Vrije Universiteit Amsterdam

Master Thesis

Limit Order Execution Probability Modeling in the Crypto-Asset Market

Author:

Mark G. de Kwaasteniet

Supervisor:

Norman J. Seeger

Amsterdam, June 29, 2022

Abstract

This paper studies limit order execution probability in the crypto-asset market by comparing parametric and non-parametric models using high-frequency trading data. The models are assessed on prediction accuracy, estimation time, and variable importance. The results indicate that non-parametric models are more accurate in modeling the execution probability of limit orders. The random forest and the kernel density estimation are the recommended models when weighing both accuracy and estimation time, with order book quantity being the most influential market factor for model accuracy. To validate the conclusion of this study, that limit order execution probability in the crypto-asset market can be accurately modeled; future research should improve this methodology by extending the data samples and taking iceberg orders into account. A real-time trading test using execution probability to improve order placement would provide an ultimate confirmation of the conclusion.

Keywords: Crypto-asset Market; Limit Order Execution; Probability Modeling;

Contents

I Introduction

This research paper aims to accurately model the execution probability when placing limit orders in the crypto-asset market. The primary focus of this paper is the comparison of parametric and non-parametric probability models and whether they accurately predict limit order execution given the information available in the order book and market trades on the Binance[1] trading platform.

In both equity and crypto-asset markets, there are considered directional and non-directional traders. Directional traders are concerned with an up-or-down price movement (e.g., asset managers). Non-directional traders, like market makers, are less dependent on a directional price movement and more on trade activity or volatility. Both trader types have multiple options when placing an order (buy or sell) into the market, but this paper only focuses on the market- and limit order option.

A market order is executed directly at the best bid- or ask price, and a limit order is placed in the market for a specific price and executed only if a counterparty in the market is willing to buy or sell at that price. A market order is therefore guaranteed to be executed, providing a probability of 100% that the order will be executed. For a limit order, this execution probability might not be 100% as it depends on multiple market factors, including available counterparties. Moreover, factors like the current bid-ask spread, order book quantities, and imbalancedness of the order book are considered to influence the execution probability (Biais et al., 1995; Hollifield et al., 2004). Although a market order will be executed immediately, the price at which it is executed is the best bid- or ask price in the order book, meaning the trader incurs a bid-ask spread premium. On the other hand, a limit order provides a more favorable price but may result in opportunity costs for the trader as the execution of the order is uncertain. This suggests that there is a crucial trade-off between the execution uncertainty and the price of an order. By modeling the execution probability of limit orders, the trader can calculate the expected value of a trade, clarifying the financial trade-off between a market- and limit order.

This study examines the trade-off by measuring the variable importance and evaluating the models on accuracy and estimation time. Concluding the research by weighing both the

[1] https://www.binance.com

accuracy and estimation time with an all-encompassing goal to provide a model that predicts accurate probabilities and thus improves the order placement strategy of high-frequency traders (e.g., market makers).

Limit order execution probability has been studied to improve the order placement strategy of directional traders, using mostly parametric models to predict the execution probability or the time to fill an order (Foucault, 1999; Lo et al., 2002; Hollifield et al., 2004). There is a lack of evidence of how the probability models hold up in high-frequency trading samples as this data is rare and expensive to acquire for equity- and commodity markets. The market makers conducting this research see no benefits in sharing these results as they experience fierce competition in this sector. With the arrival of cryptocurrency and the transparent exchanges accompanied by it, the retrieval of asset data in high frequencies has become realizable. The mentioned studies have done research in equity- and commodity markets, where limit orders have the same mechanism as in the crypto-asset market. However, the price behavior differs as the crypto-asset market is considered more volatile than the average financial asset in the equity- and commodity markets (Stosic et al., 2018; Koutmos, 2020). This gap makes for unique research as both the crypto-asset market and the data frequency have not been considered in other studies when modeling limit order execution probability.

The next section, Section II, contains the methodology and elaborates on the models and evaluation methods used in this research. It is followed by Section III, where the structure and decisions regarding the data and variables are explained. Section IV and V present and discuss the results of the estimated models, respectively. This study finds that the random forest is the most accurate model in predicting limit order execution probabilities, with the quantity in the order book being the most influential variable for the accuracy of the predictions. The random forest is relatively slow compared to the kernel density estimation when comparing the estimation time of the models. This results in a recommendation of both the random forest and the kernel density estimation, dependent on the time sensitivity of the user.

II Methodology

This section presents the process of modeling the execution probability of limit orders and evaluation methods for assessing the prediction accuracy of the models. The models used in this methodology are split into parametric and non-parametric models, distinguishable by their assumptions and estimation procedures.

A Parametric Models

Parametric models are considered to have particular assumptions to validly interpret the coefficients- and predictions of the model. These are basic assumptions regarding the data distribution of variables and residuals produced by the model. Most of the literature contains studies that use parametric survival analysis to determine how much time an order can survive before execution. This estimated time to fill is used to predict the actual execution probability of an order (Foucault, 1999; Hollifield et al., 2004; Yingsaeree, 2012; Lo et al., 2002). This study, similar to Omura et al. (2000), uses a parametric logistic regression to model the execution probability and interpret the variables related to order execution.

A.1 Logistic Regression

The logistic regression is modeled using the proposed variables in Section III, where the dependent order execution variable ($Exec_t$) is regressed on several independent variables:

$$\begin{aligned} Exec_t = c_0 + \beta_1 VOL_t^M + \beta_2 QUANT_t + \beta_3 TYPE_t + \beta_4 PRICE_t + \Sigma_{y=1}^{4} \beta_5^y THEO_t^{P,y} \\ + \beta_6 SPRD_t + \beta_7 PASTVOL_t^T + \beta_8 PASTVOL_t^M + \beta_9 INV_t^B \\ + \beta_{10} INV_t^A + \beta_{11} IMB_t + \beta_{12} BEST_t^B + \beta_{13} BEST_t^A + \epsilon_t \end{aligned} \tag{1}$$

where the cumulative logistic distribution of the order being executed within one second is given by

$$P(Exec_t) = \frac{1}{1 + \exp^{-Exec_t}} \tag{2}$$

where $P(Exec_t)$ denotes the probability of order execution. Due to the non-linear distribution of the logistic model, the Ordinary Least Squares (OLS) method cannot be used to estimate the parameters. Maximum likelihood estimation therefore provides the optimal parameters using

the log-likelihood function:

$$\log L(X, Exec; w) = \sum_{t=1}^{N} \ell_t \tag{3}$$

where ℓ_t is given by

$$\ell_t = -\log\left(\exp\left(-Exec_t(w^T X_t + c)\right) + 1\right). \tag{4}$$

This results in the minimization of the loss function:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{t=1}^{n} \log(\exp(-Exec_t(X_t^T w + c)) + 1) \tag{5}$$

where X, w, and c denote the vector with explanatory variables, the weights of the variables, and the intercept in the logistic regression model, respectively. C is considered the regularization parameter, given by $C = \frac{1}{\lambda}$ where λ is the regularization strength given by the model user. Thereby, the loss function not only minimizes the log-likelihood function, but the regularization also penalizes the weights of the variables to prevent overfitting of the model.

When the model parameters have been estimated, the standard errors can be calculated and model coefficients can be tested on whether they significantly affect the order execution variable, using the *Wald* test:

$$Z = \frac{\hat{\beta}_i}{\text{s.e.}(\hat{\beta}_i)}$$

where the hypotheses: $H_0 : \beta_i = 0$ & $H_1 : \beta_i \neq 0$ are tested with the calculated Z-score (Brooks, 2019).

To validly interpret the aforementioned test results, the following assumptions regarding the data and residuals are assessed; independent residuals, linearity between continuous variables, absence of multicollinearity, and no significant outliers (Stoltzfus, 2011). The main concern with parametric models is that, although an execution time model might produce accurate predictions, it can be inappropriate since distributional assumptions do not often hold in real market situations.

B Non-Parametric Models

Non-Parametric models do not have any prior assumptions regarding the data distribution of the variables, which makes them flexible since the data distributions are not always known beforehand or can change over time. Various studies used the random forest to estimate credit default probability in the banking sector, arguing that this non-parametric model often outperformed the logistic regression in terms of accuracy (Kruppa et al., 2013; Fantazzini and Figini, 2009). Their results form the hypothesis that non-parametric models have a significant advantage over parametric models when modeling probabilities due to their flexibility and possible outperformance. This hypothesis is confirmed by Maglaras et al. (2021), which uses a non-parametric recurrent neural network (RNN) to accurately estimate the time to fill an order and thereby predict the probability of order execution. The study finds that the RNN significantly outperforms the logistic regression benchmark model in terms of the accuracy of the probability predictions. To formally test this hypothesis in the crypto-asset market, this study uses a random forest and kernel density estimation to model the limit order execution probability without parametric assumptions.

B.1 Random Forest Classification

Classification and Regression Trees (CART)

The core of a random forest depends on the estimation process of many single classification trees and combining them into one model. CART is a categorization method where one estimates if an order is executed based on a set of questions about the independent variables. Each question aims to create a distinction between the classes in the dependent variable, execution or non-execution, to categorize the input and use it to predict the class when order execution is unknown. Partitioning lines represent the questions, where each partitioning line contains a specific description with a particular value related to the independent variable. When this description holds, the dependent variable equals the constant in that category. For instance, if the quantity of an order is equal to 2.5 Bitcoin and the bid-ask spread is 0.2$, there is a 70% probability that the order will be executed within one second (Hastie et al., 2009).

In a multivariate analysis, there are many combinations between the independent variables. This makes the question of a partitioning line, and thus the categorization process, complicated

to describe. This idea is being simplified by having binary partitioning, shown in Figure 1. Namely, if *Quantity* is lower than or equal to 2.5 Bitcoin and *Spread* is lower than 0.2$, the probability of executing an order within one second equals 75%. Binary partitioning is created by subsequently splitting the dependent variable's distribution into two regions. It can be visualized in a binary tree when put in chronological order. The tree nodes represent the binary partitionings, where the observations in the data sample are separated into orders where the question did and did not hold. The predictions are considered accurate if the nodes can distinguish the separated data samples by executed and non-executed orders.

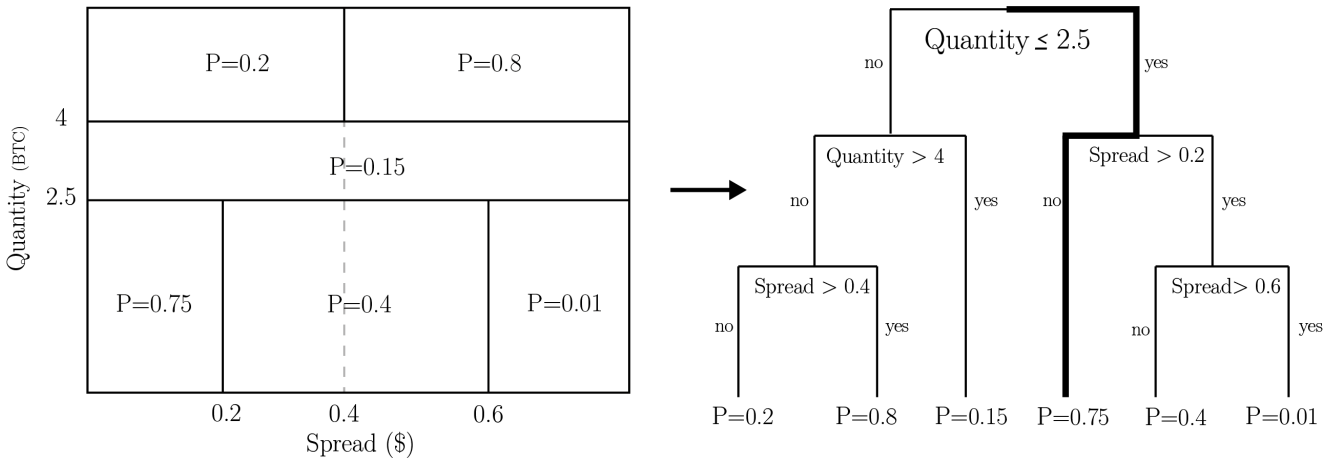


Figure 1. Binary partitioning converting into a binary tree. This figure illustrates the conversion of binary partitioning lines into a binary tree. The partitioning lines are constructed with Boolean questions related to the independent variables to sort the observations on the dependent order execution variable. *P* denotes the probability of order execution. *Quantity* and *Spread* are related to the quantity in the order book and the spread between the bid and ask, further disclosed in Section III.

To estimate the order execution variable, the binary tree can be written into a regression model, with the constant (c_i) and the partitioning regions (R_i):

$$Y_t = \sum_{i=1}^{k} c_i * I_{\{(X)\in R_i\}} \tag{6}$$

where X and I are the vectors of independent variables and the indicator function measuring the binary partitionings of the independent variables, respectively. To construct the regions and accurately predict the order execution, the classification tree makes use of the Gini Index measure. Which is an index that measures the amount of separation between the executed and

non-executed orders made by the binary partitionings:

$$1 - \sum_{i=0}^{K} (\hat{p}_{mi})^2 \tag{7}$$

where $\hat{p}_{mi}$ is given by

$$\hat{p}_{mi} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = i). \tag{8}$$

K can take two values; 0 if the order is not executed within H seconds and 1 if it is. N_m is the number of observations where order execution equals i in that specific region R_m, with m representing the specific node in the decision tree.

Figure 2 illustrates 1300 orders that are taken from the data of this study. It shows that if *Quantity* is higher than 2.5 Bitcoin, the binary partitioning leaves 700 orders, of which 600 orders were executed within one second. The Gini Index first takes the separation of the dependent variable on both sides of the binary partitioning into account to calculate the Gini Index of the end leaf nodes. After which, it takes the Gini Index of the end leaf nodes to calculate the Gini Index for *Quantity*. A perfect separation in Figure 2 would have been 700 non-executed orders if *Quantity* is smaller or equal to 2.5 Bitcoin and 600 executed orders if *Quantity* is larger than 2.5 Bitcoin.

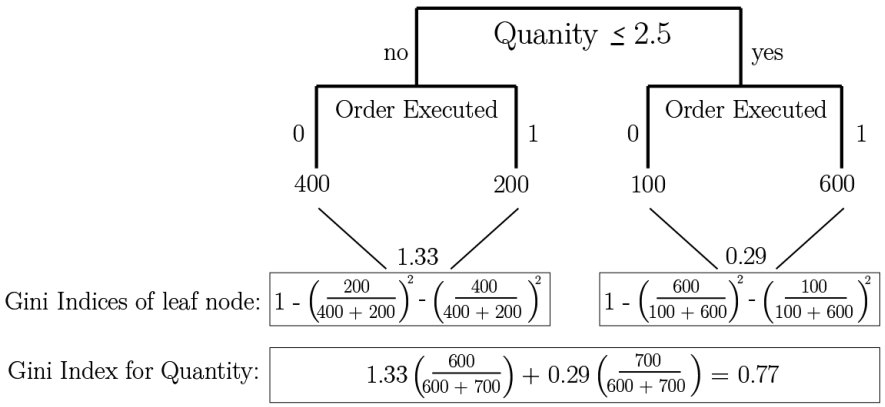


Figure 2. Calculation of the Gini Index. The figure shows the calculation of the Gini Index for a leaf node and, subsequently, an independent variable. A hypothetical situation is sketched where the split in the dependent variable is made by the Boolean question regarding a *Quantity* value.

The Gini Index is calculated separately over all binary partitionings to construct the full tree with multiple independent variables. The binary partitioning with the lowest Gini Index ends up in the tree's center. This center node represents a binary partitioning of the independent

variable that can make the largest separation in whether an order is executed within one second or not. After that selection, an iterative process begins where the Gini Index of other nodes is re-calculated, sorted in ascending order, and attached to the root node. As long as additional nodes improve the separation of the dependent variable, the iterative process continues. Whenever this is not the case, the node becomes an end node, where order execution probability is estimated: $P(Exec_t) = \frac{N_1}{N_1+N_0}$. N_1 and N_0 contain the number of executed and non-executed orders within that node, respectively (Hastie et al., 2009).

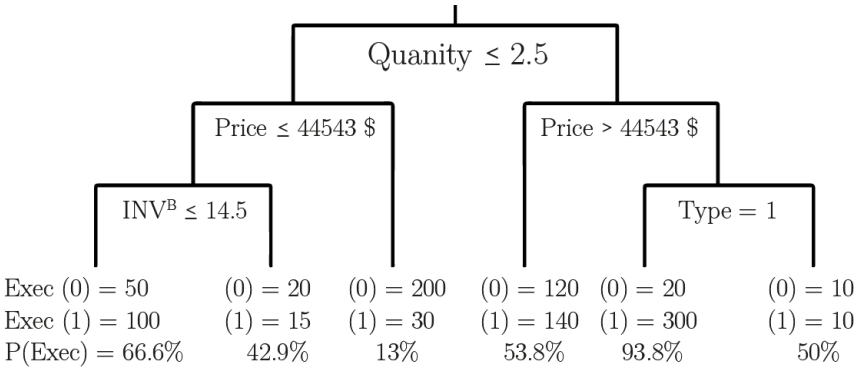


Figure 3. Binary decision tree. This figure visualizes a small hypothetical classification tree, including the probability estimations of the dependent variable based on the number of executed (1) and non-executed orders (0) in that end node.

Random Forest Classification

The Random forest was introduced by Breiman (2001) to reduce the forest error rate of binary trees. A random forest grows multiple binary trees to reduce the forest error by combining the estimated value of every binary tree. The binary trees in a random forest are all trained on a subset of the training data; this means that the binary trees all have different statements for the independent variables. Random forests can contain thousands of binary trees, resulting in high strength for the random forest if the correlation between the binary trees is low. This means that the more diverse (and acceptable) the binary trees are, the more accurate the random forest becomes.

Figure 4 provides an example where if the order quantity is larger than 2.5 Bitcoin, the first binary tree might predict an order execution probability of 50% whilst the other two predict a probability of 35% and 67%, respectively. In that case, the unweighted average majority voting of the random forest results in an execution probability of 51%.

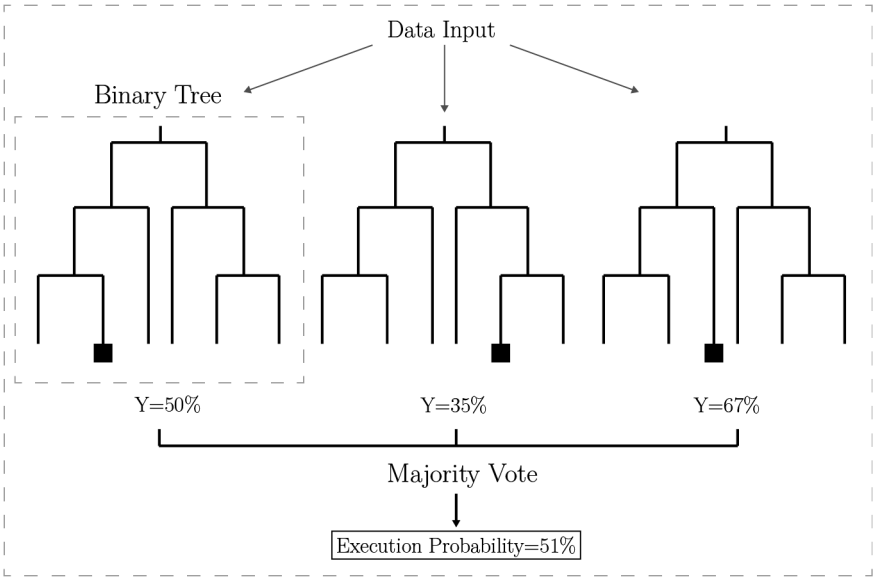


Figure 4. Random forest example. The figure sketches the process of transforming input data of the independent variables into binary trees when producing probability estimations. Combining multiple tree estimations in a majority voting for a final probability prediction.

B.2 Kernel Density Estimation

Kernel density estimation is a non-parametric method to model the probability distribution of variables based on a specific kernel. Often, economic, real-time variables are not perfectly normally distributed, making the kernel density estimation suitable for accurately mapping the distribution. First, the distribution of a single variable is visualized by plotting a histogram, where the low bins represent a low probability of that value occurring and vice versa. Then, the density of a variable is estimated using a density function $\hat{f}$:

$$\hat{f}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} K(\alpha - x_i) \tag{9}$$

where α and x_i are considered a given point in the distribution and the independent distributed sample, respectively. N is the number of observations of the variable and K is the kernel (non-negative) function. This study uses a Gaussian kernel:

$$G(X;\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{X^2}{2\sigma^2}} \tag{10}$$

where the density of X is estimated based on its data distribution given in Equation 9 by $(\alpha - x_i)$. An example is shown in Figure 5a where a Gaussian kernel is placed on every single

observation point of a variable to estimate the density using Equation 9, shown in Figure 5b.

a) Individual Gaussian Kernels **b)** Density Estimation

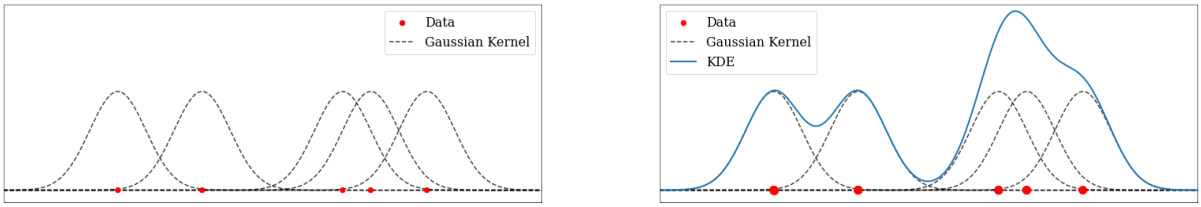


Figure 5. Kernel density estimation. The left sub figure shows the individual data points and the Gaussian kernels fitted on these single observations. The right sub figure visualizes the estimation process where the kernels are accumulated into a density function.

To estimate the most accurate density of the data, a certain bandwidth has to be specified, which determines the degree of the narrowness of the density. Too low of bandwidth will result in overfitting of the data, whereas too high of bandwidth will result in underfitting. The kernel density estimation function, including bandwidth:

$$\hat{f}(y) = \frac{1}{Nh} \sum_{i=1}^{N} K(\frac{y - x_i}{h}) \tag{11}$$

where h is the bandwidth, determined by modeling the kernel density estimation on a range of bandwidths and selecting the bandwidth that provides the highest (mean) accuracy of classifying the order execution variable.

First, the data set is split into two ways to predict the probability of an order being executed within one second. One subset contains all the observations with executed orders, and the other contains the rest. Next, the densities of the independent variables in the subsets are estimated separately using a Gaussian kernel. To visualize the density of the orders that are executed, a 16-dimensional histogram plot should be created, where one observation would include values for 16 different independent variables. The log-likelihood of an observation α is calculated using the estimated kernel density model:

$$\mathcal{LL}(\alpha) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(\alpha - \mu)^2 \tag{12}$$

where the Gaussian kernel function is used. Using the log-likelihood of observation α, a Bayesian

classification is used to determine the probability of order α being executed within one second:

$$P(Exec_t) = \frac{\exp^{\mathcal{LL}(\alpha)+\log(\frac{N_i}{N_T})}}{\sum_{i=1}^{2} \exp^{\mathcal{LL}(\alpha)+\log(\frac{N_i}{N_T})}} \tag{13}$$

where N_i and N_T denote the number of observations where the dependent variable is equal to 0 or 1 and the total number of observations in the data sample, respectively.

C Ensemble Models

An Ensemble model is not considered a model in itself but rather a method to combine the output of multiple models to predict future outcomes. For example, Kuhnert et al. (2000) combines parametric (logistic regression) with non-parametric models to estimate the probability of motor vehicle injuries. Kuhnert et al. (2000) concludes that the logistic regression model can be improved in terms of both accurateness and interpretation of variables by combining non-parametric models in an ensemble model. This study will use a similar approach where parametric- and non-parametric models are combined in an ensemble model, the soft voting classifier.

C.1 Soft Voting Classifier

The soft voting classifier combines various trained models, that can be weighted according to their importance to produce the most accurate predictions. Figure 6 shows a representation of a soft voting classification model, where the models mentioned above are used as the main drivers of the predictions. The dependent variable is estimated:

$$\hat{Exec}_t = \arg\max_t \sum_{j=1}^{3} w_j \chi_A(C_j(\mathbf{x}) = t), \tag{14}$$

where C, w, and χ_A denote the model, weight of every model, and characteristics function that combines the weights and the model with a unique data sample A separated by the binary order execution variable. The underlying thought is that the errors of the individual models are (to some extent) independent of each other. Meaning that combining the predictions of these individual models will reduce the variance because the aggregated model only makes a wrong

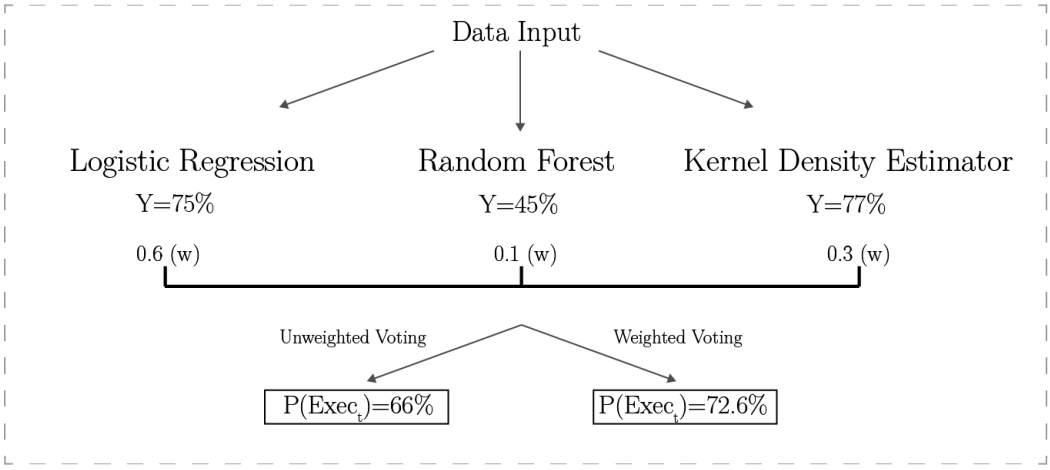


Figure 6. Soft voting classifier. The figure shows the soft voting process under unweighted and weighted voting. Weight relates to the amount of weight given to the models that are combined in the soft voting classifier.

prediction if more than half of the underlying models predict incorrectly (i.e., when unweighted voting is used) (Hastie et al., 2009).

D Model Evaluation

All the models are evaluated using the Area Under the Receiver Operating Characteristic ($AUROC$), measuring the accuracy of the model predictions on the test data set. The ROC curve in Figure 7b plots the True Positive Rate (TPR) against the False Positive Rate (FPR).

$$TPR = \frac{True\ Positive\ Predictions}{True\ Positive\ Predictions + False\ Negative\ Predictions} \tag{15}$$

$$FPR = \frac{False\ Positive\ Predictions}{False\ Positive\ Predictions + True\ Negative\ Predictions} \tag{16}$$

The construction of the ROC curve is shown in Table 7a, where the predicted probabilities are sorted in descending order, after which the actual order execution at time t is counted in the same order. Whenever the order execution equals 1, the TPR will increase with $1/N_1$ where N_1 is the total number of predictions with order execution equal to 1. Similar to the TPR, the FPR will increase with $1/N_0$ when the estimated execution of an order equals 0.

a) Sorted Predictions

$P(\widehat{Exec_t})$	$Exec_t$	TPR	FPR
0.902	1	1/5	0/5
0.700	0	1/5	1/5
0.672	1	2/5	1/5
0.629	1	3/5	1/5
0.372	1	4/5	1/5
0.205	0	4/5	2/5
0.194	0	4/5	3/5
0.172	1	5/5	3/5
0.106	0	5/5	4/5
0.045	0	5/5	5/5

b) *ROC* Curve

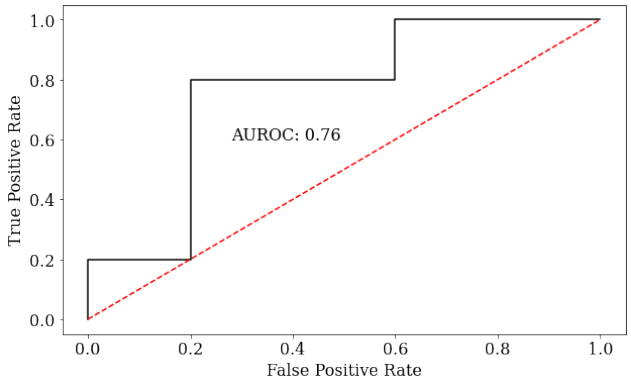


Figure 7. Sorted probability predictions into an ROC curve. The left sub figure illustrates the calculation process of the *ROC* curve, where the table values are sorted on the probability predictions, $P(\widehat{Exec_t})$. The right sub figure visualizes, the *ROC* curve and the area under the *ROC* curve (*AUROC*) when using the sorted predictions in the left sub figure. Additionally, the right sub figure contains the diagonal *ROC* curve that represents randomly guessing whether an order will be executed or not.

The *AUROC* is the proportion of the plot under the line of the *ROC* curve, resulting in a score between 0 and 1, where an *AUROC* of 1 would indicate a perfectly accurate model. The *AUROC* is scale-independent, primarily depending on how the predictions are ranked rather than measuring the specific probabilities. The *AUROC* is measuring how accurate the model is in predicting whether $\widehat{Exec_t} = 1$ or $\widehat{Exec_t} = 0$ is more likely to be happening, regardless of the preciseness of the probability (Hastie et al., 2009). For instance, the fifth prediction in Table 7a contributes to a high *AUROC* as the ranking is correct while the probability is not in line with order execution.

The *Brier* score is used as a complement to the *AUROC*, because the probability preciseness of order execution is essential to determine the expected value of a trade. The *Brier* score is used to measure the accuracy of predictions by calculating the difference between the estimated probability and the classification of the execution (0 or 1):

$$BS = \frac{1}{N}\sum_{t=1}^{N}(Exec_t - P(\widehat{Exec_t}))^2. \tag{17}$$

A limitation of the *Brier* score and *AUROC* is that they perform relatively poorly when the dependent variable is highly unbalanced, that is, more orders are executed than not executed, and vice versa. This issue arises when the order execution time is set lower than 0.5 or higher

than two seconds. Therefore, the unbalancedness of the dependent variable causes no problems for the core results of this study as these are based on a one-second execution window.

E Variable Importance

Variable importance is studied to establish essential variables when modeling probability. Related and transformed variants of these variables can be included in future research to increase the accuracy of the predictions. Contrarily, insignificant variables can be left out in future practices to reduce the model estimation time and produce more parsimonious models. Currently, studies argue that order book variables, like bid-ask spread and past trading volume, can significantly enhance the prediction performance of a probability model (Foucault, 1999; Hollifield et al., 2004; Yingsaeree, 2012; Lo et al., 2002). This study wants to confirm these results and establish new factors that could improve the model's accuracy.

The variable importance in this study is calculated using the Mean Decrease in Accuracy (MDA). MDA measures the model's performance without each variable (Fisher et al., 2018). A high value would indicate that this variable is essential in predicting whether the order will be executed within one second or not. Removing that variable causes the model to lose accuracy in predicting the execution of an order.

$$MDA_i = R^2 - \frac{1}{K}\sum_{k=1}^{K} R^2_{k,i} \tag{18}$$

where R^2 is considered the reference score:

$$R^2 = \frac{\sum_t (Exec_t - \widehat{Exec_t})^2}{\sum_t (Exec_t - \overline{Exec_t})^2}. \tag{19}$$

i denotes the independent variable and K is considered the number of iterations in which the values in the column of variable i are shuffled randomly. This creates corrupted data frames on which the R^2 is computed for measuring accuracy decrease.

III Data

This section describes the data samples and variables constructed in this study, where the Crypto-asset data is obtained through a partnership with Blocktraders. This study has been given access to high-frequency trading data of cryptocurrencies supplied from the trading platform Binance.

A Data Description

Three two-hour samples of the exchange rate BTC/USD are taken in 2021/2022 selectively, emphasizing different volatilities. Bitcoin is traded relatively often, compared to other cryptocurrencies, providing the study with more make and take order observations than other cryptocurrencies.

Table 1. Sample Statistics

Sample	Start Date	Make Orders	Take Orders	Avg. Freq. (sec)	Stdev	Volatility (%)
1	2022-03-28 11:00:00.098592	838,432	212,481	00.008587	58.24	4.23%
2	2022-04-24 11:00:00.069087	647,696	128,191	00.011115	41.38	2.42%
3	2022-05-23 11:00:00.057229	1,061,368	323,329	00.006783	63.51	4.18%

This table contains descriptive statistics of the three samples used in this study. The Start Date is given in nanoseconds, the average frequency denotes the average time between an order book update, and the volatility is retrieved from Wallabit Media LLC (2022).

Each period contains two hours of order book data regardless of the number of orders. The frequency of order book observations is dependent on the order book updates at Binance. This means that the interval between observations is not constant, as shown in Table 1. The data consists of two data samples:

i *Order Book Data:* This data contains the bid- and ask prices and quantities of the exchange rate BTC/USD per order book update of Binance. A maximum order book depth of nine ticks are used, to limit the size of the data set and thus the need for more processing power.

Date Time	bidsP_0	...bidsP_9	bidsQ_0	...bidsQ_9	asksP_0	...asksP_9	asksQ_0	...asksQ_9
2022-03-28 11:00:00.098592	44564.4	...44561.8	20.533	... 0.128	44564.5	...44568.2	0.201	... 0.001

ii *Take Order Data:* This data contains the BTC/USD orders that are executed on Binance, including the price, quantity, and type of the order (buy or sell).

Date Time	Price	Volume	Type
2022-03-28 11:00:00.098592	44564.4	0.011	0

The two data types will be concatenated into one data frame where the take order data is attached to the order book data sorting on the order book updates of Binance. The sample periods are carefully chosen based on the Bitcoin Volatility Index, sampling two market crashes (March and May) and a medium volatility period in April 2022 (Wallabit Media LLC, 2022). However, the sample periods are short relative to all the data available in a year. This implies that the results can suffer from a bias due to short-term irregularities.

B Variables

B.1 Order Book Data

The bid-ask spread variable ($SPRD$) at the time of the order placement is an indication of the current liquidity in the market. A liquid market has a significant positive effect on the execution probability due to large trading activity (Biais et al., 1995; Omura et al., 2000).

$$SPRD_t = A^P_{0,t} - B^P_{0,t} \tag{20}$$

0 denotes the first price in the order book, which corresponds to the best price available, and t denotes the time at which the specific order is placed.

$QUANT$ contains the order book quantity denoted in units of Bitcoin at the time and side of the make order. There is a positive relationship between the quantity in the order book and the time to fully execute a make order due to the First In, First Out order placement principle of Binance. This means that an order is placed at the end of the queue and will therefore take relatively longer to execute if the queue is already long.

The order book imbalance (IMB) measures the difference in the quantities at the best bid- and ask price. Empirical evidence shows that this relates to short-term price movement and has a negative relationship with the execution probability (Nevmyvaka et al., 2006; Wang and Zhang, 2006).

$$IMB_t = \frac{A^Q_{0,t}}{B^Q_{0,t}} \tag{21}$$

Order book inventory (INV) contains the unexecuted quantity in the order book. It is found to be negatively related to the execution probability of a limit order as the trading aggressiveness of investors decreases when the inventory in the order book is relatively large (Ranaldo, 2004).

$$INV_t^A = \sum_{k=0}^{9} A_{k,t}^Q \tag{22}$$

$$INV_t^B = \sum_{k=0}^{9} B_{k,t}^Q \tag{23}$$

To calculate theoretical prices of the cryptocurrency, the bid- and ask prices have to be combined to find a neutral mid-point that does or does not consider slippage and price behavior. The unweighted price ($UNWG^P$) is calculated using the best bid- and best ask price available in the market regardless of the bid or ask quantity.

$$UNWG_t^P = \frac{B_{0,t}^P + A_{0,t}^P}{2} \tag{24}$$

The volume-weighted average price ($VOLWG^P$) is calculated using both the prices and quantities of the bids and asks. According to Cao et al. (2009), this price incorporates the sentiment of market participants at that point in time.

$$VOLWG_t^P = \frac{(B_{0,t}^Q \times B_{0,t}^P + A_{0,t}^Q \times A_{0,t}^P)}{(B_{0,t}^Q + A_{0,t}^Q)} \tag{25}$$

The micro price ($MICRO^P$) is calculated by cross multiplication of both the quantities and prices of bids and asks, reflecting the assumption of informed market participants that believe large order quantities imply reverting momentum in the market (Stoikov, 2018).

$$MICRO_t^P = \frac{(B_{0,t}^Q \times A_{0,t}^P + A_{0,t}^Q \times B_{0,t}^P)}{(B_{0,t}^Q + A_{0,t}^Q)} \tag{26}$$

The deep price ($DEEP^P$) is calculated using the bid- and ask prices combined with a quantity threshold (0.2 Bitcoin). This price ensures that a certain quantity can be bought before the price increases due to low order book quantities. An example can be found in Figure 8, where more than 0.2 Bitcoins can be sold and bought at approximately 44543.0$ and 44544.5$, respectively. This clearly shows how the unweighted midpoint price (44543.5$) is not representative of the available prices and quantities in the order book.

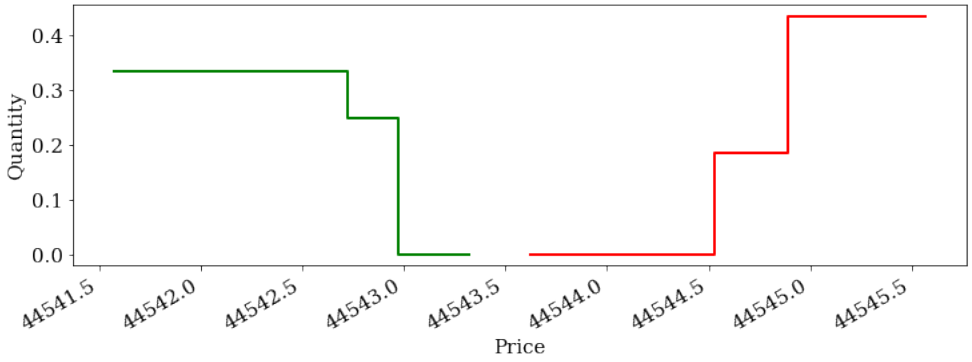


Figure 8. Order book bid- and ask price. This figure illustrates an order book where the accumulated quantity is plotted over the price of the Bitcoin. The green line represents the bid orders and the red line represents the ask orders. It is used to emphasize the utility of the deep price variable ($DEEP^P$).

B.2 Take Order Data

The order price ($PRICE$) and type ($TYPE$) variables indicate the price of a make order (denoted in US Dollars) and whether the order (make or take) is a bid (0) or an ask (1), respectively. These variables are retrieved directly from the data sample without any processing.

Take order volume (VOL^T) contains the volume of the order taken from the market, that is, an order which is either sold (bid) or bought (ask) at Binance at time point t. This means that a take order with order type 0 reduces the quantity at the bid side of the order book. This variable is taken directly from the take order data sample.

Order Book Data

Timestamp	Bid'Q^0
2022-03-27 10:30:00.0209	20.533
2022-03-27 10:30:00.0601	21.022
2022-03-27 10:30:00.0929	21.000

Take Order Data

Timestamp	Take'O'Quantity
2022-03-27 10:30:00.0528	0.011
2022-03-27 10:30:00.2547	0.090
2022-03-27 10:30:01.0150	0.004

Merged Data

Timestamp	Make'O'Quantity	Cancelled Quantity
2022-03-27 10:30:00.0601	0.500	0.000
2022-03-27 10:30:00.0929	0.000	0.022
2022-03-27 10:30:00.1259	0.000	0.000

Figure 9. Calculation of make order volume (VOL^M). This figure shows the process of creating the make order volume variable by calculating the quantity differences in the order book and adjusting these differences with the take order data in the same time interval.

Make order volume (VOL^M) contains the volume of an order that is placed in the market at time point t:

$$VOL^M_{y,t} = \left(Q^y_t - Q^y_{t-1} + VOL^T_{y,t-1}\right) \times I_{\{1|Q^y_t > Q^y_{t-1}\}} \tag{27}$$

where y and Q denote the type of the order and the quantity on that side of the order book, respectively. Moreover, I depicts an indicator function that equals to 1 whenever the quantity of the specific type order has increased over the last order book update. Figure 9 illustrates how the make order variable is constructed, merging the data frames to adjust the change in quantity with the take order variable. The past make- and take volume variables ($PASTVOL$) contain the summation of order volumes over the last second,

$$PASTVOL^M_{1,t} = \sum_{i=0}^{1} VOL^M_{t-i} \tag{28}$$

$$PASTVOL^T_{1,t} = \sum_{i=0}^{1} VOL^T_{t-i}. \tag{29}$$

Both make and take order variables are essential in determining order execution ($Exec_t$). This is the dependent binary variable which equals 1 when the order is executed within H seconds and 0 if the order is not fully executed within H seconds. Multiple time windows between 0.2 and 5 seconds are applied, reflecting the trader's patience. To construct this dependent variable, assumptions regarding order book behavior have to be made.

- A make order at time t will be placed on top of the current quantity in the market, using the First In, First Out principle of Binance. It will be placed at the end of the queue, waiting to be executed.
- A make order at time t will not be (partially) canceled throughout the time it takes to execute the order.
- Future make orders can be canceled, reducing the quantity in the order book and reducing the queue of the make order.
- It is assumed that a make order will be fully executed when the order book quantity at the time of placement is equal to the sum of the future canceled- and take order volumes. Important to note is that a make order can only be fully executed at time t if there is a take order of that type within the last order book update. Otherwise, this assumption could initiate an execution based on a canceled order, which is theoretically impossible.

C Data Visualization

The data samples are clean and balanced after checking on incorrect inputs. Data is assumed clean when it has only correct inputs and balanced when there are no missing values that require imputation. By looking at the summary statistics of the order book data in Table 2, it becomes clear that there are no missing and incorrect input values in this data set (e.g., a negative value for minimum price).

Table 2. Descriptive Statistics

	count	mean	std	min	25%	50%	75%	max
VOL^M	2547496	0.238	0.942	0.000	0.000	0.010	0.100	538.638
$PRICE$	2547496	38303.251	7228.597	30330.000	30480.500	39575.300	47184.300	47417.000
$QUANT$	2547496	3.755	8.714	0.001	0.862	2.396	4.926	526.304
$PASTVOL^T$	2547496	2.161	8.593	0.000	0.094	0.421	1.493	389.495
$PASTVOL^M$	2547496	6.995	12.091	0.000	1.495	3.862	8.250	549.239
$TYPE$	2547496	0.500	0.500	0.000	0.000	0.000	1.000	1.000
$SPRD$	2547496	0.129	0.310	0.040	0.100	0.100	0.100	60.800
$DEEP^P$	2547496	38303.249	7228.599	30330.350	30480.550	39575.350	47184.250	47408.500
$UNWG^P$	2547496	38303.251	7228.597	30330.350	30480.550	39575.350	47184.250	47408.500
$MICRO^P$	2547496	38303.250	7228.596	30330.136	30480.510	39575.321	47184.297	47402.836
$VOLWG^P$	2547496	38303.252	7228.598	30330.032	30480.527	39575.334	47184.290	47416.998
INV^B	2547496	6.996	8.040	0.018	2.541	4.863	8.760	139.001
INV^A	2547496	8.995	28.145	0.039	2.678	5.014	8.902	551.752
IMB	2547496	82.390	710.163	0.000	0.157	1.036	6.745	102818.000
ASK^P	2547496	38303.316	7228.599	30330.700	30480.600	39575.400	47184.300	47417.000
BID^P	2547496	38303.186	7228.596	30330.000	30480.500	39575.300	47184.200	47400.000

This table contains the summary statistics for the independent variables processed from the initial data samples. Count considers the number of observations. The subsequent columns are based on the Bitcoin perpetual prices, expressed in US dollars. A, B, T, M and P are corresponding to Ask, Bid, $Take$, $Make$ and $Price$, respectively.

Figure 10 shows the balancedness of the dependent variable ($Exec_t$), which should be balanced for valid interpretation of the accuracy metrics. The figure indicates the need for manually adjusting training data sets when estimating the probability of order execution within 0.2 or more than 2 seconds. The results for the variable importance, model evaluation, and model estimation times are based on order execution within 1 second. The one-second distribution indicates that this variable's balancedness poses no danger for valid interpretation or a need for under- or over-sampling.

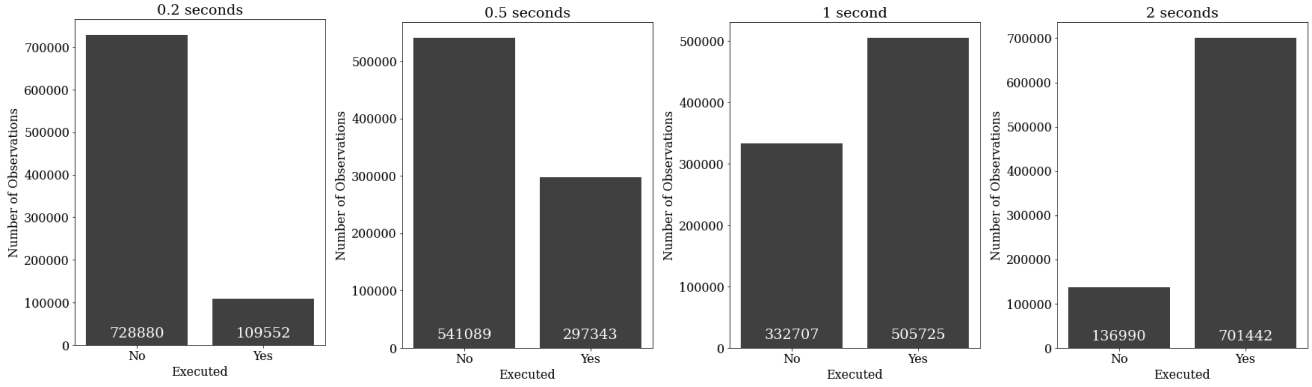


Figure 10. Distributions of the order execution variable ($Exec_t$). The figure visualizes the distributions per time window denoted in seconds. The number of observations in the category (executed or non-executed) are plotted in the bins.

To validly interpret the coefficients of the logistic regression model, the absence of multicollinearity, outlier significance, linearity between continuous variables, and independence of residuals is tested. Table 3 excludes the variables $DEEP^P$, $UNWG^P$, $MICRO^P$, $VOLWG^P$, ASK^P, BID^P, $SPRD$ as they indicate the possibility of multicollinearity and are therefore excluded from the logistic regression. The rest of the variables could be included, having a correlation of less than 0.7.

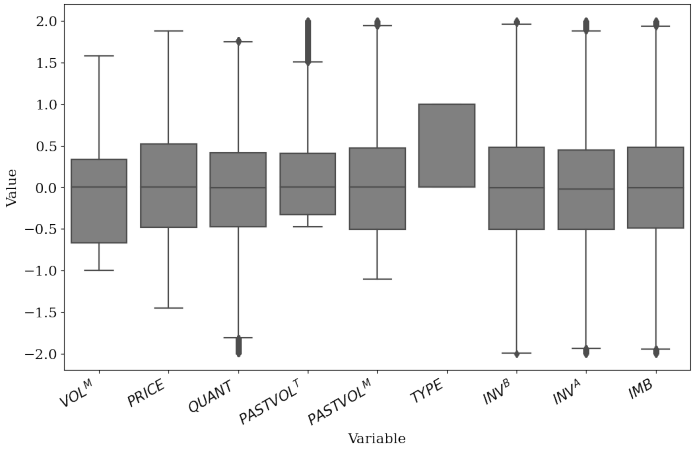
Table 3. Correlation Matrix

	VOL^M	$PRICE$	$QUANT$	$PASTVOL^T$	$PASTVOL^M$	$TYPE$	INV^B	INV^A	IMB
VOL^M	1 .0								
$PRICE$	0.0024	1 .0							
$QUANT$	0.0143	0.0262	1 .0						
$PASTVOL^T$	0.0085	0.0854	-0.1093	1 .0					
$PASTVOL^M$	0.0278	0.0901	0.3671	0.2634	1 .0				
$TYPE$	0.0042	0.0341	0.0227	-0.0046	-0.0094	1 .0			
INV^B	0.0026	0.0602	0.1620	0.0135	0.0803	-0.1619	1 .0		
INV^A	0.0009	0.0212	0.1050	-0.0187	0.0404	0.1554	-0.4294	1 .0	
IMB	-0.0008	0.0069	-0.0323	-0.0027	-0.0169	0.2467	-0.6304	0.5994	1.0

This table contains the correlation statistics between the independent variables. The *Pearson* correlation coefficient is used to denote the correlation between the variables. Variables with a correlation higher than 0.7 are excluded from the table and thus excluded from the logistic regression due to the possibility of multicollinearity.

The box plots in Figure 11a indicate that the independent variables do not contain major outliers after pre-processing the data. This means that the assumption regarding outliers can be assumed to hold. Figure 11b contains the results of the log odds regression, which is used to determine the linearity between the independent variables and the order execution variable. The results suggest that all variables have significant p-values and that the assumption regarding linearity holds.

a) Box plot Independent Variables



b) Log Odds Regression

	coef	**std err**	**z**	**P> |z|**	**[0.025**	**0.975]**
$VOL^M : \log VOL^M$	0.1704	0.003	51.359	0.000	0.164	0.177
$PRICE : \log PRICE$	7.2300	0.067	108.174	0.000	7.099	7.361
$QUANT : \log QUANT$	0.0531	0.000	181.609	0.000	0.053	0.054
$PASTVOL^T : \log PASTVOL^T$	-0.0154	0.000	-39.830	0.000	-0.016	-0.015
$PASTVOL^M : \log PASTVOL^M$	-0.0288	0.000	-133.774	0.000	-0.029	-0.028
$TYPE : \log TYPE$	2.859e-07	2.64e-09	108.136	0.000	2.81e-07	2.91e-07
$INV^B : \log INV^B$	0.0165	0.001	24.452	0.000	0.015	0.018
$INV^A : \log INV^A$	0.0055	0.000	34.846	0.000	0.005	0.006
$IMB : \log IMB$	-0.0006	8.51e-06	-65.137	0.000	-0.001	-0.001
VOL^M	-0.5389	0.006	-87.113	0.000	-0.551	-0.527
$PRICE$	-85.0423	0.786	-108.171	0.000	-86.583	-83.501
$QUANT$	-0.3233	0.002	-187.779	0.000	-0.327	-0.320
$PASTVOL^T$	0.0816	0.002	38.260	0.000	0.077	0.086
$PASTVOL^M$	0.1791	0.001	148.747	0.000	0.177	0.182
$TYPE$	-0.0769	0.005	-15.640	0.000	-0.087	-0.067
INV^B	-0.0562	0.002	-22.871	0.000	-0.061	-0.051
INV^A	-0.0357	0.001	-37.517	0.000	-0.038	-0.034
IMB	0.0054	8.01e-05	67.884	0.000	0.005	0.006
$Constant$	3.413e+05	3156.632	108.136	0.000	3.35e+05	3.48e+05

This table contains the summary statistics of the linear regression between the dependent variable ($Exec_t$) and the independent variables and log odds. The log odds variables are created by interacting the dependent variable with the log transformed version of the same independent variable.

Figure 11. Additional distributional assumption tests. The left sub figure illustrates the box plots of the pre-processed independent variables that had a $Pearson$ correlation of less than 0.7. The right sub figure reports the linear coefficients of the independent variables and the log odds to show linearity between the independent and dependent variables. The March data sample is used to construct both sub figures.

At last, Figure 12 plots the residuals of the logistic regression on its observations, showing no signs of autocorrelation in the residuals. Therefore, this study can validly interpret the logistic regression coefficients since all the distributional assumptions hold.

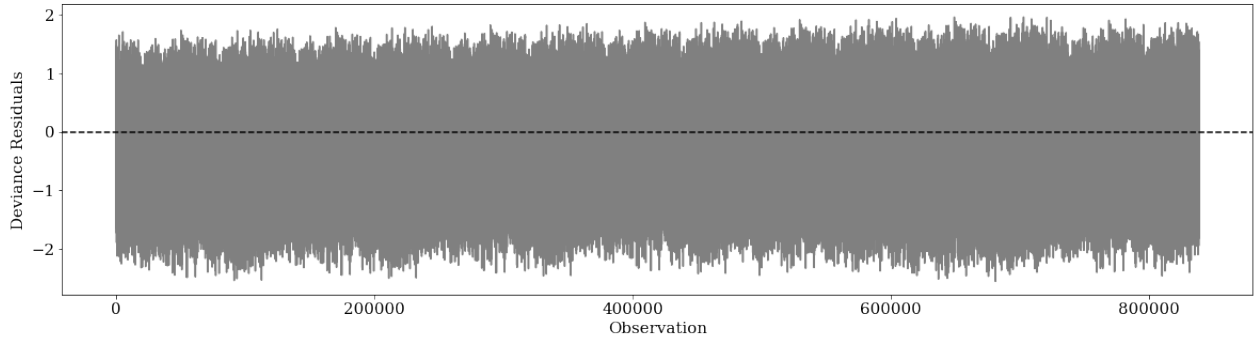


Figure 12. Residuals logistic regression. The figure visualizes the residuals of the logistic regression over the observations in the March data sample. The figure indicates that the residuals are independently distributed.

IV Results

This section compares model accuracy, estimation times, and the essential variables for modeling order execution probability. Providing the results with economic interpretations and evaluating them with the findings of previous studies.

A Model Accuracy Comparison

Since prediction accuracy is the most essential factor in modeling limit order execution, the core results of this study contain the accuracy comparison of the parametric and non-parametric models. Table 13a indicates that all the non-parametric and ensemble models outperform the parametric logistic regression model in terms of the *AUROC*. This means that when the logistic regression should predict whether it is more likely that an order will be executed or not executed within one second, it will produce more faulty predictions (e.g., a high probability prediction when the true order is not executed within one second).

a) *AUROC* Results

		Parametric	Non-Parametric		Ensemble
	Size (N)	Logistic	RF	KDE	SVC
March	167,686	0.726827	0.970120	0.852698	0.929939
	335,372	0.726399	0.989837	0.893700	0.965258
	503,059	0.726823	0.995123	0.912625	0.979307
	670,745	0.728207	0.996826	0.924794	0.985723
April	129,539	0.728305	0.977285	0.854610	0.941673
	259,078	0.728171	0.993245	0.896458	0.972881
	388,617	0.727161	0.997181	0.916835	0.984547
	518,156	0.725485	0.998299	0.930352	0.990048
May	212,273	0.722756	0.974700	0.893671	0.950768
	424,547	0.722949	0.991047	0.931577	0.978447
	636,820	0.722759	0.995187	0.948627	0.986787
	849,094	0.722732	0.996807	0.959263	0.990820

This table contains the AUROC scores of all the estimated models in this study over different time periods. The size column depicts the training size that is used to fit the model and estimate the model parameters.

b) *ROC* Curve

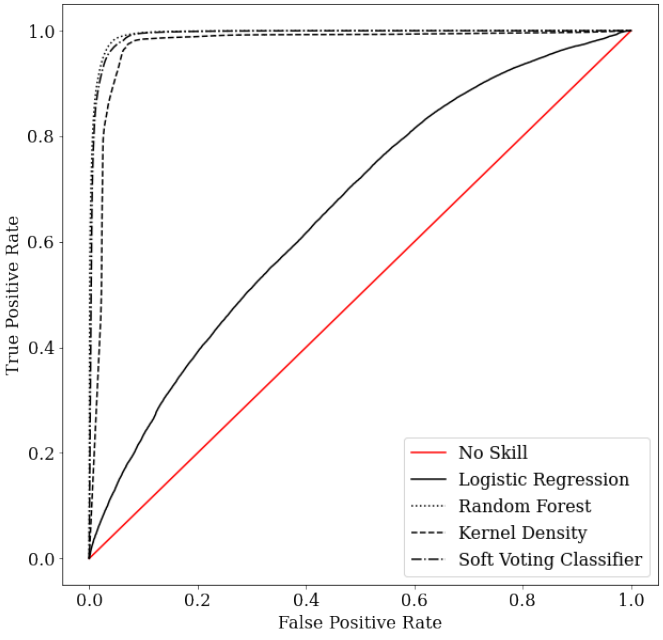


Figure 13. Results of the AUROC metric. The left sub figure reports the *AUROC* results of the various parametric and non-parametric models over the three data samples when trained on different sample sizes. The right sub figure visualizes the *ROC* curve by plotting the True Positive Rate over the False Positive Rate.

Moreover, the table indicates that the $AUROC$ metric on the test set increases as the model is trained on a more extensive data set. It is noticeable that the soft voting classifier and the kernel density estimation benefit most from this increase in training size. These results mainly contribute to the robustness of the models using various training sizes because the significance of this increase is difficult to interpret. This is due to the $AUROC$ measuring the ranked prediction instead of the absolute probability values. Figure 13b illustrates the various ROC curves of the estimated models, providing a visualization of the accuracy of the estimated models. This shows that the random forest has the steepest ROC curve, resulting in the biggest area under the curve ($AUROC$).

In line with Table 13a, Table 4 suggests that the parametric logistic regression underperforms with respect to the other probability models when measuring the accuracy with the $Brier$ score. For example, a $Brier$ score of 0.04 would indicate that, on average, the difference between the classification of an order being executed (0 or 1) and the predicted probability is 20%. In this case, on average, a probability of 80% will be estimated when an order is executed within one second. This example holds for the accuracy score of the random forest, which is approximately 0.04, varying over different training sizes. Thereby, the random forest creates probability predictions that are closest to the classification of the order execution.

Table 4. $Brier$ score Results

	Size (N)	Logistic Regression	Random Forest	Kernel Density	Voting Classifier
March	167,686	0.203542	0.071419	0.165168	0.114059
	335,372	0.203748	0.041636	0.131352	0.090575
	503,059	0.203513	0.028055	0.115875	0.078557
	670,745	0.202927	0.021880	0.106028	0.071695
Mean Decrease		0.000014	0.021682	0.024646	0.017751
April	129,539	0.198952	0.062393	0.156368	0.106563
	259,078	0.199013	0.034231	0.124221	0.083764
	388,617	0.199101	0.021902	0.109136	0.072472
	518,156	0.199230	0.016427	0.099257	0.066101
Mean Decrease		-0.000074	0.020246	0.023616	0.017045
May	212,273	0.174909	0.053979	0.114534	0.086319
	424,547	0.174854	0.029592	0.086492	0.066214
	636,820	0.174966	0.020302	0.074123	0.057377
	849,094	0.174943	0.016433	0.066663	0.052682
Mean Decrease		-0.000029	0.016839	0.020205	0.014471

This table contains the $Brier$ scores of all the estimated models in this study over different time periods. The size column depicts the training size that is used to fit the model and estimate the model parameters. The mean decrease denotes how much the $Brier$ score of the model decreases when the training size increases by 20%.

In addition, Table 4 reports the average decrease in *Brier* score when training the model on bigger sample sizes. For the kernel density estimation, increasing the training set by four times the observations would translate to reducing the difference between the probability predictions and the classification of order execution by approximately 9.5%. As high-frequency traders can use this probability to compute the expected value of a trade, this increase in accuracy might make a considerable difference in whether or not a trade should be made. Conversely, the logistic regression on average does not improve in accuracy when the training size increases.

Both Table 13a and 4 indicate that the models produce robust accuracy results regarding the different months. The models are ranked in the same order in every data sample based on their prediction accuracy. The random forest has the best, and the logistic regression has the worst accuracy performance in both the $AUROC$ and *Brier* scores. This also implies that the information to model execution probability does not change drastically over the course of several months. Models have to be re-trained to achieve high prediction accuracy over the order execution, but it seems that models do not need to be revised every quarter of the year.

As previous studies were primarily interested in the relationships between market variables and the probability of execution, it is hard to compare the model evaluation results with empirical evidence. However, this study concludes that logistic regression is performing acceptable using the $AUROC$ rule of thumb, constructed by Hosmer Jr et al. (2013). In contrast, the non-parametric and ensemble models perform excellent and outstanding in predicting the probability of executing a limit order. This study concludes that if model estimation time plays no role, non-parametric models tend to outperform parametric models in modeling order execution probability, with the size of the training set positively influencing the prediction accuracy. This conclusion is robust when using various execution time windows (0.2, 0.5, 1, 2, and 5 seconds), reported in Table A11.

B Estimation Time Comparison

As model estimation time forms a crucial part of modeling probability in high-frequency trading situations, this section is dedicated to comparing models regarding their estimation times.

Table 5 provides a clear comparison of the slow and fast models, where kernel density

estimation is the fastest model to estimate and thereby the most adequate for high-frequency trading. The estimation has been done over a data sample of a maximum of 2 hours, indicating that some models can be re-estimated every 5 seconds. In contrast, the random forest needs approximately 5 minutes, after which it can be re-estimated. For some traders, 5 minutes might not pose any limitations as they trade on stable assets where their model does not need fast re-estimation. Other traders might favor kernel density estimation as they trade in dynamic market situations where the data distributions of the variables change within minutes.

Table 5. Model Estimation Times

Sample Size	Logistic Regression	Random Forest	Kernel Density	Soft Voting Classifier
849,094	2.7814	232.8424	1.5312	237.6260
636,820	2.9602	177.3763	1.0572	181.6265
424,547	1.4371	119.9754	0.6561	121.8128
212,273	0.8101	60.7256	0.2629	61.6622
Mean Increase Time	0.6571	57.3723	0.4228	58.6546
849,094	9.7478	55.0574	1.7160	238.7228
636,820	8.8598	41.1364	1.1329	181.7178
424,547	4.3343	26.8375	0.6587	122.4682
212,273	2.7571	13.0689	0.2735	63.0468
Mean Increase Time	2.3302	13.9962	0.4808	58.5587

This table contains the time in seconds of estimating every model on various sample sizes where the dependent variable is the execution of an order within one seconds. The results are constructed using the March data sample. The upper part of the table is the time when estimating the model using only a single processor of the computer. The lower part of the table is the time when estimating the models parallel, i.e., using all the processors that the computer currently has available.

The second half of Table 5 contains the computed times with parallel estimation using all the available processors in the computer. The results indicate that the estimation time of the random forest can be reduced to a quarter of the original estimation time when more processing power is used. Thereby, estimation time in modeling execution probability might not always represent a limitation when enough processing power is available to the user.

Taking these estimation times and the results of Section A into consideration, this study concludes that the random forest would be the preferred model to use in a high-frequency trading situation. The kernel density estimation model is recommended if the user weighs estimation time relatively heavier than accuracy. Robustness checks will primarily be done on the random forest and the kernel density estimation, as these models show the highest potential for high-frequency trading situations.

C Variable Importance

Variable importance provides insight into the impact of each independent variable on the accuracy of the models. This analysis aims to provide essential variables that future studies can incorporate to improve the accuracy of limit execution probability models. Table 6 indicates that for the logistic regression, random forest, and soft voting classifier, the quantity in the order book ($QUANT$) contributes the most in terms of prediction accuracy. This result is in line with the results of Biais et al. (1995), reporting that the order book quantity significantly affects the order execution probability. As the quantity in the order book can be translated to the queue of an order, it results in a higher execution probability if the order queue is lower.

This variable should be studied more in-depth to develop more variants of order book quantity ($QUANT$) that, when included, could increase the accuracy of the model. For instance, Biais et al. (1995) also included the order quantity of the opposite order book side and found that this had a significant positive relation to the order execution probability. Moreover, a variable with the average order quantity of the past H seconds could also be included in the model, representing order book quantity behavior.

Table 6. Mean Decrease in Accuracy (MDA) Results

	Logistic Regression	Random Forest	Kernel Density	Soft Voting Classifier
$TYPE$	0.0000	0.0062	0.0198	0.0197
$PASTVOL^T$	0.0000	0.0255	0.0273	0.0249
IMB	0.0006	0.0256	0.1411	0.0890
$SPRD$	0.0009	0.0000	0.0001	0.0000
BID^P	0.0010	0.0001	0.0001	0.0112
ASK^P	0.0013	0.0000	0.0001	0.0107
$PRICE$	0.0017	0.0078	0.2139	0.1148
$VOLWG^P$	0.0052	0.0162	0.2141	0.1263
INV^A	0.0081	0.0205	0.2088	0.1171
$UNWG^P$	0.0088	0.0074	0.2141	0.1117
INV^B	0.0103	0.0595	0.2243	0.1526
$MICRO^P$	0.0120	0.0140	0.2121	0.1294
VOL^M	0.0209	0.0000	0.0151	0.0131
$DEEP^P$	0.0247	0.0103	0.2165	0.1202
$PASTVOL^M$	0.0502	0.0334	0.1659	0.1248
$QUANT$	0.1111	0.1187	0.1947	0.1796

This table contains the Mean Decrease in Accuracy (MDA) of every independent variable in the parametric, non-parametric and ensemble models, where the execution within one second is estimated. The MDA values are visualized in a bar plot in Figure A16.

In the logistic regression, random forest, and soft voting classifier, the past making volume ($PASTVOL^M$) and inventory bid (INV^B) are, next to the order book quantity, two variables that should be included in future research to enhance prediction accuracy. The importance of the inventory bid variable is in line with the results of Ranaldo (2004), reporting a significant

negative relationship with the order execution probability. The past making volume variable is constructed without prior empirical evidence, making this variable new and attractive for studies to extend research on. At last, the table shows that the Mean Decrease in Accuracy (MDA) results of the kernel density estimation contradict the results of the other models, reporting all theoretical price variables as essential variables.

C.1 Logistic Regression Coefficients

The logistic regression coefficients provide this study with in-depth information about the sign and statistical significance of the relation between the independent variables and the order execution probability. In addition, unbiased inference of the coefficients is possible as the assumptions regarding the data distributions and residuals are tested in Section III.

In line with the results of Table 6, the logistic regression coefficients in Table 14a indicate that the order quantity ($QUANT$) is significant at a 1% significance level. Order quantity has a negative relationship with the probability of execution, which is similar to the reported results of the studies as mentioned earlier (Biais et al., 1995; Hollifield et al., 2004). This study concludes from the signs of the coefficients that the probability of executing a limit order within one second increases when the queue in the order book decreases.

a) Coefficient Significance

	Coefficients	P-Values
VOL^M	-0.35249	0.00000
$PRICE$	0.21612	0.00000
$QUANT$	-0.98394	0.00000
$PASTVOL^T$	0.35240	0.00000
$PASTVOL^M$	0.88529	0.00000
$TYPE$	-0.01202	0.06477
INV^B	-0.20602	0.00000
INV^A	-0.20199	0.00000
IMB	-0.02958	0.00000

b) Absolute Values of the Coefficients

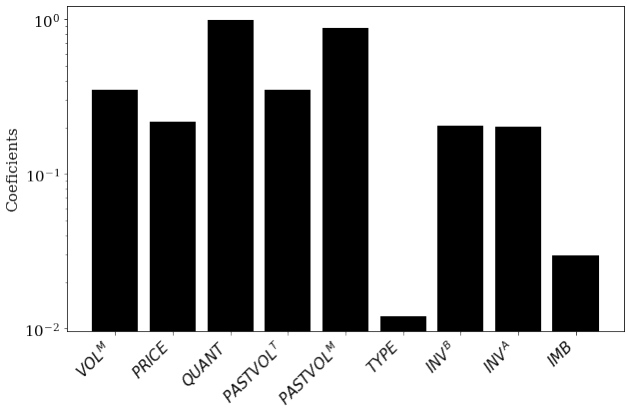


Figure 14. Logistic regression coefficients. The left sub figure reports the coefficients and the p-values of the coefficients, produced by the logistic regression and tested by the $Wald$ test, respectively. The right sub figure visualizes the absolute values of the coefficients in a bar graph.

Table 14a indicates that the coefficients of the last second of make order volume ($PASTVOL^M$) and bid inventory (INV^B) are in line with the MDA analysis as well, both having a significant

influence on the order execution. Concluding that, if many orders are placed in the last second, and the bid inventory is low, the probability of executing an order will be relatively high.

Although they did not seem to have a significant impact in the MDA analysis, all the included variables are significant at 1% significance level, except for $TYPE$. The significance of the inventory ask- and imbalance variable is in line with the results of Biais et al. (1995) and Nevmyvaka et al. (2006), negatively interacting with the order execution probability. It is important to note that these conclusions are only linked to the logistic regression. This model constructs regression coefficients that the other models do not necessarily produce and thereby cannot be similarly interpreted.

D Robustness

This section is dedicated to testing the results in different scenarios, considering different hyper parameters of the probability models and out-of-sample data to reproduce the accuracy scores of the random forest and kernel density estimation. The March data sample is used to perform these proposed robustness tests.

Pre-processing is considered an essential factor in model estimation since outliers and distributions of unprocessed data can influence the model parameters. A model's prediction accuracy is often enhanced by adequately pre-processing the data before fitting the model (Hastie et al., 2009). To test the robustness of the models, this study uses three pre-processing techniques given in Table 7. The accuracy results of the random forest and the kernel density estimation are stable, indicating that the models are very robust regarding different pre-processing techniques. Both models have lower accuracy scores when pre-processing has been done. The prediction accuracy of the logistic regression increases the most when the data is pre-processed.

Table 7. Pre-processing Robustness

	Size	Measure	Logistic Regression	Random Forest	Kernel Density	Voting Classifier
Non-Processing	754,588	AUROC	0.659130	0.997320	0.980736	0.994399
		Brier score	0.226291	0.016924	0.030331	0.043224
Log Transform	754,588	AUROC	0.718479	0.99759	0.907346	0.985832
		Brier score	0.205979	0.01695	0.121377	0.062319
Standardized	754,588	AUROC	0.690748	0.995041	0.944336	0.986171
		Brier score	0.216486	0.022657	0.091435	0.071144
Log/Standardized	754,588	AUROC	0.722230	0.966674	0.964667	0.965845
		Brier score	0.204866	0.038646	0.066246	0.067296

This table contains the model evaluation scores of the proposed models when performing three pre-processing techniques to the independent variables before model estimation. The size depicts the training size on which the model is fitted.

Interpreting the *Brier* score of the logistic regression, pre-processing would translate to an increase in accuracy of approximately 2% on average when measuring the difference between the estimated probability and the actual classification of order execution. This is in line with the expectation of this study as the logistic regression is a parametric model with strong assumptions regarding the data distribution.

The out of sample robustness is tested by training the models on one month and measuring their accuracy on another. Table 8a and 8b present the month of the training set on the horizontal axis and the month of the test set on the vertical axis. The tables indicate that both the random forest and the kernel density estimation lose accuracy when predicting the probability of executing a limit order out of sample. According to the $AUROC$, the random forest still performs better than randomly guessing whether or not the order will be executed. The accuracy of the random forest, measured by the difference between the estimated probability and the true classification, decreases by approximately 34% on average when predicting out of sample. This means that although the $AUROC$ indicates some predictability, the model loses its applicability in high-frequency trading situations when using the actual probabilities. The kernel density estimation performs equally as well as randomly guessing the order execution, as it has no out-of-sample predictability ($AUROC$=0.5). This indicates that when using these probability models in practice, there is a need for continuous estimation since a single estimated model does not perform adequately over different periods.

Table 8. Out of Sample Robustness

	AUROC			*Brier* score		
	March	April	May	March	April	May
March	0.9944	0.6024	0.6457	0.0257	0.2381	0.4784
April	0.6714	0.9962	0.6279	0.3883	0.0203	0.3247
May	0.6610	0.6156	0.9935	0.2027	0.2272	0.0238

This table contains the model evaluation scores (both *AUROC* and *Brier* score) of the random forest when fitting the model on one sample month (horizontal axis) and measuring its accuracy on another sample month (vertical axis).

(a) Random Forest

	AUROC			*Brier* score		
	March	April	May	March	April	May
March	0.9711	0.5000	0.5000	0.0401	0.6028	0.3782
April	0.5000	0.9785	0.5000	0.3782	0.0336	0.6028
May	0.5000	0.5000	0.9689	0.7255	0.7255	0.0340

This table contains the model evaluation scores (both *AUROC* and *Brier* score) of the kernel density estimation when fitting the model on one sample month (horizontal axis) and measuring its accuracy on another sample month (vertical axis).

(b) Kernel Density Estimation

The hyper-parameters of the random forest and kernel density estimation can be tweaked to increase accuracy or reduce overfitting. The robustness of the random forest is tested by measuring the increase or decrease in prediction accuracy when adjusting various hyper parameters.

i Trees: number of binary trees in the random forest.

ii Depth: the maximum depth of the tree, meaning the maximum number of vertical split points that the binary tree may have.

iii Minimum Split: the minimum number of observations that the node in the tree has to have before the split; if it is lower than this number, it converges to an end node.

Table 15a reveals that the maximum depth of the binary trees in the random forest is the most crucial factor when modeling probability as both accuracy metrics mainly improve when the maximum depth of the binary trees increases.

a) Random Forest

Trees	Depth	*AUROC* Minimum Split			*Brier* score Minimum Split		
		2	4	6	2	4	6
50	10	0.8068	0.8069	0.8071	0.1804	0.1800	0.1803
	20	0.9622	0.9626	0.9624	0.0760	0.0759	0.0770
	50	0.9669	0.9681	0.9706	0.0445	0.0444	0.0446
	100	0.9669	0.9681	0.9706	0.0445	0.0444	0.0446
100	10	0.8077	0.8074	0.8078	0.1799	0.1801	0.1801
	20	0.9626	0.9629	0.9627	0.0754	0.0758	0.0773
	50	0.9679	0.9690	0.9711	0.0443	0.0441	0.0443
	100	0.9679	0.9690	0.9711	0.0443	0.0441	0.0443
150	10	0.8077	0.8076	0.8083	0.1801	0.1801	0.1799
	20	0.9630	0.9632	0.9632	0.0750	0.0752	0.0764
	50	0.9685	0.9696	0.9714	0.0442	0.0440	0.0442
	100	0.9685	0.9696	0.9713	0.0442	0.0440	0.0442

This table contains the model evaluation scores of the random forest (both AUROC and Brier score) when various hyper parameters of the random forest are tweaked. Where Trees, Depth and Minimum Split are explained in this section.

b) Kernel Density Estimation

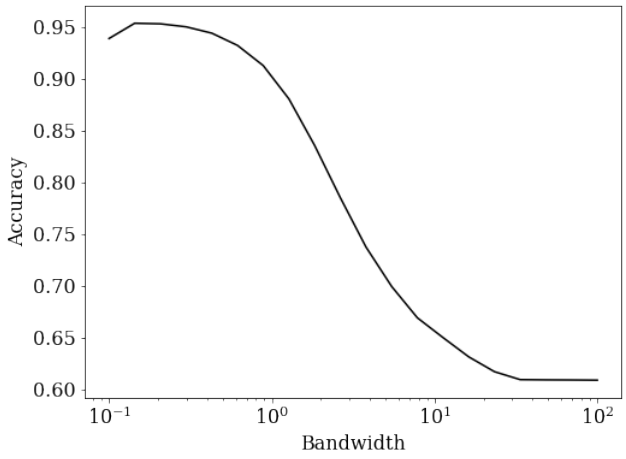


Figure 15. Hyper parameter robustness. The left sub figure reports the *AUROC* and *Brier* scores of the random forest, tested on various hyper parameter combinations. The right sub figure visualizes the accuracy of the kernel density estimation when adjusting the bandwidth parameter.

The kernel density estimation has only one hyper parameter that can be tweaked to test the robustness of the model. Figure 15b illustrates the results of the robustness test of the kernel density estimation when fitting the model under different bandwidths. It indicates that the bandwidth is essential for the model's accuracy and that too high or too low bandwidth will decrease its performance. This contradicts the hyper parameter results of the random forest, where an increase in Trees or Depth primarily increases the model's performance.

At last, this study tests the models on different cryptocurrency exchange rates; APE/AUD, ADA/GBP, and YFI/EUR. These exchange rates are considered to have no direct relationship with BTC/USD regarding back-end technology, absolute price in the order book, or fiat

currency. Table 9 indicates that the accuracy results in section A are robust over different cryptocurrencies as the random forest performs superior in all the three cryptocurrency samples. The other models also perform robustly over the different samples, except for the logistic regression on the YFI/EUR rate, as it scored visibly better than on the other exchange rates and previous tests in this study.

Table 9. Robustness over the Crypto-Asset Market

Sample	Measure	Logistic Regression	Random Forest	Kernel Density	Soft Voting Classifier
APE	*AUROC*	0.709011	0.942336	0.863032	0.908795
	Brier Score	0.216938	0.102522	0.193441	0.128938
ADA	*AUROC*	0.754487	0.999651	0.989240	0.998170
	Brier Score	0.200135	0.006741	0.034736	0.035030
YFI	*AUROC*	0.942478	0.999898	0.997963	0.999814
	Brier Score	0.084973	0.001406	0.007777	0.012996

This table contains the model evaluation scores of the proposed models on different crypto-assets; APE/USD, ADA/GBP, and YFI/EUR. The three data samples consist of one hour periods, which are drawn randomly over June 2022. Additionally, random oversampling is used to compensate for the unbalancedness of the dependent variable in the samples.

V Discussion

This section discusses the practical relevance of the results, limitations of the methodology, and assumptions to clarify the real trade applicability of the recommended models. Recommendations for future research are created based on the limitations and difficulties experienced by modeling execution probabilities.

A Practical Relevance

To illustrate the applicability of probability models by high-frequency traders, a probability grid is created to improve the efficiency of an order placement strategy. This probability grid sketches a hypothetical situation in which the make order size and price are the varying variables that traders can control. Table 10a illustrates the probability grid estimated by the random forest, which can be reproduced for every Binance order book update. It is noticeable that the probability grid provides illogical probabilities regarding the order size and price. That is, an order with a smaller size will always be executed earlier, ceteris paribus. The same goes for order price, where a better-priced order will always be executed earlier, ceteris paribus. This reasoning is inconsistent with the random forest's probability grid, which hints at an overfitted model.

Table 10. Probability Grid of Order Execution

	Order Size					
Order Price	0.00001	0.00010	0.00100	0.01000	0.10000	1.00000
44543.4	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
44543.3	0.72	0.73	0.73	0.72	0.71	0.54
44543.2	0.42	0.43	0.43	0.43	0.43	0.4
44543.1	0.42	0.43	0.43	0.43	0.43	0.4
44543.0	0.41	0.41	0.41	0.41	0.41	0.38
44542.9	0.4	0.4	0.4	0.4	0.4	0.37
44542.8	0.4	0.4	0.4	0.4	0.39	0.36

This table contains the predicted probabilities of executing an order within one second, given the independent variables; Order Quantity (1.5), Spread (0.1), Past Taking/ Making Volume (2), Inventory Bid/Ask (15), Imbalance (0.02).

(a) Random Forest

	Order Size					
Order Price	0.00001	0.00010	0.00100	0.01000	0.10000	0.50000
44543.4	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
44543.3	0.48529	0.48529	0.485289	0.485283	0.485224	0.484961
44543.1	0.484415	0.484415	0.484414	0.484408	0.484349	0.484086
44542.9	0.48354	0.48354	0.483539	0.483533	0.483474	0.483211
44542.7	0.482665	0.482665	0.482665	0.482659	0.482599	0.482336
44542.5	0.481791	0.481791	0.48179	0.481784	0.481725	0.481462
44542.3	0.480916	0.480916	0.480915	0.480909	0.48085	0.480587

This table contains the predicted probabilities of executing an order within one second, given the independent variables; Order Quantity (1.5), Spread (0.1), Past Taking/ Making Volume (2), Inventory Bid/Ask (15), Imbalance (0.02).

(b) Logistic Regression

The core of a logistic regression is its cumulative logistic distribution, where order execution is regressed on the independent variables. This gives a one-directional effect of the coefficients on the probability of executing an order. Thereby, we see in Table 10b that when holding all other variables equal, a bigger size or deeper price will always have a lower probability of

execution.

To construct this probability grid, the original data set has been manipulated by creating additional orders with selected sizes and prices. At every make order, 25 orders are added with sizes ranging from 0.00001 to 1 Bitcoin and prices that lay 0.1 to 0.5 $ deeper in the order book, after which these orders are treated and analyzed as real make orders. The study has chosen to create hypothetical orders instead of analyzing the order book beyond the best bid or ask, as this would demand a more extensive sample period since fewer orders are placed behind the best bid or ask. This adjustment in the methodology provides the study with data that has a wider variety of order sizes and price depths.

These probability grids provide a foundation for calculating the expected return for; to be placed orders. It is noticeable that limit order placement becomes more profitable than market orders if the model predicts an execution probability of 100% in that particular market situation. In this situation, the trader is able to generate a guaranteed profit of at least the current bid-ask spread. Due to the high liquidity in the crypto-asset market (Bitcoin in particular), it is often not profitable to place limit orders when the predicted probability is lower than 100% as the bid-ask spread is often too tight and therefore would reduce the expected profit relative to a market order. However, the probability grid can be adjusted to the user's interests, given the make order sizes and prices of interest. Providing the trader with the option to predict the probability of execution when placing an order deeper into the order book, thereby increasing the expected profit.

B Limitations

The make orders are obtained by measuring the quantity changes in the order book. The make orders are therefore limited by one per order book update of Binance, when in fact, multiple orders could have been placed in the market between order book update $t-1$ and t. Since this study has no access to the individual orders and order IDs, it implicitly takes a sum over the placed orders in the market and assumes that one order is placed. This forms a limitation, as is it possible that the actual orders are smaller than the orders placed in the created data set. This study, consequently, trains models on manipulated data, not fully taking trading frequency into consideration.

A second limitation is that the best bids and asks have primarily been analyzed to construct the make orders in the market. Orders can be placed and thus analyzed deeper in the order book, which would create a more diverse data set where order prices would have a greater variety of differences with the best bid or ask price. This study has considered this limitation when establishing the methodology but concluded that it would not have been feasible on a single computer to analyze deeper placed make orders. The main reason is that deeper placed make orders happen less frequently, requiring this study to extend the sample size when the sample size is already close to a million observations. To counter this limitation when creating the probability grid, hypothetical orders with various prices and sizes are placed at every make order in the sample. This provides the study with a greater variety of orders, assuming that these orders would not affect market behavior. This assumption forms a different limitation as the orders might affect market behavior and thereby could produce biased outcomes. The significance of this bias depends on the liquidity and size of the crypto asset.

The third limitation of this study relates to iceberg orders, which haven't been mentioned in this study but play a big role in estimating the time to fill an order. Iceberg orders are considered large orders that are too big to place directly into the market and are therefore subsequently placed in small portions without providing this information in the order book. The rationale of trading platforms to place orders in this manner relates to the strong reactions (positive and negative) of market participants (Frey and Sandås, 2009). This study has not taken iceberg orders into account when modeling order execution probability. That means that the estimated models could predict a high probability of order execution, although there could be an iceberg order placed in the order book without the model accounting for it.

C Future Research

Future research should look at the availability of specific make order data, where the orders that are placed in the market are reported and tracked. In that case, the data quality will go up as both the make order sizes and the time points of order placement will be more accurate. Moreover, it is more transparent when orders are canceled or partially filled in the market, making the data more complete, which could improve the model predictions. Usually, this data regarding the make orders of other traders are private and thereby not available. If this is the

case, this recommendation would only hold for high-frequency traders that place many orders themselves, which they could use to create this data set. In addition, this recommendation is related to the diversity of the data samples, including the orders that are placed deeper in the order book. In this case, future research will not have to assume that the hypothetical placed orders do not affect market behavior. This makes the data less biased and will increase the accuracy of the models when applying the models in real high-frequency trading situations.

Secondly, future research should take iceberg orders into account when forming the methodology. Iceberg orders can be observed by using time series data and analyzing recurrent patterns of order placement. If there is a recurring pattern in the order book, a binary variable could be able to describe whether or not a current iceberg order is placed. It is essential for the practical relevance that this binary variable is created and included in the proposed models of the study. In that case, the predicted probability of order execution is likely to be lower when there is, in fact, an iceberg order in the order book.

At last, future research could extend the methodology of this study by including the accuracy comparison test, developed by DeLong et al. (1988) to statistically compare two $AUROC$ scores. DeLong et al. (1988) has derived an asymptotic approach to statistically test whether the $AUROC$ of one model significantly differs from the $AUROC$ of another model. This study compares the accuracy metrics by economic interpretation rather than statistical measures. By statistically testing the difference between two $AUROC$ scores, future studies could formally conclude that one model significantly outperforms another model. Including these statistical measures will strengthen the conclusion of the study.

VI Conclusion

This study examines whether or not limit order execution probability can be accurately modeled using parametric and non-parametric models in the crypto-asset market. To reach a conclusion, models are compared using the $AUROC$ and $Brier$ score, the importance of the independent variables is established, and the applicability to real-time situations is presented using a probability grid.

The model comparison results indicate that the random forest model is superior regarding the data samples' accuracy measures. In addition, the kernel density estimation is superior in estimation time and performs well regarding accuracy. Thereby, this study concludes that the random forest is recommended in modeling execution probability when there is no need for model re-estimation every five to 10 seconds. However, the kernel density estimation is recommended when the model should be re-estimated frequently due to the dynamic market behavior of the crypto asset.

The variable importance analysis indicates that the order quantity is the variable that has the most impact on the accuracy in the majority of models. This indicates that the order queue is significant when placing an order and that an order will have a higher execution probability within 1 second if the order quantity in the order book is relatively low.

At last, the probability grid shows the applicability of the execution probability models for order placement strategies. Although the probabilities are considered accurate, this study has not been able to consistently produce a probability grid that incorporates the logical relationship between order prices and sizes.

This study concludes that the execution probability of limit orders in the crypto-asset market can be accurately modeled using order book and take order data from Binance. This study provides a clear basis for high-frequency traders to improve their order placement strategy and increase potential returns by using a probability grid before placing an order.

References

Belgiu, M. and Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS journal of photogrammetry and remote sensing*, 114:24–31.

Biais, B., Hillion, P., and Spatt, C. (1995). An empirical analysis of the limit order book and the order flow in the paris bourse. *the Journal of Finance*, 50(5):1655–1689.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Brooks, C. (2019). *Introductory Econometrics for Finance*. Cambridge University Press.

Cao, C., Hansch, O., and Wang, X. (2009). The information content of an open limit-order book. *Journal of Futures Markets: Futures, Options, and Other Derivative Products*, 29(1):16–41.

CoinMarketCap (2022). Binance trade volume and market listings.

DeLong, E. R., DeLong, D. M., and Clarke-Pearson, D. L. (1988). Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics*, pages 837–845.

Dixon, M. (2018). A high-frequency trade execution model for supervised learning. *High Frequency*, 1(1):32–52.

Fantazzini, D. and Figini, S. (2009). Random survival forests models for sme credit risk measurement. *Methodology and computing in applied probability*, 11(1):29–45.

Fisher, A., Rudin, C., and Dominici, F. (2018). Model class reliance: Variable importance measures for any machine learning model class, from the "rashomon" perspective. *arXiv preprint arXiv:1801.01489*, 68.

Foucault, T. (1999). Order flow composition and trading costs in a dynamic limit order market. *Journal of Financial markets*, 2(2):99–134.

Frey, S. and Sandås, P. (2009). The impact of iceberg orders in limit order books. In *AFA 2009 San Francisco Meetings Paper*.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Hollifield, B., Miller, R. A., and Sandås, P. (2004). Empirical analysis of limit order markets. *The Review of Economic Studies*, 71(4):1027–1063.

Hosmer Jr, D. W., Lemeshow, S., and Sturdivant, R. X. (2013). *Applied logistic regression*, volume 398. John Wiley & Sons.

Koutmos, D. (2020). Market risk and bitcoin returns. *Annals of Operations Research*, 294(1):453–477.

Kruppa, J., Schwarz, A., Arminger, G., and Ziegler, A. (2013). Consumer credit risk: Individual probability estimates using machine learning. *Expert Systems with Applications*, 40(13):5125–5131.

Kuhnert, P. M., Do, K.-A., and McClure, R. (2000). Combining non-parametric models with logistic regression: an application to motor vehicle injury data. *Computational Statistics & Data Analysis*, 34(3):371–386.

Lo, A. W., MacKinlay, A. C., and Zhang, J. (2002). Econometric models of limit-order executions. *Journal of Financial Economics*, 65(1):31–71.

Maglaras, C., Moallemi, C. C., and Wang, M. (2021). A deep learning approach to estimating fill probabilities in a limit order book. *Available at SSRN 3897438*.

Nevmyvaka, Y., Feng, Y., and Kearns, M. (2006). Reinforcement learning for optimized trade execution. In *Proceedings of the 23rd international conference on Machine learning*, pages 673–680.

Omura, K., Tanigawa, Y., and Uno, J. (2000). Execution probability of limit orders on the tokyo stock exchange. *Available at SSRN 252588*.

Philip, R. (2020). Machine learning in a dynamic limit order market. *Available at SSRN 3630018*.

Ranaldo, A. (2004). Order aggressiveness in limit order book markets. *Journal of Financial Markets*, 7(1):53–74.

Segal, M. R. (2004). Machine learning benchmarks and random forest regression.

Stoikov, S. (2018). The micro-price: a high-frequency estimator of future prices. *Quantitative Finance*, 18(12):1959–1966.

Stoltzfus, J. C. (2011). Logistic regression: a brief primer. *Academic Emergency Medicine*, 18(10):1099–1104.

Stosic, D., Stosic, D., Ludermir, T. B., and Stosic, T. (2018). Collective behavior of cryptocurrency price changes. *Physica A: Statistical Mechanics and its Applications*, 507:499–509.

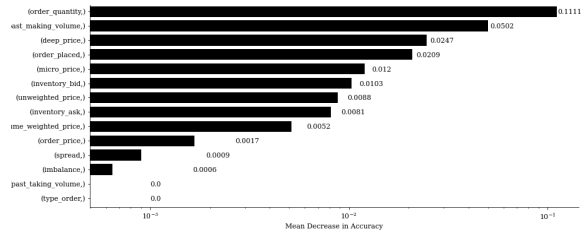
Wallabit Media LLC (2022). Bitcoin Volatility Index.

Wang, J. and Zhang, C. (2006). Dynamic focus strategies for electronic trade execution in limit order markets. In *The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)*, pages 26–26. IEEE.

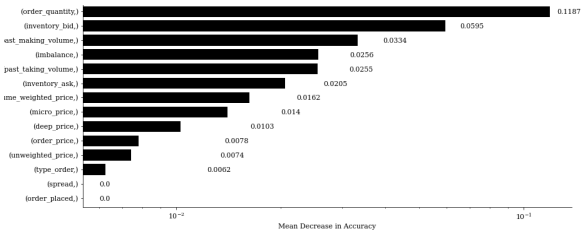
Yingsaeree, C. (2012). *Algorithmic trading: Model of execution probability and order placement strategy*. PhD thesis, UCL (University College London).

A Appendix

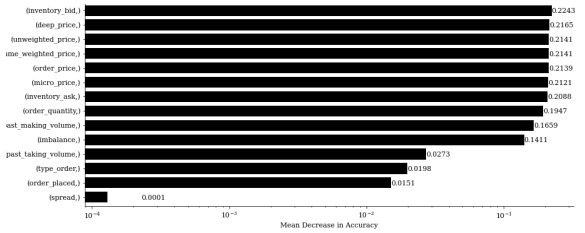
a) Logistic Regression model



b) Random Forest model



c) Kernel Density Estimation model



d) Soft Voting Classifier

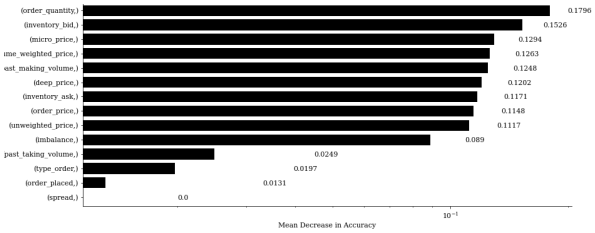


Figure 16. MDA statistics. This figure contains the visualized MDA statistics of the proposed models in this study. Using a log-scaled horizontal axis to plot the mean decrease in accuracy of all the independent variables.

Table 11. Results Evaluation Metrics - Time Windows

Time (sec)	Sample	Size (N)	Measure	Logistic Regression	Random Forest	Kernel Density	Voting Classifier
0.2	**March**	503,059	*AUROC*	0.754226	0.996726	0.934882	0.989730
			Brier Score	0.098631	0.014054	0.055374	0.037959
	April	388,617	*AUROC*	0.793181	0.997794	0.954916	0.994606
			Brier Score	0.055979	0.007425	0.030080	0.020437
	May	636,820	*AUROC*	0.743428	0.996484	0.938097	0.987801
			Brier Score	0.122971	0.018397	0.063798	0.045971
0.5	**March**	503,059	*AUROC*	0.725263	0.995870	0.935628	0.986277
			Brier Score	0.194591	0.023371	0.093472	0.067686
	April	388,617	*AUROC*	0.750509	0.997733	0.949464	0.992251
			Brier Score	0.133562	0.013152	0.062182	0.043756
	May	636,820	*AUROC*	0.711589	0.995574	0.936704	0.985430
			Brier Score	0.213011	0.025722	0.096739	0.072252
2	**March**	503,059	*AUROC*	0.762556	0.996298	0.964657	0.990123
			Brier Score	0.118867	0.013187	0.046352	0.037720
	April	388,617	*AUROC*	0.737925	0.997226	0.957891	0.991429
			Brier Score	0.196297	0.015878	0.076640	0.058676
	May	636,820	*AUROC*	0.764353	0.995457	0.970397	0.991463
			Brier Score	0.060495	0.007483	0.024645	0.019929
5	**March**	503,059	*AUROC*	0.823138	0.995622	0.985238	0.995008
			Brier Score	0.019882	0.001942	0.006946	0.005843
	April	388,617	*AUROC*	0.790459	0.997702	0.977941	0.994575
			Brier Score	0.074181	0.005722	0.027772	0.021868
	May	636,820	*AUROC*	0.784093	0.990410	0.983686	0.990532
			Brier Score	0.003128	0.000273	0.001425	0.001007

This table contains the *AUROC* and *Brier* scores of all the estimated models in this study using different dependent variables with execution time as the adjusted factor. The time column denotes the time in seconds before an order is or is not executed. The size column depicts the training size that is used to fit the model and estimate the model parameters.