International Conference on Culture, Economy and Social Sciences (ICCESS 2026)

High-Frequency Volatility Prediction for Bitcoin

Junyu Wei
International Business School Suzhou at XJTLU, Suzhou, China

Abstract:Cryptocurrency markets operate continuously and generate high-frequency order book data that provides detailed information about market liquidity, depth, and order flow. However, the high dimensionality and volatility of limit order book data make short-term price prediction challenging. This study investigates the prediction of future mid-price movements for BTC/USDT using minute-level limit order book data from Binance. The dataset contains 741,600 order book snapshots collected from January 2022 to May 2023, with five price levels on both the bid and ask sides. We construct key order book features, including mid-price, bid-ask spread, order imbalance, and market depth, and develop classification models to predict price direction across multiple forecasting horizons. The study provides a baseline framework for high-frequency cryptocurrency price prediction and establishes a feature engineering pipeline that can be further extended using advanced deep learning architectures, including Transformer and DeepLOB models.

Keywords: Cryptocurrency; Bitcoin; Limit Order Book; High-Frequency Trading; Price Prediction; Order Imbalance; Deep Learning

1. Introduction

Cryptocurrency markets operate arounsd the clock and attract both retail and institutional traders. Unlike traditional equity markets, crypto exchanges do not close overnight or on weekends. This creates a continuous stream of order book data that reflects supply and demand in real time. The limit order book (LOB) records all outstanding buy and sell orders at different price levels. It provides detailed information about liquidity, depth and order flow (Gould et al., 2013).

Short-term price prediction in crypto markets is difficult. Prices are volatile and the data is noisy. The relationship between order book states and future price changes is not stable. It varies across exchanges, time periods and market conditions (Kolm, Turiel and Westray, 2023). Traditional statistical models struggle with the high dimensionality of LOB data. They also fail to capture the complex interactions between price levels.

Deep learning offers a promising approach. Models such as DeepLOB combine convolutional and recurrent layers to learn spatial and temporal patterns from raw order book inputs (Zhang, Zohren and Roberts, 2019). More recently, transformer models have gained attention because self-attention can handle long-range dependencies without sequential processing (Vaswani et al., 2017). However, most transformer studies in crypto use daily or hourly data. Few apply transformers directly to high-frequency LOB snapshots.

This paper investigates high-frequency volatility prediction for BTC/USDT using LOB features. We use minute-level order book snapshots from Binance, covering January 2022 to May 2023. We extract features such as mid-price, spread, order imbalance and depth from 5 levels of the book. We then build classification models to predict the direction of future mid-price movement at multiple horizons. The analysis provides a baseline for LOB-based prediction in crypto markets. It also establishes the feature engineering pipeline for more advanced models such as transformers and DeepLOB.

2. Data and Variables

2.1 Data Source

The dataset consists of 741,600 minute-level order book snapshots for the BTC/USDT trading pair on Binance. It covers the period from 1 January 2022 to 30 May 2023. Each snapshot records the price and volume at 5 levels on both the bid and ask sides. The data includes 24 raw columns in total.

The mid-price ranges from approximately 15,500 to 48,200 USDT. The mean spread is 0.55 USDT, which corresponds to about 0.25 basis points. Order imbalance has a mean near

zero. This indicates that bid and ask volumes are roughly balanced on average. The standard deviation of log returns is 0.084%, which reflects high-frequency noise in minute-level data.

2.2 Feature Variables

We derive several features from the raw order book data. Table 2 lists all variables used in the analysis.

Table 1. Descriptive Statistics of Key Variables

Variable	Count	Mean	Std	Min	Median	Max
mid_price	741,377	27,269.10	8,928.52	15,512.93	23,881.99	48,164.33
spread (USDT)	741,377	0.550	0.893	0.010	0.100	76.360
spread (bps)	741,377	0.254	0.374	0.002	0.047	34.351
order_imbalance	741,377	0.004	0.555	-1.000	0.006	1.000
total_bid_vol	741,377	2.437	10.129	0.002	0.711	2,041.40
total_ask_vol	741,377	2.429	7.309	0.002	0.706	1,802.57
log_return	741,370	0.000	0.001	-0.049	0.000	0.045

Table 2. Feature Variable Definitions

Variable	Definition	Source
ask_i, bid_i	Limit order price at level i (i = 1 to 5)	LOB snapshot
ask_vol_i, bid_vol_i	Quoted volume at level i	LOB snapshot
mid_price	(best_ask + best_bid) / 2	Derived
spread	best_ask - best_bid	Derived
spread_bps	spread / mid_price × 10000	Derived
order_imbalance	(total_bid_vol - total_ask_vol) / (total_bid_vol + total_ask_vol)	Derived
weighted_mid	Volume-weighted mid-price using best level	Derived
bid_depth, ask_depth	Total notional depth across 5 levels	Derived
log_return	log(mid_t / mid_{t-1})	Derived
volatility_k	Rolling standard deviation of log returns (k = 10, 60)	Derived

Order imbalance measures the relative difference between total bid and ask volumes across 5 levels. A positive value suggests more buying pressure. A negative value suggests more selling pressure. Spread measured in basis points removes the effect of price level changes over time. Rolling volatility captures recent price variation at two different window lengths.

2.3 Feature Engineering Code

The following Python code shows how features are computed from the raw LOB data.

Code 1: Feature extraction from order book snapshots

```
import pandas as pd
import numpy as np
df = pd.read_csv('BTCUSDT_book_snapshot_5.csv')
df['timestamp'] = pd.to_datetime(df['timestamp'])

# Mid-price and spread
df['mid_price'] = (df['asks[0].price'] + df['bids[0].price']) / 2
df['spread'] = df['asks[0].price'] - df['bids[0].price']
df['spread_bps'] = df['spread'] / df['mid_price'] * 10000
 # Order imbalance across 5 levels
df['total_bid_vol'] = sum(df[f'bids[{i}].amount'] for i in range(5))
df['total_ask_vol'] = sum(df[f'asks[{i}].amount'] for i in range(5))
df['order_imbalance'] = (
   (df['total_bid_vol'] - df['total_ask_vol'])
   / (df['total_bid_vol'] + df['total_ask_vol'])
)# Log return and volatility
df['log_return'] = np.log(df['mid_price'] / df['mid_price'].shift(1))
df['volatility_60'] = df['log_return'].rolling(60).std()
```

3. Exploratory Analysis

3.1 Mid-Price Dynamics

Figure 1 shows the BTC/USDT mid-price during the first 24 hours of the sample. The price moves between 46,200 and 47,800 USDT. It displays typical intraday fluctuations with short-term trends and reversals.

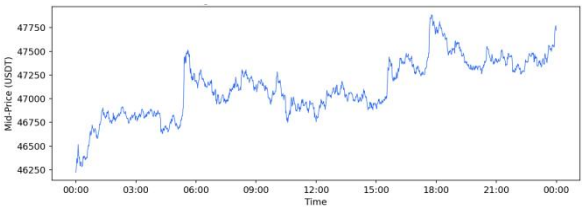

Figure 1. BTC/USDT Mid-Price During the First 24 Hours (1-Minute Resolution).

3.2 Spread Analysis

Figure 2 shows the distribution of bid-ask spreads. Most observations have very tight spreads below 1 USDT. In basis point terms, the median spread is about 0.047 bps. This is consistent with BTC/USDT being one of the most liquid pairs on Binance. However, the distribution has a long right tail. Some snapshots show spreads exceeding 10 USDT during periods of low liquidity or rapid price movement.

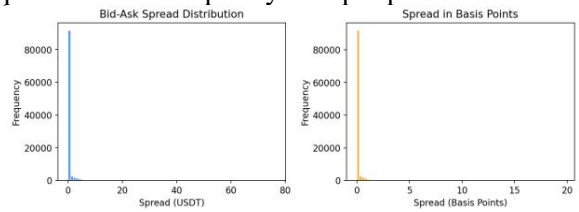

Figure 2. Distribution of Bid-Ask Spread in USDT (left) and Basis Points (Right).

3.3 Order Imbalance and Future Returns

Figure 3 plots order imbalance against the 10-step future return. There is a weak positive relationship. Positive imbalance tends to be followed by positive returns. Negative imbalance tends to be followed by negative returns. However, the scatter is very wide. This suggests that order imbalance alone is not a strong predictor of short-term price direction.

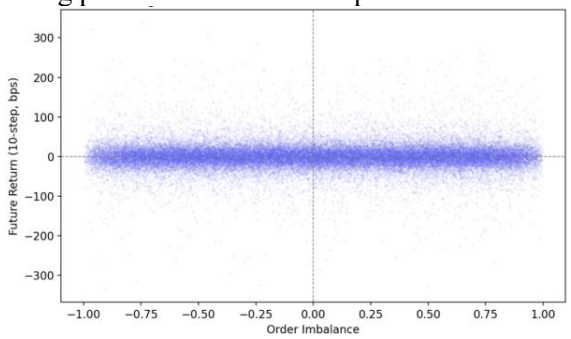

Figure 3. Scatter Plot of Order Imbalance vs. 10-Step Future Mid-Price Return.

3.4 Order Book Depth

Figure 4 shows a heatmap of order book depth across 5 bid and 5 ask levels during the first 60 minutes. The colour intensity represents the volume at each level. Depth varies substantially across time and across levels. Some snapshots have concentrated volume at the best level, while others show deeper liquidity at outer levels.

3.5 Intraday Volatility

Figure 5 displays the 60-step rolling volatility for the first week of the sample. Volatility is not constant. It shows clear clusters where high-volatility periods are followed by more high-volatility periods. This pattern is consistent with well-known volatility clustering in financial markets (Cont, 2001). It also suggests that volatility features may carry predictive information for future price movements.

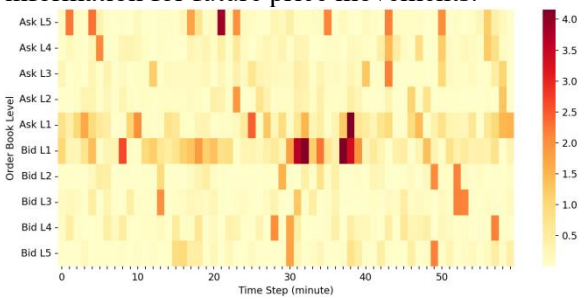

Figure 4. Order Book Depth Heatmap (First 60 Minutes).

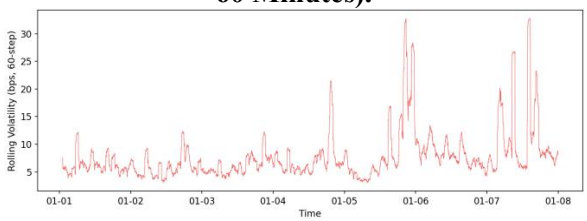

Figure 5. Rolling Volatility (60-Minute Window) during the First Week.

4. Prediction Framework

4.1 Label Construction

We construct directional labels for three prediction horizons: k = 5, 10 and 20 steps. At each time step t, the future return is computed as (mid_{t+k} - mid_t) / mid_t. We then apply a threshold equal to 0.3 times the standard deviation of the future return. If the return exceeds the threshold, the label is +1 (up). If it falls below the negative threshold, the label is -1 (down). Otherwise, the label is 0 (stationary).

Code 2: Label construction for directional prediction

```
# Generate directional labels
for h in [5, 10, 20]:
    df[f'future_return_{h}'] = (
        df['mid_price'].shift(-h) - df['mid_price']
    ) / df['mid_price']
    threshold = df[f'future_return_{h}'].std() * 0.3
    df[f'label_{h}'] = 0
    df.loc[df[f'future_return_{h}'] > threshold,
f'label_{h}'] = 1
    df.loc[df[f'future_return_{h}'] < -threshold,
f'label_{h}'] = -1
```

Table 3 shows the label distribution for each horizon.

Table 3. Label Distribution by Prediction Horizon

Horizon	Down (-1)	Stationary (0)	Up (+1)
k = 5	209,294 (28.25%)	322,491 (43.53%)	209,142 (28.23%)
k = 10	208,536 (28.15%)	322,505 (43.53%)	209,886 (28.33%)
k = 20	207,955 (28.07%)	323,473 (43.66%)	209,499 (28.28%)

The stationary class accounts for about 43% of observations at all horizons. The up and down classes are roughly balanced at about 28% each. This three-class setup is standard in LOB prediction studies (Zhang, Zohren and Roberts, 2019).

4.2 Baseline Model: Random Forest

We use a Random Forest classifier as the baseline model. The input features include spread, order imbalance, depth, volatility and log return. The data is split chronologically. The first 70% is used for training and the remaining 30% for testing. Features are standardised using the training set statistics.

Code 3: Random Forest baseline model

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report

features = ['spread', 'spread_bps', 'order_imbalance',
            'total_bid_vol', 'total_ask_vol',
            'bid_depth', 'ask_depth',
            'volatility_10', 'volatility_60', 'log_return']

X = df_clean[features].values
y = df_clean['label_10'].values

split = int(len(X) * 0.7)
X_train, X_test = X[:split], X[split:]
y_train, y_test = y[:split], y[split:]

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

rf = RandomForestClassifier(
    n_estimators=100, max_depth=10, random_state=42
)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)
print(classification_report(y_test, y_pred))
```

5. Results

5.1 Classification Performance

Table 4 reports the classification results for the Random Forest baseline at each prediction horizon.

Table 4. Random Forest Classification Results

Horizon	Accuracy	Precision	Recall	F1 (macro)
k = 5	0.4442	0.4271	0.4329	0.4236
k = 10	0.4422	0.4282	0.4336	0.4216
k = 20	0.4376	0.4242	0.4325	0.4011

The accuracy ranges from 43.8% to 44.4% across the three horizons. For a three-class problem with a near-uniform prior, random guessing would yield about 33% accuracy. The Random Forest achieves a noticeable improvement over this baseline. However, the F1 scores remain below 0.45, which indicates that the model has difficulty separating the three classes reliably.

Performance decreases slightly as the horizon increases from k = 5 to k = 20. This is expected. Longer horizons introduce more noise and reduce the signal-to-noise ratio. The macro F1 at k = 20 drops to 0.40, which suggests that simple features lose predictive power over longer intervals.

5.2 Confusion Matrices

Figure 6 shows the confusion matrices for all three horizons. The stationary class is predicted most frequently. This is partly because it is the largest class. The model shows a tendency to confuse the up and down classes with the stationary class.

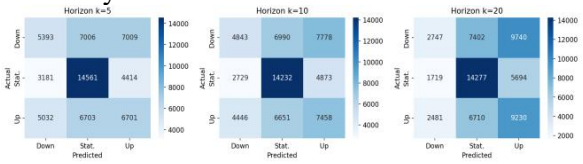

Figure 6. Confusion Matrices for the Random Forest Baseline at K = 5, 10, 20.

5.3 Feature Importance

Figure 7 shows the feature importance scores

from the Random Forest model at k = 10. Log return is the most important feature. Volatility measures also rank highly. Spread and order imbalance contribute less but are still informative. The dominance of return and volatility features suggests that recent price momentum and variation carry the strongest signal for short-term direction prediction.

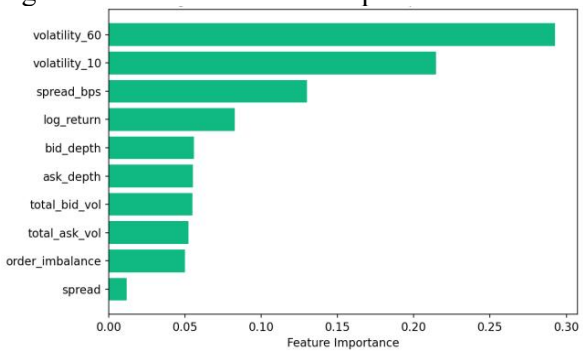

Figure 7. Feature Importance Scores from the Random Forest Model (k = 10).

6. Discussion

The results show that LOB features can predict short-term BTC/USDT price direction above the random baseline. However, the improvement is modest. This is consistent with findings from other studies on LOB-based prediction in cryptocurrency markets. Kolm, Turiel and Westray (2023) show that stationary order-flow representations can improve prediction stability. Bilokon and Qiu (2023) compare LSTM and Transformer architectures and find that both struggle with highly noisy high-frequency data.

The Random Forest model serves as a reasonable starting point. It captures nonlinear relationships between features and labels. However, it does not model temporal dependencies. Each observation is treated independently. This is a clear limitation for time-series data where the sequence of order book states matters.

Deep learning models address this limitation. DeepLOB uses convolutional layers to capture cross-level spatial patterns and LSTM layers to model temporal dynamics (Zhang, Zohren and Roberts, 2019). Transformer models use self-attention to learn which parts of the input sequence are most relevant for prediction (Vaswani et al., 2017). These architectures have shown better performance on similar tasks in equity markets. Their application to cryptocurrency LOB data is a natural next step.

One additional challenge is anomaly detection. Letteri (2025) shows that outlier detection in Bitcoin LOB data is important for model robustness. Order book snapshots during flash crashes or extreme events can distort training. Incorporating anomaly labels into the modelling pipeline could improve prediction quality.

7. Conclusion

This paper presents an analysis of high-frequency BTC/USDT limit order book data for short-term price direction prediction. We use 741,600 minute-level order book snapshots from Binance covering January 2022 to May 2023. We extract features including mid-price, spread, order imbalance, depth and volatility from 5 levels of the book. We construct three-class directional labels at horizons of 5, 10 and 20 steps.

A Random Forest baseline achieves 44.4% accuracy at k = 5, which exceeds the random baseline of 33%. Log return and volatility are the most important features. Performance decreases at longer horizons. These results establish a feature engineering pipeline and evaluation framework for future work.

Future research should apply deep learning models such as DeepLOB and Transformer to the same dataset. These models can capture both spatial and temporal patterns that the Random Forest cannot. Multi-exchange data and cross-venue features could also improve prediction accuracy. Anomaly detection and regime-aware training are additional directions worth exploring.

References:

[1]Bilokon, P. and Qiu, Y. (2023) 'Transformers versus LSTMs for electronic trading', arXiv preprint arXiv:2309.11400.

[2]Cont, R. (2001) 'Empirical properties of asset returns: Stylized facts and statistical issues', Quantitative Finance, 1(2), pp. 223-236.

[3]Cont, R., Kukanov, A. and Stoikov, S. (2014) 'The price impact of order book events', Journal of Financial Econometrics, 12(1), pp. 47-88.

[4]Farooq, A., Uddin, M.I., Adnan, M., Alarood, A.A., Alsolami, E. and Habibullah, S. (2024) 'Interpretable multi-horizon time series forecasting of cryptocurrencies by leverage temporal fusion transformer', Heliyon, 10(22), e40142.

[5]Gould, M.D., Porter, M.A., Williams, S., McDonald, M., Fenn, D.J. and Howison, S.D. (2013) 'Limit order books', Quantitative Finance, 13(11), pp. 1709-1742.

[6]Kolm, P.N., Turiel, J. and Westray, N. (2023) 'Deep order flow imbalance: Extracting alpha at multiple horizons from the limit order book', Mathematical Finance, 33(4), pp. 1044-1081.

[7]Letteri, I. (2025) 'A comparative analysis of statistical and machine learning models for outlier detection in Bitcoin limit order books', arXiv preprint arXiv:2507.14960.

[8]Lim, B., Arik, S.O., Loeff, N. and Pfister, T. (2021) 'Temporal Fusion Transformers for interpretable multi-horizon time series forecasting', International Journal of Forecasting, 37(4), pp. 1748-1764.

[9]Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M. and Iosifidis, A. (2017) 'Forecasting stock prices from the limit order book using convolutional neural networks', in Proceedings of the 2017 IEEE 19th Conference on Business Informatics (CBI), pp. 7-12.

[10]Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I. (2017) 'Attention is all you need', in Advances in Neural Information Processing Systems, 30, pp. 5998-6008.

[11]Wallbridge, J. (2020) 'Transformers for limit order books', arXiv preprint arXiv:2003.00130.

[12]Zhang, Z., Zohren, S. and Roberts, S. (2019) 'DeepLOB: Deep convolutional neural networks for limit order books', IEEE Transactions on Signal Processing, 67(11), pp. 3001-3012.