Computational Economics (2020) 55:901–923
https://doi.org/10.1007/s10614-019-09922-x

Classifier Based Stock Trading Recommender Systems for Indian stocks: An Empirical Evaluation

V. Vismayaa[1] · K. R. Pooja[1] · A. Alekhya[1] · C. N. Malavika[1] · Binoy B. Nair[2] · P. N. Kumar[1]

Accepted: 27 September 2019 / Published online: 1 October 2019

Abstract
Recommender systems that can suggest the user when to buy and sell stocks can be of immense help to those who wish to trade in stocks but are constrained by their limited knowledge of stock market dynamics. Traditionally, the trading recommendations have been generated on the basis of technical analysis. However, recent research in the field indicates that soft computing/data mining based recommender systems are also capable of generating profitable trading recommendations. An attempt has been made in this study to generate novel classifier based stock trading recommender systems that employ historical stock price data and technical indicators as input features. Moreover, there have been very few studies on the effectiveness recommender systems in the context of India, the world's sixth largest economy and home to one of the world's largest stock exchanges: the Bombay Stock Exchange (BSE). This study presents an empirical evaluation the effectiveness of five single classifier and six ensemble classifier based recommender systems on a total of 293 stocks drawn from the BSE. Recommender system performance for each stock is evaluated based on classification accuracy and eight economic performance measures. Results indicate that the proposed approach can indeed be used successfully for generating profitable trading recommendations.

Keywords Indian · Stock · Trading · Recommender · Classification · Technical indicators

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10614-019-09922-x) contains supplementary material, which is available to authorized users.

✉ Binoy B. Nair
b_binoy@cb.amrita.edu

1 Department of Computer Science and Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India

2 Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India

1 Introduction

An intelligent stock trading recommender system capable of recommending buy/sell decisions to the user can be a great asset to an individual attempting to profit by trading in stocks. However, due to the nonlinear nature of stock price movements, it is considered to be an extremely challenging task. Work over the past century, e.g. in Cowles (1933) and the Efficient market hypothesis proposed by Fama (1969, 1995) even suggests that it is not possible to obtain excess returns from the markets. Though a number of studies tend to support this hypothesis, e.g. Caporale et al. (2016), there is a large body of empirical evidence, e.g. in Atsalakis and Valavanis (2009b) and Nair and Mohandas (2015a) to suggest that soft computing based techniques can be successfully employed to forecast stock price movements and hence, can be utilized for developing a successful trading strategy. However, there are two major aspects of such soft computing based recommender systems that play a significant role in their ability to make successful trading recommendations: (a) selection of the appropriate soft computing technique and (b) the choice of the optimal input feature set to ensure that the model trained using the selected soft computing technique learns the patterns in stock price movements and is capable of making profitable trading recommendations. This study is presents an innovative technique to address both these aspects by proposing the design and empirical validation of classifier based stock trading recommender systems.

Soft computing/data mining based techniques have been widely used in financial applications, for e.g. in stock signal prediction (Luo and Chen 2013), stock index and stock price prediction (Zarandi et al. 2013; Banik et al. 2007; Dai et al. 2012), exchange rate forecasting (Khashei and Bijari 2011). Atsalakis and Valavanis (2009b) presents a survey of application of soft computing techniques in financial forecasting. Soft computing/data mining based approaches tend to outperform the traditional techniques such as regression, however, it is observed that there are very few studies available with respect to the effectiveness of classifier based recommender systems. This is especially true in case of India, which also happens to be one of the largest emerging economies.

Selection of the optimal input feature set can significantly affect the performance of a forecasting/recommender system. As was observed in Nair and Mohandas (2015a), there appears to be no consensus on the input features that should be used to ensure high prediction accuracy. Historical stock time series data itself has been used as input, for e.g. in Atsalakis and Valavanis (2009a) and Saad et al. (1998).Technical indicators have been used, with varying degrees of success, for e.g. in Chan and Teong (1995), Dempster et al. (2001), Kuo and Chen (2006), Lee and Chen (2007) and Nair and Mohandas (2015b). Fundamental indicators and macroeconomic indicators have also been used, for e.g. in Abbasi and Abouec (2008) and Ballings et al. (2015) etc. However, since the recommender systems evaluated in the present study are designed to recommend trading decisions over a short time frame, fundamental and macroeconomic indicators might not be suitable as input features. In the present study, impact of three different input feature sets, (a) the daily stock data, (b) daily stock data along with technical indicators with all technical indicator window lengths set to the naïve assumption of the minimum i.e. two-days (this feature set is referred to, later in this

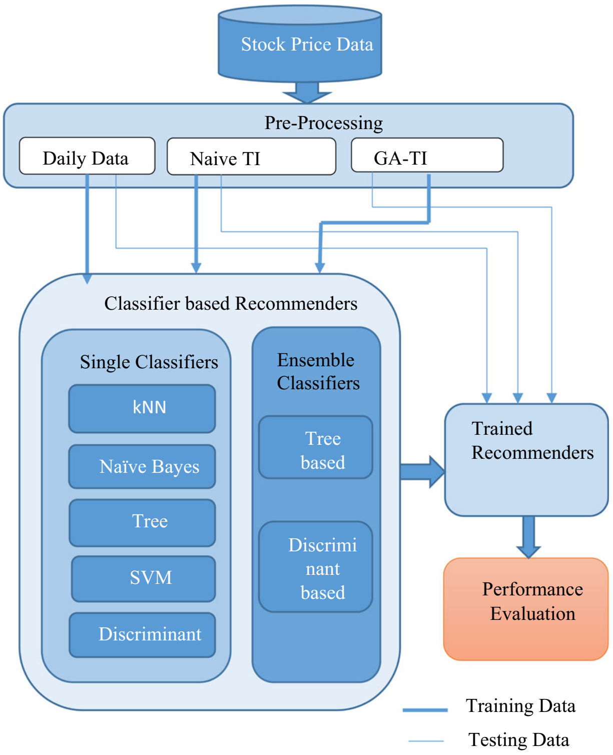

Fig. 1 Recommender system evaluation process block diagram

study as: daily data-naïve TI) and (c) daily stock data along with technical indicators with the optimal window length identified using Genetic Algorithms (GA) (this feature set is referred to, later in this study as: daily data-GA-TI), on the performance of each of the recommenders considered is also empirically evaluated on 293 stocks drawn from Group 'A' stocks in the Bombay Stock Exchange (BSE). An attempt is also made to: (1) empirically determine if ensemble classifier based recommender systems are capable of outperforming single classifiers, as in Ballings et al. (2015) and (2) to identify the optimal set of input features that can result in optimal performance. In addition to accuracy, the financial implications of following the recommendations made by each classifier is also evaluated for all the 293 stocks using eight financial performance measures, as employed in Nair and Mohandas (2015b).

Remainder of the paper is organized as follows: Sect. 2 presents the methodology followed in design of the recommender systems, Sect. 3 presents the results and the conclusions are presented in Sect. 4.

2 Methodology

Block diagram of the recommender system evaluation process is presented in Fig. 1.

2.1 Data Extraction

First step in the process is the extraction of historical stock price data. Daily data (daily open, high, low, close, adjusted close and traded volume) for 293 firms that constitute the Group 'A' firms in the Bombay Stock Exchange (BSE) is extracted from ("Yahoo! Finance India," 2015).

2.2 Pre-processing

The second step, preprocessing, involves extraction of features from the daily stock price data. As discussed in the previous section, three different input feature sets are evaluated to identify the optimal input feature set that can generate the optimal recommender performance. First feature set consists of six features (the daily stock price data alone), i.e. each input sample to the recommender at time t, can be represented as:

$$\boldsymbol{X_t} = \left\{ Open_t \; High_t \; Low_t \; Volume_t \; Close_t \; Adj\_Close_t \right\}.$$

The second and third feature sets consist of the daily stock price data along with the technical indicator values at that instant. A total of nineteen technical indicators are considered in the present study based on the results reported in Nair et al. (2010) and Nair and Mohandas (2015b). The technical indicators can be classified into three broad categories:

(a) Price based indicators: (1) Stochastics: consisting of two indices, namely, stochastic %K and %D, (2) Relative Strength Index (RSI), (3) Price Rate of Change (ProC), (4) Ulcer Index (UI), (5) Moving Average Convergence Divergence (MACD) consisting of two indices: Nine Period Moving Average (NPMA) and MACD Line, (6) Ease of Movement (EM), (7) Ultimate Oscillator (UO), (8) Acceleration (Acc), (9) Momentum (Mom), (10) William's %R, (xi) Highest High (HH), (12) Lowest Low (LL).

(b) Volume based indicators: (1) Percentage Volume Oscillator (PVO), (2) Positive Volume Index(PVI), (3) Negative Volume Index(NVI), (4) On-balance Volume (OBV).

(c) Overlays: (1) Twenty-five day Moving Average (MA (25)), (2) Sixty-five day Moving Average (MA (65)), (3) Bollinger Bands composed of three indices: Bollinger Upper (BU), Bollinger Mid (BM) and Bollinger Lower(BL).

Detailed discussions on the indicators listed above and their calculation can be found in Eng (1998) and Jobman (1998). Indicators listed above are widely used by the trading community, however, not all indicators are useful under all market conditions. Stock traders tend to select relevant technical indicators and the parameters used for calculating the technical indicators, largely based on their experience. Hence, the second feature set used in this study takes the naïve assumption and fixes calculation window size for all the technical indicators to their lowest possible values (i.e. 2 days). The third feature set employs Genetic Algorithms (GAs) to find the optimal technical indicator window size. The objective function to be maximized was the profit factor

(PF). Population size of 40 was considered with elitist selection, uniform crossover and Gaussian mutation.

2.3 Recommender System Design

Once the features are extracted, the next step is to train the recommender system using the feature set. All the recommender systems considered in the present study use classification as the primary technique for recommending stock trading decisions. The recommenders generate 'Buy' or 'Hold' recommendations on a daily basis, based on the 3-month ahead stock direction forecast. If the stock price is likely to go up by more than 5% from the current price, 3 months from the current date (taken as 65 days-ahead, considering 5-day week for the BSE and holidays). The value of 5% was chosen due to the fact that some of the largest stock broking firms in India impose a maximum brokerage charge for small value trades that is typically quoted as a fixed amount or 2.5% of the traded value per trade, whichever is lower. However, for larger value trades, a brokerage of around 0.5% (this varies slightly depending on the broker and the type of trading plan a customer opts for) of the traded value is charged. A detailed description can be found in HDFC securities Ltd. (2018) and ICICI securities Ltd. (2018). Two trades (one buy and one sell) need to be completed for generating profit (or loss). Hence, for the trader to generate a profit even for small value transactions after paying the brokerage charges, the profits should exceed 5%.

The decision table used for training the recommender systems, consisting of $\boldsymbol{N}$ samples and $\boldsymbol{M}$ features takes the form

$$\boldsymbol{D} = \left(\begin{array}{l|c} \boldsymbol{X}_t & \boldsymbol{C}_t \\ \boldsymbol{X}_{t-1} & \boldsymbol{C}_{t-1} \\ \vdots & \vdots \\ \boldsymbol{X}_{t-(\boldsymbol{N}-1)} & \boldsymbol{C}_{t-(\boldsymbol{N}-1)} \end{array} \right)$$

where $\boldsymbol{X}_t = \left\{ x_{1t}\ x_{2t}\ \ldots\ x_{Mt} \right\}$ are the $\boldsymbol{M}$ feature values at day t.

And the corresponding output class at time t, given by

$$\boldsymbol{C}_t = \begin{cases} Hold, x_{t+65} - x_t < 0.05 \\ Buy, x_{t+65} - x_t \geq 0.05 \end{cases}$$

Hence, it becomes a binary classification problem with two classes, C_{1t} = 'Hold' and C_{2t} = 'Buy'.

Two categories of classifiers were evaluated in this study: Single classifiers and Ensemble classifiers.

Single classifiers attempt to classify the data points using a single learner while ensemble classifiers use an ensemble of weak learners to generate a classifier. Single classifiers have been used, for e.g. in Nair et al. (2011). The only study apart from the one presented in this paper was on the evaluation of ensemble classifiers for stock direction prediction (Ballings et al. 2015) for European stocks. No such study on Indian stocks appears to have been carried out, so far. Following classifiers were considered for generating the recommenders:

2.3.1 K-Nearest Neighbor (kNN) Classifier

The kNN classifier (Han et al. 2006) can be considered a 'lazy' classifier since it does not require any training. All the training samples need to be stored in the memory and once a test sample $\boldsymbol{X_t}$ is available, the closest k samples to $\boldsymbol{X_t}$ are identified from the training dataset based on some distance measure such as the Euclidean distance. Now $\boldsymbol{X_t}$ is simply assigned the class to which the majority of its k nearest neighbors in the training set belong.

2.3.2 Naïve Bayes Classifier(NB)

The Naïve Bayes (Han et al. 2006) classifier, based on the Bayes theorem, attempts to find the posterior probabilities for both the classes C_{1t} and C_{2t}, given by $P(C_{1t}|\boldsymbol{X_t}) = \frac{P(C_{1t})p(\boldsymbol{X_t}|C_{1t})}{p(\boldsymbol{X_t})}$ and $P(C_{2t}|\boldsymbol{X_t}) = \frac{P(C_{2t})p(\boldsymbol{X_t}|C_{2t})}{p(\boldsymbol{X_t})}$. The sample is assigned to the class for which the posterior probability is higher, with the Naïve assumption being: $p(\boldsymbol{X_t}|C_{it}) = p(x_{1t}|C_{it}).p(x_{2t}|C_{it}).\cdots.p(x_{2t}|C_{it}), i = 1, 2$

2.3.3 Linear Discriminant Analysis Classifier (DISC)

Discriminant analysis (Teknomo 2015; Tufféry 2011) is also based on Bayes theorem, however, the probability, $p(\boldsymbol{X_t}|C_{1t})$ and $p(\boldsymbol{X_t}|C_{2t})$ are modeled as multivariate normal distributions with the density function being represented by:

$$p(\boldsymbol{X_t}|C_{it}) = \frac{1}{\sqrt{(2\pi)^M \boldsymbol{\Sigma}_i}} e^{-\frac{1}{2}(X_t-\mu_i)^T \Sigma_i^{-1}(X_t-\mu_i)}, \text{ where } i = 1,\ 2. \tag{1}$$

where $\boldsymbol{\Sigma}_i$ is the $M \times M$ dimensional covariance matrix for the i-th class; $\boldsymbol{\mu}_i$ is the $M \times 1$ dimensional vector of means of the M features in the i-th class.

The sample $\boldsymbol{X_t}$ is assigned to the class i for which $P(C_{it})p(\boldsymbol{X_t}|C_{it})$ is higher.

Since linear discriminant is employed in the present study, $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}$.

As in the Bayes classifier, the sample X_t is assigned to the class for which $P(C_{it})p(\boldsymbol{X_t}|C_{it})$, where i = 1,2; is the highest.

2.3.4 Decision Tree Based Classifier (TREE)

Classification tree employed in the present study attempts to successively partition $\boldsymbol{D}$ by selecting one among the $\boldsymbol{M}$ features into smaller subsets (nodes) such that the Gini impurity at each node of the tree is minimized. The Gini impurity at node n is given by

$$Gini_n = 1 - \left(p_{C_1}^2 + p_{C_2}^2\right) \tag{2}$$

where $Gini_n$ = Gini impurity at node n. p_{C_1}, p_{C_2} are the fractions of classes C_1 and C_2 that reach the node n.

In this study, the partitioning stops at a node n (resulting in a leaf node) if $Gini_n = 0$ or if the number of samples in a node drops below 10.

2.3.5 SVM Classifier

SVM classifier used in the present study attempts to find, from the decision table $\boldsymbol{D}$, the optimal hyperplane of the form $C(\boldsymbol{X}) = \boldsymbol{\beta}^T \boldsymbol{X} + b$, that separates the two classes C_1 and C_2 (represented typically using class labels + 1 and − 1) such that the hyperplane is at maximum distance from points in the two classes. For a given sample $\boldsymbol{X_t}$ SVM with linear kernel is used in the present study. Detailed discussion on SVM classifier can be found in Soman et al. (2010).

2.3.6 Ensemble Classifiers

Bagging, as proposed in Breiman (1994) creates an ensemble of classifiers by training the classifiers on randomly drawn $\boldsymbol{N}$ samples from the dataset, with replacement. It must be noted that $\boldsymbol{N}$ is the number of samples in the dataset.

Two boosting techniques are also evaluated: Adaboost (Freund and Schapire 1997) and RobustBoost (Yoav Freund 2009)

2.4 Performance Evaluation Measures

Performance of the recommender systems was evaluated on two fronts:

(a) Efficiency of classification, as measured by classification accuracy and
(b) Economic performance as evaluated by eight performance measures suggested in Brabazon and O'Neill (2008).

Considering the total number of trades as TT and the profit or loss made in each transaction (assuming a transaction cost of 0.5%) be represented as p_i, where i = 1,2,…,TT. This set of profit (loss) generated for each transaction can be represented as: $\boldsymbol{P} = \{ p_1 \ p_2 \ \ldots \ p_{|\boldsymbol{P}|} \}$

The set of trades that result in profit can be represented as: $\boldsymbol{S} = \{ p_1 \ p_2 \ \ldots \ p_{|\boldsymbol{S}|} \}$, $p_i \geq 0$, i= 1,2,…,|$\boldsymbol{S}$| and $\boldsymbol{S} \subseteq \boldsymbol{P}$

The set of trades that result in loss can be represented as: $= \{ p_1 \ p_2 \ \ldots \ p_{|\boldsymbol{L}|} \}$, $p_i < 0$, i= 1,2,…, |$\boldsymbol{L}$| and $\boldsymbol{L} \subset \boldsymbol{P}$

Where $|\boldsymbol{S}| + |\boldsymbol{L}| = |\boldsymbol{P}|$, $\boldsymbol{L} \cup \boldsymbol{S} = \boldsymbol{P}$ and $\boldsymbol{S} \cap \boldsymbol{L} = \phi$

1. Total Profit (TP): Total profit or loss made considering all trades, $TP = \sum_{p \in \boldsymbol{P}} p$
2. Average Profit (AP): Average profit or loss made considering all trades, $AP = \frac{1}{|\boldsymbol{P}|} \sum_{p \in \boldsymbol{P}} p$
3. Profit per Successful Trade (P/ST): Profit considering only the profit-making trades, $P/ST = \frac{1}{|\boldsymbol{S}|} \sum_{s \in \boldsymbol{S}} s$

Table 1 Distribution of stocks across sectors

Sector	Stocks (% of total)
Banking and finance	19.11
Infrastructure and realty	13.65
Pharma and healthcare	7.51
Utilities	7.51
Automotive	6.83
Chemicals and plastics	6.48
IT and consulting	6.48
Textiles and retail	4.78
Mining/metals	4.44
FMCG	4.1
Oil and gas	4.1
Transport, logistics and ports	2.73
Broadcasting and publishing	2.73
Household appliances	1.02
Telecom	1.71
Hotels	0.68
Sugar, tea/coffee and Breweries	2.39
Diversified and misc	3.75

4. Loss per Loss making Trade (L/LT) : Loss considering only the loss-making trades, $L/LT = \frac{1}{|\boldsymbol{L}|}\sum\limits_{l \in \boldsymbol{L}} l$
5. Maximum Drawdown (MD): The lowest profit(or the highest loss) incurred among all trades executed, $MD = \min\limits_{p \in \boldsymbol{P}} p$
6. Total Trades (TT): Total number of trades, $TT = |P|$
7. Profit Factor (PF) : Ratio of the total profits generated by the profit-making trades to the total loss generated by the loss-making trades, $PF = \frac{\sum_{s \in \boldsymbol{S}} s}{\sum_{l \in \boldsymbol{L}} l}$
8. Win ratio (WR): Ratio of the total profit-making trades to the total loss-making trades, $WR = \frac{|\boldsymbol{S}|}{|\boldsymbol{L}|}$

3 Results and Discussion

A comprehensive study comprising of 293 group ‘A’ stocks drawn from the BSE was carried out. The broad distribution of stocks across the sectors is given in Table 1. Stock details are presented in "Appendix 10".

The daily stock data consists of the daily Open price, Close price, High Price, Low Price, Adjusted Closing Price and Volume. Time frame considered was from May 25, 2012 to March 30, 2015. Data up to December 31, 2014 was used for training and the last 3 months data for testing.

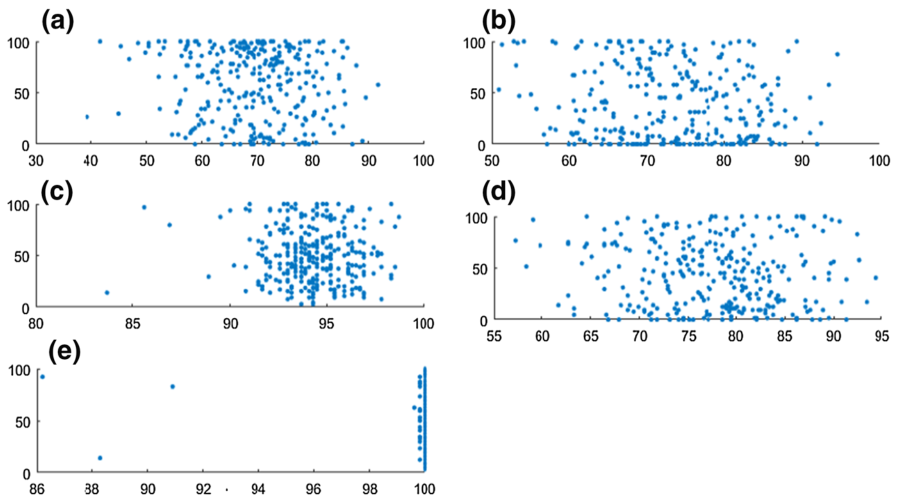

Fig. 2 Plots of training accuracy vs testing accuracy for standalone classifier based recommenders trained using only daily data: **a** NB, **b** SVM, **c** Tree, **d** discriminant and (e) kNN. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

The effectiveness of each classifier was evaluated as the first step. It was observed that decision tree classifiers were able to outperform all other single classifiers. It was also observed that incorporating technical indicators into the input feature set improved the accuracy of the classification process. Classification accuracies for each of the 293 stocks for each of the three different input feature sets is presented in "Appendix" 1, 2, and 3, respectively. Figure 2 presents the scatter plots of the testing accuracy vs the training accuracy (in percentages) for all stocks considered and each of the single classifiers considered in the present study trained using only the daily stock data. It must be noted that each point on the plot represents the testing vs training accuracy of one stock.

It can be observed from Fig. 2 (as well as from "Appendix 1"), that apart from decision tree based and kNN based recommenders, the training accuracies of other three classifiers varied widely from stock to stock. It was also observed that in case of kNN classifier based recommender systems (Fig. 2e), while the training accuracy was close to 100% for most of the stocks, the forecasting accuracy for the testing dataset was poor. This could be attributed to the fact that by its very nature, there is no real 'learning' taking place in case of kNN classifier. Decision tree classifier based recommender offered relatively better performance for the testing data, however it can be seen from Fig. 2c that while the training accuracies were between 90% -98% for most of the stocks, accuracy for the test data showed wide variation with the average accuracy of around 50%.

Figure 3 shows the scatter plots of the testing accuracy vs the training accuracy (in percentages) for all 293 stocks considered and each of the single classifiers considered in the present study trained using the daily stock data and technical indicators with the naïve assumption of setting all the technical indicator window sizes to their minimum (i.e. 2 days). Details for each stock has been presented in "Appendix 2".

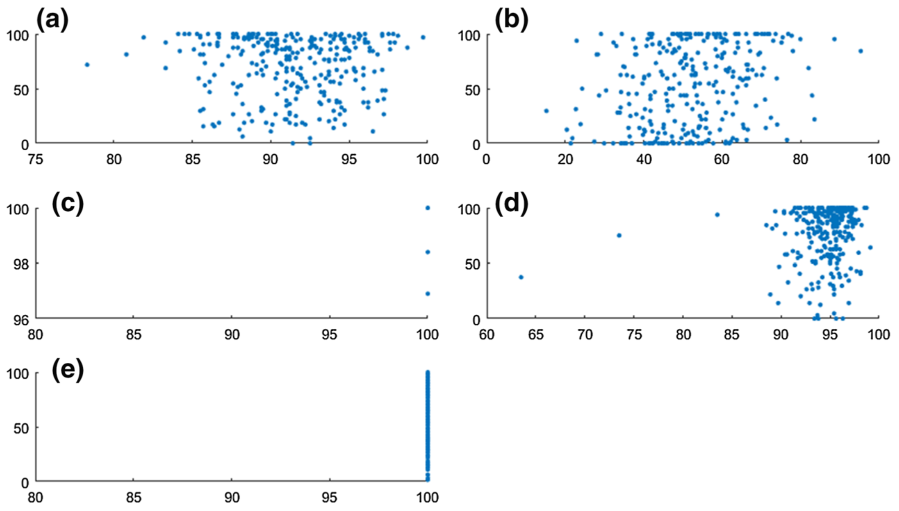

Fig. 3 Plots of training accuracy vs testing accuracy for standalone classifier based recommenders trained using daily data-naïve TI for: **a** NB, **b** SVM, **c** Tree, **d** Discriminant, **e** kNN. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

It can be seen from Fig. 3 (and "Appendix 2") that the performance of single classifiers shows significant improvement with the addition of technical indicators as additional features for all except SVM and kNN based recommenders. The training accuracies for naïve Bayes based recommenders (see Fig. 3a), improved, on an average from around 70% (when trained using daily data alone) to around 90%, with the test data accuracies also demonstrating a marked improvement. The improvement was most marked for decision tree based recommender system with training accuracy of around 100% and testing accuracy between 96–100% for all 293 stocks. Discriminant classifier based recommenders, as can be observed from Figs. 2d, 3d, also showed a marked improvement in accuracy. It must be noted that in Fig. 3c only three distinct data points are visible which is due to the fact that for of the 293 stocks, the test data accuracies were found to be 98.4% for 12 stocks, 96.9% for one stock (MARUTI) and 100% for the rest of the stocks.

Performance of single classifiers when the technical indicator window sizes are tuned using GA, for all the 293 stocks, is presented in Fig. 4. Performance figures for each stock has been listed in "Appendix 3".

It is observed that optimization of the window lengths of TIs using GA did not offer any discernable improvement in accuracy for any of the single classifiers considered.

As the next step, the accuracy of ensemble classifier based recommender systems was evaluated. Adaboost, Robustboost and bagging ensemble classifier based recommender systems were evaluated. Two weak learners namely, discriminant and tree were considered. Discriminant weak learner based ensemble classifier recommenders were evaluated first, followed by tree weak learner based ensemble classifier recommenders. Figure 5 presents the scatter plots of testing vs training accuracy for Adaboost

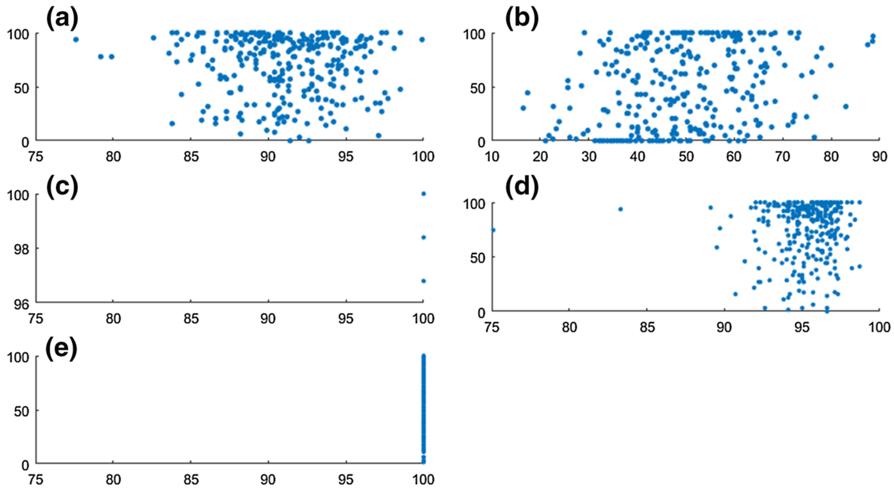

Fig. 4 Plots of training accuracy vs testing accuracy for standalone classifier based recommenders trained using daily data-GA-TI for: **a** NB, **b** SVM, **c** Tree, **d** Disc, **e** kNN. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

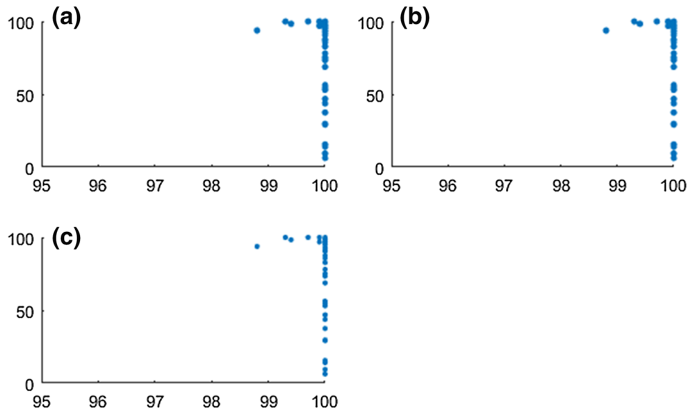

Fig. 5 Plots of training accuracy vs testing accuracy for Adaboost ensemble classifier based recommenders employing discriminant as the weak learner trained using: **a** daily data, **b** daily data-naïve TI and **c** daily data-GA-TI. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

classifier based recommender systems. Discriminant was used as the weak learner in this case. Accuracy values for each of the 293 stocks is presented in "Appendix 4".

It is observed that for 198 of the 293 stocks, the test accuracy was 100%, 72 stocks generated test accuracy between 90.1–99.9% and 7 stocks, between 80–90%. The results remained consistently same for all three different input feature set combinations.

Training vs testing accuracy scatter plots for bagging classifier based recommender systems is presented in Fig. 6. Accuracies for each stock can be obtained from "Appendix 5".

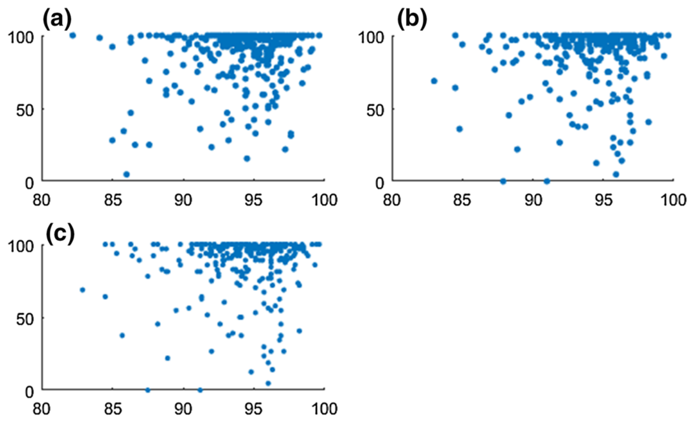

Fig. 6 Plots of training accuracy vs testing accuracy for bagging classifier based recommenders employing discriminant as the weak learner trained using: **a** daily data, **b** daily data-naïve TI and **c** daily data-GA-TI. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

It is observed that bagging classifier based recommenders using discriminant as weak learner were able to generate testing accuracy in excess of 60% for 261 of the total 293 stocks when trained using the daily stock data alone. This dropped by one to a total of 260 stocks for the other two feature sets. However, while only 219 stocks were able to generate accuracies in the range 80–100% for bagging-discriminant recommenders trained using daily data alone, this number rose up to 233 and 234 stocks for bagging-discriminant recommenders trained using daily data-GA-TI input feature set and daily data-naïve TI feature set, respectively. Only 85,86 and 83 stocks reported test accuracy of 100% for daily data, daily data-naïve TI and daily data-GA-TI trained bagging-discriminant classifiers. Accuracies for training data were also found to be poorer when compared to Adaboost-discriminant based recommenders with training accuracies of only 3,4 and 4 stocks corresponding to daily data alone, daily data-naïve TI and daily data-GA-TI input feature sets being above 99%. For 259, 262 and 263 stocks respectively, the training accuracies were in the range 90.1–99%.

Accuracies for Robustboost ensemble classifier based recommender systems for all the stocks considered in the present study is presented in Fig. 7. Discriminant was used as the weak learner. Performance details for each of the 293 stocks can be found in "Appendix 6".

It is observed that 134 of the 293 stocks considered, show an accuracy of 100% for the test data when Robustboost ensemble classifier based recommenders employing discriminant as the weak learner are trained using daily data alone (Fig. 7a). This number slightly to 138 stocks each for the other two input feature sets. Test data classification accuracies were between 90.1 and 99.9% for 105 stocks when Robustboost-discriminant classifier based recommenders are trained using daily data, while for the other two input feature set combinations, this improved slightly to 110 stocks in both cases.

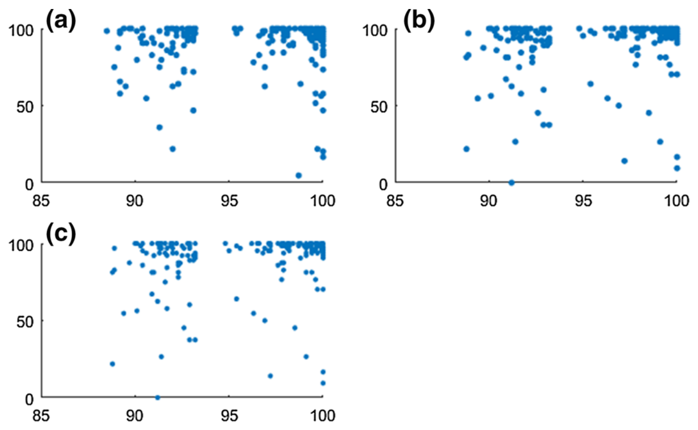

Fig. 7 Plots of training accuracy vs testing accuracy for Robustboost ensemble classifier based recommenders employing discriminant as the weak learner trained using: **a** daily data, **b** daily data-naïve TI and **c** daily data-GA-TI. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

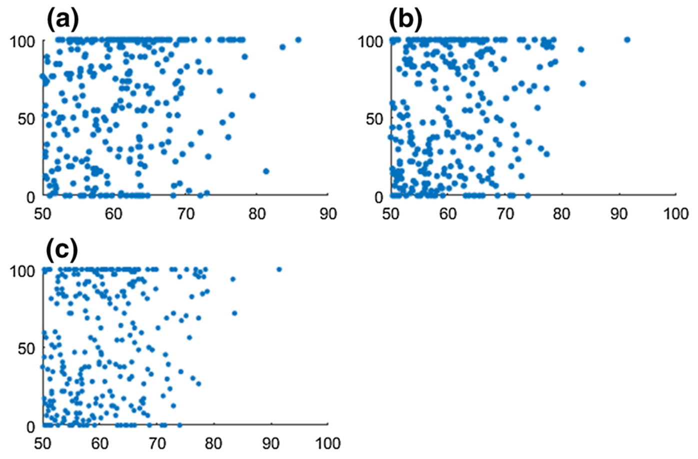

Fig. 8 Plots of training accuracy vs testing accuracy for Adaboost ensemble classifier based recommenders employing tree as the weak learner trained using: **a** daily data, **b** daily data-naïve TI and **c** daily data-GA-TI. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

Figure 8 presents the scatter plots of testing vs training accuracy for Adaboost classifier based recommender systems. Tree was used as the weak learner in this case. Accuracy values for each of the 293 stocks is presented in "Appendix 7".

Compared to any of the discriminant weak learner classifier based recommenders, Adaboost ensemble classifiers with tree as weak learner performed poorly. When trained using only daily stock data, the recommenders could not correctly classify even a single test data point for 33 of 293 stocks considered. This further worsened for the other two input feature sets to 44 stocks. 100% test data classification accuracies were obtained for only 64 of the 293 stocks when trained using only the daily data while this worsened to 55 and 55 stocks each when trained using daily data-naïve

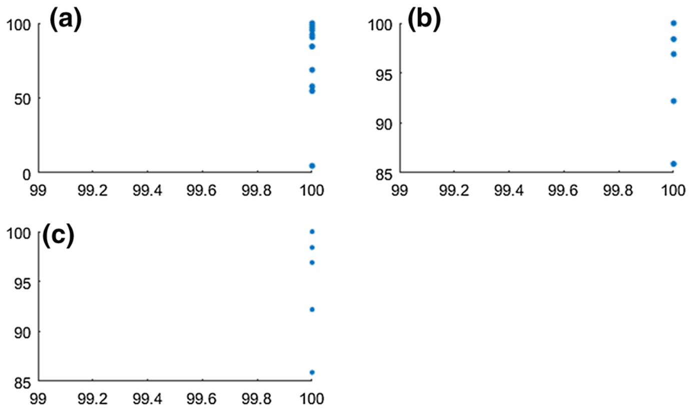

Fig. 9 Plots of training accuracy vs testing accuracy for bagging classifier based recommenders employing discriminant as the weak learner trained using: **a** daily data **b** daily data-naïve TI and **c** daily data-GA-TI. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

TI and daily data-GA-TI feature sets. This could be attributed to very poor training results since only three stocks demonstrated training accuracy in excess of 80% for all the three input feature sets.

Accuracies for bagging ensemble classifier based recommender systems for all the stocks considered in the present study is presented in Fig. 9. Tree was used as the weak learner. Performance details for each of the 293 stocks can be found in "Appendix 8".

It is observed that for bagging classifier based recommenders employing discriminant as the weak learner, the training accuracy for all the 293 stocks for all three input feature sets was uniformly 100%. For testing data however, 264 stocks showed an accuracy of 100% when trained using daily data alone. Training using daily data-naïve TI feature set resulted in 270 stocks demonstrating a testing accuracy of 100%. This slightly improved to 272 stocks with 100% accuracy, when the recommenders were trained using training data-GA-TI input feature set.

Accuracies for Robustboost ensemble classifier based recommender systems using tree as the weak learner for all the stocks considered in the present study is presented in Fig. 10. Performance details for each of the 293 stocks can be found in "Appendix 9".

It is observed from the results that the training accuracy for Robustboost ensemble classifier based recommenders employing tree as the weak learner for all the 293 stocks for all three input feature sets was uniformly 100%, similar to the results obtained for bagging ensemble classifier based recommender using tree as weak learner. Testing performance was, however, slightly better compared to bagging-tree ensemble based recommenders with 277 of 293 stocks classifying 100% of the test data correctly when trained using daily data alone. This further improved to 280 stocks that showed 100% test accuracy when the input feature set employed was daily data-naïve TI as well as when daily data-GA-TI was employed. Moreover, considering all the stocks and all the three different input feature sets, the worst test accuracy observed was

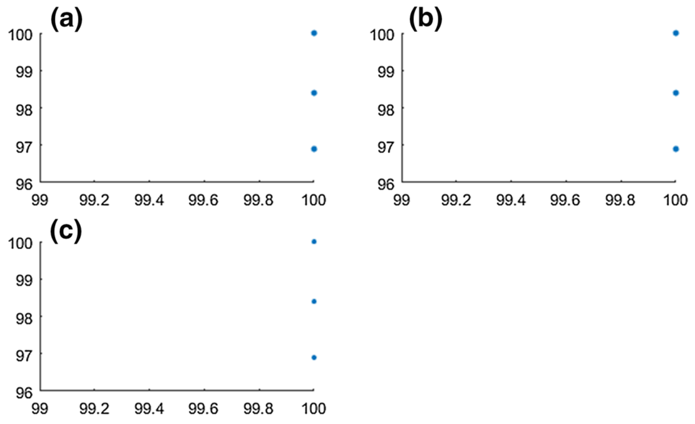

Fig. 10 Plots of training accuracy vs testing accuracy for Robustboost ensemble classifier based recommenders employing tree as the weak learner trained using: **a** daily data, **b** daily data-naïve TI and **c** daily data-GA-TI. Training accuracy (%) is on x axis and testing accuracy (%) is on the y axis

96.9%, while 290, 292 and 292 stocks showed test accuracy in excess of 98% when trained using daily data, daily data-naïve TI and daily data-GA-TI, respectively. Of all the recommender systems considered in the present study, Robustboost ensemble classifier based recommenders with tree as weak learner and trained using either daily data-naïve TI or daily data-GA-TI were seen to demonstrate highest accuracy, followed very closely by its third variant, trained using only the daily data.

As the next step, the economic performance of each type of recommender system is evaluated. A summary of the average profits (AP) generated for different classifier based recommender systems is presented in Tables 2 and 3. Both the tables present the number of stocks for which the AP was less than zero rupees (< 0), equal to zero rupees (0), between zero and hundred rupees (0–100) and greater than two hundred rupees (> 200). It must be noted that AP can become equal to zero under two conditions, namely, the resultant sum of all the profits and losses (considering transaction cost of 0.5% always) should sum to exactly zero or there should be no trades (i.e. TT for the stock = 0) at all. The first case was encountered rarely while the second case was observed much more frequently. A summary analysis of the number of trades is presented in Table 5.

From Table 2, it is seen that of all the single classifier based recommenders considered, Tree based recommenders trained using daily data—naïve TI and daily data-GA-TI, yield the smallest number of stocks with average negative returns. It is also seen that SVM based recommenders, in more than 40% of the cases for all the three input feature sets, recommend ‘Hold’ for the entire time frame considered. kNN classifier based recommenders trained using daily data, were also seen to be effective, however, the number of stocks for which the average returns turned out to be negative were three times more than that for the tree based recommenders discussed above.

From Table 3, it can be observed that Robustboot ensemble classifier based recommenders result in minimal number of stocks with negative average returns for all the three input feature set combinations.

Table 2 Number of stocks with average profits in five ranges, for single classifiers considered

Input feature set	Recommender	No. of stocks with avg. profit range (in Rs.)				
		<0	0	0–100	100–200	>200
Daily data	NB	15	58	188	19	13
	SVM	13	127	131	14	8
	Tree	20	12	224	21	16
	Discriminant	15	75	171	21	11
	kNN	19	10	228	21	15
Daily data-naïve TI	NB	107	49	121	9	7
	SVM	96	137	49	5	6
	Tree	5	77	177	18	16
	Discriminant	54	75	143	9	12
	kNN	106	72	98	9	8
Daily data-GA-TI	NB	108	51	116	10	8
	SVM	97	143	40	5	8
	Tree	5	79	175	18	16
	Discriminant	62	69	141	9	12
	kNN	102	75	99	9	8

Table 3 Number of stocks with average profits in five ranges, for ensemble classifiers considered

Input feature set	Recommender	Learner	No. of stocks with avg. profit range (in Rs.)				
			<0	0	0–100	100–200	>200
Daily data	Adaboost	Discriminant	11	76	173	19	14
	Adaboost	Tree	49	151	74	8	11
	Bag	Discriminant	22	76	155	21	19
	Bag	Tree	6	70	175	22	20
	Robustboost	Discriminant	15	64	173	20	21
	Robustboost	Tree	5	69	177	22	20
Daily data-naïve TI	Adaboost	Discriminant	11	76	173	19	14
	Adaboost	Tree	83	145	55	4	6
	Bag	Discriminant	24	81	157	20	11
	Bag	Tree	6	77	176	18	16
	Robustboost	Discriminant	17	74	168	21	13
	Robustboost	Tree	5	77	177	18	16
Daily data-GA-TI	Adaboost	Discriminant	11	76	173	19	14
	Adaboost	Tree	83	145	55	4	6
	Bag	Discriminant	26	80	156	20	11
	Bag	Tree	5	78	176	18	16
	Robustboost	Discriminant	17	74	168	21	13
	Robustboost	Tree	5	77	177	18	16

Table 4 Average MD for recommender systems considered

Type	Recommenders	Learners	Average MD (in Rs.)		
			Daily data	Daily data-naïve TI	Daily data-GA-TI
Single classifiers	NB		– 7.86	– 81.80	– 81.36
	SVM		– 8.98	– 61.15	– 54.69
	Tree		– 9.15	– 0.59	– 0.59
	Discriminant		– 12.33	– 56.51	– 57.39
	kNN		– 9.83	– 77.82	– 77.30
Ensemble classifiers	Adaboost	Discriminant	– 20.87	– 20.87	– 20.87
	Adaboost	Tree	– 45.06	– 74.35	– 74.35
	Bag	Discriminant	– 24.52	– 46.36	– 45.99
	Bag	Tree	3.52	– 0.35	– 0.29
	Robustboost	Discriminant	– 17.50	– 22.05	– 22.05
	Robustboost	Tree	3.92	– 0.29	– 0.29

Maximum drawdown (MD) is another important parameter that determines the recommender performance. Ideally a recommender should be able to prevent large negative returns from trades. MD is a measure of the worst loss from among all the trades carried out. In the present study, if there are no loss-making trades reported for a stock, the lowest profit generated is taken as the MD. Table 4 presents the average MD per stock for proposed recommender systems. Average MD for each recommender is calculated as:

$$AvgMD_{Recommender} = \frac{1}{293}\sum_{i=1}^{293} MD(i)_{Recommender} \tag{3}$$

Where:

Recommender = Recommender considered.

$AvgMD_{Recommender}$ = Average MD for the recommender considered over all 293 stocks.

It can be seen from Table 4 that Robustboost ensemble classifier based recommenders employing tree as the weak learner tend to generate the best average MD when trained using daily data. Average MD values showed slight worsening for daily data-naïve TI and daily data-GA-TI based recommenders. Worst performance is demonstrated by NB based recommenders when trained using Daily data-naïve TI.

Total number of trades (TT) carried out during the time frame under consideration is a measure of how efficient the recommender is at identifying trading points. Ideally, a recommender should generate recommendations such that maximum profit can be realized using minimum number of trades. This will help the trader minimize the transaction costs that he/she might have to pay to the brokerage firm.

Table 5 presents the average TT per stock for proposed recommender systems. Average TT for each recommender is calculated as:

Table 5 Average TT for recommender systems considered

Type	Recommenders	Learners	Average TT		
			Daily data	Daily data-naïve TI	Daily data-GA-TI
Single classifiers	NB		36	27	27
	SVM		21	29	28
	Tree		26	17	16
	Discriminant		22	23	23
	kNN		26	23	22
Ensemble classifiers	Adaboost	Discriminant	17	17	17
	Adaboost	Tree	31	32	32
	Bag	Discriminant	28	20	20
	Bag	Tree	26	17	17
	Robustboost	Discriminant	26	18	18
	Robustboost	Tree	26	17	17

$$AvgTT_{Recommender} = \frac{1}{293}\sum_{i=1}^{293} TT(i)_{Recommender} \tag{4}$$

where:

$Recommender$ = Recommender considered.

$AvgTT_{Recommender}$ = Average TT for the recommender considered over all 293 stocks.

From Table 5, it is seen that incorporating technical indicators into the input feature set along with the daily data results in reduction in the average number of trades recommended by Bagging and Robustboost ensemble classifier based recommenders. This trend is also observed in the case of Tree and NB based recommenders and to a small extent, in kNN based recommenders, as well.

It must be noted that, in addition to the TT, the profit factor (PF) and the win-ratio (WR) must also be considered while evaluating any recommender system. As described in Sect. 2.4 above, PF is the ratio of the profit from profit-making trades and the loss from loss-making trades. Ideally, for a recommender, the number of loss-making trades should minimal and even if loss-making trades do exist, the total loss from those loss-making trades should be much lower when compared to the total profit from profit-making trades. Hence, PF should be as high as possible for a good recommender system.

Number of stocks falling within four broad ranges of PF for each of the recommender system considered, is presented in Table 6. It must be noted that the heading 'NA' in Table 6 refers to the number of stocks for which PF could not be calculated (due to reasons such as no trade being executed during the time frame under consideration). The heading 'Inf' in Table 6 refers to the number of stocks for which the PF was infinity (i.e. the loss from loss-making trades was zero, implying no loss-making trade in the time frame). The number of stocks, for which the loss from loss-making trades exceeds the profit from profit-making trades, and thus, have PF < 1 are listed under head (< 1). Number of stocks, for which the loss from loss-making trades is less than the profit from profit-making trades have PF > 1 are listed under head (> 1).

It can be seen from Table 6 that kNN based recommenders tend to generate only profit-making trades during the time frame considered, for 221 of the 293 stocks. This is closely followed by Robustboost and Bagging ensemble classifier based recommenders using tree as learner and trained using daily data. However, it must be noted that for kNN based recommenders, the number of stocks with PF < 1 was at least three times higher than the above two ensemble classifier based recommenders.

Number of stocks falling within four broad ranges of WR for each of the recommender system considered, is presented in Table 7. It must be noted that the heading ‘NA’ in Table 7 refers to the number of stocks for which WR could not be calculated (again, due to reasons such as no trade being executed during the time frame under consideration). The heading ‘Inf’ in Table 7 refers to the number of stocks for which the WR was infinity (i.e. the number of loss-making trades was zero). The number of stocks, for which the number of loss-making trades exceeds the number of profit-making trades, and thus, have WR < 1 are listed under head (< 1). Number of stocks, for which the number of loss-making trades is less than the number of profit-making trades have WR > 1 are listed under heading (> 1).

It is observed from Table 7 that that kNN based recommenders tend to outperform other recommenders while considering the WR. Similar to the situation observed in the case of PF, this is closely followed by Robustboost and Bagging ensemble classifier based recommenders using tree as learner and trained using daily data. However, in this case too, for kNN based recommenders, the number of stocks with WR < 1 around three times higher than the above two ensemble classifier based recommenders.

4 Conclusions

A comprehensive empirical study involving 293 Group-A stocks from the Indian stock exchange BSE for evaluating the effectiveness of classifier based recommender systems in recommending stock trading decisions for Indian stocks, was carried out. Five single classifier and six ensemble classifier based recommender systems were considered. The effectiveness of three different input feature sets on recommender system performance was also empirically evaluated. It was observed that Robustboost and Bagging ensemble classifier based recommender systems employing tree as the weak learner were able to generate high profits, while at the same time minimizing the number of loss-making trades. kNN classifier based recommenders performed the best among single classifiers considered, however, the number of stocks for which the recommendations ended up in loss, was much higher for kNN based recommenders when compared to the above two recommenders. Another interesting observation made was that the recommender systems showed their best performance in terms of PF and WR, which indicate the ability of the recommender to recommend profitable trades, when trained using daily stock price data. Based on the results, it can be said that

Table 6 Number of stocks within each PF range for recommender systems considered

Type	Recommenders	Learners	No. of stocks with PF range											
			Daily data				Daily data-naïve TI				Daily data-GA-TI			
			NA	< 1	> 1	Inf	NA	< 1	> 1	Inf	NA	< 1	> 1	Inf
Single classifiers	NB		58	15	62	158	48	108	87	50	50	109	89	45
	SVM		127	13	33	120	137	96	45	15	141	98	36	18
	Tree		12	20	52	209	77	5	5	206	79	5	5	204
	Discriminant		75	15	34	169	75	54	80	84	68	62	82	81
	kNN		10	19	43	221	71	106	68	48	71	104	69	49
Ensemble classifiers	Adaboost	Discriminant	76	11	18	188	76	11	18	188	76	11	18	188
	Adaboost	Tree	150	49	78	16	145	83	57	8	145	83	57	8
	Bag	Discriminant	76	22	55	140	81	24	52	136	80	25	52	136
	Bag	Tree	70	6	7	210	77	6	5	205	78	5	5	205
	Robustboost	Discriminant	64	15	35	179	74	17	24	178	74	17	24	178
	Robustboost	Tree	69	5	4	215	77	5	5	206	77	5	5	206

Table 7 Number of stocks within each WR range for recommender systems considered

Type	Recommenders	Learners	No. of stocks with WR range											
			Daily data				Daily data-naïve TI				Daily data-GA-TI			
			NA	< 1	> 1	Inf	NA	< 1	> 1	Inf	NA	< 1	> 1	Inf
Single classifiers	NB		58	16	61	158	48	111	84	50	50	108	90	45
	SVM		127	13	33	120	137	101	40	15	141	101	33	18
	Tree		12	18	54	209	77	5	5	206	79	5	5	204
	Discriminant		75	20	29	169	75	59	75	84	68	68	76	81
	kNN		10	16	46	221	71	106	68	48	71	107	66	49
Ensemble classifiers	Adaboost	Discriminant	76	11	18	188	76	11	18	188	76	11	18	188
	Adaboost	Tree	150	47	80	16	145	90	50	8	145	90	50	8
	Bag	Discriminant	76	22	55	140	81	24	52	136	80	90	49	8
	Bag	Tree	70	6	7	210	77	6	5	205	78	5	5	205
	Robustboost	Discriminant	64	15	35	179	74	17	24	178	74	17	24	178
	Robustboost	Tree	69	5	4	215	77	5	5	206	77	5	5	206

ensemble classifier based recommender systems employing tree as the weak learner and trained using the daily stock price data as the input feature set, can be successfully employed for trading in Indian stocks.

References

Abbasi, E., & Abouec, A. (2008). Stock price forecast by using neuro-fuzzy inference system. *Proceedings of World Academy of Science, Engineering and Technology, 36*(December), 320–323.

Atsalakis, G. S., & Valavanis, K. P. (2009a). Forecasting stock market short-term trends using a neuro-fuzzy based methodology. *Expert Systems with Applications, 36*(7), 10696–10707. https://doi.org/10.1016/j.eswa.2009.02.043.

Atsalakis, G. S., & Valavanis, K. P. (2009b). Surveying stock market forecasting techniques—Part II: Soft computing methods. *Expert Systems with Applications, 36*(3 PART 2), 5932–5941. https://doi.org/10.1016/j.eswa.2008.07.006.

Ballings, M., Van Den Poel, D., Hespeels, N., & Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications, 42*(20), 7046–7056. https://doi.org/10.1016/j.eswa.2015.05.013.

Banik, S., Chanchary, F. H., Rouf, R. A., & Khan, K. (2007). Modeling chaotic behavior of Dhaka Stock Market Index values using the neuro-fuzzy model. In *10th International conference on computer and information technology* (pp. 1–6). https://doi.org/10.1109/ICCITECHN.2007.4579362.

Brabazon, A., & O'Neill, M. (2008). Natural computing in computational finance: An introduction. *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-540-77477-8_1.

Breiman, L. (1994). Bagging predictors. *Technical Report*. https://doi.org/10.1007/BF00058655.

Caporale, G. M., Gil-Alana, L., Plastun, A., & Makarenko, I. (2016). Intraday anomalies and market efficiency: A trading robot analysis. *Computational Economics, 47*(2), 275–295. https://doi.org/10.1007/s10614-015-9484-9.

Chan, K. C. C., & Teong, F. K. T. F. K. (1995). Enhancing technical analysis in the forex market using neuralnetworks. In *Proceedings of ICNN'95—international conference on neural networks*, *2*. https://doi.org/10.1109/ICNN.1995.487561.

Cowles 3rd, A. (1933). Can stock market forecasters forecast? *Econometrica, 1*(3), 309–324. https://doi.org/10.2307/1907042.

Dai, W., Wu, J.-Y., & Lu, C.-J. (2012). Combining nonlinear independent component analysis and neural network for the prediction of Asian stock market indexes. *Expert Systems with Applications, 39*(4), 4444–4452. https://doi.org/10.1016/j.eswa.2011.09.145.

Dempster, M. A. H., Payne, T. W., Romahi, Y., & Thompson, G. W. P. (2001). Computational learning techniques for intraday FX trading using popular technical indicators. *IEEE Transactions on Neural Networks, 12*(4), 744–754. https://doi.org/10.1109/72.935088.

Eng, W. F. (1998). *Technical analysis of stocks, options and futures: Advanced trading systems and techniques*. Chicago: Irwin Professional Publishing.

Fama, E. F. (1969). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance, 25*(2), 383–417.

Fama, E. F. (1995). Random walks in stock market prices. *Financial Analysts Journal, 51*(1), 75–80. https://doi.org/10.2469/faj.v51.n1.1861.

Freund, Y. (2009). A more robust boosting algorithm. arXiv:0905.

Freund, Y., & Schapire, R. (1997). A desicion-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55,* 119–139. https://doi.org/10.1006/jcss.1997.1504.

Han, J., Kamber, M., & Pei, J. (2006). Data mining: concepts and techniques. San Francisco: Morgan Kauffman.

HDFC Securities Ltd. (2018). Brokerage Charges, Retrieved October 26, 2018, from https://www.hdfcsec.com/article/brokeragecharges-3044.

ICICI Securities Ltd. (2018), Pricing, Retrieved October 26, 2018, from https://www.icicidirect.com/idirectcontent/Home/Pricing.aspx.

Jobman, D. R. (Ed.). (1998). *Technical analysis for futures traders*. New Delhi: Vision Books.

Khashei, M., & Bijari, M. (2011). Exchange rate forecasting netter with hybrid artificial neural networks models. *Journal of Mathematical and Computational Science, 1*(1), 103–125.

Kuo, M.-H., & Chen, C.-L. (2006). An ETF trading decision support system by using neural network and technical indicators. In *The 2006 IEEE international joint conference on neural network proceedings* (pp. 2394–2401). https://doi.org/10.1109/IJCNN.2006.247064.

Lee, C. T., & Chen, Y. P. (2007). The efficacy of neural networks and simple technical indicators in predicting stock markets. In *International conference on convergence information technology, ICCIT 2007* (pp. 2292–2297). https://doi.org/10.1109/ICCIT.2007.4420595.

Luo, L., & Chen, X. (2013). Integrating piecewise linear representation and weighted support vector machine for stock trading signal prediction. *Applied Soft Computing Journal, 13*(2), 806–816. https://doi.org/10.1016/j.asoc.2012.10.026.

Nair, B. B., Dharini, N. M., & Mohandas, V. P. (2010). A stock market trend prediction system using a hybrid decision tree-neuro-fuzzy system. In *Proceedings—2nd international conference on advances in recent technologies in communication and computing, ARTCom 2010* (pp. 381–385). https://doi.org/10.1109/ARTCom.2010.75.

Nair, B. B., & Mohandas, V. P. (2015a). Artificial intelligence applications in financial forecasting—A survey and some empirical results. *Intelligent Decision Technologies, 9*(2), 99–140. https://doi.org/10.3233/IDT-140211.

Nair, B. B., & Mohandas, V. P. P. (2015b). An intelligent recommender system for stock trading. *Intelligent Decision Technologies, 9*(3), 243–269. https://doi.org/10.3233/IDT-140220.

Nair, B. B., Mohandas, V. P., & Sakthivel, N. R. (2011). Predicting stock market trends using hybrid ant-colony-based data mining algorithms: An empirical validation on the Bombay Stock Exchange. *International Journal of Business Intelligence and Data Mining, 6*(4), 362–381. https://doi.org/10.1504/IJBIDM.2011.044976.

Saad, E. W., Prokhorov, D. V., & Wunsch, D. C. (1998). Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Transactions on Neural Networks/A Publication of the IEEE Neural Networks Council, 9*(6), 1456–1470. https://doi.org/10.1109/72.728395.

Soman, K. P., Diwakar, S., & Ajay, V. (2010). *Insight into data mining-theory and practice* (1st ed.). New Delhi: Prentice Hall of India.

Teknomo, K. (2015). Discriminat analysis tutorial, Retrieved October 26, 2018, from https://people.revoledu.com/kardi/tutorial/LDA/.

Tufféry, S. (2011). *Data mining and statistics for decision making. Data mining and statistics for decision making*. West Sussex: Wiley. https://doi.org/10.1002/9780470979174.

Yahoo! Finance India. (2015).

Zarandi, M. H. F., Zarinbal, M., Ghanbari, N., & Turksen, I. B. (2013). A new fuzzy functions model tuned by hybridizing imperialist competitive algorithm and simulated annealing. Application: Stock price prediction. *Information Sciences, 222,* 213–228. https://doi.org/10.1016/j.ins.2012.08.002.