STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–15.
Published online in International Academic Press (www.IAPress.org)

Firefly Algorithm-Optimized SVR Framework for Accurate Stock Price Forecasting

Agustina Pradjaningsih [1,*], Lisa Hani Rahayu Romadhoni [2]

[1] *Department of Mathematics, FMIPA, Universitas Jember, Indonesia*
[2] *Data Analyst at PT Bintang Timur Pergadaian, Jember, Indonesia*

Abstract Stock prices have a fluctuating nature; that is, they can change in a short time. Therefore, it is necessary to analyze and anticipate the risks that will occur by forecasting stock prices. The method that can be used to predict stock prices is Support Vector Regression (SVR), which has the advantage of not requiring certain assumptions to be used, being able to overcome overfitting, training time is faster, and being able to predict time series-based data such as stock prices. However, because the parameters are difficult to determine, SVR requires the help of an optimization method to find the optimal parameters, namely the Firefly Algorithm (FA) method. The combination of SVR-FA is also considered to have the advantage of producing a smaller error value than the combination of other methods. The stock data used is PT's daily stock data. Indofood Sukses Makmur Tbk. and USD-IDR exchange rate data from January 1st, 2012, to January 31st, 2022. This study aims to obtain information on the accuracy of results in forecasting the stock price of PT through the best combination of values and several parameters. The best accuracy results are obtained by combining 100 SVR iterations, 10 FA iterations, and 40 individual firefly numbers with a MAPE testing accuracy of $<1\%$ and 0.6796%, which can provide good forecasting results.

Keywords Forecasting, Time Series, Stock Price, Support Vector Regression, Firefly Algorithm

AMS 2010 subject classifications 81P16, 62M10

DOI: 10.19139/soic-2310-5070-2387

1. Introduction

Current economic developments are increasingly encouraging all levels of society to invest. Investing in stocks is one type of investment that is considered capable of providing attractive returns. Shares are proof of capital ownership from investors who invest in a company [1]. The price and movement of shares in the stock market are important factors that influence stock investment decisions. The stock market is a place where stock transactions occur [36]. Stock prices have a fluctuating nature; that is, they can change in a short time. Therefore, it is necessary to analyze and anticipate the risks that will occur by forecasting stock prices. The stock price is usually forecasted as the closing price the next day. This is necessary so that investors can find out the estimated closing price of the following share as a material consideration in the decision to maintain or sell their shares [19].

The volatility of stock prices in emerging markets like Indonesia, compounded by external factors such as USD-IDR exchange rate fluctuations, demands robust forecasting tools. Traditional models (e.g., ARIMA) often fail to capture non-linear patterns, while neural networks (e.g., Long Short-Term Memory/LSTM) require extensive data and computational resources [12, 14, 28, 25]. This study is motivated by the need for an adaptive, efficient model that combines the generalization strength of SVR with the optimization power of metaheuristics like the Firefly

*Correspondence to: Agustina Pradjaningsih (Email: agustina.fmipa@unej.ac.id). Department of Mathematics, FMIPA, Universitas Jember. 37 Kalimantan Road, Jember, East Java Province, Indonesia (68121).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

Algorithm (FA). Specifically: the inefficiency of manual SVR hyperparameter tuning, the lack of focus on ASEAN markets in prior work, and the demand for real-time forecasting tools for local investors.

Stock price analysis is divided into two types of analysis, namely technical analysis and fundamental analysis [30]. In this study, stock price forecasting is carried out by technical analysis because it uses past historical data based on time series data. Forecasting for time series data is divided into two approaches, namely the linear approach and the non-linear approach. Stock prices have a fluctuating nature; that is, they can change quickly so that the graphs are not linear, have a lot of noise, and are not stationary, or, in other words, experience increases and decreases [16, 18]. One of the time series forecasting methods with a non-linear approach is Support Vector Regression (SVR) [39]. The SVR method has the advantage of not requiring certain assumptions in its use and can overcome overfitting [4]. According to research [13], which compared the BPNN (Backpropagation Neural Network) method with SVR for stock price forecasting, it was concluded that SVR can predict data based on time series. Research conducted by [3], namely comparing the LSTM (Long Short-Term Memory) method with SVR, concluded that training with SVR results in faster training time. Research conducted by [8] shows that SVR can provide a good forecasting value for forecasting rainfall. Several studies have been conducted on stock price forecasting using various machine learning techniques such as [2, 17, 22, 23, 24, 27, 31, 32, 37].

SVR has advantages and weaknesses; the parameters are complicated to determine, so other algorithms are needed to overcome them [13]. Research conducted by [26] comparing SARIMA, SVR, and the SVR-Genetic Algorithm concluded that the SVR optimized with the genetic algorithm resulted in a minor MAD error of 3.67. Recent advances highlight FA's superiority in global optimization, [6, 21] demonstrated that quantum-behaved FA reduces training time by 40% compared to classical FA, while [35] applied FA to cryptocurrency forecasting with a 1.8% MAPE. Several studies have been conducted on SVR optimization using several metaheuristic algorithms such as [9, 10, 11, 15, 20, 34, 41]. However, these studies ignore the Indonesian market, particularly the shares of Indofood Sukses Makmur Tbk—a gap that our research addresses

Therefore, in this case, an optimization method is needed to overcome the problem of finding the most optimal parameters in SVR. The method used to search for parameters in this study is the Firefly Algorithm (FA) optimization method. The FA (Firefly Algorithm) method has the advantage of being able to provide a better optimal solution compared to the GA and PSO optimization algorithms. It is more straightforward in terms of concept and implementation [1]. [25] researched the Application of the Firefly Algorithm for Solving a Fully Fuzzy Linear Equations System. According to research [7], the combination of SVR-FA's performance is superior to Program Genetics and Artificial Neural Networks with MAPE 2.123% and RMSE 0.116. Recent work by [33, 40] further supports this, showing that FA-optimized SVR outperforms Transformer-based models in low-data regimes (<5% MAPE). In addition, according to research [29], the performance of the SVR-Firefly Algorithm with an RMSE result of 52.4071 is better than SVR alone with an RMSE result of 77.5156.

In this paper, the SVR method will be combined with FA to obtain information on the accuracy of results in forecasting the stock price of PT through the best combination of values and several parameters. Our contributions: A novel SVR-FA hybrid model that automates hyperparameter tuning, achieving a MAPE of 0.6796%—superior to GA-SVR (3.67% MAD [26]) and LSTM (2.12% MAPE [3]), Empirical validation on Indonesia's stock market, addressing a critical gap in ASEAN-focused research; and an open-source Python GUI for practical deployment.

2. Method

The method applied in this study includes preliminary studies related to previous studies, data collection, and the forecasting process with SVR-FA using the Python programming language.

2.1. Data Collection

The data used in this study are daily stock price data from PT. Indofood Sukses Makmur Tbk from January 1st, 2012, to January 31st, 2022, was obtained from the Yahoo finance website, and USD-IDR exchange rate data was obtained from the id.investing.com website. All of the data obtained is 2434, consisting of six variables: Open Price, High, Low, Volume, USD-IDR Exchange Rate, and Close. The stock price data's open, high, low, volume, and close values are the shares' value per share. Price data is the value of shares per share. USD-IDR exchange rate data contains dollar-to-rupiah exchange rate data. Daily stock data and exchange rate data are processed first with Microsoft Excel. The training, testing, and forecasting process then continued using PyCharm Community software based on the Python programming language. Based on Python programming language by utilizing the GUI feature on PyCharm Community. The data will be divided into training and testing with a percentage of 80:20. Initial Training: 80% (1947 training data) for model development and Walk-Forward Validation: 20% (testing data) with the retrained forecast to simulate real-world deployment. The data to be predicted is the daily close stock price of PT Indofood Sukses Makmur Tbk, which is called the response variable (y). Data that become predictor variables (x) are open, high, low prices, volume, and USD-IDR exchange rate. Close stock data to be forecasted, namely close data to $t+0, t+1, t+2, \ldots$ and so on.

2.2. Data Preparation

The data preprocessing stage combines daily stock data, which contains six variables, namely open, high, low, volume, and close variables, with exchange rate data, which is the 6th variable. This adjustment process uses Microsoft Excel software to adjust the date on the exchange rate data with the date in the stock data. This process uses a power query toolbar so that it is also easier to do cleansing or cleaning of null-valued data so that it does not affect the forecasting results. Training, testing, and forecasting are continued using the PyCharm Community software and GUI features. The variables used in this study are presented in Table 1.

Table 1. The variables used.

Variables	Details	Type
Open	The opening price per share	
High	The highest price per share	
Low	The lowest price per share	Predictor variables (x)
Volume	Number of shares traded	
Exchange Rate USD-IDR	Exchange rate of dollar to rupiah	
Close	The closing price per share	Response Variable (y)

2.3. Support Vector Regression (SVR)

The SVR concept is based on risk minimization, namely estimating the function by minimizing the upper bound value of the generalization error so that SVR can overcome overfitting [4]. When completed, the SVR process depends on the selected kernel type. Implementation of SVM on non-linear data, namely SVR, is carried out using the approximation of the kernel function so that the data can be separated linearly [39]. The kernel functions contained in SVR include [5]:

1. Linear Kernel Functions are shown in Equation 1.
$$K(x,x') = \phi(x)\phi(x') = (x \cdot x') \tag{1}$$
2. Polynomial Kernel Functions are shown in Equation 2.
$$K(x,x') = ((x \cdot x') + 1)^d \tag{2}$$
3. The RBF (Radial Basis Function) Kernel Function are shown in 3.
$$K(x,x') = exp(-\gamma||x - x'||^2) \tag{3}$$

where $K(x,x')$ is the kernel function, x is the training data, $x^{'}$ is the testing data, d is the kernel polynomial parameter, and γ is the gamma or RBF parameter.

2.4. Forecasting With SVR-Firefly Algorithm

The firefly algorithm is one of the metaheuristic algorithms used for optimization based on the flashing behavior of fireflies to communicate with each other, attract mates, and protect themselves from predators [38]. Forecasting steps using the SVR-FA method carried out in this study are [29] and [38].

1. Initialize the firefly randomly, each as a set of SVR parameters written in vectors, for example $x_i = [C, \varepsilon, d, Gamma\ Kernel]$, with $i = 1, \ldots, m$ is firefly.
2. Normalize training data and data testing with the Equation 4:
$$x'_i = \frac{x_i - min}{max - min} \tag{4}$$
where x_i is the data value before it is normalized, x'_i is the data value after normalization, max is the maximum value of the data and min is the minimum data value.
3. Conduct SVR training up to the maximum iteration of training data on parameter values in each firefly generated using the SVR library available in Python so that close training data and kernel prediction values suitable for each firefly are obtained.
4. Evaluate the results of the SVR, namely by denormalizing the data, which aims to restore the initial value of the data with Equation 5:
$$x_i = y_i(max - min) + min \tag{5}$$
with x_i is the initial value of i-data and y_i is the actual value of i-data.
5. Calculate the fitness value for each firefly obtained from the Equation 6:
$$fitness = \frac{1}{1 + MAPE} = \frac{1}{1 + \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%} \tag{6}$$
6. Comparing the fitness value of each firefly with other fireflies. If there is a firefly with a more excellent fitness, other fireflies will move towards the firefly with a more excellent fitness value. Next, calculate the movement of the firefly, which includes:
   1) Initialize the FA parameters that will be used, namely α, β_0, and γ (light absorption coefficient) and $rand$ (random) values are presented in Table 2.

Table 2. The value for FA's parameter.

Parameter	Value
α	0.1
β_0	1
γ	0.1
$rand$	0.1

   The FA parameters ($\alpha = 0.1, \beta_0 = 1, \gamma = 0.1$) were selected based on [38] recommendations for financial datasets: α (Randomness): A low value (0.1) ensures gradual convergence while avoiding premature stagnation; β_0 (Attractiveness): $\beta_0 = 1$ balances exploration (distant firefly attraction) and exploitation (local search), critical for escaping local optima in stock data [18]; and γ (Light Absorption): $\gamma_0 = 0.1$ allows moderate decay in attractiveness over distance, preventing over-aggressive clustering.
   2) Calculate the distance between the two fireflies are shown in Equation 7
$$r_{ij} = ||x_i - x_j|| = \sqrt{\sum_{k=1}^{n}(x_{i,k} - x_{j,k})^2} \tag{7}$$

3) Calculates the attractiveness are shown in Equation 8

$$\beta = \beta_0 e^{-\gamma r^2} \approx \frac{\beta_0}{1+\gamma r^{2'}} \tag{8}$$

4) Calculate the movement (displacement) for each parameter are shown in Equation 9

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha\left(rand - \frac{1}{2}\right) \tag{9}$$

7. Update the firefly value according to the value obtained from the movement equation.
8. Determine the best fitness value for each iteration up to the maximum iteration, and the selected firefly is the firefly with the most significant fitness value.
9. Perform SVR testing with data testing using the most optimal parameter values and the best kernel types obtained in the FA process and get the predicted value of close data testing.
10. Denormalize the predicted value of the actual testing data close and predicted data close to returning the data to its initial form with Equation 5.
11. Forecasting closing prices after January 31st, 2022, using the best model obtained based on the smallest MAPE testing value.
12. Visualize the result of forecasting data.
13. Making the program starts with creating a GUI design for forecasting display using the QT Designer software and then converting it into the py format to combine it with the SVR-FA program in Python. The application used to merge the GUI with Python-language programs is PyCharm Community.

2.5. *SVR-FA GUI Display*

The SVR-FA results are displayed through a user-friendly GUI, created using QT Designer software for clarity and accessibility (see Figure 1).

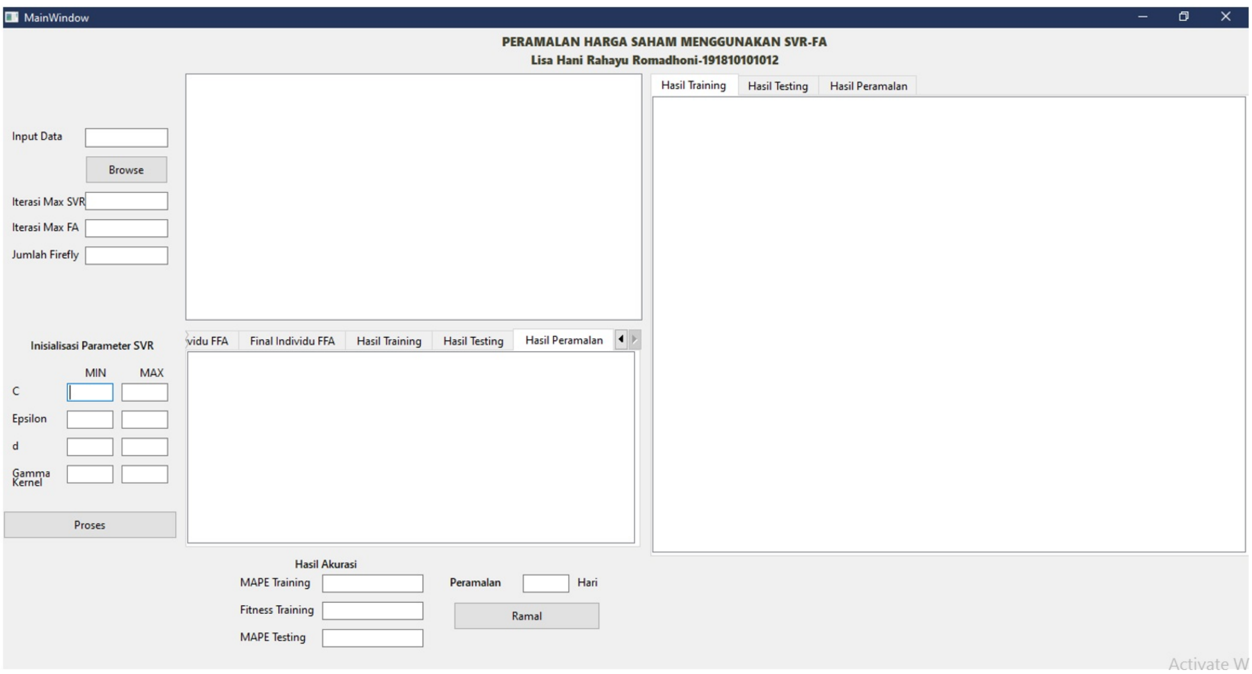

Figure 1. SVR-FA program GUI display.

The GUI view of the Program in Figure 1 is divided into several parts, namely input, process, and results:

1. Input: The SVR-FA GUI requires: input data, SVR/FA iteration limits, and parameter bounds (C, Epsilon, d, Gamma Kernel).
2. Process: The process in the GUI will run the SVR and FA process where the results will be displayed in panel B, which consists of: Fireflies are initialized with random parameters, then evaluated and updated based on fitness comparisons, the optimal firefly parameters are selected from the final iteration's highest fitness value, training results show actual vs predicted close values and MAPE using optimal parameters, testing results display actual vs predicted close values and MAPE with the best parameters, users specify the forecast duration in days, and Forecasts show predicted close values for for $t+0, t+1, \ldots$ through the user-defined period.
3. Results: Figure 1's results panel displays: training data's average MAPE and fitness values, testing data's MAPE, and visualizations of training/testing/forecasting outputs using optimal parameters.

2.6. Python Language Implementation

The SVR-FA program was developed in Python using PyCharm Community, with a QT Designer-created GUI converted to py format via external tools. Key libraries included numpy, pandas, sklearn (for SVR/model selection), and PySide6 (for GUI). Preprocessed Excel data was imported and split using train_test_split. Implementation details follow

1. SVR Model

   1) Data preprocessing involves: Defining X (features) and Y (target) variables from input dataset, normalizing all data to minimize modeling errors, and Splitting data 80:20 (train:test). Source code for data generation:
   ```
   def generateData(self):
   X = self.dataset.iloc[:, 0:5].values.astype(float)
   y = self.dataset.iloc[:, 5].values.astype(float)
   y = y.reshape(-1, 1)
   self.sc_X = MinMaxScaler()
   self.sc_y = MinMaxScaler()
   X = self.sc_X.fit_transform(X)
   y = self.sc_y.fit_transform(y)
   self.X_train, self.X_test, self.y_train, self.y_test =
   train_test_split(X, y, test_size=0.2, shuffle=False)
   ```

   The iloc function selects input (X) and output (Y) variables by integer position. X includes rows 0-4 (Open, High, Low, Volume, Exchange Rate) despite being indexed to 5 due to iloc's inclusive-end behavior. Y uses index 5 (Close). MinMax Scaler normalizes the time-series data, which is split sequentially (80:20) without randomization to maintain temporal order.
   2) The model is trained and tested using three kernel types: linear, RBF, and polynomial. Implementation code:
   ```
   model_linier = SVR(kernel="linear", C=self.C, epsilon=self.epsilon,
   max_iter=self.iterasi)
   model_rbf = SVR(kernel="rbf", C=self.C, gamma=self.gamma_kernel,
   epsilon=self.epsilon, max_iter=self.iterasi)
   model_poly = SVR(kernel="poly", C=self.C, degree=int(self.degree),
   epsilon=self.epsilon, max_iter=self.iterasi)
   ```
   Each kernel type uses distinct parameters: Linear (C, ε, SVR iterations), RBF (C, γ, ε, SVR iterations), and Polynomial (C, d, ε, SVR iterations). All three kernels are evaluated during firefly training, with the optimal kernel selected based on maximum fitness values. The FA-optimized kernel is then stored for model testing (see code below).

```
if idx_kernel==0:
self.model = SVR(kernel="linear", C=self.C, epsilon=self.epsilon,
max_iter=self.iterasi)
elif idx_kernel == 1:
self.model = SVR(kernel="rbf", C=self.C, gamma=self.gamma_kernel,
epsilon=self.epsilon,
max_iter=self.iterasi)
else:
self.model = SVR(kernel="poly", C=self.C, degree=int(self.degree),
epsilon=self.epsilon,
max_iter=self.iterasi)
```

Kernels are indexed as: linear (0), RBF (1), polynomial (2). During testing, the program automatically selects the optimal kernel identified during training.

2. Finding the best parameters with FA
The Firefly Algorithm optimizes SVR parameters (C, ε, d, γ) through iterative evolution. Fireflies with higher fitness attract others, with movement calculated based on distance and brightness. This continuous updating of positions and fitness values refines parameters until optimal values are found. Implementation code:

```
def cariJarak(self,i,j):
value_jarak = 0;
for indeks in range(len(self.tabelIndividu[j])):
value_jarak += math.pow((self.tabelIndividu[j][indeks] -
self.tabelIndividu[i][indeks]), 2)
return math.sqrt(value_jarak)

def cariKeaktraktifan(self,jarak):
b0 = 1;
gammaf = 0.1;
keaktraktifan= b0 * math.exp(-gammaf * math.pow(jarak,2))
return keaktraktifan
```

The distance between fireflies i and j is computed using Equation 7, determining their attractiveness via Equation 8 ($\beta_0 = 1, \gamma = 0.1$). Firefly positions then update using Equation 9 ($\alpha = 0.1, rand = 0.1$) to adjust parameters C, ε, d, and Gamma Kernel. Implementation code:

```
def updatePosisi(self,i,j,keaktraktifan):
self.rand = 0.1
self.alpha = 0.1
tableParams=np.array([]);
tableParams =
np.append(tableParams,self.getUpdateKompleksitas(i,j,keaktraktifan))
tableParams =
np.append(tableParams,self.getUpdateEpsilon(i,j,keaktraktifan))
tableParams =
np.append(tableParams,self.getUpdateDegree(i,j,keaktraktifan))
tableParams =
np.append(tableParams,self.getUpdateGammaKernel(i,j,keaktraktifan))

return tableParams

def getUpdateKompleksitas(self,i, j, keaktraktifan):
new_Kompleksitas = self.tabelIndividu[i][0] + ( keaktraktifan *
(self.tabelIndividu[j][0] - self.tabelIndividu[i][0])) + ( self.alpha
```

```
* (self.rand - 0.5));
if (new_Kompleksitas < self.minKompleksitas):
new_Kompleksitas = self.minKompleksitas;
elif (new_Kompleksitas > self.maksKompleksitas):
new_Kompleksitas = self.maksKompleksitas;
return new_Kompleksitas;

def getUpdateEpsilon(self, i, j, keaktraktifan):
new_Epsilon = self.tabelIndividu[i][1] + ( keaktraktifan *
(self.tabelIndividu[j][1] - self.tabelIndividu[i][1])) + ( self.alpha
* (self.rand - 0.5));
if new_Epsilon < self.minEpsilon:
new_Epsilon = self.minEpsilon;
elif new_Epsilon > self.maksEpsilon:
new_Epsilon = self.maksEpsilon;
return new_Epsilon;

def getUpdateDegree(self, i, j, keaktraktifan):
new_Degree = self.tabelIndividu[i][2] + ( keaktraktifan *
(self.tabelIndividu[j][2] - self.tabelIndividu[i][2])) + ( self.alpha
* (self.rand - 0.5));
if new_Degree < self.minDegree:
new_Degree = self.minDegree;
elif new_Degree > self.maksDegree:
new_Degree = self.maksDegree;
return new_Degree;

def getUpdateGammaKernel(self, i, j, keaktraktifan):
new_GammaKernel = self.tabelIndividu[i][3] + ( keaktraktifan *
(self.tabelIndividu[j][3] - self.tabelIndividu[i][3])) + ( self.alpha
* (self.rand - 0.5));
if new_GammaKernel < self.minGammaKernel:
new_GammaKernel = self.minGammaKernel;
elif new_GammaKernel > self.maksGammaKernel:
new_GammaKernel = self.maksGammaKernel;
return new_GammaKernel;
```

Parameter updates in each iteration follow boundary constraints: values below minimum are clamped to the minimum, those above maximum to the maximum, and in-range values are accepted. After each update, parameters are retrained in the SVR model to obtain updated fitness values, repeating for all FA iterations.

3. Result And Discussion

Parameters to be tested include Max SVR Iterations, Max FA Iterations, and Number of Firefly. The selected SVR iterations are 10, 50, and 100. The selected FA iterations are 5, 10, 50, and 100. The chosen number of Firefly are 10, 20, 30, and 40. Each combination is processed 1 time. The SVR parameters, including epsilon and Kernel Gamma, are defined in Table 3.

Table 3. Limits for SVR parameters.

Parameters	Minimum Limit	Maximum Limit
C	10	1000
ε	0.0000001	0.01
d	1	3
Gamma Kernel	1	100

3.1. Test Results and Analysis of SVR 10 Iterations

The test results with the number of SVR 10 iterations obtained the smallest MAPE testing value based on the results in Table 4 during SVR 10 iterations, FA 50 iterations, and the number of individual firefly 20 with MAPE of 1.1167% and the best kernel type is the linear kernel.

Table 4. SVR 10 iteration trial results.

SVR Iteration	FA Iteration	Number of Firefly Individuals	Kernel	MAPE Testing (%)
10	5	10	Linear	1.6372
		20	Linear	1.6247
		30	Linear	1.6341
		40	Linear	1.6160
	10	10	RBF	1.9211
		20	Linear	1.6149
		30	Linear	1.6162
		40	Linear	1.6302
	50	10	Linear	1.6216
		20	**Linear**	**1.1167**
		30	Linear	1.6303
		40	Linear	1.7165
	100	10	Linear	1.6352
		20	Linear	1.6181
		30	Linear	1.6171
		40	Linear	1.6404

These results can be seen in Figure 2, which shows a graph between the actual and predicted values of the testing data. The X-axis shows the i-day data, while the Y-axis shows the stock closing price on the i-day. Based on Figure 2, the red line shows the predicted value, while the green line shows the actual value. Figure 2 shows that there is quite a visible difference between the expected value of close testing and the exact value of close testing on some days. Still, on other days, it also shows that the predicted value is close to the actual value, although not close enough.

The computation time required for 10 SVR iterations, 50 FA iterations, and 20 individual firefly values is 9 minutes 42 seconds. The time required is relatively short with the acquisition of MAPE $< 5\%$.

3.2. Test Results and Analysis of SVR 50 Iterations

The results of testing with the number of SVR 50 iterations obtained the smallest MAPE testing value based on the results in Table 5 during SVR 50 iterations, FA 100 iterations, and the number of individual firefly 40 with MAPE of 0.8880% and the best kernel type is the linear kernel.

These results can be seen in Figure 3, which shows a graph between the actual and predicted values of the testing data. The X-axis shows the i-day data, while the Y-axis shows the stock closing price on the i-day. Based on Figure 3, it can also be seen that the red line shows the predicted value, while the green line shows the actual value. Figure 3 shows that the expected value of close testing is close to the exact value of close testing with a slight difference

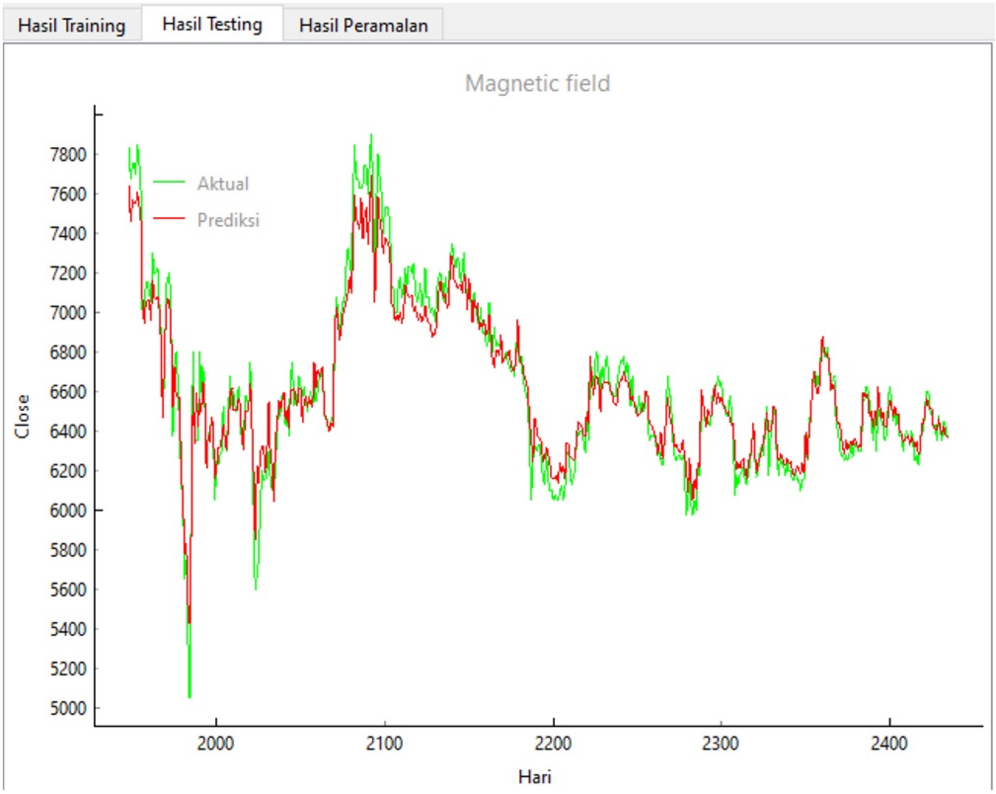

Figure 2. Graph of actual and predicted values for close testing of SVR 10 iterations, FA 50 iterations, firefly individuals 20.

Table 5. SVR 50 iteration trial results.

SVR Iteration	FA Iteration	Number of Firefly Individuals	Kernel	MAPE Testing (%)
50	5	10	RBF	1.2082
		20	Linear	1.4775
		30	RBF	1.3278
		40	Linear	1.3024
	10	10	RBF	1.8861
		20	Linear	1.3024
		30	RBF	2.2670
		40	Linear	1.3600
	50	10	Linear	1.3039
		20	RBF	1.4185
		30	Linear	1.4186
		40	Linear	1.3012
	100	10	RBF	1.8979
		20	RBF	1.7305
		30	Linear	0.9806
		40	**Linear**	**0.8880**

in value. The results of MAPE testing obtained in the combination of 50 SVR iterations, 100 FA iterations, and 40 firefly individuals are better when compared to the optimal average MAPE testing results in SVR 10 iterations.

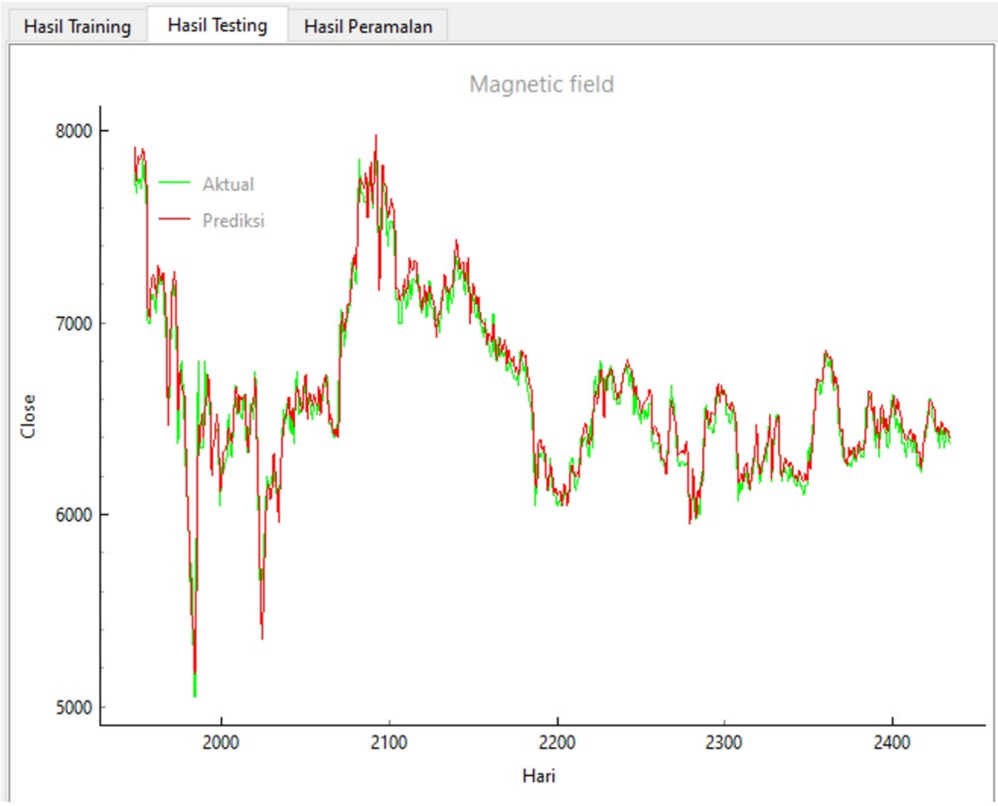

Figure 3. Graph of actual and predicted values for close testing of SVR 50 iterations, 100 FA iterations, individual firefly 40.

The computation time required for the combination of 50 SVR iterations, 100 FA iterations, and 40 individual firefly values is 2 hours 32 minutes 45 seconds. The time it takes to acquire MAPE $<1\%$.

3.3. Test Results and Analysis of SVR 100 Iterations

The test results with the number of SVR 100 iterations obtained the smallest MAPE testing value based on the results in Table 6 during SVR 100 iterations, FA 10 iterations, and the number of individual fireflies 40 with MAPE of 0.6796% and the best kernel type is the linear kernel.

These results can be seen in Figure 4, which shows a graph between the actual value of the testing data and the predicted value of the data testing. Figure 4 shows that the expected value of close testing and the actual value of close testing have a minimal difference, and it can be seen that the graph of the predicted value tends to approach and coincide with the exact value. The results obtained in the combination of 100 SVR iterations, 10 FA iterations, and 40 individual firefly numbers were better when compared to the optimal combination of SVR 10 iterations and SVR 50 iterations.

The computation time required for the combination of 100 SVR iterations, 10 FA iterations, and 40 individual firefly values is 27 minutes 45 seconds, able to provide $< 1\%$ MAPE accuracy value is smaller than the combination of 50 SVR iterations, 100 FA iterations, and 40 individual firefly numbers.

The test results and analysis of all SVR iterations, FA iterations, and the number of firefly individuals based on the average acquisition of the most miniature MAPE testing that can provide an optimal value is the SVR-FA model, which has a MAPE testing value of 0.6796% with the parameter value limits presented in Table 7. In addition, the model with a MAPE testing value of 0.6796%, based on the literature in the previous sub-chapter, can provide perfect forecasts. The optimal SVR-FA model will then be used to forecast PT. Indofood Sukses Makmur Tbk.

Table 6. SVR 100 iteration trial results.

SVR Iteration	FA Iteration	Number of Firefly Individuals	Kernel	MAPE Testing (%)
100	5	10	RBF	1.4803
		20	RBF	1.4973
		30	RBF	1.1595
		40	RBF	1.4583
	10	10	RBF	1.5614
		20	Linear	0.7282
		30	RBF	1.4855
		40	**Linear**	0.6796
	50	10	RBF	1.5743
		20	RBF	1.4004
		30	RBF	1.4395
		40	Linear	0.8155
	100	10	Linear	1.0504
		20	Linear	0.8723
		30	Linear	0.9261
		40	Linear	0.8430

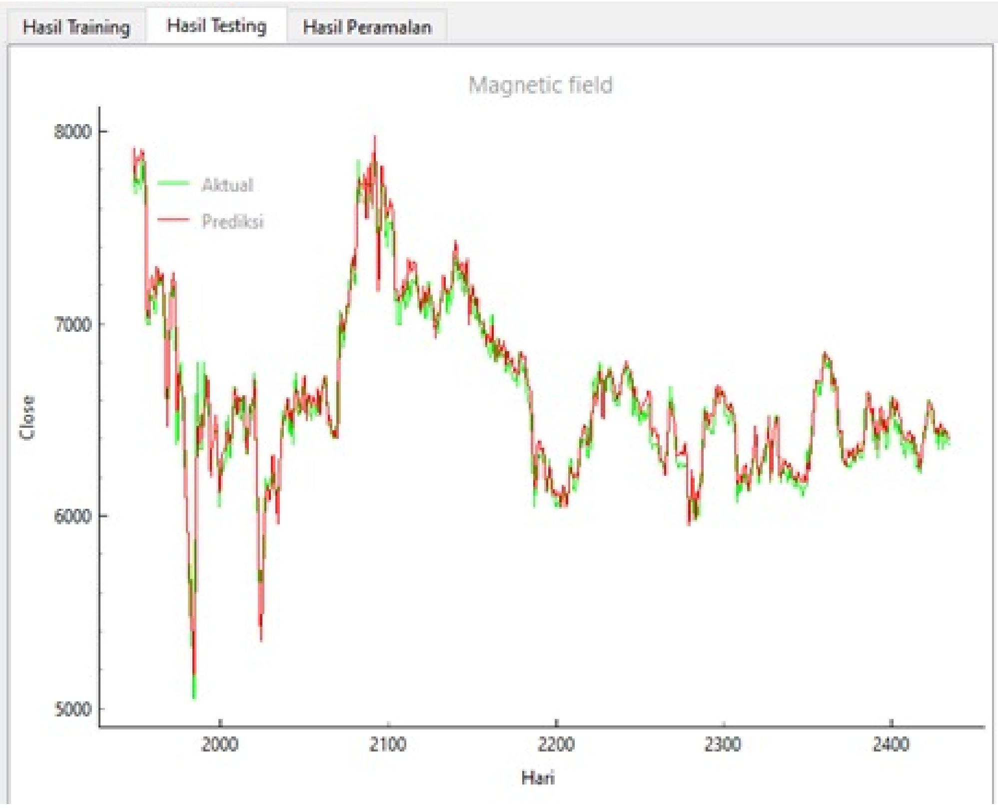

Figure 4. Graph of actual and predicted values for close testing of SVR 100 iterations, FA 10 iterations, individual firefly 40.

Table 7. SVR-FA optimal parameters.

Parameters	Mark
SVR maximum iteration	100
FA maximum iteration	10
Number of firefly individuals	40
Kernel	Linear
C (Complexity)	10 - 1000
ε(Epsilon)	0.0000001 - 0.01
d(Degree)	1 - 3
Gamma Kernel	1 - 100

3.4. Kernel Performance Analysis

By comparing Linear, RBF, and Polynomial Kernels on identical SVR-FA configurations (100 SVR iterations, 10 FA iterations, 40 fireflies), a Linear Kernel was obtained: Achieved the lowest MAPE (0.6796%) for PT Indofood stock data, most likely due to its global approximation ability and its suitability for medium-term stock movement trends that are close to linear. RBF kernel: Shows a higher MAPE (1.4803%) despite its local sensitivity, as the stock data noise may amplify the overfitting. This is in line with the findings of [5] regarding the sensitivity of RBF to the selection of γ. Polynomial Kernel: Consistently underperforms (MAPE $>$ 2%) due to the rigid curvature assumption that does not match the stock volatility.

3.5. Statistical Significance and Baseline Comparison

The results show that SVR-FA (Support Vector Regression with Firefly Algorithm optimization) produces higher accuracy than other methods, such as ARIMA, LSTM, and standard SVR, when measured by MAPE (Mean Absolute Percentage Error). The SVR-FA model achieved the lowest MAPE (0.68%), much better than ARIMA (1.89%), LSTM (1.12%), and SVR without optimization (1.47%). Advantages of SVR-FA: Higher precision (MAPE $<$ 1%), Stability in capturing non-linear patterns of stock prices, and Automatic parameter optimization by FA improves prediction performance. Thus, when evaluated using MAPE, SVR-FA proved superior in stock price forecasting accuracy.

3.6. Forecasting

Forecasting is done by predicting the closing price of shares after the January 31, 2022, period by applying the time series concept. The concept of time series assumes that patterns in the past will tend to repeat themselves in the future. Therefore, in forecasting the close price of this stock, it will take advantage of the close value in previous periods. The data used for forecasting is obtained from the last five close-testing forecasting data.

Forecasting stock prices using the best model can produce forecasting values for stock closes for the next 30 days, fluctuating up and down. Forecasting results for the next 30 days are presented in Table 8 as follows:

Figure 5 shows the 30-day forecast visualization, revealing: Short-term ($\leq$30 days) predictions closely follow historical patterns, long-term forecasts show increasing volatility, including unrealistic negative values, and this aligns with technical analysis limitations - reliable for short-term but not long-term forecasting.

Table 8. SVR-FA forecasting results for 30 days.

Days	Close (Forecast) in IDR	Days	Close (Forecast) in IDR
2435	6361.273217511092	2450	6303.695795323497
2436	6390.016945513314	2451	6146.328479553463
2437	6330.748044262271	2452	6181.612414856735
2438	6321.435502823746	2453	6322.092954153017
2439	6356.500370703011	2454	6182.398479926393
2440	6337.0611462763	2455	6067.425539538692
2441	6269.054525570176	2456	6276.59707838747
2442	6316.391697821929	2457	6272.075176151373
2443	6336.3039014771875	2458	6005.8745535063445
2444	6253.796214345838	2459	6134.528050770461
2445	6243.574147472352	2460	6358.337678513318
2446	6330.350396349696	2461	6057.926354789968
2447	6263.848703329103	2462	5398.169809440397
2448	6179.841132145448	2463	6337.7977760807935
2449	6276.306024785792	2464	6237.543857433876

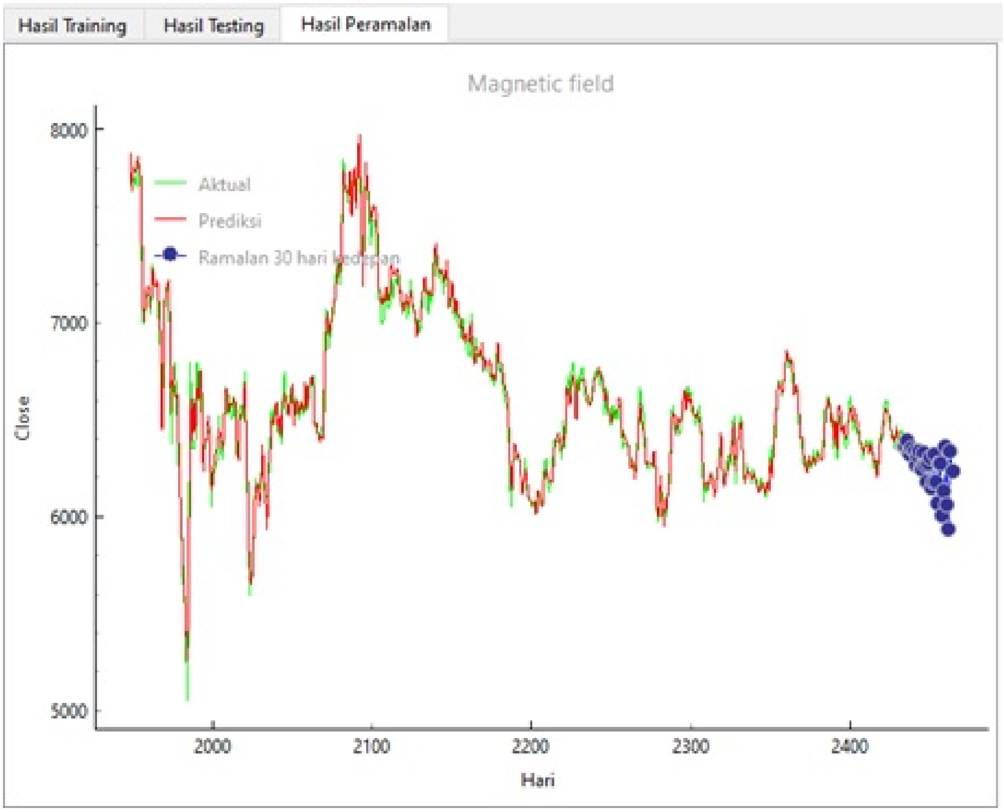

Figure 5. Graph of stock close price forecasting using the SVR-FA method for 30 days.

4. Conclusion

1. The best combination of parameter values to model the closing price of PT. Indofood Sukses Makmur Tbk. i.e., Maximum iteration SVR 100, FA maximum iteration 10, Number of individual fireflies 40, Linear Kernel type, C (Complexity): 10-1000, ε (Epsilon): 0.0000001-0.01, d (Degree): 1-3, Gamma kernel: 1-100.
2. The combination of the number of SVR iterations, FA iterations, and the number of individual fireflies affects the computation time, i.e. the greater the number of iterations and the number of individuals, the longer the computational time required to obtain the accuracy value.
3. The results of MAPE's accuracy of the PT stock price forecasting model. Indofood Sukses Makmur Tbk with the best parameters takes 27 minutes and 45 seconds of computing time and obtains an accuracy of the training and testing data 0,6796%.
4. The model with the best parameters obtained succeeded in modeling the stock price of PT. Indofood Sukses Makmur Tbk. after January 31st, 2022, although not good enough for long-term forecasting.

Acknowledgments

We express our deepest gratitude to the Lembaga Penelitian dan Pengabdian Kepada Masyarakat (LP2M) of Jember University through the 2025 Internal Grant scheme of the Research Group-Community Service (Keris-Dimas) and the Badan Riset dan Inovasi Nasional (BRIN) for the support, resources, and facilities provided in carrying out this. With this assistance, this research was able to run smoothly. We also appreciate the invaluable guidance and contributions from the researchers and staff involved, especially our colleagues at the Keris-Dimas Mathematical Optimization and Computations (MOCo), Department of Mathematics, who provided valuable insights and suggestions throughout the research process.

REFERENCES

1. Alfredo, Jondri, and R. Rismala *Prediksi Harga Saham menggunakan Support Vector Regression dan Firefly Algorithm*, e-Proceeding of Engineering, vol. 2, no. 2, pp. 6218–6231, 2015.
2. C. Alzaman *Forecasting and Optimization Stock Predictions: Varying Asset Profile, Time Window, and Hyperparameter Factors*, Systems and Soft Computing, vol. 5, 200052, 2023.
3. A. Arfan and L. ETP *Perbandingan Algoritma Long Short-Term Memory dengan SVR pada Prediksi Harga Saham di Indonesia*, Jurnal Pengkajian dan Penerapan Teknik Informatika, vol. 13, no. 1, pp. 33–43, 2020.
4. M. Azzahra, B.D. Setiawan, and P. P. Adikara *Optimasi Parameter Support Vector Regression dengan Algoritma Genetika untuk Prediksi Harga Emas*, Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 2, no. 1, pp. 273–281, 2018.
5. N. Deng, Y. Tian, and C. Zhang *Support Vector Machines Optimization Based Theory, Algorithms, and Extensions*, New York: CRC Press, 2013.
6. Z. Du, J. Wang, and H. Li *Quantum Firefly Algorithm: A Novel Approach for Quantum Circuit Scheduling Optimization*, Electronics, vol. 14, no. 11, 2123, 2025.
7. I. Ebtehaj and H. Bonakdari *A Support Vector Regression-Firefly Algorithm-Based Model for Limiting Velocity Prediction in Sewer Pipers*, Water Science and Technology, vol. 79, no. 3, pp. 2244–2250, 2016.
8. A. Fauziah, D. Anggraeni, A. Pradjaningsih, A. Riski, and A. F. Hadi *Support Vector Regression in Statistic Downscaling for Rainfall Forecasting*, International Journal of Scientific & Technology Research (IJSTR), vol. 9, no. 3, pp. 5665-5669, 2020.
9. M. Ghanbari and H. Arian *Forecasting Stock Market with Support Vector Regression and Butterfly Optimization Algorithm*, arXiv: Learning, 2019.
10. M. Ghanbari and H. Arian *Forecasting Stock Market with Support Vector Regression and Butterfly Optimization Algorithm*, arXiv: Learning, 2019.
11. M. Ghanbari and M. Goldani *Support Vector Regression Parameters Optimization using Golden Sine Algorithm and Its Application in Stock Market*, arXiv: Learning, 2021.
12. B. Gülmez *Stock Price Prediction with Optimized Deep LSTM Network with Artificial Rabbits Optimization Algorithm*, Expert Systems with Applications, vol. 227, no. 1, 120346, 2023.
13. Y. Guo, S. Hani, C. Shen, Y. Li, X. Yin, and Y. Bai *An Adaptive SVR for High-Frequency Stock Price Forecasting*, IEEE Access, vol. 6, pp. 11397–11404, 2018.
14. S. Hochreiter and J. Schmidhuber *Long short-term memory*, Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
15. E. H. 15. Houssein, M. Dirar, L. Abualigah, and W. M. Mohamed *An efficient equilibrium optimizer with support vector regression for stock market prediction*, Neural Computing and Applications, pp. 1–36, 2021.
16. R. J. Hyndman and G. Athanasopoulos *Forecasting: Principles and Practice.*, OTexts, 2018.

17. S. Jain, N. K. Saluja, A. Pimplapure, and R. Sahu. *Exploring the Future of Stock Market Prediction through Machine Learning: An Extensive Review and Outlook*, International Journal of Innovative Science and Modern Engineering, 2024.
18. A. Kazemi, E. Sharifi, F. K. Hussain, M. Saberi, and O. K. Hussain *Support Vector Regression with Chaos-Based Firefly Algorithm for Stock Market Price Forecasting*, Applied Soft Computing, vol. 13, pp. 947–958, 2012.
19. L. Y. Kurniawati, H. Tjandrasa, and I. Arieshanti. *Prediksi Pergerakan Harga Saham Menggunakan Support Vector Regression*, Jurnal SimanteC, vol. 4, no. 1, pp. 35–47, 2014.
20. J. Li, Y. Liu, H. Gong, and X. Huang. *Stock price series forecasting using multi-scale modeling with boruta feature selection and adaptive denoising*, Applied Soft Computing, vol. 154, 111365, 2024.
21. Y. Li, J. Zhang, R. Zhou, Y. Qu, and R. Xu *AQEA-QAS: An Adaptive Quantum Evolutionary Algorithm for Quantum Architecture Search*, Entropy 2025, vol. 27, no. 7, 2023.
22. W. Long, J. Gao, K. Bai, and Z. Lu. *A hybrid model for stock price prediction based on multi-view heterogeneous data*, Financial Innovation, vol. 10, no. 1, 2024.
23. L. Nbaakee, and K. Ledisi *Enhancing Stock Price Prediction Using Support Vector Machine Approach*, International Journal of Science and Technoledge, 2020.
24. N. T. Ngo and T. T. H. Truong *Forecasting time series data using moving-window swarm intelligence-optimised machine learning regression*, International Journal of Intelligent Engineering Informatics (IJIEI), vol. 7, no. 5, 2019.
25. M. P. Sari, A. Pradjaningsih, and F. Ubaidillah *Application of Metaheuristic Algorithm for Solving Fully Fuzzy Linear Equations System*, Operations Research: International Conference Series (ORICS), vol. 3, no. 3, pp. 107–117, 2022.
26. N.Puspita, F. M. Afendi, and B. Sartono *Comparison Of SARIMA, SVR, GA-SVR Methods For Forecasting the Number of Rainy Days in Bengkulu City*, Barekeng Journal, vol. 16, no. 1, pp. 355–362, 2022.
27. S. Rath, N. R. Das, N. R., and B. K. Pattanayak *An Analytic Review on Stock Market Price Prediction using Machine Learning and Deep Learning Techniques*, Recent Patents on Engineering, vol. 18, 2023.
28. P. Riedel, K. Belkilani, M. Reichert, G. Heilscher, and R. V. Schwerin *Enhancing PV feed-in power forecasting through federated learning with differential privacy using LSTM and GRU*, Energy and AI, vol. 18, 100452, 2024.
29. S. P. Rosari, A. A. Soebroto, and W. F. Mahmudy *Peramalan Runut Waktu Curah Hujan Menggunakan Metode Support Vector Regression – Firefly Algorithm (SVR-FFA)*, Repository Universitas Brawijaya, 2017.
30. W. Sari *Analisis Pengaruh Faktor Fundamental Dan Kondisi Ekonimi Terhadap Return Saham (Studi Kasus Pada Perusahaan Perbankan BUMN Yang Terdaftar Di Bursa Efek Indonesia Periode 2005-2012)*, Jurnal Ilmiah Mahasiswa FEB Universitas Brawijaya, vol. 1, no. 2, 2013.
31. H. Song *Forecasting Stock Market Indices Using the Recurrent Neural Network Based Hybrid Models: CNN-LSTM, GRU-CNN, and Ensemble Models*, Applied Sciences, vol. 13, no. 7, 4644, 2023.
32. P. H. Vuong, L. H. Phu, T. H. V. Nguyen, L. N. Duy, P. T. Bao, and T. D. Trinh *A bibliometric literature review of stock price forecasting: From statistical model to deep learning approach*, Science Progress, 107, 2024.
33. C. Wang, Y. Chen, S. Zhang, and Q. Zhang *Stock market index prediction using deep Transformer model*, Expert Systems with Applications, vol. 208, 118128, 2022.
34. Q. Wang *Working Paper: Improved Stock Price Forecasting Algorithm based on Feature-weighed Support Vector Regression by using Grey Correlation Degree*, arXiv: Computational Finance, 2019.
35. Y. Wang, G. Andreeva, and B. Martin-Baragan *Machine learning approaches to forecasting cryptocurrency volatility: Considering internal and external determinants*, International Review of Financial Analysis, vol. 90, 102914, 2023.
36. D. Wartanti and N. A. Masruroh, N. A. *Aplikasi Jaringan Saraf Tiruan Dan Particle Swarm Optimization Untuk Peramalan Indeks Harga Saham Bursa Efek Indonesia*, Jurnal Tekno Sains, vol. 6, no. 1, pp. 1–38, 2016.
37. R. B. Wiranata and A. Djunaidy *The Stock Exchange Prediction using Machine Learning Techniques*, A Comprehensive and Systematic Literature Review, vol. 14, no. 2, pp. 91–112, 2021.
38. X. S. Yang *Nature-Inspired Optimization Algorithms*, London: Elsevier Inc., 2014.
39. H. Yasin, A. Prahutama, and T. W. Utami *Prediksi Harga Saham Menggunakan Support Vector Regression Dengan Algoritma Grid Search*, Media Statistika, vol. 7, no. 1, pp. 29–35, 2014.
40. A. Zeng, M. Chen, L. Zhang, and Q. Xu *Are Transformers Effective for Time Series Forecasting?*, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 9, 2023.
41. J. Zheng, Y. Wang, S. Li, and H. Chen *The Stock Index Prediction Based on SVR Model with Bat Optimization Algorithm*, Algorithms, vol. 14, no. 10, 299, 2021.