Selection of a Machine Learning Based Model for Prediction of Commodity Futures Index

Shom Prasad Das[1], andSudarsanPadhy[2]
[1]Department of Computer Science and Engineering, National Institute of Science and Technology
Palur Hills, Berhampur, Odisha761008, India
[2]Silicon Institute of Technology, Silicon Hills, Bhubaneswar, Odisha 751024, India

Abstract- The hugenumber of factors including economic, political, environmental, psychological, complex behavior of noisy and non-stationary data has made financial forecasting from time-series as one of the most interesting as well as most challenging problems. This study presents the use of variants of three most promising machine learning regression models i.e., Support Vector Machines (SVMs), Multiple Kernel Learning (MKL),and Extreme Learning Machine (ELM) for forecasting indexvalues of the commodity futures traded on the National Commodity & Derivatives Exchange Limited in India.The past studies concluded that, forecasting performance of models may vary using different performance evaluation measures (criteria). The objectives of this paper are to: (i) empirically study and compare the performance of the different variants of three promising machine learning models, and (ii) apply multi-criteria decision making (MCDM) technique (Technique for Order Preference by Similarity to Ideal Solution i.e., TOPSIS) to rank the forecasting models. The results show that the use of exclusive performance measure may lead to untrustworthyconclusions;however this situation can be overcome by the use of multi criteria decision making techniques.

Keywords- Machine learning, Support vector machines (SVMs), Multiple kernel learning (MKL), Multi-criteria decision making, Commodity futures index,Financial time series

I. INTRODUCTION

The financial market is a complex, evolutionary, and non-linear dynamical system [1]. In the business environment, the ability to forecast a variety of different financial variables accurately and efficiently is necessary to ensure the development of proper strategies and to avoid the potential risk of financial losses [6]. The financial forecasting from time series data is considered as one of the most interesting and challenging problems because of huge number of factors like economic, political, environmental, and psychological, and the complex nature of data which govern the forecasting mechanism. In the recent years data mining techniques have been implemented successfully to provide improved decision supportto the stakeholders in the financial domainsuch as predicting the prices (values) of different financial instruments, bankruptcy threats, and credit scores.

Forecasting financial time-series data has been studied since the 1980s by different researchers with the aim of beating the financial market. The improvement in machine learning framework and successful application of machine learning in various domains including financial domain, have inspired researchers to use different machine learning tools and techniques to predict financial markets. The machine learning frameworks are data-driven, non-parametric models, and they let "the data speak for themselves" [31]. Vapnik and his coworkers introduced the use of support vector machinesto overcome the difficulties of neural networks (NNs) such as getting trapped in local minima, overfitting to training data, and lengthy training time[35]. Unlike most of the traditional learning machines that adopt the Empirical Risk Minimization Principle, SVMs implement the Structural Risk Minimization Principle, which seeks to minimize an upper bound of the generalization error rather than minimize the training error. This will result in better generalization than conventional techniques [12]. Chen and Shih [6] proposed a model using SVM and NN for forecasting six major Asian stock market. Tay and Cao [32] and Cao and Tay [4] developed pricing models for five specific financial futures in the US market using SVMs, while Gestel et al. [11] used an LS-SVM for T-bill rates and stock index pricing in the US and German markets. Several authors have proposed financial instrument pricing using kernel based SVM, with their simulated results showing that the SVM method outperforms the NN one [4, 6, 11,24].

Although, the kernel based SVM techniques has been used for forecasting the financial instruments but the selection of kernel function and the free parameters of the learning mechanism were purely done empirically with no general guidelines and theoretical justification to it. Unsuitably chosen kernel function and the associated free parameters may lead to significantly poor performance[9, 18, 41].In addition, due to noisy, non-stationary, varying distribution, and unstructured nature of financial data use of a single kernel function may not be efficient to solve complex financial problems. Several researchers have used multiple kernel learning[2, 27]to deal with the above problems by fusion of kernels [10, 21, 38,41].MKL framework mitigates the threat of flawed kernel

choice to certain degree by taking a set of kernels and deriving a weight for each kernel such that predictions are computed based on weighted sum of the kernels. As MKL framework uses combination of multiple heterogeneous basic kernels, the process to obtain optimal convex combination of kernels takes more time than the single kernel based SVM.

The machine learning techniques i.e., single kernel SVM and multiple kernel learning only focuses on the forecasting accuracy, and rarely analyzed the computational time required to execute the model that is a very important factor in forecasting task, since good or bad time complexity indicates applicability of the model in practices [41]. Huang and his coworkers [13-14] proposed a promising supervised technique, called extreme learning machinehaving comparable accuracy and fast forecasting speed to solve some real-life problems and Li et al. [22] used this technique for stock market prediction. However, the comparative study of performances of SVM and ELM by Liu et al. [20] demonstrated that ELMs has potential to yield generalization behavior as good as the SVM when the size of training sample is large. So when large data is not available application of ELM may not be suitable, but SVM technique is suitable for small and large data set.

The evaluation of algorithms in general is a central issue in the different fields like artificial intelligence, operations research, machine learning, and data mining and knowledge discovery [30].The performance evaluation of learning methods is an important topic in financial instruments' prediction, sincea minor improvement in forecasting performance can have a positive impact on a financial investment. Previous studies show that, performances in financial prediction may differ using different performance measures and under different conditions [6,31]. Rokach [28] proposed that the algorithm selection can be considered as multi criteria decision making (MCDM) problem and MCDM techniques can be used to methodicallyselect the appropriate algorithm. To our best knowledge, up to date, use of MCDM technique to evaluate the performance of forecasting models is very limited.

The main objective of the study is to evaluate and compare some promising machine learning methods using six criteria measures like root mean square error (RMSE), mean absolute percentage of error (MAPE), directional symmetric (DS), weighted directional symmetric (WDS), R squared (R^2), and computational time complexity. For the performance evaluation a multi criteria decision making based Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) method is used to rank the prediction models. The TOPSIS is one of the MCDM approaches known for reliable assessment of results, fast computation process, easily interpretable and ease of use and understanding [39-40]. Moreover, some studies [3, 19, 25, 29, 36, 42] have implemented TOPSIS to solve MCDM problem successfully.

Ourempirical study is designed to assess five prediction models (SVM for regression with radial basis function: SVM+RBF, SVM for regression with polynomial kernel function: SVM+POLY, multiple kernels SVM learning: MKL, ELM with radial basis activation function: ELM+RADBAS, and ELM with sigmoidal activation function: ELM+SIGMOID)using six performance measures mentioned above for forecasting the values of commodity index on futures Dhaanya, traded in the National Commodity & Derivatives Exchange (NCDEX) Limited in India. The commodity futures index under consideration in our experiment highlights the importance of agriculture in India and provides a benchmark to the Indian agriculture futures sector. In the experiment, due to nonlinear nature of financial data, only non-linear kernels like radial basis and polynomial are used in the SVM for regression and MKL models. Similarly, nonlinear activation functions are used in the ELM regression models. In our experiment, we use rolling-over training-testing samples for datasets because the dynamics of financial market would change and we try to make our model adapt and robust to the changes. Our experimental results show that, this rolling-over technique obtain stable and better result.

The rest of the paper is structured as follows. In Sections 2, we provide a summary of the SVM for regression model, MKL regression model, and ELM regression model. Section 3 introduces the TOPSIS techniqueused in our study. In Section 4, the research design for our study is presented with brief discussion on data used, data preprocessing, input features selection, performance criteria, and implementation of five machine learning models and the TOPSIS technique. Section 5, provides computational results and analysis on the results of our study. Finally, Section 6 gives the conclusions of the study with a brief discussion of the findings and possible future work.

II. FORECASTING MODELS

A. SVM for regression

Vapnik et al. [35] developed an SVM technique for regression [12], which we briefly describe below.

Given a training dataset $\{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_l}, y_l)\}$ (where each $\mathbf{x_i} \in X \subset R^n$, and X denotes the input sample space), and matching target values $y_i \in R$ for $i = 1, \ldots, l$ (where l denotes 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 when $\mathbf{x}$ is not in the training set.

The estimation function, f, is defined as

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

where $\mathbf{w} \in R^m$, $b \in R$ is the bias, and $\mathbf{\Phi}$ denotes a nonlinear function from R^n to high-dimensional space R^m ($m > n$). $f(\mathbf{x})$ is determined by minimizing the risk

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

where L_{ε} is the extension of the ε-insensitive loss function originally proposed by Vapnik [35]. C is a user-specified constant known as the regularization parameter, which is used to modulate the trade-off between empirical and generalization errors.

The problem (***P***) is solved using the primal-dual method to obtain

$$f(\mathbf{x}) = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*)(\mathbf{\Phi(x_i)}^T \mathbf{\Phi(x)}) + b,$$

$$= \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) K(\mathbf{x_i}, \mathbf{x}) + b. \tag{3}$$

Here, $K : X \times X \to R$ is the Mercer kernel defined as

$$K(\mathbf{x}, \mathbf{z}) = \mathbf{\Phi(x)}^T \mathbf{\Phi(z)} . \tag{4}$$

$\{\alpha_i\}_{i=1}^{l}$ *and* $\{\alpha_i^*\}_{i=1}^{l}$ are the solutions to the primal quadratic optimization problem

$$\text{Maximize } Q(\alpha_i, \alpha_i^*) = \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) - \varepsilon \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) K(\mathbf{x_i}, \mathbf{x_j}), \tag{5}$$

subject to

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

$$(ii) \quad 0 \le \alpha_i \le C, \ 0 \le \alpha_i^* \le C \text{ , where } i = 1, \ldots, l \tag{7}$$

The bias (b) is calculated using the Karush-Kuhn-Tucker (KKT) conditions [35], that is,

$$b = y_i - \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) K(\mathbf{x_j}, \mathbf{x_i}) - s\varepsilon \text{ where } s = 1 \ \textit{for} \ 0 < \alpha_i < C \text{ and } s = -1 \ \textit{for} \ 0 < \alpha_i^* < C. \tag{8}$$

B. Multiple kernel learning regression

The support vector machines for regression method presented in equation (1) uses a single function Φ and hence single kernel function K is used. If the dataset for the learning method has a locally varying distribution, using single kernel like in SVR may not be efficient to catch up the varying distribution. Instead of using one single kernel function, numerous kernel functions are combined. This learning mechanism using multiple kernels i.e., Multiple Kernel Learning (MKL) model provides a more flexible framework to extract information, patterns and forecast data more efficiently and effectively. This technique mitigates the risk of erroneous single kernel selection in support vector machines based learning to some degree by taking a set of kernels and deriving a weight for each kernel such that predictions are made based on a weighted sum of several kernels. In the multiple kernels learning mechanism an equivalent kernel is created by a linear convex combination of series of base kernels and is used in the learning mechanism for the given set of data. For example for M number of kernel functions $\Phi(x)$ can be represented as:

$$\Phi(x) = \left[\sqrt{\delta_1}\Phi_1(x), \sqrt{\delta_2}\Phi_2(x), \ldots, \sqrt{\delta_M}\Phi_M(x)\right]^T \tag{9}$$

where $\delta_1, \delta_2, \ldots, \delta_M$ are weights of kernel functions. With the introduction of multiple kernels, the objective function and constraints for multiple kernel regression problems become

$$\min_{\delta} \min_{w,b} C\left[\sum_{i=1}^{l} (\zeta_i + \zeta_i')\right] + \frac{1}{2}\|w\|^2 \text{ subject to}$$

$$y_i - w^T \Phi(x_i) - b \le \varepsilon + \zeta_i,$$

$$w^T \Phi(x_i) + b - y_i \le \varepsilon + \zeta_i',$$

$$\zeta_i, \zeta_i' \ge 0,$$

$$\delta_s \ge 0, s = 1, 2, \ldots, M,$$

$$\sum_{s=1}^{M} \delta_s = 1, \tag{10}$$

where $\Phi(x)$ is the vector of function mappings of equation (9). By introducing the Lagrangian multipliers, the Eq. (10) can be transformed to the following form:

$$\min_{\delta} \max_{\alpha_i, \alpha_i^*} \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) - \varepsilon \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) K(x_i, x_j),$$

subject to

$$\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0,$$
$$0 \le \alpha_i \le C,\ 0 \le \alpha_i^* \le C,\ i = 1,2,...,l$$
$$\delta_s \ge 0, s = 1,2,...,M,$$
$$\sum_{s=1}^{M}\delta_s = 1 \qquad (11)$$

where

$$\begin{aligned} K(x_i, x_j) &= \Phi(x_i)^T \Phi(x_j) \\ &= \delta_1(\Phi_1(x_i)^T \Phi_1(x_j)) + \delta_2(\Phi_2(x_i)^T \Phi_2(x_j)) \\ &\quad + ... + \delta_M(\Phi_M(x_i)^T \Phi_M(x_j)) \\ &= \delta_1 K_1(x_i, x_j) + \delta_2 K_2(x_i, x_j) + ... + \delta_M K_M(x_i, x_j) \\ &= \sum_{s=1}^{M}\delta_s K_s(x_i, x_j) \end{aligned}$$

$$\min_{\boldsymbol{\beta} \in R^{n_h \times m}} \left[\frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{C}{2}\sum_{i=1}^{N}\left\|\mathbf{e}_i\right\|^2 \right] \text{ subject to } \mathbf{H}\boldsymbol{\beta} = \mathbf{Y} - \mathbf{E}, \qquad (15)$$

The bias (b) is computed using Karush-Kuhn-Tucker (KKT) condition [35], similar to the solution of single kernel in equation (8).

In order to solve the multi-objective optimization problem in equation (11), various techniques have been proposed by different researchers. Here we discussed a simple and efficient technique proposed by Rakotomamonjy et al. [27] called SimpleMKL for solving the MKL problem. The SimpleMKL is two-stage optimization algorithm, in the first stage; the problem in equation (11) is optimized using sequential minimal optimization (SMO) [26] keeping the weight vector δ fixed. In the next stage, the Lagrange multipliers α and α^* are kept fixed and the weight vector δ is computed using reduced gradient method [23]. Further details on SimpleMKL algorithm can be found in Rakotomamonjy et al. [27]

C. Extreme learning machine (ELM)

The ELM algorithm was proposed by Huang et al. [13-14] to overcome the slow learning speed of feed-forward neural networks, which is due to their slow gradient-based learning algorithms and iterative determinations of all the network parameters.The ELM algorithm has been shown to have a shorter training time, with generalization performance results very close to those of the SVM.

(12)

is weighted sum of M kernel functions $K_1, K_2, ..., K_M$, corresponding to $\Phi_1, \Phi_2, ..., \Phi_M$, respectively.

If we can find the values of δ, α and α^* by solving Eq.(11), the multiple kernel form of regression hyperplance corresponding to single kernel of equation (1) would be,:

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\, K(x_i, x) + b, \qquad (13)$$

Given a supervised learning problem with N arbitrary distinct training samples $(\mathbf{x_i}, \mathbf{y_i})$, where $\mathbf{x_i} = [x_{i1}, x_{i2}, ..., x_{in}]^T \in R^n$ and $\mathbf{y_i} = [y_{i1}, y_{i2}, ..., y_{im}]^T \in R^m$, ELM learning is realized in two stages. In the first stage, we randomly generate the weight and bias parameters between the input and single hidden layer neurons. Then, outputs at the neurons in the hidden layer are produced using appropriate activation functions, *g(x)*. The remaining free tuning parameters (i.e., output weights between the hidden neurons and output nodes) are optimized in the training phase. This is the most distinctive feature of an ELM network, when compared with traditional feed-forward neural networks.

In an ELM network, the relationship between output vector $\mathbf{o_j}$ and input vector $\mathbf{x_j}$ is given by

$$\sum_{i=1}^{n_h}\boldsymbol{\beta}_i\, g(\mathbf{w_i} \cdot \mathbf{x_j} + b_i) = \mathbf{o_j},\ \ j = 1,2,...,N \qquad (14)$$

where $\mathbf{w_i} = [w_{i1}, w_{12}, ..., w_{in}]^T$ is the weight vector connecting the *i*-th hidden neuron and the input neurons, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{im}]^T$ is the output weight vector connecting the *i*-th hidden neuron and the output neurons, b_i is the bias (threshold) of the *i*-th hidden neuron, and n_h is the total number of neurons in the hidden layer.

In the second stage, the output weight vectors $\boldsymbol{\beta}_i$, $i=1,2,...,n_h$ are obtained by solving

where C is the penalty coefficient of the training error, and $\mathbf{e}_i \in R^m$ is the error vector with respect to the i-th training sample.

$$\mathbf{H}(\mathbf{w}_1,\ldots,\mathbf{w}_{n_h},b_1,\ldots,b_{n_h},\mathbf{x}_1,\ldots,\mathbf{x}_N) = \begin{bmatrix} g(\mathbf{w}_1.\mathbf{x}_1+b_1) & \ldots & g(\mathbf{w}_{n_h}.\mathbf{x}_1+b_{n_h}) \\ \vdots & \ldots & \vdots \\ g(\mathbf{w}_1.\mathbf{x}_N+b_1) & \ldots & g(\mathbf{w}_{n_h}.\mathbf{x}_N+b_{n_h}) \end{bmatrix}_{N\times n_h} \quad (16)$$

where

$$\boldsymbol{\beta}=\begin{bmatrix}\boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_{n_h}^T\end{bmatrix}_{n_h\times m}, \ \mathbf{Y}=\begin{bmatrix}\mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T\end{bmatrix}_{N\times m}, \text{ and } \mathbf{E}=\begin{bmatrix}\mathbf{e}_1^T \\ \vdots \\ \mathbf{e}_N^T\end{bmatrix}_{N\times m} \quad (17)$$

The solution of (2) is [15]

$$\boldsymbol{\beta}^* = \left(\mathbf{H}^T\mathbf{H}+\frac{\mathbf{I}_{n_h}}{C}\right)^{-1}\mathbf{H}^T\mathbf{Y}, \quad (18)$$

where $\mathbf{I}_{n_h}$ is an identity matrix with dimension n_h, if the number of training samples is greater than or equal to the number of nodes in the hidden layer (i.e., $N \geq n_h$). Otherwise, it is

$$\boldsymbol{\beta}^* = \mathbf{H}^T\left(\mathbf{H}\mathbf{H}^T+\frac{\mathbf{I}_N}{C}\right)^{-1}\mathbf{Y}, \quad (19)$$

where $\mathbf{I}_N$ is an identity matrix with dimension N.

III. TECHNIQUE FOR ORDER PREFERENCE BY SIMILARITY TO IDEAL SOLUTION (TOPSIS) METHOD

TOPSIS, one of the multi criteria decision making method used in our study, was proposed by Hwang and Yoon [15], which is briefly describe below. TOPSIS selects one optimal solution for real-world problem from some alternative available solutions, where all alternate solutions can be used to solve the current problem. Our concern is to select an ideal solution that is superior to all other alternatives. In order to find the most appropriate solution, TOPSIS technique assumes that, most preferred alternative solution have shortest distance from the positive ideal solution and largest distance from the negative ideal solution. In TOPSIS, the positive-ideal solution is a solution that maximizes the "benefit" criteria (criteria which improve the performance when the performance measure increases in value) and minimizes the cost criteria (criteria which improves the performance when the performance measure decreases in value), where negative ideal solution, does exactly the opposite. The operational procedure of TOPSIS can be summarized as follows:

Suppose the current problem has *J* alternatives (i.e., forecasting models), and each of the alternatives has *N* number of criteria (i.e., performance measures).Let x_{ij}denote the value for the ith criterion $C_i$of the jth alternative A_j.

Step 1: Calculate the normalized decision matrix (r_{ij}) where:

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^{J} x_{ij}^2}}, \ j=1,\ldots,J \text{ and } i=1,\ldots,N \quad (20)$$

Step 2: Calculate the weighted normalized decision matrix (v_{ij}) where:

$$v_{ij} = r_{ij}\times w_i, \ j=1,\ldots,J \text{ and } i=1,\ldots,N \quad (21)$$

where w_i is the weight of the ithcriterion or feature and $\sum_{i=1}^{N} w_i = 1$.

Step 3: Determine the positive ideal (S^+) and negative ideal (S^-) solutions.

$$S^+ = \{v_i^+ \mid i=1,2,\ldots,N\} = \left\{\left(\max_{1\leq j\leq J} v_{ij} \mid i\in I'\right), \left(\min_{1\leq j\leq J} v_{ij} \mid i\in I''\right)\right\}, \quad (22)$$

$$S^- = \{v_i^- \mid i=1,2,\ldots,N\} = \left\{\left(\min_{1\leq j\leq J} v_{ij} \mid i\in I'\right), \left(\max_{1\leq j\leq J} v_{ij} \mid i\in I''\right)\right\}, \quad (23)$$

where I' is associated with the benefit criteria and I'' is associated with the cost criteria.

For the evaluation of our regression models,

Step 4: Calculate the separation measures using the N-dimensional Euclidean distance. The separation of each alternative from the positive ideal solution is calculated as:

$$D_j^+ = \sqrt{\sum_{i=1}^{N} (v_{ij} - v_i^+)^2}, \; j = 1, ..., J. \tag{24}$$

And the separation of each alternative from the negative-ideal solution is calculated as:

$$D_j^- = \sqrt{\sum_{i=1}^{N} (v_{ij} - v_i^-)^2}, \; j = 1, ..., J. \tag{25}$$

Step 5: Compute the R_j^+ that measures relative closeness to the ideal solution.

$$R_j^+ = \frac{D_j^+}{(D_j^+ + D_j^-)}, \; j = 1, ..., J \tag{26}$$

Step 6: Rank the alternatives by the non-increasing order of R_j^+ ratio value (i.e. highest R_j^+ value is ranked 1).

IV. RESEARCH DESIGN AND IMPLEMENTATION

In this section, we describe our procedure and experimental set-up for the study as shown in Fig. 1. A list of models consider for our study is shown in Table 1 along with their description.

The experiment was carried out according to the following steps:

Input: Time series data

Output: Ranking of regression models

START

Step 1: Collect the raw time series data from the source (In our study source: NCDEX website)

Step 2: Feed the raw time series data to preprocess module to compute and prepare input feature (see Section4.2 for detail description and implementation setup)

Step 3: Use the preprocessed data sets prepared in step 2 to train and test the five regression models. (see Section 4.4, 4.5, and 4.6 for detail description)

Step 4: Calculate the six performance measures (criteria) described in Section 4.3 for each regression model. The results are summarized in Tables 6.

Step 5: Generate the rankings of regression models using TOPSISas per the steps for computation given in Section 4.7. The results obtainedfrom Step 4 are used as inputs to the TOPSIStechnique.

Step 6: The rankings of regression models using TOPSISare summarized in Tables 7.

END

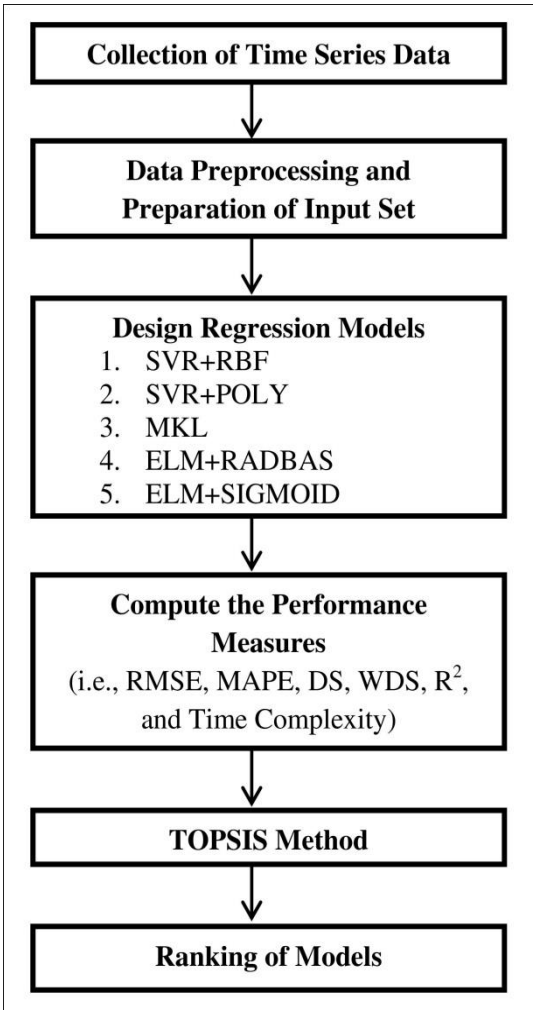


Fig.1: Flowchart of the process for the proposed model

Table 1 A list of models for our experiment

Sl.No.	Description of the model	Abbreviation
1	SVM regression with radial basis kernel function	SVR+RBF
2	SVM regression with polynomial kernel function	SVR+POLY
3	Multiple kernel learning regression with fusion of radial basis and polynomial kernels	MKL
4	Extreme learning machine regression with radial basis activation function	ELM+RADBAS
5	Extreme learning machine regression with sigmoidal activation function	ELM+SIGMOID

A. Data description

Our forecasting model uses real index on commodity futures (i.e., Dhaanya) data collected from the NCDEX Ltd. of India (http://www.ncdex.com). A total of 1,146 daily trading data points of the Dhaanyafuture commodity index were collected from NCDEX between November 1, 2011 and November 30, 2015. Each index data point consists of the daily opening price, low price, high price, closing price, and trading date. The period during which the data were collected includes many important and significant economic events; thus, we consider these data to be suitable for training and testing our models. Table 2 describes the dataset in terms of high, low, mean, and median prices, 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). The raw daily closing index prices are plotted in Fig. 2. Table 2 and Fig. 2 shows that, the skewness value of the dataset is less than zero i.e., the dataset is left skewed distribution (most values are concentrated on 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 more than three i.e., leptokurtic distribution (sharper than a normal distribution, thicker tails, and high probability for extreme values). This clearly shows that, our Dhaanya index time series data do not exhibit identical statistical properties at each point of time.

Table 2 Description of dataset for NCDEX Dhaanya Index

Parameter (for entire collection period)	High	Low	Mean	Median	Standard deviation	Kurtosis	Skewness
Price	3024.97	1285.45	2438.988	2469.35	342.0939	3.257389	-1.67454

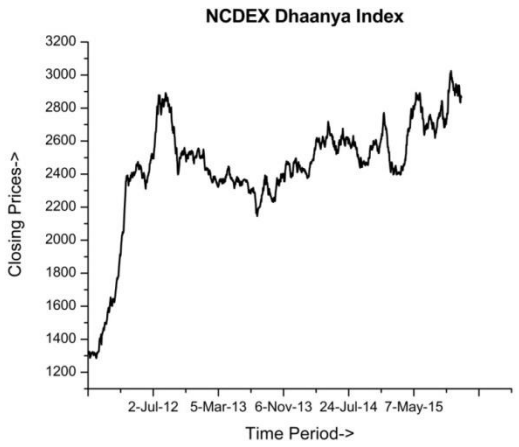


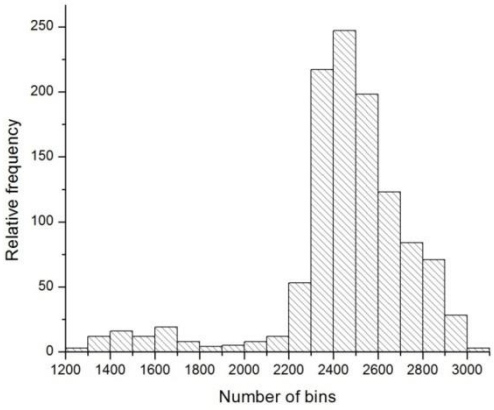
Fig.2: Closing index values of NCDEX Dhaanya Index (entire dataset)

B. Data preprocessing and model inputs selection

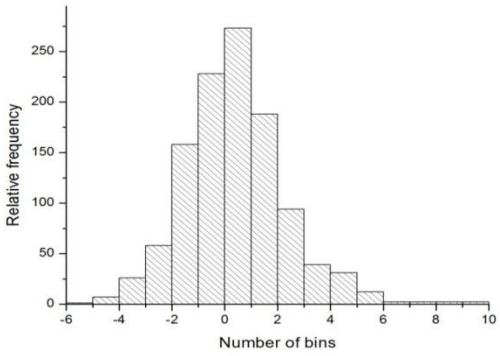
In order to improve the predictive power of the machine learning models used in our study, we transformed the original closing price of the index into a five-day Relative Difference in Percentage of price (RDP). This transformation makes the distribution of data more symmetric and to follows a normal distribution more closelyas illustrated in Fig. 3(b) and Fig. 3(c)[32-34]. Further, financial technical indicators are computed to include as input features to our models. In our study, four-lagged RDP values (i.e., RDP-5, RDP-10, RDP-15, and RDP-20), one transformed closing index price (EMA5) calculated by subtracting a five-day exponential moving average from closing price, and three financial technical indicators are considered as input features for the models. The input features are prepared based on the previous literature, feedback from the domain experts, and previous research work [7-8, 17, 32-34].The selections of input features are done very carefully so that, the trend in the closing prices are removed by use of RDP transformed inputs, original closing price information are retained by EMA5 input variable, volatility of the market is fed to the models by use of psychological line (PSY) indictor, stochastic (*%K*) indicator maintains thetrend of the index with respect to highest and lowest index values, and the Larry William's (*%R*) capture the overbought/oversold momentum of the index. The output variable (RDP+5) is computed by initially smoothing the closing price of the index with a three-day exponential moving averages. The description and formula of selected input features and output variables are presented in Table 3.

Having treated the 1,146 collected raw time series data points; we obtained 1,123 data points transformed in terms of input features with dates starting from December 1, 2011 to November 30, 2015. To render the forecasting model more robust and reliable, we divide the proposed data into four datasets (D-I, D-II, D-III, and D-IV). Each dataset consists of data for commodity index for 30 months (2 years and 6 months). Then, each dataset is subdivided into training and testing sets with 24 months and 6 months of data, respectively.

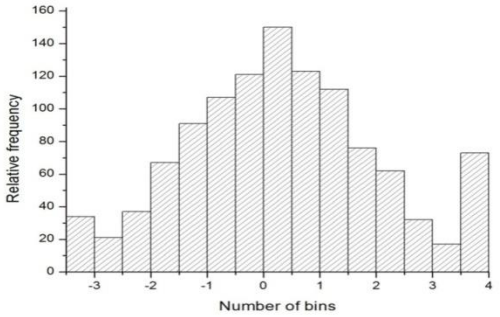
Table 4 summarized the details on the rolling-over training cum validation and testing sample for the D-I, D-II, D-III, and D-IV datasets. The Fig. 3(b) shows thatthere are outliers in the data set. Subsequently these outliers in data may make it difficult or time-consuming to converge to an effective solution for the machine learning models. In our study, the datavalues outside limits of ± 2 standard deviations are considered as outliers data andwinsorsing techniques were used to replace the values of outlier data. The input and output variables are normalized in the range -1.0 to 1.0 before being used in the training and testing phase.



(a) Dhaanya closing price



(b) RDP+5 values



(c) RDP+5 values after removal of outliers data

Fig.3: Histograms of (a) Dhaanya closing price, (b) RDP+5 values, and (c) RDP+5 values after removal of outliers data

Table 3 Input and output variables used in this study

Sl. No.	Indicator	Indicator Description & Formula
Input variables		
1	RDP-5	$(RDP-5)_i = \frac{CP_i - CP_{i-5}}{CP_{i-5}} \times 100$
2	RDP-10	$(RDP-10)_i = \frac{CP_i - CP_{i-10}}{CP_{i-10}} \times 100$
3	RDP-15	$(RDP-15)_i = \frac{CP_i - CP_{i-15}}{CP_{i-15}} \times 100$
4	RDP-20	$(RDP-20)_i = \frac{CP_i - CP_{i-20}}{CP_{i-20}} \times 100$
5	EMA5	$(EMA5)_i = CP_i - (\overline{EMA_5})_i$

6	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 price in the previous 13 days.
7	Stochastic indicator *%K*	Stochastic *%K* compares where a security's price closed relative to its price 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 price and *HHP* is the highest high index price over the last *N* periods.
8	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 price and *HP* is the highest index price over the last *N* periods.
Output variables		
1	RDP+5	$(RDP+5)_i = \frac{\overline{CP_{i+5}} - \overline{CP_i}}{\overline{CP_i}} \times 100$ $\overline{CP_i} = (\overline{EMA_3})_i$
Notation: *i*: *i*-th day [*i days (i=1,2,…,N) counted from reference date, December 1, 2011, in the experiment*] CP_i: closing index value of *i*-th day; $(\overline{EMA_N})_i$: N-day exponential moving average of i-th day		

Table 4 Rolling-over training-testing samples for datasets

Datasets	Training	Testing
D-I	1st Dec, 2011 – 30th Nov, 2013 (Sample Size: 597)	1st Dec, 2013 – 31st May, 2014 (Sample Size: 141)
D-II	1st Jun, 2012 – 31st May, 2014 (Sample Size: 588)	1st Jun, 2014 – 30th Nov, 2014 (Sample Size: 127)
D-III	1st Dec, 2012 – 30th Nov, 2014 (Sample Size: 566)	1st Dec, 2014 – 31st May, 2015 (Sample Size: 128)
D-IV	1st Jun, 2013 – 31st May, 2015	1st Jun, 2015 – 30th Nov, 2015

	(Sample Size: 547)	(Sample Size: 130)

C. Performance measures

The performance of the alternate models was evaluated using the standard sixmetrics, shown in Table 5. The performance criteria are selected to include measuring the regression models in different aspect. For example, RMSE is selected to measure the error in terms of absolute value, MAPE and R^2 are used for relative error, DS provides an indication of the correctness of the predicted direction, WDS measures both the magnitude of the prediction error and the direction, and time complexity measures the time required for executing the model. In the ideal case in forecasting task, the RMSE, MAPE, WDS, and time complexity should have minimum values while other criteria like DS and R^2, should have maximum value. In the study, all performance criteria were considered equally important while measuring the predictive performance of the forecast models.

Table 5 Definitions of performance metrics and desired values

Performance Metrics	Definition	Desired Values
RMSE	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - y_i)^2}$	Minimum
MAPE	$\frac{1}{N}\sum_{i=1}^{N}\left\lvert\frac{(p_i - y_i)}{y_i}\right\rvert$ *100%	Minimum
DS	$\frac{100}{N}\sum_{i=1}^{N} d_i$ $d_i = \begin{cases} 1 & \text{if } (y_i - y_{i-1})(p_i - p_{i-1}) \geq 0 \\ 0 & \text{otherwise} \end{cases}$	Maximum
WDS	$\sum_{i=1}^{N} d_i \lvert y_i - p_i \rvert \Big/ \sum_{i=1}^{N} d'_i \lvert y_i - p_i \rvert$ $d_i = \begin{cases} 0 & \text{if } (y_i - y_{i-1})(p_i - p_{i-1}) \geq 0 \\ 1 & \text{otherwise} \end{cases}$ $d'_i = \begin{cases} 1 & \text{if } (y_i - y_{i-1})(p_i - p_{i-1}) \geq 0 \\ 0 & \text{otherwise} \end{cases}$	Minimum
R^2	$1 - \frac{\sum_{i=1}^{N}(y_i - p_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$	Maximum
Time	Computational Time Complexity	Minimum

where N is the total number of sample data, y_i is the actual output value, and p_i is the predicted output value of the i-th sample, and $\bar{y}$ is the mean of actual output

In the subsequent sections 4.4, 4.5, and 4.6, we present the details on computational techniques used and the experimental setups in SVM, MKL, and ELM regression models in our study. All the experiments were executed on an Intel Core i7 CPU @ 2.10 GHz, with 6 GB primary memory. Moreover, for the purpose of getting a more general result, we repeat each experiment 30 times on every data set.

D. SVM Implementation

We implemented Vapnik's SVM regression technique using LIBSVM, which is an SVM tool box [5]. SVMs for financial time series forecasting commonly use the polynomial kernel ($k(x,y)=(x.y+1)^d$) or the Gaussian kernel ($k(x,y)=\exp((-1/\square^2)\|x-y\|^2)$), d is the degree of the polynomial kernel and $\square$ is the width (bandwidth) of the Gaussian kernel. We used the Gaussian kernel (radial basis) function as the kernel function for the SVR+RBF model and the polynomial kernel were used for the SVR+POLY model. To train the standard SVM regression model, free parameters of the SVM regression model and that of the kernel function were determined using a grid search [16] with 5-fold cross validation. Different pairs of (C, $\square$) were used in the experiments for SVR+RBF model, where C is the regularization parameter and $\square$ is the bandwidth of the radial basis kernel function. Similarly, a different pair of (C, $\boldsymbol{d}$) used for SVR+POLY model, where d is the degree of the polynomial kernel function. We keep the ε (insensitive loss function radius) parameter constant at a reasonable value (i.e., 0.0001), because the number of support vectors decreases as ε increases, when ε is greater than 0.01 [4]. The result with the minimum error was selected for comparison in our study. For the experiments, we prepared a search table that takes C values in the range 0.01 to 100, $\square$values in the range 0.01 to 100, and $\boldsymbol{d}$ values in the rage of 1 to 5. After obtaining the final values of free parameters in each case using the grid search technique, both the forecasting models (i.e., SVR+RBF and SVR+POLY), were trained with the entire training dataset for all the four datasets (i.e., D-I, D-II, D-III, and D-IV) to create the final forecasting model. The out-of-sample (testing) performances of the models are tabulated in Table 6.

E. MKL Implementation

The radial basis function (RBF) and polynomial kernels have frequently been used in financial market forecast problem, e.g. [6, 17, 37]. In the experiment for the MKL, we have used the SimpleMKL [27] regression as discussed in section 2.2.

For the multiple-kernel learning experiment, we have considered fusion of RBF and polynomial kernel functions. The $\boldsymbol{C}$ (regularization parameter) was fixed to 1 and the ε (insensitive loss function radius) parameter was fixed to 0.0001. In case for RBF kernels, 37 different settings of kernel parameter $\square$(bandwidth) were considered in the range from 0.01 to 100. From 0.01 to 0.09 with a step size of 0.01, 0.1 to 0.9 with step size of 0.1, 1 to 10 with step size of 1, and 10 to 100 with step size 10 [38]. Similarly, for the polynomial kernels, 5 different setting of kernel parameter $\boldsymbol{d}$(degree) were considered [$\boldsymbol{d} \in \{1,2,\ldots,5\}$].In total there are 42 (37 RBF kernels and 5 polynomial kernels) were used for the experiment. Training and testing for all the data sets (i.e., D-I, D-II, D-III, and D-IV) were experimented and the out-of-sample (testing) performances of the model are tabulated in Table 6.

F. ELM Implementation

In the implementation of ELM regression, we implemented the regression model with two different activation functions (i) ELM with radial basis activation function: ELM+RADBAS, and (ii) ELM with sigmoidal activation function: ELM+SIGMOID). The programming codes used for may be downloaded from the following webpage: http://www.ntu.edu.sg/home/egbhuang/elm_codes.html. Basic ELM code has been used which requires only the number of hidden nodes to be tuned. We experimented using different numbers nodes in the hidden layer (in the ELM) and found that the optimal performance when the number of hidden nodes were more than the sample size. The optimal performance of this model for testing (out-of-sample) is tabulated in Table 6.

G. TOPSIS Implementation

In the implementation of TOPSIS method, we have five alternative forecasting models as described in Table 1 and each alternative models have six criteria (performance measures) as given in Table 5. The alternative forecasting models and their respective criteria are input to the TOPSIS method. The TOPSIS method runs as per the steps given in Section 3. The weights required to compute the weighted normalized decision matrix are always set by experts and is nontrivial. In this study in order to simplify the experiment, equal weights are set for each of six criteria by maintaining $\sum_{i=1}^{N} w_i = 1$ (where w_i is the weight of the ith criterion). In table 5, criterion (performance indicator) with desired value of maximum are set as "benefit criteria" and the criterion with minimum desired value are set as "cost criteria" for TOPSIS procedure. Finally, by using the TOPSIS method, the ranking

of the forecasting models according to their general performances is obtained and is summarized in Table 7. The rankings of the forecasting models are reached according to the value of measures of relative closeness to the ideal solution (R_j^+). The higher value of R_j^+ means that, it is closer to the distance from the ideal solution and it is further from the negative ideal solution.

V. EXPERIMENTAL RESULTS AND DISCUSSION

The experimental results for the SVR+RBF model, the SVR+POLY model, the MKL model, the ELM+RADBASmodel, and the ELM+SIGMOID modelin terms of performance for the testing (out-of-sample) phase is presented in Table 6.It is observed that average result of MKL forecasting model showed better performance in terms of RMSE, MAPE, DS, and R^2 performance measures compared to other four forecasting models. The SVM+RBF performed better for the WDS criterion and SVM+POLY performed better in terms of computational time complexity criterion.The results tabulated in Tables 6 indicate that no forecasting model attains the best performance across all performance measures (criteria) and therefore, one might draw different conclusions about the best performing forecasting model depending on the performance evaluation measured used. The numbers in bold denote the average best performance and numbers in underlined represent the average second best performance.Table 7, shows that SVR+RBF has highest R_j^+ value and hence ranked top. The MKL forecasting model is next in the ranking. The ELM+RADBAS and ELM+SIGMOD succeeded the MKL model respectively. In this study, SVR+POLY value is lowest in the ranking.

Table 6 Performance of out-of-sample (testing) results of SVR+RBF,SVR+POLY, MKL, ELM+RADBAS, and ELM+SIGMOID model.

Forecasting Models	Datasets	RMSE	MAPE	DS	WDS	R^2	Time
	D-I	0.0236	0.3462	93.5719	0.037	0.9857	0.111
	D-II	0.0255	0.2362	96.8254	0.0448	0.9831	0.1
	D-III	0.0293	0.2432	98.4252	0.0218	0.9878	0.097
	D-IV	0.0388	0.4963	93.7984	0.0327	0.9835	0.093
SVR+RBF	**Average**	0.0293	0.3305	95.6552	**0.0341**	0.9850	0.1003
	D-I	0.0843	0.6973	83.5714	0.1833	0.8182	0.072
	D-II	0.0987	0.9395	86.5079	0.1192	0.7474	0.063
	D-III	0.1165	0.7284	87.4015	0.1648	0.8088	0.059
	D-IV	0.12	2.6036	87.5969	0.0856	0.8428	0.056
SVR+POLY	**Average**	0.1049	1.2422	86.2694	0.1382	0.8043	**0.0625**
	D-I	0.0204	0.2924	95.7142	0.0226	0.9893	1.9036
	D-II	0.0221	0.171	96.8254	0.0444	0.9873	1.808
	D-III	0.0268	0.1922	97.6378	0.029	0.9898	1.7007
	D-IV	0.0323	0.4321	95.3488	0.0406	0.9886	1.5552
MKL	**Average**	**0.0254**	**0.2719**	**96.3816**	0.0342	**0.9888**	1.7419
	D-I	0.0349	0.5029	89.2857	0.1394	0.9689	0.234
	D-II	0.0427	0.3075	92.8571	0.0689	0.9527	0.1872
	D-III	0.1093	0.6498	89.7637	0.1364	0.8318	0.1872
	D-IV	0.0874	0.7966	96.124	0.0446	0.9165	0.2496
ELM+RADBAS	**Average**	0.0686	0.5642	92.0076	0.0973	0.9175	0.2145
	D-I	0.0325	0.4003	90	0.0936	0.9728	0.1872
	D-II	0.0346	0.3531	91.2698	0.1196	0.9689	0.1872
	D-III	0.105	0.6035	84.2519	0.4317	0.8447	0.2028
	D-IV	0.0704	1.0463	96.8992	0.0411	0.9459	0.0144
ELM+SIGMOD	**Average**	0.0606	0.6008	90.6052	0.1715	0.9331	0.1479

Table 7 TOPSIS rankings for forecasting models of our study.

Forecasting Models	TOPSIS (R_j^+)	Ranks
SVR+RBF	**0.9531**	1
MKL	0.6307	2
ELM+RADBAS	0.5929	3
ELM+SIGMOD	0.4617	4
SVR+POLY	0.3499	5

VI. CONCLUSIONS AND FUTURE WORK

Although different machine learning models have been used for forecasting the financial instruments by different researcher in literature, we propose a broad framework for ranking multiple regression models with multiple performance criteria using multi criteria decision making model.The data used in our experiments were real data and hence, the results can be considered more practical.

In this study, we attempted to propose a two-stage framework to rank multiple forecasting models using TOPSIS as MCDM technique. In the first stage, performance score is computed for each forecasting (regression) model to evaluate whether superior or inferior performance of one forecasting model over another. The second stage applies MCDM based TOPSIS method to rank the forecasting models. The proposed framework has been analyzed empirically using real financial time series data on commodity futures index. Although the results shows that, MKL forecasting model performed better than other models in terms of RMSE, MAPE, DS, and R Square performance measures and SVM+RBF performed better than other models in terms of WDS and time complexity but the SVR+RBF regression model ranked top in the ranking list of forecasting models when all the measures are taken into consideration. The ranking on MKL degraded because of its higher time complexity. For the decision making where the time is not a major concern, the MKL should be the most preferred method.

As a future research direction, there is a possibility to rank the forecasting models using other different MCDM techniques a like Analytic Hierarchy Process (AHP), Preference Ranking Organization METHod for Enrichment of Evaluations (PROMETHEE), VIKOR, and Data Envelopment Analysis (DEA) which will help to further evaluate the strength of the proposed framework.A limitation of the study is the limited dataset. Despite the available data size being comparatively small, reasonably good results were attained for forecasting the commodity futures index. The proposed hybrid model should provide better forecasting results given larger volumes of data. In this research, we developed the hybrid model by combining machine learning regression models and MCDM technique for solving the function approximation problem and rank the models based on multiple criteria. We feel that the methodology proposed in this paper is quite promising and successful application to non-linear and highly complex financial time-series data suggests useful application of the proposed model in other domains.

VII. ACKNOWLEDGEMENT

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.

VIII. COMPLIANCE WITH ETHICAL STANDARDS

The authors declare that there are no conflicts of interest (either financial or non-financial) regarding the publication of the paper.

IX. REFERENCES

[1]. Abu-Mostafa YS, Atiya AF (1996) Introduction to financial forecasting. Applied Intelligence 6: 205-213.

[2]. Bach F, Lanckriet G, Jordan M (2004) Multiple kernel learning, conic duality, and the SMO algorithm. Proceeding of the International Conference on Machine Learning, pp. 41-48.

[3]. Bai C, Dhavale D, Sarkis J (2014). Integrating Fuzzy C-Means and TOPSIS for performance evaluation: An application and comparative analysis. Expert Systems with Applications, 41(9), 4186-4196.

[4]. Cao L J , Tay F E H (2003) Support vector machine with adaptive parameters in financial time series forecasting. IEEE Transactions on Neural Networks 14(6): 1506–1518.

[5]. Chang CC, Lin C J (2011) LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2(3):27

[6]. Chen W-H, Shih J-Y, Wu S (2006) Comparison of support vector machines and back propagation neural networks in forecasting the six major Asian stock markets. International Journals Electronics Finance 1 (1): 49-67.

[7]. Chih-Ming H (2013) A hybrid procedure with feature selection for resolving stock/futures price forecasting problems. Neural Computing and Applications 2013: 651-671. doi: 10.1007/s00521-011-07214

[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. International Journal of Machine Learning and Cybernetics 1–15. doi: 10.1007/s13042-015-0359-0

[9]. Duan K, Keerthi S, Poo AN (2003) Evaluation of simple performance measures for tuning SVM hyperparameters. Neurocomputing 51, 41-59.

[10]. Fletcher T, Hussain Z, Shawe-Taylor J (2010) Multiple kernel learning on the limit order book. JMLR: Workshop and Conference Proceedings 11: 167–174.

[11]. Gestel T V, Suykens JAK, Baestaens DE, Lambrechts A, Lanckriet G, Vandaele B, Moor BD ,Vandewalle J (2001) Financial time-series prediction using least squares support

vector machines within the evidence framework. IEEE Transactions on Neural Networks 12(4): 809–821.
[12]. Haykin S (2010) Neural Networks and Learning Machines, 3rd Edition. PHI Learning Private Limited.
[13]. Huang G, Zhu Q, Siew C (2004) Extreme learning machines: A new learning scheme of feedforward neural networks. Proc. Int. Joint Conf. Neural Netw. (2): 985-990.
[14]. Huang G, Zhu Q, Siew C (2006) Extreme Learning Machines: Theory and Applications. Neurocomputing 70 (1):489–501.
[15]. Hwang CL, Yoon KS (1981). Multiple attribute decision making. Berlin: Spring-verlag.
[16]. Keerthi SS (2002) Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. IEEE Transactions on Neural Networks 13(5):1225–1229.
[17]. Kim K (2003) Financial time series forecasting using support vector machines. Neurocomputing 55: 307-319.
[18]. Kwok J T-Y (2000) The evidence framework applied to support vector machines. IEEE Transaction on Neural Networks 11(5), 1162-1173.
[19]. Li H, Adeli H, Sun J, & Han JG (2011). Hybridizing principles of TOPSIS with case-based reasoning for business failure prediction.Computers& Operations Research, 38(2), 409-419.
[20]. Liu X, Gao C, Li P (2012) A comparative analysis of support vector machines and extreme learning machines. Neural Networks 33: 58-66. doi: 10.1016/j.neunet.2012.04.002
[21]. Li X, Huang X, Deng X, Zhu S (2014) Enhancing quantitative intra-day stock return predicting by integrating both market news and stock prices information. Neurocomputing 142: 228-238.
[22]. Li X, Xie H, Wang R, Cai Y, Cao J, Wang HM, Deng X (2014) Empirical analysis: stock market prediction via extreme learning machine. Neural Comput&Applic. doi 10.1007/s00521-014-1550-z.
[23]. LuenbergerDG & YeY(1984). Linear and nonlinear programming (Vol. 2). Reading, MA: Addison-wesley.
[24]. Min JH, Lee Y-C (2005) Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. Expert Systems with Applications 28: 603-614.
[25]. Peng Y, Wang G, Kou G, & Shi Y (2011). An empirical study of classification algorithm evaluation for financial risk prediction. Applied Soft Computing, 11(2), 2906-2915.
[26]. PlattJC(1999). 12 fast training of support vector machines using sequential minimal optimization. Advances in kernel methods, 185-208.
[27]. Rakotomamonjy A, Bach F, Canu S, Grandvalet Y (2008) SimpleMKL. J. Mach. Learn. Res. 9: 2491–2521.
[28]. Rokach L (2010). Ensemble-based classifiers. Artificial Intelligence Review, 33(1-2), 1-39.
[29]. Sangaiah AK, Subramaniam PR, &Zheng X (2015). A combined fuzzy DEMATEL and fuzzy TOPSIS approach for evaluating GSD project outcome factors. Neural Computing and Applications, 26(5), 1025-1040.
[30]. Smith-Miles KA (2008) Cross-disciplinary perspectives on meta-learning for algorithm selection. ACM Comput. Surv. 41(1): 1-25. doi: 10.1145/1456650.1456656
[31]. Tay FEH, Cao L (2001) Application of support vector machines in financial time series forecasting. Omega 29: 309-317.
[32]. Tay FEH, Cao L (2002) Modified support vector machines in financial time series forecasting. Neurocomputing 48: 847-861.
[33]. Thomason M (1999) The practitioner methods and tool: a basic neural network-based trading system project revisited (parts 1 and 2). J. Comput. Intell. in Finance, vol. 7, no. 3, pp. 36–45.
[34]. Thomason M (1999), The practitioner methods and tool: a basic neural network-based trading system project revisited (parts 3 and 4). J. Comput. Intell. in Finance, vol. 7, no. 4, pp. 35–48.
[35]. Vapnik V (1995) The Nature of Statistical Learning Theory. Springer N Y. ISBN 0-387-94559-8
[36]. Wang YJ (2014). The evaluation of financial performance for Taiwan container shipping companies by fuzzy TOPSIS. Applied Soft Computing,22, 28-35.
[37]. Wei H, Nakamori Y, Wang S-Y (2005) Forecasting stock market movement direction with support vector machine. Comput. Oper. Res. 32 (10): 2513-2522. (doi: 10.1016/j.cor.2004.03.016)
[38]. Yeh C-Y, Huang S-W, Lee S-J (2011) A multiple-kernel support vector regression approach for stock market price forecasting. Expert Systems and Applications 38: 2177-2186.
[39]. Yurdakul MYTIC (2010) Development of a quick credibility scoring decision support system using fuzzy TOPSIS. Expert Systems and Applications 37: 567-574.
[40]. Zanakis SH, Solomon A, Wishart N, &Dublish S (1998). Multi-attribute decision making: A simulation comparison of select methods.Europeade4en journal of operational research, 107(3), 507-529.
[41]. Zhang X, Hu L, Zhang L (2013) An efficient multiple kernel computation method for regression analysis of economic data. Neurocomputing 118: 58-64.
[42]. Zhu X, Wang F, Wang H, Liang C, Tang R, Sun X, & Li J (2014). TOPSIS method for quality credit evaluation: A case of air-conditioning market in China. Journal of Computational Science, 5(2), 99-105.