Intraday Stock Trend Prediction Using Sentiment Analysis

A dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Technology

in

Computer Science

by

Bhaskar Tiwari

under the guidance of

Dr. Diganta Mukherjee

Associate Professor

Sampling and Official Statistics Unit

Indian Statistical Institute

Kolkata-700108, India

July 2018

CERTIFICATE

This is to certify that the dissertation titled **"Intraday Stock Trend Prediction Using Sentiment Analysis"** submitted by **Bhaskar Tiwari** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bona fide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Diganta Mukherjee

Associate Professor,

Sampling and Official Statistics Unit,

Indian Statistical Institute,

Kolkata-700108, INDIA.

Acknowledgments

I would like to express my heartfelt gratitude to my advisor, **Dr. Diganta Mukherjee**, Sampling and Official Statistics Unit, Indian Statistical Institute, Kolkata, for his gracious encouragement and valued constructive criticism that has driven me to complete the dissertation successfully.

I would also like to thank my parents and my sister for their continued support and encouragement.

Abstract

Traditionally, most of the studies involved in the prediction of stock price movement, using machine learning techniques, have utilized the historical stock price data and the technical indicator extracted from the data. However, such strategies do not use information about live news events that affect the stock prices. In this thesis, we focus on the publicly available tweets on twitter as a source of real time news. We extract the sentiment information from the tweets and use them (along with the stock data) to predict the intra-day price movement in the Indian stock market.

We design a sentiment analyzer that identifies the sentiment of each tweet and sorts the tweet into one of three clusters (Trade, Feedback, and Miscellaneous). We modify our sentiment features using a cluster importance factor, a score that quantifies the importance of each clusters in the context of market importance, and use these modified features for price movement prediction via machine learning models. At the end of this thesis, we show that our trained models consistently outperform the random walk baseline accuracy of 33% for three-way stock trend classification and that the cluster importance factor improves the prediction accuracy of the random forest model.

Keywords: *Stock prediction, sentiment analysis, machine learning, random forest*

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Can we consistently predict the movement of stock prices in the market? This question has garnered continued attention from academia and industry, because of the potential ramifications of a positive answer. However, there are a few studies that discredit this notion of stock market predictability. Fama's *efficient market hypothesis*[9] and Malkiel's *random walk theory*[16] of asset price movement cast serious doubts on the ability to predict stock prices.

Behavioural finance posits the idea that the stock market is an aggregate of all human actions, their greed and their irrationality[21]. It directly challenges one of the assumptions behind the efficient market hypotheses, such as the inherent rationality of investors in the market[22]. In 2004, Andrew Lio[15] introduced the *adaptive market hypothesis*, which states that opportunities for arbitrage exist because of the inefficiencies present in the market, but these opportunities grow smaller as information about inefficiencies disseminate among the public. These theories have fuelled a lot of research on stock price prediction using machine learning models.

Traditionally, most of the studies involved in the prediction of stock price movement using machine learning techniques have utilized the historical stock price data and the technical indicator extracted from the data[[24], [4], [13]]. However, such strategies have the disadvantage of not using information about live news events, that affect the stock prices[12]. In the last 10 years, a lot of research regarding stock prediction have focussed on this "real- world news" component to augment the technical analysis of stocks. E. Junqué de Fortuny et al.[7] and Leung[14] used the sentiment information extracted from different "real-world news" corpus along with technical indicator to predict day-end stock prices (returns).

In this thesis, we focus on the publicly available tweets on twitter as a source of real time news. We extract the sentiment information from the tweets and use them (along with the stock data) to take to predict the price movement. There have been various studies documenting the correlation between the sentiment expressed in tweets and stock performance. The most famous among them is by Bollen, Mao, Zeng[2]. They showed that the public sentiment extracted from twitter feeds is strongly correlated with the Dow Jones Industrial Average and using such sentiment information appreciably increases the prediction accuracy over standard models. A recent work by Pagolu, Raddy, panda, Majhi[20] utilizes a machine learning based sentiment analyzer to find the tweet sentiment class (positive, negative , or neutral). They use this sentiment information to predict the future stock prices for Microsoft. In our work, we use the sentiment analysis phase presented in this paper as the basis of our work and introduce a way to identify and segregate the different categories of tweets in order to get effective sentiment features.

We focus our attention on the prediction of intra-day price trends (30 minute intervals) in the Indian stock market using the sentiment information derived from twitter data. In recent years, there have been a number of studies focussed on intra-day trend prediction using sentiment in stock markets. Geva, Zahavi[10], in one such study, examined the sentiment scores based on textual news data in addition to the technical features analyzed from stock data to forecast returns (better than S&P index).

1.2 Our Contribution

In this thesis, we use machine learning techniques to predict the direction of stock price movement at every 30 minute interval. We use both information about stock prices (order book) and public sentiment (twitter data) for our analysis. Our contributions are summarized as follows:

- We design a sentiment analyzer that identifies the sentiment of each tweet and sorts the tweet into one of three clusters (Trade, Feedback, and Miscellaneous).
- We introduce the concept of cluster importance factor, a score that quantifies the importance of each clusters in the context of market importance.
- We propose a method to integrate the cluster importance factor with the sentiment features derived from twitter data. This gives us the final sentiment features. We use these sentiment features along with the stock price features in ML models- SVM and random forest- for price movement prediction.
- We show that both our trained models consistently outperform the random walk baseline accuracy of 33% for three-way stock trend classification (up, down, no change).
- We show that the cluster importance factor improves the prediction accuracy of the random forest model.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2: This chapter provides an outline of the data preparation process- data collection, pre-processing, and feature extraction- process for stock and sentiment data.
- Chapter 3: This chapter discussed the process of sentiment analysis via ML models and the post-processing of sentiment data for stock trend analysis.
- Chapter 4: This chapter examines the role of cluster importance factor and the performance of ML algorithms- SVM and Random forest- on the stock trend prediction task.
- Chapter 5: This chapter concludes our analysis and outlines potential opportunities for improvement in future work.

Chapter 2

Data Preparation

This chapter provides an outline of the data preparation process for stock and sentiment data. In particular, we examine the pre-processing and the feature extraction phases of the data preparation process.

In this thesis, we use twitter data for sentiment analysis and the order book data for stock price information. For the purpose of analysis, we focus our attention on three of the biggest financial institutions listed on the NSE, namely, ICICIBANK, AXISBANK, and HDFCBANK. We collect order book and twitter data for these stocks for the 3-month period, 1st September 2017 to 30th Novemeber 2017.

2.1 Twitter Data

2.1.1 Data Collection

The twitter dataset used in this thesis is collected using a custom crawler, which relies upon twitter's native search functionality to extract the relevant tweets. The crawler fetches data from twitter based on relevant search keywords. We used the name of the company and the shortened form of the company name (eg. ICICI for ICICBANK) as relevant keywords in our data collection strategy.

We extracted a total of 1,19,116 tweets. An overview of the breakdown of the total tweets amongst the different stocks is given in Table 2.1.

	AXISBANK	**ICICBANK**	**HDFCBANK**
Tweets	22884	49357	46875

Table 2.1: An overview of the breakdown of the total tweets amongst the different stocks

2.1.2 Data Pre-Processing

The tweets obtained from the data collection phase have a lot of undesirable parts, such as pictures, email address, url links, among others that need to be removed. We also need to remove stop words (a, an, the, is, etc). These words occur very frequently in the language, but they don't express any emotion. We employ a two component pre-processor for cleaning the tweets:

- Tokenizer: This splits the tweet into its individual component phrases based on spaces and stores the phrases in a list. The tokenizer also removes any stop words from the phrase list.
- Reg-ex Cleaner: This takes the output phrase list from the tokenizer and removes the url links, pictures, email addresses, and user mentions (@) from the phrase list. The reg-ex cleaner returns a list of relevant words that make up the tweet.

2.1.3 Feature Extraction

We use a Sent2Vec[19] model, which is a modified version of the Continuous Bag of Words[[18], [17]] model, to get the textual (semantic) representation of the tweets. We provide a brief overview of the model below.

A sent2vec model can be viewed as a fully connected neural network with one hidden layer (linear activation). The size of the hidden layer corresponds to the dimensionality of our final text representation (100 in our case). The input and output layers have the same number of neurons as the vocabulary size. The weight matrix between the input and the hidden layer gives us the textual representation.

In a continuous bag of words (CBOW) model, the neural network tries to learn the representation of each word by looking at the surrounding words in the sentence. A CBOW model that looks at all the remaining words in the sentence in order to get the representation of the target word works similarly to our sent2vec model. One major difference between sent2vec and CBOW is that in sent2vec, we learn the text embedding for all the n-grams in the sentence, and not just uni-grams. The text representation of a sentence is the average of the text embeddings of its constituent n-grams (including uni-grams).

This model gives us a 100 dimensional feature vector for each tweet. We use these features for the purpose of sentiment analysis in Chapter 3.

2.2 Stock Price Data

2.2.1 Data Source

We use order book data for stock price information. The "order book" has information about all the active buy (bid) and sell (ask) requests for that stock in the market (along with the quantities). In addition, the order book has information about the last traded price of the stock. In the context of an order book, a trade occurs when the best bid (highest price buy order) and best ask (lowest price sell order) prices match. The order book is a very high frequency data source, with each 6 hour trading day contributing about 15000 data points.

2.2.2 Data Pre-Processing

We down sample our order book data in order to collect samples at every 30 minutes interval, during the regular trading hours.

2.2.3 Feature Selection

In our work, we choose the top 5 bid and the top 5 ask prices in the order book as our stock price features. Our class label, $\mathrm{y_i}$ is defined as follows:

- $\mathrm{y_i} = 1$, if next interval's "last traded price" is greater than the current "last traded price".
- $\mathrm{y_i} = -1$, if next interval's "last traded price" is less than the current "last traded price".

- $\mathrm{y_i} = 0$, if next interval's "last traded price" is equal than the current "last traded price".

Here, i is the current data point in the down-sampled data.

Chapter 3

Sentiment Analysis

This chapter examines the process of sentiment analysis using the 100 dimensional vectorized twitter data described in chapter 2. We discuss two broad topics in this chapter - sentiment classification and the post processing of the sentiment data for stock trend analysis.

The main focus of this thesis is the sentiment classification phase. It is a three step process:

- Clustering: Division of the tweets into meaningful clusters.
- Sentiment Labelling: Creation of a small labelled data set from the twitter corpus, where each tweet is manually classified as positive (1), negative (-1), or neutral (0).
- Classification: Annotation of unlabelled data from the twitter corpus into one of the three sentiment classes.

In data post-processing, we transform the output of the sentiment classifier - fully labelled sentiment data - into a set of 9 sentiment features. We also introduce the concept of cluster importance score, which in conjunction with the 9 sentiment features gives us the final sentiment feature vectors.

3.1 Clustering

In the clustering phase, we divide the tweets into meaningful clusters. Manually looking at the tweets, we figure out two identifiable clusters, but we apply DBSCAN[8] clustering algorithm to identify all meaningful clusters. DBSCAN algorithm identifies clusters based on the notion of density, i.e., it identifies clusters as high density regions separated by low density areas. Each cluster in DBSCAN is a set of core points, where a core point is a data sample that has atleast a predefined number of neighbouring points within a predefined distance. The algorithm finds clusters by starting with a core point and recursively looking at the neighbours of this core point to find other core points. Each cluster also contains non-core points (boundary points), which are not core points but are present in the neighbourhood(as defined by the predefined distance) of core points[1].

We observe three meaningful clusters from the output of the clustering algorithm. We summarize the clusters as follows:

- Trade Cluster: This cluster includes tweets that talk about market movement, strategic partnership, products or services offered, or any other factor that has a direct impact on the market price. In the stock trend analysis, this is the most important sentiment cluster. Table 3.1 shows a sample of the tweets in the trade cluster.

- Feedback: This cluster includes the tweets that talk about customer issues and feedback. Table 3.2 shows a sample of the tweets in the customer feedback cluster.

- Miscellaneous: This cluster includes the tweets that are not part of the first two clusters. We tried to find meaningful sub-clusters in this cluster, but in that case, either the size of the individual sub-clusters become too small or the sub-clusters are not meaningful. In the stock trend analysis, this is the least important sentiment cluster. Table 3.3 shows a sample of the tweets in the customer feedback cluster.

Tweets
@axisbank up 5% on talks of bain capital eyeing stake
@axis bank tumbles over 9% as asset quality worsens @suchetadalal
@yatra @axisbank offer flat rs 1000 off on domestic flight booking of rs 6000 or more
Axis bank 's sangram singh appointed ceo of freecharge after acquisition is finalised fintech
Buy oneplus 3t at 25 999mrp 29 999 rs2000 cashback using @axisbank creditdebit cards on amazon.in
Market live sensex extends losses at open nifty holds 10 200 @axisbank tanks 7% axis bank was biggest

Table 3.1: Sample tweets in the Trade Cluster

Tweets
@axisbank what's wrong with your app while trying to add beneficiary
Don't expect anything from ab staff pls close ur account at first available opprtunity
@axisbank please let us know why atm don't have money if we use card also stop charges of 2%
Muhpetaala amazing initiative by @axisbank undertake precautions whenever use internet banking
Friends everyone should now have muhpetaala you can not forget this axisbank
@axisbank worse service mailed 50 times for home loan query always sending auto mail now never prefer

Table 3.2: Sample tweets in the Feedback Cluster

Tweets
Dropped a mail pls check
My contact details **********
Sales in credit card for axis bank in bhubaneswar
@noidatrafficpol A burnt wagon r is lying on road in front of axis bank
hi to reiterate we wish to inform that axis bank does not levy any charges on any pos transaction.
Date 7th september 2017 casting for axis bank advertisement looking for actors with wheatish complexion

Table 3.3: Sample tweets in the Miscellaneous Cluster

3.2 Sentiment Labelling

After we get the relevant clusters, we create a labelled corpus of tweets and their sentiments. For this purpose, we select 500 tweets from each cluster and manually label their sentiment as positive (+1), negative (-1), or neutral (0). This gives us a labelled dataset of 4500 tweets (1500 tweets for each stock).

3.3 Classification

For the classification of non-annotated data, we train two classifiers- Random forest classifier and Support vector machines with a Gaussian radial basis function as kernel- on our labelled data. As discussed in section 3.2, we have 1500 labelled tweets for each stock. Out of these 1500 tweets, we randomly select 10% (150) tweets for testing the accuracy of the trained model on the tweet data for individual stocks. From the remaining 4050 labelled tweets, we create a 90-10 split. We use 3645 tweets for training and cross validation purpose, and use the remaining 405 tweets to test the overall accuracy of the trained model.

In the following subsections, we examine the performance of our classifiers. For each classifier, we perform K-fold cross-validation to find out the best model parameters. Finally, we measure the performance of both classifiers on the following test data sets:

- 405 labelled tweets for checking overall model accuracy .
- 150 labelled tweets for checking model accuracy on AXISBANK tweets.
- 150 labelled tweets for checking model accuracy on ICICIBANK tweets.
- 150 labelled tweets for checking model accuracy on HDFCBANK tweets.

3.3.1 Random Forest

Random forest[11] is an ensemble machine learning method that uses the concept of bagging and de-correlation of the tress in the ensemble to improve the accuracy of decision tree models. In bagging, random forest perform bootstrapping to get M different training sets (from our single training set) and build M independent decision tress using the bootstrapped training sets. The output of the bagging phase is the average of the output of the independently built trees. For the classification task, a majority vote is taken over the outputs of the individual decision trees. This process helps to reduce the variance of the decision tree model.

In addition to bagging, random forest also de-correlates the independently built trees by randomly choosing a subset of the available features (predictors) to build the tree. This procedure avoids the possibility of the independently constructed trees being highly correlated on account of one or two very strong predictors.

In our analysis, we use the number of trees (M) in the random forest model as our parameter of interest.

Parameter Optimization

We vary

$$M \in \{10, 20, ..., 100, 200, ..., 1000, 2000, ..., 18000\} \tag{3.1}$$

and run 10-fold cross validation on the training set. Figure 3.1 shows the cross-validation accuracy for each value of M. We find that the parameter that maximizes the cross validation accuracy is M = 300. With this value of the parameter, we finally train our classifier with the entire training data.

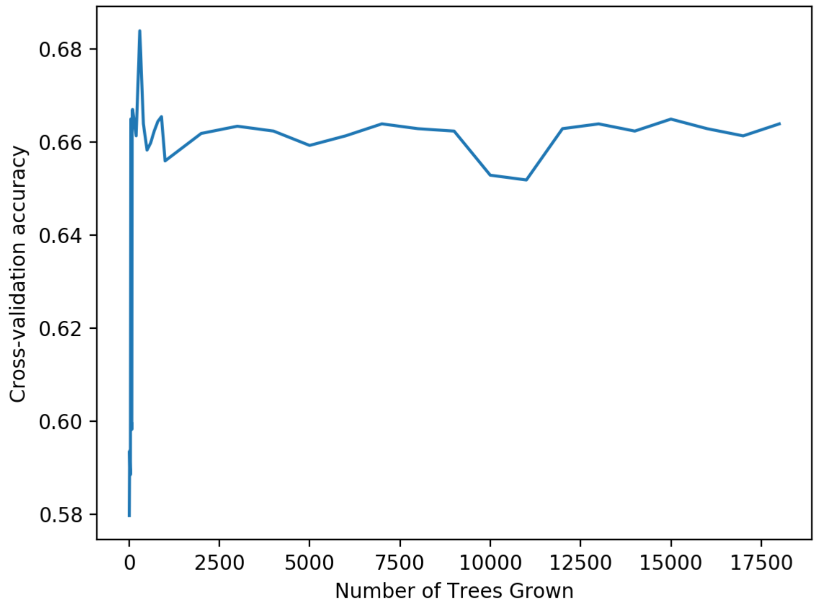

Figure 3.1: Cross validation accuracy from varying the number of trees in the random forest

Test Results

We test our classifier on all four test data sets. Table 3.4 summarizes the performance on all four data sets. Our optimized classifier with M = 300 gives an overall accuracy of 67.8%

	Overall	**AXISBANK**	**ICICBANK**	**HDFCBANK**
Accuracy	67.88%	66.22%	66.56%	68%

Table 3.4: Performance of Random Forest Classifier(M=300) on all four test data sets

3.3.2 Support Vector Machines

SVM[[6],[3],[23]] can be used either for classification or regression problems. In this thesis, we have have SVM for classification. As a classifier, SVM tries to find a set of hyperplanes that separate the data into 2 classes, in high dimensional space. From this set of hyperplane, the SVM algorithm selects the one with the maximum distance (margin) from any points on either side of the hyperplane (decision boundary). SVM finds the hyperplane with the maximum margin by solving the following optimization problem:

$$\min\{\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\epsilon_i\}\ s.t.\ y_i(w^T\varphi(x_i) + b) \geq 1 - \epsilon_i,\ \epsilon_i \geq 0,\ C > 0 \tag{3.2}$$

where w is the normal to the hyperplane; b is the offset; x_i is the i^{th} data point; y_i is the class label of the i^{th} data point; φ is the non-linear function that transforms the input space to a higher dimensional space; ϵ_i is the slack variable that control the amount of misclassification tolerated by the mode; C is the regularization coefficient that controls the trade-of between the simplicity of the decision boundary (in the original input space) and the misclassification error tolerated.[5]

Solution to the problem posed in equation 3.2 will give us the following classifier,

$$y(x) = sign(w^T\varphi(x) + b) \tag{3.3}$$

Using kernel functions, the solution to the dual of the problem posed in equation 3.2 gives us the final classifier (without the need need to know the function φ):

$$y(x) = sign(\sum_{i=1}^{n}\alpha_i y_i K(x_i, x) + b) \tag{3.4}$$

where $K(x_i, x)$ is the kernel function. In our thesis, we have used a Gaussian radial basis function kernel, i.e.

$$K(x_i, x) = \exp(-\gamma||x - x_i||^2) \tag{3.5}$$

where $\gamma = \frac{1}{\sigma^2}$. So, SVM with a Gaussian RBF kernel has 2 parameters of interests- C and γ

Parameter Optimization

We vary

$$C \in \{10^{-2}, ..., 10^{6}\} \; and \; \gamma \in \{10^{-5}, ..., 10^{6}\} \tag{3.6}$$

and run 10-fold cross validation on the training set. Figure 3.2 shows the cross-validation accuracy from varying both C and γ. We find that the parameters that maximize the cross validation accuracy are $C = 10^5$ and $\gamma = 10^{-2}$. With these value of the parameter, we finally train our classifier with the entire training data.

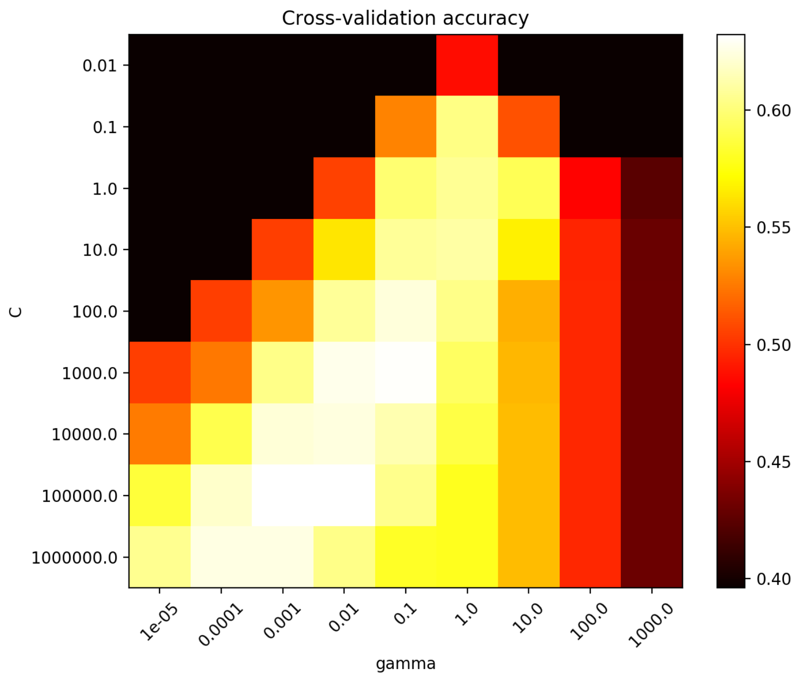

Figure 3.2: Cross validation accuracy from varying C and γ in the SVM

Test Results

We test our classifier on all four test sample data sets. Table 3.5 summarizes the performance on all four data sets. Our optimized classifier with $C = 10^5$ and $\gamma = 10^{-2}$ gives an overall accuracy of 70.8%.

	Overall	**AXISBANK**	**ICICBANK**	**HDFCBANK**
Accuracy	70..83%	70%	70.44%	72.66%

Table 3.5: Performance of SVM with Gaussian RBF kernel($C = 10^5$ and $\gamma = 10^{-2}$) on all four test data sets

3.4 Classifier Comparison

Based on the results presented in Table 3.4 and Table 3.5, we see that the Support vector machine performs better (in terms of accuracy) than the Random forest classifier on all four test data sets. We use our optimized SVM to annotate the sentiments of all unlabelled tweets.

3.5 Data Post-Processing

3.5.1 Sentiment Feature Extraction

We present the steps used to extract the features from tweets annotated with sentiment labels.

- Drop all other information from tweet data, except for date-time and sentiment score of the tweet.
- For each cluster, separate the tweets based on the sentiment value. We get 3 sentiment groups, and each sentiment group has the tweets belonging to the same sentiment class (positive, negative, or neutral).
- In each sentiment group, we make half an hour intervals, wherein each interval has the total number of tweets (occurrence number) in that time period. This gives us 48 half-hour periods in a day.
- On trading days, we keep the half-hour periods from 09:30 -15:30 (trading hours). But we merge all other half-hour periods into one. This merged data will correspond to the time period 15:30 on one day to 09:00 on the next trading day. The occurrence number for this period is the sum of all the occurrence numbers in the merged time

frame. On holidays, we keep merging the occurrence number in the half-hour time periods till 09:00 on the next trading day. The merging process remains the same. This way, we get 13 time periods in a day.

- After the completion of the previous step, for each of the 13 time periods, we get the number of tweets in that time period. We get this information for each of the 3 sentiment groups in all 3 clusters, i.e. we get 9 such scores. Finally, we divide each of these scores by the total number of tweets gathered in that time frame (irrespective of the sentiment class or the cluster) to get all the 9 sentiment features.

3.5.2 Cluster Importance

We introduce a concept of cluster importance factor to emphasize the level of importance of each cluster in our analysis. This factor has the following two properties:

- It is a real number between 0 and 1.
- The sum across all three clusters should be 1.

We multiply each of the 9 sentiment features with their corresponding cluster importance factor to get the final sentiment features. We use these features in conjunction with the features selected from order book data for the final stock trend analysis. We employ an empirical analysis to find out the suitable importance factor. This will be further explained in the next chapter.

Chapter 4

Stock Price Trend Analysis

In this chapter, we evaluate the performance of machine learning learning algorithms- Random Forest and Support vector machines with Gaussian Kernel- on the stock trend prediction task. Continuing from chapter 3, we further examine the role of cluster importance in trend prediction and outline a method to determine this empirical factor.

We incorporate the features extracted during the sentiment analysis phase with the 10 price features extracted from the order book. This gives us a total of 19 features for stock trend analysis.

4.1 Cluster Importance

We vary our cluster importance factors as follows:

- $CI_{trade}, CI_{feedback}, CI_{misc} \in \{0.05, 0.1, 0.15, 0.2, \ldots , 0.75, 0.8, 0.85, 0.9\}$
- $CI_{trade} + CI_{feedback} + CI_{misc} = 1$
- $CI_{trade} > CI_{feedback} > CI_{misc}$ (emphasizes relative importance of the clusters)

CI_{trade}, $CI_{feedback}$, and CI_{misc} are the cluster importance factors for the trade, feedback, and miscellaneous clusters respectively.

For each tuple (CI_{trade}, $CI_{feedback}$, CI_{misc}), we get a different set of sentiment features. During the parameter optimization phase of modelling, we select the best cluster importance tuple and do forecasting using the corresponding sentiment features (along with the common price features from order book). In this thesis we have used a set of 9 such tuples: {(0.6,0.3,0.1), (0.65,0.3,0.15), (0.65,0.25,0.1), (0.5,0.4,0.1), (0.5,0.3,0.2), (0.55,0.35,0.10), (0.70,0.25,0.05), (0.55,0.3,0.15), (0.60,0.35,0.05)}

4.2 Trend Analysis

4.2.1 Parameter Optimization and Forecasting

In order to properly handle time series data for optimization and forecasting, we use a sliding window approach to divide the data set into multiple training and test/validation sets. We use 2 months of data for training and the succeeding 1 week data for test/validation purposes. As we have used 3 months of data in this thesis, we get 4 such training and test set partitions. Figure 4.1 illustrates this sliding window approach.

Parameter Optimization

For parameter optimization, we use the training and validation sets described in sliding window 1 in Fig 4.1. We train the data on the training set and measure the cross-validation accuracy on the validation set. We select those parameters which maximize the cross-validation accuracy for that model.

To find the optimal cluster importance tuple, we repeat the cross validation procedure for each tuple. We select the tuple whose cross-validation accuracy is the highest. The sentiment features corresponding to this tuple (along with the stock price features) are then used forecasting.

Forecasting

For forecasting, we use the training and validation sets described in sliding window 1, 2, and 3 in Fig 4.1. We train the data on the training set and measure the cross-validation accuracy on the validation set. Finally, we report the forecasting accuracy of the model on three test/validation sets.

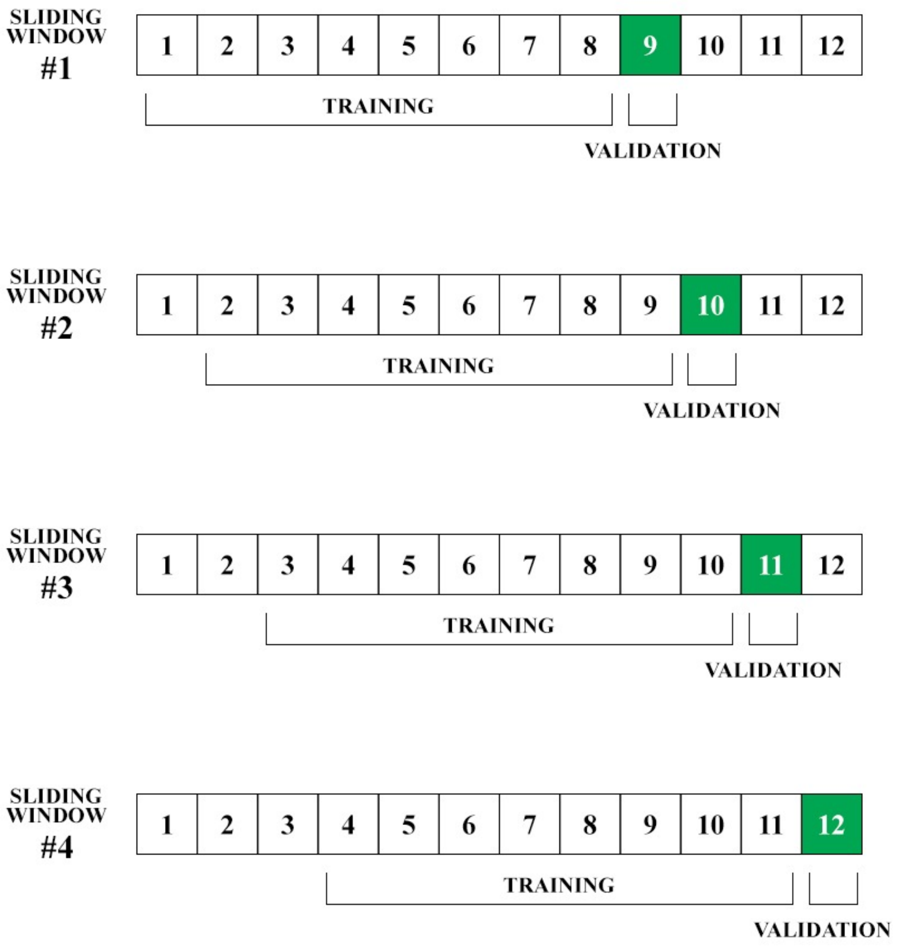

Figure 4.1: Sliding window approach for parameter optimization and forecasting.

4.2.2 Random Forest

Parameter Optimization

In a Random Forest model, the number of trees (M) in the model is our only parameter of interest. We vary

$$M \in \{10, 20, ..., 100, 150, 200, 250, 300, ..., 1000, 2000, ..., 8000\}. \tag{4.1}$$

We run the parameter optimization process described in section 4.2.1 for each cluster importance tuple and the number of trees (M) in our random forest model. Table 4.1 shows the best cross-validation accuracy for each cluster importance tuple for all three stocks. Table 4.2 shows the model parameter (M) that maximizes the cross validation accuracy (corresponding to the best cluster importance tuple) for all three stocks.

Cluster Importance Tuple	**AXISBANK**	**ICICBANK**	**HDFCBANK**
(0.6,0.3,0.1)	58%	53%	54%
(0.5,0.4,0.1)	57%	56%	59%
(0.5,0.3,0.2)	54%	57%	58%
(0.65,0.3,0.15)	57%	56%	57%
(0.65,0.25,0.1)	53%	54%	57%
(0.55,0.3,0.15)	57%	56%	56%
(0.55,0.35,0.10)	56%	53%	54%
(0.70,0.25,0.05)	55%	55%	57%
(0.60,0.35,0.05)	56%	53%	57%

Table 4.1: Best cross-validation accuracy for each cluster importance tuple for all three stocks

	AXISBANK	**ICICBANK**	**HDFCBANK**
Number of trees(M)	3000	3000	5000
Cluster Importance Tuple	(0.6,0.3,0.1)	(0.5,0.3,0.2)	(0.5,0.4,0.1)

Table 4.2: Model parameter (M) that maximizes the cross validation accuracy (corresponding to the best cluster importance tuple) for all three stocks.

Forecasting

We perform forecasting on one week test windows, as described in section 4.2.1. Table 4.3 summarizes the performance of our optimized classifier for all three stocks on each of the three forecasting sliding window test periods.

Stock Symbol	Window2	Window3	Window4
AXISBANK	45%	54%	48%
ICICIBANK	44%	53%	50%
HDFCBANK	44%	57%	48%

Table 4.3: Performance of our optimized Random forest classifier for all three stocks on each of the three forecasting sliding window test periods.

4.2.3 Support Vector Machines

Parameter Optimization

As discussed in chapter 3, there are two parameters of interest in an SVM with Gaussian RBF kernel- the regularization coefficient(C) and gamma(γ). We vary

$$C \in \{10^{-2}, ..., 10^{6}\} \ and \ \gamma \in \{10^{-5}, ..., 10^{6}\} \tag{4.2}$$

We run the parameter optimization process described in section 4.2.1 for each cluster importance tuple and the SVM parameters. The cross-validation accuracy for an SVM does not change with variation in the cluster importance tuple for any of the stocks. So, we arbitrarily choose (0.6,0.3,0.1) as the cluster importance tuple. For all stocks, we get the highest cross validation accuracy with (C, γ)=(10^3, 10^{-1})

Forecasting

We perform forecasting on one week test windows, as described in section 4.2.1. Table 4.4 summarizes the performance of our optimized classifier for all three stocks on each of the three forecasting sliding window test periods.

Stock Symbol	**Window2**	**Window3**	**Window4**
AXISBANK	42%	44%	48%
ICICIBANK	43%	46%	48%
HDFCBANK	44%	48%	50%

Table 4.4: Performance of our optimized SVM classifier for all three stocks on each of the three forecasting sliding window test periods.

4.2.4 Model Comparison

Based on the results presented in Tables 4.5, 4.6, 4.7, we draw the following conclusions:

- Our models perform better than the baseline 33% accuracy (as per the random walk hypothesis) for three-way stock trend classification.
- Random forest classifier (with sentiment data and the cluster importance factor) performs better than SVM in most situations, even though the baseline performance (with stock data only) of SVM is better than random forest for almost all the cases considered. For HDFCBANK data, both the classifiers perform equally well during forecasting in Window 2 and Window 4 (Fig. 4.1).
- The performance of our random forest model improves (appreciably in some cases) with the use of cluster importance factor. However, the performance of SVM remain unaffected by the introduction of this score. Even though the cluster importance factor do not have an impact on SVM performance, the introduction of basic sentiment features improves the model accuracy (in most cases).

The lack of improvement in the performance of the SVM model with cluster importance factor is an interesting result. We need more data to test whether this result is specific to the data set we used or is it a general trend with SVM.

We compare the performance of our non-linear models - SVM and Random forest- with a simple linear model, logistic regression. Table 4.8 shows the result of the comparison. We observe that the linear model performs better than SVM in all scenarios. From our analysis, we also find that the performance of random forest model is comparable to the performance of the linear model. The small size of our dataset could account for this aberration.

Model	Features	Window 2	Window 3	Window 4
Random Forest	Stock	42%	43%	46%
	Stock+Sentiment	44%	49%	46%
	Stock+Sentiment+Cluster Importance	45%	54%	48%
Support Vector Machines	Stock	40%	44%	48%
	Stock+Sentiment	42%	44%	48%
	Stock+Sentiment+Cluster Importance	42%	44%	48%

Table 4.5: Comparison between the performance of SVM and Random forest for different feature settings on AXISBANK data

Model	Features	Window 2	Window 3	Window 4
Random Forest	Stock	43%	46%	48%
	Stock+Sentiment	43%	50%	50%
	Stock+Sentiment+Cluster Importance	44%	53%	50%
Support Vector Machines	Stock	40%	44%	48%
	Stock+Sentiment	43%	46%	48%
	Stock+Sentiment+Cluster Importance	43%	46%	48%

Table 4.6: Comparison between the performance of SVM and Random forest for different feature settings on ICICIBANK data

Model	Features	Window 2	Window 3	Window 4
Random Forest	Stock	44%	40%	44%
	Stock+Sentiment	44%	50%	47%
	Stock+Sentiment+Cluster Importance	44%	57%	50%
Support Vector Machines	Stock	44%	42%	48%
	Stock+Sentiment	44%	48%	50%
	Stock+Sentiment+Cluster Importance	44%	48%	50%

Table 4.7: Comparison between the performance of SVM and Random forest for different feature settings on HDFCBANK data

Stock Symbol	**Model**	**Window 2**	**Window 3**	**Window 4**
AXISBANK	SVM	42%	44%	48%
	Random Forest	45%	54%	48%
	Logistic Regression	43%	56%	53%
ICICIBANK	SVM	43%	46%	48%
	Random Forest	44%	53%	50%
	Logistic Regression	44%	51%	53%
HDFCBANK	SVM	44%	48%	50%
	Random Forest	44%	57%	50%
	Logistic Regression	46%	52%	56%

Table 4.8: Comparison of the performance of the non linear classifiers-SVM and Random forest- against linear logistic regression classifier for all three stocks.

Chapter 5

Future Work and Conclusion

In this chapter, we conclude this thesis with closing remarks about our work and outline potential opportunities for improvement in future work.

5.1 Conclusion

In this thesis, we have used two non-linear models - SVM and Random forest- to predict the intra-day stock price movement for three stocks on the NSE. Our main contributions in this thesis are:

- We designed a sentiment analyzer that identifies the sentiment of each tweet and sorts the tweets into one of three clusters (Trade, Feedback, and Miscellaneous).
- We introduced the cluster importance factor, a score that quantifies the importance of each clusters in the context of market importance.

We used a compound sentiment score (combination of the cluster importance factor and the sentiment score extracted from the output of the sentiment analyzer) in conjunction with the features extracted from order book data to predict the stock price movement.

In chapter 3, we found that an SVM (Gaussian RBF kernel) based sentiment analyzer performs better than a random forest classifier with an overall accuracy of 70.8%. This result is in agreement with one of the recent studies on twitter sentiment analysis for stock price prediction[20].

In chapter 4, we explored an empirical approach to find the best cluster importance factor for each stock. We used the optimal factor in combination with the sentiment features extracted in chapter 3, to get the final sentiment features. Using these sentiment features along with the order book ones, we found that a random forest model performed better than the SVM model for forecasting stock price movement. With the random forest model, we observed that the cluster importance factor had an appreciable impact (in some cases) on the accuracy of the prediction task in comparison to using just the stock features or the combination of stock and simple sentiment features. Our random forest model consistently outperformed the random walk baseline accuracy of 33% for three-way stock trend classification on all three stocks.

In our analysis, we observed a few results that demand a more through analysis on a larger data-set to fully understand the implications of the results, such as:

- Cluster importance factor did not have any impact on the performance of the SVM model.
- Linear model like logistic regression performed better than SVM in the prediction task (performance of logistic regression and random forest in different settings were comparable).

5.2 Future Work

In this thesis, we have analysed 3 months of stock data. While 3 months of order books amounts to more than 1,00,000 data points, down-sampling at every 30 minutes does not leave us with too many data points. Analysing an entire year's data would give a lot more data points to explore the anomalous results discussed in the previous section. Moreover, with more data points, we can apply neural network models like CNN and LSTM .

While twitter is an admirable source of market information, we can include other sources of market information- financial articles, blog-posts, Blomberg report, etc.- in our analysis to get a complete picture of public sentiment.

In our work, we have focussed on the stocks of only 3 companies listed on the NSE. We can expand our work to include other companies on the exchange. This will provide a better understanding of the cluster importance factor's impact on the prediction accuracy for different stocks.

Bibliography

[1] Dbscan algorithm. `http://scikit-learn.org/stable/modules/clustering.html#dbscan`, accessed: 2018-05-12

[2] Bollen, J., Mao, H., Zeng, X.: Twitter mood predicts the stock market. Journal of computational science 2(1), 1–8 (2011)

[3] Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on Computational learning theory. pp. 144–152. ACM (1992)

[4] Chenoweth, T., Obradovic, Z., Lee, S.S.: Embedding technical analysis into neural network based trading systems. Applied Artificial Intelligence 10(6), 523–542 (1996)

[5] Christopher, M.B.: Pattern recognition and machine learning. Springer-Verlag New York (2016)

[6] Cortes, C., Vapnik, V.: Support-vector networks. Machine learning 20(3), 273–297 (1995)

[7] De Fortuny, E.J., De Smedt, T., Martens, D., Daelemans, W.: Evaluating and understanding text-based stock price prediction models. Information Processing & Management 50(2), 426–441 (2014)

[8] Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd. vol. 96, pp. 226–231 (1996)

[9] Fama, E.F.: The behavior of stock-market prices. The journal of Business 38(1), 34–105 (1965)

[10] Geva, T., Zahavi, J.: Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news. Decision support systems 57, 212–223 (2014)

[11] James, G., Witten, D., Hastie, T., Tibshirani, R.: An introduction to statistical learning, vol. 112. Springer (2013)

[12] Kalev, P.S., Liu, W.M., Pham, P.K., Jarnecic, E.: Public information arrival and volatility of intraday stock returns. Journal of Banking & Finance 28(6), 1441–1467 (2004)

[13] Kim, K.j.: Financial time series forecasting using support vector machines. Neurocomputing 55(1-2), 307–319 (2003)

[14] Leung, J.W.: Application of machine learning: automated trading informed by event driven data. Ph.D. thesis, Massachusetts Institute of Technology (2016)

[15] Lo, A.W.: The adaptive markets hypothesis: Market efficiency from an evolutionary perspective (2004)

[16] Malkiel, B.G., McCue, K.: A random walk down Wall Street, vol. 8. Norton New York (1985)

[17] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[18] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)

[19] Pagliardini, M., Gupta, P., Jaggi, M.: Unsupervised learning of sentence embeddings using compositional n-gram features. arXiv preprint arXiv:1703.02507 (2017)

[20] Pagolu, V.S., Reddy, K.N., Panda, G., Majhi, B.: Sentiment analysis of twitter data for predicting stock market movements. In: Signal Processing, Communication, Power and Embedded System (SCOPES), 2016 International Conference on. pp. 1345–1350. IEEE (2016)

[21] Shefrin, H.: Beyond greed and fear: Understanding behavioral finance and the psychology of investing. Oxford University Press on Demand (2002)

[22] Shiller, R.J.: From efficient markets theory to behavioral finance. Journal of economic perspectives 17(1), 83–104 (2003)

[23] Vapnik, V.: The nature of statistical learning theory. Springer science & business media (2013)

[24] Wang, J.L., Chan, S.H.: Stock market trading rule discovery using two-layer bias decision tree. Expert Systems with Applications 30(4), 605–611 (2006)