Chapter 3

Predicting Direction of Movement of Stock Price and Stock Market Index

This study addresses problem of predicting direction of movement of stock price and stock market index for Indian stock markets. The study compares four prediction models, Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF) and Naive Bayes (NB) with two approaches for input to these models. The first approach for input data involves computation of ten technical parameters using stock trading data (open, high, low & close prices) while the second approach focuses on representing these technical parameters as trend deterministic data. Accuracy of each of the prediction models for each of the two input approaches is evaluated. Evaluation is carried out on 10 years of historical data from 2003 to 2012 of two stocks namely Reliance Industries and Infosys Ltd. and two stock market indices CNX Nifty and S&P Bombay Stock Exchange (BSE) Sensex. Experimental results suggest that for the first approach of input data where ten technical parameters are represented as continuous values, Random Forest outperforms other three prediction models on overall performance. Experimental results also show that the performance of all the prediction models improve when these technical parameters are represented as trend deterministic data.

3.1 Introduction

Predicting stock and stock price index is difficult due to uncertainties involved. There are two types of analysis which investors perform before investing in a stock. First is the fundamental analysis. In this, investors look at intrinsic value of stocks, performance of the industry and economy, political climate etc. to decide whether to invest or not. On the other hand, technical analysis is the evaluation of stocks by means of studying statistics generated by market activity, such as past prices and volumes. Technical analysts do not attempt to measure a security's intrinsic value but instead use stock charts to identify patterns and trends that may suggest how a stock will behave in the future. Efficient market hypothesis states that prices of stocks are informationally efficient; which means that it is possible to predict stock prices based on the trading data (Malkiel and Fama). This is quite logical as many uncertain factors like political scenario of country, public image of the company, etc. will start reflecting in the stock prices. So, if the information obtained from stock prices is pre-processed efficiently and appropriate algorithms are applied, trend of stock or stock price index may be predicted.

Since years, many techniques have been developed to predict stock trends. Initially, classical regression methods were used to predict stock trends. Since stock data can be categorized as non-stationary time series data, non-linear machine learning techniques have also been used. ANN (Mehrotra, Mohan, and Ranka) and SVM (Vapnik) are two machine learning algorithms which are most widely used for predicting stock and stock price index movement. Each algorithm has its own way to learn patterns. ANN emulates functioning of a human brain to learn by creating network of neurons while SVM uses the spirit of Structural Risk Minimization (SRM) principle.

3.2 Related Work

(Hassan, Nath, and Kirley) proposed and implemented a fusion model by combining the Hidden Markov Model (HMM), ANN and Genetic Algorithms (GA) to forecast financial market behaviour. Using ANN, the daily stock prices are transformed to independent sets of values that become input to HMM. (Wang and Leu) developed a prediction system useful in forecasting mid-term price trend in Taiwan stock mar-

ket. Their system was based on a recurrent neural network trained by using features extracted from Autoregressive Integrated Moving Average (ARIMA) analysis. Empirical results showed that the network trained using 4-year weekly data was capable of predicting up to 6 weeks market trend with acceptable accuracy. Hybridized soft computing techniques for automated stock market forecasting and trend analysis was introduced in (Abraham, Nath, and Mahanti). They used Nasdaq-100 index of Nasdaq Stock Market with Neural Network for one day ahead stock forecasting and a neuro-fuzzy system for analysing the trend of the predicted stock values. The forecasting and trend prediction results using the proposed hybrid system were promising. (Chen, Leung, and Daouk) investigated the probabilistic neural network (PNN) to forecast the direction of index after it was trained by historical data. Empirical results showed that the PNN-based investment strategies obtained higher returns than other investment strategies. Other investment strategies that were examined include the buy-and-hold strategy as well as the investment strategies guided by forecasts estimated with the random walk model and the parametric GMM models.

A very well-known SVM algorithm developed by (Vapnik) searches for a hyper plane in higher dimension to separate classes. SVM is a very specific type of learning algorithm characterized by the capacity control of the decision function, the use of the kernel functions and the scarcity of the solution. (Huang, Nakamori, and Wang) investigated the predictability of SVM in forecasting the weekly movement direction of NIKKEI 225 index. They compared SVM with Linear Discriminant Analysis, Quadratic Discriminant Analysis and Elman Backpropagation Neural Networks. The experiment results showed that SVM outperformed the other classification methods. SVM was used in (Kim) to predict the direction of daily stock price change in the Korea Composite Stock Price Index (KOSPI). Twelve technical indicators were selected to make up the initial attributes. This study compared SVM with Back-propagation Neural Network (BPN) and Case-Based Reasoning (CBR). It was evident from the experimental results that SVM outperformed BPN and CBR.

Random Forest creates n classification trees using sample with replacement and predicts class based on what majority of trees predict. The trained ensemble, therefore, represents a single hypothesis. This hypothesis, however, is not necessarily contained within the hypothesis space of the models from which it is built. Thus,

ensembles can be shown to have more flexibility in the functions they can represent. This flexibility can, in theory, enable them to over-fit the training data more than a single model would, but in practice, some ensemble techniques (especially bagging) tend to reduce problems related to over-fitting of the training data. (Tsai et al.) investigated the prediction performance of the classifier based on ensemble method to analyse stock returns. The hybrid methods of majority voting and bagging were considered. Moreover, performance using two types of classifier ensembles were compared with those using single baseline classifiers (i.e. Neural Networks, Decision Trees, and Logistic Regression). The results indicated that multiple classifiers outperform single classifiers in terms of prediction accuracy and returns on investment. (Sun and Li) proposed new financial distress prediction (FDP) method based on SVM ensemble. The algorithm for selecting SVM ensemble's base classifiers from candidate ones was designed by considering both individual performance and diversity analysis. Experimental results indicated that SVM ensemble was significantly superior to individual SVM classifier. (Ou and Wang) used total ten data mining techniques to predict price movement of Hang Seng index of Hong Kong stock market. The approaches included Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbor (KNN) Classification, Naive Bayes based on Kernel Estimation, Logit Model, Tree based Classification, Neural network, Bayesian Classification with Gaussian Process, SVM and Least Squares - Support Vector Machine (LS-SVM). Experimental results showed that the SVM and LS-SVM generated superior predictive performance among the other models.

It is evident from the above discussions that each of the algorithms in its own way can tackle this problem. It is also to be noticed that each of the algorithm has its own limitations. The final prediction outcome not only depends on the prediction algorithm used, but is also influenced by the representation of the input. Identifying important features and using only them as the input rather than all the features may improve the prediction accuracy of the prediction models. A two-stage architecture was developed in (Hsu et al.). They integrated Self-Organizing Map (SOM) and Support Vector Regression (SVR) for stock price prediction. They examined seven major stock market indices. Specifically, the self-organizing map was first used to decompose the whole input space into regions where data points with similar statistical

distributions were grouped together, so as to contain and capture the non-stationary property of financial series. After decomposing heterogeneous data points into several homogenous regions, SVR was applied to forecast financial indices. The results suggested that the two stage architecture provided a promising alternative for stock price prediction. Genetic Programming (GP) and its variants have been extensively applied for modelling of the stock markets. To improve the generalization ability of the model, GP have been hybridized with its own variants (Gene Expression Programming (GEP), Multi Expression Programming (MEP)) or with the other methods such as Neural Networks and boosting.

The generalization ability of the GP model can also be improved by an appropriate choice of model selection criterion. (Garg, Sriram, and Tai) worked to analyse the effect of three model selection criteria across two data transformations on the performance of GP while modelling the stock indexed in the New York Stock Exchange (NYSE). Final Prediction Error (FPE) criteria showed a better fit for the GP model on both data transformations as compared to other model selection criteria. (Nair et al.) predicted the next day's closing value of five international stock indices using an adaptive ANN based system. The system adapted itself to the changing market dynamics with the help of Genetic Algorithm which tuned the parameters of the Neural Network at the end of each trading session.

The study in (Ahmed) investigated the nature of the causal relationships between stock prices and the key macro-economic variables representing real and financial sector of the Indian economy for the period March, 1995 to March, 2007 using quarterly data. The study revealed that the movement of stock prices was not solely dependent on behaviour of key macro-economic variables. (Mantri, Gahan, and Nayak) estimated the volatilities of Indian stock markets using Generalized Autoregressive Conditional Heteroskedasticity (GARCH), Exponential GARCH (EGARCH), Glosten-Jagannathan-Runkle GARCH (GJR-GARCH), Integrated GARCH (IGARCH) & ANN models. This study used fourteen years of data of BSE Sensex & NSE Nifty to estimate the volatilities. It was concluded that there was no difference in the volatilities of Sensex & Nifty estimated under the GARCH, EGARCH, GJR GARCH, IGARCH & ANN models.

(Mishra, Sehgal, and Bhanumurthy) tested for the presence of nonlinear de-

pendence and deterministic chaos in the rate of returns series for six Indian stock market indices. The result of analysis suggested that the returns series did not follow a random walk process. Rather it appeared that the daily increments in stock returns were serially correlated and the estimated Hurst exponents were indicative of marginal persistence in equity returns. (Liu and Wang) investigated and forecast the price fluctuation by an improved Legendre Neural Network by assuming that the investors decided their investing positions by analysing the historical data on the stock market. They also introduced a random time strength function in the forecasting model. The Morphological Rank Linear Forecasting (MRLF) method was proposed by (Araújo and Ferreira). An experimental analysis was conducted and the results were compared to Multilayer Perceptron (MLP) networks and Time-delay Added Evolutionary Forecasting (TAEF) method.

This study focuses on comparing prediction performance of ANN, SVM, Random Forest and naive Bayes algorithms for the task of predicting stock and stock price index movement. Ten technical parameters are used as the inputs to these models. A Trend Deterministic Data Preparation Layer which converts continuous-valued inputs to discrete ones is proposed. Each input parameters in its discrete form indicates a possible up or down trend determined based on its inherent property. The focus is also to compare the performance of these prediction models, when the inputs are represented in the form of real values and trend deterministic data. All the experiments are carried out using 10 years of historical data of two stocks - Reliance Industries and Infosys Ltd. and two indices S&P BSE Sensex and CNX Nifty. Both stocks and indices are highly voluminous and vehemently traded in and so they reflect Indian Economy as a whole.

3.3 Research Data

Ten years of data of total two stock market indices (CNX Nifty and S&P BSE Sensex) and two stocks (Reliance Industries & Infosys Ltd.) from Jan 2003 to Dec 2012 is used in this study. All the data is obtained from http://www.nseindia.com/ and http://www.bseindia.com/ websites. These data form our entire data set. Percentage wise increase and decrease cases (days) of each year in the entire data set are shown in Table 3.1.

Table 3.1: Number of increase and decrease cases (days) percentage in each year in the entire data set of S&P BSE Sensex

Year	**Increase**	**Increase (%)**	**Decrease**	**Decrease (%)**	**Total**
2003	146	58.63	103	41.37	249
2004	136	54.18	115	45.82	251
2005	147	59.04	102	40.96	249
2006	148	59.92	99	40.08	247
2007	139	55.82	110	44.18	249
2008	114	46.72	130	53.28	244
2009	127	52.70	114	47.30	241
2010	134	53.39	117	46.61	251
2011	116	47.15	130	52.85	246
2012	128	51.82	119	48.18	247
Total	1335	53.94	1139	46.06	2474

This study uses 20% of the entire data as the parameter selection data. This data is used to determine design parameters of predictor models. Parameter selection data set is constructed by taking equal proportion of data from every year of the ten years. The proportion of percentage wise increase and decrease cases in each year is also maintained. This sampling method enables parameter setting data set to be better representative of the entire data set. The parameter selection data is further divided into training and hold-out set. Each of the set consists of 10% of the entire data. Table 3.2 depicts the number of increase and decrease cases (days) for parameter selection data set. These statistics are for S&P BSE Sensex. Similar data analysis is done for CNX Nifty, for Reliance Industries and Infosys Ltd.

Optimum parameters for predictor models are obtained by means of experiments on parameter selection data. After that, for comparing ANN, SVM, Random Forest and Naive Bayes, comparison data set is devised. This data set comprises of entire

ten years of data. It is also divided in training (50% of the entire data) and hold-out (50% of the entire data) set. Details of this data set of S&P BSE Sensex is shown in Table 3.3. These experimental settings are same as in (Kara, Acar Boyacioglu, and Baykan).

Table 3.2: Number of increase and decrease cases (days) in each year in the parameter setting data set of S&P BSE Sensex

Year	**Training (Days)**			**Holdout (Days)**		
	Increase	**Decrease**	**Total**	**Increase**	**Decrease**	**Total**
2003	15	10	25	15	10	25
2004	14	11	25	14	11	25
2005	15	10	25	15	10	25
2006	15	10	25	15	10	25
2007	14	11	25	14	11	25
2008	11	13	24	11	13	24
2009	13	11	24	13	11	24
2010	13	12	25	13	12	25
2011	12	13	25	12	13	25
2012	13	12	25	13	12	25
Total	135	113	248	135	113	248

There are some technical indicators through which one can predict the future movement of stocks. Here in this study, total ten technical indicators as employed in (Kara, Acar Boyacioglu, and Baykan) are used. These indicators are shown in Table 3.4. Two approaches for the representation of the input data are employed in this study. The first approach uses continuous-valued representation, i.e., the actual time series, while the second one uses trend deterministic representation (which is discrete in nature) for the inputs. Both the representations are discussed here.

Table 3.3: Number of increase and decrease cases (days) in each year in the comparison data set of S&P BSE Sensex

Year	Training (Days)			Holdout (Days)		
	Increase	**Decrease**	**Total**	**Increase**	**Decrease**	**Total**
2003	73	52	125	72	52	124
2004	68	58	126	67	58	125
2005	74	51	125	73	51	124
2006	74	50	124	73	50	123
2007	70	55	125	69	55	124
2008	57	65	122	57	65	122
2009	64	57	121	63	57	120
2010	67	59	126	66	59	125
2011	58	65	123	58	65	123
2012	64	60	124	63	60	123
Total	669	572	1241	661	572	1233

3.3.1 Continuous Representation - The Actual Time Series

Ten technical indicators calculated based on the formula as shown in Table 3.4 are given as inputs to predictor models. It is evident that each of the technical indicators calculated based on the above mentioned formula is continuous-valued. The values of all technical indicators are normalized in the range between [-1, +1], so that larger value of one indicator do not overwhelm the smaller valued indicator. Performance of all the models under study is evaluated for this representation of inputs.

3.3.2 Discrete Representation - Trend Deterministic Data

A new layer of decision is employed, which converts continuous-valued technical parameters to discrete value, representing the trend. This layer, proposed in this study, is thereby named as the "Trend Deterministic Data Preparation Layer".

Table 3.4: Selected technical indicators & their formula (Kara, Acar Boyacioglu, and Baykan)

Name of Indicators	**Formula**
Simple n(10 *here*)-day Moving Average (SMA)	$\frac{C_t+C_{t-1}+\cdots+C_{t-9}}{n}$
n(10 *here*)-day Exponential Moving Average (EMA)	$EMA(k)_t = EMA(k)_{t-1} + \alpha \times (C_t - EMA(k)_{t-1})$
Momentum (MOM)	$\mathrm{C}_t - C_{t-9}$
Stochastic K% (STCK%)	$\frac{C_t-LL}{HH-LL} \times 100$
Stochastic D% (STCD%)	$\frac{\sum_{i=0}^{n-1} K_{t-i}}{10}\%$
Relative Strength Index (RSI)	$100 - \frac{100}{1+(\sum_{i=0}^{n-1} UP_{t-i}/n)/(\sum_{i=0}^{n-1} DW_{t-i}/n)}$
Moving Average Convergence Divergence (MACD)	$MACD(n)_{t-1} + \frac{2}{n+1} \times (DIFF_t - MACD(n)_{t-1})$
Larry William's R% (LWR)	$\frac{HH-C_t}{HH-LL} \times -100$
Accumulation/Distribution (A/D) Oscillator (ADO)	$\frac{[(H_t-O_t)+(C_t-L_t)]}{[2\times(H_t-L_t)]} \times 100$
Commodity Channel Index (CCI)	$\frac{M_t-SM_t}{0.015D_t}$

Here, C_t is the closing price, O_t is the opening price, L_t is the low price and H_t the high price of t^{th} day, $DIFF_t = EMA(12)_t - EMA(26)_t$, EMA is Exponential Moving Average, $EMA(k)_t = EMA(k)_{t-1} + \alpha \times (C_t - EMA(k)_{t-1})$, α is a smoothing factor which is equal to $\frac{2}{k+1}$, k is the time period of k-day Exponential Moving Average, $EMA(k)$ is the k-day moving average, LL and HH imply the lowest low and the highest high during the look back period (10 days here), respectively. $M_t = \frac{H_t+L_t+C_t}{3}$, $SM_t = \frac{(\sum_{i=1}^{n} M_{t-i+1})}{n}$, $D_t = \frac{\sum_{i=1}^{n} |M_{t-i+1} - SM_t|)}{n}$, UP_t means upward price change while DW_t is the downward price change at time t.

Each technical indicator has its own inherent property through which traders generally predict the stock's up or down movement. The job of this new layer is to convert this continuous values to '+1' or '-1' by considering this property during the discretization process. This way, the input data to each of the predictor models is converted to '+1' and '-1', where '+1' indicates up movement and '-1' shows down movement.

Simple Moving Average (SMA) and Exponential Moving Average (EMA) help smooth price action and filter out the noise. SMA and EMA of 10-days will hug prices quite closely and turn shortly after prices turn. So when SMA and EMA at time t is greater than at time $t-1$, it suggests up movement for stock i.e. '+1' and vice-a-versa for down movement i.e. '-1'.

Stochastic K% (STCK%), Stochastic D% (STCD%) and Larry Williams R% are stochastic oscillators. These oscillators are clear trend indicators for any stock. When stochastic oscillators are increasing, the stock prices are likely to go up and vice-a-versa (Kim). MACD, RSI, CCI and A/D oscillators also follow the stock trend. Using these indicator values, the trend deterministic input set is prepared and given to the predictor models. Performance of all the models under study is evaluated, for this representation of inputs also.

3.4 Prediction Models

3.4.1 ANN Model

Inspired by functioning of biological neural networks, ANN are a dense network of inter-connected neurons which get activated based on inputs. A three layer feed-

forward neural network is employed in this study (Mehrotra, Mohan, and Ranka Han, Kamber, and Pei). Inputs for the network are ten technical indicators which are represented by ten neurons in the input layer. Output layer has a single neuron with log-sigmoid as the transfer function. This results in a continuous value output between 0 and 1. A threshold of 0.5 is used to determine the up or down movement prediction. For the output value greater than or equal to 0.5, prediction is considered to be the up movement, else the down movement. Each of the hidden layer's neurons employed tan-sigmoid as the transfer function. The architecture of the three-layered feed-forward ANN is illustrated in Figure 3.1.

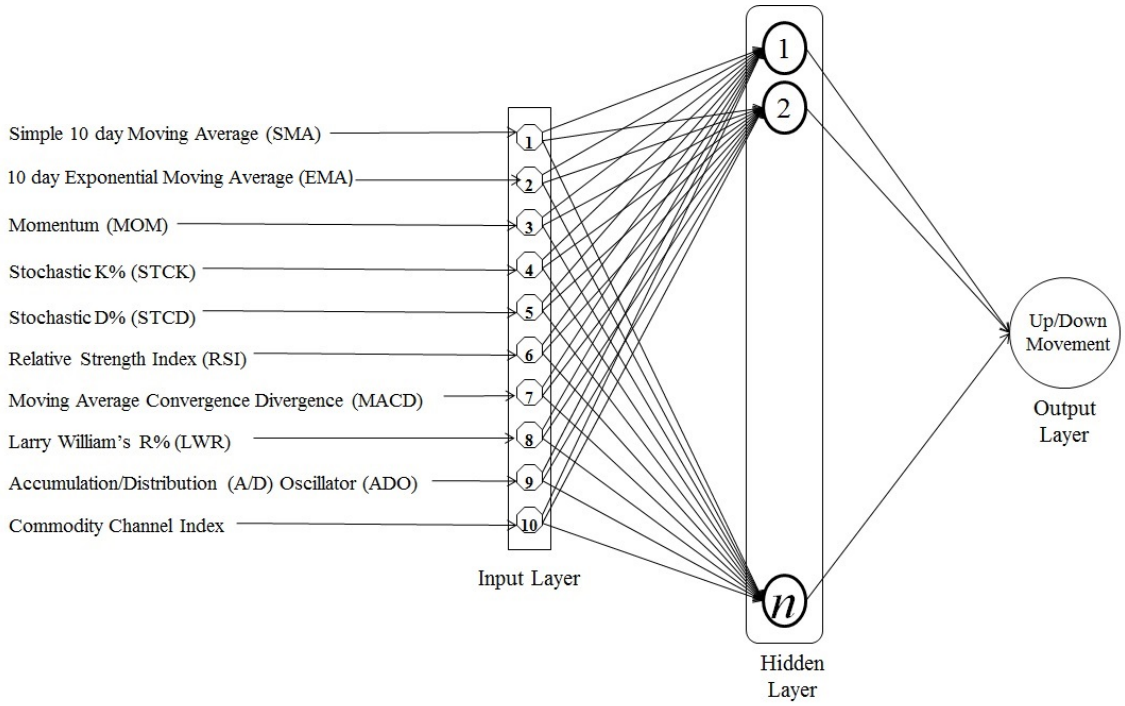

Figure 3.1: Architecture of ANN model (Kara, Acar Boyacioglu, and Baykan)

Gradient descent with momentum is used to adjust the weights, in which, at each epochs, weights are adjusted, so that a global minimum can be reached. Comprehensive parameter setting experiments to determined parameters for each stocks and indices are performed in this study. The ANN model parameters are number of hidden layer neurons (n), value of learning rate (lr), momentum constant (mc) and number of epochs (ep). To determine them efficiently, ten levels of n, nine levels of mc and ten levels of ep are tested in the parameter setting experiments. Initially, value of lr is fixed to 0.1. These parameters and their values which are tested are

summarized in Table 3.5.

These settings of parameters yield a total of $10 \times 10 \times 9 = 900$ treatments for ANN for one stock. Considering two indices and two stocks, total of 3600 treatments for ANN are carried out. The top three parameter combinations that resulted in the best average of training and holdout performance are selected as the top three ANN models for comparison experiments on comparison data set. For these top performing models, learning rate lr is varied in the interval of [0.1, 0.9].

Table 3.5: ANN parameters and their values tested in parameter setting experiments

Parameters	**Value(s)**
Number of Hidden Layer Neurons (n)	10,20,$\cdots$,100
Epochs (ep)	1000, 2000, $\cdots$,10000
Momentum Constant (mc)	0.1, 0.2, $\cdots$,0.9
Learning Rate (lr)	0.1

3.4.2 SVM Model

SVM was first introduced by (Vapnik). There are two main categories for SVMs: Support Vector Classification (SVC) and Support Vector Regression (SVR). SVM is a learning system using a high dimensional feature space. (Khemchandani, Chandra, et al.) stated that in SVM, points are classified by means of assigning them to one of the two disjoint half spaces, either in the pattern space or in a higher-dimensional feature space.

The main objective of SVM is to identify maximum margin hyper plane. The idea is that the margin of separation between positive and negative examples is maximized (Xu, Zhou, and Wang).

SVM finds maximum margin hyper plane as the final decision boundary. Assume that $x_i \in R^d, i = 1, 2, \cdots, N$ forms a set of input vectors with corresponding class labels $y_i \in \{+1, -1\}, i = 1, 2, \cdots, N$. SVM can map the input vectors $x_i \in R^d$ into a high dimensional feature space $\Phi(x_i) \in H$. A kernel function $K(x_i, x_j)$ performs the mapping $\phi(\cdot)$. The resulting decision boundary is defined in Equation 3.1.

$$f(x) = sgn(\sum_{i=1}^{N} y_i \alpha_i \cdot K(x, x_i) + b) \tag{3.1}$$

To get the values of α_i, involved in Equation 3.1, quadratic programming problem shown in Equation 3.2 is solved.

$$\begin{aligned} Maximize \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j \cdot y_i y_j \cdot K(x_i, x_j) \\ Subject\ to\ 0 \leq \alpha_i \leq c \\ \sum_{i=1}^{N} \alpha_i y_i = 0, i = 1, 2, \ldots, N \end{aligned} \tag{3.2}$$

The trade-off between margin and misclassification error is controlled by the regularization parameter c. The polynomial and radial basis kernel functions are used in this study and they are shown in Equations 3.3 and 3.4 respectively.

$$Polynomial \quad Function : K(x_i, x_j) = (x_i \cdot x_j + 1)^d \tag{3.3}$$

$$Radial \quad Basis \quad Function : K(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2) \tag{3.4}$$

where d is the degree of polynomial function and γ is the constant of radial basis function.

Choice of kernel function, degree of kernel function (d) in case of polynomial kernel, gamma in kernel function (γ) in case of radial basis kernel and regularization constant (c) are considered as the parameters of SVM in this study. To determine them efficiently, four levels on d, ten levels of γ and four to five levels of c are tested in the parameter setting experiments. These parameters and their values which are tested are summarized in Table 3.6.

For one stock, these settings of parameters yield a total of 20 and 40 treatments for SVM employing polynomial and radial basis kernel functions respectively. Considering two indices and two stocks, total of 240 treatments for SVM are carried out. One parameter combination for each of the polynomial kernel SVM and radial basis kernel SVM, that resulted in the best average of training and holdout performance is selected as the top SVM model for comparison experiments.

Table 3.6: SVM parameters and their values tested in parameter setting experiments

Parameters	**Values (polynomial)**	**Values (radial basis)**
Degree of Kernel Function (d)	$1, 2, 3, 4$	-
Gamma in Kernel Function (γ)	-	$0.5, 1.0, 1.5, \cdots, 5.0, 10.0$
Regularization Parameter (c)	0.5, 1, 5, 10, 100	0.5, 1, 5, 10

3.4.3 Random Forest

Decision tree learning is one of the most popular techniques for classification. Its classification accuracy is comparable with other classification methods, and it is very efficient. The classification model learnt through these techniques is represented as a tree and is known as a decision tree. ID3 (Quinlan, "Induction of decision trees"), C4.5 (Quinlan, *C4. 5: programs for machine learning*) and CART (Breiman et al.) are decision tree learning algorithms. Details can be found in (Han, Kamber, and Pei).

Random Forest belongs to the category of ensemble learning algorithms (Breiman). It uses decision tree as the base learner of the ensemble. The idea of ensemble learning is that a single classifier is not sufficient for determining class of test data. Reason being, based on sample data, classifier is not able to distinguish between noise and pattern. So, it performs sampling with replacement, such that, given n trees that are to be learnt, are based on these data set samples. In the experiments performed in this study, each tree is learnt using 3 features selected randomly. After creation of n trees, when testing data is used, the decision which majority of trees come up with is considered as the final output. This also avoids problem of over-fitting. Implementation of random forest algorithm in this study is summarized in the Algorithm 1.

Number of trees (*ntrees*) in the ensemble is considered as the parameter of Random Forest. To determine it efficiently, it is varied from 10 to 200 with increment of 10 each time during the parameter setting experiments. For one stock, these settings of parameter yield a total of 20 treatments. Considering two indices and two

stocks, total of 80 treatments are carried out. The top three parameter values that resulted in the best average of training and holdout performance are selected as the top three Random Forest models for the comparison experiments.

Algorithm 1 Random Forest

Input: training set D, number of trees in the ensemble k

Output: a composite model $M*$

1: **for** $i = 1$ to k **do**
2: Create bootstrap sample D_i by sampling D with replacement
3: Select 3 features randomly
4: Use D_i and randomly selected three features to derive tree M_i
5: **end for**
6: return $M*$

3.4.4 Naive Bayes Classifier

Naive Bayes classifier assumes class conditional independence (Han, Kamber, and Pei Markov and Larose). Given test data, Bayesian classifier predicts the probability of data belonging to a particular class. To predict probability it uses concept of Bayes' theorem. Bayes' theorem is useful in calculating the posterior probability, $P(C|X)$, from $P(C)$, $P(X|C)$, and $P(X)$. Bayes' theorem states that

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)} \tag{3.5}$$

Here, $P(C|X)$ is the posterior probability which tells us the probability of hypothesis C being true given that event X has occurred. In this work, hypothesis C is the probability of belonging to class Up/Down and event X is our test data. $P(X|C)$ is a conditional probability of occurrence of event X, given hypothesis C is true. It can be estimated from the training data. The working of naive Bayesian classifier, or simple Bayesian classifier, is summarized as follows.

Assume that, m classes $C_1, C_2, \ldots, C_m$ and event of occurrence of test data, X, is given. Bayesian classifier classifies the test data into a class with the highest probability. By Bayes' theorem (Equation (3.5)),

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}, i = 1, 2, \cdots, m \tag{3.6}$$

Given data sets with many attributes $(A_1, A_2, ..., A_n)$, it would be extremely computationally expensive to compute $P(X|C_i)$. In order to reduce computation in evaluating $P(X|C_i)$, the naive assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the tuple (i.e. that there are no dependence relationships among the attributes). Therefore,

$$\begin{aligned} P(X|C_i) &= \prod_{k=1}^{n} P(x_k|C_i) \\ &= P(x_1|C_i) \times P(x_2|C_i) \times \cdots \times P(x_n|C_i) \end{aligned} \tag{3.7}$$

Here, x_k denotes the value of attribute A_k for tuple X. Computation of $P(x_k|C_i)$ depends on whether it is categorical or continuous. If A_k is categorical, $P(x_k|C_i)$ is the number of observations of class C_i in training set having the value x_k for A_k divided by the number of observations of class C_i in the training set. If A_k is continuous-valued, Gaussian distribution is fitted to the data and the value of $P(x_k|C_i)$ is calculated based on Equation 3.8.

$$\begin{aligned} f(x, \mu, \sigma) &= \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \\ &\ so\ that, \\ P(x_k|C_i) &= f(x_k, \mu_{C_i}, \sigma_{C_i}) \end{aligned} \tag{3.8}$$

Here, μ_{C_i} and σ_{C_i} are the mean (i.e., average) and standard deviation, respectively, of the values of attribute A_k for training tuples of class C_i. These two quantities are then plugged into Equation 3.8 together with x_k, in order to estimate $P(x_k|C_i)$. $P(X|C_i)P(C_i)$ is evaluated for each class C_i in order to predict the class label of X. The class label of observation X is predicted as class C_i, if and only if

$$P(X|C_i)P(C_i) > P(X|C_j)P(C_j) \quad for\ \ 1 \leq j \leq m; j \neq i \tag{3.9}$$

Bayesian classifiers also serve as a theoretical justification for other classifiers that do not explicitly use Bayes' theorem. For example, under specific assumptions, it

can be demonstrated that many neural networks and curve-fitting algorithms output the maximum posteriori hypothesis, as does the naive Bayesian classifier.

3.5 Experimental Evaluation

This section discusses about evaluation measures, experimental methodology and results of the experimentations.

3.5.1 Evaluation Measures

Accuracy and F-measure are used to evaluate the performance of proposed models. Computation of these evaluation measures require estimating Precision and Recall which are evaluated from True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN) (Han, Kamber, and Pei Markov and Larose). These parameters are defined in Equations 3.10, 3.11, 3.12 and 3.13.

$$Precision_{positive} = \frac{TP}{TP + FP} \tag{3.10}$$

$$Precision_{negative} = \frac{TN}{TN + FN} \tag{3.11}$$

$$Recall_{positive} = \frac{TP}{TP + FN} \tag{3.12}$$

$$Recall_{negative} = \frac{TN}{TN + FP} \tag{3.13}$$

Precision is the weighted average of precision positive and negative while Recall is the weighted average of recall positive and negative. Accuracy and F-measure are estimated using Equations 3.14 and 3.15 respectively. It is to be noticed that, the value range of these measures is between 0 and 1, where 0 indicates the worst performance while 1 indicates the best performance.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.14}$$

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.15}$$

Accuracy of a classifier on a given test set is the percentage of test set observations that are correctly classified by the classifier. It is also to be mentioned that F-measure used throughout the thesis is traditional/balanced F-measure or F_1-score.

3.5.2 Experimental Methodology & Results

Experiments are carried out in two phases.

First phase of experimentation considers input as the continuous-valued data. The best parameter combinations are identified by means of experiments on parameter setting data set for each of the prediction models. These parameter combinations with corresponding accuracy and f-measure during parameter setting experiments are reported in Tables 3.7, 3.8 and 3.9. It is to be noted that there are no parameters to be tuned for naive Bayes classifier.

Comparison data set (which is a whole data set) is used to compare the performance of various prediction models. 50% observations of these data set are used to learn various prediction models along with the parameters identified during parameter tuning experiments. Remaining 50% observations are used as holdout observations and accuracy reported in the chapter shows the accuracy on these holdout observations. Table 3.10 reports average accuracy and f-measure of each of the models during comparison experiment. Average accuracy and f-measure reported are averaged over the top performing models. It can be seen that naive Bayes with the Gaussian process is the least accurate while Random Forest is the most accurate with average accuracy of nearly 84%. Figure 3.2 depicts the prediction process when data is continuous-valued.

Second phase of experimentation is identical to the first one except that the input to the models is trend deterministic data. The idea is depicted in Figure 3.3. Tables 3.11, 3.12 and 3.13 show result of best performing combinations for ANN, SVM and Random Forest respectively during parameter setting experiments. It is to be noted that when data is represented as trend deterministic data, naive Bayes classifier is learnt by fitting multivariate Bernoulli distribution to the data. Results on comparison data set for all the proposed models are reported in Table 3.14. Final comparison shows that all the models perform well with discrete data input but SVM, random forest and naive Bayes performance better than ANN. The accuracy of SVM, Random Forest and naive Bayes is nearly 90%.

3.6 Discussions

Stock market data is an example of non-stationary data. At particular time there can be trends, cycles, random walks or combinations of the three. It is desired that if a particular year is part of a cycle, say a bullish one, the proposed models should follow this pattern for trend prediction. Same can be considered for a trending year. However, usually stock values of a particular year are not isolated and there are days with random walks. Stock values are also affected by external factors creating trends and state of the country's economy. Political scenarios are also the influencing factors which may result in cycles.

Table 3.7: Best three parameter combinations of ANN model and their performance on continuous-valued parameter setting data set

CNX Nifty			
	ep:*n*:*mc* & *lr*=0.1		
	10000:20:0.6	7000:10:0.7	7000:10:0.9
Accuracy	0.8434	0.8450	0.8558
F-measure	0.8614	0.8606	0.8686
S&P BSE Sensex			
	ep:*n*:*mc* & *lr*=0.1		
	1000:80:0.1	2000:40:0.2	10000:100:0.1
Accuracy	0.7968	0.7827	0.7723
F-measure	0.7743	0.7982	0.7862
Infosys Ltd.			
	ep:*n*:*mc* & *lr*=0.1		
	1000:70:0.7	8000:150:0.7	3000:10:0.3
Accuracy	0.7417	0.7023	0.6949
F-measure	0.7581	0.7098	0.7412
Reliance Industries			
	ep:*n*:*mc* & *lr*=0.1		
	8000:50:0.6	6000:40:0.4	9000:20:0.5
Accuracy	0.6356	0.6326	0.6898
F-measure	0.6505	0.6116	0.7067

It can be seen from the results that all the models perform well when they are learnt from continuous-valued inputs but the performance of each of the models is further improved when they are learnt using trend deterministic data. The reason behind the improved performance is justified in the remainder of this section.

Table 3.8: Best two parameter combinations (one for each type of kernel) of SVM model and their performance on continuous-valued parameter setting data set

CNX Nifty		
	Kernel:Polynomial	Kernel:RBF
	c=100,d=1	c=0.5,γ=5
Accuracy	0.8427	0.8057
F-measure	0.8600	0.8275
S&P BSE Sensex		
	Kernel:Polynomial	Kernel:RBF
	c=100,d=1	c=0.5,γ=5
Accuracy	0.8136	0.7823
F-measure	0.8321	0.8015
Infosys Ltd.		
	Kernel:Polynomial	Kernel:RBF
	c=0.5,d=1	c=0.5,γ=5
Accuracy	0.8139	0.7836
F-measure	0.8255	0.7983
Reliance Industries		
	Kernel:Polynomial	Kernel:RBF
	c=0.5,d=1	c=1,γ=5
Accuracy	0.7669	0.6881
F-measure	0.7761	0.7023

Trend deterministic data is prepared by discretizing the continuous-valued data. The idea is based on the intuition that each continuous-valued parameters when

compared with its previous day's value indicates the future up or down trend. The data is discretized based on these heuristics. When this data is given as the input to the model, we are already inputting the trend based on each input parameters. It is actually the situation where each of the input parameters signifies about the probable future trend and we have the actual future trend to identify the transformation from probable trends to the correct trend. This is a step forward because original dataset is converted to trend deterministic data set. Now prediction models have to determine co-relation between the input trends and output trend. Though it is non-linear, it is easy to create a model which can transform input trends to the output trend.

Table 3.9: Best three parameter combinations of random forest model and their performance on continuous-valued parameter setting data set

CNX Nifty			
	ntrees		
	140	20	30
Accuracy	0.9148	0.9146	0.9099
F-measure	0.9186	0.9185	0.9162
S&P BSE Sensex			
	ntrees		
	80	50	70
Accuracy	0.8819	0.8719	0.8786
F-measure	0.8838	0.8742	0.8802
Infosys Ltd.			
	ntrees		
	50	110	200
Accuracy	0.8138	0.8059	0.8132
F-measure	0.8202	0.8135	0.8190
Reliance Industries			
	ntrees		
	160	60	150
Accuracy	0.7368	0.7441	0.7450
F-measure	0.7389	0.7474	0.7478

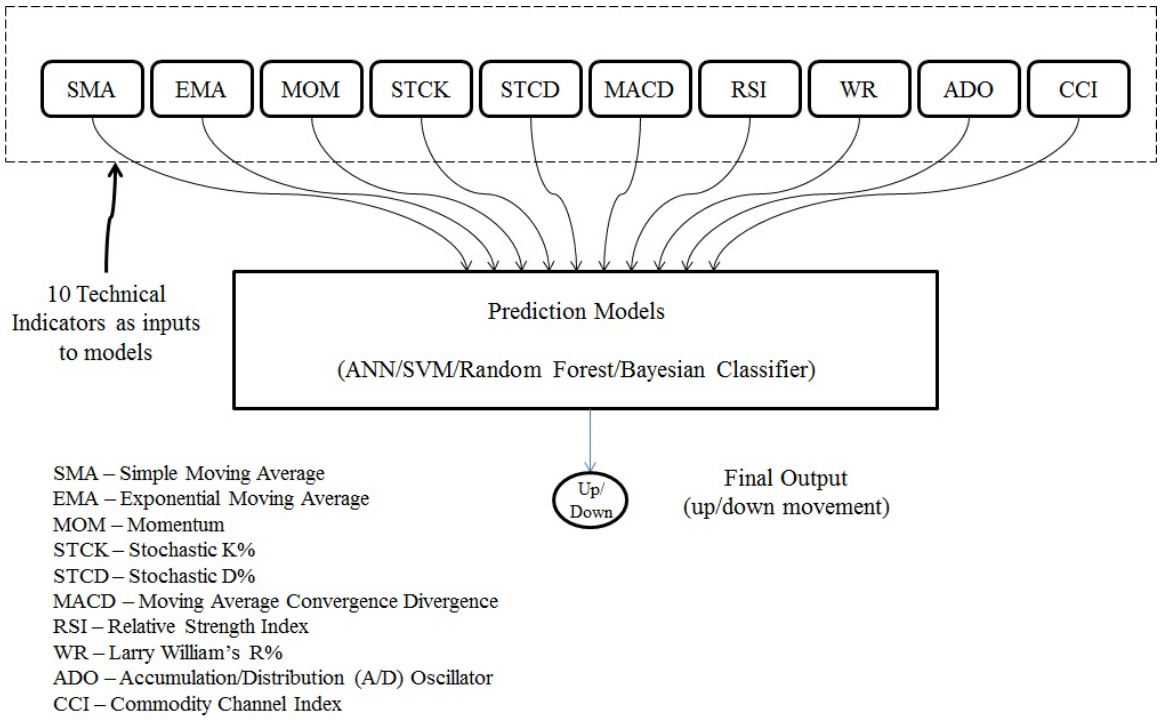

Figure 3.2: Predicting with continuous-valued data

Table 3.10: Performance of prediction models on continuous-valued comparison data set

Stock/Index	**Prediction Models**			
	ANN (Kara, Acar Boyacioglu, and Baykan)		**SVM**	
	Accuracy	F-measure	Accuracy	F-measure
S&P BSE Sensex	0.7839	0.7849	0.7979	0.8168
CNX Nifty 50	0.8481	0.8635	0.8242	0.8438
Reliance Industries	0.6527	0.6786	0.7275	0.7392
Infosys Ltd.	0.7130	0.7364	0.7988	0.8119
Average	0.7494	0.7659	0.7871	0.8029
Stock/Index	**Random Forest**		**Naive Bayes(Gaussian)**	
	Accuracy	F-measure	Accuracy	F-measure
S&P BSE Sensex	0.8775	0.8794	0.7354	0.7547
CNX Nifty 50	0.9131	0.9178	0.8097	0.8193
Reliance Industries	0.7420	0.7447	0.6565	0.6658
Infosys Ltd.	0.8110	0.8176	0.7307	0.7446
Average	0.8359	0.8399	0.7331	0.7461

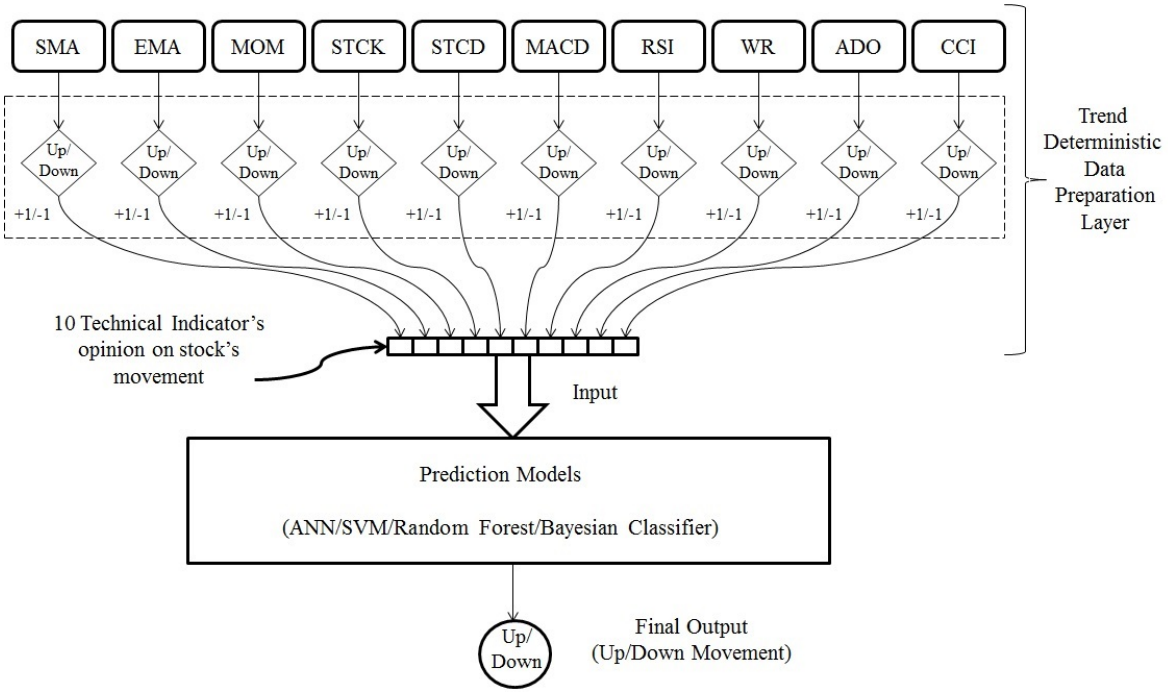

Figure 3.3: Predicting with trend deterministic data

Further to notice is that for any stock or index, there are scenarios, when a stock or index is trading at some value, say 200, then due to some external factors, it may start trading at higher price, say 400, and then stabilize at that higher value. If prediction model is given direct continuous-valued input, it is possible that it tries to establish relation between the values in 200 and that in 400, which is not required as far as predicting future trend is considered.

Each parameter is relative while signifying future trend. It means that the important thing is how its value has changed with respect to previous days rather than the absolute value of change. Therefore, trend deterministic data which is discrete in nature is basically the statistical indication of whether the stocks are over-bought or over-sold and is value independent. Hence, these input parameters, when represented as probable future trends serve as a better measure of stocks condition rather than the scenario, when they are represented as continuous-valued.

3.7 Conclusions

The task focused in this study is to predict direction of movement for stocks and stock price indices. Prediction performance of four models namely ANN, SVM, Random Forest and Naive Bayes is compared based on ten years (2003-2012) of historical data

of CNX Nifty, S&P BSE Sensex, Infosys Ltd. and Reliance Industries from Indian stock markets. Ten technical parameters reflecting the condition of stock and stock price index are used to learn each of these models.

Table 3.11: Best three parameter combinations of ANN model and their performance on discrete-valued parameter setting data set

CNX Nifty			
	ep:*n*:*mc* & *lr*=0.2		
	4000:50:0.8	1000:100:0.6	3000:70:0.3
Accuracy	0.8703	0.8740	0.8729
F-measure	0.8740	0.8768	0.8801
S&P BSE Sensex			
	ep:*n*:*mc* & *lr*=0.1		
	6000:100:0.4	2000:30:0.3	4000:90:0.1
Accuracy	0.8563	0.8728	0.8717
F-measure	0.8632	0.8771	0.8759
Infosys Ltd.			
	ep:*n*:*mc* & *lr*=0.1		
	6000:50:0.1	4000:70:0.2	9000:80:0.4
Accuracy	0.8531	0.8717	0.8468
F-measure	0.8600	0.8742	0.8503
Reliance Industries			
	ep:*n*:*mc* & *lr*=0.2		
	1000:100:0.1	4000:90:0.9	8000:100:0.5
Accuracy	0.8573	0.8747	0.8808
F-measure	0.8620	0.8799	0.8826

Table 3.12: Best two parameter combinations (one for each type of kernel) of SVM model and their performance on discrete-valued parameter setting data set

CNX Nifty		
	Kernel:Polynomial	Kernel:RBF
	c=1,d=1	c=1,γ=4
Accuracy	0.9010	0.8808
F-measure	0.9033	0.8838
S&P BSE Sensex		
	Kernel:Polynomial	Kernel:RBF
	c=1,d=1	c=5,γ=1.5
Accuracy	0.8959	0.8780
F-measure	0.8980	0.8810
Infosys Ltd.		
	Kernel:Polynomial	Kernel:RBF
	c=0.5,d=1	cc=1,γ=3
Accuracy	0.8895	0.8865
F-measure	0.8916	0.8880
Reliance Industries		
	Kernel:Polynomial	Kernel:RBF
	c=1,d=1	c=0.5,γ=4
Accuracy	0.9221	0.8923
F-measure	0.9229	0.8932

A Trend Deterministic Data Preparation Layer is proposed on the basis of the fact that each technical indicator has its own inherent property through which traders generally predict the stocks' up or down movement. Utilizing these heuristics, the trend deterministic data preparation layer converts each of the technical parameter's

continuous value to +1 or −1, indicating probable future up or down movement respectively.

Table 3.13: Best three parameter combinations of random forest model and their performance on discrete-valued parameter setting data set

CNX Nifty			
	ntrees		
	30	120	20
Accuracy	0.8913	0.8973	0.8969
F-measure	0.8934	0.8990	0.9005
S&P BSE Sensex			
	ntrees		
	20	90	110
Accuracy	0.8886	0.8981	0.9011
F-measure	0.8914	0.9012	0.9028
Infosys Ltd.			
	ntrees		
	50	60	70
Accuracy	0.9035	0.8964	0.9004
F-measure	0.9051	0.8980	0.9019
Reliance Industries			
	ntrees		
	30	10	40
Accuracy	0.9079	0.9088	0.9070
F-measure	0.9085	0.9098	0.9078

Experiments with continuous-valued data show that naive Bayes (Gaussian Process) model exhibits least performance with 73.3% accuracy and random forest with highest performance of 83.56% accuracy. Performance of all these models is improved

significantly when they are learnt through trend deterministic data. ANN is slightly less accurate in terms of prediction accuracy compare to other three models which perform almost identically. The accuracy of 86.69%, 89.33%, 89.98% and 90.19% is achieved by ANN, SVM, Random Forest and Naive Bayes (Multivariate Bernoulli Process) respectively.

Table 3.14: Performance of prediction models on discrete-valued comparison data set

Stock/Index	**Prediction Models**			
	ANN		**SVM**	
	Accuracy	F-measure	Accuracy	F-measure
S&P BSE Sensex	0.8669	0.8721	0.8869	0.8895
CNX Nifty 50	0.8724	0.8770	0.8909	0.8935
Reliance Industries	0.8709	0.8748	0.9072	0.9080
Infosys Ltd.	0.8572	0.8615	0.8880	0.8898
Average	0.8669	0.8714	0.8933	0.8952
Stock/Index	**Random Forest**		**Naive Bayes**	
	Accuracy	F-measure	Accuracy	F-measure
S&P BSE Sensex	0.8959	0.8985	0.8984	0.9026
CNX Nifty 50	0.8952	0.8977	0.8952	0.8990
Reliance Industries	0.9079	0.9087	0.9222	0.9234
Infosys Ltd.	0.9001	0.9017	0.8919	0.8950
Average	0.8998	0.9017	0.9019	0.9050

Trend Deterministic Data Preparation Layer proposed in this chapter exploits inherent opinion of each of the technical indicators about stock price movement. The layer exploits these opinions in the same way as the stock market's experts. In earlier researches, the technical indicators were used directly for prediction while this study first extracts trend related information from each of the technical indicators and then utilizes the same for prediction, resulting in significant improvement in accuracy. The proposal of this Trend Deterministic Data Preparation Layer is a distinct contribution to the research. Owing to the noteworthy improvement in the prediction accuracy, the proposed system can be deployed in real time for stocks' trend prediction, making investments more profitable and secure.