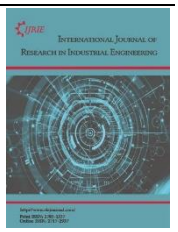
E-ISSN: 2717-2937 | P-ISSN: 2783-1337

International Journal of Research in Industrial Engineering

www.riejournal.com

Int. J. Res. Ind. Eng. Vol. 12, No. 3 (2023) 234–272.



Paper Type: Research Paper

Hybrid Metaheuristic Artificial Neural Networks for Stock Price Prediction Considering Efficient Market Hypothesis

Milad Shahvaroughi Farahani[1,*], Hamed Farrokhi-Asl[2], Saeed Rahimian[1]

[1] Department of Finance, Khatam University, Tehran, Iran; m.shahvaroughi@khatam.ac.ir; s.rahimian@khatam.ac.ir.

[2] Sheldon B. Lubar College of Business, University of Wisconsin-Milwaukee, Milwaukee, WI, USA; farrokh5@uwm.edu.

Citation:

Shahvaroughi Farahani, M., Farrokhi-Asl, H., & Rahimian, S. (2023). Hybrid metaheuristic artificial neural networks for stock price prediction considering efficient market hypothesis. *International journal of research in industrial engineering*, *12*(3), 234-272.

Received: 08/09/2022 Reviewed: 12/10/2022 Revised: 19/11/2022 Accepted: 01/12/2022

Abstract

Investigating stock price trends and determining future stock prices have become focal points for researchers within the finance sector. However, predicting stock price trends is a complex task due to the multitude of influencing factors. Consequently, there has been a growing interest in developing more precise and heuristic models and methods for stock price prediction in recent years. This study aims to assess the effectiveness of technical indicators for stock price prediction, including closing price, lowest price, highest price, and the exponential moving average method. To thoroughly analyze the relationship between these technical indicators and stock prices over predefined time intervals, we employ an Artificial Neural Network (ANN). This ANN is optimized using a combination of Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Harmony Search (HS) algorithms as meta-heuristic techniques for enhancing stock price prediction. The GA is employed for selecting the most suitable optimization indicators. In addition to indicator selection, PSO and HS are utilized to fine-tune the Neural Network (NN), minimizing network errors and optimizing weights and the number of hidden layers simultaneously. We employ eight estimation criteria for error assessment to evaluate the proposed model's performance and select the best model based on error criteria. An innovative aspect of this research involves testing market efficiency and identifying the most significant companies in Iran as the statistical population. The experimental results clearly indicate that a hybrid ANN-HS algorithm outperforms other algorithms regarding stock price prediction accuracy. Finally, we conduct run tests, a non-parametric test, to evaluate the Efficient Market Hypothesis (EMH) in its weak form.

Keywords: Technical indicators, Artificial neural network, Genetic algorithm, Harmony search, Particle swarm optimization algorithm, Efficient market hypothesis.

1 | Introduction

Various tmethods for predicting stock prices include technical analysis, mathematical and statistical approaches, and econometric models. The primary motivation behind the stock market prediction is rooted in the Efficient Market Hypothesis (EMH) and the pursuit of financial gains. EMH is of particular significance because it serves as a foundational concept underpinning the effectiveness of prediction. An efficient market implies that all available information is reflected in stock prices. Conversely, an inefficient market suggests the presence of individuals and entities with privileged access to additional market information. It underscores the idea that information can significantly impact the decisions made by investors, consequently leading to changes in stock prices [1].

Corresponding Author: m.shahvaroughi@khatam.ac.ir

https://doi.org/10.22105/riej.2023.361216.1336

EMH rests on two fundamental assumptions: 1) all investors have equal access to identical information, and 2) there are no superior or premier strategies or systems that can consistently outperform the market [2].

In the realm of Machine Learning (ML), the concept of Artificial Neural Networks (ANNs) stands out as a powerful tool known for its remarkable capabilities, such as identifying intricate patterns (e.g., pattern recognition), making predictions, conducting clustering, and more [3]. Institutional investors have embraced artificial intelligence-based methods, including ML and Deep Learning (DL), to enhance their profit maximization strategies [4]. Financial data exhibit two prominent characteristics: nonlinearity and irregularity [5]. While econometric and time series models can be effective under certain assumptions, such as data stationarity [6], using NNs can help tackle these challenges effectively.

NNs are a widely adopted tool for predicting stock prices due to their proficiency in recognizing both linear and nonlinear relationships between inputs and outputs [7]. The financial world, including the stock market, is known for its inherent complexity and chaos [8]. By harnessing the ability of NNs to identify nonlinear relationships, we can enhance traditional analytical methods and other computational techniques for stock market prediction [9]. Beyond stock market forecasting, NNs find utility in various financial applications. Experimental and trading systems leverage them to track commodity and futures markets, conduct Forex trading, assist with financial planning, and predict corporate stability and bankruptcy probabilities [10]. Banks employ NNs to evaluate loan applicants and estimate bankruptcy risk. Financial managers utilize them to optimize portfolio planning for profitable investments. With increased interest in tools and methods for maximizing profitability and minimizing risk as investment levels rise, various prediction models, such as statistical methods, technical analysis, fundamental analysis, and linear regression, have been employed. None of these methods, however, have consistently proven their ability to predict the market precisely. They exhibit varying performances across different criteria, including predictability, accuracy, and computational complexity.

Due to the multifaceted nature of factors influencing stock prices, achieving precise predictions is challenging. Consequently, this paper proposes a hybrid approach. It involves using the Genetic Algorithm (GA) for feature selection, focusing on identifying the most relevant input variables. Furthermore, two meta-heuristic algorithms, Harmony Search (HS) and Particle Swarm Optimization (PSO), are employed to enhance the predictive capabilities of the ANN. These algorithms work together to minimize network errors, optimize weights, and determine the optimal number of hidden layers. Eight estimation criteria have been introduced to evaluate and compare the model's performance based on error metrics. The experimental results demonstrate that the hybrid ANN-HS algorithm exhibits the best performance.

The primary objective of this study is to predict stock prices using AI-based and econometric models. Additionally, it aims to explore the implications of the EMH on market efficiency and the applicability of various techniques and models, such as technical and fundamental analysis. Furthermore, the study evaluates the effectiveness of GA for feature selection. The comparison of methods and models is conducted based on error metrics and fitness functions, with the hypothesis that a combination of AI-based and econometric models, along with financial input data, can enhance model robustness and accuracy.

Notably, the study distinguishes itself in two key ways. Firstly, it conducts tests to assess the EMH, a crucial aspect often overlooked in previous research that frequently proceeds without considering this assumption. Secondly, the study encompasses a broader scope by selecting significant companies in Iran across various industries as the statistical population, in contrast to previous articles that have typically focused on predicting the index or share price of a limited number of companies.

This paper combines financial data as input variables, AI-based and econometric models, and statistical techniques to create a robust predictive model. The subsequent sections of this paper are structured as follows: Section 2 provides a literature review. Section 3 outlines the proposed algorithm and examines EMH. Section 4 delves into the experimental process and presents the results, and finally, Section 5 offers concluding remarks.

2 | Literature Review

The stock market is influenced by a multitude of factors, including economic conditions, psychological emotions and expectations, and political events. These factors collectively contribute to the stock market's inherent volatility, nonlinearity, and discontinuity [11]. Furthermore, technological advancements and communication systems have amplified the speed of stock market processes, intensifying stock price fluctuations. Consequently, parties and sectors such as banks, financial institutions, major investors, and brokers must prioritize rapid trading [12]. Investors primarily aim for profit maximization, which has prompted researchers to seek methods or models to achieve this objective [13].

The literature presents two main perspectives on market efficiency. The first perspective asserts the efficiency of the market and its resistance to the prediction of market returns [14]. The second perspective argues for the weak efficiency of the market, emphasizing a low degree of serial correlation and transaction costs, particularly in emerging markets [15]. Proponents of the EMH in the second group contend that future price changes cannot be predicted using historical data. In contrast, those who challenge the EMH assert the presence of anomalies that undermine the theory of efficient markets, as exemplified in various empirical studies [1]. These anomalies represent deviations from theoretical expectations [16].

The advancement of computer technology over the past decades has facilitated the development of optimization methods, resulting in the compilation of numerous techniques during this period. Mathematical optimization models encompass various types, including linear, nonlinear, integer, binary, probabilistic, and complex models [17]. Broadly speaking, methods for solving optimization problems can be categorized into two primary groups: exact and meta-heuristic. Accurate methods comprise algorithms such as simplex, linear, and dynamic programming, including integer programming and its derivatives. In contrast, unlike exact methods, meta-heuristic methods offer viable solutions within acceptable timeframes for large-scale problems. The expansion and diversification of applications of these methods in recent decades have garnered interest among researchers across various scientific disciplines and fields, including design engineering, aerodynamics, robotics, telecommunications, chemical process simulation, control, transportation, logistics, and supply chain management. The widespread utilization of these algorithms highlights their practical suitability and effectiveness. Noteworthy advantages of meta-heuristic methods include accelerated computational speeds, user-friendliness, and other benefits previously mentioned.

The structural framework of various optimization algorithms can be outlined in *Fig. 1*. As previously discussed, meta-heuristic algorithms find applications in diverse fields. In the context of this paper, these meta-heuristic algorithms play a pivotal role in training the ANN and serving as an optimization method. To represent the history of ML in *Fig. 2*, along with corresponding expert phrases, you can follow these guidelines provided in reference [18]. In *Fig. 2*, the methods, NNs, and tools can be categorized into green, red, and blue colors. It is evident that with each successive phase, ML has evolved and become more sophisticated, enabling it to perform a wide range of tasks, including graphical tasks, reasoning-based tasks, and decision-making, among others. In its early stages, ML experiments focused on the concept of computers recognizing patterns in data and learning from them. Over time, these foundational experiments have paved the way for developing increasingly complex ML techniques. Although ML algorithms have existed for quite some time, the rapid and effective application of intricate algorithms for handling big data is a more recent advancement.

ML has progressed from automating manual data entry to tackling more intricate tasks like fraud detection and insurance risk assessment. It is in these areas that ML truly shines. A significant advantage of ML lies in its ability to discern what the human eye might overlook. ML models are adept at identifying complex patterns that could easily elude human analysis.

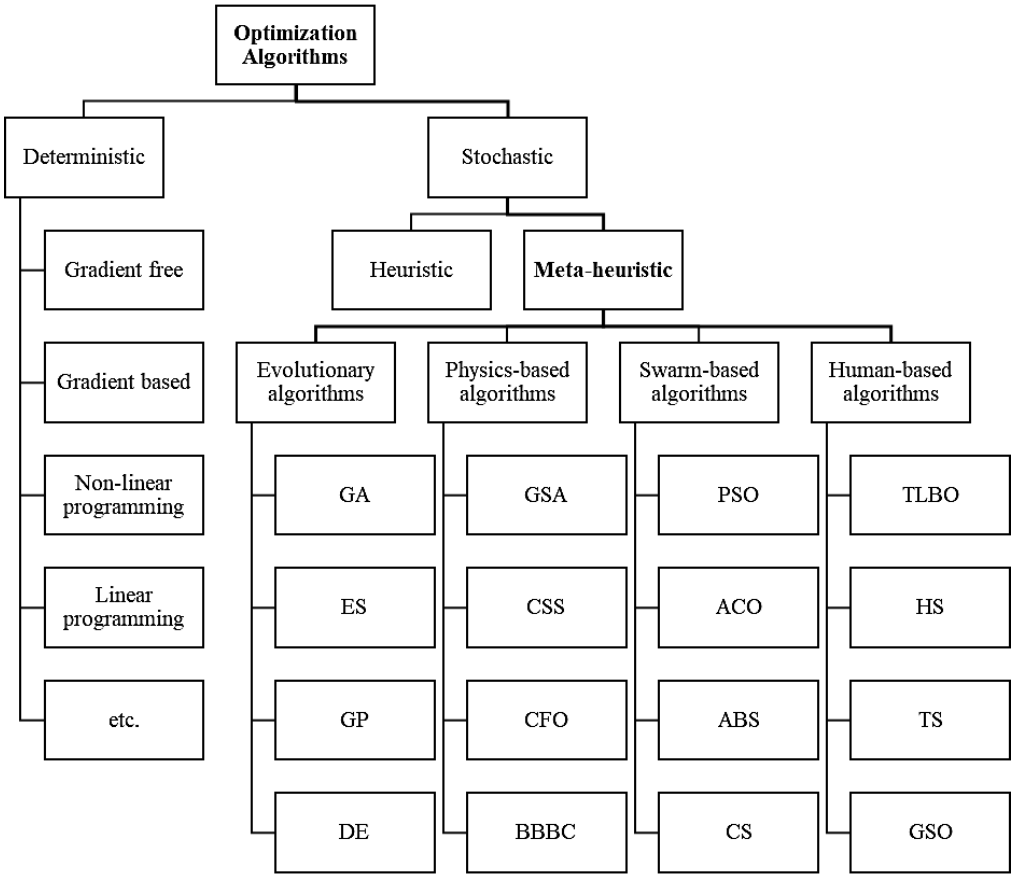


Fig. 1. Optimization algorithms.

Fig. 1. represents the position of meta-heuristic algorithms among other optimization algorithms.

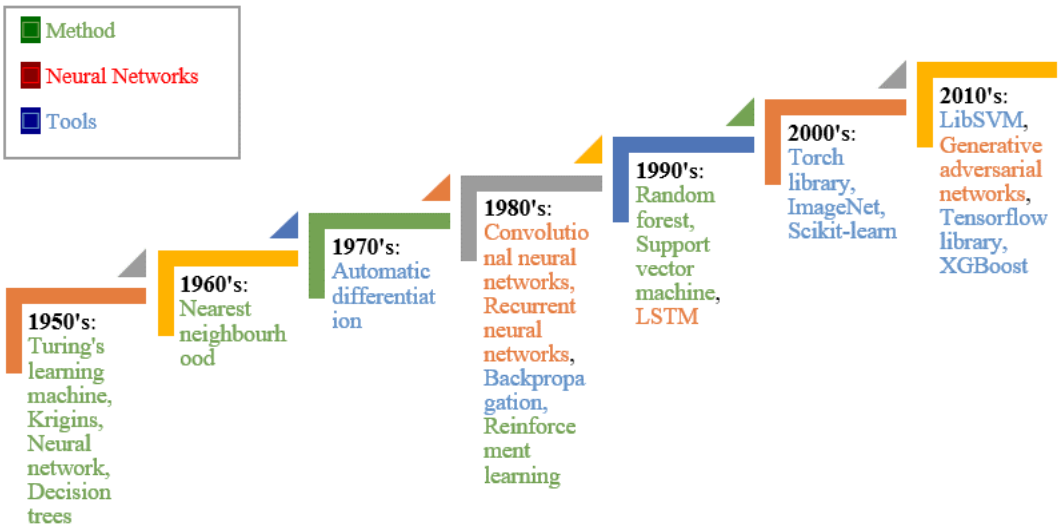


Fig. 2. ML history.

ML can automate nearly any task that relies on data-defined patterns or a set of rules. *Table 1* provides a concise overview of various ML techniques. To better understand the relationship between categories and AI components, *Fig. 3* is depicted [19].

Table 1. ML components and techniques.

Name	Definition
Regression	Regression methods help to predict or explain a particular numerical value based on a set of prior data.
Classification	Classification methods predict or explain a class value.
Clustering	Group or cluster observations that have similar characteristics.
Dimensionality reduction	Remove the least important information (sometimes redundant columns) from a data set.
Ensemble methods	Combining several predictive models (supervised ML) to get higher quality predictions than each of the models could provide on its own.
NN and DL	NNs aim to capture nonlinear patterns in data by adding layers of parameters to the model.
Transfer learning	It refers to re-using part of a previously trained neural net and adapting it to a new but similar task.
Supervised learning	Allows you to collect data or produce a data output from a previous ML deployment.
Unsupervised learning	Helps you find all kinds of unknown patterns in data.
Reinforcement learning	Taking suitable action to maximize reward in a particular situation.
Natural language processing	A widely used technique to prepare text for ML.

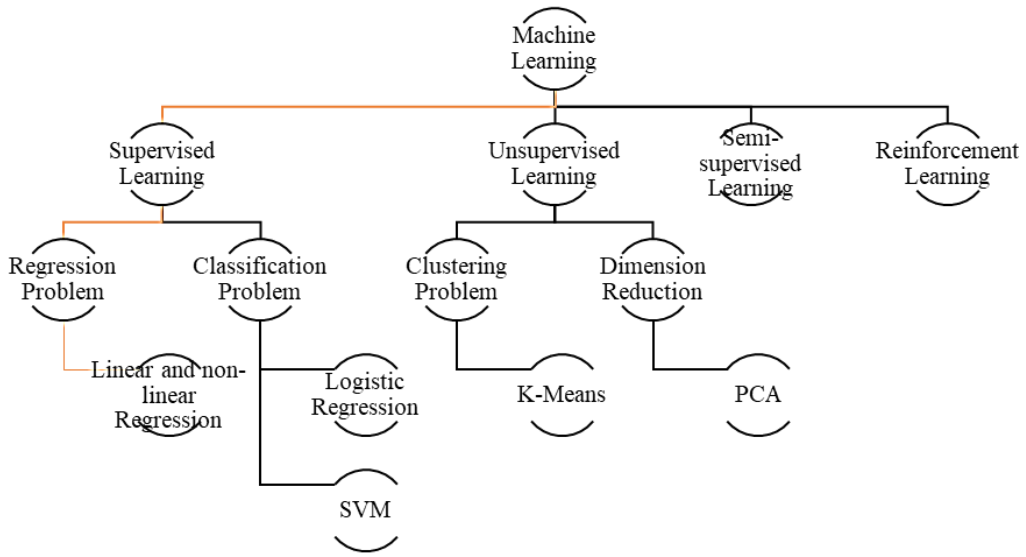


Fig. 3. ML subsets.

The experimental findings and research studies consistently demonstrate the robustness of ANNs in forecasting stock prices [20]. Researchers continually seek models to enhance prediction accuracy, reduce errors, and expedite calculations. In this regard, meta-heuristic algorithms are highly effective, offering optimization and model selection capabilities. ANNs serve as a powerful tool for various forecasting tasks, including stock prices, stock returns, exchange rates, and inflation. Notably, they outperform traditional statistical models [21].

Göçken et al. [22] employed a hybrid ANN based on GA and HS to predict the Turkish price index, incorporating technical indicators as input variables. The results underscored the robustness of the hybrid ANN-HS approach. Hassanin et al. [23] explored the use of Grey Wolf Optimization (GWO) for providing initial solutions to ANNs, highlighting the superiority of the hybrid GWO-ANN method. Faris et al. [24] utilized a Multi-Verse Optimizer (MVO) to optimize weights and biases for multilayer perceptron networks, comparing their performance with several other algorithms. Rather et al. [25] emphasized the burgeoning interest in hybrid forecasting approaches, concluding that hybrid ANNs contribute to model robustness.

Yang et al. [26] predicted Chinese stock prices using an ensemble of multilayer feedforward networks, incorporating the Backpropagation Neural Network (BPNN) for training and a bagging approach for

ensemble formation [27]. Their results indicated the somewhat predictable nature of the Chinese market using this approach with satisfactory accuracy, precision, and recall. Zhang et al. [9] proposed a system that predicts stock price movement and growth/decline rate intervals within predefined durations. They employed a random forest model to classify stocks into four primary classes and achieved superior results in terms of market volatility prediction. Ahmed et al. [28] utilized Ant Colony Optimization (ACO) for stock price prediction from the Nigerian stock exchange, demonstrating the robustness of ACO when compared to other criteria. Ghanbari and Arian [29] employed Support Vector Regression (SVR) and Butterfly Optimization Algorithm (BOA) for stock market prediction, achieving significant improvements in parameter optimization and model performance. Chandana [30] introduced a hybrid prediction method for stock prices based on LSSVR and ML, simplifying calculations while enhancing accuracy. Rajesh et al. [31] explored ensemble learning techniques to predict S&P500 trends, favoring Random Forest, SVM, and K-nearest neighbors' classifiers as the most powerful predictive models. Lv et al. [32] compared traditional ML algorithms with advanced deep NN models to predict index component stocks, highlighting the conventional ML algorithms' superiority under no transaction cost and DNN models' better performance when considering transaction costs.

Zaman [33] identified a weak form of market efficiency in the largest stock market of Bangladesh, using parametric and non-parametric tests to test EMH. Shahvaroughi Farahani and Razavi Hajiagha [34] applied ANN to predict indices such as S&P500, DAX, FTSE100, NASDAQ, and DJI. They utilized meta-heuristic algorithms like Social Spider Optimization (SSO) and Bat Algorithm (BA) for network training. GAs were used for feature selection, and various loss functions were employed for error evaluation. Time series models like ARMA and ARIMA were also incorporated into their analysis.

Ranjbarzadeh et al. [35] reviewed AI methods for diagnosing brain tumors using MRI images, employing supervised, unsupervised, and DL techniques alongside image segmentation methods. Farahani et al. [36] examined the impact of Covid-19 vaccines on economic conditions and sustainable development goals, utilizing ANN and the Beetle Antennae Search (BAS) algorithm for prediction. The study revealed that both AI and econometric models yielded similar results, with AI optimization models enhancing robustness.

In the context of supply chain design, Tirkolaee et al. [37] addressed critical issues such as supplier selection, order size identification, and order allocation across three supply chain levels. Their objectives encompassed cost minimization, product value maximization, and supply chain reliability maximization. Sensitivity analyses were conducted on key model parameters to assess supply chain reliability. For further insights into the applications of ANN and algorithms, please refer to Abiodun et al. [38] and Tkáč and Verner [39].

Table 2 overviews previous research on forecasting stock prices using NNs and the methods considered.

Table 2. Some current research on forecasting stock price using ANN and other methods.

Author(s)	Title	Methodology	Results
Pierdzioch and Risse [40]	A machine-learning analysis of the rationality of aggregate stock market forecasts	Testing rationality of aggregate stock market forecasts based on an ML algorithm called Boosted Regression Trees (BRT)	Rational The Expectations Hypothesis (REH) did not reject short-term forecasts, but for a longer term, there was evidence against the REH
Zhong and Enke [41]	Predicting the daily return direction of the stock market using hybrid ML algorithms	Prediction of future stock market index returns using DNN and ANN	The results showed the superiority of DNNs using two PCAs over other hybrid ML algorithms based on classification accuracy

Table 2. Continued.

Author(s)	Title	Methodology	Results
Aytac et al. [42]	Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques	Proposing a novel hybrid model based on Long Short-Term Memory (LSTM), NN and Empirical Wavelet Transform (EWT) along with Cuckoo Search (CS) algorithm to the prediction of crypto currency time series.	Their proposed model succeeded in capturing the nonlinear features of the crypto currency time series.
Jiang et al. [43]	The two-stage ML ensemble models for stock price prediction by combining mode decomposition, Extreme Learning Machine (ELM) and Improved Harmony Search (IHS) algorithm	Introducing a new model by combining three models means Empirical Mode Decomposition (EMD) (or Variational Mode Decomposition (VMD)), ELM and HIS algorithm for stock price prediction	The results showed the superiority of their model based on accuracy and stability compared to the other models.
Behravan and Razavi [44]	Stock price prediction using ML and swarm intelligence	Phase I. uses an automatic clustering algorithm for clustering data. Phase II. using a hybrid regression model based on PSO and SVR for training each cluster, parameter tuning and feature selection.	On average, the proposed method has shown 82.6% accuracy in predicting stock price in 1-day.
Kumar Chandar [45]	Grey Wolf optimization-Elman NN model for stock price prediction	Optimize parameters of ENN using ENN and GWO	The results showed the high predictability of the proposed model and outperformed the benchmark models taken for comparison.
Wei and Cheng [46]	A novel graph convolutional feature-based convolutional NN for stock trend prediction	Proposing a novel method for stock trend prediction using Graph Convolutional feature Convolutional Neural Network (GC–CNN) model, in which both stock market information and individual stock information are considered	The experimental analysis and results presented the superiority of the proposed GC–CNN-based method over several stock trend prediction methods and stock trading strategies.
Kumar Chandar [47]	Hybrid models for intraday stock price forecasting based on ANNs and metaheuristic algorithms	Offering nine new integrated models for forecasting intraday stock price based on the potential of three ANNs.	The results proved that the PSO-BPNN model outperformed other considered models.
Shahvaroughi Farahani and Razavi Hajiagha [34]	Prediction of international stock indices using hybrid models	Training ANN using SSO and BA algorithms.	Better performance of Hybrid model
Shahvaroughi Farahani et al. [48]	Hybrid meta-heuristic ANNs for stock price prediction considering EMH	Testing market efficiency, training ANN using HS and PSO algorithms, technical indicators as input variables, and GA as feature selection.	The best results were obtained in the HS and almost in PSO algorithms with the lowest loss functions.

Researchers have been actively exploring the hybridization of methods to improve results. It is evident that simple algorithms (those without hybridization) exhibit the following characteristics:

I. Increased computation time.
II. Greater model complexity.
III. Vulnerability to local minima or maxima trapping.
IV. Rapid convergence.

First and foremost, we believe that assessing market efficiency should precede any attempts at predicting stock prices using models or methods. This assessment serves as a litmus test for the effectiveness of the considered models or methods inefficiently predicting market trends and volatilities and their potential impacts on the market. Secondly, it's worth noting that not all technical indicators need to be considered as input variables. Hence, a method such as GA becomes essential for feature selection. Thirdly, optimizing and fine-tuning the ANN is crucial. Therefore, HS and PSO are employed as optimization methods. Finally, comparing the results with previous studies helps demonstrate the robustness of the model.

This paper aims to address the following key questions:

I. Does utilizing a GA for feature selection impact the speed and quality of results?
II. Can an ANN effectively predict the target stock price?
III. Can meta-heuristic algorithms optimize the performance of an ANN?

For a more comprehensive understanding of the limitations of early models, interested readers can refer to *Table B1*.

3 | Hybrid Meta-Heuristic Artificial Neural Networks for Stock Price Prediction

3.1 | Technical Indicators

This section outlines the methodology employed for selecting input variables. In each case, a set of 42 technical indicators is considered as input variables. Technical indicators play a pivotal role in the analysis of stock prices, as they are derived from specific formulas designed for assessing stock prices or market indices through graphical tools. The literature encompasses a wide array of technical indicators, making selecting the most crucial ones a key determinant for profitability among stock market investors [38]. To address this challenge, we employ GA to identify the most relevant input variables. GA serves as the mechanism to pinpoint the indicators that exert the most significant influence on forecasting performance. *Table 3* provides an exhaustive list of all the technical indicators under consideration [34].

Table 3. Important and most common technical indicators as input variables.

Technical Indicators and their Formulas	
$\text{Diff} = \text{Close}_{\text{Today}} - \text{Close}_{\text{Yesterday}}$	$M_{\text{Open}} = \text{Open}_{\text{Today}} - \text{Open}_{\text{Yestrerday}}$
Close	$M_{\text{High}} = \text{High}_{\text{Today}} - \text{High}_{\text{Yestrerday}}$
High	$M_{\text{Low}} = \text{Low}_{\text{Today}} - \text{Low}_{\text{Yestrerday}}$
Low	$M_{\text{Close}} = \text{Close}_{\text{Today}} - \text{Close}_{\text{Yestrerday}}$
Open	$\text{Acc}_{\text{Open}} = \text{M Open}_{\text{Today}} - \text{M Open}_{\text{Yestrerday}}$
$\text{SMA}(5) = \frac{(\text{Close1} + \text{Close2} + \cdots + \text{Close5})}{5}$	$\text{Acc}_{\text{Close}} = \text{M Close}_{\text{Today}} - \text{M Close}_{\text{Yestrerday}}$
$\text{SMA}(6) = \frac{(\text{Close1} + \text{Close2} + \cdots + \text{Close6})}{6}$	$\text{Acc}_{\text{High}} = \text{M High}_{\text{Today}} - \text{M High}_{\text{Yestrerday}}$
$\text{SMA}(10) = \frac{(\text{Close1} + \text{Close2} + \cdots + \text{Close10})}{10}$	$\text{Acc}_{\text{Low}} = \text{M Low}_{\text{Today}} - \text{M Low}_{\text{Yestrerday}}$
$\text{SMA}(20) = \frac{(\text{Close1} + \text{Close2} + \cdots + \text{Close20})}{20}$	$\%K = \left[\frac{(\text{Close} - \text{Low})}{(\text{High} - \text{Low})}\right] * 100$

Table 3. Continued.

Technical Indicators and their Formulas	
EMA(5)Today $= \frac{\text{Close Today} * k + \text{EMA(5)Yestarday} * (1-k)}{5}$	%D = 3 − day SMA of %K
$K = \frac{2}{5+1}$. EMA(5)0 = SMA(5)	
EMA(6)Today $= \frac{\text{Close Today} * k + \text{EMA(6)Yestarday} * (1-k)}{6}$	Slow%K == Fast%D
$K = \frac{2}{6+1}$. EMA(5)0 = SMA(6)	
EMA(10)Today $= \frac{\text{Close Today} * k + \text{EMA(10)Yestarday} * (1-k)}{10}$	Slow%D = 3 − day SMA of %D
$K = \frac{2}{10+1}$. EMA(10)0 = SMA(10)	
EMA(20)Today $= \frac{\text{Close Today} * k + \text{EMA(20)Yestarday} * (1-k)}{20}$	William's%R $= \frac{\text{(Highest high − Close)}}{\text{(Highest high − Lowset low)}}$
$K = \frac{2}{20+1}$. EMA(20)0 = SMA(20)	
$TMA(5) = \frac{\left(SMA(1) + SMA(2) + \cdots + SMA(5)\right)}{5}$	$RSI = 100 - \frac{100}{1+RS}$. $RS = \frac{\text{Average Gain}}{\text{Average Loss}}$
$TMA(6) = \frac{\left(SMA(1) + SMA(2) + \cdots + SMA(6)\right)}{6}$	Middle Band = SMA(20)
$TMA(10) = \frac{\left(SMA(1) + SMA(2) + \cdots + SMA(10)\right)}{10}$	Upper Band = MA(TP. n) + m ∗ σ[TP. n]
$TMA(20) = \frac{(SMA(1) + SMA(2) + \cdots + SMA(20))}{20}$	Lower Band = MA(TP. n) − m ∗ σ[TP. n]
$AccDist = Acc\ Dist_{Yesterday} + Volume * CLV$	$MP = \frac{(High + Low)}{2}$
$CLV = \frac{[(Close - Low) - (High - Close)]}{High - Low}$	ROC $= \frac{\text{(Close today − Close N previous day)}}{\text{Close N previous day}}$
MACD = EMA(12) − EMA(26)	
Signal MACD = EMA(MACD. 9) = MACD Today ∗ 0.2 + (Signal MACD Yesterday ∗ (0.8))	Typical Price $= \frac{(High + Low + Close + Open)}{4}$
Weighted Close $= \frac{((Close * 2) + High + Low)}{4}$	

Table 3 shows stochastic indicators (%K and %D) have two types: fast and slow. Also, low and high are the minimum and maximum prices of the n periods ago. About the RSI indicator, average gain and average loss are calculated as follows:

Average Gain= [(previous Average Gain) x 13 + current Gain] / 14.

Average Loss= [(previous Average Loss) x 13 + current Loss] / 14.

According to the Bollinger Band indicator, MA stands for moving average, TP is typical price, n is equal to the number of periods, which is usually 20 and m is standard deviation and often equals 20. The last one, $\sigma[TP.n]$, is equal to the standard deviation during the n period of *TP*.

3.2 | Artificial Neural Network

The study initially employs a simple ANN without any additional algorithms. Subsequently, a hybrid ANN approach is employed to select input variables and determine the number of input and hidden

layers. Similar to the methodology employed by Göçken et al. [22], a Multilayer Perceptron (MLP) is utilized in this research, comprising three layers (two layers for input and output variables and one layer for the hidden layer). The input layer incorporates 42 input variables, effectively consisting of 42 neurons. Since the output layer pertains to a single variable, it contains one neuron. The inclusion of hidden layers enhances the model's capacity to detect complexity.

A trial-and-error approach is adopted to determine the number of hidden layer neurons in the standard NN model. A range of 1 to 32 neurons in the hidden layer is explored, and the optimal number of neurons that yields the highest accuracy is selected. Additionally, error-back propagation is employed as the network's training method. The Levenberg-Marquardt optimization algorithm is utilized to minimize errors [49]. The training process involves 1000 epochs, with the initial training rate set at 0.01 and gradually decreasing to 0.001 to achieve greater precision. To enhance ANN recognition and predictive capabilities, two threshold functions are incorporated. The first recognizes linear qualifications, while the second identifies nonlinear qualifications within the model. The output function of the hidden layer is the sigmoid function, while a linear function represents the output layer's threshold function. *Fig. 4* illustrates the architectural design of the proposed NN [22].

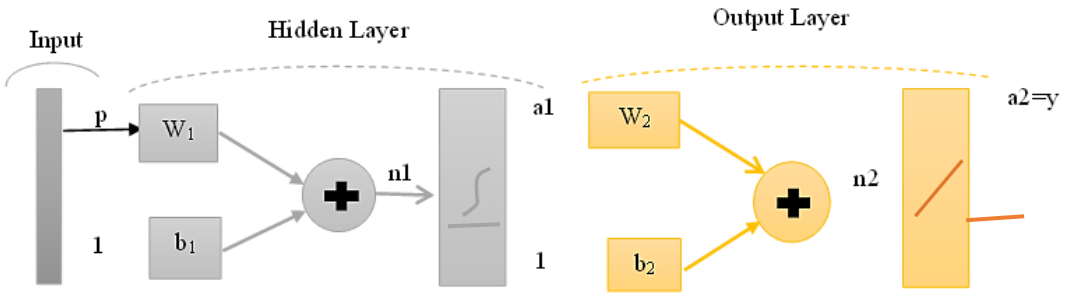


Fig. 4. Architecture of the proposed NN.

The notations used in *Fig. 4* are summarized as follows:

P : The input patterns.

b_1 : The vector of bias weights on the hidden neurons.

W_1: The weight matrix between the 0th (i.e., input) and 1th (i.e., hidden) layers.

a_1 : The vector containing the outputs from the hidden neurons.

n_1: The vector containing net-inputs going into the hidden neurons.

a_2 : The column-vector coming from the second output layer.

n_2: The column-vector containing the net inputs going into the output layer.

W_2: The synaptic weight matrix between the 1st (i.e., hidden) layer and the 2nd (i.e., output) layer.

b_2: The column-vector containing the bias inputs of the output neurons.

In each layer of the NN, weights and biases play a crucial role [1]. These weights and biases are summed, passing the result through a nonlinear activation function [50]. These activation functions are control parameters, limiting the output values [51]. Moreover, recognizing nonlinear characteristics within the hidden layer is important, necessitating carefully selecting an appropriate nonlinear activation function [52].

The proposed hybrid approach in this study unfolds in two main steps. Firstly, technical indicators are calculated, and the optimal indicators are selected using GA. Subsequently, different hybrid ANN models are employed to predict the closing price, and their prediction errors are compared. To facilitate this analysis, the stock price data from 2013 to 2018 are divided into two subsets: one for training, consisting of 70% of the observations, and the other for testing and validation, comprising the remaining 30%. The performance of the models is assessed using eight different evaluation criteria. In this research, 42 technical indicators are utilized as input variables. These variables are normalized within the range of 0 to 1 using *Eq. (1)*, rendering them suitable for input into the models. In *Eq. (1)*, the numerator i represents the data point. Moreover, *Fig. 5* represents the research methodology.

$$\widetilde{S_i} = \frac{(S_i - S_{min})}{S_{max} - S_{min}}. i = 1 \dots N. \quad (1)$$

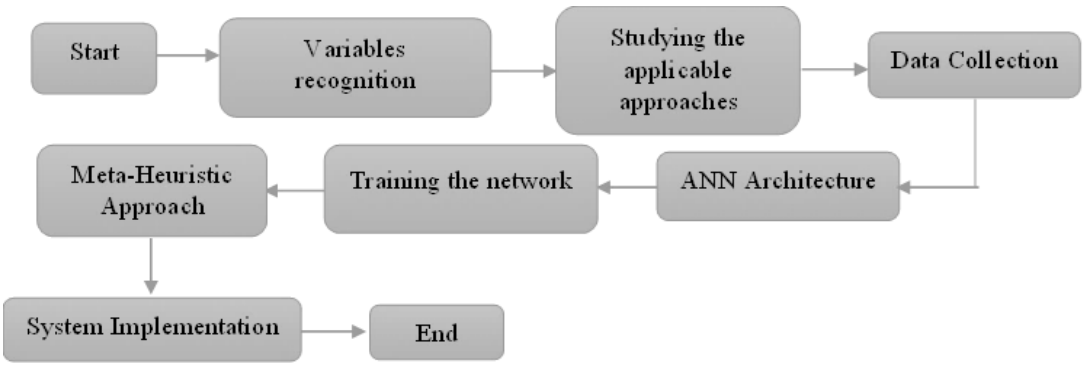


Fig. 5. Research methodology [53].

Algorithm parameter tuning is a critical aspect of AI-based algorithms. Some parameters, like crossover and mutation, are set based on existing literature, while others are chosen according to the default settings of the corresponding software. The hyper-parameters subject to tuning encompass the number of neurons, activation function, optimizer, learning rate, batch size, and epochs.

To determine the optimal number of neurons, a trial-and-error approach is employed. Neuron counts ranging from one to 32 are tested, and the architecture that yields the best number of neurons and fitness value is selected based on the loss function [54]. For the activation function, two categories are considered: nonlinear and linear. MATLAB software is used to identify the most suitable activation function, exploring options like sigmoid, tanh, and exponential. Leveraging the LM optimization algorithm, the best activation function type, learning rate, batch size, and epochs are determined.

Additional details regarding parameter tuning are summarized in *Table 4*. The batch size signifies the number of samples passed through the network in a single iteration.

Table 4. Tuning the parameters and layers of ANN.

No	Parameters	Values
1	Neurons	1, 32
2	Activation	0, 5
3	Learning rate	0.01, 1
4	Batch size	100, 1000
5	Epochs	1, 1000
6	Layer 1	1, 3
7	Layer 2	1, 3

3.3 | Genetic Algorithm-Artificial Neural Network Model

In this study, a binary solution representation is employed to encode solutions. Each chromosome consists of 47 bits, each representing the presence or absence of an input variable (technical indicator).

Specifically, "1" denotes the existence of the corresponding variable, while "0" signifies its non-existence. An additional five bits are allocated to express a number within the range of 1-32 (25=32), indicating the number of neurons in the hidden layer.

The GA operates with a population size of 20 [55]. The initial population is generated randomly. The loss function is the Mean Square Error (MSE), with the input variables comprising technical indicators. The number of neurons in the hidden layer is estimated based on minimizing the MSE rate. The smallest MSE is considered the optimal choice for the subsequent forecasting period. The initial number of epochs is set to 100 to accelerate the algorithm's training process and progressively increase it to achieve improved results. Initially, the training (learning) rate is set at 0.01 to facilitate extensive exploration, expanding the search space. Subsequently, it is systematically reduced during training to obtain more precise outcomes. The number of epochs can be increased to 1000 to enhance result accuracy. The parameters considered for the GA are summarized in *Table 5*. *Fig. 6* also represents the related flowchart of the Genetic Algorithm-Artificial Neural Network (GA-ANN) [22].

Table 5. GA parameters.

Output Error	Output Activation Function	Input Activation Function	Mutation Rate	Crossover Rate	Number of Generation	Population Size
SSE	Logistic	Logistic	0.1	0.9	50	20

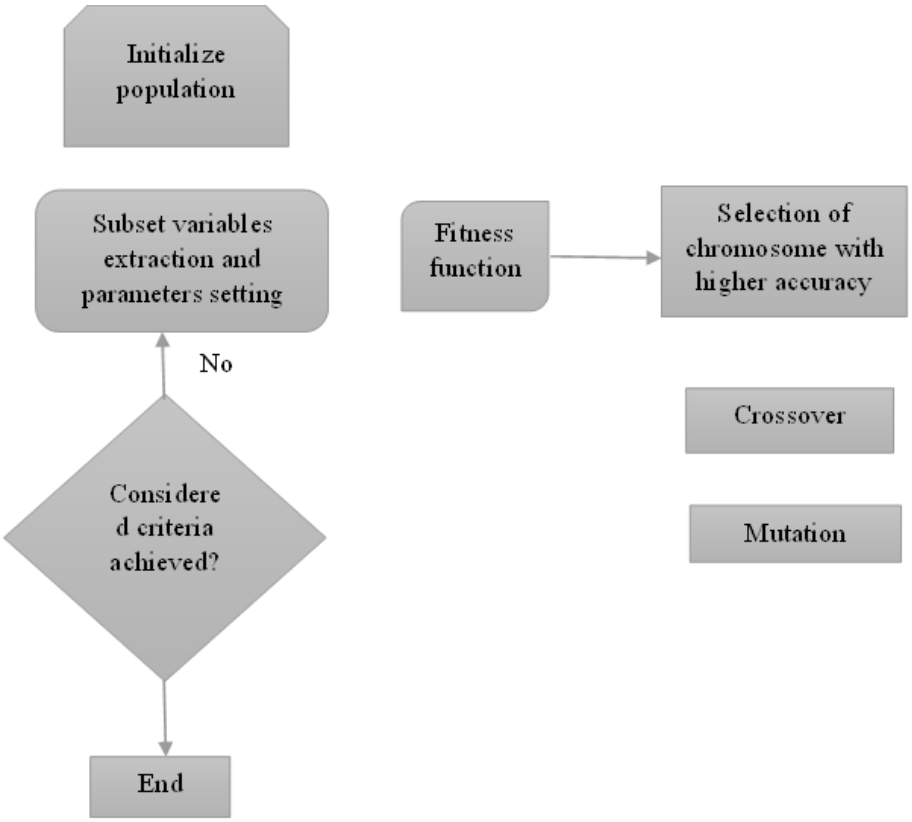


Fig. 6. Considered GA flowchart for training ANN.

Within the GA framework, parent selection is performed using the roulette wheel method, with a crossover rate set at 0.8. A one-point crossover technique is applied for crossover operations. The mutation is incorporated with a rate of 0.2, utilizing a binary mutation approach. Among the 20 parents and 20 children generated, the 20 best individuals are selected to form the new generation. This process continues until a

termination condition is met. One of the termination conditions entails repeating the best-found solution for up to 100 iterations. If this condition is not met, the maximum iteration count is checked, with the maximum allowable iterations set at 2000. The mutation and crossover operators are visually depicted in *Fig. 7*.

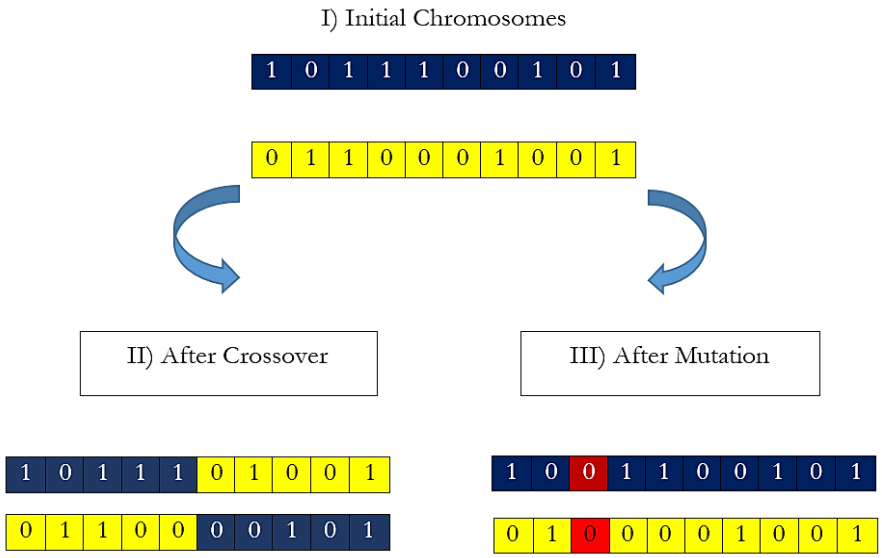


Fig. 7. Crossover and mutation operator.

Offspring, representing newly generated solutions resulting from the crossover of two-parent strings, are primarily produced with a relatively high probability, typically within the range of 0.8 to 0.95. Conversely, mutation involves the alteration of certain digits within a string, resulting in the creation of novel solutions. The pseudocode for the GA used in hyperparameter tuning is outlined as *Algorithm 1*.

Algorithm 1. GA-ANN pseudocode.

Function *GENETIC-ALGORITHM (population, FITNESS-FN)* ***returns*** *an individual*
Inputs: *population, a set of individuals*
FITNESS-FN, a function that measures the fitness of an individual
Repeat
new_population <-- empty set
For *i= = 1* ***to*** *SIZE (population)* ***do***
x <-- RANDOM-SELECTION (population, FITNESS-FN)
y <-- RANDOM-SELECTION (population, FITNESS-FN)
Child <-- REPRODUCE (x, y)
If *(small random probability)* ***then*** *child <-- MUTATE (child)*
Add child to new_population
Population <-- new_population
Until *some individual is fit enough, or enough time has elapsed*
Return *the best individual in the population, according to FITNESS-FN*

3.4 | Optimization Algorithm (PSO-ANN & HS-ANN)

PSO algorithm begins with the initial population of solutions and moves toward optimal solutions in sequential iterations. In each iteration, two solutions are updated (X_j^{Gbest} and$X_j^{i.pbest}$), representing the best solution found by all particles and the best solution found by a particle j, respectively. The foundation of PSO is that each moment, each particle sets its location in searching space with the best location currently existing and the best location in its neighborhood [56]. Accordingly, speed movement and the next particle location are obtained as:

$$V_{j+1}^{i} = W_j V_j^i + c_1 r_1 \left(X_j^{i.pbest} - X_j^i\right) + c_2 r_2 \left(X_j^{Gbest} - X_j^i\right), \quad (2)$$

$$X_{j+1}^{i} = X_j^i + V_{j+1}^i, \quad (3)$$

where

c_1: Importance of personal best.

c_2 : Importance of best neighborhood.

Usually, $c_1 + c_2 = 4$ [57]. Accelerate constant c_1 and c_2 represent the particle stochastic acceleration weight toward the personal best (pbest) and the global best (gbest). The c_1 and c_2 need to be chosen in such a way that the search speed decreases over time. $r_1$and $r_2$are random numbers between 0.1. V_j^i and X_j^i are speed and location of the jth particle in ith iteration. *Table 4* illustrates the parameters of PSO. *Fig. 8* represents the process of training ANN with PSO [58]. The pseudocode of Particle Swarm Optimization-Artificial Neural Network (PSO-ANN) is also shown is *Algorithm 2*.

Table 6. PSO parameters.

Parameters	Size
Upper bound	1.5
Lower bound	-1.5
C_1	1.5
C_2	2.5
Max iteration	2000

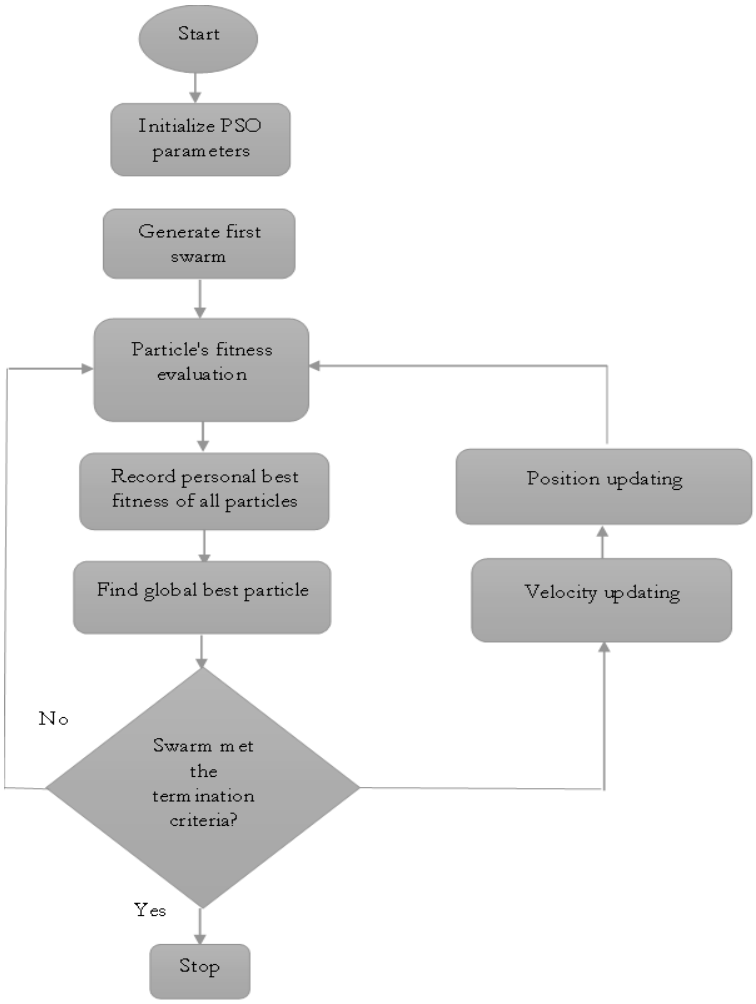


Fig. 8. Considered PSO flow chart for training ANN.

Algorithm 2. PSO-ANN Pseudocode.

For each particle
Initialize particle
END
Do
For each particle
Calculate fitness value
If the fitness value is better than the best fitness value (pBest) in history
set current value as the new pBest
End
Choose the particle with the best fitness value of all the particles as the gBest
For each particle
Calculate particle velocity according to equation (2)
Update particle position according to equation (3)
End

To fine-tune the parameters of the PSO algorithm, two distinct methods are employed: offline and online approaches. In the offline method, meta-optimization is utilized to adjust the parameters of PSO using an external optimizer. In contrast, the online method encompasses two techniques:

I. Self-adaptation: this approach involves including some or all of the optimizer's behavioral parameters in the search space, allowing them to be optimized alongside the primary problem.
II. Meta-adaptation: in this technique, an external optimizer dynamically adjusts the parameters of another optimizer during the optimization process.

In our study, we work with training and theoretical datasets. Instead of relying on random values for particle initialization, we use the training data. For this purpose, we reference Fei Ye's article on tuning PSO hyperparameter estimation [59]. In this study, we use the HS algorithm to train ANN and find the fittest number of input and hidden layers. The HS consists of three basic phases: initialization, improvisation of a harmony vector and updating the HM, described below respectively [60]. In addition, other parameters of HS should be determined. These parameters are Harmony Memory Size (HMS), which is equal to 100; Harmony Memory Considering Rate (HMCR), which is equal to 0.95 and Pitch Adjusting Rate (PAR), which is 0.3; and Bandwidth (BW), which is 0.2. We can show the HM with HMS*(N+1), which in this study N is 42. Running HS algorithm including multiple steps (the HS parameters are determined in *Table 7*):

Step 1. Parameter initialization.

Step 2. Harmony memory initialization.

Step 3. Update harmony memory.

Step 4. Checking the stopping criterion.

Table 7. HS parameters.

Parameters	Size
Lower bound	-11
Upper bound	11
HMS	11
NHMS	100
Max iteration	1000
HMCR	0.75
PAR	0.05
Fret Width (FW)	0.1
FW-damp	0.95

Table 5 is equivalently employed as the BW, one factor affecting the algorithm's time complexity and performance. The BW value is a random number between 0 and 1, intended to possess both exploratory and exploitative characteristics. The underlying concept involves initiating the search with a broad exploration of the entire domain and dynamically adjusting it as we approach the optimal solution. FW-Damp, on the other hand, refers to an influence exerted on an oscillatory system that serves to reduce, restrict, or dampen its oscillations. *Fig. 9* illustrates the ANN training process with HS [61].

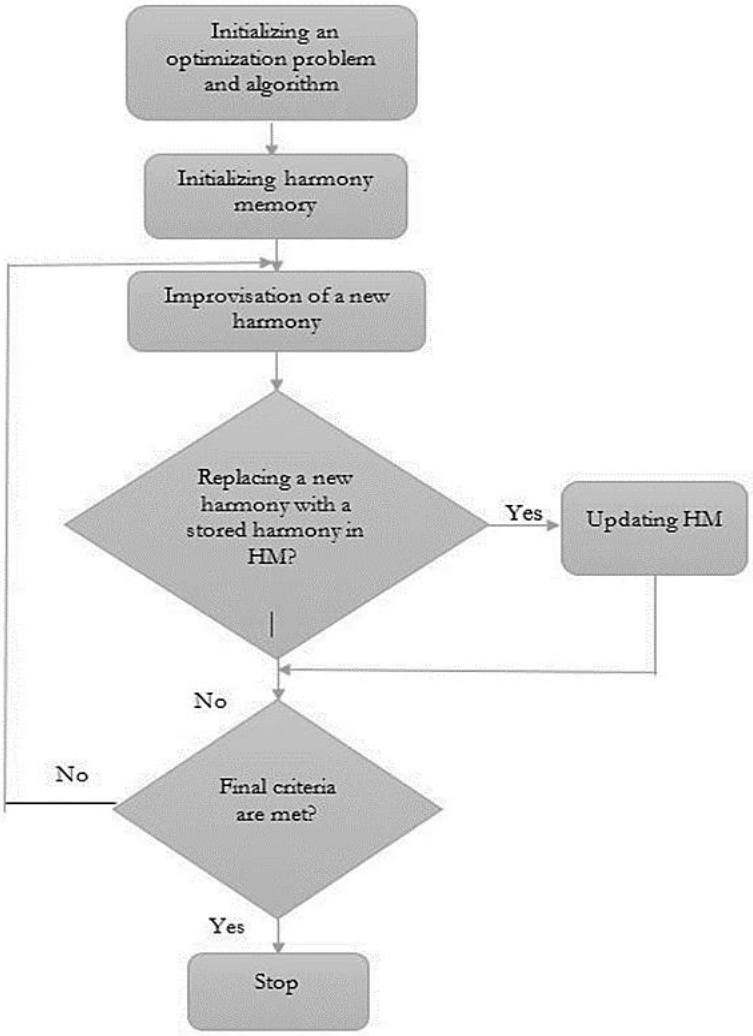


Fig. 9. Considered HS flow chart for training ANN.

The pseudocode for Harmony Search-Artificial Neural Network (HS-ANN) is shown in *Algorithm 3*. Like other algorithms, the HS algorithm needs parameter tuning. As a result, Kang et al. [62] is used as a reference for this purpose [48].

Algorithm 3. HS-ANN Pseudo-Code.

for (j=1 to n) do
if (r1< HMCR) then
Xnew(j) = Xa(j) where a ∈ (1, 2…, HMS)
if (r2< PAR) then
Xnew(j) = Xnew(j) ± r3 × BW where r1, r2, r3∈ (0,1)
endif
else
Xnew(j) = LBj + r × (UBj -LBj), where r ∈ (0,1)
endif
endfor

3.5 | Loss Functions

This study employs a range of pre-existing loss functions available in MATLAB to identify the optimal model performance, aiming for the highest accuracy and lowest error. These loss functions are summarized in *Table 8*, which are readily accessible within MATLAB. Ultimately, the accuracy of the various methods is compared based on the values of the loss function.

Table 8. Most common loss functions.

Error Criterion Formula	Error Criterion
$MAE=\frac{1}{n}\sum_{i=1}^{n}\|ei\|$	Mean absolute error
$MSE=\frac{1}{n}\sum_{i=1}^{n}ei^2$	MSE
$RMSE=\sqrt{\frac{1}{n}}\sum_{i=1}^{n}ei^2$	Root MSE
$MARE=\frac{1}{n}\sum_{i=1}^{n}\left\|\frac{ei}{ai}\right\|$	Mean absolute relative error
$MSRE=\frac{1}{n}\sum_{i=1}^{n}\left\|\frac{ei}{ai}\right\|$ ^2	Mean squared root error
$RMSRE=\sqrt{\frac{1}{n}}\sum_{i=1}^{n}\left\|\frac{ei}{ai}\right\|$ ^2	Root mean squared relative error
$MAPE=\frac{100}{n}\sum_{i=1}^{n}\left\|\frac{ei}{ai}\right\|$	Mean absolute percentage error
$MSPE=\frac{100}{n}\sum_{i=1}^{n}\left\|\frac{ei}{ai}\right\|$ ^2	Mean squared prediction error

3.6 | Testing Efficient Market Hypothesis

One of the pivotal considerations for investors before making stock market investment decisions is the assessment of market efficiency. Consequently, measuring stock market efficiency is paramount and can exert a profound impact. In modern economies, market efficiency bears significant implications because it facilitates accurate stock price determination and optimizes capital allocation [62]. Market efficiency, when realized, signifies a state of information transparency among market participants, the absence of mispricing in the stock market, and the inability to earn abnormal returns. Under such circumstances, technical analysis becomes less applicable.

Various methods exist for evaluating the EMH. Given that financial data typically exhibit abnormal characteristics with skewness and kurtosis, we have employed a non-parametric test, namely the run test. The run test is a statistical procedure that assesses whether a data sequence follows a random distribution pattern. It examines the occurrence of similar events separated by dissimilar events.

In investing, a run test holds significance for investors to discern whether the dataset they are utilizing is generated randomly or influenced by an underlying variable. In inefficient markets, pertinent information affecting asset prices is not readily accessible, rendering it challenging to ascertain or predict accurate asset prices. It, in turn, obscures the true value of financial holdings, resulting in a weakened market. The selected test for evaluating the randomness of the data is the run test.

Table 9. Run test notification.

Term	Description
Observed	The number of runs in the sample
Expected	$1+\frac{2\times A\times B}{N}$
Variance	$\frac{2\times A\times B\times(2\times A\times B-N)}{N^2(N-1)}$
A	The number of observations above K
B	The number of observations below or equal to K
N	The number of observations

Since the article's main focus is on ANN, a brief and complementary explanation is provided about EMH. It could be possible to use the Kolmogorov-Smirnov goodness of fit test to check if a sample comes from a population with a specific distribution [63]. The randomness of data is also evaluated using a run test [64].

Table 10. Kolmogorov-Smirnov (K-S) hypothesis.

H_0:	There is a specific distribution in the data structure.
H_1:	There is no specific distribution in the data structure.
Test statistic:	The Kolmogorov-Smirnov test statistic is defined as.

4 | Results and Discussion

4.1 | Data Statistics and Results

A set of 42 technical indicators is employed for stock price prediction, encompassing 41 input variables and 1 output variable, which serves as the target variable representing the closing price for the following day. The analysis spans a time frame from the commencement of 2013 to the conclusion of 2018, approximately 5 years, with data recorded daily. The stocks under consideration belong to different sectors, namely Fakhouz, Zagros, Khodro, Fameli, and Kechad, operating in the steel, petrochemical, automotive, copper production, and mining industries. The Risk Laboratory at Khatam university serves as the data source for this study. It is worth noting that some data may be missing in certain intervals. To address this issue, various approaches are available, with the current research opting for a direct write-off method, a common and default approach in many software applications.

The selection of the most relevant indicators is facilitated through a GA. *Table 11* provides a comprehensive overview of the companies, time frame, and data volume incorporated into the study. These five companies were selected due to 1) availability and easy access to data, and 2) they are important and well-known companies in their industry.

Table 11. Statistical description of data.

Target Indicator	Number of Output Layer	Number of Input Variables	Number of Data (After Normalizing)	Number of Data (Before Normalizing)	Time Interval	Symbol
Closing price	1	42	923	1176	2013-2018	Fakhouz
Closing price	1	42	963	1230	2013-2018	Zagros
Closing price	1	42	1086	1213	2013-2018	Khodro
Closing price	1	42	997	1231	2013-2018	Fameli
Closing price	1	42	974	1211	2013-2018	Kechad

4.2 | Artificial Neural Network

To begin with, stock prices for each company are predicted using an ANN without applying any additional algorithms. *Fig. 10* visually illustrates the architecture of the ANN, comprising an input layer, a hidden layer, and an output layer. In this depiction, the input layer consists of 42 nodes, the hidden layer consists of 10 nodes and the output layer consists of 1 node. The activation function employed in the hidden layer is the tangent sigmoid, while the output layer employs a simple linear activation function.

The network is trained using 70% of the available data, with the remaining 30% allocated for validation and testing purposes. The training algorithm utilized for the network is the Levenberg-Marquardt algorithm. One of the primary loss functions employed is the MSE. The training iterations are initially set to 1000, representing the software's default setting. The subsequent figure illustrates the optimal

performance, considering three segments: training, validation, and testing. Regression analysis and corresponding figures are presented below, accompanied by an estimation criterion.

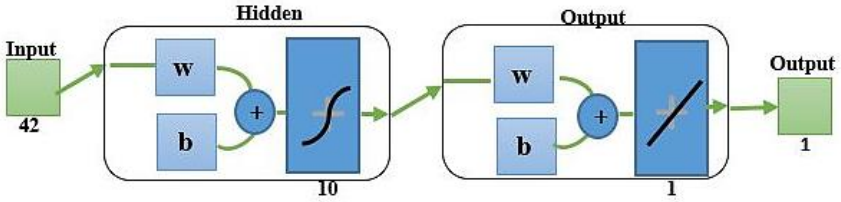
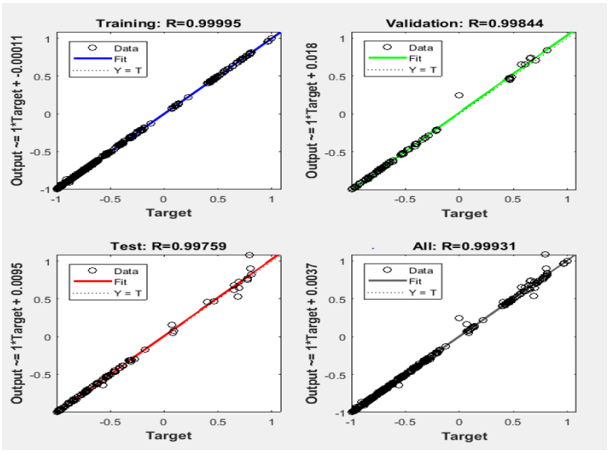
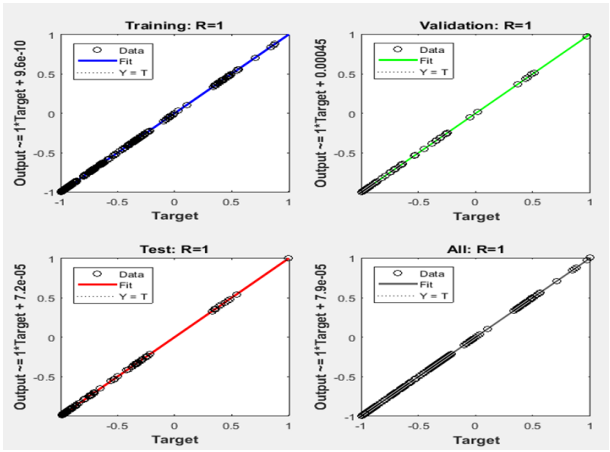


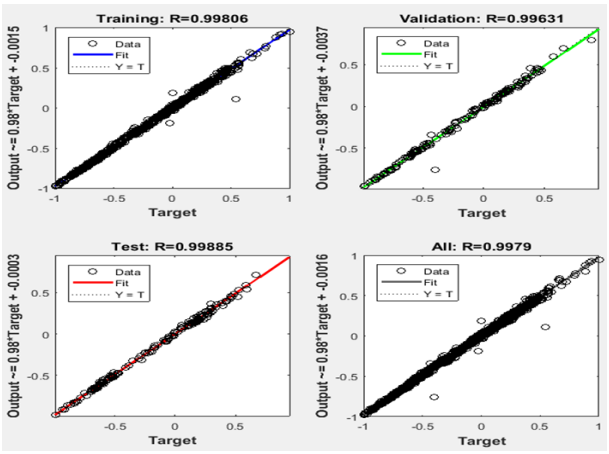
Fig. 10. The structure of ANN.



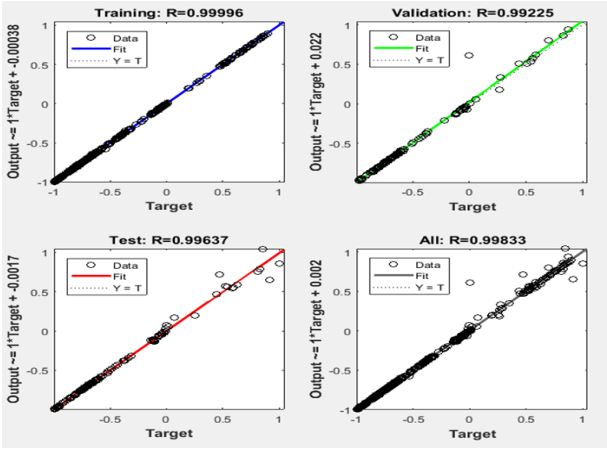
a.



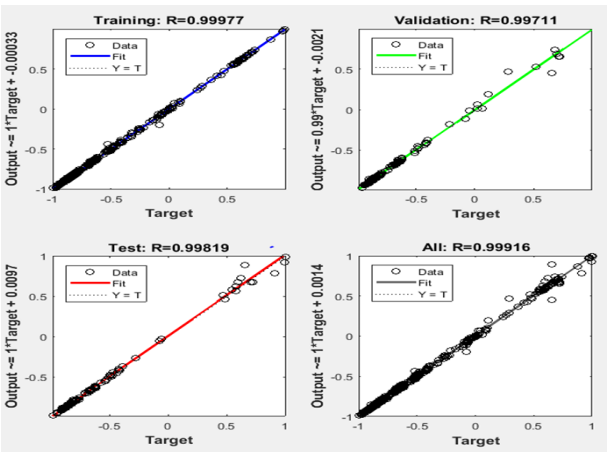
b.



c.



d.



e.

Fig. 11. ANN regression for five companies: a. ANN regression (Fakhouz); b. ANN regression (Zagros); c. ANN regression (Khodro); d. ANN regression (Fameli); e. ANN regression (Kechad).

To effectively train and test our model, it is essential to partition our dataset into three distinct subsets.

Training set: this set comprises data used to train the model and enable it to learn the underlying patterns and features in the dataset. The same training data is repeatedly fed into the NN architecture during each training epoch, allowing the model to grasp the dataset's intricacies progressively. It is crucial for the training set to encompass a diverse range of inputs to ensure that the model is trained comprehensively, enabling it to make accurate predictions on unseen data in the future.

Validation set: the validation set consists of data separate from the training set and is employed to assess the model's performance during training. This validation process provides valuable insights that aid in fine-tuning the model's hyperparameters and configurations. Essentially, it acts as a critic, indicating whether the training is progressing in the right direction. While the model is being trained on the training set, its performance is simultaneously evaluated on the validation set after each training epoch. The primary purpose of this separation is to prevent overfitting, ensuring that the model generalizes well to unseen data rather than merely memorizing the training set.

Test set: the test set is an independent dataset employed to evaluate the model's performance once the training is completed. It provides an unbiased assessment of the final model's accuracy, precision, and performance metrics. It answers the fundamental question of how well the model performs on new, unseen data.

As evident from the results, all five regression models fit the data exceptionally well, as indicated by the R-squared values exceeding 99% for all five companies. Further results details are presented in *Table 12*, where error rates are provided for each dataset. The MSE varies across the datasets, with the lowest and

highest values belonging to Zagros and Khodro. SSE and SAE represent the Sum Squared Error and Sum of Absolute Errors, respectively, which are additional metrics used to assess the model's performance.

Table 12. Training, validation and testing error (before using GA).

Symbol	Training Error	Validation Error	Testing Error	MSE	MAE	SSE	SAE	R^2
Fakhouz	0.003	0.02	0.0452	0.0117	0.52	0.1087	4.837	0.9993
Zagros	4.02e-08	6.01e-04	7.01e-04	9.7e-05	0.003	0.0092	2.6	0.9999
Khodro	0.076	0.14	0.046	0.080	0.0174	0.8780	18.86	0.9979
Fameli	0.0014	0.028	0.15	0.052	0.55	0.65	5.44	0.9983
Kechad	0.008	0.076	0.0795	0.028	0.73	0.281	7.12	0.9916

4.3 | Hybrid GA-ANN

Now, we apply the GA to select the most suitable input and hidden layers for the ANN. To achieve this, we must determine key parameters such as the population size, the number of generations, the mutation rate, and the crossover rate.

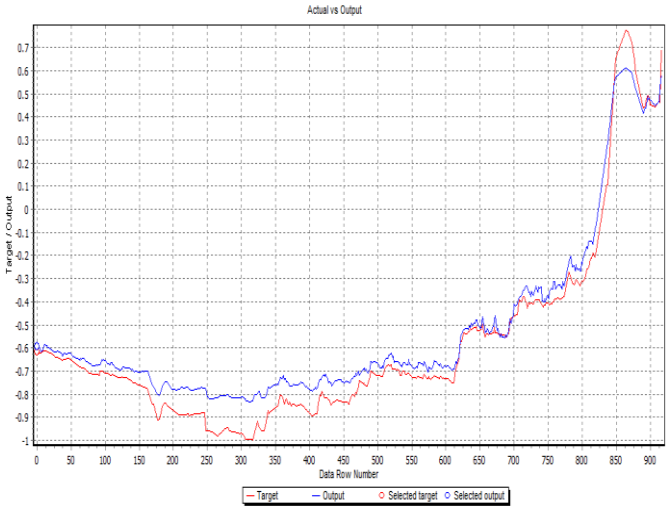
Similar to the previous sections, we employ the GA separately for each company. A summary of the results is presented in *Table 13*. Notably, the error rates for several companies, including Fakhouz, Fameli, and Kechad, have decreased. For the remaining two companies, the error rates are closely aligned. It suggests that the use of the GA has positively impacted model performance. Please refer to *Tables (B2)* and *(B3)* in the appendix for more detailed information.

Table 13. Training, validation and testing error (after using GA).

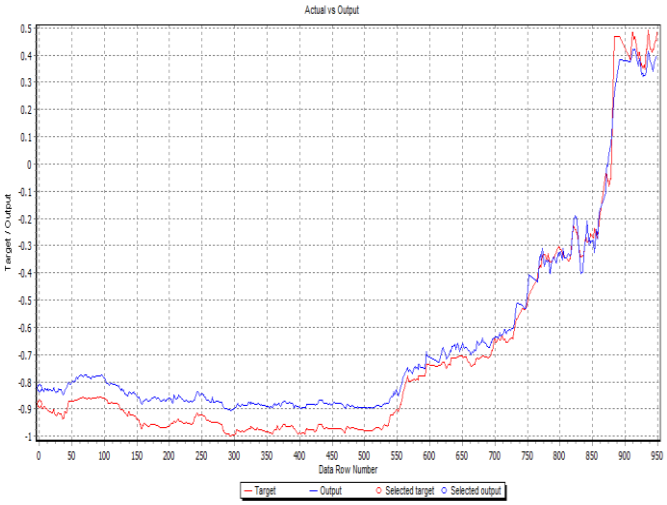
Symbol	Training Error	Validation Error	Testing Error	MSE	MAE	SSE	SAE	R^2
Fakhouz	3.01e-04	1.33e-04	2.95e-04	2.75e-04	0.0084	0.2297	6.99	0.9996
Zagros	2.02e-04	1.45e-04	1.66e-04	1.88e-04	0.0064	0.1798	6.112	0.9989
Khodro	0.0013	0.0032	0.0015	0.0016	0.0263	1.7331	28.46	0.9959
Fameli	0.0015	7.57e-04	4.75e-04	0.0012	0.0153	1.2240	15.22	0.9985
Kechad	2.46e-04	1.83e-04	0.0022	5.25e-04	0.0088	0.5108	8.5220	0.9999

Indeed, after the implementation of GA, we observed a decrease in the error rate and a simultaneous increase in the R-Squared rate. This outcome shows the efficiency of GA as a feature selection method.

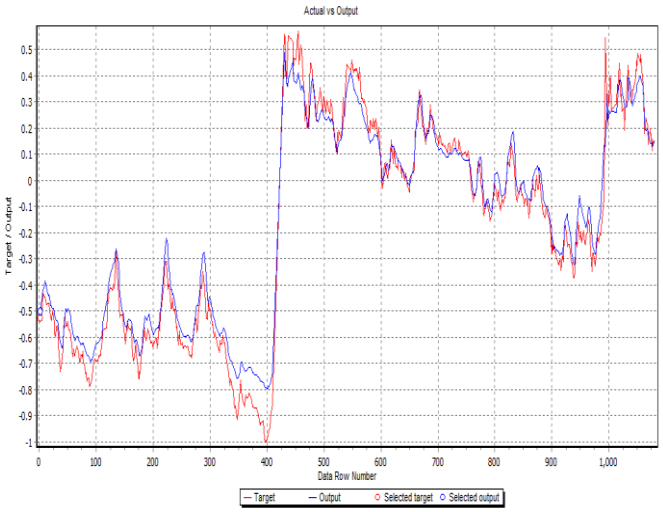
Figs. 12(a) to *12(e)* pertain to the testing phase of the network. In these figures, the horizontal line represents the closing price, while the vertical line denotes the output range, which has been normalized to fall within the range of [1, -1]. The blue line represents the output results generated by the NN, in line with the red line, which represents the input variables. For a more comprehensive overview, please consult *Tables B4* and *B5* in the appendix, which detail the results and the selected most relevant technical indicators.


Actual vs Output
Target / Output
Data Row Number
Target
Output
Selected target
Selected output

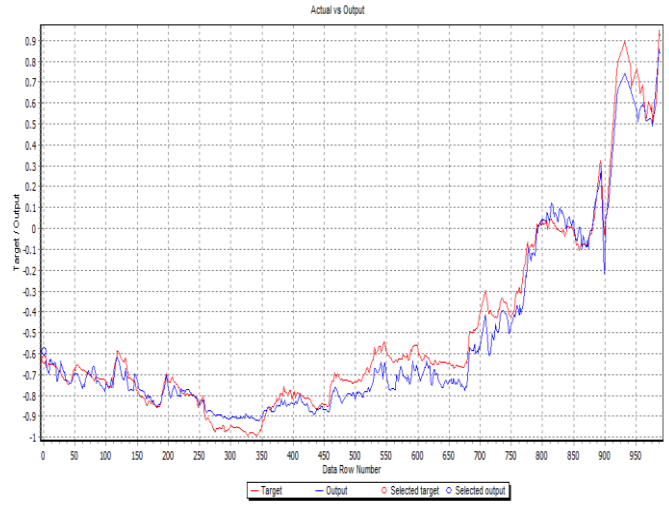
a.


Actual vs Output
Target / Output
Data Row Number
Target
Output
Selected target
Selected output

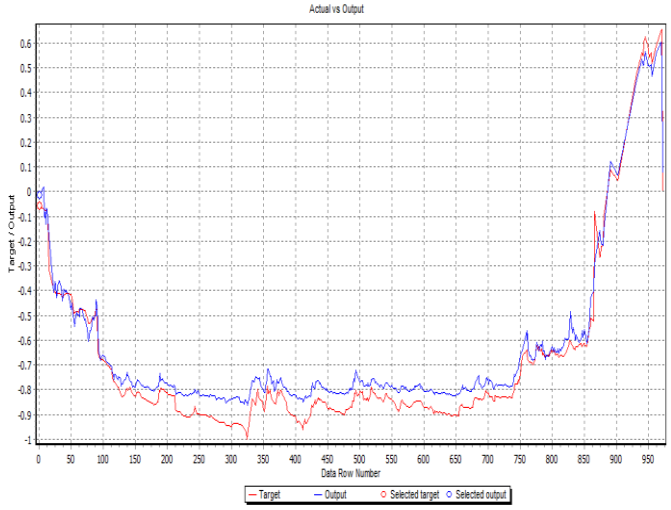
b.


Actual vs Output
Target / Output
Data Row Number
Target
Output
Selected target
Selected output

c.



d.



e.

Fig. 12. Target vs. Output trend for five symbols: a. Target versus output trend (Fakhouz); b. Target versus output trend (Zagros); c. Target versus output trend (Khodro); d. Target versus output trend (Fameli); e. Target versus output trend (Kechad).

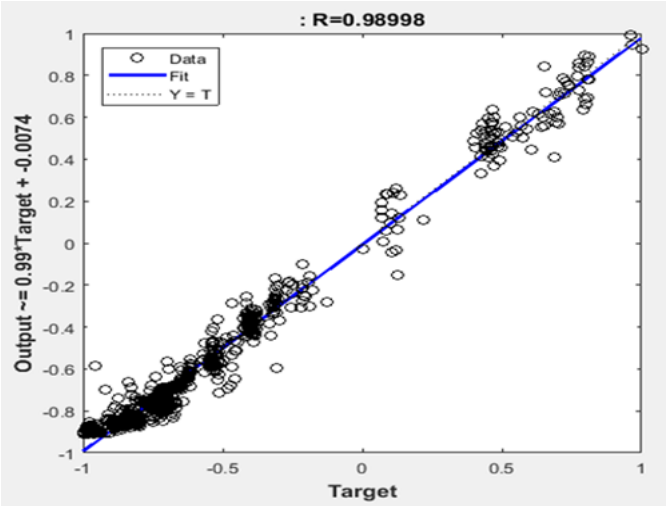
The blue and red lines are aligned and coincidentally but are not fully matched, and it can be better to apply different methods mentioned at the end.

4.4 | Hybrid PSO-ANN

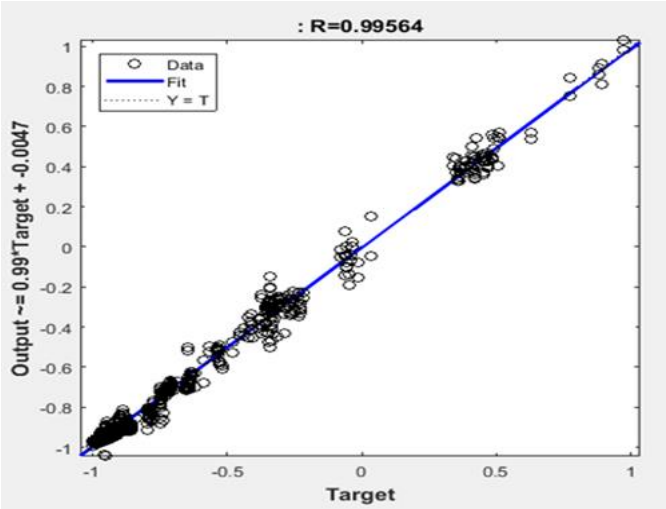
To train the NN using the PSO algorithm, we follow seven key steps as outlined below:

I. Data collection.
II. Network creation.
III. Network estimation.
IV. Weights and biases initialization.
V. Network training using PSO.
VI. Network validation.
VII. Network utilization.

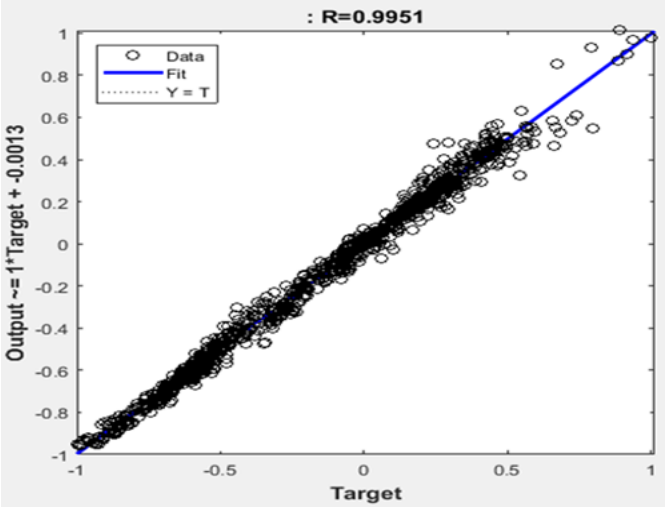
We require a fitness function to predict the stock price, specifically the closing price. The algorithm is initialized, including population and speed, with initial personal best (pbest) and global best (gbest) values. The values for C_1 and C_2 are set for a given number of iterations, which in this case is 1000. It's important to note that the network is a feedforward NN. The regression results are visualized in *Figs. 13(a)* to *13(e)*.


: R=0.98998
Data
Fit
Y = T
Output ~= 0.99*Target + -0.0074
Target

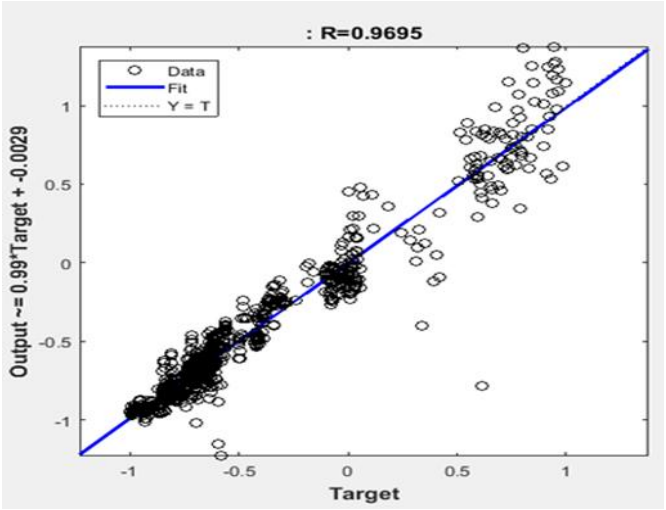
a.


: R=0.99564
Data
Fit
Y = T
Output ~= 0.99*Target + -0.0047
Target

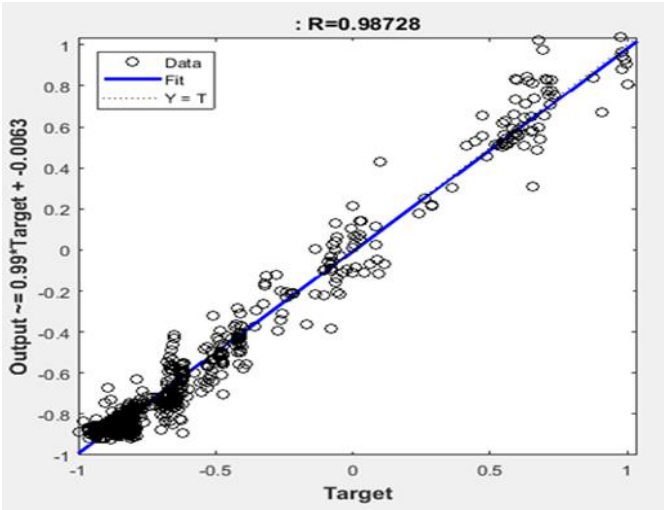
b.


: R=0.9951
Data
Fit
Y = T
Output ~= 1*Target + -0.0013
Target

c.



d.



e.

Fig. 13. ANN-PSO regression for five symbols: a. ANN-PSO regression (Fakhouz): b. ANN-PSO regression (Zagros): c. ANN-PSO regression (Khodro): d. ANN-PSO regression (Fameli): e. ANN-PSO regression (Kechad).

From the figures, it's evident that this algorithm exhibits strong predictive capabilities and a good fit. However, there is still room for improvement. *Figs. 13(a)* to *13(c)* exhibit the highest R-squared values, while the remaining figures do not perform as strongly.

Figs 13(d) and *13(e)* notably contain more data points that deviate further from the trend line, indicating some prediction variance.

Table 14 provides additional insights into the hybrid ANN-PSO model's training, validation, and testing. *Table 9* presents eight error measurements along with R-squared values and the identification of the best particle for each of the five companies. Specifically, the best particles are numbered 8, 8, 1, 9, and 10 for Fakhouz, Zagros, Khodro, Fameli, and Kechad, respectively.

Table 14. Hybrid ANN-PSO.

Symbol	MSE	RMSE	MAE	MAPE	MSRE	MARE	RMSRE	RMSPE	R^2	Best Particle
Fakhouz	3.06e-05	0.0005	0.005	0.0003	0.0018	3.17e-06	1.4e-05	0.001	0.989	8
Zagros	1.5e-05	0.0003	0.0002	0.0003	0.0038	3.05e-06	2.0e-05	0.02	0.995	8
Khodro	1.0e-05	0.0004	0.0004	-0.0001	0.0002	1.97e-06	4.0e-05	0.0042	0.995	1
Fameli	1.4e-05	0.0011	0.004	0.0007-	0.004	7.98e-06	2.0e-05	0.0002	0.969	9
Kechad	4.0e-05	0.0006	0.014	0.0019-	0.0351	2.0e-05	6.0e-06	0.006	0.987	10

4.5 | Hybrid HS-ANN

Like other algorithms, e.g., GA and PSO, several steps are followed for training the network and solving the problem. The network structure is Feed Forward ANN (FFANN). First, the number of iterations is 1000, and to achieve better results, it is increased to 5000. Finally, the result after 5,000 iterations is illustrated in *Table 15*.

Table 15. Hybrid ANN-HS.

Symbol	MSE	RMSE	MAE	MAPE	MSRE	MARE	RMSRE	RMSPE	R^2
Fakhouz	5.51e-07	0.0002	5.49e-15	6.88e-11	8.98e-07	6.86e-12	0.00002	0.002	0.9992
Zagros	2.56e-07	0.00005	5.10e-15	2.30e-11	4.08e-08	2.08e-13	0.000005	0.0005	0.9996
Khodro	3.09e-08	0.00001	4.70e-15	4.23e-11	6.19e-09	4.23e-13	0.000001	0.0001	0.9996
Fameli	1.37e-07	0.00003	5.55e-15	3.88e-11	1.22e-07	3.88e-13	0.000003	0.0003	0.9993
Kechad	3.05e-07	0.00005	5.99e-15	4.36e-11	1.4e-07	4.36e-13	0.000005	0.0005	0.9995

The results of *Table 15* show that HSA is a powerful algorithm in prediction because all of the R-squareds are high and close to 1. The rate of MSE and most of the other error measurements are better than PSO algorithms.

4.6 | Comparing Results

In this section, this study's obtained results are compared with similar studies. The comparison results are summarized in *Table 16*.

Table 16. Comparative study.

Author(s)	Proposed Approaches	Data Type	MSE	MAE	R2
Ghasemiyeh et al. [65]	GA-ANN	Train	0.0074	0.0584	0.9866
		Test	0.0079	0.0585	
	PSO-ANN	Train	0.0013	0.0253	0.9895
		Test	0.0014	0.0260	
	ICS-ANN	Train	0.0076	0.0720	0.9972
		Test	0.0068	0.0694	
Sedighi et al. [66]	ARIMA-SVM	Final outcome	1.0042	0.0142	0.9969
	SVM-RF	Final outcome	0.000295	0.0245	0.9966
	ANFIS-SVM	Final outcome	3.5849	0.0117	0.9995
	FA-MSVR	Final outcome	0.0014	0.0130	0.9986
Safa and Panahian [67]	HS-ANN	Final outcome	0.02776	0.05177	0.9641
Emamverdi et al. [68]	ANN	Final outcome	0.00030	0.0174	0.9893
	ARIMA	Final outcome	0.00042	0.0162	0.9795
Zheng et al. [69]	Wavelet Neural Networks (WNNs)	Final outcome	0.00510	6.742e-04	0.9877
Dong et al. [70]	One-step ahead and multi-step ahead predictions	Final outcome	0.0043	0.1043	0.9012
Wang et al. [71]	Delayed Neural Network (DNN)	Final outcome	1.60e-03	1.00e-07	0.9955
Sin and wang [72]	Ensembles of NN	Final outcome	2.05e-05	2.045e-09	0.9963
Current research	ANN	Train	0.01768	0.036408	0.9973
		Test	0.06578	0.00621	
	GA-ANN	Train	0.00070	0.0130	0.9984
		Test	0.00045	0.000532	
	PSO-ANN	Train	0.000042	0.00392	0.9993
		Test	0.00431	0.000216	
	HS-ANN	Train	3.0258E-07	5.366E-15	0.9995
		Test	0.000061402	0.00042	

As per *Table 11*, the HS and, to a similar extent, the PSO algorithms yield the best results with the lowest loss functions. These results are based on each algorithm's average R-squared values, indicating their effectiveness in predictive performance.

For further insights and detailed information regarding the number of layers in each input and hidden layer, activation functions, and other related details, you can refer to *Table B6* in the appendix.

4.7 | EMH Testing

First, the series should be checked to determine whether they follow a normal distribution. Since the sample is more than 100, the Kolmogorov-Smirnov normality test is used separately for each symbol.

Table 17. Test of normality.

Fakhouz						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Adj.Closing Price;	.209	923	.000	.752	923	.000
Zagros						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Adj.Closing Price;	.234	963	.000	.711	963	.000
Khodro						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Adj.Closing Price;	.097	1086	.000	.963	1086	.000
Fameli						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Adj.Closing Price;	.242	997	.000	.793	997	.000
Kechad						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Adj.Closing Price;	.266	973	.000	.650	973	.000
a. Lilliefors significance correction						

As it is clear, the significance of all symbols is less than 5%, which means the series is not normal. So, the non-parametric test is used for checking EMH.

Table 18. Run test.

Fakhouz	
	Adj.Closing Price;
Test value[a]	-.57223850
Cases < test value	620
Cases >= test value	303
Total cases	923
Number of runs	2
Z	-30.327
Asymp. Sig. (2-tailed)	.000

Table 18. Continued.

Zagros	
	Adj.Closing Price;
Test value[a]	-.71050612
Cases < test value	680
Cases >= test value	283
Total cases	963
Number of runs	12
Z	-30.200
Asymp. Sig. (2-tailed)	.000
Khodro	
	Adj.Closing Price;
Test value[a]	-.16692865
Cases < test value	504
Cases >= test value	582
Total cases	1086
Number of runs	12
Z	-32.299
Asymp. Sig. (2-tailed)	.000
Fameli	
	Adj.Closing Price;
Test value[a]	-.52997966090
Cases < test value	698
Cases >= test value	299
Total cases	997
Number of runs	2
Z	-31.522
Asymp. Sig. (2-tailed)	.000
Kechad	
	Adj.Closing Price;
Test value[a]	-.64985740
Cases < test value	716
Cases >= test value	257
Total cases	973
Number of runs	9
Z	-30.558
Asymp. Sig. (2-tailed)	.000
a. Mean	

In all the tables, the sigma value is consistently less than 0.05, indicating that the data is not random. It implies that the market is not efficient. In inefficient markets, crucial information that impacts asset prices is not readily available. Consequently, it becomes challenging to determine or predict asset prices accurately. This situation can result in financial holdings not reflecting their true value, contributing to a weak market.

5 | Conclusion

This paper employs a neural network-based approach to predict stock prices for five symbols representing various industries: Fakhouz, Zagros, Khodro, Fameli, and Kechad. The process involves selecting essential technical indicators, such as SMA, EMA, TMA, etc., as input variables using GAs. Subsequently, the NN is trained using two meta-heuristic algorithms, HS and PSO. Different loss functions are computed for each algorithm after optimizing the indicators and weights through GA. *Table (B7)* in the appendix presents each algorithm's comprehensive loss function values. HS exhibits the lowest training and testing errors, while ANN demonstrates the highest forecasting error. Evaluating model performance with new data, referred to as testing performance, proves to be a reliable indicator of predictive performance.

The primary advantages of HS and PSO include speeding up calculations, reducing model complexity, increasing network accuracy, and facilitating model utilization. Ultimately, HS, PSO, and ANN yield the lowest error rates. Given the market's inefficiency, past information significantly influences future stock prices, allowing for accurate stock price forecasting.

In summary, the most significant finding is the high predictability of ANN. The results underscore the effectiveness of GA as a feature selection method, improving model robustness. Moreover, employing meta-heuristic algorithms like PSO and HS for optimization enhances R-squared values or reduces error estimates.

Key considerations for future research include investigating different parameters, such as the number of hidden layers, activation functions, and alternative HS models like HIS. Additionally, fine-tuning GA parameters, such as crossover and mutation rates, holds potential interest. Lastly, future studies should explore other novel meta-heuristic algorithms like Aquila Optimization Algorithm (AO) and Bald Eagle Search (BES) optimization algorithms.

Acknowledgments

We would like to express our special gratitude to our parents and the God we trust.

Funding

Any organization does not fund this paper.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Mehwish, N., & Yasir, B. T. (2015). The efficient market hypothesis: a critical review of the literature. *The IUP journal of financial risk management*, *12*(4), 48–63.

[2] Guerrien, B., & Gun, O. (2011). Efficient market hypothesis : what are we talking about ? *Analysis*, (56), 19–30. http://paecon.net/PAEReview/issue56/GuerrienGun56.pdf

[3] Bhowmik, P. (2019). Research study on basic understanding of artificial neural networks. *Global journal of computer science and technology*, *19*(4), 5–7.

[4] Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: statistical arbitrage on the S&P 500. *European journal of operational research*, *259*(2), 689–702.

[5] Caginalp, G., & Desantis, M. (2011). Nonlinearity in the dynamics of financial markets. *Nonlinear analysis: real world applications*, *12*(2), 1140–1151. DOI:10.1016/j.nonrwa.2010.09.008

[6] Khashei, M., & Bijari, M. (2011). A novel hybridization of artificial neural networks and ARIMA models for time series forecasting. *Applied soft computing journal*, *11*(2), 2664–2675.

[7] De Oliveira, F. A., Nobre, C. N., & Zárate, L. E. (2013). Applying artificial neural networks to prediction of stock price and improvement of the directional prediction index - case study of PETR4, Petrobras, Brazil. *Expert systems with applications*, *40*(18), 7596–7606. DOI:10.1016/j.eswa.2013.06.071

[8] Abolfazli, N., Eshghali, M., & Fatemi Ghomi, S. M. T. F. (2022). Pricing and coordination strategy for green supply chain under two production modes. *IEEE 2022 systems and information engineering design symposium, sieds 2022* (pp. 13–18). IEEE. DOI: 10.1109/SIEDS55548.2022.9799373

[9] Zhang, J., Cui, S., Xu, Y., Li, Q., & Li, T. (2018). A novel data-driven stock price trend prediction system. *Expert systems with applications*, *97*, 60–69. DOI:10.1016/j.eswa.2017.12.026

[10] Jóhannsson, Ó. S. (2020). *Forecasting the icelandic stock market using a neural network* (Master Thesis, Reykjavík). https://skemman.is/bitstream/1946/36421/1/MSC_OMAR_2020.pdf

[11] Hadavandi, E., Ghanbari, A., & Abbasian-Naghneh, S. (2010). *Developing an evolutionary neural network model for stock index forecasting. dvanced intelligent computing theories and applications. ICIC 2010. communications in computer and information science.* (Vol. 93 CCIS, pp. 407–415). Springer, Berlin, Heidelberg. DOI: 10.1007/978-3-642-14831-6_54

[12] Hull, J. C. (2012). *Risk management and financial institutions, + Web Site* (Vol. 733). John Wiley & Sons. https://www.google.com/books/edition/

[13] Prasanna, S. (2013). An analysis on stock market prediction using data mining techniques. *International journal of computer science & engineering technology (IJCSET)*, *4*(02), 49–51.

[14] Fama, E. F. (1998). Market efficiency, long-term returns, and behavioral finance. *Journal of financial economics*, *49*(3), 283–306.

[15] Gimba, V. K. (2012). Testing the weak-form efficiency market hypothesis: evidence from nigerian stock market. *CBN journal of applied statistics*, *3*(1), 117–136.

[16] Fama, E. F. (1965). The behavior of stock-market prices. *The journal of business*, *38*(1), 34–105.

[17] Dixit, A., Mishra, A., & Shukla, A. (2019). Vehicle routing problem with time windows using meta-heuristic algorithms: A survey. In *Advances in intelligent systems and computing* (Vol. 741, pp. 539–546). Springer. DOI: 10.1007/978-981-13-0761-4_52

[18] Dramsch, J. S. (2020). 70 Years of Machine learning in geoscience in review. *Advances in geophysics*, *61*, 1–55. DOI:10.1016/bs.agph.2020.08.002

[19] Sindayigaya, L., & Dey, A. (2022). Machine learning algorithms: a review. *International journal of science and research (IJSR), 11*(8), 1127-1133.

[20] Iuhasz, G., Tirea, M., & Negru, V. (2012). Neural network predictions of stock price fluctuations. *Proceedings - 14th international symposium on symbolic and numeric algorithms for scientific computing, synasc 2012* (pp. 505–512). IEEE. DOI: 10.1109/SYNASC.2012.7

[21] Huang, W., Lai, K. K., Nakamori, Y., Wang, S., & Yu, L. (2007). Neural networks in finance and economics forecasting. *International journal of information technology & decision making*, *6*(01), 113–140.

[22] Göçken, M., Özçalici, M., Boru, A., & Dosdołru, A. T. (2016). Integrating metaheuristics and artificial neural networks for improved stock price prediction. *Expert systems with applications*, *44*, 320–331.

[23] Hassanin, M. F., Shoeb, A. M., & Hassanien, A. E. (2016). *Grey wolf optimizer-based back-propagation neural network algorithm.* 2016 12th international computer engineering conference (ICENCO) (pp. 213–218). IEEE. https://ieeexplore.ieee.org/abstract/document/7856471

[24] Faris, H., Aljarah, I., & Mirjalili, S. (2016). Training feedforward neural networks using multi-verse optimizer for binary classification problems. *Applied intelligence*, *45*(2), 322–332. DOI:10.1007/s10489-016-0767-1

[25] Rather, A. M., Sastry, V. N., & Agarwal, A. (2017). Stock market prediction and Portfolio selection models: a survey. *Opsearch*, *54*(3), 558–579. DOI:10.1007/s12597-016-0289-y

[26] Yang, B., Gong, Z. J., & Yang, W. (2017). *Stock market index prediction using deep neural network ensemble*. 2017 36th chinese control conference (CCC) (pp. 3882–3887). IEEE. DOI: 10.23919/ChiCC.2017.8027964

[27] Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: bagging, boosting, and hybrid-based approaches. *IEEE transactions on systems, man and cybernetics part C: applications and reviews*, *42*(4), 463–484. DOI:10.1109/TSMCC.2011.2161285

[28] Kabir Ahmed, M., Maksha Wajiga, G., Vachaku Blamah, N., & Modi, B. (2019). Stock market forecasting using ant colony optimization based algorithm. *American journal of mathematical and computer modelling*, *4*(3), 52. DOI:10.11648/j.ajmcm.20190403.11

[29] Ghanbari, M., & Arian, H. (2019). *Forecasting stock market with support vector regression and butterfly optimization algorithm*. https://doi.org/10.48550/arXiv.1905.11462

[30] Chandana, P. H. (2019). A survey on soft computing techniques and applications. *International research journal of engineering and technology (IRJET) 6*(4), 1258–1266.

[31] Rajesh, P., Srinivas, N., Vamshikrishna Reddy, K., Vamsipriya, G., Vakula Dwija, M., & Himaja, D. (2019). Stock trend prediction using ensemble learning techniques in python. *International journal of innovative technology and exploring engineering*, *8*(5), 150–154.

[32] Lv, D., Yuan, S., Li, M., & Xiang, Y. (2019). An empirical study of machine learning algorithms for stock daily trading strategy. *Mathematical problems in engineering*, *2019*. DOI:10.1155/2019/7816154

[33] Zaman, S. (2019). Weak form market efficiency test of Bangladesh stock exchange: an empirical evidence from dhaka stock exchange and chittagong stock exchange. *Journal of economics, business and accountancy ventura*, *21*(3), 285. DOI:10.14414/jebav.v21i3.1615

[34] Shahvaroughi Farahani, M., & Razavi Hajiagha, S. H. (2021). Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. *Soft computing*, *25*(13), 8483–8513. DOI:10.1007/s00500-021-05775-5

[35] Ranjbarzadeh, R., Caputo, A., Tirkolaee, E. B., Jafarzadeh Ghoushchi, S., & Bendechache, M. (2023). Brain tumor segmentation of MRI images: a comprehensive review on the application of artificial intelligence tools. *Computers in biology and medicine*, *152*, 106405. DOI:10.1016/j.compbiomed.2022.106405

[36] Farahani, M. S., Esfahani, A., & Alipoor, F. (2022). The application of machine learning in the corona era, with an emphasis on economic concepts and sustainable development goals. *International journal of mathematical, engineering, biological and applied computing*, *1*(2), 95–149. DOI:10.31586/ijmebac.2022.519

[37] Tirkolaee, E. B., Mardani, A., Dashtian, Z., Soltani, M., & Weber, G. W. (2020). A novel hybrid method using fuzzy decision making and multi-objective programming for sustainable-reliable supplier selection in two-echelon supply chain design. *Journal of cleaner production*, *250*, 119517. DOI:10.1016/j.jclepro.2019.119517

[38] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A. E., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: a survey. *Heliyon*, *4*(11). DOI:10.1016/j.heliyon.2018.e00938

[39] Tkáč, M., & Verner, R. (2016). Artificial neural networks in business: two decades of research. *Applied soft computing journal*, *38*, 788–804. DOI:10.1016/j.asoc.2015.09.040

[40] Pierdzioch, C., & Risse, M. (2018). A machine-learning analysis of the rationality of aggregate stock market forecasts. *International journal of finance and economics*, *23*(4), 642–654. DOI:10.1002/ijfe.1641

[41] Zhong, X., & Enke, D. (2019). Predicting the daily return direction of the stock market using hybrid machine learning algorithms. *Financial innovation*, *5*(1), 1–20. DOI:10.1186/s40854-019-0138-0

[42] Altan, A., Karasu, S., & Bekiros, S. (2019). Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques. *Chaos, solitons and fractals*, *126*, 325–336. DOI:10.1016/j.chaos.2019.07.011

[43] Jiang, M., Jia, L., Chen, Z., & Chen, W. (2022). The two-stage machine learning ensemble models for stock price prediction by combining mode decomposition, extreme learning machine and improved harmony search algorithm. *Annals of operations research*, *309*(2), 553–585. DOI:10.1007/s10479-020-03690-w

[44] Behravan, I., & Razavi, S. M. (2020). Stock price prediction using machine learning and swarm intelligence. *Journal of electrical and computer engineering innovations (JECEI)*, *8*(1), 31–40.

[45] Kumar Chandar, S. (2021). Grey Wolf optimization-elman neural network model for stock price prediction. *Soft computing*, *25*(1), 649–658. DOI:10.1007/s00500-020-05174-2

[46] Wei, L. Y., & Cheng, C. H. (2012). A hybrid recurrent neural networks model based on synthesis features to forecast the Taiwan stock market. *International journal of innovative computing, information and control*, *8*(8), 5559–5571.

[47] Chandar, S. Kumar. (2021). Hybrid models for intraday stock price forecasting based on artificial neural networks and metaheuristic algorithms. *Pattern recognition letters*, *147*, 124–133. DOI:10.1016/j.patrec.2021.03.030

[48] Accounting, M., Farahani, M. S., Nejad, M., Moghaddam, F., & Ramezani, A. (2023). Forecasting Tehran price index (TEPIX) using novel meta-heuristic algorithms keywords. *International journal of finance & managerial accounting*, *8*(28), 185–216.

[49] Monfared, J. H., Alinejad, M. A., & Metghalchi, S. (2012). A comparative study of neural network models with box Jenkins methodologies in prediction of Tehran price index (TEPIX). *Financial engineering and securities management (portfolio management)*, 3(11), 1-16. https://sid.ir/paper/197900/en

[50] Ahmed, J., Jafri, M. N., Ahmad, J., & Khan, M. I. (2007). Design and implementation of a neural network for real-time object tracking. *International journal of computer and information engineering*, *1*(6), 1816–1819.

[51] Haider, A., & Hanif, M. N. (2009). Inflation forecasting in pakistan using artificial neural networks. *Pakistan economic and social review*, *47*(2), 123–138. https://www.jstor.org/stable/25825345

[52] Chopra, S., Yadav, D., & Chopra, A. N. (2019). Artificial neural networks based indian stock market price prediction : before and after demonetization. *International journal of swarm intelligence and evolutionary computation*, *8*(1), 1–7.

[53] Ravichandran, K. S., Thirunavukarasu, P., Nallaswamy, R., & Babu, R. (2005). Estimation of return on investment in. *Journal of theoretical and applied information technology*, *3*, 44–54.

[54] Hawaldar, I. T., Rohit, B., & Pinto, P. (2017). Testing of weak form of efficient market hypothesis: evidence from the Bahrain bourse. *Investment management and financial innovations*, *14*(2), 376–385. DOI:10.21511/imfi.14(2-2).2017.09

[55] 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. DOI:10.1080/01969722.2017.1285162

[56] Davallou, M., & Azizi, N. (2017). The investigation of information risk pricing; evidence from adjusted probability of informed trading measure. *Financial research journal*, *19*(3), 415–438.

[57] Eberhart, R., & Kennedy, J. (1995). *New optimizer using particle swarm theory*. IEEE Proceedings of the international symposium on micro machine and human science (pp. 39–43). IEEE. DOI: 10.1109/mhs.1995.494215

[58] Yang, X. S. (2020). Nature-inspired optimization algorithms: challenges and open problems. *Journal of computational science*, *46*, 101104. https://doi.org/10.1016/j.jocs.2020.101104

[59] Al Masud, M. A., Paul, S. K., & Azeem, A. (2014). Optimisation of a production inventory model with reliability considerations. *International journal of logistics systems and management*, *17*(1), 22–45.

[60] Ye, F. (2017). Particle swarm optimization-based automatic parameter selection for deep neural networks and its applications in large-scale and high-dimensional data. *PLoS one*, *12*(12), 1-36.

[61] Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: harmony search. *Simulation*, *76*(2), 60–68. DOI:10.1177/003754970107600201

[62] Kang, J., Kwon, S., Ryu, D., & Baik, J. (2021). HASPO: harmony search-based parameter optimization for just-in-time software defect prediction in maritime software. *Applied sciences*, *11*(5), 2002. DOI:10.3390/app11052002

[63] Leković, M. (2018). Evidence for and against the validity of efficient market hypothesis. *Economic themes*, *56*(3), 369–387. DOI:10.2478/ethemes-2018-0022

[64] Pervez, M., Harun Ur Rashid, M., Asad Iqbal Chowdhury, M., & Rahaman, M. (2018). International journal of economics and financial issues predicting the stock market efficiency in weak form: a study on dhaka stock exchange. *International journal of economics and financial issues*, *8*(5), 88–95.

[65] 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.

[66] Sedighi, M., Jahangirnia, H., Gharakhani, M., & Fard, S. F. (2019). A novel hybrid model for stock price forecasting based on metaheuristics and support vector machine. *Data*, *4*(2), 75. https://doi.org/10.3390/data4020075

[67] Safa, M., & Panahian, H. (2019). Ranking P/E predictor factors in Tehran stock exchange with using the harmony search meta heuristic algorithm. *Journal of investment knowledge*, *8*(29), 67–82.

[68] Emamverdi, G., Karimi, M. S., Khakie, S., & Karimi, M. (2016). Forecasting the total index of tehran stock exchange. *Financial studies*, *20*(1), 55–68.

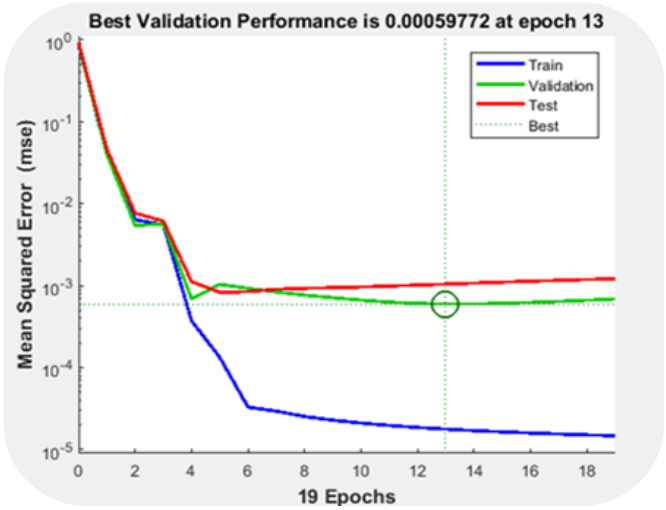
[69] Zheng, T., Fataliyev, K., & Wang, L. (2013). *Wavelet neural networks for stock trading* [presentation]. Independent component analyses, compressive sampling, wavelets, neural net, biosystems, and nanoengineering xi (Vol. 8750, pp. 83–92).

[70] Dong, Guanqun., Kamaladdin, F., & Wang, L. (2013). One-step and multi-step ahead stock prediction using backpropagation neural networks. *ICICS 2013 - conference guide of the 9th international conference on information, communications and signal processing* (pp. 1–5). IEEE. DOI: 10.1109/ICICS.2013.6782784

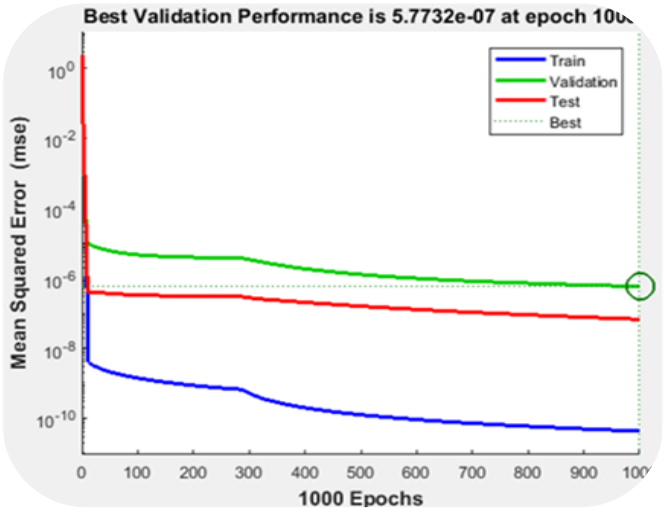
[71] Wang, L., Chan, F. F., Wang, Y., & Chang, Q. (2016). *Predicting public housing prices using delayed neural networks* (pp. 3589–3592). IEEE.

[72] Sin, E., & Wang, L. (2017). Bitcoin price prediction using ensembles of neural networks. *2017 13th international conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD)* (pp. 666–671). IEEE.

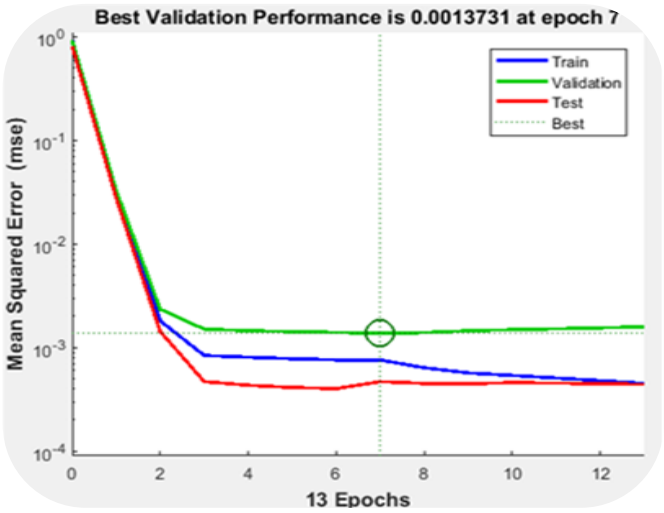
Appendix



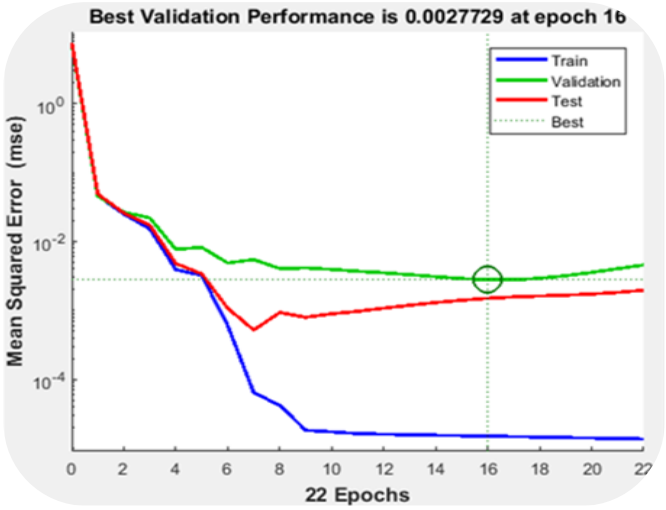
a.



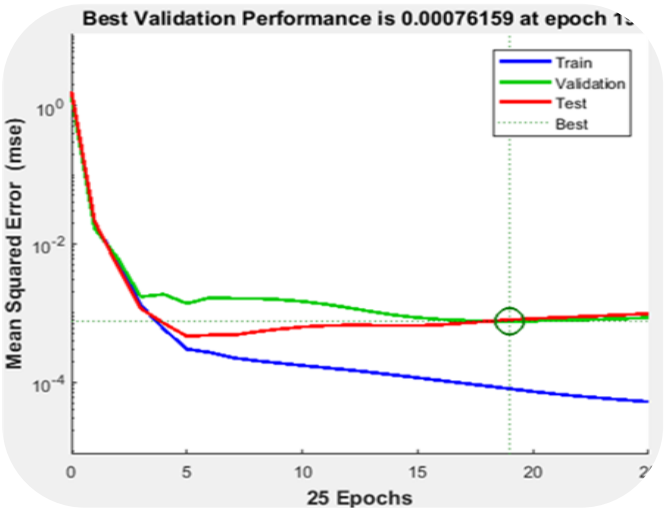
b.



c.



d.



e.

Fig. A1. a. the best validation performance(Fakhouz); b. the best validation performance(Zagros); c. the best validation performance(Khodro); d. the best validation performance(Fameli); e. the best validation performance(Kechad).

Table B1. Limitations of the previous methods.

No	Methods	Purpose	Limitations
1	ARIMA[1]	Forecasting and clustering	-Incompatible with nonlinear time series -Working with a large data set and more data - Decreasing speed with a larger data - Susceptive to noise
2	BPNN[2]	Forecasting	- Actual performance depends on initial values - Slow convergent speed - High possibility of local minima/maxima trap
3	CART[3]	Classification and forecasting	- The stability of the network will be affected by a little change in training data
4	GP[4]	Classification and forecasting	-Generates" black box" models which are difficult to interpret. -This type of calculation can be expensive
5	GRNN[5]	Classification and forecasting	- This method needs more memory space to store the model - This type of calculation can be expensive because of its huge size

[1] Autoregressive Integrated Moving Average (ARIMA) model

[2] Back Propagation Neural Network (BPNN)

[3] Classification and Regression Trees (CART)

[4] Gaussian Process (GP)

[5] Generalized Regression Neural Network (GRNN)

Table B1. Continued.

No	Methods	Purpose	Limitations
6	Hierarchical clustering	Clustering	-The length of each time series is the same because of the euclidean distance - Compatible with small datasets because of its quadratic computational complexity
7	HMM[1]	Clustering, classification and clustering	- Very sensitive due to personal tuning - Sensitive to a larger dataset. So, the processing may take a longer time. - Needs to determine the number of clusters in advance
8	K-Mean	clustering	- Sensitive to noise - Limitations and shortage in clustering - Due to poor scalability, incompatible and unable to handle long-time series
9	KNN[2]	Classification and forecasting	- Needs to determine the number of nearest neighbors at first - This type of calculation can be expensive - Memory constraint/restriction - Network performance depends on the local structure of the data
10	Logistic Regression (LR)	Classification and forecasting	- Sensitive to outliers - Determinative assumptions
11	LSTM[3]	Classification and forecasting	- The high possibility of overfitting - Sensitive to initial random weights - Needs more memory to train
12	MLP[4]	Classification and forecasting	-Almost slow convergence - be sensitive to local minima while it can affect the performance. - Hard to scale
13	PSO[5]	Forecasting	- The possibility of local optimum increases in high dimensional space - Low convergence rate
14	RBF[6]	Classification and forecasting	-Slower classification process of this method proportional to MLP
15	RF[7]	Classification and forecasting	- Complicated calculations due to the creation of trees - Decision trees require less time via RF.
16	RNN[8]	Classification and forecasting	-Hardness of training
17	SOM[9]	Clustering and classification	- Lack of performance power against time series of unequal length because of the difficulty involved in determining the scale of weight vectors - Sensitive to outliers
18	SVM[1]	Classification and forecasting	-Sensitive to outliers - Sensitive to tuning
19	SVR[1]	Forecasting	- Sensitive to personal tuning and setting free parameters
20	ANN[1]	Classification and forecasting	- Overfitting - Local minima/maxima trap - Training and the network performance depend on parameter setting and tuning - Limited exploration and exploitation, which leads to the use of the optimization algorithm - Sensitive to data structure

[1] Hidden Markov Model (HMM)
[2] K Nearest Neighbor (KNN)
[3] Long Short-Term Memory (LSTM)
[4] Multi-Layer Perceptron (MLP)
[5] Particle Swarm Optimization (PSO)
[6] Radial Basis Function (RBF)
[7] Random Forest (RF)
[8] Recurrent Neural Network (RNN)
[9] Self-Optimizing Maps (SOM)
[10] Support Vector Machine (SVM)
[11] Support Vector Regression (SVR)
[12] Artificial Neural Network (ANN)

Table B2. Loss function for ANN before and after using GA.

Error Before Using GA				Error After Using GA		
Fakhouz	**Training**	**Selection**	**Testing**	**Training**	**Selection**	**Testing**
SSE	1246.79	409.51	403.57	103.5	37.39	41.56
MSE	2.2464	2.2256	2.1933	0.186	0.203	0.22
RMSE	1.4988	1.4918	1.4809	0.431	0.4508	0.4752
NSE	12.59	11.6286	12.7699	1.07	1.04	1.23
ME	989.909	325.64	322.606	135.33	47.35	51.24
Zagros	**Training**	**Selection**	**Testing**	**Training**	**Selection**	**Testing**
SSE	263.69	88.36	82.2	91.26	31.83	41.67
MSE	0.4602	0.465	0.432	0.152	0.167	0.219
RMSE	0.6783	0.6819	0.657	0.399	0.409	0.468
NSE	2.7339	2.9913	2.46	1.03	1.04	1.02
ME	296.79	99.3	94.43	120.3	40.97	49.27
Khodro	**Training**	**Selection**	**Testing**	**Training**	**Selection**	**Testing**
SSE	721.35	244.33	244.14	220.23	71.52	61.01
MSE	1.109	1.13	1.13	0.338	0.33	0.28
RMSE	1.05	1.06	1.06	0.582	0.575	0.53
NSE	6.29	6.14	6.27	1.85	1.78	1.76
ME	690.2	232.66	232.2	253.9	82.3	72.11
Fameli	**Training**	**Selection**	**Testing**	**Training**	**Selection**	**Testing**
SSE	230.63	82.51	52.61	221.22	71.89	57.8
MSE	0.38	0.416	0.265	0.37	0.36	0.29
RMSE	0.621	0.645	0.5154	0.608	0.6	0.54
NSE	1.62	1.68	1.67	1.52	1.6	1.74
ME	224.09	79.94	56.39	238.5	77.94	66.19
Kechad	**Training**	**Selection**	**Testing**	**Training**	**Selection**	**Testing**
SSE	300.7	99.11	99.25	1594.5	522.9	511.08
MSE	0.51	0.51	0.51	2.72	2.69	2.63
RMSE	0.717	0.714	0.71	1.65	1.64	1.62
NSE	2.91	3.64	2.75	17.27	15.68	12.51
ME	345.05	114.37	114.1	12.14	397.48	389.4

Table B3. ANN-GA structure for determining weight.

Symbol	Structure	Weight	Tra… Error	Val… Error	Tes… Error	AIC	Correlation	R^2	Tra… Error (AE)	Val… Error (AE)	Itr	Tra… Algo
Fakhouz	19-50-1	1101	0.045	0.04	0.044	-2987	0.997	0.96	0.066	0.064	7	LM
Zagros	16-40-1	721	0.063	0.06	0.061	-3724	0.996	0.96	0.067	0.066	7	LM
Khodro	11-17-1	222	0.057	0.06	0.060	-6022	0.99	0.97	0.055	0.059	6	LM
Fameli	15-19-1	343	0.054	0.05	0.052	-4976	0.99	0.97	0.059	0.06	12	LM
Kechad	16-25-1	451	0.06	0.06	0.06	-4430	0.99	0.94	0.06	0.06	8	LM

Table B4. Selection of most important technical indicators using GA.

Symbol	Technical Indicators	Selected Using GA	Selected (Deleted)
Fakhouz	Open, High, Low, WC, EMA (5), EMA (6), EMA (10), EMA (20), MACD, RS, Lowest Low, %K, SMA (5), SMA (6), SMA (10), SMA (20), TMA (5), TMA (6), TMA (10), TMA (20), Diff, MOpen, MHigh, MLow, MClose, AccOpen, AccHigh, AccLow, AccClose, AccDist, Fast %K, Fast %D, Slow %K, Slow %D, %R, RSI, Middle Band, Upper Band, Lower Band, MP, ROC, TP	Open, High, Low, WC, EMA (6), MACD, RS, Lowest Low, %K, Upper Band, SMA(5), SMA(10), SMA(20), TMA(20), MLow, AccClose, %R	(25)-17

Table B4. Continued.

Symbol	Technical Indicators	Selected Using GA	Selected (Deleted)
Zagros		Low, MP, MOpen SMA (5), EMA (5), EMA (10), EMA (20), EMA (6), MACD, TP, Highest High, ROC, SMA (3), %R	(26)-16
Khodro		High, ROC, %R, EMA (20), EMA (6), EMA (10), Lowest Low, SMA (20), TMA (5), TMA (6), TMA (20)	(31)-11
Fameli		Low, EMA (3), EMA (5), EMA (20), RS, RSI, Highest High, Lowest Low, %D, ROC, SMA (5), SMA (6), SMA (10), TMA (5)	(27)-15
Kechad		Open, Upper Band, Lower Band, %D, %R, Highest High, Lowest Low, RS, SMA (5), SMA (6), TMA (5), TMA (6), TMA (10), Mhigh, Mclose, AccClose	(26)-16

Table B5. The most important selected technical indicators.

LOW	EMA (6)	EMA (20)	SMA (5)	SMA (6)	SMA (20)	TMA (20)
ROC	RSI	R%	MACD	Highest high	Lowest low	TMA (5)

Table B6. Parameters of the models.

Symbol Fakhouz	Network Element Parameters	Algorithm ANN	GA-ANN	PSO-ANN	HS-ANN
	Input layers	42	19	19	19
	Hidden layers	10	50	20	6
	The activation function in the hidden layer	Tan-Sigmoid	Logistic	Tan-Sigmoid	Tan-Sigmoid
	Activation function in the output layer	Simple Linear	Simple Linear	Simple Linear	Simple Linear
	Training function	LM	LM	LM	LM
Zagros	**Parameters**	**ANN**	**GA-ANN**	**PSO-ANN**	**HS-ANN**
	Input layers	42	16	16	16
	Hidden layers	10	40	15	15
	The activation function in the hidden layer	Tan-Sigmoid	Logistic	Tan-Sigmoid	Tan-Sigmoid
	Activation function in the output layer	Simple Linear	Simple Linear	Simple Linear	Simple Linear
	Training Function	LM	LM	LM	LM
Khodro	**Parameters**	**ANN**	**GA-ANN**	**PSO-ANN**	**HS-ANN**
	Input layers	42	11	11	11
	Hidden layers	10	17	12	14
	The activation function in the hidden layer	Tan-Sigmoid	Logistic	Tan-Sigmoid	Tan-Sigmoid
	Activation function in the output layer	Simple Linear	Simple Linear	Simple Linear	Simple linear
	Training function	LM	LM	LM	LM

Table B6. Continued.

Fameli	Parameters	ANN	GA-ANN	PSO-ANN	HS-ANN
	Input layers	42	15	15	15
	Hidden layers	10	19	15	4
	The activation function in the hidden layer	Tan-Sigmoid	Logistic	Tan-Sigmoid	Tan-Sigmoid
	Activation function in the output layer	Simple Linear	Simple Linear	Simple Linear	Simple Linear
	Training function	LM	LM	LM	LM
Kechad	**Parameters**	**ANN**	**GA-ANN**	**PSO-ANN**	**HS-ANN**
	Input layers	42	16	16	16
	Hidden layers	10	25	16	15
	The activation function in the hidden layer	Tan-Sigmoid	Logistic	Tan-Sigmoid	Tan-Sigmoid
	Activation function in the output layer	Simple Linear	Simple Linear	Simple Linear	Simple Linear
	Training function	LM	LM	LM	LM

Table B7. Evaluation criteria error for ANN, GA-ANN, PSO-ANN and HS-ANN.

SAE	SSE	RMSPE	RMSRE	MARE	MSRE	MAPE	MAE	RMSE	MSE	Fakhouz
4.837	0.1087						0.52		0.003	Tr error ANN
6.99	0.2297						0.0084		0.000301	Tr error GA-ANN
		0.0014	0.000014	3.18E-06	-0.0018	0.000318	0.005	0.00059	0.000036	Tr error PSO-ANN
		0.0021	0.00002	-6.9E-10	8.98E-05	-6.7E-08	-5.5E-15	0.0002	5.51E-07	Tr error HS-ANN
		0.02138	0.000213	0.00117	0.00329	-0.00023	0.0057	0.0096	0.04527	Tes error ANN
		0.0036	0.000036	3.61E-06	7.98E-07	0.000761	-7.1E-06	8.42E-05	0.000295	Tes error GA-ANN
		0.096	0.00096	0.01	0.00049	0.111	-0.00068	0.0096	0.0093	Tes error PSO-ANN
		0.013	0.00011	-0.0013	0.00006	-0.1305	0.0008	0.0011	0.00012	Tes error HS-ANN
SAE	**SSE**	**RMSPE**	**RMSRE**	**MARE**	**MSRE**	**MAPE**	**MAE**	**RMSE**	**MSE**	**Zagros**
4.837	0.1087						0.003		4.02E-08	Tr error ANN
6.112	0.1798						0.0064		0.000202	Tr error GA-ANN
		0.02	0.00002	3.05E-06	-0.0038	0.0003	0.00024	0.00038	0.000015	Tr error PSO-ANN
		0.0005	5.10E-06	-2.08E-11	4.08E-08	-2.07E-09	-5.10E-15	0.00005	2.56E-07	Tr error HS-ANN
		5.80E-03	5.81E-05	-2.38E-06	1.96E-07	1.30E-03	6.33E-03	4.31E-02	7.01E-04	Tes error ANN
		2.00E-02	2.00E-04	-1.39E-03	1.70E-04	4.40E-02	-2.56E-03	7.12E-05	5.08E-09	Tes error GA-ANN
		0.51	0.00051	0.00043	0.0065	-0.04	0.00036	0.0051	0.0026	Tes error PSO-ANN
		0.0113	0.00011	-0.0013	0.00006	-0.1305	0.0008	0.0012	0.00014	Tes error HS-ANN
SAE	**SSE**	**RMSPE**	**RMSRE**	**MARE**	**MSRE**	**MAPE**	**MAE**	**RMSE**	**MSE**	**Khodro**
18.86	0.878						0.0174		0.076	Tr error ANN
28.46	1.7331						0.0263		0.0013	Tr error GA-ANN
		4.20E-03	4.00E-05	-1.97E-04	-0.0002	-0.00019	0.0004	0.0004	1.00E-05	Tr error PSO-ANN
		0.0001	1.00E-06	-4.23E-13	6.19E-09	-4.23E-11	-4.70E-15	0.00001	3.09E-08	Tr error HS-ANN
		4.20E-02	4.20E-03	-3.37E-03	5.10E-03	-3.10E-02	1.65E-03	7.40E-03	4.69E-02	Tes error ANN
		3.00E-01	3.00E-03	-8.19E-05	6.00E-03	1.20E-02	-6.42E-05	2.80E-04	1.50E-03	Tes error GA-ANN
		0.068	0.00068	0.0008	0.0003	0.087	-0.00038	0.0068	0.0046	Tes error PSO-ANN
		0.002	0.000026	0.0004	0.00007	0.0421	-0.0001	0.00026	7.01E-06	Tes error HS-ANN
SAE	**SSE**	**RMSPE**	**RMSRE**	**MARE**	**MSRE**	**MAPE**	**MAE**	**RMSE**	**MSE**	**Fameli**
5.45	0.6505						0.55		0.0014	Tr error ANN
15.22	1.224						0.0153		0.0015	Tr error GA-ANN
		2.00E-04	2.00E-05	-7.98E-06	-0.004	-0.0007	0.004	0.0011	1.40E-04	Tr error PSO-ANN
		0.0003	3.00E-06	-3.88E-13	1.22E-07	-3.88E-11	-5.55E-15	0.00003	3.70E-07	Tr error HS-ANN
		1.00E-02	1.00E-04	1.42E+00	1.80E-05	-9.00E-02	-1.23E-03	4.32E-03	1.57E-01	Tes error ANN
		1.00E-02	1.00E-04	-3.16E-03	1.00E-03	7.50E-03	-1.91E-05			Tes error GA-ANN
	1.00E-03	4.75E-04								Tes error PSO-ANN
		0.029	0.00029	0.0003	0.0039	-0.03	-0.00068	0.0029	0.00086	Tes error HS-ANN
		0.00615	0.00006	-0.00065	0.00017	-0.0653	0.0004	0.000615	0.00003	Kechad
SAE	**SSE**	**RMSPE**	**RMSRE**	**MARE**	**MSRE**	**MAPE**	**MAE**	**RMSE**	**MSE**	**Tr error ANN**
7.123	0.2817						0.73		0.008	Tr error GA-ANN
8.522	0.5108						0.0088		0.000246	Tr error PSO-ANN
		6.00E-03	6.00E-05	-2.00E-05	-0.0351	-0.0019	0.01	0.0006	4.00E-05	Tr error HS-ANN
		0.0005	5.00E-06	-4.36E-13	1.40E-07	-4.36E-11	-5.99E-15	0.00005	3.05E-07	Tes error ANN
		3.20E-01	3.20E-03	2.40E-04	1.86E-01	-2.20E-03				Tes error GA-ANN
	1.86E-02	3.60E-03	7.95E-02							Tes error PSO-ANN
		5.40E-02	5.40E-03							Tes error HS-ANN