MS&E448 Final Paper Report
High Frequency Algorithmic Trading

Francis Choi George Preudhomme Nopphon Siranart
Roger Song Daniel Wright

June 11, 2017

Abstract

In this paper, we assess the informational value of the limit order book in predicting near-term price movements within a machine learning framework. We analyze a corpus of discretely sampled time series data on exchange-traded securities to derive features based on order book characteristics such as price, volume, time, and order flow. Our model demonstrates the potency of random forest and logistic regression in predicting a directional price movement classifier that, with further research, may potentially translate to a production-grade trading strategy.

1 Introduction

The proliferation of high-frequency algorithmic trading in the United States originates with the electronification of equities markets in the late 1990s, pioneered by the now-defunct Island ECN which began matching orders in 1997. By 2001, the SEC had compelled exchanges to decimalize and discontinue the lucrative practice of maintaining the traditional minimum price variation (MPV) of 1/8th of USD. Quote-driven markets are characterized by the presence of a centralized auctioneer (such as a NYSE specialist) publishing a single pair of bid and ask quotes for a security at which buyer and sellers must transact at with the auctioneer for a given indefinite quantity. The profitability of the central auctioneer was maintained by widening or narrowing the quote spread to a level commensurate with the order flow, with a guaranteed minimal floor of 1/8th USD between the bid and ask. Following decimalization, this highly lucrative model vanished overnight with quotes compressing to a much narrower spread (Furfine, 2003). Bar a few exceptions in the FX and OTC space, quote-driven markets have been replaced by a central limit-order-book (LOB) model in which any participant is free to post a buy or sell order at a better price than the prevailing bid/ask if they believe they have superior information than the market.

To attract liquidity in a post-decimalization electronic marketplace, many exchanges began offering maker rebates to attract trading volume. The passage of Regulation National Markets System (Reg NMS) by the SEC in 2005 cemented the growth of alternative trading

systems (dark pools) that did not have adhere to the protected quote rules that created inefficiencies, such as opportunities for latency arbitrage against Security Information Processors (SIPs) in calculating and disseminating the National Best Bid and Offer (NBBO). Advances in computational technology and made possible the speed and cost compression made high-frequency trading (HFT) viable. By 2015, HFTs accounted for over 55% of trading volume in US equities (Gerig, 2015), and present as at least one counterparty in +70% of all trades based on Nasdaq trade and quote (TAQ) data sampling (Khashanah, 2014).

The informational value of high-frequency TAQ data (OHara, 1997) and limit order books (Avellaneda, 2008) have been already well documented in academia and by industry practitioners. Since the late 2000s, however, HFT volume and profits have stagnated, if not shrunk as large banks scaled back their propriety trading arms the 2008 financial crisis and Dodd-Frank capital regulations. A TABB Group/Deutsche Bank Research white paper estimates HFT firms revenues in the US to have shrunk from USD 7.2bn in 2009 to USD 1.3bn in 2014. (Kaya, 2016). With more money and homogenous strategies chasing after a finite amount of alpha, the signal/noise ratio falls, and the lions share of profits tend to go to large market makers who can the afford the large capital expenditures to maintain a latency advantage (Laughlin, 2014).

In the same period, driven by same breakthroughs in computational power, the domain of machine learning has witnessed notable innovations in nonlinear classification algorithms in support vector machines (Boser, 1992) and random forests (Breiman, 2001). Markets generate much granular and noisier signals than they did in the past. We hypothesize that this sort of data may lend well to machine learning frameworks to identify patterns in data. Our paper builds upon existing work on order book dynamics modeling with support vector machines (Kercheval, 2013) and market microstructure feature selection (Kearns, 2013). Our approach aims to apply a guided learning framework utilizing random forest and logistic regression algorithms to predict near-future price movement from an expanded set of features derived from a discretely sampled limit order book for a given security.

2 Literature Review

Existing academic research on modeling limit order book (LOB) dynamics cluster into one of two approaches to understanding data: statistical modeling vs. machine learning. In the first approach, statistical properties of the LOB for a given asset are developed and conditional quantities are then derived and modeled (i.e. akin to modeling residuals via the Ornstein-Uhlenbeck process this quarter). In contrast, machine learning methods represent data from the LOB in a systematic manner, then generalize the data so that unseen data can be recognized and classified based on the generalization (Kerchval, 2013).

In the early days of algorithmic trading, exploiting the full potential of the latter approach was hamstrung by the limitations computing power and accessibility of granular, timely data (OHara, 1997). Given these constraints, the statistical predictive models commonly resorted to GARCH or FIGARCH derivative processes. While GARCH/FIGARCH methods were powerful tools for modeling longer-termed behavior of securities, they were observed

to be universally unstable and unsatisfactory for intraday-or-higher-frequency predictions (Andersen, 1994; Guillaume, 1995).

For our aims, other statistical predictive methods (utility maximization, fundamentals-driven, minimizing market impact, lattice-particle diffusion, agent based, etc.) were deemed poor candidates for modeling high frequency LOB behavior given their unrealistic or unverifiable assumptions (Avellaneda, 2008; Gould, 2013).

Instead, we sought to pivot our research into exploring the use of machine learning (ML) tools to model statistical regularities in the LOB. Recent literature has shown a wide range of successes in using ML to answer challenges in market microstructure and high frequency trading in fields such as: reinforcement learning for optimized trade execution, price movement prediction from order book states, and optimal trade routing (Kearns, 2013). Notably, we were inspired from the potency of support vector machines (SVM) to accurately extend predictions for high frequency LOB dynamics (Kercheval, 2013). We believed that we could build on Kercheval and Zhangs 2013 work by expanding the ML toolkit to additional algorithms, expanding the corpus of order book features, optimizing the hyperparameters, and tweaking the classifier label.

3 Data Processing

The following sections illustrate how we extracted features and labels from the data, along with challenges faced in working with the Thesys platform with respect to simulation runtimes.

3.1 Limit Order Book Dynamics

A limit order book is a set of all the active orders in a market for a given security s at a point in time t as $L_s(t)$. A LOB has a is partitioned as a bids book (B_s^{bid}) and asks (A_s^{ask}), with each level of the respective books l_n^{bid} and l_n^{ask} ordered by price (p) and depth (w). Changes in the LOB can be characterized as a discrete cdlg process (Gould, 2013) where an arriving limit order $x_s = (p_x, w_x, t_x)$ adds to or removes liquidity from to the corresponding level of the limit order book. The top (inner) level of the book is defined as the best bid or ask price available at a given time.

We utilize consolidated limit order book data from Thesys Technologies education access. Thesys compiles historical direct feed trade and quote (TAQ) data at microsecond intervals from all 13 exchanges: AMEX, ARCA, BATS, BATSY, BX, CBOE, CHX, EDGA, EDGX, IEX, INET, NSX, and PSX, although we did not have access to NYSE data due to licensing agreement issues. To access this data, we were provided authentication keys to a Jupyter ipython notebook hosted on Thesys servers.

In the aim of limiting the scope of our project for computational tractability purposes, we extracted second-by-second consolidated order book data from Thesys for the top five levels of the bid and ask. For sampled intraday time, we subtracted 15 minutes from the beginning and end of the trading day (9:45AM 3:45PM) to account for opening and closing

volatility. Thus, for each security, we extract the state of the order book every second for 60 seconds per minute, 60 minutes per hour, 6 hours per trading day. Every state of the order book is populated with 10 levels with each containing price, volume, active number of orders, and time, totalling 864,000 data points per day to train and test our LOB model.

While we initially sought to run our testing and trading simulations via the Python API, we observed that the inbuilt simulator could not adequately handle trades at the speed desired for the quantity of data tested. Thus, we pickled our data locally to accelerate our simulation runtime. A downside of this methodology is that we were unable to realistically simulate the challenges posed by partial fills and slippage as only the Thesys simulator injects orders into a historical real-time TAQ stream. Nevertheless, we believed that the loss of realism was not severely hampered as our strategy was simulated with a single buy/sell quantity designed to capture only the top of book at any given time.

To build our training algorithm, we opted to use off-the-shelf open source machine learning tools via scikit libraries (sklearn.ensemble ExtraTreesClassifier) along with additional Python libraries (pandas, numpy, matplotlib.pyplot, etc.) for data manipulation.

3.2 Features

To capture information in the limit order book into a format digestible by our machine learning model, we constructed a feature vector $\bar{v}(t)$ derived from the current state of the LOB $L(t)$ (along with lagged features) for every sampled period t. We primarily drew upon the body of features documented in Kercheval (2013) as well as Kearns (2013) for the following:

Basic Set	**Description** ($i = level\ index$, $n = 5$)
$v_1 = \{P_i^{ask}, P_i^{bid}, V_i^{ask}, V_i^{bid}\}_{i=1}^n$	Price and volume (n levels)
Time-insensitive Set	
$v_2 = \{(P_i^{ask} - P_i^{bid}), (P_i^{ask} + P_i^{bid})/2\}_{i=1}^n$	bid-ask spreads and mid-prices
$v_3 = \{P_n^{ask} - P_1^{ask}, P_1^{bid} - P_n^{bid}, \lvert P_{i+1}^{ask} - P_i^{ask}\rvert, \lvert P_{i+1}^{bid} - P_i^{bid}\rvert\}_{i=1}^n$	price differences
$v_4 = \{\frac{1}{n}\sum_{i=1}^n P_i^{ask}, \frac{1}{n}\sum_{i=1}^n P_i^{bid}, \frac{1}{n}\sum_{i=1}^n V_i^{ask}, \frac{1}{n}\sum_{i=1}^n V_i^{bid}\}$	mean prices and volume
$v_5 = \{\sum_{i=1}^n (P_i^{ask} - P_i^{bid}), \sum_{i=1}^n (V_i^{ask} - V_i^{bid})\}$	accumulated differences
$v_6 = \{\frac{1}{n}\sum_{i=1}^n (P_i^{ask}/V_i^{ask}), \frac{1}{n}\sum_{i=1}^n (P_i^{bid}/V_i^{bid})\}$	volume-weighted average price
$v_7 = \{\sum_{i=1}^n (V_i^{ask} - V_i^{bid})\}$	volume imbalance
Time-sensitive Set	
$v_8 = \{dP_i^{ask}/dt, dP_i^{bid}/dt, dV_i^{ask}/dt, dV_i^{bid}/dt\}_{i=1}^n$	P and V first derivatives
$v_9 = \{d^2P_i^{ask}/dt^2, d^2P_i^{bid}/dt^2, ^2\, dV_i^{ask}/dt^2, d^2V_i^{bid}/dt^2\}_{i=1}^n$	P and V second derivatives
$v_{10} = \{d\sum_{i=1}^n (N_i^{asks} - N_i^{bids})/dt\}$	orders imbalance first derivative
$v_{11} = \{d^2\sum_{i=1}^n (N_i^{asks} - N_i^{bids})/dt^2\}$	order imbalance second derivative
$v_{12} = \{d\frac{1}{n}\sum_{i=1}^n (P_i^{ask}/V_i^{ask})/dt, d\frac{1}{n}\sum_{i=1}^n (P_i^{bid}/V_i^{bid})/dt\}$	VWAP first order derivative

Table 1: Feature vector sets.

By concatenating the features in the above table, we construct the feature vector $\bar{v}(t)$ for the state of the order book at time t. We sought to capture all the first order factors and several second-order effects that arise in the interplay between influence price, volume, time,

and order flow. We opted to limit the time lag factor to two periods following consultation with Greg Kapoustin at AlphaBetaWorks on the informational value of time decay during the project. Nevertheless, random forest algorithms are by design are robust against surplus features. In future developments, we may seek to grow the feature vector with exogenously derived information distinct from the order book.

3.3 Label Construction

A machine learning system takes a set of features as inputs and trains an algorithm to assign likelihoods for labels that classify the data. Features that may have great explanatory reach will invariably perform poorly if the labels they seek to predict are ill constructed. Thus, label quality is paramount in our project. To this aim, we iterated over three successive label models over the course of the project as our understanding of the data and nuances of market micro-structure evolved.

Our initial intuition was to construct a midpoint-change binary label that sought to generate probabilities for $P(\text{Midpoint}_t < \text{Midpoint}_{t+1})$ and $P(\text{Midpoint}_t > \text{Midpoint}_{t+1})$. However, this model failed to account for the issue of spread as the distance between the bid and ask, which could significantly change over the course of trading. A midpoint-crossing label would not be able to account for this nuance.

Thus, our second iteration of our label sought to incorporate this insight via a three-way spread-crossing label that predicted the likelihoods for an upwards cross, downwards cross or spread stationarity, defined respectively as $P(P_{t+1}^{\text{bid}} > P_t^{\text{ask}})$, $P(P_{t+1}^{\text{ask}} < P_t^{\text{bid}})$, or $P(P_{t+1}^{\text{bid}} < P_t^{\text{ask}}$ and $P_{t+1}^{\text{ask}} > P_t^{\text{bid}})$. However, we realized this label did not accurately reflect the true measure at hand: profit and loss. An up or down classifier that consistently delivers above 50:50 odds may still lose money if the average loss is significantly greater than the average gain per trade.

In our final iteration, we arrived at a set of two labels that took account of both issues raised by the simple midpoint-crossing and the trinary spread-crossing labels. We constructed a binary classifier that predicted the probability of PnL over the next period exceeding a threshold value.

- **AREA** - This label is a time-weighed PnL over the next period ("area under the price movement curve"). The PnL is measured from the midpoint price for the aim of simplifying the label to be generalizable for both long/short signals without having to compute bid/ask specific values.

- **VWAP** - The volume-weighted average price (VWAP) is calculated based on the inner bid and ask quantities. The label calculates whether the VWAP goes up or down in the window.

4 Methodologies

First, we used machine learning algorithms to train a model to predict the labels described in the section above. We trained the model on a rolling window basis. In other words, we retrain the model everyday using the two-week data prior to the trading day. After that, based on these predicted labels we got from the model, we will execute the trade according to our trading strategy explained below.

4.1 Prediction Model

We decided to use two models: random forest and logistic regression. Each is described below and we give our justification for choosing each as well as the drawbacks. There is a lot of literature on each, so we only present a summary of the methods here.

Random Forest is a machine learning method that is a generalization of random decision trees. In the training portion, a forest of random decision trees are created. This is done by randomly choosing a subset of data and features at each branch in the tree and using the best split. After training, new data is run through the trees to get a prediction. The prediction is the most common result and the confidence for this prediction would be the fraction of trees that predict this result. Random decision trees tend to overfit and, while they have a low bias, have very high variance. Random forests address this problem by constructing and averaging out many different decision trees. Thus the random forest method is very resistant to overfitting, irrelevant features, and so on. This is the primary reason we chose the random forest model, because overfitting can be very dangerous in the context of algorithmic trading. A weakness of the random forest model, however, is that it can be computationally expensive and is not interpretable.

Logistic Regression is a form of regression often used when the dependent variable of interest is a binary event (such as VWAP up or down, or AREA positive or negative). The left-hand side of the regression is the "log-odds ratio," which is the log of the ratio of data that have a label of 1 given the predictors over the rest of the data. The right-hand side is a standard linear function of the features, where the coefficients are to be estimated. The coefficients are usually found with maximum likelihood estimation. Strengths of logistic regression are its speed and interpretability: the coefficients correspond to the change in the log-odds ratio for a unit increase in the given feature. One drawback, however, is that it assumes a certain functional form, unlike random forest, and so is more susceptible to overfitting.

4.2 Trading Strategy

A problem we ran across in the earlier iterations of our strategy was the issue of inventory management. The nave approach to this issue was to build up a net inventory of long and short positions until closing out the difference at the end of day. Naturally, this approach was

flawed as it became a directional bet on whichever way the market was trending during the day (and hopefully, where our label was correctly predicting), resulting in a poor substitute for a beta replication trade.

Instead, we chose to take each discrete trading interval as a PnL event to close out all previous position and place new bets. The size of the trade will be exponentially weighted by the classifier probabilities. The random forest and logistic regression methods allow us to gauge the confidence of our labels and size our bets accordingly, an improvement over the support vector machine approach taken by Kercheval and Zhang (2013).

5 Experiments and Results

We conducted simulations primarily on data from January 2015. The training period was from January 5, 2015 to January 16, 2015. And the testing period was from January 19, 2015 to January 30, 2015. We chose securities (stocks and ETFs) based on liquidity, selecting those with the highest liquidity. The tickers we chose were SPY, XLF, VXX, EEM, IWM, UVXY, USO, QQQ, XLE, MSFT, CSCO, IVV,

Within the training period, we tuned certain parameters. The two main parameters that were tuned were the trading window length and the decay rate in constructing the AREA label (which is time-weighted, but where later times are weighted less according to the decay rate). Two heatmaps are given below in Fig. 1. You can observe that for XLE, the optimal parameters are a window of 7 seconds and a decay of 0.7, whereas for XLF the optimal parameters are 15 seconds and 0.9. In our simulations, we found that tuning the model for each stock had a significant effect in the strategy's performance.

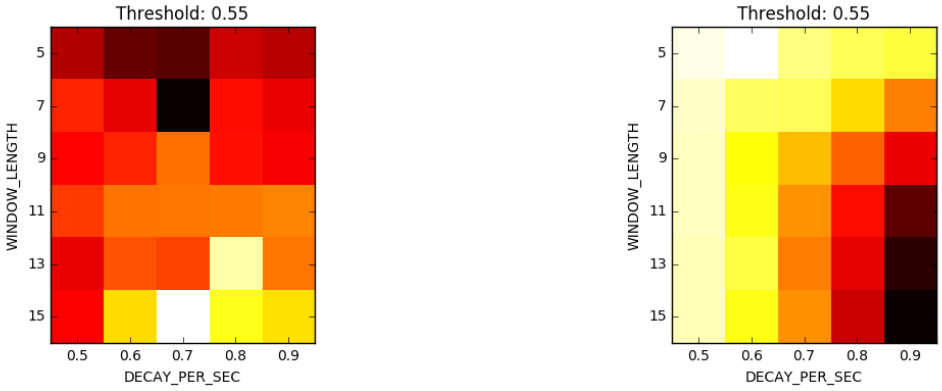

Figure 1: Heat map of accuracy for different decay and window length parameters. (Left) XLE (Right) XLF.

To investigate the practical importance of choosing the right hyperparameters, we computed the PnL per trade for various securities under different parameter choices. These are plotted in Fig. 2. We can see that the parameters clearly have an effect on profitability

(especially with MSFT in this example) and that different securities have different optimal parameter choices.

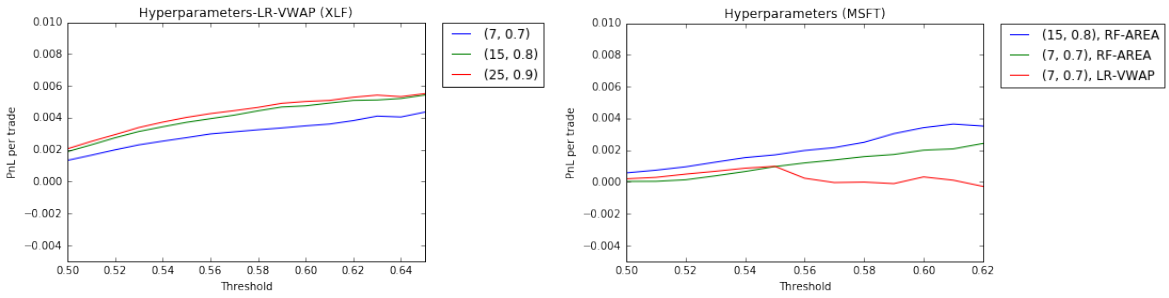

Figure 2: PnL per trade for XLF (top) and MSFT (bottom) for different model and threshold choices.

For each instrument, we tested both models' prediction accuracy against the prediction threshold. More precisely, this means the percent of the time that the model's prediction is correct when we consider only those predictions that meet the threshold (that is, when taking into account only those predictions which the model is at least as confident about as the threshold specifies. These accuracies come from only one set of parameters. We would expect the accuracy numbers to improve if we tuned each model for each ticker. Below, Fig. 3 shows the accuracy for the logistic regression and Fig. 3 shows the accuracy for the random forest model. Fig. 3 plots the difference in accuracy (the random forest's accuracy minus the logistic regression's).

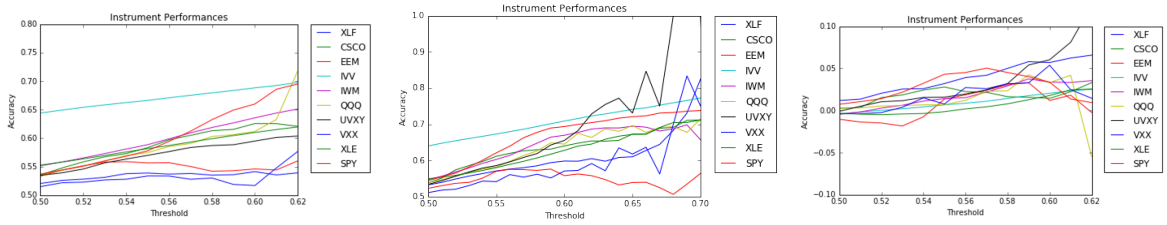

Figure 3: Accuracy of the model vs. threshold (Left) Logistic Regression (Center) Random Forest(Right)Difference in accuracies of the model (random forest accuracy minus the logistic regression accuracy.

First of all the plots are encouraging for both models: we achieve an accuracy of greater than 50%, even at the 50% threshold level, and the accuracy increases with an increasing threshold, as we would expect. (The exception to this last observation is the fact that after a certain threshold, the number of data points we have that achieve the threshold becomes very low, so the accuracy numbers become very sensitive to just a small number of correct or incorrect predictions. The figures do not show this region.) Secondly, we note that in general the random forest model achieves a better accuracy. Fig. 3 shows that for most instruments, the difference is greater than 0, meaning that the random forest's accuracy is

better. Furthermore, as threshold increases, the accuracy difference increases, meaning that the random forest model is even more accurate for higher thresholds.

We investigated two models (random forest and logistic regression) and two labels (volume-weighted midprice movement and the AREA label) to use, giving four total combinations. Fig. 4 shows PnL per trade results for each of the four model options, as well as threshold choice, for XLF and MSFT. For XLF, we see that using logistic regression with the VWAP label performs best at all thresholds, whereas with MSFT, logistic regression with the AREA label performs best at lower thresholds and the random forest with the AREA label performs best at higher thresholds. If we executed our strategy in reality, the choice of model and label would be essentially another parameter to tune, and these results show that different models and labels are better suited for different securities.

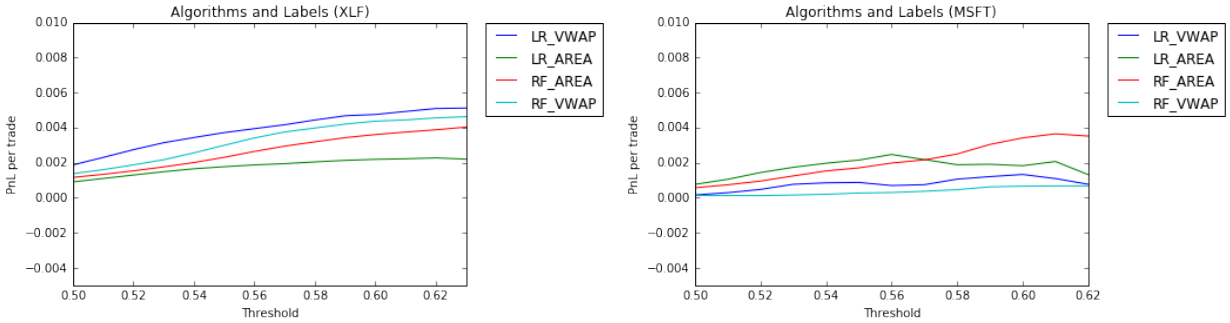

Figure 4: PnL per trade for XLF (left) and MSFT (right) for different model and threshold choices.

6 Performance

The strategy was run on each of the selected securities. For example, Fig. 5 shows the profit and loss (PnL) for XLF over the testing period. Clearly the strategy performs quite well. The PnL steadily increases over the period, indicating a high Sharpe ratio.

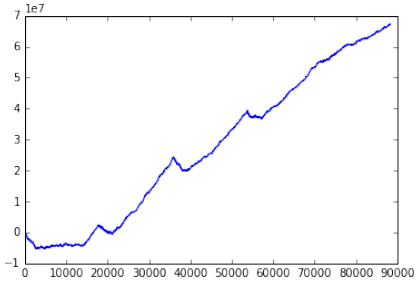

Figure 5: Profit and loss over the testing period for XLF.

Below in Fig. 6 we show the profit per trade for all tickers under two different model settings and a range of threshold choices. In general, the random forest model performs

better than the logistic regression model, but again we note that this depends on the ticker. For certain tickers, the logistic regression model performs better, although these are an exception to the general pattern.

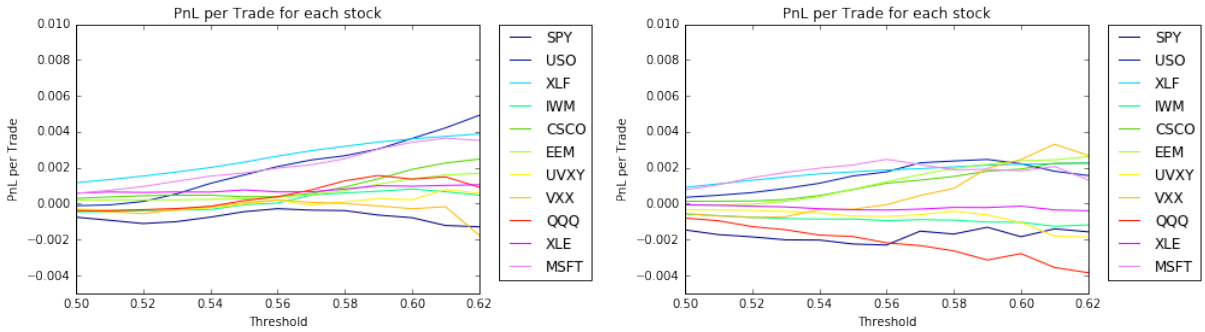

Figure 6: PnL per trade for all stocks across thresholds, for random forest (top) and logistic regression (bottom) models using the AREA label. The parameters are a window of 15 seconds and a decay parameter of 0.8.

Table 2 shows the profit for each security for our testing, the number of trades, and the profit per trade. Profit per trade is the primary measure of success for us. This measurement is more applicable to high-frequency settings (in which calculating the Sharpe ratio is sometimes difficult and does not always have the same interpretation as in lower-frequency trading). For all but two securities, we make a positive profit. The profit per trade is also very encouraging, at generally a few cents per trade. Baron, et. al., found that high-frequency traders in S&P options earned a profit of $0.25 per contract, which would correspond to $0.0025 per trade in our context. Thus, we are in line with the industry-standard profit, which is an encouraging sign. Once again, the point is made that certain model setups perform better for certain tickers.

	XLF	**SPY**	**MSFT**	**CSCO**	**IVV**	**IWM**	**VXX**	**EEM**	**UVXY**	**USO**	**QQQ**	**XLE**
Profit	36.0	57.6	2.4	25.4	-200.5	3.8	6.8	5.0	-1.8	24.4	28.7	12.1
Trades	10968	31580	1509	19184	151321	10345	1871	2234	1289	3987	7846	10291
PnL/Trade	0.33	0.18	0.16	0.13	-0.13	0.04	0.36	0.22	-0.14	0.61	0.36	0.12
Profit	63.9	18.2	0.8	24.9	-77.5	-0.6	6.6	5.8	6.5	12.8	22.2	2.3
Trades	16798	5183	2768	23425	60781	31111	4756	2284	2575	3336	6090	2591
PnL/Trade	0.40	0.35	0.03	0.11	-0.13	-0.01	0.14	0.25	0.25	0.38	0.36	0.09

Table 2: Profit, number of trades, and profit per trade for each security. On the top, the model here is random forest with AREA label, window of 15 seconds, decay of 0.8, threshold of 0.6. On the bottom, we use the VWAP label (the other parameters are the same).

7 Discussion

In this section we first examine one security to come up with potential explanations for the model's sub-par performance on that security. Then we discuss in more generality, the successes and limitations of the model, and directions for future improvements. Overall, we were encouraged by our results, but noticed that the strategy performed poorly on the ticker IVV (an S&P 500 ETF).

During further analysis, Fig. 7, we see that in the first two panels, the price initially moves up and then declines to below the original price over the trading window; in the third panel, the opposite happens. The predicted label in the first two cases was an increase in price, and in the third, a decrease. (The model used here is the random forest model with the AREA label, window of 15 seconds, decay of 0.8, and threshold of 0.6.) This is partly what were are trying to capture in our refined label, so that we can capitalize on movements like this. In fact, this is exactly the reason we use time-weighting and the decay parameter in the construction of our AREA label. However, our strategy dictates that we always close out our trade after exactly the trading window. This is why the VWAP label still performed better in some instances, and especially in the case of IVV (although we still lose money, we lost less money overall) even though we felt that our AREA label better captures PnL.

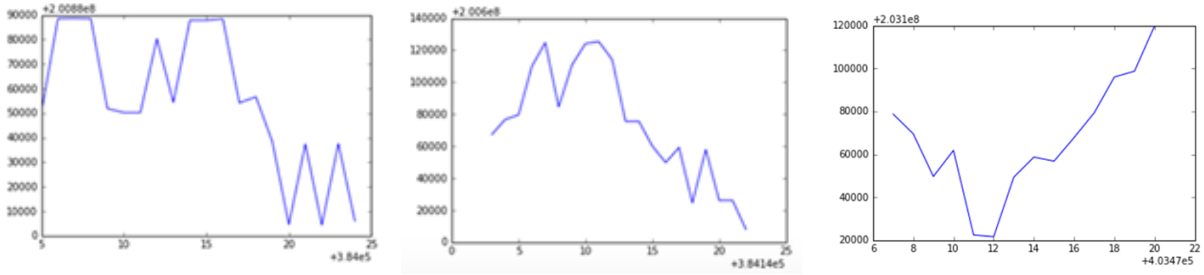

Figure 7: Mid-price movement of IVV for illustrative trading windows.

Another illustration of the subtle issues involved is given in Fig. 8. In the first panel the model dictates not to trade, perhaps because of the initial stagnation. However, over the window, the price clearly rises, indicating that buying would have been profitable. With a high decay parameter, this later rise in price would be severely discounted. Tuning the decay parameter is clearly very important.

Using our AREA label, we hope that we make the correct trade at the peak or trough within the window, but this is not guaranteed. To further improve our strategy, we could try to come up with ways to detect and close out trades early in these situations where the prediction is initially correct, but ends up incorrect at the end of the trading window, which is more in line with our AREA label.

Additional testing also revealed that we could actually make a profit on IVV if we chose a window length of 60s and a less harsh decay, such as 0.95-0.99, due to the overall less volatile nature of the movements for IVV. However, while this increases the performance of IVV, it did not do as well on other stocks. Because of the computation times, we ran a single

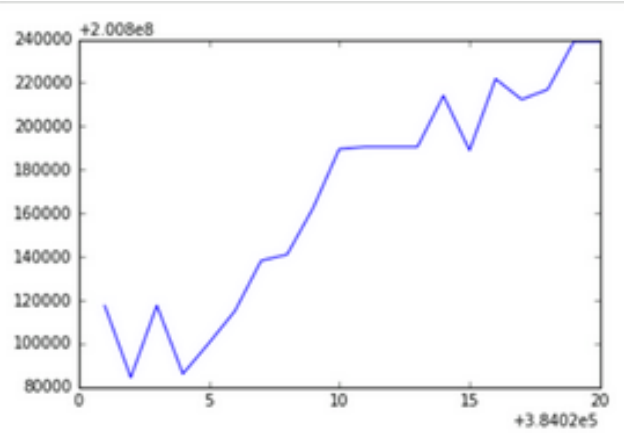

Figure 8: Mid-price movement of IVV over a trading window to illustrate the significance of the decay parameter.

model (RF, decay = 0.8, window = 15, threshold = 0.6) over all stocks to generalize as best we could.

7.1 Successes and Limitations

In evaluating our strategy, there are some strengths and weaknesses that stand out. As for strengths, first and foremost, the models have a reasonably good accuracy. This is the most encouraging since because predictive power should in some way be able to be converted into a trading strategy.

Secondly, we note that we get the relatively high profit/trade results, with small variance if we use a longer training period. This indicates that we have been successful in converting the predictive power of our models into a trading strategy. We also believe that a strength of our approach is it can be generalized and applied to many stocks and ETFs. Finally, a major strength is that the strategy performed well even during the tumultuous period of the financial crisis in 2008.

We now discuss some potential limitations of the model. High prediction accuracy does not necessarily lead to profit. So even though our model appears to have predictive power, this does not in and of itself guarantee successful trading. Our trading strategy is still relatively simple, and could be improved to be more assured of profiting on the predictions of the model. Next, the hyperparameters of the model have to be tuned for each ticker. This is takes time and computational resources (which, for us, are very limited, but one could imagine that in a different setting, like a wealthy hedge fund, this would be less of an issue). Finally, another drawback of our approach is that the random forest model is hard to interpret. With the logistic regression model, the coefficients have an interpretation with regards to the log-odds ratio of the outcome. With the random forest, which was generally the better model, there is no easy interpretation: one cannot interpret the forest that results, and the feature importances can be counterintuitive.

7.2 Areas for Improvement

Naturally, although our strategy has given promising results, there are aspects that could be improved given more time and resources. Most importantly would be to address the fact that the strategy trades at the mid-price, rather than at the bid and ask prices. This decision was made to test the preliminary efficacy of the strategy. The promising results indicate that there perhaps is a way to convert this to a strategy that would be profitable trading at the bid and ask. We note that in some cases, the profit/trade exceeds the spread, and so would make money as is. However, we would want to be more clever and perhaps use a market-making-like strategy around our predictions.

Even without adapting the strategy to trade at the bid and ask prices, there are some things that could be enhanced. For example, using more data to train and test. In this regard, we are constrained by our limited time and resources, but we feel that it is always advantageous to incorporate more data and do more extensive testing. For example, we could stress test with tumultuous historical periods in the market and hypothetical data meant to assess the robustness of the strategy. Including more features would be another improvement. Because Thesys only provides data on the orderbook, we did not incorporate any external data. The model would be probably improved, however, if we included features such as the VIX and data from related securities to help make our predictions.

8 Conclusion

In this project, we sought to use the tools of machine learning to predict short-term price movements based on order book data, and then build a strategy to profit on these predictions. We chose to use random forest and logistic regression models and constructed appropriate labels to indicate when a trade should be made. Then we built a strategy on top of this model, and tuned the hyperparameters of the model over a training period to try to maximize profit. We conducted simulations on a testing period. The results are promising because the strategy is able to make a consistent profit for most of the securities that we tested. To improve the strategy, the main aspects we would like to change are to adapt the strategy to trade at the bid and ask prices and to make the training, testing, and model more sophisticated.

References

[1] Flepp R., et al. The Liquidity Advantage of the Quote-Driven Market: Evidence from the Betting Industry, The Quarterly Review of Economics and Finance. 2016.

[2] Furfine, C. Decimalization and Market Liquidity, Economic Perspectives. Federal Reserve Bank of Chicago, 4Q/2003. https://pdfs.semanticscholar.org/b0c0/67fab9c85c97b19027eadb7502926d7ef509.pdf

[3] OHara, M. High Frequency Data in Financial Markets: Issues and Applications, Journal of Empirical Finance, 1997. http://www.sciencedirect.com/science/article/pii/S0927539897000030

[4] Avellaneda, M. and Stoikov, S. High-frequency trading in a limit order book, Quantitative Finance, 2008. https://www.math.nyu.edu/faculty/avellane/HighFrequencyTrading.pdf

[5] Khashanah, K., Florescu, I., and Yang, S. High-Frequency Trading: A White Paper, IRRC Institute, September 2014.

[6] Kaya, O. High-frequency trading - Reaching the limits, Deutsche Bank Research, May 24, 2016.

[7] Laughlin, G. Insights into High Frequency Trading From the Virtu Initial Public Offering, 2014. https://online.wsj.com/public/resources/documents/VirtuOverview.pdf

[8] Andersen, T., Bollerslev, T., Intraday Seasonality and Volatility Persistence in Foreign Exchange and Equity Markets. Kellogg Graduate School of Management, Northwestern University, Working Paper # 193. 1994.

[9] Guillaume, D., Pictet, O., Dacorogna, M., On the Intraday Performance of GARCH Processes. Working Paper, Olsen and Associates, Zurich, Switzerland. 1995.

[10] Boser, E., Guyon, I., and Vapunik, V. A Training Algorithm for Optimal Margin Classifiers, 92 Proceedings of the Fifth Annual Workshop On Computational Learning Theory, 1992. http://w.svms.org/training/BOGV92.pdf

[11] Breiman, L. Random Forests, Department of Statistics, University of California, 2001. https://www.stat.berkeley.edu/ breiman/randomforest2001.pdf

[12] Kercheval, A. and Zhang, Y. Modeling high-frequency limit order book dynamics with support vector machines. University of Florida, 2013. http://www.math.fsu.edu/ aluffi/archive/paper462.pdf

[13] Kearns, M. and Nevmyvaka, Y. Machine Learning for Market Microstructure and High Frequency Trading. University of Pennsylvania, 2013. https://www.cis.upenn.edu/ mkearns/papers/KearnsNevmyvakaHFTRiskBooks.pdf

[14] Baron, M., J. Brogaard, A. Kirilenko. The Trading Profits of High-Frequency Traders. 2012. https://faculty.chicagobooth.edu/john.cochrane/teaching/35150_advanced_investments/Baron_B