

**DEVELOPMENT OF A WEB BASED APPLICATION FOR STOCK
PRICE PREDICTION**

BY

ENECHUKWU RANSOME CHUKWUBUIKEM

(RUN/CMP/20/9564)

**A PROJECT SUBMITTED TO
THE DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF NATURAL SCIENCES,
REDEEMER'S UNIVERSITY, EDE, OSUN STATE, NIGERIA**

**IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
AWARD OF THE DEGREE OF BACHELOR OF SCIENCE (BSc.) IN
COMPUTER SCIENCE**

AUGUST, 2023

DECLARATION

I Enechukwu Ransome Chukwubuikem (RUN/CMP/20/9564), hereby declare that this project was carried out by me in the Department of Computer Science, Faculty of Natural Sciences, Redeemer's University, Ede, Osun State, Nigeria. To the best of my knowledge, this work has not been presented elsewhere for the award of a degree or any other purpose.

Enechukwu, Ransome Chukwubuikem

.....
Student

.....
Signature & Date

CERTIFICATION

We, the undersigned hereby certify that this project was carried out by Enechukwu, Ransome Chukwubuikem RUN/CMP/20/9564 in the Department of Computer Science, Faculty of Natural Sciences, Redeemer's University, Ede, Osun State, Nigeria. To the best of our knowledge, this work has not been presented elsewhere for the award of a degree or any other purpose.

Dr. T. A. Olowookere
.....
Supervisor

.....
Signature & Date

Prof. A. O. Ogunde
.....
Head of Department

.....
Signature & Date

Prof. C. O. Akanbi
.....
External Examiner

.....
Signature & Date

DEDICATION

First and foremost, I would be dedicating this project to God Almighty, thanking him for his guidance and grace. To my parents, of blessed memories, who instilled in me the values of hard work, perseverance, and faith, I am forever grateful. Though they are no longer with me, their love and support continue to inspire me. I hope to honor their legacy through my work.

ACKNOWLEDGMENT

I would like to express my genuine and deep gratitude to my sisters for their unwavering love and support, which has played a vital role in the success of my program at this esteemed institution. I extend my heartfelt appreciation to my supervisor, Mrs. T. O. Ojewunmi, for dedicating her time, effort, and guidance throughout this project, ensuring that its objectives were met.

I also want to acknowledge the collective efforts of both the academic and non-academic staff of the Department of Computer Science. Starting from the Academic Staff, Dr(Mrs) A. A. Kayode, Dr(Mrs) O. O. Olaniyan, Dr(Mrs) B. O. Oguntunde, Prof. A. O. Ogunde, Dr. M. O. Odim, Prof. S. A. Arekete, Dr. T. A. Olowookere, Mrs Adio, Mr Fagba, Mr Oyetade, Mr Adekunle, Mr Olorunfemi, Mr Victor, Mrs Onifade, Mr Agbolade, Dr. Abolarinwa. Their contributions have immensely enriched my experience during my stay, and I am truly grateful for their invaluable support. To each and every one of them, I extend my sincerest wishes for abundant blessings.

My sincerest thanks goes out to my friend, Daniel Enamudu a.k.a Danikoko, thank you so much brother. For all the late night google meets, and for all the spare time you sacrificed just to help out, i say a very big thank you.

To Adaimoabasi, thank you so much for your input, even with a very tight deadline, you were able to help. May God raise men and women to help you when you need them, Amen.

To Akinyode Ilerioluwa, thank you my friend for your assistance throughout the entirety of this project.

TABLE OF CONTENTS

Content	Page
TITLE PAGE	i
DECLARATION	ii
CERTIFICATION	iii
DEDICATION	iv
ACKNOWLEDGMENT	v
TABLE OF CONTENTS	vi
LIST OF TABLES	x
LIST OF FIGURES	xi
ABSTRACT	xiii
CHAPTER ONE INTRODUCTION	1
1.1 Background to the Study	1
1.2 Statement of the Problem	1
1.3 Aim and Objectives of the Study	2
1.4 Overview of Methodology	2
1.5 Scope of the Study	2
1.6 Significance of the Study	2
1.7 Definition of Terms	3
1.8 Organisation of the Project	3

CHAPTER TWO	REVIEW OF LITERATURE	5
2.0	Introduction	5
2.1	Stock Markets	5
2.1.1	Factors that Influence the Stock Market	5
2.2	Stock Prices	6
2.2.1	Stock Price Prediction	7
2.2.2	History of Stock Price Prediction	7
2.2.2.1	Early Approaches	7
2.2.2.2	Technical Analysis	8
2.2.2.3	Fundamental Analysis	8
2.2.2.4	Ensemble Analysis	9
2.2.3	Failure of Stock Price Prediction Systems	9
2.2.4	Success of Stock Price Prediction Systems	10
2.2.5	Importance of Stock Price Prediction to Stock Market Stakeholders	11
2.3	Artificial Intelligence	13
2.4	Machine Learning	14
2.5	Deep Learning	15
2.5.1	Deep Learning Architectures	16
2.5.1.1	AlexNet	17
2.5.1.2	VGGNet	19
2.5.1.3	GoogleNet	20
2.5.1.4	MobileNet	21
2.5.1.5	ResNet	23

2.5.1	Long Short Term Memory	25
2.6	Related Works	26
CHAPTER THREE	METHODOLOGY	36
3.0	Introduction	36
3.1	Data Collection	36
3.2	Existing System Analysis	37
3.3	Architecture of the Proposed System	38
3.3.1	Preprocessing Module	40
3.3.2	Training Module	41
3.4	Performance Metrics	42
3.5	Predictions	43
3.6	UML Diagrams	43
3.6.1	Use Case Diagram	43
3.6.2	Activity Diagram	45
CHAPTER FOUR	SYSTEM IMPLEMENTATION AND RESULTS DISCUSSION	46
4.0	Introduction	46
4.1	Data Analysis	46
4.2	Visualisation of the Stock Data	50
4.3	Feature Engineering	53
4.3.1	Creating a new dataframe with close column	54
4.4	Model Building	55
4.4.1	Result of Prediction	57

4.5 Performance Evaluation	57
4.6 Application Development	57
CHAPTER FIVE	SUMMARY, CONCLUSION AND
	RECOMMENDATIONS
5.0 Introduction	60
5.1 Summary	60
5.2 Conclusion	60
5.3 Limitations of the Study	61
5.4 Recommendations	61
5.5 Future Works	62
REFERENCES	63
APPENDICES	67
Appendix A	67
Appendix B	74

LIST OF TABLES

Table	Page
2.1 Summary of Related Works	31

LIST OF FIGURES

Figure	Page
2.1 Sub-fields of Artificial Intelligence	13
2.2 The Hidden Layers of a Deep Neural Network	16
2.3 Classification of Deep Learning Architecture	17
2.4 Architecture of AlexNet	18
2.5 VGGNet Architecture	19
2.6 Inception Module	21
2.7 Inception Module GoogleNet	21
2.8 MobileNets Architecture	22
2.9 Residual Block	24
2.10 ResNet Architectures	24
2.11 LSTM Networks	26
3.1 Collected Data	36
3.2 Architecture of Existing System	38
3.3 Architecture of Proposed System	39
3.4 Hidden layers of LSTM	41
3.5 Use Case Diagram of the System	44
3.6 Activity Diagram of the System	45
4.1 Collected Data	46
4.2 Stock information of Amazon	47
4.3 Stock information of Microsoft	47
4.4 Stock information of Google	48
4.5 Stock information of Apple	48
4.6 Statistical Information about Amazon	49
4.7 Statistical Information about Microsoft	49
4.8 Statistical Information about Google	50

4.9	Statistical Information about Apple	50
4.10	Closing Price of Microsoft and Amazon	51
4.11	Closing Price of Apple and Google	52
4.12	Visualisation of the sales volume for Apple and Google	52
4.13	Visualisation of the sales volume for Microsoft and Amazon	53
4.14	Apple Dataset	54
4.15	Creating dataframe with the close column	54
4.16	Importing the MinMaxScaler	55
4.17	Creating the training Dataset	55
4.18	Building the LSTM Model	56
4.19	Creating the testing Dataset	56
4.20	Result of prediction	57
4.21	Stock Prediction System	58
4.22	Time Series Data	58
4.23	Forecasted Data	59

ABSTRACT

The stock market is one of the most important financial markets, it is a market where people gather to buy and sell shares of publicly listed companies come together for trading purposes. Study has shown that financial analysts face difficulties in analysing and forecasting the market manually and predicting the accurate or close to the accurate movement of the stock prices will reduce risk in the stock market. Therefore, the aim of this project is to develop a web application for predicting stock prices using Long Short Term Memory (LSTM). Stock prices of four companies were collected and Long Short Term Memory was used to analyse and predict the stock prices, the model accuracy test gave a Root Mean Squared Error result of 7.2. The application was developed using Hyper Text Markup Language (HTML) and Cascade Style Sheet (CSS) for the frontend development and streamlit a framework of Python was used for the backend development. The web application developed shows the movement of the stock prices and the prediction of one year's worth of future stock prices of a few companies. Individuals interested in buying stocks can view the movement and check future predictions on the application.

Keywords: Stock Market, Long Short Term Memory, Stock Price Prediction, Web-application

CHAPTER ONE

INTRODUCTION

1.1 Background to the Study

Financial markets are critical to increasing economic efficiency by getting funds transferred from those who have no practical use for them to those who do. Well-functioning financial markets produce high economic growth and poor-performing financial markets are the main reason many countries remain desperately poor (Mishkin, 2019).

The stock market is one of the most followed financial markets where people always speculate on where the market is heading, get excited when they brag about their big killings, and get depressed when they suffer a loss which in turn interprets the stock market as a place where people can get rich or poor easily (Mishkin, 2019). The stock market has given business owners and entrepreneurs an advantage to raise capital and expand their businesses. It has allowed investors to generate profits and increase their wealth.

The highest single-day decline in the stock market happened on a Monday, October 19, 1987, now known as Black Monday, Dow Jones Industrial Average (DJIA) falling by 22% (Dilallo, 2023), revealing how volatile the stock prices and market can be.

1.2 Statement of the Problem

Since the beginning, investors have aimed to predict stock prices. It is difficult for financial analysts to analyse and forecast the market by manually churning large volumes of data generated by stock markets (Saud & Shakya, 2019).

Predicting the accurate movement of stock prices can lessen risk in the stock market and increase the rate of investment and performance of the stock market. Machine learning models can improve the accuracy of prediction.

1.3 Aim and Objectives of the Study

The aim of this study is to develop a web application for predicting stock prices using Long Short-Term Memory (LSTM).

The specified objectives to achieve this aim are to:

- i. gather stock price data,
- ii. train a model for predicting stock price,
- iii. test the trained model,
- iv. deploy the model in a web application and
- v. evaluate the usability of the system.

1.4 Scope of the Study

This project will gather data from three companies to predict their stock values.

1.5 Overview of the Methodology

- i. The stock price was obtained from yahoo finance and divided into 60%-40% testing and training data.
- ii. The model was trained using LSTM deep learning algorithms
- iii. The test model was evaluated using Root mean square error (RMSE)
- iv. The system was developed using streamlit, HTML, and CSS.

1.6 Significance of the Study

Developing a web application for stock price prediction is important because:

- i. It helps investors know the essential worth of security before investing,
- ii. It reduces the high risk and insecurity of investing,
- iii. It helps expand the stock market positively,
- iv. Improves the economy of a country.

1.7 Definition of Terms

Security: any financial claim or piece of property that is susceptible to ownership.

Stock prices: The stock price is the current value or price at which a share of stock is exchanged on the stock market.

Investors: a person or organisation that puts money into property, business, or financial schemes with the expectation of gaining a profit.

Funds: a sum of money set aside for a particular purpose.

Recurrent Neural Network (RNN): is a deep learning technique that is utilised for speech recognition, voice recognition, time series production, and natural language processing.

Long Short Term Memory (LSTM): is a type of Recurrent Neural Network that can learn long-term dependencies.

Deep learning: is a subset of machine learning that enables systems to cluster data and make predictions with incredible accuracy using three or more layers of neural network.

1.8 Organisation of the Project

- i. Chapter One (Introduction) presents the background of study, statement of the problem, the aim and the objectives of the project.

- ii. Chapter Two (Literature Review) provides a review of literature of the study and also, a review of existing algorithms used as benchmarks for this study.
- iii. Chapter Three (Methodology) provides in depth description and explanation of the algorithms to be used in this study.
- iv. Chapter Four (Implementation of Design) shows the implementation of the system, the process involved in developing the eye detection system and the software infrastructure used to achieve the set of objectives and documentation.
- v. Chapter Five (Conclusion and Recommendation) comprises the concluding part of the project.

CHAPTER TWO

LITERATURE REVIEW

2.0 Introduction

This chapter discusses the literature review and related works

2.1 Stock Markets

A stock market is a venue where individuals looking to buy and sell shares of publicly listed companies come together for trading purposes (Khude, 2020).

Since it enables businesses to swiftly access public capital, a well-functioning stock market is thought to be essential for economic growth. The stock market's fundamental objective is to offer businesses cash so they may finance and grow their operations. The secondary objective is to allow investors to participate in publicly traded businesses' earnings (CFI Team, 2022).

2.1.1 Factors that Influence the Stock Market

This section explains the economic factors that affect the stock market.

Interest Rates: are set by the Federal Reserve to make borrowing money more or less expensive and keep currency inflation within an established target rate. Since stock values are tied to how profitable their primary companies are, profitability may be limited when interest rates are high and consumers and businesses are buying less stuff. A low-interest rate makes the economy's development easier (Speed Trader, 2023).

Inflation: is the rate at which a currency devalues each year. The stock market anticipates some changes in inflation rates, so not every announcement will impact

the market if it is consistent with expectations. However, if the announcement significantly exceeds the changes anticipated by investors, it will have a negative market impact. High inflation hurts businesses because it devalues the currency (Speed Trader, 2023).

Politics: On the stock market, politics on a national and worldwide level can have a significant impact. Because politics and economics are not directly related, it is challenging for traders to understand how political news may affect the market. The US's trade-off with China is an example of how politics can affect the market (Speed Trader, 2023).

Foreign Markets: The health of foreign markets can sometimes have an impact on the domestic market. The main determinant is how dependent the domestic market is on those foreign markets (Speed Trader, 2023).

Unemployment rate and jobs report: The relationship between unemployment rates and the stock market can be complicated. Job reports can either reassure or frighten investors, which can influence longer-term movements in the stock market (Speed Trader, 2023).

Economic Growth and Projections: Economic growth reports and projections are common triggers for intra-day and multi-day movements in the stock market given that the economic growth announcement must fulfil investor expectations for both present and future growth in order to produce a substantial rise in stock prices (Speed Trader, 2023).

2.2 Stock Prices

The stock price is the current price that a share of stock is trading for on the market (CFI Team, 2023). It represents the cost at which investors are ready to purchase or sell shares of that particular company at a given point in time.

The stock price is normally expressed in dollars per share and can fluctuate during the trading day as buyers and sellers carry out trades on stock exchanges. Stock prices are frequently monitored by investors and traders because they represent a company's perceived value and market mood (Investopedia, 2023).

2.2.1 Stock Price Prediction

Stock price prediction refers to the attempt to forecast the future price movements of a stock or a collection of stocks. It uses a variety of approaches, strategies, and models to analyse historical data, market patterns, and other pertinent elements in order to make an educated prediction regarding the direction and amount of future price changes (Investopedia, 2023).

The primary reason for investors to sell out their stock price is continuous unrest in the stock market. It is impossible to predict stock market movement with complete accuracy, but by predicting stock market movement more precisely based on historical data analysis, investors' losses can be greatly reduced. (Anup. Mahbubur, Al Almin, Sahab, & Sarnali, 2022).

Stock price prediction is a hotly debated topic in several domains, including trading, finance, statistics, and computer science. Specialised traders inspect stocks using fundamental and technical research to make business decisions (Gomathy, Linganna, & Reddy, 2021).

2.2.2 History of Stock Price Prediction

The history of stock price prediction is intertwined with the development of financial markets and advancements in technology. For numerous years, the prediction of stock prices has been of interest to investors and researchers alike. Although it is difficult to make precise predictions about stock prices due to the intricate and financial markets' ever-changing character, a multitude of techniques and strategies have been devised to anticipate future stock prices. Here is a brief overview of the history of stock price prediction.

2.2.2.1 Early Approaches

The Random Walk Theory: Developed by economist Burton Malkiel in 1973, this theory suggests that stock prices follow a random path and are unpredictable in the short term (Malkiel, 1973).

Efficient Market Hypothesis (EMH): Proposed by Eugene Fama in 1970, According to EMH, stock prices fully represent all available information, making persistent outperformance of the market unachievable (Fama, 1970).

2.2.2.2 Technical Analysis

Moving Averages: A popular technical indicator for smoothing out price volatility by determining the average closing price over a given time period (Achelis, 1995).

Bollinger Bands: Developed by John Bollinger, this indicator uses standard deviations to create upper and lower bands around the moving average, providing a measure of price volatility (Bolliber, 2001).

2.2.2.3 Fundamental Analysis

(Hochreiter & Schmidhuber, 1997) company's earnings and financial performance such as the Price-to-Earnings (P/E) ratio and discounted cash flow (DCF) analysis (Graham & Dodd, 1934).

Artificial Neural Networks (ANN): ANNs have been applied to stock price prediction using historical price and volume data as input to train a network to forecast future prices (Refenes, Zaprains, & Francis, 1993).

Support Vector Machines (SVM): SVMs have been used to predict stock price movements by finding patterns in historical data (Chang & C., 2011).

Recurrent Neural Networks (RNN) and Long Short-Term Memory: These deep learning models have shown promise in capturing temporal dependencies and have been used for stock price prediction (Hochreiter & Schmidhuber, 1997).

2.2.2.4 Ensemble Methods

Random Forests: An ensemble learning method that combines multiple decision trees to make predictions (Breiman, 2001).

Gradient Boosting: A machine learning technique that builds an ensemble of weak predictors by sequentially fitting new models to the residuals of the previous models (Friedman, 2001).

2.2.3 Failure of Stock Price Prediction Systems

While there have been numerous attempts to predict stock prices, it is widely recognised that accurately forecasting stock prices, it is widely recognised that accurately forecasting stock prices consistently is challenging. The failure of stock price prediction can be attributed to several factors including the efficient market hypothesis, the complexity of financial markets and the limitations of prediction

models. Here are some works that discussed the challenges and failures of stock price prediction.

Fama (1970) introduced the efficient market hypothesis and argued that stock prices fully reflect all available information, making it difficult to consistently predict future prices (Fama, 1970).

Malkiel (2012) analysed the performance of equity mutual funds and concludes that active fund managers struggle to consistently beat the market, reinforcing the difficulty of predicting stock prices (Malkiel B. G., 2012).

Brown & Cliff (2004) investigated the relationship between investor sentiment and stock market returns and find that sentiment indicators have limited predictive power for short-term stock prices (Brown & Cliff, 2004).

2.2.4 Success of Stock Price Prediction Systems

Stock price prediction systems aim to forecast the future movement of stock prices based on historical data and various analytical techniques. However, it's important to note that predicting stock prices with high accuracy consistently is a complex and challenging task, and no system can guarantee absolute success (Khedkar, Choudhari, & Dhok, 2019).

While there have been numerous approaches and models developed for stock price prediction, the success of these systems varies significantly. Some systems have shown promising results and achieved moderate success in predicting stock prices over short-term periods, while others have struggled to provide consistent and accurate predictions (Khedkar, Choudhari, & Dhok, 2019).

It's worth mentioning that the stock market is influenced by a wide range of factors, including economic indicators, company performance, geopolitical events, investor sentiment and market dynamics. These factors introduce a level of unpredictability and volatility that makes accurate prediction difficult (Khedkar, Choudhari, & Dhok, 2019).

The success of stock price prediction systems is often evaluated using metrics such as accuracy, precision, recall and mean absolute error (MAE). Researchers and practitioners in the field of financial forecasting continuously work on developing new algorithms, incorporating machine learning and artificial intelligence techniques and refining existing models to improve prediction accuracy (Khedkar, Choudhari, & Dhok, 2019).

It's important to approach stock price prediction with caution and consider them as one of many tools for investment decision making. Investors should conduct thorough fundamental analysis, assess market trends and consider other relevant factors before making any investment decisions (Khedkar, Choudhari, & Dhok, 2019).

2.2.5 Importance of Stock Price Prediction to Stock Market Stakeholders

Stock price prediction is of significant importance to various stakeholders in the stock market and here are some reasons why

Investors: Stock price prediction assists investors in making informed decisions on whether to buy, hold, or sell stocks. Investors can identify possible investment opportunities, minimise risk, and maximise returns by analysing and forecasting stock prices. Accurate forecasts allow investors to successfully timing trades and capitalise on market trends (Vargas et al, 2020).

Traders: Traders, especially short-term traders, heavily rely on stock price predictions to execute their trading strategies. They aim to capitalise on short-term price movements and volatility. Timely and accurate predictions can help traders identify entry and exit points, optimise their trading positions and maximise profits (Vargas et al, 2020).

Financial Institutions: Stock price projections are used by banks, hedge funds, and other financial institutions to influence their investment strategies and portfolio management. These institutions frequently manage significant amounts of money and require accurate forecasts in order to distribute capital efficiently, reduce risks, and produce returns for their clients and shareholders (Vargas et al, 2020).

Fund Managers: Fund managers responsible for mutual funds, pension funds and other investment vehicles closely monitor stock price predictions. They rely on these predictions to evaluate the performance of their existing portfolio, adjust asset allocations and make investment decisions that align with their fund's objectives. Accurate predictions are crucial for managing risk and delivering competitive returns to their investors (Vargas et al, 2020).

Regulatory Authorities: Regulatory bodies such as securities commissions and financial regulators, closely monitor stock market activities and ensure fair and orderly trading. Stock price predictions help regulators identify irregularities, market manipulation, or potential risks to investor protection. These predictions assist in maintaining market integrity and enforcing compliance with regulations (Vargas et al, 2020).

Risk Management: Stock price prediction aids in risk management by providing stakeholders with an understanding of potential market fluctuations and volatility.

Investors can assess the risk-return trade-off and adjust their portfolios accordingly. By incorporating stock price predictions into risk models, portfolio managers can optimise asset allocation and diversification strategies to mitigate risk exposure (Chong & Han, 2019).

Analysts and Researchers: Stock price prediction is crucial for researchers and analysts in conducting studies, market analysis and developing investment strategies. Accurate predictions provide insights into market behaviour, investor sentiment as well as the effect of various factors on stock prices. This knowledge enhances the understanding of financial markets and supports the development of robust investment models (Bertoluzzo, Bortoli, & Delprete, 2020).

Company Management: Stock price predictions are important for company management teams to gauge market sentiment and make strategic decisions. They help management assess the impact of different factors on their company's stock price, plan capital raising activities and evaluate the potential benefits of stock buybacks (Mishra & Sridhar, 2021).

2.3 Artificial Intelligence

The simulation of human intelligence in computers that have been educated to think like people and mimic their actions is referred to as artificial intelligence. The ability of artificial intelligence to rationalise and execute actions that have the best likelihood of reaching a certain goal is its ideal feature. Machine learning and deep learning are subsets of artificial intelligence (Frankenfield, 2022). Figure 2.1 shows the sub-fields of artificial intelligence.

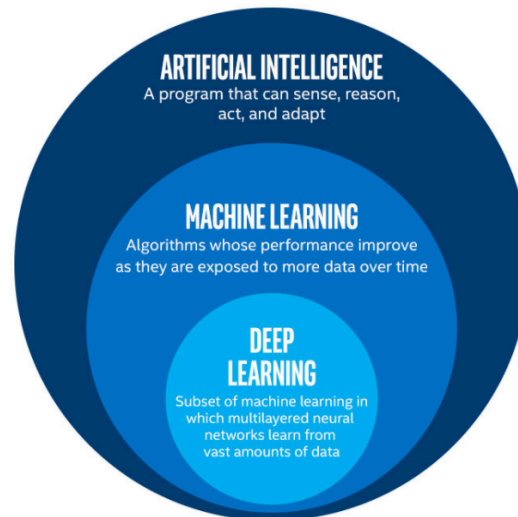


Figure 2.1: Sub-fields of Artificial Intelligence (Source: Singh S., 2018)

2.4 Machine Learning

Machine learning is an artificial intelligence subset that focuses on developing systems that learn or improve performance based on the data they ingest. As we engage with banks, purchase online, or use social media, machine learning algorithms come into play to make our experience efficient, smooth, and secure (Oracle, 2023).

Machine learning models fall into three primary categories

Supervised machine learning: Its use of labelled datasets to train algorithms to reliably identify data or predict outcomes defines it. As input data is fed into the model, the weights are adjusted until the model is adequately fitted. This is performed as part of the cross-validation procedure to guarantee that the model does not overfit or underfit. Neural networks, naive bayes, linear regression, logistic regression, random forest, and support vector machine (SVM) are some of the approaches used in supervised learning (IBM, 2023).

Unsupervised machine learning: Machine learning algorithms are used to examine and cluster unlabelled datasets. These algorithms uncover hidden patterns or data grouping without the need for human interaction. This method is ideal for exploratory data analysis, cross-selling strategies, consumer segmentation, and visual pattern identification because to its ability to find similarities and contrasts in data. It is also used to reduce the number of features in a model through the dimensionality reduction technique (IBM, 2023).

Reinforcement machine learning: is a machine learning model comparable to supervised learning, except that the algorithm is not trained on sample data. This model train as it goes by using trial and error. A sequence of successful outcomes will be reinforced to develop the best recommendation or policy for a given problem (IBM, 2023).

2.5 Deep Learning

Deep learning is a machine learning subset that is essentially a three- or more-layered neural network. These neural networks attempt to replicate the human brain behaviour by studying from massive amounts of data. While a single-layer neural network can still produce approximate predictions, additional hidden layers can assist optimise and tune for accuracy (IBM, 2023).

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars) (IBM, 2023). Figure 2.2 shows the hidden layers of a deep neural network.

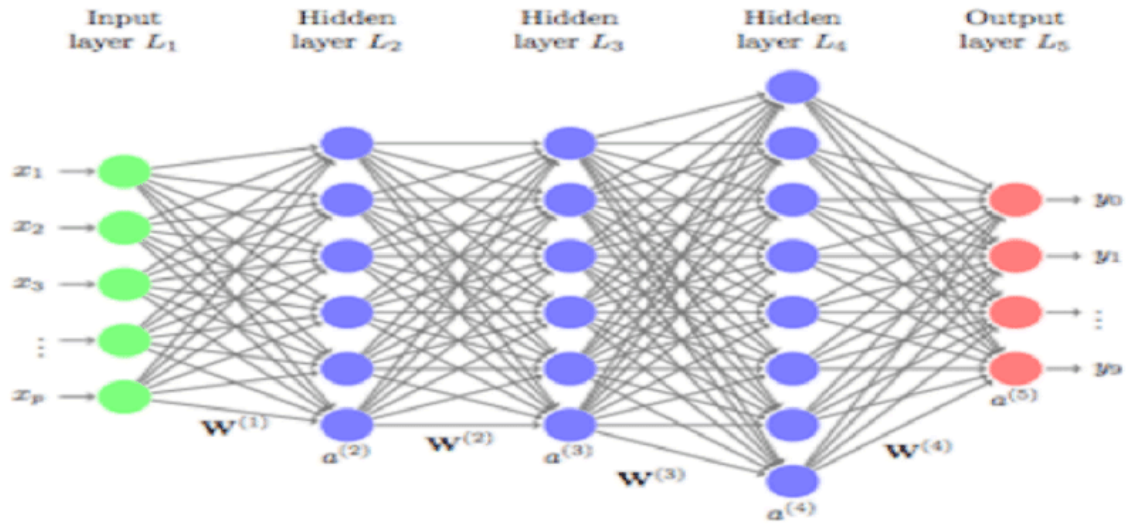


Figure 2.2: The Hidden layers of a Deep Neural Network (Source: ResearchGate, 2023)

2.5.1 Deep Learning Architectures

Connectionist architectures have been around for over 70 years, but it is the emergence of new architectures and the use of graphical processing units (GPUs) that have propelled them to the forefront of artificial intelligence. Deep learning is not a single approach, but rather a set of algorithms and structures that can be used to solve a variety of problems (Madhavan & Jones, 2021).

Although deep learning is not a new concept, it is experiencing tremendous growth as a result of the combination of extensively layered neural networks and the usage of GPUs to expedite their computations. The rise of big data has also played a role significantly to this increase. Because deep learning is based on training neural

networks with example data and rewarding their success, having more data is important for building these complex deep learning structures (Madhavan & Jones, 2021).

Deep learning refers to a wide range of architectures and algorithms. In this perspective, we will investigate six deep learning architectures produced during the last two decades. Among these designs, long short-term memory (LSTM) and convolutional neural networks (CNNs) stand out as not only the oldest but also the most commonly used in a variety of applications (Madhavan & Jones, 2021). Figure 2.3 shows the classification of deep learning architecture.

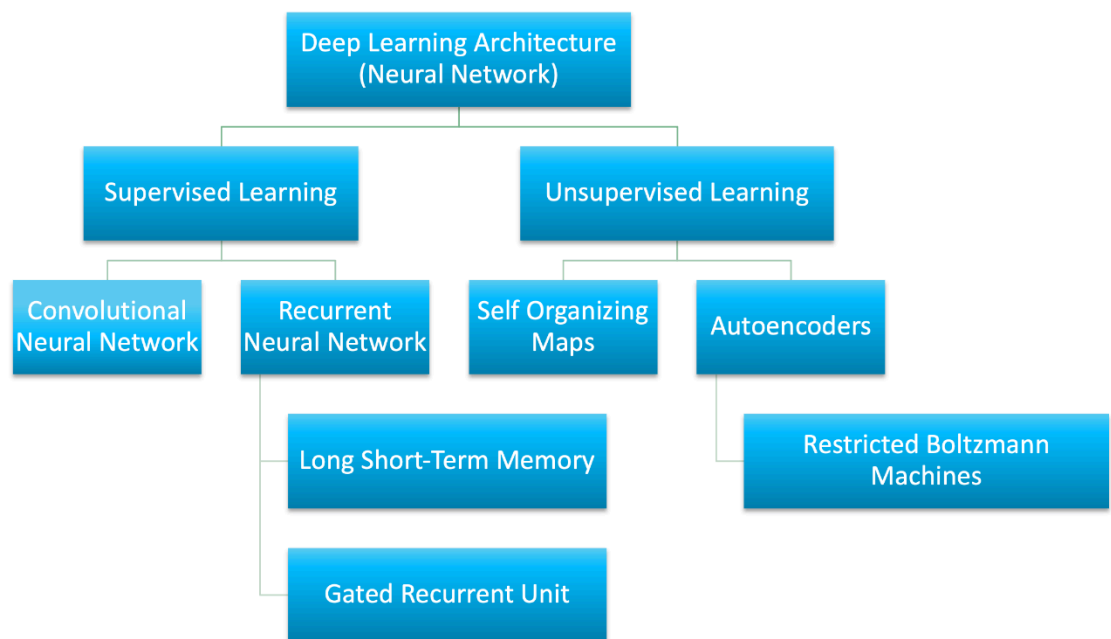


Figure 2.3: Classification of Deep Learning Architecture (Source: Madhavan & Jones, 2021)

2.5.1.1 AlexNet

AlexNet architecture garnered great recognition for its innovative usage of Convolutional Networks in the field of Computer Vision. It had great success in 2012 by winning the ImageNet ILSVRC competition, with a top-5 error rate of 15.4% compared to the next closest network's rate of 26.2%. While inspired by the LeNet-5 architecture, AlexNet was distinguished by its increased depth, size, and inclusion of stacked convolutional layers (Bezdan & Bacanin, 2019).

Unlike LeNet-5, which used the tanh activation function, AlexNet used the ReLU function. It also made use of the cross-entropy loss function. Notably, AlexNet's training method used a significantly larger dataset. AlexNet used a subset of the ImageNet dataset, whereas LeNet-5 was trained using the MNIST dataset, which consisted of 50,000 grayscale images classified into 10 classes. This subset included one million colour photos classified into 1,000 categories (Bezdan & Bacanin, 2019).

The input images for AlexNet were required to have a resolution of 224 x 224 pixels. The architecture itself consisted of five convolutional layers, three 2x2 max pooling layers, and two fully connected layers. To combat overfitting, a regularisation technique called dropout was employed specifically in the fully connected layers. The total number of parameters in AlexNet amounted to 60 million (Bezdan & Bacanin, 2019). Figure 2.4 shows AlexNet architecture.

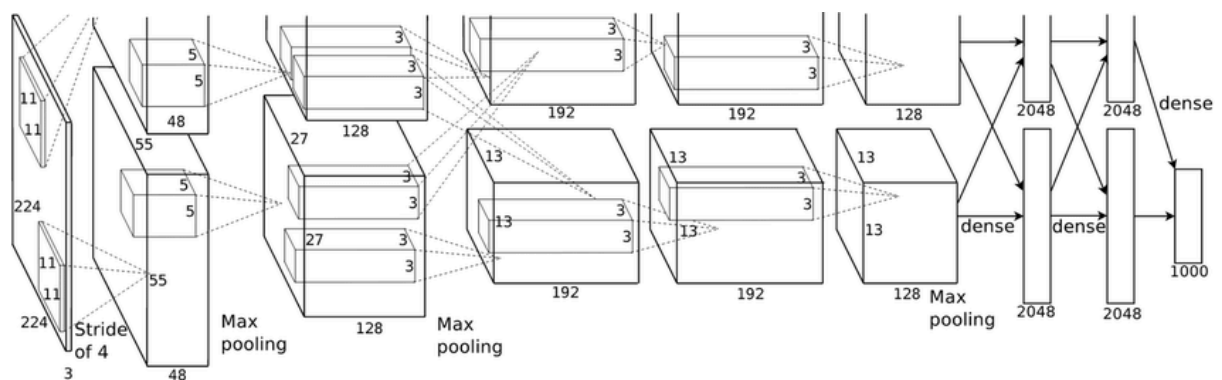


Figure 2.4: Architecture of AlexNet (Source: ResearchGate, 2023)

2.5.1.2 VGGNet

The VGG Network, developed by Karen Simonyan and Andrew Zisserman in 2014, was notable for its deep architecture. It introduced the concept that network depth plays a vital role in improving the accuracy of recognition and classification in Convolutional Neural Networks (CNNs). VGGNet, depicted in Figure 2.5, employed 3x3 filters. The authors provided insight into the reasoning behind this choice, explaining that two consecutive 3x3 filters effectively created a receptive field of 5x5, while three consecutive 3x3 filters resulted in a receptive field of 7x7. Furthermore, the number of filters in the VGGNet architecture doubled after each max-pooling operation (Bezdan & Bacanin, 2019). Figure 2.5 shows the architecture of VGGNet.

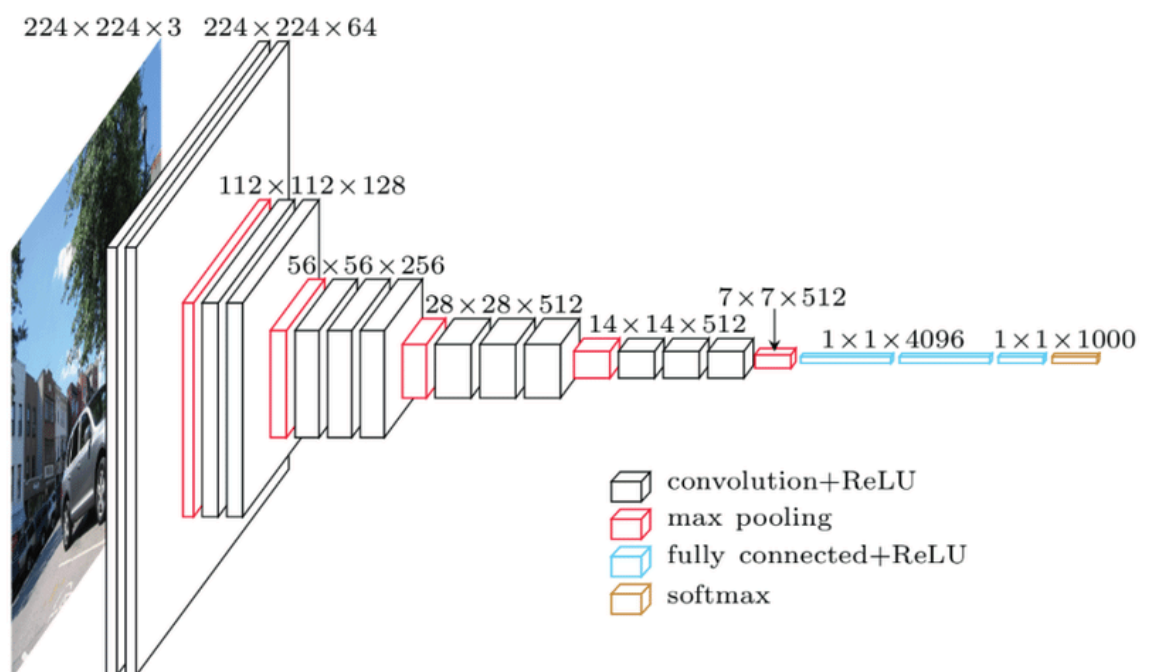


Figure 2.5: VGGNet Architecture (Source: ResearchGate, 2023)

2.5.1.3 GoogleNet

The winner of the ILSVRC competition in 2014 was GoogleNet. GoogleNet, depicted in Figure 2.7, is widely regarded as the first modern Convolutional Neural Network (CNN) architecture that deviates from a simple sequence of convolution and pooling layers. Instead, it introduced the concept of the inception architecture, which can be seen as a type of Network in Network (NIN), as exemplified in Figure 2.6. The inception module within this architecture incorporates skip connections, forming mini-modules that are repeated throughout the network (Bezdan & Bacanin, 2019).

The inception module played a significant role in reducing the number of parameters in the network. GoogleNet utilised approximately 7 million parameters, representing a drastic reduction of 9 times compared to its predecessor, AlexNet, which employed 60 million parameters. Moreover, VGGNet utilised roughly three times more parameters than AlexNet. GoogleNet employed nine inception modules and eliminated all fully connected layers by utilising average pooling to transition from a spatial dimension of 7x7 with 1024 channels to a dimension of 1x1 with 1024 channels. This elimination effectively removed a substantial number of parameters that did not contribute significantly to the network's performance (Bezdan & Bacanin, 2019).

In terms of data augmentation, GoogleNet employed multiple crops of the same image during training. This technique enhanced the network's ability to generalise.

Additionally, there have been subsequent versions of GoogleNet, with the most recent being Inception-v4 (Bezdan & Bacanin, 2019). Figure 2.6 shows the inception module and Figure 2.7 shows the inception module GoogleNet architecture.

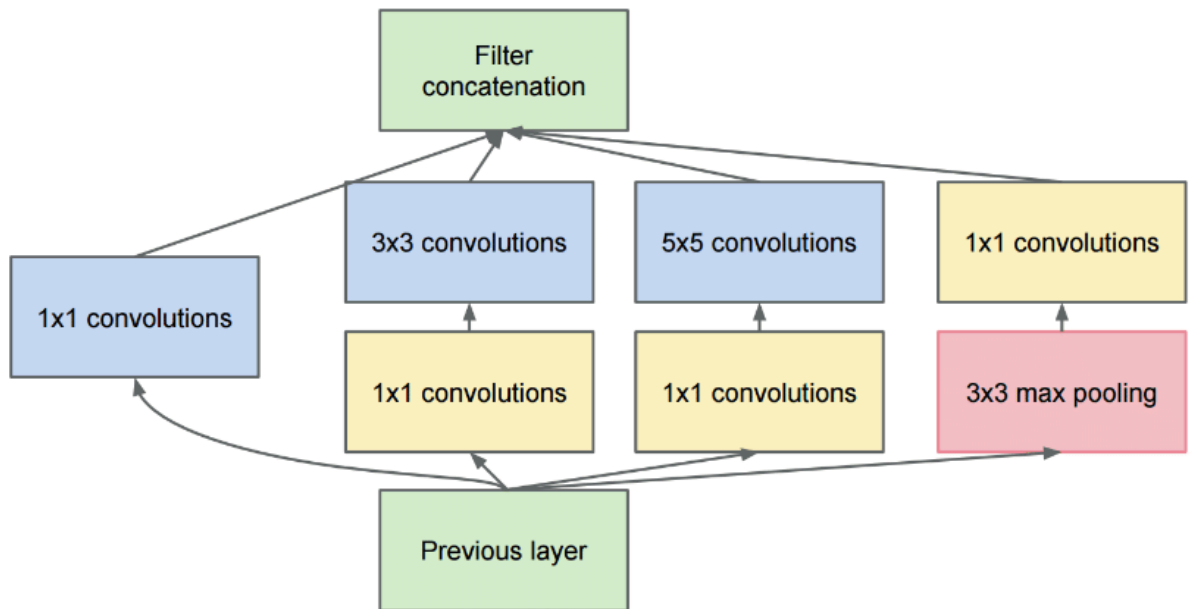


Figure 2.6: Inception Module (Source: ResearchGate, 2023)

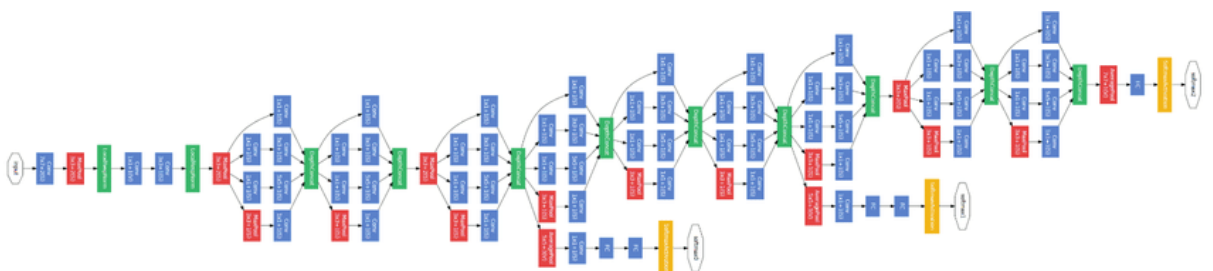


Figure 2.7: Inception Module GoogleNet Architecture (Source: ResearchGate, 2023)

2.5.1.4 MobileNet

The MobileNet model utilises depth wise separable convolutions, which are a type of factored convolutions. This approach breaks down a regular convolution into two parts: a depth wise convolution and a pointwise convolution, where the latter uses a

1x1 filter. In MobileNets, the depth wise convolution applies a filter to each input channel separately. The pointwise convolution then combines the outputs of the depth wise convolution by employing a 1x1 convolution. A typical convolution operation performs both filtering and combining of inputs to generate new outputs in a single step. In contrast, the depthwise separable convolution separates this process into two distinct layers: one for filtering and another for combining (Howard, et al., 2017). The MobileNet architecture is defined in Figure 2.8

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5 \times$	Conv dw / s1	$3 \times 3 \times 512$ dw
	Conv / s1	$1 \times 1 \times 512 \times 512$
	Conv dw / s2	$3 \times 3 \times 512$ dw
	Conv / s1	$1 \times 1 \times 512 \times 1024$
	Conv dw / s2	$3 \times 3 \times 1024$ dw
	Conv / s1	$1 \times 1 \times 1024 \times 1024$
	Avg Pool / s1	Pool 7×7
	FC / s1	1024×1000
	Softmax / s1	Classifier

Figure 2.8: MobileNets Architecture (Source: ResearchGate, 2023)

2.5.1.5 ResNet

The Residual Network (ResNet) emerged as the victor in the ILSVRC 2015 competition and stood out for its impressive 152-layer architecture. ResNet is constructed using a building block called a residual block, as depicted in Figure 11.

By stacking these residual blocks together, the network achieves its depth. Each residual block consists of two 3x3 convolutional layers and periodically doubles the number of filters while downsampling spatially using a stride of 2. Notably, ResNet incorporates special skip connections and utilises batch normalisation after every convolutional layer (Bezdan & Bacanin, 2019).

Optimising deeper models poses challenges, but ResNet tackles this by employing skip connections. These connections allow the activation from one layer to be fed directly to another layer, enabling the training of exceptionally deep networks while avoiding issues such as the vanishing and exploding gradient problems. To reduce the number of parameters, ResNet does not include fully connected layers, except for a final fully connected layer responsible for outputting the 1000 classes. Interestingly, ResNet stands out as the first architecture to surpass human performance, showcasing its superior capabilities (Bezdan & Bacanin, 2019). Figure 2.9 shows the residual block and Figure 2.10 shows the ResNet architecture.

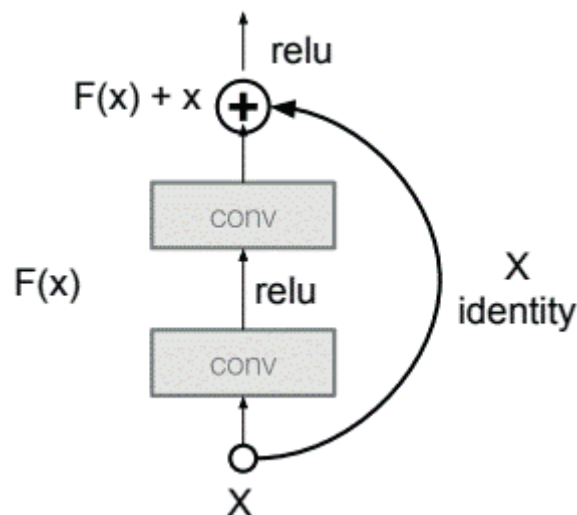


Figure 2.9: Residual Block (Source: ResearchGate, 2023)

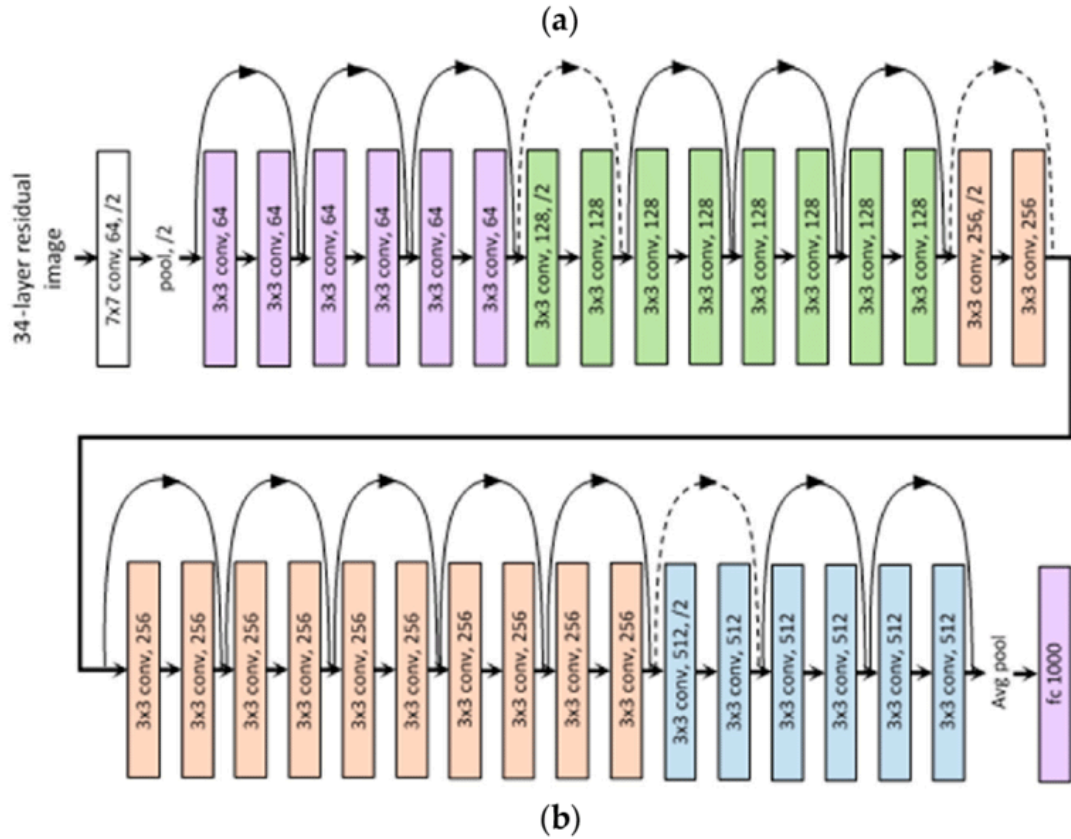


Figure 2.10: ResNet Architecture (Source: ResearchGate, 2023)

2.5.2 Long Short Term Memory

Hochreiter and Schmidhuber created the Long Short-Term Memory (LSTM) in 1997, which has grown in prominence in recent years as a recurrent neural network (RNN) architecture used for a variety of purposes. LSTMs are common in everyday devices such as cell phones, and IBM used them in IBM Watson to produce exceptional conversational voice recognition (Madhavan & Jones, 2021).

Unlike traditional neuron-based neural networks, the LSTM introduced a new concept called a memory cell. This memory cell's value can be retained for a short or lengthy

time, depending on its inputs. This capability allows the cell to remember important information rather than just its most recent computation (Madhavan & Jones, 2021).

The LSTM memory cell is made up of three gates that control the flow of information into and out of the cell. The input gate controls when fresh information can enter the memory, while the forget gate controls the forgetting of previous information, allowing the cell to recall new data. Finally, the output gate determines when the information stored in the cell is used as output. In addition, the cell contains weights that govern each gate. These weights are optimised depending on the resulting network output error using a training approach such as Backpropagation Through Time (BPTT) (Madhavan & Jones, 2021).

More recently, Convolutional Neural Networks (CNNs) and LSTMs have been employed in developing systems for image and video captioning. In these systems, the CNN handles the processing of images or videos, while the LSTM is trained to convert the output from the CNN into natural language captions (Madhavan & Jones, 2021). Figure 2.11 shows LSTM networks.

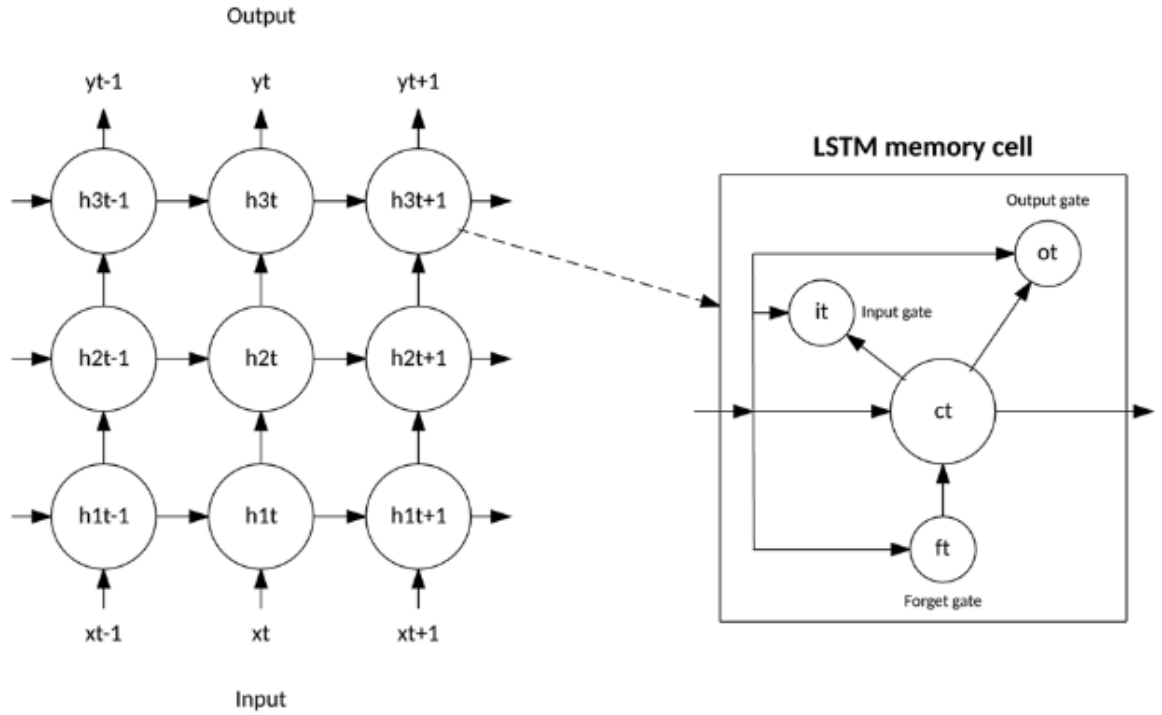


Figure 2.11: LSTM Networks (Source: Madhavan & Jones, 2021)

2.6 Related Works

Shen *et al.* (2020) researched short-term stock market prediction using two algorithms: feature engineering using FE + RFE + PCA and the LSTM model. The dataset used consisted of 3558 stocks from the Chinese stock market, and data was acquired using a web-scraping approach from Sina Finance web pages and the SWS Research website. Four types of data were collected: basic data, trade data, financial data, and other reference data, and the solution was implemented in three stages. The first stage was feature selection, which ensured that the features chosen were highly effective. Second, data was evaluated in order to do dimensionality reduction, and the final section is the work's key contribution, which was to develop a prediction model of target stocks. 5 main components were 89.03% accurate, 10 principal components were 89.35% accurate, 15 principal components were 89.39% accurate, 20 principal

components were 89.30% accurate, and 29 principal components were 90.29% accurate. The study solely looked at predictions for the short term.

Wen *et al.* (2020) researched stock price forecasting using the PCA-LSTM model. The data set used in this paper was Pingan bank stock information, including stock trading data from January 4, 2016, to December 28, 2018 (731 trading days). The data set was divided into 60% training, 20% verification, and 20% test sets, and the performance of the PCA-LSTM model was then evaluated using Root Mean Square Error (RSME) and Mean Absolute Percentage Error (MAPE). The PCA-LSTM model was compared to CNN, MLP, and Moving Average models, with Pingan Bank's test set data serving as the test object. As model evaluation indicators, the RSME and MAPE were employed. When compared to the other three comparison models, the PCA-LSTM model showed a superior fitting degree of prediction curve and true value curve on the test set, demonstrating that its prediction accuracy was higher. If a larger dataset had been employed, the research would have provided a more accurate forecast.

Saud *et al* (2019) did an analysis of the lookback period for stock price prediction using RNN variants with a dataset from Aug 8, 2007, to Nov 5, 2018, for NIB and NABIL. The data was preprocessed before being used for training and testing the deep recurrent neural network, with a 9:1 split of the dataset. The look-back time in the research work was adjusted from 2 to 30, and ten tests were performed for each value of look-back period utilised with deep RNN models and MAPE were captured. The lowest MAPE values obtained when predicting the stock price of NIB using GRU are lower than the minimum MAPE values obtained when predicting the stock price of NABIL using LSTM and VRNN, and a similar pattern was found when predicting the stock price of NABIL. The highest MAPE values achieved when forecasting the

stock price of NIB using GRU and LSTM are much lower than the maximum MAPE values obtained when predicting the stock price using VRNN, and a similar pattern was also found when predicting the stock price of NABIL. The study also discovered that the highest MAPE values were obtained for look-back durations of more than 15 years. The study concluded that GRU outperforms LSTM while both models outperform VRNN in predicting stock values. More technical indicators could be employed to improve stock price prediction.

Banerjee *et al.* (2020) made an attempt to predict the stock market trend using culmination of predictive modelling and regression algorithms. The model utilised current stock prices from the data sets acquired and divided the data into multiple subparts that were used to train and test the algorithm. The models were created and trained using supervised machine learning algorithms. Simple Linear Regression (SLR), Multiple Linear Regression (MLR), Multiple Linear Regression with Interactions, Principal Component Analysis (PCA), and Support Vector Machines (SVM) were the algorithms employed. SLR has a prediction accuracy of 58%, while MLR has a prediction accuracy of 69%. The algorithms' accuracy was too poor.

Singh *et al.* (2021) proposed a machine learning system for stock price prediction with the dataset of three different companies from the date 1/1/2012 to 10/10/2021 obtained from Yahoo Finance. The data was separated into training and testing segments using an 80:20 ratio, and the data was scaled to normalise the data range between 0 and 1 so that the algorithm could learn patterns from the data more easily. Following the scaling, an RNN was constructed for the dataset and initialised with a sequential regressor. The acquired data was fed into the LSTM model and trained for prediction, after which testing data was constructed from the scaled values, transformed to a Numpy array, and reshaped. The model was evaluated using root

mean square error, and the results were displayed graphically. The RMSE of LSTM prediction was compared to ARIMA, Linear Regression, and SVM, and LSTM produced the best results with an RMSE of 0.43.

Pramod *et al.* (2020) proposed an article about stock price prediction using LSTM. LSTM networks are a type of recurrent neural network (RNN) that can solve linear problems and learn context-specific temporal dependencies. The dataset was gathered, divided into testing and training data, the algorithm was trained to predict the price, and the data was finally visualised. The data was being pre-processed and scaled, limiting the factors. The model preparation includes cross-validation, which is a well-founded, anticipated execution of the model using the preparation information. The programme was able to anticipate the share price with an extremely low loss and mistake rate, with a 0.0024 loss rate. The prediction accuracy will improve as the number of data sets, layers, and LSTM modules increases.

Bhattacharjee *et al.* (2019) reviewed the statistical methods approach and machine learning methods approach to predict stock prices. Simple Moving Average (SMA), Weighted Moving Average (WMA), Exponential Smoothing, and the Naïve Approach were used as statistical methods, and machine learning methods included Simple Linear Regression (SLR), Ridge Regression, Lasso Regression, K-Nearest Neighbours (KNN), Random Forest, Support Vector Machine (SVM), Single Layer Perception (SLP), and Multi-Layer Perception (MLP). Tesla stock market data from 29-06-2010 to 17-03-2017 and Apple stock market data from 02-01-2014 to 31-12-2018 were used as datasets in the system. MSE values for statistical methods were found to be around 322.4 to 28.8 for the Tesla dataset and 125.4 to 11.2 for the Apple dataset, while MAPE values were found to be approximately 6.5 to 1.7 for the Tesla dataset and 4.7 to 1.3 for the Apple dataset. MSE values for machine learning

algorithms, on the other hand, vary from 10.3 to 3.8 for the Tesla dataset and 7.4 to 3.5 for the Apple dataset, while MAPE values range from 1.13 to 0.69 for the Tesla dataset and 1.09 to 0.7 for the Apple dataset. This revealed a significant difference in the average mistakes of statistical methods and machine learning methods, demonstrating that machine learning algorithms are more appropriate than traditional statistical methods.

Guo (2022) proposed an article on stock price prediction using ARIMA, GARCH, and LSTM. The article uses Yahoo Finance to choose the S&P500 index and transaction data for each trading day from September 26th, 2001 to September 25th, 2021. The data was separated into training and testing data, with 70% of the data used to train the model and the remaining observations used for testing and validation. The average error while using the ARIMA model was 3.19%, 1.65% when using the GARCH model, 0.40% when using the LSTM model, and 0.27% when using the Mixed Model.

Xiao *et al.* (2022) reviewed an article on stock price prediction using Autoregressive Integrated Moving Average Model (ARIMA) and Long Short-Term Memory (LSTM). The LSTM served as the foundation for employing the ARIMA-LSTM hybrid model, which combines linear and nonlinear components to give better results than a single method. To compare and assess the ensemble approach and various other methods, certain generally used methods such as MAE, MSE, and RMSE were utilised to compare and evaluate the ARIMA-LSTM hybrid model performance. The RMSE using ARIMA was 0.043768 and 0.028828 using LSTM. The RMSE and ARIMA models are well suited to data with a single dimension, and the ARIMA-LSTM hybrid model outperformed other financial models in terms of prediction accuracy.

Wu *et al* (2022) proposed a framework using S_I_LSTM model for price prediction based on sentiment analysis and multiple data sources. As the study object, five stocks from EastMoney.com were chosen. EastMoney.com provided the dataset, which included 2351 new articles and 33500 forum posts in 3377 transaction days. When the RMSE of the prediction was compared, the RMSE using only transaction data was 2.731571, 2.62012 using only technical indicators, 2.775747 using only sentiment index, and 2.696610 combining various data sources. The prediction's RSME was compared to other similar approaches, and LSTM was shown to be the best accurate model.

Table 2.1: Summary of Related Works

S/N	Author(s) (year)	Title	Methods	Results	Limitations
1	Saud and Shakya (2019)	Analysis of lookback period for stock prediction with RNN variants: A case study on banking sector of NEPSE	VRNN, LSTM, and GRU	The average MAPE obtained with GRU for NIB and NABIL prediction is 4.74 and 4.71, and LSTM was 5.58 and 5.06 respectively	More technical indicators can be included for better prediction

2	Bhattacharjee and Bhattacharja (2019)	Stock Price Prediction: A comparative Study between Traditional Statistical Approach and Machine Learning Approach	SMA, WMA, Exponential Smoothing, Naïve Approach, SLR, RRR, Lasso Regression, KNN, Random Forest, SVM, SLP	MAPE values for statistical methods ranges within 6.5 to 1.7 approximately for Tesla and 4.7 to 1.3 approximately for Apple, MAPE values for machine learning range between 1.13 to 0.69 approximately for Tesla and 1.09 to 0.7 approximately for Apple.	Statistical methods have prediction accuracy.
3	Shen and Shafiq (2020)	Short-term stock market price trend prediction	Feature engineering using FE +	The accuracy of 5 principal components was 89.03%,	Focused more on short-term stock market

		using a comprehensive deep learning system	RFE + PCA and LSTM	10 principal components was 89.35%, 15 principal components was 89.39%, 20 principal components was 89.30%, 29 principal components was 90.29%	price trend prediction
4	Wen, Lin, and Nie (2020)	Research of stock price prediction based on PCA-LSTM model	PCA-LSTM	The model achieved a higher accuracy with 0.221 RMSE and 1.667% MAPE compared to other models	The dataset used is too small
5	Banerjee, Dabeeru, and Lavanya (2020)	Stock Market Prediction	Simple Linear Regression (SLR),	SLR accuracy of the prediction is 58%, MLR	The accuracy is too low.

			Multiple Linear Regression (MLR), MLR with interactions, PCA and SVM	accuracy of the prediction is 69%.	
6	Pramod and Mallikarjuna (2020)	Stock Price Prediction using LSTM	LSTM	The loss rate is 0.0024	A greater number of data sets will increase the prediction accuracy.
7	Singh, Gutta, and Hadaeghi (2021)	Stock Prediction Using Machine Learning	RNN, LSTM	The RSME value using LSTM is 0.43	No web application.
8	Guo (2022)	Stock Price Prediction Using Machine Learning	ARIMA, LSTM, GARCH	The average error rate for the prediction using ARIMA is 3.19%, using	ARIMA and GARCH have low prediction accuracy

				GARCH is 1.65%, using LSTM is 0.40%, and using Mixed Model is 0.27%	
9	Xiao and Su (2022)	Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average	ARIMA, LSTM	The RMSE using ARIMA is 0.043788, using LSTM is 0.028828	The noises could not be covered by the hybrid model.
10	Wu, Liu, Zou, and Weng (2022)	S_I_LSTM: stock price prediction based on multiple data sources and	S_I_LSTM	The RMSE using S_I_LSTM is 2.696610	Less consideration in training labelled data.

		sentiment analysis			
--	--	-----------------------	--	--	--

CHAPTER THREE

METHODOLOGY

3.0 Introduction

This chapter explains the methods that were implemented to develop this project. It also describes how the data is collected and the algorithms implemented during this project.

3.1 Data Collection

Initially proposed the data collection is from Kaggle. To get a more up-to-date data, the data of Apple for around 10 years has been collected from yahoo finance. The data contains information about the stock such as High, Low, Open, Close, Adjacent close and Volume. The Close column has been selected to train and test my model. Figure 3.1 shows the dataset collected.

	Open	High	Low	Close	Adj Close	Volume
Date						
2013-01-02	19.779285	19.821428	19.343929	19.608213	16.837112	560518000
2013-01-03	19.567142	19.631071	19.321428	19.360714	16.624596	352965200
2013-01-04	19.177500	19.236786	18.779642	18.821428	16.161528	594333600
2013-01-07	18.642857	18.903570	18.400000	18.710714	16.066452	484156400
2013-01-08	18.900356	18.996071	18.616072	18.761070	16.109701	458707200
...
2023-02-28	147.050003	149.080002	146.830002	147.410004	147.410004	50547000
2023-03-01	146.830002	147.229996	145.009995	145.309998	145.309998	55479000
2023-03-02	144.380005	146.710007	143.899994	145.910004	145.910004	52238100
2023-03-03	148.039993	151.110001	147.330002	151.029999	151.029999	70668500
2023-03-06	153.789993	156.300003	153.460007	153.830002	153.830002	87410100

Figure 3.1: Collected Data

The data has been split into training and testing set using ratio 60:40 respectively.

3.2 Existing System Analysis

Singh, Gutta, & Hadaeghi. (2021) proposed a system for predicting future price of a stock using machine learning model. The data for three companies from different sectors from date 1/1/2012 to 10/10/2021 was collected from yahoo finance. The datasets collected includes data from Tech company, Banking sector and Food service to understand movement of stock prices in different sectors. The data was divided into training and testing segments using an 80:20 ratio where 80% was for training and 20% was for testing and the data was scaled ranging the data from 0 to 1. The RNN (Recurrent neural network) was initialised and the LSTM model was built, the model was trained by providing batch size and epochs. The testing data was converted to a NumPy array and RMSE (root mean squared error) was used to evaluate the model. The architecture is shown in figure 3.2

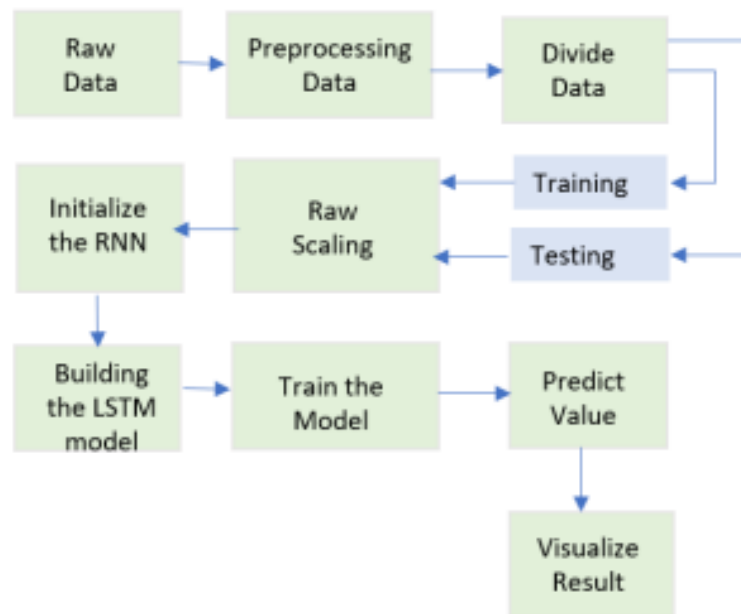


Figure 3.2: Architecture of Existing System (Source: Singh, Gutta, & Hadaeghi, 2021)

3.3 Architecture of the Proposed System

The architecture is divided into three modules: preprocessing, training and testing.

Figure 3.3 shows the architecture of the proposed system.

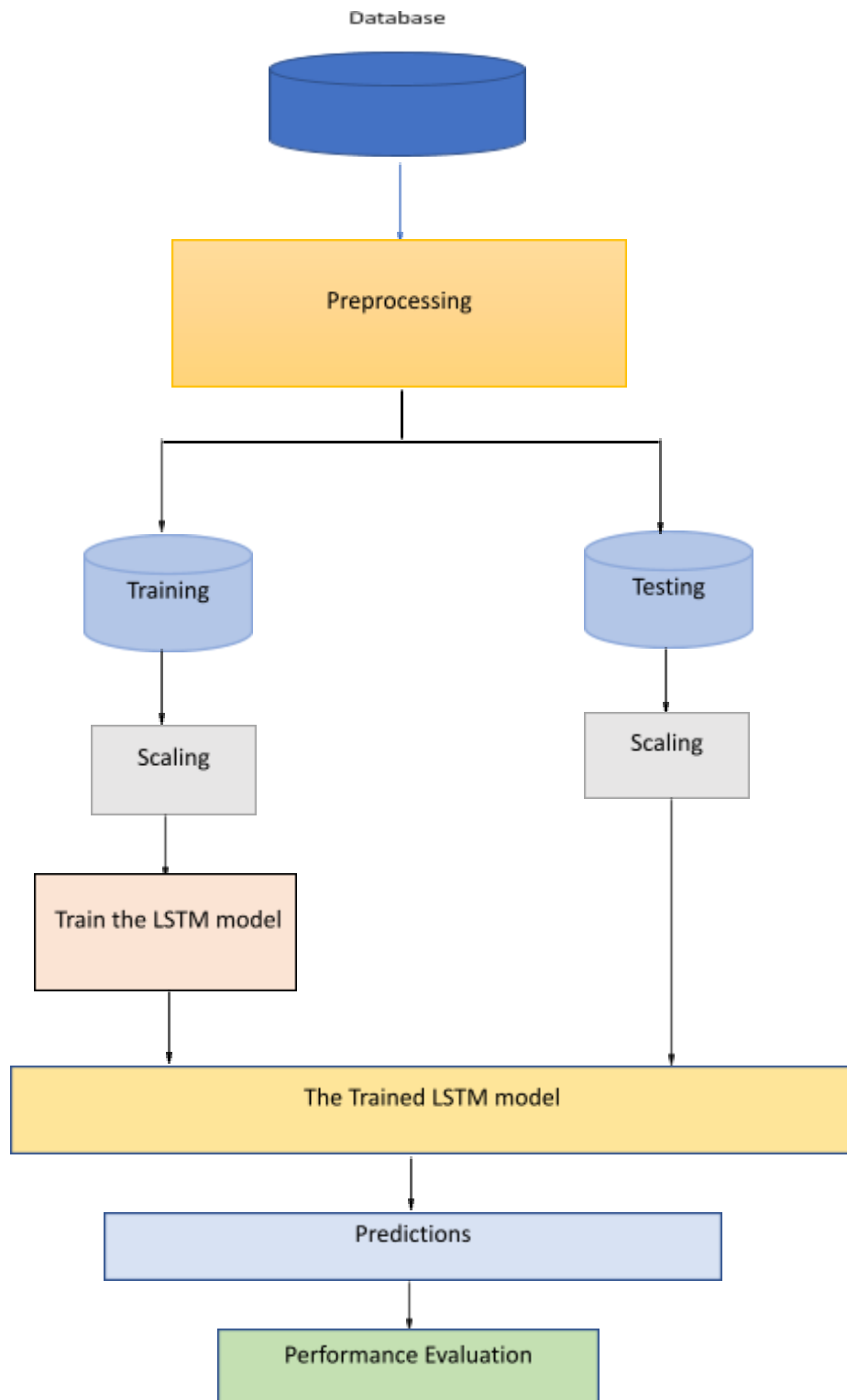


Figure 3.3: Architecture of Proposed system

3.3.1 Preprocessing Module

The dataset is pre-processed in order for the algorithm to easily learn patterns from it. It is used to increase the model's accuracy and interpretability while lowering the time and resources necessary to train the model and preventing overfitting. Checking for null values, partitioning the dataset, and normalisation are all part of this process.

Data Cleaning

Identifying and handling missing, erroneous or inconsistent data to ensure the data quality and integrity. The missing values are being handled by deleting rows or columns if the missing values are limited, however this is done cautiously considering the impact on the overall dataset.

Data Normalization

It converts a dataset's numerical features into a common scale. Min-Max Scaling reduces the data to a set range of 0 to 1, preserving the original distribution while ensuring that all values fall within the range. Normalisation reduces all data features to a common scale, preventing specific features from dominating the learning process due to their greater magnitude. It encourages fair comparisons and improves the dependability and interpretability of outcomes (geeksforgeeks, 2021)

Train-Test Split

The dataset is then divided into two sections, one for training the model and one for evaluating its performance. This is done to determine how well the model generalises previously unknown data. During the learning process, the training set is used to update the model's weights and biases, whereas the test set is used to evaluate the

model's performance and estimate its capacity to generate predictions on fresh unknown data (IBM, 2023).

3.3.2 Training Module

The machine learning model is long short-term memory.

Long Short-Term Memory

LSTM networks are a recurrent neural network (RNN) modification that was designed to handle instances where RNNs fail. It has been intended to almost fully eliminate the vanishing gradient problem while leaving the training model unchanged. Because LSTMs provided us with a wide range of parameters such as learning rates and input and output biases, there was no need for precise tuning (geeksforgeeks, 2021). Figure 3.4 shows the hidden layers of LSTM.

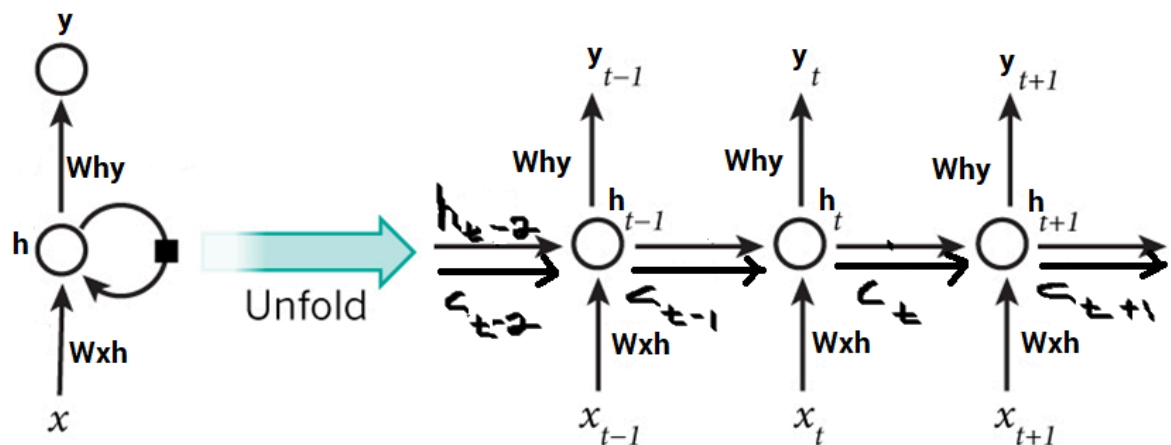


Figure 3.4: Hidden layers of LSTM (Source: geeksforgeeks, 2021)

The LSTM hidden layer is a gated unit or gated cell. It is composed of four layers that interact to create the cell's output as well as the cell state. These two items are subsequently passed on to the following concealed layer. Three logistic sigmoid gates and one tanh layer make up an LSTM. Gates are used to limit the quantity of data that

travels through the cell. They determine which information is required by the next cell and which is rejected. The result is often in the 0-1 range, with 0 indicating reject all and 1 indicating include all (geeksforgeeks, 2021).

3.4 Performance Metrics

They are used to tell if the programmer is making progress and put a number on it. Using it yields better performance from the model. The metrics for this research are Mean Absolute Error, Mean Squared Error, and Root Mean Squared Error.

Mean Absolute Error (MAE)

It is the simplest to grasp because it is an average inaccuracy. It computes the average absolute difference between predicted and actual values. It gives you a straightforward and understandable a measurement of prediction accuracy. Lower MAE values indicate higher performance (ResearchGate, 2023).

$$\frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (3.1)$$

Mean Squared Error (MSE)

It computes the square of the mean of the expected and original values. It is more common than MAE because it penalises greater faults that are more beneficial in practise. It emphasises bigger prediction mistakes, making it susceptible to outliers. Lower MSE values suggest higher performance (ResearchGate, 2023).

$$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (3.2)$$

Root Mean Squared Error (RMSE)

It is expressed in the same units as the target variable and offers a measure of the standard deviation of the prediction errors. Lower values of RSME indicate better performance (ResearchGate, 2023).

$$3.5 \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

(3.3)Predictions

The trained LSTM model is used to make prediction on unseen data. The input sequence for the next time step is prepared and passed through the model to obtain the predicted stock price. The process is repeated subsequent time steps to generate a sequence of future price predictions.

3.6 UML Diagrams

UML diagrams are visual representations of software programmes. Grady Booch, James Rumbaugh, Ivar Jacobson, and Rational Software Corporation created this notation particularly for object-oriented design. UML's scope has grown over time and it is now used in a variety of software engineering projects. Today, the Object Management Group (OMG) recognises UML as the standard for modelling software development (smartdraw, 2023).

3.6.1 Use Case Diagram

In this use case diagram, there are two primary actors, the user and the system. The user can interact with the system through two use cases Enter Data and View Results. The Enter Data use case allows the user to input relevant data for stock price

prediction into the system and the View Results use case enables the user to view the predicted stock prices.

The system actor performs two essential use cases Predicted Prices and Save Results. The Predicted Prices use case represents the system's ability to analyse the entered data and generate stock price predictions, the Save Results use case indicates the system's capability to store the predicted results for reference or analysis. Figure 3.5 shows the use case diagram.

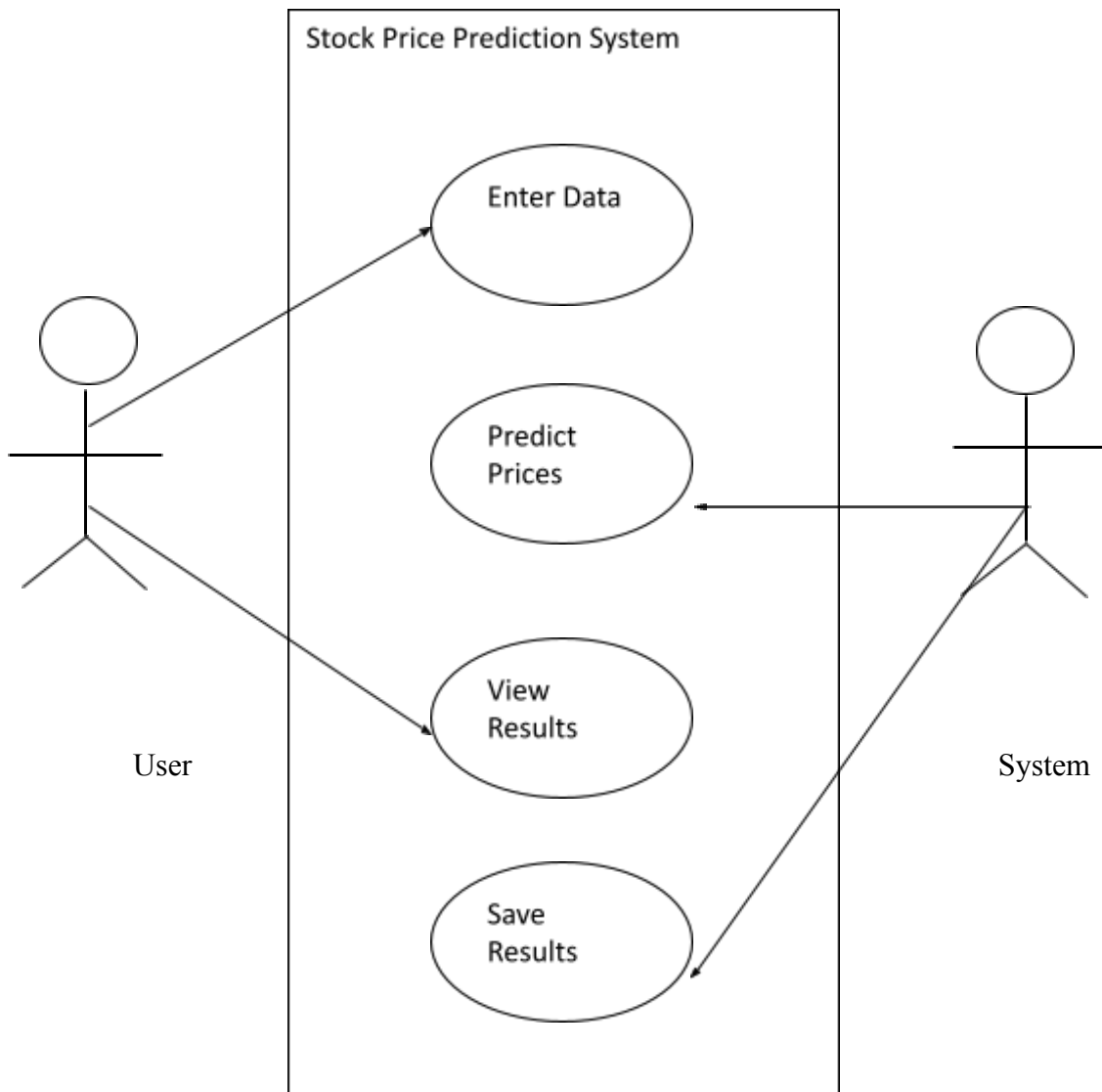


Figure 3.5: Use Case Diagram of the System

3.6.2 Activity Diagram

The process starts with Login to Application then proceeds to the Start Prediction activity, representing the initiation of the stock prices prediction process, it then proceeds Collect Data activity where it gathers the necessary data required for the prediction. Once the data is collected it moves to the Analyse Data activity, where it performs analysis and computations on the collected data. After analysis the system enters the Generate Prediction activity where it uses the analysed data to generate the stock price prediction and finally is prediction is displayed to the user through the Display Prediction activity. Figure 3.6 shows the activity diagram.

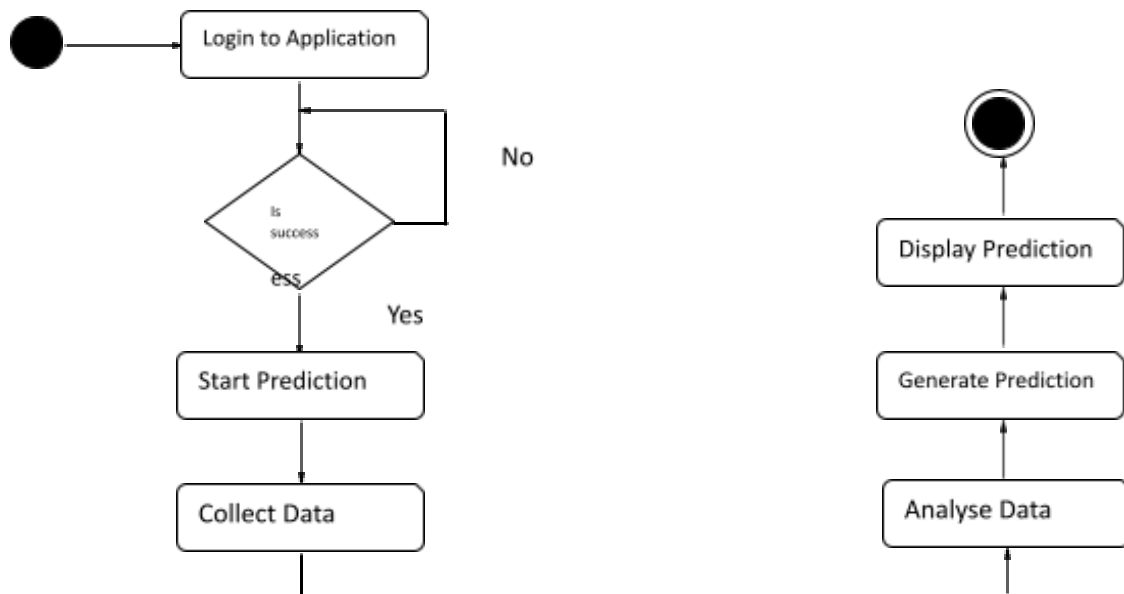


Figure 3.6: Activity Diagram of the System

CHAPTER FOUR


SYSTEM IMPLEMENTATION AND TESTING

4.0 Introduction

This chapter talks about the system implementation and testing

4.1 Data Analysis

Data analysis examines, cleans, transforms, and interprets data to extract meaningful insights and make informed decisions. It involves various techniques and tools to improve your comprehension of the data and its underlying trends. Figure 4.1 shows the collected data.



Date	Open	High	Low	Close	Adj Close	Volume	company_name
2022-07-27	152.580002	157.330002	152.160004	156.789993	155.859329	78620700	APPLE
2022-07-28	156.979996	157.639999	154.410004	157.350006	156.416000	81378700	APPLE
2022-07-29	161.240005	163.630005	159.500000	162.509995	161.545380	101786900	APPLE
2022-08-01	161.009995	163.589996	160.889999	161.509995	160.551315	67829400	APPLE
2022-08-02	160.100006	162.410004	159.630005	160.009995	159.060196	59907000	APPLE
...
2023-07-21	131.339996	131.369995	128.419998	130.000000	130.000000	133265000	AMAZON
2023-07-24	130.309998	131.660004	128.350006	128.800003	128.800003	45591100	AMAZON
2023-07-25	129.309998	129.580002	128.529999	129.130005	129.130005	39236700	AMAZON
2023-07-26	126.510002	129.080002	126.110001	128.149994	128.149994	53789900	AMAZON
2023-07-27	131.000000	132.630005	130.679993	132.339996	132.339996	21140962	AMAZON

Figure 4.1: Collected Data

Figure 4.2, figure 4.3, figure 4.4 and figure 4.5 show the info about each of the stock data Amazon, Microsoft, Google and Apple respectively

```

<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 250 entries, 2022-08-01 to 2023-07-28
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Open            250 non-null   float64
1   High            250 non-null   float64
2   Low             250 non-null   float64
3   Close           250 non-null   float64
4   Adj Close       250 non-null   float64
5   Volume          250 non-null   int64
6   company_name    250 non-null   object
dtypes: float64(5), int64(1), object(1)
memory usage: 15.6+ KB

```

Figure 4.2: Stock information of Amazon

```

<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 250 entries, 2022-08-01 to 2023-07-28
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Open            250 non-null   float64
1   High            250 non-null   float64
2   Low             250 non-null   float64
3   Close           250 non-null   float64
4   Adj Close       250 non-null   float64
5   Volume          250 non-null   int64
6   company_name    250 non-null   object
dtypes: float64(5), int64(1), object(1)
memory usage: 15.6+ KB

```

Figure 4.3: Stock information of Microsoft

```

<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 250 entries, 2022-08-01 to 2023-07-28
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Open            250 non-null   float64
1   High            250 non-null   float64
2   Low             250 non-null   float64
3   Close           250 non-null   float64
4   Adj Close       250 non-null   float64
5   Volume          250 non-null   int64
6   company_name    250 non-null   object
dtypes: float64(5), int64(1), object(1)
memory usage: 15.6+ KB

```

Figure 4.4: Stock Information of Google

```

<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 250 entries, 2022-08-01 to 2023-07-28
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Open            250 non-null   float64
1   High            250 non-null   float64
2   Low             250 non-null   float64
3   Close           250 non-null   float64
4   Adj Close       250 non-null   float64
5   Volume          250 non-null   int64
6   company_name    250 non-null   object
dtypes: float64(5), int64(1), object(1)
memory usage: 15.6+ KB

```

Figure 4.5: Stock Information of Apple

The statistical info about each of the stock data Amazon, Microsoft, Google and Apple are being shown in figure 4.6, figure 4.7, figure 4.8, and 4.9 respectively

	Open	High	Low	Close	Adj Close	Volume
count	251.000000	251.000000	251.000000	251.000000	251.000000	2.510000e+02
mean	110.176056	111.827450	108.490877	110.158685	110.158685	6.678457e+07
std	16.538930	16.678758	16.511643	16.550138	16.550138	2.619729e+07
min	82.800003	83.480003	81.430000	81.820000	81.820000	2.826480e+07
25%	96.035000	97.544998	94.195000	96.259998	96.259998	5.093825e+07
50%	105.260002	106.790001	104.330002	105.660004	105.660004	5.953410e+07
75%	125.474998	127.385002	124.220001	126.195000	126.195000	7.298745e+07
max	143.910004	146.570007	142.000000	144.779999	144.779999	2.231334e+08

Figure 4.6: Statistical Information about Amazon

	Open	High	Low	Close	Adj Close	Volume
count	252.000000	252.000000	252.000000	252.000000	252.000000	2.520000e+02
mean	158.262421	160.090595	156.715476	158.500555	158.133178	7.069926e+07
std	17.186132	16.990510	17.532395	17.282006	17.417059	2.334023e+07
min	126.010002	127.769997	124.169998	125.019997	124.656982	1.623975e+07
25%	145.817505	147.360004	144.072502	145.924995	145.488323	5.402915e+07
50%	154.719994	157.095001	153.360001	155.154999	154.694229	6.772575e+07
75%	170.209999	171.330002	168.829998	170.402500	169.768894	8.140258e+07
max	196.020004	198.229996	195.449997	196.250000	196.250000	1.647624e+08

Figure 4.7: Statistical Information about Microsoft

	Open	High	Low	Close	Adj Close	Volume
count	251.000000	251.000000	251.000000	251.000000	251.000000	2.510000e+02
mean	105.375454	106.867988	104.210036	105.545677	105.545677	2.707043e+07
std	11.841132	11.918700	11.863595	11.891424	11.891424	1.110734e+07
min	85.510002	86.550003	83.449997	83.489998	83.489998	8.567800e+06
25%	95.759998	97.349998	94.470001	95.840000	95.840000	2.064670e+07
50%	102.879997	104.220001	101.860001	103.629997	103.629997	2.424990e+07
75%	116.625000	118.350002	116.315002	117.599998	117.599998	3.024690e+07
max	131.800003	133.600006	129.179993	129.869995	129.869995	9.779860e+07

Figure 4.8: Statistical Information about Google

	Open	High	Low	Close	Adj Close	Volume
count	251.000000	251.000000	251.000000	251.000000	251.000000	2.510000e+02
mean	273.603705	276.788725	270.703426	273.852310	272.799603	2.923583e+07
std	36.921937	37.034596	36.800666	36.954404	37.434146	1.091271e+07
min	217.550003	220.410004	213.429993	214.250000	212.649246	9.200800e+06
25%	243.160004	245.305000	240.264999	242.514999	241.041595	2.249105e+07
50%	261.690002	266.480011	260.290009	263.100006	261.870209	2.669820e+07
75%	296.335007	302.385010	293.990005	299.884995	299.230865	3.277870e+07
max	361.750000	366.779999	352.440002	359.489990	359.489990	8.610200e+07

Figure 4.9: Statistical Information about Apple

4.2 Visualisation of the Stock Data

Data visualisation is the graphical representation of information and data. It is a potent tool that allows us to understand complex data patterns, identify trends, and communicate insights effectively. In this short note, we explore the significance and benefits of data visualisation.

Insightful Communication: Visualisation transforms raw data into easily understandable charts, graphs, and plots. By presenting information visually, it

enables clear and concise communication of complex findings to both technical and non-technical audiences. Visuals have the advantage of being more engaging, memorable, and impactful than raw numbers or text.

Pattern Recognition: Visualisation helps us recognise patterns, trends, and outliers within datasets that might otherwise be challenging to identify. It provides a visual context that aids in understanding the relationships between variables, enabling data-driven decision-making.

Figure 4.10 and figure 4.11 show the visualisation of the closing price of each of the stock data

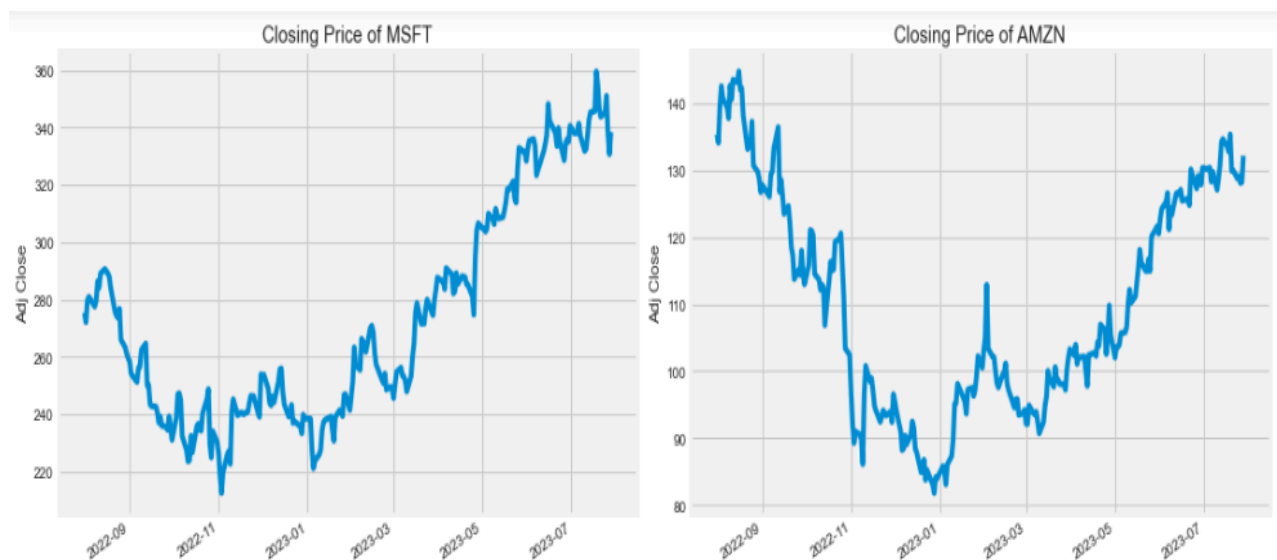


Figure 4.10: Closing Price of Microsoft and Amazon

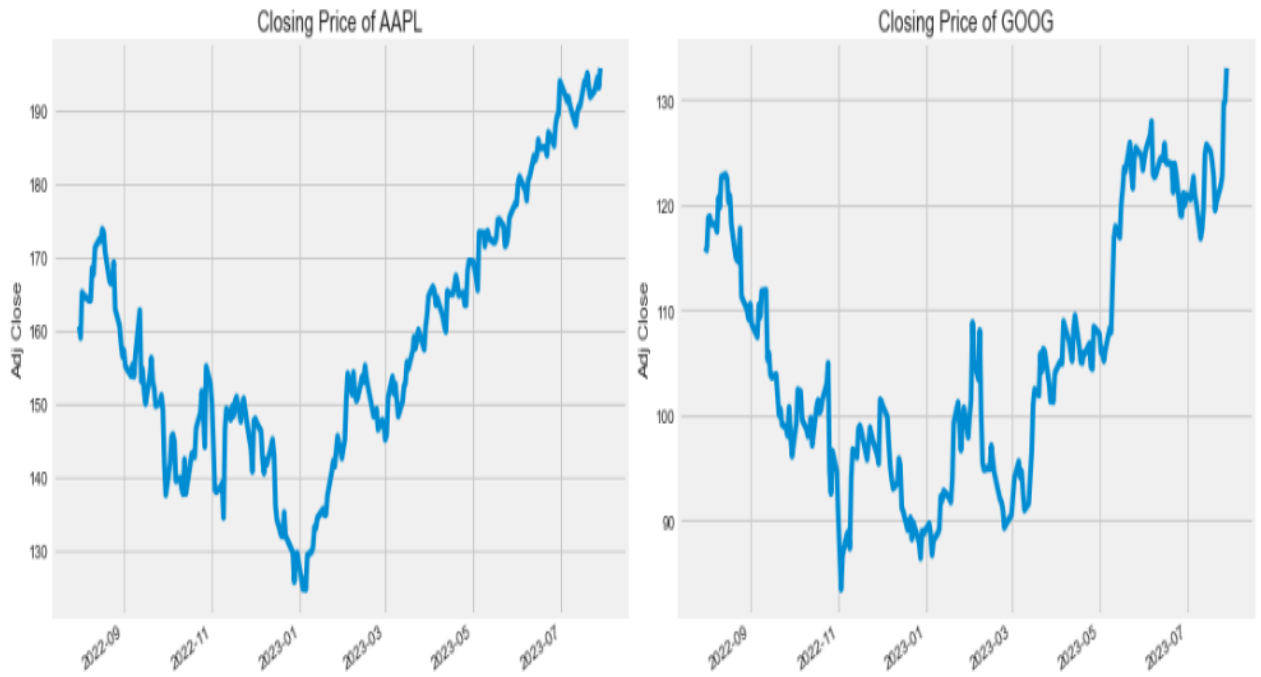


Figure 4.11: Closing Price of Apple and Google

Figure 4.12 and figure 4.13 show the visualisation of the sales volume of each of the stock data

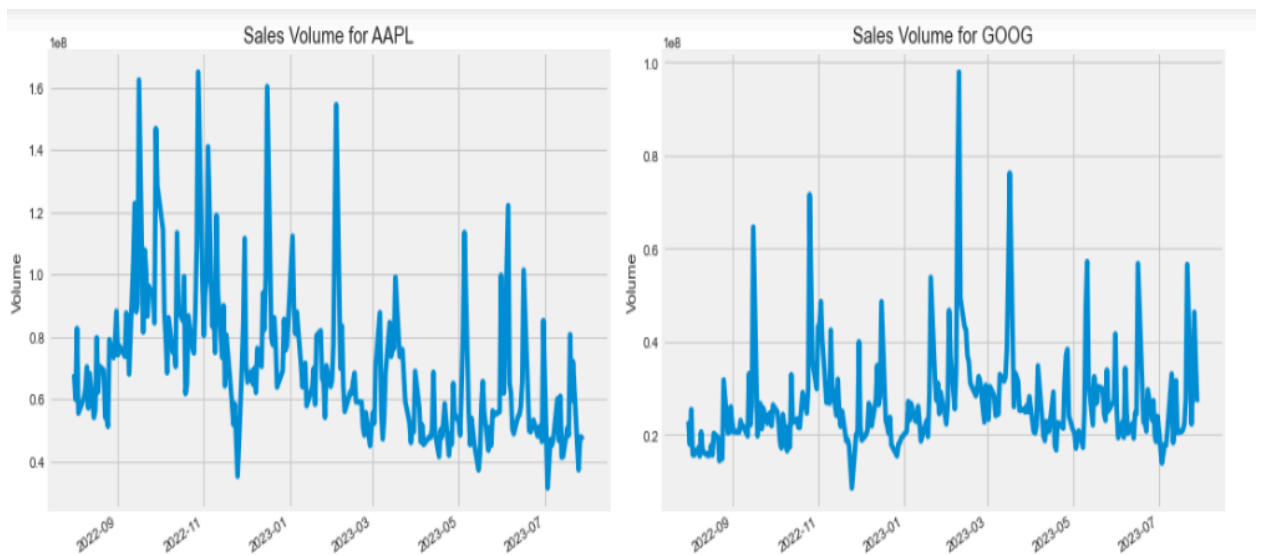


Figure 4.12: Visualisation of the sales volume for apple and google

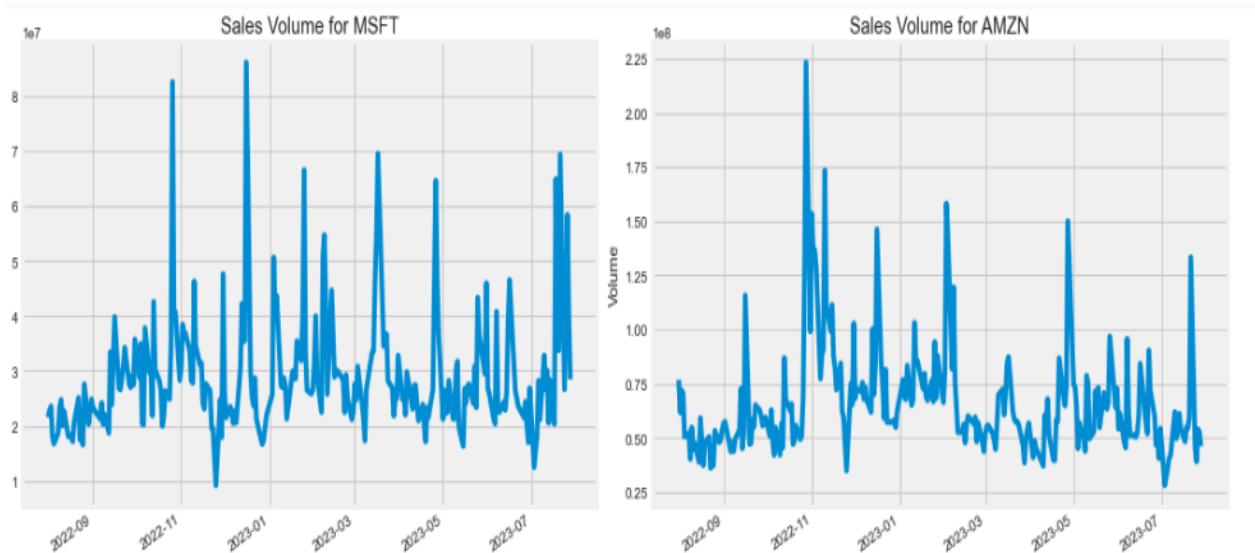


Figure 4.13: Visualisation of the sales volume for Microsoft and amazon

4.3 Feature Engineering

Feature engineering is a critical process in data analysis and machine learning. It involves selecting, transforming, and creating relevant features from raw data to improve the performance and interpretability of predictive models.

A sample dataset set was used, Figure 4.14 shows the apple dataset.

Out[18]:

	Open	High	Low	Close	Adj Close	Volume
Date						
2012-01-03	14.621429	14.732143	14.607143	14.686786	12.482926	302220800
2012-01-04	14.642857	14.810000	14.617143	14.765714	12.550011	260022000
2012-01-05	14.819643	14.948214	14.738214	14.929643	12.689337	271269600
2012-01-06	14.991786	15.098214	14.972143	15.085714	12.821991	318292800
2012-01-09	15.196429	15.276786	15.048214	15.061786	12.801659	394024400
...
2023-07-24	193.410004	194.910004	192.250000	192.750000	192.750000	45377800
2023-07-25	193.330002	194.440002	192.919998	193.619995	193.619995	37283200
2023-07-26	193.669998	195.639999	193.320007	194.500000	194.500000	47471900
2023-07-27	196.020004	197.199997	192.550003	193.220001	193.220001	47460200
2023-07-28	194.669998	196.630005	194.139999	195.830002	195.830002	48254600

Figure 4.14: Apple dataset

4.3.1 Creating a new dataframe with close column

Figure 4.15 shows the codes used to create the dataframe, figure 4.16 shows the importation of the MinMaxScaler and figure 4.17 shoes the creation of the training data.

```
In [20]: # Create a new dataframe with only the 'Close column'
data = df.filter(['Close'])
# Convert the dataframe to a numpy array
dataset = data.values
# Get the number of rows to train the model on
training_data_len = int(np.ceil( len(dataset) * .95 ))

training_data_len
```

Out[20]: 2766

Figure 4.15: Creating dataframe with the close column

```
In [21]: from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(dataset)

scaled_data
```

Figure 4.16: Importing the MinMaxScaler

```
In [22]: # Create the training data set
# Create the scaled training data set
train_data = scaled_data[0:int(training_data_len), :]
# Split the data into x_train and y_train data sets
x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])
    if i<= 61:
        print(x_train)
        print(y_train)
        print()

# Convert the x_train and y_train to numpy arrays
x_train, y_train = np.array(x_train), np.array(y_train)

# Reshape the data
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
# x_train.shape
```

Figure 4.17: Creating the training dataset

4.3 Model Building

Model building is a fundamental process in data analysis and machine learning. It involves the creation, training, and evaluation of predictive or descriptive models using data to make informed decisions and extract valuable insights. In this research, we made use of LSTM (Long Short Term Memory) model in training the data.

The codes used to build the model are shown in the figure 4.18 and figure 4.19

```
In [23]: from keras.models import Sequential
from keras.layers import Dense, LSTM

# Build the LSTM model
model = Sequential()
model.add(LSTM(128, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(64, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
model.fit(x_train, y_train, batch_size=1, epochs=1)

2706/2706 [=====] - 128s 38ms/step - loss: 0.0012
```

Figure 4.18: Building the LSTM model

```
In [24]: # Create the testing data set
# Create a new array containing scaled values from index 1543 to 2002
test_data = scaled_data[training_data_len - 60: , :]
# Create the data sets x_test and y_test
x_test = []
y_test = dataset[training_data_len:, :]
for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])

# Convert the data to a numpy array
x_test = np.array(x_test)

# Reshape the data
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1 ))

# Get the models predicted price values
predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)

# Get the root mean squared error (RMSE)
rmse = np.sqrt(np.mean(((predictions - y_test) ** 2)))
rmse

5/5 [=====] - 9s 66ms/step
```

Out[24]: 7.2005115071545385

Figure 4.19: Creating the testing dataset

4.3.1 Result of Prediction

The result of the prediction is shown in figure 4.20



Figure 4.20: Result of Prediction

iv.4 Performance Evaluation

The performance metrics used to evaluate the performance of the model is Root Mean Squared Error which gave a result of 7.2 which is about 4.8 percentage error when compared with the stock values. It is one of the best performance metrics to evaluate a time series data.

iv.5 Application Development

Application development is the process of designing, building, and implementing software or hardware solutions to address specific needs and challenges in various domains. It involves a systematic approach to creating efficient, reliable, and user-friendly systems.

In this research, the application was developed using the following technologies, python, streamlit (a python framework), HTML and CSS. Visual studio code was

used as the software for developing the system. The figures 4.21 and 4.22 shows the interface of the application.

Stock Prediction App

Select stock

AAPL

Years of Prediction:

1

1

4

Loading data...done!

Raw Data

	Date	Open	High	Low	Close	Adj Close	Volume
2,150	2023-07-20 00:00:00	195.09	196.47	192.5	193.13	193.13	59,581,200
2,151	2023-07-21 00:00:00	194.1	194.97	191.23	191.94	191.94	71,917,800
2,152	2023-07-24 00:00:00	193.41	194.91	192.25	192.75	192.75	45,377,800
2,153	2023-07-25 00:00:00	193.33	194.44	192.92	193.62	193.62	37,283,200

Figure 4.21: Stock Prediction System



Figure 4.22: Time Series Data

Forecast Data

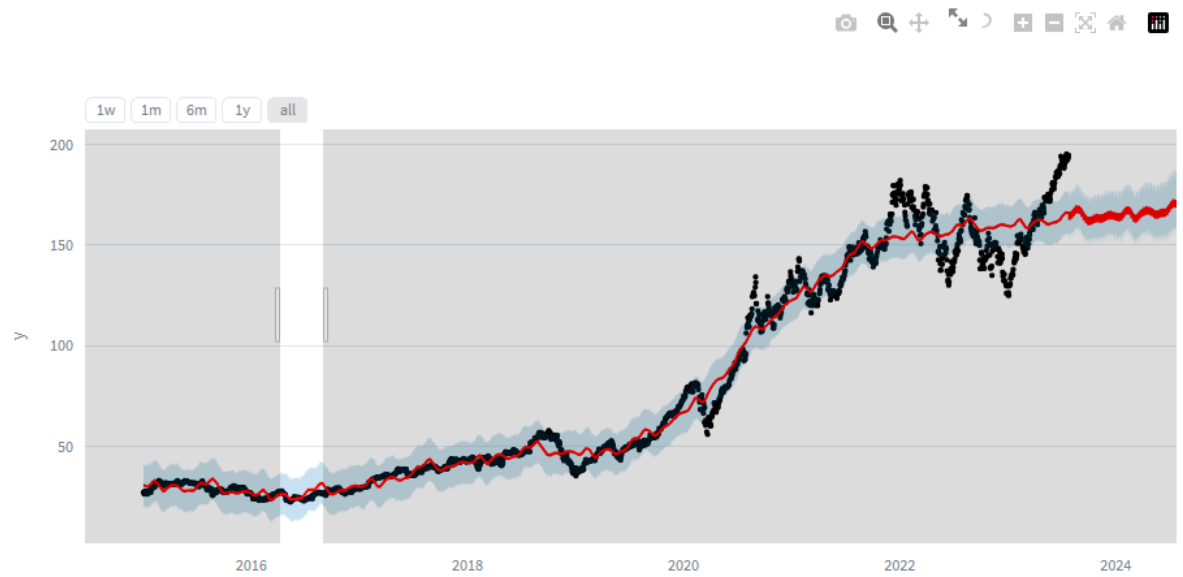


Figure 4.23: Forecast Data

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.0 Introduction

This chapter talks about the summary, conclusion, recommendations, future works and limitations of this study.

5.1 Summary

The summary of this project is that using the knowledge in artificial intelligence, as well as machine learning, techniques under machine learning have been reviewed and used in creating a model that predicts the future of some different stock markets such as Apple, Microsoft, Google and Amazon.

LSTM (long Short Term Memory) was the algorithm used in creating the model as well as the programming language used in writing the code was in python.

5.2 Conclusion

Under this research study, there is proper explanation about Artificial Intelligence (AI) as well as In-depth knowledge of machine learning and their various applications as to the content of this research work. We discussed extensively on supervised learning, unsupervised learning and Reinforcement learning, “Machine learning in Stock Market as a field of study”. We learnt about the Stock market as an investment and how to make use of machine learning techniques to create a model to predict the future of all these stocks for investment.

5.3 Limitation

Model training and updates: LSTM models require regular retraining to adapt to changing market conditions. Maintaining an up-to-date model in a web application can be resource intensive and time-consuming.

Uncertain market behaviour: Financial markets can be influenced by unforeseen events and unpredictable factors. LSTM models might struggle to capture these sudden changes, leading to less reliable predictions.

User expertise: Users of the web application might not have a deep understanding of machine learning or limitations of LSTM models. As a result, they might place too much reliance on the predictions, leading to potential financial risks.

5.4 Recommendations

User Authentication: Implement user authentication and secure data handling to protect users' data and maintain privacy.

Model Performance Improvement: Continuously fine-tune and optimise the LSTM model. Experiment with hyperparameters, different architectures, and alternative neural network approaches to enhance prediction accuracy.

Explain ability and Interpretability: Work on techniques to make the model's predictions more interpretable to users. This could involve providing insight into the factors influencing the predictions.

Sentiment Analysis: Consider integrating sentiment analysis of news articles or social media data related to stocks. Sentiments can impact stock prices and could enhance the accuracy of predictions.

Real-Time Updates: Develop a mechanism to provide real-time updates for stock prices and prediction, allowing users to stay informed about changes as they happen.

5.5 Future Works

Multivariate Time Series: Expand the model to handle multivariate time series data, incorporating additional features such as trading volume, technical indicators, and macroeconomic factors.

Portfolio Management: Extend the application to include features to users to manage and track their investment portfolios based on the predictions.

Explainable AI: Incorporate methods to explain the model's predictions, helping users understand the reasoning behind each forecast.

Multi-Asset Prediction: Expand the application to predict prices for other financial instruments such as commodities, cryptocurrencies, or foreign exchange.

Mobile App: Consider developing a mobile version of the application for users to access predictions and insights on the go.

REFERENCES

- Achelis, S. B. (1995). *Techincal Analysis from A to Z*.
- Anup, M., Md. Mahbubur, R., Al Almin, B., Md. Sahab, Z., & Sarnali, B. (2022, January). Stock Market Prediction: A Time Series Analysis. 390-400.
- Banerjee, S., Dabeeru, N., & Lavanya, R. (2020, July). Stock Market Prediction. *International Journal of Innnovative Technology and Engineering(IJITEE)*, 9(9).
- Bertoluzzo, M., Bortoli, L., & Delprete, C. (2020). Artificail intelligence and financial markets: Stock price prediction with a deep learning aproach with a deep learning approach. *Journal of Risk and Financial Management*.
- Bezdan, T., & Bacanin, N. (2019). *Convolutional Neural Network Layers and Architectures* (Vol. I).
- Bhattacharjee, I., & Bhattacharja, P. (2019). Stock Price Prediction: A comparative Study between Traditional Statistical Approach and Machine Learning Approach. *4th International Conference on Electrical Information and Communication Technology (EICT)*. doi:10.1109/EICT48899.2019.9068850
- Bolliber, J. (2001). *Bollinger on Bollinger Bands* (Vol. I).
- Bosco, J., & Khan, F. (2018). Stock Market Prediction and Efficiency Analysis using Recurrent Neural Network. *Stock Market*.
- Breiman, L. (2001). *Random Forests* (Vol. I).
- Brown, G. W., & Cliff, M. T. (2004). Investor Sentiment and the Near-Term Stock Market. *Journal of Empirical Finance*, I.
- CFI Team. (2022, November 28). *CFI*. Retrieved from CFI: <https://corporatefinanceinstitute.com/resources/wealth-management/stock-market>
- CFI Team. (2023, March 21). *CFI*. Retrieved from CFI: <https://corporatefinanceinstitute.com/resources/capital-markets/stock-price/#:~:text=What%20is%20Stock%20Price%3F%20The%20term%20stock%20price,ideally%20reflects%20the%20value%20of%20the%20company%20itself>.
- Chang, C., & C., L. (2011). *LIBSVM: A library for support vector machines*.
- Chong, E., & Han, H. (2019). Stock market prediction using machine learning algorithms. *Journal of Physics*, I.
- Dilallo, M. (2023, January 18). *The Motley Fool*. Retrieved from The Motley Fool: <https://www.fool.com/investing/stock-market/basics/crashes/#:~:text=When%20the%20>

20debt%20bubble%20burst%2C%20it%20caused%20the,the%20biggest%20single-day%20decline%20in%20stock%20market%20history.

Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *Journal of Finance*.

Frankenfield, j. (2022, july 6). *investopedia*. Retrieved from investopedia: <https://www.investopedia.com/terms/a/artificial-intelligence-ai-asp>

Friedman, J. (2001). *Greedy Function Approximation: A Gradient Boosting Machine*.

geeksforgeeks. (2021, june 25). Retrieved from geeksforgeeks: <https://www.geeksforgeeks.org/understanding-of-lstm-networks/>

Gomathy, C., Linganna, & Reddy, P. (2021, April). The Stock Market Prediction System. *International Research Journal of Engineering and Technology (IRJET)*, VIII(04).

Graham, B., & Dodd, D. L. (1934). *Security Analysis*.

Guo, Y. (2022). *Stock Price Prediction Using Machine Learning*.

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

Howard, A., Zhu, M., Chen, B., Lalenichenko, D., Wang, W., Weyand, T., . . . Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*.

IBM. (2023). *IBM*. Retrieved from IBM: <https://www.ibm.com/topics/machine-learning>

IBM Cloud Education. (2020, August 17). *IBM*. Retrieved from IBM: <https://www.ibm.com/in-en/cloud/learn/neural-networks>

Investopedia. (2023, January). *Investopedia*. Retrieved from Investopedia : <https://www.investopedia.com/terms/s/stockprice.asp>

Investopedia. (2023, January). *Investopedia* . Retrieved from Investopedia : <https://www.investopedia.com/terms/s/stock-price-prediction.asp>

Khedkar, S., Choudhari, J., & Dhok, J. (2019, May). A Survey on Stock Market Prediction Using Machine Learning Techniques. *Procedia Computer Science*, 1387-1394.

Khude, D. (2020, november 21). *Stock Market Library*. Retrieved from SAMCO: <https://www.samco.in/knowledge-center/articles/what-is-market-definition-and-types-of-stock-market/>

Madhavan, S., & Jones, T. (2021, January 24). *IBM Developer*. Retrieved from IBM Developer: <https://developer.ibm.com/articles/cc-machine-learning-deep-learning-architectures/>

- Malkiel, B. G. (1973). *A Random Walk Down Wall Street* (Vol. I).
- Malkiel, B. G. (2012). Returns from Investing in Equity Mutual Funds 1971 to 2011. *Journal of Investment Consulting*.
- Mishkin, F. S. (2019). *The Economics of Money, Banking and Financial Markets* (12th ed.). United Kingdom: Pearson Education.
- Mishra, A., & Sridhar, V. (2021). Can social media predict stock market movements? Evidence from an emerging market. *Journal of Business Research*, 494-508.
- Oracle. (2023). *Oracle*. Retrieved from Oracle: <https://www.oracle.com/ng/artificial-intelligence/machine-learning/what-is-machine-learning/>
- Pramod, B., & Mallikarjuna, S. P. (2020, June). Stock Price Prediction using LSTM. *Test Engineering and Management*, 83, 5246-5251.
- Refenes, A., Zaprains, A., & Francis, G. (1993). *On the Predictability of Stock Prices: A Case for High and Low Prices*.
- ResearchGate. (2023). *ResearchGate*. Retrieved from https://www.researchgate.net/figure/Structural-representation-of-a-Deep-Neural-Network-with-three-hidden-Layers_fig1_342292618
- ResearchGate. (2023, May 29). *ResearchGate*. Retrieved from Deep Learning in Remote Sensing: A Review - Scientific Figure on ResearchGate.: https://www.researchgate.net/figure/Architecture-of-AlexNet-as-shown-in-2_fig2_319955230
- Saud, A. S., & Shakya, S. (2019). Computer Intelligence and Data Science. *Procedia Computer Science* (pp. 788-798). Elsevier B. V.
- Shen, J., & Shafiq, M. O. (2020). Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of Big Data*.
- Singh, S. (2018, May 27). *Towards Data Science*. Retrieved from Towards Data Science: <https://towardsdatascience.com/cousins-of-artificial-intelligence-dda4edc27b55>
- Singh, S., Gutta, S., & Hadaeghi, A. (2021). Stock Price Prediction Using Machine Learning. *Computer Research*, 22.
- smartdraw. (2023). *smartdraw*. Retrieved June 7, 2023, from smartdraw: <https://www.smartdraw.com/uml-diagram/>
- Speed Trader. (2023). *speed trader*. Retrieved from speed trader.

- Strader, T. J., J., J., ROOT, T. H., & Huang, Y.-H. (. (2020). Machine Learning Stock Market Prediction Studies: Review and Research Directions. *Journal of International Technology and Information Management*, 28(4).
- Tupe-Waghmare, P. (2021). Prediction of Stocks and Stock Price using Artificial Intelligence: A Bibliometric Study using Scopus Database. *Library Philosophy Practice*.
- Vargas, F., Sant'Anna, A., Ruiz, D., & Merschmann, L. (2020). Stock price prediction: A systematic literature review. *Expert Systems with Applications*.
- Wen, Y., Lin, P., & Nie, X. (2020). Research of Stock Price Prediction Based on PCA-LSTM Model. *IOP Conference Series*. doi:10.1088/1757-899X/790/1/012109
- Wu, S., Liu, Y., Zou, Z., & Weng, T.-H. (2022). S_I_LSTM: stock price prediction based on multiple data sources and sentiment analysis. *Connection Science*, 44-62.
- Xiao, D., & Su, J. (2022). *Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average*. Central University of Finance and Economics. Beijing: Hindawi. Retrieved 2022

APPENDICES

Appendix A: Source Code for Long Short Term Memory

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
plt.style.use("fivethirtyeight")
%matplotlib inline
# For reading stock data from yahoo
from pandas_datareader.data import DataReader
import yfinance as yf
from pandas_datareader import data as pdr
import warnings
warnings.filterwarnings('ignore')
yf.pdr_override()
# For time stamps
from datetime import datetime
# The tech stocks we'll use for this analysis
tech_list = ['AAPL', 'GOOG', 'MSFT', 'AMZN']
# Set up End and Start times for data grab
tech_list = ['AAPL', 'GOOG', 'MSFT', 'AMZN']
end = datetime.now()
start = datetime(end.year - 1, end.month, end.day)
for stock in tech_list:
    globals()[stock] = yf.download(stock, start, end)
company_list = [AAPL, GOOG, MSFT, AMZN]
company_name = ["APPLE", "GOOGLE", "MICROSOFT", "AMAZON"]
for company, com_name in zip(company_list, company_name):
    company["company_name"] = com_name
df = pd.concat(company_list, axis=0)
df
```

```

AAPL.describe()

plt.figure(figsize=(15, 10))
plt.subplots_adjust(top=1.25, bottom=1.2)
for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Adj Close'].plot()
    plt.ylabel('Adj Close')
    plt.xlabel(None)
    plt.title(f'Closing Price of {tech_list[i - 1]}')
plt.tight_layout()

plt.figure(figsize=(15, 10))
plt.subplots_adjust(top=1.25, bottom=1.2)
for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Volume'].plot()
    plt.ylabel('Volume')
    plt.xlabel(None)
    plt.title(f'Sales Volume for {tech_list[i - 1]}')
plt.tight_layout()

ma_day = [10, 20, 50]
for ma in ma_day:
    for company in company_list:
        column_name = f'MA for {ma} days'
        company[column_name] = company['Adj Close'].rolling(ma).mean()

fig, axes = plt.subplots(nrows=2, ncols=2)
fig.set_figheight(10)
fig.set_figwidth(15)

AAPL[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50
days']].plot(ax=axes[0,0])
axes[0,0].set_title('APPLE')

GOOG[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50
days']].plot(ax=axes[0,1])
axes[0,1].set_title('GOOGLE')

```

```

MSFT[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50
days']].plot(ax=axes[1,0])
axes[1,0].set_title('MICROSOFT')
AMZN[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50
days']].plot(ax=axes[1,1])
axes[1,1].set_title('AMAZON')
fig.tight_layout()
for company in company_list:
    company['Daily Return'] = company['Adj Close'].pct_change()
# Then we'll plot the daily return percentage
fig, axes = plt.subplots(nrows=2, ncols=2)
fig.set_figheight(10)
fig.set_figwidth(15)
AAPL['Daily Return'].plot(ax=axes[0,0], legend=True, linestyle='--', marker='o')
axes[0,0].set_title('APPLE')
GOOG['Daily Return'].plot(ax=axes[0,1], legend=True, linestyle='--', marker='o')
axes[0,1].set_title('GOOGLE')
MSFT['Daily Return'].plot(ax=axes[1,0], legend=True, linestyle='--', marker='o')
axes[1,0].set_title('MICROSOFT')
AMZN['Daily Return'].plot(ax=axes[1,1], legend=True, linestyle='--', marker='o')
axes[1,1].set_title('AMAZON')
fig.tight_layout()
plt.figure(figsize=(12, 9))
for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Daily Return'].hist(bins=50)
    plt.xlabel('Daily Return')
    plt.ylabel('Counts')
    plt.title(f'{company_name[i - 1]}')
plt.tight_layout()
closing_df = pdr.get_data_yahoo(tech_list, start=start, end=end)['Adj Close']
# Make a new tech returns DataFrame

```

```

tech_rets = closing_df.pct_change()
tech_rets.head()
sns.jointplot(x='GOOG', y='GOOG', data=tech_rets, kind='scatter', color='seagreen')
# We'll use joinplot to compare the daily returns of Google and Microsoft
sns.jointplot(x='GOOG', y='MSFT', data=tech_rets, kind='scatter')
# We can simply call pairplot on our DataFrame for an automatic visual analysis
# of all the comparisons
sns.pairplot(tech_rets, kind='reg')
# Set up our figure by naming it returns_fig, call PairPLot on the DataFrame
return_fig = sns.PairGrid(tech_rets.dropna())
# Using map_upper we can specify what the upper triangle will look like.
return_fig.map_upper(plt.scatter, color='purple')
# We can also define the lower triangle in the figure, including the plot type (kde)
# or the color map (BluePurple)
return_fig.map_lower(sns.kdeplot, cmap='cool_d')
# Finally we'll define the diagonal as a series of histogram plots of the daily return
return_fig.map_diag(plt.hist, bins=30)
plt.figure(figsize=(12, 10))
plt.subplot(2, 2, 1)
sns.heatmap(tech_rets.corr(), annot=True, cmap='summer')
plt.title('Correlation of stock return')
plt.subplot(2, 2, 2)
sns.heatmap(closing_df.corr(), annot=True, cmap='summer')
plt.title('Correlation of stock closing price')
rets = tech_rets.dropna()
area = np.pi * 20
plt.figure(figsize=(10, 8))
plt.scatter(rets.mean(), rets.std(), s=area)
plt.xlabel('Expected return')
plt.ylabel('Risk')
for label, x, y in zip(rets.columns, rets.mean(), rets.std()):

```

```

plt.annotate(label, xy=(x, y), xytext=(50, 50), textcoords='offset points', ha='right',
va='bottom',

               arrowprops=dict(arrowstyle='-', color='blue',
connectionstyle='arc3,rad=-0.3'))

# Get the stock quote
df = pdr.get_data_yahoo('AAPL', start='2012-01-01', end=datetime.now())

# Show tech data
df

plt.figure(figsize=(16,6))
plt.title('Close Price History')
plt.plot(df['Close'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.show()

# Create a new dataframe with only the 'Close column
data = df.filter(['Close'])

# Convert the dataframe to a numpy array
dataset = data.values

# Get the number of rows to train the model on
training_data_len = int(np.ceil( len(dataset) * .95 ))

training_data_len

from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(dataset)

scaled_data

# Create the training data set
# Create the scaled training data set
train_data = scaled_data[0:int(training_data_len), :]

# Split the data into x_train and y_train data sets
x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])

```

```

y_train.append(train_data[i, 0])
if i<= 61:
    print(x_train)
    print(y_train)
    print()

# Convert the x_train and y_train to numpy arrays
x_train, y_train = np.array(x_train), np.array(y_train)

# Reshape the data
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
# x_train.shape
from keras.models import Sequential
from keras.layers import Dense, LSTM

# Build the LSTM model
model = Sequential()
model.add(LSTM(128, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(64, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
model.fit(x_train, y_train, batch_size=1, epochs=1)

# Create the testing data set
# Create a new array containing scaled values from index 1543 to 2002
test_data = scaled_data[training_data_len - 60: , :]
# Create the data sets x_test and y_test
x_test = []

```

```

y_test = dataset[training_data_len:, :]
for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])
# Convert the data to a numpy array
x_test = np.array(x_test)
# Reshape the data
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1 ))
# Get the models predicted price values
predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)
# Get the root mean squared error (RMSE)
rmse = np.sqrt(np.mean(((predictions - y_test) ** 2)))
rmse
# Plot the data
train = data[:training_data_len]
valid = data[training_data_len:]
valid['Predictions'] = predictions
# Visualize the data
plt.figure(figsize=(16,6))
plt.title('Model')
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')
plt.show()

```


Appendix B: Source Code for Streamlit

```
import streamlit as st
from prophet import Prophet
from datetime import date
import yfinance as yf
from prophet.plot import plot_plotly
from plotly import graph_objs as go
START = "2015-01-01"
TODAY = date.today().strftime("%Y-%m-%d")
st.title("Stock Prediction App")
stocks = ('AAPL', 'GOOG', 'MSFT', 'AMZN')
selected_stocks = st.selectbox("Select stock", stocks)
num_of_years = st.slider("Years of Prediction: ", 1, 4)
period = num_of_years * 365
@st.cache_data
def load_data(ticker):
    data = yf.download(ticker, START, TODAY)
    data.reset_index(inplace=True)
    return data
data_load_state = st.text("load data...")
data = load_data(selected_stocks)
data_load_state.text("Loading data...done!")
st.subheader('Raw Data')
st.write(data.tail())
def plot_raw_data():
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=data['Date'], y=data['Open'], name='stock_open'))
    fig.add_trace(go.Scatter(x=data['Date'], y=data['Close'], name='stock_close'))
    fig.layout.update(title_text="Time Series Data", xaxis_rangeslider_visible=True)
    st.plotly_chart(fig)
plot_raw_data()
```

```

df_train = data[['Date', 'Close']]
df_train = df_train.rename(columns={'Date':'ds','Close':'y'})
m = Prophet()
m.fit(df_train)
future = m.make_future_dataframe(periods=period)
forecast = m.predict(future)
st.subheader('Forecast data')
st.write(forecast.tail())
st.write("Forecast Data")
fig1 = plot_plotly(m,forecast)
new_color = 'red'
for trace in fig1['data']:
    trace['line']['color'] = new_color
fig1.update_layout(plot_bgcolor='white')
st.plotly_chart(fig1)

```