Machine Learning on Limit Order Book

Alex Monahan Ramin Ahmari Jiayu Wu

Abstract

In this paper, we propose a framework that utilizes a data processing and machine-learning framework to analyze data from the HSBC limit order book (LOB) on March 20th, 2013. By characterizing each row in the limit order book as a multidimensional element, the model proposed in this paper offers a prediction of whether the HSBC stock price will increase or decrease for each entry of the LOB. The predictions of the model were subsequently compared to the realized price-movements of the stock. The model bases fundamental predictions on features created from the ten best ask and bid prices in the LOB. We address the imbalanced data problem and the methods we employed to overcome it. The results from the random forest model showed a great performance with accuracy above 95%, and we also found volume is the major driver of short-term mid-price movement.

1 Introduction

The world is shifting towards high frequency trading (HFT). As trading floors at large investment banks, such as Goldman Sachs, continue to downsize, high frequency trading firms, such as Citadel, continue to rise. As noted by Kercheval & Zhang, in 2012, approximately 84% of stock trades were completed through high frequency trading.

Fundamentally, LOBs match buyers and sellers in an order-driven market, as electronic trading systems aggregate the available orders into a single LOB. Currently, the NYSE, London Stock Exchange, and NASDAQ operate with hybrid LOB systems (Gould et al. 2013). Orders are entered into the LOB in a typical "first-in first-out" queue format. Some exchanges allow hidden orders, which, of course, decrease the transparency of the given market (e.g. a dark pool). Most systems, however, do not allow hidden buy or sell orders.

As further details regarding the LOB, limit orders can be canceled at any time if the order has not been executed, and the LOB is simply reconstructed without the previous order. In an LOB system, there are two main types of "players:" liquidity providers and liquidity takers. Liquidity providers (e.g. HFT firms) offer to either buy or sell at certain bid and ask prices, and liquidity takers are available to buy or sell at these given prices. As described in Homework 1 of this course, due to adverse selection and information asymmetry, there is always a positive bid-ask spread.

Due to advances in technology, data-packed LOBs are available in real-time to traders throughout market hours. Consequently, the LOB of a given equity can be used as an input to high-frequency trading models, and past LOB data can be utilized to test HFT trading strategies as firms try to generate alpha for their investors.

2 Task

The task of this project is to do multi-classification using limit order book data. The goal is to predict the direction of mid-price movement. For example, we take a row of features of the limit order book at a specific moment, and predict whether the mid-price will go *upward*, *downward*, or is *stationary* at Δt later.

We define mid-price at time t as

$$\text{mid-price}_t = (\text{best-bid-price}_t + \text{best-ask-price}_t)/2 \tag{1}$$

If $\text{mid-price}_{t+\Delta t} > \text{mid-price}_t$, then the label of row_t is *upward*.

If mid-price$_{t+\Delta t}$ = mid-price$_t$, then the label of row$_t$ is *stationary.*
If mid-price$_{t+\Delta t}$ < mid-price$_t$, then the label of row$_t$ is *downward.*

3 Data

We use the data `2014-03-20-HSBC_L2_Proj.csv`, which has 26708 messages sent by the exchange. The fields include 10 best ask price/volume, 10 best bid price/volume, and a timestamp.

An important part of this project is to preprocess the raw L2 data to reconstruct the limit order book for each moment during the trading hours.

In the dataset we are given, we have a rows like

SEQ	TIMESTAMP	...	BP1	BSize1	AP1	ASize1	...
...	...	...	...	...	...	...	...
7	2014-03-20 09:30:00.112	...	76.2	9600	*	*	...
...	...	...	...	...	...	...	...

Notice that each message may only contain some bid price and its corresponding size and not necessarily contain all information about the limit order book at that moment specified by "TIMESTAMP". For example, at SEQ 7, AP1 and ASize1 are just "*".

Thus, in order to recover the limit order book, we use the first limit order book snapshot sent at 2014-03-20 09:20:00.163, and then iterate through every row while updating the book from the most recent snapshot.

We notice that some rows have "NA" for some bid/ask price/volume. In the case of "NA", we keep the value of the last limit order book snapshot.

After we have recovered a dataset of snapshots of the limit order book, in order to do multi-classification, we want to label each row using one of $\{0(\textit{downward}),\ 1(\textit{stationary}),\ 2(\textit{upward})\}$. Using rules defined in Section 2, we want to find the first row whose timestamp is at least Δt ahead of the current row, and then compare its mid-price with the current row's mid-price. We use $\Delta t = 5$ seconds.

Now we have a label for each row. We then extract features from the raw features of each row, and then use extracted features to do the training, and finally compare the prediction results with the true labels.

One thing worth noting is that the price of HSBC on 2014-03-20 does not vary much. For example, the open mid-price for the day is 76.225 while the close mid-price is 76.025. The resulting problem is that around 80% of the labels are 1(*stationary*). Thus, we have an imbalanced dataset problem. For example, we can just predict the label to be 1(*stationary*), then we will have an accuracy of 80%. To avoid this problem, we downsample rows with a label of 1(*stationary*), to make the number of each class more balanced. After down sizing *stationary* rows, the ratio of *stationary*:(*upward*+*downward*) is around 1:1. Then we see that an absolutely biased predictor that only predicts *stationary* will obtain an accuracy of 50%, which is the baseline for our models.

Finally, we split the entire dataset into train data (70%), validation data (10%), and test data (20%).

4 Feature Extraction

After the data had been pre-processed, features had to be determined to analyze the HSBC LOB data. As noted in Gould et al., analysis of LOB data has taken a variety of forms, drawing on concepts from physics, mathematics, economics, finance, psychology, and statistics, and there still does not remain an objectively most effective method of analysis. For instance, the Smith-Farmer-Guillemot-Krishnamurthy model uses a Poisson process with rate $\mu/2$ to model market orders and the cancellations of existing limit orders at the "die" rate v.

In our paper, we constructed different features from the ten best ask & bid prices and volumes for the data.

There are some basic features directly available in limit order book data. We include the following features for a row:

$$\begin{aligned}
&\text{AP}_i,\ i\text{th best ask price}\\
&\text{ASize}_i,\ \text{volume of } i\text{th best ask price}\\
&\text{BP}_i,\ i\text{th best bid price}\\
&\text{BSize}_i,\ \text{volume of } i\text{th best bid price}
\end{aligned} \tag{2}$$

We also used the square of the range of the ask price. The initial hypothesis for this feature, as an example, is that a wider range in ask prices is correlated with higher demand for the given asset, and, thus, a greater value for the square of range of the ask price would signal a greater probability of the stock price increasing at the next instant. Similarly, this can be applied to the bid price and thus, we also used the square of the range of the bid price for our model.

$$\begin{aligned}
\text{Square of range of ask price} &= (\text{AP10} - \text{AP1})^2\\
\text{Square of range of bid price} &= (\text{BP10} - \text{BP1})^2
\end{aligned} \tag{3}$$

The bid/ask and volume imbalances are also good indicators of price movement. It is believed that the more imbalanced they are, the more likely the price is going to change in the near future.

$$\begin{aligned}
\text{Bid-ask imbalance} &= (\text{BP1} - \text{AP1})/(\text{BP1} + \text{AP1})\\
\text{Volume imbalance} &= \text{BSize1}/(\text{BSize1} + \text{ASize1})
\end{aligned} \tag{4}$$

Furthermore, we also consider the bid-ask spread.

$$\text{Bid-Ask spread} = \text{AP1} - \text{BP1} \tag{5}$$

5 Models

5.1 Logistic Regression

Logistic regression is a statistical regression model dependent on a categorical variable and developed in 1958 by statistician David Cox.

In our case, this categorical variable would be $\{0(\textit{downward}), 1(\textit{stationary}), 2(\textit{upward})\}$ as outlined in Section 3: Data. As we have more than two categories, we apply logistic regression in a one-vs-rest fashion, which means we do logistic regression classification for each class by viewing other classes as one other class.

The logistic model estimates the probability of one of the categorical outcomes based on a number of specified predictors, which are the features we have alluded to earlier in the paper, by establishing a relationship between the respective categories and the predicting features based on an estimation of probabilities via the cumulative logistic distribution. This is calculated with the logistic function

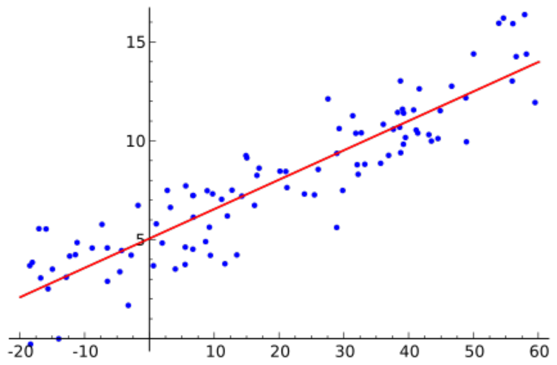

Figure 1: Logistic Regression
(Source: RaphaelQS, Wikipedia Commons, March 2016)

$$f(x) = \frac{L}{1 + e^{-t}} \tag{6}$$

for which
L = the curve's maximum value
t = a linear function of a single explanatory variable x

Estimations are generally be derived through the ordinary least squares calculation as the logistic regression model can be expressed as a generalized linear model.

Regression coefficients are generally estimated through maximum likelihood estimation and since there is no closed-form expression for this, iterative processes must be used (e.g. stochastic gradient descent or Newton's method).

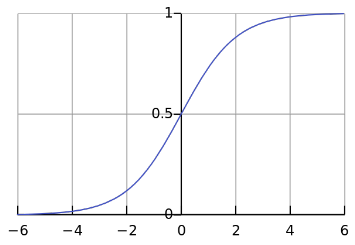

Figure 2: Standard Logistic Sigmoid Function
(Source: Aflafla1, Wikipedia Commons, July 2014)

5.2 Support Vector Machine

Support Vector Machines (SVMs) are machine learning models that estimate a categorical output based on certain features that we identified as a non-probabilistic binary classifier.

Graphically, it does so by dividing a set of points with a line in a way that the gap between the points on either side, and the line, is as large as possible (indicating a clear and evident separation and classification). When handling new data, this line will be evaluated to establish the categorization of the new example data, determining on which side of the gap the new example point fall to formulate its classification.

By using a kernel trick, which maps inputs into high-dimensional feature spaces, Support Vector Machines can perform non-linear classifications.

The classifier of a Support Vector Machine for a p-dimensional vector is defined by the (p-1)-dimensional hyperplane that separates the examples as described above. The optimal hyperplane is defined to be associated with the biggest margin to its closest example point. This hyperplane is denoted as the maximum-margin hyperplane which defines the maximum-margin classifier. It is this maximum-margin hyperplane that a Support Vector Machine establishes.

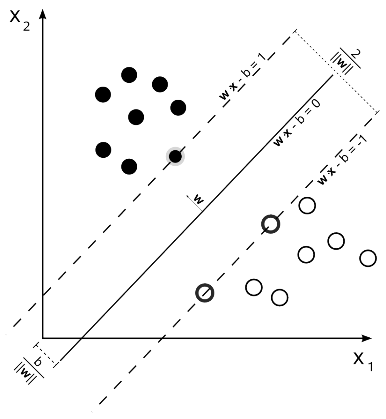

Figure 3: Maximum Margins / Gaps Created Through Hyperplane
(Source: Peter Buch, Wikipedia Commons, June 2011)

5.3 Random Forest

Random decision forests are an ensemble learning method, first devised by Tin Kam Ho and extended by Leo Breiman and Adele Cutler, that can be used for both regression and classification. Ensemble learning algorithms are models that, in a divide-and-conquer approach, use multiple learning algorithms to boost their prediction accuracy and modeling behavior.

Random decision forests in particular have gained their name by establishing numerous decision trees (a popular choice for data mining purposes as they are invariant under scaling and certain other transformations of the extracted features that are inputted into them) when being trained on the training data. However, each tree is a in itself only a weak learner due to its low bias but high variance, yet together they can lead to a strong learner. This process is represented graphically on Figure 3 below. Thus, for classification, the mode of the different decision tree is outputted and for regression, the mode of the mean prediction of the different decision trees is outputted.

Achieving the strong learner from the weak learners is being done through Bootstrap Aggregating which in this case works as follows:
For some number of trees T:

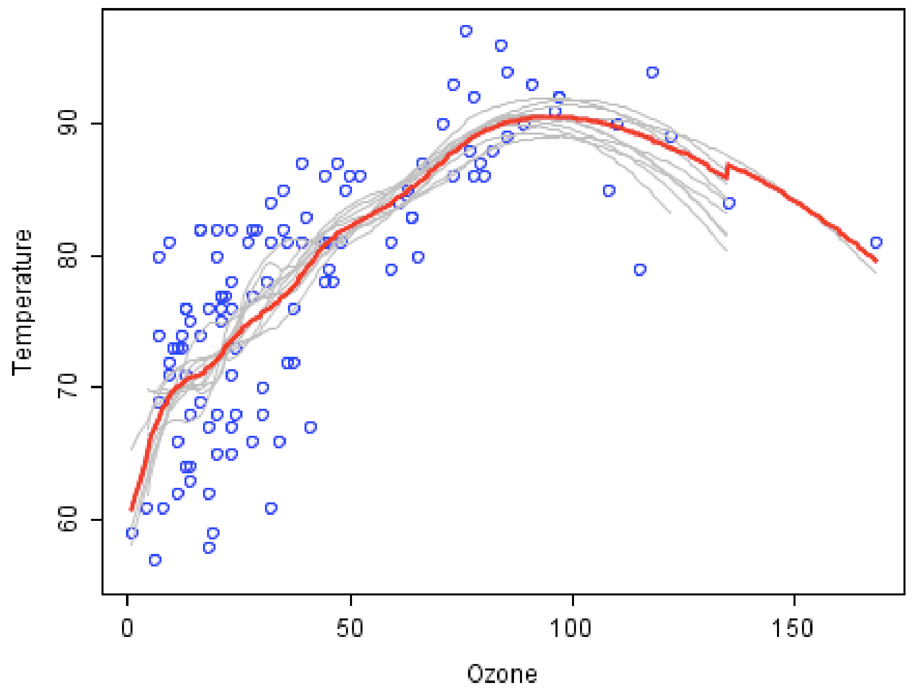

Figure 4: Creating a Strong Learner (Red) By Aggregating Weak Learners (Gray) Over the Data (Blue)
(Source: Future Perfect at Sunrise, Wikipedia Commons, May 2012)

1. Sample N cases at random (with replacement). This creates a subset of the data that is usually aimed to be about two thirds of the entire data set.
2. Repeat for each node: m predictor variables are randomly selected from all predictor variables with the value of m being much smaller than the total amount of predictor variables. The predictor variable that achieves the best binary split (binary splitting is a technique for increasing the efficiency of numerical evaluations) on that node splits it.

As decision trees trend to overfit their training sets, random decision trees actively correct themselves for this phenomenon by averaging the decision trees, trained on different parts of the same training data. This is to increase the variance of the algorithm as it already has low bias due to the nature of the decision trees.

Random decision forests can rank the importance of variables within its respective regressions or classifications.

6 Evaluation Metrics

We use accuracy, precision, recall and the F1 score as our evaluation metrics. Since precision, recall and the F1 score are based on binary labels, in our multi-classification case, we calculate these scores for each class, namely viewing other classes together as one other class, and then calculate the average of them.

Accuracy	Precision	Recall	F1 score
(TP+TN)/(TP+FP+TN+FP)	TP/(TP+FP)	TP/(TP+FN)	2(precision · recall)/(precision+recall)

7 Results

We used the `scikit-learn` Python package to do the training and testing. We first tested a logistic regression model with a one-vs-rest scheme. We further tested a support vector machine model with an rbf kernel and a one-vs-rest scheme, and finally also tested a random forest model with different numbers of trees.

Here we show a table of the results we accomplished.

	accuracy	average precision	average recall	average F1-score
Logistic Regression	0.6262	0.6276	0.5950	0.6059
SVM, rbf kernel	0.5381	0.8322	0.4326	0.3940
Random Forest, $N_{trees} = 1$	0.9315	0.9255	0.9337	0.9293
Random Forest, $N_{trees} = 10$	**0.9647**	**0.9589**	**0.9685**	**0.9633**
Random Forest, $N_{trees} = 50$	0.9569	0.9523	0.9583	0.9552

First, it is evident that the random forest model performed very well, with all five metrics above 0.95. One reason might be that random forest is an ensemble learning method as it utilizes a number of decision trees and aggregates their output by keeping the improved performance on the part of data it has not performed well on. Another reason might be that every time a node is split, the model only considers a random subset of features, which decreases the model's variance significantly.

Furthermore, we observe that logistic regression performs only slightly better than the baseline of 50%. A possible explanation is that the underlying probability distribution of the features is not matching to the logistic distribution assumed by the logistic regression model.

Surprisingly, the support vector machine model also performs poorly with about 0.63 accuracy. Although Kercheval & Zhang showed that support vector machines performs fairly well on AAPL L2 data, it fails to demonstrate a convincing performance on our HSBC L2 data. One possible explanation is that the minimum tick size in their data is much smaller than the one in our data, and thus they would have been able to label each row much more accurately than we could. Because of the large minimum tick size and the small price volatility, many rows in our data are labeled as *stationary*. However, there might have been price movements within the minimum tick size's range. Thus, this "fuzzy" labeling actually results in the loss of information because the effect of changes of some features is not reflected in the label itself.

It is also interesting to see the importances of the respective features that the random forest model returns. Here we show a plot of the five most important features returned by the random forest model.

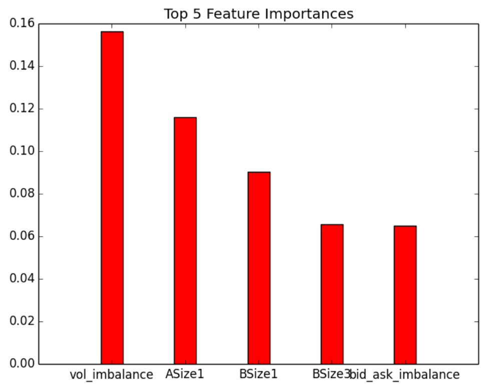

Figure 5: Five Most Important Features Returned by the Random Forest Model

According to our figure, volume-related features seem to dominate when it comes to predicting the mid-price movement. The most important feature is volume imbalance (BSize1/(BSize1+ASize1)). Indeed, fundamental economic theories suggest that imbalanced supplies and demands lead to a change of price.

The importances of the respective features that we established show that volume is a major driver of short-term mid-price movement.

8 Conclusion

In this project, we applied machine learning methods to analyze HSBC L2 data. We formulated the problem as a multi-classification problem, predicting the mid-price movement as *downward*, *stationary*, or *upward*. The random forest model showed a surprisingly good performance with an accuracy higher than 0.95, while other models, such as logistic regression and support vector machine, only had a mediocre performance. The importance of the respective features returned by the random forest model showed that in the short-term, which is 5 seconds in our case, volume is the major driver of mid-price movement.

Although HSBC data was used in this proposed model, vast amounts of historical LOB data exist, and the proposed model could be tested with LOBs for a variety of other equities. Using these other equities, the model could be further refined as data size is a key factor in establishing an accurate machine learning model.

Furthermore, additional features could be generated. Feature extraction is the most difficult, and arguably the most important, aspect when applying a machine learning algorithm to generate effective, profitable HFT strategies. Thus, with more complex, mathematically intensive features generated for each LOB entry and passed into the machine-learning algorithm, the accuracy of the model should be improved even further. Proprietary data would be very useful in this case. As seen currently across the hedge fund industry, competitors are searching everywhere for unique sources of information to include within their features. As noted in one article from the *Wall Street Journal*, numerous hedge funds are attempting to use satellites and natural language processing of news to access useful proprietary data that can be inputted into their HFT models. These concepts and ideas would of course, applied to our models, further improve our accuracy.

9 References

Gould, Martin, Mason Porter, Stacy Williams, Mark McDonald, Daniel Fenn, and Sam Howison. "Limit Order Books." *Quantitative Finance* 13.11 (2013): 1709-742. Web.

Hope, Bradley. "Tiny Satellites: The Latest Innovation Hedge Funds Are Using to Get a Leg Up." *The Wall Street Journal*. Dow Jones & Company, 14 Aug. 2016. Web. 09 Dec. 2016.

Kercheval, Alec, and Yuan Zhang. "Modeling High-frequency Limit Order Book Dynamics with Support Vector Machines." Florida State University, 24 Oct. 2013. Web.

Smith, Eric, Doyne Farmer, Laszlo Gillemot, and Supriya Krishnamurthy. "Statistical Theory of the Continuous Double Auction." *Quantitative Finance* 3 (2003): n. pag. Web.