Modeling high frequency limit order book dynamics with machine learning models

Shuyue Fu, Shengdong Liu, Xinyu Peng

Abstract

We proposed a machine learning framework to capture certain patterns within the high frequency limit order books of E-mini S&P500 future on 2015/09/17. By characterizing each entry in a limit order book with a vector of attributes such as book imbalance, weighted average price, we are able to rebuild the limit order book into a dataset containing possible prediction attributes and directional price movement as our target value. With the help of Logistic regression and Support vector machines, we are able to build learning models for the limit order book to predict next price movement whenever there is a trade entry in the LOB. Experiments with real data prove that features selected by the proposed data framework are effective in the high frequency environment. We also look into the impacts of iceberg orders on our models.

Keywords: high frequency trading; limit order book; logistic regression; support vector machines; iceberg orders; machine learning

1. Introduction

A limit order is an order to buy or sell a given quantity of stock at a specified limit price or better. The size is the number of shares to be bought or sold. An order remains in the order book until fully executed, i.e. until its size is zero as a result of trades. Partial executions occur as a result of trades for less than the entire size of the order.

In this project, we used the limit order book of E-mini S&P500 Future of CME GLOBEX on September 17th 2015. We applied learning models on certain attributes like liquidity derived from this limit order book to capture the dynamics and forecast movements of the execution price. We also looked into the impacts of iceberg orders on our model.

2. Model and Methodology

2.1 Logistic Regression

Logistic regression classifier takes features and multiply each one by a weight and then adds them up. Then the result will be put into the sigmoid function, and return a number between 0 and 1. Anything above 0.5 will be classified as a 1 while anything below 0.5 will be classified as a 0. One can also think logistic regression as a probability estimation.

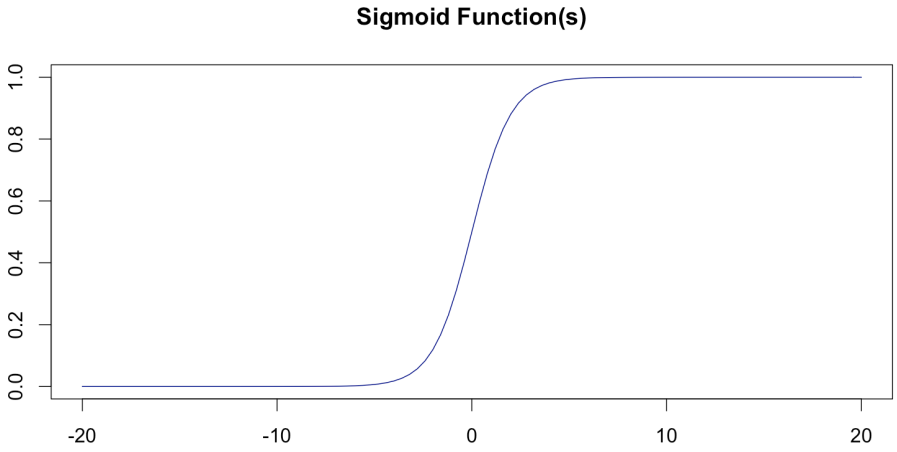


Figure 2.1

In this project, we use gradient ascent to build our logistic regression model. Gradient ascent is based on the idea that in order to find the maximum point on a function, the best way to move is through the direction of the gradient. The gradient is defined as the symbol ∇, and the gradient of a function f(x, y) is given by the equation:

$$\nabla f(x,y) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix}$$

The gradient ascent algorithm shown in Figure 2.2 takes a step in the direction given by the gradient. The gradient operator will always point to the direction of the greatest increase. The magnitude, or step size, is given by the parameter α. In vector notation, the gradient ascent algorithm is:

$$\boldsymbol{w} := \boldsymbol{w} + \alpha \nabla_{\mathbf{w}} f(w)$$

Such step is repeated until we reach a stopping condition: either a specified number of steps or the algorithm is within a certain tolerance margin.

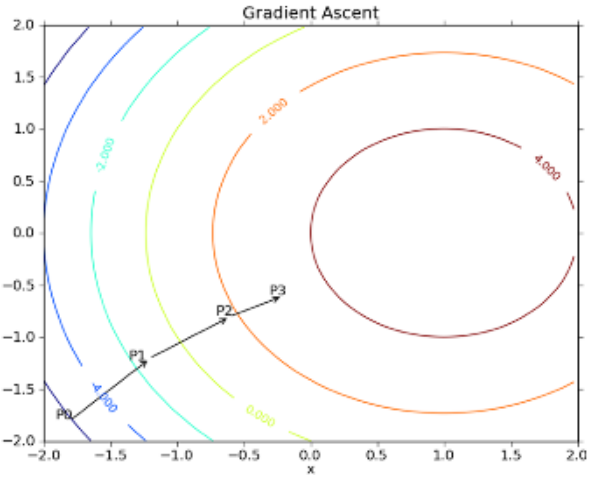


Figure 2.2

2.2 Support Vector Machines

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

In SVM, data points are classified and divided into different disjoint spaces. So its main objective is to construct the separation hyper-plane, such that the margin of separation between states is maximized. The hyper plane is called the "maximum-margin hyper-plane," in which the distances between the hyper-plane and the nearest point (*support vectors*) from different groups are maximized. For binary case, the decision boundary is defined mathematically as:

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

The mapping is performed by a kernel function $K(x, x_i)$, the choice of the kernel function is one of the parameters of SVM in most non linear cases, radial kernel is the most common choice.

3. Feature selection and visualization

3.1 Data overview

In this project, we build our model based on the limit order book in 2015/09/17. The shows the original structure of the limit order book. There are two types of record updates: when there is an order submission or cancelation, a book record update will show up and when there is a trade, a trade record update will show up. This structure is also known as the microstructure data.

```
00:00:18.146,BOOK10DEEPREC,
47,28,198325,59,33,198350,42,26,198375,74,41,198400,63,35,198425,142,31,198450,67,36,19
8475,84,32,198500,33,24,198525,17,12,198550,198575,7,9,198600,22,38,198625,24,30,198650
,37,52,198675,33,54,198700,38,113,198725,25,49,198750,30,86,198775,32,82,198800,42,263
00:00:19.194,BOOK10DEEPREC,
47,28,198325,59,33,198350,42,26,198375,74,41,198400,63,35,198425,142,31,198450,67,36,19
8475,84,32,198500,33,24,198525,117,13,198550,198575,7,9,198600,22,38,198625,24,30,19865
0,37,52,198675,33,54,198700,38,113,198725,25,49,198750,30,86,198775,32,82,198800,42,263
00:00:19.195,TRADEREC,1,198575,25133,7
00:00:19.195,TRADEREC,1,198575,25134,7
00:00:19.195,TRADEREC,2,198575,25136,7
00:00:19.195,TRADEREC,1,198575,25137,7
00:00:19.195,TRADEREC,1,198575,25138,7
00:00:19.195,TRADEREC,1,198575,25139,7
00:00:19.195,BOOK10DEEPREC,
59,33,198350,42,26,198375,74,41,198400,63,35,198425,142,31,198450,67,36,198475,84,32,19
8500,33,24,198525,117,13,198550,1,1,198575,198600,22,38,198625,24,30,198650,37,52,19867
5,33,54,198700,38,113,198725,25,49,198750,30,86,198775,32,82,198800,42,263,198825,27,76
```

Figure 3.1

Every book record provides time, 10 levels of bid and ask price and their corresponding quantity. Every trade record provides time, the trade price and volume. In other words, this microstructure has provided us the price level and liquidity state of the market in every micro-second. Therefore, the interpretation of the microstructure is the most significant challenge. What kind of pattern is contained in microstructure? What features or attributes can we extract from these data? How to rebuild the LOB so that data can be implemented in building predictive models?

We select some basic indicators which are listed in table below, and try to visualize these indicators to show whether they possess possible prediction power or patterns. We plot the histogram of indicators both when the price is going up and down to see if different value can lead to different price movements. We also plot the scatter point using different colors and different indicators value as x and y axis to see if combing different indicators value can help us divide the target value. (The directional movements of price). Figure 3.1 and 3.2 are several examples.

Liquidity Attributes	Price Attributes	Trend Attributes
best ask/bid size	average ask price	volume change
total ask/bid size	average bid price	number of consecutive trades with same direction
size imbalance	weighted average price	changes of all liquidity attributes
		changes of all price attributes

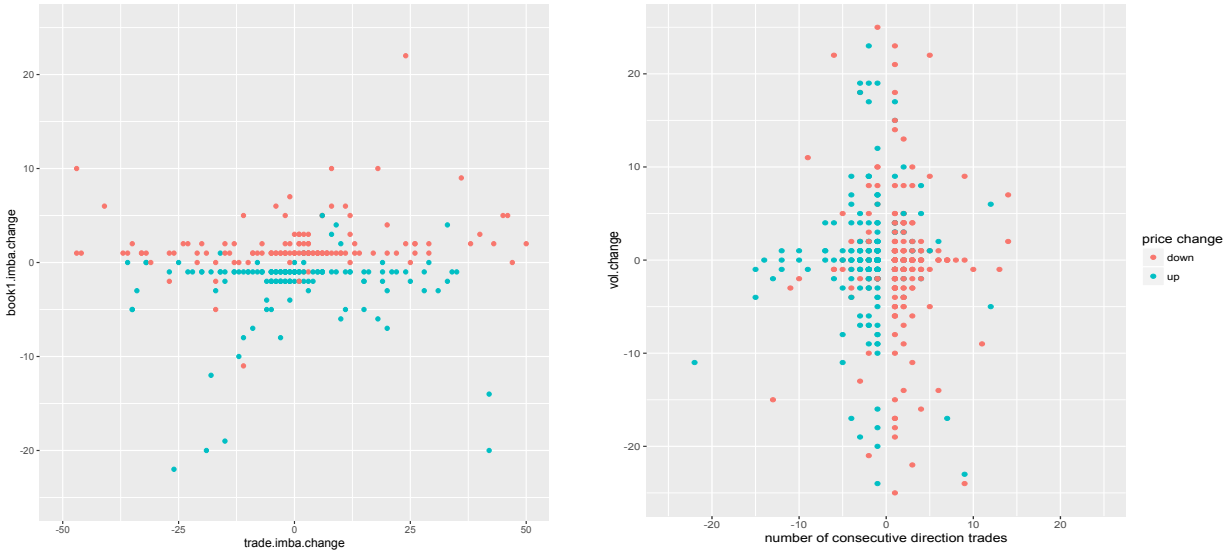


Figure 3.2

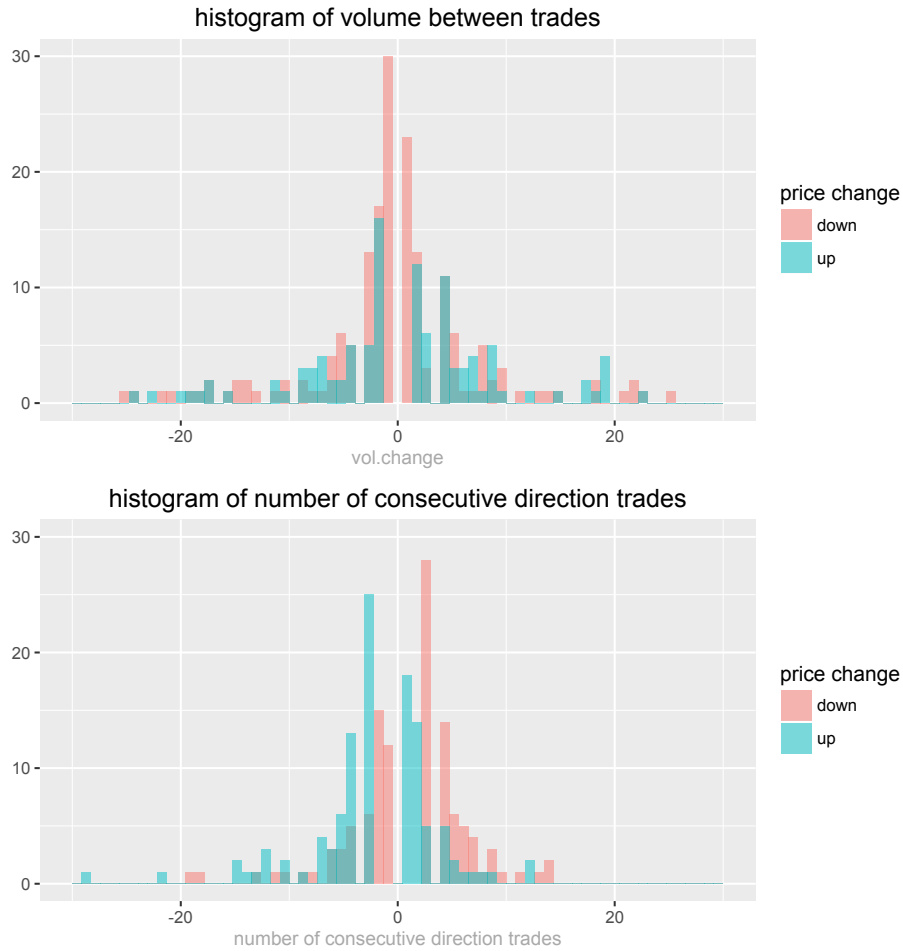


Figure 3.3

3.2 Building Dataset

We select the indicators that shows the prediction power like the above examples, use them as the attribute (inputs) to form the dataset we plan to use in building machine learning models. And we use the directional price movement as the target.

Whenever a trade happens, we calculate the current liquidity and price attributes and deltas of these attributes with previous trade updates and previous book update. And use the next execute price movement as target value.

If we add more delta of attributes, the dataset will become really large and affect the efficiency of the model. So for the sake of efficiency, which is extremely important in high frequency trading, we only use two datasets to build our model:

I. Price and liquidity attributes plus deltas of these attributes with previous ***one*** book updates and previous ***three*** trade updates
II. Price and liquidity attributes plus deltas of these attributes with previous ***three*** book updates and previous ***three*** trade updates

Note: The definition and detailed attributes we used are listed in appendix 1.

4. Price movement prediction

4.1 Logistic Regression

The result of the model:

LAG	TRAINING SET	TESTING SET	UP_RATE	DOWN_RATE	TOTAL ACCURACY
LAG1	80.00%	20.00%	74.57%	84.39%	79.52%
LAG1	20.00%	80.00%	60.61%	90.72%	75.74%
LAG1	50.00%	50.00%	72.96%	84.67%	78.85%
LAG3	80.00%	20.00%	70.72%	74.58%	72.64%
LAG3	20.00%	80.00%	54.39%	71.57%	62.99%
LAG3	50.00%	50.00%	81.77%	55.31%	68.40%

Note: Lag1 represents the first dataset listed in section 3.2 and Lag2 represents the second dataset.

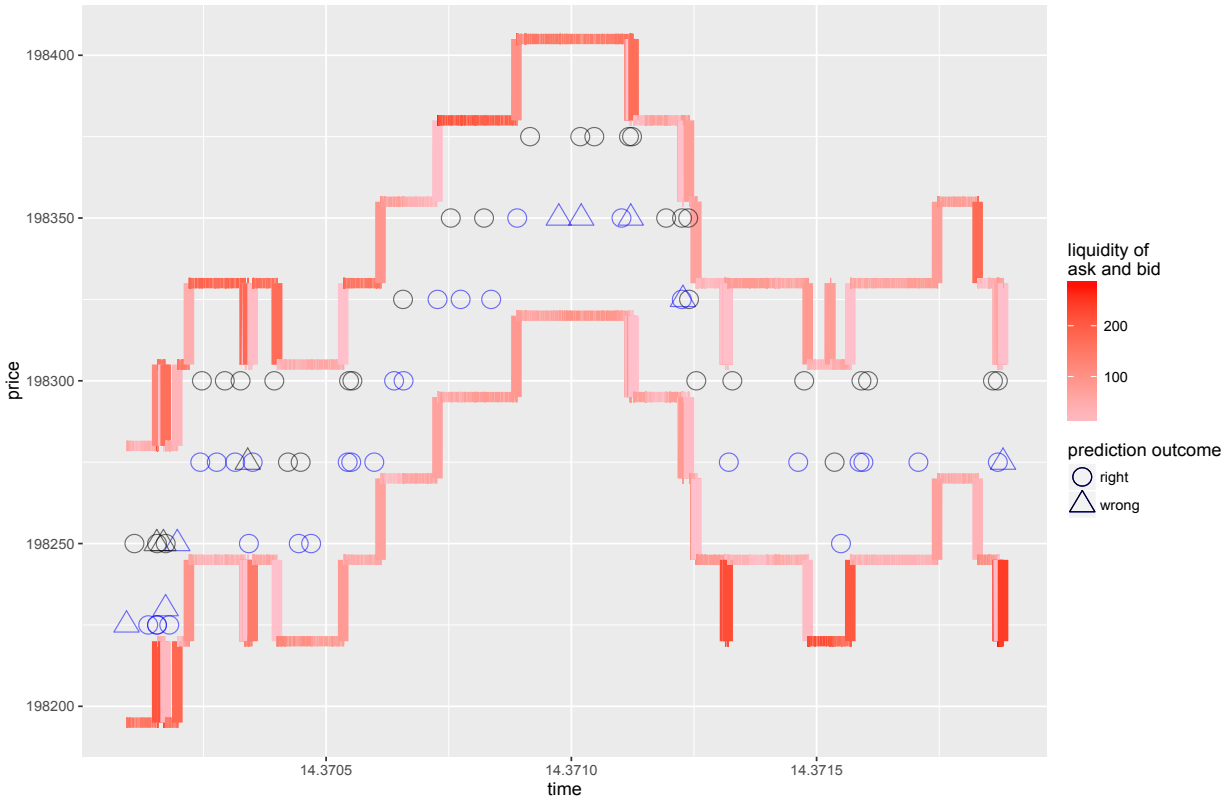


Figure 4.1

Figure 4.1 shows logistic regression's first 70 prediction results. The red lines represent the price movement of best ask and best bid price where the gradient color shows the liquidity state of ask side and bid side. The circle shows the right predictions while the triangle represents wrong predictions. And the blue circle/triangle means the model predicts the price will go up while the black one means the model predicts the opposite.

Conclusion:

1. When more training data are used, prediction accuracy will be higher.

2. From the dataset which includes one previous book update, down-accuracy is always higher than up-accuracy, which indicates that the price has a downward moving trend. But in the dataset which includes three previous book updates. When 50% of the data are used as training set, up accuracy is higher. This is because the model makes more predictively for up at the cost of lower the down accuracy.

4.2 Support Vector Machine

Like Logistic regression, two different datasets described in section 3.2 are used to train SVM model. One of them includes indicators from one previous book update and the other one includes indicators from three previous book updates. For each dataset, the effects of different training to testing set ratio to prediction accuracy are also compared. The training/testing ratios are: 80%-20%, 20%-80% and 50%-50%.

The effects of different kernel functions used in the SVM model on the final results are also tested. And the radial function is proved to be the most efficient kernel function.

LAG	TRAINING SET	TESTING SET	UP ACCURACY	DOWN ACCURACY	TOTAL ACCURACY
LAG 1	80%	20%	85.57%	86.46%	86.02%
LAG 1	20%	80%	84.56%	84.86%	84.71%
LAG 1	50%	50%	85.05%	85.71%	85.38%
LAG 3	80%	20%	86.27%	84.84%	85.55%
LAG 3	20%	80%	83.22%	84.14%	83.68%
LAG 3	50%	50%	85.69%	84.26%	84.97%

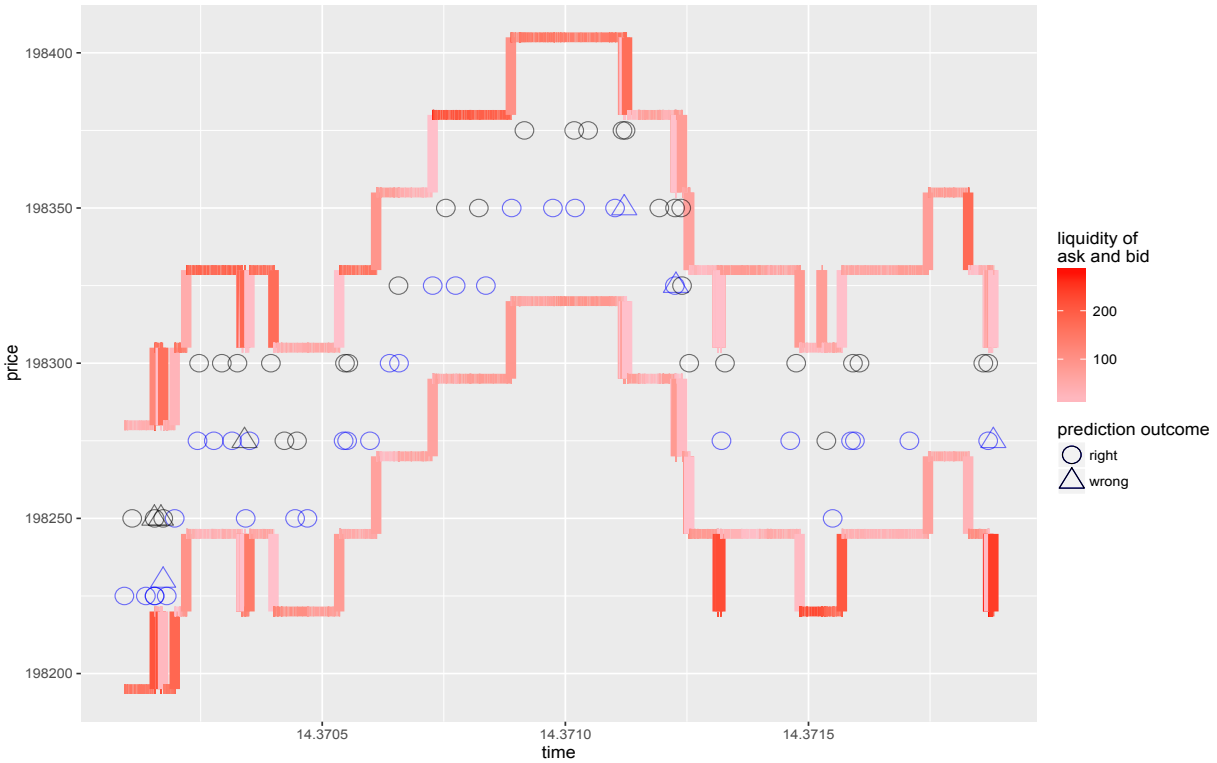


Figure 4.2

The results of the prediction using SVM model are as follow:

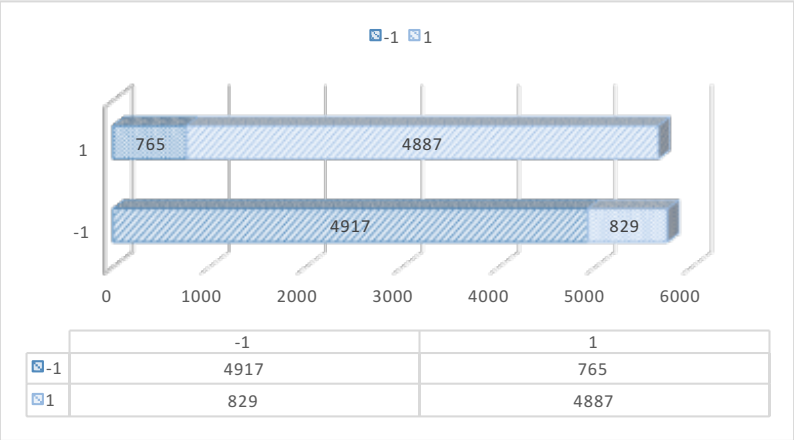
1. Using inputs with 1 previous book update yields a better prediction accuracy than using inputs with two or more previous book updates, given the same training to testing set ratio.

2. The prediction accuracy is the highest with a training to testing set ratio of 80%-20%, and the training to testing set ratio of 20%-80% has the worst prediction accuracy.

3. Throughout the process of this project, more and more financial indicators are included into the model, and the performance of the model became better and better.

4. Given the same data set, the difference between the correction rate of predicting up and down trend is smaller when using 50%-50% training to testing set ratio than using 80%-20% training to testing set ratio.

Below is the graphic representation of the best prediction model of SVM:



	-1	1
-1	4917	765
1	829	4887

5. Iceberg order detection

5.1 Iceberg Order Overview

As is described above, the model we are trying to build is mainly based on certain patterns of the liquidity state in the market we derived from the open *limit order book* (LOB) of CME GLOBEX Futures. But at CME, LOB may not display the "true" liquidity state of the market. Iceberg orders are also an order type supported by GLOBEX.

An iceberg order is a special type of limit order, where only a fraction of the total order size is shown in the LOB at any one time with the remainder of volume hidden. The use of iceberg orders

can benefit the owner by removing the informational disadvantage associated with having their intentions publicly known and avoiding being exploited by subsequent participants.

As a consequence of the existence of iceberg orders, the actual liquidity is quite different from the liquidity we observed from the LOB. For example, the table below shows that a trade happened on price 1986.25 with volume 2 while LOB showed that there were only 1 lot available on book at price 1986.25 at that time. This is due to the fact that there were iceberg orders placed on the level one of ask side.

Time	Order Type	Leve1 Bid Size	Leve1 Bid Price	Level 1 Ask Price	Level1 Ask size
00:14:05.407	Submit/Cancel	1	198625	198650	27

Time	Order Type	Volume	Price
00:14:05.407	**Trade**	2	198625

Time	Order Type	Leve1 Bid Size	Leve1 Bid Price	Level 1 Ask Price	Level1 Ask size
00:14:05.407	Submit/Cancel	3	198625	198650	27

Table 5.1

Therefore, in order to make a better prediction, we need to find out the real liquidity in the market. In other words, we try to detect the iceberg orders hidden behind the LOB and add the hidden lots back to the original book.

5.2 Detection Algorithm

In this section, the logic behind how the algorithm works is described. Basically, the algorithm we wrote for detecting iceberg orders on GLOBEX applies a pattern recognition approach which seeks out a certain combination of events that allows and iceberg order to be identified and tracked.

First sets of events that enable us to identify iceberg order are events like the trade record updates show in table 6.1 where trade volume is larger than the corresponding book quantity. The difference of the trade volume and the displayed book quantity should be accounted as the iceberg order size. However, as we can see in table 5.1, after the trade happened, the following book record update showed there were still two lots in the level of 1986.25. This means that the orders placed on 1986.25 had not been fully executed yet, and it is not likely to decided whether there's still iceberg orders in the remaining orders. So in this scenario, we can only identify the existence of an iceberg order and part of its size. We cannot confirm the real size of the iceberg order.

Second sets of events are similar to the first ones but a lot more ideal. As is showed in table 5.2, Like the first sets of events, a trade happened with a volume larger than the displayed book quantity. Contrary to the first scenario, the book record updates after the trade showed that the best ask price changed from 1986.75 to 1987.00, which might lead to the conclusion that the order placed on 1986.75 had been fully executed.

Time	Order Type	Level Bid Size	Level Bid Price	Level 1 Ask Price	Level1 Ask size
00:14:38.289	Submit/Cancel	25	198650	198675	1

Time	Order Type	Volume	Price
00:14:38.289	**Trade**	2	198675

Time	Order Type	Level Bid Size	Level Bid Price	Level 1 Ask Price	Level1 Ask size
00:14:38.289	Submit/Cancel	26	198650	198700	1

Table 5.2

However, the mechanics on CME GLOBEX allows a trader to place an iceberg order with a max show (the peak of the iceberg). So when the above scenario happens, it could simply mean the 'peak' of the iceberg order has been executed. We need to check the next trade to make sure all the order on 1986.75 has been executed: If the ask level has not moved from 00:14:38.289 to next trade and next trade took place at price level other than 1986.75. We can conclude that the size we inferred for the iceberg order is its total amount. Otherwise, it may still be part of the iceberg like first scenario.

Third sets of events happened when bid-ask spread is relative large. There might be iceberg orders placed in the spread. As is shown in table 5.3, the trade took place in the spread at price 1985.50. This indicates the existence of iceberg placed at that price with unknown quantity. Again, this is the type of iceberg order we can detect but cannot confirm its total hidden size.

Time	Order Type	Level Bid Size	Level Bid Price	Level 1 Ask Price	Level1 Ask size
00:08:48.774	Submit/Cancel	30	198525	198575	25

Time	Order Type	Volume	Price
00:08:48.774	**Trade**	1	198550

Time	Order Type	Level Bid Size	Level Bid Price	Level 1 Ask Price	Level1 Ask size
00:08:48.774	Submit/Cancel	30	198525	198575	25

Table 5.3

5.3 The result of the algorithm

Based on the algorithm above, we are able to identify 11091 iceberg orders among all 235967 trades in one day. And the total size we found that was hidden is 446167 lots.

Type	*Number*	*Hidden Size*
Iceberg in the spread	1191	4210
Iceberg we cannot decide size	3519	118120
Iceberg we can decide size	6381	323837
Total iceberg orders found	11091	446167

Table 5.4

And the following graph shows the time series of the trade price and iceberg order sizes. The orange line represents the price movement of whole day and the grey bar is corresponding size of the iceberg we detected and the *black bar shows the real quantity in the LOB*. (the displayed book quantity plus the hidden quantity.)

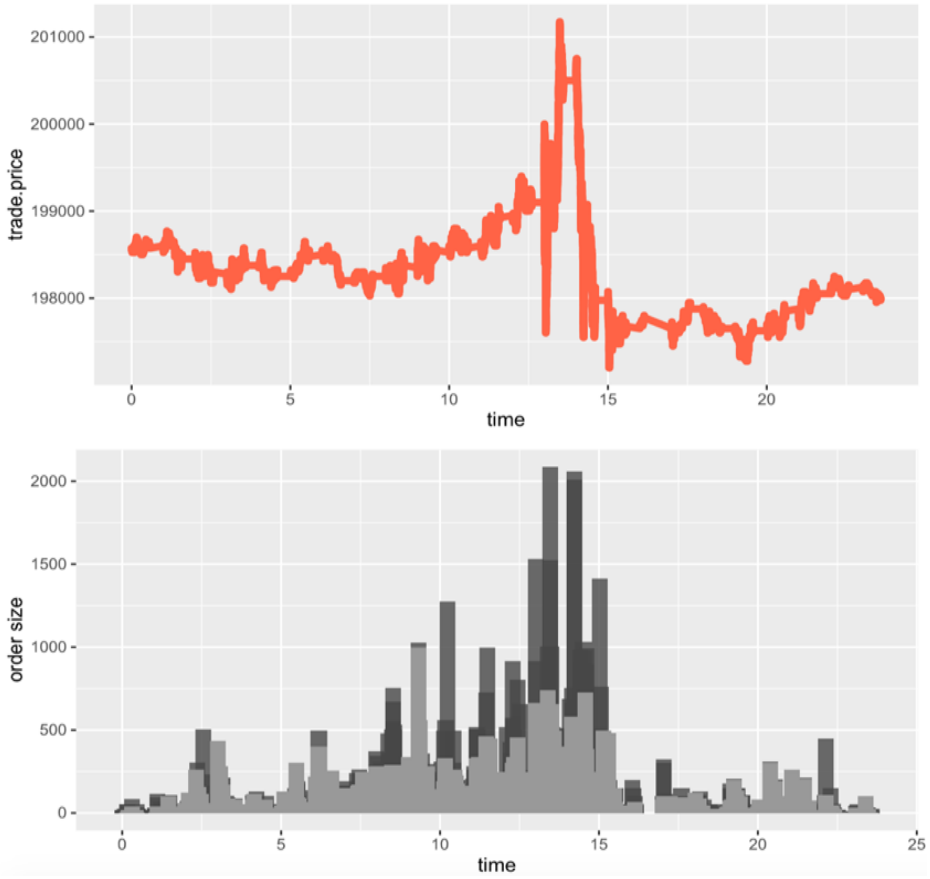


Figure 5.1

Next, we add the iceberg order size we detected back into the book quantity and rebuild our machine learning model. And do the prediction again. The prediction rate we acquire from SVM model is slightly lower than the origin one (about 0.3%). The reason behind this is because of certain mechanics on GLOBEX we described above that in most the time, we cannot find out whether there is an iceberg order. And in some circumstances, even we can detect the existence of an iceberg orders, we cannot know for sure how large is the actual size and when did the owner placed the iceberg order. Therefore, the prediction accuracy is not as good as we'd expected. There does exist several algorithms that provide ways to infer the actual size and time of the iceberg orders. De Winne and D'hondt [2007] and Bessem-binder et al, [2009] use regression models on 2002-2003 Euronext equities data. Huang and Hautsh [2007] build a Bayesian model with a Bernouli likelihood function using logit multiple regression. But due to the difficulty and the fact that all of these methods are based on probability inference, so we did not try to implement these algorithms in this project.

6. Conclusion

1. The indicators we select do have certain prediction power.
2. Logistic Regression gave us a prediction accuracy around 75% while Support Vector Machines gave us a much better accuracy rate around 86%.
3. The separation of training and testing set has a great impart on the performance of learning models.
4. There is a large amount of iceberg orders hidden in the limit order book which means the liquidity we observe from the LOB is not accurate.

Reference

[1] Hugh L. C., and Robert Woodmansey. (2013) 'Prediction of Hidden Liquidity in the Limit Order Book of GLOBEX Futures'. 'The Journal of Trading', 8(3), 68-95.

[2] Alec N.Kercheval and Yuan Zhang. (2013) 'Modeling high-frequency limit order book dynamics with support vector machines'.

[3] ÁLVARO CARTEA & SEBASTIAN JAIMUNGAL (2013), 'Modelling Asset Prices for Algorithmic and High-Frequency Trading'. 'Applied Mathematical Finance', Vol. 20

[4] MARCO AVELLANEDA and SASHA STOIKOV(2006), 'High-frequency trading in a limit order book '. 'Quantitative Finance', Vol. 8

Appendix 1: Datasets

$$\text{average bid price} = \frac{\sum_{i=1}^{10} Price_{Level\ i}^{bid} \times Quantity_{Level\ i}^{bid}}{\sum_{i=1}^{10} Quantity_{Level\ i}^{bid}}$$

$$\text{average ask price} = \frac{\sum_{i=1}^{10} Price_{Level\ i}^{ask} \times Quantity_{Level\ i}^{ask}}{\sum_{i=1}^{10} Quantity_{Level\ i}^{ask}}$$

$$\text{book volume imbalance} = \sum_{i=1}^{10} Quantity_{Level\ i}^{bid} - \sum_{i=1}^{10} Quantity_{Level\ i}^{ask}$$

$$\text{weighted book price} = \frac{\sum_{i=1}^{10} (P_i^{ask} \times Quantity_i^{ask} + P_i^{bid} \times Quantity_i^{bid})}{\sum_{i=1}^{10} (Quantity_{Level\ i}^{ask} + Quantity_{Level\ i}^{bid})}$$

Dataset one:

bestasksize.lag3	bestbidsize.lag3	imba.lag3	vol.lag3
bestasksize.lag2	bestbidsize.lag2	imba.lag2	vol.lag2
bestasksize.lag1	bestbidsize.lag1	imba.lag1	vol.lag1
book1.best.ask.size.change	book1.best.bid.size.change	book1.bid.price.change	book1.ask.price.change
book1.ask.size.change	book1.bid.size.change	book1.imba.change	book1.weighted.price.change
book1.best.ask.size	book1.best.bid.size	book1.ask.size	book1.bid.size

Dataset two:

bestasksize.lag3	bestbidsize.lag3	imba.lag3	vol.lag3
bestasksize.lag2	bestbidsize.lag2	imba.lag2	vol.lag2
bestasksize.lag1	bestbidsize.lag1	imba.lag1	vol.lag1
book1.best.ask.size.change	book1.best.bid.size.change	book1.bid.price.change	book1.ask.price.change
book1.ask.size.change	book1.bid.size.change	book1.imba.change	book1.weighted.price.change
book1.best.ask.size	book1.best.bid.size	book1.ask.size	book1.bid.size
book2.best.ask.size.change	book2.best.bid.size.change	book2.bid.price.change	book2.ask.price.change
book2.ask.size.change	book2.bid.size.change	book2.imba.change	book2.weighted.price.change
book2.best.ask.size	book2.best.bid.size	book2.ask.size	book2.bid.size
book3.best.ask.size.change	book3.best.bid.size.change	book3.bid.price.change	book3.ask.price.change
book3.ask.size.change	book3.bid.size.change	book3.imba.change	book3.weighted.price.change
book3.best.ask.size	book3.best.bid.size	book3.ask.size	book3.bid.size