Order Book Based Market Simulation Using a State Machine

B. A. N. P. Kumarasiri

Order Book Based Market Simulation Using a State Machine

B. A. N. P. Kumarasiri
Index No : 13000616

Supervisors: Dr. Chamath Keppetiyagama
Dr. Mindika Premachandra

December 2017

Submitted in partial fulfillment of the requirements of the B.Sc in Computer Science Final Year Project (SCS4124)

Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Candidate Name: Ms. Pumudinee Kumarasiri

………………………………………………

Signature of Candidate Date:

This is to certify that this dissertation is based on the work of Ms. Pumudinee Kumarasiri under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor Name: Dr. Chamath Keppetiyagama

………………………………………………

Signature of Supervisor Date:

Supervisor Name: Dr. Mindika Premachandra

………………………………………………

Signature of Supervisor Date:

Abstract

Stock market trading has evolved into an electronic market, where trading happens according to automated algorithms. Trading algorithm developers are developing more competitive algorithms for tomorrow which can beat yesterdays and traders are trying to survive in this competitive market. Both trading algorithm developers and traders may benefit from a model which predicts the market impact caused from a specific limit order.

This research studies the market impact of limit orders using a state machine with stable, upward or downward states by observing order book and trade data. An order book representative factor named, State Factor has been introduced in order to represent an order book instance as a single floating point value. All information in a single order book view has been taken into consideration when introducing this State Factor. Market state identification is done using trade data. As the machine learning approach, a Hidden Markov Model has been used for prediction. Model training is done using observed order book data and derived states from trade data. The trained model predicts the posterior probabilities of the market being in stable, upward or downward states for a given limit order or a limit order sequence. Continuous learning approach has been used to train and test the model.

Model evaluation is done by comparing the real probability values with the predicted posterior probabilities. The model gives promising posterior probability results with at most 4% deviation. Also, the model predicts the state sequence which is likely to occur in a particular trading day, with an accuracy between 53 – 60 %. The results of the modelling and analysis are presented such that they can be used as an aid to future work.

Keywords: Market impact, Hidden Markov Model, State Machine, Order Book

Preface

The basis for this research originally came from my passion and the experience I had at my internship at DirectFN. I was inspired to do a research on order book simulation since this was a problem which came to me at the internship.

This dissertation is organized in six chapters. The comparison done in Chapter 1.1 in Table 1.1 is entirely my own work. The Chapter 3 contains the design which was proposed by me with the help of supervisors, for this research study. Apart from the Hidden Markov Model concept, all other work in Chapter 3 are my own work. The State Factor introduced in Chapter 3.3.4 is based on the ideas gathers from referred papers but the equation was designed by me. Implementation details included in Chapter 4 are my own work and each diagram, each result was acquired by me through model training. Evaluation method used in Chapter 5 is not defined by me, but I evaluated the data by myself. Conclusions stated in Chapter 6 are also my own work.

Acknowledgement

I would like to express my sincere gratitude to my supervisors Dr. Chamath Keppetiyagama and Dr. Mindika Premachandra for guiding and encouraging me throughout the research. Valuable insights given by them, were a tremendous support to me. I am also grateful to Dr. Kasun Gunawardena and Dr. K.D. Sandaruwan for their advices.

I am thankful to our research coordinator Dr. H.E.M.H.B. Ekanayake for his guidance and keeping the research work on-track. I convey my heartiest appreciation for all the lecturers who gave feedback at proposal defense and interim presentations.

My special thanks goes to Mr. Arjuna Nanayakkara and Mr. Ruwan Aluthgedara at DirectFN, for their kind support throughout the research work. They helped me by providing the data sources according to my requirement and giving feedback for the research.

Finally my thanks goes to my loving family and friends for being with me in my hard situations and for the comments and encouragement given by them.

Table of Content

List of Figures

List of Tables

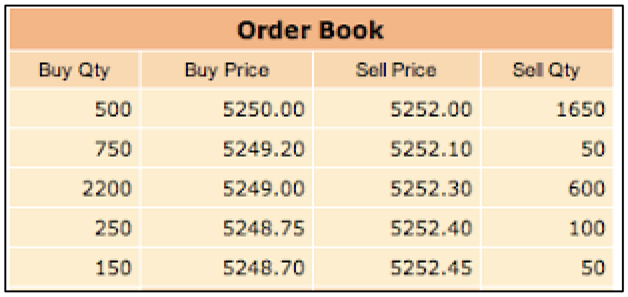
List of Equations

Chapter 1

Introduction

A stock exchange plays a major role in a country's financial state. Stock market trading, which had existed for centuries, has evolved into an electronic market, where trading happens according to automated algorithms. People who used to trade following the order book have moved on to relying on trading algorithms since they are way better than manually engaging with High Frequency Trading. This evolution has resulted in predicting, simulating different parameters in the stock market with the use of huge data streams, to provide more information for traders, asset managers, brokers and policy makers.

Order book, which is a special data structure used by stock markets to organize each incoming buy or sell order, is a major interest among researchers in this domain because of its' ability to showcase the market sentiment or the trader belief towards a particular instrument. Order flow is visible to every market player in real-time and a trade happens whenever an order arrives for buying or selling at the best currently available price. Each trade happens at least with one order from the order book. Therefore, in order-driven markets, the total market behavior can be modeled through the order book.



Order Book			
Buy Qty	Buy Price	Sell Price	Sell Qty
500	5250.00	5252.00	1650
750	5249.20	5252.10	50
2200	5249.00	5252.30	600
250	5248.75	5252.40	100
150	5248.70	5252.45	50

Figure 1.1 - Order Book Representation

As shown above, order book is organized in two lists, i.e. buy side and sell side according to the price level. Buy price is sorted in descending order while sell price is sorted in ascending order.

Topmost row showcase the best bid (buy) and best ask (sell) prices at the current time. Order book is updated in real time, based on trades, adding a new order etc.

1.1 Background to the Research

The growth of computer speed and algorithmic development has created many possibilities in trading. Usage of trading algorithms to execute daily transactions has also grown rapidly, with the dawn of the electronic market. American markets and European markets generally have a higher proportion of algorithmic trades than other markets, and has a range as high as an 80% proportion.

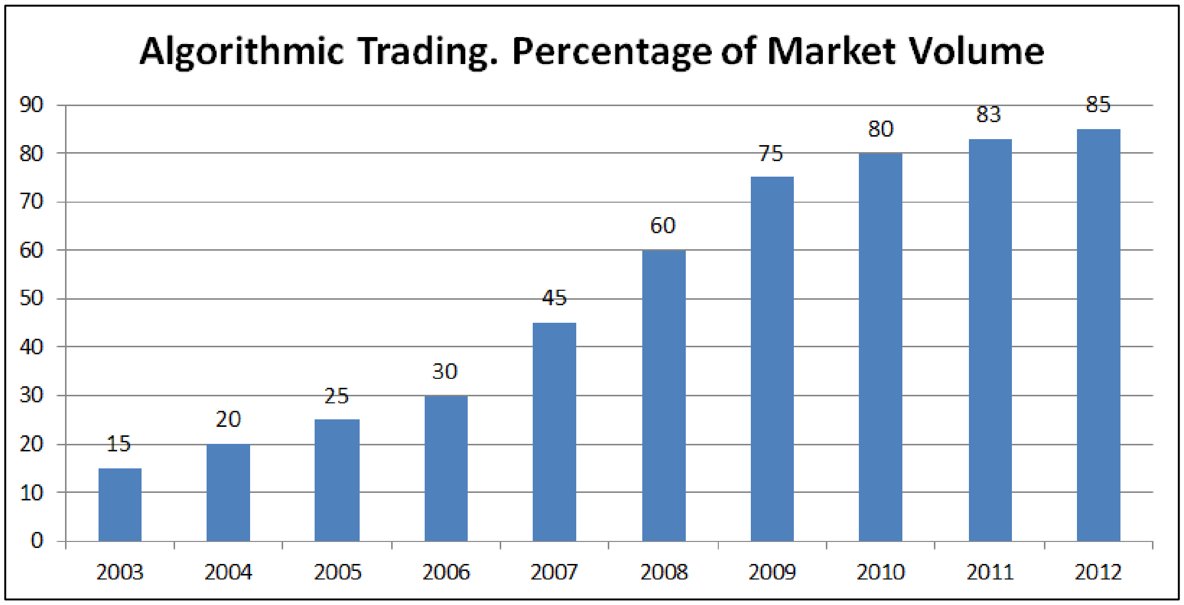


Figure 1.2 - Algorithmic Trading Market Volume Evolution [1]

Even though Algorithmic Trading has achieved a huge attention within past years, there seems to be a significant scarcity in publically available academia relevant to this domain due the monetary value they hold. However, implementing new and challenging trading algorithms is an emerging research area. To further encourage trading algorithm developers, some organizations such as Quantopian [2] provides a platform to compete against other developers using trading algorithms to make profits and to build better algorithms.

Testing the market impact caused by a specific order, is still a main challenge faced by trading algorithm developers and trader decision support systems. They cannot carry out the testing mechanism in the real market, because a small bug in the algorithm or in the system, will cost them a huge amount of money. Testing methods used at present are stated below.

Testing Method	**Issues**
Online testing websites e.g. Quantopian [2] - allow users to upload their algorithms and test them with a provided dataset	Fails to test the market impact of the algorithm because this uses a provided dataset
Standalone software e.g. MetaTrader4 [3] – platform to practice trading	Consists of hypothetical orders Does not provide the real market atmosphere
Online exchange simulators e.g. Investopedia exchange simulator [4] - real time platform to practice trading	Not suitable to test automated trading algorithms because it takes about 20 minutes to execute one order and fails to test the market impact
Other exchange simulators	Publically not accessible

Table 1.1 – Testing methods used at present

According to Table 1.1, there isn't any publically available method to test a particular trading algorithm from market impact aspect. However, proper testing platforms must exist, given that algorithmic trading has achieved a vast popularity. It is unfortunate that those solutions may not be publically available. Chapter 2 will also highlight this problem, based on related work.

1.2 Research Problem and Research Question

After each trade, that particular transaction creates an impact to the existing trade price[1]. Based on this impact, the next best bid and best offer prices in the order book will be changed. Since best bid or best offer price is most likely to become the next trade price, the impact occurred from the previous trade can be affected to the next trade.

Knowing what would be the next market state or the next trade price trend, is important for a trader because he can make a profit from that information. If the next market state is known

[1] Further explained in Chapter 2.3 under Market Impact

beforehand, algorithmic developers can test their trading algorithms in a challenging market environment and design more competitive algorithms. But the inability to provide such information has influenced the trader and has reduced the effectiveness of trading algorithms. The above research problem has led to the following research question.

Can we identify an upward, downward or stable trend in the market, by analyzing order book data and the incoming buy/sell order details?

Since order book is a collection of buy/sell orders and every trade should occur at least with one order from the order book, the current order book represents the current market state and the general market direction at a particular time. When new orders are added to the current market state, it will represent new market states. By adding a new order, the best bid/ best offer price can get changed and the next trade might depend on the added order price.

1.3 Methodology

The proposed research study follows a continuous learning approach because price patterns in different months may vary and the trained parameters would not match for the testing phase. The model will be trained using a cumulative dataset. Predictions will be done for the next day, using the model which was trained up to the current date. This methodology ensures that parameters get learned continuously. Methodology will be further discussed in Chapter 3.

1.4 Outline of the Dissertation

This dissertation is organized in six main chapters. Precise introduction is given in Chapter 1. The document proceeds to describe relevant background information and related work in Chapter 2. Chapter 3 focuses on the proposed research design and will be described using key design considerations. Chapter 4 showcases the implementation process, highlighting the dataset, tools and equations. Chapter 5 will elaborate the evaluation process along with the acquired results for the proposed model. Finally, Chapter 6 will explain the conclusion, limitations and future work.

1.5 Definitions

Financial domain has many domain specific words, which needs to be understood in order to get a clear picture about the proposed research study.

Instrument - an abbreviation used to uniquely identify publicly traded shares of a particular stock on a particular stock market. A stock instrument may consist of letters, numbers or a combination of both.

Limit Order – Both sell and buy orders can be defined as limit orders. In order to place a limit order, both order price and order quantity have to be mentioned. Limit orders can be canceled after a specific time period.

Market Order – Only the order quantity has to be mentioned, to place a market order. It will get matched with the best available current price. Market orders are not added to the order book.

Market By Order – One type of order book representation. Order book view is organized as order by order.

Market By Price – Other type of order book representation. Orders having same order price are concatenated and shown as one order. Splits define how many independent orders are shown in this particular order. This research study uses the order book view in this format.

1.6 Delimitations of Scope

Classification of order book data will be done only for bid and ask related data. The research will target the whole market point of view, not from the trader's or market-maker's perspective. As an initial step, only one instrument will be taken into consideration to analyze patterns between order book states. Market impact within different instruments (cross instrument interference) will not be considered. Market impact caused by different buy/sell orders within the same instrument, will be analyzed.

1.7 Chapter Summary

This chapter gave a detailed introduction to the research study. It further explained about the research problem and the research question to be addressed. Methodology, important definitions and the delimitation of scope were described in detail. Hence, the dissertation will proceed with a clear foundation.

Chapter 2

Literature Review

Even though financial market related research is an emerging area, the number of publically available research studies are limited due to the monetary value of these researches. Most of the researchers will earn money than publishing their work. Difficulty of acquiring a proper order book dataset has also limited the number of order book simulation researches done in this area.

This section provides a discussion that relates to the research study. At first it discusses about whether the stock market can be represented using a state machine. Then it discusses various approaches taken to simulate the order book for different intentions. In many scenarios order book simulation has taken place to forecast trader related parameters. The typical price, relative strength index, moving average are some of them [5]. Finally it discusses about the literature from the market impact aspect.

2.1 State Machine Representation

A state machine is any device that stores the status of something at a given time and can operate on input to change the status and/or cause an action or output to take place for any given change. A finite state machine is a model of a system that shows the different possible states the system can attain and the transitions that occur as the system moves from one state to another. We can use it in modeling a trading system.

Ismaila Oladimeji et al has modeled the stock market using a Hidden Markov Model (HMM). HMM is a finite set of states, each of which is associated with a probability distribution.

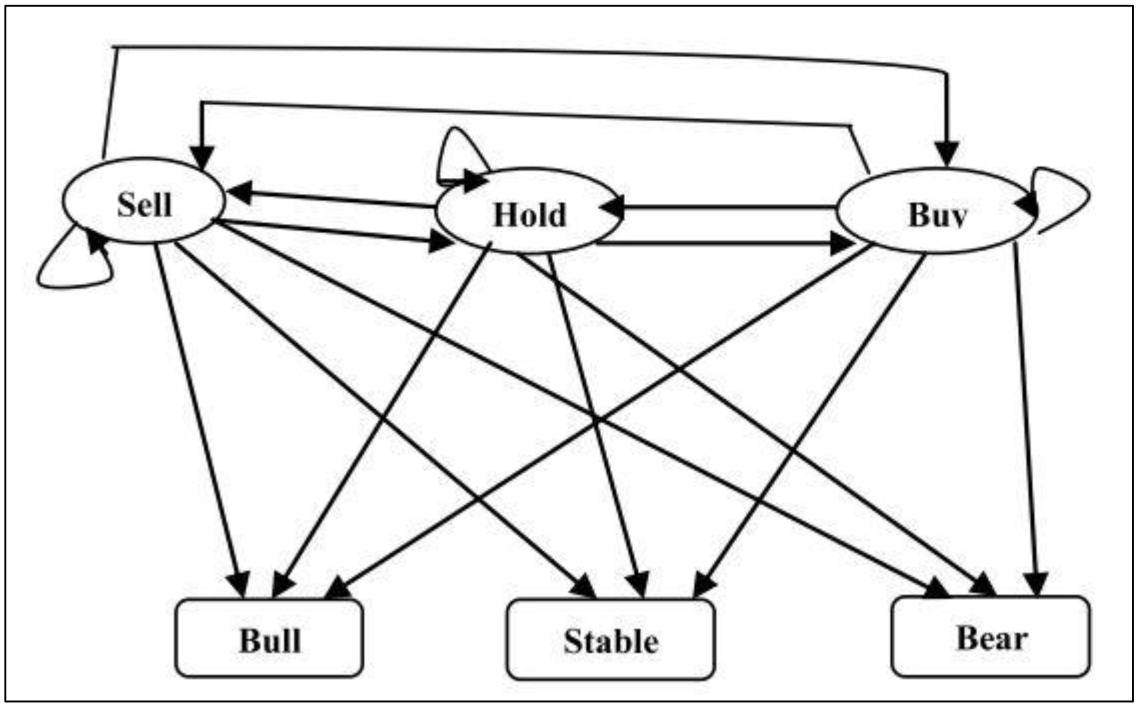


Figure 2.1 - Three State HMM

Figure 2.1 shows the map of stock market operations in terms of an HMM. Hidden states (Sell, Hold and Buy) and observation instruments (Bull, Stable and Bear) have been decided in the above state representation. The situations of stock market are Bear (when prices are falling), Bull (when the market is generally rising) and stable (no change) represent the observation instruments [5].

S.P. Meyn et al provide a comprehensive description on Markov Chains and Stochastic Stability [6]. According to them, the following states (Figure 2.2) represent, whether a hypothetical stock market is exhibiting a bull market, bear market or stagnant market trend during a given week. They have identified the distribution over states can be written as a stochastic row vector.

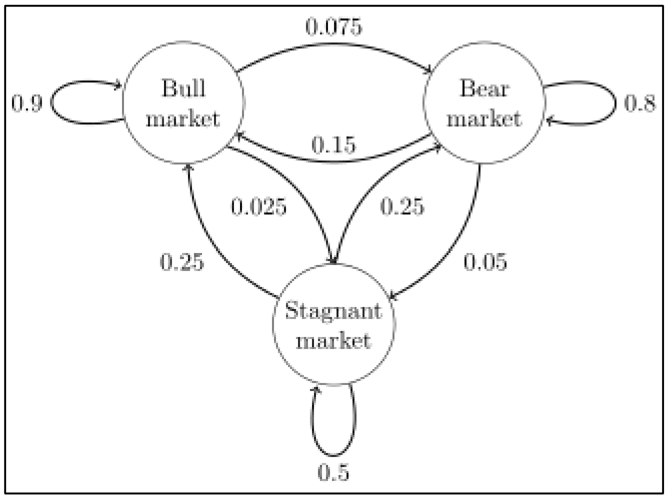


Figure 2.2 - State Representation of a Stock Market

According to above scenarios it is positive that a stock market can be represented using a state machine.

2.2 Order Book Simulation

Stock markets consist of specialized traders and market-makers, to provide liquidity and volume to the market. Responsibility of market-makers is to set prices and volumes for buying and selling. Kim, Adlar J. et al [7] model the market as a dynamic system and propose a reinforcement learning algorithm that learns profitable market-making strategies when run on this model. To model the order flow from market-maker's perspective, an Input-Output Hidden Markov Model has been used. Main objective of this study is to derive a profitable market-making strategy.

Forecasting the dynamics of the order book accurately, is crucial in a variety of application scenarios, for example:

- A broker charged with executing a trade: The broker's ability to forecast the behavior of the order book would help him to optimize the trade execution schedule to decide how to partition the total number of units to be traded into small amounts, how to time each small order, and how aggressively to trade it

- An asset manager contemplating a portfolio rebalancing: A predictor of the order book dynamics would help in forecasting the cost occurred by the price impact of the trades required to perform the rebalancing
- An exchange or a regulator charged with maintaining orderly markets: Order book forecasting tools help exchanges and regulators to quickly detect the possibility of abnormal events [8]

Most of the execution algorithms proposed in this research domain attempts to produce trading decisions neglecting the order book information. However, as stated above the order book contains important information to forecast market direction. Therefore, Deepan Palguna and Ilya Pollak [8] have proposed several non-parametric methods to predict short-term mid-price changes in limit order books. The average of the best ask and the best bid is called the mid-price. Even though asset managers with large portfolio turnovers and brokers with considerable trading volumes would benefit from this study, still it fails to forecast the market impact.

Weibing Huang, Charles-Albert Lehalle, and Mathieu Rosenbaum [9] have proposed a model which analyzes the properties of the full order book by dividing time interval of interest into periods based on mid-price. This model is capable of analyzing pre-trade cost in a simple and efficient way. The historical order flow, which is another important public information has not been used in this study.

To describe an order book state, both order price and order quantity are important. Su Chen, Chen Hu and Yijia Zhou have applied 4-variate Hawkes Process to simulate arrivals of orders, and measure the market impact caused by different sizes of orders. They have discovered that bigger the liquidation size, the larger the market impact is. They have stated that ‘It’s quite clear to see that, most of the orders are actually of size 10000, 5000, 15000, etc. strongly depending on the dataset. Investor rarely place orders which size is a weird number’ [10], which is not a valid argument since there exists order quantities in some electronic markets, which do not align with this statement.

Neural networks are found to perform well for modeling the distribution of the best ask and best bid prices, with significantly better performance relative to logistic regression. Justin Sirignano has developed a spatial neural network, which is computationally efficient and specifically designed to take advantage of the spatial structure of limit order books [11]. Importance of this model is that it allows to model price movements, which lay deep in the limit order book (i.e., many levels beyond the best bid and best ask).

According to the order book simulation literature analysis, various contributions made to this research domain can be found. However, exploring trader familiar parameters such as trade price, trade quantity, mid-price etc. and placing a better buy/sell order is the intention of a trading algorithm. To test such a trading algorithm, we need to have an order book which operates exactly similar to the real order book and we need to forecast how a particular buy/sell order will impact the real stock market.

2.3 Market Impact

Market impact is the effect that a market participant has when it buys or sells an asset. It is the extent to which the buying or selling moves the price against the buyer or seller, i.e., upward when buying and downward when selling. If a particular stock has a higher demand (many buy orders) then the buy price increases, i.e., price moves upward. If a particular stock has a higher supply (many sell orders) then the sell price decreases, i.e., price moves downward. Knowing the market impact caused by a specific buy/sell order beforehand, would be helpful for both traders and trading algorithm developers.
Market impact has two aspects.

1. Impact caused within different orders among the same instrument
2. Impact caused within different instruments in a particular stock market

However, this research study focuses only on the first aspect.

Kyle’s Lambda is a most common and simple method defined to measure the market impact [12]. For short periods, market impact λ is,

$$\lambda = \frac{|\Delta \text{Price}_t|}{\text{Volume}_t}$$

Equation 4- Kyle's Lambda Function

Volume is typically measured as turn-over or the value of shares traded, not the number. Under this measure, a highly liquid stock is one that experiences a small price change for a given level of trading volume. However, this measurement gives the market impact after trading happens, not beforehand.

Yoshihiro Yura et al have introduced the Knudsen number of the financial Brownian particle motion and its asymmetric version (on the buy and sell sides) [13]. Even though the Knudsen number was not considered previously, asymmetric Knudsen numbers are crucial in finance in order to detect asymmetric price changes. By using Knudsen number, they have characterized conditions for the market dynamics. They have concluded that the number of limit orders in the inner layer can be a good indicator of the **market stability** and directional movement of financial markets. It would only detect the direction of the price motion at the early stage while volatility is blind to the price direction.

Esteban Moro et al have empirically studied the market impact of trading orders [14]. They have mainly focused on large trading orders that are executed incrementally, which also known as hidden orders. They have found that market impact is strongly concave and it gets increased when the order size gets increased.

According to H.M. Soner [15], market impact can be divided as temporary impact and permanent impact. After a trade happens, a certain fraction of price change remains (permanent impact) but the remaining price change decays in time (temporary impact) (Figure 2.3). However, this research study focuses on the total market impact and it is important to note that there is a visible market impact after a particular trade.

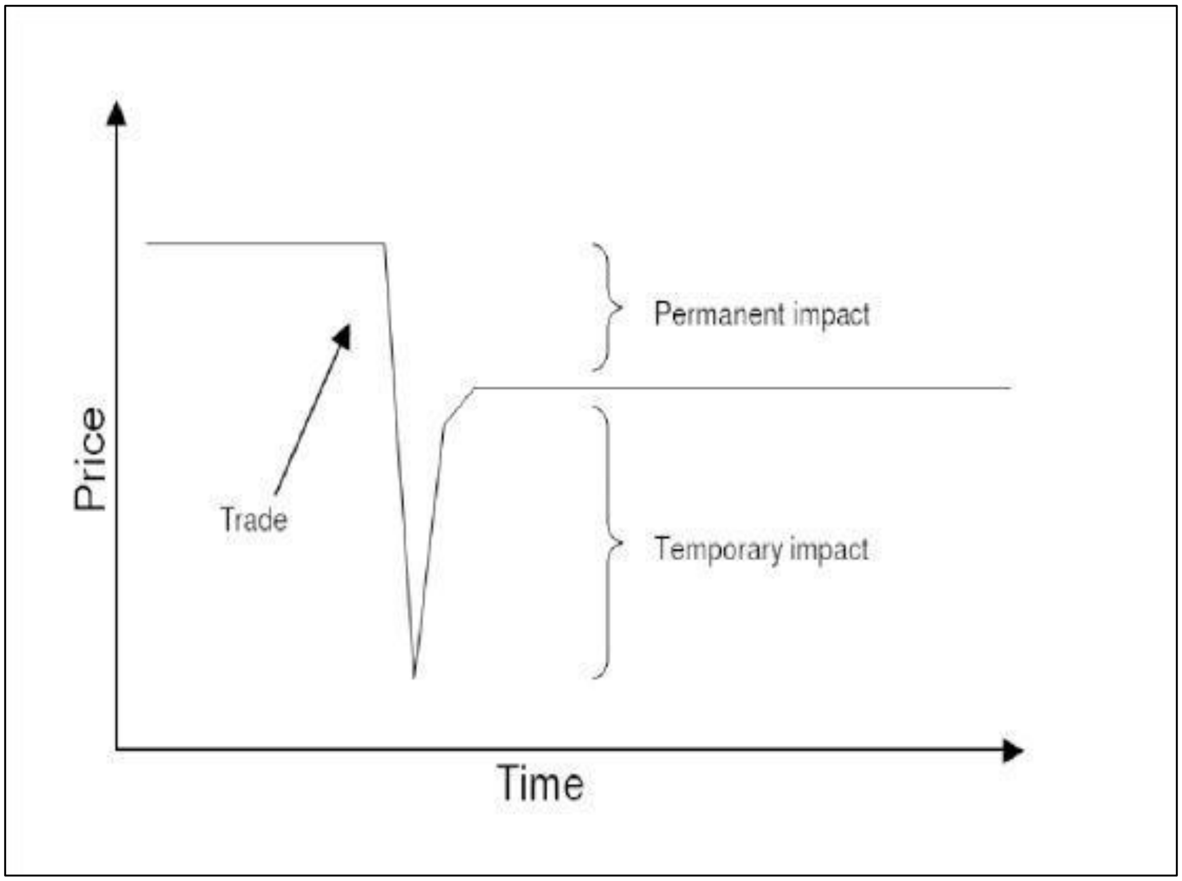


Figure 2.3 - Variation of Market Impact after a Trade

This chapter provides a wider context of the research on State Representation, Order Book Simulation, Market Impact and the related work relevant to the research problem mentioned in Chapter 1.2. Furthermore, it provide theories which will benefit the proposed research study. The application of those details would be described in Chapter 3 and 4.

Chapter 3

Design

This section describes the selected approach for the proposed research study and the design considerations, based on the research problem and the literature survey. Furthermore, a high level design architecture is provided and each component is explained with detailed descriptions.

3.1 Design Considerations

The main intention of this research is to predict the next market state for a given buy/ sell order. Hence the research approach needs to observe different patterns within order book data in order to provide an accurate prediction. Therefore, proposed research study follows a continuous learning approach other than following a traditional learning approach. This will be further discussed in Chapter 4 under Implementation.

3.2 Design Overview

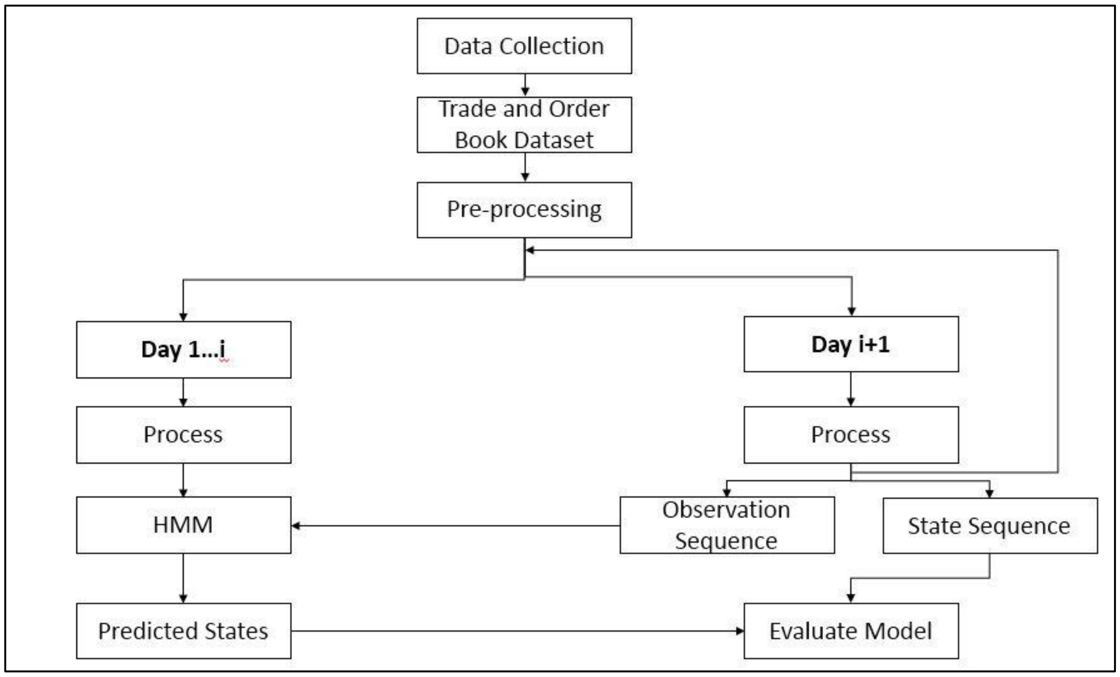


Figure 3.0-1 - High Level Design Architecture

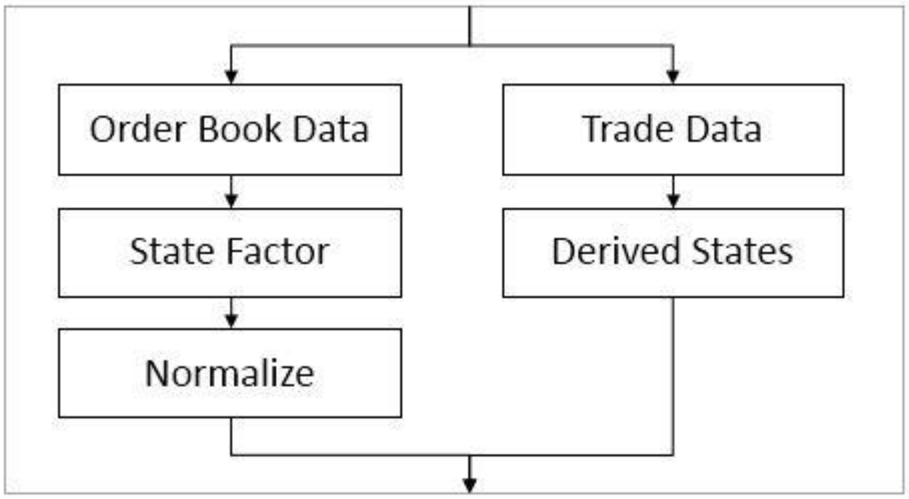


Figure 3.0-2 - Process Component of Design Diagram

Acquired dataset will be pre-processed to ease the analysis. By following the continuous learning approach, Hidden Markov Model will be trained using 'Day 1 to i' dataset. The model will be evaluated by comparing the predicted state sequence for 'Day i+1' from the model and the real state sequence occurred on 'Day i+1'. Then processed data in 'Day i+1' will be added to the training dataset, model parameters will be learned again and the predictions will be done for 'Day i+2' ($1< i < n$, n = number of days).

3.3 Design Components

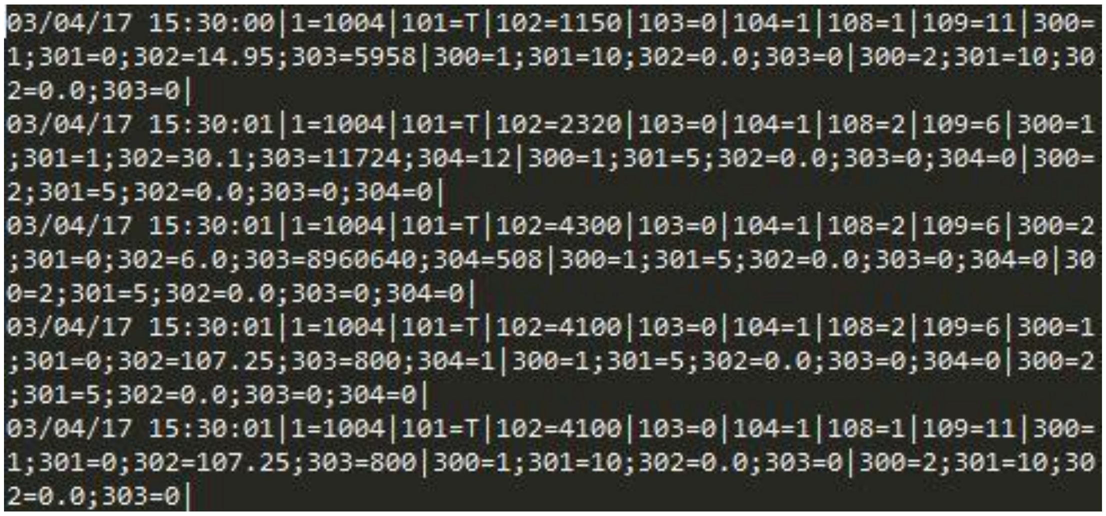
A detailed explanation of the design components will be stated in this section.

3.3.1 Data Collection and Dataset

As the first step stock market data had to be collected continuously. A Middle East stock exchange was used to acquire real time order book update data and trade data. The data was collected during market hours (12.30 p.m. – 5.30 p.m. Sunday to Thursday) from 3rd April, 2017 to 13th July, 2017. Cron, which is a time based job scheduler was used to run the data acquiring program periodically at fixed intervals during specific dates.

3.3.2 Pre-Processing

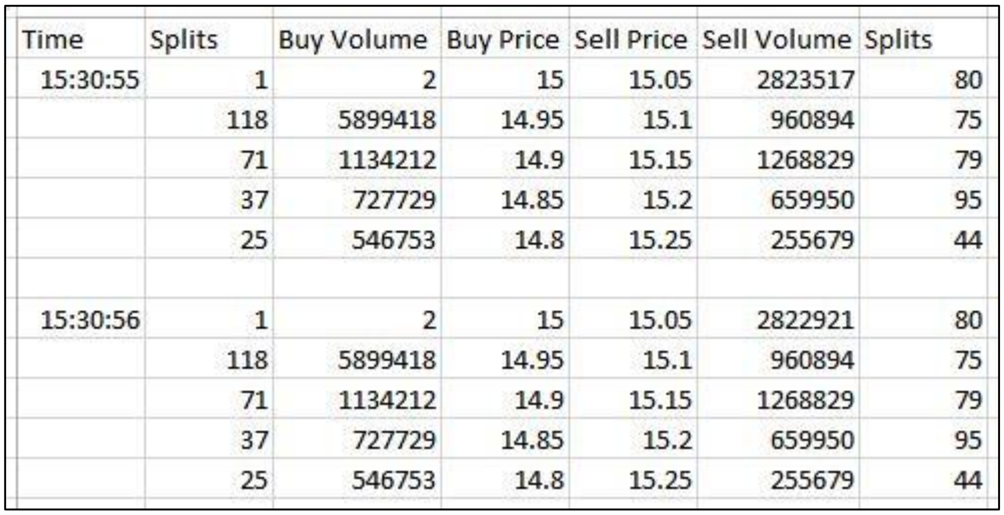
A basic data pre-processing approach was taken into consideration in order to ease analyze and to provide quality data. The real time data, gathered from the servers had a raw, frame structure and it had to be changed into the relevant order book instance views.



```
03/04/17 15:30:00|1=1004|101=T|102=1150|103=0|104=1|108=1|109=11|300=
1;301=0;302=14.95;303=5958|300=1;301=10;302=0.0;303=0|300=2;301=10;30
2=0.0;303=0|
03/04/17 15:30:01|1=1004|101=T|102=2320|103=0|104=1|108=2|109=6|300=1
;301=1;302=30.1;303=11724;304=12|300=1;301=5;302=0.0;303=0;304=0|300=
2;301=5;302=0.0;303=0;304=0|
03/04/17 15:30:01|1=1004|101=T|102=4300|103=0|104=1|108=2|109=6|300=2
;301=0;302=6.0;303=8960640;304=508|300=1;301=5;302=0.0;303=0;304=0|30
0=2;301=5;302=0.0;303=0;304=0|
03/04/17 15:30:01|1=1004|101=T|102=4100|103=0|104=1|108=2|109=6|300=1
;301=0;302=107.25;303=800;304=1|300=1;301=5;302=0.0;303=0;304=0|300=2
;301=5;302=0.0;303=0;304=0|
03/04/17 15:30:01|1=1004|101=T|102=4100|103=0|104=1|108=1|109=11|300=
1;301=0;302=107.25;303=800|300=1;301=10;302=0.0;303=0|300=2;301=10;30
2=0.0;303=0|
```

Figure 3.0-3 - Order Book Data Frame Structure

Above data frames contain the incremental updates of how the order book behaved during a particular trading day. Each frame shows an action happened to the current order book, i.e. trade, new order, cancel order etc. and it consists of different tags which further describe what is the current status of order book. For an example, 300=1 represents updated orders in the buy side and 300=2 shows orders in sell side. By analyzing data frames, following order book views can be generated.



Time	Splits	Buy Volume	Buy Price	Sell Price	Sell Volume	Splits
15:30:55	1	2	15	15.05	2823517	80
	118	5899418	14.95	15.1	960894	75
	71	1134212	14.9	15.15	1268829	79
	37	727729	14.85	15.2	659950	95
	25	546753	14.8	15.25	255679	44
15:30:56	1	2	15	15.05	2822921	80
	118	5899418	14.95	15.1	960894	75
	71	1134212	14.9	15.15	1268829	79
	37	727729	14.85	15.2	659950	95
	25	546753	14.8	15.25	255679	44

Figure 3.0-4 - Order Book Instances View Converted from Raw Data

3.3.3 Placed Order Details

Deriving the buy/sell order sequence placed to the market, can be done by analyzing the difference between order book views.

Figure 3.3 shows two consecutive instances of order book view, derived from data frames. When analyzing these two instances, it is possible to note that the only change happened within this second is 1st row in 6th column. A buy order of price 15.05 and quantity 596 has placed to the market at 15:30:55 and a trade has been occurred. As a result of that action, the topmost sell order quantity has been reduced to 2822921 at 15:30:56. Likewise, all the placed order details can be derived from the dataset itself.

3.3.4 State Factor

Order book captures the current market state and the general market direction at a given time. In order to compare two different states, there needs to be an order book representative factor, i.e., state factor.

When introducing the state factor, following facts were considered.

1. Deepan Palguna and Ilya Pollak [8] have used mid-price to predict short term changes in limit order books. The average of the best ask and the best bid is called the mid-price and it is capable of representing the approximate market price.

2. Esteban Moro et al [14] have found that market impact gets increased when the order size gets increased. Therefore order volume is important when creating the state factor.

3. When calculating the Knudsen number, Yoshihiro Yura et al [13] have considered the market depth with respect to the best price. 0 1 2 3 4 5 6 7 are depth indicators in the following figure.

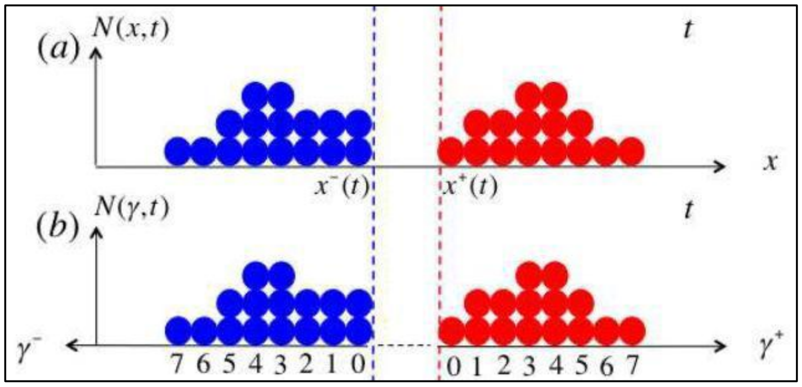


Figure 3.0-5 - Depth Indicators

However, when creating the state factor, weight values from five to one were given in the decreasing order to each row in the order book (Only 5 rows are there). Since the order book is a sorted structure and first row orders can cause a larger impact, larger weight value was given to the first row.

4. VWAP (Volume Weighted Average Price) of a stock over a specified market period is simply the average price paid per share during that period, so the price of each transaction in the market is weighted by its volume [16]. In order to define the state factor, similar weighting mechanism as the VWAP was used.

Considering above facts, state factor was defined as follows.

$$\frac{\sum_{i=1}^{n} \left[\frac{\left(Buy\,Volume(i) * Buy\,Price(i) + Sell\,Price(i) * Sell\,Volume\,(i)\right)}{(Buy\,Volume(i) + Sell\,Volume(i)}\right] * Depth(i)}{Total\,Depth}$$

n=5, $i\ \varepsilon$ *Set of rows*

Equation 5 - State Factor Function

The proposed state factor represents a price value between best bid and best ask prices. Volume, price and row depth were taken into consideration when introducing it. Hence it is a better indicator to represent an order book instance, than a mere price value.

3.3.5 Normalization

Calculated state factor values have spread over a wide range. In order to ease the evaluation, all the state factor values have been normalized using following method.

Min = Minimum State Factor

Max = Maximum State Factor

Normalized State Factor = 100((State Factor – Min) / (Max – Min))

Equation 6 - Normalization

3.3.6 Derived State

Stock market trends are described relevant to the overall direction of price movement. If the price movement has an upward direction it is an uptrend and if the price movement has a downward direction the trend is known as downtrend (Figure 3.5) [17].

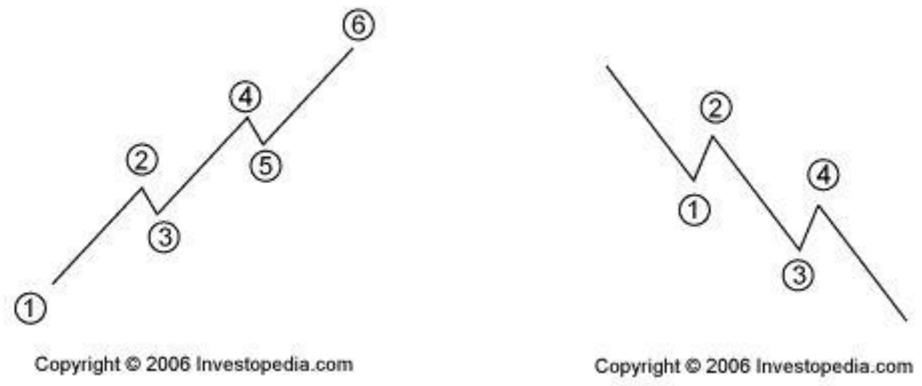

1
2
3
4
5
6
Copyright © 2006 Investopedia.com
2
1
4
3
Copyright © 2006 Investopedia.com

Figure 3.0-6 - Upward (left) and Downward (right) Trends

But trends are identified within a longer time period (years). Since the proposed research study focuses on short term predictions, consecutive price movements are used to derive the market state.

As stated in figure 2.3 it is clear that after a trade, there is a visible market impact. This impact can be analyzed using trade price movement variations. By considering two consecutive trade prices P1 and P2, market states can be derived as follows.

If (P1– P2) = 0 then State = Stable

If (P1 – P2) > 0 then State = Downward

If (P1 – P2) < 0 then State = Upward

By analyzing the trade data stream, relevant states after every trade can be identified.

3.3.7 Justification for the Proposed Model

Different techniques are available for the prediction of stock market. Some popular methods of these are Neural Network [11], Data Mining, Hidden Markov Model [7] and Hawkes Process [10]. From this Hidden Markov Model is the most leading machine learning techniques in stock market index prediction area [5].

When comparing Markov Chains with HMM, Markov Chain only focuses about the probabilities between state transitions. It does not identify what input causes that state transition. However, HMM consider about the hidden variable which affects to the state transition. Since the proposed research study focuses on predicting the market state for a given order, predictions have to be based on given input. Therefore, HMM is selected as the proposed model in this study.

3.3.8 Hidden Markov Model

The Hidden Markov Model (HMM) is a finite set of states, each of which is associated with a probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are “hidden" to the outside; hence the name Hidden Markov Model [5]. In order to define a HMM completely, following elements are needed.

1. The number of states of the model, N. Let denote the set of states, S = {S1, S2, …, SN}, where Si, i =1,2,….N is an individual state.
2. The number of observation instruments per state, M. Let denote the set of instruments V = {V1, V2,…VM}, where Vi, i =1,2,….M is an individual instrument
3. A set of state transition probabilities, that is, A= [aij], where aij = p{qt+1=Sj\qt=Si}, 1≤i,j≤N ; t=1, 2,… where qt denotes the current state.
4. The observation instrument probability distribution in each of the states, B = {bj(k)}, where bj(k)=P(Vk\Sj), 1≤j≤N, 1≤k≤M
5. The initial state distribution, π = [πi], where πi = P{q1 = Si},1≤ i ≤ N, such that $\sum_{i=1}^{N} \pi i = 1$

Proposed research study focuses on predicting market stability for a given order. Market stability is decided by trades and order book is the input to that decision. According to the given Hidden Markov Model definition, observation and state sequences are identified as follows, in this research context.

3.3.8.1 Observation Sequence

State predictions are made based on the observed data. In the financial market context, each trader can observe how the order book behaves throughout the day. Traders tend to place their orders by depending on the decisions they take by observing order book information. Therefore, order book details are identified as the observation sequence. For each order book instance, a state factor value is calculated (Chapter 3.3.4) and the value is normalized (Chapter 3.3.5).

According to H.M. Soner [15], after a trade happens a clearly visible market impact may occur. It can be either result in a stable price, price increment or a price decrement. Since a trade is the triggering point for a market impact, the idea is to consider a trade to trade epoch and to calculate the average state factor.

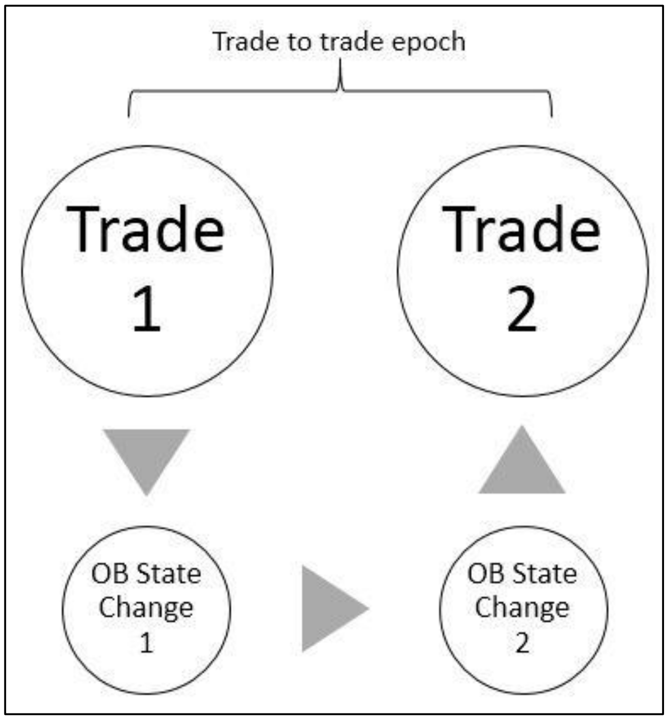


Figure 3.0-7 - Trade to Trade Epoch

Figure 3.6 shows that, after trade 1 occur, order book will result in OB State Change 1. Then because of a new order/ cancel order/ amend order, it can be moved into OB State Change 2. After that, trade 2 occurs. Average state factor value of OB State Change 1 and OB State Change 2 is identified as the observed value. Observation sequence contains a sequence of such derived state factor values.

3.3.8.2 State Sequence

By observing the state factor value sequence, market state; i.e., stable, upward or downward has to be derived. Even though the current market state and the order book details are visible, trader decisions towards the next possible trade are hidden from the total market aspect. For an example, for two traders as A and B, both can see the same order book. However, by depending on their personal decisions the next trade may vary. Therefore, the nth observation in a chain of observations is influenced by a hidden variable called personal decision of the trader. With the effect of the hidden variable, market state can get changed as stable, upward or downward. State sequence is generated by analyzing the trade details as explained in Chapter 3.3.6.

Hidden Markov Model training and evaluation is explained in Chapter 3.2 and it will be further discussed in Chapter 4.

Chapter 4

Implementation

This chapter discusses about the implementation details with respect to the proposed approach in Chapter 3. The chapter flow is also similar to the Chapter 3. Concepts and ideas introduced earlier will be described more and the problems occurred during the implementation and how they have been dealt will also be described in this chapter.

4.1 Technologies and Tools

MATLAB R2013a – MATLAB is a high level technical language and fourth generation programing language. Also a powerful, computing and interactive environment developed by Math Works [16]. Collaboration of specific capabilities like image and signal processing, numeric and neural computing, data analysis and visualization, programming and algorithm development, GUI with program written by other languages like C, C++, Java, Fortran and Python. MATLAB also provides functions relevant to Hidden Markov Models which ease the task of model training and predicting, i.e., *hmmtrain*, *hmmestimate, hmmgenerate, hmmdecode*, *hmmviterbi*.

4.2 Implementation Process

4.2.1. Design Overview

As stated in Chapter 3.2, dividing the dataset into testing and training datasets would not be an advantage in this research context. For an example, the model can be trained using April, May and June data, and based on the trained model, predictions can be done for the month July. But price patterns in different months may vary and the trained parameters may not match for the testing phase. Therefore, this needs an online learning approach where parameters get learned continuously. Then the next day predictions can be done by the model which was trained until the current day.

4.2.2 State Factor

Proposed research study targets an order book based market simulation approach. Therefore, the research began with analyzing only the order book data. In order to identify different states in the market, the first approach was to introduce a representative factor for each order book instance view and identifying different states from those values using a threshold.

The order book representative factor was named as "State Factor" and different approaches were taken into introduce the equation. Each order book view was defined with a separate State Factor value.

1. State Factor was first introduced as a mid-price value by following Deepan Palguna et al. and Su Chen et al. approaches.

   Mid price = (best bid price + best ask price)/2

   Even though best bid and best ask prices have a higher impact to the next trade price, order volume which is another important factor to the market impact, has left out. Since that data needed to be included in the state factor calculation, the equation was updated as follows.
2. By following the VWAP approach the new state factor was defined as,

   $$\frac{(Best\ Buy\ Volume * Best\ Buy\ Price + Best\ Sell\ Price\ * Best\ Sell\ Volume\ )}{(Best\ Buy\ Volume + Best\ Sell\ Volume)}$$

   Using the above updated equation, a more meaningful equation could be derived for the State Factor. However, the dataset provides 5 rows of order book data and the equation has been built on using only the first row. The values in other four rows, shows the trader belief towards that particular instrument. Some buyers have an intention to buy that stock for a lesser price than the best bid price and some sellers have an intention to sell that stock for a higher price than the best ask price. Those intentions are shown in the below four rows in the order book. After a trade occurs, first row will get matched and second row order may become next best orders. Therefore, the neglected four rows also have a potential towards the market impact and that needed to be included in the equation.
3. With the above argument and considering the approach taken by Yoshihiro Yura et al, the equation was updated by introducing depth values for the order book rows. The updated equation was introduced as the Equation 2 in Chapter 3.3.4.

4.2.3 Threshold Value

After introducing the state factor value, the next target was to derive the market state, using the calculated factor. A threshold value needed to separate the states as stable, upward or downward. As an initial value, threshold was selected as follows.
For two consecutive order book instances O1 and O2, is the calculated state factors are F1 and F2,

```
Difference = F2 – F1
Threshold (T) = 0.01
if difference < T and difference > - T, then "Stable"
if difference < - T, then "Downward"
if difference > T, then "Upward"
```

It is important to note that the threshold value was defined only by analyzing the dataset from the surface level. The idea was to update the threshold value using trial and error method. However, this value was not used at the end.

4.2.4 HMM using Threshold Value

Using the state details derived above, observation and state sequences were defined as follows.
Observation Sequence – state factors
State Sequence – relevant state between two state factors, derived using the threshold value

Example:

Observations:	15.0138	15.0138	14.9987	14.9987	14.9986
Difference :	-	0	-0.0151	0	-0.0001
States :	-	Stable	Downward	Stable	Stable

Even though observation and state sequences are defined as above, according to the HMM concept this is incorrect. HMM make predictions based on observations and the predicted state may have an effect from a hidden variable. However, in this approach states are derived using observation values (based on the threshold) and that does not need a HMM. This does not need a machine learning approach and if the observation value and the threshold can be identified correctly, then the relevant state can also be derived directly.

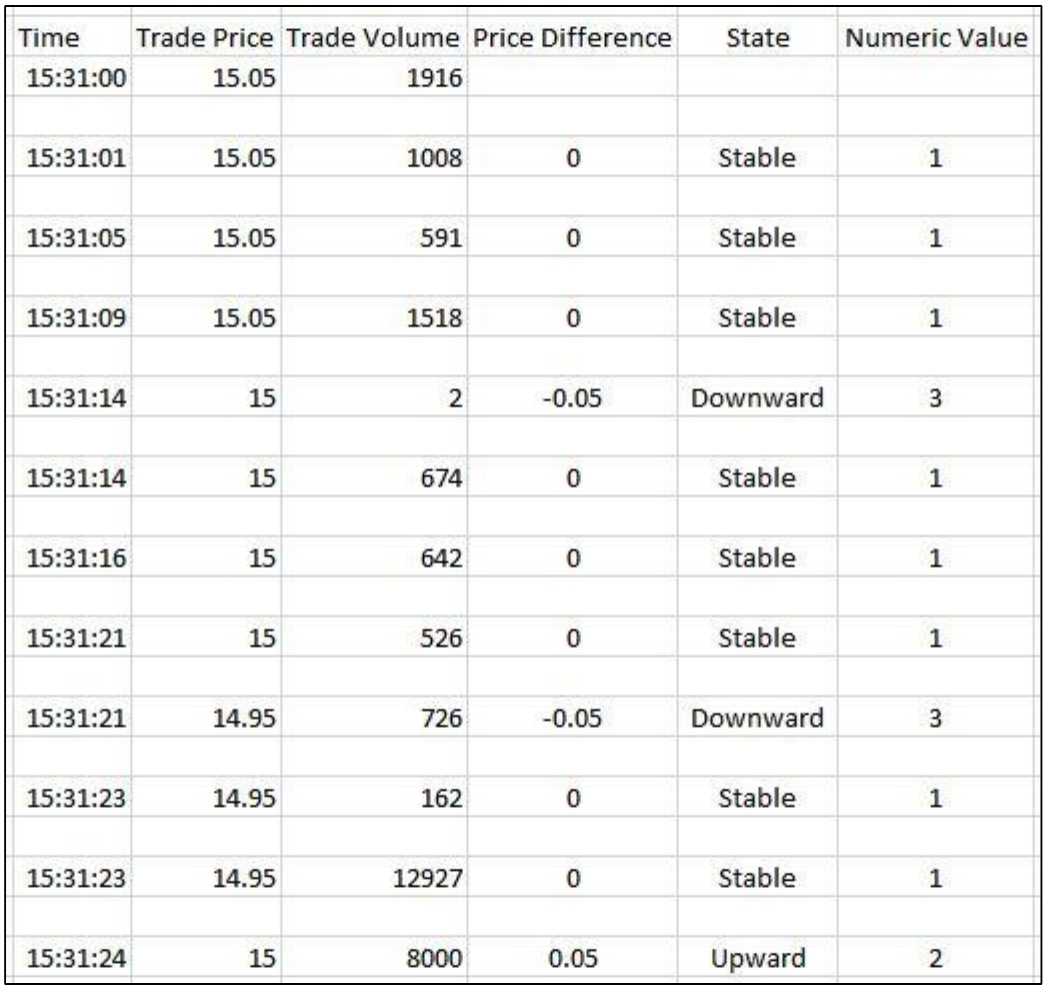
However, other than using a randomly guessed threshold value, a suitable method has to be found to identify separate states. DirectFN uses percentage change in the market trade price as a stability measure but that information was not accessible through the dataset. Therefore, another approach has to be followed based on trade data to derive stability measure of the market.

4.2.5 HMM using Trade and Order Book Data

The new approach derives the stability measure i.e., stable, upward or downward as stated in Chapter 3.3.6 using trade data. The Hidden Markov Model is designed as stated in Chapter 3.3.8 and the model implementation is as follows.

April 3rd, 2017 dataset will be taken as an example to explain the model training phase.

Trade Data:
By comparing the difference between two consecutive trade prices, stability measure is derived. When implementing the HMM, MATLAB only allows numeric values > 0 as the sequences. Therefore, stable = 1, upward = 2, downward = 3 were given.



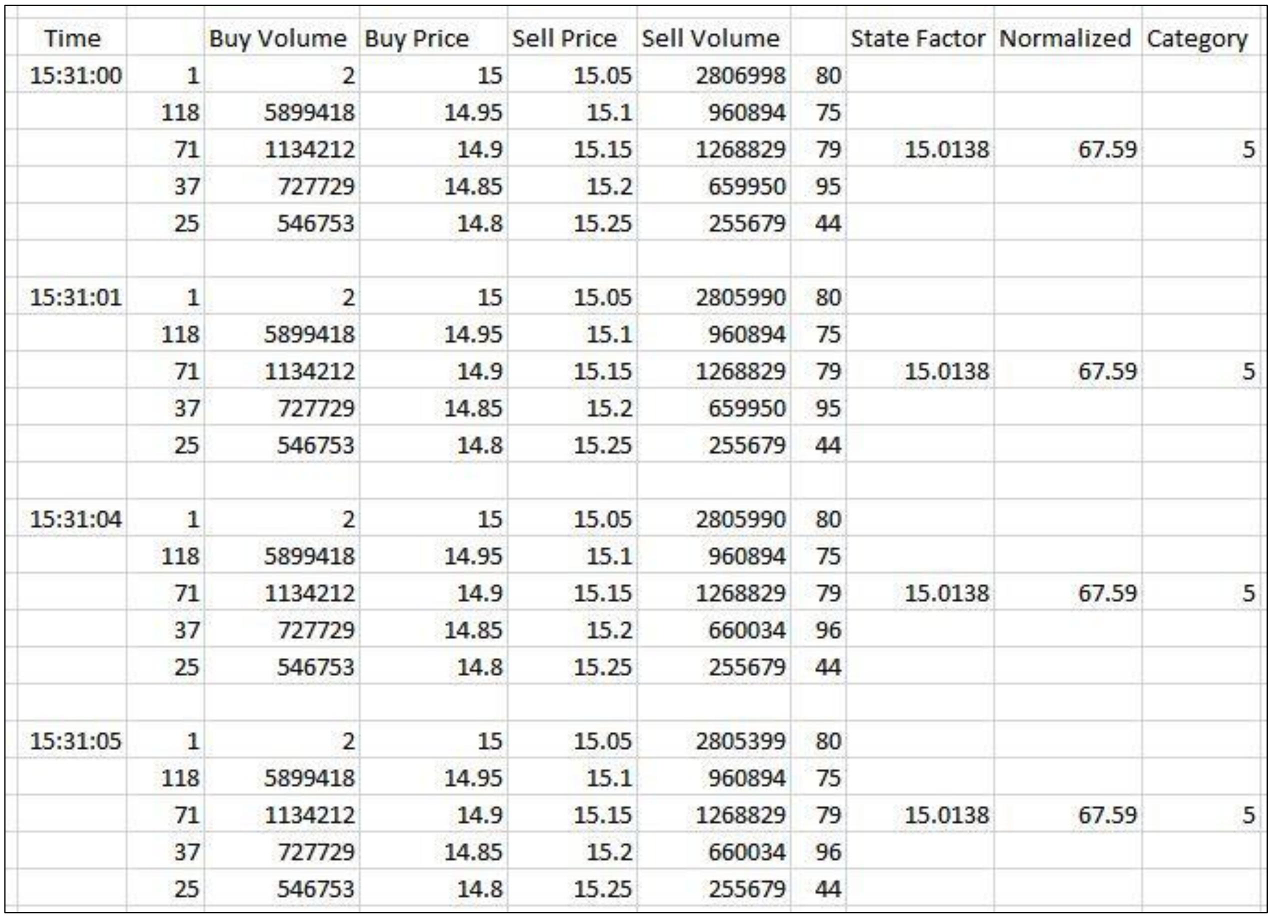
Time	Trade Price	Trade Volume	Price Difference	State	Numeric Value
15:31:00	15.05	1916			
15:31:01	15.05	1008	0	Stable	1
15:31:05	15.05	591	0	Stable	1
15:31:09	15.05	1518	0	Stable	1
15:31:14	15	2	-0.05	Downward	3
15:31:14	15	674	0	Stable	1
15:31:16	15	642	0	Stable	1
15:31:21	15	526	0	Stable	1
15:31:21	14.95	726	-0.05	Downward	3
15:31:23	14.95	162	0	Stable	1
15:31:23	14.95	12927	0	Stable	1
15:31:24	15	8000	0.05	Upward	2

Figure 4.0-1- Trade Data with Derived States

Order Book Data:

A State Factor value will be calculated for each order book instance and the value will be normalized using Equation 3. Normalized values are divided into six categories, to ease the analysis and the categories will be numbered as below since sequences should contain numeric values > 0.

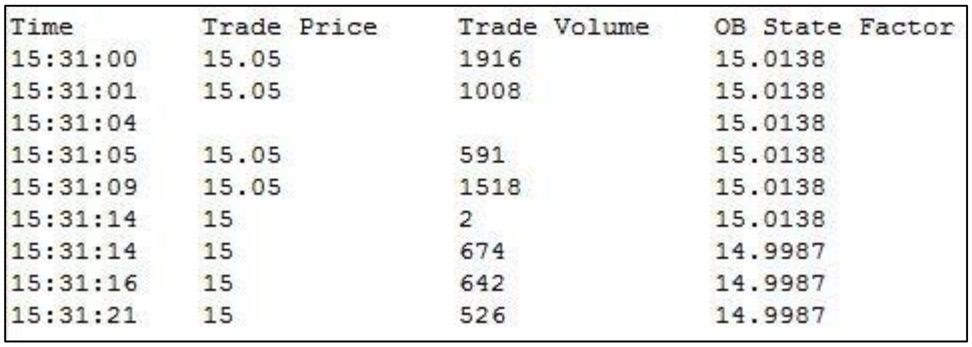
```
if StateFactor>=15 && StateFactor <30
           Factors(:,increment) = 1;
elseif StateFactor >=30 && StateFactor <40
           Factors(:,increment) = 2;
elseif StateFactor >=40 && StateFactor <50
           Factors(:,increment) = 3;
elseif StateFactor >=50 && StateFactor <60
           Factors(:,increment) = 4;
elseif StateFactor >=60 && StateFactor <70
           Factors(:,increment) = 5;
elseif StateFactor >=70 && StateFactor <80
           Factors(:,increment) = 6;
end
```



Time		Buy Volume	Buy Price	Sell Price	Sell Volume		State Factor	Normalized	Category
15:31:00	1	2	15	15.05	2806998	80			
	118	5899418	14.95	15.1	960894	75			
	71	1134212	14.9	15.15	1268829	79	15.0138	67.59	5
	37	727729	14.85	15.2	659950	95			
	25	546753	14.8	15.25	255679	44			
15:31:01	1	2	15	15.05	2805990	80			
	118	5899418	14.95	15.1	960894	75			
	71	1134212	14.9	15.15	1268829	79	15.0138	67.59	5
	37	727729	14.85	15.2	659950	95			
	25	546753	14.8	15.25	255679	44			
15:31:04	1	2	15	15.05	2805990	80			
	118	5899418	14.95	15.1	960894	75			
	71	1134212	14.9	15.15	1268829	79	15.0138	67.59	5
	37	727729	14.85	15.2	660034	96			
	25	546753	14.8	15.25	255679	44			
15:31:05	1	2	15	15.05	2805399	80			
	118	5899418	14.95	15.1	960894	75			
	71	1134212	14.9	15.15	1268829	79	15.0138	67.59	5
	37	727729	14.85	15.2	660034	96			
	25	546753	14.8	15.25	255679	44			

Figure 4.0-2- Order Book Data with Derived Categories

Figure 4.3 shows both trade and order book details and how the trade to trade epoch is calculated. 15:31:00 and 15:31:01 both time points have a trade and the order book view has updated because of the trade. State Factor is shown as 15.0138. However, at 15:31:04 a trade has not occurred and the order book has updated due to new order adding, order cancellation or order amendment. At 15:31:05 a trade has occurred and the observation value at this point would be the average of 15:31:04 and 15:31:05 time points' OB State Factor values.



Time	Trade Price	Trade Volume	OB State Factor
15:31:00	15.05	1916	15.0138
15:31:01	15.05	1008	15.0138
15:31:04			15.0138
15:31:05	15.05	591	15.0138
15:31:09	15.05	1518	15.0138
15:31:14	15	2	15.0138
15:31:14	15	674	14.9987
15:31:16	15	642	14.9987
15:31:21	15	526	14.9987

Figure 4.3- Trade to Trade Epoch Calculation

Example observation and state sequences are,

Observation:	5	5	5	5	5	5	6	6	6	5
State:	1	1	3	1	1	1	3	1	1	2

After the sequences have been generated, the Hidden Markov Model can be built.

MATLAB Statistics and Machine Learning Toolbox™ functions related to hidden Markov models are:

- *hmmgenerate* - Generates a sequence of states and emissions from a Markov model
- *hmmestimate* - Calculates maximum likelihood estimates of transition and emission probabilities from a sequence of emissions and a known sequence of states
- *hmmtrain* - Calculates maximum likelihood estimates of transition and emission probabilities from a sequence of emissions
- *hmmviterbi* - Calculates the most probable state path for a hidden Markov model
- *hmmdecode* - Calculates the posterior state probabilities of a sequence of emissions

Since, both observation and the state sequences can be derived from the dataset, '*hmmestimate'* function was used to estimate the transition and emission probabilities. If only the observation sequence was available, '*hmmtrain'* function has to be used to estimate the parameters.

```
[estimatedTR,estimatedEM] = hmmestimate(OBFactors,States);
```

Transition and Emission probability matrices for April 3rd, 2017 is as follows.

$$\texttt{estimatedTR} = \begin{matrix} 0.7612 & 0.1194 & 0.1194 \\ 0.5437 & 0.0684 & 0.3879 \\ 0.568 & 0.3861 & 0.0452 \end{matrix} \qquad \texttt{estimatedEM} = \begin{matrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

Figure 4.4 – Estimated Transition and Emission Matrices

Using the above estimated transition and emission probabilities for April 3rd, 2017, market stability probabilities can be predicted for April 4th, 2017.

```
[pStates] = hmmdecode(Factors,trans,em);
```

Since April 4th, 2017 dataset is available, the observation sequence and trade price variation; i.e., state changes are already available. Probabilities of the market stability, for the given observation sequence, can be generated using the trained HMM.

Predicted pState probabilities are as follows for a small observation sequence from April 4th, 2017.

Observations Sequence	5	5	5	5
Stable	0.7612	0.7122	0.7022	0.7001
Upward	0.1194	0.1452	0.1500	0.1512
Downward	0.1194	0.1426	0.1478	0.1487

Table 2 – Posterior Probabilities

If the trader places an order O1 to the market where the order book results in the category = 5, then there is a 0.7612 probability of market state becoming stable due to O1.

April 4th, 2017 observation sequence is used for the prediction in order to evaluate the model accuracy and to update model parameters. Otherwise, predictions can be generated for any observation sequence. Model results will be explained in detail in Chapter 5.

Chapter 5

Results and Evaluation

This chapter evaluates the research implementation and analyzes all statistical results. Evaluation is done in two main parts; i.e., Performance evaluation and Accuracy Evaluation. Accuracy Evaluation is mainly focused because false results may cost a lot of money for the user. Since a continuous evaluation approach has been used, the dataset have not been divided into training and testing datasets.

All the evaluation was done in a computer with following configurations. Operating System: Windows7 (64 bit), RAM: 4 GB, Processor: Intel Core i5, Processor Speed: 1.80 GHz.

5.1 Dataset

Order book data was collected from 3rd April, 2017 to 13th July, 2017 and trade data was collected from 3rd April, 2017 to 22nd June, 2017. During this period some datasets have been corrupted due to the termination of the server connection from DirectFN end. After informing them, the connection was re-established.

Each order book data file is around 50MB and contains around 250,000 order book update data frames and each trade data file is around 15MB and contains around 100,000 trade data frames. This huge dataset contains the total market behavior and for the proposed research study, evaluations will be done for considering only one instrument.

5.2 Accuracy Evaluation

Prediction accuracy is more important in this research as stated before. There are two methods to evaluate the predicted state sequence and they are as follows.

5.2.1 Using hmmgenerate function

‘*hmmgenerate’* function generates a sequence of states and observations from a Hidden Markov Model. By using the trained HMM, the next day state sequence can be predicted using this function and the similarity between the predicted state sequence and the real state sequence can also be measured. Accuracy was measured using following equation.

```
[Predicted Seq,Predicted States] = hmmgenerate(1000,trans,em);
```

Accuracy = 100 * numel (find (Real States==Predicted States))/numel(Real States);

numel = Number of elements

According to the above equations, results were taken as follows.

Date	Accuracy
19-Apr	55.6407
23-Apr	55.4181
24-Apr	57.9062
25-Apr	59.1976
5-Jun	57.0395
6-Jun	57.0829
7-Jun	55.65
8-Jun	57.5728
11-Jun	55.792
12-Jun	53.0926
18-Jun	56.085
19-Jun	59.5522

Table 3 – hmmgenerate Accuracies

Predictions done using ‘*hmmgenerate’* function has managed to maintain the predicted sequence accuracy, above 53%. However, directly predicting ‘this is what the market is going behave like’ is not acceptable in this domain because of the uncertainty of order prices. Therefore it would be much better to indicate the results using a probabilistic measure.

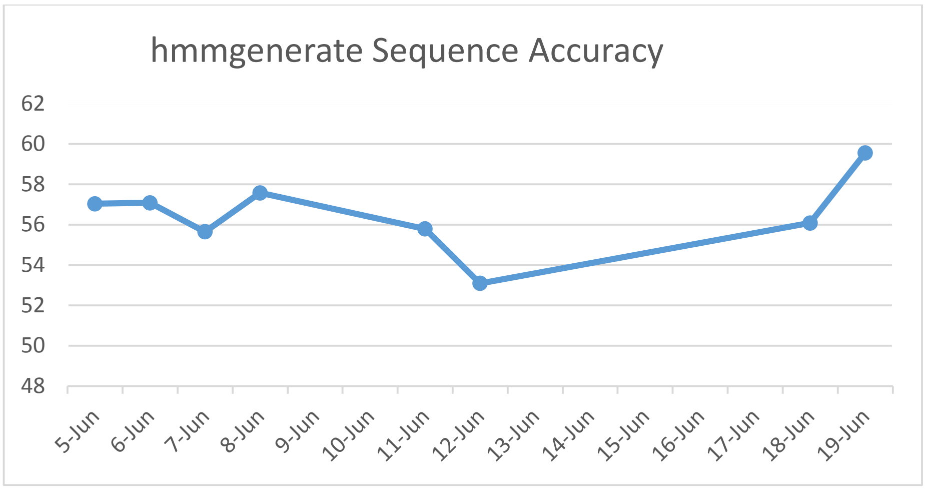


Figure 5.1 - hmmgenerate Sequence Accuracy

But these results may also benefit the trader, in a following situation. If the trader is aware about, how the market states are going to behave in the next day, for an example, if he knows that tomorrow between 1.30 p.m. and 2.00 p.m. the market is going to show an upward trend, then he can get ready beforehand and he can sell the shares he already has for a higher price. That way he will make more profit than placing a sell order to the stock market without having any clue.

However, the main intention of the proposed research study is to predict the impact caused by a particular buy/sell order. Therefore, another approach has to be used to achieve that objective.

5.2.2 Using hmmdecode function

'*hmmdecode'* function calculates the posterior state probabilities of a sequence of observations. The posterior state probabilities are the conditional probabilities of being at state k at step i, given the observed sequence. Using this function, the state probability can be calculated for a trader given the particular limit order. An example of such derivation was shown in Table 2.

It is important to note that the emission matrix in Figure 4.4 contains lot of zeros. That is because at 4th April, 2017 the model has seen only one day's sequence patterns. When the model is continuously trained, the emission matrix get updated. For an example, following are the model parameters trained until 12th April, 2017.

```
Transition matrix = [0.7802   0.1051   0.1148
                     0.5802   0.0282   0.3916
                     0.5382   0.4423   0.0195]
```

$$\text{Emission matrix} = \begin{bmatrix} 0 & 0.0390 & 0.2718 & 0 & 0 & 0.6596 & 0.0296 \\ 0 & 0.0396 & 0.2080 & 0 & 0 & 0.7238 & 0.0286 \\ 0 & 0.0373 & 0.2120 & 0 & 0 & 0.7223 & 0.0284 \end{bmatrix}$$

A prediction to be acceptable, the model should at least have seen different types of sequences. Therefore, before predicting the stability measures, the HMM was trained using 5 days data (April 03, 06, 10, 11, 12) and the predictions were done for the 19th April, 2017.
In order to compare the prediction probability and the real probability, following probability measures were taken into consideration.

For the real state sequence,

Stable probability = Number of stable occurrences / Total number of state occurrences

Upward probability = Number of upward occurrences / Total number of state occurrences

Downward = Number of downward occurrences / Total number of state occurrences

For the posterior state probabilities,

Mean and Standard Deviation values of stable, upward and downward posterior probabilities are considered

Following probability measures were taken after predicting April 19th, 2017 state sequence.

	Stable	**Up**	**Down**
Real Probability	0.7199	0.1396	0.1405
Predicted Mean Probability	0.761	0.1203	0.1187
Predicted Standard Deviation	0.0334	0.018	0.0154

For April 20th, 2017:

	Stable	**Up**	**Down**
Real Probability	0.7434	0.1296	0.127
Predicted Mean Probability	0.7268	0.1385	0.1347
Predicted Standard Deviation	0.0122	0.0069	0.0053

For April 23rd, 2017:

	Stable	**Up**	**Down**
Real Probability	0.7075	0.1458	0.1467
Predicted Mean Probability	0.7529	0.1239	0.1232
Predicted Standard Deviation	0.013	0.0072	0.0057

For April 24th, 2017:

	Stable	**Up**	**Down**
Real Probability	0.7601	0.12	0.12
Predicted Mean Probability	0.7352	0.1342	0.1306
Predicted Standard Deviation	0.002	0.0011	0.0008

For April 25th, 2017:

	Stable	**Up**	**Down**
Real Probability	0.7629	0.1183	0.1189
Predicted Mean Probability	0.7725	0.0703	0.1572
Predicted Standard Deviation	0.064	0.0736	0.0122

Graphical representation for April data using error bars, is as follows.

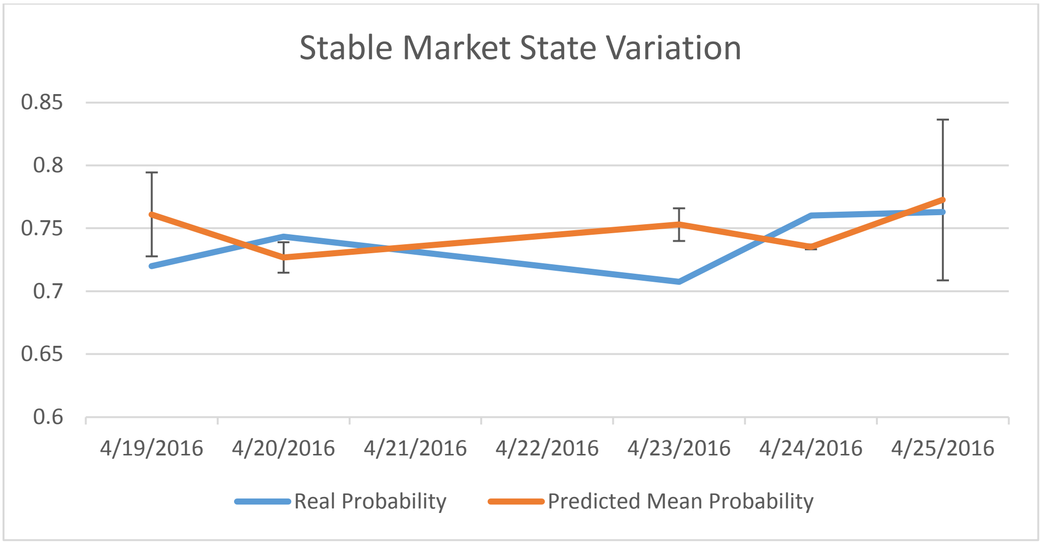


Figure 5.2- Stable Market Variation (April)

For the April dataset, Figure 5.2 shows the variation between how the real market has behaved in a particular trading day and what probabilities has the model predicted for that particular day. On 19th and 20th April, 2017, error bounds are closer to the real probability values, but fails to get in between the value. On 23rd and 24th April, 2017, error bounds have deviated from the real probability value and the predicted posterior probability is not anywhere closer to the real value. However, 25th April, 2017 shows that both real probability value and predicted posterior probability are in between the error bound.

However, it is important to highlight that there is only less than 0.1 deviation between the real behavior and the predicted behavior.

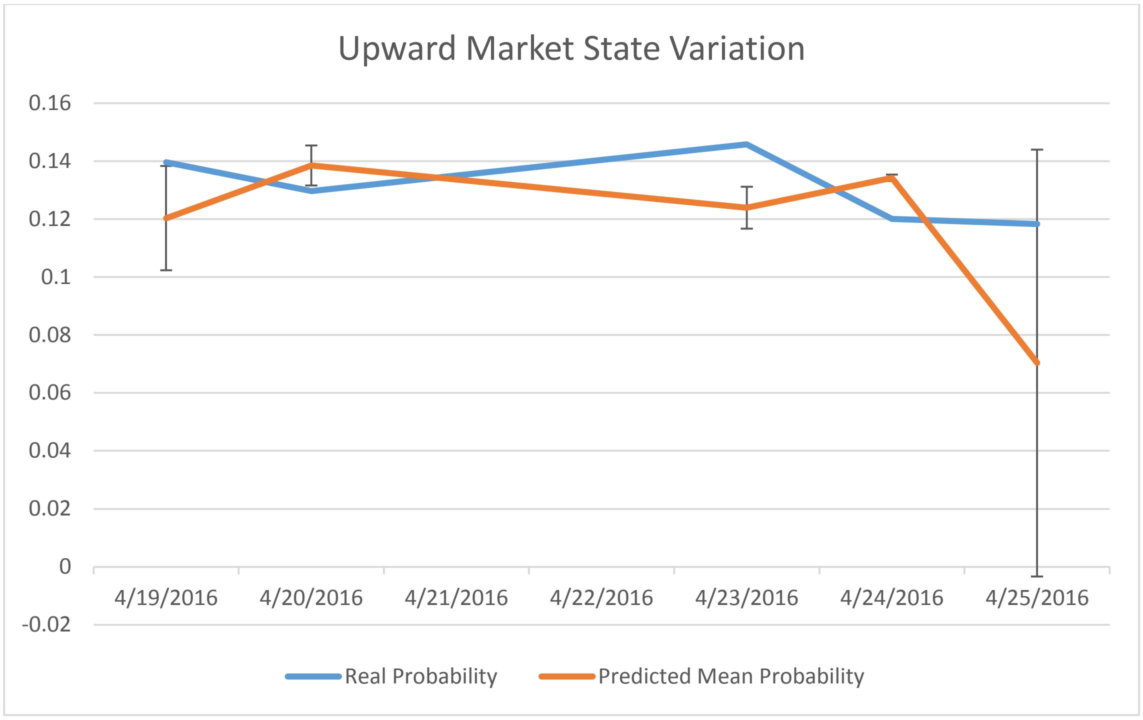


Figure 5.3 - Upward Market State Variation (April)

Figure 5.3 shows that the real probability of upward market state variation, agrees with the predicted values on 19th and 25th April, 2017, but not on 20th, 23rd and 24th April 2017. Again in Figure 5.4, downward market state variation is predicted only on 25th April, 2017.

According to the results generated for April month, the model has achieved a higher accuracy, when the model has been trained with more sequence patterns. By training from 3rd April, 2017 to 24th April, 2017, the model has been able to output an accurate state prediction for 25th April, 2017.

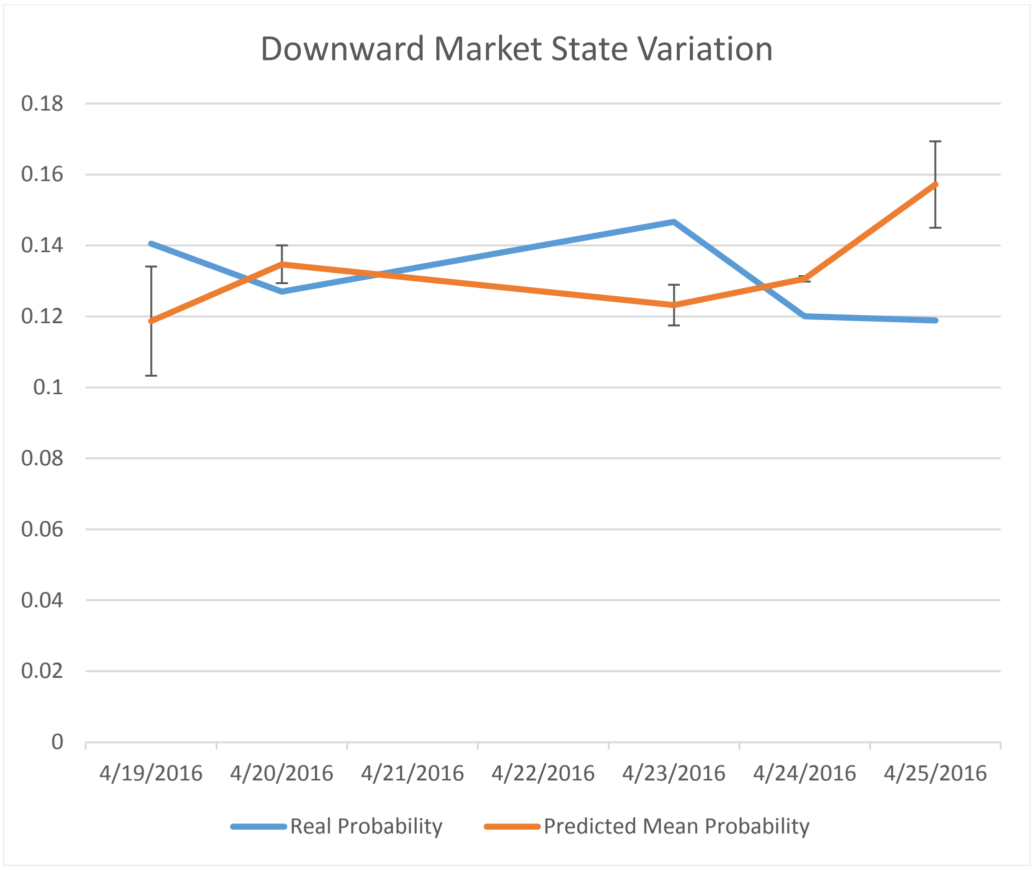


Figure 5.4 - Downward Market State Variation (April)

According to the research scope, only one instrument has been considered for the analysis and trade price variation for that instrument is as follows.

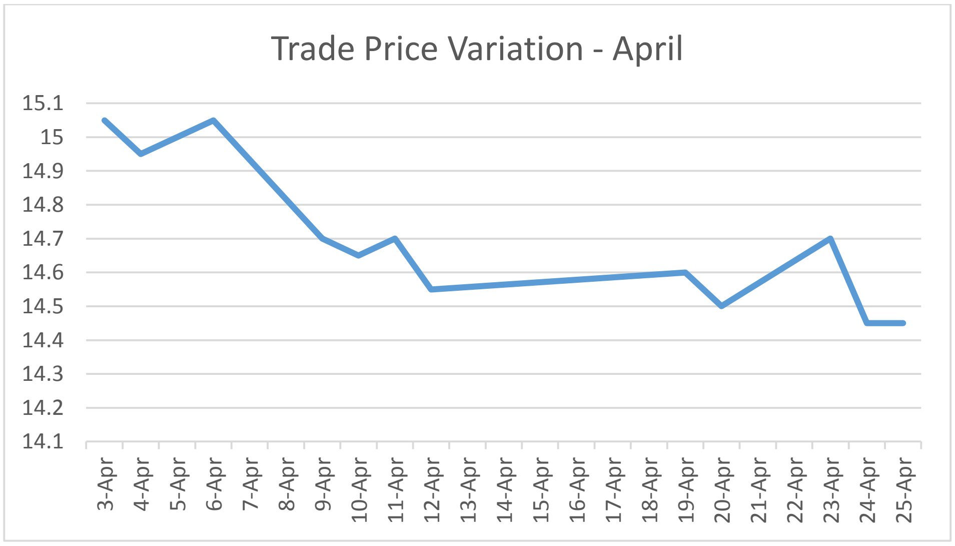


Figure 5.5 - Trade Price Variation (April)

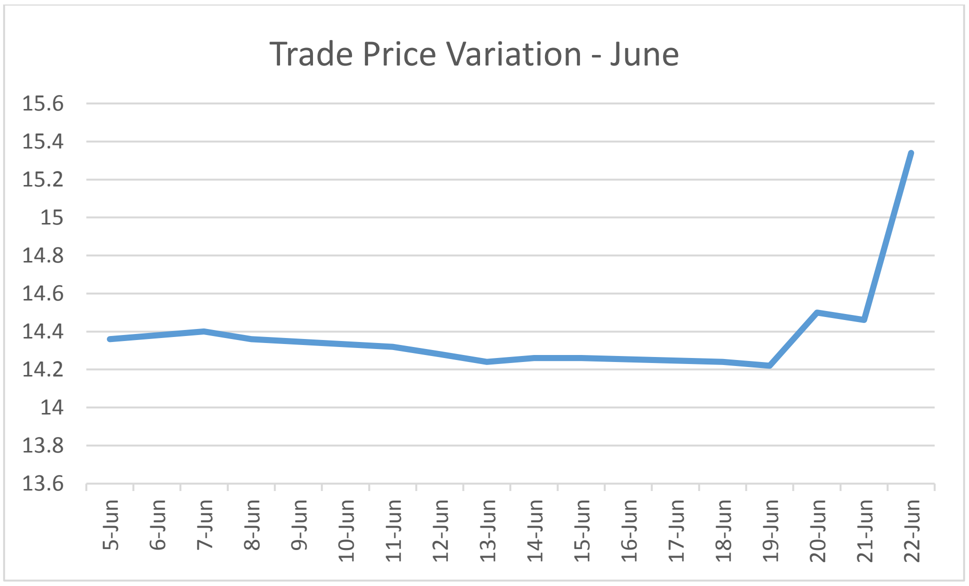


Figure 5.6 - Trade Price Variation (June)

From April to June, the market has an overall downward price trend. Even though the model was trained using a cumulative dataset, the transition and emission probabilities were resulted as follow, when trained until June 8th, 2017.

Transition matrix = [0.7876 0.1013 0.1111; 0.5863 0.0206 0.3932; 0.5380 0.4468 0.0152];

$$\text{Transition matrix} = \begin{bmatrix} 0.7876 & 0.1013 & 0.1111; \\ 0.5863 & 0.0206 & 0.3932; \\ 0.5380 & 0.4468 & 0.0152 \end{bmatrix};$$

$$\text{Emission matrix} = \begin{bmatrix} 0 & 0.0001 & 0.0160 & 0.8872 & 0.0008 & 0.0828 & 0.0131; \\ 0 & 0 & 0.0169 & 0.8689 & 0.0008 & 0.0991 & 0.0143; \\ 0 & 0.0002 & 0.0173 & 0.8700 & 0.0008 & 0.0970 & 0.0147 \end{bmatrix};$$

When the predictions were made using above probabilities for June 11th, 2017 following results were acquired.

	Stable	**Upward**	**Downward**
Real Probability	0.708	0.1521	0.1399
Predicted Mean Probability	nan	nan	Nan
Predicted Standard Deviation	nan	nan	Nan

For June 12th, 2017:

	Stable	**Upward**	**Downward**
Real Probability	0.7621	0.1173	0.1206
Predicted Mean Probability	nan	nan	Nan
Predicted Standard Deviation	nan	nan	Nan

Since the emission matrix fails to identify stability measures when the observation category = 1 (first column of the matrix contains zero), and since June 11th and June 12th observation sequences contain category = 1 observations, the predictions are resulted as 'nan' as shown above.

As a remedy for the above problem, to predict June month's stability measures, the model was trained only using June datasets. The result was as follows.

For June 11th, 2017:

	Stable	**Upward**	**Downward**
Real Probability	0.7146	0.1455	0.1399
Predicted Mean Probability	0.7378	0.1321	0.1301
Predicted Standard Deviation	0.0104	0.0049	0.0055

For June 12th, 2017:

	Stable	**Upward**	**Downward**
Real Probability	0.6841	0.1578	0.1581
Predicted Mean Probability	0.7438	0.1293	0.1269
Predicted Standard Deviation	0.0011	0.0006	0.0004

In this approach, the model identifies posterior probabilities for June 11th and June 12th observation sequences. Therefore, it is more efficient to use a model which was trained using a dataset at most 1 month prior to the prediction date.

Following are the results acquired for June month's predictions.

For June 13th, 2017:

	Stable	Upward	Downward
Real Probability	0.7182	0.1426	0.1392
Predicted Mean Probability	0.746	0.1281	0.1259
Predicted Standard Deviation	0.0011	0.0006	0.0004

For June 14th, 2017:

	Stable	Upward	Downward
Real Probability	0.7096	0.1447	0.1457
Predicted Mean Probability	0.7348	0.1338	0.1314
Predicted Standard Deviation	0.002	0.0012	0.0008

For June 15th, 2017:

	Stable	Upward	Downward
Real Probability	0.7194	0.1397	0.1409
Predicted Mean Probability	0.7414	0.1305	0.1281
Predicted Standard Deviation	0.0022	0.0013	0.0009

For June 18th, 2017:

	Stable	Upward	Downward
Real Probability	0.7629	0.121	0.1161
Predicted Mean Probability	0.7403	0.131	0.1287
Predicted Standard Deviation	0.0025	0.0014	0.001

For June 19th, 2017:

	Stable	Upward	Downward
Real Probability	0.7623	0.1215	0.1162
Predicted Mean Probability	0.729	0.1364	0.1347
Predicted Standard Deviation	0.0141	0.0067	0.0074

For June 20th, 2017:

	Stable	Upward	Downward
Real Probability	0.7623	0.1215	0.1162
Predicted Mean Probability	0.729	0.1364	0.1347
Predicted Standard Deviation	0.0141	0.0067	0.0074

Graphical representation for June data using error bars, is as follows.

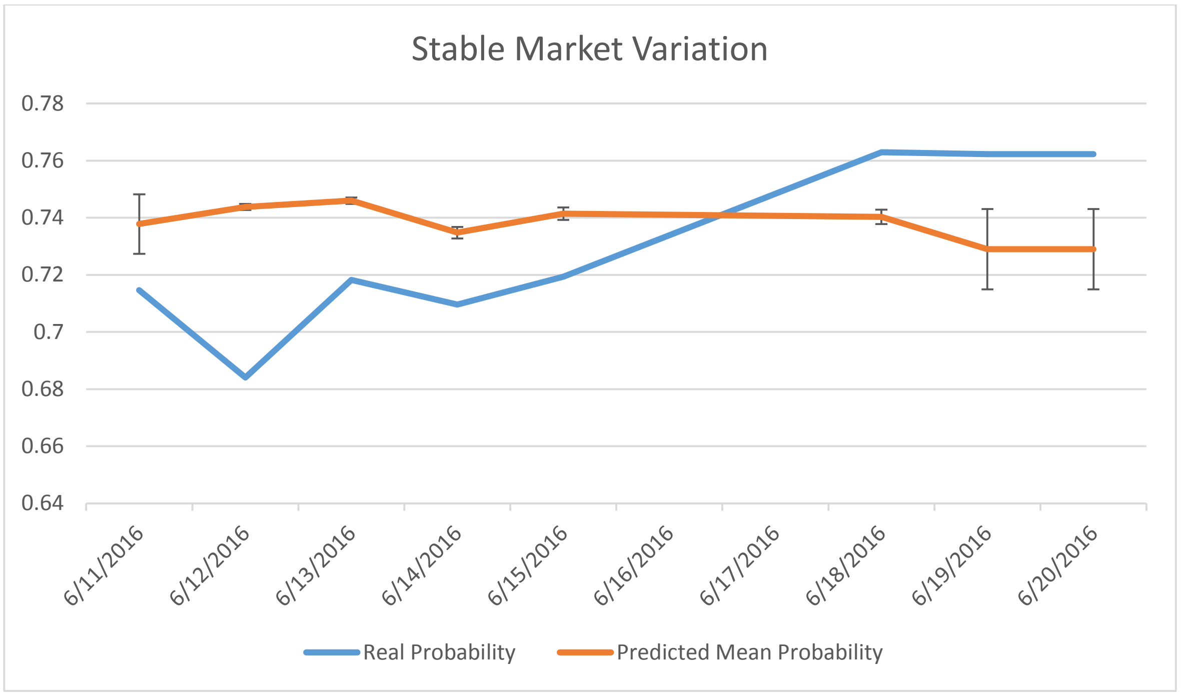


Figure 5.7 - Stable Market Variation (June)

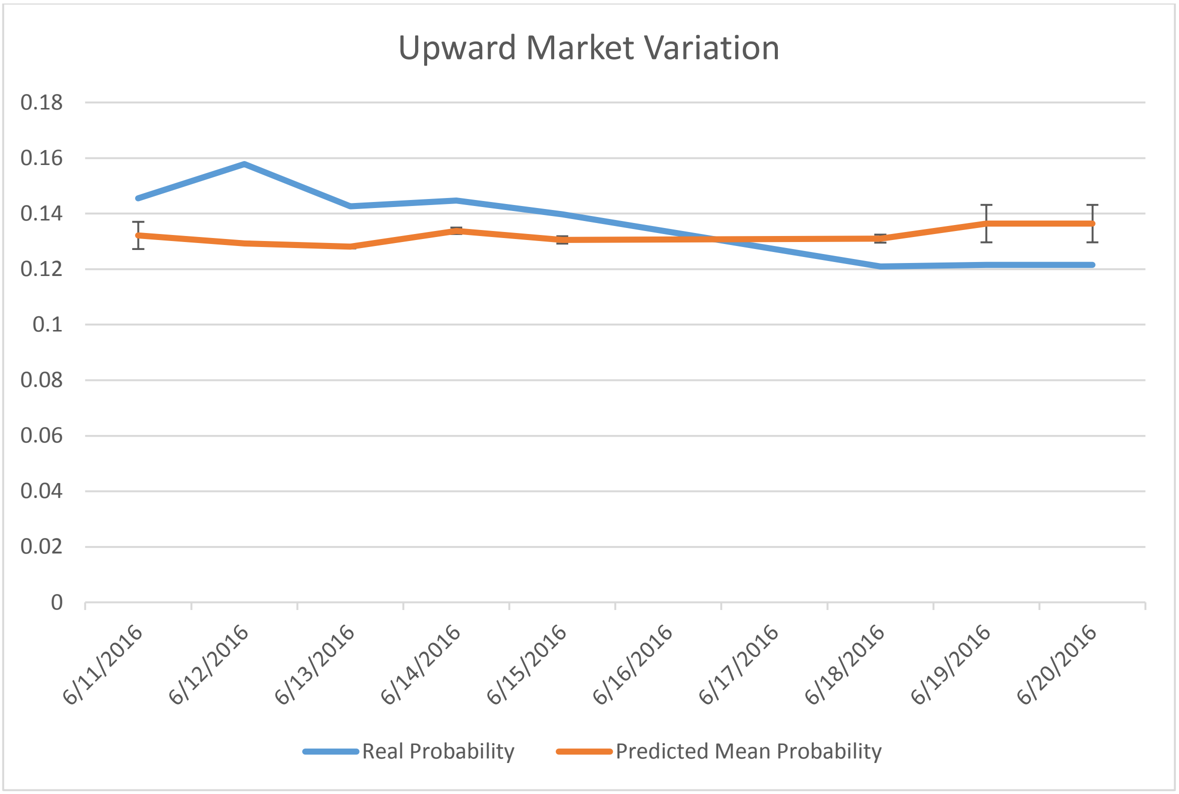


Figure 5.8 - Upward Market Variation (June)

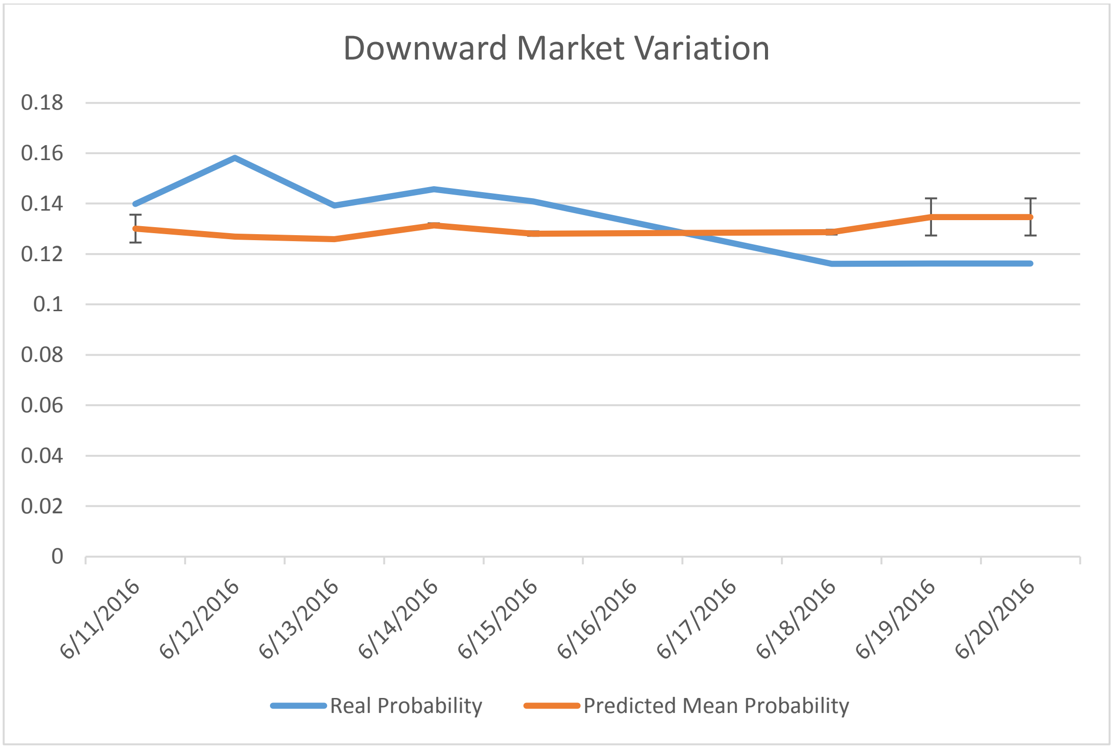


Figure 5.9 - Downward Market Variation (June)

The prediction for month June has an at most deviation of 4%. When comparing upward and downward graphs, it is clearly visible.

5.3 Performance Evaluation

Since the proposed research study uses a machine learning approach and a huge dataset has been used for model training, performance of the model needs to be evaluated. Following is a graphical representation of the Performance Evaluation for training April data. Minimum, average and maximum training times were considered by training the same dataset for 3 times. Sequence size of the dataset was also considered to observe the impact of sequence size, to the model training time. Training time was measured from the '*hmmestimate*' function execution time and all time measurements are in seconds.

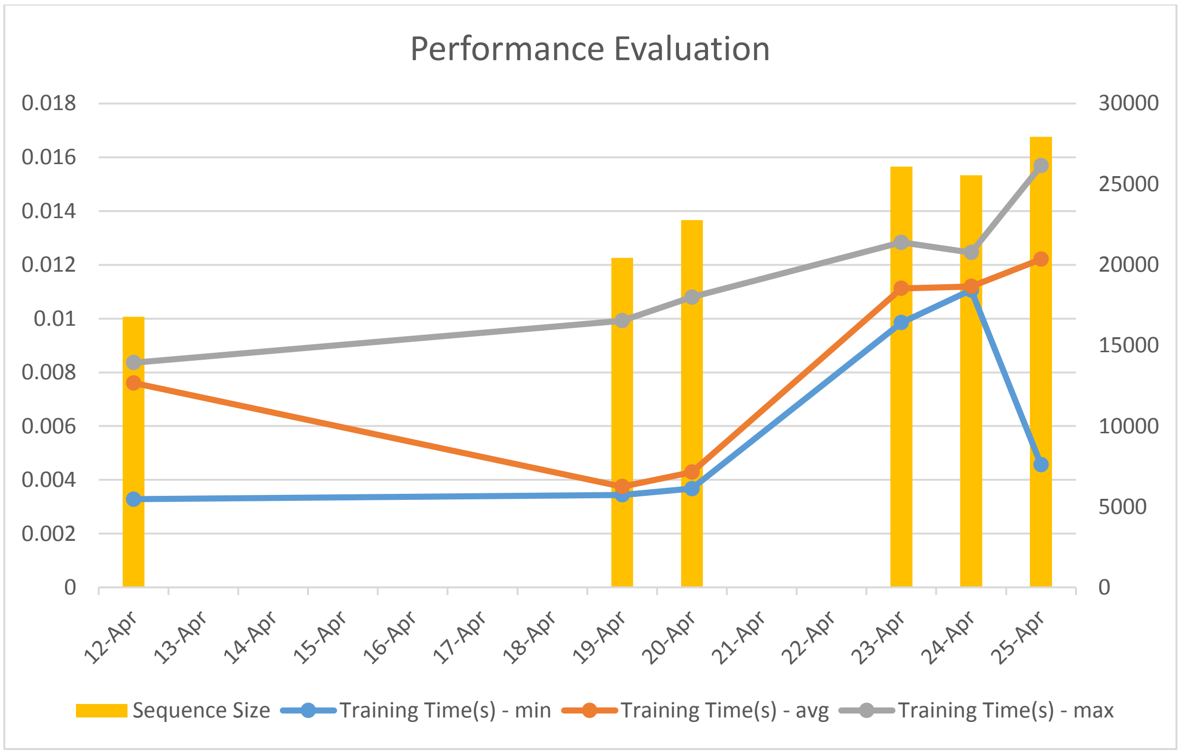


Figure 5.10 - Performance Evaluation

According to above evaluation, the training time gets increased along with the sequence size. For the same dataset, different training times have been observed depending on the efficiency the model works, at each time. Average training time lays between 0.003 and 0.013 seconds.

Chapter 6

Conclusions

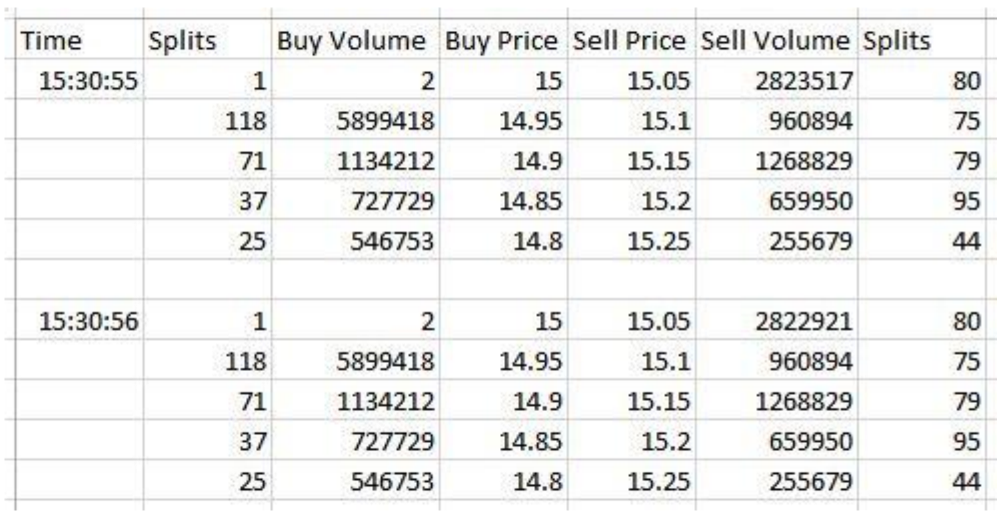
6.1 Introduction

This chapter focuses on concluding the evaluation results obtained in Chapter 5 and concluding about the research problem and research question stated under Chapter 1, based on the evaluation results. Limitations and implications for future research work, will also be discussed through this chapter.

6.2 Conclusions about Research Question

As stated in Chapter 1, the research question is, “Can we identify an upward, downward or stable trend in the market, by analyzing order book data and the incoming buy/sell order details**?”** Even though the implemented model doesn’t give a 100% accuracy, it is capable of identifying the market trend by observing the order book details. Incoming buy/ sell orders can be derived into an order book state factor if the current order book view is known. Given that the order book is visible to each trader, it is possible. For an example (Figure 5.1), if the trader views the following order book at 15:30:55 and decides to place a buy order of price 15.05 and quantity 596, we can deduce that it will result in an order book view similar to 15:30:56. After that, state factor can be calculated, normalized and market impact prediction probability can be derived from the model.

However, for a highly volatile market, predicting within minutes would not be effective. Therefore, the proposed model provides next day predictions. But still, as the research question highlights, trend identification can be done by analyzing order book observations.



Time	Splits	Buy Volume	Buy Price	Sell Price	Sell Volume	Splits
15:30:55	1	2	15	15.05	2823517	80
	118	5899418	14.95	15.1	960894	75
	71	1134212	14.9	15.15	1268829	79
	37	727729	14.85	15.2	659950	95
	25	546753	14.8	15.25	255679	44
15:30:56	1	2	15	15.05	2822921	80
	118	5899418	14.95	15.1	960894	75
	71	1134212	14.9	15.15	1268829	79
	37	727729	14.85	15.2	659950	95
	25	546753	14.8	15.25	255679	44

Figure 6.1 - Order Book Views

6.3 Conclusions about Research Problem

The main research problem was the inability of providing an indicator for market impact, for a given buy/sell order. This had influenced the trader and had reduced the effectiveness of trading algorithms. With the proposed model, such indicator for market impact is provided for a given sequence of observations. However, due the low accuracy of the implemented model, it needs to be tested more before practically using it. Research problem has been solved to some extent because the predictions can be done to identify the market impact via stability measures. Predictions for April month converges towards the real values and predictions for month June does not provide sufficient results. Therefore, more testing using a wide range of data, has to be done in order to provide a more accurate model.

6.4 Limitations

The major limitation that became apparent during the progress of the research, is predicting the stability measurements for the next day (i+1th Day). First intention was to derive the predictions for a short period of time, i.e., predictions to estimate the market impact beforehand, for an order which is going to place now. However, considering the higher market volatility and the inability to use a time related measurement for the model, the predictions were done for i+1th day. Therefore, by using a model trained up to the current date, state predictions can be done for a given order or given order sequence which will be placed on the next day.

6.5 Contributions

Following are the main contributions to the financial domain through this research study.

- Proposing a representative factor, i.e., State Factor to clearly indicate the information in one order book instance, as a single value. Based on different literature relevant to the financial domain, State Factor has been introduced by combining already published methods in order to achieve the proposed research idea. As a suggestion, a resource person from Millennium Information Technologies advised to include the time information to the state factor calculation. That approach would be explored as future work.

- A trained and tested Hidden Markov Model for the financial domain, to make predictions about the market impact caused from a given buy/ sell order, based on order book and trade information. Even though the model accuracy has to be increased for this model to be practically used, this approach is a good foundation towards prediction markets. Trading algorithm developers will have a publically available platform to test their implemented algorithms, in terms of market impact. Such algorithm may emit buy/ sell order signals and the model can predict how the market get an impact from those sequence of signals. Particular trader may benefit from the model by being aware about the market impact which would be caused from his buy/sell order.

- The difficulty of acquiring a proper order book dataset has limited the number of researches done on exploring order book data patters. Even though, order book data has gathered, most of the researches are done focusing long term predictions (Predictions for years) (Esteban Moro). Since this research has explored on order book data patters throughout the day and predictions are made for a shorter time period compared with years, this study would be a huge contribution for any researcher focusing on day by day order book data.

6.6 Implications for Further Research

There are several implications which can be explored as future work.

- Updating the state factor calculation in terms of time as stated above. By including the time information, predictions can be done for a particular time in the next day. As an improvement state factor can be updated by giving buy side a positive indicator and sell side a negative indicator. The motivation to indicate with different signs is because, if a particular stock has a higher demand (many buy orders) then the buy price increases, i.e., price moves upward. If a particular stock has a higher supply (many sell orders) then the sell price decreases, i.e., price moves downward. This approach can be explored since this research study has not considered it.

- As Chapter 1.7 states the research was evaluated with only one market instrument in order to complete the study within given time constraints. But the proposed model can be applied to any instrument (basic configurations may need to changed). Hence, the model can be tested against different market instruments for a longer time period and results can be evaluated.

- Derived state approach under Chapter 3.3.6 can be changed to derive the states using short term trends. Whether it will result in a way to increase the model accuracy, cannot be predicted. However, this approach can be followed as a future work.

References:

[1] Glantz, M., Kissell, R., Mun, J. and Paul, K. (2013). Multi-asset risk modeling: Techniques for a Global Economy in an Electronic and Algorithmic Trading. San diego: Elsevier academic Press, p.258.

[2] "Quantopian", Quantopian.com, 2017. [Online]. Available: https://www.quantopian.com. [Accessed: 29- Jul- 2017].

[3] "MetaTrader 4 Platform for Forex Trading", Metatrader4.com, 2017. [Online]. Available: https://www.metatrader4.com/en. [Accessed: 29- Jul- 2017].

[4] "Investopedia Stock Simulator", Investopedia.com, 2017. [Online]. Available: http://www.investopedia.com/simulator/. [Accessed: 29- Jul- 2017].

[5] I. Oladimeji, "Forecasting Shares Trading Signals With Finite State Machine Variant", Multidisciplinary Engineering Science and Technology, vol. 3, no. 4, pp. 4488-4493, 2016.

[6] H. Kaspi, S. Meyn and R. Tweedie, "Markov Chains and Stochastic Stability", Journal of the American Statistical Association, vol. 92, no. 438, p. 792, 1997.

[7] A. Kim, C. Shelton and T. Poggio, "Modeling Stock Order Flows and Learning Market-Making from Data", SSRN Electronic Journal, no., p. 8, 2002.

[8] D. Palguna and I. Pollak, "Mid-Price Prediction in a Limit Order Book", IEEE Journal of Selected Topics in Signal Processing, vol. 10, no. 6, pp. 1083-1092, 2016.

[9] W. Huang, C. Lehalle and M. Rosenbaum, "Simulating and Analyzing Order Book Data: The Queue-Reactive Model", Journal of the American Statistical Association, vol. 110, no. 509, pp. 107-122, 2015.

[10] S. Chen, C. Hu and Y. Zhou, "Order Book Simulator and Optimal Liquidation Strategies", no. 3, pp. 1-20, 2010.

[11] J. Sirigano, "Deep Learning for Limit Order Books", arXiv:1601.01987v7 [q-fin.TR], vol. 1, 2016

[12] Kyle, Albert (November 1985). "Continuous Auctions and Insider Trading". Econometrica. 56 (3): 129–176.

[13] Y. Yura, H. Takayasu, D. Sornette and M. Takayasu, "Financial Knudsen number: Breakdown of continuous price dynamics and asymmetric buy-and-sell structures confirmed by high-precision order-book information", Physical Review E, vol. 92, no. 4, 2015.

[14] E. Moro, J. Vicente, L. Moyano, A. Gerig, J. Farmer, G. Vaglica, F. Lillo and R. Mantegna, "Market impact and trading profile of hidden orders in stock markets", Physical Review E, vol. 80, no. 6, 2009.

[15] H. SONER, Trading with Market Impact. Swiss Finance Institute, 2015, pp. 7-10.

[16] S. Kakade, M. Kearns, Y. Mansour and L. Ortiz, "Competitive algorithms for VWAP and limit order trading", Proceedings of the 5th ACM conference on Electronic commerce - EC '04, 2004.

[17] I. Staff, "Uptrend", Investopedia, 2017. [Online]. Available: https://www.investopedia.com/terms/u/uptrend.asp?ad=dirN&qo=investopediaSiteSearch&qsrc=0&o=40186. [Accessed: 18- Dec- 2017].

Appendix

CalculateStateFactor.m

```
function stateFactor = calculateStateFactor(matrix)
[rows] = size(matrix);
depth = [5
         4
         3
         2
         1];
factors = [5,1];
for row = 1 : rows
     factors(row) = ((matrix(row, 3) * matrix(row, 4)) + ((matrix(row, 5) *
matrix(row, 6)))) / (matrix(row, 3) + matrix(row, 6));
end

weightedFactors = factors * depth;
stateFactor = weightedFactors/15;
end
```

Process.m

```
TradeMatrix = xlsread(filename,sheet2);
OBMatrix = xlsread(filename,sheet1);
[rows, columns] = size(TradeMatrix);
[OBrows, OBcolumns] = size(OBMatrix);
increment = 1; X =ones(1,1);Y =ones(1,1); Z = ones(1,1);
old = []; new =[];
States = ones(1,1);
Factors = ones(1,1);
increse = 1; stableCount = 0; upCount = 0; downCount = 0;
count = 1; temp=0; tempSFCalc = 0; noOfElements = 0;

for row = 3 : 2 : rows
    time = TradeMatrix(row,1);
    str = datestr(time, 'HH:MM:SS');
    sd1 = datenum(str,'HH:MM:SS' );

    price = TradeMatrix(row,2);
    Y(:, increment) = price;
    old = TradeMatrix(row-2,2);
    new = price;
    diff = old-new;
    if diff==0  %S
        States(:,increment) = 1;
        stableCount = stableCount + 1;
    elseif diff< 0  %U
        States(:,increment) = 2;
        upCount = upCount + 1;
```

```
elseif diff> 0  %D
    States(:,increment) = 3;
    downCount = downCount +1;
end

for OBrow = count : 6: count+6
    if OBrow == OBrows-4
        break
    end
    count = count + 6;
    temp = calculateStateFactor(OBMatrix(OBrow:OBrow+4,1:7));
    normalizedSF = normalize(temp);
    OBtime = OBMatrix(OBrow,1);
    OBstr = datestr(OBtime, 'HH:MM:SS');
    sd2 = datenum(OBstr,'HH:MM:SS' );
    if (sd1>sd2)
        tempSFCalc = tempSFCalc + normalizedSF;
        noOfElements = noOfElements + 1;
    elseif(sd1<sd2)
        count = count - 6;
        T = X(1,increment-1);
        tempSFCalc = tempSFCalc + T;
        noOfElements = noOfElements + 1;
        break;
    elseif(sd1==sd2)
        tempSFCalc = tempSFCalc + normalizedSF;
        noOfElements = noOfElements + 1;
        break
    end
end

if tempSFCalc == 0
    tempSFCalc = X(1,increment-1);
else
    tempSFCalc = tempSFCalc / noOfElements;
end

X(:,increment)= tempSFCalc;   %tempSFCalc;
Z(:,increment)= normalizedSF;
if tempSFCalc>=10 && tempSFCalc<20
    Factors(:,increment) = 1;
elseif tempSFCalc>=20 && tempSFCalc<30
    Factors(:,increment) = 2;
elseif tempSFCalc>=30 && tempSFCalc<40
    Factors(:,increment) = 3;
elseif tempSFCalc>=40 && tempSFCalc<50
    Factors(:,increment) = 4;
elseif tempSFCalc>=50 && tempSFCalc<60
    Factors(:,increment) = 5;
elseif tempSFCalc>=60 && tempSFCalc<70
    Factors(:,increment) = 6;
elseif tempSFCalc>=70 && tempSFCalc<80
    Factors(:,increment) = 7;
end
```

```
    increment = increment +1;
    tempSFCalc = 0;
    noOfElements = 0;
end
```

CurrentDayProbabilities.m

```
total = size(States);
stableProb = stableCount / total(1,2);
upProb = upCount / total(1,2);
downProb = downCount / total(1,2);
```

hmm.m

```
tic
[estimateTR,estimateE] = hmmestimate(Factors,States);
toc

[seq,s] = hmmgenerate(1000,trans,em);
[pStates, log, forward, backward] = hmmdecode(Factors,trans,em);
```