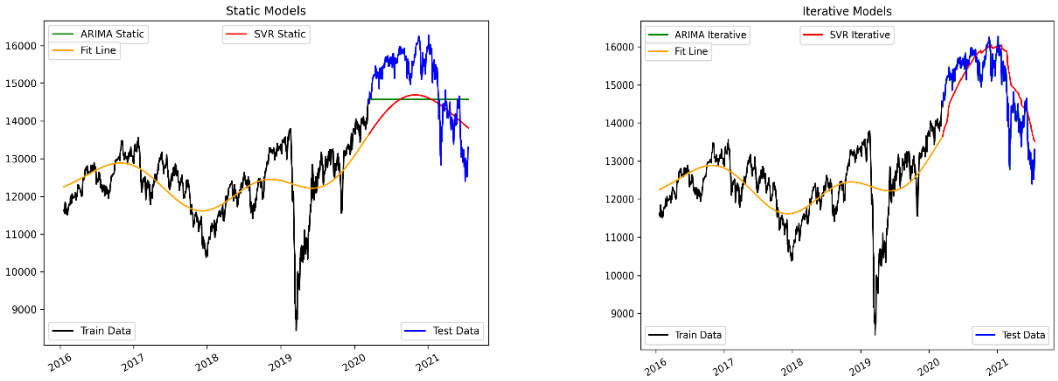

Figure 4.6 Static and iterative model prediction for DAX

4.4.4 Nikkei 225

Table 4.7 shows the performance of models on the Nikkei 225 index. The ARIMA model outperformed naïve and SVR in the static models. ARIMA iterative marginally outperformed naïve in iterative models.

Table 4.7 NI225 Evaluation

Model	RMSE	MAPE	MAE	MSE
Naïve static	1529.62	4.45	1210.93	2339745
ARIMA static	1386.46	4.04	1105.08	1922275
SVR static	2417.37	7.8	2162.27	5843693
Naïve iterative	346.59	0.98	273.54	120128
ARIMA iterative	346.54	0.98	273.32	120090
SVR iterative	1090.88	3.26	909.94	1190011

Figure 4.7 shows the fit line, prediction line, and train and test data of static and iterative models. The pattern of the Nikkei 225 index in test data is different from other indices. All

other indices, namely the Nifty, DJIA, and DAX, moved up first and then downwards, but the Nikkei 225 moved only downwards in test data. Still, the SVR model's final price is near to the final close price in the test data.

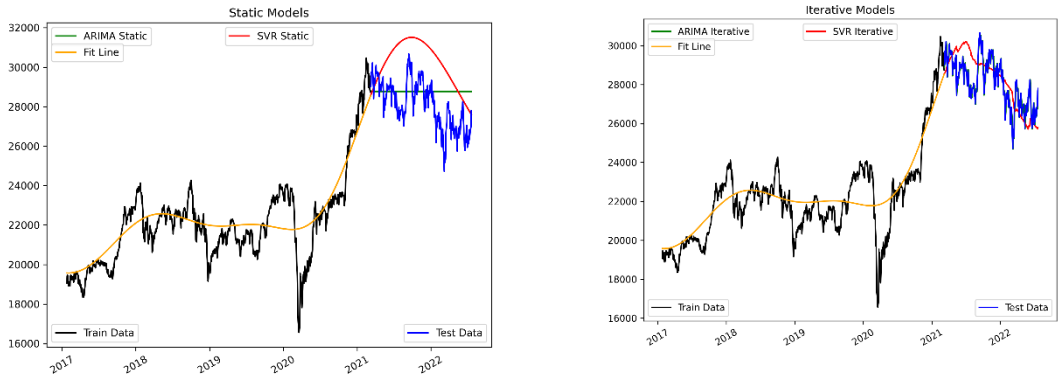

Figure 4.7 Static and iterative model prediction for NI225

4.4.5 SSE Composite Index

Table 4.8 shows the performance of models on the SSE composite index. SVR static outperformed both naïve and ARIMA static in static models. While in iterative models, naïve outperformed other iterative models.

Table 4.8 SSE Evaluation

Model	RMSE	MAPE	MAE	MSE
Naïve static	185.91	4.78	164.48	34563
ARIMA static	168.5	4.14	140.12	28392
SVR static	156.12	3.73	124.52	24375
Naïve iterative	33.9	0.75	25.39	1149
ARIMA iterative	34.26	0.76	25.79	1174
SVR iterative	117.06	2.76	92.2	13703

Figure 4.8 shows the prediction prices by ARIMA and SVR for both static and iterative models. Further, Figure 4.8 clearly shows that the SSE composite index pattern is completely different from other global indices. Still, the SVR static model is able to closely predict the future pattern.

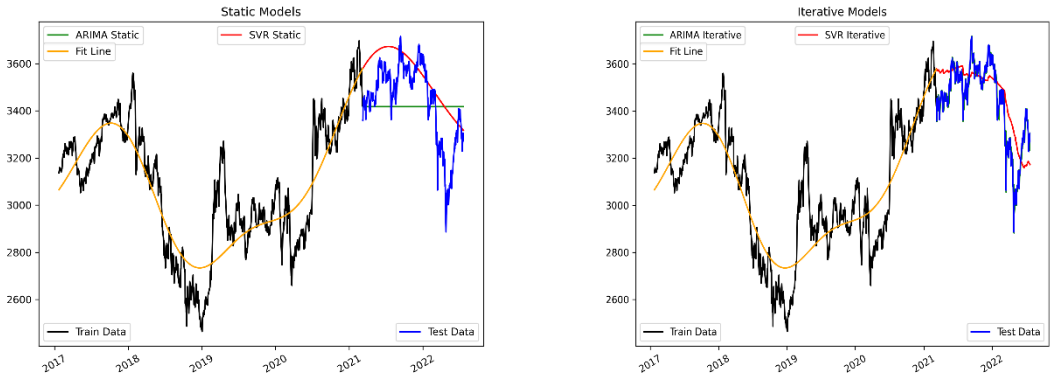

Figure 4.8 Static and iterative model prediction for SSE

4.5 Significant Outcomes

Significant outcomes from the analysis are as follows:

- ARIMA static outperforms SVR static in the DJIA and NI225 indices. In DAX, both have similar performance. This suggests SVR's low performance in developed economies.
- SVR static exhibits lower MAPEs for Nifty and SSE compared to DJIA, DAX, and NI225, indicating better predictability in emerging economies.

- The Naïve model performs better than ARIMA iterative and SVR iterative models in next-day price prediction. This indicates that the short-term prediction ability of both ARIMA iterative and SVR iterative is similar to random forecasting.
- The prices predicted by ARIMA in the long term do not exhibit variation.

In summary, the study suggests that the SVR static model has better predictability than the ARIMA static model over the long term in emerging economies. Unlike ARIMA static, SVR static is able to suggest both future prices and patterns.

This chapter is based on the study:

Beniwal, M., Singh, A., & Kumar, N. " A comparative study of static and iterative models of ARIMA and SVR to predict stock indices prices in developed and emerging economies." **International Journal of Applied Management Science (2023)**

5 Long-term Price Forecasting using Optimized GA and SVR

5.1 Overview

Predicting long-term stock index prices is a challenging and debatable task. Most of the studies focus on predicting next-day stock prices. However, those are not useful to long-term investors and traders. This study attempts to predict up to a year's daily prices of global stock indices using daily data of close prices. This study fills a gap in the existing literature by focusing on long-term stock index price forecasting, which is crucial for practical applications in the financial markets. Moreover, the empirical analysis highlights the superior performance of a rolling forward-validation approach over cross-validation in predicting long-term stock prices. A forward-validating Genetic Algorithm Optimization for Support Vector Regression (OGA-SVR) is used to efficiently forecast multi-step ahead long-term global stock indices. Further, the performance of the model is compared with that of Support Vector Regression (SVR), Grid Search based Support Vector Regression (GS-SVR), Genetic algorithm-based Support Vector regression (GA-SVM), and state-of-the-art Long Short-Term Memory (LSTM) algorithms.

The models are empirically tested on daily data from five global stock indices time series, namely the Nifty, Dow Jones Industrial Average (DJIA), DAX performance index (DAX), Nikkei 225 (NI225), and Shanghai Stock Exchange composite index (SSE). Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) are used for the evaluation. The result shows the OGA-SVR model outperforms other models in predicting the long-term prices of global indices. Further, the OGA-SVR model has the potential to

forecast the long-term underlying future pattern of index prices, which can be used to build trading and risk mitigation systems for investors and traders.

5.2 Background

Benjamin Graham once said that a stock market is a voting machine in the short run, but in the long run, it is a weighing machine (Graham & David, 1965). The stock index prices are also biased toward an upward direction in the long term. However, Investing or trading in the stock market is risky. Still, the stock market attracts many investors and traders because of its ease of accessibility, the possibility of diversification, and the potential to give lucrative returns.

With the advent of electronic trading and an explosion in stock data, researchers have explored the use of artificial intelligence to predict the stock market. Machine learning, a subset of artificial intelligence, has the capability to handle non-linear, noisy, huge, and complex data with ease. LSTM and SVM/SVR are the most commonly used machine learning algorithms for forecasting financial time series (Henrique et al., 2018). This study experiments with Support Vector Regression (SVR) and LSTM algorithms to predict the stock indices' prices.

To implement machine learning algorithms successfully, optimizing the hyperparameters to achieve the best possible model architecture is often necessary. However, refining hyperparameters through a trial-and-error approach can be time-consuming for a human ML expert. An alternative approach is to use Automated Machine Learning (AutoML), which can automate several phases of machine learning, such as data preprocessing, feature selection, and hyperparameter optimization. Hyperparameter

optimization is a subfield of automated machine learning (AutoML) that automates the selection of hyperparameters for machine learning algorithms, leading to the creation of optimized models that are tailored to specific datasets and algorithms (Drachal & Pawłowski, 2021; Taljard, 2021). The common approaches for automated hyperparameter optimization include Grid-Search (GS), Random Search, Bayesian Optimization, and Genetic Algorithms (GA) (Drachal & Pawłowski, 2021).

Grid search performs an exhaustive search with a predetermined set of hyperparameters to find the optimum combination of hyperparameters for a given model (Bergstra & Bengio, 2012). The genetic algorithm, on the other hand, is a heuristic optimization technique that emulates natural selection and evolution to find the optimum hyperparameters (Holland, 1992). SVR has two important hyperparameters, C and epsilon, that control the trade-off between model complexity and error. Achieving high prediction accuracy using a single machine-learning method can be challenging, and combining soft computing methods can improve accuracy (Lu et al., 2021). In this study, grid-search and genetic algorithms are utilized to optimize these two hyperparameters of the SVR model.

Cross-validation is a commonly used technique for estimating the performance of a model on out-of-sample data, and it is widely used for both model selection and assessment (Schnaubelt, 2019). One of the initial validations is the leave-one-out cross-validation proposed by Stone (1974) and Allen (1974). Out of several extensions of leave-one-out cross-validation, k-fold cross-validation (Geisser, 1975) is the most common (Schnaubelt, 2019). K-fold cross-validation randomly splits data in k equally size subset. The model is trained on the k-1 subset and tested on the remaining subset. This method assumes that

observations are independent and identically distributed. However, in time series analysis, this assumption does not hold true. An alternative to cross-validation is forward validation. Schnaubelt (2019) empirically studied common validation schemes and concluded that forward-validation techniques provide more accurate estimates of the out-of-sample error.

In forward validation, data is divided into consecutive intervals, with each interval representing a specific time period rather than being randomly split. The model is trained on earlier intervals and tested on later intervals, making it especially helpful for time series data. Therefore, this study's optimized genetic algorithm-based SVR model (OGA-SVR) uses a specific type of forward validation known as "rolling window forward validation" to estimate the model's performance on out-of-sample data. The rolling window forward validation takes into account any temporal patterns that may exist in the time series data.

Machine learning involves organizing data into input features and output targets, but the task becomes more complex when it comes to multi-step prediction. Multi-step prediction involves mapping multiple output targets to input features. This process can be particularly challenging when attempting to map long-term prices, such as daily prices over a year, to input features. In practice, including long-term price data as an input feature is often not feasible. Most studies predict next-day stock prices, with some predicting intraday prices (Nazareth & Reddy, 2023). Most studies use daily data as a sampling frequency, so the prediction frequency is also typically daily. However, accurately predicting stock prices over longer time horizons remains a challenging problem in machine learning. Many traders and investors are interested in predicting long-term prices, such as yearly predictions. Rouf et al. (2021), in their literature review, reported 24 studies, and those studies predicted

intraday, daily, weekly, or up to 90 days, but no study predicted yearly prices. This gap motivated this study to predict the global indices' long-term prices using the SVR and LSTM algorithms. This study trains SVR and LSTM to identify patterns based on time dependency. As time is known for the future and provided as input to the models, our models can predict stock prices for any future duration, such as monthly, quarterly, yearly, etc., using daily historical data.

This study experiments with predicting the long-term prices of the five global indices with five models. The five global indexes are the Nifty from India, the Dow Jones Industrial Average (DJIA) from the US, the DAX performance index (DAX) from Germany, the Nikkei 225 (NI225) from Japan, and the Shanghai Stock Exchange composite index (SSE) from China. These are stock indices from the world's top five economies in terms of Gross Domestic Product (GDP) (*Countries by GDP*, 2022). The study optimizes the hyperparameters of the SVR model using a genetic algorithm and rolling window forward validation. The rolling forward-validation approach enables the model to adapt to new data and avoid overfitting, while the genetic algorithm helps to explore a large search space of possible solutions and find the optimal ones. The Optimized Genetic Algorithm based on SVR (OGA-SVR) predicts one year ahead of stock indices prices. The optimized hybrid model OGA-SVR predictions are compared with those of SVR, Grid Search-based SVR (GS-SVR), Genetic Algorithm-based SVR (GA-SVR), and LSTM. The models are evaluated using Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) on out-of-sample data.

The objective of this study is to contribute to the field of stock price prediction by using daily historical data to predict one year ahead of stock prices. This study aims to demonstrate the superior performance of a rolling forward-validated genetic algorithm support vector regression (SVR) model over cross-validated SVR and genetic algorithm SVR models. The study also seeks to optimize the OGA-SVR model to improve its predictions in comparison to other SVR models and a double-layer long short-term memory (LSTM) deep learning model. Furthermore, the study aims to analyze and predict the stock prices of multiple global indices to evaluate the optimized model's robustness and ability to perform well on different time series data. Ultimately, this research endeavors to advance the development of effective long-term prediction models for stock prices and help investors and traders make informed decisions over long-term time horizons.

5.3 Proposed Methodology

5.3.1 Genetic Algorithm (GA)

The inspiration for Genetic Algorithms (GAs) came from Darwin's theory of evolution, which involves simulating the survival of fitter creatures and their genes (De, 1988; Holland, 1992; Mirjalili, 2019). Figure 5.1 shows the steps involved in GA.

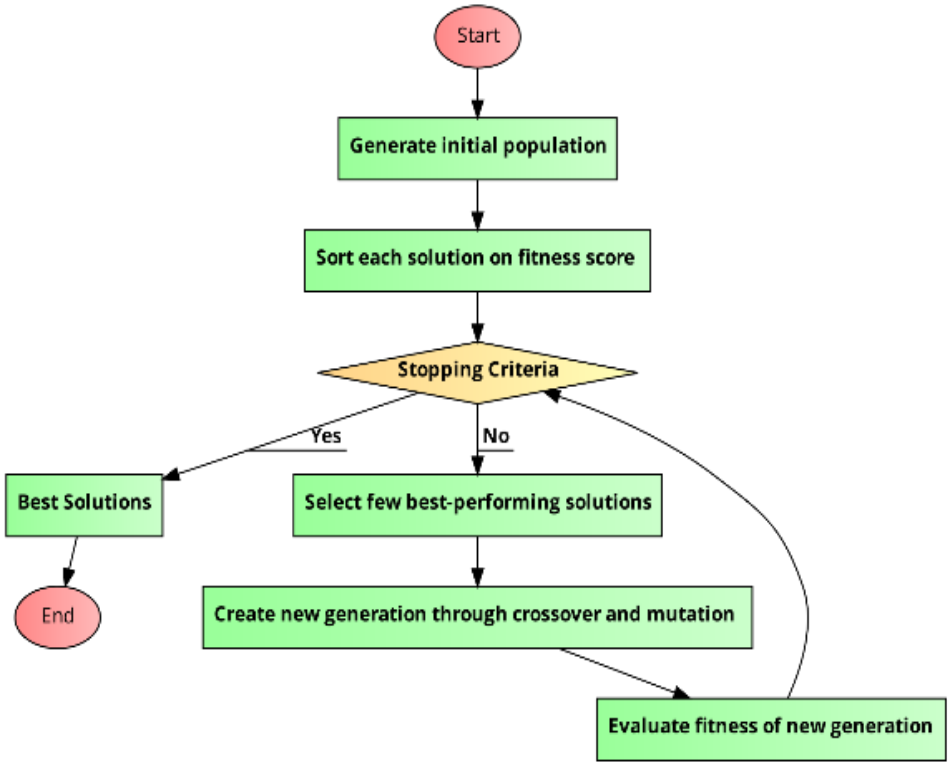

Figure 5.1 Genetic Algorithm Flow

When the problem is computationally demanding and difficult to solve, GA can be used to obtain a solution that is close to optimal (Chung & Shin, 2020; Kramer & Kramer, 2017). GAs are often used in ML and time series forecasting, where they can be used to identify optimal hyperparameters for ML models and to search for the best possible models for predicting future values in time series data. GA involves six stages: initialization, fitness calculation, selection, crossover, mutation, and termination condition check (Pal & Wang, 1996). Figure 5.2 shows the GA operations.

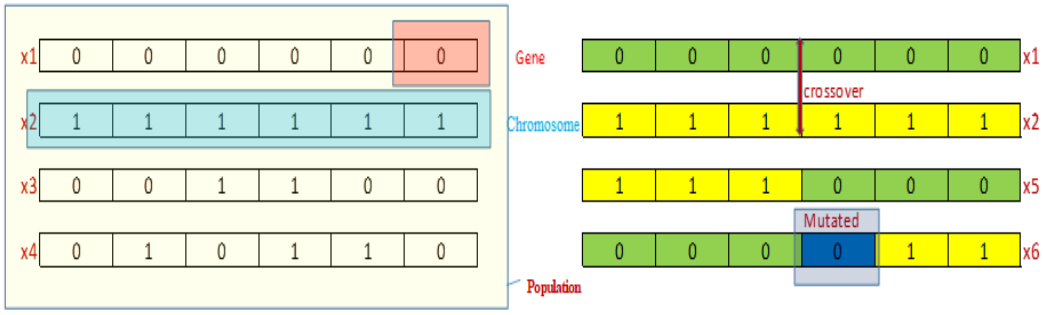

Figure 5.2 Genetic Algorithm Operations

GA starts by randomly generating a population of chromosomes that represent potential solutions to a problem. These solutions are sorted based on a fitness score like accuracy, RMSE, etc. If the stopping criteria are met, the best solutions are returned; otherwise, a new generation is created through crossover and mutation. The new generation is evaluated again, and the process continues iteratively until the stopping criteria are met.

5.3.2 Rolling Window Forward Validation

One of the crucial tools for assessing the performance of regression and classification techniques is cross-validation (Bergmeir & Benítez, 2012). Cross-validation is widely utilized for evaluating how well a model performs on data that has not been seen during training, and it is commonly employed for both selecting and evaluating models. Cross-validation involves dividing a dataset into multiple subsets, or folds, for model training and testing. One-fold is held out as a validation set for evaluation, while the remaining folds are used for training. This process is repeated for each fold, and the performance metrics are averaged to estimate the performance of generalization on unseen or out-of-sample data. The

validation set is important to optimize hyperparameters and prevent overfitting of the model during training. K-fold cross-validation, a variation of cross-validation, is the most common (Schnaubelt, 2019). K-fold cross-validation divides the dataset into k equal-sized subsets, where k is a user-defined positive integer. 5-fold cross-validation and 10-fold cross-validation are frequently used. After division, the model is trained on k-1 partitions and evaluated on the unseen partition. This process is repeated k times, with each subset serving as the validation set once for training a model. The evaluation metric, such as accuracy or mean squared error, is recorded after each run. Lastly, the average is computed across all k runs for the final evaluation metric. This approach is a more reliable estimation of the performance of unseen data.

Cross-validation assumes that observations are independent and identically distributed, which may not hold true in time series analysis. As an alternative, forward validation can be used. Forward validation is a technique that partitions the dataset into consecutive subsets, based on time instead of random assignment. This technique is particularly useful for time series data, where the temporal order of observations is important. By training the model on earlier subsets and validating it on later subsets, forward validation simulates the real-world scenario in which the model is deployed to predict future values based on historical data. One of the variants of forward validation is rolling window forward validation. In rolling window forward validation, the size of the subsets remains constant, but they are shifted forward by a certain number of observations at each step. Rolling window forward validation can be especially helpful for stock time series data,

where recent data is more relevant in predicting the future. Figure 5.3 visually explains cross-validation and rolling window forward validation.

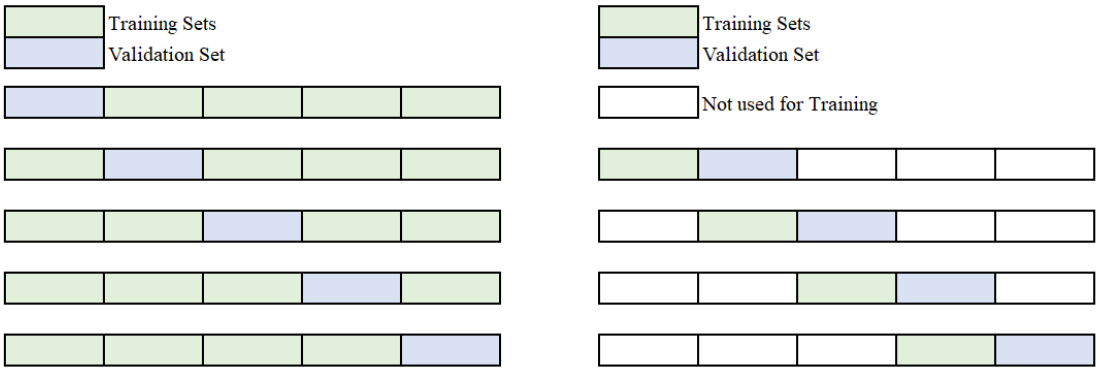

Figure 5.3 Five-Fold Cross Validation and Rolling Window Forward Validation

5.3.3 Grid Search

Grid search (GS) is a technique that involves searching through a pre-defined set of hyperparameters to identify the optimal combination of hyperparameters for a given model by performing an exhaustive search (Bergstra & Bengio, 2012). GS divides parameters into grids of the same length within a certain range, with each point representing a set of parameters. Traversing all the points in the grid enables obtaining the optimal solution (Sun et al., 2021). The model is then trained and evaluated for each combination of hyperparameters in the grid, and the optimal combination is selected based on the performance metrics. GS is commonly used with supervised ML algorithms, such as SVR. For example, when using SVR, one can fine-tune hyperparameters such as the penalty parameter 'C' and the gamma parameter. A GS can test all possible combinations of values within specified ranges. For instance, the grid may include values of the penalty parameter

(1, 10, 100) and gamma parameter (0.001, 0.01, 0.1). Figure 5.4 shows an example of the combination of the parameters in the grid search. GS can try and evaluate each combination of these hyperparameters, and the combination with the best performance on the cross-validation dataset is selected. GS uses brute force. As the number of hyperparameters and their ranges increases, the number of models to be trained and evaluated increases exponentially. Therefore, this process can become computationally expensive, especially when dealing with many hyperparameters and a large dataset.

C/Gamma	0.001	0.01	0.1
1	1 , 0.001	1 , 0.01	1 , 0.1
10	10 , 0.001	10 , 0.01	10 , 0.1
100	100 , 0.001	100 , 0.01	100 , 0.1

Figure 5.4 Grid Search Example

5.3.4 Prediction Models and Assumptions

This study experiments with five models, namely Support Vector Regression (SVR), Grid Search based Support Vector Regression (GS-SVR), Genetic Algorithm based Support Vector Regression (GA-SVR), Optimized Genetic Algorithm Support Vector Regression (OGA-SVR), and Long Short-Term Memory (LSTM). The study optimized the predictions of OGA-SVR using a genetic algorithm and rolling window forward validation to tune the hyperparameters of SVR. The other models, SVR, GS-SVR, GA-SVR, and LSTM, are used as baseline models to compare the performance of the optimized model, OGA-SVR. In this study, the Radial Bias Kernel (RBF) and default value of gamma in the Scikit-learn Python library are used for all models of SVR. The default value of gamma in the Scikit-learn Python library is ‘scale’ as shown in the equation Eq. (5.1):

$$Thank\ you\ for\ reaching\ out.Scale = \frac{1}{n\_feature \times Var(X)} \qquad (5.1)$$

The other two important hyperparameters of SVR are C and epsilon, which play a critical role in training the model. The cost parameter, denoted by C, is a regularization parameter that balances the trade-off between minimizing the training error and the complexity of the model. When C is small, the margin is wider, and the model allows for more training errors. Conversely, a larger C value leads to a narrower margin and fewer training errors. In SVR, C controls the degree to which the margin is allowed to be violated in the training data. Epsilon is another threshold parameter in SVR that sets the minimum distance between the predicted and actual values before an error is counted. If the predicted value is within the epsilon distance of the actual value, it is considered accurate and has zero error. Any predicted value beyond the epsilon distance of the actual value is considered inaccurate and has a non-zero error. Choosing a smaller value of epsilon increases the sensitivity of the model to errors, whereas a larger value of epsilon results in a less sensitive model. These two parameters of SVR are optimized in other models of SVR. Finally, the predictions made by the models are evaluated using RMSE and MAPE. Figure 5.5 shows the general flow chart of prediction algorithms.

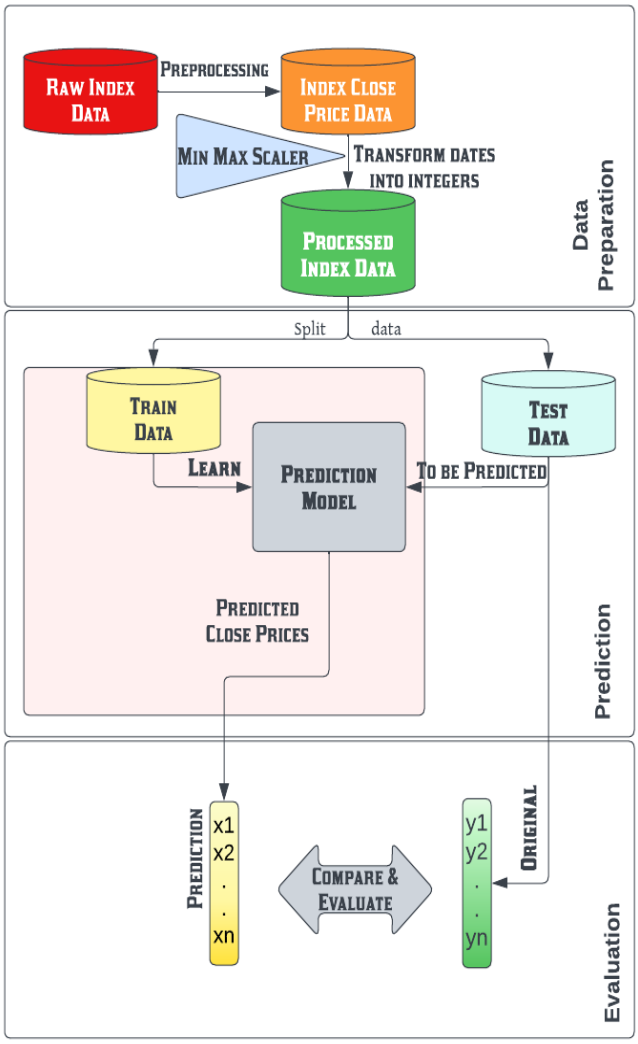

Figure 5.5 Flowchart of prediction methods

The complete process is done in three steps: data preparation, prediction, and evaluation. The following are the assumptions of the prediction models:

- The models assume the future pattern of the stock prices is related to historical data. This is against the random market hypothesis but aligns with technical analysis.

- The experimental model OGA-SVR assumes that rolling forward validation is more appropriate than cross-validation to improve predictability for stock market forecasting.

5.4 Experimental Setups

5.4.1 SVR

SVR is the base model for this study. The model SVR is trained without hyperparameter optimization, and default values of hyperparameters are used. This is helpful in evaluating other models' efficiency. Table 5.1 lists the default value of SVR. Fig. 8 shows the flowchart of the prediction algorithm of SVR.

Table 5.1 SVR hyperparameters

Hyperparameter	Values
Kernel	RBF
C	1
Epsilon	0.1
Gamma	Scale

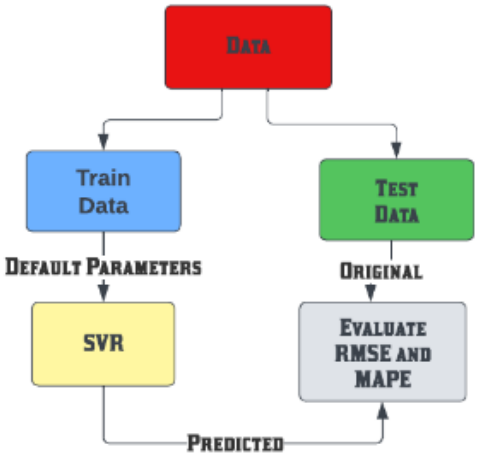

Figure 5.6 Flowchart SVR prediction algorithm

5.4.2 GS-SVR

The Grid Search-based Support Vector Regression (GS-SVR) algorithm uses a grid search approach to optimize the hyperparameters C and epsilon. Specifically, the Scikit-learn Python library's grid search function estimates the performance of different hyperparameters using cross-validation (CV). Table 5.2 shows the input to the grid search. Figure 5.7 shows the flowchart of GS-SVR prediction.

Table 5.2 Grid Search Parameters

Parameters	Values
param_grid	C: [0.001, 0.01, 0.1, 1, 10, 100] Epsilon: [0.1, 0.5, 0.1, 0.15, 0.2, 0.25]
scoring	neg_root_mean_squared_error
cv	10

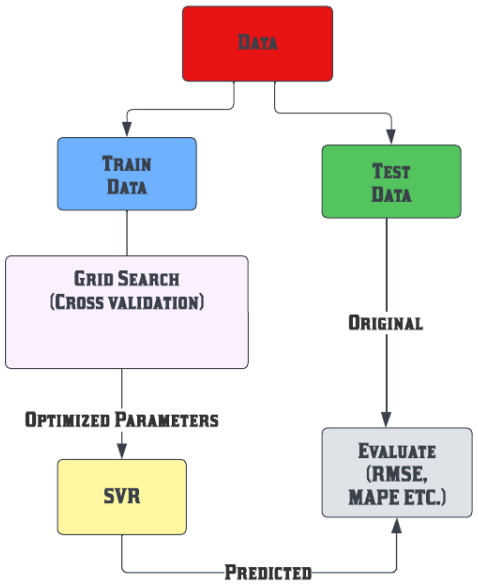

Figure 5.7 Flowchart GS-SVR prediction algorithm

In this study, using grid search, 10-fold cross-validation was used to evaluate the performance of different hyperparameter combinations, with the RMSE used as the loss measurement. This approach aids in the training process by identifying the optimal combination of hyperparameters, which are then utilized in the Support Vector Regression (SVR) algorithm to make predictions about future prices.

5.4.3 GA-SVR

Genetic algorithm-based Support Vector Regression (GA-SVR) used a genetic algorithm instead of a grid search to optimize the hyperparameters C and epsilon. GS and GS are both techniques used for hyperparameter optimization but differ in their approaches. Grid search is a brute-force approach that evaluates the performance of all possible hyperparameter combinations within a pre-defined range. On the other hand, genetic algorithms are inspired by the natural selection process and the survival of the fittest. Figure 5.8 shows the steps of the genetic algorithm in this study.

Figure 5.8 Genetic Algorithm Steps

Genetic algorithms use an iterative process to evaluate and optimize hyperparameters. In each iteration, the algorithm selects a set of hyperparameters, creates new sets by combining and mutating them, and evaluates their performance. The best-performing hyperparameters are then selected for the next iteration, and the process is

repeated until exit criteria are met. In the GA-SVR process, full training data is utilized to converge on the best parameter.

The initial population comprises pairwise combinations of values of C and epsilon from Table 5.3. Each value from the C list is paired with every value from the epsilon list, resulting in 36 initial solutions. The top 50% of individuals in the population are selected as parents and allowed to undergo crossover with randomly created new children. Further, the mutation is permitted in 20% of the population, allowing the insertion of any random value in the individual solution, subject to the limit of maximum and minimum values of C and epsilon from Table 5.3.

Table 5.3 Chromosome Initial Population

Chromosome	Initial Values
C	0.001, 0.01, 0.1, 1, 10, 100
Epsilon	0.1, 0.5, 0.1, 0.15, 0.2, 0.25

The exit criterion in this study is a maximum number of generations, which is set at 30. Similarly, Yun et al. (2021) also used 30 generations while optimizing their machine-learning models. The fitness function of GA-SVR aims to minimize the RMSE of the fit value of prices and training prices. Figure 5.9 shows the flowchart of GA-SVR.

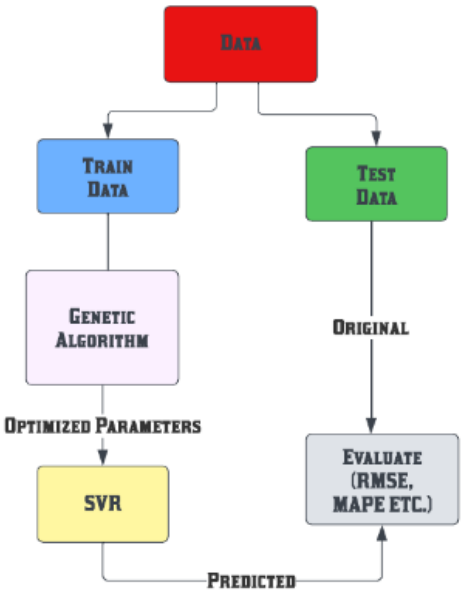

Figure 5.9 Flowchart GA-SVR prediction algorithm

5.4.4 OGA-SVR

Optimized Genetic Algorithm-based Support Vector Regression (OGA-SVR) utilizes rolling window forward validation along with a genetic algorithm to optimize the hyperparameters C and epsilon of SVR. The genetic algorithm aims to find the best parameter on full training data, unlike grid search. In the Scikit-learn Python library, grid search has cross-validation inbuilt into it to estimate the performance on unseen data. In contrast to grid search, genetic algorithms do not inherently include cross-validation as a part of the optimization process on training data. This can lead to variations in the performance of the model on unseen data. Further, the assumption of independent and identically distributed observations, which is made in cross-validation, is not valid in time series analysis. Forward validation is proposed as an alternative to cross-validation.

According to the empirical study of Schnaubelt (2019), forward-validation techniques were found to provide more accurate estimates of the out-of-sample error than other validation schemes. Hence, in this study, to optimize the performance on unseen data, rolling window forward validation is incorporated in the fitness function of the genetic algorithm in the model OGA-SVR. Figure 5.10 shows the flowchart of the OGA-SVR prediction algorithm.

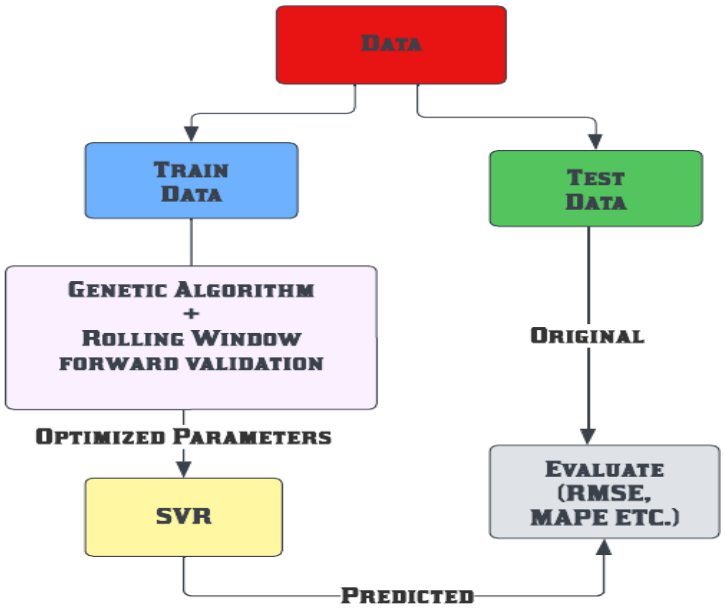

Figure 5.10 Flowchart OGA-SVR prediction algorithm

In this study, rolling window forward validation divides the data year-wise and trains the model on one year's data while validating it on the next year's data. In the next step, the model is trained on the data it was validated on and is further validated in the next year of

the currently trained window. This process continues until the training reaches the last year of data. The fitness function of OGA-SVR aims to minimize the average RMSE of all validated data, whereas the fitness function of the genetic algorithm aims to minimize the RMSE of the full training dataset. The remaining operations of OGA-SVR, including population initialization, crossover, and mutation, are similar to those of GA-SVR, as discussed in the preceding section.

5.4.5 LSTM

The Long Short-Term Memory (LSTM) algorithm is a powerful tool for processing sequential data, such as stock time series data. In this study, after trial and error, two LSTM layers and a dense output layer are stacked sequentially. The LSTM layers are recurrent neural networks that can handle sequential data, while the dense layer is a fully connected neural network layer that processes the output of the LSTM layers to produce a final output. Using two LSTM layers makes this architecture deeper than a single LSTM layer architecture, which can handle complex patterns of stock prices.

Additionally, the study added dropout layers to reduce overfitting and make the model more robust. Dropout is a regularization technique commonly used in neural networks to reduce overfitting. The dropout process involves the selective omission of a portion of the neurons in a layer during training by setting them to zero. Figure 5.11 shows the architecture of the LSTM model.

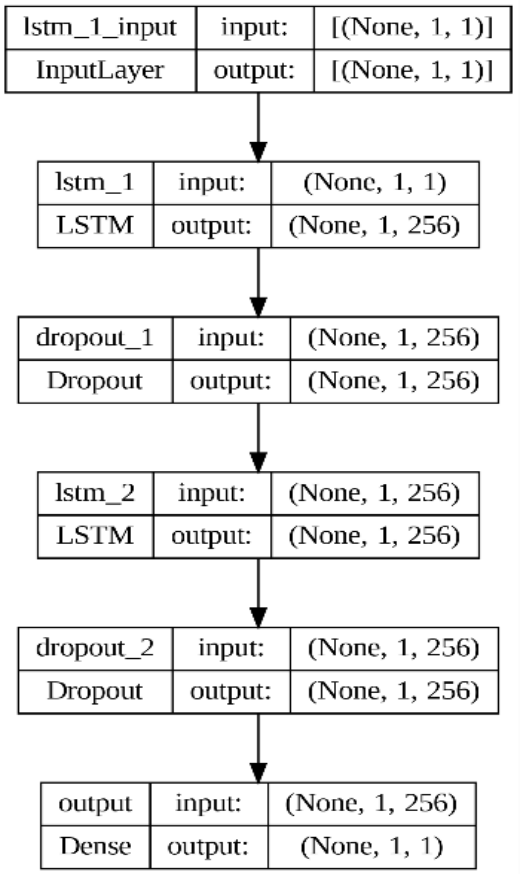

Figure 5.11 LSTM architecture

This dropout approach helps the neural network learn more resilient features, preventing it from becoming overly reliant on any one feature. The number of epochs is 100, the batch size is 32, and the dropout percentage is 20% for all indices.

5.5 Findings

This study experiments with five global indices: the Nifty from India, the Dow Jones Industrial Average (DJIA) from the US, the DAX performance index (DAX) from Germany, the Nikkei 225 (NI225) from Japan, and the Shanghai Stock Exchange composite index (SSE) from China. These stock indices represent the world's top five economies in terms of GDP. The study obtains 10 years of data for all indices from Yahoo Finance, spanning from January 1st, 2013 to December 31st, 2022. The training set uses data from January 1st, 2013, to December 31st, 2021, and the testing set reserves data from January 1st, 2022, to December 31st, 2022. The data is then divided into training and testing sets, with a ratio of 90:10 for all models. In GS-SVR, the training data is then split for 10-fold cross-validation. In contrast, in OGA-SVR, the data is divided year-wise to utilize consecutive years of data in training and validation in rolling window forward validation. The data is preprocessed for training by removing fields such as Open, High, and Low prices and the Adjusted Close and Volume fields. The data is then reduced to only include the date index and close price. The date index is encoded into integers ranging from 1 to n. The encoded values of dates and close prices are further transformed using the min-max scaler equation shown in Eq. (5.2)

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{5.2}$$

where X is the feature matrix, X_{min} and X_{max} represent the minimum and maximum values of X, respectively. Since input features rely on previous prices, the models can predict prices for any time in the future using only future dates. After training, the models predict future prices using future dates from the testing data. The data is then inverse-transformed

and compared to the testing data. The study evaluates the results using RMSE and MAPE in the training and testing data. RMSE is used to compare the model on the same dataset, while MAPE is used to compare models on a different dataset.

5.5.1 Nifty

Table 5.4 shows the results for NIFTY during training and testing. There is no difference in performance between SVR and GS-SVR, as their train and test RMSE and MAPE values are the same. This is because the best-estimated parameters by grid search for GS-SVR are the same as the default parameters of SVR. In contrast, GA-SVR performs better in both the training and testing data than SVR and GS-SVR.

Table 5.4 Nifty

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
SVR	765.38	2787.05	7.24	15.29
GS-SVR	765.38	2787.05	7.24	15.29
GA-SVR	729.72	1721.7	6.64	8.87
OGA-SVR	1016.26	1296.24	9.18	6.36
LSTM	663.57	1522.84	5.61	7.72

Due to its strong capability to handle sequential data, LSTM has the second-best performance on the test data and the best on the training data. The optimized model, OGA-SVR, performs best on the test data but worst on the training data. Fig. 14 shows the models' prediction chart, where the gray line divides the training and testing data.

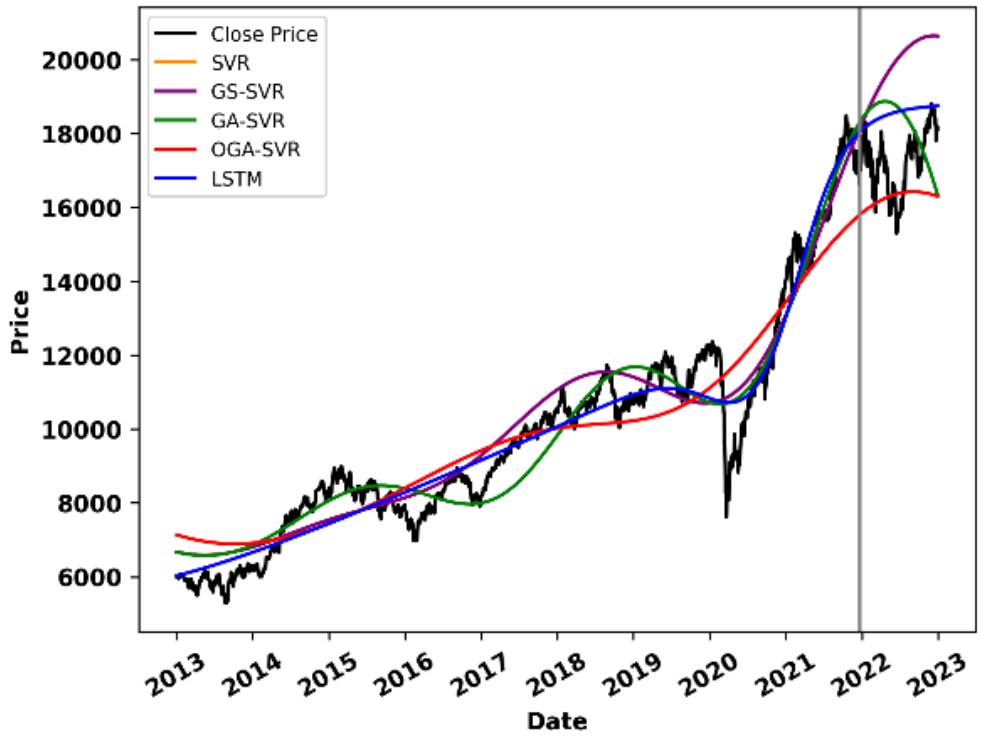

Figure 5.12 Performance of models on NIFTY

5.5.2 DJIA

Table 5.5 presents the performance of all models on DJIA. Similar to the Nifty results, SVR and GS-SVR exhibit similar performance. However, both SVR and GS-SVR demonstrate poor performance compared to the other models. GA-SVR exhibits some improvement in training and testing data compared to SVR and GS-SVR. LSTM exhibits the lowest RMSE and MAPE on training data but the second-best on testing data. The optimized model exhibits the worst performance on training data but the best on testing data. Figure 5.13 visualizes the performance of all models.

Table 5.5 DJIA

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
SVR	1315.96	7064.22	5.04	20.28
GS-SVR	1315.96	7064.22	5.04	20.28
GA-SVR	1244.76	6653.96	4.44	18.95
OGA-SVR	1591.99	1961.45	5.98	4.87
LSTM	1625.81	2827.79	5.01	7.39

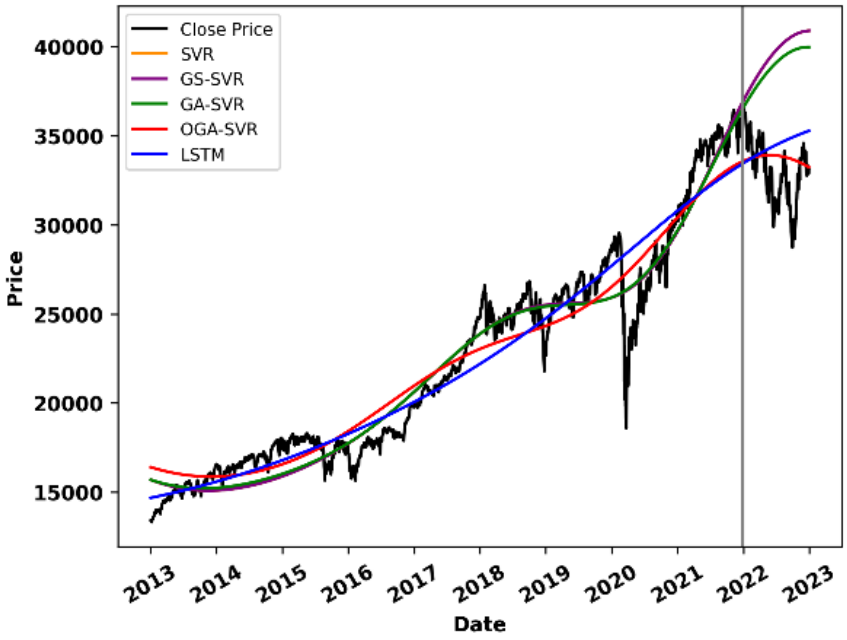

Figure 5.13 Performance of models on DJIA

5.5.3 DAX

SVR has the worst performance on testing data. The GS-SVR performs better than the SVR, but both show poor performance compared to other models. The GA-SVR model performs best on training data compared to all other models but does not generalize well on testing data, indicating overfitting. Figure 5.14 illustrates the fit and prediction prices of all

models. Interestingly, unlike the previous LSTM performance on NIFTY and DJIA, the performance of LSTM is lower than that of GA-SVR on DAX. The optimized model OGA-SVR shows the lowest performance on training but the best on testing data. Table 5.6 presents the performance of all models on DAX.

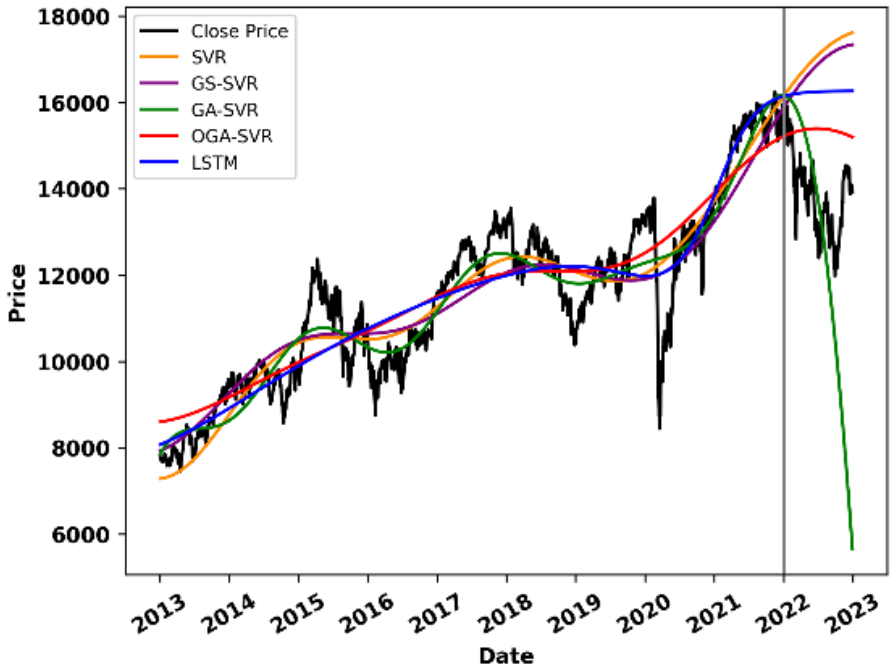

Figure 5.14 Performance of models on DAX

Table 5.6 DAX

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
SVR	721.98	3416.08	5.14	23.62
GS-SVR	774.81	3207.71	5.4	21.99
GA-SVR	656.02	3094.12	4.54	15.49
OGA-SVR	843.06	1761.95	6.02	11.7
LSTM	779.22	2554.55	5.47	17.63

5.5.4 Nikkei 225

In contrast to previous results, the best performance on SVR model is the best on testing data. Figure 5.15 shows the performance of all models.

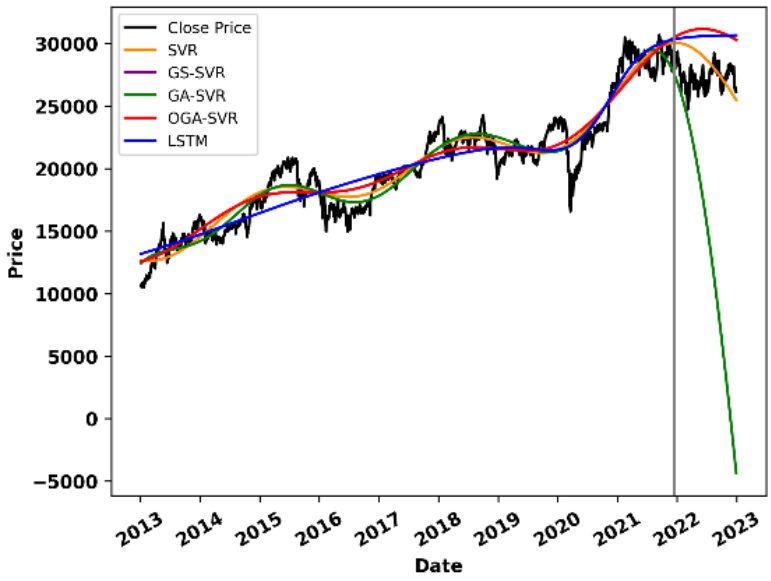

Figure 5.15 Performance of models on Nikkei 225

The RMSE is the minimum for GA-SVR on training data but the highest on test data. Hence, it is overfitting the testing data. The GS-SVR and OGA-SVR models have similar results as both models estimate the same hyperparameters. The LSTM model is the second-best on testing data. Table 8 presents the performance of all models on the Nikkei 225.

Table 5.7 Nikkei 225

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
SVR	1262.73	2042.3	5.38	6.4
GS-SVR	1379.07	3690.73	5.77	13.16
GA-SVR	1168.82	15487.49	4.91	45.11
OGA-SVR	1379.07	3690.73	5.77	13.16
LSTM	1507.74	3329.79	6.4	11.86

5.5.5 SSE

Table 9 presents the performance of all models on SSE. The GA-SVR performs worst on test data. Fig. 18 shows the fit and predicted prices of all models on SSE. The best performance exhibited by GS-SVR. The performance of SVR is close to that of GS-SVR. The performance of the optimized model is less than that of SVR, GS-SVR, and LSTM on testing data.

Table 5.8 SSE

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
SVR	311.85	419.06	7.33	12.43
GS-SVR	341.66	421.39	7.66	12.1
GA-SVR	269.61	931.85	6.67	28.1
OGA-SVR	313.06	502.3	7.53	15.08
LSTM	193.46	501.21	4.57	14.41

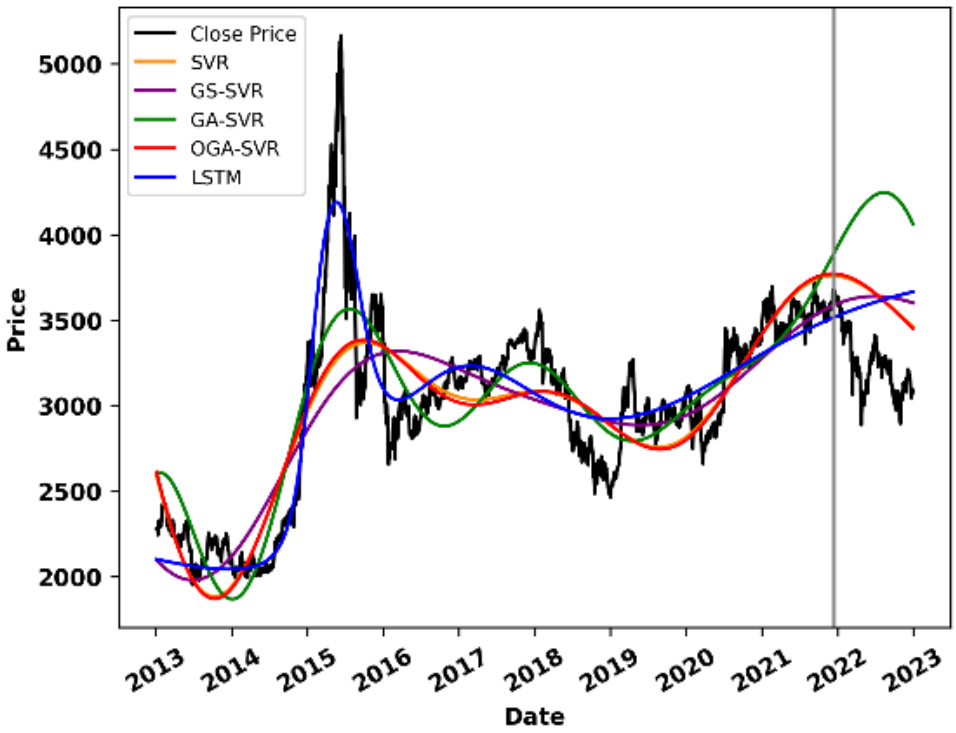

Figure 5.16 Performance of models on SSE

5.5.6 Consolidated result

To compare the performance of all models on all datasets, the study consolidated the results in Table 5.9. The GA-SVR model exhibits good performance on testing data but worst on testing data. This result implies that the genetic algorithm finds the hyperparameters that overfit the training data. The reason for overfitting can be attributed to the absence of validation on unseen data. However, there is also an absence of validation on unseen data in the SVR model, but its performance on testing data is better than GA-SVR. The MAPE of training data of GA-SVR is lesser than the MAPE of SVR. This indicates that finding the best parameter using training datasets does not guarantee good performance on testing data.

There is an increase in the RMSE value of GS-SVR compared to the RMSE values of SVR and GA-SVR. However, the performance of GS-SVR is better than GA-SVR but less than SVR on the training dataset. GS-SVR uses 10-fold cross-validation, which should increase the model's generalization ability on the testing dataset. However, the results do not comply. The reason why 10-fold cross-validation does not improve the result of GS-SVR compared to SVR can be attributed to the violation of the assumption that observations are independent and identically distributed. In stock time series data, this assumption of cross-validation does not hold true. Hence, 10-fold cross-validation does not improve the performance of GS-SVR over SVR.

Table 5.9 Consolidated

Model	Train MAPE	Test MAPE
SVR	6.03	15.60
GS-SVR	6.22	16.56
GA-SVR	5.44	23.30
OGA-SVR	6.90	10.23
LSTM	5.41	11.80

The training MAPE of the LSTM model is better compared to all other models. Hence, the LSTM network is able to fit the complex pattern of the stock market better than other models. Further, it also performs second best on unseen test data. This result clearly reflects the outstanding ability of LSTM to handle sequential data. Based on MAPE on the training datasets, the optimized model OGA-SVR does not fit the prices as well as other models. However, the performance of testing datasets is the best for OGA-SVR. This

suggests that the stock prices are noisy, and fitting the model well on the training datasets can degrade the performance of the testing datasets. The OGA-SVR models used rolling window forward validation. This helps the models compromise their performance on the training dataset to predict well on the testing dataset. Hence, in the stock market time series, a higher RMSE or MAPE value on training datasets suggests that the model is bad. Further, the results also suggest that training a model on stock time series using rolling window forward validation is a better alternative to cross-validation to predict on testing datasets.

5.5.7 Managerial Insights

The results clearly demonstrate the excellent forecasting ability of the proposed OGA-SVR approach. Therefore, this study recommends the use of the OGA-SVR model for long-term forecasting of stock market prices due to its superior performance on testing data. The study also highlights the importance of using rolling window forward validation to avoid overfitting on the training data and to improve the generalization ability of the model on testing data. Cross-validating grid search or genetic algorithms alone cannot generalize well on unseen testing data. Therefore, managers and investors should be cautious about relying on predictions solely based on grid search or genetic algorithms with SVR. This study also demonstrates that LSTM is also an appropriate algorithm for long-term stock prediction. Future research could explore the use of other models and validation techniques for stock market time series data to improve the accuracy of the predictions. The insights from this study can assist managers in making better investment decisions and understanding the limitations of the models used for stock market forecasting.

5.6 Significant Outcomes

From the findings, the following are the significant outcomes of the chapters:

- The optimized model, OGA-SVR, outperformed all compared models on testing data. The optimized model exhibited better generalization ability by avoiding overfitting in the training data.
- The generic algorithm was overfitting in the training process and hence performed poorly on testing data.
- Cross-validation did not improve the result of SVR, which suggests that observations in financial time series data are not independent.
- Rolling window forward validation is more suitable for estimating the future performance of stock time series data.
- The performance of LSTM was good on both training and testing data.

In summary, the study demonstrates that the proposed model OGA-SVR outperforms the state-of-the-art model LSTM for forecasting global stock indices. The model may be utilized by investors to make informed decisions.

The work in this chapter is based on the following publication:

Beniwal, M., Singh, A., & Kumar, N. (2023). Forecasting long-term stock prices of global indices: A forward-validating Genetic Algorithm optimization approach for Support Vector Regression**. Applied Soft Computing, 110566.**

6 Deep Learning Models for Long-term Price Forecasting

6.1 Overview

Deep machine learning algorithms play an important role in facilitating the development of predictive models for the stock market. However, most studies focus on predicting next-day stock prices using deep learning, limiting the usability of the predictive model for investors. There is a gap in the literature for studies that predict daily prices for up to a year. Deep learning is a subset of neural networks that can automatically extract features from raw data without the need for manual feature engineering.

This study extensively explores the ability of deep learning models to predict out-of-sample the daily prices of global stock indices over the long term, up to a year. The performance of six models, including Deep Neural Network (DNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (Bi-LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN), is compared using RMSE and MAPE. The models predict the long-term daily prices of five global stock indices, namely the Nifty, the Dow Jones Industrial Average (DJIA), the DAX performance index (DAX), the Nikkei 225 (NI225), and the Shanghai Stock Exchange composite index (SSE).

The results confirm the superiority of LSTM for predicting long-term daily prices. The Bi-LSTM does not improve the result of the LSTM but performs better than other algorithms. CNN overfits the training data and poorly forecasts the long-term stock prices of global indices based on the testing data. The study can be further applied to different frequencies of data, such as 1-minute, 5-minute, 15-minute, etc.

6.2 Background

Various machine learning algorithms have been used in stock market analysis, such as Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), and Naive Bayes (NB). These algorithms have demonstrated varying degrees of success in predicting stock prices. The accuracy of predictions is subject to a margin of error, which is influenced by the choice of algorithm employed (Nikou et al., 2019b). However, neural networks have emerged as robust and effective machine learning algorithms that can efficiently handle noisy and nonlinear data to forecast time series (Yu & Yan, 2020). Among neural networks, deep learning has become a popular approach for analyzing stock market data due to its superior performance in prediction tasks. Deep learning enables the creation of computational models that consist of multiple layers of processing, which can learn to represent data with varying degrees of abstraction (Lecun et al., 2015). Currently, there is a growing trend among asset management companies and investment banks to allocate more research funds toward the development of artificial intelligence techniques, particularly in the field of deep learning (Jiang, 2021a). Deep learning models have shown better performance compared to linear and other machine learning models in stock market prediction tasks, owing to their ability to effectively handle large volumes of data and identify complex nonlinear relationships between input features and prediction targets (Jiang, 2021b).

One advantage of deep learning is its ability to automatically extract features from raw data (Liang et al., 2017; D. Wu et al., 2022), which eliminates the need for feature engineering and improves the accuracy of forecasting. Deep learning models are composed

of multiple layers of interconnected neurons, with each layer responsible for extracting higher-level features from the input data. Numerous deep-learning architectures have been created to address diverse problems and the inherent structure of datasets (Bhandari et al., 2022). In this study, some frequently used deep learning models such as Deep Neural Networks (DNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (Bi-LSTM), Gated Recurrent Units (GRU), and Convolutional Neural Networks (CNN) are explored.

DNN includes dense hidden layers with a hierarchical topology (Thakkar & Chaudhari, 2021). DNNs can learn multiple levels of features from raw input data using multiple layers of nonlinear transformations. An RNN is a specialized type of Artificial Neural Network (ANN) that can handle sequential inputs through the incorporation of internal feedback connections between neurons (K. Kumar & Haider, 2021b). LSTM, a specific type of RNN, is a neural network architecture that possesses the ability to retain memory, making it well-suited for processing and predicting significant events that have longer intervals and time delays within time series data (Lin et al., 2021). However, LSTM can only learn from past information (Alkhatib et al., 2022). Bi-LSTM, a variant of LSTM, can learn from past as well as future information because it has two hidden layers with opposite directions connected with the same output (Alkhatib et al., 2022; Houssein et al., 2022; Qin et al., 2018). Similar to LSTM, GRU can handle sequence data while simplifying the complex computations involved in LSTM (S. Zhang et al., 2023). While CNN is inspired by computer vision, it can also be designed for financial data (Hoseinzade & Haratizadeh, 2019b; Shah et al., 2022).

In the field of stock market prediction, most of the research has focused on predicting the next day's price (Rouf et al., 2021), using iterative methods to predict prices for the entire test data. This approach has occasionally achieved high accuracy but is not always useful for traders seeking long-term predictions. For long-term prediction, machine learning algorithms need to provide multi-output predictions. However, structuring machine learning algorithms for multi-output long-term predictions can be a tedious task and sometimes impractical. To address this limitation, this study proposes an approach for stock market prediction that leverages the time dependency of stock prices. Instead of predicting only the next day's price, machine learning algorithms are trained to learn the patterns of price fluctuations in relation to time. By exploiting the time dependency of prices, the study aims to predict long-term prices with higher accuracy and precision. To test the robustness of the approach, the study experiments with the ML algorithms on stock indices of the top five economies in terms of Gross Domestic Product (GDP) (*Countries by GDP*, 2022), namely the Nifty, the Dow Jones Industrial Average (DJIA), the DAX performance index (DAX), the Nikkei 225 (NI225), and the Shanghai Stock Exchange composite index (SSE).

The proposed approach contributes to the field of predicting long-term stock prices using deep learning. Further, it can provide traders and investors with more accurate and reliable long-term predictions. This can help them make informed investment decisions and reduce risk. Additionally, the proposed approach can be used to develop advanced trading strategies, such as trend-following or mean-reverting. Overall, this study contributes to the ongoing research efforts aimed at improving the accuracy and reliability of stock market

predictions using advanced deep learning algorithms. Hence, this study contributes in the following ways:

- Most predictive studies in the stock market emphasize next-day prices. In contrast, this study presents an approach to exploit price patterns in relation to time dependency to forecast long-term stock prices of global indices.
- Extensive experiments are conducted on the top five GDPs to evaluate the predicting robustness of LSTM and other deep learning algorithms in the long term.
- The research addresses multiple questions regarding the ability to forecast long-term stock prices of global indices, such as whether RNN performs better than DNN, whether it is better to use Bi-LSTM instead of LSTM, whether GRU outperforms LSTM, and which algorithm from DNN, RNN, LSTM, Bi-LSTM, GRU, and CNN is the least suitable.
- The approach helps investors gauge the market outlook for the long term and make informed decisions.
- Furthermore, the study may inspire other researchers to develop trading and risk management systems for the long-term horizon.

6.3 Deep Learning Models

6.3.1 Deep Neural Network (DNN)

ANNs were proposed in the 1940s as the simplest model to mimic the way human brains work in processing information and learning from it. However, ANNs' learning becomes challenging if the data increases in size (Awad & Khanna, 2015a). Multi-layer Perceptron (MLP) is a variant of feedforward ANN and is the foundation of DNN (Sarker,

2021b). A fully connected MLP having more than or equal to two hidden layers is referred to as a DNN in this study. DNNs are highly scalable and can handle large and complex datasets with ease. Hinton et al. (2006) first proposed DNNs. In a DNN, there are three distinct layer types, starting with the input layer, followed by two or more hidden layers, and concluding with an output layer. The input layer receives and processes the input data, which is then transformed through the subsequent hidden layers via activation functions. Each hidden layer comprises multiple neurons, and the output of each neuron is fully connected to all neurons in the next layer, creating a dense layer. Lastly, the output layer produces the network's final output, which can either be a single value or a vector of values. Figure 6.1 shows the architecture of a typical DNN.

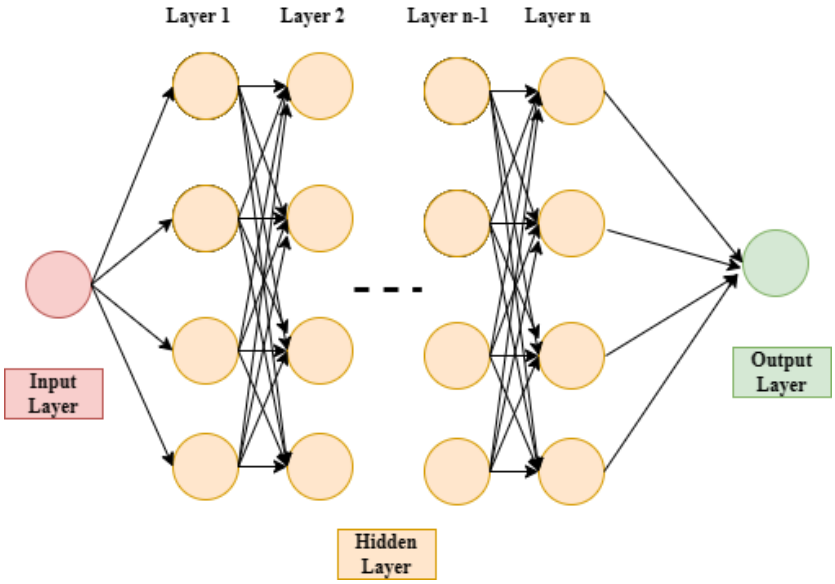

Figure 6.1 DNN Architecture

6.3.2 Recurrent Neural Network (RNN)

DNNs are feedforward neural networks where information only flows in one direction, from the input layer through the hidden layers to the output layer. Feedforward neural networks encounter several difficulties when processing temporally related data, such as time series, including managing varying-length sequences (Tsantekidis et al., 2022). In contrast, RNNs have a feedback mechanism where information can flow back into the network. Hence, RNNs can take in a sequence of data and use it to build a kind of memory that helps them understand the current and past data in that sequence. Elman (1990) first introduced simple RNN (J. Zhang & Man, 1998). In this paper, SimpleRNN from TensorFlow is used, which is fully connected. A fully connected RNN layer means that each neuron in the layer is connected to every neuron in the next layer. Figure 6.2 shows the architecture of a simple RNN.

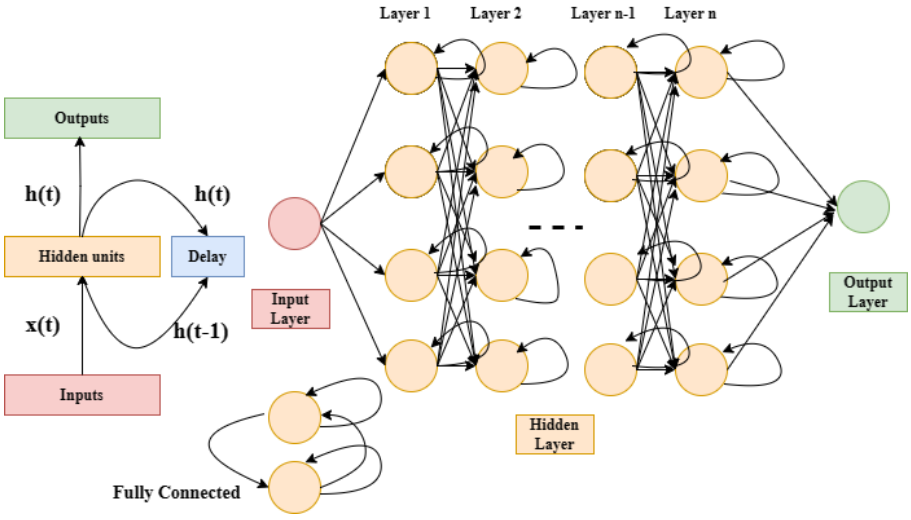

Figure 6.2 RNN Architecture

6.3.3 Bi-directional Long Short-Term Memory (Bi-LSTM)

LSTM was discussed earlier. Bidirectional RNN (Bi-RNN) was proposed by Schuster and Paliwal in 1997 (Schuster & Paliwal, 1997). Later, Graves and Schmidhuber (2005) proposed Bi-LSTM. The architecture of Bi-LSTM is similar to LSTM, but LSTMs utilize past information to make predictions. However, LSTMs do not incorporate future information in their predictions, which could limit their ability to capture certain patterns in the data. Bi-LSTM can obtain not only past context information but also future context information (Y. Wang et al., 2019). LSTMs utilize past information in order to make predictions. However, LSTMs do not consider future information in their predictions, which may limit their ability to capture certain patterns in the data. Figure 6.3 shows the architecture of Bi-LSTM.

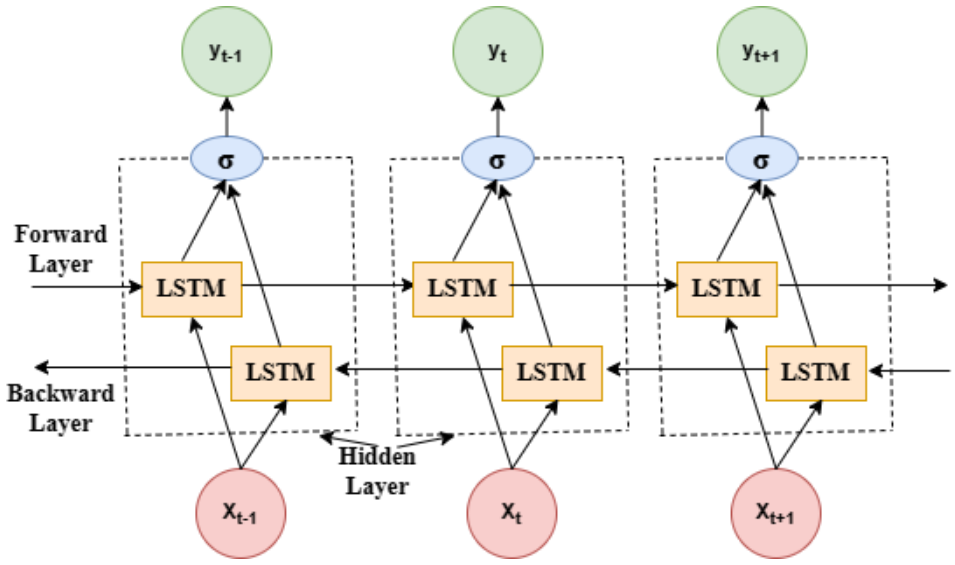

Figure 6.3 Bi-LSTM Architecture

6.3.4 Gated Recurrent Unit (GRU)

The GRU was proposed by Kyunghyun Cho (2014). A GRU is a subtype of RNN that shares some similarities with LSTM. However, GRU is characterized by the presence of only two gates, namely a reset gate and an update gate. Like LSTM, GRU also largely tackles the vanishing gradient problem (Shen et al., 2018). Further, due to its simpler structure, GRU trains faster (Samsani et al., 2022) and is computationally more efficient (Xia et al., 2022) than LSTM. Figure 6.4 shows the architecture of the GRU unit.

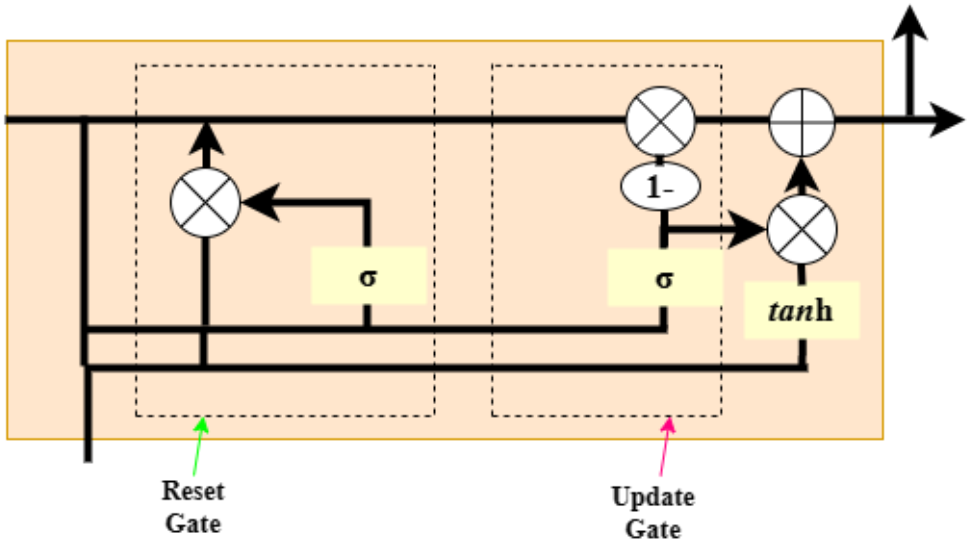

Figure 6.4 GRU Unit

There are two gates in the GRU, namely the reset gate and the update gate. The reset gate plays a crucial role in determining the extent to which the previous hidden state should be discarded and the new input should be included in the current hidden state. The update gate, on the other hand, determines the extent to which the previous hidden state should be retained and how much of the new input should be added to the hidden state.

6.3.5 Convolutional Neural Network (CNN)

CNN in its primitive form was first introduced by Yann LeCun (Bhatt et al., 2021; LeCun et al., 1998) inspired by the work of Kunihiko Fukushima (Fukushima, 1980). However, the focus on CNN increased after the introduction of a deep convolutional neural network also known as Alexnet (Hinton et al., 2012; Krizhevsky et al., 2017).

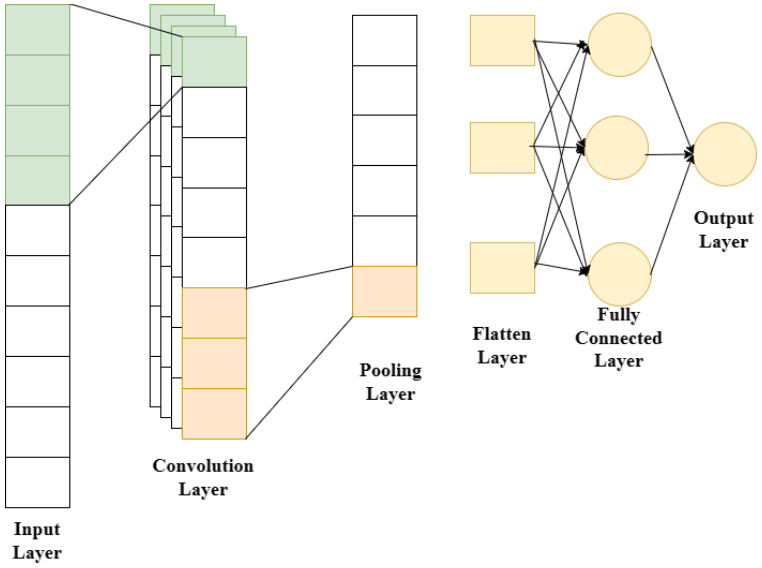

Figure 6.5 1D-CNN Architecture

Figure 6.5 shows the architecture of a one-dimensional CNN. In 1D-CNN, the convolutional kernel is an array of one dimension. As shown in the figure, a typical CNN architecture comprises several layers, including the input layer, convolutional layer, pooling layer, fully connected layer, and output layer. Although CNN is used for image recognition, it can also process time series data (H. Wang et al., 2021).

6.4 Proposed Methodology

6.4.1 Prediction method

In this study, all deep learning models have the same configuration, so the comparison can be unbiased. Figure 6.6 shows the architecture of the deep learning models. The first layer of each model is the input layer. The dates are converted into an integer from 1 to n. Along with integer dates, this layer also has an input of a transformed array of scaled close prices using the min-max scaler formula in Eq. (6.1)

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{6.1}$$

Assuming X is the feature matrix, X_{min} and X_{max} correspond to its minimum and maximum values, respectively. The input layer gives output to the model's first layer. The first layer of the model is the corresponding fully connected neuron cell, such as DNN, RNN, LSTM, Bi-LSTM, GRU, or CNN. There are 256 fully connected neuron units. This layer has a ReLU activation function. Deep learning models can sometimes can overfit the training data. Hence, it is important to design a prediction method that tackles the problem of overfitting the training data. To handle the overfitting in training data, a dropout layer with 256 units is added after the first layer. The dropout rate is kept at 20% in the dropout layer for all models.

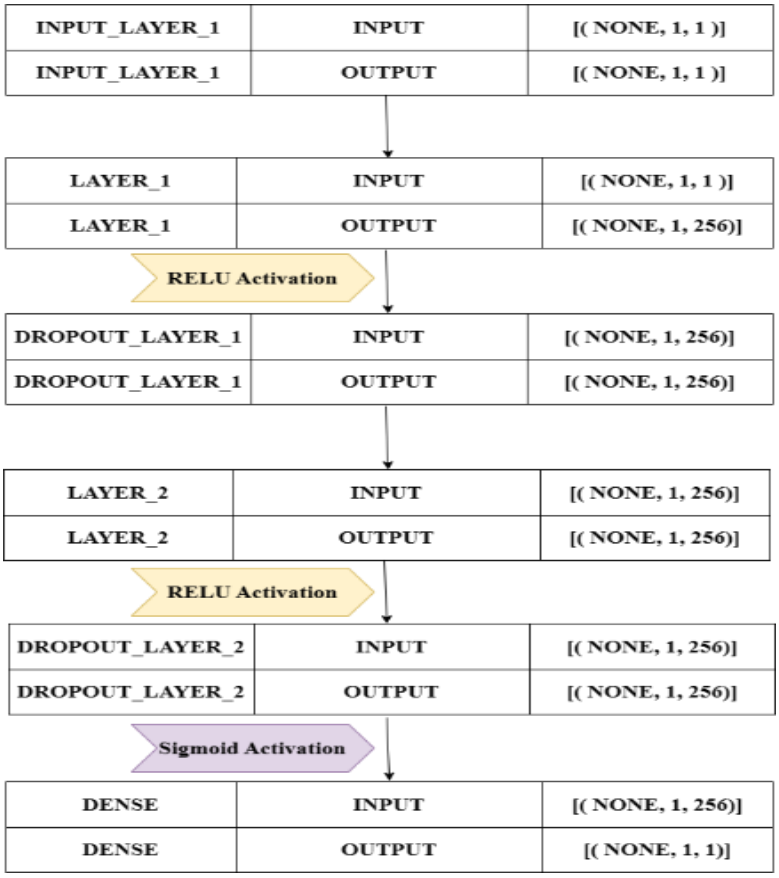

Figure 6.6 Deep learning model Layers architecture

To make the learning deeper, a second layer of deep learning units is added. The second layer also has 256 units. This layer also has a ReLU activation function. A second dropout layer is again stacked on top of the deep learning layer with the same 20% dropout rate and 256 units. Finally, a dense layer with a single neuron is added with a sigmoid activation function. The data is input into the models with a batch size of 32. The models are compiled and trained using the Adam optimizer and mean squared error as a loss function. For all models, 100 epochs are used to train them. The output from the model is inversely

transformed and compared with the original prices. The final evaluation of the models is done based on RMSE and MAPE.

6.4.2 Experimental Setup

In this study, an investigation is conducted using five prominent global indices, which are the Nifty, the Dow Jones Industrial Average (DJIA), the DAX performance index (DAX), the Nikkei 225 (NI225), and the Shanghai Stock Exchange composite index (SSE). These indices belong to India, the USA, Germany, Japan, and China, respectively. The countries are the top five economies in the world. As the stock market is reflected in the economy of a country, these indices from diverse economies can help test the robustness of the deep learning model. The data is collected from Yahoo Finance for a span of around ten years, from January 1st, 2013, to February 28th, 2023. Such a long period helps the model train on different phases of the market, such as bull, bear, and stagnant. The last year's data is kept for testing purposes, while the rest of the data is utilized for training. Upon completion of training, the trained models make predictions on future prices using dates from the testing data. The prices obtained from the testing phase are inverse-transformed and compared to the predicted values. The efficacy of the models is evaluated using RMSE on both training and testing data. MAPE is used to compare the results obtained from different indices because the value of indices varies from market to market. Figure 6.7 shows the overall experimental design.

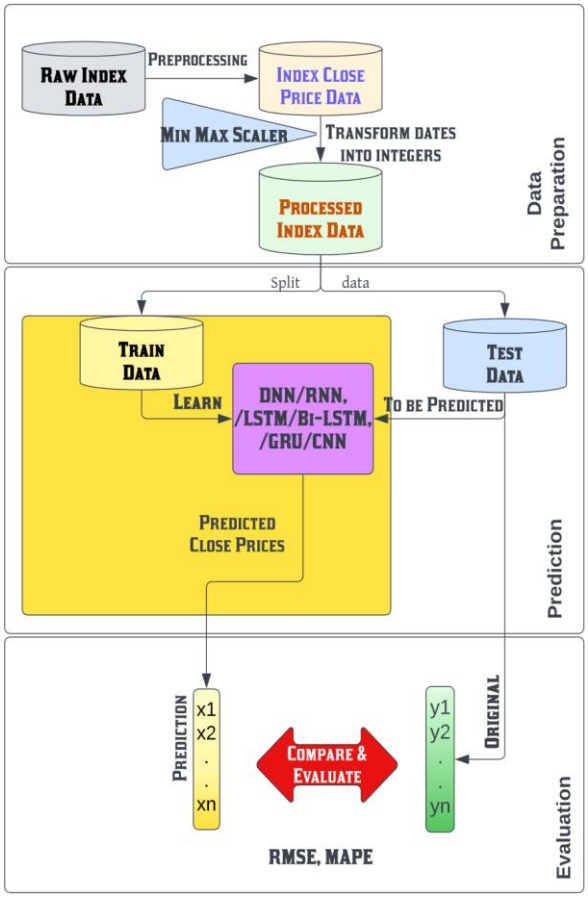

Figure 6.7 Prediction Framework

The process has three stages, namely data processing, prediction, and evaluation. Data processing involves learning and encoding data. In the prediction stage, deep learning models are used to forecast future prices. Lastly, in the evaluation state, models are evaluated using RMSE and MAPE.

6.5 Findings

6.5.1 Nifty

Table 6.1 shows the results of all models on the Nifty index. In the training phase, CNN performed the best and LSTM performed the worst. Further, GRU is the second-best in terms of RMSE and MAPE. However, in the testing phase, LSTM performed the best, and CNN performed the second worst. This indicates that fitting well in the training phase does not guarantee better performance in the testing phase. Figure 6.8 shows visually the price pattern and prediction in the training and testing phases. The gray line separates the training and testing periods.

Table 6.1 Nifty

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
DNN	503.34	1255.32	3.52	6.2
RNN	510.74	1353.29	3.56	6.8
LSTM	581.37	881.38	4.51	3.95
Bi-LSTM	576.12	1064.63	4.46	4.96
GRU	506.26	1224.16	3.6	5.98
CNN	499.28	1326.14	3.49	6.64

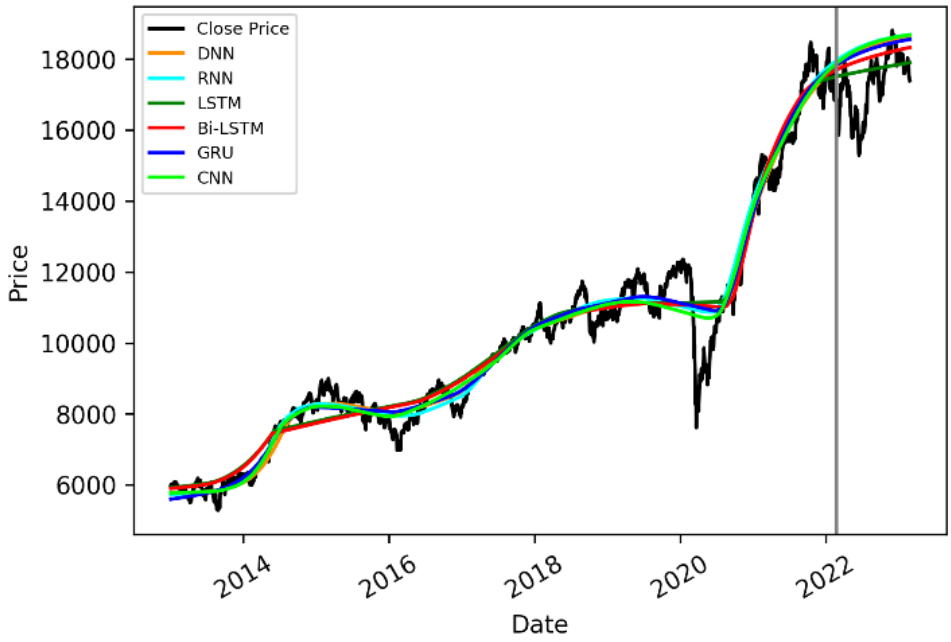

Figure 6.8 Nifty predictions

6.5.2 DJIA

Table 6.2 shows the results of all models on the DJIA index. The performance of CNN in the training phase is the best in the DJIA index, similar to the performance of CNN on Nifty. DNN is the second-best in terms of training RMSE and MAPE. GRU performed worst in the training period, and LSTM is the second worst. However, the scenario changes in the testing phase. The LSTM model again performed the best, while the CNN model performed the worst. Figure 6.9 shows the performance of deep learning models during the training and testing periods. The MAPE values are higher in the DJIA compared to the Nifty index.

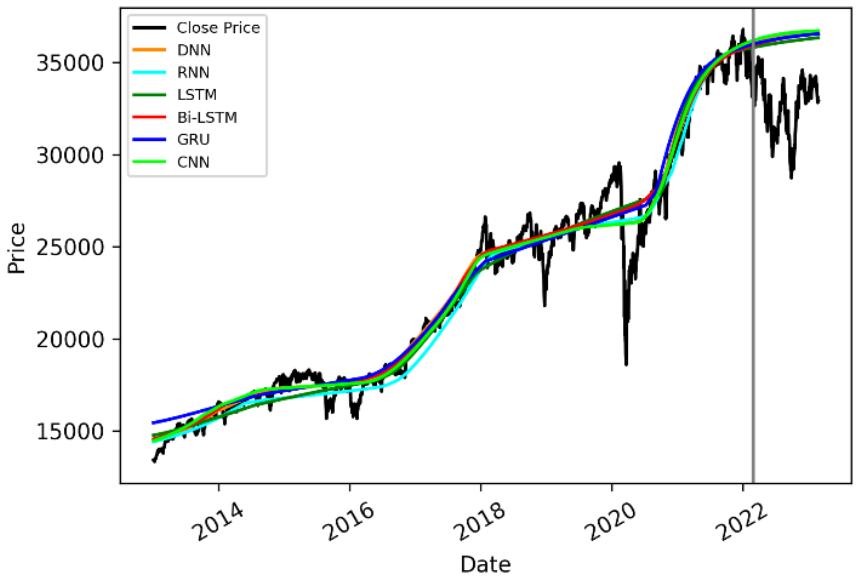

Figure 6.9 DJIA predictions

Table 6.2 DJIA

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
DNN	957.45	4027.82	2.76	11.67
RNN	1011.53	3953.16	3.26	11.42
LSTM	1041.41	3736.37	3.1	10.71
Bi-LSTM	991.63	3928.81	2.82	11.34
GRU	1085.22	3942.12	3.43	11.39
CNN	937.4	4145.69	2.69	12.05

6.5.3 DAX

Table 6.3 shows the performance of deep learning models on the DAX performance index. In contrast to the performance of CNN on the Nifty and DJIA, CNN's performance is not the best during the training phase. The DNN model performed slightly better than the CNN model. Following a similar pattern in the previous two indices, the LSTM model performed the worst in terms of the training RMSE and training MAPE. However, the LSTM is the best-performing model during testing in DAX as well. The MAPE values are higher in DAX compared to the Nifty and DJIA. Figure 6.10 visualizes the pattern of prices both during the training and testing phases.

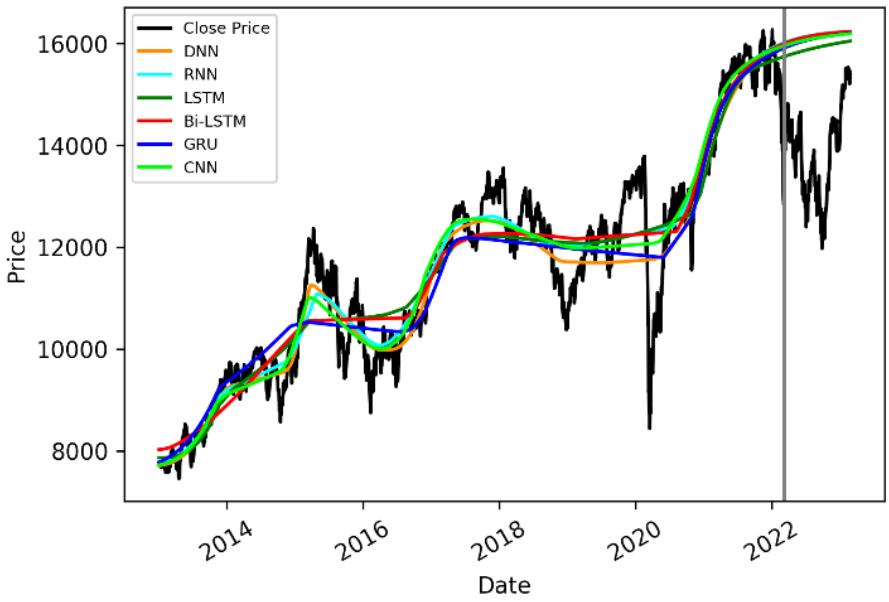

Figure 6.10 DAX predictions

Table 6.3 DAX

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
DNN	596.34	2453.73	3.71	17.06
RNN	601.07	2479.58	3.67	17.26
LSTM	684.68	2246.89	4.53	15.48
Bi-LSTM	681.42	2473.32	4.52	17.21
GRU	679.04	2414.38	4.45	16.76
CNN	598.2	2421.44	3.74	16.82

6.5.4 Nikkei 225

Table 6.4 reports the results of the six models on the Nikkei 225. The results are similar to previous experiments. The CNN model shows superiority during the training phase, and the GRU model outperforms other models during the testing period. The MAPE values are smaller compared to the DJIA and DAX but bigger than the Nifty. Figure 6.11 shows the chart of the original close price and fit and prediction prices during the testing and training periods, respectively.

Table 6.4 Nikkei 225

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
DNN	1016.41	3013.85	3.91	10.8
RNN	996.18	2932.33	3.89	10.49
LSTM	1421.02	3080.97	5.88	11.06
Bi-LSTM	1259.34	2908.46	5.14	10.4
GRU	1058.77	2610.49	4.23	9.25
CNN	990.39	2989.09	3.8	10.71

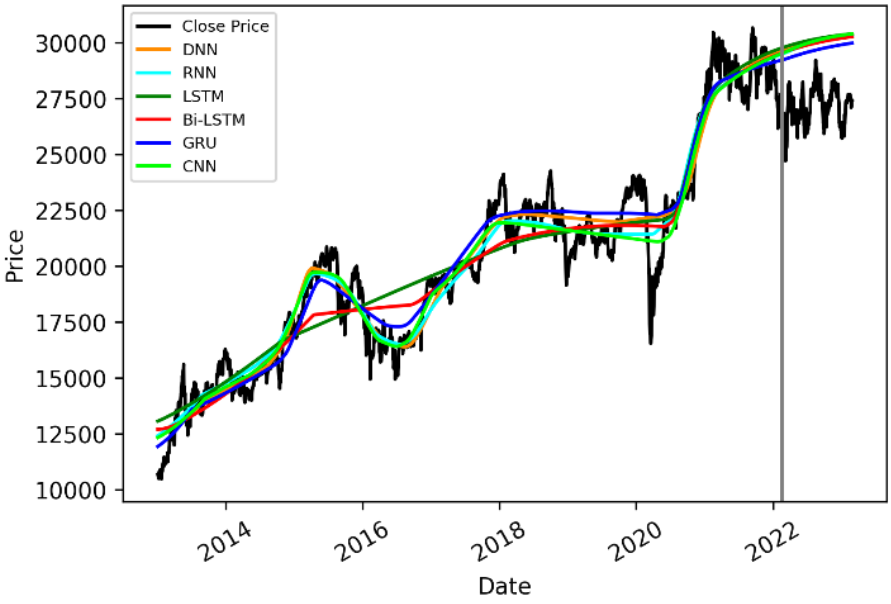

Figure 6.11 NI225 predictions

6.5.5 SSE

Table 6.5 shows the performance of the models on the SSE index. RNN fits the best on the training data of the SSE index, and LSTM fits the worst. The Bi-LSTM models performed the best on testing data. The LSTM model is in second place. The CNN models again performed poorly on the testing data. Figure 6.12 visualizes the results of all models. It can be seen that the CNN predicted prices are farthest from testing close prices.

Table 6.5 SSE

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE
DNN	188.75	324.18	4.8	9.5
RNN	183.04	452.4	4.49	13.7
LSTM	234.99	357.77	5.8	10.68
Bi-LSTM	198.46	354.77	4.89	10.57
GRU	184.49	395.5	4.49	11.89
CNN	189.66	546.76	4.71	16.77

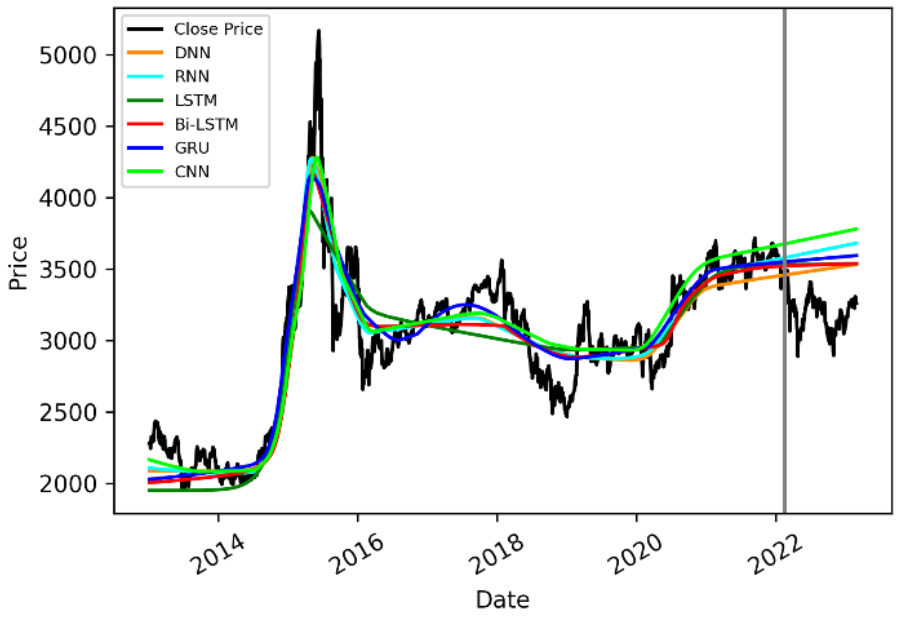

Figure 6.12 NI225 predictions

6.5.6 Consolidated

In all models, the CNN model tends to overfit the training prices and underfit the testing prices. In contrast, the LSTM model tends to underfit the training price but performs the best on the testing data. Table 6.6 shows the consolidated result. To compare the models between the five indices, RMSE cannot be used as the scale of close prices is different for each index. Hence, MAPE is appropriate for comparison. Except for the LSTM, the Bi-

LSTM is superior to all other models. This can be associated with the fact that LSTM and Bi-LSTM have a lot of similarities in architecture. The performance of GRU and DNN is the same on testing data. However, DNN fits better on the training data compared to the GRU. GRU also has a gated mechanism like LSTM and Bi-LSTM but it is not able to outperform these models to predict long-term prices.

Table 6.6 Consolidated

Model	Train MAPE	Test MAPE
DNN	3.74	11.05
RNN	3.77	11.93
LSTM	4.76	10.38
Bi-LSTM	4.37	10.90
GRU	4.04	11.05
CNN	3.69	12.60

Table 6.7 visualizes the best and worst performers during the training and testing phases. RNN is the second-worst performing, which might be due to a vanishing gradient problem.

Table 6.7 Overview of Models' Results

	Training		Testing	
	The Best	**The Worst**	**The Best**	**The Worst**
Nifty	CNN	LSTM	LSTM	RNN
DJIA	CNN	GRU	LSTM	CNN
DAX	CNN	LSTM	LSTM	RNN
NI225	CNN	LSTM	GRU	LSTM
SSE	RNN	LSTM	Bi-LSTM	CNN

CNN performed the poorest compared to other models, and this might be because CNN is designed to work with computer vision and images. It can be used for time series

analysis, but with a similarly deep learning architecture, it does not well predict long-term prices.

6.5.7 Managerial Insights

Based on the analysis, the study found that the LSTM model is superior to other models in predicting long-term stock prices for the five indices considered. The LSTM model may perform poorly on training data compared to other models, but it performed well on testing data, indicating that it is an appropriate model to predict long-term stock prices of global indices. However, the CNN model tends to overfit the training data and underfit the testing data, making it unsuitable for predicting long-term stock prices. Furthermore, the study found that the performance of the models varies across the different indices, indicating that the stock market trends in different regions are not identical. Additionally, the study highlights the importance of selecting the appropriate evaluation metrics when comparing the performance of the models. In this case, MAPE is used because it accounts for the difference in scale among the indices. Overall, the findings of this study could be useful for investors and traders who rely on stock price predictions to make informed investment and risk mitigation decisions. The LSTM model, in particular, could be a valuable tool for predicting long-term stock prices, providing a competitive edge in the stock market.

6.6 Significant Outcomes

- In forecasting long-term prices using deep learning models, the LSTM performed the best on Nifty, DJIA, and DAX.
- Out of all the deep learning models used, the Bi-LSTM also performed better than other models except the LSTM.
- 1-D CNN overfits the training data, performs poorly on testing data, and is hence least appropriate to predict long-term stock prices. RNN also performed poorly on testing data.
- Based on the training and testing of MAPE, GRU's forecasting ability for long-term prices of global indices is inferior to LSTM, Bi-LSTM, and DNN.

In summary, among the compared deep learning models, the LSTM model is the most appropriate for forecasting long-term prices.

This chapter is based on the following paper:

Beniwal, M., Singh, A., & Kumar, N. "Forecasting Multistep Daily Stock Prices for Long-Term Investment Decisions: A study of Deep Learning models on Global Indices." (Communicated)

7 Conclusion and Future Work

This chapter serves as a conclusion to the study, discussing its limitations and providing insights into future directions. Furthermore, it suggests the potential applications of the research findings.

7.1 Conclusions

7.1.1 Trading Framework using LSTM and Technical Analysis

In this study, a model based on LSTM and technical indicators was developed. The empirical study conducts a series of experiments on standard models such as LSTM and LSTM-o and some created models such as *fma*-LSTM-o, *atr*-LSTM-o, and *a*-*m*-LSTM-o. The study used the model's parameters as round numbers that traders generally use to avoid underfitting and overfitting.

Confusion matrix components such as accuracy, weighted prediction, weighted recall, and weighted F1-score are used as evaluation parameters. Apart from these, prediction return, prediction drawdown, and the number of transactions are also used to evaluate the models. From the results of the experiments, it can be concluded that a high value of weighted precision, weighted recall, or weighted F1 score does not make a model better. It cannot be assumed that a model with a high F1score will yield higher returns. Some models with high F1-score give poor returns. The study emphasizes that the prediction return, prediction drawdown, and number of transactions are more practical criteria for evaluation.

The study innovatively and intuitively uses technical indicators to transform prices. Some studies improve the accuracy of classification but do not compare returns. Further, lots

of studies lack a buy-and-hold model as a baseline (Basak et al., 2019; Borovkova & Tsiamas, 2019; Kim et al., 2006). However, the proposed model improves returns over the baseline buy-and-hold return. It provides an alternative for investors to think beyond the passive buy-and-hold strategy.

7.1.2 Performance Comparison of ARIMA and SVR

This work analyzes and evaluates the performance of static and iterative ARIMA and SVR models. Further, a comparison is made with the baseline static and iterative Naïve models. The data of five global indices, namely, Nifty, DJIA, DAX, NI225, and SSE, are split for training and testing in a 75:25 ratio for both static ARIMA and SVR models. In the static models, a long-term prediction of price is made at once. For all the test days, the naïve static model assumes that the price remains the same as on the last day of the training day. The models are evaluated using RMSE, MAPE, MAE, and MSE. Based on these parameters, the SVR model performs the best for the Nifty and SSE indices in the static models. Both of these indices belong to emerging economies. In the DAX index, ARIMA static and SVR static both performed equally. The ARIMA static model outperforms SVR static in the DJIA and NI225 indices. Notably, ARIMA performed better than SVR static in developed economies. From the result, it could be inferred that SVR static has better predictability power in emerging economies than in developed economies. Additionally, ARIMA predicted that prices do not fluctuate in the long term. It is not giving any future directional information to the investors. On the other hand, SVR static is able to capture past patterns in all indices and provide directional information to investors. Hence, it can be concluded that for long-term prediction, SVR does have predictability.

In iterative models of ARIMA and SVR, the models are first trained on 75% of the data. Once trained, the models predict only the next day's price. Before predicting the upcoming day, the iterative models are retrained with the original price of the predicted day. This process makes the models dynamic and adjustable to changes in the behavior of the time series. The Naïve iterative model assumes the next day's price is the same as the previous day's price. ARIMA iterative and SVR iterative are compared to the Naïve model based on RMSE, MAPE, MAE, and MSE. This study shows that the Naïve model performed better than both models, i.e., the ARIMA and SVR iterative models. This result indicates that the Naïve model is best for prediction in next-day price prediction. ARIMA iterative is similar to the Naïve model. Though SVR performs the worst based on evaluation parameters, its prices still reflect the underlying trend. The best performance by the naïve model shows that in the short term, the prediction abilities of ARIMA and SVR are close to random forecasting.

This study empirically demonstrates that the SVR static model has better predictive power than other models over the long term. Additionally, the SVR models predict the price trend and eliminate noise. The experiments also show that the SVR has better predictability in emerging economies such as India and China. Further, in developed countries, the predicted price pattern of the SVR model reflects similarly to the test data, which can be useful to investors. ARIMA and Naïve model fails to add that futuristic knowledge to investors. In iterative models, the SVR and ARIMA fail to beat the Naïve models. Hence, market efficiency makes it difficult to predict the price for the next day by learning the

underlying pattern. Using this study, investors can be more confident using SVR static to predict the long-term pattern of prices in developing countries like India and China.

7.1.3 Long-term Price Forecasting using Optimized GA and SVR

This study proposed a model based on forward-validating genetic algorithm and support vector regression to forecast the multistep long-term stock prices. While using daily data for prediction, the study avoided using statistical calculations and technical indicators based on historical data. This study only used dates, and close prices are input and output features, respectively. As dates were known input variables for the long-term duration, the models predicted daily close prices for up to a year.

After empirical evaluation using RMSE and MAPE, the study found that the optimized model OGA-SVR outperformed all baseline models on testing data. The optimized model exhibited better generalization ability by avoiding overfitting in the training data. The model used rolling window forward validation, which helped it perform well on unseen future data. The results suggest that only genetic algorithm based SVR model was overfitting in the training process and hence performed poorly on testing data. The study also found that cross-validation did not improve the result of SVR, as observations in time series data are not independent. Therefore, rolling window forward validation is more suitable for estimating the future performance of stock time series data.

After OGA-SVR, LSTM has the best performance among other models. The performance of LSTM is good on both training and testing data, which affirms its outstanding ability to handle sequential data and suitability for stock time series data.

Although the LSTM model showed good performance on training and testing data with two dropout layers, further optimization of hyperparameters such as the number of neurons, batch size, and number of epochs can improve performance.

7.1.4 Deep Learning Models for Long-term Price Forecasting

The study extensively explores the ability of deep learning models to predict the daily prices of global stock indices over the long term, up to a year. The performance of six models, including DNN, RNN, LSTM, Bi-LSTM, GRU, and CNN, is compared using RMSE and MAPE. The models predict the long-term daily prices of five global stock indices, namely the Nifty, DJIA, DAX, NI225, and SSE.

The results confirm the superiority of LSTM for predicting long-term daily prices. The LSTM performed the best on the Nifty, DJIA, and DAX. Overall, among the compared models, the MAPE on all indices demonstrates that LSTM is most appropriate to predict long-term stock prices. The Bi-LSTM also performed better than other models except for the LSTM. This suggests that processing input by Bi-LSTM in reverse order does not improve stock prediction over LSTM. CNN overfits the training data and performs poorly on the testing data. Therefore, among the compared models, it is least appropriate to predict long-term stock prices. Similarly, RNN also performed poorly on testing data. According to the literature, GRU is computationally efficient, but its forecasting ability is inferior to that of LSTM, Bi-LSTM, and DNN, as indicated by the results.

7.2 Limitations

In the study of LSTM and technical analysis models, the parameters are chosen manually rather than algorithmically. A dynamic parameter assignment by the models will make the modes more robust. The researchers may further compare the models with other neural networks such as CNN, DNN, and RNN.

In the comparative study of the ARIMA and SVR, only close prices and dates are used as input to the models. Researchers can include fundamental analysis, technical analysis, sentiment analysis, or evolutionary algorithms to improve the forecasting capabilities of the machine learning algorithm. Further, this study does not generate buy or sell signals. Researchers can further develop a trading or decision system to help investors in their decision-making.

In forecasting long-term prices using GA and SVR, it only tested the models on global indices, which have a tendency to go up and are comparatively less volatile than individual stock prices. It will be interesting to explore whether the approach produces similar results for individual stock predictions. Particularly, the use of only a single input variable limits the models' performance. The performance can be improved by feeding prices and predicting them from sampling frequencies such as monthly, quarterly, and yearly.

In the study of forecasting long-term prices using deep learning models, the architecture of the models is selected manually. Hence, using automated Machine Learning (AutoML) might improve results by optimizing the hyperparameters. Apart from dropout, other regularization techniques such as forward validation and moving window training may enhance the results.

7.3 Future Work

- The methods and models utilized in this study can be further explored to develop trading and risk management systems. These systems can assist investors, traders, and analysts in making better-informed decisions.
- The study focused on daily data frequency; however, it can be applied to small frequency data such as 1 min, 5 min, 15 min, 1 hour, etc. Hence, it will be interesting to evaluate the model efficiently in shorter timeframes.
- The study can be utilized to further develop a dynamic parameter assignment mechanism for LSTM and technical analysis models to enhance their robustness. This can be achieved by determining the optimal parameters algorithmically based on the characteristics of the dataset.
- Expand the input features by incorporating fundamental or sentiment analysis. This can potentially improve the forecasting capabilities of machine learning algorithms.
- Explore the use of automated Machine Learning (AutoML) techniques to optimize the architecture and hyperparameters of deep learning models. This can lead to improved performance by automatically fine-tuning the model configuration.

- Some of the machine learning algorithms are not evaluated in the study, such as random forests, XGboost, decision trees, etc. It will be interesting to explore those algorithms and compare the proposed models' performance.

7.4 Applications

- A trading system based on the long-term prediction of daily prices can be built to enhance the decision-making of traders and investors.
- Further, using the long-term stock forecasting model of this study, stock portfolios can be optimized.
- The long-term prediction models, especially OGA-SVR, can be effectively applied to other time series forecasting tasks, such as sales forecasting, inventory forecasting, electricity demand forecasting, and more.
- Based on the models' predictions, investors can develop systems to take positions in the derivatives market to hedge their risks or enhance their returns.

Journal Publications

Journal Publications

S. No.	Paper	Index	Status
1	Beniwal, M., Singh, A., & Kumar, N. "Alternative to Buy-and-Hold: Predicting Indices Direction and Improving Returns Using a Novel Hybrid LSTM Model." **International Journal on Artificial Intelligence Tools (2023).**	SCIE (Impact Factor: 1.1)	Published
2	Beniwal, Mohit, Archana Singh, and Nand Kumar. "A comparative study of static and iterative models of ARIMA and SVR to predict stock indices prices in developed and emerging economies." **International Journal of Applied Management Science** 15.4 (2023): 352-371.	Scopus and ESCI (Impact Factor: 0.7)	Published
3	Beniwal, M., Singh, A., & Kumar, N. (2023). Forecasting long-term stock prices of global indices: A forward-validating Genetic Algorithm optimization approach for Support Vector Regression. **Applied Soft Computing**, 110566.	SCIE (Impact Factor: 8.7)	Published
4	Beniwal, M., Singh, A., & Kumar, N. "Forecasting Multistep Daily Stock Prices for Long-Term Investment Decisions: A study of Deep Learning models on Global Indices." **Engineering Applications of Artificial Intelligence**	SCIE (Impact Factor: 8)	Under Review
5	Beniwal, Mohit, Archana Singh, and Nand Kumar. "Predicting Long Term Prices of Nifty Index using Linear Regression and ARIMA: A comparative study." **International Journal of Accounting, Business and Finance** 2.1 (2022): 31-41.	Google Scholar	Published

Conferences

Table: Conferences

S. No.	Conference Paper
1	Beniwal, M., Singh, A., & Kumar, N. (2023, May). Predicting Next Quarter Nifty 50 Price using Genetic Algorithm and Support Vector Regression. In 2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC) (pp. 631-635). IEEE.
2	Beniwal, M., Singh, A., & Kumar, N. (2023, Jan). Predicting Nifty index Using Linear Regression And Arima: A Comparative Study . In 2nd International Conference on Business and Management (ICBM 2023)
3	Beniwal, M., Singh, A., & Kumar, N. (2023, July). Does support vector regression predict long-term stock index price better than linear regression . In International Conference on Financial Market and Corporate Finance (ICFMCF 2023).

References

Achelis, S. B. (2001). *Technical Analysis from A to Z*. McGraw Hill New York.

Achelis, & Steven. (2013). *Technical Analysis from A to Z*. McGraw Hill Education. http://www.equis.com/free/taaz/acknowledgments.html

Adebiyi, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Stock price prediction using the ARIMA model. *Proceedings - UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, UKSim 2014*, 106–112. https://doi.org/10.1109/UKSim.2014.67

Ahmed, S., Alshater, M. M., Ammari, A. El, & Hammami, H. (2022). Artificial intelligence and machine learning in finance: A bibliometric review. *Research in International Business and Finance*, *61*, 101646. https://doi.org/https://doi.org/10.1016/j.ribaf.2022.101646

al Galib, A., Alam, M., & Rahman, R. M. (2014). Prediction of stock price based on hidden Markov model and nearest neighbour algorithm Prediction of stock price based on hidden Markov model and nearest neighbour 263. In *Int. J. Information and Decision Sciences* (Vol. 6, Issue 3).

Alkhatib, K., Khazaleh, H., Alkhazaleh, H. A., Alsoud, A. R., & Abualigah, L. (2022). A New Stock Price Forecasting Method Using Active Deep Learning Approach. *Journal of Open Innovation: Technology, Market, and Complexity*, *8*(2). https://doi.org/10.3390/joitmc8020096

Allen, D. M. (1974). The Relationship Between Variable Selection and Data Agumentation and a Method for Prediction. *Technometrics*, *16*(1), 125–127. https://doi.org/10.1080/00401706.1974.10489157

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1), 53. https://doi.org/10.1186/s40537-021-00444-8

Awad, M., & Khanna, R. (2015a). Deep Neural Networks. In *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Apress.

Awad, M., & Khanna, R. (2015b). *Support Vector Regression In: Efficient Learning Machines*. Apress.

Bao, D., & Yang, Z. (2008). Intelligent stock trading system by turning point confirming and probabilistic reasoning. *Expert Systems with Applications*, *34*(1), 620–627. https://doi.org/10.1016/j.eswa.2006.09.043

Barber, B. M., & Odean, T. (2000). Trading is hazardous to your wealth: The common stock investment performance of individual investors. *Journal of Finance*, *55*(2), 773–806. https://doi.org/10.1111/0022-1082.00226

Basak, S., Kar, S., Saha, S., Khaidem, L., & Dey, S. R. (2019). Predicting the direction of stock market prices using tree-based classifiers. *North American Journal of Economics and Finance*, *47*, 552–567. https://doi.org/10.1016/j.najef.2018.06.013

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, *191*, 192–213. https://doi.org/10.1016/j.ins.2011.12.028

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, *13*, 281–305.

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Dahal, K. R., & Khatri, R. K. C. (2022). Predicting stock market index using LSTM. *Machine Learning with Applications*, *9*, 100320. https://doi.org/10.1016/j.mlwa.2022.100320

Bhatt, D., Patel, C., Talsania, H., Patel, J., Vaghela, R., Pandya, S., Modi, K., & Ghayvat, H. (2021). CNN variants for computer vision: History, architecture, application, challenges and future scope. In *Electronics (Switzerland)* (Vol. 10, Issue 20). MDPI. https://doi.org/10.3390/electronics10202470

Borovkova, S., & Tsiamas, I. (2019). An ensemble of LSTM neural networks for high-frequency stock market classification. *Journal of Forecasting*, *38*(6), 600–619. https://doi.org/10.1002/for.2585

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992a). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 144–152.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992b). A Training Algorithm for Optimal Margin Classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*.

Box, G. E., Gwilym M. Jenkins, & G. Reinsel. (1970). Time series analysis: forecasting and control Holden-day San Francisco. *BoxTime Series Analysis: Forecasting and Control Holden Day*.

Brock, W., Josef Lakonishok, & Blake LeBaron. (1992). Simple Technical Trading Rules And The Stochastic Properties. *The Journal of Finance*, *47*(5), 1731–1764.

Budiharto, W. (2021). Data science approach to stock prices forecasting in Indonesia during Covid-19 using Long Short-Term Memory (LSTM). *Journal of Big Data*, *8*(1). https://doi.org/10.1186/s40537-021-00430-0

Chaudhari, K., & Thakkar, A. (2023). Neural network systems with an integrated coefficient of variation-based feature selection for stock price and trend prediction. *Expert Systems with Applications*, 119527. https://doi.org/10.1016/j.eswa.2023.119527

Chen, A.-S., Leung, M. T., & Daouk, H. (2003). Application of neural networks to an emerging ÿnancial market: forecasting and trading the Taiwan Stock Index. In *Computers & Operations Research* (Vol. 30). www.elsevier.com/locate/dsw

Chen, S., & Ge, L. (2019). Exploring the attention mechanism in LSTM-based Hong Kong stock price movement prediction. *Quantitative Finance*, *19*(9), 1507–1515. https://doi.org/10.1080/14697688.2019.1622287

Chen, Y., Wu, J., & Wu, Z. (2022). China's commercial bank stock price prediction using a novel K-means-LSTM hybrid approach. *Expert Systems with Applications*, *202*, 117370. https://doi.org/10.1016/j.eswa.2022.117370

Chhajer, P., Shah, M., & Kshirsagar, A. (2022). The applications of artificial neural networks, support vector machines, and long–short term memory for stock market prediction. *Decision Analytics Journal*, *2*, 100015. https://doi.org/https://doi.org/10.1016/j.dajour.2021.100015

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *ArXiv Preprint ArXiv:1406.1078*. http://arxiv.org/abs/1406.1078

Chowdhury, M., Howe, J. S., & Lin, J.-C. (1993). The relation between aggregate insider transactions and stock market returns. *Journal of Financial and Quantitative Analysis*, *28*(3), 431–437.

Christoffersen, P. F., & Diebold, F. X. (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science*, *52*(8), 1273–1287. https://doi.org/10.1287/mnsc.1060.0520

Chung, H., & Shin, K. S. (2018). Genetic algorithm-optimized long short-term memory network for stock market prediction. *Sustainability (Switzerland)*, *10*(10). https://doi.org/10.3390/su10103765

Chung, H., & Shin, K. shik. (2020). Genetic algorithm-optimized multi-channel convolutional neural network for stock market prediction. *Neural Computing and Applications*, *32*(12), 7897–7914. https://doi.org/10.1007/s00521-019-04236-3

Cochrane H., J. (1999). *New facts in finance*.

Cortes, C., Vapnik, V., & Saitta, L. (1995). Support-Vector Networks Editor. In *Machine Leaming* (Vol. 20). Kluwer Academic Publishers.

Countries by GDP. (2022). Https://Www.Populationu.Com/Gen/Countries-by-Gdp.

Dash, R. K., Nguyen, T. N., Cengiz, K., & Sharma, A. (2021). Fine-tuned support vector regression model for stock predictions. *Neural Computing and Applications*. https://doi.org/10.1007/s00521-021-05842-w

De, K. (1988). Learning with Genetic Algorithms: An Overview. In *Machine Learning* (Vol. 3).

DeMark, T. R. (1994). *The new science of technical analysis* (Vol. 45). John Wiley & Sons.

Doroudyan, M. H., & Niaki, S. T. A. (2021). Pattern recognition in financial surveillance with the ARMA-GARCH time series model using support vector machine. *Expert Systems with Applications*, *182*. https://doi.org/10.1016/j.eswa.2021.115334

Drachal, K., & Pawłowski, M. (2021). A review of the applications of genetic algorithms to forecasting prices of commodities. *Economies*, *9*(1). https://doi.org/10.3390/economies9010006

Du, Y. (2018). Application and analysis of forecasting stock price index based on combination of ARIMA model and BP neural network. *Chinese Control and Decision Conference (CCDC). IEEE*, 2854–2857.

Eames, F. L. (1894). *The New York Stock Exchange*. TG Hall.

Edgar, T. W., & Manz, D. O. (2017). Chapter 6 - Machine Learning. In T. W. Edgar & D. O. Manz (Eds.), *Research Methods for Cyber Security* (pp. 153–173). Syngress. https://doi.org/https://doi.org/10.1016/B978-0-12-805349-2.00006-6

Ellis, C. D. (1975). The Loser's Game. *Financial Analysts Journal*, *31*(4), 19–26. https://doi.org/10.2469/faj.v31.n4.19

Elman, J. L. (1990). Finding Structure in Time. *Cognitive Science*, *14*(2), 179–211. https://doi.org/10.1207/s15516709cog1402_1

Fama, E. F. (1965). The behavior of stock-market prices. *The Journal of Business*, *38*(1), 34–105.

Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*, *25*(2), 383–417. http://www.jstor.orgURL:http://www.jstor.org/stable/2325486Accessed:30/04/200806:47

Fama, E. F., & French, K. R. (1988). Permanent and temporary components of stock prices. *Journal of Political Economy*, *96*(2), 246–273.

Farsi, M. (2021). Application of ensemble RNN deep neural network to the fall detection through IoT environment. *Alexandria Engineering Journal*, *60*(1), 199–211. https://doi.org/https://doi.org/10.1016/j.aej.2020.06.056

Fukushima, K. (1980). Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biological Cybernetics* , *36*, 202.

Gao, R., Zhang, X., Zhang, H., Zhao, Q., & Wang, Y. (2022). Forecasting the overnight return direction of stock market index combining global market indices: A multiple-branch deep learning approach. *Expert Systems with Applications*, *194*. https://doi.org/10.1016/j.eswa.2022.116506

Geisser, S. (1975). The predictive sample reuse method with applications. *Journal of the American Statistical Association*, *70*(350), 320–328. https://doi.org/10.1080/01621459.1975.10479865

Gerlein, E. A., Mcginnity, M., Belatreche, A., & Coleman, S. (2016). Evaluating machine learning classification for financial trading: an empirical approach. *Expert Systems with Applications*, *54*, 193–207.

Graham, B., & David, L. Dodd. (1965). The Intelligent Investor. *New York: Harper & Row*.

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, *18*(5–6), 602–610. https://doi.org/10.1016/j.neunet.2005.06.042

Gregory Mankiw, N., Romer, D., & Shapiro, M. D. (1991). Stock Market Forecastability and Volatility: A Statistical Appraisal. *The Review of Economic Studies*, *58*(3), 455–477. http://restud.oxfordjournals.org/

Gu, W., & Peng, Y. (2019). Forecasting the market return direction based on a time-varying probability density model. *Technological Forecasting and Social Change*, *148*. https://doi.org/10.1016/j.techfore.2019.119726

Gülmez, B. (2023). Stock price prediction with optimized deep LSTM network with Artificial Rabbits Optimization Algorithm. *Expert Systems with Applications*, 120346. https://doi.org/10.1016/j.eswa.2023.120346

Guyon, I., Boser, B., & Vapnik, V. (1993). *Automatic Capacity Tuning of Very Large VC-dimension Classifiers In: Advances in Neural Information Processing Systems*.

Hamayel, M. J., & Owda, A. Y. (2021). A Novel Cryptocurrency Price Prediction Model Using GRU, LSTM and bi-LSTM Machine Learning Algorithms. *AI*, *2*(4), 477–496. https://doi.org/10.3390/ai2040030

Hardy, Q. (2012). *Jeff Hawkins Develops a Brainy Big Data Company*. The New York Times International. https://nyti.ms/46xOp1v

Heckman, L. (2013). *NASDAQ: A Guide to Information Sources*. Routledge.

Henrique, B. M., Sobreiro, V. A., & Kimura, H. (2018). Stock price prediction using support vector regression on daily and up to the minute prices. *Journal of Finance and Data Science*, *4*(3), 183–201. https://doi.org/10.1016/j.jfds.2018.04.003

Hinton, G. E., Krizhevsky, A., & Sutskever, I. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25*(1106–1114), 1.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, *18*(7), 1527–1554.

Hiransha, M., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2018). NSE Stock Market Prediction Using Deep-Learning Models. *Procedia Computer Science*, *132*, 1351–1362. https://doi.org/10.1016/j.procs.2018.05.050

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8).

Holland, J. H. (1992). Genetic Algorithms. *Scientific American*, *267*(1), 66–73. https://doi.org/10.2307/24939139

Hoseinzade, E., & Haratizadeh, S. (2019a). CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, *129*, 273–285. https://doi.org/https://doi.org/10.1016/j.eswa.2019.03.029

Hoseinzade, E., & Haratizadeh, S. (2019b). CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, *129*, 273–285. https://doi.org/10.1016/j.eswa.2019.03.029

Houssein, E. H., Dirar, M., Abualigah, L., & Mohamed, W. M. (2022). An efficient equilibrium optimizer with support vector regression for stock market prediction. *Neural Computing and Applications*, *34*(4), 3165–3200. https://doi.org/10.1007/s00521-021-06580-9

Ince, H., & Trafalis, T. B. (2008). Short term forecasting with support vector machines and application to stock price prediction. *International Journal of General Systems*, *37*(6), 677–687. https://doi.org/10.1080/03081070601068595

Jegadeesh, N., & Titman, S. (1993). Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency. *The Journal of Finance*, *48*(1), 65–91. https://doi.org/10.1111/j.1540-6261.1993.tb04702.x

Jiang, W. (2020). *Applications of deep learning in stock market prediction: recent progress*. https://doi.org/10.1016/j.eswa.2021.115537

Jiang, W. (2021a). Applications of deep learning in stock market prediction: Recent progress. In *Expert Systems with Applications* (Vol. 184). Elsevier Ltd. https://doi.org/10.1016/j.eswa.2021.115537

Jiang, W. (2021b). Applications of deep learning in stock market prediction: Recent progress. In *Expert Systems with Applications* (Vol. 184). Elsevier Ltd. https://doi.org/10.1016/j.eswa.2021.115537

Johannessen, J.-A., & Johannessen, J.-A. (2017). The south sea and mississippi bubbles of 1720. *Innovations Lead to Economic Crises: Explaining the Bubble Economy*, 59–87.

Kaczmarek, T., & Perez, K. (2022). Building portfolios based on machine learning predictions. *Economic Research-Ekonomska Istrazivanja* , *35*(1), 19–37. https://doi.org/10.1080/1331677X.2021.1875865

Kanwal, A., Lau, M. F., Ng, S. P. H., Sim, K. Y., & Chandrasekaran, S. (2022a). BiCuDNNLSTM-1dCNN — A hybrid deep learning-based predictive model for stock price prediction. *Expert Systems with Applications*, *202*. https://doi.org/10.1016/j.eswa.2022.117123

Kanwal, A., Lau, M. F., Ng, S. P. H., Sim, K. Y., & Chandrasekaran, S. (2022b). BiCuDNNLSTM-1dCNN — A hybrid deep learning-based predictive model for stock price prediction. *Expert Systems with Applications*, *202*. https://doi.org/10.1016/j.eswa.2022.117123

Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, *38*(5), 5311–5319. https://doi.org/10.1016/j.eswa.2010.10.027

Kazem, A., Sharifi, E., Hussain, F. K., Saberi, M., & Hussain, O. K. (2013). Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Applied Soft Computing Journal*, *13*(2), 947–958. https://doi.org/10.1016/j.asoc.2012.09.024

Khan, A. H., Cao, X., Katsikis, V. N., Stanimirovic, P., Brajevic, I., Li, S., Kadry, S., & Nam, Y. (2020). Optimal Portfolio Management for Engineering Problems Using Nonconvex Cardinality Constraint: A Computing Perspective. In *IEEE Access* (Vol. 8, pp. 57437–

57450). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2020.2982195

Khashei, M., & Hajirahimi, Z. (2019). A comparative study of series arima/mlp hybrid models for stock price forecasting. *Communications in Statistics: Simulation and Computation*, *48*(9), 2625–2640. https://doi.org/10.1080/03610918.2018.1458138

Khodaee, P., Esfahanipour, A., & Mehtari Taheri, H. (2022). Forecasting turning points in stock price by applying a novel hybrid CNN-LSTM-ResNet model fed by 2D segmented images. *Engineering Applications of Artificial Intelligence*, *116*. https://doi.org/10.1016/j.engappai.2022.105464

Kim, M. J., Min, S. H., & Han, I. (2006). An evolutionary approach to the combination of multiple classifiers to predict a stock price index. *Expert Systems with Applications*, *31*(2), 241–247. https://doi.org/10.1016/j.eswa.2005.09.020

Kolen, J. F., & Kremer, S. C. (2001). *A field guide to dynamical recurrent networks*. John Wiley & Sons.

Kramer, O., & Kramer, O. (2017). *Genetic algorithms*. Springer.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Kumar, K., & Haider, M. T. U. (2021a). Enhanced Prediction of Intra-day Stock Market Using Metaheuristic Optimization on RNN–LSTM Network. *New Generation Computing*, *39*(1), 231–272. https://doi.org/10.1007/s00354-020-00104-0

Kumar, K., & Haider, M. T. U. (2021b). Enhanced Prediction of Intra-day Stock Market Using Metaheuristic Optimization on RNN–LSTM Network. *New Generation Computing*, *39*(1), 231–272. https://doi.org/10.1007/s00354-020-00104-0

Kumar, R. (2016). 3 - Efficient capital markets and its implications. In R. Kumar (Ed.), *Valuation* (pp. 73–91). Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-12-802303-7.00003-6

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*, 116659. https://doi.org/https://doi.org/10.1016/j.eswa.2022.116659

Kurani, A., Doshi, P., Vakharia, A., & Shah, M. (2023). A Comprehensive Comparative Study of Artificial Neural Network (ANN) and Support Vector Machines (SVM) on Stock Forecasting. In *Annals of Data Science* (Vol. 10, Issue 1, pp. 183–208). Springer

Science and Business Media Deutschland GmbH. https://doi.org/10.1007/s40745-021-00344-x

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. In *Nature* (Vol. 521, Issue 7553, pp. 436–444). Nature Publishing Group. https://doi.org/10.1038/nature14539

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2323. https://doi.org/10.1109/5.726791

Lee, M. C., Chang, J. W., Yeh, S. C., Chia, T. L., Liao, J. S., & Chen, X. M. (2022). Applying attention-based BiLSTM and technical indicators in the design and performance analysis of stock trading strategies. *Neural Computing and Applications*, *34*(16), 13267–13279. https://doi.org/10.1007/s00521-021-06828-4

Leigh, W., Purvis, R., & Ragusa, J. M. (2002). Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support. *Decision Support Systems*, *32*(4), 361–377. https://doi.org/https://doi.org/10.1016/S0167-9236(01)00121-X

Leung, M. T., Daouk, H., & Chen, A.-S. (2000). Forecasting stock indices: a comparison of classification and level estimation models. *International Journal of Forecasting*, *16*(2), 173–190. www.elsevier.com/locate/ijforecast

Lev, B., & Thiagarajan, S. R. (1993). Fundamental Information Analysis. *Journal of Accounting Research*, *31*(2), 190–215. http://www.jstor.org/stable/2491270

Li, X., & Sun, Y. (2020). Stock intelligent investment strategy based on support vector machine parameter optimization algorithm. *Neural Computing and Applications*, *32*(6), 1765–1775. https://doi.org/10.1007/s00521-019-04566-2

Liang, H., Sun, X., Sun, Y., & Gao, Y. (2017). Text feature extraction based on deep learning: a review. In *Eurasip Journal on Wireless Communications and Networking* (Vol. 2017, Issue 1). Springer International Publishing. https://doi.org/10.1186/s13638-017-0993-1

Lin, Y., Yan, Y., Xu, J., Liao, Y., & Ma, F. (2021). Forecasting stock index price using the CEEMDAN-LSTM model. *North American Journal of Economics and Finance*, *57*. https://doi.org/10.1016/j.najef.2021.101421

Liu, N. K., & Lee, K. K. (1997). 2002 An Intelligent Business Advisor System for Stock Investment. *Expert Systems*, *14*(3), 129–139.

Liu, Y., Gong, C., Yang, L., & Chen, Y. (2020). DSTP-RNN: A dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction. *Expert Systems with Applications*, *143*. https://doi.org/10.1016/j.eswa.2019.113082

Lo, A. W., & MacKinlay, A. C. (1998). Stock market prices do not follow random walks: Evidence from a simple specification test. *The Review of Financial Studies*, *1*(1), 41–61.

Lohrmann, C., & Luukka, P. (2019). Classification of intraday S&P500 returns with a Random Forest. *International Journal of Forecasting*, *35*(1), 390–407. https://doi.org/https://doi.org/10.1016/j.ijforecast.2018.08.004

Lu, W., Li, J., Li, Y., Sun, A., & Wang, J. (2020). A CNN-LSTM-based model to forecast stock prices. *Complexity*, *2020*. https://doi.org/10.1155/2020/6622927

Lu, W., Li, J., Wang, J., & Qin, L. (2021). A CNN-BiLSTM-AM method for stock price prediction. In *Neural Computing and Applications* (Vol. 33, Issue 10, pp. 4741–4753). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/s00521-020-05532-z

Lv, P., Shu, Y., Xu, J., & Wu, Q. (2022). Modal decomposition-based hybrid model for stock index prediction. *Expert Systems with Applications*, *202*. https://doi.org/10.1016/j.eswa.2022.117252

Madge, S., & Bhatt, S. (2015). Predicting stock price direction using support vector machines. *Independent Work Report Spring*, *45*.

Mahmoodi, A., Hashemi, L., Jasemi, M., Mehraban, S., Laliberté, J., & Millar, R. C. (2022). A developed stock price forecasting model using support vector machine combined with metaheuristic algorithms. *OPSEARCH*. https://doi.org/10.1007/s12597-022-00608-x

Malkiel, B. G. (1973). A Random Walk Down Wall Street. *Norton & Co, New York*.

Malkiel, B. G. (2003a). *The Efficient Market Hypothesis and Its Critics*.

Malkiel, B. G. (2003b). The Efficient Market Hypothesis and Its Critics . *Journal of Economic Perspectives*, *17*(1), 59–82. https://doi.org/10.1257/089533003321164958

Mcinish, T., Upson, J., & Wood, R. A. (2014). The flash crash: Trading aggressiveness, liquidity supply, and the impact of intermarket sweep orders. *Financial Review*, *49*(3), 481–509. https://doi.org/10.1111/fire.12047

McNally, S., Roche, J., & Caton, S. (2018). Predicting the Price of Bitcoin Using Machine Learning. *Proceedings - 26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2018*, 339–343. https://doi.org/10.1109/PDP2018.2018.00060

Mehtab, S., Sen, J., & Dutta, A. (2021). Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models. *Machine Learning and Metaheuristics Algorithms, and Applications: Second Symposium, SoMMA 2020, Chennai, India*.

Mexmonov, S. (2020). World Experience of Stock Exchange Operations. *Архив Научных Исследований*, *33*(1).

Michie, R. (2001). *The London stock exchange: A history*. OUP Oxford.

Mirjalili, S. (2019). Genetic algorithm. In *Studies in Computational Intelligence* (Vol. 780, pp. 43–55). Springer Verlag. https://doi.org/10.1007/978-3-319-93025-1_4

Murphy, J. J. (1999). *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. Penguin.

Naik, N., & Mohan, B. R. (2019). Study of stock return predictions using recurrent neural networks with LSTM. *Communications in Computer and Information Science*, *1000*, 453–459. https://doi.org/10.1007/978-3-030-20257-6_39

Naughton, T., Truong, C., & Veeraraghavan, M. (2008). Momentum strategies and stock returns: Chinese evidence. *Pacific Basin Finance Journal*, *16*(4), 476–492. https://doi.org/10.1016/j.pacfin.2007.10.001

Nayak, S. C., Misra, B. B., & Behera, H. S. (2017). Artificial chemical reaction optimization of neural networks for efficient prediction of stock market indices. *Ain Shams Engineering Journal*, *8*(3), 371–390. https://doi.org/10.1016/j.asej.2015.07.015

Nazareth, N., & Reddy, Y. Y. R. (2023). Financial applications of machine learning: A literature review. In *Expert Systems with Applications* (Vol. 219). Elsevier Ltd. https://doi.org/10.1016/j.eswa.2023.119640

Neely, C., Weller, P., & Dittmar, R. (1997). Is Technical Analysis in the Foreign Exchange Market Profitable? A Genetic Programming Approach. *Journal of Financial and Quantitative Analysis*, *32*(4).

Nikou, M., Mansourfar, G., & Bagherzadeh, J. (2019a). Stock price prediction using DEEP learning algorithm and its comparison with machine learning algorithms. *Intelligent Systems in Accounting, Finance and Management*, *26*(4), 164–174. https://doi.org/10.1002/isaf.1459

Nikou, M., Mansourfar, G., & Bagherzadeh, J. (2019b). Stock price prediction using DEEP learning algorithm and its comparison with machine learning algorithms. *Intelligent Systems in Accounting, Finance and Management*, *26*(4), 164–174. https://doi.org/10.1002/isaf.1459

Nobre, J., & Neves, R. F. (2019). Combining Principal Component Analysis, Discrete Wavelet Transform and XGBoost to trade in the financial markets. *Expert Systems with Applications*, *125*, 181–194. https://doi.org/10.1016/j.eswa.2019.01.083

Nyberg, H. (2011). Forecasting the direction of the US stock market with dynamic binary probit models. *International Journal of Forecasting*, *27*(2), 561–578. https://doi.org/10.1016/j.ijforecast.2010.02.008

Pai, P. F., & Lin, C. S. (2005). A hybrid ARIMA and support vector machines model in stock price forecasting. *Omega*, *33*(6), 497–505. https://doi.org/10.1016/j.omega.2004.07.024

Pal, S. K., & Wang, P. P. (1996). *Genetic algorithms for pattern recognition*. CRC press.

Paper, R., Jarrett, J. E., & Kyper, E. (2011a). ARIMA Modeling With Intervention to Forecast and Analyze Chinese Stock Prices. *International Journal of Engineering Business Management*. www.intechopen.com

Paper, R., Jarrett, J. E., & Kyper, E. (2011b). ARIMA Modeling With Intervention to Forecast and Analyze Chinese Stock Prices. *International Journal of Engineering Business Management*, *3*(3), 53–58. www.intechopen.com

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *Expert Systems with Applications*, *42*(1), 259–268. https://doi.org/10.1016/j.eswa.2014.07.040

PESARAN, M. H., & TIMMERMANN, A. (1995). Predictability of Stock Returns: Robustness and Economic Significance. *The Journal of Finance*, *50*(4), 1201–1228. https://doi.org/10.1111/j.1540-6261.1995.tb04055.x

Petram, L. O. (2011). *The world's first stock exchange: how the Amsterdam market for Dutch East India Company shares became a modern securities market, 1602-1700*. Universiteit van Amsterdam [Host].

Pettit, R. R., & Venkatesh, P. C. (1995). Insider trading and long-run return performance. *Financial Management*, 88–103.

Poterba, J. M., & Summers, L. H. (1988). Mean reversion in stock prices: Evidence and implications. *Journal of Financial Economics*, *22*(1), 27–59.

Qian, B., & Rasheed, K. (2007). Stock market prediction with multiple classifiers. *Applied Intelligence*, *26*(1), 25–33. https://doi.org/10.1007/s10489-006-0001-7

Qian, X.-Y., & Gao, S. (2017). Financial Series Prediction: Comparison Between Precision of Time Series Models and Machine Learning Methods. *ArXiv Preprint ArXiv:1706.00948*. http://arxiv.org/abs/1706.00948

Qian, Y., Fan, Y., Hu, W., & Soong, F. K. (2014). On the training aspects of Deep Neural Network (DNN) for parametric TTS synthesis. *2014 IEEE International Conference on*

Acoustics, Speech and Signal Processing (ICASSP), 3829–3833. https://doi.org/10.1109/ICASSP.2014.6854318

Qin, L., Yu, N., & Zhao, D. (2018). Applying the convolutional neural network deep learning technology to behavioural recognition in intelligent video. *Tehnicki Vjesnik*, *25*(2), 528–535. https://doi.org/10.17559/TV-20171229024444

Ranjan, A., Mahadani, & Kumar, A. (2022). Stock Price Prediction Using Deep Learning-Based Univariate and Multivariate LSTM and RNN. In S. and S. J. and R. A. Sikdar Biplab and Prasad Maity (Ed.), *Proceedings of the 3rd International Conference on Communication, Devices and Computing* (pp. 95–103). Springer Singapore.

Rather, A. M. (2021). LSTM-based Deep Learning Model for Stock Prediction and Predictive Optimization Model. *EURO Journal on Decision Processes*, *9*. https://doi.org/10.1016/j.ejdp.2021.100001

Roberts, H. V. (1959). Stock-Market "Patterns" and Financial Analysis: Methodological Suggestions. *The Journal of Finance*, *14*(1), 1–10. http://www.jstor.org/stable/2976094

Rouf, N., Malik, M. B., Arif, T., Sharma, S., Singh, S., Aich, S., & Kim, H. C. (2021). Stock market prediction using machine learning techniques: A decade survey on methodologies, recent developments, and future directions. In *Electronics* (Vol. 10, Issue 21). MDPI. https://doi.org/10.3390/electronics10212717

Samsani, S. S., Mutahira, H., & Muhammad, M. S. (2022). Memory-based crowd-aware robot navigation using deep reinforcement learning. *Complex and Intelligent Systems*. https://doi.org/10.1007/s40747-022-00906-3

Sarker, I. H. (2021a). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science*, *2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1

Sarker, I. H. (2021b). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. In *SN Computer Science* (Vol. 2, Issue 6). Springer. https://doi.org/10.1007/s42979-021-00815-1

Schnaubelt, M. (2019). A comparison of machine learning model validation schemes for non-stationary time series data. *FAU Discussion Papers in Economics*, *11*. http://hdl.handle.net/10419/209136

Schslkopf, B., Burges, C., & Vapnik, V. (1996, July 16). Incorporating Invariances in Support Vector Learning Machines. *Artificial Neural Networks—ICANN 96*.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional Recurrent Neural Networks. *IEEE Transactions On Signal Processing*, *45*(11).

Schwager, J. D. (1995). *Technical analysis* (Vol. 43). John Wiley & Sons.

Selvamuthu, D., Kumar, V., & Mishra, A. (2019). Indian stock market prediction using artificial neural networks on tick data. *Financial Innovation*, *5*(1), 1–12.

Shah, J., Vaidya, D., & Shah, M. (2022). A comprehensive review on multiple hybrid deep learning approaches for stock prediction. In *Intelligent Systems with Applications* (Vol. 16). Elsevier B.V. https://doi.org/10.1016/j.iswa.2022.200111

Shen, G., Tan, Q., Zhang, H., Zeng, P., & Xu, J. (2018). Deep learning with gated recurrent unit networks for financial sequence predictions. *Procedia Computer Science*, *131*, 895–903. https://doi.org/10.1016/j.procs.2018.04.298

Singh, R., & Srivastava, S. (2017). Stock prediction using deep learning. *Multimedia Tools and Applications*, *76*(18), 18569–18584. https://doi.org/10.1007/s11042-016-4159-7

Smith, B. M. (2004). *A history of the global stock market: from ancient Rome to Silicon Valley*. University of Chicago press.

Smith, C. F. (1929). The Early History of the London Stock Exchange. *The American Economic Review*, *19*(2), 206–216. http://www.jstor.org/stable/1807309

Solares, E., De-León-Gómez, V., Salas, F. G., & Díaz, R. (2022). A comprehensive decision support system for stock investment decisions. *Expert Systems with Applications*, *210*. https://doi.org/10.1016/j.eswa.2022.118485

StatisticsTimes.com. (2020, August). *World GDP Ranking*. https://statisticstimes.com/economy/world-gdp-ranking.php

Stone, M. (1974). Cross-Validatory Choice and Assessment of Statistical Predictions. In *Journal of the Royal Statistical Society. Series B (Methodological)* (Vol. 36, Issue 2).

Sun, Y., Ding, S., Zhang, Z., & Jia, W. (2021). An improved grid search algorithm to optimize SVR for prediction. *Soft Computing*, *25*(7), 5633–5644. https://doi.org/10.1007/s00500-020-05560-w

Taljard, J. (2021). The Use of Genetic Algorithms for Automated Machine Learning in Trend Prediction in Time Series Data: A Review. *University of Cape Town*.

Teixeira Zavadzki de Pauli, S., Kleina, M., & Bonat, W. H. (2020). Comparing Artificial Neural Network Architectures for Brazilian Stock Market Prediction. *Annals of Data Science*, *7*(4), 613–628. https://doi.org/10.1007/s40745-020-00305-w

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. In *Expert Systems with Applications* (Vol. 177). Elsevier Ltd. https://doi.org/10.1016/j.eswa.2021.114800

Thakkar, A., & Chaudhari, K. (2022). Information fusion-based genetic algorithm with long short-term memory for stock price and trend prediction. *Applied Soft Computing*, *128*. https://doi.org/10.1016/j.asoc.2022.109428

Ticknor, J. L. (2013). A Bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications*, *40*(14), 5501–5506. https://doi.org/10.1016/j.eswa.2013.04.013

TimesofIndia. (2023). In charts: Demat accounts surge to 11 crore in January 2023. *The Times of India*. https://timesofindia.indiatimes.com/business/markets/in-charts-demat-accounts-surge-to-11-crore-in-january-2023-but-active-nse-accounts-dip-for-seventh-straight-months/articleshow/97946068.cms

Tsantekidis, A., Passalis, N., & Tefas, A. (2022). Chapter 5 - Recurrent neural networks. In A. Iosifidis & A. Tefas (Eds.), *Deep Learning for Robot Perception and Cognition* (pp. 101–115). Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-32-385787-1.00010-5

Vapnik, V. (1995). The Nature of Statistical Learning Theory. In *The Nature of Statistical Learning Theory*. Springer Verlag.

Vapnik, V., Golowich, S., & Smola, A. (1997). *Support vector method for function approximation, regression estimation, and signal processing, neural information processing systems*. MIT Press Cambridge.

Wafi, Ahmed. S., Hassan, H., & Mabrouk, A. (2015). Fundamental Analysis Models in Financial Markets – Review Study. *Procedia Economics and Finance*, *30*, 939–947. https://doi.org/https://doi.org/10.1016/S2212-5671(15)01344-1

Wang, H., Wang, J., Cao, L., Li, Y., Sun, Q., & Wang, J. (2021). A Stock Closing Price Prediction Model Based on CNN-BiSLSTM. *Complexity*, *2021*. https://doi.org/10.1155/2021/5360828

Wang, Y., Chen, Q., Ding, M., & Li, J. (2019). High precision dimensional measurement with convolutional neural network and bi-directional long short-term memory (LSTM). *Sensors (Switzerland)*, *19*(23). https://doi.org/10.3390/s19235302

Wang, Y., Zhao, Y., & Addepalli, S. (2021). Practical Options for Adopting Recurrent Neural Network and Its Variants on Remaining Useful Life Prediction. *Chinese Journal of Mechanical Engineering*, *34*(1), 69. https://doi.org/10.1186/s10033-021-00588-x

Wijaya, Y. B., Kom, S., & Napitupulu, T. A. (2010). Stock price prediction: Comparison of Arima and artificial neural network methods - An Indonesia stock's case. *Proceedings - 2010 2nd International Conference on Advances in Computing, Control and Telecommunication Technologies, ACT 2010*, 176–179. https://doi.org/10.1109/ACT.2010.45

WorldData.info. (2021, June 20). *The 50 Largest Economies of the World*. Eglitis-Media. https://www.worlddata.info/largest-economies.php

Wu, D., Wang, X., & Wu, S. (2022). Jointly modeling transfer learning of industrial chain information and deep learning for stock prediction. *Expert Systems with Applications*, *191*. https://doi.org/10.1016/j.eswa.2021.116257

Wu, J. M. T., Li, Z., Herencsar, N., Vo, B., & Lin, J. C. W. (2021). A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. *Multimedia Systems*. https://doi.org/10.1007/s00530-021-00758-w

Xia, H., Huang, K., & Liu, Y. (2022). Unexpected interest recommender system with graph neural network. *Complex and Intelligent Systems*. https://doi.org/10.1007/s40747-022-00849-9

Yadav, S. S., & Jadhav, S. M. (2019). Deep convolutional neural network based medical image classification for disease diagnosis. *Journal of Big Data*, *6*(1), 113. https://doi.org/10.1186/s40537-019-0276-2

Yang, F., Chen, J., & Liu, Y. (2021). Improved and optimized recurrent neural network based on PSO and its application in stock price prediction. *Soft Computing*. https://doi.org/10.1007/s00500-021-06113-5

Yeh, C. Y., Huang, C. W., & Lee, S. J. (2011). A multiple-kernel support vector regression approach for stock market price forecasting. *Expert Systems with Applications*, *38*(3), 2177–2186. https://doi.org/10.1016/j.eswa.2010.08.004

Yu, P., & Yan, X. (2020). Stock price prediction based on deep neural networks. *Neural Computing and Applications*, *32*(6), 1609–1628. https://doi.org/10.1007/s00521-019-04212-x

Yun, K. K., Yoon, S. W., & Won, D. (2021). Prediction of stock price direction using a hybrid GA-XGBoost algorithm with a three-stage feature engineering process. *Expert Systems with Applications*, *186*. https://doi.org/10.1016/j.eswa.2021.115716

Zhang, D., & Lou, S. (2021). The application research of neural network and BP algorithm in stock price pattern classification and prediction. *Future Generation Computer Systems*, *115*, 872–879. https://doi.org/10.1016/j.future.2020.10.009

Zhang, J., & Man, K.-F. (1998). Time series prediction using RNN in multi-dimension embedding phase space. *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, *2*, 1868–1873.

Zhang, J., Teng, Y. F., & Chen, W. (2019). Support vector regression with modified firefly algorithm for stock price forecasting. *Applied Intelligence*, *49*(5), 1658–1674. https://doi.org/10.1007/s10489-018-1351-7

Zhang, J., Zeng, Y., & Starly, B. (2021). Recurrent neural networks with long term temporal dependencies in machine tool wear diagnosis and prognosis. *SN Applied Sciences*, *3*(4), 442. https://doi.org/10.1007/s42452-021-04427-5

Zhang, Q., Zhang, P., & Zhou, F. (2022). Intraday and interday features in the high-frequency data: Pre- and post-Crisis evidence in China's stock market. *Expert Systems with Applications*, *209*. https://doi.org/10.1016/j.eswa.2022.118321

Zhang, S., Luo, J., Wang, S., & Liu, F. (2023). Oil price forecasting: A hybrid GRU neural network based on decomposition–reconstruction methods. *Expert Systems with Applications*, *218*. https://doi.org/10.1016/j.eswa.2023.119617

Zou, B., Wang, H., Li, H., Li, L., & Zhao, Y. (2022). Predicting stock index movement using twin support vector machine as an integral part of enterprise system. *Systems Research and Behavioral Science*, *39*(3), 428–439.