Appl Intell (2016) 45:1148–1165
DOI 10.1007/s10489-016-0801-3

# Novel hybrid SVM-TLBO forecasting model incorporating dimensionality reduction techniques

**Shom Prasad Das[1] · N. Sangita Achary[1] · Sudarsan Padhy[2]**

Published online: 8 July 2016


**Abstract** In this paper, we present a highly accurate forecasting method that supports improved investment decisions. The proposed method extends the novel hybrid SVM-TLBO model consisting of a support vector machine (SVM) and a teaching-learning-based optimization (TLBO) method that determines the optimal SVM parameters, by combining it with dimensional reduction techniques (DR-SVM-TLBO). The dimension reduction techniques (feature extraction approach) extract critical, non-collinear, relevant, and de-noised information from the input variables (features), and reduce the time complexity. We investigated three different feature extraction techniques: principal component analysis, kernel principal component analysis, and independent component analysis. The feasibility and effectiveness of this proposed ensemble model were examined using a case study, predicting the daily closing prices of the COMDEX commodity futures index traded in the Multi Commodity Exchange of India Limited. In this study, we assessed the performance of the new ensemble model with the three feature extraction techniques, using different performance metrics and statistical measures. We compared our results with results from a standard SVM model and an SVM-TLBO hybrid model. Our experimental results show that the new ensemble model is viable and effective, and provides better predictions. This proposed model can provide technical support for better financial investment decisions and can be used as an alternative model for forecasting tasks that require more accurate predictions.

**Keywords** Support Vector Machine (SVM) · Teaching-Learning-Based Optimization (TLBO) · Dimensional reduction · Commodity futures contract · Financial time series

## 1 Introduction

In the business environment, we desire precise and efficient forecasts of various kinds of financial variables, which can then be used to develop successful strategies and avoid large losses [7]. Over the last three decades, many researchers have considered financial time-series data prediction, with the prime objective of beating the financial market. Financial forecasting is an interesting and challenging field, because there are a huge number of factors (e.g., economic, political, environmental, and psychological) that must be considered during the forecasting process. Financial time series data are intrinsically noisy, non-stationary, and deterministically chaotic [4, 39]. The noise in financial time-series data are caused by a lack of information regarding the historical behavior of financial markets, which makes it hard to map the future and past values. The non-stationary and chaotic nature of the data indicates that the data distribution varies over time and is unpredictable.

Because of successful developments in different computational intelligence techniques, researchers have started to apply computational intelligence approaches to financial

✉ Shom Prasad Das
shomdas@yahoo.com

[1] Department of Computer Science and Engineering, National Institute of Science and Technology, Palur Hills, Odisha, 761008, India

[2] Silicon Institute of Technology, Silicon Hills, Bhubaneswar, Odisha, 751024, India

markets. Example techniques include artificial neural networks (ANNs), support vector machines (SVMs), genetic algorithms (GAs), particle swarm optimization (PSO), and fuzzy technologies. Vapnik et al. [44] introduced SVM methods to overcome the problems of ANNs (such as getting trapped in local minima, overfitting to training data, and long training times). Since then, several authors have proposed financial instrument pricing using SVMs. For example, Tay and Cao [39] and Cao and Tay [4] developed pricing models for five specific financial futures in the US market using SVMs, and Van Gestel et al. [43] used LS-SVM (least squares support vector machine) for the T-bill (treasury bill) rate and stock index pricing in the United States (US) and German markets. Their experimental results showed that SVM performed well when applied to financial markets and produced good predictions Sapankevych and Sankar [38] published an exhaustive survey on SVMs for time series data prediction. They surveyed papers in the areas of financial market prediction, electric utility load forecasting, environmental states, weather prediction, and reliability forecasting. In their survey, they noted that the selection of free parameters for the SVM and the kernel function had a significant influence on the forecast. The experimental result by Kim [21] showed that SVM predictions are sensitive to these free parameters, and that it is important to select optimal values. Improper selection of free parameters can cause over- or under-fitting problems [21]. To select the optimal SVM parameter(s) and kernel function, several studies proposed a hybrid model combining an SVM and optimization techniques using PSO, GAs, artificial bee colonies (ABCs), differential evolution (DE), ant colony optimization (ACO), simulated annealing (SA), and so on [19, 29, 30, 46, 47]. However, the optimization model used to select the optimal parameter(s) itself introduces additional user-specified controlling parameter(s), making the user's task even more complex. To determine the influence of the user-specified controlling parameter(s) of the optimization technique on the forecasting result, Das and Padhy [8] proposed a novel hybrid SVM-TLBO model. In the hybrid model, the teaching-learning-based optimization (TLBO) algorithm proposed by Rao et al. [37] is used to optimize the SVM parameters and kernel function, which does not require user-specified control parameters. Their extensive experimental results showed that the SVM-TLBO novel hybrid model outperformed the standard SVM model and the SVM+PSO model proposed by Lin et al. [29].

Das and Padhy [8] proposed a novel hybrid regression model for forecasting the value of the multicommodity futures index (COMDEX) traded on the Multi Commodity Exchange of India Limited (MCX). They considered 17 technical indicators as input variables (features) to the SVM regression model, and predicted closing values of the futures index from 1, 3, and 5 days ahead. The 17 technical indicators in their study were selected from different literature surveys based on work by Hsu [12], Huang and Tsai [13], Ince and Trafalis [18], Kim [21], Kim and Han [22], Kim and Lee [23], Lai et al. [25], Liang et al. [27], and Tsang et al. [41], and feedback from domain experts. A detailed description of these technical indicators can be found in Appendix A. When modeling financial time series using SVM regression, technical indicators are used as input and must be very carefully selected and identified. Including irrelevant technical indicators as input to the regression model may lead to noise. Training an SVM model with the inherited noisy data may cause it to fit to undesirable data, resulting in an inappropriate approximation function and loss of the generalization. Moreover, the model could under- or over-fit to the noisy data [2]. Additionally, almost all stock prediction techniques use many approaches such as statistics, mathematical models, machine learning and artificial intelligence to determine potential future rules. However, these approaches require high quality features (inputs) before they can learn any knowledge, because inaccurate information can produce erroneous results [6].

To minimize the noise and collinearity in the data and to extract the most relevant information for knowledge discovery, we can use dimensionality reduction techniques like independent component analysis (ICA), principal component analysis (PCA), kernel principal component analysis (KPCA), and factor analysis (FA). The fundamental objectives of dimensionality reduction are to identify and extract more reliable information and effective features from the original data [40]. Our literature survey identified different dimensional reduction techniques used in machine learning and showed that the generalization capabilities of these methods are improving. Cao et al. [3] compared PCA, KPCA and ICA applied to SVM classification. Cai et al. [1] applied dimensionality reduction in SVM using PCA, KPCA and ICA. Ekenel and Sankur [10] applied ICA and PCA dimensionality reduction methods to facial recognition. Lu et al. [32] used ICA for dimensionality reduction with support vector regression (SVR) for financial timeseries data forecasting. Ince and Trafalis [17] used KPCA and factor analysis (FA) as dimensionality reduction techniques with SVM to predict stock prices. Lu [31] developed a hybrid model using non-linear independent component analysis (NLICA), SVR and PSO to forecast the stock index. Zhai et al. [48] used PCA with SVM to improve material identification of loose particles in sealed electronic devices Their experimental results showed that this model was effective. An improved SVM model was proposed by Kuang et al. [24], which integrated SVM, KPCA, and chaotic PSO for intrusion detection.The focus of this research was to incorporate some well-known

dimensionality reduction techniques (feature extraction approaches) into the novel hybrid SVM-TLBO regression model proposed by Das and Padhy [8], to develop an improved prediction model (DR-SVM-TLBO) that forecasts COMDEX. Chang and Wu [6] concluded from their experimental study that dimensional reduction using feature extraction techniques is superior to feature selection techniques. We considered PCA, a non-linear version of PCA with a kernel based trick (KPCA), and ICA to reduce the dimensionality of the input variables (features), because these three techniques are well-known methods [1, 3, 17, 33] More detailed information regarding the PCA, KPCA, and ICA algorithms can be found in [6, 14–17, 20, 31, 32, 42] and Appendix B. We compared standard SVM without dimension reduction and the novel hybrid SVM-TLBO [8] model with our proposed model. The expected benefits of the proposed model are that it can: (1) reduce the dimension of the financial time-series data, (2) reject noisy inputs, (3) reduce the computational complexity, and (4) produce a more generalized model.

The experimental results of our commodity futures index data show that the forecasting results of the proposed ensemble model are more accurate than those of the standard SVM regression model and the novel hybrid SVM-TLBO model [8]. To measure and compare the forecasting performances of the models, we used the root mean square error (RMSE), mean absolute error (MAE), normalized mean square error (NMSE), directional symmetric (DS), and Diebold-Mariano (DM) statistical test. The outcome of this study provides empirical confirmation of the usefulness of the proposed model when forecasting in the financial domain.

The contribution of this study and experiment was to design a new ensemble system by integrating two approaches: dimensionality reduction techniques using feature extraction methods and the novel hybrid SVM-TLBO regression model. A small improvement in the forecasting performance can minimize the stakeholders' risk and lead to a considerable investment profit. The proposed model produced a system that revealed three key characteristics: (1) the resulting model reduced the input features (technical indicators) from seventeen to six features that had a greater than 95 % cumulative variance; (2) the time complexity of the proposed model was less than the benchmark models; and (3) the forecasts were more accurate than those of the benchmark models. The organization of this paper is as follows. Section 2 provides a summary of SVM for regression, TLBO optimization methods, and the novel hybrid SVM-TLBO model. In Section 3, we describe the proposed hybrid model architecture and experiments, followed by the empirical results and discussion in Section 4. Section 5 concludes the study with a brief discussion of our findings and future work.

## 2 Novel hybrid SVM-TLBO model

### 2.1 SVM for regression

Vapnik et al. [44] developed an SVM technique for regression. The method was presented in Hykins [11] as follows.

Given a training data set $\{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$, where each $x_i \in X \subset R^n$ ($X$ denotes the input sample space) and matching target values $y_i \in R$ for $i = 1, \ldots, l$ (where $l$ corresponds to the size of the training data), the objective of the regression problem is to find a function $f : R^n \to R$ that can approximate the value of $y$ for $x$ not in the training set.

The estimating function $f$ is defined as

$$f(x) = (w^T \Phi(x)) + b, \tag{1}$$

where $w \in R^m b \in R$ is the bias, and $\Phi$ denotes a nonlinear function from $R^n$ to high-dimensional space $R^m (m > n)$. The aim is to find $w$ and $b$ such that the value of $f(x)$ can be determined by minimizing the risk.

$$R_{\text{reg}}(f) = C \sum_{i=1}^{n} L_\epsilon(y_i, f(x_i)) + \frac{1}{2} \|w\|^2 . \tag{2}$$

Here, $L_\epsilon$ is the extension of the $\epsilon$-insensitive loss function originally proposed by Vapnik et al. [44], which is defined as

$$\mathrm{L}_\epsilon = \left\{ \begin{array}{ll} |\mathrm{y} - \mathrm{z}| - \epsilon, & |\mathrm{y} - \mathrm{z}| \geq \epsilon \\ 0, & \text{otherwise} \end{array} \right\}. \tag{3}$$

By introducing the slack variables $\zeta_i$ and $\zeta_i^*$, the problem in (2) can be reformulated to the following.

($\boldsymbol{P}$) Minimize $C[\sum_{i=1}^{l} (\zeta_i + \zeta_i')] + \frac{1}{2}\|w\|^2$ subject to

$$\begin{array}{l} (i) y_i - w^T \Phi(x_i) - b \leq \epsilon + \zeta_i, \\ (ii) w^T \Phi(x_i) + b - y_i \leq \epsilon + \zeta_i', \\ (iii) \zeta_i \geq 0, \\ (iv) \zeta_i' \geq 0, \end{array} \tag{4}$$

where $i = 1, \ldots, l$, and $C$ is a user-specified constant known as a regularization parameter.

We can solve ($\boldsymbol{P}$) using the primal dual method to get the following dual problem.

Determine $(\{\alpha_i\}_{i=1}^{l} and \left\{\alpha_i^*\right\}_{i=1}^{l})$ ($\alpha_i$ and $\alpha_i*$ are the respective Lagrange multipliers for constraints (i) and (ii) of the primal quadratic optimization problem $\boldsymbol{P}$ in (4)), that maximize the objective function

$$\begin{aligned} Q(\alpha_i, \alpha_i^*) = & \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) - \epsilon \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) \\ & - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) K(x_i, x_j), \end{aligned} \tag{5}$$

subject to

(1)

$$\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, \text{ and} \tag{6}$$

(2)

$$0 \le \alpha_i \le C, 0 \le \alpha_i^* \le C. \tag{7}$$

Here, $i = 1, \ldots, l$, and $K : X \times X \rightarrow R$is the Mercer kernel defined by

$$K(x, z) = \Phi(x)^T \Phi(z). \tag{8}$$

The solution of the primal dual method yields

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\Phi(x_i), \tag{9}$$

where $b$ is calculated using the Karush-Kuhn-Tucker conditions. That is,

$$\begin{aligned} &\alpha_i(\varepsilon + \zeta_i - y_i + w^T\Phi(x_i) + b) = 0, \\ &\alpha_i^*(\varepsilon + \zeta_i^* + y_i - w^T\Phi(x_i) - b) = 0, \end{aligned} \tag{10}$$

$$(C - \alpha_i)\zeta_i = 0 \text{ and } (C - \alpha_i^*)\zeta_i^* = 0\,, \text{ where } i = 1, \ldots, l \tag{11}$$

Because $\alpha_i \bullet \alpha_i^* = 0$, and $\alpha_i$ and $\alpha_i^*$ cannot simultaneously be non-zero, there exists some $i$ for which either $\alpha_i \in (0, C)$ or $\alpha_i^* \in (0, C)$. Hence, $b$ can be computed using

$$\begin{aligned} b &= y_i - \sum_{j=1}^{l}(\alpha_j - \alpha_j^*)K(x_j, x_i) - \varepsilon \quad for\ 0 < \alpha_i < C, \\ b &= y_i - \sum_{j=1}^{l}(\alpha_j - \alpha_j^*)K(x_j, x_i) + \varepsilon \quad for\ 0 < \alpha_i^* < C. \end{aligned} \tag{12}$$

The $x_i$ corresponding to $0 < \alpha_i < C$ and $0 < \alpha_i^* < C$ are called *support vectors*. Using the expressions for $w$ and $b$ in (9) and (12), $f(x)$ can be computed using

$$\begin{aligned} f(x) &= \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)(\Phi(x_i)^T\Phi(x)) + b, \\ &= \sum_{i=1}^{\ell}(\alpha_i - \alpha_i^*)K(x_i, x) + b. \end{aligned} \tag{13}$$

Note that we do not require the function $\Phi$ to compute $f(x)$, which is an advantage of using the kernel.

### 2.2 Teaching-learning-based optimization technique

Teaching learning based optimization (TLBO) is a recently established novel and effective meta-heuristic population based optimization algorithm [37]. TLBO uses a certain population of solutions to find the global solution, in a similar way as other nature-inspired algorithms such as PSO, GA, and ABC. The TLBO algorithm is based on a simulation of a traditional learning process, that is, the transfer of knowledge within a classroom atmosphere. The algorithm consists of two stages: (i) learners (students) first acquire knowledge from a teacher (*teacher phase*); and (ii) they enhance their knowledge by interacting with their peers (*student phase*). The TLBO population consists of a group of learners. There are decision variables, similar to other optimization algorithms. The different decision variables in TLBO are equivalent to the different subjects offered to students, and the students' results are analogous to the 'fitness' value of the optimization problem.

#### *2.2.1 Steps in the TLBO algorithm*

The following steps of the TLBO algorithm were described by Rao et al. [37].

Step 1: Define the optimization problem and create a solution space: In the initial phase, we identify the decision variable(s) in the problem to be optimized and assign them a range (minimum and maximum of the variable) where we will search for the optimal solution. If the solution spaces and ranges are not properly defined, then there is a chance that the optimization will take more time.

Step 2: Identify the fitness function: In this step, we design or identify the fitness function, which accurately represents how well the optimized solution fits our problem using a single number. The TLBO algorithm uses the fitness function to evaluate its candidate solutions and obtains the optimal solution by minimizing (or maximizing) $f(X)$ over the range of values of the decision variables $(X)$, where $f(X)$ is the fitness function.

Step 3: Initialize the learners (or students): Each learner (based on the population size) is initialized using random values for each of the decision variables (within the appropriate ranges). The $i$-th learner is represented by row vector$X_i$, defined as

$$X_i = [x_{i,1}, x_{i,2}, x_{i,3}, \ldots, x_{i,D}], i = 1, 2, \ldots, N, \tag{14}$$

where$D$ is the number of decision variables, and N is the total number of learners. Each decision variable $x_{i,j}$is randomly assigned a value using

$$x_{i,j} = x_j^{\min} + rand() * (x_j^{\max} - x_j^{\min}) j = 1, 2, \ldots, D \tag{15}$$

where $x_j^{\min}$ and $x_j^{\max}$ are the minimum and maximum values of the $j$-th variable of $i$-th learner, and $rand()$is a function that returns a random number between 0 and 1.

Step 4: Teacher phase

a. Compute the mean value of each of the learners' decision variables and denote the population mean as $X_{mean} = [\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_j, \ldots, \overline{x}_D]$, where $\overline{x}_j = \frac{\sum_{i=1}^{N} x_{i,j}}{N}$
b. Compute the fitness values of each learner $X$ based on the fitness function $f(X)$.The learner with the best fitness value (solution) is identified as the teacher ($X_{teacher}$) for the teacher phase.
c. Now the teacher ($X_{teacher}$) transfers their knowledge and tries to improve the fitness of other learners ($X_i$) by shifting the mean of the learners towards the teacher using

$$X_{new} = X_i + rand() * (X_{teacher} - (TF) * X_{mean}), \quad for\ i = 1, 2, \ldots, N, \tag{16}$$

where,

$$TF = round[1 + rand(0, 1)]. \tag{17}$$

Here, $TF$ is the teaching factor (either 1 or 2), and $rand()$ is a random number function that returns a number between 0 and 1.

Note that $TF$is not a parameter of the TLBO algorithm. The value of $TF$is not provided as input to the TLBO, but its value is randomly chosen with equal probability by the algorithm using (17).

d. If the updated solution ($X_{new}$) is better than the existing solution ($X_i$), then we accept the new solution, otherwise we reject it.

Step 5: Student phase

In the student phase, the learners (students) enhance their knowledge by interaction with other peer learners in the classroom. The practice of mutual interaction between learners (students) tends to increase the knowledge of the learner. Therefore, an individual learner learns if the other individuals have more knowledge.

a. Randomly select any two solutions $X_i$ and $X_j$ such that $i \neq j$.
b. Solution $X_i$interacts with solution $X_j$ If $f(X_i)$,that is, the fitness value of $X_i$ is better (superior) than fitness value of $X_j$, then we update $X_i$to $X_{new}$ using

$$X_{new} = X_i + rand() * (X_i - X_j) \tag{18}$$

otherwise, we update it to

$$X_{new} = X_i + rand() * (X_j - X_i) \tag{19}$$

Step 6: Iterate until the termination criteria are satisfied

We then repeat Steps 4 and 5 until our termination conditions are satisfied, i.e., the average value of the fitness function for all learners does not improve, or we reach the maximum number of generations. The $X_i$that minimizes $f(X_i)$for a minimization problem (or $X_i$that maximizes $f(X_i)$for a maximization problem) is the final solution to the optimization problem.

A three-dimensional graphical illustration of single learner $X_i$ searching for the optimal solutions is presented in Fig. 1. The initial stage represents the status of the decision variables obtained by the learner for each of the parameters in the optimization problem, as in (15) $X_{mean}$ and $X_{teacher}$ represent the mean and current best status among all the learners (populations). The updated $X_{new}$ is the status of the learner after the teacher phase, which is updated based on (16). $X_{new}$ after the student phase represents the learner status after interacting with its peers in the student phase, where the status is updated using either (18) or (19) Note that the fitness value (i.e., distance between the $X$'s and corresponding $f(X$'s)) of each learner improves after each phase.

### 2.3 The novel hybrid SVM-TLBO regression model

The novel hybrid SVM-TLBO model proposed by Das and Padhy [8] predicts using SVM regression and uses TLBO to determine the SVM parameters. The hybrid model was designed to work in a two-dimensional solution space, that is, to optimize $C$ and $\sigma$, where $C$ is the regularization parameter of the SVM regression model and $\sigma$ is the bandwidth of the radial basis (Gaussian) kernel function. The values of different parameters used in the SVM-TLBO novel hybrid model are presented in Table 1(a) and (b). The flow chart of the SVM-TLBO hybrid regression model is shown in Fig. 2. The raw time series financial data are processed to prepare the input set (features), and then the TLBO algorithm selects the optimal free parameters for the

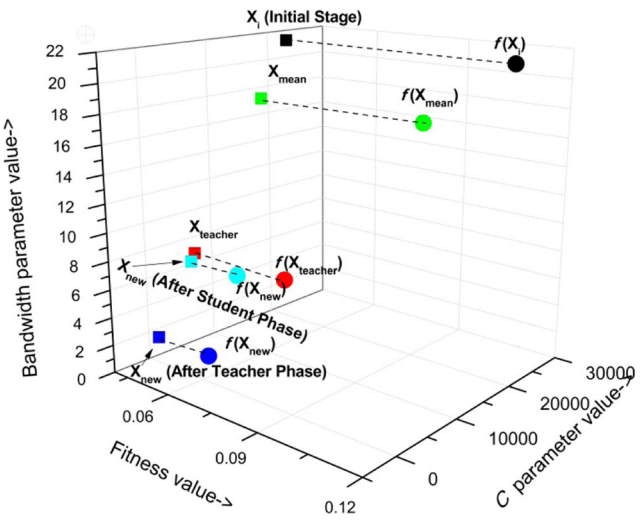



**Fig. 1** Three-dimensional graphical illustration of the learner (population) searching for optimal solutions, in the teacher and student phases

**Table 1** (a) SVM and (b) TLBO parameters used in experiments [8]

| SVM Parameter | Parameter Value |
|---|---|
| $C$ (regularization parameter) | 0.01 to 35,000 [28] |
| $\sigma$ (bandwidth) | 0.0001 to 32 [28] |
| $\varepsilon$ | 0.0001 [28] |
| Kernel type | Radial basis (Gaussian) |
| (a) SVM parameters | |
| TLBO Parameter | Parameter Value |
| Population size (learners/students) | 15 [34, 36] |
| Maximum iterations | 30 [34, 36] |
| Optimization category | Minimum |
| (b) TLBO parameters | |

SVM regression model. We evaluate the fitness function for the optimization algorithm using the RMSE of the SVM regression results. In the training phase we apply the SVM regression model for each set of parameter ($C$ and $\sigma$) values obtained by the TLBO algorithm. These multiple executions of the SVM regression model in the training phase increase the computational time, but this is the only overhead involved in the hybrid model. After determining the optimal parameters for the training data set, we apply the trained model to the test (out-of-sample) data to evaluate the performance of the forecasting model.

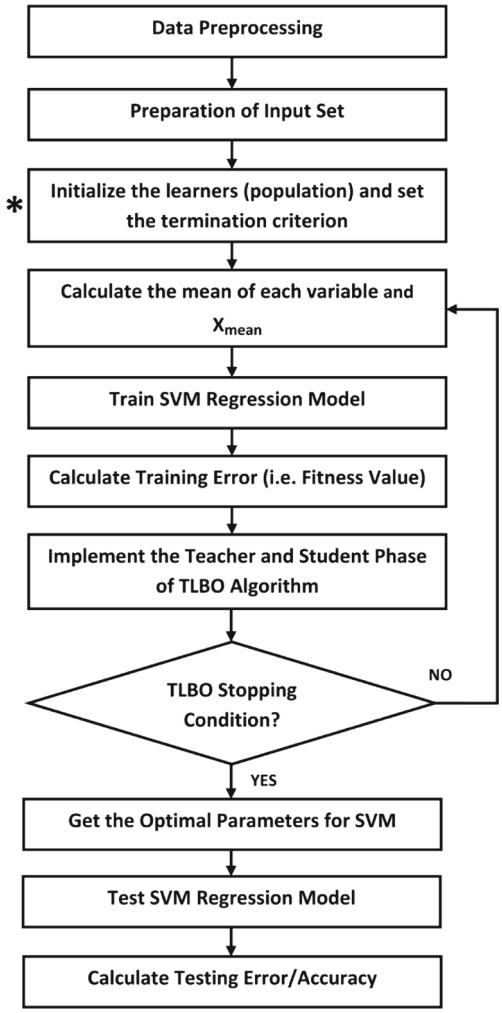



**Fig. 2** Flow chart representation of the novel hybrid SVM-TLBO regression model [8]

## 3 Proposed ensemble model architecture and methodology

We propose a new ensemble model called DR-SVM-TLBO for predicting financial time series, particularly the values of the energy commodity futures index. The proposed model is presented as a flowchart in Fig. 3. In the first step, the raw original time-series data collected from the

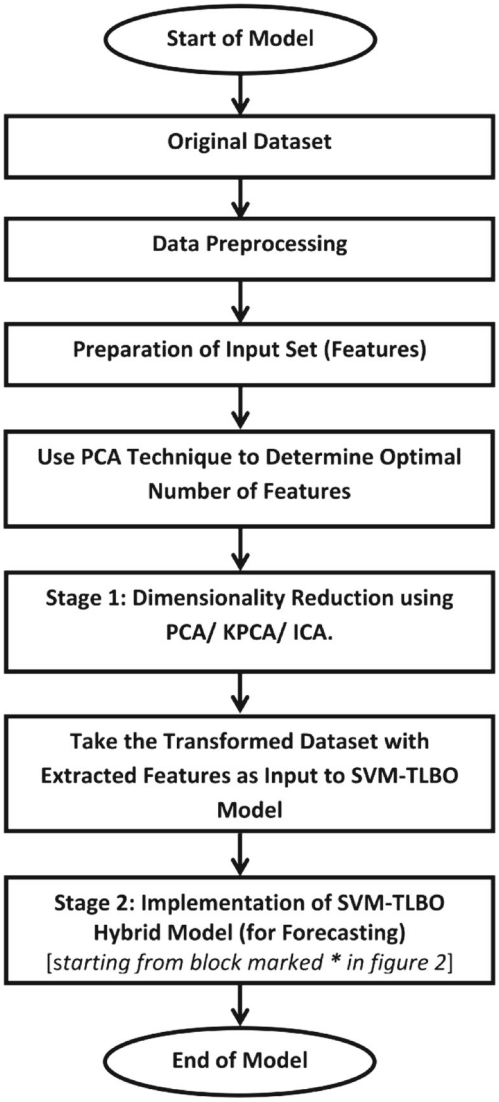



**Fig. 3** Flowchart for the proposed DR-SVM-TLBO ensemble model

market (i.e., MCX COMDEX) is input into the model for preprocessing and is used to calculate the technical indicators (see Appendix A). Detailed explanations of the data collection methodology, data, and preprocessing are given in Section 3.1. To determine the optimal number of features (to reduce dimensionality) from the 17 normalized input technical indicators, we used PCA dimension reduction to explain at least 95 % of the cumulative variance in our data set. Then the flow chart is divided into two stages: (1) dimensional reduction (critical feature extraction), and (2) implementation of SVM-TLBO hybrid model. In the dimensionality reduction stage, we apply feature extraction techniques using PCA, KPCA, and ICA to the normalized data. The number of critical features to be extracted is equal to the optimal number of input features ($N$) determined according to PCA to account for 95 % of the cumulative variance. After the dimension reduction step, we construct an input dataset containing the extracted features (reduced in size) In the SVM-TLBO stage, we apply the model to the reduced features. Here the training dataset is used to find the optimal values for the free parameters of the SVM regression model and the kernel function. The SVM-TLBO hybrid model used in the second stage is similar to the model developed by Das and Padhy [8] and presented in Fig. 2 The only difference is that we have omitted the first two blocks (i.e., data preprocessing and input preparation). These changes were required because data preprocessing and input preparation steps are already included in the proposed model. A detailed overview of the computation techniques used in our study is presented in Section 3.3 To forecast the values of the commodity futures index for a new data pattern, $X$, we must first apply the dimension reduction technique to extract the optimal feature values. Then, the trained SVM regression model is used to predict the value for the new data pattern. The out-of-sample (test) data are historical data, so the desired index values are known, and we can easily calculate the forecast performance.

### 3.1 Experimental data

To examine the effectiveness of the improved forecasting model, we applied it to real COMDEX data collected from the MCX (http://www.mcxindia.com) [8]. We collected daily trading series data points from January 1, 2010, to May 7, 2014, and used them as training and testing data. The total number of data samples in the time frame was 1,332. The time-series data consist of daily opening price, low price, high price, closing price, and traded date. We used 17 technical indicators as the inputs. The raw daily prices were used to calculate the technical indicators as per the details given in Appendix A. The data period includes many important and significant economic events so we consider this data to be appropriate for training the proposed models. Table 2 describes the dataset in terms of high, low, mean, and median values, as well as standard deviation, kurtosis (measure of the flatness of the distribution), and skewness (degree of asymmetry of the distribution close to its mean). Table 2 shows that the skewness value of the dataset is less than zero i.e., the dataset is left skewed (most values are concentrated to the right of the mean, with the extreme values to the left), there are lot of spikes in the dataset, and the kurtosis value is less than three i.e., it is a platykurtic distribution (flatter than a normal distribution with a wider peak). After processing the 1,332 raw data points, we obtained 1,307 transformed data points with dates from February 1, 2010 to May 7, 2014. The technical indicators were normalized to the range [0, 1] to minimize forecasting errors and to prevent variables with large numeric ranges from overwhelming the other data. The min-max normalization process was applied to the input technical indicators and the output closing prices. The technical indicators and closing prices were normalized using

$$\bar{x}_i^d = \frac{x_i^d - \min\left(x_i^d|_{i=1}^N\right)}{\max\left(x_i^d|_{i=1}^N\right) - \min\left(x_i^d|_{i=1}^N\right)}, \quad d = 1, 2, \ldots, 17 (number\ of\ input \text{ variable}) \tag{20}$$

where $\bar{x}_i^d$ is the normalized value, $x_i^d$is the original value, $\min(x_i^d|_{i=1}^N)$ is the minimum value in the original input data, $\max(x_i^d|_{i=1}^N)$ is the maximum value in the original input data, and $N$ is the total number of trading days.

The normalized data were segregated into training and test groups, approximately in the ratio of 5:1. The data were divided into training and testing samples based on previous work [7, 8, 21, 31, 45]. The ratio of training to test data used by Chen et al. [7] was 9:1, Das and Padhy [8] used approximately 5:1, Kim [21] and Lu [31] used 4:1, and Wang and Wang [45] used approximately 6:1. In our case, 1085 data points were used for training with 5-fold cross-validation, and the remaining 222 were used to test the model. We considered three different forecasts of the closing prices: (1) 1 day ahead; (2) 3 days ahead; and (3) 5 days ahead.

**Table 2** Brief description of the MCX COMDEX dataset

| Parameter (for entire collection period) | High | Low | Mean | Median | Standard deviation | Kurtosis | Skewness |
|---|---|---|---|---|---|---|---|
| Value | 4689.6 | 2504.53 | 3539.56 | 3692.17 | 473.53 | −0.4959 | −0.7378 |

### 3.2 Performance measures and statistical test

The performance of the proposed model was evaluated using standard parametric statistical metrics: RMSE, MAE, NMSE, and DS [4, 7, 39]. The descriptions and definitions of these performance criteria are given in Table 3. The accuracy of the direction of the prediction is provided by DS (in %). Larger DS values indicate a better forecast. These parametric statistical tests require a distributional assumption (i.e., the data are normally distributed) and are not robust to outliers, so they may occasionally produce ambiguous results. Therefore, we also used nonparametric techniques to evaluate the significance of any differences in the test (out-of-sample) performance of the proposed model compared with the benchmark models. We applied the DM statistical test [9], which is a nonparametric statistical test extensively used for forecasting model validation, especially in economics and finance. In the DM test, the null hypothesis states that the two forecasting methods have the same forecasting accuracy, while the alternative hypothesis is that the two forecasting methods have different levels of accuracy. The null hypothesis of equal forecasting accuracy is rejected at the 5 % significance level; that is, if the computed absolute value of the DM statistic is greater than 1.96 (i.e., |DM value| $>$ 1.96). In this study, we used the square-error criteria as the loss function in the DM test.

### 3.3 Computation techniques

We implemented Vapnik's SVM regression technique using LIBSVM, which is a SVM tool box [5]. We used the Gaussian kernel (radial basis) function, because it performs well under general smoothness assumptions. All the experiments were executed on an Intel Core i7 CPU @ 2.10 GHz, with 6 GB primary memory. We wrote our own code to implement TLBO for the SVM-TLBO hybrid regression model. The TLBO algorithm was defined in two dimensions, to optimize $\sigma$ (bandwidth) of the Gaussian kernel parameter and $C$ (regularization parameter) of the SVMs. In our experimental runs of the TLBO algorithm, there were no significant changes to $\sigma$ and $C$ after 25–30 iterations, when using a population size (learners/students) of 15. Pawar and Rao [34] and Rao and Patel [36], observed that TLBO only requires a small population and few iterations (generations). With this in mind, we fixed the maximum number of iterations for the TLBO to 30, with a population size 15. According to Tay and Cao [39], SVRs are insensitive to $\varepsilon$ (the extent to which deviations are tolerated) if it is a reasonable value. Cao and Tay [4] observed that the number of support vectors decreases as $\varepsilon$ increases. Thus, we chose $\varepsilon = 0.0001$. The SVM parameter range of $C$ was set to 0.01–35,000, and the range of $\sigma$ (bandwidth parameter of Gaussian kernel) was set to 0.0001–32 [28].

For the proposed ensemble DR-SVM-TLBO model, we designed our own code for dimensionality reduction using the PCA, KPCA, and ICA techniques which we implemented in R (https://www.r-project.org). The ensemble model was implemented according to the flowchart in Fig. 3. As previously discussed, we determined the optimal number of features from the original 17 technical indicators (features) using PCA. We selected the number of dimensions based on the PCA results that accounted for at least 95 % of the cumulative variance in the dataset. The cumulative variance of the PCA result is presented in Fig. 4. The optimal number of features is six, because the cumulative variance of the first six principal components (PC1 to PC6) was 95.41 %. So we set the optimal number of features (components) for all the dimension reduction techniques to six. In the implementation of the KPCA technique,

**Table 3** Performance evaluation metrics and their definitions

| Performance Metrics | Calculation |
|---|---|
| RMSE | $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - y_i)^2}$ |
| NMSE | $\frac{1}{\sigma^2 n}\sum_{i=1}^{n}(y_i - p_i)^2$ where: $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y})^2 and \overline{y} = \sum_{i=1}^{n} y_i$ |
| MAE | $\frac{1}{n}\sum_{i=1}^{n}\lvert y_i - p_i\rvert$ |
| DS | $\frac{100}{n}\sum_{i=1}^{n} d_i$ $d_i = \{^{1 if (y_i - y_{i-1})(p_i - p_{i-1}) \geq 0}_{0 otherwise}$ |

* Note that $n$ is the total number of data, $y_i$ is the actual output value, and $p_i$ is the predicted output value of the i-th sample data

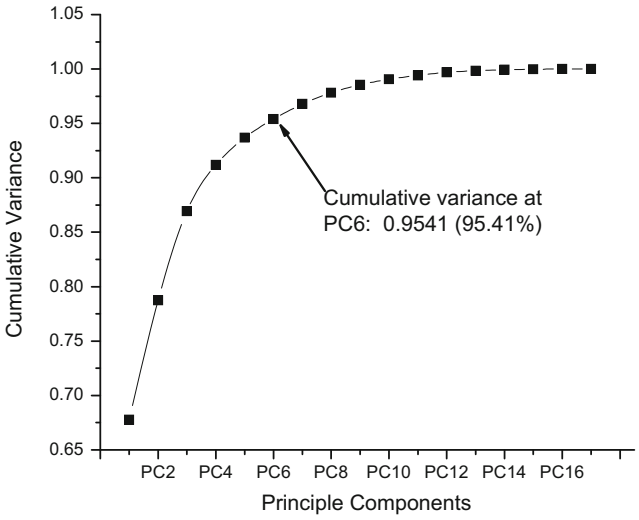



**Fig. 4** Cumulative variance for the PCA technique

we used a Gaussian kernel (radial basis) with a bandwidth parameter of 0.01 and for ICA, we used the fast ICA algorithm [15].

The dimensional reduction in the proposed ensemble model uses PCA, KPCA, and ICA methods. So there are three different variants of the proposed model: 1) $DR_{PCA}$-SVM-TLBO, the proposed model with PCA for dimensionality reduction; 2) $DR_{KPCA}$-SVM-TLBO, the proposed model with KPCA for dimensionality reduction; and 3) $DR_{ICA}$-SVM-TLBO the proposed model with ICA for dimensionality reduction. We ran the new ensemble DR-SVM-TLBO algorithm as per the flow chart in Fig. 3. The simulation results are shown in Table 4. We compared the results of all three variants (i.e., $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO), the new ensemble model with the standard SVM regression without dimension reduction, and the novel hybrid SVM-TLBO model [8]. We used a sequential optimization based algorithm to train the SVM regression, because it is fast and efficient for large data sets.

**Table 4** Model performance with respect to the RMSE and the optimal parameters for standard SVM, SVM-TLBO novel hybrid, $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO

| Forecasting Cases | **Model** | Optimal Value | | Performance | |
|---|---|---|---|---|---|
| | | $C$ | $\sigma$ | RMSE | Average Computational Time (milliseconds) |
| 1-day-ahead forecast | SVM* | 100 | 0.01 | TR: 0.0601 | TR: 867 |
| | | | | TE: 0.0992 | TE: 41 |
| | SVM-TLBO* | 2376.1150 | 2.1725 | TR: 0.0363 | TR: 8173 |
| | | | | TE: 0.0440 | TE: 45 |
| | $DR_{PCA}$-SVM-TLBO# | 1.5222 | 2.0021 | TR: 0.0151 | TR:496 |
| | | | | TE: 0.0435 | TE: 27 |
| | $DR_{KPCA}$-SVM-TLBO# | 37.3934 | 0.7444 | TR:0.0144 | TR:385 |
| | | | | TE: 0.0219 | TE: 29 |
| | $DR_{ICA}$-SVM-TLBO# | 35.3597 | 0.6499 | TR: 0.0150 | TR: 279 |
| | | | | TE: 0.0305 | TE: 3 |
| 3-days-ahead forecast | SVM* | 10000 | 0.0001 | TR: 0.0614 | TR:2674 |
| | | | | TE: 0.0922 | TE: 41 |
| | SVM-TLBO* | 25.5730 | 0.0234 | TR: 0.0333 | TR: 3483 |
| | | | | TE: 0.0408 | TE: 45 |
| | $DR_{PCA}$-SVM-TLBO# | 2.0044 | 4.5064 | TR: 0.0240 | TR:823 |
| | | | | TE: 0.0354 | TE: 26 |
| | $DR_{KPCA}$-SVM-TLBO# | 11.0621 | 2.8712 | TR: 0.0241 | TR:2545 |
| | | | | TE:0.0292 | TE: 29 |
| | $DR_{ICA}$-SVM-TLBO# | 79.9654 | 1.2028 | TR: 0.0250 | TR:1922 |
| | | | | TE: 0.0326 | TE: 31 |
| 5-days-ahead forecast | SVM* | 0.1 | 1 | TR: 0.0613 | TR: 4292 |
| | | | | TE: 0.1403 | TE: 41 |
| | SVM-TLBO* | 198.8539 | 0.1818 | TR: 0.0434 | TR: 9927 |
| | | | | TE: 0.0599 | TE: 39 |
| | $DR_{PCA}$-SVM-TLBO# | 6.5362 | 4.7105 | TR: 0.0282 | TR:2263 |
| | | | | TE: 0.0509 | TE: 31 |
| | $DR_{KPCA}$-SVM-TLBO# | 21.6708 | 2.5298 | TR: 0.0289 | TR:3775 |
| | | | | TE:0.0479 | TE: 28 |
| | $DR_{ICA}$-SVM-TLBO# | 1.3585 | 7.1256 | TR:0.0275 | TR:817 |
| | | | | TE: 0.0537 | TE: 29 |

TR represents the training phase and TE represents the testing phase

Bold values represent the best performance. * Result based on Das and Padhy [8]. # Proposed model

**Table 5** Comparison of the out-of-sample results with respect to the RMSE, MAE, NMSE, and DS of the standard SVM, SVM-TLBO novel hybrid, $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO ensemble models

| Forecasting Cases | Model | RMSE | MAE | NMSE | DS |
|---|---|---|---|---|---|
| 1-day-ahead forecast | SVM* | 0.0992 | 0.0967 | 4.32E -05 | 58.79 |
| | SVM-TLBO* | 0.044 | 0.033 | 8.49E -06 | 56.02 |
| | $DR_{PCA}$-SVM-TLBO# | 0.0435 | 0.0225 | 8.30E-06 | 55.09 |
| | $DR_{KPCA}$-SVM-TLBO# | 0.0219 | 0.0147 | 2.10E -06 | 52.78 |
| | $DR_{ICA}$-SVM-TLBO# | 0.0305 | 0.0183 | 4.06E -06 | 57.87 |
| 3-days-ahead forecast | SVM* | 0.0922 | 0.0754 | 2.98E -05 | 51.54 |
| | SVM-TLBO* | 0.0408 | 0.0333 | 7.37E -06 | 52.85 |
| | $DR_{PCA}$-SVM-TLBO# | 0.0354 | 0.0279 | 5.56E -06 | 51.39 |
| | $DR_{KPCA}$-SVM-TLBO# | 0.0292 | 0.0231 | 3.79E -06 | 52.77 |
| | $DR_{ICA}$-SVM-TLBO# | 0.0326 | 0.0252 | 4.64E -06 | 53.24 |
| 5-days-ahead forecast | SVM* | 0.1403 | 0.1341 | 1.13E -04 | 50.93 |
| | SVM-TLBO* | 0.0599 | 0.0442 | 1.58E -05 | 52.53 |
| | $DR_{PCA}$-SVM-TLBO# | 0.0509 | 0.038 | 4.20E -05 | 50.25 |
| | $DR_{KPCA}$-SVM-TLBO# | 0.0479 | 0.0363 | 1.00E-05 | 50.46 |
| | $DR_{ICA}$-SVM-TLBO# | 0.0537 | 0.0406 | 6.90E -05 | 50 |

*Bold values represent the best performance.* * *Result based on Das and Padhy* [8]. # *Proposed model*

**Table 6** Diebold-Mariano statistic: DM test values and p-values (in parentheses) for the MSE loss function

| Model/ Forecasting cases | 1-day-ahead forecast | 3-days-ahead forecast | 5-days-ahead forecast |
|---|---|---|---|
| Standard SVM | 31.4622* | 20.377* | 19.4791* |
| | (<2.2E -16) | (<2.2E -16) | (<2.2E -16) |
| SVM-TLBO | 6.9828* | -0.4223 | 2.5113* |
| | (3.517E -11) | (0.6733) | (0.0132) |
| $DR_{PCA}$-SVM-TLBO | 2.5026* | 3.018* | 2.9088* |
| | (0.013) | (0.0029) | (0.004) |
| $DR_{ICA}$-SVM-TLBO | 2.5593* | 0.286 | 3.5703* |
| | (0.0111) | (0.7752) | (0.0004) |

* *Significance level of 5%, |DM test |>1.96*

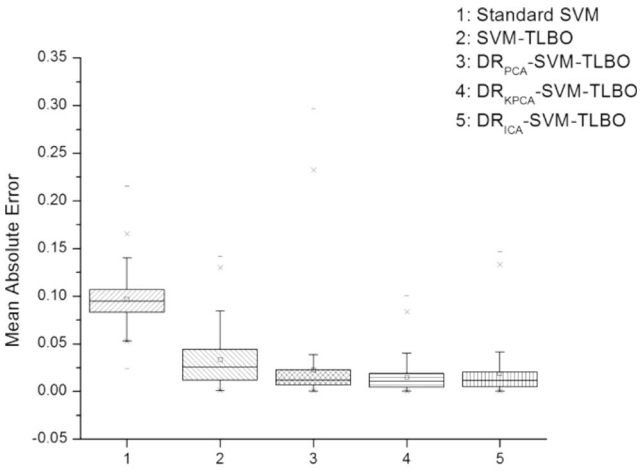
**Table 7** Percentage (%) improvement of the proposed new ensemble DR-SVM-TLBO (all three variants) model over the benchmark novel hybrid SVM-TLBO [8] model for out-of-sample data, with respect to RMSE and MAE

| Performance Metrics | Forecasting Cases/ Models | $DR_{PCA}$-SVM-TLBO | $DR_{KPCA}$-SVM-TLBO | $DR_{ICA}$-SVM-TLBO |
|---|---|---|---|---|
| RMSE | 1-day-ahead | 1.14 % | 50.23 % | 30.68 % |
| | 3-days-ahead | 13.24% | 28.43% | 20.10 % |
| | 5-days-ahead | 15.03 % | 20.03 % | 10.35 % |
| MAE | 1-day-ahead | 31.82 % | 55.45 % | 44.55 % |
| | 3-days-ahead | 16.22 % | 30.63 % | 24.32 % |
| | 5-days-ahead | 14.03 % | 17.87 % | 8.14 % |

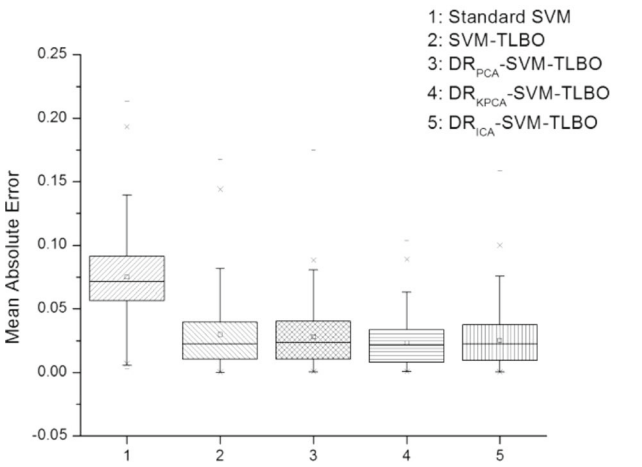
## 4 Experimental results and discussion

In this section, we present our experimental results regarding the efficiency of the proposed new ensemble model. The RMSE results and average computational time in milliseconds for all five models in the training phase (in-sample) and testing phase (out-of-sample), and the optimal values of $C$ and $\sigma$ are presented in Table 4. The testing phase (out-of-sample) RMSE, MAE, and NMSE values presented in Table 5 show that the new ensemble DRSVM-TLBO model (all three variants) outperformed the standard SVM regression and SVM-TLBO novel hybrid models in all three forecasting cases. This is because the parameters ($C$ and $\sigma$) of the standard SVM were selected using a traditional grid search method whereas the optimal values of $C$ and $\sigma$ for SVM-TLBO, $DR_{PCA}$-SVM-TLBO $DR_{KPCA}$-SVM-TLBO and $DR_{ICA}$-SVM-TLBO were obtained using the TLBO algorithm starting at random values within the defined solution space. In addition to the selection of the optimal SVM and kernel parameters, the dimension reduction techniques (i.e., PCA, KPCA, and ICA) extracted the input features from the original input set (17 technical indicators). The extracted input features contain less noise and more refined information. This changes the optimal values (of $C$ and $\sigma$) derived by the optimization process and produced superior forecasting models. Table 4 clearly shows that the dimension reduction techniques used in our models reduced the average computational time and increased the accuracy when compared to the benchmark models. With respect to the DS performance metric, standard SVM performed better than rest of the models in the 1-day-ahead forecasts, $DR_{ICA}$-SVM-TLBO performed best in the 3-days-ahead forecasts, and SVM-TLBO performed best for the 5-days-ahead forecasts. Financial market practitioners evaluate forecasting models using both the minimum forecast error and directional accuracy [26]. The aim is to get a directional accuracy (DS value) of over 50 % [7]. In our study, the DS values for the benchmark and the proposed new ensemble forecasting models were greater than 50 % in all the forecasting cases. Table 5 clearly shows that the proposed ensemble model with KPCA (i.e. $DR_{KPCA}$-SVM-TLBO) outperformed the other models under consideration in this study. This is because the nonlinear kernel based PCA (i.e., KPCA) can include more discriminatory information to improve the accuracy of the forecasting model. The number in bold corresponds to the best performance.

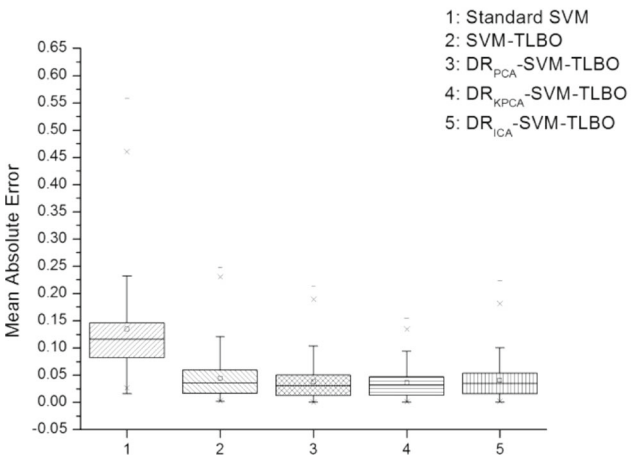
Table 6 summarizes the DM statistic with the $p$-values for the DM test given in parentheses. We compared the proposed ensemble $DR_{KPCA}$-SVM-TLBO forecasting model with two benchmark models (i.e. standard SVM and novel hybrid SVM-TLBO) and two variants of proposed models (i.e. $DR_{PCA}$-SVM-TLBO and $DR_{ICA}$-SVM-TLBO) for the 1-, 3-, and 5-days-ahead forecast cases. The results




(a) Box plot for 1-day-ahead forecast model




(b) Box plot for 3-days-ahead forecast model




(c) Box plot for 5-days-ahead forecast model

**Fig. 5** Box plot of MAE values using standard SVM, SVM-TLBO novel hybrid, and our proposed new ensemble models (i.e. $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO): (**a**) 1-day-ahead forecast, (**b**) 3-days-ahead forecast, and (**c**) 5-days-ahead forecast

in Table 6 show that the $p$-values were smaller than the chosen significance level (i.e., 5 %) and the DM test values were greater than 1.96 except for the SVM-TLBO and $DR_{ICA}$-SVM-TLBO models, when applied to the 3-days-ahead forecast. The absolute value of the DM test results of $DR_{KPCA}$-SVM-TLBO compared to SVM-TLBO was 0.4223 (p-value: 0.6733) and for $DR_{KPCA}$-SVM-TLBO compared to $DR_{ICA}$-SVM-TLBO was 0.286 (p-value: 0.7752). These values are less than 1.96, so we cannot reject the zero hypothesis at the 5 % significance level. That is, the experimental difference between the forecasting performance of these models is not significant and might be due to stochastic variations. From these observations, we can conclude that the proposed DR-SVM-TLBO (all three variants) yields more accurate predictions than the benchmark models. And among the proposed ensemble models, $DR_{KPCA}$-SVM-TLBO performed the best. Table 7 gives the percentage improvements of the proposed ensemble model for all three variants (i.e. $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO) over the benchmark novel hybrid SVM-TLBO model for the out-of-sample (test) data with respect to the RMSE and MAE. Figure 5a, b, and c show box plots of the MAE for the 1-, 3-, and 5-days-ahead forecasts, respectively, for the standard SVM, SVM-TLBO novel hybrid, and the proposed models (i.e. $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO). The middle square in each box plot represents the MAE. The box plots clearly show that $DR_{KPCA}$-SVM-TLBO has the smallest range and smallest standard error deviation (denoted by the lines above and below the box). This shows that the $DR_{KPCA}$-SVM-TLBO model outperformed all the other models.

## 5 Conclusions and future work

In this study, we extended the novel hybrid SVM-TLBO model by incorporating dimension reduction techniques. To reduce the number of input variables (features), we used three well known dimensional reduction techniques: PCA, KPCA, and ICA. We used multicommodity futures index data collected from MCX to examine the feasibility of the proposed ensemble model. Our models performed better than existing methods. Our conclusions are summarized as follows

1. The average computational time results (Table 4) suggest that reducing the number of input variables (features) decreased the computational time.
2. Our empirical results show that DR-SVM-TLBO (i.e. $DR_{PCA}$-SVM-TLBO, $DR_{KPCA}$-SVM-TLBO, and $DR_{ICA}$-SVM-TLBO) produced better predictions than the standard SVM regression method and the SVM-TLBO hybrid model. Among the three variants of the proposed ensemble model, $DR_{KPCA}$-SVM-TLBO performed the best.
3. $DR_{KPCA}$-SVM-TLBO improved the RMSE by 50.23 % (for the 1-day-ahead forecast), 28.43 % (for the 3-days-ahead forecast), and 20.03 % (for the 5-days-ahead forecast), when compared with the SVM-TLBO hybrid regression model. The $DR_{KPCA}$-SVM-TLBO model also improved the MAE result by 55.45 % (1-day-ahead), 30.63 % (3-days-ahead), and 17.87 % (5-days-ahead), when compared with the SVM-TLBO novel hybrid regression model. There were similar improvements in terms of MAE and RMSE for the other two variants of the proposed model (i.e. $DR_{PCA}$-SVM-TLBO and $DR_{ICA}$-SVM-TLBO).
4. The results of the DM statistical test (Table 6) show that all the DM tests comparing the proposed model ($DR_{KPCA}$-SVM-TLBO) with the other models yielded values greater than 1.96 (the threshold value at the 5 % significance level). The corresponding $p$-values lie within the 5 % significance level in all cases except $DR_{KPCA}$-SVM-TLBO compared to SVM-TLBO and $DR_{KPCA}$-SVM-TLBO compared to $DR_{ICA}$-SVM-TLBO, for the 3-days-ahead forecast. The DM test confirms that the predictive accuracy of our proposed model is statistically significantly better than that of the benchmark models.

In this study, we selected quantitative technical indicators (features) based on previous research work by different researchers in this area and feedback from a domain expert. We could improve the predictive performance by including non-quantitative factors like data from breaking news and social media, efficient macroeconomics factors, and psychological factors. One limitation of this study is that we used a relatively small dataset. Despite this, we achieved reasonably good forecasts. The proposed hybrid model should provide better forecasting results when applied to larger volumes of data. The successful application of our proposed model to non-linear and highly complex financial time-series data suggests that it may be useful in other domains.

**Acknowledgments** We would like to express our gratitude to the National Institute of Science and Technology (NIST), for the facilities and resources provided at the Data Science Laboratory at NIST for the development of this study.

**Compliance with ethical standards**

**Conflict of interests** The authors declare that there are no conflict of interests (either financial or non-financial) regarding the publication of the paper.

## Appendix A: Technical indicators (features) used in this study

**Formulas for technical indicators (features)**

Notation: $i$: $i$-th day [*i days* ($i = 1, 2, \ldots, N$) *since a reference date (February 1, 2010, in the experiment)*]

$HP_i$: highest index value of $i$-th day

$LP_i$: lowest index value of $i$-th day

$OP_i$: open index value of $i$-th day

$CP_i$: closing index value of $i$-th day

| Sl No. | Technical Indicator Name | Technical Indicator Description & Formula |
|---|---|---|
| 1 | 10-day moving average | The most current 10-day average closing value of the financial instrument. $MA_{10,i} = \frac{\sum_{j=i-9}^{i} CP_j}{10}$ |
| 2 | 20-day bias | Closing value and the moving average value deviation for the past 20 days. $BIAS_{20,i} = \frac{CP_i - MA_{20,i}}{MA_{20,i}}, where MA_{20,i} = \frac{\sum_{j=i-19}^{i} CP_j}{20}$ |
| 3 | Moving average convergence/divergence (MACD) | MACD is the change between a 26-day and 12-day exponential moving average (EMA). ** $MACD_i = EMA_{12,i} - EMA_{26,i}, where$ $EMA_{N,i} = (CP_i - EMA_{N,i-1}) \times (2/(N+1)) + EMA_{N,i-1}$ **EMA gives more weight to recent values and decreasing weight to older data. |
| 4 | Stochastic indicator $\%K$ | Stochastic $\%K$ compares where a security's value closed relative to its value range over a given period. (In this experiment, we used a period of 9 days.) $\%K_i = \frac{(CPi - LLP)}{(HHP - LLP)} \times 100$ where *LLP* is the lowest low index value and *HHP* is the highest high index value over the last $N$ periods. |
| 5 | Stochastic indicator $\%D$ | Moving average of $\%K$ (three-period simple moving average) $\%D_i = \frac{\sum_{j=0}^{2} \%K_{i-j}}{3}$ |
| 6 | Stochastic slow $\%D$ | Moving average of $\%D$ (three-period simple moving average) $\%SD_i = \frac{\sum_{j=0}^{2} \%D_{i-j}}{3}$ |
| 7 | Larry William's $\%R$ | Larry William's $\%R$ is a momentum indicator that measures overbought/oversold levels. (In this experiment, we used a period of 9 days.) $\%R_i = \frac{(HP - CP_i)}{(HP - LP)} \times 100$ where *LP* is the lowest index value and *HP* is the highest index value over the last $N$ periods. |
| 8 | Rate of change (ROC) | Ratio of current closing value to the value a certain number of periods ($n$ periods) ago. (In this experiment, we used a period of 10 days.) $ROC_i = \frac{CP_i}{CP_{i-n}} \times 100$ where $CP_{i-n}$ is the Closing Index Value of the (i-n)-th day. |
| 9 | Relative strength index (RSI) | RSI is a momentum oscillator that compares the magnitude of recent gains to the magnitude of recent losses. (In this experiment, we used a period of 5 days.) $RSI_i = \frac{AG_i}{AG_i + AL_i} \times 100,$ $where$ $G_i = \begin{cases} CP_{i-1} - CP_i, if CP_i > CP_{i-1} \\ 0 \end{cases}$ $and\ L_i = \begin{cases} CP_{i-1} - CP_i, if CP_i < CP_{i-1} \\ 0 \end{cases}$ $AG_i = \frac{4}{5} \times AG_{i-1} + \frac{1}{5} \times G_i and AL_i = \frac{4}{5} \times AL_{i-1} + \frac{1}{5} \times L_i$ |

**Formulas for technical indicators (features)**

Notation: $i$: $i$-th day [*i days (i=1,2,...,N) since a reference date (February 1, 2010, in the experiment)*]

$HP_i$: highest index value of $i$-th day

$LP_i$: lowest index value of $i$-th day

$OP_i$: open index value of $i$-th day

$CP_i$: closing index value of $i$-th day

| Sl No. | Technical Indicator Name | Technical Indicator Description & Formula |
|---|---|---|
| 10 | Commodity channel index (CCI) | CCI measures the variation of a security's value from its statistical mean. (In this experiment, we used a period of 24 days.) $CCI_i = \frac{TP_i - MATP_i}{0.015 \times MD_i}$, $where$ $TP_i = \frac{HP_i + LP_i + CP_i}{3}$, $MATP_i = \frac{\sum_{j=i-23}^{i} TP_j}{24}$, $MD_i = \frac{\sum_{j=i-23}^{i} \lvert TPj - MATPi \rvert}{24}$ where $TP_i$ is the typical value for the $i$-th day, $MATP_i$ is the 24-day simple moving average of the typical value for the $i$-th day, and $MD_i$ is the 24-day mean deviation for the $i$-th day. |
| 11 | Psychological line | Psychological line is the volatility indicator based on the number of time intervals that the market was rising during the preceding period. (In this experiment, we used a period of 13 days.) $PSY_i = \frac{TDU_i}{13} \times 100\%$ where $TDU_i$ is the total number of days with regard to the rise in index value in the previous 13 days. |
| 12 | Buying/selling momentum indicator | Buying/selling momentum indicator (26 days) $BSMI_i = \frac{\sum_{j=i-25}^{i} (HP_j - OP_j)}{\sum_{j=i-25}^{i} (OP_j - LP_j)}$ |
| 13 | Buying/selling willingness indicator | Buying/selling willingness indicator (26 days) $BSWI_i = \frac{\sum_{j=i-25}^{i} (HP_j - CP_{j-1})}{\sum_{j=i-25}^{i} (CP_{j-1} - LP_j)}$ |
| 14 | Momentum | Momentum measures the amount that a security's value has changed over a given period (4 days) $MO_i = CP_i - CP_{i-4}$ |
| 15 | Disparity 5 | Measures the distance between the current value and the moving average over 5 days $DIS_{5,i} = \frac{CP_i}{MA_{5,i}}$ where $MA_{5,i}$ is the 5-day moving average for the $i$-th day. |
| 16 | Disparity 10 | Measures the distance between the current value and the moving average over 10 days $DIS_{10,i} = \frac{CP_i}{MA_{10,i}}$ where $MA_{10,i}$ is the 10-day moving average for the $i$-th day. |
| 17 | Moving average oscillators (MAO) | Value oscillator that displays the difference between two moving averages of different lengths (5 and 10 days) $MAO_i = \frac{MA_{5,i} - MA_{10,i}}{MA_{5,i}}$ where $MA_{5,i}$ and $MA_{10,i}$ are the 5- and 10-day moving averages for the $i$-th day. |

## Appendix B: Dimensionality reduction techniques used in this study

The objective of a dimension reduction technique is to reduce the dimension (number of features) of the input from a high-dimensional space to a low dimensional subspace. Dimensional reduction methods can be divided into two types: (i) feature selection and (ii) feature extraction. In feature selection, a subset of features is selected from the originals. In feature extraction new features are computed by transforming the original features. We present brief reviews of the dimensional reduction methods based on feature extraction that were used in our study. That is, PCA, KPCA, and ICA.

### B.1 Principal component analysis (PCA)

Principal component analysis is a well-known linear statistical approach for feature extraction. The objective is to reduce the dimension of the input features from the original

dataset [20]. It uses an orthogonal transformation to convert a set of $N$ patterns (samples) of $l$ possibly correlated features into a set of $N$samples of $m(\leq l)$ uncorrelated features called principal components (PCs). The transformation mechanism is designed such that the first principal component (PC1) has the highest possible variance, the second principal component (PC2) is orthogonal to the PC1 and accounts for next highest variance, and so on for the other PCs.

The PCA procedure is briefly described as follows

Step 1: Input $N$patterns (samples) $X_1, X_2, \ldots, X_N$ that each have $l$ features ($X_j \in R^l$). Each vector $X_j$for $j = 1, 2, \ldots, N$ is such that the mean value of the features in $X_j$ is zero (that is, we subtract the mean value of the original feature from each feature value).

Step 2: Compute the covariance matrix

$$C = \frac{1}{N}\sum_{k=1}^{N} X_k X_k^T \tag{B.1}$$

The $ij$-th element of matrix $C$is

$$C_{ij} = \frac{1}{N}\sum_{k=1}^{N} X_k(i) X_k(j) \tag{B.2}$$

where $X_k(i)$denotes the $i$th component of the $X_k$sample.

Step 3: Calculate $l$ eigenvalues of $C$ and arrange them in non-increasing order $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_l$. For each eigenvalue $\lambda_i$, $i = 1, 2, \ldots, l$, compute an associated eigenvector $\alpha_i \in R^l$of matrix $C$ using an eigenvector decomposition technique [35].

Step 4: Choose the $m \leq l$ largest eigenvalues (choose the smallest integer $m$, so that $\lambda_{m-1} - \lambda_m$ is large or $\sum_{i=1}^{m} \lambda_i \geq t \sum_{i=1}^{N} \lambda_i$ where $t = 0.95$ if we wish to retain 95 % variance in the transformed data, where $\sum_{i=1}^{N} \lambda_i$ represents the total variance).

Step 5: Use the eigenvectors (column vectors) $\alpha_1, \alpha_2, \ldots, \alpha_m$ to form the transformation matrix.

$$A = [\alpha_1 \alpha_2 \ldots \alpha_m] \tag{B.3}$$

Step 6: Transform each pattern $X_i$ in the original space $R^l$ to the vector $Y_i$ in the m-dimensional space $R^m (m < l)$ using

$$Y_i = A^T X_i, i = 1, 2, \ldots, N \tag{B.4}$$

So the $j$th component $Y_i(j)$ of $Y_i$ is the projection of $X_i$ on $\alpha_i$(i.e., $Y_i(j) = \alpha_j^T X_i$).

### B.2 Kernel principal component analysis (KPCA)

In the PCA technique, each input pattern (sample) in $R^l$ is linearly projected onto a lower dimensional subspace. This is appropriate when the data approximately lie on a linear manifold (for example a hyperplane). However, in many applications the input data lie on a low dimensional nonlinear manifold. Then it is more appropriate to use KPCA, which is a nonlinear dimensional reduction technique. In this method the input patterns $X_i \in R^l$ for $i = 1, 2, \ldots, N$ (where $N$ is the number of input samples) are first mapped onto a space $H$ with more than $l$ dimensions using a nonlinear mapping $\phi : R^l \rightarrow H$[42]. Their images $\phi(X_i)$ are projected along the orthonormal eigenvectors of the covariance matrix of $\phi(X_i)$'$s$. These projections only involve the inner product of the $\phi(X_i)$'$s$ in $H$, $\phi$ is not explicitly known, and it is difficult to construct a kernel function. So we use $K$defined by $K : R^l \times R^l \rightarrow R$ such that

$$K(X_i, X_j) = < \phi(X_i), \phi(X_j) > \tag{B.5}$$

(where $<,>$ denotes inner product in $H$) to compute the inner products involved in the projections leading to the computation of $Y_i$'$s$ having fewer dimensions $m(m < l)$than $X_i$'$s$. It has been proved that the components $Y_i(k), k = 1, 2, \ldots, m$ of the $Y_i$'$s$ are uncorrelated and the first $q(\leq m)$ principal components have maximum mutual information with respect to the inputs, which justifies the use of the method for dimensionality reduction.

The KPCA procedure is given in the form of the following algorithm.

Step 1: Input the data patterns (samples) $X_i \in R^l$ for $i = 1, 2, \ldots, N$ (where $N$ is the number of input samples).

Step 2: Choose a kernel function $K : R^l \times R^l \rightarrow R$ and compute the kernel matrix $K_1$whose $ij$-th element is equal to $K(X_i, X_j)$ for $i, j = 1, 2, \ldots, l$

Step 3: Compute the eigenvalues and eigenvectors of$K_1$. Arrange the eigenvalues in non-increasing order $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_l$. Let the corresponding eigenvectors be $a_1, a_2, \ldots, a_l$.

Step 4: Choose $m$dominant eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_m (m \leq l)$[choose the smallest integer $m$ such that $\lambda_{m-1} - \lambda_m$is large or $\sum_{i=1}^{m} \lambda_i \geq t \sum_{i=1}^{N} \lambda_i$, where $t = 0.95$ if we wish to retain 95% of the variance in the transformed data, and $\sum_{i=1}^{N} \lambda_i$represents the total variance], and normalize the corresponding eigenvectors $a_1, a_2, \ldots, a_m$ using

$$a_k' = \frac{a_k}{\|a_k\| \sqrt{\lambda_k}}, k = 1, 2, \ldots, m \tag{B.6}$$

Step 5: For each $X_i, i = 1, 2, \ldots, N$, compute the $m$ projections $Y_i(k)$ of $\phi(X_i)$ onto each of the orthonormal

eigenvectors $a_{k'}$'$s$, $k = 1, 2, \ldots, m$, *i.e.*,

$$Y_i(k) = \sum_{j=1}^{l} a_k^i(j) K(X_i, X_j), k = 1, 2, \ldots, m \quad \text{(B.7)}$$

### B.3 Independent component analysis (ICA)

Independent component analysis (ICA) is a relatively new statistical method [14, 16]. ICA does not transform uncorrelated components or factors, but instead attempts to find statistically independent components or factors in the transformed vectors. The primary goal of this method is to find representations of non-Gaussian data, so those components are statistically independent or as independent as possible [16].

In ICA, we assume that $l$measured variables $X = [x_1, x_2, \ldots, x_l]^T$ can be expressed as linear combinations of $n$ unknown latent source components $S = [s_1, s_2, \ldots, s_n]^T$, i.e.,

$$X = AS \quad \text{(B.8)}$$

where $A_{l \times l}$ is an unknown mixing matrix. Here, we consider that $l \geq n$ if $A$ is a full rank matrix. $S$ is the latent source data that cannot be directly observed from the input mixture data, $X$. The basic ICA objective is to estimate the latent source components, $S$, and unknown mixing matrix $A$ from $X$ with appropriate assumptions on the statistical properties of the source distribution. The basic ICA model for feature transformation aims to find a de-mixing matrix $W_{l \times l}$ that can be written as

$$Y = WX \quad \text{(B.9)}$$

where $Y = [y_1, y_2, \ldots, y_n]^T$ is the independent component vector. The elements of $Y$must be statistically independent and are called independent components (ICs). Here, $W = A^{-1}$(i.e., the de-mixing matrix $W$is the inverse of mixing matrix $A$). The ICs $(y_i)$ can be used to compute the latent source signals $s_i$.

Many algorithms can perform the ICA. The fixed-point fast ICA method presented by Hyvärinen and Oja [15] is the most popular. We used fixed-point fast ICA in our experimental study. In this algorithm, PCA is first used to transform the original input vectors $(X)$ to a set of new uncorrelated vectors with zero means and unity variance. This process reduces the dimension of $X$ and consequently reduces the number of $Y$. Then, the uncorrelated vector obtained by PCA is used to estimate the independent components vectors $(Y)$ and the transformed matrix using the fixed point algorithm.

## References


1. Cai LJ, Zhang JQ, Zongwu CAI, Kian Guan LIM (2006) An empirical study of dimensionality reduction in support vector machine. Neural Network World 16(3):177–192
2. Cao LJ (2003) Support vector machines experts for time series forecasting. Neorocomputing 51:321–339
3. Cao LJ, Chua KS, Chong WK, Lee HP, Gu QM (2003) A comparison of PCA, KPCA and ICA for dimensional reduction in support vector machines. Neurocomputing 55(1):321–336
4. Cao LJ, Tay FEH (2003) Support vector machine with adaptive parameters in financial time series forecasting. IEEE Trans Neural Netw 14(6):1506–1518
5. Chang CC, Lin CJ (2011) LIBSVM: A library for support vector machines. ACM Trans Intell Syst Technol (TIST) 2(3):27
6. Chang PC, Wu JL (2015) A critical feature extraction by kernel PCA in stock trading model. Soft Comput 19(5):1393–1408
7. Chen WH, Shih JY, Wu S (2006) Comparison of support-vector machines and back propagation neural networks in forecasting the six major Asian stock markets. International Journal of Electronic Finance 1(1):49–67
8. Das SP, Padhy S (2015) A novel hybrid model using teaching–learning-based optimization and a support vector machine for commodity futures index forecasting. Int J Mach Learn Cyber:1–15. doi:10.1007/s13042-015-0359-0
9. Diebold FX, Mariano RS (1995) Comparing predictive accuracy. J Bus Econ Stat 13(3):253–263
10. Ekenel HK Sankur B (2004) Feature selection in the independent component subspace for face recognition. Pattern Recogn Lett 25(12):377–1388
11. Haykin S (2010) Neural Networks and Learning Machines. 3rd Edition, PHI Learning Private Limited
12. Hsu CM (2013) A hybrid procedure with feature selection for resolving stock/futures price forecasting problems. Neural Comput Applic 22(3–4):651–671. doi:10.1007/s00521-011-07214
13. Huang CL, Tsai CY (2009) A hybrid SOFM-SVR with a filter based feature selection for stock market forecasting. Expert Syst Appl 36(2):1529–1539. doi:10.1016/j.eswa.2007.11.062
14. Hyvarinen A, Karhunen J, Oja E (2001) Independent component analysis. Wiley, New York
15. Hyvarinen A, Oja E (1997) A fast fixed-point algorithm for independent component analysis. Neural Comput 9(7):1483–1492
16. Hyvarinen A, Oja E (2000) Independent component analysis: algorithms and applications. Neural networks 13(4):411–430
17. Ince H, Trafalis TB (2007) Kernel principal component analysis and support vector machines for stock price prediction. IIE Trans 39(6):629–637
18. Ince H, Trafalis TB (2008) Short term forecasting with support vector machines and application to stock price prediction. Int J Gen Syst 37(6):77–687. doi:10.1080/03081070601068595
19. Jiang M, Jiang S, Zhu L, Wang Y, Huang W, Zhang H (2013) Study on parameter optimization for support vector regression in solving the inverse ECG problem. Comput Math Methods Med Article ID 158059. doi:10.1155/2013/158056

20. Jolliffe IT (2002) Principle components analysis 2nd Edition. Springer, New York
21. Kim KJ (2003) Financial time series forecasting using support vector machines. Neurocomputing 55(1):307–319
22. Kim KJ, Han I (2000) Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. Expert Syst Appl 19(2):125–132
23. Kim KJ, Lee WB (2004) Stock market prediction using artificial neural networks with optimal feature transformation. Neural Comput Applic 13(3):255–260. doi:10.1007/s00521-004-0428-x
24. Kuang F, Zhang S, Jin Z, Xu W (2015) A novel SVM by combining kernel principal component analysis and improved chaotic particle swarm optimization for intrusion detection. Soft Comput 19:1187–1199. doi:10.1007/s00500-014-1332-7
25. Lai RK, Fan CY, Huang WH, Chang PC (2009) Evolving and clustering fuzzy decision tree for financial time series data forecasting. Expert Syst Appl 36(2):3761–3773. doi:10.1016/j.eswa.2008.02.025
26. Leung MT, Daouk H, Chen AS (2000) Forecasting stock indices: a comparison of classification and level estimation models. Int J Forecast 16(2):173–190
27. Liang X, Zhang H, Xiao J, Chen Y (2009) Improving option price forecasts with neural networks and support vector regressions. Neurocomputing 72(13):3055–3065. doi:10.1016/j.neucom. 2009.03.015
28. Lin HT, Lin CJ (2003) A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods Technical report, University of National Taiwan Department of Computer Science and Information Engineering, March 1–32
29. Lin SW, Ying KC, Chen SC, Lee ZJ (2008) Particle swarm optimization for parameter determination and feature selection of support vector machines. Expert Syst Appl 35(4):1817–1824
30. Liu S, Tian L, Huang Y (2014) A comparative study on prediction of throughput in coal ports among three models. Int J Mach Learn Cybern 5(1):125–133. doi:10.1007/s13042-013-0201-5
31. Lu CJ (2013) Hybridizing nonlinear independent component analysis and support vector regression with particle swarm optimization for stock index forecasting. Neural Comput Applic 23(7–8):2417–2427. doi:10.1007/s00521-012-1198-5
32. Lu CJ, Lee TS, Chiu CC (2009) Financial time series forecasting using independent component analysis and support vector regression. Decis Support Syst 47(2):115–125
33. Musa AB (2014) A comparison of 1-regularizion, PCA, KPCA and ICA for dimensionality reduction in logistic regression. Int J Mach Learn Cybern 5(6):861–873. doi:10.1007/s13042-013-0171-7
34. Pawar PV, Rao RV (2013) Parameter optimization of machining using teaching-learning-based optimization algorithm. Int J Adv Manuf Technol 67:995–1006
35. Porikli F, Haga T (2004) Event detection by eigenvector decomposition using object and frame features. IEEE Conference In Computer Vision and Pattern Recognition Workshop 2004(CVPRW'04):114–114
36. Rao RV, Patel V (2014) A multi-objective improved teaching-learning based optimization algorithm for unconstrained and constrained optimization problems. Int J Ind Eng Comput 5(1):1–22. doi:10.5267/j.ijiec.2013.09.007
37. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315
38. Sapankevych NI, Sankar R (2009) Time series prediction using support vector machines: a survey. IEEE Comput Intell Mag 4(2):24–38. doi:10.1109/MCI.2009.932254
39. Tay FE, Cao LJ (2002) Modified support vector machines in financial time series forecasting. Neurocomputing 48(1):847–861
40. Tsai CF, Hsiao YC (2010) Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. Decis Support Syst 50(1):258–269. doi:10.1007/s00500-014-1350-5
41. Tsang PM, Kwok P, Choy SO, Kwan R, Ng SC, Mak J, Tsang J, Koong K, Wong TL (2007) Design and implementation of NN5 for Hong Kong stock price forecasting. Eng Appl Artif Intell 20(4):453–461. doi:10.1016/j.engappai.2006.10.002
42. Twining CJ, Taylor CJ (2003) The use of kernel principal component analysis to model data distributions. Pattern Recogn 36(1):217–227
43. Van Gestel T, Suykens JA, Baestaens DE, Lambrechts A, Lanckriet G, Vandaele B, Vandewalle J (2001) Financial time series prediction using least squares support vector machines within the evidence framework. IEEE Trans Neural Netw 12(4):809–821
44. Vapnik V (1995) The nature of statistical learning theory. Springer, NY
45. Wang J, Wang J (2015) Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks. Neurocomputing 156:68–78
46. Wang S, Meng B (2011) Parameter selection algorithm for support vector machine. Prog Environ Sci 11:538–544. doi:10.1016/j.proenv.2011.12.085
47. Wu CH, Tzeng GH, Lin RH (2009) A Novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression. Expert Syst Appl 36(3):4725–4735. doi:10.1016/j.eswa.2008.06.046
48. Zhai G, Chen J, Wang S, Li K, Zhang L (2015) Material identification of loose particles in sealed electronics devices using PCA and SVM. Neurocomputing 148:222–228. doi:10.1016/j.neucom.2013.10.043


**Shom Prasad Das** received his B.Tech. from Berhampur University and M.Tech. from Biju Patnaik University of Technology (BPUT). He is currently working as a Faculty at National Institute of Science & Technology, Berhampur, India, Odisha. He has total of 16 years of experience in Industry and Academics. He is currently pursuing his Ph.D. under BPUT. His area of interest is in machine learning, data analytics, and object oriented programming.

**N Sangita Achary** currently pursuing her M.Tech in Computer Science and Engineering at National Institute of Science & Technology, Berhampur, Odisha, India. Her areas of research interest are on artificial neural network, and data mining.

**Sudarsan Padhy** is currently Professor of Computer Science & Engineering at Silicon Institute of Technology, Bhubaneswar. Prior to this he was Director at Institute of Mathematics and Applications, Bhubaneswar, India and Professor of Mathematics at Utkal University, India. He obtained his Ph.D. degree in Mathematics from Utkal University in 1979 and pursued Postdoctoral research at University of Freiburg, Germany during 1980-81. He has over sixty published research papers and five books to his credit extending over fluid dynamics, finite difference and finite element method for solving partial differential equations, operation research, parallel algorithms, computational finance, machine learning, and computational biology.