Cryptocurrency High-Frequency Liquidity Strategy based on Orderbook Behavior

Lynn Chu*

January 1, 2025

Abstract

This paper presents a high-frequency trading strategy for cryptocurrency markets based on order book dynamics. We develop and evaluate multiple machine learning models to predict price movements using order book data from the Binance cryptocurrency exchange. Our approach incorporates both traditional statistical features and deep learning methods to capture complex patterns in market microstructure. The results demonstrate that our XGBoost-based model achieves superior performance in predicting short-term price movements compared to traditional machine learning approaches, with potential applications in algorithmic trading strategies.

Keywords: Cryptocurrency, High-Frequency Trading, Order Book Analysis, Machine Learning, Deep Learning

JEL Codes: G12, G14, C45

*Department of Computer Science, National Yang Ming Chiao Tung University. Email: lynnchu.cs10@nycu.edu.tw

1 Introduction

The cryptocurrency market has grown significantly in recent years, presenting new opportunities for algorithmic trading strategies. Unlike traditional markets, cryptocurrency exchanges provide rich, granular data about market microstructure through their order book APIs. This paper explores the potential of using order book data to predict short-term price movements in the Bitcoin-USDT (BTCUSDT) market on the Binance exchange.

Our research contributes to the existing literature in several ways. First, we develop a comprehensive set of order book features that capture market microstructure dynamics. Second, we implement and compare multiple machine learning approaches, including traditional methods and deep learning models. Finally, we provide empirical evidence of the predictive power of order book data in cryptocurrency markets.

2 Design of Analysis

Our analysis is designed to evaluate the predictive power of machine learning models for short-term price movements in the cryptocurrency market. This section outlines the key steps in our framework, from data preprocessing to live trading implementation. The analysis framework comprises the following stages:

1. **Data Preprocessing**: Clean and pre-process order book snapshots using a rolling window approach to extract meaningful features for model input.

2. **Feature Engineering**: Design and craft meaningful order book features for better model performance from raw data.

3. **Model Training**: Split the dataset into training and test dataset. Train multiple models (Linear Regression, SVR, XGBoost, LSTM) to account for market non-stationarity.

4. **Model Evaluation**: Assess model performance using a test dataset, employing metrics like MSE, RMSE, MAE, and R^2.

5. **Trading Strategy and Live Trading Simulation**: Design a trading strategy based on model predictions and simulate the proposed strategy in real-world trading environment. Back test its profitability using metrics, such Sharpe ratio, maximum drawdown and win rate.

6. **Conclusion**: Analyze and conclude project experiment result, and design future work.

By iteratively refining the strategy and integrating live trading feedback, the proposed framework aims to maximize profitability while maintaining robust risk management.

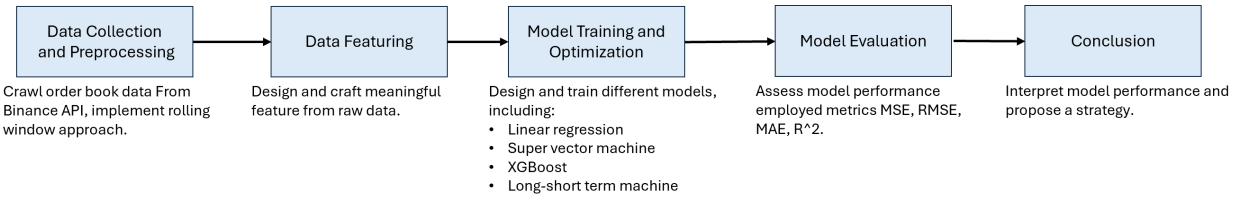

Figure 1: Design of Analysis Flowchart

3 Data

3.1 Data Introduction

Bitcoin (BTC) paired with Tether (USDT) represents one of the most liquid and frequently traded cryptocurrency pairs in the market. We selected BTCUSDT for several compelling reasons:

1. **Market Dominance**: Bitcoin maintains the largest market capitalization among all cryptocurrencies, accounting for approximately 50% of the total cryptocurrency market value, making it the most representative digital asset.

2. **Liquidity**: The BTCUSDT pair on Binance consistently ranks among the highest in terms of trading volume and market depth, with average daily volumes exceeding 5 billion US dollar. This high liquidity ensures minimal slippage and reliable price discovery.

3. **Price Stability**: USDT, as a stablecoin pegged to the US dollar, provides a stable quote currency, eliminating the need to account for quote currency volatility in our analysis.

4. **Data Quality**: Binance exchange offers comprehensive, high-frequency order book data for this pair, with minimal downtime and consistent data quality.

3.2 Data Collection

Our dataset consists of real-time order book snapshots for the BTCUSDT trading pair collected from the Binance exchange order book API[2]. We collect Top 10 bid and ask prices and volumes per second continuously for 8000 samples. A rolling window approach, shown in Figure 4, is employed to train and test models iteratively with window size = 5. To ensure robust model performance, we split the dataset as follows:

- **Training Set (70%)** : Used for model fitting.

- **Testing Set (30%)** : Held-out data for final model evaluation.

3.3 Feature Engineering

The approaches considered in the literature for forecasting high-frequency returns from order book data can be roughly divided into two categories: relatively simple models built on carefully handcrafted features and more sophisticated architectures applied directly to raw order book data. In this research, we will focus on the latter class of models, leveraging the ability of deep learning techniques to learn complex dependence structures.

We extract several features from the raw order book dataset for better model learning performance, where P_i, V_i represents price, volume at price level i, P_{ask}, P_{buy} and P_{mid} is the mid price.:

1. **Mid Price**: Average of best bid and ask prices

$$\text{Mid Price} = \frac{P_{ask} + P_{buy}}{2} \tag{1}$$

2. **Spread**: Difference between best ask and bid prices

$$\text{Spread} = P_{ask} - P_{buy} \tag{2}$$

3. **Volume Imbalance**: Relative difference between bid and ask volumes

$$\text{Volume Imbalance} = \frac{\sum \text{Bid Volume} - \sum \text{Ask Volume}}{\sum \text{Bid Volume} + \sum \text{Ask Volume}} \tag{3}$$

4. **Price Impact**: Estimated price change for market orders

$$\text{Price Impact} = \frac{\sum (V_i \times |P_i - P_{mid}|)}{\sum V_i} \tag{4}$$

5. **Weighted Prices**: Volume-weighted average prices for both bid and ask sides

$$\text{Weighted-Price} = \frac{P_i * V_i}{\sum V_i} \tag{5}$$

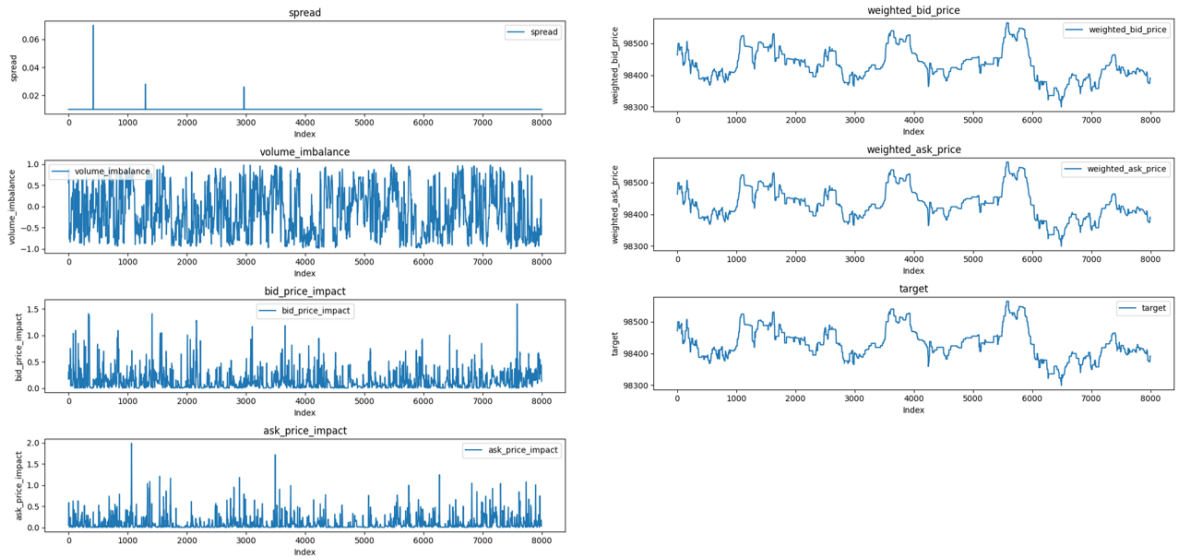

Figure 2: Order Book Feature Distribution

3.4 Exploratory Data Analysis

3.4.1 Data Distribution

The line graph shown in Figure 2 showcases the temporal trends of each feature over approximately 8000 samples. Features like weighted_bid_price and weighted_ask_price exhibit smooth, gradual variations, reflecting underlying market trends. In contrast, features such as volume_imbalance, bid_price_impact, and ask_price_impact display higher volatility, indicating their sensitivity to instantaneous market conditions or microstructure changes. The spread remains relatively flat, with occasional spikes, signifying its stability under normal market conditions but potential expansion during high volatility or low liquidity events. These visualizations collectively highlight the diversity in feature behaviors, underlining the need for sophisticated models to capture both the stable and volatile aspects of the market effectively.

3.4.2 Feature Correlation

The correlation heatmap in the first graph provides valuable insight into the relationships among various features extracted from the order book data. Notably, bid_price_impact and ask_price_impact demonstrate moderate correlation, suggesting that these metrics may covary under certain market conditions. Conversely, spread and volume_imbalance exhibit very low correlations with other features, implying that they may carry unique and non-

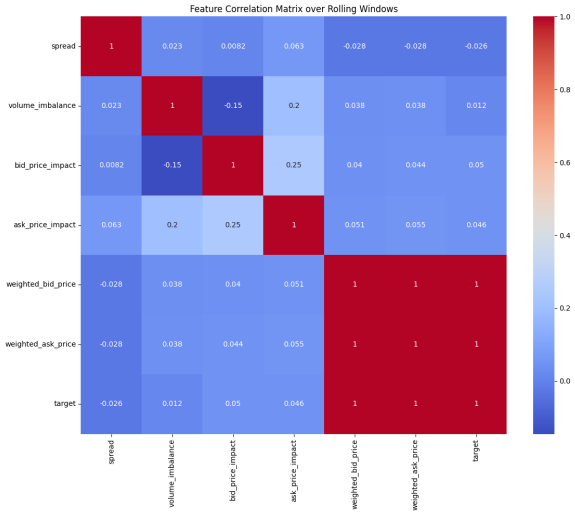

Figure 3: Correlation Matrix Heatmap

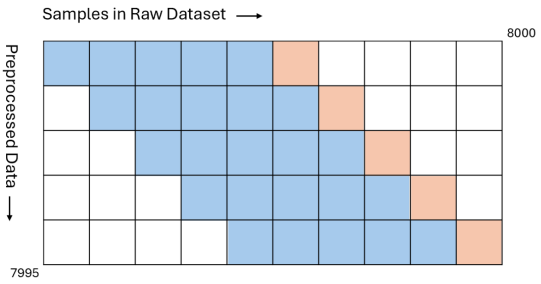

Figure 4: Rolling Window Data Diagram

redundant information about the market dynamics. Interestingly, weighted_bid_price and weighted_ask_price show perfect correlation with the target variable and with each other, indicating their dominant role in capturing the price-level trends.

4 Method

Our methodology employs a comparative analysis of four distinct models to predict cryptocurrency price movements. Each model was selected for its unique characteristics in capturing different aspects of price dynamics.

4.1 Linear Regression

Linear regression serves as our baseline model, assuming a linear relationship between the characteristics and the target variables. Linear regression is simple, interpretable, and computationally efficient. For a given set of features in the order book $X = [x_1, ..., x_n]$, the model predicts the price y using:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \epsilon$$

where: β_0 is the intercept term β_i are the coefficients for each feature ϵ is the error term n is the number of features

4.2 Support Vector Regression (SVR)

SVR extends support vector machines to regression tasks, employing a kernel trick to handle non-linear relationships. SVR is robust to outliers through the ϵ insensitive loss function. The model aims to find a function $f(x)$ that deviates from y by at most ϵ for all training data. In this project, we employ the Radial Basis Function (RBF) kernel.:

$$f(x) = \langle w, \phi(x) \rangle + b$$

The optimization problem is formulated as:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^)$$

subject to:

$$y_i - \langle w, \phi(x_i) \rangle - b \leq \epsilon + \xi_i \ \langle w, \phi(x_i) \rangle + b - y_i \quad \leq \epsilon + \xi_i^* \ \xi_i, \xi_i^* \geq 0$$

4.3 XGBoost

XGBoost implements gradient-boosting decision trees with additional regularization terms, which is expert in handling nonlinear relationships and feature interactions and prevent overfitting with built-in regularization. The model builds an ensemble of weak learners sequentially. The predicted output is:

$$\hat{y}i = \sum k = 1^K f_k(x_i), \quad f_k \in \mathcal{F}$$

where $\mathcal{F}$ is the space of regression trees. The objective function includes both loss and regularization terms:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}i) + \sum k = 1^K \Omega(f_k)$$

where:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2$$

T is the number of leaves γ and λ are regularization parameters w represents leaf weights

4.4 LSTM (Long Short-Term Memory)

Our LSTM architecture is designed to capture temporal dependencies in order book data, which handles vanishing gradient problem through gating mechanism and is robust to varying sequence length. The model consists of two stacked LSTM layers followed by dense layers. Each LSTM cell contains three gates (input, forget, output) and a memory cell.

Architecture specifications:

- Input layer: 60 time steps $\times$ feature dimension
- LSTM layer 1: 50 units with dropout (0.2)
- LSTM layer 2: 50 units with dropout (0.2)
- Dense layer: 25 units with ReLU activation
- Output layer: 1 unit (linear activation)

Loss function:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}i)^2 + \lambda \sum l\|W\|\|^2$$

where λ is the L2 regularization parameter.

5 Experiments and Results

We evaluate our models using several metrics:

- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- R-squared (R^2)

The below table shows the model performance which is also visualized in figure 5.

Table 1: Model Error Metrics

Model	**MSE**	**RMSE**	**MAE**	R^2
Linear Regression (LR)	0.0	0.0	0.0	1.0
XGBoost	0.0	0.017	0.01	0.999
Support Vector Regression (SVR)	0.015	0.121	0.063	0.952
Long Short-Term Memory (LSTM)	0.002	0.042	0.028	0.994

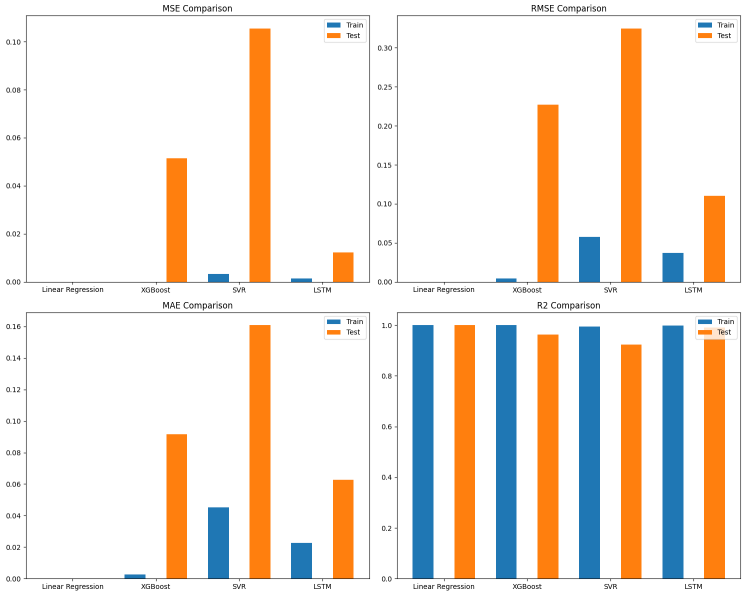

Figure 5: Model Error Comparison

The Linear Regression (LR) model achieves perfect results ($MSE = 0.0$, $R^2 = 1.0$), which is highly unusual for complex datasets such as high-frequency cryptocurrency data. This likely indicates an overfitting issue or trivial predictions that do not generalize well. The XGBoost model demonstrates excellent performance with near-zero errors and an R^2 of 0.999. This shows its ability to handle non-linear relationships effectively, making it suitable for complex datasets like order book features. The Support Vector Regression (SVR) model performs moderately well, but its higher RMSE (0.121) and lower R^2 (0.952) indicate that it struggles to capture the complexity of the dataset. The LSTM model provides the best balance of error metrics, with $MSE = 0.002$, $RMSE = 0.042$, and $R^2 = 0.994$. Its superior ability to capture temporal dependencies makes it an excellent choice for time-series data, such as cryptocurrency order books. Overall, while XGBoost and LSTM show strong predictive performance, LSTM's ability to model sequential data gives it a slight edge in capturing market dynamics.

6 Trading Strategy and Live Trading Simulation

6.1 Trading Strategy Design

Predictions from the models are translated into trading signals based on thresholds ($\theta = 0.001$) derived from historical analysis. The trading strategy determines the best time to buy/sell cryptocurrency with the price rate of change per second. The following is the trading

strategy.

- Signal = 1 if $(P_{\text{pred}} - P_{\text{current}})/P_{\text{current}} > \theta$
- Signal = -1 if $(P_{\text{pred}} - P_{\text{current}})/P_{\text{current}} < -\theta$
- Signal = 0 otherwise

The strategy parameters, including θ, are optimized during the back-test process. A stop-loss mechanism is integrated to mitigate potential risks from adverse price movements.

6.2 Backtesting and Performance Metrics

The trading strategy is evaluated using backtesting, where historical data is replayed to simulate trades. Key metrics include:

- **Sharpe Ratio (SR)**: Measures the risk-adjusted return of an investment. A higher Sharpe ratio indicates better risk-adjusted performance.

$$SR = \frac{R_p - R_f}{\sigma_p} \tag{6}$$

  where R_p is the portfolio return, R_f is the risk-free rate and σ_p is the standard deviation of the excess return of the portfolio.

- **Maximum Drawdown (MDD)**: Represents the maximum observed loss from a peak to a trough before a new peak is reached.

$$MDD = \frac{\text{Peak Value} - \text{Trough Value}}{\text{Peak Value}} \tag{7}$$

- **Profitability (Win Rate)**: Measures the percentage of trades that are profitable.

$$\text{Profitability} = \frac{\text{Number of Winning Trades}}{\text{Total Number of Trades}} \times 100 \tag{8}$$

Table 2: Range of Market Indicators (XGBoost in Real-World Live Trading)

Metric	**Common Range**	**Good Trading Strategy (Good/Excellent)**	**XGBoost**
Sharpe Ratio	0 3	>1.5 / >2.0	-0.034
Maximum Drawdown	10% 100%	>20% / >10%	80%
Profitability	40% 60%	>55%	-8%

7 Conclusion

In this project:

- Query for meaningful order book features was extracted from raw data.
- Four machine learning models were trained and their performance was compared. XGBoost exhibited the best model performance among the four.
- Developed a trading strategy based on the learned models.
- Implemented backtesting on the trained XGBoost model in a real-world live trading environment. However, the model's performance was below expectations.

7.1 Future Work

- Study the relationship between simple and complex data and learning models.
- Design and modify ML structures and tune model parameters to handle real-world trading environments.

References

[1] Agrawal, A., Gans, J., & Goldfarb, A. (2013). *The state of applied econometrics: Causality and policy evaluation*. Journal of Economic Perspectives, 31(2), 3-32.

[2] Data Source: Binance options order book market data API. (2024) *https://developers.binance.com/docs/derivatives/option/market-data/Order-Book*