DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY,

COLLEGE OF ENGINEERING, PUNE

CERTIFICATE

This is to certify that the dissertation titled

Prediction of Stock Price Movement Using Various Ensemble Models

has been successfully completed

By

Pradeep Kumar Mahato

MIS No: 121222011

and is approved for the partial fulfillment of the requirements for the degree of

Master of Technology, Computer Engineering

Dr. Vahida Z. Attar
Project Guide,
Department of Computer Engineering
and Information Technology,
College of Engineering, Pune,
Shivaji Nagar, Pune-411005.

Dr. J. V. Aghav
Head,
Department of Computer Engineering
and Information Technology,
College of Engineering, Pune,
Shivaji Nagar, Pune-411005.

June 2014

Acknowledgments

I express my deepest gratitude towards my guide Prof. Dr. Vahida Z. Attar for her constant help and encouragement throughout the project and also for providing me infrastructural facilities to work in. I have been fortunate to have a guide who gave me the freedom to explore on my own and at the same time helped me plan the project with timely reviews and constructive comments, suggestions whenever required. A big thanks to her for having faith in me throughout the project and helping me walk through the new avenues of research papers and publications.

I would like to thank Dr. Aniruddha Pant for his continuous guidance throughout the project; he has constantly encouraged me to remain focused on achieving my goal. I also like to convey my sincere gratitude to Dr. J. V. Aghav(HOD), all faculty members and staff of Department of Computer Engineering and Information Technology, College of Engineering, Pune for all necessary cooperation in the accomplishment of dissertation. I would take this opportunity to thanks all those teachers, staff and colleagues who have constantly helped me grow, learn and mature both personally and professionally throughout the process. Last but not least, I would like to thank my family and friends, who have been a source of encouragement and inspiration throughout the project.

Pradeep Kumar Mahato

College of Engineering, Pune

Abstract

Stock market prediction has always been an interesting research topic among researchers mainly due to its capital gain by trading stocks and or to understand the information hidden in stock market data. Many machine learning algorithms and statistical models have been proposed by researchers for stock price prediction and stock price movement prediction. In this paper we examine various machine learning techniques and integrate them using different ensemble models for predicting the direction of change of stock price, whether the stock price will increase or decrease for the following day relative to the current day's stock price. With knowledge of the movement of stock price, an investor can make profitable decision and reduce risk-return trade-off. We reduced this prediction model to binary classification problem and used previous year's historical data along with technical indicators for better accuracy.

The goal of this project is to design and implement various generative and non-generative ensemble methods for predicting the stock price movement. As the time series data in stock prediction is highly complex and noisy, we normalized our dataset. Then we tune the hyper-parameters of the machine learning algorithms and we examine each algorithm using factors like accuracy, f1_measure, harmonic accuracy and uncertainty score and finally combine the best algorithms using different customized ensemble methods. We design and implement various ensembles models, We also implemented different deeplearning techniques like autoencoder and Denoising autoencoder for stock prediction, but it didn't show any significant accuracy than our previous results.

Our approach demonstrates the prediction accuracy is increased in different ensemble approach rather than using individual methods. Bagging methods gives us the best accuracy for our datasets.

Contents

List of Figures:

List of tables

Chapter 1

Introduction

Stock prediction is an interesting problem, many researchers have attempted to predict the future stock prices using various ARIMA models, mathematical and statistical models, machine learning techniques like support vector machine (SVM) and neural network techniques like recurrent neural network. In this thesis we use ensemble models to predict future momentum of stock.

1.1 Stock Market and Stock Prediction

Stock prediction is a process of forecasting the future value of the stock of a company. A stock is a unit of ownership of a company. More number of stocks will lead to greater ownership stake in the company. A stock market is place where people trade (buy or/and sell) stocks, options and futures, and commodities. Buyers and sellers exchange trade together via a platform provided by stock exchanges. In India, the prominent stock exchanges are National stock exchange and Bombay Stock Exchange. Investors buy or sell stocks according to their analysis. The stock price of a company is determined by demand and supply of the stock. So if any person wants to buy, somebody else has to sell, and vice versa. Now if greater number of people willing to buy a stock than sell the stock at current price, the stock price goes up. When the price goes gets so high that people are not willing to buy any more, the price starts to drop. Similarly if greater numbers of people are willing to sell a stock than buy the stock, the stock price decreases. In stock market, people make profit by buying stocks that they believe that it will worth more in future. Thus making profit by selling it in future. In similar fashion, if any stock is continuously decreasing, sell the stock at current price and buy it when the stock price has become very less from the selling price.

Due to non-linear and volatile behavior of stock market, it is not possible to make profitable decisions and get returns consistently. Thus it is impossible to correctly predict the future stock price. We need a model that predicts the future stock price to a satisfactory accuracy and predict when and which stock to buy or sell. This problem

can be transformed into an engineering problem and solved by application of engineering methods like data mining or machine learning.

Many investors made stock prediction using gut feeling or they evaluate the performance of the company performance to make investment decision. Nowadays mostly stock prediction is done using statistical methods or machine learning techniques. Many researchers have proposed various prediction algorithms which uses historical data for stock forecasting. Our prediction model attempts to predict the movement of stock price in the next day by using historical data of the stock. With this knowledge, an investor can make profitable decisions and reduce the risk

1.2 Previous theories related to Stock Prediction

Random Walk: It states that stock price movement will not follow any pattern nor historic stock price data can be used to predict future price. It follows efficient market hypothesis.

Efficient Market Hypothesis (EMH): It states that for any given time, the stocks are information efficient. In other words beating the market is impossible and no trading strategies can be used for making profit. At any given time, the stock is fairly valued. So any investor buys a stock under the impression that the stock price will increase or investor sells a stock assuming the stock price will go down. According to EMH, stock market is efficient, no way to identify a bargain stocks. At any one time, anyone and everyone already knows all relevant information about the stock, thus buying or selling the stock to achieve any profit is only a matter of chance rather than skill.

There are three forms of EMH:

Weak Efficient Market Hypothesis: This market is efficient market, reflecting all the market information instantly. The past prices of a stock have no effect on future price so no technical analysis can be applied for stock prediction. Stock market adjusts any new information quickly and fully.

Semi-strong Efficient Market Hypothesis: This market is efficient market reflecting all public information instantly. As public information is reflected quickly, no fundamental analysis can be applied for stock prediction. Thus an investor cannot

make profits by taking investment decision based on any new information. Weak efficient market hypothesis is a subset of this form.

Strong Efficient Market Hypothesis: This market is strongly efficient, reflecting all past stock price data, current and private information. No fundamental or technical analysis can be applied for stock prediction. Weak EMH and Semi-strong EMH are subset of this form.

The Random Walk and Efficient Market Hypothesis is highly disputed theory, because:

* Every individual investor view information differently, therefore each investor will evaluate stocks differently.

* Stocks take time to respond to information. Thus investors who act on this information first can take advantage of it and make huge profits.

* Stock prices can be affected by human error and emotional decisions.

* Investors had proven they can profit from market-analysis.

Fundamental Analysis: It is method to value stocks by examining economic and financial factors related to the company. The financial statements of a company reveal information like debt load, margin, price multiple, and book value, price / earnings ratio and many other metrics. Many economic factors like inflation, interest-rates, price of important commodities like oil, different metal and currencies also affect the trading decision using fundamental analysis. Most of time the current stock value will always be overvalued or undervalued , using fundamental factors about the company, an investor can make trading decisions to buy or sell a stock and make profit. The current stock price always tends to move towards its real stock value of the company. If the real value of the stock is more than the current market stock price, any good analyst would buy the stock as the current market stock price will rise to move towards the stock real price. Similarly if the real stock price is below the current market price, it is better to sell the stock as the current market stock price will move towards the real stock price. Thus the current market price will decrease. Thus to estimate real stock price, analysts does fundamental analysis on the current and future overall health of the company

Technical Analysis: It is a method to value stocks by using historic stock price data and volume. It is assumed that price moves in trends or patterns and any patterns in historical stock price data will tend to repeat in future. Technical analysis is done using charts, tables, statistical methods like regression, Bayesian. Stock charts illustrate the stock price movement of any stock. Technical analyst find different pattern as stock trends tends to persist over time. Any stock can be seen in uptrend, downtrend or range bound swings back and forth. Volume is an importance metric in technical analysis, it shows the participation of crowd. Technical indicators measure the relationship of current stock price to past price. Thus many prominent technical indicators like relative strength index, moving averages, rate of change are also used in technical analysis. Recently many machine learning technique and artificial techniques are used for technical analysis.

1.3 Motivation

Stock prediction has always been interesting topic for researchers, due to its monetary rewards, complexity and unpredictable nature. With the proliferation of computer technology more calculating and data processes abilities are accessible to individual / institutional researchers. The major motivation for our work is to understand any hidden pattern in the stock market using machine learning algorithms. Secondly to design a prediction model that can continually predict future stock value with greater accuracy. Thus make better trading decisions and earn money.

1.4 Problem Definition

The goal of this thesis is to implement various ensemble models and find out the best prediction model applicable to all our dataset.

1.5 Thesis Outline

The rest of the thesis is organized as follows:

In chapter 2 we give a brief description of the importance papers and literature that we have studied or utilized as part of our literature survey. In Chapter 3, we present background to all the models and ensemble methods used in thesis. In Chapter 4, we present our proposed model to tackle the problem of stock prediction, objectives and scope of this thesis. In chapter 5, we show our various experiment and models. In

Chapter 6, we do result analysis to find the best prediction model for each dataset. Finally Conclusion and Future Work is mentioned in Chapter 7.

Chapter 2

Literature Survey

Stock Prediction has been considered as both interesting and difficult research topic in the field of finance and engineering due to its potential financial gain, as the financial time series are inherently noisy, dynamic and deterministically chaotic. However, many researchers have strived for proving the predictability of the financial market by doing fundamental analysis, technical analysis or using some sort of hybrid model involving both. However researchers have proposed many mathematical models for stock prediction. In recent years many models based on machine learning techniques are proposed, such as neural networks, decision trees genetic algorithms, rough set theory, support vector machines and hidden Markov model to build a financial prediction model. Ensemble model are combination of individual machine learning algorithms. The accuracy of ensemble models is greater than individual models.

2.1 Stock Prediction and Ensemble Models

Dietterich [1]has reviewed all the ensemble methods with proper mathematical proofs, Matteo and Valentini [2]has reviewed the ensemble methods extensibly covering all possible taxonomies related to ensemble models. They have also shown applications of ensemble models in astronomy and astrophysics. Brofos [3]proposed an ensemble of random forest classifier, support vector machine and relevance vector machine classifiers, including an ensemble of k-nearest neighbor classifiers. Using relief-f algorithm for identifying the informative features of the dataset. And implemented with customized boosting algorithms. This model achieved an accuracy of 70% on data from Global Industry Classification Standard finance sectors. This model was made to predict stock price for next quarter, it seems truly justifiable how information gathered in one quarter is not useful for prediction in next quarter.

Zaamout [4]implemented novel ensemble approached to improve the neural network classification power. The first approach combines the output of a set of neural network together via another neural network. And the second improves the accuracy of the

neural network, by feeding the output of neural network and repeat this process until the error is reduced sufficiently. This is single link chaining and multiple link chaining model, better accuracy is achieved at the cost of computing time. Nikunj [5]have applied an online ensemble learning algorithm for a very large dataset. Online version of bagging and boosting are used to increase the accuracy as well as speed of classification problem. Upadhaya et al. [6] Categorizes stock performance in the stock market into three categories GOOD, average and poor using multi logistic regression. This model had been constructed using seven financial ratios, book value, paid/sales and earnings per share, percentage change in operating profit, percentage change in net sales, price to cash earning per shares, and price to book value. This model has shown promising 56% profit on Indian stock market.

Shen et al. [7] proposed prediction model based on SVM to predict the next-day stock trend by exploiting the correlation between the stock prices of stocks trading at the beginning or end of US markets as the possible features for a stock. Increasing more number of features the accuracy of the prediction model increases. For single feature prediction oil price, Australian dollar and FTSE shows an accuracy of 67.2%, 66.4% and 65.2%. Their model showed an accuracy of 70.8% for predicting 70.8%. SVM and MART were compared for a multi feature dataset, SVM had shown 74.4% compared to 70.3% accuracy for MART. SVM seems to outperform regression and GLM on the same dataset. SVM is a good prediction model, many researchers have used in their proposed model. Chen et al. [8] Examined different machine learning techniques for stock prediction, Logistic Regression, K-nearest neighbors algorithm, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and Support Vector Machine for predicting S&P 500 index. Logistic Regression was the best model among them.Using support vector machine Yuan [9]proposed an integrated model of Principal component analysis and Support vector Machine for stock prediction for Korean stock exchange and Hangseng index for next following day. The PCA-SVM approach seems to be working for different stock with accuracy greater than 65%.

Pham et al. [10] implemented a framework integrating previous historical data with user predictions. Two prediction models is used, first one is based on regression model using historic data to learn and second prediction model learns from user input, using weighted majority algorithm. In weighted majority algorithm , each user is given some weight and at the end of the day, weights of the user is updated if the user predicts

wrong, else it is kept the same. Probabilistic combinations of two algorithms are done, though this approach is suitable for real time trading situation.K-nearest neighbor (kNN) approach is a suitable algorithm for data mining problems, Khalid et al. [11]used kNN and non-linear regression approach to predict Jordanian stock exchange. They concluded that kNN is a robust algorithm and predictions were nearly similar to the stock price. An efficient prediction model can be judged how it accurately predicts increase or decrease in returns in real time data. Yen et al [12]proposed an ensemble model for predicting returns using multilayer perceptron, classification and regression tree and logistic regression. They compared individual models, majority voting approach and bagging also shown different homogeneous and heterogeneous models accuracy. Greater accuracy is achieved when more number of classifiers is used. They concluded that their given model is better than simple buy and hold strategy. Panda et al [13] proposed an extension to Multilayer Perceptron based model for predicted S&P 500 and DJIA indices. Optimizing the model with Particle Swarm Optimization, this model is believed to be computationally more efficient, takes less training time and show better accuracy. They have predicted for next 1,3,5,7 days. They have used mean absolute percentage error MAPE) as the metric for showing their results. The MAPE increases with increasing the number of test set and it has decreased the computation time by a factor of 4. Suresh Kumar and Elango [14] proposed that isotonic regression has better accuracy in predicting stock price than existing approaches like artificial neural network approach. They used stock price of Infosys technology to predict whether the stock price will go up or down. Their analysis proves that isotonic regression performs belter than other functions such as Gaussian Process, linear regression, SMO regression and multilayer perceptron. Kwon and moon [15] proposed a neuro-genetic hybrid for stock prediction by not fixing the members of ensemble but dynamically chooses the members that perform well for the days with similar context's to present day by computing in terms of Euclidean distance. Chitra and Uma [16]proposed a novel two level ensemble learning approach based on Radial Basis Function network (RBF), K - Nearest Neighbor (NN) and Self Organizing Map (SOM), it perform better than the individual classifiers. In the first level ten diverse models of these classifiers are generated by varying the architecture, the best rule is used to train in the second level. Finally the second heterogeneous ensemble gives the required prediction. They measured the performance of the classifiers by using Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and Normalized

Mean Square Error (NRMSE). The paper does not give any detail about the ensemble technique used in the second level. Grosan and Abraham [17]proposed a genetic programming ensembles of Multi-Expression Programming (MEP) and Linear Genetic Programming (LEG), the goal was to minimize Root Mean Squared Error (RMSE), Maximum Absolute Percentage Error(MAP) and Mean Absolute Percentage 4Error (MAPE)and maximize Correlation Coefficient (CC). A given parameters were passed to NSGAII for combining MEP and LEG. The results outperformed the ANN and FZ approach. Chen, Yang and Abraham [18]proposed a flexible neural trees (FNT) ensemble for stock index modeling. The structure and parameter of FNT is optimized using Particle Swarm Optimization algorithm. Local Weighted Polynomial Regression (LWPR) ensemble model is used to ensemble FNTs.

Qian, Rasheed [19] proposed a ensemble stock market prediction using Artificial Neural Network (ANN), NN, Decision Tree. For ANN they used Levenberg-Margquardt learning algorithm with sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. They used simplest weighted voting and stacking method to combine the classifiers, But they couldn't achieve significant accuracy. Shah [20] proposed a Stock prediction Model based on Machine Learning using Support Vector Machine (RBFkernel) using Adaboost algorithm for Cost Parameter C from 512 to 65356. This paper touched many aspects of stock prediction like the importance of indicators and linguistic analysis of Financial News Results. They got a satisfactory accuracy of 60%. Pulido, Castillo and Melin [21] proposed Genetic Optimization of ensemble neural networks for complex time series prediction of the Mexican exchange. They implemented a Genetic Algorithm to optimize the ensemble 5neural network architectures for each of the modules, and thus to find a neural network architecture that yields optimum results in each of the Time Series that dataset. Wu, Chen and Liu [22]proposed Ensemble model of Intelligent Paradigms for Stock Market Forecasting which is an ensemble of SVM and ANNs. The performance of the method is measured in terms of Mean Absolute Percentage Error (RMSE). Tsai, Lin, Yen and Chen [23] proposed predicting stock returns by classifier ensembles using MLP, CART and LR. They proved how homogeneous and heterogeneous ensembles are more efficient than the single classifier ensembles. They show both majority voting and bagging method of ensemble learning. Rao and Hong [24] proposed and unique analysis of Hidden Markov Models and Support Vector Machines in Financial

Applications, they also show how Efficient Market Hypothesis and Technical indicators can be used for forecasting stock prices and movements. They used the Simlio engine to train both the HMM and SVM to predict future stock movements. Si, Mukherjee, Liu , LI , Deng [25] has proposed a unique stock prediction model which leverage topic based sentiments from Twitter to help predict the stock Market. They first utilize a continuous Dirichlet Process Mixture model to learn the daily topic set, then for each topic sentiments according to its opinion words then regress the stock index to predict the market. This model has shown an significant accuracy of 68 %. Schumaker and Chen [26]has proposed a quantities stock prediction system based on financial news. They have used support vector regression to classify news. Soni and Srivastava [25] proposed classification of Indian stock market data using Machine learning algorithms like CART, Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA). They proved how CART algorithm gives greater accuracy than LDA and QDA. Shen, Jian and Zang [26] proposed an algorithm that exploits the temporal correlation among global stock markets and various financial products to predict the next-day stock trend with the aid of SVM. There focus was in the correlation between the Closing prices of the markets that stop trading right before or at the beginning of US markets, and they belief that data of overseas stock and other financial markets, especially those having strong temporal correlation with the upcoming US trading day, should be useful to machine learning based predictor which they proved in their results.

2. 2 Deep learning

Before the advent of deep learning algorithms, classification of semantic content of a document were an unsolved problem, Document classification has many application like organizing journals and online information, spam filtering and topic based search engine. Geoffrey Hinton [27] has successfully developed a neural network that reduces the dimension of a large corpus of documents using deep belief networks of Restricted Boltzmann Machine RBM. Similarly yoshua bengio proposed denoising AutoEncoder [28] that works similar to Restricted Boltzmann Machines. Stacked denoising AutoEncoder works faster than RBM'S. It captures the most possible patterns in a dataset also reduces the dimension of the dataset by stacking different number of neurons on each layer. Denoising AutoEncoder can take noisy data as input

and find out best possible patterns in it. Stacked denoising AutoEncoder was used for handwritten digit recognition problem, it outperform the deep belief network proposed by geoffery hinton.

Xie et al [29] has proposed a unique ensemble methods using deep representation of AutoEncoders. They proposed horizontal and vertical voting ensemble method. This model achieved a good performance in ICML 2013 black box challenge. the vertical voting ensemble model is similar to majority voting, here the AutoEncoder technique are applied to the datasets in 3 levels and their predictions are added to make the final prediction of the test set.

Till now very few work has been done for stock prediction with deep learning.With development of techniques like AutoEncoder and denoising AutoEncoder, this can be applied to noisy stock market data to find the hidden patterns on its own. Till now very few work has been done for stock prediction with deep learning. Stock prediction can be performed with Vertical voting approach.

Chapter 3

Background

3.1 Deep Learning

3.1.1 AutoEncoder:

It is artificial neural network used for learning efficient pattern in the data. It learns a compressed representation of data without losing any importance pattern in the dataset. The architecture of an AutoEncoder is similar to Multilayer perceptron. It comprise of input layer, hidden layer and output layer. The number of neurons in input layer should be similar to the number of neurons in the output layer. And the number of neurons present in the hidden layer are less than equal to the number of neurons in the hidden layer. Also the goal of this neural network is to reconstruct the inputs using mean error or cross entropy function, rather than predicting anything.

Consider an input $x \epsilon [0,1]^d$, it is mapped to the hidden layer $y \epsilon [0,1]^d$ through a deterministic mapping like $y = s(\mathcal{W}x + b)$, where s is a non-linearity function like sigmoid, tanh, etc. Then y is mapped back into a reconstruction z of same shape as x through another similar transformation $z = s(\mathcal{W}'y + b')$. Here $\mathcal{W}' = \mathcal{W}^T$, the parameters $\mathcal{W}, b, b'$ are optimized by minimizing the average reconstruction error. The reconstruction error can be measured in traditional squared error loss $L(x, z) = \|x - z\|^2$ or using cross-entropy function

$$L(x, z) = -\sum_{k=1}^{d} [x_k \log z_k + (1 - x_k) log(1 - z_k)]$$

The diagram for an autoencoder is show below Fig:3.1.1

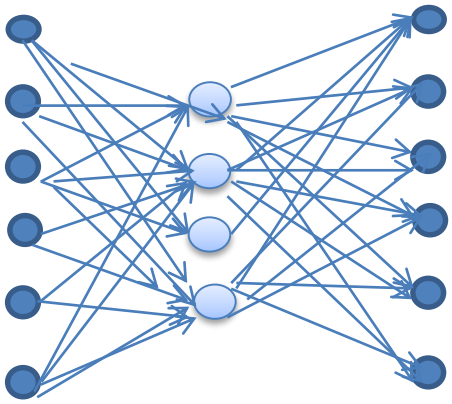


Figure3.1.1 AutoEncoder Architecture

The algorithm for training an AutoEncoder :

For each input x from $\mathcal{X}$

Feed –forward pass to compute all the activation unit at all hidden layers, then at the output layer

Measure the error using squared error loss or cross entropy loss function

Back propagate the error and perform weight updates

With no other constraint besides minimizing the reconstruction error, then an AutoEncoder can learn the identity function, Bengio[28] also proved that if we have more number of neurons on the hidden layer then input, it will yield useful representaton

3.1.2 Denoising AutoEncoder:

Denoising AutoEncoder is similar to an AutoEncoder but more powerful than AutoEncoder, in terms of discovering more robust features from input dataset, and prevent it from learning identity. We should train the AutoEncoder to reconstruct the

input from a corrupted version of it. The denoising AutoEncoder is stochastic AutoEncoder. It is also susceptible to noisy data. Based on architechture,it is similar to AutoEncoder only difference is here we have to train the AutoEncoder to reconstruct the input from a corrupted version of it. It tries to encode the input and try to undo the effect of a corruption process stochastically applied to the input of AutoEncoder.

3.2 Vertical Voting : [29]

It is a majority voting approach in which we have three different representation of the dataset, which will be used in each iteration for prediction.

Consider a dataset $\mathcal{D}$ of d features. We will used autoencoder to generate two reperesentation of this dataset by the following pseudo code

Our autoencoder will have d number of neurons in the input layer and d -1 number of neurons on the hidden layer and d – 2 number of neuron on the output layer, thus when the autoencoder get trained, we do a forward pass and get a compressed representation of our original dataset $\mathcal{D}$ of d to $\mathcal{D}_1$ of d-2 dimensions.

We repeat the above process again with $\mathcal{D}_1$ to get a new dataset of $\mathcal{D}_2$ of d-4 dimensions.

After we get the datasets we perform Vertical voting method:

for each row $r_1\ r_2\ r_3\ from$ datasets $D_1\ D_2\ D_3$

$$y_1 = prediction\ of\ r_1 using\ an\ base\ classifier\ C$$

$$y_2 = prediction\ of\ r_2 using\ an\ base\ classifier\ C$$

$$y_3 = prediction\ of\ r_3 using\ an\ base\ classifier\ C$$

$$y_{pred} = Majority\ Vote\ of\ (y_1 + y_2 + y_3)$$

3.3 Metrics:

In this thesis we are considering 5 metrics that we will be using in our different models.

Some models are based on confusion matrix

Confusion Matrix: It is a contingency table that shows the performance of an algorithm. The column of the matrix represents the instances in predicted class and rows of the matrix represents the instances in original class.

$$\begin{bmatrix} true\_positve & false\_positive \\ false\_negative & true\_negative \end{bmatrix}$$

True_positive : The number of instances which are predicted as positive class and originally it is also in positive class.

False_positive: The number of instances which are predicted as positive class but originally they are in negative class

False_negative: The number of instances which are predicted as negative class but originally they are in positive class.

True_negative: The number of instance which are predicted as negative class and originally also it is in negative class.

Accuracy : It is the ratio of total number of correct prediction to total number of predictions. Correct predictions means those are predicted positive class and originally it is positive class and predicted negative class and originally it is also in negative class.

Accuracy = $\frac{true_{postive} + true_{negative}}{true_{positive} + false_{positive} + true_{negative} + false\_negative}$

F1_measure: It is the harmonic mean of precision and recall, it measure the accuracy of the test.

$f1 = \frac{2 * precision * recall}{precision + recall}$

Where $precision = \frac{true_{positive}}{true_{positive} + false\_positive}$ and $recall = \frac{true_{positive}}{true_{positive} + false\_negative}$

Harmonic score = It is the harmonic means of original positive class and predicted positive class.

Harmonic score = $\frac{2*(true_{postive}+false_{negative})*(true_{positive}+false_{positive})}{2*true_{positive}+false_{negative}+false_{positive}}$

Uncertainty also called as proficiency, it calculates the nominal association of positive class to negative class

Uncertainty score = $\frac{L+LTP+LFP+LFN+LTN}{L+LP+LN}$

where,

Total instances = (*# of positive instances + # of negative instances*)

L = $Total\ instances * \log(Total\ Instances)$

LP = (# of *positive instances*) * $log\ \frac{\#\ of\ positive\ instances}{Total\ instances}$

LN = (# of *negative instances*) * $log\ \frac{\#\ of\ positive\ instances}{Total\ instances}$

LTP = $true_{positive} * \log(\frac{true_{positive}}{(true_{positive}+false_{positive})*(true_{positive}+false_{negative})})$

LTP = $false_{positive} * \log(\frac{false\_positive}{(true_{positive}+false_{positive})*(false_{positive}+true_{negative})})$

LTP = $false\_negative *$
$\log(\frac{false\_positive}{(true_{positive}+false_{negative})*(true_{negative}+false_{negative})})$

LTP = $true_{negative} * \log(\frac{true_{negative}}{(true_{negative}+false_{positive})*(true_{negative}+false_{negative})})$

Average sameness : It is measure of how two similar classifier predicts the same class.

3.4 Ensemble Models

3.4.1 Walk-forward Approach:

For each dataset we apply all the 12 chosen machine learning algorithms and get the output in walk forward approach. Our ensemble models should perform better than this accuracy. In Walk forward approach we train our dataset on first n number of rows from the beginning i.e. $\{ 1, 2, 3, .. n^{th}\ row \}$ and predict on $(n + 1)th$ row. Next we move our window by 1, training set will be from $\{ 2,3,4 .. (n + 1)^{th}\ row \}$ and testing set will be $(n + 2)^{th}$row, and so on.

3.4.2 Majority Voting :

It is the most used ensemble technique. It combines outputs from different classifiers into one output. It find the maximum voted class among odd number of votes and by taking the sum of every votes. For example I am having three classifier which classify an instance as 1 or -1. If I have predictions like 1 1 -1 than the sum is positive thus it is a positive class. Similarly if -1 1 -1 are my predictions, then my sum is negative. Thus my final prediction is negative.

It is the most used ensemble approach, we will be doing majority voting with different number of classifiers, thus to get best possible combination of classifiers.

Majority voting with k best classifiers. Pseudo code is as follows:

For each row r in dataset:

$\mathcal{X}$ = { r , r+1, ..r+500} as training set

$\mathcal{y}$ = { r + 501 } as test set

Do 10 fold cross validation on the training set $\mathcal{X}$ to get k best classifiers based on accuracy

Train the k best classifier on $\mathcal{X}$ and predict on $\mathcal{y}$

Take the sum of all predictions

If sum is positive assign predicted variable as positive class else negative class

3.4.2 Hybrid Majority Voting:

This approach is similar to majority voting, here we choose the classifier for majority voting based on intersection of metrics.

Hybrid Majority Voting approach

Pseudo code

For each row r in dataset:

$\mathcal{X}$ *= { r , r+1, ..r+500 } as training set*

$\mathcal{y}$ *= { r + 501 } as test set*

Do 10 fold cross validation on the training set $\mathcal{X}$ *to get intersection set of all the best classifiers based on each metrics: accuracy, f1_score, harmonic accuracy ,Uncertainty coefficient and Average sameness.*

Train the 3 best classifier on $\mathcal{X}$ *and predict on* $\mathcal{y}$

Take the sum of all predictions

If sum is positive assign predicted variable as positive class else negative class

3.4.3 Stacking

It is a non-generative ensemble algorithm. This is a fusion algorithms having two layered architecture. All algorithms at the first layer are trained on the given training set, and then a combiner algorithm at the second layer is trained using predictions of the first level as inputs. This method is used for increasing the accuracy from weak learners. The combiner algorithm is termed as Meta learner. Instead of output from the base classifier, the probability predictions [30] can also be passed as input to the meta layer

Algorithm

Input: $\mathcal{M} = \{m_1, m_2, m_3, \dots, m_k\}$ Machine learning classifiers at the first layer

$\mathcal{X} = \{x_1^d, \dots, x_n^d\}$ n Training set of dimension d

$\mathcal{Z} = \{ z_1^d, \ldots, z_p^d \}$ p Intermediate test set of dimension d

$M =$ Meta Learner

Output: Prediction of a m test vector each of dimension d $\mathcal{Y} = \{ y_1^d, \ldots y_m^d \}$

$\mathcal{X}'$ meta training set for meta learning in second layer

$\mathcal{Y}'$ meta test for meta learning in second layer

For each row r in $\mathcal{Z}$:

I = Instance vector

For each classifier m in $\mathcal{M}$:

p = Train the classifier with $\mathcal{X}$ and predict on row vector r

Add prediction p to I

Add I to $\mathcal{X}'$

For each row r in $\mathcal{Y}$:

I = Instance vector

For each classifier m in $\mathcal{M}$:

p = Train the classifier with $\mathcal{X}$ and predict on row vector r

Add prediction p to I

Add I to $\mathcal{Y}'$

$prediction$:Train meta learner $M$with meta training set $\mathcal{X}'$ and predict on $\mathcal{Y}'$

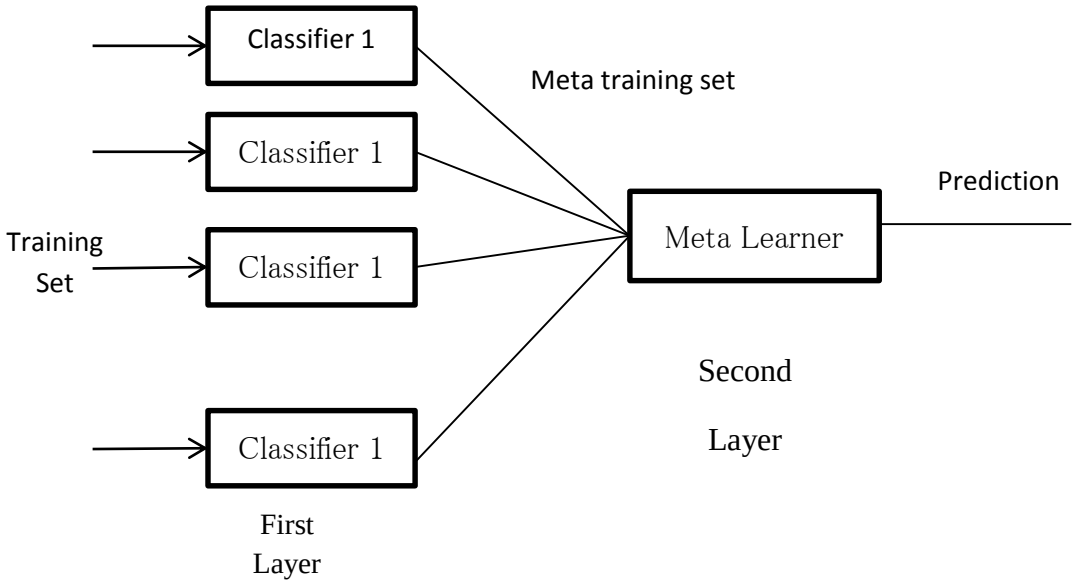


Figure 3.4.1 Stacking method

3.4.4 Bagging

It stands for bootstrap aggregating. Different training data subset are randomly generated from the training set. Each subset training data is trained by a different classifier. Then simple majority voting of each output from each classifier is defined as the output of this ensemble model.

Algorithm:

Input:

- Training dataset with class labels $\mathfrak{w}_1, \mathfrak{w}_2 \ldots \mathfrak{w}_c$ c classess
- Weak learning algorithms $\mathfrak{m}_1, \mathfrak{m}_2, \mathfrak{m}_3 \ldots \mathfrak{m}_T$
- T represents the number of interations
- Percent (or Fraction) F to create bootstrapping training data

for t = 1 to T

Take bootstrapped replica S_t by randomly drawing F percent of S.

Build the classifier $\mathfrak{m}_t$ with S_t as hypothesis h_t

Add hypothesis h_t to ensemble ξ

Test: Test Set X

for each row $x_t in X$

Evaluate the ensemble ξ *on* x_t

Let $v_{t,j} = \begin{Bmatrix} 1\,, & if\ h_t picks\ class\ \varpi_j \\ 0, & Otherwise \end{Bmatrix}$

Total vote received by each class $V_j = \sum_{t=1}^{T} v_{t,j}\ j = 1 \ldots c$

Choose the class that receives highest vote as the final classification

Chapter 4

Proposed Models and Design

4.1 Problem Analysis

Stock prediction is a challenging problem, it has been extensively analyzed by different mathematical models and machine learning techniques. Due to its non-linear dependency on many factors like volatility, time-series, economic situation and many more. Modeling this problem mathematically is non-trivial. However minimizing some risk, an acceptable accuracy can be achieved. Neural networks can be applied due to its non-linearity. Fuzzy system had also been applied for stock prediction problem. State-of-art model like support vector machines have shown high accuracy rate for stock prediction.

In this paper we are trying to solve this problem using ensemble techniques. Ensemble methods are methods that combine a set of machine learning techniques either linear or non-linear way to give one single predictions. Ensemble techniques are believed to give better accuracy than individual classifiers. Ensemble methods are similar to classifiers, it attempts to extract patterns from the data and translate this pattern to usable knowledge. According to Dietterich [1], following reason shows why ensemble models are better than individual classifiers:

1. A learning algorithm finds out a hypothesis *h* from an set of hypothesis H, by taking ensemble of all the classifiers we can get an good approximation of our true hypothesis

2. In many problems, a true hypothesis cannot be represented by any of hypothesis of H, by taking linear or non-linear combination of many hypothesis, it may be possible to get accuracy close to our true hypothesis.

Stock price movement prediction deals with forecasting the next momentum of any stock, whether increase or decrease. In simple terms we are categorizing the next price movement into two classes, this can be transformed to a binary classification problem and try to solve it using machine learning technique.

. 4.2 Objectives

Our goal is to develop a stock prediction model that correctly forecasts the future momentum of the stocks of commodity market by analyzing historical data.

In order to achieve this goal, we have laid down a list of sub-goals that help to achieve our final goal.

* Identifying various machine learning algorithms that can forecasts stock price

* Identifying various ensemble models that can be used for stock prediction

* integrate various methods and examine different ensemble models

The scope of this thesis is limited to Indian stocks in commodity market

4.3 Proposed Model

We proposed an ensemble model to predict the price movement of a stock. Initially we examine different machine learning techniques. Choosing best 9 methods we will apply to our ensemble models. We will be examining various generative and non-generative ensemble models to get best prediction model. The proposed model is shown below:

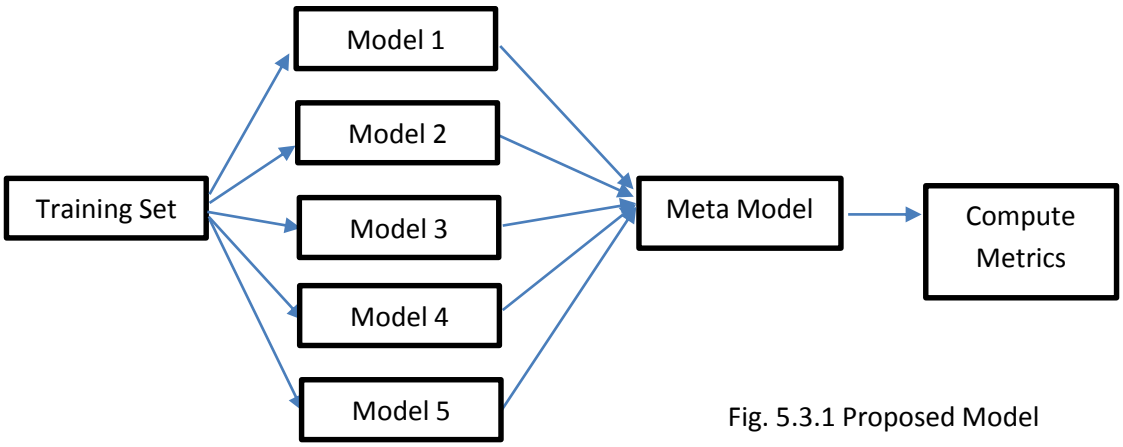


Fig. 5.3.1 Proposed Model

4.4 System Requirement

Hardware Processor Intel i3, Ram 4 +GB, Storage 160 GB, Frequency 2.93 GHz.

Operating System Ubuntu 12.04 LTS. 32 bit.

Programming Language JAVA, Python

IDE Programming is done in Eclipse IDE environment.

WEKA 3.6.11 Machine Learning Package [32]

4.5 Dataset

The dataset for our thesis have been provided by AlgoAnalytics Financial Consultancy Pvt. Ltd [31]. It is a proprietary dataset comprising of three different stocks from commodity market. The dataset are namely CDATA, FDATA, GDATA, NDATA, ODATA and SDATA. In each dataset, the columns are normalized with mean as 0 and standard deviation as 1. The features of each dataset are trade secrets, thus it can't be disclosed. Following table gives data description about each dataset.

Dataset	Number of Instances	Number of Features
CDATA	2073	10
FDATA	1740	10
GDATA	2129	9
NDATA	2610	24
ODATA	2077	14
SDATA	2126	10

Table 3.1: Information about all datasets.

Chapter 5

Experiments and Models

Model 1: Using Vertical Voting with support vector machine as classifier

We choose vertical voting approach, combined three different dataset representation using Denoising Autoencoder. Table 5.1.1 shows the result.

Dataset	Accuracy	F1_measure	Harmonic accuracy	Uncertainty coefficient
CDATA	0.4782	0.4696	0.4890	0.7909
FDATA	0.4867	0.4724	0.4850	0.4943
GDATA	0.4900	0.4917	0.4994	0.4959
NDATA	0.4978	0.4953	0.4962	0.4990
ODATA	0.4720	0.4573	0.4842	0.4886
SDATA	0.4755	0.4608	0.4845	0.4899

Table 5.1.1: Vertical Voting performance

Though deep learning seems to be a promising machine learning technique but we got bad results in our dataset.

Model – 2: Value of Individual Classifiers

We have taken training set as 300,400,500 to find the best possible training size. Here we have taken each individual classifier and followed walk forward mode. Table 5.2.1 shows performance for CData and FData dataset .

Using Individual machine learning algorithms, Table 5.2.2 shows performance for GData and NData with different training size of 300,400,500 with walk forward approach.

Using Individual machine learning algorithms, Table 5.2.3 shows performance for OData and SData with different training size of 300,400,500 with walk forward approach.

	CData			FData		
	Training Data			Training Data		
Classifiers	300	400	500	300	400	500
Multilayer Perceptron	0.5491	0.5444	0.5369	0.5405	0.5492	0.5439
Logistic	0.5480	0.5391	0.5509	0.5839	0.5803	0.5853
SVM	0.5378	0.5299	0.5363	0.5482	0.5446	0.5585
VotedPerceptron	0.5374	0.5402	0.5388	0.5657	0.5583	0.5739
kNN	0.5322	0.5327	0.5356	0.5384	0.5265	0.5512
Logistic Modeled Trees	0.5332	0.5372	0.5312	0.5636	0.5666	0.5617
SimpleLogistic	0.5282	0.5391	0.5388	0.5643	0.5560	0.5674
J48	0.5271	0.5238	0.5273	0.5517	0.5568	0.5560
NaiveBayes	0.5203	0.5234	0.5222	0.5370	0.5590	0.5495
DecisionTable	0.5169	0.5057	0.5048	0.5510	0.5500	0.5601

Table 5.2.1 Performance of CData and FData on different training size

Classifiers	GData			NData		
	Training Size			Training Size		
	300	400	500	300	400	500
Multilayer Perceptron	0.5357	0.5470	0.5283	0.5421	0.5206	0.5161
Logistic	0.5505	0.5447	0.5506	0.5482	0.5537	0.5623
Smo (SVM)	0.5532	0.5534	0.5444	0.5456	0.5464	0.5528
VotedPerceptron	0.5357	0.5302	0.5228	0.5360	0.5342	0.5376
kNN	0.5148	0.4976	0.5074	0.4991	0.4997	0.5014
Logistic Modeled Trees	0.5483	0.5558	0.5438	0.5400	0.5487	0.5528
SimpleLogistic	0.5510	0.5581	0.5419	0.5365	0.5482	0.5519
J48	0.5373	0.5372	0.5372	0.5178	0.5197	0.5304
NaiveBayes	0.5425	0.5273	0.5382	0.5008	0.5052	0.4995
DecisionTable	0.5527	0.5581	0.5512	0.5300	0.5265	0.5200

Table 5.2.2 Performance of GData and NData on different training size

	OData			SData		
	Training Size			Training Size		
Classifiers	300	400	500	300	400	500
Multilayer Perceptron	0.5282	0.5431	0.5363	0.5401	0.5505	0.5512
Logistic	0.5576	0.5694	0.5668	0.5582	0.5645	0.5685
Smo (SVM)	0.5593	0.5616	0.5681	0.5685	0.5500	0.5524
VotedPerceptron	0.5435	0.5485	0.5541	0.5763	0.5523	0.5629
kNN	0.5214	0.5179	0.5264	0.5280	0.5220	0.5308
Logistic Modeled Trees	0.5661	0.5832	0.5675	0.5532	0.5819	0.5759
SimpleLogistic	0.5638	0.5838	0.5745	0.5576	0.5645	0.5777
J48	0.5502	0.5401	0.5363	0.5593	0.5622	0.5691
NaiveBayes	0.5395	0.5383	0.5305	0.5461	0.5461	0.5345
DecisionTable	0.5350	0.5221	0.5267	0.5565	0.5779	0.5648

Table 5.2.3 Performance of OData and SData on different training size

Model 3: Optimized Result

As we see with training data size of 500, all machine learning algorithms perform good for each dataset. So we further optimize each classifier and calculate result with training data size 500 and testing data size 1 following the walk forward approach. Table 5.3.1 shows the result.

Model 4: Majority Voting

MajorityVoting taking best 3 classifier based on accuracy with training size 500 and test size 1 with walk forward approach. Table 5.4.1 shows the results.

MajorityVoting taking best 5 classifier based on accuracy with training size 500 and test size 1 with walk forward approach. Table 5.4.2 shows the results.

MajorityVoting taking best 7 classifier based on accuracy with training size 500 and test size 1 with walk forward approach. Table 5.4.3 shows the results.

	CData	FData	GData	NData	OData	SData
Logistic	0.5509	0.5853	0.5506	0.56	0.5668	0.5685
VotedPerceptron	0.5509	0.5829	0.5339	0.5452	0.5624	0.5654
SimpleLogistic	0.5509	0.5813	0.5469	0.55	0.5668	0.5746
Locally Weighted Learning	0.5420	0.5780	0.5401	0.5490	0.5789	0.5685
SVM	0.5420	0.5756	0.5370	0.5627	0.5560	0.559
Multilayer Perceptron	0.5337	0.5731	0.5370	0.5352	0.5414	0.5574
Logistic Modeled Trees	0.5382	0.5707	0.5487	0.5447	0.5668	0.5808
kNN	0.5312	0.5617	0.5228	0.5185	0.5350	0.5401
BayesNet	0.5038	0.5585	0.5512	0.52	0.5267	0.5709
Decision Table	0.5050	0.5585	0.55	0.52	0.5292	0.5648
NaiveBayes	0.5038	0.5585	0.5501	0.52	0.5267	0.571
J48	0.5273	0.5560	0.5425	0.5304	0.5363	0.569

Table 5.3.1 Optimized algorithms using training size of 500

Dataset	**Accuracy**	**Harmonic Accuracy**	**F1_score**	**Uncertainty Score**
CData	0.5212	0.6182	0.6203	0.5099
FData	0.5844	0.6572	0.5980	0.5421
GData	0.5528	0.7111	0.7111	0.5373
NData	0.5619	0.5932	0.5376	0.5321
OData	0.575	0.6037	0.5353	0.5403
SData	0.5814	0.5814	0.6586	0.5397

Table 5.4.1 Majority voting with 3 classifier

Dataset	**Accuracy**	**Harmonic Accuracy**	**F1_score**	**Uncertainty Score**
CData	0.55	0.6151	0.5787	0.5235
FData	0.5827	0.6535	0.5948	0.5415
GData	0.5503	0.7097	0.7097	0.5068
NData	0.5538	0.5837	0.5352	0.5274
OData	0.5692	0.5980	0.5350	0.5367
SData	0.5808	0.6572	0.6031	0.5394

Table 5.4.2 Majority voting with 5 classifier

Dataset	Accuracy	Harmonic Accuracy	F1_score	Uncertainty Score
CData	0.5378	0.6027	0.5764	0.5172
FData	0.5819	0.6521	0.5938	0.5411
GData	0.5469	0.6941	0.6922	0.5200
NData	0.5619	0.5932	0.5376	0.5321
OData	0.5706	0.6004	0.5365	0.5375
SData	0.5790	0.6523	0.5984	0.5388

Table 5.4.3 Majority voting with 7 classifier

Hybrid Majority Voting: We follow the hybrid majority voting approach with training data size as 500 and test size as 1 with walk forward approach. Table 5.4.4 shows the result.

Dataset	Accuracy	Harmonic Accuracy	F1_score	Uncertainty Score
CData	0.5561	0.7033	0.7124	0.5297
FData	0.5959	0.6691	0.6024	0.5429
GData	0.5565	0.7135	0.7104	0.2797
NData	0.5548	0.6985	0.5325	0.5283
OData	0.5882	0.5660	0.5331	0.5312
SData	0.5900	0.6649	0.6041	0.5451

Table 5.4.4 Hybrid Majority voting

Model 5: Stacking

Taking 9 classifier in the first layer, they are:

- Logistic
- SMO
- VotedPerceptron
- Multilayer Perceptron
- kNN
- Logistic Modeled Trees
- J48

- Simple logistic
- Naïve Bayes

We are taking the above classifier at first level, with training_data size as 500, intermediate test_size as 200 and final test_data size as 1 with walk forward approach. Logistic Regression as the meta learner. Instead of taking outputs from the classifier at first layer as the input to second layer, we are taking confidence or probabilistic prediction distribution. Table 5.5.1 shows the result.

Dataset	**Accuracy**	**Harmonic Accuracy**	**F1_score**	**Uncertainty Score**
CData	0.6071	0.6269	0.5265	0.5625
FData	0.5804	0.6036	0.5291	0.5441
GData	0.7688	0.6	0.5384	0.5366
NData	0.5620	0.5764	0.5169	0.5327
OData	0.5688	0.6	0.5384	0.5366
SData	0.6631	0.6927	0.5480	0.6055

Table: 5.5.1 stacking with probabilistic prediction

Taking the output of the individual base classifier that is 1 or -1 as the input to Meta classifier. Here we have Logistic as the Meta classifier, with training data size as 500, intermediate test size as 200 and final test data size as 1 with walk forward approach. Table 5.5.2 shows the result.

Dataset	**Accuracy**	**Harmonic Accuracy**	**F1_score**	**Uncertainty Score**
CData	0.6359	0.657	0.5308	0.5841
FData	0.6809	0.7067	0.5439	0.6213
GData	0.8512	0.8641	0.5476	0.8010
NData	0.6225	0.6447	0.5309	0.5739
OData	0.5644	0.59166	0.5328	0.5340
SData	0.7075	0.7353	0.5523	0.6453

Table: 5.5.2 Stacking with Outputs

Model 6: Bagging

We have applied rotation forest with bagging. As subset training dataset are made, they are given into rotation forest ensemble. Rotation forest further divide the subsets into small subsets in which principal component analysis is performed. We applied the walk forward approach with training data size of 500 and test data size of 1. Table 5.6.1 shows the output on various datasets.

Dataset	**Accuracy**	**Harmonic Accuracy**	**F1_score**	**Uncertainty Score**
CData	0.6059	0.6300	0.5335	0.5616
FData	0.617	0.6544	0.5524	0.5696
GData	0.8180	0.8337	0.5469	0.7617
NData	0.5725	0.5957	0.5283	0.5391
OData	0.5070	0.5493	0.5451	0.5032
SData	0.6366	0.6722	0.5538	0.5839

Table 5.6.1 Bagging with rotation forest.

We have applied random subspace with bagging. As subset training dataset are made, they are given into random subspace ensemble. Random subspace divides the dataset into small training data subset, and for each subset randomly number of features are selected. These subset are fed into individual classifiers. Final predictions are done using majority voting of each classifiers. We applied the walk forward approach with training data size of 500 and test data size of 1. Table 5.6.2 shows the output on various datasets.

Dataset	**Accuracy**	**Harmonic Accuracy**	**F1_score**	**Uncertainty Score**
CData	0.7167	0.7352	0.5348	0.6550
FData	0.7390	0.7640	0.5524	0.6764
GData	0.7851	0.7809	0.4904	0.7259
NData	0.5868	0.6137	0.5341	0.5484
OData	0.5809	0.6153	0.5434	0.5444
SData	0.7641	0.7894	0.5593	0.7020

Table 5.6.2 Bagging with random subspace.

Chapter 6

Result Analysis

DeepLearning showing very bad accuracy. Following may be some reasons

1. The more number of neurons are needed in a layer to fully encode the feature of a data
2. Should have enough training iterations for each layer.

If the input vector is a poor representation of the data, so poor features are learned by the auto encoder, thus hidden layer does not learn any pattern at the end of the calculation. Thus we have better features defined for our dataset, to extract better patterns from out dataset.

Best individual Classifier for each dataset, varying the size of training data and optimized dataset is shown in Table 6.1.1

	Classifier	Accuracy	Training size
CData:	Logistic	0.550	500
FData	Logistic	0.5853	500
GData	Decision Table	0.5581	400
NData	Logistic	0.5623	500
OData	Simple Logistic	0.5838	400
SData	Logistic Modeled Trees	0.5819	400

Table 6.1.1 Best Individual Classifier

Comparing different majority voting approach. Hybrid majority voting approach is giving better accuracy for every dataset, only Ndata majority voting with 5 classifier is showing better accuracy.

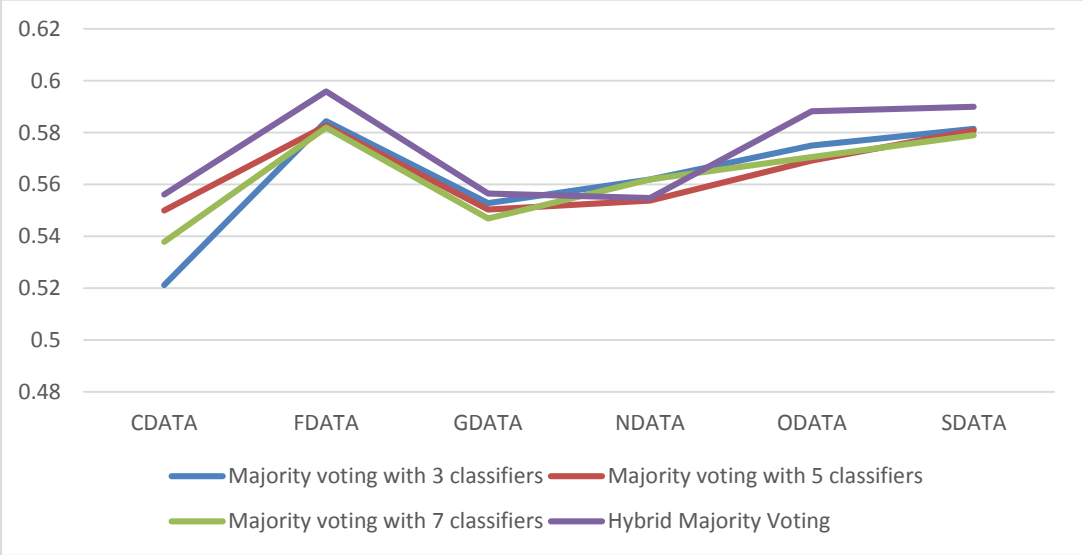


Figure: 6.1 Majority Voting comparison

Stacking :

As we can see in Figure:6.2 taking probability distribution as output is showing better accuracy than taking outputs of each classifer

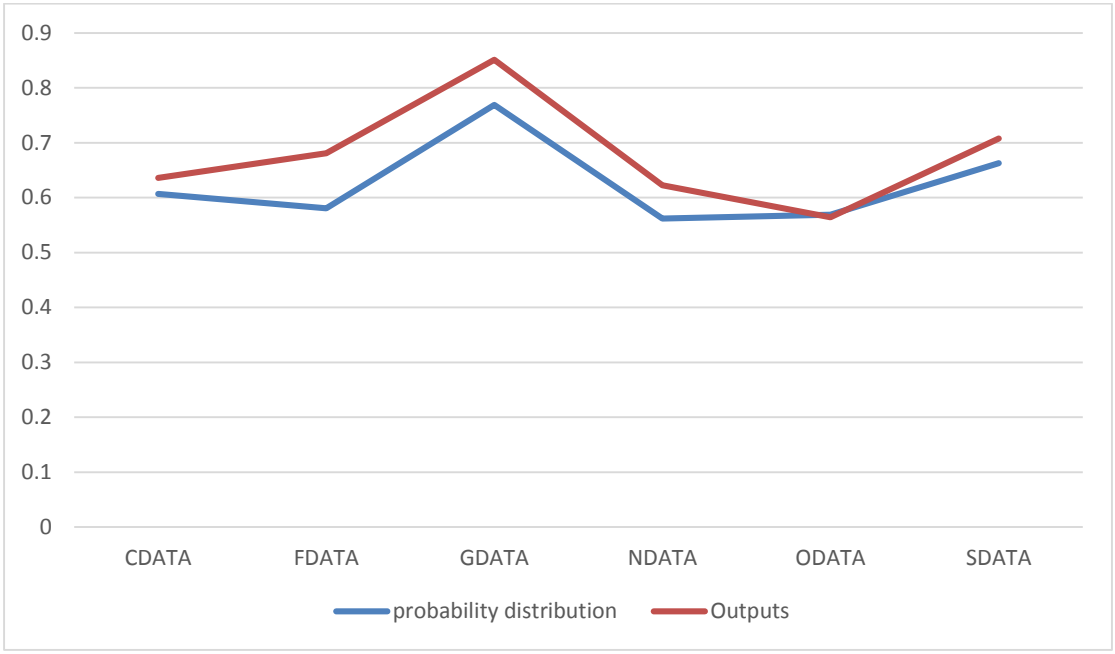
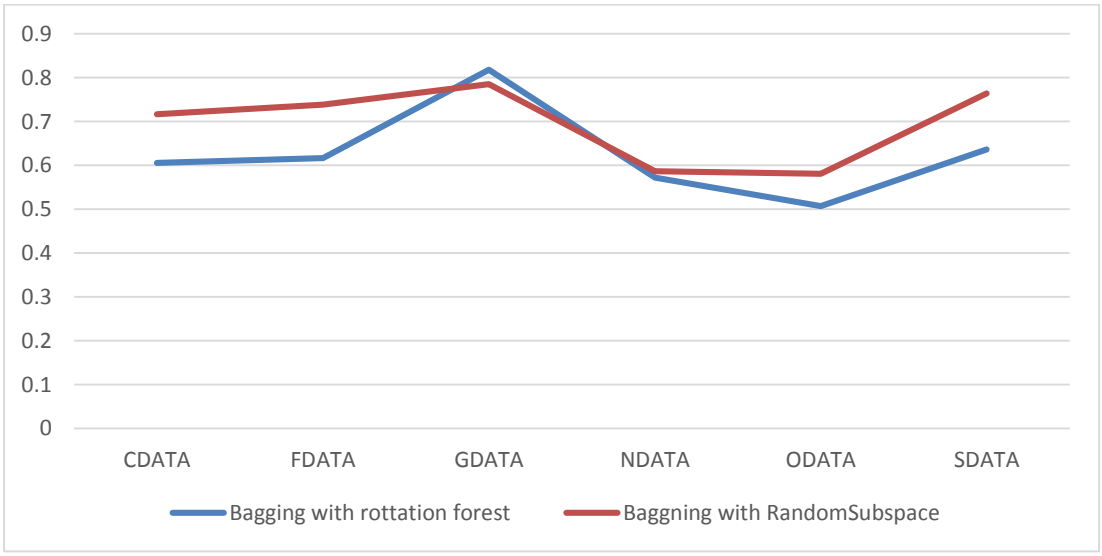


Figure: 6.2 Stacking comparison

Bagging:

As we can see in Figure 6.Bagging with Random subspace gives better accuracy than bagging with rotation forest except for Gdata.



:

Figure: 6.3 Bagging comparison

Best prediction model for each datasets:

Bagging with random subspace outperforms every other models, however for gdata stacking with Output is the best prediction model as shown in Figure 6.4

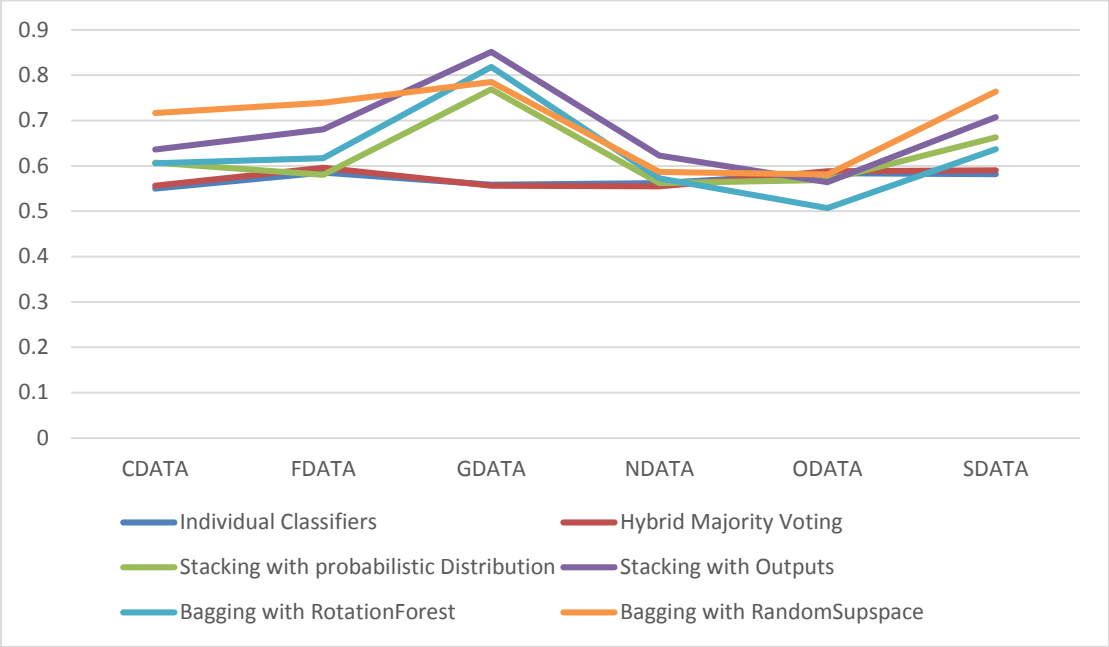


Figure 6.4 Best prediction model

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

All ensemble method has shown better accuracy than individual methods. Majority Voting is the most used ensemble method and it is guaranteed to give higher accuracy than individual classifier only if the classifiers are independent.

Stacking methods has worked significantly good with Logistic machine learning technique as meta classifier. Bagging with Random subspace outperforms all the methods except for gdata.

With this prediction accuracy, we can certainly make profitable trading decisions and earn money.

5.2 Future Scope

The future work can comprise of implementing more unique ensemble models to have better accuracy. Great amount of research is done on designing ensemble models. Better features should be designed to increase the accuracy of stock prediction. More accuracy can be achieved if we tune the hyperparameters of different machine learning algorithms and ensemble models.

Deep learning is a hot topic, sooner or later it will be applied to stock prediction.

Bibliography

[1] T. G. Dietterich, "Ensemble Methods in Machine Learning," in *Proceedings of the First International Workshop on Multiple Classifier Systems*, London, Springer-Verlag, 2000, pp. 1-15.

[2] Matteo Re, Giorgio Valentini and Oleg Okun, Ensembles in Machine Learning, Berline: Springer-Verlag, 2011.

[3] J. Brofos, "Ensemble Committees for Stock Return," 4 April 2014. [Online]. Available: http://arxiv.org/abs/1404.1492.

[4] John Z. Zhang and Khobaib Zaamout,, "Improving Neural Networks Classification through Chaining," in *Artificial Neural Networks and Machine Learning – ICANN 2012*, Berlin Heidelberg, Springer, 2012, pp. 288-295.

[5] Stuart, Nikunj C. Oza and Russell, "Online Bagging and Boosting," in *Eighth International Workshop on Artificial Intelligence and Statistic*, Florida, Morgan Kaufmann, 2001, p. 105–112.

[6] A. Upadhyay, G. Bandhopadhyay and A. Dutta, "Forecasting Stock Performance in Indian Market using Multinomial," *Journal of Buisness Studies*, vol. 3, no. 3, pp. 16-39, 2012.

[7] Tongda Zhang, Haomiao Jiang and Shunrong Shen, "Stock Market Forecasting Using Machine Learning Algorithm," 2012.

[8] Jingwei, Ye Nan and Chen Ming, "Forecasting the Direction and Strength of Stock Market Movement," 2013.

[9] In-Chan Choi and Yanshan Wang,, "Market Index and Stock Price Direction Prediction using Machine Learning Techniques," Elsevier Ltd, Korea, 2013.

[10] Hung Pham, Andrew Chien and Youngwhan Lim, "A Framework for Stock Prediction," Stanford, 2011.

[11] Khalid Alkhatib, Hassan Najadat and Ismail Hmeidi, "Stock Price Prediction Using K-Nearest Neighbor (kNN) Algorithm," *International Journal of Business, Humanities and Technology*, vol. 3, no. 3, pp. 32-44, 2013.

[12] Yan-Mi, David C. Yen,, Yuah-Chiao Lin, and Chih-Fong Tsai, , "Predicting stock returns by classifier ensembles," *Applied Soft Computing*, vol. 11, no. 2, pp. 2452-2459, 2011.

[13] Ritanjali Majhi, G. Panda, G. Sahoo and Abhishek Panda, "Prediction os S&P 500 and DJIA Stock Indices Using Particle Swarm Optimization Technique," IEEE, Rourkela, 2008.

[14] K.K.Sureshkumar and Dr.N.M.Elango,, "An Efficient Approach to Forecast Indian Stock Market Price and their Performance Analysis," *International Journal of Computer Applications,* vol. 34, no. 5, pp. 4103-5942, 2011.

[15] B. R. Moon and Y. K. Kwon, "Evolutionary Ensemble for Stock Prediction.," in *Genetic and Evolutionary Computation Conference*, Seattle, 2004,.

[16] Dr. A. Chitra and S. Uma, "An Ensemble Model of Multiple Classifiers for Time Series Prediction," *International Journal of Computer Theory and Engineering,*, vol. 2, no. 3, pp. 454-458, 2010.

[17] Ajith Abraham and Crina Grosan, , "Ensemble of Genetic Programming Models for Designing Reactive Power Controller," *Proceedings of the Fifth International Conference on Hybrid Intelligent Systems,* pp. 277-284, 2005.

[18] ,Ajith Abraham, Bo Yang,Jiwen Dong and Yuehui Chen,, "Time-series forecasting using flexible neural tree model," *Information Sciences: an International Journal,* vol. 174, no. 3-4, pp. 219-235, 2005.

[19] P, Salamon and Hansen L. K , "Neural network ensembles," *IEEE Trans. Pattern analysis and machine intelligence,* vol. 12, no. 10, pp. 993-1001, 1990.

[20] V. Shah, "Machine Learning Techniques for Stock Prediction," vatsals , [Online]. Available: http://www.vatsals.com/Work.aspx.

[21] P. Melin, O. Castillo and M. Pulido, "GENETIC OPTIMIZATION OF ENSEMBLE NEURAL NETWORKS," *International Journal of Innovative,* vol. 9, no. 10, pp. 4151-4166, 2013.

[22] Zhen Liu, Yuehui Chen and Qiang Wu, "Ensemble model of Intelligent Paradigms for Stock Market Forecasting," in *Workshop on Knowledge Discovery and Data Mining*, 2008.

[23] Yan-Mi, David C. Yenb,, Chiao Linc, and Chih-Fong Tsaia, , "Predicting stock returns by classifier ensembles," *The Impact of Soft Computing for the Progress of Artificial Intelligence,* vol. 11, no. 2, p. 2452–2459, 2011.

[24] J. H. Satish Rao, "Analysis of Hidden Markov Models and Support," Berkley, 2010.

[25] Jianfeng Si,, Arjun Mukherjee, Bing Liu, Qing Li and Huay, "Exploiting Topic based Twitter Sentiment for Stock Prediction," ACL, 2013.

[26] Hsinchun, Robert P. Schumaker and Chen, "Textual Analysis of Stock Market Prediction Using Breaking," in *Conference on Information Systems*, 2006.

[27] Simon Osinder, Yee-Whye Teh and Geoffrey E. Hinton, , "A fast learning algorithm for deep belief nets," in *Neural Computation*, 2006.

[28] H. L. Y. B. P.-A. M. Pascal Vincent, Pascal Vincent, Hugo Larochelle and Yoshua Bengio, "Extracting and Composing Robust Features with Denoising AutoEncoder," Universit´e de Montr´eal, 2008.

[29] ,Zhang chuang, Bing xu and Jingjing Xie, "Horizontal and Vertical Ensemble with Deep Representation for Classification," in *ICML workshop*, 2013.

[30] ZENKO and DZEROSKI , "Is Combining Classifiers with Stacking Better than selecting the best one," in *Machine Learning*, Netherland, 4 Kluwer Academic Publishers, 2004, p. 255–273.

[31] A. Pant, *AlgoAnalytics Financial Consultancy Pvt Ltd.*

[32] Mark Hall, Eibe Frank, Geoffrey Holmes and Bernhard P, *The WEKA Data Mining Software: An Update.*