ROBUST BACKPROPAGATION NEURAL NETWORK USING DATE PALM SEED GROWTH ALGORITHM FOR STOCK MARKET PREDICTION

TENGKU NURUL AIMI BALQIS BINTI TENGKU MALIM BUSU

MASTER OF SCIENCE (STATISTICS)
UNIVERSITI UTARA MALAYSIA
2025

Awang Had Salleh
Graduate School
of Arts And Sciences

Universiti Utara Malaysia

PERAKUAN KERJA TESIS / DISERTASI
(Certification of thesis / dissertation)

Kami, yang bertandatangan, memperakukan bahawa
(We, the undersigned, certify that)

TENGKU NURUL AIMI BALQIS TENGKU MALIM BUSU

calon untuk Ijazah **MASTER OF SCIENCES**
(candidate for the degree of)

telah mengemukakan tesis / disertasi yang bertajuk:
(has presented his/her thesis / dissertation of the following title):

"ROBUST BACKPROPAGATION NEURAL NETWORK USING DATE PALM SEED GROWTH ALGORITHM FOR STOCK MARKET PREDICTION"

seperti yang tercatat di muka surat tajuk dan kulit tesis / disertasi.
(as it appears on the title page and front cover of the thesis / dissertation).

Bahawa tesis/disertasi tersebut boleh diterima dari segi bentuk serta kandungan dan meliputi bidang ilmu dengan memuaskan, sebagaimana yang ditunjukkan oleh calon dalam ujian lisan yang diadakan pada : **04 September 2024.**
That the said thesis/dissertation is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on: ***04 September 2024.***

Pengerusi Viva: *(Chairman for VIVA)*	**Assoc. Prof. Dr. Nazrina Aziz**	Tandatangan *(Signature)*
Pemeriksa Luar: *(External Examiner)*	**Assoc. Prof. Dr. Zamira Hasanah Zamzuri**	Tandatangan *(Signature)*
Pemeriksa Dalam: *(Internal Examiner)*	**Dr. Norhayati Yusof**	Tandatangan *(Signature)*
Nama Penyelia/Penyelia-penyelia: *(Name of Supervisor/Supervisors)*	**Dr. Saadi Ahmad Kamaruddin**	Tandatangan *(Signature)*
Nama Penyelia/Penyelia-penyelia: *(Name of Supervisor/Supervisors)*	**Assoc. Prof. Dr. Nor Aishah Ahad**	Tandatangan *(Signature)*

Tarikh:
(Date) ***04 September 2024***

Permission to Use

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences
UUM College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok

Abstrak

Rangkaian Neural Rambatan Balik (BPNN) merupakan salah satu model yang paling lazim digunakan bagi ramalan pasaran saham disebabkan kemampuannya sebagai penganggar universal. Namun begitu, proses latihan BPNN berasakan Kaedah Kuasa Dua Terkecil (OLS) cenderung menghasilkan anggaran pemberat yang tidak teguh apabila terdapat nilai terpencil dalam data. Akibatnya, prestasi ramalan model BPNN terjejas. Bagi menangani isu ini, kajian ini mencadangkan pendekatan alternatif dengan menggantikan OLS kepada algoritma Pertumbuhan Biji Kurma Kuasa Dua Median Terkecil (DPSG-LMedS). Pendekatan ini bertujuan untuk meningkatkan ketepatan ramalan pada tahap pencemaran data yang berbeza dalam pasaran saham. DPSG-LMedS melibatkan lima fasa, iaitu melatih rangkaian secara berulang dengan meminimumkan ralat median yang dianggarkan, membuang nilai terpencil berdasarkan sisihan piawai yang teguh, melatih semula menggunakan data yang telah disaring, dan menghentikan proses apabila ralat LMedS terbaik memenuhi kriteria yang ditetapkan. Seterusnya prestasi model dinilai menggunakan data simulasi dan data sebenar. Dalam analisis simulasi, ketepatan model baharu dinilai berdasarkan tahap pencemaran data yang berbeza (0% hingga 65%), konfigurasi lag input (5 hingga 45), dan nod tersembunyi (5 hingga 45). Data sebenar bagi harga penutupan pasaran saham FBM KLCI digunakan untuk membandingkan prestasi model baharu dengan BPNN dan BPNN bersama LMedS. Model dengan prestasi terbaik ditentukan berdasarkan nilai terendah bagi Ralat Punca Min Kuasa Dua (RMSE) dan Ralat Punca Min Kuasa Dua Geometrik (GRMSE). Keputusan daripada analisis simulasi menunjukkan bahawa model baharu berprestasi baik pada semua tahap pencemaran data, dengan konfigurasi lag input yang sederhana dan nod tersembunyi yang terendah. Perbandingan menggunakan data sebenar menunjukkan bahawa model baharu mengatasi prestasi model-model lain. Model baharu ini menawarkan model peramalan yang lebih dipercayai dan dijangka dapat menyokong pelabur, ahli ekonomi, pembuat dasar, serta institusi kewangan dalam membuat keputusan yang lebih tepat dan berinformasi. Selain itu, ia turut menyumbang kepada pembangunan teknik rangkaian neural yang lebih teguh bagi aplikasi ramalan kewangan.

Kata Kunci: Algoritma Pertumbuhan Biji Kurma, Kuasa Dua Median Terkecil, Nilai Terpencil, Rangkaian Neural Rambatan Balik Teguh, Ramalan Pasaran Saham.

Abstract

Backpropagation Neural Network (BPNN) is one of the most commonly used models for stock market prediction due to its ability as universal estimators. However, the Ordinary Least Squares (OLS)-based training in BPNN leads to non-robust weightage estimates in the presence of outliers. Consequently, it affects the prediction performance of the BPNN model. Addressing this issue, this study proposes an alternative approach by replacing OLS with Date Palm Seed Growth Least Median Squares (DPSG-LMedS) algorithm. This approach aims to improve the prediction accuracy at different levels of data contamination in stock market. DPSG-LMedS involve five phases which are training the network iteratively by minimizing the median of estimated errors, removing outliers based on robust standard deviation, retraining on the cleaned data, and stopping once the best LMedS errors meet the setting criteria. Next, the model performance is evaluated using simulated and real data. In simulation analysis, the accuracy of the new model is assessed based on different levels of data contamination (0% to 65%), input lags (5 to 45), and hidden node (5 to 45) configurations. Real data of FBM KLCI stock market closing prices is used to compared the performance of the new model with BPNN and BPNN with LMedS. The best-performing model is determined based on the lowest values of Root Mean Square Error (RMSE) and Geometric Root Mean Square Error (GRMSE). Results from simulated analysis shows that the new model performed well at all levels of data contamination with configuration moderate lags input and lowest hidden nodes. Comparison using real data indicate that the new model outperformed other models. This new model offers a more reliable predicting model and is expected to support investors, economists, policymakers, and financial institutions in making more accurate and informed decisions. Additionally, it contributes to the development of robust neural network techniques for financial prediction applications.

Keywords: Date Palm Seed Growth Algorithm, Least Median Square, Outliers, Robust Backpropagation Neural Network, Stock Market Prediction.

Acknowledgement

IN THE NAME OF ALLAH, THE MOST GRACIOUS AND THE MOST MERCIFUL.

First, praise be to Allah S.W.T. Blessings and greetings to the noble man Muhammad S.A.W, his family and friends. I'm so grateful, with His permission, grace, guidance and protection, I was finally able to complete this thesis with great success. I would like to express my sincere and deepest appreciation to everyone who was contributed in completing this thesis. It was a great honor to pursue my studies under the supervision of Dr. Saadi bin Ahmad Kamaruddin and Associate Professor Dr. Nor Aishah binti Ahad. Their continuous encouragement, advice, motivation, and patience in guiding me throughout the completion of this thesis are tremendous. Both have given me much inspiration and opened my vision for contributing to the body of knowledge.

In addition, I would like to thank my family for their continuous moral and emotional support. They are the source of strength for me to complete my studies. Furthermore, I am grateful to my postgraduate friends at UUM who have given their encouragement and brilliant ideas throughout my studies.

Last but not least, thanks to Universiti Utara Malaysia (UUM) for the support to carry out this research work in such a wonderful environment that makes it feels so comfortable and enjoyable to study at UUM. Millions of thanks to the Ministry of Education, for supporting this research under the Malaysia FRGS Grant Scheme -

FRGS/1/2021/STG06/UUM/02/4. Last but not least, may the master's work bring me closer to Him and my passion goal towards the eternal.

Table of Contents

Permission to Use ii
Abstrak iii
Abstract iv
Acknowledgement v
Table of Contents vii
List of Tables xi
List of Figures xii
List of Appendices xiii
List of Abbreviations xiv

CHAPTER 1 INTRODUCTION 1
1.1 Introduction 1
1.2 Research Background 1
1.3 Problem Statement 10
1.4 Research Questions 12
1.5 Research Objectives 13
1.6 Scope of the Study 13
1.7 Significance of the Research 14
1.8 Summary 15

CHAPTER 2 LITERATURE REVIEW 16
2.1 Introduction 16
2.2 Stock Market in Malaysia 17
2.3 Neural Network in Stock Market Prediction Globally 18
2.4 Backpropagation Neural Network 20
2.5 Backpropagation Neural Network in Stock Market Prediction with the Presents of Outliers 21
2.6 Outlier Percentage 24
2.7 Solving Techniques for Outliers' Problem 24
  2.7.1 Metaheuristic Approach 25

2.7.2 Least Median Square 26
2.8 Performance Validation of Time Series Neural Network models for Stock Market Prediction 27
2.9 Convergence Evaluation for Proposed Algorithm 29
2.10 Summary 30
CHAPTER 3 METHODOLOGY 32
3.1 Introduction 32
3.2 Data Background 33
3.2.1 Data Collection 35
3.2.2 Descriptive Analysis 35
3.2.3 Boxplot of outlier detection 40
3.3 Preprocessing 41
3.3.1 Data Normalization 42
3.3.2 Data Partitioning 42
3.4 Backpropagation Neural Network Modelling 48
3.4.1 Input Layer 49
3.4.2 Output Layer 49
3.4.3 Hidden Nodes 49
3.4.4 Data Preprocessing 51
3.4.5 Data Splitting 51
3.4.6 Design and Architecture 52
3.4.7 Training Algorithm 52
3.4.8 Time Lags 52
3.5 Backpropagation Neural Network (BPNN) for Stock Market Prediction 53
3.5.1 Least Median Squares (LMedS) 63
3.5.2 Date Palm Seed Growth Algorithm (DPSG) 63
3.5.3 DPSG-LMedS 64
3.5.4 Enhanced BPNN model using DPSG-LMedS 66
3.6 Evaluation of Prediction Model using Error Measure 69
3.6.1 Root Mean Square Error 69
3.6.2 Geometric Root Mean Square Error 70

3.7 Convergence Test 71

3.8 Summary 72

CHAPTER 4 DATA ANALYSIS AND DISCUSSION 73

4.1 Introduction 73

4.2 Descriptive Analysis 74

4.3 Outliers Detection 79

4.4 Performance Comparisons for three models 81

4.5 Enhanced Backpropagation Neural Network on Simulated Dataset 82

4.6 Convergence Test 88

4.7 Forecasting 89

4.8 Summary 90

CHAPTER 5 CONCLUSION 91

5.1 Review Summary 91

5.2 Limitation 93

5.3 Recommendation 93

REFERENCES 95

APPENDIX A PYTHON CODE FOR ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL 118

APPENDIX B PERFORMANCE OF THE BPNN, BPNN WITH LMEDS AND ENHANCED BPNN MODEL ON REAL DATA, FBM KLCI STOCK MARKET 126

APPENDIX C PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET I - RMSE TRAINING VALUE 132

APPENDIX D PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET I - RMSE TESTING VALUE 137

APPENDIX E PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET I - GRMSE TRAINING VALUE 141

APPENDIX F PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET I - GRMSE TESTING VALUE 145

APPENDIX G PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET II - RMSE TRAINING VALUE 149

APPENDIX H PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET II - RMSE TESTING VALUE 153

APPENDIX I PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET II - GRMSE TRAINING VALUE 157

APPENDIX J PERFORMANCE OF THE ENHANCED BACKPROPAGATION NEURAL NETWORK (BPNN) MODEL ON SIMULATED DATASET II - GRMSE TESTING VALUE 161

APPENDIX K CONVERGENCE TEST WITH DIFFERENT NUMBER OF LAG AND DIFFERENT NUMBER OF EPOCHS 165

List of Tables

Table 3.1 *Input Lags Outliers Percentage for Enhanced BPNN model on Datasets* 44
Table 3.2 *Function for Simulated Dataset* 46
Table 3.3 *Variables of input lags* 47
Table 3.4 *Partitioning Tabulation for FBM KLCI Data* 48
Table 3.5 *Error Measures to Assess the Forecasting Model Performance* 70
Table 4.1 *Descriptive Statistics for FBM KLCI data* 75
Table 4.2 *Data Partitioning for Training and Testing Sets for FBM KLCI data* 79
Table 4.3 *Percentage of outliers for different lags of FBM KLCI Dataset* 81
Table 4.4 *Comparisons of RMSE and GRMSE value for Three Different Models using FBM KLCI Stock Market Closing Price Dataset* 82
Table 4.5 *The comparison of RMSE and GRMSE value in the training phase using Data Set I* 83
Table 4.6 *The comparison of RMSE and GRMSE value in the testing phase using Data Set I* 84
Table 4.7 *The comparison of RMSE and GRMSE value in the training phase using Data Set II* 86
Table 4.8 *The comparison of RMSE and GRMSE value in the testing phase using Data Set II* 87
Table 4.9 *Prediction 1-step ahead, 2-step ahead, and 3-step ahead* 89

List of Figures

Figure 3.1. Research Flowchart 34
Figure 3.2. Illustration of Death Cross and Golden Cross in Stock Market Trends ..39
Figure 3.3 Example of Box-plot 41
Figure 3.4. Connectionist Feedforward Backpropagation of DPSG-LMedS model .46
Figure 3.5 Process of Neural Network Model 50
Figure 3.6. Backpropagation Neural Network 54
Figure 3.7. Information Flow in the Forward Path of a Two-Layer Neural Network57
Figure 3.8. Information Flow in the Backward Path of a Two-Layer Neural Network 60
Figure 3.9. Contribution of the enhanced algorithm 67
Figure 3.10. Enhanced Backpropagation Neural Network 68
Figure 3.11. Example of Convergence Result 71
Figure 4.1. FBM KLCI stock market closing prices from 2nd January 2018 to 30th December 2022 74
Figure 4.2. 50-day and 200-day Simple Moving Average..76
Figure 4.3. Daily return of FBM KLCI and bands at ± 119.64636 standard deviation 77
Figure 4.4. Subset splitting for FBM KLCI stock market closing prices 78
Figure 4.5. Outliers which exist in FBM KLCI Stock Prices for different lags 80
Figure 4.6. Convergence Test from different epoch 88

List of Appendices

Appendix A Python Code for Enhanced Backpropagation Neural Network (BPNN) Model 118
Appendix B Performance of the BPNN, BPNN with LMedS and Enhanced BPNN Model on Real Data, FBM KLCI stock market 126
Appendix C Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - RMSE Training value 132
Appendix D Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - RMSE Testing value 137
Appendix E Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - GRMSE Training value 141
Appendix F Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - GRMSE Testing value 145
Appendix G Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - RMSE Training value 149
Appendix H Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - RMSE Testing value 153
Appendix I Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - GRMSE Training value 157
Appendix J Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - GRMSE Testing value 161
Appendix K Convergence Test with different number of lag and different number of epochs 165

List of Abbreviations

ACO	Ant Colony Algorithm
AGA-BPNN	Adaptive Genetic Algorithm In The Backpropagation Neural Network
AI	Artificial Intelligence
ANN	Artificial Neural Network
ARIMA	Autoregressive Integrated Moving Average
BA	Bat Algorithm
BCO	Bee Colony Optimization Algorithm
BP	Backpropagation
BPNN	Backpropagation Neural Network
CMAC NN	Cerebellar Model Articulation Controller Neural Network
COA	Cuckoo Optimization Algorithm
COVID-19	Coronavirus Disease 2019
CS	Cuckoo Search
DANN	Dynamic Artificial Neural Network
DBN	Deep Belief Network
DL	Deep Learning
DNN	Deep Neural Network

DPSG	Date Palm Seed Growth
DPSG-LMedS	Date Palm Seed Growth Least Median Square
DRNN	Deep Recurrent Neural Network
ECS	Enhanced Cuckoo Search
EMD	Empirical Mode Decomposition
EMD-HW	Empirical Mode Decomposition and Holt-Winter
EMH	Efficient Market Hypothesis
ESG	Environmental, Social, and Governance
FA	Firefly Algorithm
FBM KLCI	Financial Times Stock Exchange (FTSE) Bursa Malaysia Kuala Lumpur Composite Index
FFNN	Feedforward Neural Network
FNN	Feedforward Neural Network
FT	Fourier Transform
FTSE	Financial Times Stock Exchange
GA	Genetic Algorithm
GARCH	Generalized Auto-Regressive Conditional Heteroskedasticity
GARCH-MLP	Generalized Autoregressive Conditional Heteroscedasticity MLP
GDP	Gross Domestic Product
GRMSE	Geometric Root Mean Square Error

GRU	Gated Recurrent Unit
HW	Holt-Winter
IMF	Intrinsic Mode Function
KLSE	Kuala Lumpur Stock Exchange
LMedS	Least Median Square
LSTM	Long Short-Term Memory
LTA	Least Trimmed Absolute Value
LTS	Least Trimmed Squares
LTSD	Least Trimmed Symmetry Distance
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MASE	Mean Absolute Scaled Error
MBB	Moving Block Bootstrap
MFNNs	Multilayered Feedforward Neural Networks
ML	Machine Learning
MLP	Multi-Layer Perceptron
MCO	Movement Control Order
MSE	Mean Square Error
NASDAQ	National Association of Securities Dealers Automated Quotations

NN	Neural Networks
OLS	Ordinary Least Squares
PSO	Particle Swarm Optimization
PSO-BPNN	Particle Swarm Optimization Optimized Backpropagation Neural Network
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
RSD	Robust Standard Deviation
S&P	Standard & Poor's
SMA	Simple Moving Average
SOS	Symbiotic Organisms Search
SOSFFNN	Symbiotic Organisms Search Feedforward Neural Network
SVM	Support Vector Machines
TSE	Tehran Stock Exchange
TheilU	Theil’s Ustatistic

CHAPTER 1
INTRODUCTION

1.1 Introduction

This chapter explains the research background, problem statement, research questions, research objectives, the scope of the study, and the significance of this research accordingly. There are altogether seven sections in this chapter. The research background of the study is presented in Section 1.2 and consists of stock market prediction focused on Financial Times Stock Exchange (FTSE) Bursa Malaysia Kuala Lumpur Composite Index (FBM KLCI) dataset. Section 1.3 highlights the problem statement of this research. Furthermore, the research questions and the research objectives are presented in Section 1.4 and Section 1.5 respectively. Then, Section 1.6 explains the scope and limitations of this research work. In Section 1.7, the significance of the research work is presented according to the research objectives. Last but not least, the summary of this chapter is presented in Section 1.8.

1.2 Research Background

Decreasing economy in Malaysia due to a lacks of initiatives is one of the hot problems. The Chief Economist Bank Muamalat Malaysia Bhd, Mohd Afzanizam states that the ringgit appears to be undergoing a technical correction as it rose relatively high in early January 2024 (BERNAMA, 2024). This problem shows that the economy in Malaysia is not good, and it is one of the reasons why investors need to learn how to invest in stock markets.

Bursa Malaysia, established in 1930 under the name Singapore Stockbrokers' Association, holds the distinction of being the first formal securities exchange. Over time, the exchange underwent several name changes. In 2004, it transitioned from the Kuala Lumpur Stock Exchange (KLSE) to Bursa Malaysia, driven by the goal of enhancing its customer-centric and market-oriented approach.

The exchange offers comprehensive services, including settlement, depository services, listing, exchange functions, and clearing operations, all of which are fully integrated. By the close of 2008, a fully electronic trading system had been introduced. According to the exchange's website (Kenton, 2020), around 900 firms are eager to utilize this system for fundraising through various business practices. Ensuring the accuracy of predicted values is crucial, emphasizing the significance of employing the most effective trading system.

Stock market forecasting presents a prominent and highly significant endeavor within the realm of economics. This challenge arises from the presence of outliers in stock market data. Predicting the behavior of the stock market is widely recognized as one of the most formidable tasks in this field (Jin et al., 2020). Al-Mashhadani et al. (2021) emphasize that stock price prediction is an extremely challenging task due to the complexity and numerous aspects involved. Furthermore, the stock market remains characterized by its volatility and dynamic nature, further complicating prediction efforts (Zhang et al., 2021). Consequently, stock market prediction poses a considerable dilemma for investors seeking to make informed decisions on where to allocate their funds for profitable returns (Gandhmal & Kumar, 2019).

Stock market prediction involves the utilization of input variables encompassing fundamental indicators, technical indicators, and external factors, as discussed by Kumar et al. in 2020. The first type of input variable is fundamental indicators including turnover, expenses, annual reports, assets and liabilities and income statements. The second type is technical indicators that include parameters like Open Price, Close Price, High Price, Low Price, and Moving Averages (Kumar et al., 2020). Last but not least, external factors that encompass Oil Price, Gold Price, Commodity Price, and Exchange Rate are also the type of input variable that can be in stock market prediction. Various types of variables can be employed to predict stock market behavior. However, this research specifically concentrates on utilizing closing prices as the primary dataset for prediction due to its significance in the stock market, alongside the varying severity of outlier issues in the data.

Starting from 2019, the whole world is forced to endure economic hardship due to the rapid spread of the Coronavirus Disease 2019 (COVID-19) (Khanthavit, 2021). The impacts of COVID-19 on the economy have been significant and disadvantageous (Hasanat et al., 2020). The unanticipated recession triggered by the pandemic has severely impacted multiple economic sectors, leaving the economy in a state of ongoing uncertainty (Gamal et al., 2021). In the Malaysian stock market, the pandemic was found positively impact the FBM KLCI and sectoral indices, despite the implementation of the Movement Control Order (MCO) and the cessation of most economic activities. This scenario worsened the problem of outlier values that exist in the FBM KLCI stock market dataset (Basuony et al., 2021). The problem of outliers was identified in the prediction of the Malaysian stock market, as highlighted by Seong and Salleh in 2022.

Outliers are typically regarded as random occurrences that cannot be predicted (Naidoo & Du, 2022). In the FBM KLCI dataset, the presence of outliers becomes apparent during the training phase when lag variables are introduced. Consequently, this study opts for univariate analysis, selecting closing prices as the core dataset. Nonetheless, the creation of additional variables becomes necessary to address outlier problems.

According to the findings by Rusiecki (2012), an outlier within a dataset is a numerical value that exhibits significant deviation from the rest of the data points. These outliers can exert a noticeable impact on measures of central tendency, especially the mean, as noted by Mishra et al. in 2019. Within a dataset, outliers can disrupt modeling accuracy and influence the estimated parameters, particularly in statistical analysis.

Backpropagation was defined as a supervised learning algorithm used for training artificial neural networks. It played a crucial role in minimizing the error between predicted and actual outputs by adjusting the weights of the network through a process known as gradient descent. This algorithm was essential for enabling neural networks to learn from data and improve their performance over time. When it comes to the training process of backpropagation in neural networks, outliers can significantly affect the weight adjustment process. Even a single outlier can propagate its effects throughout the network, potentially leading to inaccuracies in the final results. The significant impact of a single outlier is that it can distort the learning process of the neural network, causing it to focus disproportionately on the outlier rather than the general pattern of the data. This can lead to poor generalization and reduced prediction accuracy. In the context of stock market prediction, this means that the model may make erroneous predictions, resulting in potential financial losses for investors and speculators.

The presence of outliers in time series data is a common occurrence, typically ranging from 1% to 10% in routine data. Outliers represent data points that deviate substantially from the established patterns within the majority of the dataset, as explained by Rusiecki in 2013. These values may deviate markedly from the typical sample values, either due to measurement errors or because they reflect significant features within the data. Previous studies have indicated that the existence of such outliers can pose a challenge to conventional least square analysis methods, potentially making them formidable competitors.

Various linear and nonlinear time series approaches are employed to predict stock market behavior with the aim of minimizing prediction errors, such as Autoregressive Integrated Moving Average (ARIMA) (Hafiz et al., 2019), Neural Network (NN) (Averitt & Natarajan, 2018), Long Short-term Memory (LSTM) (Lv et al., 2021), and Recurrent Neural Network (RNN) (Reddy et al., 202). However, ARIMA models are sensitive towards outliers (Agnieszka & Magdalena, 2018). In such cases, preprocessing the data to eliminate these outliers may be necessary before applying ARIMA models. Additionally, ARIMA models tend to underperform when applied to long-term forecasts due to the assumption of stationarity in the underlying time series data, which may not hold true for long-term predictions (Wang et al., 2023).

Nonetheless, ARIMA encounters challenges when dealing with practical nonlinear problems. Nonlinearity is employed to characterize situations where there is not a straightforward or direct relationship between an independent variable and a dependent variable (Hayes, 2021). Consequently, as Ma and Ihler noted in 2020, linear models tend to outperform more complex structural models. Furthermore, noise is a prevalent problem in many forecasting domains, necessitating the application of noise-resistant methods for stock market prediction. Accurately forecasting stock market prices is undeniably a challenging task (Yiing & Thim, 2015).

In the realm of Artificial Intelligence (AI), Artificial Neural Networks (ANNs) were utilized to enhance the accuracy of stock market prediction, as emphasized by Bhardwaj et al. (2020). ANNs outperform traditional statistical methods due to their effective handling of both linear and nonlinear time series data, whether it is noisy or not (Ashour et al., 2018). A notable advantage of ANNs is their ability to operate without requiring prior information about the systems of interest. Since their versatility as function approximators, ANNs have gained significant attention from practitioners across diverse fields.

Utilizing an AI approach enables the implementation of advanced automation and computational methods to enhance results while reducing errors, as highlighted by Chuan et al. (2021). Numerous AI techniques, including Genetic Algorithms, Decision Trees Algorithms, Support Vector Machines (SVM), Neural Networks

(NN), Deep Learning (DL), and Machine Learning (ML), can be employed to develop models capable of addressing complex challenges.

However, NN is frequently used in AI, especially ANNs. ANNs offer numerous advantages in the realm of stock market prediction. As discussed across various research articles, ANNs excel in managing complex, non-linear relationships between input and output variables, rendering them well-suited for stock market prediction, which are influenced by a multitude of factors (Yetis et al., 2014; Selvamuthu et al., 2019; Chhajer et al., 2022). Moreover, they possess the capability to learn from historical data and adapt to evolving market conditions, enhancing their utility in predicting future stock prices (Yetis et al., 2014; Bing et al., 2012).

Furthermore, ANNs can integrate technical analysis components, such as moving averages and trading volumes, into their predictions, leading to heightened accuracy (Selvamuthu et al., 2019; Oh, 2022). Additionally, ANNs are adept at handling substantial volumes of data and discerning patterns that may elude human analysts, resulting in more accurate predictions (Oh, 2022). Through a sufficient period of fund simulation, ANNs can provide dependable results, empowering investors to make confident decisions without the need for daily data analysis (Oh, 2022).

Overall, ANNs offer several advantages in stock market prediction, encompassing their adeptness at handling intricate relationships, adapting to market dynamics, and incorporating technical analysis. These attributes make ANNs a valuable tool for projecting future stock prices and assisting investors in making well-informed choices.

Another significant concern pertains to the presence of outliers within stock market data, which can exert a notable influence on the approximated model, particularly when employing the least squares method. Addressing outliers in a time series commonly involves identifying these outliers and subsequently applying intervention models to investigate their effects. Tsay (1988) notes that the iterative approach necessitates multiple iterations between outlier detection and model parameter estimation.

In this context, two particularly useful methods for characterizing data in terms of location and dispersion are metrics for mean and variance. When data is devoid of outliers, the sample mean $\bar{x}$ and variance s^2 of a sample $X_N = \{x_i\}_{i=1}^{N}$ typically provide reliable estimates of location and dispersion. However, even a solitary observation exhibiting significant variability can wield a substantial impact on both the sample mean and dispersion matrix when the data is tainted. Consequently, employing robust model estimation techniques proves advantageous in situations involving contaminated data (Bakar & Midi, 2019). It was referred to as contaminated data because extreme values or anomalies could significantly distort the dataset. Failure to account for the effects of contaminated data could lead to inaccurate forecasts.

Nonetheless, the influence of outliers on the neural network training process using real stock market data, which may contain contaminated data, results in these outliers affecting the data and propagating into subsequent lags. The impact of even a single outlier on the dataset was significant, indicating that the presence of multiple outliers could further disrupt the network learning process, as highlighted by Jang et al.

(2015). This scenario can ultimately lead to erroneous network training and inaccurate predictions regarding future stock market behavior.

In the context of time series data, the term "lag" signifies a specific time interval, allowing for the emergence of autocorrelation (Naidoo & Du, 2022). Autocorrelation, as explained by Linden and Adams in 2010, refers to the tendency of instances in a time series to exhibit correlation with preceding instances.

Furthermore, the backpropagation neural network (BPNN) exhibits strong performance in time series forecasting, particularly when applied to stock price time-series data, as indicated by Ghasemiyeh et al. (2017). However, it's worth noting that the backpropagation learning algorithm, which relies on minimizing the Ordinary Least Square (OLS) of the Mean Square Error (MSE) cost function, lacks robustness in the presence of outliers, potentially resulting in errors during the data training process (El-Melegy et al., 2009). The MSE, a fundamental backpropagation learning technique employed in multi-layer feedforward neural networks (MFNNs), quantifies the disparity between the desired and actual output (Samantaray & Sahoo, 2020).

Hence, this study aims to enhance the robustness of the MSE in the backpropagation algorithm, which is susceptible to violations due to outliers. This is achieved by substituting OLS with Least Median Squares (LMedS) estimators, capable of handling up to 50% outliers. However, it's worth noting that, as Rusiecki et al. pointed out in 2014, LMedS exhibits notably low efficiency, and errors associated with LMedS cannot be minimized through gradient algorithms. To enhance the efficiency of the BPNN, the research employed a metaheuristic algorithm.

The term 'Metaheuristic' was initially introduced by Glover in 1986 and was later described as nature inspired by Askari et al. in 2020. The concept of "nature-inspired computing" involves the development of algorithms that tackle optimization problems by simulating natural phenomena or biological attributes, as explained by Ma et al. in 2022. Examples of metaheuristic algorithms include those inspired by plant biology, such as the firefly algorithm, whale algorithm, and particle swarm optimization algorithms (Gupta et al., 2020; Zhao et al., 2021). Furthermore, metaheuristic algorithms represent a contemporary approach to fortifying the BPNN against problems like outliers, as highlighted by Mamoudan et al. in 2023.

Hence, the primary objective of this research was to propose a new metaheuristic algorithm termed the Date Palm Seed Growth (DPSG) optimization algorithm. This innovative approach is inspired by the growth mechanism of date palm seeds, a common agricultural practice in the Middle East where dates are cultivated in sandpits and covered with stones. The key focus is to enhance the performance of the BPNN model, thereby improving its effectiveness in forecasting stock market trends. This enhanced BPNN model is equipped to autonomously adapt and effectively manage stock market datasets that are afflicted by challenges like outliers problem.

1.3 Problem Statement

ANNs are extensively employed in stock market prediction analysis due to their ability to effectively map nonlinear relationships between input and output variables. Specifically, BPNN is well-suited for handling large-scale, complex data tasks that exhibit nonlinearity. Nonlinearity referred to the complex and non-proportional relationship between stock market inputs (such as prices and indicators) and outputs (predictions). In the context of BPNN, the OLS estimators are commonly used. However, it's crucial to note that the backpropagation algorithm based on OLS struggles to address issues related to outliers.

In practice, big data sets, such as FBM KLCI data, often present challenges associated with outliers. Outliers tend to emerge due to the influence of uncommon and non-repetitive events. These outliers significantly impact the forecasting process by substantially reducing forecast accuracy and introducing bias into parameter estimations, as indicated by Hosseinioun in 2016.

Stock market prediction involves multiple influencing factors; however, this study specifically examines univariate data due to the substantial impact of outliers on historical price movements. The rationale for utilizing a univariate approach is to facilitate a controlled analysis of the BPNN response to extreme fluctuations in stock prices as a singular time-dependent variable (Chen et al., 2022). Prior research by Anis and Bahar (2021) has highlighted that outliers in stock prices can significantly compromise forecasting models, resulting in poor generalization and increased prediction errors. By isolating stock price trends and excluding additional variables, the effectiveness of the enhanced BPNN model in mitigating the adverse effects of outliers could be evaluated.

Moreover, when working with the FBM KLCI dataset that exhibits outliers, performance errors and network over-fitting problems become prevalent.

Consequently, the accuracy of stock market predictions is adversely affected when dealing with the FBM KLCI dataset afflicted by outliers.

Metaheuristic algorithms offer a range of advantages, including resilience to collinearity and outliers. Building on the inspiration drawn from the growth mechanism of date palm seeds, a metaheuristic algorithm was developed to minimize LMedS. To address this challenge, this study introduces the LMedS estimator, which is proficient at accommodating up to 50% of outliers. Regrettably, as noted in the literature by Rusiecki et al. in 2014, LMedS exhibits notably low efficiency, and its errors remain unmitigated by gradient algorithms.

Various metaheuristic techniques, such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and the Firefly Algorithm (FA), were employed to enhance LMedS. These methods aimed to improve the accuracy and robustness of the estimator but continued to face challenges related to convergence speed, scalability, and escaping local optima. To address these challenges, this research proposed the Date Palm Seed Growth Optimization Least Median Square (DPSG-LMedS) algorithm to mitigate these limitations.

1.4 Research Questions

The objectives of this research are to investigate several questions pertaining to the challenges associated with univariate data, as outlined below:

1. How severe outliers problems of FBM KLCI dataset?
2. How to improve the predictive accuracy of BPNN model?

3. How the performance of the enhanced BPNN model?
4. Is the enhanced BPNN model reliable?

1.5 Research Objectives

The primary aim of this research is to enhance BPNN model, and in pursuit of this objective, the following specific goals are set:

1. To identify the severity of outliers problems within the FBM KLCI dataset.
2. To develop an enhanced BPNN model for FBM KLCI stock market.
3. To compare the performance of the enhanced with the ordinary BPNN model and BPNN-LMedS model.
4. To check the reliability of the enhanced BPNN model.

1.6 Scope of the Study

The analyzed data focused on the Malaysian stock market, referred to as the FBM KLCI stock market (Al-Mashhadani et al., 2021). The FBM KLCI stock market data was collected by extracting information from the Yahoo Finance website. The dataset encompasses daily records spanning from 2nd January 2018 to 30th December 2022, with a specific emphasis on univariate data, specifically the closing prices. The time frame of the dataset purposely includes the stock market data during COVID-19 events (Khairudin et al., 2023).

1.7 Significance of the Research

The enhanced BPNN has the capability to reduce network errors by addressing the challenges posed by outliers. By integrating DPSG-LMedS algorithm into BPNN model, the network errors can be reduced while addressing the outliers. This leads towards the model efficiency and the prediction accuracy (Chen et al., 2024). This study holds substantial importance for various stakeholders in the stock market. Importantly, the utilization of the enhanced model in the research is anticipated to yield enhanced profits for individuals with an interest in the stock market, including speculators and investors. Moreover, since the optimization of the model been used, the investors can enhance the accuracy of predictions regarding stock prices and market movements. This leads to better-informed trading decisions, potentially increasing profitability.

Accurate stock market predictions were crucial for economists in understanding market trends and making informed economic forecasts. The enhanced BPNN model provided a reliable tool for analyzing market behavior, aiding in the development of economic policies and strategies. This study contributed to the academic community by presenting a robust neural network model that could be utilized for various predictive analyses. Researchers could build upon this work to explore further improvements and applications of BPNN in different domains.

For policymakers, accurate stock market predictions were essential for maintaining economic stability. The enhanced BPNN model assisted in monitoring and regulating market activities, ensuring the stability of the FBM KLCI as a measure of the national economy within an interconnected global economic landscape.

The study served as an educational resource for students pursuing studies in finance, economics, and data science. It provided insights into the application of advanced neural network models in real-world scenarios, fostering a deeper understanding of predictive modeling techniques. Overall, this study demonstrated the practical benefits of the enhanced BPNN model in improving prediction accuracy and supporting various industry players and stakeholders in making informed decisions.

1.8 Summary

In this chapter, this research emphasizes the driving force behind the research and delve into the common challenges encountered in big data like outliers. Consequently, this research formulates research questions aimed at tackling these problems. Subsequent to the development of the research questions, this research outlines the study's objectives in the following section. Finally, this research portrays the study's scope and underscores its significance in the concluding part of this chapter.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

Chapter two extensively explores relevant literature concerning stock market prediction using backpropagation neural networks (BPNN). Additionally, it delves into the literature related to enhancing BPNN through the application of metaheuristic approaches. Section 2.2 provides an overview of the literature pertaining to the stock market in Malaysia, while Section 2.3 presents a review of previous studies regarding neural networks in stock market prediction globally. Section 2.4 presents the application of BPNN model in previous research. The literature that specifics the use of BPNN model in stock market prediction and outliers' problem is discussed in Section 2.5. Moreover, Section 2.6 has discussed the solving techniques for outliers' problem which has been divided into two subsections; Section 2.6.1 delved into the metaheuristic approach while focusing on the least median square (LMedS). The component in BPNN modelling has been explained in eight subsections under Section 2.7. Furthermore, Section 2.8 addresses the validation of time series neural network models for stock market prediction. Based on the previous study, the convergence test for the enhanced model has been explained in Section 2.9. Finally, Section 2.10 offers a comprehensive summary of the chapter's contents.

2.2 Stock Market in Malaysia

The stock market in Malaysia, represented by Bursa Malaysia, has played a pivotal role in the nation's economic growth. Early research on the Malaysian stock market, such as Lai and Lau (2004), traced its historical development from its beginnings in the 1960s, highlighting the influence of commodity trading and government policies. Numerous studies, including the work of Lean and Smyth (2010), have examined the efficiency of the Malaysian stock market. Researchers have employed various methodologies to test whether the market adheres to the Efficient Market Hypothesis (EMH) and to what degree. The impact of macroeconomic variables on stock market performance in Malaysia has been a recurring theme. The relationship between economic indicators like Gross Domestic Product (GDP) growth, inflation, and interest rates, and stock market movement has been studied (Ho, 2019). Malaysia's prominence in Islamic finance has been investigated in numerous studies. Hassan et al., (2022) examined the performance and growth of Islamic indices and funds in Malaysia. Corporate governance practices and transparency in financial reporting have been scrutinized. The study by Boadi and Amegbe in 2017 explored the relationship between corporate governance quality and stock market performance. Recent research by Barbosa et al. in 2023 explored the integration of Environmental, Social, and Governance (ESG) factors into investment decisions and their implications for the Malaysian stock market.

Research has shown that the development of the Malaysian stock market, as measured by market capitalization, has a positive but statistically insignificant relationship with the country's economic growth (Zulikifli et al., 2024). This suggests that while the stock market plays an important role, other factors also contribute to Malaysia's economic growth.

At the same time, the performance of the FBM KLCI has been resilient, with the index emerging as the second-best performer in ASEAN in the second half of 2023, closing the year at 1,454.7 points (Taharem & Fitriyah, 2023). This highlights the market's capability to adapt and thrive, solidifying Bursa Malaysia's role as a viable platform for fundraising and investing.

In terms of predicting stock market movements, studies have shown that neural network models can be effective in forecasting stock prices in various markets around the world, including Malaysia (Bursa, 2024). Therefore, the implementation of neural network in stock market prediction globally were discussed further in Section 2.3 to understand it further.

2.3 Neural Network in Stock Market Prediction Globally

Neural networks were used to predict stock prices in various markets around the world. These models use historical stock data, such as prices and trading volume, as input to predict future stock prices. The neural network attempts to learn the underlying patterns and relationships in the data, which can then be used to make predictions. However, predicting stock prices is a challenging task due to the complexity and volatility of financial markets, and the accuracy of these predictions can vary. Additionally, it is essential to use a robust evaluation method and consider the uncertainty of the predictions to get a more realistic result.

There were many studies in the literature that used neural networks to predict stock prices in various markets around the world. The study by Abdouli et al. (2020) used a long short-term memory (LSTM) neural network to predict Tehran Stock Exchange (TSE). The authors found that the LSTM model outperformed traditional time series models, such as Autoregressive Integrated Moving Average (ARIMA) in terms of prediction accuracy.

Another study by Guresen et al. (2011) applied a Multi-Layer Perceptron (MLP) model, to predict National Association of Securities Dealers Automated Quotations (NASDAQ) stock exchange. The results showed that the MLP model more accurate compared to generalized autoregressive conditional heteroscedasticity MLP (GARCH-MLP) and dynamic artificial neural network (DANN) model.

A study by Liu et al. (2021) used a deep neural network (DNN) to predict the dataset provided by Jane Street, which originated from major stock exchanges around the world. Jane Street was a financial services firm that engaged in trading a diverse array of asset classes across over 200 trading venues in 45 countries worldwide. The authors found that the neural network model outperformed others machine learning (ML) models, such as Xgboost algorithm and random forest algorithm, in terms of prediction accuracy.

Research by Lv et al. (2021) used an enhanced model called as LightGBM-optimized LSTM model to predict stock prices in the Shanghai and Shenzhen 300 indexes stock market, and they found that the LightGBM-optimized LSTM model outperformed other deep network models which is Gated Recurrend Unit (GRU) and recurrent neural network (RNN).

Overall, these studies suggest that neural network models can be effective in predicting stock prices in various markets around the world. However, it is important to keep in mind that stock market prediction is a challenging task, and the results can be affected by various factors such as market conditions, economic indicators, and company specific events.

Last but not least, previous research shows that BPNN model is an effective and widely used approach for predicting stock prices, outperforming other models in terms of accuracy and problem-solving ability. The research that has been used BPNN model in predicting stock market has been show in Section 2.4.

2.4 Backpropagation Neural Network

Backpropagation (BP) is a fundamental tool in machine learning for efficient training (deep) neural networks (Brunel et al., 2019). BP operated through two main processes, namely forward propagation and backward propagation. BPNN was utilized in various studies, such as image recognition (Du et al., 2022), game development (Bhasin & Vaishali, 2017), and healthcare (Torres et al., 2022).

In the context of the stock market, the BPNN model was used to predict the stock market. BPNN performs well in time series forecasting when it comes to stock price time series data (Ghasemiyeh et al., 2017). Prior knowledge about systems of interest is not needed. Due to their universal capacity as a function approximator, BPNNs have successfully captured the attention of many practitioners in many fields. However, in the presence of outliers that may cause errors in the data training process, the backpropagation learning algorithm based on the minimization of the mean square error (MSE) cost function is not completely robust. Several studies show the presence of outliers as discussed in Section 2.5.

2.5 Backpropagation Neural Network in Stock Market Prediction with the Presents of Outliers

The limitations of BPNN in handling outliers have been discussed in a previous studies. Research by Chan et al. (2022) suggests that BPNN can be highly sensitive to outliers, which can negatively impact prediction accuracy, making it less suitable for datasets with extreme variations.

In addressing this issue, the selection of an appropriate loss function plays a critical role, as it applies to individual data points to quantify the prediction error. Traditional loss functions, such as Mean Squared Error (MSE), often amplify the influence of outliers due to squared error penalization. Alternative loss functions, such as Mean Absolute Error (MAE) and Huber Loss, offer more robust solutions by reducing the disproportionate impact of extreme values.

At a broader level, the cost function aggregates the loss function across the dataset, guiding the optimization process to improve overall model performance. Therefore, careful selection of the loss function directly influences the cost function, shaping the effectiveness of BPNN in stock market prediction, particularly in datasets containing outliers (Zhao et al., 2024).

Moreover, previous research has compared the forecasting performance of different neural network models, including the BPNN model, for stock market returns. The results showed that the cerebellar model articulation controller neural network (CAMC NN) model made more accurate forecasts than the BPNN model (Selvamuthu et al., 2019).

Outliers or extreme values can have a significant impact on the results of stock market analysis and lag variable. The data can skew statistical measures such as the mean and standard deviation and can lead to unreliable or misleading conclusions if not properly handled.

Several studies investigated the presence of outliers in stock market data. For example, a study by Chen et al. (2022) analyzed the Taiwan stock market and found that outliers had a significant impact on the performance of technical trading rules. Another study by Zhao et al. (2021) analyzed the Shanghai Stock Exchange and found that outliers had a significant impact on the results of event study analysis.

There are several methods that can be used to handle outliers in stock market analysis. One common method is to simply remove them from the dataset, although this approach has the potential to bias the results. Other methods include using robust statistical measures, such as the median and interquartile range, and using outlier detection algorithms, such as the Tukey method or the Z-score method. The prediction result has been combined using the median.

A bias in the parameter estimation and the outliers on the point forecast may give an effect to the forecast accuracy where it will decrease drastically. Outliers can have deterious effects on statistical analyses. It also can result in parameter estimation biases invalid inferences and weak votality forecasts in financial data (Hosseinioun, 2016). In numerous real data sets, outliers are a frequent occurrence. In the research by Hampel et al. (1974), it showed that outliers typically occur in normal data ranges which are from 1% to 10%. To address the outlier problem, two approaches may be employed. One approach involves detecting outliers prior to constructing the model with high-quality data, a process known as outlier diagnostics (Limas et al., 2004; Rousseeuw & Leroy, 1987). Another approach was to employ resistant or robust methods, which remained reliable even when some data was tainted

Last but not least, the outlier percentage has also been focused on in BPNN model in order to get cater to the outliers. Previous research regarding the percentage of outliers that can be catered for has been shown in Section 2.6.

2.6 Outlier Percentage

The outlier percentage plays a crucial role in refining predictive models, enabling analysts to enhance forecast accuracy. Numerous studies have implemented robust techniques to improve the performance of BPNN by addressing the impact of outliers. Research has shown that a significant portion of literature on outlier percentages originates from time series data analysis, as highlighted by Blázquez-García et al. (2021).

In the research by Zhang and Qu (2021), the adaptive genetic algorithm in the backpropagation neural network (AGA-BPNN) accommodated only 5% of outliers. Furthermore, research addressed 20% of outliers using least trimmed squares (LTS) estimators (Beliakov et al., 2011). Additionally, the firefly algorithm applied to the least median squares estimator (FFA-LMedS), least trimmed absolute value (LTA), and least median of squares (LMedS) improved convergence with 50% of outliers (Kamaruddin et al., 2016; Rusiecki et al., 2014). Finally, Wang and Suter (2003) demonstrated that the LTS method handled 40% of outliers, while their proposed approach, least trimmed symmetry distance (LTSD), accommodated up to 60% of outliers.

2.7 Solving Techniques for Outliers' Problem

Dealing with outliers is a crucial step in data analysis and machine learning. Outliers can significantly impact the results of statistical analyses and the performance of prediction models. One of the techniques that can be used to handle the presence of outliers in financial time series data is a metaheuristic algorithm. Various articles have demonstrated the effectiveness of these hybrid neural network metaheuristic approaches in enhancing stock market prediction performance (Elhoseny et al., 2022, Mamoudan et al., 2023).

2.7.1 Metaheuristic Approach

Metaheuristic algorithm is the optimization algorithm where it is inspired by animals or nature (Amiri et al., 2024). It is random to find the new methods in order to get the solutions that are optimum or close to optimal response. The word random in developing the algorithms means that the local optimal solution isn't restricted to a specific answer. The metaheuristic approach can help to enhance the model. There are a lot of metaheuristic algorithms that have been developed in previous studies. Ghasemiyah et al. (2017) developed a novel algorithm that integrates multiple bio-inspired optimization methods, including the Ant Colony Algorithm (ACO), Bee Colony Optimization Algorithm (BCO), Bat Algorithm (BA), Particle Swarm Optimization (PSO), Cuckoo Optimization Algorithm (COA), and Firefly Algorithm (FA).

Moreover, in the study by Farahani and Hajiagha, (2021), an artificial neural network (ANN) was optimized using metaheuristic algorithms, including cuckoo search (CS), enhanced cuckoo search (ECS), genetic algorithm (GA), and particle swarm optimization (PSO). The study compared the performance of the optimized models with non-weighted models in predicting stock price indices. Moreover, there is also a study that focused on intraday stock price forecasting and found that the Particle Swarm Optimization Optimized Backpropagation Neural Network (PSO-BPNN) model yielded the highest prediction accuracy among the tested models (Kumar et al., 2020).

Furthermore, there is a study that proposed an efficient hybrid symbiotic organisms search feedforward neural network (SOSFFNN) model for stock price prediction. The study combined global optimization metaheuristic approaches of symbiotic organisms search (SOS), PSO, and GA with the Feedforward Neural Network (FFNN) model for effective and efficient prediction of stock price indices (Pillay & Ezugwu, 2019). These studies demonstrate the use of metaheuristic algorithms to enhance BPNN models in stock market prediction, leading to improved accuracy and performance.

Zhao et al. (2021) used particle swarm and whale optimization algorithm to improve backpropagation neural network. Gupta et al. (2020) found that plant-biology inspired algorithm is superior efficiency compared to latest firefly algorithm.

Other than that, LMedS is the other technique that can also help to enhance the BPNN model as being discussed in Section 2.6.2.

2.7.2 Least Median Square

The least median square (LMedS) method is a robust regression method which means that it is not sensitive to outliers or other violation of assumption of the usual normal model (Massart et al., 1986). According to Farida (2019), LMedS is a robust estimator for the presence of outliers and has a high breakdown value, showing better results compared to OLS in stock market prediction. Moreover, other research also proved that LMedS is better than OLS method in predicting regression parameter on data with up to 3% of the percentage of outliers (Foss et al., 2001).

According to the research by Faraz and Khaloozadeh (2020), this research predicts the Iran stock market closing price by using the enhanced model, Least Squares Generative Adversarial Network (LSGAN). The result shows that the LSGAN outperformed the ordinary model, Generative Adversarial Network (GANs) in stock market prediction by using least-squares loss function and z-score method to remove outliers.

2.8 Performance Validation of Time Series Neural Network models for Stock Market Prediction

Validation of the time series models is really needed in order to the prediction of stock market. There are a few types of validation such as bootstraps. Awajan et al. (2018) validated the model using five techniques namely, Moving Block Bootstrap (MBB), Fourier Transform (FT), Holt-Winter (HW), Intrinsic Mode Function (IMF) and Empirical Mode Decomposition (EMD) applied to daily stock market from six countries. The stock market data were obtained from indices representing the US-S&P 500, Sri Lanka, Netherlands, Malaysia, France and Australia.

The findings from the study by Awajan et al. (2018) indicated that Empirical Mode Decomposition and Holt-Winter (EMD-HW) bagging forecasting results demonstrated greater accuracy compared to these fourteen forecasting methods. The evaluation was based on five error measures: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Theil's U-statistic (TheilU), and Mean Absolute Scaled Error (MASE).

Other than that, the research by Dantas and Oliveira (2018) focuses on enhancing time series forecasting by combining bootstrap aggregation, clusters, and exponential smoothing. The research utilizes the modified Moving Blocks Bootstrap (MBB) algorithm proposed by Bergmeir et al. (2016) to conduct resampling, which is crucial for improving the accuracy of stock market prediction models. The data used in the research by Ren et al. (2018) included three types of stock market which are Standard & Poor's (S&P), Dow Jones and NASDAQ.

Block bootstrap methods encompass several variations, including non-overlapping, overlapping, and circular approaches, each of which had potential for application in time series modeling if appropriate methodologies had been available. In the study conducted by Akerstrom (2020), the training data was divided into two subsets: the training set and the test set. The test set was required to be sufficiently large to yield statistically significant results and to represent the overall dataset comprehensively. The generalized training data was utilized to develop a model. To achieve the optimal model, the cost function needed to be minimized.

Validating the performance of time series neural network models on out-of-sample data is a critical step before deploying such models for real-world stock market prediction. However, it is equally important to test the convergence and stability of the enhanced BPNN model itself to ensure it is optimizing properly and producing reliable forecasts. The following section examined the techniques for assessing the convergence of the enhanced BPNN model for stock market prediction.

2.9 Convergence Evaluation for Proposed Algorithm

The previous study has discussed the performance of ML models, including deep learning (DL) models, in predicting stock prices (Sonkavde et al., 2023). According to the study by Rizvi and Khalid (2024), the study compares the performance of various DL models in predicting stock prices and discusses the importance of selecting appropriate features and hyperparameters for improving the accuracy of predictions. Moreover, there also a study that proposes a hybrid data analytics framework that combines various stock-related information to improve the prediction performance of ML models (Daradke, 2022).

Epochs played a role in training machine learning models for stock market prediction by facilitating iterative learning and allowing fine-tuning of model parameters. Understanding and optimizing the number of epochs was essential for developing robust predictive models that generalized well to new market conditions.

Epochs and Root Mean Square Error (RMSE) value was used in previous research related to stock market predictions that run the convergence tests (Moraitis et al., 2021). Epochs and iterations are important parameters in training neural networks in order to minimize the cost function (Mehmood et al., 2023) The performance of ANN evaluated by using MSE convergence progression versus epochs in stock market predictions, while the RMSE value is used to measure the accuracy of the predictive results (Moraitis et al., 2021).

According to the research by Kalaiselvi et al. in 2018, the maximum number of epoch that being used in that study is 1000 epochs after predict the stock market with BPNN model. Moreover, there is also another study that use 100 up to 1000 epochs in stock price prediction using BPNN based on gradient descent with momentum and adaptive learning Rate (Dwiarso et al., 2017). Furthermore, in the research by Dahal et al. (2023), the research only use 30 epochs in order to test the convergence for LSTM and Gated Recurrent Unit (GRU) model. The convergence has been tested in the Indian stock market prediction using ANN with 16 to 1000 epochs (Selvamuthu et al., 2019).

The number of epochs directly influenced the convergence of the adopted algorithm. A small number of epochs could cause the method to converge at a local minimum, whereas excessive epochs might lead to overlearning. The maximum number of training epoch is 1000, but the stability of the MSE curve usually converged after 22 to 91 epochs for an ANN model, according to Moraitis et al. (2021).

2.10 Summary

In summary, there are still a lot of things that need to be improved. Furthermore, in this research, univariate time series was applied. The outlier's problem can be seen from previous research (Vishwakarma et al., 2020). Therefore, this study aims to enhance the BPNN model to improve stock market forecasting accuracy, achieve the lowest possible error rate, and mitigate the impact of outliers.

CHAPTER 3
METHODOLOGY

3.1 Introduction

This chapter explains in detail the methodology used in this research. The structure of this chapter is according to the research objectives. There are a few parts to answer the first objective. In Section 3.2, this thesis explained the background of the data which is a real dataset of the Financial Times Stock Exchange (FTSE) Bursa Malaysia Kuala Lumpur Composite Index (FBM KLCI) dataset, how the data has been collected and the diagnostic test that needed to test the outlier's problem has been discussed in two subsections. Then, in Section 3.3, the process of preprocessing data was explained in this part which is data normalization and data partitioning. After that, to answer the second objective, Section 3.3.1 and Section 3.32 explained the data normalization and data partitioning process respectively. In Section 3.4, the backpropagation neural network (BPNN) for stock market prediction explains the process and how the ordinary BPNN model works. Next, section 3.5 elaborates on the process of the evaluation of the prediction model using error measures where the error measures were discussed in different sub-sections which is Section 3.5.1 regarding Root Mean Square Error (RMSE) and Section 3.5.2 discusses the Geometric Root Mean Square Error (GRMSE). After that, the convergence test was further discussed in Section 3.6. Lastly, the summary of Chapter 3 was provided in Section 3.7.

3.2 Data Background

A flowchart illustrating the four phases corresponding to the four objectives of this research was developed based on Figure 3.1.

In the first phase, the data is collected, and a diagnostic test is conducted to achieve the first objective which is to identify the severity of outliers problems of the FBM KLCI dataset.

Following this, in the second phase, data preprocessing was conducted by normalizing and partitioning the dataset. Subsequently, the BPNN model was enhanced by replacing the Ordinary Least Square (OLS) cost function with the Least Median Square (LMedS).

In Phase 3, the simulated and real datasets were used in experimentations. Then, the comparison of the model's performance for both BPNN model and enhanced BPNN models using error measures is performed.

In the last phase, before starting to do the prediction, the model validation was used the time series moving block bootstrap (MBB). Last but not least, the stock market predicted for multi-step ahead using the validated model.

Figure 3.1 shows the flowchart of research activities that were conducted in every phase.

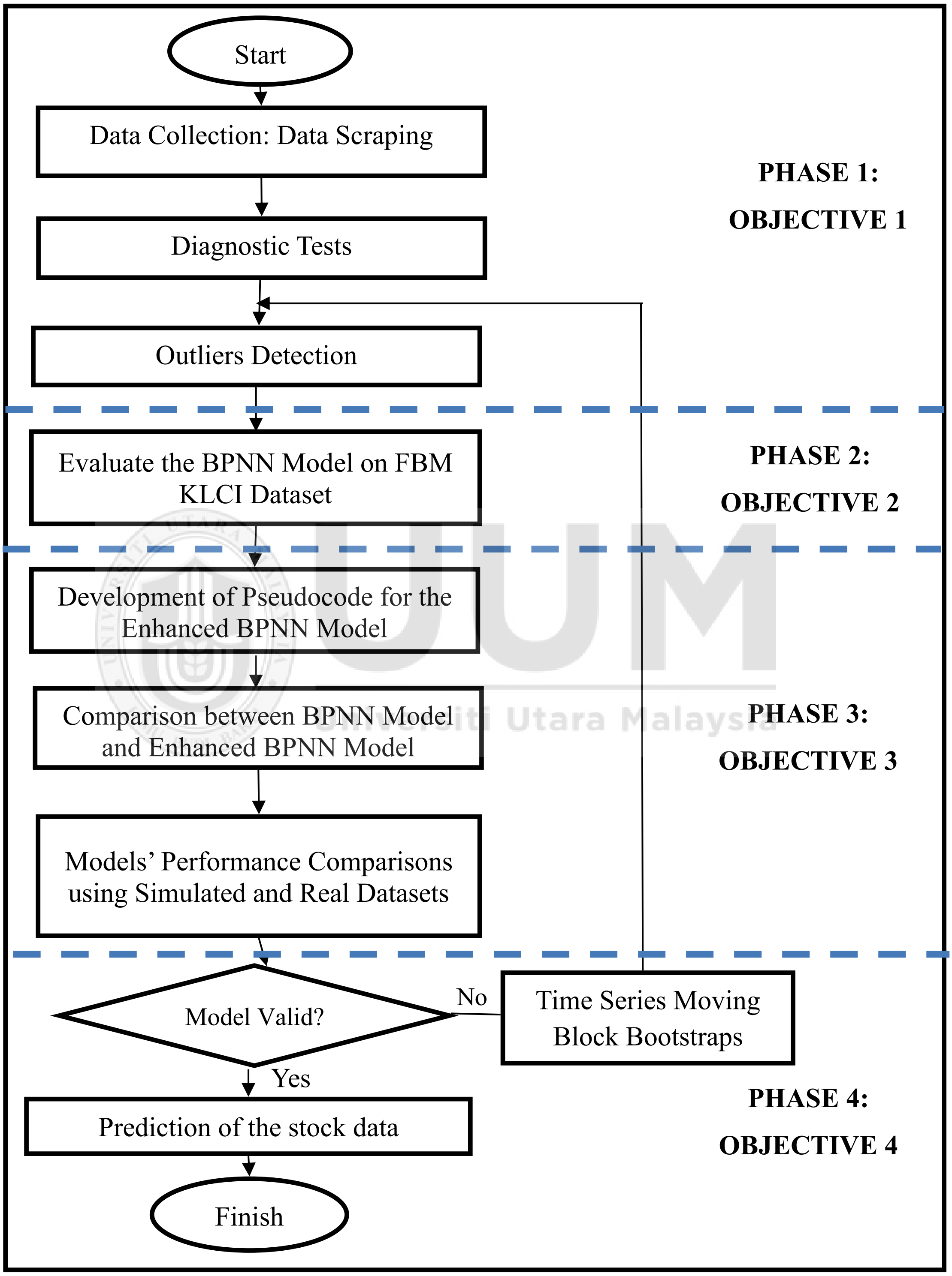

Figure 3.1. Research Flowchart

3.2.1 Data Collection

Yahoo Finance served as the primary data source for this research. Secondary data was utilized, as the Yahoo Finance website contained 1,222 daily closing price data points, as suggested by Vijh et al. (2020), covering the period from January 2, 2018, to December 30, 2022, for stock market prediction. Primary data collection was deemed unnecessary, as the Yahoo Finance dataset met the research requirements and provided a sufficiently large dataset for forecasting. The data was analyzed using both the BPNN model and enhanced BPNN model, which were developed in Spyder software by executing Python code.

3.2.2 Descriptive Analysis

Descriptive statistics are essential in stock market prediction as they help to summarize and understand the data. Here are some key descriptive statistics that can be used:

1. **Mean Price**: The mean provides the average value of stock prices over a specific period. It helps in understanding the central tendency of the data.

2. **Median Price**: The median is the middle value of the stock prices when arranged in ascending to descending order. It is useful for understanding the central tendency, especially when the data has outliers.

3. **Standard deviation**: This measures the dispersion or variability of stock prices from the mean. A higher standard deviation indicates more volatility in the stock prices.

4. **Variance**: Variance is the square of the standard deviation and provides a measure of the spread of stock prices around the mean.

5. **Minimum and Maximum Price**: These values indicate the lowest and highest stock prices within a specific period, providing insights into the range of price fluctuations.

6. **Skewness and Kurtosis**:

   Skewness can help the neural network understand the asymmetry in the distribution of stock prices, which can be important for capturing market anomalies.

   Kurtosis provides information about the "tailedness" of the distribution, helping the neural network understand the probability of extreme price movements.

7. **Simple Moving Average (SMA)**:

   Moving averages can be used as features to smooth out short-term fluctuations and highlight longer-term trends. They are commonly used in technical analysis and can be valuable inputs for neural networks. SMA_50 (50-Day

Simple Moving Average) represents the average closing price over the past 50 days. It reflects short-term trends in the stock price. SMA_200 (200-Day Simple Moving Average) represents the average closing price over the past 200 days. It reflects long-term trends in the stock price.

The stock price was considered as **upward trend** when:

i) SMA_50 is above SMA_200: If the 50-day SMA crosses above the 200-day SMA, it is generally considered a bullish signal, indicating an upward trend. This is known as a "Golden Cross." Golden Cross indicates potential upward momentum and is often seen as a buy signal.

ii) When Both SMAs are Rising: If both the 50-day and 200-day SMAs are rising, it suggests a strong upward trend.

The stock price was considered as **downward trend** when:

i) When SMA_50 is Below SMA_200: If the 50-day SMA crosses below the 200-day SMA, it is generally considered a bearish signal, indicating a downward trend. This is known as a "Death Cross." Death Cross indicates potential downward momentum and is often seen as a sell signal.

ii) When Both SMAs are Falling: If both the 50-day and 200-day SMAs are falling, it suggests a strong downward trend.

There are a few common cases and ways to interpret it as follow:

i) ***If the 50-day SMA is steadily rising and is above the 200-day SMA:*** This indicates a sustained upward trend.

ii) ***If the 50-day SMA crosses above the 200-day SMA (Golden Cross):*** This signals a potential bullish trend and may be a good time to consider buying.

iii) ***If the 50-day SMA is below the 200-day SMA and both are declining:*** This suggests a sustained downward trend.

iv) ***If the 50-day SMA crosses below the 200-day SMA (Death Cross):*** This signals a potential bearish trend and may be a good time to consider selling.

Figure 3.2 illustrated the example of Death Cross and Golden Cross to be clearer how the crosses could be identified.

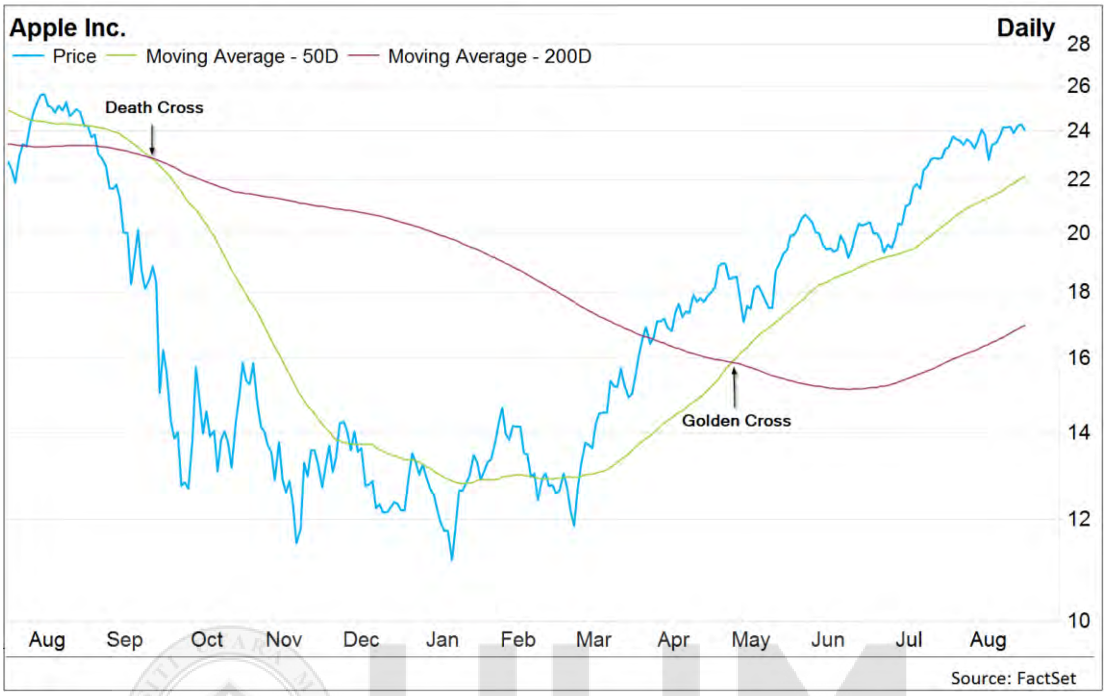

Figure 3.2. Illustration of Death Cross and Golden Cross in Stock Market Trends
Note: Adapted from Simmons (2018)

8. Return Analysis: Returns over different periods can be used as features to represent the performance of stocks. This helps the neural network understand the profitability and growth trends.

9. Frequency Distribution: Creating features based on the frequency distribution of stock prices can help the neural network understand the overall distribution and patterns in the data.

By incorporating these descriptive statistics as features, neural networks can gain a more comprehensive understanding of the stock market data, leading to better predictions and analysis.

3.2.3 Boxplot of outlier detection

In this section, an assessment was conducted to determine whether the data exhibited an outlier problem. The problem of outliers was detected in stock market prediction. The boxplot method, as suggested by McGrill et al., (1978) was employed to identify and examine any potential outliers in daily closing prices of FBM KLCI stock market. Selected order statistics form the basis of the boxplot approach, which is used to locate outlaying observations. In particular, the technique is based on determining the data set's sample quartiles, or hinges, and then building outlier fences (a lower fence and an upper fence). Outliers may occur in the data set if any observations fall outside (in either direction) of the built fences. The lower and upper fence can be seen in Figure 3.3.

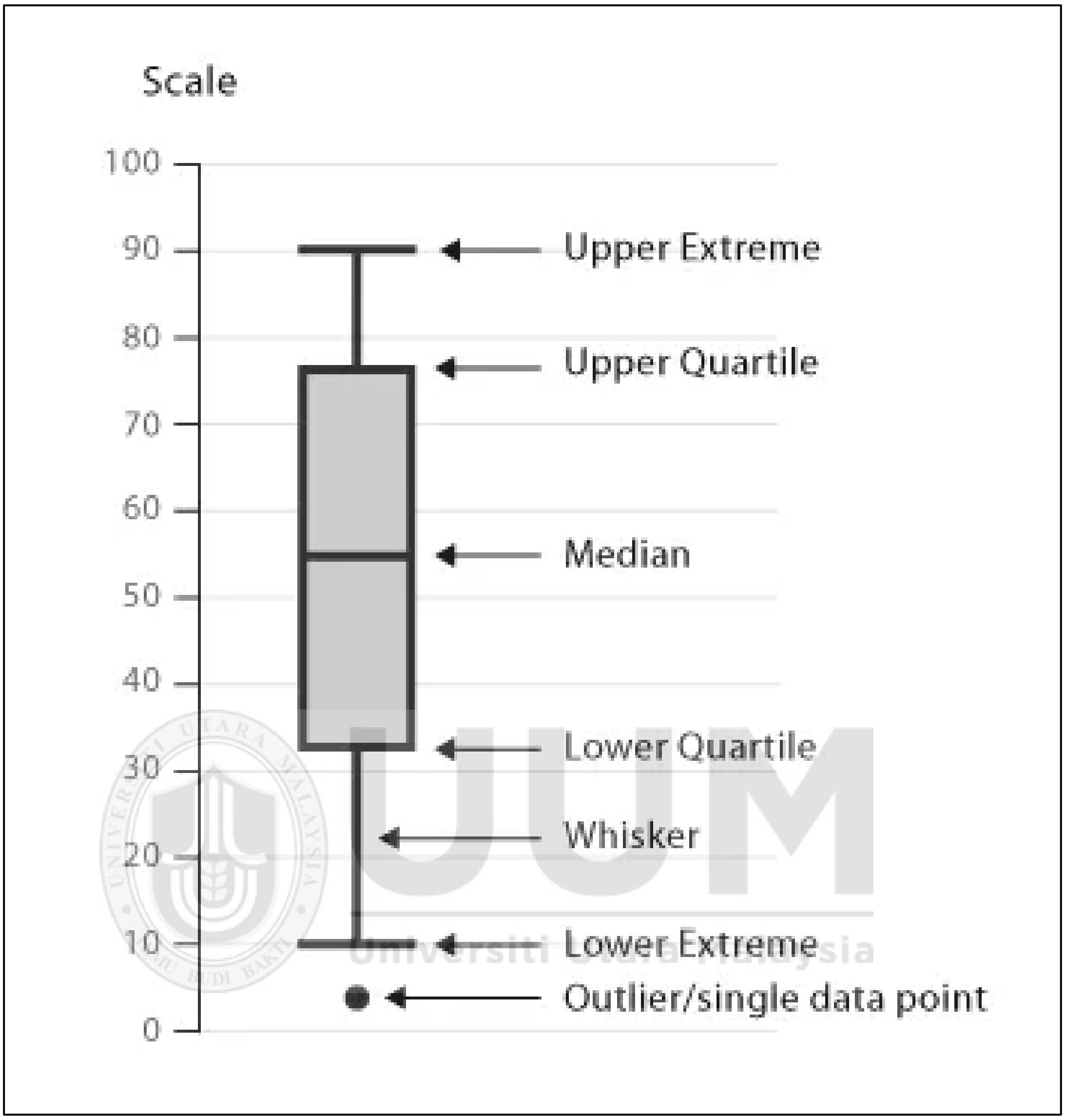

Figure 3.3 Example of Box-plot
Note: Adapted from Chinaza (2023)

3.3 Preprocessing

Data preprocessing is required in neural network analysis. This process involves data normalization and data partitioning.

3.3.1 Data Normalization

Data normalization is important in order to scale the observed values for better neural network learning. Data normalization can reduce the amount of time needed for training by ensuring that each feature is trained on the same scale (Nayak et al., 2014). There are a lot of methods to do the normalization data such as min-max normalization median and median absolute deviation, sigmoid normalization, decimal scaling normalization, z-score normalization, and median normalization. Min-max normalization was implemented, where the data inputs were mapped into a predefined range [-1;1]. This technique normalizes the value of the attribute A of a data set according to its minimum and maximum values. It converts a value a of the attribute A to $\hat{a}$ in the range [low; high] (Zhang et al., 2021) by computing:

$$\hat{a} = \text{low} + \frac{(\text{high}-\text{low})*(\text{a}-\min \text{A})}{\max \text{A} - \min \text{A}} \tag{3.1}$$

All out-of-sample values below min A and above max A were mapped to low and high respectively, after considering the minimum, min A, and maximum, max A, values reported in the sample data set.

3.3.2 Data Partitioning

Based on the research by Sharma et al., (2021), in order to get 100% specificity and sensibility in neural network, data partitioning is divided into two parts which are 85% training set and 15% testing set. Neural network training sufficient if the real data involves a large sample size like more than thousands or hundreds (Wei et al., 2015). This research split the data into training and testing sets using the 'iloc' method by selecting rows based on their index position.

In the context of Python's pandas library, 'iloc' stood for "integer location" and was a method used for accessing and retrieving data from DataFrame objects using integer-based indexing. The iloc method allowed for the selection of specific rows and columns from a DataFrame by providing integer indices, enabling data retrieval based on position rather than label.

The method takes two arguments: the starting index and the ending index of the rows to be selected. The training set consisted of the rows that fell between the starting and finishing indexes; the testing set consisted of the remaining rows. The size of the dataset depends on the percentage of the partitioning.

The two variations of simulated datasets, designated as Data Set I and Data Set II, to evaluate the model's performance. The input nodes number was the most fundamental parameter, as it corresponded to the number of lagged observations used to portray the time management of underlying pattern (Zhang, 2001). The parameters varied across levels of 5, 10, 15, 20, 25, 30, 35 and 40 input lags for enhanced BPNN modelling.

For Data Set I and II showed in Table 3.1, all possible combinations of hidden nodes and input lags were investigated with varying proportions of outliers ranging from 0%, to 65% which extend previously published experiments with such datasets (Ghani et al., 2018). The simulated data, which included Data Set I and II, were required to evaluate the effectiveness of the model.

Table 3.1

Input Lags Outliers Percentage for Enhanced BPNN model on Datasets

No.	**Data Description**	**Notations**	**Outliers**	**Input Lags**	**Hidden Nodes**
1	Real Datasets	FBM KLCI Stock Market Closing Prices	62%		
2	Simulated 1-Dimensional Data	Data Set I	0%		
3	Simulated 1-Dimensional Data	Data Set II	5% 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55%, 60% 65%	5, 10, 15, 20, 25, 30, 35 40	5, 10, 15, 20, 25, 30, 35, 40, 45

Data Set I: To evaluate the algorithm developed for application on FBM KLCI stock market data, was employed as an approximation task. The function defined in Equation 3.2 was used to assess several robust algorithms.

A lot of previous studies have applied this function (Liano, 1996; Chen & Jain, 1994; Chuang, 2020; El-Melegy et al., 2009; Rusiecki, 2005). The independent, x and dependent variable, y were used to generate the simulated data (Chuang, 2020). The equation 3.2 defines this function:

$$y = |x|^{-2/3} \tag{3.2}$$

where,

x = independent variable,

y = dependent variable.

The independent variable, x can generate the data points with a step 0.01 and in the range [-2, 2], and then the dependent variable, y can be determined by Equation 3.2.

Data Set II: The second 1-D function to be approximated was a function considered in many articles (Chen & Jain, 1994; Chuang et al., 2004) defined as:

$$y = \frac{\sin(x)}{x} \tag{3.3}$$

where,

x = independent variable,

y = dependent variable.

With a 0.1 step, the independent variable was sampled within the interval [-7.5, 7.5].

For clearer, Table 3.2 was presented the simulated dataset for this research study.

Table 3.2

Function for Simulated Dataset

No.	Data Type	Notation	Function
1	Simulated 1 - Dimensional Data	Data Set I	$y = \lvert x\rvert^{-2/3}$
2	Simulated 1 - Dimensional Data	Data Set II	$y = \frac{sin(x)}{x}$

Figure 3.4 shows the connectionist feedforward backpropagation of DPSG-LMedS model.

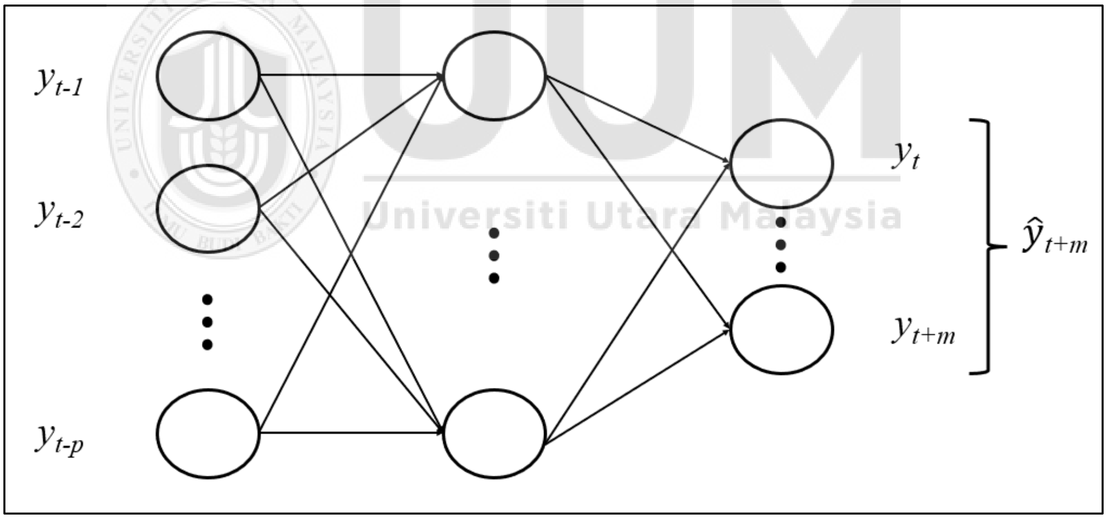

Figure 3.4. Connectionist Feedforward Backpropagation of DPSG-LMedS model

Here $y_{t-1}, \dots, y_{t-p}$ are input values (daily FBM KLCI closing prices data) for p^{th} lag. and $\varepsilon_{t-1}, \varepsilon_{t-2}, \dots, \varepsilon_{t-p}$ are forecast errors for q^{th} lag and y_{t+1} is forecast values of FBM KLCI indices as i^{th}-step-ahead, $i = 1,2, \ldots, m$,. Table 3.3 exhibits the variables of input lags.

Table 3.3

Variables of input lags

Time (*t*)	y_t	Input Lag 1	Input Lag 2	Input Lag 3	…	Input Lag p
1	y_1	-	-	-	-	-
2	y_2	y_1	-	-	-	-
3	y_3	y_2	y_1	-	-	-
4	y_4	y_3	y_2	y_1	-	-
$\vdots$	$\vdots$	y_4	y_3	y_2	-	-
		y_5	y_4	y_3	-	-
		y_6	y_5	y_4	$\vdots$	y_p
	y_{t-1}	$\vdots$	$\vdots$	$\vdots$	-	$\vdots$
T	y_t	y_{t-1}	y_{t-2}	y_{t-3}	$\cdots$	y_{t-p}

Following the data partitioning process, the original FBM KLCI stock market dataset was systematically divided into two distinct subsets: training and testing. The training set, comprising 85% of the total sample size, was designed to facilitate the model's learning of underlying trends and patterns in stock price movements. The remaining 15% was designated as the testing set, ensuring an evaluation of predictive accuracy and model generalization. Table 3.4 presents the partitioning tabulation for FBM KLCI data.

Table 3.4

Partitioning Tabulation for FBM KLCI Data

Series Part	**ANN Data Partitioning Terms**	**Time (t)**	**Actual values at time, t**	**Fitted Values**	**Forecasted Values**
Model Estimation Part	Training Set (85%)	1	$y_{1(train)}$	$\hat{y}_{1(train)}$	-
		2	$y_{2(train)}$	$\hat{y}_{2(train)}$	-
		3	$y_{3(train)}$	$\hat{y}_{3(train)}$	-
		$\vdots$	$\vdots$	$\vdots$	-
		$y_{(train)}$	$y_{t(train)}$	$\hat{y}_{t(train)}$	-
Model Evaluation Part	Testing Set (15%)	$t+1$	y_{t+1}	-	$\hat{y}_{1(forecast)}$
		$t+2$	y_{t+2}	-	$\hat{y}_{2(forecast)}$
		$\vdots$	$\vdots$	-	$\vdots$
				-	
		$t+n$	y_{t+n}	-	$\hat{y}_{n(forecast)}$

3.4 Backpropagation Neural Network Modelling

Backpropagation Neural Network (BPNN)s have gained significant attention in various domains, including stock market prediction, due to their ability to learn complex nonlinear relationships from data (Vargas et al., 2022). The basic structure of a BPNN consists of an input layer, one or more hidden layers, and an output layer (Pellegrino et al., 2022).

The backpropagation algorithm uses gradient descent to update the weights and biases of the network, aiming to minimize the mean squared error between the predicted and actual outputs (Zhang et al., 2024). Data splitting, data preprocessing, design and architecture, training algorithm, and time lags are also important in BPNN model. The following section has discussed further regarding input layer, hidden layers, and an output layer, data splitting, data preprocessing, design and architecture, training algorithm, and time lags.

3.4.1 Input Layer

Input layer in backpropagation neural network (BPNN) is the layer where data is introduced into the system (Priddy, 2007). In the context of backpropagation, which is a supervised learning algorithm for training neural networks, the input layer's primary function is to transmit the input features to the subsequent layers in the network. This process enables the network to learn and make predictions based on the input data effectively.

3.4.2 Output Layer

The output layer in a BPNN consists of a hidden layer with 8 neurons and is used to minimize error between target and output (Primadusi et al., 2016). In BPNN model, the output layer is the layer with adjustable hidden-to-output weights.

3.4.3 Hidden Nodes

The hidden layer in neural networks is an unobservable layer of nodes that calculate the weighted sum of their input nodes and pass the sum, adjusted for a bias, to the next node in the network (Averitt & Natarajan, 2018). Hidden nodes in neural networks are randomly generated additive or radial basis function nodes that can work as universal approximators in incremental extreme learning machines (Huang et al., 2006). In the research by Uzair and Jamil, in 2020, the different number of layers has been tested in order to get the most accurate results.

Figure 3.5 illustrates the architecture and computational flow of the neural network model. The diagram consists of three key layers: the input layer, the hidden layer, and the output layer. The input layer receives multiple features represent past data points fed into the network. These inputs are transmitted through weighted connections to the hidden layer, where neurons process and transform the information using activation functions. The output layer then generates predictions correspond to future values based on the model’s learned patterns. The connections between layers signify the propagation of information, demonstrating how the neural network refines input data to produce accurate forecasts.

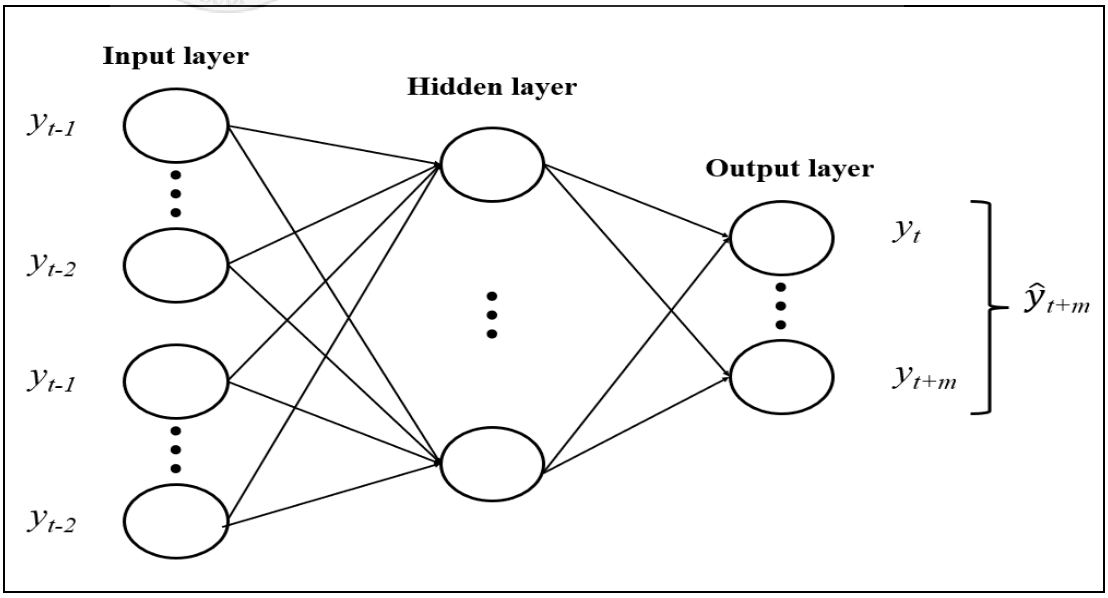

Figure 3.5 Process of Neural Network Model

3.4.4 Data Preprocessing

Data preprocessing is also one of the steps that is essential in stock market prediction. It involves cleaning, transforming, and preparing the data to make it suitable for analysis. In preprocessing data, normalization and feature selection are also involves in order to improve the accuracy of stock market prediction using neural network (Ican & Celik, 2017).

3.4.5 Data Splitting

The process of splitting the data is a crucial step in stock market prediction by using any machine learning model. The data splitting involves dividing the available data into training, and testing sets. The training set is used to train the model and the testing set is used to evaluate the model's performance on unseen data.

The research by Shen and Shafiq in 2020, used the splitting of the data into the training and testing set. The result shows that the system achieves overall high accuracy for stock market prediction by conducting comprehensive evaluations on frequently used machine learning models. The previous study also discusses the importance of splitting the data to train, find the model selection and estimate model prediction error or accuracy (Kumbure et al., 2022).

3.4.6 Design and Architecture

The design and architecture for each type of neural network model is different. Therefore, before the neural network model has been used for analysis, the design of the neural network model is really needed.

3.4.7 Training Algorithm

Backpropagation is a widely used training algorithm for neural networks. In the context of stock market prediction, researchers have applied traditional backpropagation to train neural networks to learn patterns and trends in historical stock data.

3.4.8 Time Lags

Time lags in neural networks were significant, particularly when processing time-series data or sequences. Utilizing time series with input vector lags improved the accuracy of forecasting stock market indices (Surakhi et al., 2021). Hadi (2006) proposed a methodology for reducing data requirements in hydrological time series forecasting by employing Box-Jenkins models to identify “lag components” and developing a compact network structure.

According to Kamaruddin et al. (2019), the study utilized time lags of 5, 10, 15, 20, 25, 30, 35, and 40. The findings indicated that the optimal configuration for the Killer Whale-Backpropagation (KW-BP) Algorithm was 30-30-30 for input lags, error lags, and hidden nodes, respectively.

One previous study by Hsieh et al. (2011) applied a BPNN to predict future stock prices using various input nodes and found that the number of nodes in the hidden layer affected the model's convergence efficiency and prediction accuracy. The study also determined that the optimal configuration for the hidden layer was achieved when the number of nodes was twice that of the input nodes. Similarly, Utomo et al. (2017) found that the number of input nodes, the number of hidden layer neurons, and the number of training iterations were significant factors affecting forecast accuracy.

3.5 Backpropagation Neural Network (BPNN) for Stock Market Prediction

In stock market prediction, BPNNs can handle large-scale data tasks and identify the patterns that are often present in financial data. This ability makes BPNNs a powerful tool for predicting future stock prices and market trends. To better understand how BPNNs operate, consider the flowchart depicted in Figure 3.6. This figure illustrates the steps involved in implementing the original BPNN model. It shows the process of data collection, preprocessing, model determination, training, and validation.

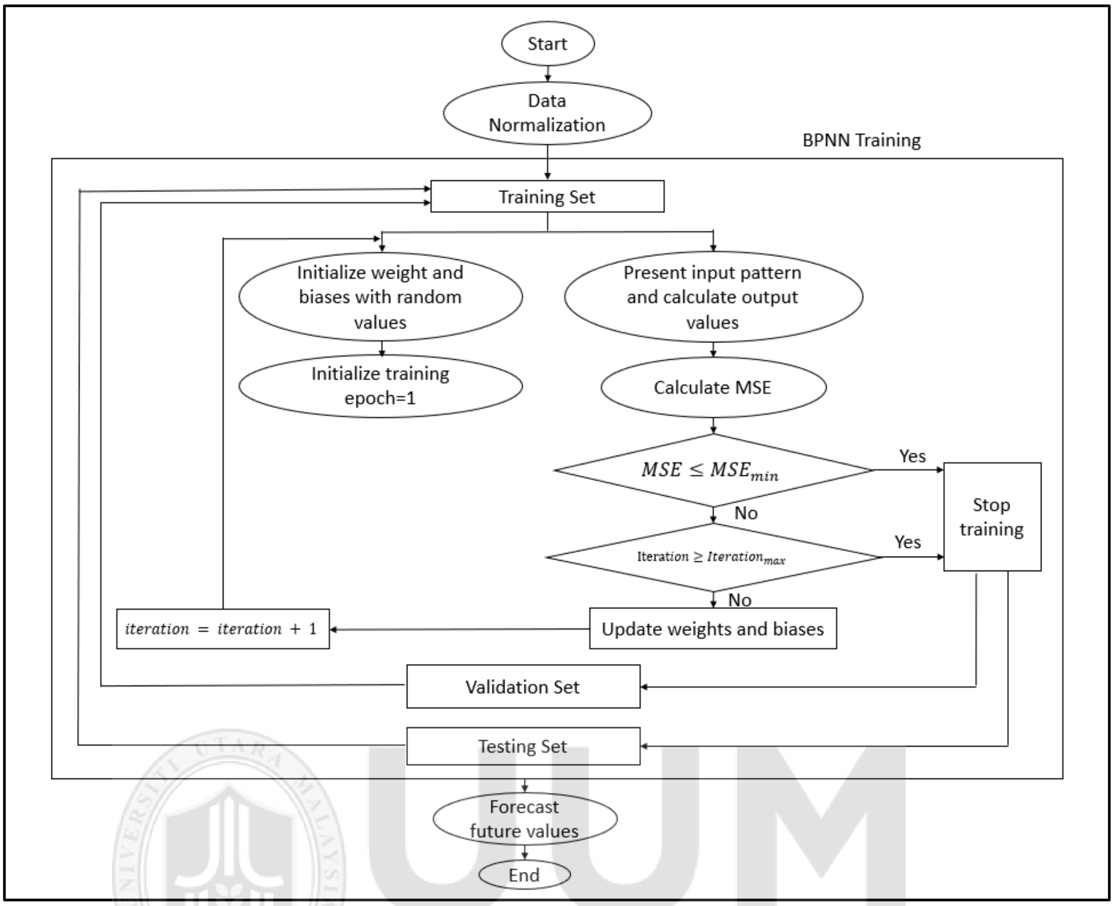

Figure 3.6. *Backpropagation Neural Network*

The multilayer perceptron was the main concern.

Step 1: Time Series Input Data

$$y_t = x_{t-1}, x_{t-2}, \cdots, x_{t-p} \tag{3.4}$$

where,

y_t = input values (daily closing stock market data),

$x_{t-1}, \ldots, x_{t-p}$ = p^{th} lag input values (daily closing stock market data),

t-p = lags for inputs,

Step 2: Initialization of random weight from input layer to hidden layer.

$$\Gamma_{\alpha,\beta}^{input\to hidden} = \pm\frac{1}{2n}\sum_{\alpha=1}^{n}\frac{1}{|x_t|} \tag{3.5}$$

where

α = row connecting hidden nodes with input nodes,

β = column connecting hidden nodes with input nodes,

n = total number of inputs,

y_t = inputs of network,

Γ = vector of weight.

Step 3: Approximate $\hat{y}_{t+\alpha}$

The formula for autoregressive (AR) model is

$$y_{t+\alpha} = f\left(y_{t-1}, \dots, y_{t-p}\right) + \varepsilon_t, \tag{3.6}$$

becoming

$$y_{t+\alpha} = f(y_t) + \varepsilon_t, \tag{3.7}$$

where

y_t = actual values (daily closing data of the stock market) at time t,

y_{t+i} = stock market data predict values at i^{th}-step-ahead, i= 0,1,2,…m,

$x_{t-1}, \dots, x_{t-p}$ = p^{th} lag input values (daily closing stock market data),

ε_t= the errors of the model at time t,

t = time,

$f(.)$ = function of nonlinear.

The expression in equation 3.7 is the approximation of nonlinear to f:

$$\hat{y}_{t+\alpha} = f(y_t), \tag{3.8}$$

where

$\hat{y}_{t+\alpha}$ = the approximated of forecast values at i^th^-step-ahead, i= 0, 1, 2, …, m,

f (.) = the evaluated of nonlinear function at y_t

y_t = real values (daily closing stock market data) at time t,

Step 4: Approximate the output for the first layer. The approximation of the nonlinear to f is the feedforward network given by

$$y_j = f_1(y_{t-1}, \dots, y_{t-p})$$

$$\therefore y_j = f_1\left(\sum_{\alpha=1}^{l} w_{\alpha j} y_t\right), \tag{3.9}$$

where

$\Gamma_{i\beta}$ = An input layer to hidden layer connection's random weight matrix ranges from -1 to 1,

f_1 = tangent sigmoidal function.

Therefore, nonlinear mapping from pass data to projections of future data is represented by the feedforward network. Consider the weight matrix of a link between the output and hidden layers as comparable $w_{\beta k}$. Figure 3.7 exhibits the link between output layer and output.

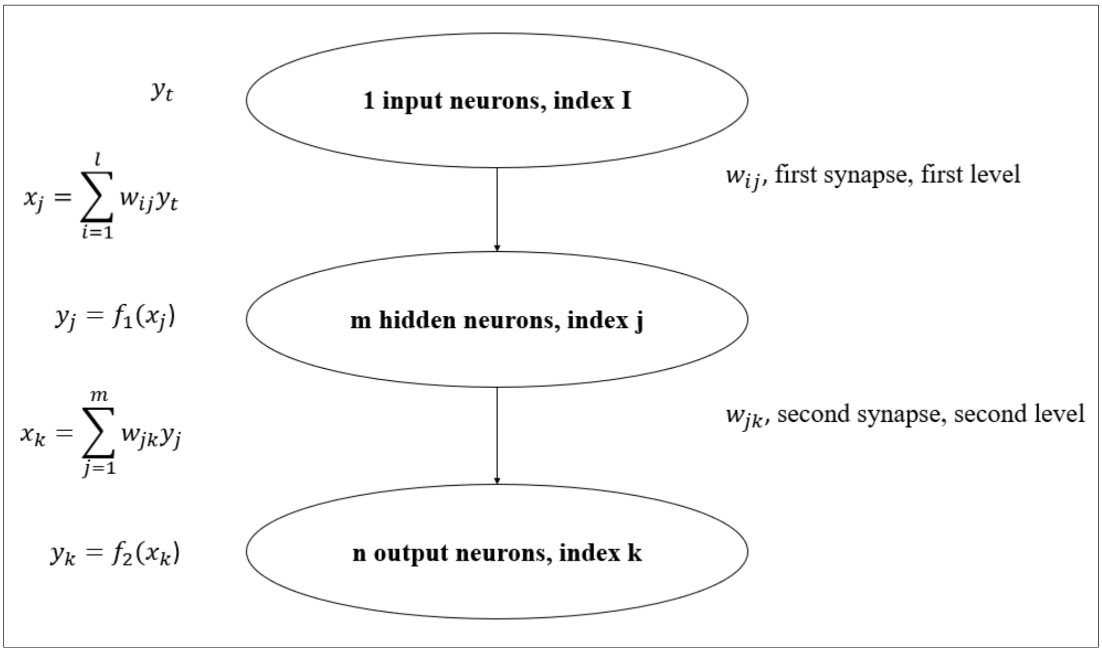

Figure 3.7. Information Flow in the Forward Path of a Two-Layer Neural Network

Step 5: Define the first synapse which is input to hidden layer activation function, $f(.)$.

$$\boldsymbol{f_1}(.) = \frac{\mathbf{sinh}\,(.)}{\mathbf{cosh}\,(.)} = \frac{\boldsymbol{exp}^{(.)} - \boldsymbol{exp}^{-(.)}}{\boldsymbol{exp}^{(.)} + \boldsymbol{exp}^{-(.)}}$$

$$= 2\left(\frac{1}{1 + exp^{-2(.)}}\right) - 1 \tag{3.10}$$

where f_{tanh}: $\Re \rightarrow [-1;1]$

Becoming

$$y_j = \tanh\left(\sum\nolimits_{\alpha=1}^{l} w_{\alpha j} y_t\right)$$

$$= 2\left(\frac{1}{1 + exp^{-2\left(\sum_{\alpha=1}^{l} w_{\alpha j} y_t\right)}}\right) - 1 \tag{3.11}$$

Step 6: Initialization of random weights from hidden to output layer

$$\boldsymbol{w_{j.k}^{hIdden \to output}} = \pm \frac{\mathbf{1}}{\mathbf{2n}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n}} \frac{\mathbf{1}}{\boldsymbol{f\left(\sum w_{\alpha.j}^{\alpha nput \to h\alpha dden} y_j\right)}} \tag{3.12}$$

where,

k = column connecting hidden nodes with input nodes

β = column connecting hidden nodes with input nodes

n = the total number of inputs

y_β= the inputs of network

w = the vector weight

$f(.)$ = function of repetitive /linear activation (equal to 1)

Step 7: Find the second synapse's activation function, $f(.)$, which connects the hidden layer to the output layer.

$$f_2(.) = 1(.), \tag{3.13}$$

where $f_{purelin}$: $\Re \to [-1;1]$

becoming

$$\begin{aligned} y_k &= 1\left(\sum_{j-1}^{m} w_{jk}\, y_j\right) \qquad (3.14) \\ &= 1(x_k) \\ &= x_k \\ &= \sum_{j-1}^{m} w_{jk}\, y_j \end{aligned}$$

Step 8: The output neurons' maximum likelihood estimator, or accumulated error signal, is calculated by Equation 3.16 after pattern t presentation.

$$\varepsilon_t = \frac{1}{2}\sum_{k=1}^{n}[y_t - y_k]^2 \qquad (3.15)$$

where

ε_t = errors at time t,

y_i = actual output

y_k = target output.

Step 9: The primary objective is to minimize ε_{avg} by adjusting the free parameters $w_{\beta k}$ and $w_{i\beta}$. Equation 3.16's partial derivatives were computed in relation to the weights, $w_{\beta k}$.

So, for the method illustration, let

$$y_t - y_k = y_k - y_k^d \qquad (3.16)$$

Therefore

$$\frac{\partial \varepsilon_t}{\partial w_{jk}} = \frac{1}{T}\sum_{t=1}^{T}\left(y_k - y_k^d\right)\frac{\partial y_k}{\partial x_k}\frac{\partial x_k}{\partial w_{jk}} \qquad (3.17)$$

$$= \frac{1}{T}\sum_{t=1}^{T}\left(y_k - y_k^d\right)f'(x_k)y_j$$

The whole goal is to reduce ε_t by modifying the free parameters $w_{\beta k}$ and $w_{i\beta}$.

Step 10: Calculating the partial derivatives of Equation 3.14 with regard to the weights $w_{\beta k}$. is necessary to reach the goal. Thus,

$$\frac{\partial \varepsilon_t}{\partial w_{jk}} = \frac{1}{T}\sum_{t=1}^{T}\left(y_k - y_k^d\right)\frac{\partial y_k}{\partial x_k}\frac{\partial x_k}{\partial w_{jk}}$$

$$= \frac{1}{T}\sum_{t=1}^{T}\left(y_k - y_k^d\right)f'(x_k)y_j \qquad (3.18)$$

The chain-rule was utilized to calculate the partial derivatives in Equation 3.18. Let

$$\delta_k = \left(y_k - y_k^d\right) f'(x_k) \tag{3.19}$$

According to the chain-rule, if

$$y = f(g(x)) \text{ then } \frac{dy}{dg}\frac{dg}{dx} \tag{3.20}$$

As a result, changing k δ_k in Equation 3.19 results in

$$\frac{\partial \varepsilon_t}{\partial w_{jk}} = \frac{1}{T}\sum_{t=1}^{T} \partial_k y_j \tag{3.21}$$

Partial derivatives of β existed because the network was considered to be totally connected to a set of k. The cumulative gradient of the network's second level consisted of these derivatives (Equation 3.21), following the flow in Figure 3.8.

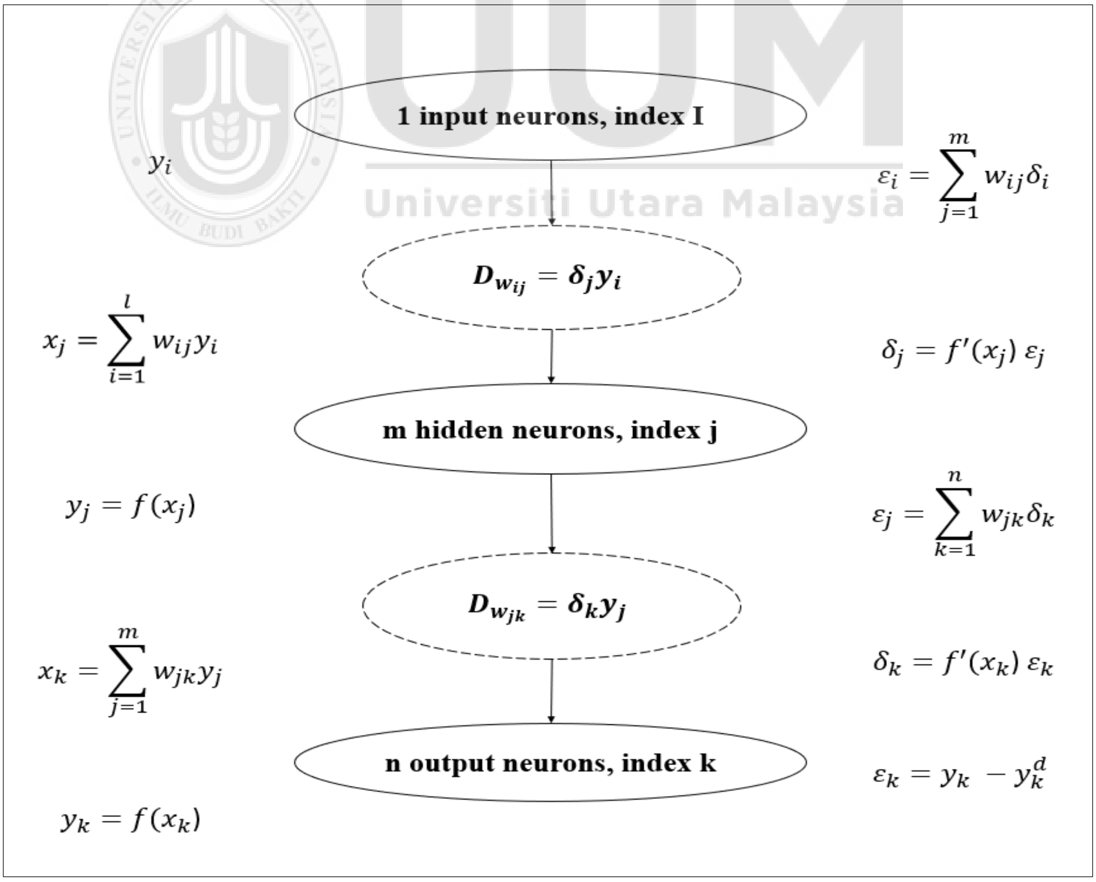

Figure 3.8. Information Flow in the Backward Path of a Two-Layer Neural Network

Step 11: The partial derivatives of Equation 3.14 with respect to $w_{i\beta}$ are determined in a similar manner (the network in the first level). The chain-rule is applied twice to

$$\frac{\partial \varepsilon_t}{\partial w_{\alpha j}} = \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{n}\left(y_k - y_k^d\right)\frac{\partial y_k}{\partial x_k}\frac{\partial x_k}{\partial y_j}\frac{\partial y_j}{\partial x_j}\frac{\partial x_j}{\partial w_{\alpha j}}$$

$$= \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{n}\left(y_k - y_k^d\right)f'(x_k)w_{jk}f'\left(x_j\right)y_\alpha. \tag{3.22}$$

The Equation 3.22 is simplified by substituting the auxiliary term ∂_k which produces

$$\frac{\partial \varepsilon_t}{\partial w_{\alpha j}} = \frac{1}{T}\sum_{t=1}^{T}\left[\sum_{k=1}^{n}\partial_k w_{jk}\, f'\left(x_j\right)\right]y_\alpha. \tag{3.23}$$

The Equation 3.23 is simplified by substituting an additional auxiliary term, where

$$\delta_j = \sum_{k=1}^{n}\partial_k w_{jk}\, f'\left(x_j\right) \tag{3.24}$$

becoming Equation 3.25. This is the first network level in cumulative gradient.

$$\frac{\partial \varepsilon_t}{\partial w_{\alpha j}} = \frac{1}{T}\sum_{t=1}^{T}\delta_j y_\alpha \tag{3.25}$$

Let's consider a single training pattern to describe the backpropagation algorithm. The outputs y_i and y_β are computed on the forward path as shown in Figure 3.6. The level one and level two partial derivatives are represented on the backward route by the products $\delta_k y_\beta$ and $\delta_\beta y_i$ respectively. In other words, the auxiliary terms δ_β and δ_k [Equations 3.24 and 3.25] implicitly carry error information throughout the network.

Step 12: Based on the stopping criteria, steps 1 through 13 should be repeated until the training stops.

Step 13: Take the NAR model's residuals and add them as additional inputs to the network.

Step 14: Estimated $\hat{y}_{t+\alpha}$

$$\boldsymbol{y_{t+\alpha} = \quad f\left(y_{t-1}, \dots, y_{t-p}, \varepsilon_{t-1}, \dots, \varepsilon_{t-q}\right) + \varepsilon_{t+\alpha}} \tag{3.26}$$

$$\hat{y}_{t+\alpha} = \quad f\left(\hat{y}_{t-1}, \dots, \hat{y}_{t-p}, \hat{\varepsilon}_{t-1}, \dots, \hat{\varepsilon}_{t-q}\right) \tag{3.27}$$

$$\hat{y}_{t+\alpha} = \quad y_{t+\alpha} = f(y_t + \varepsilon_t) + \varepsilon_{t+\alpha} \tag{3.28}$$

$$\hat{y}_{t+\alpha} = \quad \hat{y}_{t+\alpha} = f(\hat{y}_t + \hat{\varepsilon}_t) \tag{3.29}$$

where

y_t = actual values (daily returns stock market data) at time t,

$y_{t-1}, \dots, y_{t-p}$ = p^{th} lag input values (daily returns stock market data),

$\hat{y}_{t+\alpha}$ = estimated values in forecast at i^{th}-step-ahead, i= 0, 1, 2,…m,

$y_{t+\alpha}$ = stock market data predict values at i^{th}-step-ahead, i= 0, 1, 2,…m,

$f(.)$ = the function of nonlinear,

ε_t = the errors values of forecast at time t,

$\hat{\varepsilon}_{t+\alpha}$ = estimated errors of the model at i^{th}-step-ahead, i = 0, 1, 2, …, m.

Step 15: Same as Step 12.

A normal distribution with a mean of zero is represented as $\varepsilon_t \sim N\left(0,\sigma^2\right)$ where σ^2 denotes the variance. The error terms ε_t were regarded as independent and identically distributed random variables (iid.). The homoscedasticity assumption or that the error term variance, $\sigma_\varepsilon{}^2$, is constant across the time, is made. Additionally, in each instance where $E\left(\varepsilon_t \varepsilon_t - 1 = 0\right)$, the error terms are uncorrelated, i.e., meaning that there is no covariance between them. Since there is no correlation between the independent variables and the error terms, $E\left(\varepsilon_t x_{jt} = 0\right)$ is the covariance. Finally, the error terms comply with the normality assumption.

3.5.1 Least Median Squares (LMedS)

In this phase, Mean Square Error (MSE) was replaced with LMedS. The process continued from Step 15 in Phase 1, followed by the subsequent step.

Step 16: Start with a minimal value of $med(\hat{\varepsilon}_t{}^2) = 0$

Step 17: Identify the predicted output value, y_k

$$\hat{\varepsilon}_{1t} = \frac{1}{2}\sum_{k=1}^{n}[y_\alpha - y_k]^2 \tag{3.30}$$

where, $\hat{\varepsilon}_{1t}$ is the expected errors at time *t,* while y_k and y_t denoted as predicted and actual output respectively.

3.5.2 Date Palm Seed Growth Algorithm (DPSG)

DPSG is the metaheuristic algorithm where it is inspired by the growth of date palm seed. The farmers in the Middle East have their own way so that the roots of palm trees can be strong and survive in arid and dry soil conditions. This plant can grow in dry, arid, barren lands, even in lands that are often hit by terrible desert storms. The strength of the palm tree actually lies in the roots it possesses. That is why this tree is considered a resilient tree.

3.5.3 DPSG-LMedS

The LMedS estimator has low efficiency and is not enough. Therefore, the DPSG algorithm need to hybrid with LMedS become Date Palm Seed Growth Least Median Square (DPSG-LMedS), which enhanced the model's efficiency and accuracy in predicting the stock market.

The DPSG-LMedS algorithm was developed in this research and is formulated based on the Pseudo-code presented below.

Date Palm Seed Growth (DPSG)

```
1.  Initialize seed: Create a variable representing the date palm seed.
2.  Place seed in soil: Plant the seed in a pot or soil, ensuring it's adequately buried.
3.  Initialize stone: Create a variable representing the stone or weight.
4.  Place stone on top: Position the stone on the soil surface above the planted seed.
5.  Set water and sunlight conditions: Ensure the pot or soil receives appropriate
    water and sunlight.
6.  While seed is not a tree:
7.      If seed receives adequate water and sunlight:
8.          Allow time for growth: Let the seed absorb water and sunlight for a
            certain duration.
9.          Monitor growth conditions: Check if the seed is sprouting or growing.
10.     Else:
11.         Adjust water and sunlight: Provide more water and sunlight for optimal
            growth.
12.     End If
13.     If the seed's roots are strong enough to lift the stone:
14.         Remove the stone: Automatically lift the stone from the soil.
15.     End If
16. End While
17. Date palm tree is fully grown.
```

Step 18: Find the median of estimated errors at time *t*, DPSG-LMedS into BPNN model

$$\varepsilon_{med} = med(\hat{\varepsilon}_{1t}) = \frac{1}{2}\sum_{k=1}^{n}[y_\alpha - y_k]^2 \tag{3.31}$$

where,

$\hat{\varepsilon}_{1t}$ = the expected errors at time *t,*

y_k = predicted output,

y_t = actual output.

Step 19: Minimize ε_{med} by iterative training until criterion function ε_{med} is below minimal value in Step 18.

Step 20: If $med(\hat{\varepsilon}_{1t}) < med(\hat{\varepsilon}_t)$, replace $med(\hat{\varepsilon}_t)$ with $med(\hat{\varepsilon}_{1t})$

Step 21: Stop training once the stopping criterion is achieved. Keep the current best value of $med(\hat{\varepsilon}_t)$

Step 22: Remove outliers from the current dataset using the best criterion function value $\varepsilon_{med} = med(\hat{\varepsilon}_t)$ and current robust standard deviation (RSD). After initial training, RSD is calculated

$$\sigma_r = \Gamma * \left(1 + \frac{5}{(N-p)}\right)\sqrt{\varepsilon_{med}} \tag{3.32}$$

where,

ε_{med} = the best achieved LMedS error value,

N = training set size,

p = the input vector dimension,

$1 + \frac{5}{(N-p)}$ = factor to compensate the effect of small sample size.

Γ = the constant to provide better efficiency for the clean data with Gaussian noise.

Step 23: Retrain the network on the reduced data without outliers, minimizing ε_{med}. Remove from the training set all patterns associated with residuals exceeding a threshold based on the RSD

$$\varepsilon_{\alpha}^{2} \geq 2.5 * \sigma_{\tau} \tag{3.33}$$

Step 24: Stop the network if the network LMedS performance achieved any of the stopping criteria. Otherwise go to Step 21. After the network stops, the best LMedS error value, $\varepsilon^{*}{}_{med}$ can be achieved (Mahsereci et al., 2017)

3.5.4 Enhanced BPNN model using DPSG-LMedS

The BPNN model exhibited issues with its estimators, as its cost function was not entirely robust. Consequently, to address this limitation, the model was enhanced by minimizing error using the DPSG algorithm and replacing OLS with LMedS.

The process of the enhanced algorithm begins with the initialization of the $E*D$ roots population and relevant parameters, followed by setting the generation number $K=1$. The algorithm then evaluates F_i for all individuals and categorizes them into the main-roots and lateral-roots groups based on sorted F_i values.

Subsequent operations include main-roots regrowth, nutrient adjustment, and main-roots branching, defined by $E = E + n_i$. Lateral-roots undergo similar regrowth and nutrient adjustments, followed by dead-root elimination. Further optimization steps involve non-dominated sorting, farthest candidate selection, and recording the best solution. The stop criterion is verified, and if unmet, the generation number increments by $K = K + 1$, continuing the process until convergence. Figure 3.9 presents a flowchart detailing the steps of an enhanced algorithm aimed at improving the BPNN model.

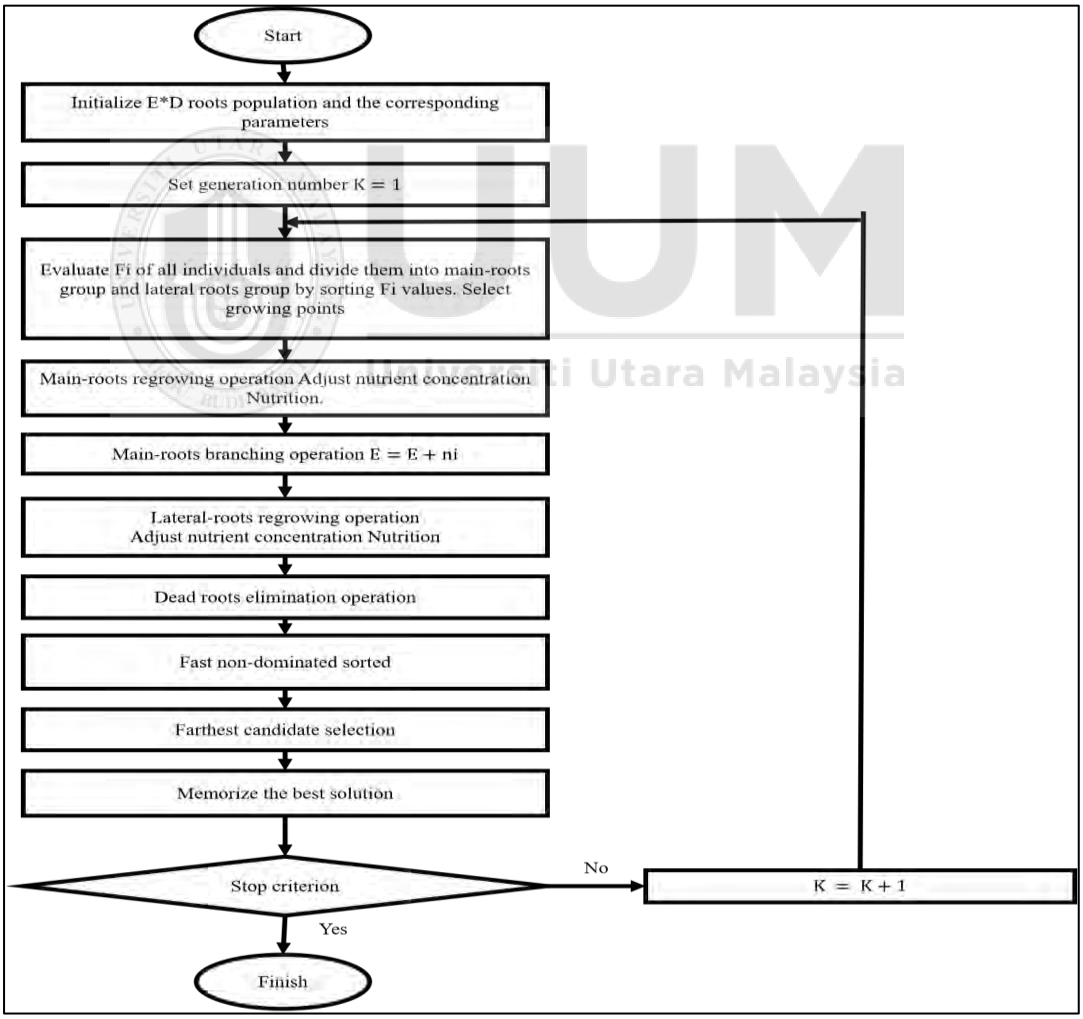

Figure 3.9. Contribution of the enhanced algorithm

Figure 3.10 provides a clearer depiction of the enhanced BPNN model, highlighting the contribution of the enhanced algorithm in the MSE calculation process within the model.

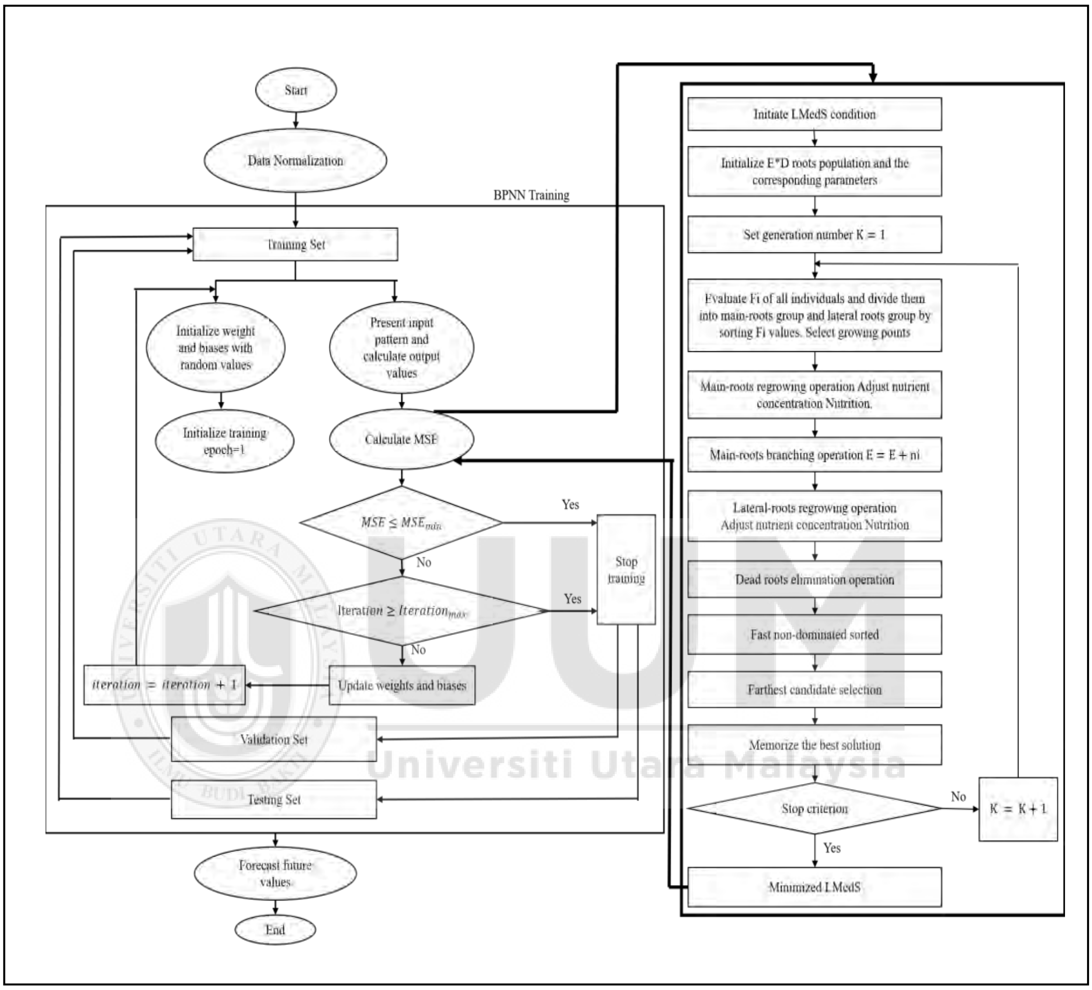

Figure 3.10. Enhanced Backpropagation Neural Network

3.6 Evaluation of Prediction Model using Error Measure

The models were evaluated by using two types of error measure which are Root Mean Square Error (RMSE) and Geometric Root Mean Square Error (GRMSE). The details of the function and formula for both error measures were explained in the next section.

3.6.1 Root Mean Square Error

Root Mean Square Error (RMSE) is used to explain how tightly the data is clustered around the line of best fit (Cinembiri et al., 2023). The error measure or the standard criterion also known as RMSE is used by mean experts that frequently used to compare to the model's anticipated performance and to assess a model's applicability to a particular set of data (Kantz & Schreiber, 2004). The RMSE can be calculated by using equation 3.34.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t-1}^{n}\left(Y_t - \hat{Y}_t\right)^2} \qquad (3.34)$$

where,

n = the number of predictions,

$\hat{Y}_t$ = the forecasted value at interval *t*,

Y_t = the actual value at interval *t*.

The model that gives the smallest value in RMSE is a fit model to forecasting.

3.6.2 Geometric Root Mean Square Error

Geometric Root Mean Square Error (GRMSE) serves as a method for addressing the issue of outliers, which often impacts the precision of error measurements, especially when dealing with a notably large error term resulting from an inaccurate forecast (Domanski & Wieclawski, 2015; Kolassa, 2016). The GRMSE can be defined as follows:

$$GRMSE = \left(\prod_t^n \left[Y_t - \hat{Y}_t\right]^2\right)^{\frac{1}{2n}} \tag{3.35}$$

where,

n = the number of predictions,

$\hat{Y}_t$ = the forecasted value at interval *t*,

Y_t = the actual value at interval *t*.

It is common for forecasters to utilize multiple error measures to ensure the consistency and accuracy of the result in evaluation. Table 3.5 presented a summary of the error measures that were used to assess the performance of each model and to select the best model.

Table 3.5

Error Measures to Assess the Forecasting Model Performance

No.	Error Measures	Notation	Formula
1.	Root Mean Square Error	RMSE	$\sqrt{\frac{1}{n}\sum_{t-1}^{n}\left(Y_t - \hat{Y}_t\right)^2}$
2.	Geometric Root Mean Square Error	GRMSE	$\left(\prod_t^n \left[Y_t - \hat{Y}_t\right]^2\right)^{\frac{1}{2n}}$

*Remark: The least error measures values are the best model

3.7 Convergence Test

In this section, the research used convergence test to validate the proposed algorithm of DPSG-LMedS. Convergence test is a common technique to validate an algorithm as suggested (Mutinda & Geletu, 2025). Based on the learning curve that be analyzed, the value of RMSE from each lag were tested with the maximum number of epochs is 1000. In a converging learning curve, the RMSE value decreased steadily as the number of epochs increased. Eventually, the RMSE stabilized, indicating that the model had learned the underlying patterns in the data and that further training did not significantly improve performance.

Figure 3.11 illustrates an example of a convergent model.

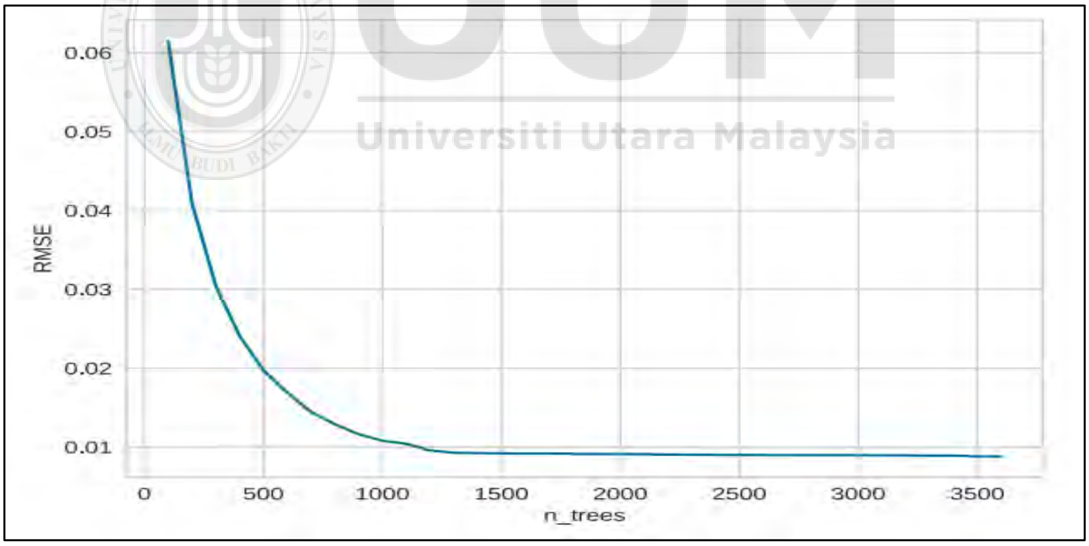

Figure 3.11. Example of Convergence Result

Note: Adapted from Song et al. (2022)

3.8 Summary

This chapter presented the methodology implemented for the study. The first phase encompassed preprocessing steps, including data scraping and diagnostic tests. In the second phase, an enhanced BPNN model was developed for FBM KLCI dataset. The performance of both the ordinary and enhanced models was evaluated using simulation data and FBM KLCI datasets. Subsequently, error measures specifically RMSE and GRMSE were employed to select the optimal model. The final phase involved validating the model and conducting forecasting on the dataset.

CHAPTER 4
DATA ANALYSIS AND DISCUSSION

4.1 Introduction

This part explains the result from the analysis of the backpropagation neural network (BPNN). The chapter is divided into five sections, which respect to the four objectives of the study. The data background explained in Section 4.2. The first objective was to identify the severity of outliers' problems of Financial Times Stock Exchange (FTSE) Bursa Malaysia Kuala Lumpur Composite Index (FBM KLCI) dataset identified based on the analysis in Section 4.3 Diagnostic Test. In Section 4.3, the test for outliers problem was discussed. In order to achieve the second objective which is to evaluate the performance of the ordinary models on real FBM KLCI dataset which consist of outliers' problems, Section 4.4 presented the numerical comparisons of both the BPNN model and Enhanced BPNN models, where their performance was evaluated on a real dataset. The third objective is to develop an enhanced BPNN for FBM KLCI dataset with outliers' problems. Therefore, to fulfill the third objective, Section 4.5 tested the enhanced BPNN on Simulated Dataset. After the enhanced BPNN model was tested, forecasting on the FBM KLCI stock market dataset was conducted for three-step ahead in Section 4.6. Last but not least, Section 4.7 shows the convergence test to validate the enhanced BPNN model to answer the fourth objective which is to validate the enhanced BPNN using time series bootstrap technique. In Section 4.8, the summary of this chapter has been done.

4.2 Descriptive Analysis

The data background used in this research was the FBM KLCI dataset, which was retrieved from Yahoo Finance. The data that was scraped using Python code in Spyder Software has been cleaning and proceeded to diagnostic test, performance comparison, forecasting and convergence test.

Figure 4.1 illustrated the closing prices of FBM KLCI from 2nd January 2018 to 30th December 2022. Between 2018 and early 2020, a general decline in closing prices was observed. In early 2020, a sharp drop occurred, followed by a recovery and subsequent fluctuations until the end of the year 2022. This significant decline was likely attributable to major market events, such as the onset of the COVID-19 pandemic, which resulted in substantial market volatility and losses. Following the sharp drop, the closing prices recovered and exhibited fluctuations, indicating that the market had begun to stabilize after the initial shock.

Figure 4.1. FBM KLCI stock market closing prices from 2nd January 2018 to 30th December 2022

Table 4.1 presented the results of descriptive statistics, including the mean price, median price, standard deviation, variance, minimum price, maximum price, skewness, and kurtosis. The mean price indicated that the average closing price was 1,598.29. Additionally, the median price of 1,588.08 suggested that half of the closing prices were below 1,588.08 and half were above this value. The stock prices typically deviated from the mean by 119.64636, with a higher standard deviation signifying greater volatility and a higher variance indicating a wider spread of stock prices.

Furthermore, the lowest and highest closing prices within the specified period were 1,219.72 and 1,895.18, respectively. The positive skewness of 0.36615 implied a slight right skew in the distribution of stock prices, suggesting a greater occurrence of higher prices. Lastly, the kurtosis value of 0.12583 indicated that the distribution of stock prices had slightly heavier tails compared to a normal distribution, signifying a moderate presence of extreme values.

Table 4.1

Descriptive Statistics for FBM KLCI data

Descriptive Statistics	**Value**
Mean Price	1598.29
Median Price	1588.08
Standard Deviation	119.64636
Variance	14315.25142
Minimum Price	1219.72

Table 4.1 (Continued)

Descriptive Statistics for FBM KLCI data

Descriptive Statistics	**Value**
Maximum Price	1895.18
Skewness	0.36615
Kurtosis	0.12583

Figure 4.2 illustrate the plot graph with a line of 50-day and 200-day simple moving average (SMA). It shows that there is 2 Golden Cross (GC) and 2 Death Cross (DC). Moreover, the stock price shows the sustained downward start from 2018 until mid-year 2020. Then, the price shows a sign of a good time to consider buying and after mid-year 2021 it shows a sign of a good time to consider selling until early year 2022. After that, it gave a sign to consider buying again until mid-year 2022 and it start shows the sustained downward trend until end year 2022.

Figure 4.2. 50-day and 200-day Simple Moving Average

Figure 4.3 illustrated the daily return of the FBM KLCI stock market and bands at ±119.64636 standard deviation. Around the beginning of 2020, a notable spike was observed, where the daily return value sharply increased before declining significantly. This spike likely indicated a major market event or anomaly during that period, potentially related to global events such as the COVID-19 pandemic, which had a substantial impact on financial markets. The plot revealed periods of increased volatility, during which the daily returns fluctuated more widely. These periods may have corresponded to market turbulence, economic news, or other factors influencing the asset's price. The remainder of the plot displayed relatively smaller fluctuations in daily returns, suggesting periods of more stable market conditions.

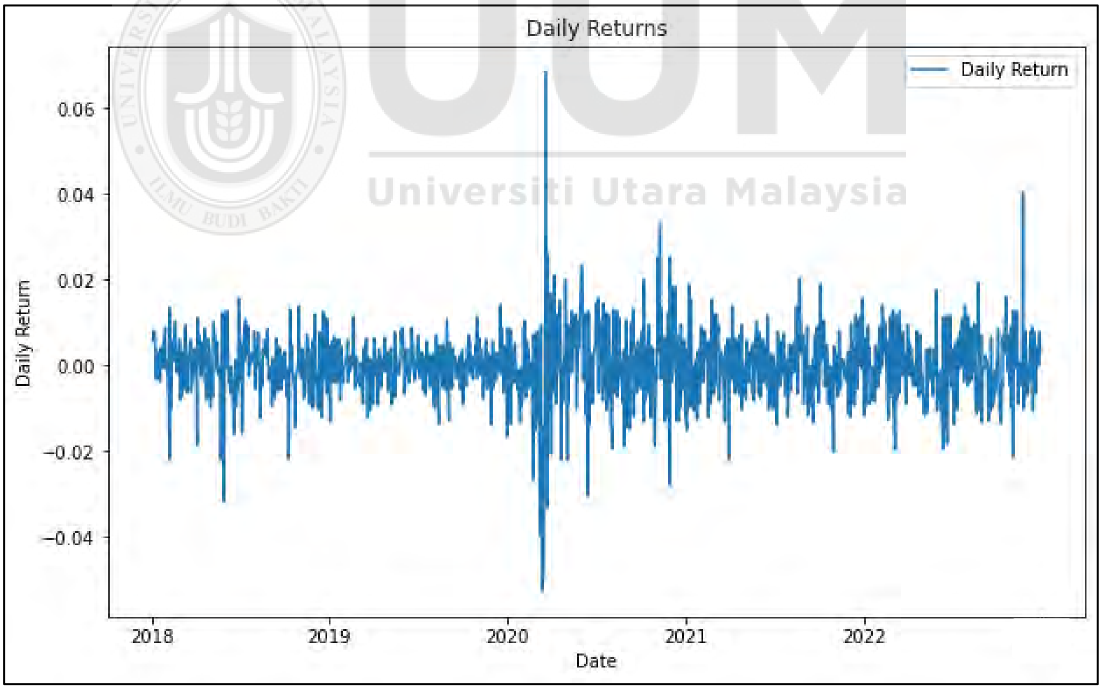

Figure 4.3. Daily return of FBM KLCI and bands at ± 119.64636 standard deviation

Figure 4.4 illustrates the subset splitting process applied to the FBM KLCI stock market closing prices. The dataset was divided into training and testing sets, with an allocation ratio of 85% and 15%, respectively. The training set encompasses the majority of historical data, ensuring that the model learns from extensive market fluctuations and trends. Meanwhile, the testing set provides a dedicated portion for evaluating predictive performance, assessing the model's ability to generalize to unseen data. This structured division is crucial for improving forecasting accuracy in financial modeling.

Figure 4.4. Subset splitting for FBM KLCI stock market closing prices

Table 4.2 presents a comprehensive breakdown of the total sample size (N) and the partitioning of data for training and testing across different data types. The table categorizes the datasets into distinct groups, specifying the number of samples allocated for training and testing. The training set comprises 85% of the total dataset, ensuring that the model learns patterns effectively, while the remaining 15% is designated for testing to evaluate predictive performance.

Table 4.2

Data Partitioning for Training and Testing Sets for FBM KLCI data

No.	Data Type	Total Sample Size (N)	Training (85%)	Testing (15%)
1	Real Dataset (FBM KLCI Stock Market Closing Prices)	1222	1039	183
2	Simulated 1–Dimensional Data (Data Set I)	2000	1500	300
3	Simulated 1–Dimensional Data (Data Set II)	2000	1500	300

4.3 Outliers Detection

In order to achieve a first objective, Section 4.3 was completed, where a diagnostic test was analyzed to check whether the outliers problems existed in the FBM KLCI stock market data.

The outliers of FBM KLCI stock prices were identified using box plot as suggested by Shehadeh et al., (2022). This research was tested the outliers using Python code for different number of lags which is 5, 10, 20, 30, 40 and 50. Based on Figure 4.5, it is clearly seen that the boxplot for each lags has some outliers in the closing price data. All the boxplot from each lag is normal. However, based on Table 4.3, the percentage of outliers from lag 40 and lag 50 have more outliers compared to the first four lags. It shows that the outliers that up to 65% outliers exist at lag 40 and lag 50.

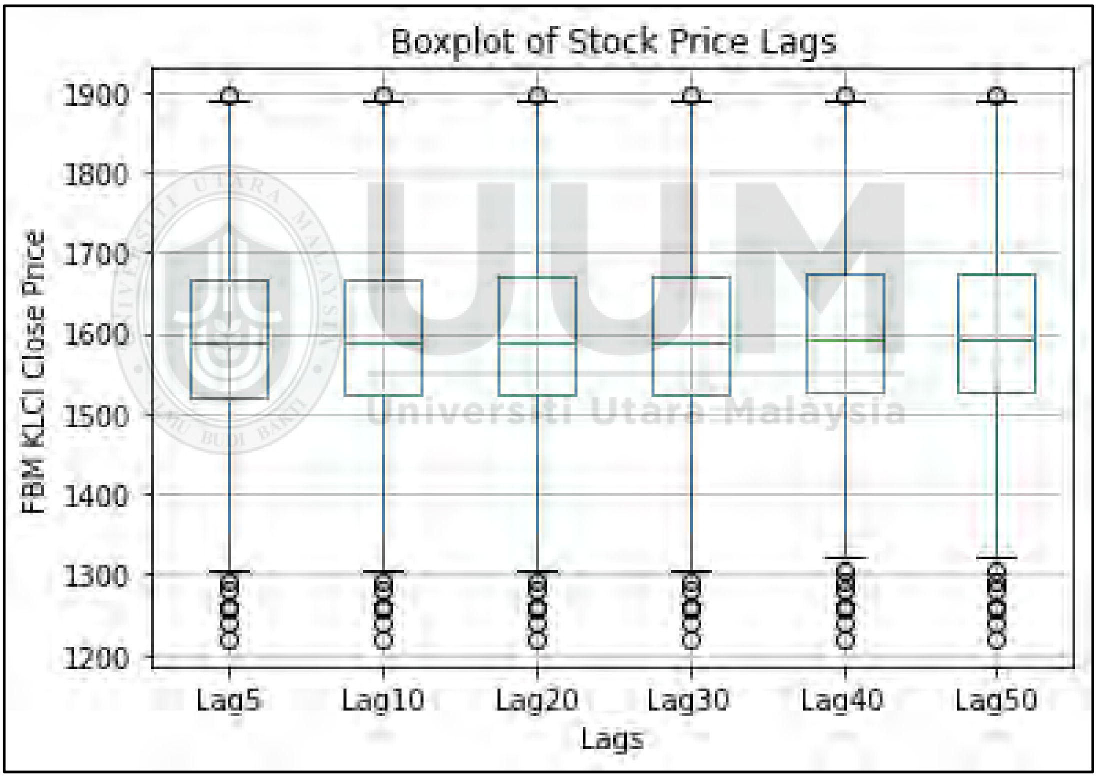

Figure 4.5. Outliers which exist in FBM KLCI Stock Prices for different lags

Table 4.3

Percentage of outliers for different lags of FBM KLCI Dataset

No. of lags	No. of Outliers	Outlier (%)
5	7000	57.33%
10	7000	57.33%
20	7000	57.33%
30	7000	57.33%
40	8010	65.52%
50	8010	65.52%

4.4 Performance Comparisons for three models

To accomplish the second objective, the real FBM KLCI dataset was analyzed by measuring the Root Mean Square Error (RMSE) and Geometric Root Mean Square Error (GRMSE) with different input lags and hidden nodes using three models which is BPNN model, BPNN with LMedS model and Enhanced BPNN models. The enhanced model by run the coding as written in **Appendix A**.

Referring to **Appendix B**, Table 4.4 was simplified and compared the three model, BPNN model, BPNN with LMedS model and Enhanced BPNN models using FBM KLCI stock market closing price dataset. The results show that the enhanced BPNN model produced the smallest value of RMSE and GRMSE compared to ordinary BPNN model. Based on the configuration from the result of RMSE and GRMSE, it shows that the performance of the enhanced model is the best when used with the higher number of input lags and the lower number of hidden nodes.

Table 4.4

Comparisons of RMSE and GRMSE value for Three Different Models using FBM KLCI Stock Market Closing Price Dataset

Type of Model	RMSE		GRMSE	
	Training	Testing	Training	Testing
BPNN	0.633592 (40-5)	0.464590 (30-25)	1.60661 (40-5)	1.455309 (30-25)
BPNN with LMedS	0.553211 (40-5)	0.461132 (15-20)	1.516604 (35-5)	1.447603 (25-20)
Enhanced BPNN	0.521857 (35-5)	0.454544 (20-10)	1.477958 (35-5)	1.441731 (20-10)

Note: number in bracket () is the configuration number

4.5 Enhanced Backpropagation Neural Network on Simulated Dataset

This research enhanced the backpropagation neural network (BPNN) by replacing the Mean Square Error (MSE) with Least Median Square (LMedS). Moreover, the enhanced BPNN was further enhanced by combining the BPNN model with a metaheuristic algorithm which was the Date Palm Seed Growth Algorithm (DPSG) algorithm. Therefore, the model was renamed as DPSG-LMedS. Then, the simulation was done after enhancing the BPNN model.

The evaluation on the enhanced model has compared the RMSE and GRMSE value from each input lag and hidden node using the Simulated Data Set I and Simulated Data Set II. Based on RMSE and GRMSE value, **Appendix C** and **E** presents the result of the enhanced model performance that was tested on Simulated Data Set I. The enhanced model performance was compared with the different number of outliers percentages (0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55%, 60% and 65%), lags input and hidden nodes.

Based on Table 4.5, the smallest value of RMSE and GRMSE were at 65% of outliers where the value was 0.309335 (20-10) and 1.300325 (20-5) respectively. Based on the performance of the enhanced model checking, the best configuration shows the smaller number of hidden nodes and the average number of input lags.

Table 4.5

The comparison of RMSE and GRMSE value in the training phase using Data Set I

Outliers (%)	**RMSE**	**GRMSE**
0	0.39221 (5-25)	1.365482 (25-5)
5	0.373162 (5-5)	1.346845 (5-5)
10	0.429335 (35-10)	1.416202 (35-10)
15	0.424814 (40-10)	1.410276 (40-10)
20	0.42305 (10-5)	1.407287 (10-5)
25	0.338947 (20-5)	1.304435 (20-5)
30	0.386238 (35-5)	1.354567 (25-25)
35	0.438536 (35-10)	1.383593 (10-15)
40	0.379449 (40-5)	1.374507 (15-5)
45	0.375665 (10-5)	1.353026 (20-5)
50	0.373053 (40-5)	1.376593 (5-30)
55	0.354103 (40-5)	1.325635 (15-5)

Table 4.5 (continued)

The comparison of RMSE and GRMSE value in the training phase using Data Set I

Outliers (%)	RMSE	GRMSE
60	0.365543 (10-5)	1.404695 (20-30)
65	**0.309335 (20-10)**	**1.300325 (20-5)**

Note: number in bracket () is the configuration number

Meanwhile, **Appendix D** and **F** analyzed in the testing phase using Simulated Data Set I to obtain the RMSE and GRMSE value respectively. Table 4.6 presents the smallest value of RMSE and GRMSE from a different number of outliers based on the testing phase using Simulated Data Set I.

However, in the testing phase, the smallest value of RMSE and GRMSE were at the 25% of outliers where the values were 0.320704 (20-5) and 1.257188 (20-5) respectively.

Table 4.6

The comparison of RMSE and GRMSE value in the testing phase using Data Set I

Outliers (%)	RMSE	GRMSE
0	0.524275 (35-20)	1.491123 (35-35)
5	0.35629 (5-5)	1.294195 (5-5)
10	0.515865 (15-20)	1.489631 (15-20)
15	0.509376 (40,10)	1.480423 (40-10)
20	0.518792 (25-5)	1.493716 (25-5)

Table 4.6 (continued)

The comparison of RMSE and GRMSE value in the testing phase using Data Set I

Outliers (%)	RMSE	GRMSE
25	**0.320704 (20-5)**	**1.257188 (20-5)**
30	0.371707 (20-5)	1.310897 (20-5)
35	0.529537 (25-35)	1.500245 (25-5)
40	0.519956 (10-5)	1.474956 (25-10)
45	0.540243 (30-5)	1.502453 (5-20)
50	0.35629 (15-5)	1.497732 (35-25)
55	0.370807 (20-5)	1.504534 (20-5)
60	0.330891 (15-5)	1.432913 (30-10)
65	0.349321 (15-5)	1.326747 (20-5)

Note: number in bracket () is the configuration number

Meanwhile, **Appendix G** and **I** shows the RMSE and GRMSE value based on Simulated Data Set II in the training phase. Table 4.7 simplifies the result from **Appendix G** and **I** to present the smallest value of RMSE and GRMSE from different number of outliers based on the training phase using Simulated Data Set II. However, in the training phase, the smallest value of RMSE and GRMSE were at the 65% of outliers where the value was 0.211321 (30-35) and 1.030024 (20-20) respectively.

Table 4.7

The comparison of RMSE and GRMSE value in the training phase using Data Set II

Outliers (%)	RMSE	GRMSE
0	0.287707 (15-15)	1.207665 (10-5)
5	0.288416 (30-10)	1.278256 (30-10)
10	0.288307 (35-15)	1.277456 (15-20)
15	0.262842 (15-5)	1.248533 (15-5)
20	0.274904 (40-25)	1.261456 (40-25)
25	0.284321 (25-5)	1.253568 (30-30)
30	0.285432 (30-15)	1.204543 (35-20)
35	0.285009 (25-10)	1.235035 (15-35)
40	0.25421 (25-45)	1.272884 (30-10)
45	0.254032 (20-45)	1.211343 (40-20)
50	0.261945 (5-30)	1.235336 (20-10)
55	0.244938 (35-10)	1.209454 (25-15)
60	0.265741 (40-45)	1.261456 (5-30)
65	**0.211321 (30-35)**	**1.030024 (20-20)**

Note: number in bracket () is the configuration number

Moreover, **Appendix H** and **I** show the RMSE and GRMSE value based on Simulated Data Set II in the testing phase. Table 4.8 presents the smallest value of RMSE and GRMSE from a different number of outliers based on the testing phase using Simulated Data Set II. However, in the testing phase, the smallest value of

RMSE and GRMSE were 0.2903942 (15-5) at the 65% of outliers and 1.2150944 (30-5) at the 10% of outliers respectively.

Table 4.8

The comparison of RMSE and GRMSE value in the testing phase using Data Set II

Outliers (%)	**RMSE**	**GRMSE**
0	0.303667 (10-5)	1.341769 (5-20)
5	0.322399 (40-5)	1.2222512 (35-5)
10	0.313086 (35-5)	**1.2150944 (30-5)**
15	0.318456 (15-5)	1.2207208 (10-5)
20	0.322231 (5-5)	1.226402 (35-5)
25	0.628105 (25-5)	1.5509367 (25-5)
30	0.322814 (10-5)	1.3063653 (5-5)
35	0.3125483 (25-5)	1.32419367 (10-5)
40	0.32102493 (10-5)	1.5487746 (5-5)
45	0.3395302 (5-5)	1.5596397 (35-5)
50	0.32139472 (30-5)	1.5421367 (15-5)
55	0.31940845 (5-5)	1.5594327 (40-5)
60	0.32193024 (20-5)	1.5509367 (30-5)
65	**0.2903942 (15-5)**	1.5829787 (40-5)

Note: number in bracket () is the configuration number

4.6 Convergence Test

The convergence test was conducted after the BPNN model had been enhanced. This research tests each lag to identify the number of epochs that were more suitable for this validation test. **Appendix K** shows the results of convergence test for each lag that was tested for a maximum number of epochs 1000.

Based on Figure 4.6, the convergence test value for the enhanced model was illustrated using the line graph. It is clearly seen that lag 25 shows better convergence compared to other lags. Furthermore, the best performance of the enhanced model shows at epoch 100 which is at lag 25.

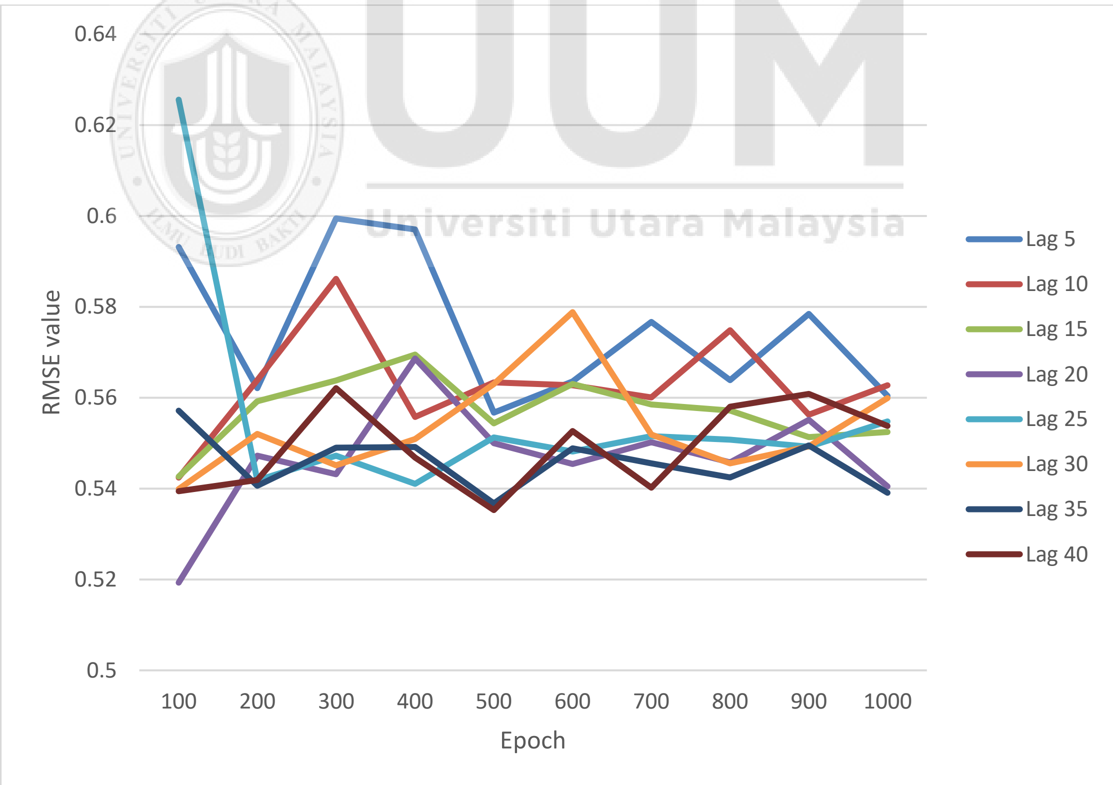

Figure 4.6. Convergence Test from different epoch

4.7 Forecasting

The final step in this analysis involved forecasting. The forecasting method implemented was a 3-step-ahead approach, in which the FBM KLCI datasets were analyzed using three models: BPNN, BPNN with LMedS, and the enhanced BPNN model. As suggested by Cheng et al. (2023), the purpose of using the 3-step-ahead prediction is to estimate the next three data points in a sequence, providing better insights into future trends and potentially enabling more accurate long-term predictions. These models were evaluated to determine the accuracy of the results in comparison to the actual value of the FBM KLCI stock market.

Each model generated three-step-ahead forecasts, which were subsequently compared to the actual closing prices. Table 4.9 demonstrated that the three-step-ahead values from the enhanced BPNN model were closer to the real values. This finding indicated that the enhanced BPNN model was more accurate compared to the BPNN model and the BPNN with LMedS model.

Table 4.9

Prediction 1-step ahead, 2-step ahead, and 3-step ahead

Model	1-step ahead	2-step ahead	3-step ahead
BPNN	1496.66	1496.42	1497.92
BPNN with LMedS	1495.06	1495.67	1497.69
Enhanced BPNN	**1494.66**	**1496.63**	**1497.66**
Real Value	1487.26	1483.38	1473.91

4.8 Summary

This chapter presented the data analysis and discussion that addressed all of the study's objectives. The first objective was fulfilled by identifying the severity of the outlier problem. Next, an enhanced BPNN model for FBM KLCI stock market was developed. Subsequently, the performance comparison of the enhanced with the ordinary BPNN model and BPNN-LMedS model were applied to FBM KLCI data and be conducted using RMSE and GRMSE to determine the model with the smallest error value. Thereafter, the reliability of the enhanced BPNN model were analyzed with varying numbers of epochs.

CHAPTER 5
CONCLUSION

5.1 Review Summary

This part explained the result from the analysis of the backpropagation neural network (BPNN). There are four sections which is diagnostic test, comparison of both BPNN model and Enhanced BPNN model using the real dataset, FTSE Bursa Malaysia Kuala Lumpur Composite Index (FBM KLCI) stock market data and last but not least, the convergence test was tested using the different number of epochs.

Based on the first objective, which was to identify the severity of outliers problems in the FBM KLCI dataset, was achieved. This research identifies that outliers exist in the closing price from FBM KLCI stock market. Based on the test of outliers, this research clearly seen that the outliers exist more in lag 40 and lag 50. Moreover, the highest percentage of the outliers got up to 65%, it shows that the enhanced model is really needed since based on the previous study, the BPNN model can only cater the outliers problem up to 50% of outliers only.

In order to evaluate the performance of the ordinary and enhanced model on real FBM KLCI dataset which contained outliers problems, a numerical comparison of the BPNN model and enhanced BPNN model was conducted on the real FBM KLCI dataset. The result shows that the RMSE and GRMSE value got the smallest value by using the enhanced BPNN model compared to ordinary BPNN model with the higher number of input lags and the lower number of hidden nodes. Furthermore, the best configuration in training and testing phase by using the BPNN model is 40-5 and 30-25 respectively. This research achieved the third objective, which was to develop an enhanced BPNN model for FBM KLCI dataset with outliers problems. The best configuration in training and testing phase is 35-5 and 20-10 respectively.

Furthermore, the result shows that smallest value based on the performance evaluation of the enhanced BPNN model using Simulated Data Set I was at 65% of outliers where the value of RMSE and GRMSE were 0.309335 (20-10) and 1.300325 (20-5) respectively. Based on the performance of the enhanced BPNN model checking, the best configuration shows the average number of input lags and the smaller number of hidden nodes. The evaluation by using Simulated Data Set II got the smallest value of RMSE and GRMSE at the 65% of outliers where the value was 0.211321 (30-35) and 1.030024 (20-20) respectively.

The last objective, which was to validate the enhanced model using time series bootstrap technique was achieved. Lastly, in the convergence test, the lag that shows the better convergence compared to other lags is lag 25.

The improved BPNN model is designed to reduce network errors by effectively handling outliers, which in turn boosts prediction accuracy. This makes the study highly valuable for a range of stock market participants. By using this enhanced model, both investors and speculators stand to benefit through more accurate predictions, potentially leading to higher profits. With the model's optimization, investors can make better-informed decisions about stock prices and market trends, increasing their chances of making profitable trades using the enhanced BPNN model with two-layer of 20-20 configurations.

Overall, all the objectives were successfully achieved. This model was expected to assist investors, economists, policymakers, and financial institutions in their forecasting activities with high accuracy.

5.2 Limitation

In this study, focus was exclusively placed on the Malaysian stock market, specifically the FBM KLCI index and only closing prices were used as input variables. Furthermore, during the backpropagation training process, OLS estimators were replaced with LMedS estimators due to their ability to effectively address issues related to outliers. Consequently, the enhanced BPNN model was utilized to forecast the closing prices of the FBM KLCI. Model performance was evaluated using Root Mean Squared Error (RMSE) and Geometric Root Mean Square Error (GRMSE) as error measures.

5.3 Recommendation

Based on the findings of this study, future research will incorporate data from multiple stock markets to analyze model performance and determine the optimal configuration. Additionally, further refinement of the enhanced BPNN model was recommended, either through the integration of alternative metaheuristic algorithms or by modifying the function to enhance performance.

Moreover, the performance of the enhanced BPNN model using existing metaheuristic algorithms such as Particle Swarm Optimization (PSO) as suggested by (Li et al., 2021), will be compared to the DPSG-LMedS BPNN model in the future research work.

REFERENCES

Abdoli, G., Mohsen, M., & Ardalani, M. E. (2020). Comparing the Prediction Accuracy of LSTM and ARIMA Models for Time-Series with Permanent Fluctuation. *SSRN Electronic Journal*, *9*. https://doi.org/10.2139/ssrn.3612487

Agnieszka, D., & Magdalena, L. (2018, September 1). Detection of outliers in the financial time series using ARIMA models. https://doi.org/10.1109/PTZE.2018.8503260

Ahmed Mohammed Gamal, A., Alqadasi, A., Ramli, N., Viswanathan, Kk., & Kuperan Viswanathan, K. (2021). The Impact of COVID-19 on the Malaysian Stock Market: Evidence from an Autoregressive Distributed Lag Bound Testing Approach*. *Journal of Asian Finance*, *8*(7). https://doi.org/10.13106/jafeb.2021.vol8.no7.0001

Akerstrom, E. (2020). *Real-time Outlier Detection using Unbounded Data Streaming and Machine Learning* (Master Thesis). Luleå University of Technology.

Al-Mashhadani, A. F. Shahatha., Hishan, S. S., Awang, H., & Alezabi, K. A. A. (2021). Forecasting Malaysian Stock Price using Artificial Neural Networks (ANN). *Journal of Contemporary Issues in Business and Government*, *27*(1).

Anis, I., & Bahar, A. (2021). Analysis Of the Bursa Malaysia Stock Prices and Forecasting. *Proceedings of Science and Mathematics*, *2*, 107–117.

Ashour, M. A. H., Jamal, A., & Helmi, R. A. A. (2018). Effectiveness of Artificial Neural Networks in Solving Financial Time Series. *International Journal of*

Engineering & Technology, *7*(4.11), 99. https://doi.org/10.14419/ijet.v7i4.11.20783

Askari, Q., Younas, I., & Saeed, M. (2020). Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. *Knowledge-Based Systems*, *195*, 105709. https://doi.org/10.1016/j.knosys.2020.105709

Averitt, A. J., & Natarajan, K. (2018). Going Deep: The Role of Neural Networks for Renal Survival and Beyond. *Kidney International Reports*, *3*(2), 242–243. https://doi.org/10.1016/j.ekir.2017.12.006

Awajan, A. M., Ismail, M. T., & AL Wadi, S. (2018). Improving forecasting accuracy for stock market data using EMD-HW bagging. *PLOS ONE*, *13*(7), e0199582. https://doi.org/10.1371/journal.pone.0199582

Azura Md Ghani, N., Bin Ahmad Kamaruddin, S., Musirin, I., & Hashim, H. (2018). Results of Fitted Neural Network Models on Malaysian Aggregate Dataset. *Bulletin of Electrical Engineering and Informatics*, *7*(2), 272–278. https://doi.org/10.11591/eei.v7i2.1177

Bakar, N. M. A., & Midi, H. (2019). The Applications of Robust Estimation in Fixed Effect Panel Data Model. *In 1st Aceh Global Conference (AGC 2018)*, 341–346.

Basuony, M. A. K., Bouaddi, M., Ali, H., & EmadEldeen, R. (2021). The effect of COVID-19 pandemic on global stock markets: Return, volatility, and bad state probability dynamics. *Journal of Public Affairs*. https://doi.org/10.1002/pa.2761

Beliakov, G., Kelarev, A., & Yearwood, J. (2011). *Robust artificial neural networks* *and outlier detection. Technical report*.

Bergmeir, C., Hyndman, Rob J., & Benítez, José M. (2016). Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation. *International Journal of Forecasting*, *32*(2), 303–312. https://doi.org/10.1016/j.ijforecast.2015.07.002

BERNAMA. (2024). Ringgit terus merosot susulan kurang pemangkin baharu. Retrieved from BH Online website: https://www.bharian.com.my/bisnes/pasaran/2024/01/1200651/ringgit-terus-merosot-susulan-kurang-pemangkin-baharu#google_vignette

Bhardwaj, S., Chandrasekhar, E., Padiyar, P., & Gadre, V. M. (2020). A comparative study of wavelet-based ANN and classical techniques for geophysical time-series forecasting. *Computers & Geosciences*, *138*, 104461. https://doi.org/10.1016/j.cageo.2020.104461

Bhasin, P., & Vaishali. (2017). Back Propogation Algorithm: An Artificial Neural Network Approach. *International Journal of Engineering Research & Technology (IJERT)*, *5*(10). ICCCS - 2017 Conference Proceedings.

Bing, Y., Hao, J. K., & Zhang, S. C. (2012). Stock Market Prediction Using Artificial Neural Networks. *Advanced Engineering Forum*, *6-7*, 1055–1060. https://doi.org/10.4028/www.scientific.net/aef.6-7.1055

Blázquez-García, A., Conde, A., Mori, U., & Lozano, J. A. (2021). A Review on Outlier/Anomaly Detection in Time Series Data. *ACM Computing Surveys*, *54*(3), 1–33. https://doi.org/10.1145/3444690

Boadi, I., & Amegbe, H. (2017). The Link between Quality of Governance and Stock Market Performance: International Level Evidence. *European Journal of Government and Economics*, *6*(1), 78–101. https://doi.org/10.17979/ejge.2017.6.1.4324

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time Series Analysis: Forecasting and Control*. John Wiley & Sons.

Brunel, A., Mazza, D., & Pagani, M. (2019). Backpropagation in the simply typed lambda-calculus with linear negation. *Proceedings of the ACM on Programming Languages*, *4*(POPL), 1–27. https://doi.org/10.1145/3371132

Bursamalaysia, A. (2023). Bursa PLCs IR Series 3: Economic Outlook 2024. Retrieved July 10, 2024, from www.bursamalaysia.com website: https://www.bursamalaysia.com/reference /ir4uprogram/recent_events/bursa-plcs-ir-series-3-economic-outlook-2024

Chan, J. Y.-L., Leow, S. M. H., Bea, K. T., Cheng, W. K., Phoong, S. W., Hong, Z.-W., & Chen, Y.-L. (2022). Mitigating the Multicollinearity Problem and Its Machine Learning Approach: A Review. *Mathematics*, *10*(8), 1283. https://doi.org/10.3390/math10081283

Chen, C.-H., Chen, P.-Y., & Chun-Wei Lin, J. (2022). An Ensemble Classifier for Stock Trend Prediction Using Sentence-Level Chinese News Sentiment and Technical Indicators. *International Journal of Interactive Multimedia and Artificial Intelligence*, *7*(3), 53. https://doi.org/10.9781/ijimai.2022.02.004

Chen, J., Dai, H., Wang, S., & Liu, C. (2024). *Improving Accuracy and Efficiency in Time Series Forecasting with an Optimized Transformer Model*. *32*(1), 1–11.

Chen, Y.-C., Kuo, S.-M., Liu, Y., Wu, Z., & Nasir, M. (2022). Improving Returns on Strategy Decisions through Integration of Neural Networks for the Valuation of Asset Pricing: The Case of Taiwanese Stock. *International Journal Financial Studies*, *10*(4), 99. https://doi.org/10.3390/ijfs10040099

Cheng, J., Huang, K., & Zheng, Z. (2023). Fitting Imbalanced Uncertainties in Multi-output Time Series Forecasting. *ACM Transactions on Knowledge Discovery from Data*, *17*(7), 1–23. https://doi.org/10.1145/3584704

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

Chinaza, E. E. (Ed.). (2024, February 28). *Descriptive Analysis with Python*. Medium; Medium. https://medium.com/@isrealemmanuel1025/descriptive-analysis-with-python-585abe5da6cb

Chinembiri, T. S., Mutanga, O., & Dube, T. (2023). Hierarchical Bayesian geostatistics for C stock prediction in disturbed plantation forest in Zimbabwe. *Ecological Informatics*, *73*, 101934. https://doi.org/10.1016/j.ecoinf.2022.101934

Chuan, T. B., Hameed, Dr. V. A., & Ramachandiran, C. R. (2021). Malaysia stock market prediction using artificial intelligence. *Journal of Applied Technology and Innovation*, *5*(1), 32.

Dahal, K. R., Pokhrel, N. R., Gaire, S., Mahatara, S., Joshi, R. P., Gupta, A., Banjade, H. R., & Joshi, J. (2023). A comparative study on effect of news sentiment on stock price prediction with deep learning architecture. *PLOS ONE*, *18*(4), e0284695. https://doi.org/10.1371/journal.pone.0284695

Daniel, F. (2019). Solve the Outlier in Multiplication Linear Regression Models with Robust's Regression Method Least Median of Squares (LMS) Estimator. *BAREKENG: Jurnal Ilmu Matematika Dan Terapan*, *13*(3), 145–156. https://doi.org/10.30598/barekengvol13iss3pp145-156ar884

Dantas, T. M., & Cyrino Oliveira, F. L. (2018). Improving time series forecasting: An approach combining bootstrap aggregation, clusters and exponential smoothing. *International Journal of Forecasting*, *34*(4), 748–761. https://doi.org/10.1016/j.ijforecast.2018.05.006

Daradkeh, M. K. (2022). A Hybrid Data Analytics Framework with Sentiment Convergence and Multi-Feature Fusion for Stock Trend Prediction. *Electronics*, *11*(2), 250. https://doi.org/10.3390/electronics11020250

de Souza Barbosa, A., da Silva, M. C. B. C., da Silva, L. B., Morioka, S. N., & de Souza, V. F. (2023). Integration of Environmental, Social, and Governance (ESG) criteria: their impacts on corporate sustainability performance. *Humanities and Social Sciences Communications*, *10*(1). https://doi.org/10.1057/s41599-023-01919-0

Debie, E., & Shafi, K. (2017). Implications of the curse of dimensionality for supervised learning classifier systems: theoretical and empirical analyses.

Pattern Analysis and Applications, *22*(2), 519–536. https://doi.org/10.1007/s10044-017-0649-0

Doma´nski, P. D., & Wieclawski, M. (2015). Memory-Based Prediction of District Heating Temperature Using GPGPU. *Progress in Automation, Robotics and Measuring Techniques: Control and Automation*. https://doi.org/10.1007/978-3-319-15796-2

Du, X., Liu, M., & Sun, Y. (2022). Cell Recognition Using BP Neural Network Edge Computing. *Contrast Media & Molecular Imaging*, *2022*(1). https://doi.org/10.1155/2022/7355233

Dwiarso, U., Pujiono, & Moch Arief , S. (2017). Stock Price Prediction Using Back Propagation Neural Network Based on Gradient Descent with Momentum and Adaptive Learning Rate. *The Journal of Internet Banking and Commerce*, *22*(3), 1–16.

Elhoseny, M., Metawa, N., & El-hasnony, I. M. (2022). A new metaheuristic optimization model for financial crisis prediction: Towards sustainable development. *Sustainable Computing: Informatics and Systems*, *35*, 100778. https://doi.org/10.1016/j.suscom.2022.100778

El-Melegy, M., Essai, M., & Ali, A. (2009). *Robust Training of Artificial Feedforward Neural Networks*.

Farahani, M. S., & Hajiagha, S. H. R. (2021). Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. *Soft Computing*, *25*(13). https://doi.org/10.1007/s00500-021-05775-5

Foss, T., Myrtveit, I., & Stensrud, E. (2001). A comparison of LAD and OLS regression for effort prediction of software projects. *In Proc. 12th European Software Control and Metrics Conference (ESCOM 2001)*, 9–15.

Gamal, A. A. M., Al-Qadasi, A. A., Noor, M. A. M., Rambeli, N., & Viswanathan, K. K. (2021). The Impact of COVID-19 on the Malaysian Stock Market: Evidence from an Autoregressive Distributed Lag Bound Testing Approach*. *Journal of Asian Finance, Economics and Business*, *8*(7), 0001–0009.

Gandhmal, D. P., & Kumar, K. (2019). Systematic analysis and review of stock market prediction techniques. *Computer Science Review*, *34*, 100190. https://doi.org/10.1016/j.cosrev.2019.08.001

Ghasemiyeh, R., Moghdani, R., & Sana, S. S. (2017). A Hybrid Artificial Neural Network with Metaheuristic Algorithms for Predicting Stock Price. *Cybernetics and Systems*, *48*(4), 365–392. https://doi.org/10.1080/01969722.2017.1285162

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, *13*(5), 533–549. https://doi.org/10.1016/0305-0548(86)90048-1

Gupta, N., Khosravy, M., Mahela, O. P., & Patel, N. (2020). Plant biology-inspired genetic algorithm: superior efficiency to firefly optimizer. *In Applications of Firefly Algorithm and Its Variants*, 193–219. Springer, Singapore.

Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, *38*(8), 10389–10397. https://doi.org/10.1016/j.eswa.2011.02.068

Hadi, N. S. (2006). Time Series Analysis for Hydrological Features: Applications of

Box-Jenkins Models to Euphrates River. *Journal of Applied Sciences*, *6*(9), 1929–1934. https://doi.org/10.3923/jas.2006.1929.1934

Hafiz, N. A. M., Ishak, N., & Rasedee, A. F. N. (2019). Analysing and Forecasting Share Price Index in Malaysia. *Journal of Computational and Theoretical Nanoscience*, *16*(12), 4930–4936. https://doi.org/10.1166/jctn.2019.8544

Hampel, F. (1974). The Influence Curve and Its Role in Robust Estimation The Influence Curve and Its Role in Robust Estimation. *Journal of the American Statistical Association*, *69*(346), 383–393.

Hasanat, M. W., Hoque, A., Shikha, F. A., Anwar, M., Hamid, Prof. Dr. A. B. A., & Tat, Prof. Dr. H. H. (2020). The Impact of Coronavirus (Covid-19) on E-Business in Malaysia. *Asian Journal of Multidisciplinary Studies*, *3*(1).

Hassan, M., Hossain, S., & Ahmed, H. (2022). Impact of Islamic Finance on Economic Growth. *Journal of Economic Cooperation and Development*, *43*(2), 101–128.

Hayes, A. (2021). *What Is Nonlinearity?* Investopedia. https://www.investopedia.com/terms/n/nonlinearity.asp

Hayes, A. (2022). *Multicollinearity*. Investopedia. https://www.investopedia.com/terms/m/multicollinearity.asp

Ho, S. Y. (2019). The macroeconomic determinants of stock market development in Malaysia: an empirical analysis. *Global Business and Economics Review*, *21*(2), 174. https://doi.org/10.1504/gber.2019.10018126

Holger Kantz, & Schreiber, T. (2015). *Nonlinear time series analysis*. Beijing: Shi Jie Tu Shu Chu Ban Gong Si Beijing Gong Si.

Hosseinioun, N. (2016). Forecasting Outlier Occurrence in Stock Market Time Series Based on Wavelet Transform and Adaptive ELM Algorithm. *Journal of Mathematical Finance*, *06*(01), 127–133. https://doi.org/10.4236/jmf.2016.61013

Hsieh, L.-F., Hsieh, S.-C., & Tai, P.-H. (2011). Enhanced stock price variation prediction via DOE and BPNN-based optimization. *Expert Systems with Applications*, *38*(11). https://doi.org/10.1016/j.eswa.2011.04.229

Huang, G.-B., Chen, L., & Siew, C.-K. (2006). Universal Approximation using Incremental Constructive Feedforward Networks with Random Hidden Nodes. *IEEE Transactions on Neural Networks*, *17*(4), 879–892. https://doi.org/10.1109/tnn.2006.875977

Ican, Ö., & Çelik, T. B. (2017). Stock Market Prediction Performance of Neural Networks: A Literature Review. *International Journal of Economics and Finance*, *9*(11), 100. https://doi.org/10.5539/ijef.v9n11p100

Jang, H., Topal, E., & Kawamura, Y. (2015). Unplanned dilution & ore loss prediction in longholestoping mines via multiple regression & artificial neural network analyses. *Journal of Southern African Institute of Mining Metallurgy*, *115*(5), 449–456.

Jin, Z., Yang, Y., & Liu, Y. (2020). Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications*. https://doi.org/10.1007/s00521-019-04504-2

K. Ritwik Reddy, B. Tarun Kumar, V. Rohit Ganesh, Polisetty Swetha, & Prakash Kumar Sarangi. (2022). *Stock Market Prediction Using Recurrent Neural*

Network. https://doi.org/10.1109/ccet56606.2022.10080154

Kalaiselvi, K., Velusamy, K., & Gomathi, C. (2018). Financial prediction using back propagation neural networks with opposition based learning. *Journal of Physics: Conference Series*, *1142*, 012008. https://doi.org/10.1088/1742-6596/1142/1/012008

Kamaruddin, S. A., Ghani, N. A. M., Rahim, H. A., & Musirin, I. (2019). Killer Whale – Backpropagation (KW-BP) Algorithm for Accuracy Improvement of Neural Network Forecasting Models on Energy-Efficient Data. *IAES International Journal of Artificial Intelligence*, *8*(3), 270–270. https://doi.org/10.11591/ijai.v8.i3.pp270-277

Kamaruddin, S., Azura, N., Ghani, M., & Ramli, N. (2016). The Improved BPNN-NAR and BPNN-NARMA Models on Malaysian Aggregate Cost Indices with Outlying Data. *Jurnal Teknologi*, *78*, 2180–3722.

Karim, Z. A., Yi, L. Q., Karim, B. A., & Zakaria, S. F. D. (2024). The Impact of the COVID-19 Pandemic on the Malaysian Stock Market. *Contemporary Issues in Finance, Investment and Banking in Malaysia*, 65–69. https://doi.org/10.1007/978-981-99-5447-6_5

Kenton, W. (2020). *Kuala Lumpur Stock Exchange (KLS)*. Investopedia; Investopedia. https://www.investopedia.com/terms/k/klse.asp

Khanthavit, A. (2021). Measuring COVID-19 Effects on World and National Stock Market Returns*. *Journal of Asian Finance, Economics and Business*, *8*(2), 0001-0013.

Khairudin, N., Sima, N., & Shariff, M. (2023). The Impact of COVID-19 Outbreaks on the Volatility of the Stock Market in Malaysia. *Malaysian Journal of Computing*, *8*(1), 1287–1300. https://doi.org/10.24191/mjoc.v8i1.19172

Kolassa, S. (2016). Evaluating predictive count data distributions in retail sales forecasting. *International Journal of Forecasting*, *32*(3), 788–803. https://doi.org/10.1016/j.ijforecast.2015.12.004

Kumar, G., Jain, S., & Singh, U. P. (2020). Stock Market Forecasting Using Computational Intelligence: A Survey. *Archives of Computational Methods in Engineering*, *28*(3). https://doi.org/10.1007/s11831-020-09413-5

Kumbure, M. M., Lohrmann, C., Luukka, P., & Porras, J. (2022). Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications*, *197*, 116659. https://doi.org/10.1016/j.eswa.2022.116659

Lai, M.-M., & Lau, S.-H. (2004). Herd Behavior and Market Stress: The Case of Malaysia. *Academy of Accounting and Financial Studies Journal*, *8*(3), 85–102.

Lean, H. H., & Smyth, R. (2010). On the dynamics of aggregate output, electricity consumption and exports in Malaysia: Evidence from multivariate Granger causality tests. *Applied Energy*, *87*(6), 1963–1971. https://doi.org/10.1016/j.apenergy.2009.11.017

Li, G., Tan, Z., Xu, W., Xu, F., Wang, L., Chen, J., & Wu, K. (2021). A particle swarm optimization improved BP neural network intelligent model for electrocardiogram classification. *BMC Medical Informatics and Decision Making*, *21*(S2). https://doi.org/10.1186/s12911-021-01453-6

Liang, Y. (2021). Stock Market Forecasting based on Artificial Intelligence Technology. In *Electronic Theses, Projects, and Dissertations*. Retrieved from https://scholarworks.lib.csusb.edu/etd/1324

Limas, M. C., Ordieres Meré, J. B., de Pisón Ascacibar, F. J. M., & González, E. P. V. (2004). Outlier Detection and Data Cleaning in Multivariate Non-Normal Samples: The PAELLA Algorithm. *Data Mining and Knowledge Discovery*, *9*(2), 171–187. https://doi.org/10.1023/b:dami.0000031630.50685.7c

Linden, A., & Adams, J. L. (2010). Applying a propensity score-based weighting model to interrupted time series data: improving causal inference in programme evaluation. *Journal of Evaluation in Clinical Practice*, *17*(6), 1231–1238. https://doi.org/10.1111/j.1365-2753.2010.01504.x

Liu, Y., Cao, C., Huang, W., & Hao, S. (2021). A Deep Neural Network Based Model For Stock Market Prediction. *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*. https://doi.org/10.1109/icbaie52039.2021.9390010

Lv, J., Wang, C., Gao, W., & Zhao, Q. (2021). An Economic Forecasting Method Based on the LightGBM-Optimized LSTM and Time-Series Model. *Computational Intelligence and Neuroscience*, *2021*, 1–10. https://doi.org/10.1155/2021/8128879

Ma, Y., Zhang, Z., & Ihler, A. (2020). Multi-Lane Short-Term Traffic Forecasting with Convolutional LSTM Network. *IEEE Access*, *8*, 34629–34643. https://doi.org/10.1109/access.2020.2974575

Ma, Z., Wu, G., Suganthan, P. N., Song, A., & Luo, Q. (2022). Performance assessment and exhaustive listing of 500+ nature-inspired metaheuristic algorithms. *Swarm and Evolutionary Computation*, 101248. https://doi.org/10.1016/j.swevo.2023.101248

Mahsereci, M., Balles, L., Lassner, C., & Hennig, P. (2017). Early stopping without a validation set. arXiv preprint arXiv:1703.09580. (n.d.).

Mamoudan, M. M., Ostadi, A., Pourkhodabakhsh, N., Fathollahi-Fard, A. M., & Soleimani, F. (2023). Hybrid neural network-based metaheuristics for prediction of financial markets: a case study on global gold market. *Journal of Computational Design and Engineering*, *10*(3), 1110–1125. https://doi.org/10.1093/jcde/qwad039

Mansor, R., Othman, M., & Kasim, M. M. (2017). Fuzzy Reasoning Based Time Series Forecasting Using Weighted Subsethood Model. *Advanced Science Letters*, *23*(9), 9094–9097. https://doi.org/10.1166/asl.2017.10030

Massart, D. L., Kaufman, L., Rousseeuw, P. J., & Leroy, A. (1986). Least median of squares: a robust method for outlier and model error detection in regression and calibration. *Analytica Chimica Acta*, *187*, 171–179. https://doi.org/10.1016/s0003-2670(00)82910-4

Mehmood, F., Ahmad, S., & Whangbo, T. K. (2023). An Efficient Optimization

Technique for Training Deep Neural Networks. *Mathematics*, *11*(6), 1360. https://doi.org/10.3390/math11061360

Mehrnaz Faraz, & Hamid Khaloozadeh. (2020). Multi-Step-Ahead Stock Market Prediction Based on Least Squares Generative Adversarial Network. *2020 28th Iranian Conference on Electrical Engineering (ICEE)*. IEEE. https://doi.org/10.1109/icee50131.2020.9260858

Mishra, P., Pandey, C., Singh, U., Gupta, A., Sahu, C., & Keshri, A. (2019). Descriptive Statistics and Normality Tests for Statistical Data. *Annals of Cardiac Anaesthesia*, *22*(1), 67. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6350423/

Mohammad Hussein Amiri, Nastaran Mehrabi Hashjin, Mohsen Montazeri, Seyedali Mirjalili, & Nima Khodadadi. (2024). Hippopotamus optimization algorithm: a novel nature-inspired optimization algorithm. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-54910-3

Moraitis, N., Tsipi, L., Vouyioukas, D., Gkioni, A., & Louvros, S. (2021). Performance evaluation of machine learning methods for path loss prediction in rural environment at 3.7 GHz. *Wireless Networks*, *27*(6), 4169–4188. https://doi.org/10.1007/s11276-021-02682-3

Mutinda, J. K., & Geletu, A. (2025). Stock Market Index Prediction Using CEEMDAN-LSTM-BPNN-Decomposition Ensemble Model. *Journal of Applied Mathematics*, *2025*(1). https://doi.org/10.1155/jama/7706431

Naidoo, V., & Du, S. (2022). A Deep Learning Method for the Detection and Compensation of Outlier Events in Stock Data. *Electronics*, *11*(21), 3465.

Nayak, S., Misra, B., & Behera, H. (2014). Impact of Data Normalization on Stock Index Forecasting. *International Journal of Computer Information Systems and Industrial Management Applications*, *6*, 257–269.

Oh, J. (2022). Development of a stock trading system based on a neural network using highly volatile stock price patterns. *PeerJ Computer Science*, *8*, e915. https://doi.org/10.7717/peerj-cs.915

Oracle. (2014). *What Is Big Data? | Oracle*. Oracle.com. https://www.oracle.com/big-data/what-is-big-data/#:~:text=The%20definition%20of%20big%20data

Pellegrino, E., Brunet, T., Pissier, C., Camilla, C., Abbou, N., Beaufils, N., … Ouafik, L. (2022). Deep Learning Architecture Optimization with Metaheuristic Algorithms for Predicting BRCA1/BRCA2 Pathogenicity NGS Analysis. *BioMedInformatics*, *2*(2), 244–267. https://doi.org/10.3390/biomedinformatics2020016

Pernía-Espinoza, A. V., Ordieres-Meré, J. B., Martínez-de-Pisón, F. J., & González-Marcos, A. (2005). TAO-robust backpropagation learning algorithm. *Neural Networks*, *18*(2), 191–204. https://doi.org/10.1016/j.neunet.2004.11.007

Persio, L., & Honchar, O. (2018). *Artificial Neural Networks architectures for stock price prediction: comparisons and applications*.

Pillay, B. J., & Ezugwu, A. E. (2019). *Metaheuristics optimized feedforward neural networks for efficient stock price prediction*. arXiv preprint arXiv:1906.10121.

Powers, J. (2022). *Curse of Dimensionality*. Built In. https://builtin.com/data-science/curse-dimensionality

Priddy, K. L. (2007). *Artificial Neural Networks : An Introduction.* New Delhi: Prentice-Hall India.

Primadusi, U., Cahyadi, A. I., & Wahyunggoro, O. (2016). The comparison of RBF NN and BPNN for SOC estimation of LiFePO4 battery. *AIP Conference Proceedings*, *1755*(1). https://doi.org/10.1063/1.4958528

Ren, L., Ren, P., & Glasure, Y. (2018). On the different forms of returns from moving average buy-sell trading rule in the stock market. *Benchmarking: An International Journal*, *25*(1), 253–258. https://doi.org/10.1108/bij-06-2016-0099

Rizvi, D. R., & Khalid, M. (2024). Performance Analysis of Stocks using Deep Learning Models. *Procedia Computer Science*, *233*, 753–762. https://doi.org/10.1016/j.procs.2024.03.264

Rousseeuw, P. J. (1984). Least Median of Squares Regression. *Journal of the American Statistical Association*, *79*(388), 871–880. https://doi.org/10.1080/01621459.1984.10477105

Rousseeuw, P. J., & Leroy, A. M. (2003). *Robust regression and outlier detection*. Hoboken, New York: Wiley-Interscience.

Rusiecki, A. (2012). Robust Learning Algorithm Based on Iterative Least Median of Squares. *Neural Processing Letters*, *36*(2), 145–160. https://doi.org/10.1007/s11063-012-9227-z

Rusiecki, A. (2013). Robust learning algorithm based on LTA estimator. *Neurocomputing*, *120*, 624–632. https://doi.org/10.1016/j.neucom.2013.04.008

Rusiecki, A., Kordos, M., Kami´nskikami´nski, T., & Gre´n, K. G. (n.d.). *Training Neural Networks on Noisy Data*. www.springerlink.com

Rusiecki, A., Kordos, M., Kamiński, T., & Greń, K. (2014). Training neural networks on noisy data. *In International Conference on Artificial Intelligence and Soft Computing*, 131–142. Springer, Cham.

Samantaray, S., & Sahoo, A. (2020). Prediction of runoff using BPNN, FFBPNN, CFBPNN algorithm in arid watershed: A case study. *International Journal of Knowledge-Based and Intelligent Engineering Systems*, *24*(3), 243–251.

Selvamuthu, D., Kumar, V., & Mishra, A. (2019). Indian stock market prediction using artificial neural networks on tick data. *Financial Innovation*, *5*(1). https://doi.org/10.1186/s40854-019-0131-7

Seong, L. K., & Salleh, R. M. (2022). A Comparative Study on Criminal Cases through Economic Indicators in Malaysia using Regression Modelling. *Enhanced Knowledge in Sciences and Technology*, *2*(1), 132–142.

Sharma, A. K., Kiran, M., Pauline Sherly Jeba, P., Maheshwari, P., & Divakar, V. (2021). Demand Forecasting Using Coupling Of Machine Learning And Time Series Models For The Automotive After Market Sector. *2021 5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT)*. https://doi.org/10.1109/iceeccot52851.2021.9708010

Shehadeh, A. A., Alwadi, S. M., & Almaharmeh, M. I. (2022). Detecting and

Analysing Possible Outliers in Global Stock Market Returns. *Cogent Economics & Finance*, *10*(1). https://doi.org/10.1080/23322039.2022.2066762

Shen, J., & Shafiq, M. O. (2020). Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of Big Data*, *7*(1). https://doi.org/10.1186/s40537-020-00333-6

Simmons, J. (Ed.). (2018). *Are Golden and Death Crosses a Good Bet?* Factset.com. https://insight.factset.com/are-golden-and-death-crosses-a-good-bet

Snigdha, A. (2021, June 12). *Guide to Prevent Overfitting in Neural Networks*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/complete-guide-to-prevent-overfitting-in-neural-networks-part-1/

Sonkavde, G., Dharrao, D. S., Bongale, A. M., Deokate, S. T., Doreswamy, D., & Bhat, S. K. (2023). Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications. *International Journal of Financial Studies*, *11*(3), 94. https://doi.org/10.3390/ijfs11030094

Surakhi, O., Zaidan, M. A., Fung, P. L., Hossein Motlagh, N., Serhan, S., AlKhanafseh, M., … Hussein, T. (2021). Time-Lag Selection for Time-Series Forecasting Using Neural Network and Heuristic Algorithm. *Electronics*, *10*(20), 2518. https://doi.org/10.3390/electronics10202518

Taharem, A., & Fitriyah, L. (2023). *Bursa Malaysia Reports RM252.4 Million Profit After Tax, Zakat And Minority Interest For Financial Year Ended 31 December 2023*.

Tan, A., Yiing, S., & Thim, C. K. (n.d.). *Prediction of Bursa Malaysia Stock Index using Autoregressive Integrated Moving Average and Artificial Neural Network*.

Torres, J. A. O., Santiago, A. M., Izaguirre, J. M. V., García, M. A., & Hernández, A. D. (2022). Hypertension Diagnosis with Backpropagation Neural Networks for Sustainability in Public Health. *Public Health. Sensors*, *22*, 5272. https://doi.org/10.3390/s22145272

Tsay, R. S. (1988). Outliers, level shifts, and variance changes in time series. *Journal of Forecasting*, *7*(1), 1–20. https://doi.org/10.1002/for.3980070102

Utomo, D., Pujiono, & Soeleman, M. A. (2017). Stock Price Prediction Using Back Propagation Neural Network Based on Gradient Descent with Momentum and Adaptive Learning Rate. *Journal of Internet Banking and Commerce*, *22*(3), 1–16.

Uzair, M., & Jamil, N. (2020). Effects of Hidden Layers on the Efficiency of Neural networks. *2020 IEEE 23rd International Multitopic Conference (INMIC)*, 1–6. IEEE. https://doi.org/10.1109/inmic50486.2020.9318195

Vargas, V. W. de , Aranda, J. A. S., Costa, R. dos S., Pereira, P. R. da S., & Barbosa, J. L. V. (2022). Imbalanced data preprocessing techniques for machine learning: a systematic mapping study. *Knowledge and Information Systems*, *65*(1), 31–57. https://doi.org/10.1007/s10115-022-01772-8

Vijh, M., Chandola, D., Tikkiwal, V. A., & Kumar, A. (2020). Stock Closing Price Prediction using Machine Learning Techniques. *Procedia Computer Science*, *167*, 599–606. https://doi.org/10.1016/j.procs.2020.03.326

Vishwakarma, G. K., Paul, C., & Elsawah, A. M. (2020). An algorithm for outlier detection in a time series model using backpropagation neural network. *Journal of King Saud University-Science*, *32*(8), 3328–3336.

Vishwakarma, G. K., Paul, C., & Elsawah, A. M. (2020). An algorithm for outlier detection in a time series model using backpropagation neural network. *Journal of King Saud University - Science*, *32*(8), 3328–3336. https://doi.org/10.1016/j.jksus.2020.09.018

Wang, G.-G., & Tan, Y. (2017). Improving Metaheuristic Algorithms With Information Feedback Models. *IEEE Transactions on Cybernetics*, *49*(2), 542–555. https://doi.org/10.1109/tcyb.2017.2780274

Wang, H., & Suter, D. (2003). Using symmetry in robust model fitting. *Pattern Recognition Letters*, *24*(16), 2953–2966. https://doi.org/10.1016/s0167-8655(03)00156-9

Wang, X., Kang, Y., Hyndman, R. J., & Li, F. (2023). Distributed ARIMA models for ultra-long time series. *International Journal of Forecasting*, *39*(3), 1163–1184. https://doi.org/10.1016/j.ijforecast.2022.05.001

Wei, H.-L., Balikhin, M. A., & Walker, S. N. (2015). A new ridge basis function neural network for data-driven modeling and prediction. *In 2015 10th International Conference on Computer Science & Education (ICCSE)*, 125–130. IEEE. https://doi.org/10.1109/iccse.2015.7250229

Yang, S., Hu, X., & Wang, Y. D. (2022). An Efficient and Explainable Ensemble Learning Model for Asphalt Pavement Condition Prediction Based on LTPP

Dataset. *IEEE Transactions on Intelligent Transportation Systems*, *23*(11), 22084–22093. https://doi.org/10.1109/TITS.2022.3164596

Yetis, Y., Kaplan, H., & Jamshidi, M. (2014). *Stock Market Prediction by Using Artificial Neural Network*. Waikoloa, HI, USA: IEEE Xplore 2014 World Automation Congress (WAC).

Yiing, A. T. S., & Thim, C. K. (2015). Prediction of Bursa Malaysia Stock Index using Autoregressive Integrated Moving Average and Artificial Neural Network. *In Malaysia Statistics Conference (MyStats 2015)*, 95.

Zhang, J., & Qu, S. (2021). Optimization of Backpropagation Neural Network under the Adaptive Genetic Algorithm. *Complexity*, *2021*, 1–9. https://doi.org/10.1155/2021/1718234

Zhang, T., Wei, D., Liu, Z., & Wu, X. (2021). Lottery preference and stock market participation: evidence from China. *China Finance Review International*. https://doi.org/10.1108/cfri-01-2021-0008

Zhang, Z., Ahmed, K. A., Hasan, M. R., Gedeon, T., & Hossain, M. Z. (2024). A Deep Learning Approach to Diabetes Diagnosis. *ArXiv Preprint ArXiv:2403.07483.*

Zhao, F., Gao, Y., Li, X., An, Z., Ge, S., & Zhang, C. (2021). A similarity measurement for time series and its application to the stock market. *Expert Systems with Applications*, *182*, 115217. https://doi.org/10.1016/j.eswa.2021.115217

Zhao, J., Nguyen, H., Nguyen-Thoi, T., Asteris, P. G., & Zhou, J. (2021). Improved Levenberg–Marquardt backpropagation neural network by particle swarm and whale optimization algorithms to predict the deflection of RC beams. *Engineering with Computers*. https://doi.org/10.1007/s00366-020-01267-6

Zhao, X., Liu, Y., & Zhao, Q. (2024). Generalized Loss-Based CNN-BiLSTM for Stock Market Prediction. *International Journal of Financial Studies*, *12*(3), 61. https://doi.org/10.3390/ijfs12030061

Zulkifli, K., Safian, S. S., Radzil, R. H. M., & Shaharuddin, N. (2024). The Impact of Stock Market Development on Economic Growth a Case of Malaysia. *Information Management and Business Review*, *16*(1), 86–104.

Appendix A

Python Code for Enhanced Backpropagation Neural Network (BPNN) Model

```
import scipy.io as sio
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers.core import Dense
import keras
import tensorflow as tf
from matplotlib import pyplot as plt
import pandas as pd

# parameters
tf.compat.v1.disable_eager_execution()
useScaling = 1
useNormalization = 1
learningRate = 1e-3
weightDecay = 1e-3 / 200
maxEpochs = 1000;
miniBatchSize = 2048
lag = 15
# hidden layer sizes
h1 = 10
h2 = 10
h3 = 1


data= pd.read_csv("C:\\Users\\HP\\Downloads\\Analysis using Python\\Coding\\Used
data\\Simulated1.csv")
```

```
uall=np.float32(data.to_numpy())

p,q = uall.shape

uall = np.float32(uall)


# normalize so that average mean and zero-variance
if useNormalization:
    normalizer = StandardScaler()
    uall_normParam = normalizer.fit(uall)
    uall_norm = uall_normParam.transform(uall)

# scale the data so that it becomes between 0 and 1
if useScaling:
    mm_scaler = MinMaxScaler(feature_range=(0, 1), copy=True)
    uall_scaled = mm_scaler.fit_transform(uall_norm)



trnsize = round(p/2);
tstsize = round(p/2);
lagx = 0

# psi_trn
psi_trn = uall_scaled[lag:trnsize-1]
lag2 = lag-1
for it in range(lag2):
    takenlagstart = lag2-it
    takenlagend = trnsize-(it+2)
    ut_trn = uall_scaled[takenlagstart:takenlagend]
    psi_trn = np.concatenate((ut_trn, psi_trn), 1)
```

```
# psi_tst
psi_tst = uall_scaled[(tstsize+lag-1):(p-1)]

lag2 = lag-1
for it in range(lag2):
    takenlagstart = tstsize+lag2-it-1
    takenlagend = p-it-2
    ut_tst = uall_scaled[takenlagstart:takenlagend]
    psi_tst = np.concatenate((ut_tst, psi_tst), 1)



yt_trn = uall_scaled[(lag+1):round(p/2)]
yt_tst = uall_scaled[(round(p/2)+lag):p]


###################################################################################
##
# define and create the MLP network
model = Sequential()
model.add(Dense(h1, input_dim=len(psi_trn[0]), activation="relu"))
model.add(Dense(h2, activation="relu"))
model.add(Dense(h3, activation="linear"))

def my_loss_fn(y_true, y_pred):

    squared_difference = tf.square(y_true - y_pred)*x (1-exp((0.35*log(10)(-
3.3)(0.06))+(0.93*1.2)+1.26)/(4000+(2.23*(-3.3))))
    return tf.reduce_mean(squared_difference, axis=-1)  # Note the `axis=-1`
```

```
#Enhanced = ElMed x (1-exp((0.35*log(10)(-3.3)(0.06))+(0.93*1.2)+1.26)/(4000+(2.23*(-
3.3))))

#RMSE
def my_loss_fn2(y_true, y_pred):
   squared_difference = tf.square((((y_pred - y_true) ** 2)))
   return tf.reduce_mean(squared_difference, axis=-1)  # Note the `axis=-1`

#MSPE
#def my_loss_fn3(y_true, y_pred):
#    squared_difference = mean_squared_error(y_true['actual'], predictions)
#    return tf.reduce_mean(squared_difference, axis=-1)  # Note the `axis=-1`

#GRMSE
def my_loss_fn4(y_true, y_pred):
   squared_difference = tf.square(np.prod((y_true / y_pred)) - 1)
   return tf.reduce_mean(squared_difference, axis=-1)  # Note the `axis=-1`

# train the model
opt = tf.keras.optimizers.legacy.Adam(lr = learningRate, decay = weightDecay)
model.compile(loss=my_loss_fn2, optimizer=opt)

# train the model
print("[INFO] training model...")
ccc=model.fit(psi_trn, yt_trn, validation_data=(psi_tst, yt_tst), epochs=maxEpochs,
batch_size=miniBatchSize)

# make predictions on the training data
print("Predicting Training Set...")
preds_trn = model.predict(psi_trn)
yhat_trn = preds_trn.flatten()
resid_trn = yt_trn.flatten() - yhat_trn
percentDiff_trn = (resid_trn / yt_trn) * 100
```

```
absPercentDiff_trn = np.abs(percentDiff_trn)
mean_trn = np.mean(absPercentDiff_trn)
std_trn = np.std(absPercentDiff_trn)
plt.figure()
jj,kk = yt_trn.shape
plt.plot(np.arange(1,jj+1), yt_trn)
plt.plot (np.arange(1,jj+1), yhat_trn, ':')
plt.xlabel("Cases (dimensionless)")
plt.ylabel("Angular Velocity (w)")
plt.title("One Step Ahead Prediction (Training Set)")
plt.show()

# make predictions on the testing data
print("Predicting Testing Set...")
preds_tst = model.predict(psi_tst)
yhat_tst = preds_tst.flatten()
resid_tst = yt_tst.flatten() - yhat_tst
percentDiff_tst = (resid_tst / yt_tst) * 100
absPercentDiff_tst = np.abs(percentDiff_tst)
mean_tst = np.mean(absPercentDiff_tst)
std_tst = np.std(absPercentDiff_tst)
plt.figure()
jj,kk = yt_tst.shape
plt.plot(np.arange(1,jj+1), yt_tst)
plt.plot (np.arange(1,jj+1), yhat_tst, ':')
plt.xlabel("Cases (dimensionless)")
plt.ylabel("Angular Velocity (w)")
plt.title("One Step Ahead Prediction (Testing Set)")
plt.legend(['Actual', 'Forecasted']);
plt.show()

# plot the residuals
plt.figure()
```

```
plt.subplot(211)
jj,kk = psi_trn.shape
plt.plot(np.arange(1,jj+1), resid_trn)
plt.xlabel("Cases (dimensionless)")
plt.ylabel("Angular Velocity Difference (w)")
plt.title("Residuals Plot (Training & Testing Set)")
plt.subplot(212)
jj,kk = psi_tst.shape
plt.plot(np.arange(1,jj+1), resid_tst)
plt.xlabel("Cases (dimensionless)")
plt.ylabel("Angular Velocity Difference (w)")
# plt.title("Residuals Plot (Testing Set)")
plt.show()

# histogram
plt.figure()
num_bins = 15
plt.subplot(211)
n, bins, patches = plt.hist(resid_trn, num_bins, facecolor='blue', alpha=0.5)
plt.title("Residuals Histogram (Training & Testing Set)")
plt.xlabel("Bins")
plt.ylabel("Frequency")
plt.subplot(212)
n, bins, patches = plt.hist(resid_tst, num_bins, facecolor='blue', alpha=0.5)
plt.xlabel("Bins")
plt.ylabel("Frequency")
plt.show()

# autocorrelation
plt.figure()
plt.subplot(211)
plt.acorr(resid_trn)
plt.title("Autocorrelation (Training & Testing Set)")
```

```
plt.xlabel("Lags")
plt.ylabel("ACF")
plt.subplot(212)
plt.acorr(resid_tst)
plt.xlabel("Lags")
plt.ylabel("ACF")
plt.show()

# crosscorrelation
plt.figure()
plt.subplot(211)
plt.xcorr(resid_trn, ut_trn.flatten())
plt.title("Crosscorrelation between Input 1 & Residuals (Training & Testing Set)")
plt.xlabel("Lags")
plt.ylabel("CCF")
plt.subplot(212)
plt.xcorr(resid_tst, ut_tst.flatten())
plt.xlabel("Lags")
plt.ylabel("CCF")
plt.show()

plt.figure()
plt.subplot(211)
plt.xcorr(resid_trn, yt_trn.flatten())
plt.title("Crosscorrelation between Input 2 & Residuals (Training & Testing Set)")
plt.xlabel("Lags")
plt.ylabel("CCF")
plt.subplot(212)
plt.xcorr(resid_tst, yt_tst.flatten())
plt.xlabel("Lags")
plt.ylabel("CCF")
plt.show()
```

```
def rmse(predictions, targets):
    """Calculate root mean squared error between two time series"""
    return np.sqrt(((predictions - targets) ** 2).mean())

rmse_train = rmse(resid_trn, yt_trn)
rmse_test = rmse(resid_tst, yt_tst)

print(rmse_train)
print(rmse_test)
```

Appendix B

Performance of the BPNN, BPNN with LMedS and Enhanced BPNN Model on Real Data, FBM KLCI stock market

		RMSE						GRMSE					
Input Lags	Hidden Nodes	BPNN		BPNN with LMedS		Enhanced		Ordinary		BPNN with LMedS		Enhanced	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing
5	5	0.662752	0.472155	0.664186	0.482093	0.662216	0.477726	1.640006	1.464304	1.638502	1.480626	1.638965	1.472271
	10	0.670994	0.480233	0.663848	0.475844	0.658419	0.479372	1.653010	1.476008	1.648521	1.472004	1.632699	1.474569
	15	0.671108	0.473039	0.663664	0.475984	0.677336	0.517913	1.652941	1.465299	1.640292	1.468027	1.665490	1.534244
	20	0.667396	0.476932	0.661143	0.475123	0.671097	0.475716	1.646816	1.470987	1.646790	1.469485	1.652891	1.469242
	25	0.670143	0.475070	0.667345	0.475632	0.670035	0.472453	1.651446	1.468344	1.645492	1.481313	1.651192	1.464443
	30	0.671609	0.474691	0.665342	0.476669	0.671516	0.475034	1.653796	1.467775	1.641000	1.462110	1.653601	1.468250
	35	0.671009	0.475874	0.665536	0.475745	0.670497	0.473805	1.652778	1.469506	1.641873	1.468805	1.651944	1.466456
	40	0.671348	0.475223	0.661543	0.471637	0.670574	0.476960	1.653305	1.468519	1.637351	1.468600	1.652046	1.471077
	45	0.670889	0.474097	0.663345	0.471425	0.671349	0.474642	1.652560	1.466871	1.651169	1.491456	1.653328	1.467687

		RMSE						GRMSE					
Input Lags	**Hidden Nodes**	**BPNN**		**BPNN with LMedS**		**Enhanced**		**Ordinary**		**BPNN with LMedS**		**Enhanced**	
		Training	**Testing**	**Training**	**Testing**	**Training**	**Testing**	**Training**	**Testing**	**Training**	**Testing**	**Training**	**Testing**
10	5	0.667005	0.478524	0.662335	0.468906	0.668429	0.476610	1.647811	1.473773	1.652466	1.479260	1.650097	1.470866
	10	0.667969	0.477312	0.663426	0.475823	0.666570	0.509837	1.649610	1.472149	1.648699	1.468774	1.652978	1.523672
	15	0.669401	0.478072	0.661645	0.481579	0.673265	0.481458	1.651682	1.473061	1.648665	1.480045	1.658330	1.478336
	20	0.666761	0.477817	0.667432	0.473670	0.669612	0.477257	1.647317	1.472630	1.647983	1.470782	1.652175	1.471836
	25	0.669853	0.476157	0.668444	0.482093	0.668177	0.471677	1.652391	1.470230	1.647791	1.470985	1.649575	1.463619
	30	0.669882	0.476003	0.663848	0.475844	0.669156	0.473831	1.652463	1.469996	1.645210	1.464032	1.651238	1.466862
	35	0.666471	0.468706	0.663664	0.475984	0.669845	0.477883	1.646972	1.459436	1.642849	1.475643	1.652331	1.472687
	40	0.671332	0.482237	0.667540	0.472878	0.668127	0.476790	1.654955	1.479337	1.645433	1.475424	1.649517	1.471111
	45	0.668847	0.473988	0.662601	0.473504	0.669553	0.476530	1.650723	1.467015	1.643901	1.478859	1.651922	1.470739
15	5	0.641549	0.499504	0.662752	0.472155	0.669429	0.478683	1.614443	1.507490	1.650097	1.470866	1.653816	1.474679
	10	0.662468	0.472787	0.670994	0.480233	0.671096	0.484405	1.643086	1.466390	1.649447	1.473357	1.656197	1.482945
	15	0.664454	0.471668	0.671108	0.473039	0.665530	0.471523	1.645210	1.464032	1.650217	1.474492	1.646923	1.463798
	20	0.668260	0.476251	0.668177	**0.461132**	0.669279	0.483240	1.651770	1.470983	1.644627	1.469324	1.653150	1.481093

		RMSE						GRMSE					
Input Lags	Hidden Nodes	BPNN		BPNN with LMedS		Enhanced		Ordinary		BPNN with LMedS		Enhanced	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing
	25	0.666444	0.473154	0.657189	0.469727	0.666460	0.473227	1.648477	1.466170	1.649850	1.476133	1.648367	1.466219
	30	0.665955	0.472750	0.656631	0.474076	0.667499	0.476321	1.647626	1.465644	1.638502	1.480626	1.650193	1.470810
	35	0.666943	0.474061	0.658087	0.478769	0.668320	0.479107	1.649110	1.467404	1.648521	1.472004	1.651620	1.474940
	40	0.667285	0.477046	0.659985	0.478587	0.667430	0.473313	1.649607	1.471770	1.640292	1.468027	1.650009	1.466398
	45	0.666583	0.473032	0.659113	0.481130	0.666095	0.471860	1.648515	1.465944	1.647819	1.476954	1.647661	1.464145
20	5	0.656130	0.492681	0.631344	0.467543	0.615003	0.522368	1.636986	1.496921	1.651770	1.470983	1.589611	1.546720
	10	0.668899	0.505632	0.653957	0.478879	0.581536	**0.454544**	1.656997	1.516662	1.648477	1.466170	1.528544	**1.441731**
	15	0.666864	0.477336	0.656075	0.479705	0.665311	0.473031	1.651281	1.472824	1.647626	1.465644	1.648632	1.466451
	20	0.665885	0.477754	0.654884	0.477493	0.666042	0.483818	1.649447	1.473357	1.652941	1.465299	1.651095	1.483060
	25	0.666289	0.478569	0.651134	0.485324	0.663865	0.472938	1.650217	1.474492	1.646816	1.470987	1.646020	1.466116
	30	0.662975	0.475147	0.654302	0.463543	0.665854	0.477046	1.644627	1.469324	1.651446	1.468344	1.649316	1.472276
	35	0.666056	0.479600	0.653422	0.475553	0.666730	0.480681	1.649850	1.476133	1.653796	1.467775	1.650935	1.477750
	40	0.665715	0.475846	0.655523	0.479434	0.663723	0.472320	1.649199	1.470487	1.652778	1.469506	1.646575	1.465981

		RMSE						GRMSE					
Input Lags	Hidden Nodes	BPNN		BPNN with LMedS		Enhanced		Ordinary		BPNN with LMedS		Enhanced	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing
	45	0.666548	0.477486	0.650494	0.483523	0.663953	0.475855	1.650505	1.472822	1.647791	1.470985	1.646110	1.470419
25	5	0.662465	0.468906	0.650469	0.499259	0.651612	0.490523	1.646907	1.461340	1.653010	1.476008	1.634116	1.494808
	10	0.664108	0.475823	0.659942	0.483758	0.663974	0.481865	1.648510	1.470854	1.652941	1.465299	1.648416	1.479606
	15	0.666430	0.481579	0.661071	0.472794	0.652654	0.474448	1.652466	1.479260	1.646816	1.470987	1.630082	1.468815
	20	0.663453	0.473670	0.656280	0.473552	0.657162	0.494181	1.648699	1.468774	1.664343	**1.447603**	1.639657	1.499030
	25	0.664186	0.482093	0.660999	0.478252	0.664310	0.477196	1.648665	1.480045	1.632699	1.474569	1.648772	1.473142
	30	0.663848	0.475844	0.660410	0.473684	0.662804	0.475522	1.647983	1.470782	1.665490	1.534244	1.646113	1.470209
	35	0.663664	0.475984	0.661486	0.476440	0.660980	0.474636	1.647791	1.470985	1.652891	1.469242	1.643324	1.468975
	40	0.662140	0.472878	0.663302	0.476905	0.661894	0.473282	1.645226	1.466477	1.651192	1.464443	1.644701	1.466898
	45	0.662623	0.473504	0.662916	0.475594	0.663262	0.475767	1.645924	1.467272	1.651281	1.472824	1.646795	1.470468
30	5	0.656277	0.482098	0.664200	0.490480	0.663412	0.474134	1.638502	1.480626	1.637351	1.468600	1.649965	1.468718
	10	0.662581	0.476177	0.657948	0.476737	0.650469	0.499259	1.648521	1.472004	1.651169	1.491456	1.634567	1.508933

		RMSE						GRMSE					
Input Lags	Hidden Nodes	BPNN		BPNN with LMedS		Enhanced		Ordinary		BPNN with LMedS		Enhanced	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing
	15	0.656869	0.473406	0.651050	0.478823	0.659942	0.483758	1.640292	1.468027	1.652466	1.479260	1.643886	1.483220
	20	0.662589	0.479835	0.659267	0.482879	0.661071	0.472794	1.647819	1.476954	1.648699	1.468774	1.645170	1.466366
	25	0.659706	**0.464590**	0.659794	0.477955	0.656280	0.473552	1.643938	**1.455309**	1.646907	1.461340	1.638110	1.468092
	30	0.661555	0.478927	0.651830	0.478924	0.660999	0.478252	1.646475	1.475839	1.648510	1.470854	1.644894	1.474373
	35	0.661486	0.476440	0.670497	0.473805	0.660410	0.473684	1.645806	1.471706	1.652466	1.479260	1.644289	1.468009
	40	0.661402	0.476905	0.670574	0.476960	0.659660	0.473114	1.645521	1.472729	1.645492	1.481313	1.642961	1.466941
	45	0.662916	0.479935	0.671349	0.474642	0.660335	0.474424	1.648247	1.477024	1.641000	1.462110	1.643844	1.468910
35	5	0.663420	0.538865	0.666864	0.477336	**0.521857**	0.461651	1.656305	1.571833	**1.516604**	1.453920	**1.477958**	1.458276
	10	0.658923	0.502608	0.665885	0.477754	0.657306	0.462124	1.647216	1.513751	1.637351	1.468600	1.643419	1.453981
	15	0.660260	0.473839	0.666289	0.478569	0.660452	0.482218	1.646790	1.469485	1.651169	1.491456	1.646278	1.480476
	20	0.659249	0.482256	0.668260	0.476251	0.664200	0.490480	1.645492	1.481313	1.633582	1.457752	1.653681	1.494168
	25	0.657189	0.469727	0.666444	0.473154	0.657948	0.476737	1.641000	1.462110	1.646475	1.475839	1.642174	1.472389
	30	0.656631	0.474076	0.665955	0.472750	0.651050	0.478823	1.640256	1.468714	1.645806	1.471706	1.632056	1.475913

		RMSE						GRMSE					
Input Lags	**Hidden Nodes**	**BPNN**		**BPNN with LMedS**		**Enhanced**		**Ordinary**		**BPNN with LMedS**		**Enhanced**	
		Training	**Testing**	**Training**	**Testing**	**Training**	**Testing**	**Training**	**Testing**	**Training**	**Testing**	**Training**	**Testing**
	35	0.658087	0.478769	0.666943	0.474061	0.659267	0.482879	1.642849	1.475643	1.645521	1.472729	1.644571	1.482091
	40	0.659985	0.478587	0.667285	0.477046	0.659794	0.477955	1.645433	1.475424	1.648247	1.477024	1.645094	1.474496
	45	0.659113	0.481130	0.666583	0.473032	0.651830	0.478924	1.643901	1.478859	1.642773	1.476733	1.632910	1.475826
	5	**0.633592**	0.467005	**0.553211**	0.476555	0.641201	0.482612	**1.606661**	1.458996	1.640957	1.482443	1.621567	1.482817
	10	0.653957	0.476273	0.663848	0.475844	0.650680	0.473403	1.638946	1.472758	1.647983	1.470782	1.632850	1.467694
	15	0.656075	0.473874	0.663664	0.475984	0.653707	0.470076	1.641873	1.468805	1.647791	1.470985	1.637999	1.463066
	20	0.653042	0.473614	0.660999	0.478252	0.650928	0.478455	1.637351	1.468600	1.645226	1.466477	1.633397	1.475163
40	25	0.659482	0.487875	0.660410	0.473684	0.657254	0.477244	1.651169	1.491456	1.651682	1.473061	1.644063	1.473698
	30	0.651335	0.466622	0.661486	0.476440	0.655640	0.478212	1.633582	1.457752	1.647317	1.472630	1.640863	1.474613
	35	0.654431	0.478879	0.661071	0.472794	0.656682	0.480110	1.638640	1.475596	1.652391	1.470230	1.642505	1.477503
	40	0.657019	0.479705	0.656280	0.473552	0.655410	0.478927	1.642773	1.476733	1.652463	1.469996	1.640648	1.476203
	45	0.655891	0.483571	0.660999	0.478252	0.653737	0.472000	1.640957	1.482443	1.653590	1.476544	1.637444	1.465467

Appendix C

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - RMSE

Training value

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
5	5	0.450142	**0.373162**	0.471096	0.485872	0.488382	0.447253	0.449358	0.487633	0.494545	0.453472	0.441219	0.386238	0.445801	0.453472
	10	0.450623	0.444911	0.487959	0.449691	0.451217	0.442434	0.445155	0.445737	0.454576	0.453490	0.453082	0.446593	0.438536	0.453490
	15	0.453731	0.432660	0.451266	0.450655	0.455530	0.451514	0.440373	0.463326	0.458522	0.452468	0.449815	0.457113	0.454685	0.452468
	20	0.452363	0.450822	0.453750	0.450142	0.448983	0.447833	0.451955	0.455046	0.463326	0.434473	0.452156	0.455053	0.454198	0.444159
	25	**0.392210**	0.453416	0.449915	0.450623	0.453772	0.452066	0.451982	0.454736	0.454547	0.453705	0.450084	0.457007	0.451367	0.434433
	30	0.455439	0.450508	0.449541	0.453731	0.449418	0.450570	0.447525	0.453733	0.453965	0.451593	0.452243	0.457787	0.453965	0.450508
	35	0.454253	0.453060	0.451264	0.452363	0.453077	0.452140	0.454863	0.456911	0.454878	0.452065	0.449983	0.457921	0.454878	0.453060
	40	0.456384	0.451690	0.449928	0.453241	0.454514	0.451744	0.454266	0.456083	0.457863	0.453592	0.451356	0.455897	0.457863	0.451690
	45	0.449853	0.450673	0.452163	0.449154	0.452084	0.452105	0.451110	0.453655	0.455484	0.455630	0.452969	0.453335	0.455484	0.450673
10	5	0.450084	0.473454	0.487633	0.453705	**0.423050**	0.449796	0.475820	0.447112	0.441219	**0.375665**	0.450759	0.458512	**0.365543**	0.471096
	10	0.452243	0.450985	0.445737	0.449075	0.455439	0.447139	0.449711	0.458512	0.453082	0.454122	0.454694	0.461534	0.444911	0.487959
	15	0.449983	0.455046	0.463326	0.448304	0.454253	0.447768	0.445921	0.461534	0.449815	0.454865	0.455046	0.463326	0.432660	0.451266

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	20	0.454685	0.453710	0.451211	0.449973	0.456384	0.450707	0.450359	0.453664	0.450759	0.438536	0.453710	0.451211	0.450822	0.453750
	25	0.451394	0.453705	0.450084	0.448466	0.449853	0.450067	0.451414	0.453429	0.454694	0.454685	0.453705	0.450084	0.434433	0.454865
	30	0.454839	0.451593	0.452243	0.452081	0.451375	0.453813	0.453730	0.452010	0.454371	0.451394	0.451593	0.452243	0.446593	0.438536
	35	0.451211	0.452065	0.449983	0.452543	0.450487	0.455934	0.452308	0.453060	0.451264	0.454839	0.452065	0.449983	0.457113	0.454685
	40	0.450084	0.453592	0.451356	0.451036	0.454158	0.454696	0.453160	0.451690	0.449928	0.456928	0.449541	0.453731	0.454995	0.454329
	45	0.452243	0.455630	0.452969	0.451470	0.453508	0.453193	0.453189	0.450673	0.452163	0.451538	0.451264	0.452363	0.452065	0.449983
15	5	0.450067	0.450239	0.436904	0.447349	0.441561	0.437947	0.436275	0.446593	0.438536	0.454122	0.454379	0.487633	0.379449	0.464192
	10	0.453813	0.451889	0.457784	0.448172	0.449761	0.444678	0.456957	0.457113	0.454685	0.454865	0.448814	0.445737	0.454122	0.454379
	15	0.455934	0.454198	0.444159	0.450985	0.445632	0.444672	0.458250	0.454122	0.458512	0.453082	0.447277	0.455134	0.454865	0.448814
	20	0.454696	0.451367	0.434433	0.450842	0.441050	0.453531	0.455401	0.454865	0.461534	0.449815	0.446213	0.453658	0.452065	0.454576
	25	0.453193	0.447277	0.455134	0.454151	0.449290	0.451888	0.451551	0.452156	0.453664	0.450759	0.447277	0.446593	0.438536	0.458522
	30	0.458664	0.446213	0.453658	0.454723	0.454012	0.454787	0.454796	0.452889	0.453429	0.454694	0.446213	0.457113	0.454685	0.459704
	35	0.450439	0.453730	0.450771	0.452362	0.452710	0.454585	0.452638	0.452491	0.452010	0.454371	0.453730	0.454995	0.454329	0.451337
	40	0.454659	0.452844	0.454832	0.455887	0.453610	0.452483	0.455879	0.454685	0.456247	0.456556	0.452844	0.455897	0.452349	0.455897
	45	0.454448	0.452793	0.453474	0.455631	0.447277	0.447041	0.454220	0.454329	0.452793	0.453474	0.452793	0.453335	0.456286	0.453335
20	5	0.453012	0.379449	0.464192	0.504881	0.456339	**0.338947**	0.387565	0.473454	0.487633	0.450239	0.436904	0.485872	0.494545	0.453472
	10	0.453259	0.454122	0.454379	0.448603	0.444484	0.449747	0.453258	0.450985	0.445737	0.451889	0.457784	0.449691	0.454865	**0.309335**

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	15	0.455867	0.454865	0.448814	0.450581	0.454418	0.458664	0.438579	0.451367	0.434433	0.454198	0.444159	0.450655	0.452156	0.454658
	20	0.455456	0.452156	0.450043	0.452381	0.459187	0.450439	0.463858	0.447277	0.455134	0.451367	0.434433	0.450142	0.452889	0.434473
	25	0.454615	0.452889	0.457007	0.453990	0.453373	0.454659	0.453012	0.455134	0.452054	0.447277	0.455134	0.450623	0.452491	0.443136
	30	0.449589	0.452491	0.457787	0.453525	0.454139	0.454448	0.453259	0.453658	0.451394	0.446213	0.453658	0.450508	0.449541	0.455895
	35	0.458457	0.455334	0.457921	0.458325	0.455175	0.455622	0.455867	0.450771	0.454839	0.459704	0.455053	0.453060	0.451264	0.454502
	40	0.453082	0.452349	0.455897	0.454477	0.454981	0.455080	0.455456	0.454832	0.456928	0.451337	0.452054	0.451690	0.449928	0.456408
	45	0.449815	0.456286	0.453335	0.452697	0.454604	0.453757	0.452564	0.453474	0.451538	0.452793	0.453474	0.450673	0.452163	0.455558
25	5	0.438536	0.494545	0.453472	0.442839	0.429941	0.446236	0.455134	0.441219	0.386238	0.445801	0.464192	0.494545	0.386238	0.445801
	10	0.454685	0.454576	0.453490	0.453279	0.457232	0.450986	0.453658	0.453082	0.446593	0.438536	0.454379	0.454576	0.446593	0.438536
	15	0.454379	0.458522	0.452468	0.444129	0.451022	0.446978	0.438536	0.449815	0.457113	0.454685	0.448814	0.458522	0.454379	0.438536
	20	0.448814	0.459704	0.455053	0.453247	0.454615	0.457823	0.454685	0.450759	0.458512	0.453082	0.446593	0.438536	0.448814	0.454685
	25	0.450043	0.451337	0.452054	0.451428	0.449589	0.445596	0.454329	0.454694	0.461534	0.449815	0.457113	0.454685	0.450043	0.454329
	30	0.452163	0.453769	0.451394	0.457187	0.458457	0.454683	0.458512	0.453082	0.446593	0.438536	0.454122	0.454379	0.457007	0.454547
	35	0.453664	0.454738	0.454839	0.452106	0.457540	0.455856	0.461534	0.449815	0.457113	0.454685	0.454865	0.448814	0.457787	0.453965
	40	0.453429	0.454961	0.456928	0.443146	0.452500	0.449045	0.453664	0.450759	0.453429	0.454694	0.452156	0.450043	0.457921	0.454878
	45	0.452010	0.452909	0.451538	0.454631	0.454226	0.455807	0.456286	0.453335	0.450673	0.452163	0.450673	0.452163	0.456286	0.453335
30	5	0.452054	0.447112	0.441219	0.430312	0.435308	0.443358	0.466238	0.445801	0.494545	0.453472	0.386238	0.445801	0.494545	0.453472

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	10	0.449541	0.458512	0.453082	0.457533	0.458698	0.426904	0.454379	0.441361	0.454576	0.453490	0.446593	0.438536	0.454368	0.434473
	15	0.451264	0.461534	0.449815	0.457075	0.463312	0.449820	0.448814	0.454685	0.458522	0.452468	0.457113	0.454685	0.432660	0.451266
	20	0.454379	0.453664	0.450759	0.453840	0.458524	0.457213	0.450043	0.454329	0.459704	0.455053	0.457113	0.454685	0.450822	0.453750
	25	0.448814	0.453429	0.454694	0.458683	0.445632	0.455091	0.457007	0.454547	0.451337	0.452054	0.454122	0.454379	0.453416	0.449915
	30	0.451264	0.452010	0.454371	0.453815	0.433680	0.457421	0.457787	0.453965	0.450508	0.449541	0.454865	0.448814	0.450508	0.449541
	35	0.449928	0.456247	0.456556	0.456241	0.451806	0.455932	0.457921	0.454878	0.453060	0.451264	0.453060	0.451264	0.453060	0.451264
	40	0.452163	0.454527	0.452791	0.454358	0.454525	0.453775	0.455897	0.457863	0.451690	0.449928	0.451690	0.449928	0.451690	0.449928
	45	0.455734	0.454724	0.454102	0.453598	0.453911	0.455819	0.450673	0.452163	0.450673	0.452163	0.450673	0.452163	0.450673	0.452163
35	5	0.460830	0.433861	0.453046	0.450952	0.441604	0.510441	**0.386238**	0.445801	0.473454	0.433861	0.473454	0.487633	0.386238	0.445801
	10	0.455368	0.444629	**0.429335**	0.432625	0.428087	0.457459	0.446593	**0.438536**	0.450985	0.444629	0.450985	0.445737	0.446593	0.438536
	15	0.463326	0.460593	0.454658	0.432792	0.453673	0.454071	0.457113	0.454685	0.451367	0.460593	0.455046	0.463326	0.457113	0.454685
	20	0.451211	0.454368	0.434473	0.454461	0.458705	0.452019	0.454122	0.454379	0.447277	0.454368	0.453710	0.451211	0.451337	0.452054
	25	0.450084	0.459592	0.443136	0.454260	0.453621	0.455734	0.454865	0.448814	0.455134	0.446593	0.438536	0.457113	0.454685	0.452054
	30	0.454878	0.454076	0.455895	0.458809	0.454601	0.460830	0.452156	0.450043	0.453658	0.457113	0.454685	0.454995	0.454329	0.451394
	35	0.450043	0.453154	0.454502	0.453593	0.453908	0.455368	0.452889	0.457007	0.450771	0.454995	0.454329	0.454736	0.454547	0.454839
	40	0.457007	0.455519	0.456408	0.452324	0.455255	0.456579	0.452349	0.455897	0.451690	0.456083	0.457863	0.451690	0.449928	0.456928
	45	0.457787	0.455269	0.455558	0.458029	0.455943	0.456026	0.456286	0.453335	0.450673	0.453655	0.455484	0.450673	0.452163	0.451538

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
40	5	0.457921	0.386238	0.445801	0.450319	0.451695	0.445107	0.494545	0.453472	**0.379449**	0.464192	**0.373053**	**0.354103**	0.379449	0.464192
	10	0.454685	0.446593	0.438536	**0.424814**	0.454041	0.449521	0.454576	0.453490	0.454122	0.454379	0.454122	0.454379	0.454122	0.454379
	15	0.454329	0.457113	0.454685	0.455704	0.447750	0.460681	0.458522	0.452468	0.454865	0.448814	0.454865	0.448814	0.459704	0.455053
	20	0.457863	0.454995	0.454329	0.451024	0.453643	0.457221	0.454865	0.455046	0.463326	0.434473	0.452156	0.450043	0.451337	0.452054
	25	0.455484	0.454736	0.454547	0.458018	0.454945	0.451081	0.438536	0.453710	0.451211	0.443136	0.452889	0.457007	0.453769	0.451394
	30	0.446593	0.453733	0.453965	0.459237	0.457546	0.452369	0.454685	0.453705	0.450084	0.455895	0.452491	0.457787	0.454738	0.454839
	35	0.457113	0.456911	0.454878	0.453535	0.455318	0.456386	0.455334	0.457921	0.454878	0.454502	0.455334	0.457921	0.453060	0.454122
	40	0.451337	0.456083	0.457863	0.455402	0.446263	0.457617	0.452349	0.455897	0.457863	0.456408	0.452349	0.455897	0.451690	0.454865
	45	0.454685	0.453655	0.455484	0.459185	0.438877	0.456187	0.451538	0.450673	0.452163	0.455558	0.450673	0.452163	0.450673	0.452156

Appendix D

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - RMSE

Testing value

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
5	5	0.557426	**0.356290**	0.553274	0.543054	0.519956	0.551161	0.554471	0.545280	0.569370	0.552980	0.545061	0.545061	0.566665	0.370807
	10	0.554884	0.552308	0.529606	0.561660	0.563652	0.554582	0.546305	0.565324	0.551468	0.555437	0.545950	0.565886	0.552171	0.566995
	15	0.570969	0.495712	0.564314	0.559771	0.564454	0.559346	0.502357	0.551305	0.552172	0.563632	0.560931	0.558982	0.558228	0.553150
	20	0.561114	0.545351	0.565647	0.560656	0.560970	0.552690	0.561663	0.559735	0.562645	0.563088	0.555058	0.558963	0.567871	0.577234
	25	0.546216	0.569370	0.560891	0.565444	0.567277	0.567565	0.557426	0.555707	0.551600	0.568916	0.545950	0.548200	0.530796	0.564730
	30	0.547025	0.563406	0.557888	0.566706	0.555114	0.560557	0.554884	0.567619	0.557726	0.540845	0.546390	0.557672	0.537356	0.569001
	35	0.562162	0.567633	0.564426	0.564762	0.564218	0.562188	0.570969	0.560931	0.557726	0.568341	0.550522	0.566300	0.552059	0.566278
	40	0.557232	0.567589	0.557729	0.571442	0.569437	0.561802	0.569545	0.555058	0.570411	0.563577	0.560258	0.567381	0.567526	0.565029
	45	0.555108	0.560634	0.561094	0.561165	0.565222	0.567859	0.559727	0.565469	0.535820	0.567979	0.566061	0.565778	0.571474	0.566133
10	5	0.566665	0.552980	0.545061	0.566665	0.525453	0.561114	0.551575	0.543054	**0.519956**	0.551161	0.566665	0.535603	0.537654	0.533048
	10	0.564171	0.555437	0.545950	0.548200	0.561187	0.546216	0.553988	0.561660	0.563652	0.554582	0.548200	0.561636	0.544785	0.539980
	15	0.575181	0.566665	0.546390	0.557672	0.564957	0.547025	0.545280	0.569370	0.560891	0.546390	0.557672	0.559771	0.564781	0.568916
	20	0.561663	0.564171	0.565886	0.552171	0.557652	0.562162	0.565324	0.551468	0.549234	0.556331	0.539980	0.560656	0.563620	0.540845
	25	0.557426	0.575181	0.558982	0.558228	0.551305	0.557232	0.560809	0.571680	0.544699	0.546762	0.543774	0.565444	0.563632	0.560931
	30	0.554884	0.566038	0.558963	0.567871	0.559735	0.555108	0.565411	0.555707	0.551600	0.564171	0.565886	0.566706	0.563088	0.555058

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	35	0.570969	0.562377	0.560838	0.562604	0.562500	0.564068	0.566940	0.567619	0.557726	0.575181	0.558982	0.564762	0.566106	0.564288
	40	0.569545	0.552978	0.559959	0.556732	0.562072	0.573422	0.571640	0.563632	0.560931	0.572932	0.567381	0.571442	0.552978	0.559959
	45	0.559727	0.574284	0.568473	0.572720	0.567598	0.573173	0.568620	0.563088	0.555058	0.566300	0.566061	0.565738	0.574284	0.568473
15	5	0.562162	0.556654	0.517091	0.546106	0.543153	0.533048	0.536036	0.552980	0.545061	0.566665	**0.356290**	0.553274	**0.330891**	**0.349321**
	10	0.557232	0.562322	0.551468	0.549234	0.556331	0.539980	0.564610	0.555437	0.545950	0.548200	0.552308	0.529606	0.549234	0.552308
	15	0.555108	0.560376	0.571680	0.544699	0.546762	0.543774	0.553434	0.572932	0.546390	0.557672	0.495712	0.564314	0.544699	0.495712
	20	0.564068	0.558390	**0.515865**	0.555707	0.551600	0.568341	0.550522	0.566300	0.565647	0.569370	0.560891	0.556649	0.551600	0.560970
	25	0.573422	0.548461	0.564339	0.567619	0.557726	0.563577	0.560258	0.567381	0.560891	0.555707	0.551600	0.563993	0.563652	0.567277
	30	0.573173	0.550966	0.574451	0.558039	0.570411	0.564985	0.561179	0.561523	0.557888	0.567619	0.557726	0.572932	0.558822	0.555114
	35	0.544478	0.566867	0.555715	0.562728	0.563311	0.561030	0.561330	0.565167	0.564426	0.555707	0.551600	0.568916	0.545950	0.564218
	40	0.541548	0.565539	0.566213	0.563369	0.566606	0.554585	0.572641	0.567984	0.563620	0.567619	0.557726	0.540845	0.546390	0.563088
	45	0.566592	0.564004	0.565804	0.567989	0.556732	0.550805	0.562125	0.561165	0.565222	0.567965	0.569476	0.565223	0.566061	0.565738
20	5	0.565283	0.363251	0.541995	0.548481	0.544478	**0.320704**	**0.371707**	0.542078	0.540243	0.545061	0.566665	**0.370807**	0.363251	0.356290
	10	0.566995	0.563911	0.563773	0.538573	0.541548	0.565283	0.557324	0.564781	0.568916	0.545950	0.548200	0.539890	0.563911	0.552308
	15	0.553150	0.560303	0.543854	0.550189	0.566592	0.566995	0.538048	0.563620	0.540845	0.546390	0.557672	0.561338	0.560303	0.495712
	20	0.577234	0.574363	0.562025	0.567994	0.557043	0.553150	0.562591	0.563632	0.560931	0.557726	0.569370	0.560891	0.555707	0.551600
	25	0.566636	0.559101	0.576556	0.556947	0.577976	0.577234	0.565334	0.563088	0.555058	0.570411	0.560970	0.552690	0.567619	0.557726
	30	0.570524	0.562729	0.558064	0.555132	0.571274	0.566636	0.564784	0.566106	0.564288	0.563311	0.567277	0.567565	0.562645	0.561811
	35	0.557726	0.568325	0.561183	0.564895	0.565565	0.570524	0.563577	0.560254	0.568173	0.567381	0.569397	0.562377	0.560838	0.568655
	40	0.570411	0.561895	0.562087	0.561401	0.564445	0.562168	0.562979	0.566061	0.565738	0.565904	0.554766	0.552978	0.559959	0.566738
	45	0.563311	0.565168	0.567169	0.564904	0.567661	0.566470	0.564631	0.558822	0.561165	0.571474	0.566133	0.574284	0.568473	0.564858
25	5	0.569055	0.549987	0.542078	0.540243	**0.518792**	0.531499	0.551330	0.543054	0.519956	0.551161	0.521659	0.530800	0.552980	0.545061

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	10	0.555707	0.561524	0.564781	0.568916	0.552118	0.546706	0.530690	0.561660	0.563652	0.554582	0.565886	0.552171	0.555437	0.545950
	15	0.567619	0.559325	0.563620	0.540845	0.555900	0.546293	0.554967	0.580582	0.558822	0.556494	0.558982	0.558228	0.561536	0.562858
	20	0.560254	0.552172	0.563632	0.560931	0.571677	0.563919	0.569055	0.561941	0.559261	0.564909	0.558963	0.567871	0.580582	0.558822
	25	0.552118	0.562645	0.563088	0.555058	0.562866	0.541581	0.555707	0.551600	0.563402	0.560254	0.529537	0.569370	0.560891	0.565834
	30	0.555900	0.568340	0.566106	0.564288	0.562058	0.564148	0.567619	0.557726	0.551884	0.551468	0.549234	0.556331	0.539980	0.560315
	35	0.571677	0.567798	0.560254	0.568173	0.565172	0.561547	0.560254	**0.529537**	0.569792	0.571680	0.544699	0.546762	0.543774	0.573150
	40	0.562866	0.568096	0.569829	0.534214	0.564522	0.562018	0.546762	0.572096	0.568818	0.555132	0.571274	0.566636	0.558390	0.515865
	45	0.562058	0.568098	0.561325	0.569911	0.568147	0.565195	0.565738	0.565221	0.565223	0.564895	0.565565	0.570524	0.548461	0.564339
30	5	0.561523	0.523496	0.535089	0.521659	0.530800	0.527016	0.515865	0.552980	0.545061	**0.540243**	0.518792	0.535603	0.537654	0.562058
	10	0.565167	0.552656	0.553147	0.561523	0.543810	0.520207	0.558982	0.555437	0.545950	0.555707	0.551600	0.561636	0.544785	0.565172
	15	0.567984	0.561536	0.562858	0.565167	0.551094	0.544458	0.558963	0.544785	0.560458	0.567619	0.557726	0.546390	0.557672	0.530796
	20	0.563380	0.580582	0.558822	0.567984	0.560888	0.551678	0.509376	0.558338	0.563402	0.551600	0.561536	0.565647	0.569370	0.537356
	25	0.560254	0.561941	0.559261	0.563380	0.541042	0.561377	0.556780	0.564678	0.551884	0.557726	0.580582	0.565886	0.552171	0.552059
	30	0.552980	0.560846	0.565834	0.560254	0.529537	0.565575	0.569187	0.548272	0.569792	0.570411	0.561941	0.558982	0.558228	0.560458
	35	0.555437	0.565040	0.560315	0.563233	0.564176	0.566282	0.544699	0.551468	0.552039	0.556331	0.539980	0.558963	0.567871	0.563402
	40	0.544785	0.572096	0.573150	0.564345	0.565000	0.568818	0.555707	0.571680	0.544699	0.546762	0.543774	0.509376	0.561636	0.551884
	45	0.557726	0.565221	0.565403	0.567965	0.569476	0.565223	0.569370	0.560891	0.566061	0.565738	0.564730	0.563235	0.560732	0.569792
35	5	0.561536	0.547613	0.559003	0.551196	0.535368	0.522871	0.565886	0.535603	0.537654	0.552980	0.545061	0.543054	0.519956	0.551161
	10	0.560376	0.540774	0.520609	0.524275	0.536603	0.560458	0.558982	0.561636	0.544785	0.555437	0.545950	0.561660	0.563652	0.554582
	15	0.551196	0.552105	0.572007	0.525988	0.556649	0.563402	0.558963	0.565886	0.552171	0.560732	0.558338	0.570411	0.560970	0.552690
	20	**0.524275**	0.567269	0.530796	0.564730	0.563993	0.551884	0.548200	0.558982	0.558228	0.556494	0.564678	0.563311	0.567277	0.567565
	25	0.525988	0.565124	0.537356	0.569001	0.572932	0.569792	0.557672	0.558963	0.567871	0.557726	0.546390	0.555707	0.551600	0.544699

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	30	0.564730	0.556780	0.552059	0.566278	0.566300	0.552039	0.557726	0.569370	0.560891	0.561536	0.565647	0.567619	0.557726	0.555707
	35	0.569001	0.569187	0.567526	0.565029	0.567381	0.569397	0.570411	0.560970	0.552690	0.560376	0.571680	0.544699	0.569370	0.560891
	40	0.572932	0.560996	0.565171	0.563763	0.565904	0.554766	0.564781	0.564171	0.565886	0.558390	0.515865	0.555707	0.558064	0.555132
	45	0.566300	0.566061	0.565738	0.565778	0.571474	0.566133	0.563620	0.575181	0.558982	0.548461	0.564339	0.567619	0.561183	0.564895
40	5	0.567381	0.370807	0.551330	0.570196	0.535603	0.537654	0.552980	0.545061	0.566282	0.543054	0.519956	0.551161	0.370807	0.551330
	10	0.565904	0.539890	0.530690	**0.509376**	0.561636	0.544785	0.555437	0.545950	0.551468	0.561660	0.563652	0.554582	0.539890	0.530690
	15	0.551600	0.561338	0.554967	0.563235	0.560732	0.558338	0.563402	0.558963	0.571680	0.544699	0.546762	0.543774	0.548200	0.558982
	20	0.557726	0.567204	0.569055	0.548150	0.556494	0.564678	0.551884	0.548200	0.567277	0.557726	0.569370	0.560891	0.555707	0.551600
	25	0.569370	0.575877	0.564041	0.561811	0.564909	0.548272	0.569370	0.560891	0.551600	0.570411	0.565886	0.552171	0.567619	0.557726
	30	0.543774	0.571908	0.568133	0.568655	0.566726	0.563573	0.551468	0.549234	0.557726	0.563311	0.558982	0.558228	0.567565	0.563919
	35	0.560891	0.572151	0.567362	0.566738	0.565099	0.556569	0.577976	0.577234	0.569370	0.567381	0.558963	0.567871	0.562377	0.541581
	40	0.552171	0.566037	0.565972	0.564858	0.549672	0.564689	0.571274	0.566636	0.572007	0.565904	0.566300	0.552039	0.552978	0.564148
	45	0.558228	0.557300	0.562345	0.565469	0.535820	0.567979	0.566061	0.565738	0.530796	0.566470	0.564631	0.558822	0.571680	0.544699

Appendix E

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - GRMSE

Training value

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
5	5	1.445223	**1.346845**	1.477029	1.500989	1.506603	1.441607	1.444452	1.434973	1.417198	1.410276	1.451130	1.427408	1.441443	1.446190
	10	1.444398	1.437884	1.504271	1.444792	1.447011	1.434398	1.438254	1.449961	1.455920	1.453478	1.441870	1.456707	1.442617	1.445433
	15	1.449710	1.421189	1.447091	1.446190	1.453353	1.447446	1.432057	1.444904	1.455499	1.446703	1.450587	1.450689	1.448218	1.446079
	20	1.451405	1.446549	1.450730	1.445433	1.443758	1.442117	1.448107	1.442758	1.449728	1.456207	1.458408	1.444398	1.446057	1.450623
	25	1.454615	1.450163	1.445129	1.446079	1.450689	1.448218	1.448131	1.445223	1.444904	1.455499	1.465102	1.451405	1.442048	1.448643
	30	1.445049	1.446016	1.444574	1.450623	1.444398	1.446057	1.441659	1.446797	1.439107	1.447177	**1.376593**	1.454615	1.446219	1.416202
	35	1.446219	1.449689	1.447058	1.448643	1.449710	1.448353	1.452288	1.452551	1.465315	1.452276	1.451202	1.445049	1.445329	1.451953
	40	1.445329	1.447670	1.445135	1.449889	1.451767	1.447773	1.451396	1.455573	1.452073	1.455246	1.452299	1.447235	1.450842	1.423138
	45	1.450842	1.446214	1.448380	1.445632	1.448259	1.448223	1.446851	1.452407	1.451439	1.447416	1.448338	1.446797	1.439107	1.443930
10	5	1.450324	1.480704	1.503867	1.444047	**1.407287**	1.445143	1.484453	1.427408	1.441443	1.416202	1.455573	1.452073	1.434973	1.417198
	10	1.447965	1.446797	1.439107	1.443930	1.453180	1.441149	1.444821	1.456707	1.442617	1.451953	1.452407	1.451439	1.449961	1.455920
	15	1.455499	1.452551	1.465315	1.442758	1.451405	1.442048	1.439429	**1.383593**	1.445927	1.423138	1.444904	1.455499	1.465102	1.445069
	20	1.446219	1.450610	1.446958	1.445223	1.454615	1.446219	1.445705	1.451202	1.451405	1.442048	1.442758	1.451405	1.442048	1.438922
	25	1.442048	1.450572	1.445401	1.443018	1.445049	1.445329	1.447284	1.452299	1.454615	1.446219	1.445223	1.454615	1.446219	1.432325
	30	1.446219	1.447523	1.448496	1.448217	1.447235	1.450842	1.450630	1.448338	1.445049	1.445329	1.444398	1.446057	1.447177	1.444189

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	35	1.449710	1.448231	1.445184	1.448927	1.445927	1.453886	1.448525	1.410276	1.451130	1.450842	1.449710	1.448353	1.452276	1.445223
	40	1.451767	1.450511	1.447191	1.446765	1.451277	1.452019	1.449784	1.453478	1.441870	1.451202	1.451767	1.447773	1.455246	1.443018
	45	1.448259	1.453390	1.449494	1.447344	1.450278	1.449827	1.449801	1.446703	1.450587	1.452299	1.448223	1.446851	1.447416	1.448217
15	5	1.458768	1.445604	1.427408	1.441443	1.433336	1.428106	1.425656	1.452678	**1.374507**	1.446797	1.439107	**1.325635**	1.444398	1.446057
	10	1.450086	1.448283	1.456707	1.442617	1.444976	1.437671	1.455385	1.450540	1.451953	1.452551	1.465315	1.444811	1.449710	1.448353
	15	1.451171	1.451336	1.436789	1.446726	1.438922	1.437589	1.457382	1.448353	1.423138	1.455573	1.452073	1.457887	1.451767	1.447773
	20	1.452724	1.447210	1.423283	1.446502	1.432325	1.450324	1.453195	1.447773	1.442758	1.444904	1.455499	1.465102	1.445069	1.448627
	25	1.454615	1.441303	1.452678	1.451250	1.444189	1.447965	1.447473	1.451130	1.445223	1.454615	1.446219	1.427408	1.441443	1.451032
	30	1.410276	1.439753	1.450540	1.452147	1.451026	1.452162	1.452204	1.441870	1.447177	1.451405	1.442048	1.456707	1.442617	1.450366
	35	1.453478	1.450609	1.446355	1.448648	1.449132	1.451909	1.449036	1.445900	1.452276	1.454615	1.446219	1.453759	1.433298	1.451130
	40	1.450587	1.449317	1.452218	1.453809	1.450444	1.448845	1.453756	1.446765	1.455246	1.445049	1.445329	1.453151	1.453478	1.441870
	45	1.451439	1.449248	1.450243	1.453409	1.450402	1.440949	1.451352	1.447344	1.447416	1.447235	1.450842	1.448921	1.446703	1.450587
20	5	1.457600	1.354567	1.466875	1.531898	1.454816	**1.304435**	1.365072	1.434973	1.417198	**1.353026**	1.446797	1.439107	1.416202	**1.300325**
	10	1.456202	1.451202	1.451583	1.443505	1.437337	1.444811	1.449985	1.449961	1.455920	1.444811	1.452551	1.465315	1.451953	1.444811
	15	1.448819	1.452299	1.443552	1.446079	1.451624	1.457887	1.428885	1.444904	1.455499	1.465102	1.445069	1.455691	1.457600	1.452011
	20	1.449955	1.448338	1.445248	1.448627	1.458768	1.445900	1.465681	1.427408	1.441443	1.451202	1.531898	1.442758	1.451405	1.442048
	25	1.452563	1.449414	1.455420	1.451032	1.450086	1.451954	1.449566	1.456707	1.442617	1.452299	1.451405	1.442048	1.454615	1.446219
	30	1.448201	1.448814	1.456642	1.450366	1.451171	1.451654	1.449933	1.447177	1.455920	1.448338	1.454615	1.446219	**1.404695**	1.451130
	35	1.447177	1.452946	1.456820	1.457370	1.452724	1.453378	1.453759	1.452276	1.446798	1.444398	1.445049	1.445329	1.453478	1.441870
	40	1.450587	1.448597	1.453808	1.451696	1.452428	1.451032	1.453151	1.455246	1.451859	1.449710	1.447235	1.450842	1.446703	1.450587
	45	1.456615	1.454364	1.450035	1.449111	1.451869	1.450642	1.448921	1.447416	1.444555	1.451767	1.447773	1.435574	1.452407	1.451439
25	5	**1.365482**	1.514594	1.450577	1.434973	1.417198	1.440302	1.447235	1.444398	1.446057	1.451495	1.450756	1.422079	1.433432	1.543961

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	10	1.451202	1.451852	1.450271	1.449961	1.455920	1.446804	1.416202	1.449710	1.448353	1.442758	1.451405	1.442048	1.414062	1.456104
	15	1.457887	1.457740	1.448754	1.436855	1.446798	1.440860	1.451953	1.451767	1.447773	1.445223	1.454615	1.446219	1.450549	1.451108
	20	1.445900	1.459548	1.452563	1.449955	1.451859	1.456615	1.423138	1.455573	1.451405	1.442048	1.416202	1.450483	1.446797	1.439107
	25	1.451954	1.447244	1.448201	1.447279	1.444555	1.438921	**1.354567**	1.452407	1.454615	1.446219	1.451953	1.450187	1.452551	1.465315
	30	1.451654	1.450630	1.447177	1.455691	1.457600	1.452011	1.451202	1.457928	1.445049	1.445329	1.423138	1.448079	1.454520	1.452276
	35	1.455499	1.452063	1.452276	1.448223	1.456202	1.453762	1.452299	1.450756	1.447235	1.450842	1.410276	1.451130	1.456292	1.455246
	40	1.444398	1.452424	1.455246	1.435574	1.448819	1.443777	1.448338	1.454281	1.448627	1.458768	1.453478	1.441870	1.441443	1.446057
	45	1.449710	1.449364	1.447416	1.451898	1.451289	1.453657	1.444904	1.455499	1.465102	1.445069	1.446703	1.450587	1.442617	1.448353
30	5	1.445329	1.441992	1.432807	1.417612	1.424416	1.436258	1.448223	1.405633	1.434973	1.417198	1.446797	1.439107	1.448627	1.458768
	10	1.450842	1.457733	1.449728	1.456207	1.458408	1.432802	1.435574	1.451202	1.449961	1.455920	1.452551	1.465315	1.451032	1.450086
	15	1.444540	1.462197	1.444904	1.455499	1.465102	1.445069	1.451405	1.452299	1.435429	1.449710	1.448353	1.444398	1.446057	1.451993
	20	1.460901	1.450483	1.446322	1.450750	1.457759	1.455824	1.454615	1.448338	1.451202	1.455573	1.452073	1.449710	1.448353	1.450580
	25	1.455680	1.450187	1.452033	1.457928	1.438956	1.452605	1.445049	1.445329	1.457887	1.452407	1.451439	1.451767	1.447773	1.455361
	30	1.456292	1.448079	1.451495	1.450756	1.422079	1.456024	1.447235	1.450842	1.445900	1.451710	1.458011	1.410276	1.451130	1.447177
	35	1.423138	1.454286	1.454784	1.454281	1.447755	1.453825	1.416202	1.451993	1.451748	1.451366	1.450378	1.453478	1.441870	1.452276
	40	1.435407	1.451755	1.449175	1.451522	1.451758	1.450625	1.451953	1.451405	1.442048	1.458057	1.451827	1.446703	1.450587	1.455246
	45	1.453868	1.452023	1.451109	1.450394	1.450826	1.453669	1.423138	1.454615	1.446219	1.446797	1.439107	1.451405	1.442048	1.447416
35	5	1.455920	1.422284	1.449952	1.446631	1.433432	1.543961	1.448627	1.458768	1.451492	1.444904	1.455499	1.465102	1.445069	1.412802
	10	1.455499	1.437500	**1.416202**	1.420771	1.414062	1.456104	1.451032	1.450086	1.459245	1.453271	1.456899	1.450850	1.451202	1.445069
	15	1.441443	1.460848	1.451953	1.420980	1.450549	1.451108	1.457928	1.434973	1.417198	1.456207	1.458408	1.458408	1.452299	1.455824
	20	1.448338	1.451492	1.423138	1.451710	1.458011	1.448389	1.450756	1.449961	1.455920	1.455499	1.465102	1.465102	1.448338	1.455920
	25	1.445329	1.459245	1.435407	1.451366	1.450378	1.453463	1.455573	1.452073	1.444398	1.446057	1.455573	1.452073	1.416202	1.455499

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	30	1.450842	1.451147	1.453868	1.458057	1.451827	1.461227	1.452407	1.451439	1.449710	1.448353	1.452407	1.447177	1.451953	1.444398
	35	1.451993	1.449742	1.451682	1.450367	1.450850	1.452952	1.451993	1.451748	1.451767	1.447773	1.450587	1.452276	1.423138	1.449710
	40	1.451405	1.453233	1.454528	1.448529	1.452810	1.454844	1.451495	1.451405	1.442048	1.451405	1.442048	1.455246	1.410276	1.354567
	45	1.454615	1.452834	1.453271	1.456899	1.453825	1.453925	1.454784	1.454615	1.446219	1.454615	1.446219	1.447416	1.448353	1.451202
40	5	1.442048	1.362943	1.439142	1.446686	1.448232	1.438326	1.434973	1.417198	1.450850	1.445049	1.445329	1.458408	1.420980	1.452299
	10	1.446219	1.440488	1.428989	**1.410276**	1.451130	1.444540	1.449961	1.455920	1.458408	1.447235	1.450842	1.465102	1.451710	1.448338
	15	1.445329	1.455573	1.452073	1.453478	1.441870	1.460901	1.444904	1.455499	1.465102	1.434973	1.417198	1.455573	1.451366	1.449710
	20	1.450842	1.452407	1.451439	1.446703	1.450587	1.455680	1.434973	1.444398	1.446057	1.446797	1.439107	1.452407	1.458057	1.451767
	25	1.448353	1.451993	1.451748	1.456933	1.452335	1.446830	1.449961	1.449710	1.448353	1.452551	1.465315	1.448353	1.450367	1.442048
	30	1.447773	1.450580	1.450921	1.458677	1.456179	1.448594	1.447177	1.451767	1.453868	1.458057	1.451827	1.447773	1.451405	1.442048
	35	1.451130	1.455361	1.452210	1.450236	1.452866	1.454520	1.452276	1.416202	1.451682	1.450367	1.450850	1.450367	1.454615	1.446219
	40	1.441870	1.454017	1.456644	1.453010	1.439784	1.456292	1.455246	1.451953	1.454528	1.448529	1.452810	1.448529	1.448627	1.458768
	45	1.554320	1.450496	1.453195	1.458581	1.429291	1.454151	1.447416	1.423138	1.455499	1.465102	1.445069	1.456899	1.451032	1.450086

Appendix F

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset I - GRMSE

Testing value

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
5	5	1.566099	**1.294195**	1.544594	1.529413	1.495997	1.541293	1.546394	1.561984	1.549773	1.557989	1.575771	1.566615	1.566550	1.569992
	10	1.551189	1.543007	1.509823	1.557372	1.560442	1.546466	1.533969	1.536896	1.554719	1.549773	1.567745	1.551764	1.555651	1.542610
	15	1.541530	1.461775	1.567745	1.567745	1.561749	1.553781	1.471023	1.560260	1.567964	1.574660	1.560319	1.558991	1.558185	1.569370
	20	1.554372	1.532608	1.560319	1.560319	1.556293	1.543596	1.557366	1.504003	1.544319	**1.502453**	1.562503	1.559993	1.561984	1.546752
	25	1.558646	1.569380	1.562503	1.562503	1.566099	1.566550	1.550834	1.562086	1.581822	1.559742	1.556371	1.565133	1.567745	1.564259
	30	1.542823	1.560068	1.556371	1.556371	1.547313	1.555651	1.546936	1.568748	1.565068	1.549136	1.524510	1.561243	1.560319	1.551189
	35	1.552060	1.566667	1.561644	1.562168	1.561330	1.558185	1.571906	1.564491	1.571189	1.534634	1.530147	1.578564	1.562503	1.541530
	40	1.567022	1.566585	1.551292	1.572645	1.569485	1.557579	1.569656	1.561475	1.554447	1.553220	1.552060	1.564148	1.556371	1.554372
	45	1.558808	1.555771	1.556482	1.561644	1.562886	1.567007	1.554369	1.563563	1.555798	1.553188	1.567022	1.567745	1.567745	1.558646
10	5	1.558808	1.544156	1.532404	1.556592	1.503232	1.556533	1.542051	1.549136	1.524510	1.566615	1.552060	1.569992	1.567745	1.547313
	10	1.549773	1.547891	1.533414	1.536826	1.556625	1.533823	1.545576	1.534634	1.530147	1.551764	1.567022	1.542610	1.560319	1.561330
	15	1.574660	1.565133	1.534176	1.551200	1.562466	1.535032	1.532411	1.542823	1.555651	1.561984	1.558808	1.569370	1.562503	1.569485
	20	1.504003	1.561243	1.563913	1.542823	1.551189	1.558120	1.563035	1.552060	1.558185	1.536896	1.567745	1.546752	1.556371	1.554322
	25	1.562086	1.578564	1.553220	1.552060	1.541530	1.550525	1.556036	1.567022	1.557579	1.560260	1.566550	1.564259	1.558808	1.561984
	30	1.568748	1.564148	1.553188	1.567022	1.554372	1.547284	1.563175	1.558808	1.567745	1.544319	1.555651	1.546936	1.549773	1.536896

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	35	1.564491	1.558459	1.556071	1.558808	1.558646	1.561086	1.565561	1.571615	1.560319	1.581822	1.558185	1.571906	1.574660	1.560260
	40	1.562503	1.544044	1.554719	1.549773	1.557989	1.575771	1.572951	1.530423	1.562503	1.565068	1.560319	1.555170	1.562503	1.577256
	45	1.556371	1.577138	1.567964	1.574660	1.566591	1.575374	1.568192	1.559742	1.556371	1.571189	1.562503	1.556601	1.556371	1.553382
15	5	1.561984	1.549632	1.491363	1.533643	1.529212	1.514292	1.518664	1.542823	1.558808	1.569992	1.566615	1.509148	1.563401	1.571615
	10	1.536896	1.558455	1.541766	1.538348	1.549136	1.524510	1.561923	1.552060	1.549773	1.542610	1.551764	1.561213	1.564501	1.480423
	15	1.560260	1.555357	1.573014	1.531556	1.534634	1.530147	1.544747	1.567022	1.574660	1.569370	1.558991	1.544319	1.558808	1.566550
	20	1.549773	1.552298	**1.489631**	1.548214	1.541934	1.567745	1.540325	1.558808	1.504003	1.546752	1.559993	1.581822	1.563035	1.555651
	25	1.574660	1.537179	1.561496	1.566615	1.551277	1.560319	1.555170	1.567745	1.562086	1.564259	1.556371	1.565068	1.562179	1.558185
	30	1.544319	1.540959	1.577394	1.551764	1.571001	1.562503	1.556601	1.560319	1.568748	1.566550	1.550834	1.549773	1.557989	1.575771
	35	1.581822	1.565439	1.548197	1.558991	1.559897	1.556371	1.556827	1.562503	1.564491	1.555651	1.546936	1.567193	1.550248	1.571615
	40	1.565068	1.563363	1.564418	1.559993	1.565029	1.546506	1.574523	1.556371	1.573214	1.558185	1.571906	1.550081	1.583020	1.480423
	45	1.571189	1.560976	1.563780	1.567201	1.564491	1.540718	1.558060	1.567964	1.574660	1.566591	1.575374	1.547306	1.572354	1.559742
20	5	1.558185	1.301676	1.527605	1.537724	1.531247	**1.257188**	**1.310897**	1.559535	1.558808	1.527646	1.524876	**1.504534**	1.566615	**1.326747**
	10	1.567745	1.560822	1.560609	1.522468	1.526826	1.562951	1.550655	1.564230	1.549773	1.549136	1.524510	1.569992	1.551764	1.552060
	15	1.560319	1.555238	1.530340	1.539802	1.565000	1.565637	1.521623	1.555168	1.574660	1.534634	1.530147	1.542610	1.558991	1.567022
	20	1.562503	1.577256	1.557890	1.567193	1.550248	1.544319	1.558808	1.566550	1.550834	1.571615	1.544319	1.569370	1.559993	1.558808
	25	1.556371	1.553382	1.580744	1.550081	1.583020	1.581822	1.563035	1.555651	1.546936	1.503433	1.581822	1.546752	1.569992	1.542823
	30	1.542823	1.558986	1.551806	1.547306	1.572354	1.565068	1.562179	1.558185	1.571906	1.567745	1.565068	1.564259	1.542610	1.504003
	35	1.552060	1.567715	1.556605	1.562361	1.563398	1.571189	1.560306	1.561984	1.556199	1.560319	1.571189	1.555651	1.569370	1.562086
	40	1.567022	1.557698	1.557996	1.556933	1.561652	1.551806	1.559379	1.536896	1.547181	1.562503	1.581822	1.563035	1.546752	1.568748
	45	1.558808	1.562781	1.565900	1.562371	1.566673	1.564809	1.561940	1.560260	1.561406	1.556371	1.565068	1.562179	1.594353	1.564491
25	5	1.561984	1.539833	1.527646	1.524876	**1.493716**	1.512063	1.549136	**1.500245**	1.571615	1.558808	1.572983	1.544319	1.558808	1.568748

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	10	1.536896	1.557115	1.562180	1.568635	1.542728	1.534575	1.534634	1.530147	**1.474956**	1.549773	1.559185	1.581822	1.549773	1.564491
	15	1.560260	1.553736	1.560358	1.525774	1.548538	1.533928	1.566550	1.550834	1.559742	1.574660	1.557952	1.565068	1.558808	1.556199
	20	1.569992	1.542823	1.560383	1.556199	1.572983	1.560834	1.555651	1.546936	1.562345	1.559165	1.555651	1.571189	1.549773	1.547181
	25	1.542610	1.558856	1.559535	1.547181	1.559185	1.526875	1.558185	1.571906	1.565180	1.552940	1.558185	1.569992	1.574660	1.561406
	30	1.569370	1.567729	1.564230	1.561406	1.557952	1.561187	1.561984	1.542823	1.562638	1.553615	1.561984	1.542610	1.504003	1.567471
	35	1.546752	1.566879	1.555168	1.567471	1.562781	1.557157	1.536896	1.552060	1.567715	1.556881	1.558185	1.569370	1.562086	1.555651
	40	1.564259	1.567356	1.570069	1.516011	1.561761	1.557870	1.560260	1.567022	1.557698	1.563401	1.561984	1.546752	1.568748	1.558185
	45	1.562781	1.567348	1.556813	1.570204	1.567425	1.562816	1.552060	1.558808	1.562781	1.564501	1.536896	1.564259	1.564491	1.561984
30	5	1.559535	1.500522	1.517274	1.497792	1.510990	1.505549	1.571615	1.549843	1.571906	1.571189	1.562763	1.560565	1.553265	1.561984
	10	1.564230	1.543546	1.544263	1.557112	1.530249	1.495709	1.542823	1.566435	1.562534	1.566211	1.562086	1.560937	**1.432913**	1.536896
	15	1.555168	1.557144	1.559165	1.562763	1.541206	1.531166	1.552060	1.567745	1.572983	1.555651	1.568748	1.574949	1.573486	1.560260
	20	1.560319	1.587174	1.552940	1.567160	1.556169	1.542057	1.567022	1.560319	1.559185	1.558185	1.571189	1.564259	1.544319	1.569992
	25	1.562503	1.557779	1.553615	1.559991	1.526064	1.556881	1.558808	1.562503	1.557952	1.561984	1.566550	1.550834	1.581822	1.542610
	30	1.556371	1.556056	1.563798	1.555145	1.509148	1.563401	1.571615	1.556371	1.549136	1.524510	1.555651	1.546936	1.565068	1.569370
	35	1.504003	1.562565	1.555244	1.559755	1.561213	1.564501	1.480423	1.549773	1.534634	1.530147	1.558185	1.571906	1.571189	1.546752
	40	1.562086	1.573634	1.575306	1.561490	1.562499	1.568469	1.559742	1.574660	1.542416	1.549136	1.524510	1.556371	1.569992	1.564259
	45	1.568748	1.562843	1.563126	1.567131	1.569504	1.562854	1.542823	1.571906	1.569992	1.534634	1.530147	1.544319	1.542610	1.553615
35	5	1.564491	1.535925	1.553265	1.541294	1.517675	1.500459	1.558808	1.559185	1.567745	1.572983	1.558185	1.571906	1.567745	1.567745
	10	1.547025	1.525646	1.496281	1.501543	1.519445	1.555457	1.549773	1.557952	1.560319	1.559185	1.561984	1.556199	1.560319	1.560319
	15	1.568825	1.542709	1.573486	1.504003	1.549601	1.560006	1.574660	1.567745	1.562503	1.557952	1.536896	1.547181	1.562503	1.562503
	20	1.560991	1.566033	1.510966	1.562086	1.560937	1.542416	1.549136	1.524510	1.556371	1.569992	1.571615	1.566615	1.569992	1.556371
	25	1.567378	1.562694	1.520602	1.568748	1.574949	1.569992	1.534634	1.530147	1.544319	1.542610	**1.497732**	1.551764	1.542610	1.555651

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	30	1.571615	1.549843	1.542612	1.564491	1.564516	1.542610	1.566550	1.550834	1.581822	1.569370	1.559742	1.558991	1.569370	1.558185
	35	**1.491123**	1.569077	1.566435	1.562534	1.566211	1.569370	1.555651	1.546936	1.565068	1.546752	1.567435	1.559993	1.546752	1.561984
	40	1.559742	1.556284	1.562763	1.560565	1.563902	1.546752	1.558185	1.571906	1.571189	1.564259	1.504003	1.550834	1.564259	1.476543
	45	1.493210	1.564147	1.563644	1.563712	1.572649	1.564259	1.502110	1.569992	1.544440	1.532101	1.562086	1.546936	1.560260	1.562503
40	5	1.504003	1.309892	1.541490	1.571615	1.518112	1.521043	1.572983	1.542610	1.566615	1.558808	1.568748	1.552060	1.541530	1.550834
	10	1.562086	1.524355	1.510831	**1.480423**	1.557263	1.531659	1.559185	1.569370	1.551764	1.549773	1.564491	1.567022	1.554372	1.546936
	15	1.568748	1.556808	1.547025	1.559742	1.555854	1.552204	1.557952	1.546752	1.558991	1.574660	1.560032	1.558808	1.558646	1.571906
	20	1.564491	1.565924	1.568825	1.536700	1.549355	1.561984	1.567745	1.564259	1.559993	1.566550	1.550834	1.549773	1.557989	1.569992
	25	1.554321	1.579618	1.560991	1.557541	1.562345	1.536896	1.560319	1.549136	1.524510	1.555651	1.546936	1.568748	1.560319	1.567435
	30	1.544319	1.573320	1.567378	1.568204	1.565180	1.560260	1.562503	1.534634	1.530147	1.542823	1.571906	1.564491	1.562503	1.504003
	35	1.581822	1.573709	1.566170	1.565193	1.562638	1.549498	1.556371	1.543552	1.555651	1.552060	1.549773	1.557952	1.556371	1.562086
	40	1.565068	1.564101	1.564005	1.562262	1.538973	1.562004	1.560565	1.563902	1.558185	1.567022	1.574660	1.567745	1.559742	1.574660
	45	1.571189	1.550588	1.558366	1.563224	1.518317	1.567146	1.563712	1.572649	1.561984	1.558808	1.549136	1.524510	1.542823	1.571906

Appendix G

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - RMSE

Training value

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
5	5	0.621849	0.762099	0.649943	0.601084	0.322232	0.761779	0.756135	0.736474	0.761514	0.339530	0.602617	0.319408	0.762255	0.766538
	10	0.755175	0.655556	0.763529	0.727068	0.656363	0.757960	0.712695	0.703071	0.678108	0.622531	0.761030	0.722989	0.752929	0.746775
	15	0.701901	0.699004	0.694056	0.654595	0.643149	0.685932	0.628105	0.777896	0.714377	0.761985	0.707293	0.738298	0.675278	0.748157
	20	0.701901	0.716917	0.618915	0.699732	0.616656	0.671680	0.693679	0.700626	0.665571	0.775970	0.760270	0.693491	0.733879	0.737100
	25	0.704822	0.659350	0.695823	0.678612	0.637604	0.675647	0.654111	0.736921	0.754066	0.764418	0.754066	0.687683	0.750911	0.751044
	30	0.739789	0.693320	0.663364	0.651883	0.669010	0.687683	0.707293	0.750911	0.797605	0.763470	0.755823	0.740089	0.749426	0.746581
	35	0.680414	0.728295	0.706282	0.631261	0.662381	0.777775	0.750449	0.716858	0.743508	0.734228	0.764020	0.748171	0.761985	0.723298
	40	0.703717	0.753300	0.717546	0.718109	0.735626	0.737629	0.748802	0.746581	0.745776	0.734341	0.740089	0.762255	0.775970	0.706282
	45	0.668707	0.730528	0.687835	0.674892	0.721213	0.733503	0.742860	0.723298	0.742905	0.733577	0.722989	0.752929	0.764418	0.717546
10	5	0.303667	0.761041	0.763557	0.488978	0.492325	0.761514	0.322814	0.602617	0.321025	0.777896	0.750449	0.716858	0.743508	0.734228
	10	0.628105	0.701594	0.760511	0.706288	0.753593	0.678108	0.622531	0.761030	0.693679	0.700626	0.748802	0.746581	0.745776	0.734341
	15	0.693679	0.706635	0.711331	0.659283	0.627584	0.714377	0.671680	0.699004	0.654111	0.736921	0.742860	0.723298	0.742905	0.733577
	20	0.654111	0.640685	0.757981	0.660057	0.678953	0.665571	0.675647	0.716917	0.707293	0.750911	0.757960	0.712695	0.706282	0.631261
	25	0.707293	0.738298	0.675278	0.700760	0.657789	0.754066	0.687683	0.659350	0.695823	0.749426	0.720756	0.766538	0.717546	0.718109
	30	0.760270	0.693491	0.733879	0.729471	0.746141	0.797605	0.777775	0.693320	0.663364	0.761985	0.720756	0.746775	0.740089	0.762255

Input Lags	Hidden Nodes	Outliers													
		0%	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**
	35	0.740598	0.725060	0.725466	0.668557	0.661120	0.728295	0.706282	0.631261	0.645380	0.775970	0.735342	0.748157	0.722989	0.752929
	40	0.761584	0.736872	0.720481	0.749989	0.677001	0.753300	0.717546	0.755823	0.628105	0.764418	0.725466	0.737100	0.737629	0.742906
	45	0.759970	0.746195	0.745656	0.733853	0.736863	0.777775	0.693320	0.764020	0.748171	0.763470	0.720481	0.751044	0.733503	0.701899
15	5	0.322547	0.755417	0.750031	0.318456	0.675262	0.736474	0.761514	0.322814	0.602617	0.761779	0.756135	0.736474	0.766538	0.290394
	10	0.696928	0.721073	0.729681	0.633985	0.770636	0.703071	0.678108	0.622531	0.761030	0.757960	0.712695	0.703071	0.746775	0.775970
	15	0.711040	0.761720	0.720756	0.577605	0.671680	0.716858	0.728295	0.706282	0.631261	0.699004	0.694056	0.714377	0.748157	0.764418
	20	0.684826	0.786814	0.720756	0.712896	0.675647	0.746581	0.753300	0.717546	0.718109	0.716917	0.618915	0.665571	0.737100	0.763470
	25	0.764401	0.743518	0.735342	0.705336	0.687683	0.723298	0.777896	0.628105	0.671680	0.659350	0.695823	0.754066	0.755823	0.761895
	30	0.719133	0.748665	0.755971	0.738210	0.777775	0.706282	0.700626	0.693679	0.675647	0.693320	0.663364	0.797605	0.764020	0.748171
	35	0.749783	0.741873	0.737537	0.761761	0.723271	0.717546	0.736921	0.750449	0.716858	0.743508	0.734228	0.740089	0.762255	0.777775
	40	0.754241	0.762835	0.737629	0.742906	0.677495	0.756135	0.750911	0.748802	0.746581	0.745776	0.734341	0.722989	0.752929	0.706282
	45	0.721649	0.750774	0.733503	0.701899	0.722917	0.744392	0.749426	0.742860	0.723298	0.742905	0.733577	0.730331	0.740720	0.717546
20	5	0.755125	0.761514	0.322814	0.602617	0.617119	0.777896	0.714377	0.750449	0.716858	0.743508	0.734228	0.743508	0.321930	0.734228
	10	0.695285	0.678108	0.622531	0.761030	0.681942	0.700626	0.665571	0.748802	0.746581	0.745776	0.734341	0.745776	0.734341	0.734341
	15	0.743129	0.711327	0.764488	0.746604	0.750473	0.736921	0.754066	0.742860	0.723298	0.742905	0.733577	0.699004	0.694056	0.733577
	20	0.748858	0.687405	0.700996	0.694896	0.657791	0.750911	0.797605	0.728295	0.706282	0.631261	0.675647	0.716917	0.618915	0.707293
	25	0.730331	0.740720	0.751165	0.651071	0.751143	0.749426	0.766538	0.753300	0.717546	0.718109	0.687683	0.659350	0.695823	0.712484
	30	0.722491	0.738738	0.763710	0.761518	0.718049	0.761985	0.746775	0.761779	0.756135	0.736474	0.777775	0.693320	0.663364	0.706282
	35	0.734710	0.746364	0.754324	0.729984	0.620632	0.775970	0.748157	0.757960	0.712695	0.703071	0.765479	0.740568	0.673217	0.764274
	40	0.744706	0.753857	0.732753	0.683587	0.725072	0.764418	0.737100	0.737629	0.742906	0.740089	0.762255	0.755823	0.734228	0.754135
	45	0.755147	0.759595	0.737498	0.725444	0.725917	0.763470	0.751044	0.733503	0.701899	0.722989	0.752929	0.764020	0.748171	0.723198
25	5	0.761779	0.756135	0.736474	0.322181	0.673217	0.628105	0.751165	0.312548	0.673217	0.665432	0.654321	0.761514	0.322814	0.602617

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	10	0.757960	0.712695	0.703071	0.676706	0.702495	0.693679	0.763710	0.676706	0.702495	0.671680	0.699004	0.678108	0.622531	0.761030
	15	0.726048	0.777896	0.765479	0.740568	0.797990	0.654111	0.754324	0.766538	0.761985	0.675647	0.716917	0.618915	0.763710	0.761518
	20	0.725912	0.700626	0.764274	0.776312	0.744186	0.707293	0.736474	0.746775	0.775970	0.687683	0.659350	0.695823	0.754324	0.729984
	25	0.725912	0.736921	0.754135	0.752916	0.731470	0.714377	0.703071	0.748157	0.764418	0.777775	0.693320	0.663364	0.732753	0.683587
	30	0.752188	0.750911	0.723198	0.684439	0.762786	0.665571	0.734710	0.737100	0.763470	0.728295	0.706282	0.631261	0.740089	0.762255
	35	0.745577	0.749426	0.723547	0.748018	0.747697	0.754066	0.744706	0.751044	0.750449	0.716858	0.743508	0.734228	0.722989	0.752929
	40	0.748392	0.740866	0.763235	0.727755	0.711229	0.797605	0.737629	0.742906	0.748802	0.746581	0.745776	0.734341	0.743508	0.734228
	45	0.746051	0.756719	0.733189	0.743517	0.744420	0.754432	0.733503	0.701899	0.742860	0.723298	0.742905	0.733577	0.745776	0.734341
30	5	0.714377	0.764630	0.322465	0.656544	0.686448	0.766538	0.755823	0.322399	0.777896	0.761514	0.321395	0.602617	0.706282	0.631261
	10	0.665571	0.684873	0.708437	0.744368	0.623470	0.746775	0.764020	0.748171	0.700626	0.678108	0.622531	0.761030	0.717546	0.718109
	15	0.754066	0.722867	0.772105	0.717105	0.746852	0.748157	0.718119	0.754012	0.736921	0.654111	0.737629	0.742906	0.699004	0.694056
	20	0.797605	0.732615	0.755648	0.712967	0.739903	0.737100	0.764940	0.763534	0.750911	0.707293	0.733503	0.701899	0.716917	0.618915
	25	0.774338	0.750449	0.716858	0.743508	0.734228	0.751044	0.753882	0.761677	0.749426	0.761779	0.756135	0.736474	0.659350	0.695823
	30	0.740789	0.748802	0.746581	0.745776	0.734341	0.702240	0.743848	0.767212	0.761985	0.757960	0.712695	0.703071	0.693320	0.671680
	35	0.745847	0.742860	0.723298	0.742905	0.733577	0.762423	0.749466	0.761985	0.775970	0.722989	0.750449	0.716858	0.743508	0.734228
	40	0.748566	0.715951	0.756389	0.719955	0.755463	0.740089	0.762255	0.775970	0.764418	0.743508	0.748802	0.746581	0.745776	0.734341
	45	0.749734	0.744072	0.761794	0.753657	0.753158	0.722989	0.752929	0.764418	0.763470	0.745776	0.742860	0.723298	0.742905	0.733577
35	5	0.747326	0.762858	0.313086	0.640688	0.400351	0.718119	0.754012	0.710415	0.740089	0.762255	0.761514	0.322814	0.602617	0.774338
	10	0.763522	0.652732	0.702240	0.743848	0.718530	0.764940	0.763534	0.744373	0.722989	0.752929	0.678108	0.622531	0.761030	0.740789
	15	0.767212	0.766538	0.762423	0.749466	0.761056	0.742860	0.723298	0.671680	0.714377	0.766538	0.628105	0.777896	0.761985	0.745847
	20	0.761985	0.746775	0.753927	0.760929	0.752819	0.715951	0.756389	0.675647	0.665571	0.746775	0.693679	0.700626	0.775970	0.742860
	25	0.775970	0.748157	0.749942	0.661494	0.763653	0.744072	0.761794	0.687683	0.754066	0.748157	0.654111	0.736921	0.764418	0.757960

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	30	0.764418	0.737100	0.740089	0.762255	0.731137	0.764020	0.761514	0.777775	0.797605	0.737100	0.707293	0.750911	0.763470	0.720756
	35	0.763470	0.751044	0.722989	0.752929	0.735963	0.734228	0.678108	0.737629	0.742906	0.751044	0.712484	0.749426	0.734250	0.720756
	40	0.751307	0.756845	0.735732	0.744692	0.742435	0.734341	0.654111	0.733503	0.701899	0.728295	0.706282	0.631261	0.755823	0.322399
	45	0.757414	0.764305	0.762182	0.760949	0.705330	0.733577	0.757960	0.712695	0.703071	0.753300	0.717546	0.718109	0.764020	0.748171
	5	0.755823	0.322399	0.768660	0.722412	0.325747	0.742860	0.737100	0.766538	0.740089	0.762255	0.731137	0.764020	0.753882	0.761677
	10	0.764020	0.748171	0.735793	0.718648	0.618797	0.757960	0.761985	0.746775	0.728295	0.706282	0.631261	0.711327	0.756211	0.734856
	15	0.759987	0.718119	0.754012	0.710415	0.706994	0.720756	0.775970	0.748157	0.753300	0.717546	0.718109	0.687405	0.750914	0.741141
	20	0.774954	0.764940	0.763534	0.744373	0.746163	0.720756	0.764418	0.737100	0.671680	0.714377	0.755823	0.322399	0.757787	0.764019
40	25	0.742716	0.753882	0.761677	0.739303	0.595732	0.755823	0.763470	0.751044	0.675647	0.665571	0.764020	0.748171	0.693679	0.777896
	30	0.756474	0.756211	0.734856	0.702923	0.764447	0.764020	0.737629	0.742906	0.687683	0.754066	0.740089	0.762255	0.654111	0.700626
	35	0.747396	0.750914	0.741141	0.758201	0.763139	0.700626	0.764274	0.701899	0.777775	0.797605	0.722989	0.752929	0.707293	0.736921
	40	0.759631	0.757787	0.764019	0.736438	0.744943	0.736921	0.754135	0.761514	0.322814	0.602617	0.761779	0.756135	0.736474	0.750911
	45	0.750198	0.760208	0.762853	0.738613	0.754243	0.750911	0.723198	0.678108	0.622531	0.761030	0.757960	0.712695	0.703071	0.749426

Appendix H

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - RMSE

Testing value

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
5	5	0.288038	0.288793	0.289039	0.286974	0.322232	0.289096	0.286921	0.288998	0.322814	0.289017	0.289367	0.288926	0.289542	0.297504
	10	0.288494	0.288579	0.288725	0.288264	0.285723	0.293982	0.287903	0.289174	0.288744	0.289134	0.286092	0.289007	0.288979	0.288791
	15	0.288985	0.288494	0.288753	0.286024	0.296632	0.288908	0.288677	0.289160	0.288815	0.288307	0.289023	0.288842	0.288307	0.289023
	20	0.288985	0.288739	0.288625	0.289257	0.286198	0.288846	0.288914	0.288787	0.288917	0.289263	0.289183	0.300569	0.289263	0.289183
	25	0.288749	0.288783	0.288991	0.289258	0.285562	0.289023	0.288846	0.288793	0.288909	0.289005	0.289339	0.288838	0.289127	0.286894
	30	0.289050	0.288994	0.289006	0.287200	0.289983	0.289309	0.289023	0.322547	0.289256	0.288817	**0.261945**	0.932413	0.289086	0.289103
	35	0.288467	0.288707	0.289069	0.286068	0.289455	0.288754	0.289309	0.285417	0.289348	0.289256	0.288817	0.287454	0.288913	0.289121
	40	0.288680	0.288784	0.288667	0.288861	0.289313	0.288842	0.288754	0.289017	0.289367	0.288756	0.288778	0.288823	0.289096	0.286921
	45	0.289083	0.288954	0.288858	0.287798	0.287951	0.288839	0.289275	0.293230	0.300569	0.289263	0.289183	0.288784	0.293982	0.287903
10	5	0.322953	0.288882	0.288758	0.307061	0.303513	0.322547	0.289256	0.288817	0.262842	0.289005	0.289339	0.288838	0.322181	0.769422
	10	0.288095	0.288430	0.288914	0.288615	0.289237	0.288998	0.322814	0.288926	0.289542	0.297504	0.288919	0.285417	0.289348	0.288914
	15	0.288932	0.288785	0.288846	0.298275	0.285338	0.289174	0.288744	0.289007	0.288979	0.288791	0.288556	0.289017	0.289367	0.288846
	20	0.289161	0.288668	0.289023	0.286791	0.288049	0.289160	0.288815	0.288739	0.288625	0.289257	0.289033	0.289134	0.286092	0.289023
	25	0.289029	0.288835	0.289309	0.287963	0.284167	0.288787	0.288917	0.288783	0.288991	0.289258	0.289067	0.288307	0.289023	0.289309
	30	0.288886	0.288852	0.288754	0.289870	0.289143	0.288793	0.288909	0.289096	0.286921	0.288710	0.288940	0.289263	0.289183	0.288754

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	35	0.288745	0.289282	0.288842	0.287698	0.287661	0.288787	0.289096	0.293982	0.287903	0.288778	0.288840	0.289127	0.286894	0.288842
	40	0.288787	0.289018	0.288835	0.289034	0.286267	0.288793	0.293982	0.288908	0.288677	0.289214	0.288800	0.289086	0.289103	0.288459
	45	0.288779	0.288715	0.288726	0.289086	0.289008	0.288838	0.288839	0.289275	0.293230	0.300569	0.289263	0.289183	0.289121	0.287451
15	5	0.322547	0.289256	0.288817	**0.262842**	0.286629	0.288739	0.288625	0.289257	0.289005	0.289339	0.288838	0.322181	0.289143	0.288793
	10	0.288754	0.288756	0.288778	0.300857	0.288514	0.288783	0.288991	0.289258	0.322547	0.289256	0.288817	0.313052	0.287661	0.288787
	15	**0.287707**	0.288923	0.288823	0.285762	0.286841	0.289160	0.288998	0.322814	0.288914	0.289096	0.286921	0.288926	0.289542	0.297504
	20	0.288710	0.288940	0.288823	0.289096	0.286921	0.288787	0.289174	0.288744	0.288846	0.293982	0.287903	0.289007	0.288979	0.288791
	25	0.288778	0.288840	0.288784	0.293982	0.287903	0.288793	0.289160	0.288815	0.289023	0.288908	0.288677	0.288745	0.289282	0.288842
	30	0.289214	0.288800	0.288852	0.288908	0.288677	0.289012	0.288787	0.288917	0.289309	0.285417	0.289348	0.288787	0.289018	0.288835
	35	0.288903	0.288793	0.288895	0.288963	0.289418	0.288839	0.288793	0.288909	0.288754	0.289017	0.289367	0.289062	0.289033	0.288864
	40	0.288808	0.288841	0.288860	0.288737	0.287293	0.288919	0.289040	0.296315	0.288842	0.289134	0.286092	0.289062	0.289067	0.289007
	45	0.288882	0.288831	0.289539	0.289463	0.289112	0.288839	0.289275	0.293230	0.300569	0.289263	0.289183	0.288864	0.288764	0.288883
20	5	0.288771	0.288998	0.322814	0.305463	0.302876	0.289174	0.288739	0.288625	0.289257	0.288998	0.322814	0.288733	0.288733	0.289048
	10	0.290313	0.289174	0.288744	0.289037	0.289357	0.289160	0.288783	0.288991	0.289258	0.289174	0.288744	0.288864	0.288864	0.288213
	15	0.289137	0.289160	0.288815	0.288780	0.289747	0.288787	0.288914	0.289096	0.286921	0.289160	0.288815	0.289007	0.289007	0.288692
	20	0.289005	0.288787	0.288917	0.287300	0.285010	0.288793	0.288846	0.293982	0.287903	0.288787	0.288917	0.288883	0.289012	0.288835
	25	0.289268	0.288793	0.288909	0.285417	0.289348	0.289012	0.289023	0.288908	0.288677	0.288793	0.288909	0.288952	0.288839	0.288954
	30	0.288989	0.289012	0.288835	0.289017	0.289367	0.288840	0.289309	0.289005	0.289339	0.288838	0.322181	0.288942	0.288931	0.288873
	35	0.288922	0.288839	0.288954	0.289134	0.286092	0.288800	0.288754	0.288926	0.289542	0.297504	0.286841	0.289160	0.288998	0.322814
	40	0.288866	0.288862	0.288903	0.285602	0.289114	0.288793	0.288842	0.289007	0.288979	0.288791	0.286921	0.288787	0.289174	0.288744
	45	0.289045	0.288865	0.289257	0.289298	0.289416	0.288841	0.322547	0.289256	0.288817	**0.254032**	0.287903	0.288793	0.289160	0.288815
25	5	0.289005	0.289339	0.288838	0.322181	0.293043	**0.284321**	0.286921	0.289747	0.288839	0.289275	0.293230	0.300569	0.289263	0.289183

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	10	0.288917	0.288919	0.289040	0.296315	0.286331	0.293982	0.287903	**0.285010**	0.289217	0.289132	0.288890	0.292332	0.288688	0.288619
	15	0.288832	0.288556	0.288733	0.289048	0.288134	0.288908	0.288677	0.289348	0.288914	0.288979	0.288791	0.288886	0.288750	0.288307
	20	0.289062	0.289033	0.288864	0.288213	0.290233	0.289137	0.288739	0.288625	0.288846	0.289208	0.288925	0.289566	0.288827	0.289263
	25	0.289062	0.289067	0.289007	0.288692	0.289305	0.289005	0.288783	0.288991	0.289023	0.285417	0.289348	0.288417	0.289156	0.290398
	30	0.288864	0.288764	0.288883	0.287496	0.289047	0.289268	0.288998	0.322814	0.289309	0.289017	0.289367	0.289132	0.288890	0.292332
	35	0.289108	0.288983	0.288952	0.290264	0.288984	0.289007	0.289174	0.288744	0.288754	0.289134	0.286092	0.289217	0.289132	0.288890
	40	0.288942	0.288931	0.288873	0.289145	0.289007	0.289012	0.289160	0.288815	0.288842	0.287321	0.289566	0.288587	0.288926	0.289542
	45	0.289003	0.288904	0.289505	0.289204	0.289032	0.288839	0.288787	0.288917	**0.254210**	0.288979	0.322547	0.289256	0.288817	0.262842
30	5	0.288841	0.288875	0.322465	0.285970	0.287657	0.289096	0.288793	0.288909	0.288435	0.285417	0.289348	0.293982	0.287903	0.288787
	10	0.289001	**0.288417**	0.289156	0.290398	0.301502	0.288839	0.289275	0.293230	0.300569	0.289263	0.289183	0.288908	0.288677	0.288793
	15	0.289217	0.289132	0.288890	0.292332	0.289927	0.288432	**0.285432**	0.288925	0.294313	0.289134	0.286092	0.288846	0.288978	0.289377
	20	0.288587	0.288926	0.289542	0.297504	0.288749	0.288998	0.322814	0.288867	0.288739	0.288625	0.289257	0.289023	0.289096	0.286921
	25	0.288758	0.289007	0.288979	0.288791	0.288886	0.289174	0.288744	0.288937	0.288783	0.288991	0.289258	0.289309	0.293982	0.287903
	30	0.289094	0.289063	0.289208	0.288925	0.289566	0.289160	0.288815	0.288983	0.312696	0.288965	0.288911	0.288754	0.288908	0.288677
	35	0.288926	0.289019	0.289118	0.289161	0.289342	0.288787	0.288917	0.288993	0.322547	0.289256	0.288817	0.253433	0.288543	**0.211321**
	40	0.288976	0.288978	0.289377	0.288566	0.289167	0.288793	0.288909	0.289681	0.322399	0.288916	0.288841	0.288875	0.288815	0.289023
	45	0.289017	0.289067	0.288980	0.289019	0.289129	0.289005	0.289339	0.288838	0.322181	0.282134	0.289001	0.288417	0.288917	0.289309
35	5	0.288931	0.289037	0.322881	0.301530	0.315615	0.288926	0.289542	0.297504	0.288839	0.289275	0.289217	0.289132	0.289263	0.289183
	10	0.288973	0.288688	0.288619	0.289191	0.294883	0.289007	0.288979	0.288791	0.322547	0.289256	0.288817	**0.244938**	0.287542	0.300451
	15	0.288684	0.288750	**0.288307**	0.289023	0.289332	0.288917	0.289309	0.285417	0.289348	0.289005	0.289339	0.288838	0.322181	0.265425
	20	0.288925	0.288827	0.289263	0.289183	0.289292	0.288909	0.288754	0.289017	0.289367	0.289023	0.288625	0.289257	0.288998	0.322814
	25	0.288867	0.288931	0.289127	0.286894	0.288954	0.296315	0.288842	0.289134	0.286092	0.289309	0.288991	0.289258	0.289174	0.288744

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	30	0.288937	0.288976	0.289086	0.289103	0.289138	0.293230	0.300569	0.289275	0.293230	0.288754	0.289096	0.286921	0.289160	0.288815
	35	0.288983	0.289079	0.288913	0.289121	0.288863	0.289161	0.289342	0.289087	0.293102	0.288842	0.293982	0.287903	0.288787	0.288917
	40	0.288993	0.289030	0.289541	0.289075	0.289710	0.288566	0.289167	0.289002	0.289414	0.288760	0.288908	0.288677	0.288793	0.288909
	45	0.288948	0.288942	0.288927	0.288858	0.286761	0.289019	0.289129	0.289008	0.289653	0.289096	0.286921	0.289160	0.288815	0.289007
40	5	0.289681	0.322399	0.288916	0.291887	0.319410	0.288307	0.289023	0.289309	0.285417	0.289348	0.288998	0.322814	0.289256	0.288817
	10	0.288977	0.288839	0.289275	0.293230	0.300569	0.289263	0.289183	0.288754	0.289017	0.289367	0.289174	0.288744	0.288756	0.288778
	15	0.289015	0.289070	0.289087	0.293102	0.287672	0.289127	0.286894	0.289377	0.288566	0.289167	0.289160	0.288815	0.288914	0.289309
	20	0.288887	0.288948	0.289002	0.289414	0.289196	0.289086	0.289103	0.289005	0.289339	0.288838	0.322181	0.288917	0.288846	0.288754
	25	0.288950	0.288913	0.289008	0.289653	**0.274904**	0.288307	0.289023	0.322547	0.289256	0.289096	0.288793	0.288909	0.289023	0.262842
	30	0.289062	0.289036	0.289258	0.289740	0.289001	0.289263	0.289183	0.285417	0.289348	0.293982	0.287903	0.288625	0.289309	0.288875
	35	0.289059	0.289090	0.289000	0.289363	0.289178	0.289127	0.286894	0.289017	0.289367	0.288908	0.288677	0.288991	0.288754	0.276743
	40	0.289003	0.289066	0.288978	0.288977	0.289136	0.289086	0.289103	0.289134	0.286092	0.288926	0.289542	0.297504	0.288842	0.288975
	45	0.289047	0.288911	0.289051	0.289224	0.289064	0.288913	0.289121	0.288946	0.279654	0.289007	0.288979	0.288791	**0.265741**	0.294607

Appendix I

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - GRMSE

Training value

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
5	5	1.277809	1.278781	1.279100	1.276531	1.341334	1.278921	1.279183	1.278509	1.278803	1.274975	1.276296	1.290338	1.277295	1.277873
	10	1.278392	1.278515	1.278691	1.278122	1.274851	1.278981	1.278916	1.278111	1.277456	1.279165	1.276380	1.275511	1.279253	1.277824
	15	1.279033	1.278395	1.278728	1.275243	1.290338	1.278995	1.279020	1.279347	1.278756	1.285775	1.277639	1.274671	1.261456	1.272884
	20	1.278603	1.278711	1.278568	1.279411	1.275511	1.278727	1.279093	1.279168	1.279432	1.275573	1.277824	1.280365	1.278997	1.279232
	25	1.278724	1.278769	1.279037	1.279430	1.274671	1.278931	1.279036	1.279108	1.278892	1.279162	1.272884	1.279661	1.279227	1.277377
	30	1.279115	1.279042	1.279057	1.276751	1.280365	1.277873	1.278302	1.278931	1.302920	1.298616	1.279232	1.279460	######	1.275554
	35	1.278359	1.278668	1.279148	1.275341	1.279661	1.278955	1.278763	1.278842	1.279076	1.279532	1.277377	1.278911	1.278913	1.279072
	40	1.278634	1.278769	1.278618	1.278878	1.279460	1.277824	1.290338	1.278503	1.278743	1.280012	1.275554	1.278801	1.278443	1.278672
	45	1.279170	1.278989	1.278865	1.277512	1.277709	1.272884	1.275511	1.274523	1.278120	1.278815	1.279742	1.279643	1.273453	1.277543
10	5	**1.207665**	1.278888	1.278727	1.307155	1.303475	1.278815	1.279742	1.279643	1.276456	1.279183	1.278509	1.278727	1.277824	1.290338
	10	1.277873	1.278302	1.278931	1.278557	1.279360	1.290338	1.279115	1.279042	1.279057	1.278916	1.278111	1.278931	1.272884	1.275511
	15	1.278955	1.278763	1.278842	1.292172	1.274445	1.275511	1.278359	1.278668	1.279148	1.279020	1.279347	1.277873	1.278302	1.278931
	20	1.279258	1.278612	1.279072	1.276234	1.277824	1.274671	1.207665	1.278888	1.278727	1.279093	1.279168	1.278955	1.278763	1.278842
	25	1.279080	1.278827	1.279445	1.277714	1.272884	1.280365	1.277873	1.278302	1.278931	1.279036	1.279108	1.278892	1.279162	1.277295
	30	1.278893	1.278852	1.278725	1.280186	1.279232	1.279661	1.302920	1.298616	1.302920	1.298616	1.278911	1.278913	1.279072	1.279253

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	35	1.278710	1.279413	1.278837	1.277419	1.277377	1.279460	1.279076	1.279532	1.279076	1.279532	1.277113	1.279079	1.278672	1.261456
	40	1.278764	1.279066	1.278828	1.279090	1.275554	1.278435	1.278743	1.280012	1.278743	1.280012	1.280673	1.279001	1.294532	1.278997
	45	1.278754	1.278671	1.278686	1.279159	1.279061	1.278634	1.278769	1.278618	1.278878	1.279460	1.279209	1.279032	1.275630	1.279227
	5	1.222416	1.279373	1.278796	**1.248533**	1.276019	1.302920	1.298616	1.302920	1.298616	1.278911	1.278913	1.279072	1.277113	1.279079
	10	1.278715	1.278717	1.278752	1.296143	1.278403	1.279076	1.279532	1.279076	1.279532	1.278801	1.278443	1.278672	1.280673	1.279001
	15	1.278735	1.278934	1.278803	1.274975	1.276296	1.278743	1.280012	1.278743	1.280012	1.278945	1.278727	1.277295	1.279209	1.279032
	20	1.278658	1.278957	**1.277456**	1.279165	1.276380	1.277824	1.290338	1.279108	1.278892	1.279162	1.278931	1.279253	1.278509	1.279183
15	25	1.278745	1.278825	1.278756	1.285775	1.277639	1.272884	1.275511	1.279115	1.279042	1.279057	1.277377	1.261456	1.278111	1.278916
	30	1.279316	1.278774	1.278842	1.278917	1.278615	1.279232	1.274671	1.278359	1.278668	1.279148	1.275554	1.278997	1.279347	1.279020
	35	1.278908	1.278765	1.278898	1.278986	1.279604	1.277377	1.280365	**1.235035**	1.278888	1.278727	1.279061	1.279227	1.279168	1.279093
	40	1.278785	1.278826	1.278851	1.278693	1.276861	1.275554	1.279661	1.277873	1.278302	1.278931	1.276019	1.277824	1.290338	1.279036
	45	1.278881	1.278815	1.279742	1.279643	1.279196	1.278509	1.279460	1.242945	1.276053	1.277824	1.290338	1.272884	1.275511	1.285353
	5	1.277356	1.279024	1.342153	1.302920	1.298616	1.278111	1.277873	1.278302	1.278931	1.272884	1.275511	1.278815	1.279742	1.279643
	10	1.280909	1.279264	1.278696	1.279076	1.279532	1.279347	1.278955	1.278763	1.278842	1.290338	**1.235336**	1.278888	1.278727	1.277295
	15	1.279228	1.279238	1.278786	1.278743	1.280012	1.279168	1.279108	1.278892	1.279162	1.275511	1.277873	1.278302	1.278931	1.279253
	20	1.279032	1.278751	1.278919	1.276851	1.273951	1.279183	1.279115	1.279042	1.279057	1.274671	1.277824	1.277113	1.279079	**1.030024**
20	25	1.279380	1.278757	1.278908	1.274472	1.279490	1.278916	1.278359	1.278668	1.279148	1.280365	1.272884	1.280673	1.279001	1.278997
	30	1.279011	1.279044	1.278811	1.279048	1.279520	1.279020	1.278911	1.278913	1.279072	1.279661	1.279232	1.279209	1.279032	1.279227
	35	1.278925	1.278817	1.278942	1.279214	1.275359	1.279093	1.278801	1.278443	1.278672	1.279460	1.277377	1.278921	1.278443	1.278672
	40	1.278852	1.278847	1.278901	1.274682	1.279181	1.279036	1.278420	1.278727	1.302920	1.298616	1.275554	1.278981	1.279063	1.278842
	45	1.279085	1.278850	1.279365	1.279421	1.279579	1.278696	1.279076	1.278931	1.279076	1.279532	1.277542	1.278995	1.279127	1.279026
25	5	1.279026	1.279468	1.278816	1.341187	1.284736	1.278727	1.276035	1.274630	1.278743	1.280012	1.290338	1.277873	1.278302	1.278931

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	10	1.278911	1.278913	1.279072	1.289262	1.275608	1.278931	1.278560	1.277295	1.279183	1.272884	1.275511	1.278955	1.278763	1.278842
	15	1.278801	1.278443	1.278672	1.279084	1.277916	1.277824	1.290338	1.279253	1.278916	1.279232	1.274671	**1.209454**	1.278888	1.278727
	20	1.279101	1.279063	1.278842	1.278009	1.280648	1.272884	1.275511	1.261456	1.279020	1.277377	1.280365	1.277873	1.278302	1.278931
	25	1.279101	1.279127	1.279026	1.278623	1.279425	1.279300	1.279183	1.278997	1.279093	1.275554	1.279661	1.279108	1.278892	1.279162
	30	1.278842	1.278712	1.278867	1.277113	1.279079	1.278478	1.278916	1.279227	1.279036	1.277650	1.279460	1.278509	1.276563	1.277863
	35	1.279160	1.278996	1.278957	1.280673	1.279001	1.278695	1.279020	1.278815	1.279742	1.279643	1.278921	1.278111	1.302920	1.298616
	40	1.278943	1.278929	1.278853	1.279209	1.279032	1.279135	1.279093	1.236435	1.278888	1.278727	1.278981	1.279347	1.279076	1.279532
	45	1.279023	1.278894	1.279684	1.279288	1.279068	1.278914	1.279036	1.277873	1.278302	1.278931	1.278995	1.279168	1.278743	1.280012
30	5	1.278804	1.278848	1.341594	1.275145	1.277409	1.278980	1.278983	1.274635	1.277824	1.277873	1.278302	1.278931	1.279101	1.279063
	10	1.279016	**1.278256**	1.279215	1.280867	1.297128	1.278911	1.278913	1.279072	**1.272884**	1.278955	1.278763	1.278842	1.279101	1.279127
	15	1.279300	1.279183	1.278868	1.283547	1.280236	1.278801	1.278443	1.278672	1.279232	1.278931	1.278560	1.277295	1.279183	1.272884
	20	1.278478	1.278916	1.279721	1.290539	1.278691	1.279115	1.279042	1.279057	1.277377	1.279108	1.278892	1.279162	1.278602	1.278509
	25	1.278695	1.279020	1.278984	1.278743	1.278873	1.278359	1.278668	1.279148	1.275554	1.277295	1.302920	1.298616	1.278679	1.278111
	30	1.279135	1.279093	1.279289	1.278916	1.279754	**1.253568**	1.278888	1.278727	1.278509	1.279253	1.279076	1.279532	1.278778	1.279347
	35	1.278914	1.279036	1.279167	1.279226	1.279467	1.277873	1.278302	1.278931	1.278111	1.261456	1.278743	1.280012	1.286567	1.278921
	40	1.278980	1.278983	1.279504	1.278451	1.279229	1.278815	1.279742	1.279643	1.279347	1.278997	1.278727	1.277824	1.290338	1.278981
	45	1.279033	1.279099	1.278986	1.279036	1.279183	1.279742	1.279643	1.278921	1.279168	1.279227	1.278931	1.272884	1.275511	1.278995
35	5	1.278914	1.279051	1.342198	1.296892	1.323468	1.277853	1.278911	1.278913	1.279072	1.279115	1.279042	1.279057	1.279183	1.277824
	10	1.278968	1.278602	1.278509	1.279254	1.286953	1.276463	1.278801	1.278443	1.278672	1.278359	1.278668	1.279148	1.278916	1.272884
	15	1.278593	1.278679	1.278111	1.279034	1.279435	1.279108	1.278892	1.279162	1.278815	1.279742	1.279643	1.290338	1.279020	1.279232
	20	1.278906	1.278778	1.279347	1.279243	1.279386	1.277295	**1.204543**	1.278888	1.278727	1.302920	1.298616	1.275511	1.279093	1.277377
	25	1.278831	1.278914	1.279168	1.276356	1.278944	1.279253	1.277873	1.278302	1.278931	1.279076	1.279532	1.274671	1.279036	1.275554

Input Lags	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	30	1.278921	1.278972	1.279116	1.279137	1.279186	1.261456	1.277353	1.298616	1.275511	1.278743	1.280012	1.280365	1.278509	1.278727
	35	1.278981	1.279108	1.278892	1.279162	1.278828	1.278997	1.279721	1.277873	1.278302	1.278931	1.278921	1.279661	1.278111	1.278931
	40	1.278995	1.279043	1.279714	1.279104	1.279934	1.279227	1.278984	1.278955	1.278763	1.278842	1.278981	1.279460	1.279347	1.263533
	45	1.278936	1.278927	1.278908	1.278818	1.276134	1.278589	1.279289	1.277824	1.290338	1.277758	1.278995	1.286453	1.279168	1.274565
40	5	1.279888	1.341460	1.278887	1.282920	1.334625	1.277824	1.274644	1.272884	1.275511	1.278911	1.278913	1.279072	1.275035	1.273053
	10	1.278965	1.278785	1.279356	1.284714	1.295953	1.272884	1.277824	1.290338	1.275533	1.278801	1.278443	1.278672	1.277645	1.274400
	15	1.279015	1.279086	1.279109	1.284559	1.277295	1.279232	1.272884	1.275511	1.279108	1.278892	1.279162	1.279115	1.279042	1.279057
	20	1.278849	1.278927	1.278998	1.279543	1.279253	1.277377	1.278302	1.279183	1.279086	**1.211343**	1.278888	1.278727	1.290338	1.279148
	25	1.278930	1.278884	1.279006	1.279849	**1.261456**	1.275554	1.278763	1.278916	1.278927	1.277873	1.278302	1.278931	1.275511	1.277295
	30	1.279077	1.279042	1.279334	1.279965	1.278997	1.278509	1.278727	1.279020	1.278884	1.278815	1.279742	1.279643	1.274671	1.279253
	35	1.279074	1.279112	1.278997	1.279469	1.279227	1.278111	1.278931	1.279093	1.279042	1.278921	1.302920	1.298616	1.280365	1.261456
	40	1.278999	1.279081	1.278967	1.278972	1.279174	1.279347	1.277873	1.278302	1.278931	1.278981	1.279076	1.279532	1.279661	1.278997
	45	1.279057	1.278880	1.279062	1.279290	1.279087	1.279168	1.278955	1.278763	1.278842	1.278995	1.278743	1.280012	1.279460	1.279227

Appendix J

Performance of the Enhanced Backpropagation Neural Network (BPNN) Model on Simulated Dataset II - GRMSE

Testing value

Input Lag	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
5	5	1.753120	1.751782	1.578203	1.512070	1.222156	1.744544	1.590608	1.618171	1.754512	1.750648	1.750782	1.690646	1.708648	1.583944
	10	1.739693	1.585780	1.754303	1.692375	1.585417	1.613666	1.650819	1.587109	1.722891	1.697171	1.705152	1.734404	1.732219	1.661379
	15	1.653088	1.648694	1.641296	1.582979	1.566051	1.676204	1.719556	1.704096	1.701879	1.624520	1.715581	1.723140	1.729678	1.748532
	20	1.342395	1.676456	1.537831	1.649273	1.531838	1.724773	1.723400	1.704347	1.548775	1.696412	1.754910	1.715466	1.706092	1.714876
	25	1.657602	1.591055	1.643938	1.617569	1.559640	1.725103	1.713860	1.751833	1.640667	1.602836	1.714876	1.709911	1.741020	1.750837
	30	1.713571	1.640111	1.596789	1.579421	1.604079	1.727451	1.685937	1.735592	1.583944	1.730716	1.750837	1.703059	1.709771	1.713571
	35	1.621557	1.694729	1.659768	1.550937	1.595059	1.708900	1.703059	1.708900	1.678171	1.737439	1.690646	1.708648	1.709911	1.621557
	40	1.655911	1.736459	1.677415	1.678133	1.706655	1.732364	1.306365	1.749885	1.756540	1.754053	1.720954	1.732522	1.703059	1.655911
	45	1.604291	1.698389	1.632100	1.612234	1.682927	1.742399	1.548775	1.652582	1.653386	1.720111	1.678329	1.723380	1.703652	1.708590
10	5	1.342395	1.749885	1.754315	1.381371	1.383413	1.650819	1.653386	1.720111	1.678329	1.583944	1.706092	1.676204	1.719556	1.704096
	10	1.548775	1.652582	1.748953	1.659441	1.736902	1.754512	1.585780	1.754303	1.692375	1.661379	1.741020	1.724773	1.723400	1.704347
	15	1.640667	1.660310	1.667643	1.588746	1.545685	1.722891	1.648694	1.641296	1.582979	1.748532	1.709771	1.725103	1.713860	1.751833
	20	**1.341769**	1.565522	1.744544	1.590608	1.618171	1.754512	1.750648	1.617569	1.559640	1.714876	1.709911	1.727451	1.685937	1.735592
	25	1.661379	1.711054	1.613666	1.650819	1.587109	1.722891	1.697171	1.579421	1.604079	1.750837	1.703059	1.708900	1.755772	1.719740
	30	1.748532	1.640789	1.703820	1.696412	1.724156	1.750782	1.624520	1.324194	1.595059	1.690646	1.708648	1.732364	1.679910	1.728447

Input Lag	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	35	1.714876	1.689388	1.690105	1.602836	1.592220	1.705152	1.639393	1.678133	1.706655	1.734404	1.732219	1.742399	1.730358	1.716971
	40	1.750837	1.708716	1.682094	1.730716	1.614880	1.715581	1.744544	1.590608	1.618171	1.723140	1.729678	1.678171	1.737439	1.665384
	45	1.748008	1.724284	1.723380	1.703652	1.708590	1.754910	1.613666	1.650819	1.587109	1.652035	1.645939	1.756540	1.754053	1.720954
	5	1.354290	1.740027	1.730774	1.220721	1.612468	1.583944	1.713860	1.751833	1.585780	1.754303	1.692375	1.653386	1.720111	1.678329
	10	1.645503	1.682967	1.696802	1.553488	1.766886	1.661379	1.685937	1.735592	1.648694	1.641296	1.582979	1.698856	1.725103	1.750782
	15	1.667158	1.751042	1.682500	1.483061	1.607155	1.748532	1.719078	1.744544	1.590608	1.618171	1.617569	1.559640	1.727451	1.705152
	20	1.667158	1.796540	1.706092	1.669819	1.613037	1.714876	1.728743	1.613666	1.650819	1.587109	1.579421	1.604079	1.708900	1.715581
15	25	1.755772	1.719740	1.706092	1.657660	1.630945	1.750837	1.690646	1.708648	1.754512	1.750648	1.542137	1.595059	1.732364	1.754910
	30	1.679910	1.728447	1.741020	1.710802	1.779804	1.754512	1.734404	1.732219	1.722891	1.697171	1.678133	1.706655	1.742399	1.754512
	35	1.730358	1.716971	1.709771	1.751113	1.686343	1.722891	1.723140	1.729678	1.701879	1.624520	1.676204	1.719556	1.704096	1.722891
	40	1.738050	1.753004	1.709911	1.718676	1.615773	1.750782	1.678171	1.737439	1.665384	1.682500	1.724773	1.723400	1.704347	1.701879
	45	1.683869	1.732050	1.703059	1.652916	1.685706	1.705152	1.756540	1.754053	1.720954	1.706092	1.747279	1.709616	1.689858	1.706092
	5	1.739496	1.750643	1.222641	1.513269	1.531187	1.690646	1.708648	1.653386	1.720111	1.678329	1.583944	1.713860	1.751833	1.741020
	10	1.642547	1.617470	1.542056	1.749791	1.622847	1.734404	1.732219	1.727451	1.617569	1.559640	1.661379	1.685937	1.735592	1.709771
	15	1.719078	1.667584	1.755888	1.724874	1.731458	1.723140	1.729678	1.708900	1.579421	1.604079	1.748532	1.725195	1.741020	1.725103
	20	1.728743	1.631100	1.651640	1.641700	1.587105	1.585780	1.754303	1.692375	1.550937	1.595059	1.714876	1.713525	1.709771	1.727451
20	25	1.697833	1.715068	1.732685	1.577792	1.732614	1.706092	1.641296	1.582979	1.678133	1.706655	1.750837	1.704096	1.709911	1.708900
	30	1.685206	1.711747	1.754512	1.750648	1.677941	1.741020	1.750782	1.714876	1.744544	1.590608	1.618171	1.704347	1.703059	1.732364
	35	1.705060	1.724535	1.722891	1.697171	1.537066	1.709771	1.705152	1.750837	1.676204	1.719556	1.704096	1.702462	1.719639	1.742399
	40	1.721698	1.737311	1.701879	1.624520	1.689407	1.709911	1.715581	1.754512	1.724773	1.723400	1.704347	1.678171	1.737439	1.665384
	45	1.739538	1.747279	1.709616	1.689858	1.690614	1.703059	1.754910	1.722891	1.701536	1.740324	1.669540	1.756540	1.754053	1.720954
25	5	1.751072	1.741203	1.707887	1.222083	1.609097	1.711747	1.754512	1.585780	1.754303	1.692375	1.754512	1.750648	1.583944	1.706092

Input Lag	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	10	1.744388	1.669676	1.654721	1.614027	1.653145	1.724535	1.722891	1.648694	1.641296	1.582979	1.722891	1.697171	1.661379	1.741020
	15	1.690828	1.779961	1.757605	1.714700	1.817523	1.737311	1.701879	1.706092	1.725103	1.744544	1.701879	1.624520	1.748532	1.709771
	20	1.690646	1.651355	1.755472	1.777042	1.720710	1.617569	1.559640	1.741020	1.727451	1.613666	1.650819	1.587109	1.714876	1.728743
	25	1.690646	1.708648	1.737758	1.735634	1.699588	1.579421	1.604079	1.709771	1.708900	1.732219	1.648694	1.750782	1.750837	1.697833
	30	1.734404	1.732219	1.686271	1.625939	1.752844	1.550937	1.595059	1.709911	1.732364	1.729678	1.744544	1.705152	1.740009	1.715581
	35	1.723140	1.729678	1.686892	1.727232	1.726699	1.678133	1.706655	1.703059	1.678171	1.737439	1.665384	1.715581	1.736015	1.754910
	40	1.727917	1.715231	1.753639	1.693570	1.667230	1.676204	1.719556	1.704096	1.756540	1.754053	1.720954	1.754910	1.751833	1.742399
	45	1.723950	1.742231	1.702462	1.719639	1.721137	1.724773	1.723400	1.704347	1.653386	1.720111	1.678329	1.685937	1.735592	1.750782
30	5	1.672257	1.756066	1.222323	1.585465	1.628772	1.725103	1.713860	1.751833	1.706092	1.585780	1.754303	1.692375	1.720972	1.583944
	10	1.599652	1.627413	1.662973	1.720972	1.539405	1.727451	1.685937	1.735592	1.741020	1.648694	1.641296	1.582979	1.676076	1.661379
	15	1.737581	1.685719	1.769415	1.676076	1.725195	1.708900	1.737439	1.754911	1.709771	1.756540	1.754512	1.750648	1.617569	1.748532
	20	1.816868	1.701536	1.740324	1.669540	1.713525	1.732364	1.754053	1.747834	1.709911	1.737439	1.722891	1.697171	1.579421	1.714876
	25	1.773451	1.731393	1.676204	1.719556	1.704096	1.742399	1.750782	1.774484	1.703059	1.754053	1.701879	1.624520	1.550937	1.750837
	30	1.715045	1.728581	1.724773	1.723400	1.704347	1.750782	1.705152	1.718225	1.729678	1.690646	1.708648	1.705334	1.678133	1.706655
	35	1.723566	1.718532	1.686424	1.718572	1.702996	1.705152	1.715581	1.741699	1.559640	1.734404	1.732219	1.744544	1.590608	1.618171
	40	1.728179	1.674703	1.741612	1.680900	1.740009	1.715581	1.678171	1.737439	1.665384	1.723140	1.729678	1.613666	1.650819	1.587109
	45	1.730170	1.720567	1.751062	1.736884	1.736015	1.754910	1.756540	1.754053	1.720954	1.653386	1.720111	1.678329	1.740009	1.715581
35	5	1.726022	1.752897	1.215094	1.562557	1.292455	1.678171	1.737439	1.665384	1.690646	1.708648	1.585780	1.754303	1.692375	1.750782
	10	1.754069	1.581568	1.653386	1.720111	1.678329	1.756540	1.754053	1.720954	1.734404	1.732219	1.648694	1.641296	1.582979	1.705152
	15	1.760610	1.759410	1.752134	1.729691	1.749757	1.719556	1.583944	1.706092	1.723140	1.729678	1.744544	1.590608	1.618171	1.715581
	20	1.751360	1.725103	1.737323	1.749504	1.735460	1.723400	1.661379	1.741020	1.617569	1.559640	1.613666	1.650819	1.587109	1.754910
	25	1.776378	1.727451	1.730488	1.592419	1.754299	1.718572	1.748532	1.709771	1.579421	1.604079	1.725103	1.750782	1.754512	1.750648

Input Lag	Hidden Nodes	Outliers													
		0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%
	30	1.755652	1.708900	1.713860	1.751833	1.699001	1.728179	1.714876	1.709911	1.550937	1.595059	1.727451	1.705152	1.722891	1.697171
	35	1.753978	1.732364	1.685937	1.735592	1.706930	1.730170	1.750837	1.703059	1.678133	1.706655	1.708900	1.715581	1.701879	1.624520
	40	1.732822	1.742399	1.706590	1.721531	1.717737	1.713860	1.751833	1.676204	1.719556	1.704096	1.732364	1.754910	1.737439	1.665384
	45	1.743368	1.755451	1.751705	1.749545	1.657548	1.685937	1.735592	1.724773	1.723400	1.704347	1.742399	1.653386	1.720111	1.678329
40	5	1.740551	1.222251	1.763151	1.684540	1.226402	1.706092	1.725103	1.754299	1.718572	1.754512	1.750648	1.678133	1.706655	1.708900
	10	1.754911	1.727425	1.706672	1.678479	1.533223	1.741020	1.727451	1.699001	1.728179	1.722891	1.697171	1.585780	1.754303	1.692375
	15	1.747834	1.678171	1.737439	1.665384	1.660183	1.709771	1.708900	1.750782	1.583944	1.701879	1.624520	1.648694	1.641296	1.582979
	20	1.774484	1.756540	1.754053	1.720954	1.723989	1.709911	1.732364	1.705152	1.661379	1.700423	1.617569	**1.559433**	1.690646	1.708648
	25	1.718225	1.737204	1.750782	1.712469	1.503197	1.703059	1.742399	1.715581	1.748532	1.590608	1.618171	1.604079	1.734404	1.732219
	30	1.741699	1.741241	1.705152	1.654488	1.755666	1.754512	1.750648	1.754910	1.714876	1.650819	1.587109	1.595059	1.723140	1.729678
	35	1.726120	1.732112	1.715581	1.744698	1.753355	1.722891	1.697171	1.744034	1.750837	1.743453	1.678133	1.706655	1.721436	1.653556
	40	1.747194	1.743979	1.754910	1.707656	1.721925	1.701879	1.624520	1.729342	1.713860	1.751833	1.676204	1.719556	1.704096	1.756754
	45	1.730893	1.748204	1.752852	1.711383	1.719571	1.677415	1.678133	1.706655	1.685937	1.735592	1.724773	1.723400	1.704347	1.704563

Appendix K

Convergence Test with different number of lag and different number of epochs

Epoch	Lag 5	Lag 10	Lag 15	Lag 20	Lag 25	Lag 30	Lag 35	Lag 40
100	0.59314	0.54241	0.54264	0.51930	0.625566	0.53989	0.55713	0.53943
200	0.56208	0.56385	0.55921	0.54721	0.54185	0.55202	0.54060	0.54189
300	0.59944	0.58615	0.56382	0.54313	0.54722	0.54515	0.54901	0.56209
400	0.59704	0.55577	0.56948	0.56862	0.54103	0.55093	0.54917	0.54684
500	0.55671	0.56334	0.55434	0.54998	0.55125	0.562976	0.53673	0.53527
600	0.56356	0.56274	0.56303	0.54542	0.54814	0.57882	0.54887	0.55263
700	0.57670	0.56004	0.55848	0.55019	0.55155	0.55188	0.54556	0.54018
800	0.56386	0.57483	0.55718	0.54572	0.55074	0.54553	0.54248	0.55799

900	0.57840	0.55622	0.55131	0.55513	0.54921	0.54924	0.54948	0.5608
1000	0.56041	0.56274	0.55245	0.54047	0.55475	0.55988	0.53903	0.55376