Soft Computing (2021) 25:8483–8513
https://doi.org/10.1007/s00500-021-05775-5

METHODOLOGIES AND APPLICATION

Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models

Milad Shahvaroughi Farahani[1] · Seyed Hossein Razavi Hajiagha[2]

Accepted: 26 March 2021 / Published online: 25 April 2021

Abstract
Today, stock market has important function and it can be a place as a measure of economic position. People can earn a lot of money and return by investing their money in the stock exchange market. But it is not easy because many factors should be considered. So, there are many ways to predict the movement of share price. The main goal of this article is to predict stock price indices using artificial neural network (ANN) and train it with some new metaheuristic algorithms such as social spider optimization (SSO) and bat algorithm (BA). We used some technical indicators as input variables. Then, we used genetic algorithms (GA) as a heuristic algorithm for feature selection and choosing the best and most related indicators. We used some loss functions such as mean absolute error (MAE) as error evaluation criteria. On the other hand, we used some time series models forecasting like ARMA and ARIMA for prediction of stock price. Finally, we compared the results with each other means ANN-Metaheuristic algorithms and time series models. The statistical population of research have five most important and international indices which were S&P500, DAX, FTSE100, Nasdaq and DJI.

Keywords Genetic algorithm · Social spider optimization algorithm · Artificial neural network · Bat algorithm

1 Introduction

People are always looking for some ways to invest their capital. Stock market is one of the main places to invest the money and capital. However, stock markets confront with different risks. Therefore, investors require forecasting stock price and it depends on several psychological, economic, etc., factors. Thus, several methods have been developed to predict stock prices. These forecasting methods aim at proposing approaches to predict index value or stock prices (Lah et al. 2019). They need different considerations due to the quality and quantity of data. Technical analysis, fundamental analysis and statistical methods are used for stock price prediction. One of the main hypotheses which should be considered and better to test it is efficient market hypothesis (EMH) (Malkiel 1989, 2003). EMH means that information has a high impact on stock prices and prices modifying themselves according to this information (Greco et al. 2019). The efficient market ensures investors that they access similar information (Naseer and Bin Tariq 2015). The efficient market is based on the assumption that no system can beat the market because if this system becomes general, everybody will use it. Thus, it negates its potential profitability.

Time series is a main method which is used for the prediction of share prices. Time series analysis deals with analyzing a series of data gathered during time. Time series are common in different fields of economy, finance, healthcare, etc (Bisgaard and Kulahci 2011). This method tries to forecast future by assuming that the previously observed pattern can be considered as the foundation to extract future behavior (Shin 2017). Heuristic algorithms are another set of methods being used for prediction. Heuristic algorithms are often used as an alternative optimization algorithm, instead on exact methods that usually deal with finding a good feasible solution without any

✉ Seyed Hossein Razavi Hajiagha
h.razavi@khatam.ac.ir

Milad Shahvaroughi Farahani
m.shahvaroughi@khatam.ac.ir

[1] Department of Finance, Faculty of Management and Finance, Khatam University, Tehran, Iran

[2] Department of Management, Faculty of Management and Finance, Khatam University, Tehran, Iran

assurance of being optimal (Kaveh and Ghazaan 2018). Heuristic algorithms are applicable in different decision problems which have complex structures, and it takes a long time to identify their characteristics. The other methods are metaheuristic algorithms. Metaheuristic algorithms are actually a set of algorithms that are applied to heuristic algorithms and simultaneously allow the use of heuristic algorithms in a large number of issues. It does not take into account the characteristics of the model and is compatible with any model and different solutions (Osman and Kelly 1996; Talbi 2009). In cases which the set of solutions is too large to being sampled completely, metaheuristics examine a set of these solutions. Since metaheuristics are usually developed based on a limited set of assumptions, they can be used for a variety of problems (Blum and Roli 2003).

Comparing with exact methods, there is no guarantee that metaheuristics can find global optimum of an optimization problem (Blum and Roli 2003; Khosravanian et al. 2018). Metaheuristic algorithms are applied to solve difficult and complicated problems in an affordable time. These algorithms usually found acceptable rather than optimal solutions for these types of problems (Talbi 2009). Gogna and Tayal (2013), Abdel-Basset et al. (2018), Wong and Ming (2019) are a sample of studies reviewed the applications of metaheuristic algorithms in different fields.

The other method is ANN which is retrieved from the function of human brain and thinking. ANN is in the subset of artificial intelligence (AI), and it is usable in different contents such as pattern recognition, classification, regression. Because most of the financial data are nonlinear and are asymmetric, ANN can recognize the relationship perfectly.

This paper aims to predict the stock price by ANN. The developed ANN is trained using some metaheuristic algorithms, including social spider optimization (SSO) and bat algorithm (BA). A group of technical measures are used as input variables. Genetic algorithm (GA) also is used as feature selection and choosing the best and most related indicators. Different loss function is used as error assessment criteria.

To evaluate the performance of the mentioned hybrid algorithm, the obtained results are compared with results of ARIMA as a time series model to predict the stock price. This obtained performance and its comparisons are done on five most important and international indices including S&P500, DAX, FTSE100, NASDAQ and DJI. The paper is structured as follows: the 2nd part reviews the available literature. The 3rd part describes ANN structure and proposed algorithm. In Sect. 4 ARIMA is used for time series forecasting. Sections 5 and 6 examine the experimental process and the results. Finally, last section means 7th part concludes the paper. You can see more results in Appendices A and B.

2 Literature review

Stock market is a place where you can invest your capital to buy or sell part of the company's assets in the form of shares (Preethi and Santhi 2012). We can see the market as a pulse of economic activities and almost country, which can be a place with high benefits for investors which they can grow their capital and money or totally their wealth. Stock market is characterized by features such as nonlinearity, discontinuity, and volatile multifaceted elements because many items affect is such as general economic situations, political actions and broker's assumptions (Hadavandi et al. 2010). Considering the amount of fluctuation in this market, a rapid decision making process is required. Therefore, it is very important that transactions are done in the shortest possible time (Barakat et al. 2016). Obtaining maximum profit is the ultimate goal of the investors. As a result, many researchers are looking for market forecasting capabilities in a variety of ways (Prasanna and Ezhilmaran 2013). According to previous studies, ANN seems a good and reasonably validated method in the prediction of stock price (Idris et al. 2015). The three most popular ANNs for stock prediction are the recurrent neural network (RNN) (Saad et al. 1998), the radial basis function (RBF) (Han et al. 2001), and multilayer perceptron (MLP). There are many methods for training the ANN and some of them are better than the others in finding the linear and nonlinear relationship. ANN uses two thresholds for exploration of linear and nonlinear qualifications. The number of layer is very important in predictability. If we use too many layers, the ANN couldn't find the fittest choice and the structure will be complicated. In addition, too few layers mean that the ANN is unable to find the global solution and nonlinear relationships. The researchers have tried to discover some methods which have high speed with high accuracy and lower the error. For this reason, the metaheuristic algorithms are used. These methods are used for the network optimization and finding the best number of input and hidden layers. The ANN models in forecasting stock price, stock return, exchange rate, inflation and imports work better than traditional statistical models (Yim and Mitchell 2002).

Gupta and Wang (2010) used feed-forward neural networks to forecast and trade the future index prices of the Standard and Poor's 500 (S&P 500). The effect of training the network with the most recent data, together with gradually subsampled past index data, has been studied in this research. They also studied the effect of past NASDAQ 100 data on the prediction of future S&P 500. A daily

trading strategy has been used, to buy/sell, according to the predicted prices, and hence calculate the directional efficiency and the rate of returns for different periods. They were able to obtain significantly higher returns compared to earlier work. There were numerous exchange-traded funds (ETFs), which attempted to replicate the performance of the S&P 500 by holding the same stocks in the same proportion as the index, and therefore, giving the same percentage returns as the S&P 500.

Zhu and Wang (2010) proposed an intelligent trading system using support vector regression optimized by genetic algorithms (SVR-GA) and multilayer perceptron optimized with GA (MLP-GA). Experimental results showed that both approaches outperformed conventional trading systems without prediction and a recent fuzzy trading system in terms of final equity and maximum drawdown for Hong Kong Hang Seng stock index.

He et al. (2013) did researches on the principles and theories in the field of financial market, and basic technical analysis methodologies about the stock market were studied and practiced with the help of Feature Selection algorithms. They used the data of Shanghai Stock Exchange Composite Index (SSECI) from 24 March 1997 to 23 August 2006 to measure twelve technical indicators for later research. The twelve chosen technical indicators were calculated, and the results were taken as the input of the Feature Selection algorithms. The three kinds of Feature Selection algorithms, principle component analysis (PCA), genetic algorithm (GA) and sequential forward selection (SFS) were studied. According to the results and analysis, PCA was the most reliable, but might be time-consuming if the input has very large dimensions. Genetic Algorithm would have a better performance since it takes the advantage of randomness in such a situation. SFS could generate the local optimal solution, but with a risk of "nesting problem".

Dong et al. (2013) first reproduced the one-step ahead prediction system from Phua et al. for the prediction of stock price. Secondly, they made some modifications and successfully outperform the original prediction system in terms of MSE value, hit-rate and absolute error. Moreover, they explored a difficult multi-step prediction problem. Firstly, they reproduced a multi-step prediction system using simple recursive algorithm. Then, they proposed an error constraint algorithm in order to obtain better weights and bias, as well as smaller accumulated errors. The results outperformed the simple recursive algorithm by observation.

Zheng et al. (2013) explored the application of a wavelet neural network (WNN), whose hidden layer was comprised of neurons with adjustable wavelets as activation functions, to stock prediction. They discussed some basic rationales behind technical analysis, and based on which, inputs of the prediction system were carefully selected. This system was tested on Istanbul Stock Exchange National 100 Index and compared with traditional neural networks. The results showed that the WNN could achieve very good prediction accuracy.

Fang et al. (2014) improved stock market prediction based on genetic algorithms (GA) and wavelet neural networks (WNN) and reported significantly better accuracies compared to existing approaches to stock market prediction, including the hierarchical GA (HGA) WNN. Specifically, they added information such as trading volume as inputs and they used the Morlet wavelet function instead of Morlet–Gaussian wavelet function in their prediction model. They also employed a smaller number of hidden nodes in WNN compared to other research work. The prediction system was tested using Shenzhen Composite Index data.

Lim et al. (2016) used delayed neural network models to predict public housing prices in Singapore. The delayed neural networks are used to estimate the trend of the resale price index (RPI) of Singapore housing from the Singapore Housing Development Board (HDB), with nine independent economic and demographic variables. The results show that the delayed neural network model is able to produce a good fit and predictions.

Göçken et al. (2016) predicted Turkish stock price index using technical indicators and hybrid ANN based on GA and harmony search (HS). The results showed that the error of hybrid metaheuristic algorithms is less than ANN. They compared the hybrid ANN-HS and ANN-GA model and found that the error of ANN-HS is less than ANN-GA.

Considering the problem of dealing with features with a similar contribution, the feature weighted SVM (FWSVM) and feature weighted K-nearest neighbor (FWKNN) are proposed to forecast market indices of stock by assigning different weights to different features (Chen and Hao 2017).

Then this model is tested on two stock markets. Comparing the results, the FWSVM and FWKNN perform better than non-weighted models.

Ghasemiyeh et al. (2017) optimized artificial neural network by metaheuristic algorithms. In their research, cuckoo search, improved cuckoo search, enhanced cuckoo search genetic algorithm, genetic algorithm and particle swarm optimization (PSO) are examined. Testing these hybrid algorithms and using 28 variables as input, the results show that particle warm outperforms other algorithms in this study.

Goli et al (2018) used various metaheuristic algorithms for improving the results and predicting demand in dairy industry too. This study used two well liked metaheuristic algorithms, such as GA and PSO, together with two more recent algorithms titled invasive weed optimization (IWO)

and cultural algorithm (CA) as feature selection and demand forecasting in dairy industry. According to the results, PSO showed the best performance in feature selection while IWO can significantly improve the prediction error.

Sin and Wang (2017) explored the relationship between the features of Bitcoin and the next day change in the price of Bitcoin using an Artificial Neural Network ensemble approach called Genetic Algorithm-based Selective Neural Network Ensemble, constructed using Multi-Layered Perceptron as the base model for each of the neural network in the ensemble. To better understand the practicality and its effectiveness in real-world application, the ensemble was used to predict the next day direction of the price of Bitcoin given a set of approximately 200 features of the cryptocurrency over a span of 2 years. Over a span of 50 days, a trading strategy based on the ensemble was compared against a "previous day trend following" trading strategy through back-testing. The former trading strategy generated almost 85% returns, outperforming the "previous day trend following" trading strategy which produced an approximate 38% returns and a trading strategy that follows the single, best MLP model in the ensemble that generated approximately 53% in returns.

Chong et al. (2017) applied three methods such as PCA, restricted Boltzmann machine (RBM) and auto encoder on the deep learning network as feature extraction with three loss functions such as root-mean-squared error (RMSE), mean absolute error (MAE), mutual information (MI) and normalized mean squared error (NMSE), to predict future market trend of South Korea. Sezer et al. (2017) employed GA for stock trading system on base of deep neural network (DNN) to anticipate buy–sell–hold. GA was used as feature selection and generates the buy–sell point in mentioned system. Later, Dixon (2018) also used a long short-term memory (LSTM) network and forecasted short-term price movements.

Zhang et al. (2018) designed a system for prediction of stock price trend which could predict stock price movement and its increase or decrease trend interval during predetermined periods. They used random forest model and trained it on historical data from a China Market to categorize the multiple clips of stocks into four major groups regard to the different kinds of their close prices. The result indicates the improvement in prediction of volatility in market and some merits such as precision and return per dealing.

Baek and Kim (2018) proposed a framework, entitled ModAugNet, consisting two modules based on LSTM: one for prevention and one for prediction. The framework is tested over two Korean stock data set. The obtained results show an improvement in different error measures.

Ahmed et al. (2019) used ant colony optimization (ACO) in forecasting stock price of Nigerian stock exchange. They compared ACO with three other algorithms including Price Momentum Oscillator, Stochastic and Moving Average. They concluded that ACO has more accuracy and lower error than other methods. Ghanbari and Arian (2019) used support vector regression (SVR) and butterfly optimization algorithm (BOA) in forecasting stock market. They presented a new BOA-SVR model based on BOA and compared it with results of 11 meta-heuristic algorithms on NASDAQ data. The result indicated that the considered model can improve the results and optimizing the SVR Parameters. On the other hand, this model has worked very well with higher performance accuracy and lower time consumption compared to other models. Chandana (2019) used a novel approach based on least square support vector regression (LSSVR) and Machine Learning. He decided to design an expert system for prediction of stock price and he hoped to help strengthen the forecast with improving the power of accuracy. Their system was successful because the computation became fewer and calculation was simpler too. Rajesh et al. (2019) used ensemble learning techniques for stock trend prediction concentrating on the stock price change percentage. They predicted S&P500 and its future trend with ensemble learning. To this aim, they considered two foreseen methods: ensemble learning and heat map. Evidences suggest that support vector machine (SVM), random forest, and K-neighbor's classifiers have more promising results compared to other methods. The accuracy of the forecast model seems upper than 51%, which illustrate 23% increase in prediction accuracy.

Pal and Kar (2019) used a hybrid approach to forecast time series of stock price by using data discretization based on fuzzistics [1; 2], where cumulative probability distribution approach (CPDA) is used to get the intervals for the linguistic values. First-order fuzzy rule generation and reduction of rule sets by rough set theory have been performed. Thereafter, forecasting of the time series data is computed from defuzzification using reduced rule base and its historical evidences. Proposed approach is applied on stock index closing price of three-time series data (BSE, NYSE, and TAIEX) as experimental data sets and the results show that the method is more effective than its counter parts.

Liu and Wang (2019), in order to address the profit bias in model evaluation, proposed a new effective metric, mean profit rate (MPR). The effectiveness of metric was measured based on the correlation between the metric value and profit of the model. Experiments on five stock daily index data among four countries show that MPR outperformed the classification metrics in correlating to profit. In view of these findings, they suggested that MPR is

a more effective metric than the classification metrics in stock trend prediction.

Lv et al. (2019) assessed different types of machine learning algorithms based on trading cost. They tried to compare traditional algorithms and advance DNN models. They used data of period 2010–2017 from different index component stocks. The random forest, naïve Bayes, logistic regression, classification and regression tree (CART), traditional machine learning algorithms are SVM, and extreme gradient boosting while the DNN architectures include deep belief network (DBN), multilayer perceptron (MLP), RNN, Stacked Auto encoders (SAE), LSTM, and gated recurrent unit (GRU). Their results indicated that each algorithm is superior than other based on transaction cost. For example, regardless of the transaction cost, traditional machine learning algorithms perform better in many directional assessment indices; however, DNN models perform better despite of transaction cost.

Zaman (2019) realized that efficiency of Bangladesh largest stock market is weak. To improve the results, he conducted parametric and nonparametric tests of DSE & CSE from 2013 to 2017. The results proved that two stock exchanges are not efficient in the weak form.

Zhou et al. (2020) investigated the SVM power in predicting stock price change direction. They used five different datasets, including technical indices, stock posts, transaction data records, news and Baidu index, and concluded that there are different ideal data source to forecast active stocks and inactive ones. Finally, they found that more active stocks can be predicted with higher accuracy for different periods of time.

Sahoo and Mohanty (2020) proposed a combination of ANN and gray wolf optimization (GWO) technique and compared the hybrid ANN-GWO with ANN. They compare these models on a dataset of Bombay stock exchange in a time period from 2004 to 2018. The performance of the ANN-GWO and ANN is evaluated according to different error measures. The comparisons illustrate that the mentioned hybrid method results better than the ANN model.

Kumar et al. (2020) reviewed and organized the published papers on stock market prediction using computational intelligence. The related papers are organized according to related datasets, input variables used, preprocessing methods, techniques used for future selection, forecasting methods and performance metrics to evaluate the models.

According to the above reviewed papers, it can be inferred that study on stock market prediction is still being raised among researchers. Also, it seems that hybrid methods are the permanent approach used in different researches. Considering the acceptance of ANN-based methods, the focus is to enhance the performance of ANN through some metaheuristics. Limitations of the previous methods are provided in Table 1 (Obthong et al. 2020).

3 Hybrid metaheuristic ANN for stock price prediction

3.1 Technical indicators

ANN includes 3 layers that the input layer is the first one. Here, some important technical indices are used as input variables of the network. Indicators are mathematical functions that are based on specific formulas for analyzing stock prices or analyzing market indices using graphical tools. Investors and managers can use them to analysis of stock market. Choosing the best and most related technical indicators is a controversial issue. To deal with this challenge, GA is used for feature selection. The considered technical indicators are illustrated in Table 2.

3.2 Artificial neural network (ANN)

Today, ANN is used in different problems. Some of the well-known applications include function approximation, classification and clustering information, save and reviewing data, optimization, etc (Versace et al. 2004). ANN can be used for a variety of topics, including time series forecasting. Because stock price data is not normal and it has some characteristics such as skewness, kurtosis, fat tail and nonlinearity, ANN can be used for recognizing these qualifications. As mentioned earlier, a typical ANN includes 3 layers: (1) Input, (2) Hidden; and (3) Output.

The number of layers in each phase is important because by changing them, the network will react differently. Thus, GA is applied for choosing important variables. The GA is used as feature selection for some reasons include (1) conceptual easiness; (2) searching a wide area of solutions instead of just examining a single point; (3) supporting multi-objective optimization; (4) GA is a stochastic process and robust to local minima/maxima; and finally (5) GA is easily paralyzed (Oreski et al. 2012). By doing this, the speed rate of calculation is increased and also the network will be prevented getting into local minima or maxima trap.

Neural network is based on learning which means each times it tries to reduce their error based on trial and error. The network has three phases: (1) training, (2) validation and (3) testing. This study includes two main parts. The first one includes calculating technical indicators and selecting the most optimal indicator by using GA. Second one includes forecasting closing price by using different hybrid ANN models and comparing their prediction error. Two metaheuristic algorithms, means SSO and BA, are used since they have had successful and brilliant results in

Table 1 Limitations of the previous methods

No	Methods	Purpose	Limitations
1	ARIMA (autoregressive integrated moving average model)	Forecasting and clustering	• Doesn't work well for nonlinear time series • Requires more data • Takes a long time processing for a large dataset
2	BPNN (back propagation neural network)	Forecasting	• Sensitive to noise • Actual performance based on initial values • Slow convergent speed • Easily converging to a local minimum
3	CART (classification and regression trees)	Classification and forecasting	• Unstable even when the training data are small changed
4	GP (Gaussian process)	Classification and forecasting	• Generates "black box" models which are difficult to interpret • Can be computationally expensive
5	GRNN (generalized regression neural network)	Classification and forecasting	• Requires more memory space to store the model • Can be computationally expensive because of its huge size
6	Hierarchical clustering	Clustering	• The length of each time series is the same because of the Euclidean distance • Useful only for small datasets because of its quadratic computational complexity
7	HMM (hidden Markov model)	Clustering, classification and clustering	• Requires parameters to be set and is based on user assumptions that may be false with the result that clusters would be inaccurate • Takes a long time processing for a large dataset
8	K-Mean	clustering	• The number of clusters must be specified in advance • Sensitive to noise • Only spherical shapes can be determined as clusters • Unable to handle long time series effectively because of poor scalability
9	KNN (K nearest neighbor)	Classification and forecasting	• The number of nearest neighbors must first be determined • Can be computationally expensive • Memory limitation • Sensitive to the local structure of the data
10	LR (logistic regression)	Classification and forecasting	• Sensitive to outliers • Strong assumptions
11	LSTM (long short-term memory)	Classification and forecasting	• Lacks a mechanism to index the memory while writing and reading the data the number of memory cells is linked to the size of the recurrent weight matrices
12	MLP (multi-layer perceptron)	Classification and forecasting	• Convergence is quite slow • Local minima can affect the training process • Hard to scale
13	PSO (particle swarm optimization)	Forecasting	• Lacks a solid mathematical foundation for analyzing future development of relevant theories
14	RBF (radial basis function neural network)	Classification and forecasting	• Classification process is slower than MLP
15	RF (random forest)	Classification and forecasting	• Requires more computational power and resources because it creates a lot of trees • Requires more time to train than decision trees
16	RNN (recurrent neural network)	Classification and forecasting	• Difficult to train
17	SOM (self optimizing maps)	Clustering and classification	• Does not work well for time series of unequal length because of the difficulty involved in determining the scale of weight vectors • Sensitive to outliers
18	SVM (support vector machine)	Classification and forecasting	• Sensitive to outliers • Sensitive to parameter selection
19	SVR (support vector regression)	Forecasting	• Sensitive to users' defined free parameters

Table 1 (continued)

No	Methods	Purpose	Limitations
20	ANN (artificial neural network)	Classification and forecasting	• Over fitting • Sensitive to parameter selection—ANNs just give predicted target values for some unknown data without any variance information to assess the prediction

various fields and researches such as prediction of the stock price and prediction of interest rate; on the other hand, they have some properties including their approximate and usually non-deterministic nature and also they are not problem-specific and flexible too (Beheshti and Shamsuddin 2013) So, stock price data, from 2013 to 2018, are split into two sections: training and testing. Then, it is analyzed with artificial intelligence algorithms and forecasting the next day closing stock price. Like Göçken et al. (2016), for training period, 70% of observations are used and for testing and validation period, the remaining 30% is used. Models are compared based on 8 criteria of prediction error. Different algorithms are used for training ANN, e.g. gradient descent backpropagation (Mozer et al. 1995), Levenberg–Marquardt (LM) backpropagation (Hao and Wilamowski 2011), BFGS quasi-Newton backpropagation (Fahad et al. 2018), Bayesian regulation backpropagation (Burden and Winkler 2008), etc.

In this study, the hidden layer neurons of the normal neural network are determined based on trial and error and it is not fixed. Due to the feature of MATLAB software, the number of hidden layers is fixed to 1. This can be considered as a kind of limitation. To this aim, 1–32 neurons are examined in hidden layer; the fittest amount of neurons with the most accuracy is chosen as ANN model. For training ANN, error-back propagation is used. LM algorithm is also used as the minimization algorithm in learning the model (Haddad and Haghighat Monfared 2012). The amount of training epochs is a thousand and for improving the results, we increased it to 2000, and the initial training rate set to 0.01 and is decreased to 0.001 to improve the accuracy of the results. ANN has two threshold functions for recognizing the linear and nonlinear qualification of the model. The output function for the hidden layer is a tangent sigmoid function which is a mathematically shifted version of the sigmoid function and it has the feature of both functions that's mean the Tahn and sigmoid and threshold function for the output layer is pure line function. We used the Tanh function for some reasons: (1) because the range of our numbers is between [1, − 1]. (2) The activation works almost always better than the sigmoid function. (3) It is capable to learn and perform more complex and nonlinear tasks. Hence the mean for the hidden layer comes out be 0 or very close to it and hence helps in centering the data by bringing mean close to 0. This makes learning for the next layer much easier.

The architecture of the proposed neural network is represented in Fig. 1.

Here, input variables are illustrated with 20 technical variables. These variables are normalized to be used as input variables using Eq. (1).

$$\widetilde{S}_i = \frac{(S_i - S_{min})}{S_{max} - S_{min}} \cdot i = 1 \ldots N \quad (1)$$

Similarly, the goal of normalization is to change the values of dataset to a common scale, without distorting differences in the ranges and it generally speeds up learning and leads to faster convergence.

Where S_i is the ith observation. Figure 2 represents the research methodology.

3.3 GA-ANN forecasting model

To select input variables, GA is used. GA is a stochastic search algorithm inspiring natural evolution (Kuo and Han 2011; Saber et al. 2013). Generally, GA seeks the approximate optimal solution by coding and decoding of a population of solutions and reproduction by crossover and mutation, as its main operators. In this study, inputs are coded using binary variables. The chromosomes are defined to contain 26 bits. Of these bits, 21 bits are associated with existence (bit value equal to 1) or nonexistence (bit value equal to 0) of input variables (technical indicators). 5 additional bits show the figure of neurons ($2^5 = 32$) in hidden layer. The population size of GA is 20 (Davallou and Azizi 2017; Kai and Wenhua 1997). The primary population is formed stochastically. Technical indicators and the number of hidden layer are entered to the GA and using ANN as its fitness function, and it is the amount of MSE reproduced as output. The fittest choice is one with the lowest MSE. To increase the training speed, the epochs are set to 100. As mentioned, 70% of data are employed for training and 30% is considered to test and validate. Table 3 illustrates the parameters of genetic algorithm.

Figure 3 illustrates the proposed GA-ANN algorithm.

According to Göçken et al. (2016), roulette wheel is used for parents' selection and crossover rate is settled 80%. The one-point crossover is also used. A mutation rate

Table 2 Important technical indicators

Row	Feature	Definition	Formula
1	Open	The first price	–
2	High	The highest price	–
3	Low	The lowest price	–
4	Close	The last price	–
5	Volume	Number of traded shares	–
6	SMA-5	Simple moving average-5 days	$\frac{(Close1+Close2+\cdots+Close5)}{5}$
7	SMA-20	Simple moving average-20 days	$\frac{(Close1+Close2+\cdots+Close20)}{20}$
8	EMA-5	Exponential moving average-5 days	$\frac{CloseToday*k+EMA(5)Yestarday*(1-k)}{5}$ $K = \frac{2}{5+1} \cdot EMA(5)0 = SMA(5)$
9	ADL	Accumulation Distribution Line	$\mathrm{ADL}_{\mathrm{Yesterday}}$ + Volume * CLV CLV = $\frac{[(Close-Low)-(High-Close)]}{High-Low}$
10	CMF	Chaikin Money Flow	(((Close − Low) − (High − Close))/(High − Low)) * Volume)/Total (Volume, 21)
11	MFI	Money Flow Index	MFI = $100 - \frac{100}{1+MoneyFlowRatio}$ Money Flow Ratio = $\frac{14PeriodPositiveMoneyFlow}{14PeriodNegativeMoneyFlow}$ Raw Money Flow = TP * Volume TP = $\frac{High+Low+Close}{3}$
12	TP	Typical Price	$\frac{High+Low+Close}{3}$
13	RSI	Relative Strength Index	$100 - \frac{100}{1+RS} \cdot RS = \frac{SMA(U)}{SMA(D)}$
14	ROC	Rate of change	$\frac{(Closetoday-CloseNpreviousday)}{CloseNpreviousday}$
15	Upper Band	Upper Band of Bollinger	SMA(20) + dev(20) * 2
16	Lower Band	Lower Band of Bollinger	SMA(20) − dev(20) * 2
17	MP	Mean Price	$\frac{(High+Low)}{2}$
18	ATR	Average True Range	Current ART = [(Prior ATR × 13) + Current TR]/14 ATR = $\left(\frac{1}{n}\right) \sum_{(i=1)}^{(n)} TRi$ TRi = A particular true range n = the number of time period
19	CCI	Commodity Channel Index	CCI = (TP − 20 Period SMA of TP)/(0.015 × Mean Deviation) (TP) = (High + Low + Close)/3 Constant = 0.015
20	DX	Directional Movement Index	+DI = $\left(\frac{Smoothed+DM}{ATR}\right) \times 100$ −DI = $\left(\frac{Smoothed-DM}{ATR}\right) \times 100$ DX = $\left(\frac{\lvert+DI-DI\rvert}{\lvert+DI+-DI\rvert}\right) \times 100$ + DM (Directional Movement) = Current High-PH PH = Previous High − DM = Previous Low-Current Low Smoothed ± DM = $\sum_{t=1}^{14} DM - \left(\frac{\sum_{t=1}^{14} DM}{14}\right) + CDM$ CDM = Current DM ATR = Average True Range

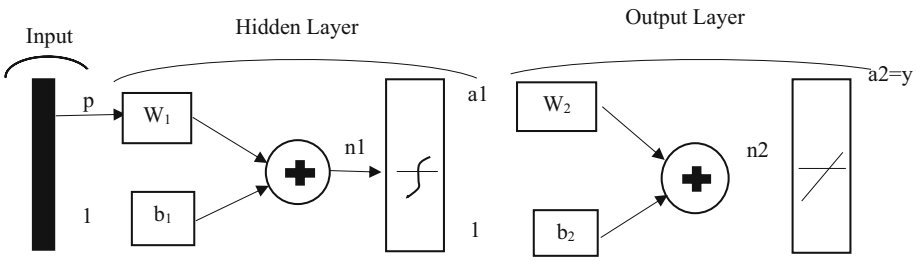


Fig. 1 The structure of the desired artificial neural network (Ghasemiyeh 2017)

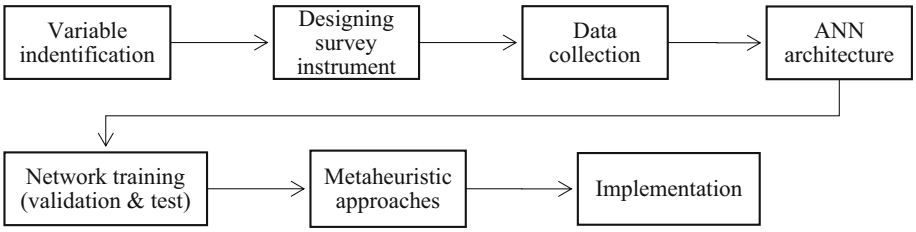


Fig. 2 Research methodology

Table 3 Parameters of GA

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	50

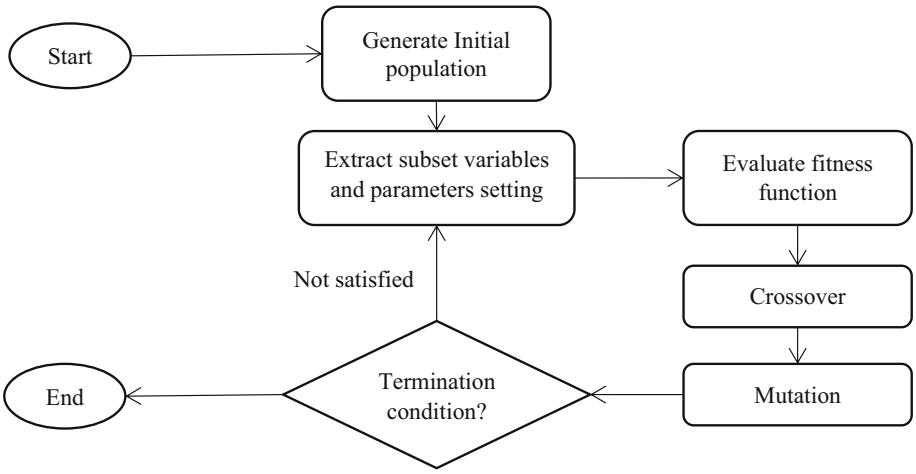


Fig. 3 Considered GA flow chart for training ANN (Liu and Wang 2019)

of 20% along with binary mutation is also used. Selecting the best chromosomes among parents and children's, new generation continues with repeating the algorithm until termination condition is satisfied. Two termination conditions used are (1) repeating the best individual to 100 generations, and (2) reaching the maximum generation condition, i.e. reproduction of 2000 generations. Different parameters such as mutation rate, crossover rate and the number of population (population size) have been set based on Göçken et al. (2016). However, since different problems have different properties such as scalability/non-scalability, dimensional dependent/independent, there are some common beliefs about the range of parameters in different researches. For example, it is better that the number of population be between 20 and 50 and the crossover and mutation rate between 80–95 and 0.5–1, respectively (Hassanat et al. 2019).

The GA pseudo-code (i.e. steps and how to get the parameters) is illustrated in Table 4.

Table 4 GA-ANN pseudo-code

```
function GENETIC-ALGPRITHM (population, FITNESS-FN) returns an individual
    inputs: population, a set of individuals
            FITNESS-FN, a function that measure 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 population, according to FITNESS-FN
```

3.4 Bat algorithm (BA)

Inspiring from echolocation behavior of microbats, the bat algorithm (BA) is proposed as a heuristic optimization algorithm (Iglesiasa et al. 2020). Mirjalili et al. (2014) proved the superiority of BA to some of the other algorithms, like GA and PSO. The echolocation of microbats is interesting and there are several parameters for simulation of its behavior such as speed, location, rate of occurrence, and loudness (Gàlvez and Iglesias 2016): every virtual and figurative bat flies stochastically with a speed υ_i at location (solution) x_i, with a rate of occurrence f_{min}, changing wavelength λ and loudness A_0. Searching and finding its prey, based on the approximation of the target, the rate of occurrence and loudness are changed and rate of pulse emission r is justified (Yang 2010). Exploration is strengthened by a local accidental walk and choosing the best continues until reaching the termination attributes (Nawi et al. 2014); to control the dynamic behavior of swarm of bats, a technique with frequency-tuning nature is used. Also, tuning the algorithm parameters can be applied balance between exploration and exploitation (Yang 2010).

The loudness may change in different ways; it can be supposed that it alters from a big positive value A_0 to a minimum fixed value A_{min}. Initially, BA started with a random population of bats, and then to renovate the location of each bat, the following formulas are used at each step:

$$\upsilon_i^{new} = \upsilon_i^{old} + (x_i + x_{best}) \times f_i \tag{2}$$

$$x_i^{new} = x_i^{old} + \upsilon_i^{new} \tag{3}$$

$$f_i = f_{min} + \varphi_1 \times (f_{max} + f_{min}) \tag{4}$$

where X_{best} is the position of the best bat, φ_1 is a random value in [0, 1], f_{max} and f_{min} are the values of max and min frequency, and here they are assumed as being 1 and 0, respectively. The initial value of the frequency of each bat is selected from the range $[f_{max}.f_{min}]$. f_i is applied to manage the velocity and the bats' movement scope (Nawi et al. 2014).

Afterwards in the local search, each bat uses a random walk to create a new alternative. To accomplish this, each bat produces a random number β. If β is greater than the pulse emission rate, the new solution is generated by Eq. (4), otherwise it is generated by Eqs. (5–8) (Tsai et al. 2014; Chou and Nguyen 2018).

$$x_i^{new} = x_i^{old} + eA_{min}^{old} \tag{5}$$

e is an incidental value from -1 to 1 and A_{min}^{old} illustrates the mean value of the all bats' loudness. Here, to optimize the generated solution in the case that β is not greater than A_i, a modification method is presented.

The main objective of this modification is to increase the diversity of the bat population using mutation and utilizing crossover which help to enhance the search efficiency. Thus, for each bat x_i, three bats $(x_{k1} \bullet x_{k2} \bullet x_{k3})$ are selected randomly in which $i \neq k1 \neq k2 \neq k3$. Now, by using the mutation and crossover operators, two below improved solutions are produced:

$$X_{opt1} = X_{k1} + a_1(X_{k2} - X_{k3}) \tag{6}$$

$$X_{opt1} = \left[X_{opt1.1}.X_{opt1.2}.\ldots.X_{opt1.n}\right] \tag{7}$$

n is the dimension of this problem.

$$X_{opt2} = \begin{cases} x_{best.i.} & \text{if } a_2 < a_3 \\ x_{i.} & \text{Otherwise} \end{cases} \tag{8}$$

$$X_{best} = [X_{best.1} \cdot X_{best.2}.\ldots.X_{best.n}] \tag{9}$$

where $a_1.a_2$ and a_3 are randomly generated numbers in [0, 1] interval. Among $X_{opt1}.X_{opt2}$ and X_i, the best one is replaced with X_i. If $\beta < A_i$ and $f(x_i) < f(x_{best})$, the new generated solution is acceptable. Accepting the new solution, the loudness and the pulse emission rate are renewed as below:

$$A_i^{new} = a \cdot A_i^{old} \quad (10)$$

$$r_i^{new} = r_i^0 \cdot [1 - exp(-\gamma * t)] \quad (11)$$

Here, a and γ are constant values, r_i^0 is the initial rate of the pulse emission and t indicates the number of iterations. In this study, the explained BA is used to modify the weight matrix of ANN. In BAT-ANN, at first, the primary population of bats is used to form the initial weight matrix. This matrix is then passed to ANN to start the training phase (Hafezi et al. 2015). Then, BA specified the best solution based on the neural network results. A local search is then performed to discover new solutions. The replacement of new acceptable solution with the best knows solution is replied until satisfaction of the termination criteria (Yang 2010). Finally, the optimal values of the weight matrix are determined. Figure 2 shows the flowchart of BAT-ANN.

It should be noted that the calculation method is adapted from Yang (2010), Golmaryami et al. (2015), and Jantan et al. (2017).

Table 5 summarizes the notation used for parameters of BA.

As in GA-ANN algorithm, different parameters such as pulse rate and velocity. have been set based on different research such as Golmaryami et al. (2015) and Hafezi et al. (2015).

The process steps of the bat algorithm are shown in Table 6.

Table 5 Bat algorithm parameters

Fitness function	Mean square error (MSE)
n	Population size (typically between 10 and 40)
N-gen	Number of generations
$A_{\max}$	Maximum loudness
A_{min}	Minimum loudness
R	Pulse rate (constant or decreasing)
Q_{min}	Frequency minimum (0)
Q_{max}	Frequency maximum (1.5)
N-iter	Number of iteration (1000)
D	Number of dimension (20)
LB	Lower bound (− 100)
UB	Upper bound (+ 100)
Q	Frequency
V	Velocity

Table 6 BA pseudo-code

Initialize the bat population Xi (i = 1, 2, … , n) and Vi
Define pulse frequency F_i
Initialize pulse rate r_i and the loudness A_i
While ($t <$ Max number of iterations)
Generate new solutions by adjusting frequency,
Updating velocities and positions [Eqs. (5) to (7)]
If (rand $> r_i$)
Select a solution among the best solutions randomly
Generate a local solution around the selected best solution
End if
Generate a new solution by flying randomly
If (rand $< A_i$ and $f(X_i) < f$ (Gbest)
Accept the new solutions
Increase r_i and reduce A_i
End if
Rank the bats and find the current Gbest
End while

The above pseudocode interpretation and more details are briefly as follows

1. Bat is initialized then passes its first population to ANN as weight's values
2. Load data
3. ANN starts training and computes the accuracy of the model
4. Bat finds the initial best solution by means of the ANN's results,
5. While $I <$ Max number of iterations

3.5 Social spider algorithm (SSA)

This kind of algorithm means social spider optimization (SSO) that is in the subset of metaheuristic, evolutionary and swarm intelligence algorithms that is modified lifestyle form of the social spiders, male and female (Mirjalili et al. 2015). They have and do various functions and operations due to their gender, each one does different tasks like mating, preying, web design, social interrelation, etc. (Luque-Chang et al. 2018). As you know, a problem may have several answers and you should find them in search space. In this algorithm, you can consider communal web as a search space. Spiders due to their positions play the role of solution (Evangeline and Abirami 2019). Web and vibration are very important for spiders because they can understand when the prey is trapped and some details about mating which are transferred along the thin strings of the web due to spiders' vibrations (Reddy et al. 2019). In vibration two things are very important: weight and distance. Spiders should change their weights regard to a fitness value. Accordingly, they execute different operations such as mating. Like genetic algorithm which is based on superiority of better individuals, the offspring with better weigh changes with weak one, else the population doesn't change. At the end of all iterations, the best spider with the best fitness seems to be the optimal choice (Yeh 2012). In training ANN with SSO algorithms, the best spider has a

role of optimal solution. Here, to find the fitness value of the spider, minimization of MSE is considered as the objective function.

Like other metaheuristic algorithm, SSA has different steps and parameters.

3.5.1 Initialization

Like any other swarm intelligence and evolutionary algorithms, the SSO algorithm begins with assigning an initial value to population and spider location. It includes two kinds of population: female f_i and male m_i spiders. The amount of population and individuals means female spiders $f_i(N_f)$ can be selected intractably which often lies at range of 65–90% and is obtained by Eq. (12) and the amount of male spiders $m_i(N_m)$ is also determined by Eq. (13):

$$N_f = floor[(0.9 - rand(0.1) * 0.25) * N \tag{12}$$

$$N_m = N - N_f \tag{13}$$

In SSO algorithm, the position of the f_i is important. Therefore, we considered some limitations means upper bound and lower bound that f_i is generated randomly between them. We have shown the initial parameters of lower and upper bound with P^{low} and P^{high} which are represented by:

$$f_{i.j}^0 = P_j^{low} + rand(0.1) * \left(P_j^{high} - P_j^{low}\right) \tag{14}$$

where $i = 1.2.\ldots.N_f$, $j = 1.2.\ldots.n$. Then m_i is also accidentally created and equates as:

$$m_{i.j}^0 = P_j^{low} + rand(0.1) * \left(P_j^{high} - P_j^{low}\right) \tag{15}$$

where $i = 1, 2, \ldots, j = 1, 2, \ldots, n$.

3.5.2 Fitness assignation

It should be noted that the size of spiders is very important and can affect the improvement in the solutions and optimizing the network and totally achieving the main goal. In the presented model, a weight W_i is assigned to the ith spider (irrespective of gender) that indicates its quality in the population S. The weight is calculated for each spider as follows:

$$\frac{J(s_i) - worst_s}{best_s - worst_s} \tag{16}$$

where $J(s_i)$ shows the fitness value of spider s_i. Equation (17) represents the values of $worst_s$ and $best_s$ as:

$$\begin{aligned} best_s &= max_{k\in[1.2....N]}(J(s_k)) \\ worst_s &= min_{k\in[1.2....N]}(J(s_k)) \end{aligned} \tag{17}$$

3.5.3 Vibration modeling

The communal web is something more than communal web and is vital for spiders according to the important things it makes possible, for example, making connection and relationship between spiders and their distance to each other. The size of vibrations means higher or lower one has different meaning. The more vibration means closer distance to each other and vice versa. In sequence, to exchange information between members i and member j of the colony, the mathematical definition of vibration is formed as follows:

$$Vib_{ij} = w_j e^{-d_{ij}^2} \tag{18}$$

where d_{ij} shows the Euclidian distance of member i and j within the colony. Spiders use these vibrations to understand the distance and transfer it from member i to member j. These 3 types of vibrations occur between i and j that are illustrated as $\text{Vib}c_i$, $\text{Vib}b_i$, and $\text{Vib}f_i$.

The individual i (s_i) receives the vibration $Vibf_i$ as a result of the sent information by the member c (s_c) that is nearer to i and also with higher weight compared to i ($w_c > w_i$).

$$Vibc_i = w_c e^{-d_{i.c}^2} \tag{19}$$

The individual i receives the vibrations $\text{Vib}b_i$ as a result of the transferred information by the member b (s_b) that has the best weight or the best fitness value of the population S as a whole.

$$\text{Vib}b_i = w_b e^{-d_{i.b}^2} \tag{20}$$

Finally, the transferred information from the member i to the closest female individual $f_{(s_f)}$ can be defined by $\text{Vib}f_i$ as:

$$\text{Vib}f_i = w_f e^{-d_{i.f}^2} \tag{21}$$

3.5.4 Female cooperative operator

The movement between spiders means absorption or excretion is based on several random criteria which is shown with symbol f_i in this article without considering gender. A random number r_m is generated uniformly in the range of [0, 1]. When r_m is smaller than a predetermined threshold PF, an attraction and a repulsion are created and shown in Eq. (22).

$$f_i^{t+1} = \begin{cases} f_i^t + a * \text{Vib}c_i * \left(s_c * f_i^t\right) + \beta * \text{Vib}b_i * \left(s_b * f_i^t\right) + \delta * (rand - 0.5) \\ \text{with probability} pf \\ f_i^t - a * \text{Vib}c_i * \left(s_c * f_i^t\right) - \beta * \text{Vib}b_i * \left(s_b * f_i^t\right) + \delta * (rand - 0.5) \\ \text{with probability} 1 - pf \end{cases} \tag{22}$$

Table 7 SSO Algorithm Parameters

Dim	Dimension (30)
Bound	100
Max-iteration	10,000
Pop-size	25
Alpha	0.99
Beta	0.7
Gamma	0.9
Fitness function	Mean square error (MSE)

where $\alpha \cdot \beta \cdot \delta$ and rand are random numbers between [0, 1] and t is the iteration number and the individuals s_c and s_b symbolize the nearest spider with a higher weight than f_i^t and the best spider in the communal web, respectively.

3.5.5 Male cooperative operator

According to weights, there are two groups of spiders. Some spiders have weights more than median the median of N_m (dominant or D) and the others have weights lower than median of the N_m (non-dominant or ND). The median weight is expressed by N_{f+m}. The position of the m_i might be equated as:

$$m_i^{t+1} = \begin{cases} m_i^t + a * Vibf_i * \left(s_f - m_i^t\right) + \delta * (rand - 0.5) if(w_{N_{f+i}} > w_{N_{f+m}}) \\ m_i^t + a * \left(\frac{\sum_{h=1}^{N_m} -h - N_f + n}{\sum_{h=1}^{N_m} * w_{N_f} + h} - m_i^t\right). \end{cases} \tag{23}$$

3.5.6 Mating operator

Mating has a specific range and takes place between dominant (D) and f_i. The mating range is equated as:

$$r = \frac{\sum_{j=1}^{n}\left(P_j^{high} - P_j^{low}\right)}{2 * n} \tag{24}$$

Spider weight is directly related to offspring. The chance of the spider with more weight is more likely to offspring and vice versa. Table 7 illustrates the parameters of the SSO algorithm.

The calculation method used in this paper is adapted from (Luque-Chang et al. 2018; Saravanan et al. 2019; Gülmez and Kulluk 2019).

The steps in the social spider algorithm to obtain the parameters are as follows:

1. Consider N as the total number of colony population; define the number of male N_m and female N_f spiders in the entire population S.
2. Initialize randomly the male and female members and calculate the radius of mating.
3. Calculate the weight of every spider of S.
4. Move the female spiders according to the female cooperative operator.
5. Move the male spiders according to the female cooperative operator.
6. Perform the mating operation.
7. If the stop criteria are met, the process is defined; go back to step 3.

4 Time series forecasting (ARMA and ARIMA)

When you check an events or sequence during time intervals, it is a kind of time series. (Hamilton 1994). You can check and examine the events in different frequencies such as yearly, monthly, weekly, daily, hourly or even in minutes or seconds. When you use past data to predict the future happens, it can be considered as prediction of single variable time series[1] and when you anticipate something more than a series, it is called multivariate time series forecasting (Granger et al. 1974; Reinsel 2003).

In autoregressive integrated moving average (ARIMA), the subsequent values of the variable are supposed to be the past observations and random errors in form of linear function and are called white noise. So we can use the same equation for the prediction of future value (Zhang 2003). ARIMA can be used to model the time series which are not station and don't have any pattern.

An ARIMA model is known with these components: (p, d, q) (Sowell 1992).

First of all, you should set the time series stationary. Because, phrase 'Auto Regressive' in ARIMA conveys that it uses its own lags for predictors as a linear regression model. We intend to check that whether predictors are dependent or independent to each other which this correlation can affect the model.

To set a time series stationary, a lot of methods are represented. Differencing is the most common one (Clements and Hendry 2000). That is, the current value minus the previous value. Due to the complicacy in the model, sometimes it needs to difference it more than one. Thus, the value of d shows the minimum number of distinct required to create the series stationary. Heretofore the time series is stationary, it means that $d = 0$ and it doesn't need differencing.

When particular lagged values of Y_t are used as predictor variables, the AR(p) model is considered as an

[1] Univariate time series forecasting.

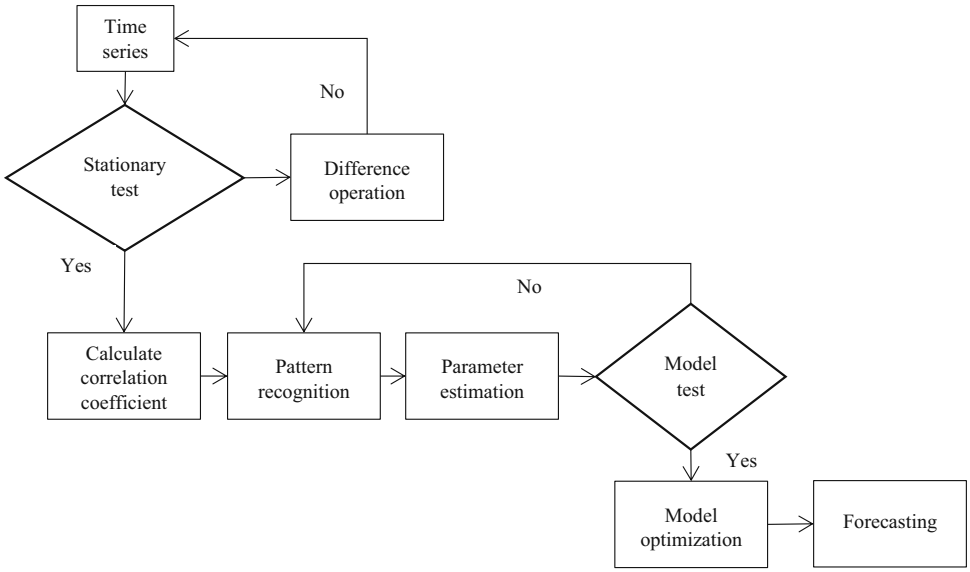


Fig. 4 ARIMA flowchart (Ma et al. 2018)

autoregressive model. Where results of a time period affect the following periods, the lags are generated.

The "p" value indicates the order. For instance, "First-order autoregressive process" can be shown as AR(1). The output variable of first-order AR in time t depends on its previous time periods $(t-1)$. The same is true in case of second- or third-order AR process that depend on data of two or three periods apart.

An AR model is one where Y_t is only related to its own lags. Here, Y_t as the lags is illustrated as (Tseng et al. 2001; Akaike 1998).

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_p Y_{t-p} + \varepsilon_1 \tag{25}$$

where $(Y_{t-1}.Y_{t-2}.\ldots.Y_{t-p})$ are the past series values (lags), $(\beta_1.\beta_2.\ldots.\beta_p)$ are the coefficient of lag estimated by the model and it also estimates α as the seperating term.

Also the moving average (MA) model equals 1 while Y_t is only depended to lagged caused by forecast errors (Said and Dickey 1984).

$$Y_t = \alpha + \varepsilon_t + \emptyset_1 \varepsilon_{t-1} + \emptyset_2 \varepsilon_{t-2} + \cdots + \emptyset_q \varepsilon_{t-q} + \varepsilon_{t-q} \tag{26}$$

where the errors are caused by autoregressive models regard to the related lags. These errors ε_t and ε_{t-1} relate to Eqs. (27)–(28):

$$Y_t = \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_0 Y_0 + \varepsilon_1 \tag{27}$$

$$Y_{t-1} = \beta_1 Y_{t-2} + \beta_2 Y_{t-3} + \cdots + \beta_0 Y_{t-n} + \varepsilon_{t-1} \tag{28}$$

They come from AR and MA models, respectively.

Through the combination of AR and MA with at least one differencing, an ARIMA model can be produced (Pai and Lin 2005). So the equation becomes:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_p Y_{t-p} + \varepsilon_1 + \emptyset_1 \varepsilon_{t-1} + \emptyset_2 \varepsilon_{t-2} + \cdots + \emptyset_q \varepsilon_{t-q} \tag{29}$$

The following diagram shows the flowchart of ARIMA model (Fig. 4).

Additional explanations and more details are as follows:

Step 1 Check stationarity: if a time series has a trend or seasonality component, it must be made stationary before we can use ARIMA to forecast.
Step 2 Difference: if the time series is not stationary, it needs to be stationarized through differencing. Take the first difference, and then check for stationarity. Take as many differences as it takes. Make sure you check seasonal differencing as well.
Step 3 Filter out a validation sample: this will be used to validate how accurate our model is. Use train test validation split to achieve this.
Step 4 Select AR and MA terms: use the ACF and PACF to decide whether to include an AR term(s), MA term(s), or both.
Step 5 Build the model: build the model and set the number of periods to forecast to N (depends on your needs).
Step 6 Validate model: compare the predicted values to the actuals in the validation sample.

Table 8 Statistical description of data

Symbol	Time interval	Number of data (before normalizing)	Number of data (after normalizing)	Number of data (after normalizing)	Number of input variables	Number of output layer	Target indicator
DJI	2018–2020	504	435	435	20	1	Closing price
DAX	2018–2020	504	436	436	20	1	Closing price
FTSE100	2018–2020	504	443	443	20	1	Closing price
NDAQ	2018–2020	504	437	437	20	1	Closing price
S&P500	2018–2020	504	438	438	20	1	Closing price

Table 9 Training, validation and testing (T.V.T) error and network architecture

indices	Architecture	Weights	Fitness	Train error	Validation error	Test error	AIC	Correlation	*R*-squared
DJI	[20-50-1]	1101	0.0224	43.1729	48.8100	44.5219	1629.23	0.9991	0.9982
DAX	[20-37-1]	815	24.4495	0.0455	0.0446	0.0409	− 988.45	0.9993	0.9986
FTSE100	[20-50-1]	1101	0.05615	12.3556	15.9880	17.8072	1232.51	0.9979	0.9957
S&P500	[20-33-1]	727	0.2096	5.0277	5.0658	4.7692	237.52	0.9989	0.9978
NDAQ	[20-18-1]	397	2.8119	0.3131	0.4033	0.3556	− 1257.5	0.9986	0.9972

5 Experimental results and findings

The main goal of this study is to forecast stock price hybridizing ANN with GA for feature selection and two metaheuristic algorithms include BA and SSO for improving the network. The five major indices of DAX, S&P500, FTSE100, DJI and NDAQ are studied in this research. The desired time interval for research is from 4 July 2018 to 4 July 2020 about 2 years. Some important technical indicators like RSI, and MACD, etc., are employed as input variables and are reduced in optimal position. Thus, 20 technical indicators are selected to predict stock price, which 19 variables are inputs and 1 variable is output or target variable that determines the next day's price.

The first step, as declared in Sect. 3 is data normalization. Data are normalized between [− 1, 1] to become ready as input variables. Table 8 is a general description of the indices, the timeframe and number of data used in this study.

5.1 Artificial neural network (ANN)

As mentioned before, ANN includes three layers. The feature of ANN used in this study is defined in Sect. 3.2. Summarily, the number of input layer is 20, output layer is 1; and hidden layer quantity varies due to trial and error. The hidden layer uses tangent sigmoid as its activation layer and the output layer uses the simple linear. The data set is divided into two sections (1) training network (70%) and (2) validation (30). LM algorithm is used for training. Mean square error (MSE) is also adapted as loss function. The related information about architecture, training and testing for each indices is represented in Table 9.

More information about training, validation and testing for DJI index is represented Table 10 and Fig. 5, for instance. Other indices are presented in "Appendix A".

5.2 GA-ANN algorithm

The GA is used for choosing the best and fittest input and hidden layers in ANN.

Therefore, Sect. 3.3 determines and describes the related parameters including size of population, the magnitude of generations, and rates of mutation and crossover. Using

Table 10 The DJI index details (T.V.T)

DJI	Training	Validation
Absolute Error	1.4611	9.9177
Network Error	4.49E−08	0
Error Improvement	3.57E−22	
Iteration	371	
Training Speed, iter/sec	7.1072	
Architecture	[20-10-1]	
TA	LM	
TSR	NEI	

Testing	Target	Output	AE	ARE
Mean	26,191.85	26,191.54	1.94	7.40E−05
SD	1394.71	1394.58	1.04	3.90E−05
Min	21,792/19	21,795.43	0.0023	9.47E−08
Max	29,551.41	29,546.42	4.99	1.70E−04

TA training algorithm; *TSR* training stop reason, *NEI* no error improvement

GA, the training, validation and testing error along with network architecture are determined according to Table 11.

Accordingly, using GA the number of input variables can be decreased to 8, while the amount of *R*-Squared is increased. The best fitness is the best technical indicators which network could recognize. For each index, as it is clear, a different number of input variables is selected, and this is due to the difference in the importance and the role of each technical indicator in the final price or target output (index). Details about selected technical indicators are represented in appendix (table ***A5).

5.3 Bat algorithm (BA)

In this section, the parameters are optimized and the network is improved improved using bat algorithm. The obtained result is illustrated in Table 12.

5.4 SSO (social spider optimization) algorithm

In this part, the global best fitness and global best solution are checked after 1000 iterations. Therefore, the error is improved using SSO. At first, the parameters set to a predetermined number and then, the network optimizes it with minimum error. Table 13 indicates the optimum error and parameters.

$\alpha \cdot \beta \cdot \delta$ are random numbers between [0, 1]. The classical SSO requires the random selection of parameters $\alpha \cdot \beta \cdot \delta$ [(22) and (23)] to control the movement of the spiders, which can affect the mentioned balance leading the algorithm to a premature convergence. The other details including the ANN structure (i.e. the number of neurons in each layer such as input layer, hidden layer and output layer) and the estimation error and the average optimum solutions are attainable too.

According to this table, it can be easily seen that error is very lower than ANN and GA-ANN network.

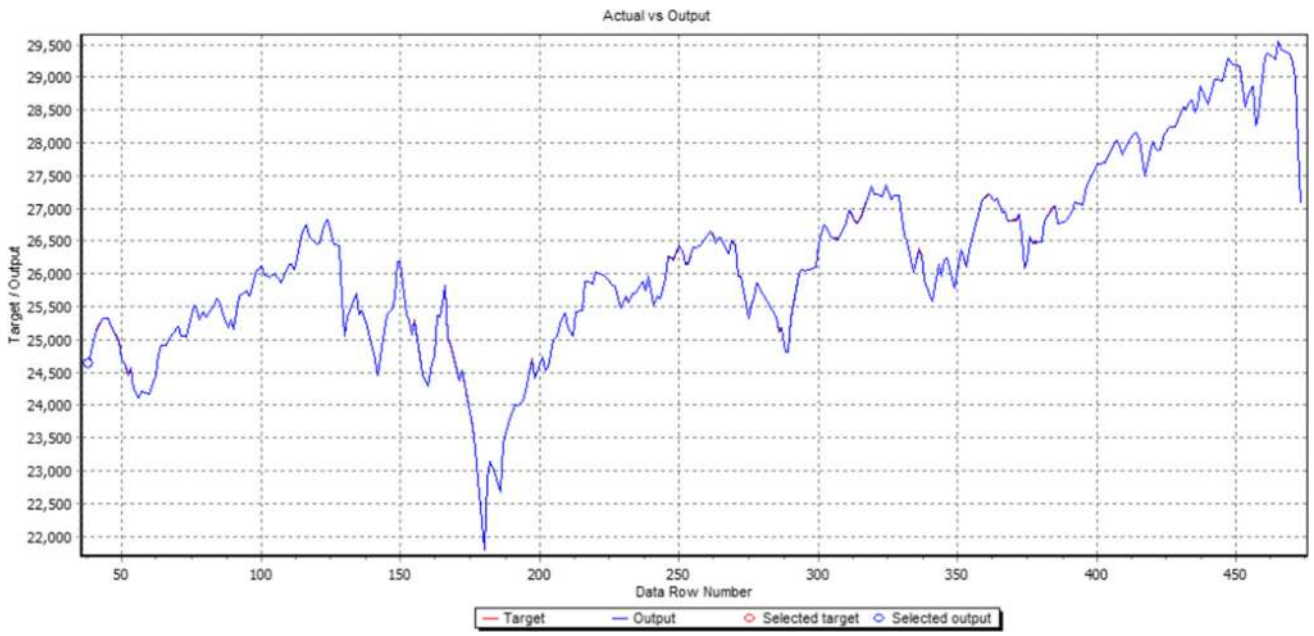


Fig. 5 Actual V.S output (testing) for DJI

Table 11 T.V.T error and network architecture after using GA

indices	Architecture	Weights	Fitness	Train error	Validation error	Test error	AIC	Correlation	*R*-squared	Best fitness
DJI	[10-16-1]	193	0.0123	39.76	73.36	81.21	− 211.23	0.9994	0.9989	89.81
DAX	[11-10-1]	131	22.18	0.0482	0.0608	0.0450	− 2339.3	0.9994	0.9987	59.65
FTSE100	[8-13-1]	131	0.0897	9.1412	12.79	11.38	− 798.78	0.9990	0.9980	74.33
S&P500	[12-30-1]	421	6.178	0.0721	0.1361	0.1618	− 1648.3	0.9999	0.9999	69.58
NDAQ	[10-21-1]	253	0.1649	5.1365	6.4543	6.0617	− 704.09	0.9994	0.9989	88.87

Table 12 Bat-ANN optimum parameters and error

Indices	Alpha	F_{min}	Gamma	*G*	*d*	Pop	*M*	Fitness function (MSE)
DJI	0.99	8.35E−03	0.9	10	10	30	1000	1.0E−55
DAX	0.99	6.43E−04	0.9	10	11	30	1000	1.0E−63
FTSE100	0.99	5.51E−05	0.9	10	8	30	1000	1.0E−31
NDAQ	0.99	5.31E−03	0.9	10	12	30	1000	1.0E−33
S&P500	0.99	1.33E−04	0.9	10	10	30	1000	1.0E−s22

Table 13 SSO-ANN optimum parameters and error

Indices	Alpha	Beta	Gamma	Epoch	Input layer	Hidden layer	Output layer	Global best fitness	Global best solution (average)
DJI	0.7665	0.6439	0.7512	250	10	10	1	1.0E−64	1.0E−44
DAX	0.7521	0.5441	0.7591	357	11	22	1	1.0E−50	1.0E−20
FTSE100	0.6314	0.5512	0.6371	550	8	13	1	1.0E−73	1.0E−51
NDAQ	0.5365	0.6891	0.8111	953	12	17	1	2.0E−68	2.0E−35
S&P500	0.871	0.752	0.667	368	10	12	1	1.0E−30	1.0E−16

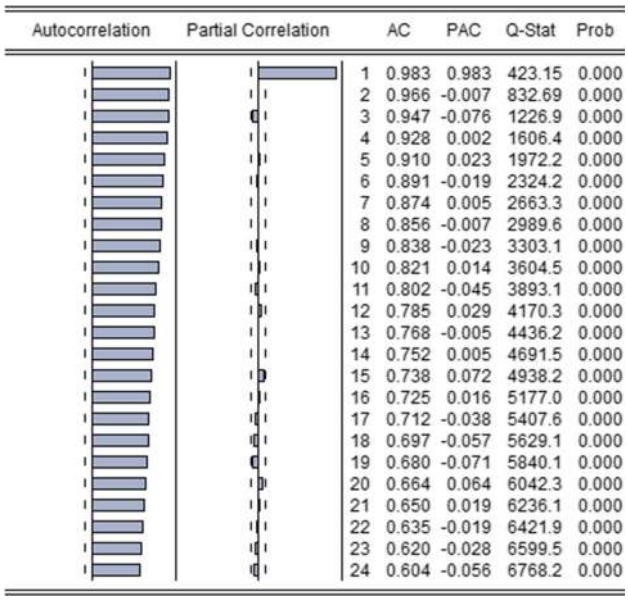


Fig. 6 Correlogram of closing price (DJI)

6 Time series forecasting (ARIMA)

The time series with financial nature usually are not stationary; they have some characteristics such as skewness and kurtosis with fat tail. Before doing everything, it seems necessary to check and recognize the stationary of the series. In this research, to find and test the stationary, Augmented Dickey Fuller Test is used. First, the stationary of each index is checked separately. The correlogram of DJI is shown in Fig. 6. Table 12 is shown the Unit root test without differencing for DJI.

From Table 14, since *t*-statistic, i.e. − 2.001110, is bigger than critical values in various significance levels (1%, 5% and 10%). Thus, series has a unit root and doesn't seem stationary. This problem is solved using ADF test.

After differencing, the series is stationary (Fig. 6). More details are represented in Table 15 (Fig. 7).

Now the series can be forecasted using ARIMA. Using Eviews 10, the degree of ARIMA is predicted. Table 16 shows the best model estimation. The used models to select criteria are summarized as can be seen in Table 17. Also,

Table 14 Unit root test without differencing (DJI)

H_0: CLOSE has a unit root			
Exogenous: Constant			
Lag Length (LLgth): 0 (Automatic—based on SIC (ABSIC), maxlag = 17)			
Prob.*	*t*-statistic		
0.2864	−2.001110	Augmented Dickey–Fuller test (AD-FT) statistic	
	− 3.445197	1% level	Test critical values
	− 2.867980	5% level	
	− 2.570265	10% level	
*MacKinnon (1996) one-sided *p*-values			

AD-FT Equation				
Dependent Variable: *D*(CLOSE)				
Method: Least Squares (LSqr)				
Date: 04/10/20 Time: 10:43				
Sample (adjusted): 2435				
Number of observations (IO): 434 after set out				
Prob	*t*-statistic	Std. error	Coefficient	Variable
0.0460	− 2.001110	0.008431	− 0.016871	CLOSE(− 1)
0.3509	0.933882	0.003227	0.003014	*C*

0.000672	Mean dependent var (MD var)	0.009184	*R*-squared (Rsqrd)
0.062874	SD dependent var (SDD var)	0.006891	Adjusted *R*-squared (ARsqrd)
− 2.697693	Akaike info criterion (AIC)	0.062657	SE of regression (SErgrs)
− 2.678924	Schwarz criterion (SC)	1.695974	Sum squared resid (Ssqrd *r*)
− 2.690285	Hannan–Quinn criteria (HQC)	587.3995	Log likelihood (LLH)
1.870658	Durbin–Watson stat (DWs)	4.004442	F-statistic (F-stat)
		0.046006	Prob (F-statistic)

Fig. 8 illustrates the Akaike information criteria, while the ARIMA forecasting summary is illustrated in Table 18.

As it is clear, the best ARIMA selected model is (4, 1, 3) with AIC value − 2.695. The above process is done over all the indices and the results are represented in "Appendix B".

7 Comparing results

In this part, some similar studies are reviewed and the obtained results are compared with them in the, as illustrated in Table 19.

It can be seen that the lowest loss functions and highest *R*-Squared are obtained using the Social Spider Optimization (SSO) and bat algorithm (BA) and these algorithms performed well.

8 Conclusions

Today, the speed of making decisions has increased. So, the stock market has been many fluctuations and volatilities. Different factors toughen up the severity of fluctuations among them can refer to major economic, politic and social changes. On the other hand, with the Coronavirus outbreak in the late of 2019, a great fluctuation is expected in stock market. Thus, using improved and well-equipped methodologies to confront these fluctuations will be a necessity. One of the main tools that can help investors is artificial intelligence (AI). AI has many applications such as pattern recognition, regression, classification.

In the current study, application of a usual ANN in forecasting stock price is compared with a hybrid meta-heuristic-based ANN. To forecast stock price, a data set is employed to train and test an ANN. Then, a hybrid ANN is developed. In the proposed hybrid ANN, genetic algorithm

Table 15 ADF test after differencing

H_0: *D*(CLOSE) has a unit root			
Exogenous: Constant			
LLgth: 0 (ABSIC, maxlag = 17)			
Prob	*t*-Statistic		
0.0000	− 19.56296	AD-FT statistic	
	− 3.445232	1% level	Test critical values:
	− 2.867995	5% level	
	− 2.570273	10% level	

AD-FT Equation

Dependent Variable: *D*(CLOSE, 2)

Method: LSqr

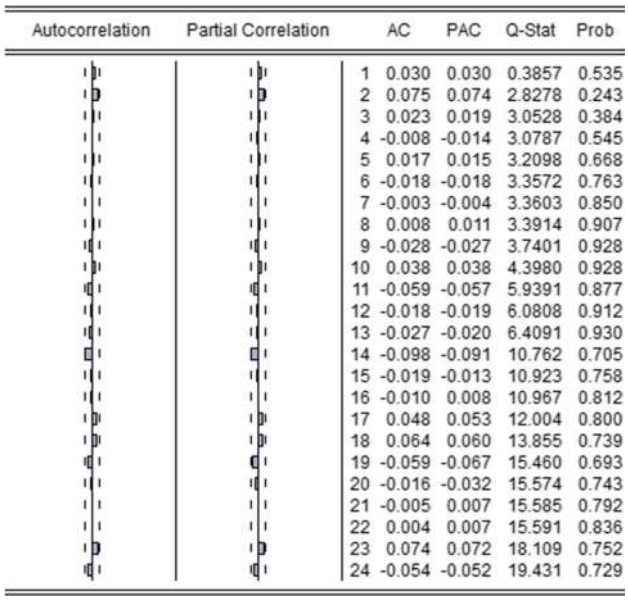
Date: 04/10/20 Time: 11:31

Sample (adjusted): 3435

Number of observations: 433 after set out

Prob	*t*-statistic	std. error	Coefficient	Variable
0.0000	− 19.56296	0.049508	− 0.968528	*D*(CLOSE(− 1))
0.8627	0.173085	0.003026	0.000524	*C*

− 0.000815	MD var	0.470327	Rsqrd
0.086398	SDD var	0.469098	ARsqrd
− 2.688280	AIC	0.062952	SErgrs
− 2.669477	SC	1.708034	SsqrdR
− 2.680857	HQC	584.0126	LLH
1.949354	DWs	382.7092	F-stat
		0.000000	PF-stat



	AC	PAC	Q-Stat	Prob
1	0.030	0.030	0.3857	0.535
2	0.075	0.074	2.8278	0.243
3	0.023	0.019	3.0528	0.384
4	-0.008	-0.014	3.0787	0.545
5	0.017	0.015	3.2098	0.668
6	-0.018	-0.018	3.3572	0.763
7	-0.003	-0.004	3.3603	0.850
8	0.008	0.011	3.3914	0.907
9	-0.028	-0.027	3.7401	0.928
10	0.038	0.038	4.3980	0.928
11	-0.059	-0.057	5.9391	0.877
12	-0.018	-0.019	6.0808	0.912
13	-0.027	-0.020	6.4091	0.930
14	-0.098	-0.091	10.762	0.705
15	-0.019	-0.013	10.923	0.758
16	-0.010	0.008	10.967	0.812
17	0.048	0.053	12.004	0.800
18	0.064	0.060	13.855	0.739
19	-0.059	-0.067	15.460	0.693
20	-0.016	-0.032	15.574	0.743
21	-0.005	0.007	15.585	0.792
22	0.004	0.007	15.591	0.836
23	0.074	0.072	18.109	0.752
24	-0.054	-0.052	19.431	0.729

Fig. 7 Correlogram of closing price after differencing (DJI)

Table 16 ARIMA forecasting

Dependent variable: *D*(CLOSE)

Method: ARMA maximum likelihood (BFGS)

Date: 04/10/20 Time: 11:41

Sample: 2435

Number of observations: 434

Failure to improve objective (non-zero gradients) after 188 iterations

Prob	*t*-statistic	Std. error	Coefficient	Variable
0.0186	2.361890	0.000731	0.001727	*C*
0.0000	− 10.50908	0.054387	− 0.571560	AR(1)
0.0000	6.938485	0.088378	0.613210	AR(2)
0.0000	4.394794	0.209860	0.922290	AR(3)
0.0000	− 13.67663	0.005546	− 0.075856	AR(4)
0.0000	44.15990	0.013955	0.616256	MA(1)
0.0000	− 25.45988	0.024209	− 0.616362	MA(2)
0.0000	− 17.12255	0.058396	− 0.999894	MA(3)
0.0000	17.57706	0.000213	0.003736	SIGMASQ

0.000672	MD var	0.052636	Rsqrd
0.062874	SDD var	0.034803	ARsqrd
− 2.695028	AIC	0.061770	SErgrs
− 2.610564	SC	1.621599	SsqrdR
− 2.661688	HQC	593.8211	LLH
1.958092	DWs	2.951641	F-stat
		0.003187	PF-stat

− 0.81 − 0.59i	− 0.81 + 0.59i	0.08	0.97	Inverted AR Roots
− 0.81 − 0.59i		− 0.81 + 0.59i	1.00	Inverted MA Roots

is used for feature selection. Then, the bat algorithm and social spider optimization are used separately for ANN parameters optimization.

In this paper, five main and important indices, such as DJI and DAX, are forecasted using ANN which is in the subset of AI. We used 20 main technical indicators as input variables. Today, many methods are used to optimization of the network. One of them is evolutionary algorithms. We used GA as an evolutionary algorithm for feature selection purpose. We could see that by using GA, the number of input variables reduced significantly. Thus, the speed of calculations and the accuracy of the network and the coefficient of determination increased. Also, two new metaheuristic algorithms including social spider algorithm and bat algorithm have been used to improve the results.

Table 17 The models used to select criteria

Model Selection Criteria Table
Dependent variable: *D*(CLOSE)
Date: 04/10/20 Time: 11:41
Sample: 1435
Number of observations: 434

HQ	BIC	AIC*	LogL	Model
− 2.661687	− 2.610563	− 2.695027	593.820926	(4, 3)
− 2.681058	− 2.669697	− 2.688467	585.397252	(0, 0)
− 2.660526	− 2.620762	− 2.686457	589.961065	(3, 2)
− 2.671159	− 2.648437	− 2.685977	586.856960	(2, 0)
− 2.671048	− 2.648326	− 2.685865	586.832798	(0, 2)
− 2.673679	− 2.656637	− 2.684792	585.599872	(1, 0)
− 2.655062	− 2.609618	− 2.684697	590.579342	(3, 3)
− 2.673553	− 2.656511	− 2.684666	585.572517	(0, 1)
− 2.668848	− 2.646126	− 2.683666	586.355450	(1, 1)
− 2.645235	− 2.588430	− 2.682279	592.054567	(4, 4)
− 2.663295	− 2.634892	− 2.681817	586.954241	(3, 0)
− 2.663246	− 2.634844	− 2.681768	586.943698	(1, 2)
− 2.663229	− 2.634827	− 2.681752	586.940082	(0, 3)
− 2.663221	− 2.634819	− 2.681743	586.938275	(2, 1)
− 2.655280	− 2.621197	− 2.677506	587.018841	(3, 1)
− 2.655068	− 2.620985	− 2.677294	586.972826	(4, 0)
− 2.654935	− 2.620852	− 2.677161	586.943976	(2, 2)
− 2.654918	− 2.620835	− 2.677144	586.940351	(0, 4)
− 2.654916	− 2.620833	− 2.677142	586.939857	(1, 3)
− 2.642085	− 2.590961	− 2.675425	589.567251	(3, 4)
− 2.646994	− 2.607230	− 2.672925	587.024642	(4, 1)
− 2.646763	− 2.606999	− 2.672694	586.974504	(2, 3)
− 2.646641	− 2.606878	− 2.672572	586.948213	(1, 4)
− 2.638654	− 2.593211	− 2.668290	587.018892	(4, 2)
− 2.638538	− 2.593094	− 2.668173	586.993524	(2, 4)

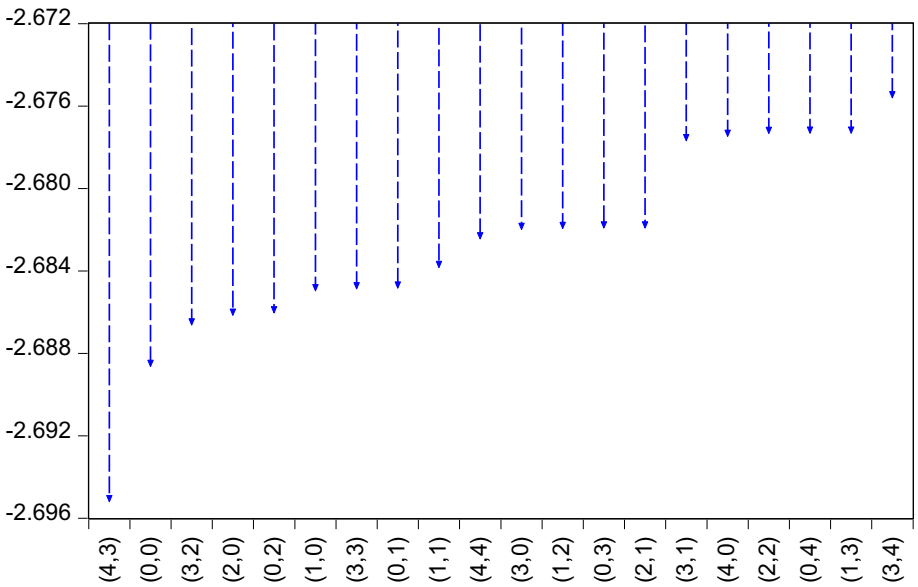


Fig. 8 Akaike information criteria (top 20 models)

Table 18 ARIMA forecasting summary

Automatic ARIMA Forecasting
Selected dependent variable: *D*(CLOSE)
Date: 04/10/20 Time: 11:41
Sample: 1435
Number of observations: 434
Forecast length: 0
Number of estimated ARMA models (No. E ARMA Ms): 25
Number of non-converged estimations (No. non-C Es): 0
Selected ARMA model: (4, 3)
AIC value: − 2.69502731054

The main advantages of using metaheuristic algorithms are as follows:

- Speed up calculations
- Reduce model complexity
- Increase the network accuracy
- Ease of using models
- High robustness
- Intelligent.

On the other hand, they have some limitations:

- In GA, there is no guarantee that the best and most related technical indicators have been selected.
- We have tried to overcome the local optima trap but it is still possible.

Comparing with previous methods, SSO and BA have had the lowest error, respectively, which could predict stock price better. As it is clear, the error of the social spider algorithm has been less, but this does not mean that this algorithm is better. Due to the difference in the time required to calculate, the complexity of the calculations, the required parameters, etc., we cannot say with certainty which one is better. But if we consider error as a measure of superiority, the social spider algorithm performed better. We used time series for the prediction of stock price too.

Table 19 Comparative study

Author and date	Proposed approaches	Data type	Data type	MSE	MAE	R^2
Gogna and Tayal (2013)	GA-ANN	Train	Train	0.0074	0.0584	0.9866
		Test	Test	0.0079	0.0585	0.9895
	PSO-ANN	Train	Train	0.0013	0.0253	0.9972
		Test	Test	0.0014	0.0260	0.9969
	ICS-ANN	Train	Train	0.0076	0.0720	0.9966
		Test	Test	0.0068	0.0694	0.9995
Sedighi et al. (2019)	ARIMA-SVM	Final outcome	Final outcome	1.0042	0.0142	
	SVM-RF	Final outcome	Final outcome	0.000295	0.0245	
	ANFIS-SVM	Final outcome	Final outcome	3.5849	0.0117	
	FA-MSVR	Final outcome	Final outcome	0.0014	0.0130	
Safa and Panahian (2018)	HS-ANN	Final outcome	Final outcome	0.02776	0.05177	0.9641
Emamverdi et al.(2016)	ANN	Final outcome	Final Outcome	0.00030	0.0174	
	ARIMA	Final Outcome	Final outcome	0.00042	0.0162	
Zheng et al.(2013)	Wavelet neural networks	Final outcome	Final outcome	0.00510	6.742E−04	0.9877
Dong et al.(2013)	One-step ahead and multi-step ahead predictions	Final outcome	Final outcome	0.0043	0.1043	0.9012
Wang et al.(2016)	Delayed neural network (DNN)	Final outcome	Final outcome	1.60E−03	1.00E−07	0.9955
Sin and Wang (2017)	Ensembles of neural network	Final outcome	Final outcome	2.05E−05	2.045E−09	0.9963
Current research	ANN	Train	Train		12.1827	0.9975
		Test	Test		13.499	
	GA-ANN	Train	Train		10.8316	0.9988
		Test	Test		19.7717	
	BA	Final outcome	Final outcome		1.0E−40	0.9993
	SSO	Final outcome	Final outcome		1.0E−52	0.999
	ARIMA	Final Outcome	Final outcome		0.0712846	0.6028

The considered model was ARIMA. Because of nonlinearity and asymmetric qualification of stock price data, ANN could predict the stock price better than time series model means ARIMA. Experiments show that hybrid models perform better to explain the model with lower error. Therefore, the main recommendation is that different new metaheuristic algorithms should be used to train the network.

Appendix A: training, validation and testing

See Tables 20, 21, 22, 23 and 24 and Figs. 9, 10, 11 and 12.

Table 20 T.V.T details (DAX)

DAX	Training	Validation
Absolute error	0.0006	0.0036
Network error	6.34E−09	0
Error improvement	9.71E−14	
Iteration	401	
Training speed, iter/sec	6.937	
Architecture	[20-10-1]	
TA	LM	
TSR	NEI	

Testing	Target	Output	AE	ARE
Mean	26.7875	26.7876	0.0006	2.00E−05
SD	1.6821	1.6821	0.0003	1.20E−04
Min	22.17	22.17	3.0E−06	9.60E−08
Max	30.61	30.61	0.0016	7.50E−05

Table 21 T.V.T details (FTSE100)

FTSE100	Training	Validation
Absolute error	0.3548	3.5899
Network error	9.19E−08	0
Error improvement	5.31E−11	
Iteration	501	
Training speed, iter/sec	7.07	
Architecture	[20-10-1]	
TA	LM	
TSR	All iteration done	

Testing	Target	Output	AE	ARE
Mean	7327.6	7327.6	0.3752	5.10E−05
SD	256.8	256.8	0.1805	2.50E−05
Min	6584.7	6585.3	1.00E−04	1.58E−08
Max	7788.4	7787.97	0.08317	1.13E−04

Table 22 T.V.T details (NDAQ)

NDAQ	Training	Validation
Absolute error	0.1354	0.261
Network error	1.50E−05	0
Error improvement	1.69E−21	
Iteration	36	
Training speed, iter/sec	6.428	
Architecture	[20-10-1]	
TA	LM	
TSR	No error improvement	

Testing	Target	Output	AE	ARE
Mean	94.74	94.69	0.2049	2.10E−03
SD	8.58	8.5	0.1568	1.70E−03
Min	76.75	77.65	3.80E−04	4.00E−06
Max	118.66	117.52	1.144	1.17E−02

Table 23 T.V.T details (S&P500)

S&P500	Training	Validation
Absolute error	0.3538	3.09
Network error	1.50E−07	0
Error Improvement	6.69E−20	
Iteration	327	
Training speed, iter/sec	7.047	
Architecture	[20-10-1]	
TA	LM	
TSR	No error improvement	

Testing	Target	Output	AE	ARE
Mean	2896.3	2896.29	0.6842	2.36E−04
SD	189.93	189.9	0.4196	1.42E−04
Min	2351.1	2353.2	8.60E−03	3.00E−06
Max	3386.14	3383.4	2.74	9.30E−05

Table 24 Selection of most important technical indicators using GA

Symbol	Technical indicators	Selected using GA	Selected (removed)
DJI	Open, High, Low, Close, Vol, SMA(5), SMA(10), EMA(5), ADL, CMF, MFI, RSI, Upper Band, Lower Band, MP, ROC, TP, DX, CCI, ATR	Open, High, Low, RS, Upper Band, SMA (5), SMA (10), ROC, Vol TP	(10)-10
DAX	Open, High, Low, Close, Vol, SMA(5), SMA(10), EMA(5), ADL, CMF, MFI, RSI, Upper Band, Lower Band, MP, ROC, TP, DX, CCI, ATR	Low, MP, SMA(5), EMA(5), TP, ROC, SMA(10), %R, ADL, RSI, MFI	(11)-9
FTSE100	Open, High, Low, Close, Vol, SMA(5), SMA(10), EMA(5), ADL, CMF, MFI, RSI, Upper Band, Lower Band, MP, ROC, TP, DX, CCI, ATR	High, ROC, %R, EMA(5), SMA(5), SMA(10), RS, RSI	(8)-12
NDAQ	Open, High, Low, Close, Vol, SMA(5), SMA(10), EMA(5), ADL, CMF, MFI, RSI, Upper Band, Lower Band, MP, ROC, TP, DX, CCI, ATR	Low, SMA(5), EMA(5), SMA(10), RS, RSI, ROC, TP, MP, ADL, VOL, CCI	(12)-8
S&P500	Open, High, Low, Close, Vol, SMA(5), SMA(10), EMA(5), ADL, CMF, MFI, RSI, Upper Band, Lower Band, MP, ROC, TP, DX, CCI, ATR	Open, Upper Band, Lower Band, RS, SMA(5), SMA(10), EMA(5), TP, RSI, High	(10)-10

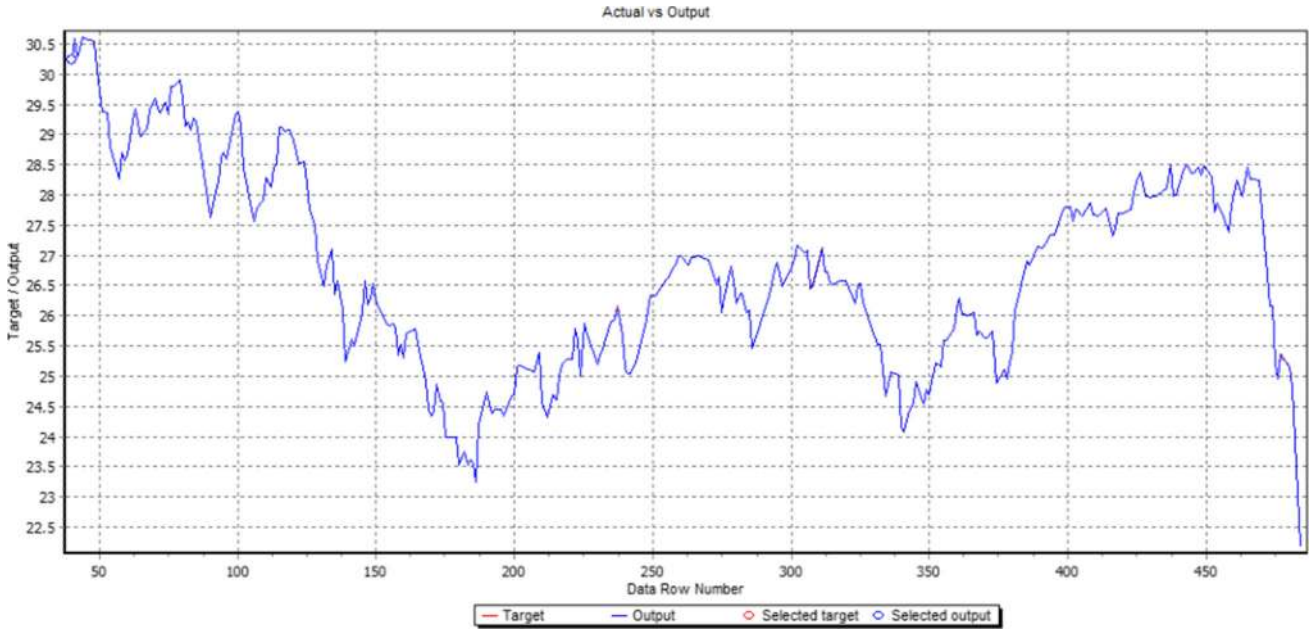


Fig. 9 Actual V.S output (testing) for DAX

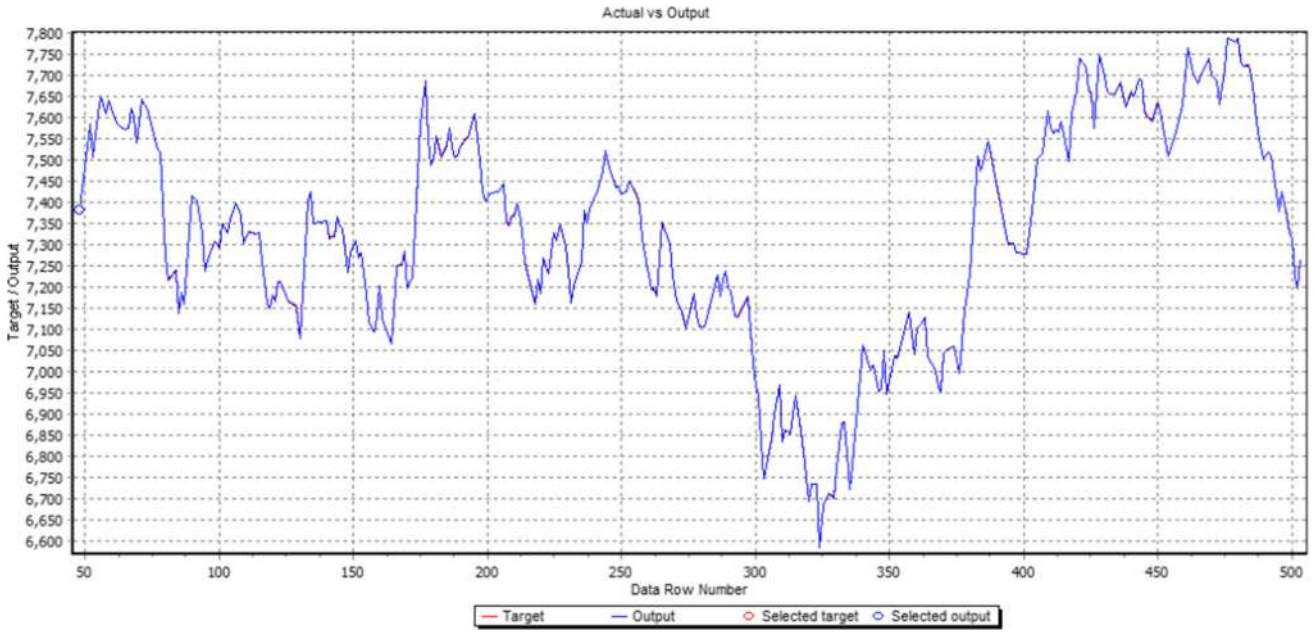


Fig. 10 Actual V.S output (testing) for FTSE100

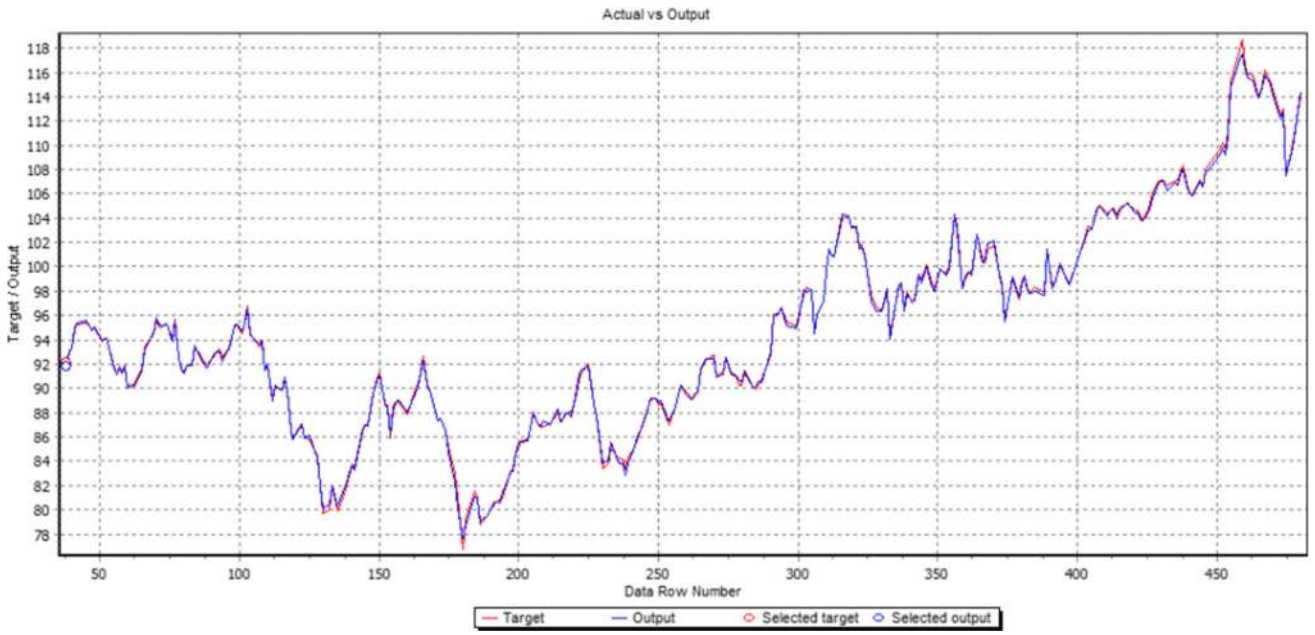


Fig. 11 Actual V.S output (testing) for NDAQ

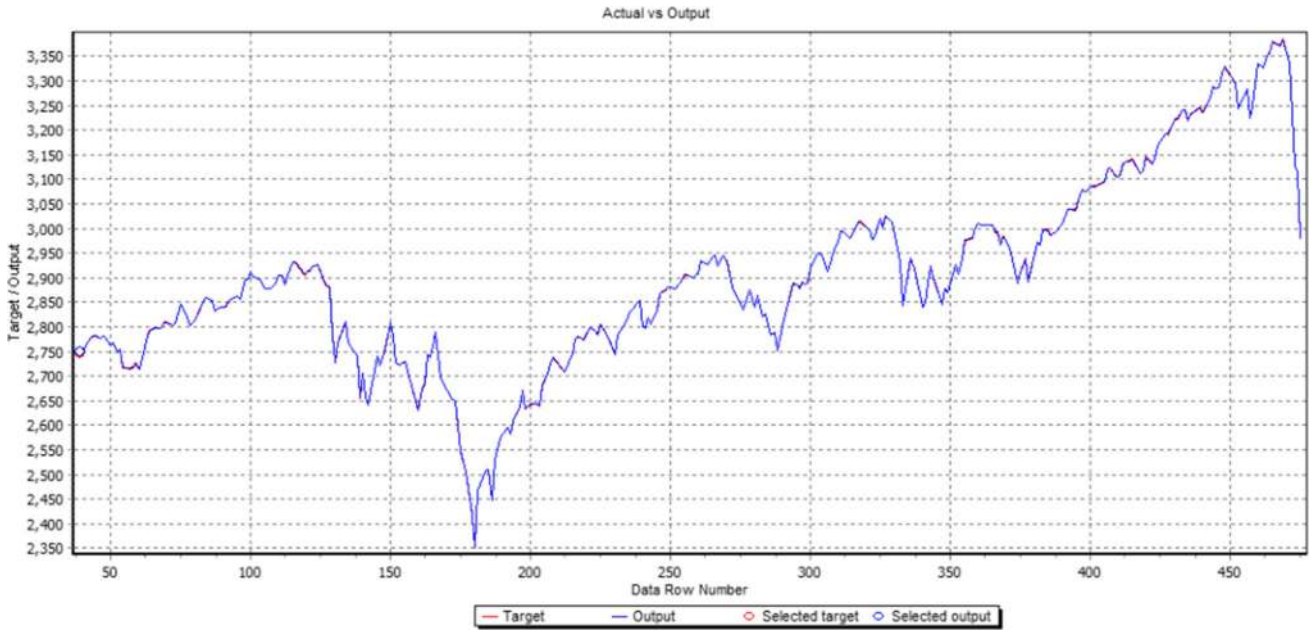


Fig. 12 Actual V.S output (testing) for S&P 500

Appendix B: time series results of indexes

See Tables 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35 and 36 and Figs. 13, 14, 15 and 16.

Table 25 Correlogram of closing price (DAX)

Autocorrelation	Partial Correlation		AC	PAC	Q-Stat	Prob
		1	0.967	0.967	410.18	0.000
		2	0.944	0.153	802.63	0.000
		3	0.920	-0.017	1176.0	0.000
		4	0.899	0.027	1532.9	0.000
		5	0.875	-0.033	1872.1	0.000
		6	0.853	-0.001	2194.9	0.000
		7	0.827	-0.053	2499.4	0.000
		8	0.803	-0.018	2786.8	0.000
		9	0.782	0.055	3060.3	0.000
		10	0.757	-0.070	3317.1	0.000
		11	0.737	0.047	3560.9	0.000
		12	0.721	0.079	3794.7	0.000
		13	0.708	0.069	4021.2	0.000
		14	0.696	0.017	4240.3	0.000
		15	0.688	0.070	4455.2	0.000
		16	0.678	-0.009	4664.5	0.000
		17	0.670	0.008	4869.2	0.000
		18	0.660	-0.042	5068.2	0.000
		19	0.648	-0.039	5260.5	0.000
		20	0.635	-0.030	5445.8	0.000
		21	0.627	0.054	5626.8	0.000
		22	0.618	0.011	5803.1	0.000
		23	0.609	0.017	5974.7	0.000
		24	0.597	-0.046	6139.8	0.000

Table 26 Unit root test without differencing (DAX)

H_0: CLOSE has a unit root			
Exogenous: Constant			
LLgth: 0 (ABSIC, maxlag = 17)			
Prob	*t*-statistic		
0.4836	− 1.596348	AD-FT statistic	
	− 3.445162	1% level	Test critical values:
	− 2.867965	5% level	
	− 2.570256	10% level	
AD-FT Equation			
Dependent variable: *D*(CLOSE)			
Method: LSqr			
Date: 04/10/20 Time: 12:05			
Sample (adjusted): 2436			
Number of observations: 435 set out			

Prob	*t*-statistic	Std. error	Coefficient	Variable
0.1111	− 1.596348	0.009615	− 0.015349	CLOSE(−1)
0.2615	1.124255	0.005497	0.006181	*C*

− 0.002100	MD var	0.005851	Rsqrd
0.038030	SDD var	0.003555	ARsqrd
− 3.699872	AIC	0.037962	SErgrs
− 3.681135	SC	0.624004	SsqrdR
− 3.692477	HQC	806.7222	LLH
1.899053	DWs	2.548326	F-stat
		0.111141	PF-stat

Table 27 Correlogram of closing price after differencing (DAX)

Autocorrelation	Partial Correlation		AC	PAC	Q-Stat	Prob
		1	-0.033	-0.033	0.4678	0.494
		2	0.082	0.081	3.3951	0.183
		3	-0.043	-0.038	4.2012	0.241
		4	0.013	0.004	4.2796	0.369
		5	-0.020	-0.013	4.4508	0.486
		6	0.021	0.018	4.6532	0.589
		7	0.006	0.010	4.6681	0.700
		8	-0.009	-0.013	4.7037	0.789
		9	0.105	0.106	9.6696	0.378
		10	0.059	0.068	11.227	0.340
		11	-0.069	-0.084	13.342	0.272
		12	-0.048	-0.056	14.377	0.277
		13	0.019	0.032	14.534	0.337
		14	-0.084	-0.079	17.687	0.221
		15	0.038	0.025	18.348	0.245
		16	0.013	0.027	18.430	0.299
		17	0.050	0.044	19.555	0.298
		18	0.016	0.013	19.672	0.352
		19	0.038	0.014	20.320	0.376
		20	-0.122	-0.109	27.136	0.131
		21	0.026	0.038	27.439	0.157
		22	-0.043	-0.029	28.296	0.166
		23	0.009	-0.001	28.337	0.203
		24	-0.005	0.013	28.347	0.246

Table 28 ADF test after differencing

H_0: *D*(CLOSE) has a unit root			
Exogenous: Constant			
LLgth: 0 (ABSIC, maxlag = 17)			
Prob	*t*-statistic		
0.0000	− 19.91917	AD-FT statistic	
	− 3.445197	1% level	Test critical values:
	− 2.867980	5% level	
	− 2.570265	10% level	

AD-FT Equation				
Dependent variable: *D* (CLOSE, 2)				
Method: LSqr				
Date: 04/10/20 Time: 12:09				
Sample (adjusted): 3436				
Number of observations: 434 after set out				
Prob	*t*-statistic	Std. error	Coefficient	Variable
0.0000	− 19.91917	0.052132	− 1.038421	*D*(CLOSE(− 1))
0.2282	− 1.206605	0.001829	− 0.002207	*C*

− 0.000760	MD var	0.478748	Rsqrd
0.052680	SDD var	0.477541	ARsqrd
− 3.693791	AIC	0.038077	SErgrs
− 3.675021	SC	0.626354	SsqrdR
− 3.686382	HQC	803.5526	LLH
1.843045	DWs	396.7733	F-stat
		0.000000	PF-stat

Table 29 ARIMA forecasting

Dependent variable: *D*(CLOSE)				
Method: BFGS				
Date: 04/10/20 Time: 12:11				
Sample: 2436				
Number of observations: 435				
Convergence achieved after 260 iterations				
Prob	*t*-statistic	Std. error	Coefficient	Variable
0.3289	− 0.977413	0.002174	− 0.002125	*C*
0.0000	− 12.52531	0.050936	− 0.637988	AR(1)
0.0000	13.55085	0.042861	0.580797	AR(2)
0.0000	− 15.98815	0.039263	− 0.627742	AR(3)
0.0000	− 18.12150	0.051270	− 0.929095	AR(4)
0.0000	8.568990	0.072039	0.617300	MA(1)
0.0000	− 15.21496	0.033027	− 0.502512	MA(2)
0.0000	19.08835	0.037062	0.707453	MA(3)
0.0000	13.27848	0.067977	0.902634	MA(4)
0.0000	21.66719	6.36E−05	0.001379	SIGMASQ

− 0.002100	MD var	0.044590	Rsqrd
0.038030	SDD var	0.024358	ARsqrd
− 3.696855	AIC	0.037564	SErgrs
− 3.603169	SC	0.599688	SsqrdR
− 3.659879	HQC	814.0660	LLH
1.871951	DWs	2.203908	F-stat
		0.020938	PF-stat

− 0.97 − 0.16i	− 0.97 + 0.16i	0.65 − 0.74i	0.65 + 0.74i	Inverted AR Roots
− 0.94 − 0.14i	− 0.94 + 0.14i	0.64 + 0.76i	0.64 − 0.76i	Inverted MA Roots

Table 30 ARIMA forecasting summary

Automatic ARIMA Forecasting
Selected dependent variable: *D*(CLOSE)
Date: 04/10/20 Time: 12:11
Sample: 1436
Number of observations: 435
Forecast length: 0
No. E ARMA Ms: 25
No. non-C Es: 0
Selected ARMA model: (4, 4)
AIC value: − 3.69664755219

Table 31 ARIMA forecasting (FTSE100)

Dependent variable: CLOSE
Method: BFGS
Date: 04/10/20 Time: 12:29
Sample: 1443
Number of observations: 443
Convergence achieved after 195 iterations

Prob	*t*-statistic	Std. error	Coefficient	Variable
0.0000	6.645851	0.085732	0.569762	*C*
0.0000	69.94883	0.011468	0.802205	AR(1)
0.0000	− 126.1606	0.006612	− 0.834216	AR(2)
0.0000	91.89403	0.010560	0.970369	AR(3)
0.8239	0.222651	0.713542	0.158871	MA(1)
0.9117	0.111016	9.007671	1.000000	MA(2)
0.8240	0.222526	0.009339	0.002078	SIGMASQ

0.575499	MD var	0.947300	Rsqrd
0.198804	SDD var	0.946574	ARsqrd
− 3.289378	AIC	0.045951	SErgrs
− 3.224694	SC	0.920628	SsqrdR
− 3.263867	HQC	735.5972	LLH
2.043668	DWs	1306.199	F-stat
		0.000000	PF-stat

− 0.08 − 1.00i	− 0.08 + 1.00i	0.97	Inverted AR Roots
	− 0.08 − 1.00i	− 0.08 + 1.00i	Inverted MA Roots

Table 32 ARIMA forecasting summary

Automatic ARIMA Forecasting
Selected dependent variable: CLOSE
Date: 04/10/20 Time: 12:29
Sample: 1443
Number of observations: 443
Forecast length: 0

No. E ARMA Ms: 25
No. non-C Es: 0
Selected ARMA model: (3, 2)
AIC value: − 3.29129512694

Table 33 ARIMA forecasting (NDAQ)

Dependent variable: *D*(CLOSE)
Method: ARMA maximum likelihood (BFGS)
Date: 04/10/20 Time: 12:38
Sample: 2437
Number of observations: 436
Convergence achieved after 33 iterations

Prob	*t*-statistic	Std. error	Coefficient	Variable
0.8122	0.237782	0.001699	0.000404	*C*
0.0000	5.317223	0.016949	0.090119	AR(1)
0.0000	− 45.21530	0.021681	− 0.980324	AR(2)
0.2343	− 1.191029	0.030373	− 0.036175	MA(1)
0.0000	42.23295	0.022787	0.962358	MA(2)
0.0000	20.69903	4.48E−05	0.000927	SIGMASQ

0.000396	MD var	0.051505	Rsqrd
0.031304	SDD var	0.040476	ARsqrd
− 4.114427	AIC	0.030664	SErgrs
− 4.058312	SC	0.404327	SsqrdR
− 4.092281	HQC	902.9450	LLH
1.851005	DWs	4.669934	F-stat
		0.000366	PF-stat

0.05 + 0.99i	0.05 − 0.99i	Inverted AR Roots
0.02 + 0.98i	0.02 − 0.98i	Inverted MA Roots

Table 34 ARIMA forecasting summary

Automatic ARIMA Forecasting
Selected dependent variable: *D*(CLOSE)
Date: 04/10/20 Time: 12:38
Sample: 1437
Number of observations: 436
Forecast length: 0

No. E ARMA Ms: 25
No. non-C Es: 0
Selected ARMA model: (2, 2)
AIC value: − 4.11442664956

Table 35 ARIMA forecasting (S&P500)

Dependent variable: *D*(CLOSE)				
Method: LSqr				
Date: 04/10/20 Time: 12:54				
Sample (adjusted): 2438				
Number of observations: 437 after set out				
Prob	*t*-statistic	Std. error	Coefficient	Variable
0.6572	0.444032	0.001216	0.000540	*C*
0.000540	MD var		0.000000	Rsqrd
0.025411	SDD var		0.000000	ARsqrd
− 4.504975	AIC		0.025411	SErgrs
− 4.495639	SC		0.281536	SsqrdR
− 4.501291	HQC		985.3371	LLH

Table 36 ARIMA forecasting summary

Automatic ARIMA Forecasting
Selected dependent variable: *D*(CLOSE)
Date: 04/10/20 Time: 12:54
Sample: 1438
Number of observations: 437
Forecast length: 0
No. E ARMA Ms: 25
No. non-C Es: 0
Selected ARMA model: (0, 0)
AIC value: − 4.50039880378

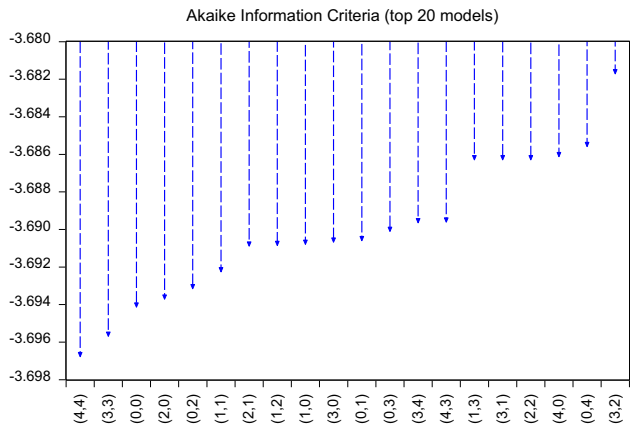


Fig. 13 Akaike information criteria (top 20 models)

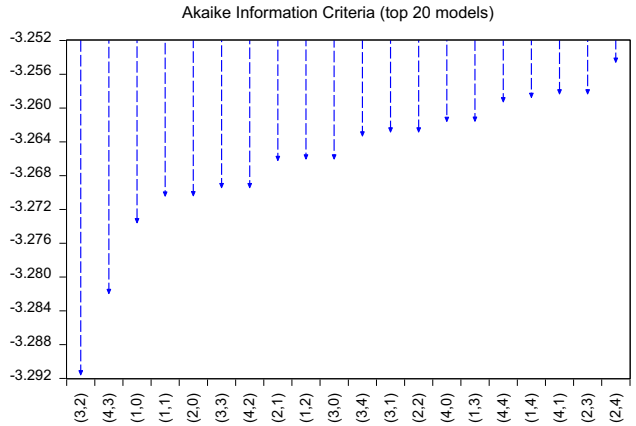


Fig. 14 Akaike information criteria (top 20 models)

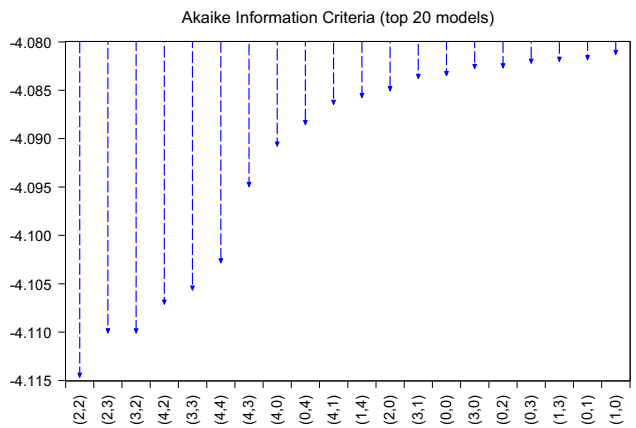


Fig. 15 Akaike information criteria (top 20 models)

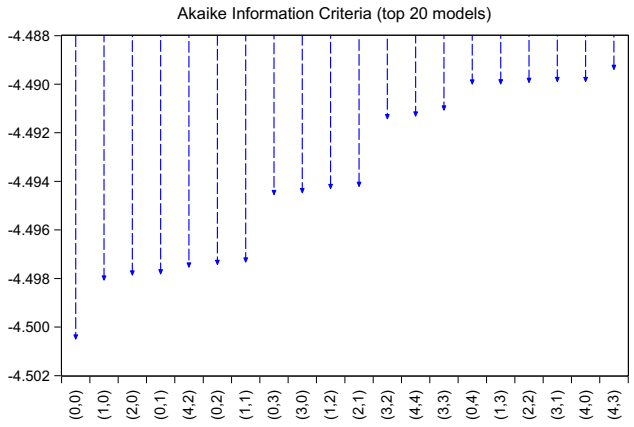


Fig. 16 Akaike information criteria (top 20 models)

Declarations

Conflict of interest The authors state that there is no conflict of interest with the publication of this paper.

Ethical approval The content of this paper doesn't involve any research on human participation.

References

Abdel-Basset M, Abdel-Fatah L, Sangaiah AK (2018) Metaheuristic algorithms: a comprehensive review. In: Sangaiah AK, Sheng M, Zhang Z (eds) Computational intelligence for multimedia big data on the cloud with engineering applications. Elsevier, Amsterdam, pp 185–231

Ahmed MK, Wajiga GM, Blamah NV, Modi B (2019) Stock market forecasting using ant colony optimization based algorithm. Am J Math Comput Model 4(3):52–57

Akaike H (1998) Autoregressive model fitting for control. In: Tanabe K, Kitagawa G, Parzen E (eds) Springer, New York, pp 153–170

Baek Y, Kim HY (2018) ModAugNet: a new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module. Exp Syst Appl 113:457–480

Barakat MR, Elgazzar SH, Hanafy KM (2016) Impact of macroeconomic variables on stock markets: evidence from emerging markets. Int J Econ Finan 8(1):195–207

Beheshti Z, Shamsuddin SMH (2013) A review of population-based meta-heuristic algorithms. Int J Adv Soft Comput Appl 5(1):1–35

Bisgaard S, Kulahci M (2011) Time series analysis and forecasting by example. Wiley, New Jersey

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 35(3):268–308

Burden F, Winkler D (2008) Bayesian regularization of neural networks. In: Livingstone DJ (ed) Artificial neural networks: methods andapplications. Humana Press, Totowa, pp 23–42

Chandana PH (2019) A survey on soft computing techniques and applications. Int Res J Eng Technol 6(4):1258–1266

Chen Y, Hao Y (2017) A feature weighted support vector machine and K-nearest neighbor algorithm for stock market indices prediction. Exp Syst Appl 80:340–355

Chong E, Han C, Park FC (2017) Deep learning networks for stock market analysis and prediction: methodology, data representations, and case studies. Exp Syst Appl 83:187–205

Chou JS, Nguyen TK (2018) Forward forecast of stock price using sliding-window metaheuristic-optimized machine-learning regression. IEEE T Ind Inform 14(7):3132–3142

Clements MP, Hendry DF (2000) Forecasting non-stationary economic time series. MIT Press, Massachusett

Davallou M, Azizi N (2017) The investigation of information risk pricing; evidence from adjusted probability of informed trading measure. Financial Res J 19(3):415–438

Dixon M (2018) Sequence classification of the limit order book using recurrent neural networks. J Comput Sci 24:277–286

Dong G, Fataliyev K, Wang L (2013) One-step and multi-step ahead stock prediction using backpropagation neural networks. In: 2013 9th International conference on information, communications & signal processing, Tainan. IEEE, pp 1–5

Emamverdi G, Karimi MS, Khakie S, Karimi M (2016) Forecasting the total index of Tehran stock exchange. Financial Stud 20(1):54–68

Evangeline D, Abirami T (2019) Social spider optimization algorithm: theory and its applications. Int J Innov Tech Explor Eng 8(10):327–332

Fahad AM, Ahmed AA, Kahar MNM (2018) Network intrusion detection framework based on whale swarm algorithm and artificial neural network in cloud computing. In: Vasant P, Zelinka I, Weber GW (eds) Intelligent computing & optimization. ICO 2018. Advances in intelligent systems and computing. Springer, Cham, pp 56–65

Fang Y, Fataliyev K, Wang L, Fu X, Wang Y (2014) Improving the genetic-algorithm-optimized wavelet neural network for stock market prediction. In: 2014 International joint conference on neural networks (IJCNN), Beijing, pp 3038–3042

Gálvez A, Iglesias A (2016) New memetic self-adaptive firefly algorithm for continuous optimisation. Int J Bio-Inspir Comput 8(5):300–317

Ghanbari M, Arian H (2019) Forecasting stock market with support vector regression and butterfly optimization algorithm. arXiv preprint arXiv:1905.11462

Ghasemiyeh R, Moghdani R, Sana SS (2017) A hybrid artificial neural network with metaheuristic algorithms for predicting stock price. Cybern Syst 48(4):365–392

Göçken M, Özçalıcı M, Boru A, Dosdoğru AT (2016) Integrating metaheuristics and artificial neural networks for improved stock price prediction. Exp Syst Appl 44:320–331

Gogna A, Tayal A (2013) Metaheuristics: review and application. J Exp Theor Artif In 25(4):503–526

Goli A, Khademi Zareh H, Tavakkoli-Moghaddam R, Sadeghieh A (2018) A comprehensive model of demand prediction based on hybrid artificial intelligence and metaheuristic algorithms: a case study in dairy industry. J Ind Syst Eng 11(4):190–203

Golmaryami M, Behzadi M, Ahmadzadeh M (2015) A hybrid method based on neural networks and a meta-heuristic bat algorithm for stock price prediction. In: 2nd International conference on knowledge-based engineering and innovation, Tehran, pp 269–275

Granger CW, Newbold P, Econom J (1974) Spurious regressions in econometrics. J Econ 2(2):111–120

Greco S, Ishizaka A, Tasiou M, Torrisi G (2019) On the methodological framework of composite indices: a review of the issues of weighting, aggregation, and robustness. Soc Indic Res 141(1):61–94

Gülmez B, Kulluk S (2019) Social spider algorithm for training artificial neural networks. Int J Bus Anal 6(4):32–49
Gupta S, Wang LP (2010) Stock forecasting with feedforward neural networks and gradual data sub-sampling. Aust J Intell Inf Process Syst 11(4):14–17
Hadavandi E, Ghanbari A, Abbasian-Naghneh S (2010) Developing an evolutionary neural network model for stock index forecasting. In: Huang DS, McGinnity M, Heutte L, Zhang XP (eds) Advance intelligent computing theories and applications. Springer, Berlin, pp 407–415
Haddad G, Haghighat Monfared J (2012) Assessment of organization performance using combined approach of balanced scorecard and fuzzy analytic network Process (case study: a branch of Iran University of Medical Sciences). Int J Manag Sci Bus Res 1(11):71–86
Hafezi R, Shahrabi J, Hadavandi E (2015) A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price. Appl Soft Comput 29:196–210
Hamilton JD (1994) Time series analysis, vol 2. Princeton, New Jersey
Han J, Kamber M, Tung AK (2001) Spatial clustering methods in data mining. In: Miller HJ, Han J (eds) Geographic data mining and knowledge discovery. Taylor & Francis, London, pp 188–217
Hao Y, Wilamowski B (2011) Levenberg–marquardt training. In: Wiliamowski BM, Irwin JD (eds) Industrial electronic handbook. CRC Press, Boca Raton, pp 12-1-12–16
Hassanat A, Almohammadi K, Alkafaween E, Abunawas E, Hammouri A, Prasath VB (2019) Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach. Information 10(12):390
He Y, Fataliyev K, Wang L (2013) Feature selection for stock market analysis. In: Lee M, Hou ZG, Kil RM (eds) Neural Information Processing. ICONIP 2013. Lecture notes in computer science, vol 8227. Springer, Berlin
Idris MA, Saiang D, Nordlund E (2015) Stochastic assessment of pillar stability at Laisvall mine using artificial neural network. Tunn Undergr Sp Tech 49:307–319
Iglesiasa A, Gálveza A, Suárezb P (2020) Swarm robotics-a case study: bat robotics. In: Yang X (ed) Nature-inspired computation and swarm intelligence: algorithms, theory and application. Academic Press, London, pp 273–302
Jantan A, Ghanem WA, Ghaleb SA (2017) Using modified bat algorithm to train neural networks for spam detection. J Theor Appl 95(24):6788–6799
Kai F, Wenhua X (1997) Training neural network with genetic algorithms for forecasting the stock price index. In: IEEE International conference on intelligent processing systems (Cat. No. 97TH8335), Beijing, China, pp 401–403
Kaveh A, Ghazaan MI (2018) Meta-heuristic algorithms for optimal design of real-size structures. Springer, Cham
Khosravanian R, Mansouri V, Wood DA, Alipour MR (2018) A comparative study of several metaheuristic algorithms for optimizing complex 3-D well-path designs. J Petrol Explor Prod Technol 8(4):1487–1503
Kumar G, Jain S, Singh UP (2020) Stock market forecasting using computational intelligence: a Survey. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-020-09413-5
Kuo R, Han Y (2011) A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem—a case study on supply chain model. Appl Math Model 35(8):3905–3917
Lah MSC, Arbaiy N, Efendi R (2019) Stock market forecasting model based on AR (1) with adjusted triangular fuzzy number using standard deviation approach for ASEAN countries. In: Piuri V, Balas VE, Borah S, Syed Ahmad SS (eds) Intelligent and interactive computing. Springer Singapore, Singapore, pp 103–114
Lim WT, Wang L, Wang Y, Chang Q (2016) Housing price prediction using neural networks. In: 2016 12th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), pp 518–522
Liu G, Wang X (2019) A new metric for individual stock trend prediction. Eng Appl Artif Intell 82:1–12
Luque-Chang A, Cuevas E, Fausto F, Zaldivar D, Pérez M (2018) Social spider optimization algorithm: modifications, applications, and perspectives. Math Probl Eng 2018
Lv D, Yuan S, Li M, Xiang Y (2019) An empirical study of machine learning algorithms for stock daily trading strategy. Math Probl Eng 2019
Ma L, Hu C, Lin R, Han Y (2018) ARIMA model forecast based on EViews software. In: IOP conference series: earth and environmental science, China, Hong Kong, vol 208, pp 1–8
Malkiel BG (1989) Efficient market hypothesis. In: Eatwell J, Newman P (eds) Finance. The New Palgrave. Palgrave Macmillan, London, pp 127–134
Malkiel BG (2003) The efficient market hypothesis and its critics. J Econ Perspect 17(1):59–82
Mirjalili S, Mirjalili S, Yang XS (2014) Binary Bat Algorithm. Neural Comput Appl 25(3–4):663–681
Mirjalili SZ, Saremi S, Mirjalili SM (2015) Designing evolutionary feedforward neural networks using social spider optimization algorithm. Neural Comput Appl 26(8):1919–1928
Mozer MC, Dodier RH, Anderson M, Vidmar L, Cruickshank R, Miller D (1995) The neural network house: an overview. In: Niklasson L, Boden M (eds) Current trends in connectionism. Erlbaum, Hillsdale, pp 371–380
Naseer M, Bin Tariq Y (2015) The efficient market hypothesis: a critical review of the literature. IUP J Financial Risk Manag 12(4):48–63
Nawi NM, Rehman MZ, Khan A (2014) A new bat based back-propagation (BAT-BP) algorithm. In: Swiątek J, Grzech A, Swiątek P, Tomczak J (eds) Advances in systems science. Advances in Intelligent systems and computing. Springer, Cham, pp 395–404
Obthong M, Tantisantiwong N, Jeamwatthanachai W, Wills G (2020) A survey on machine learning for stock price prediction: algorithms and techniques
Oreski S, Oreski D, Oreski G (2012) Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment. Exp Syst Appl 39(16):12605–12617
Osman IH, Kelly JP (1996) Meta-heuristics: an overview. In: Osman IH, Kelly JP (eds) Meta-heuristics. Springer, Boston, pp 1–21
Pai PF, Lin CS (2005) A hybrid ARIMA and support vector machines model in stock price forecasting. Omega 33(6):497–505
Pal SS, Kar S (2019) Time series forecasting for stock market prediction through data discretization by fuzzistics and rule generation by rough set theory. Math Comput Simul 162:18–30
Prasanna S, Ezhilmaran D (2013) An analysis on stock market prediction using data mining techniques. Int J Comput Sci Eng Technol 4(3):49–51
Preethi G, Santhi B (2012) Stock market forecasting techniques: a survey. J Theor Appl 46:24–30
Rajesh P, Srinivas N, Vamshikrishna Reddy K, VamsiPriya G, Dwija V, Himaja D (2019) Stock trend prediction using Ensemble learning techniques in python. Int J Innov Technol Expl Eng 8(5):150–155
Reddy KS, Panwar L, Panigrahi B, Kumar R (2019) Binary whale optimization algorithm: a new metaheuristic approach for profit-based unit commitment problems in competitive electricity markets. Eng Optim 51(3):369–389

Reinsel GC (2003) Elements of multivariate time series analysis. Springer, New York

Saad EW, Prokhorov DV, Wunsch DC (1998) Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. IEEE Trans Neural Netw 9(6):1456–1470

Saber CK, Abghari H, Erfanian M, Gholizadeh S (2013) Short-term model of optimization operation of water resources using particle swarm optimization and compared with genetic algorithm. Water Manag Res 25(4):63–72

Safa M, Panahian H (2018) P/E modeling and prediction of firms listed on the Tehran stock exchange; a new approach to harmony search algorithm and neural network hybridization. Iran J Manag Stud 11(4):765–786

Sahoo S, Mohanty MN (2020) Stock market price prediction employing artificial neural network optimized by gray wolf optimization. In: Patnail S, Ip A, Tavana M, Jain V (eds) New paradigm in decision science and management. Springer, Singapore, pp 77–87

Said SE, Dickey DA (1984) Testing for unit roots in autoregressive-moving average models of unknown order. Biometrika 71(3):599–607

Saravanan R, Sujatha P, Kadiravan G, Uthayakumar J (2019) Social spider optimization with tumbling effect based data classification model for stock price prediction. Int J Innov Tech Explor Eng 8(11):568–578

Sin E, Wang L (2017) Bitcoin price prediction using ensembles of neural networks. In: 2017 13th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), Guilin, pp 666–671

Sedighi M, Jahangirnia H, Gharakhani M, Farahani Fard S (2019) A novel hybrid model for stock price forecasting based on metaheuristics and support vector machine. Data 4(2):75

Sezer OB, Ozbayoglu M, Dogdu E (2017) A Deep neural-network based stock trading system based on evolutionary optimized technical analysis parameters. Procedia Comput Sci 114:473–480

Shin Y (2017) Time series analysis in the social sciences: the fundamentals. University of California Press, California

Sowell F (1992) Modeling long-run behavior with the fractional ARIMA model. J Monet Econ 29(2):277–302

Talbi EG (2009) Metaheuristics: from design to implementation. Wiley, New Jersey

Tsai CF, Dao TK, Yang WJ, Pan TS (2014) Parallelized bat algorithm with a communication strategy. In: Ali M, Pan JS, Chen SM, Horng MF (eds) Modern advances in applied intelligence. Lecture notes in computer science, vol 8481. Springer, Cham, pp 87–95

Tseng FM, Tzeng GH, Yu HC, Yuan BJC (2001) Fuzzy ARIMA model for forecasting the foreign exchange market. Fuzzy Set Syst 118(1):9–19

Versace M, Bhatt R, Hinds O, Shiffer M (2004) Predicting the exchange traded fund DIA with a combination of genetic algorithms and neural networks. Exp Syst Appl 27(3):417–425

Wang L, Chan FF, Wang Y, Chang Q (2016). Predicting public housing prices using delayed neural networks. In: 2016 IEEE region 10 conference (TENCON), pp 3589–3592

Wong W, Ming CI (2019) A review on metaheuristic algorithms: recent trends, benchmarking and applications. In: 2019 7th International conference on smart computing & communications (ICSCC), Sarawak, Malaysia, pp 1–5

Yang XS (2010) A new metaheuristic bat-inspired algorithm. In: González JR, Pelta DA, Cruz C, Terrazas G, Krasnogor N (eds) Nature inspired cooperative strategies for optimization (NISCO 2010). Studies in computational intelligence, vol 284. Springer, Berlin, pp 65–74

Yeh WC (2012) Simplified swarm optimization in disassembly sequencing problems with learning effects. Comput Oper Res 39(9):2168–2177

Yim J, Mitchell H (2002) A comparison of corporate failure models in Australia: hybrid neural networks, logit models and discriminant analysis. Working Paper, RMIT Business, School of Economics and Finance

Zaman S (2019) Weak form market efficiency test of Bangladesh Stock Exchange: an empirical evidence from Dhaka Stock Exchange and Chittagong Stock Exchange. J Econ Account Ventura 21(3):285–291

Zhang GP (2003) Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing 50:159–175

Zhang J, Cui S, Xu Y, Li Q, Li T (2018) A novel data-driven stock price trend prediction system. Exp Syst Appl 97:60–69

Zheng T, Fataliyev K, Wang L (2013) Wavelet neural networks for stock trading. In: Independent component analyses, compressive sampling, wavelets, neural net, biosystems, and nanoengineering XI, vol 8750. International Society for Optics and Photonics, p 87500A

Zhou Z, Gao M, Liu Q, Xiao H (2020) Forecasting stock price movements with multiple data sources: evidence from stock market in China. Phys A 542:123389

Zhu M, Wang L (2010) Intelligent trading using support vector regression and multilayer perceptron's optimized with genetic algorithms. In: The 2010 international joint conference on neural networks (IJCNN), Barcelona, pp 1–5