Graphical Abstract

BTC Trading Signal Generation: Integrating Market Microstructure and Technical Indicators

BO-LING TSENG, YUN-TSE LAN, AYE MYA THANDAR, HUEI WEN TENG

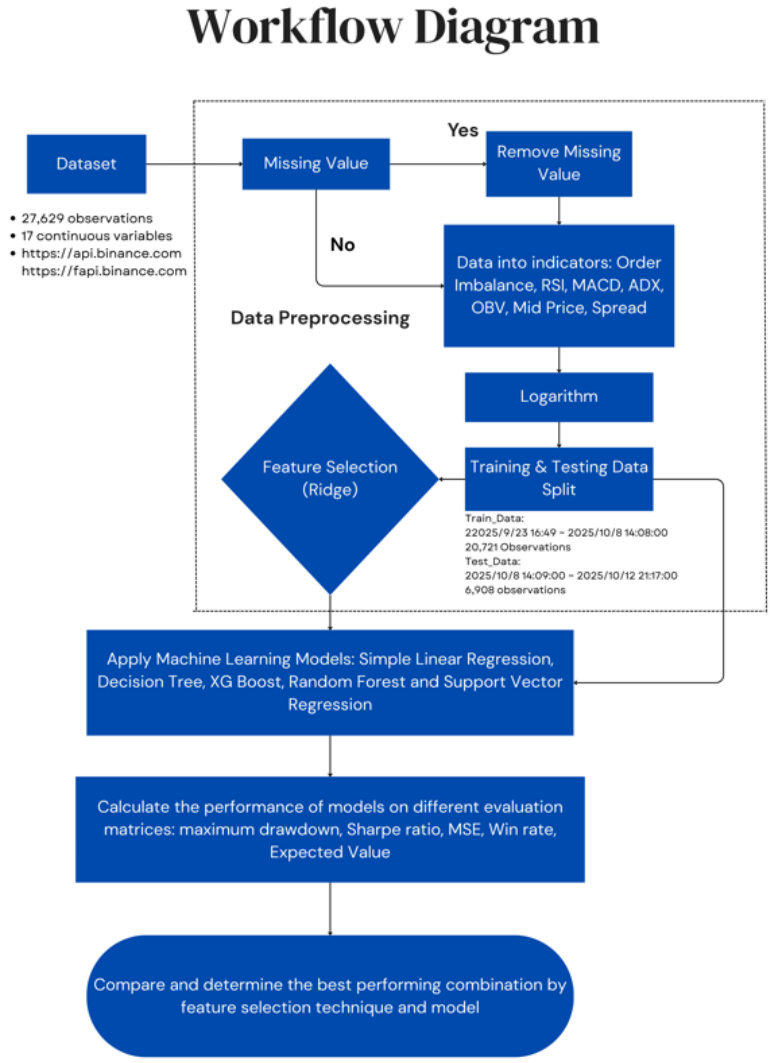

Figure 1. Research study plan

Highlights

BTC Trading Signal Generation: Integrating Market Microstructure and Technical Indicators

BO-LING TSENG, YUN-TSE LAN, AYE MYA THANDAR, HUEI WEN TENG

- Integration of market microstructure and technical indicators improves short-term Bitcoin price forecasting
- A transaction-cost-aware strategy with ATR-based dynamic risk management enhances trading performance
- XGBoost model achieves a superior Sharpe ratio of 2.42 during market volatility, outperforming linear benchmarks
- Ridge regression feature selection benefits SVR models while deteriorating the performance of tree-based algorithms
- High-frequency order book imbalance and spread provide critical predictive signals for 5-minute returns

BTC Trading Signal Generation: Integrating Market Microstructure and Technical Indicators

BO-LING TSENG, YUN-TSE LAN, AYE MYA THANDAR, HUEI WEN TENG

National Yang Ming Chiao Tung University, No 1001 Daxue Road East District, Hsinchu City, 300093, Taiwan, R.O.C.

Abstract

This paper investigates the predictive power of market microstructure features in forecasting short-term Bitcoin returns when combined with established technical indicators. While technical analysis is widely used, it often lags in capturing the rapid shifts inherent in the cryptocurrency market. To address this, we introduce high-frequency microstructure predictors—specifically order book imbalance and bid-ask spread—alongside price momentum and technical indicators. We employ four machine learning algorithms—Support Vector Regression (SVR), XGBoost, Random Forest, and Decision Tree—and compare them against a linear regression benchmark. The study further evaluates the impact of feature selection by contrasting models trained on a full feature set versus a subset selected via Ridge regression. To bridge prediction and practical trading, we implement a strategy that generates signals based on a transaction cost threshold and utilizes an Average True Range (ATR) based dynamic mechanism for take-profit and stop-loss execution. Backtesting results during a volatile market crash period demonstrate that machine learning models significantly outperform the linear benchmark. Specifically, the XGBoost model utilizing the full feature set achieved the highest risk-adjusted performance with a Sharpe ratio of 2.42 and the lowest maximum drawdown. The findings suggest that tree-based models benefit from a comprehensive feature set, while microstructure variables provide critical predictive value for identifying short-term price movements.

Keywords: Trading Strategy, Technical Indicators, Financial Forecasting, Machine Learning, Volatility Modeling
JEL: C45, C53, G11, G17

1. Introduction

Over the past decade, the cryptocurrency market has expanded rapidly and attracted growing attention from both researchers and market participants. Among digital assets, Bitcoin remains the most widely traded and most closely observed. Its high volatility and deep liquidity create opportunities for short-term trading, but they also introduce substantial uncertainty. These challenges have motivated increasing interest in developing more reliable forecasting and trading strategies.

Traditional regression-based approaches have been widely used in financial prediction due to their simplicity and interpretability. However, their linear assumptions often limit their ability to model the nonlinear and rapidly changing dynamics that characterize cryptocurrency markets. As a result, recent studies have turned toward machine learning (ML) methods that better capture complex patterns in noisy financial data. For example, Chen et al. (2020) showed that predictive performance can depend strongly on sampling design, with logistic regression performing well at daily frequencies and more flexible models performing better in high-frequency settings. Similarly, Cocco et al. (2021) demonstrated that multi-stage ML frameworks can enhance forecasting accuracy compared to single-model approaches.

Beyond model choice, feature diversity has also proven important. Studies such as Dubey and Enke (2021) and Guo et al. (2021) highlight the value of on-chain and blockchain transaction features, which offer insights into underlying network activity beyond traditional market indicators. Broader reviews like Boozary et al. (2023) emphasize that combining market, technical, on-chain, and sentiment features can strengthen ML forecasting frameworks. Other lines of work have focused on volatility prediction (Huang et al., 2022), sentiment-driven forecasts (Raju and Tarif, 2023), hybrid ML architectures (Nagula and Alexakis, 2023), and general ML-based approaches to price prediction (Dimitriadou and Gregoriou, 2023). Short-term prediction studies such as Jaquart et al. (2021) further show that ML models can perform well even during periods of fast market movement.

While these studies provide meaningful insights into prediction accuracy, fewer papers examine how such predictions translate into actual trading performance with risk considerations. Many works evaluate models using statistical metrics alone, without incorporating practical portfolio constraints such as drawdown limits, volatility targeting, or adaptive position sizing.

To address this gap, the present study proposes a short-term Bitcoin

trading strategy that integrates machine learning models with an adaptive, risk-adjusted portfolio design. Building on insights from prior ML forecasting research (Chen et al., 2020), (Cocco et al., 2021), (Jaquart et al., 2021), (Dubey and Enke, 2021) and (Guo et al., 2021), the study assesses whether modern predictive techniques can outperform traditional regression benchmarks and whether dynamic risk management can further enhance trading stability and overall performance. The goal is to contribute both to the academic understanding of Bitcoin predictability and to the practical development of data-driven trading strategies in highly volatile digital asset markets.

2. Data

2.1. Data collection and source

This study utilizes 1-minute high-frequency data for the BTC/USDT trading pair, collected from the Binance Exchange public API. The data collection covers multi-dimensional market information, including spot market order book depth, K-line data (OHLCV), and futures market open interest.

The dataset is divided into a training set, containing 21,440 observations spanning approximately 14 days (2025-09-23 16:49 to 2025-10-08 14:08), and a testing set, containing 6,189 observations spanning approximately 4 days (2025-10-08 14:09 to 2025-10-12 21:17). This sequential split ensures that the model is trained on historical data and evaluated on future data, preserving the temporal integrity required for financial time series forecasting.

2.2. Variable definitions and structure

The dataset contains 11 raw variables derived from the API, providing a comprehensive view of market activity. The definitions of these variables are presented in Table 1.

Table 1: Variable definitions and data types.

Variable	Type	Description
timestamp	String	Timestamp (YYYY/M/D HH:MM)
bid	List[Float]	Bid price array (10 levels)
bid_qty	List[Float]	Bid quantity array (10 levels)
ask	List[Float]	Ask price array (10 levels)
ask_qty	List[Float]	Ask quantity array (10 levels)
open	Float	K-line open price (USDT)
high	Float	K-line high price (USDT)
low	Float	K-line low price (USDT)
close	Float	K-line close price (USDT)
volume	Float	Trading volume (BTC)
open interest	Float	Open interest quantity

The order book structure records market depth. Bid orders are sorted from high to low price, where 'bid[0]' represents the best bid. Ask orders are sorted from low to high price, where 'ask[0]' represents the best ask.

2.3. Exploratory data analysis

2.3.1. Descriptive statistics

A comparison of descriptive statistics for key variables between the training and testing sets is provided in Table 2 and Table 3.

Table 2: Descriptive statistics for price, volume, and open interest. Note the significant increase in average trading volume in the testing set.

	Mean		Std Dev		Min		Max	
Variable	Training	Testing	Training	Testing	Training	Testing	Training	Testing
close	116,289.90	117,726.50	5,496.49	5,042.63	108,620.08	102,916.58	126,150.00	124,164.35
volume	2.210	4.602	5.374	11.548	0.001	0.007	256.385	236.415
open interest	91,086.78	86,145.70	5,102.75	10,492.80	84,426.43	70,505.59	102,237.40	98,208.10
mid price	116,289.89	117,726.45	5,496.50	5,042.72	108,620.08	102,612.02	126,188.00	124,160.48

Table 3: Descriptive statistics for spread. The testing set exhibits extreme spread outliers (Max 353.80), indicating potential liquidity stress.

	Mean		Std Dev		Min		Max	
Variable	Training	Testing	Training	Testing	Training	Testing	Training	Testing
spread	0.012	0.379	0.101	7.291	0.010	0.010	10.120	353.800

The statistics reveal distinct market regimes. While the training set covers a wider price range with slightly higher price volatility (Std Dev 5,496 vs 5,043), the testing set is characterized by significantly higher trading activity. The average volume in the testing set (4.602 BTC) is more than double that of the training set (2.210 BTC). Furthermore, Table 3 highlights a critical microstructure feature: the testing set contains extreme spread values (max 353.80 USDT), suggesting periods of liquidity shock that are absent in the training data (max 10.12 USDT).

2.3.2. Data quality assessment

A rigorous data quality check was performed. As shown in Table 4, both datasets are 100% complete with zero missing values. Time continuity checks confirmed no gaps exceeding two minutes in either set. Outliers were detected using the Interquartile Range (IQR) method. Notably, volume outliers constitute approximately 10% of both datasets, reflecting the fat-tailed, impulse-like nature of cryptocurrency trading activity rather than data errors.

Table 4: Data quality check summary. Outliers were identified using the IQR method.

Metric	Training Set	Testing Set
Total Records	21,440	6,189
Missing Values	0 (0.00%)	0 (0.00%)
Outliers - Volume	2,198 (10.25%)	613 (9.90%)
Outliers - Spread	29 (0.14%)	49 (0.79%)
Time Gaps ($>$ 2 min)	0 occurrences	0 occurrences

2.3.3. Visualization

Figure 1 displays the price trajectories. The training period (top) captures a prolonged consolidation followed by a rally, whereas the testing period (bottom) exhibits a sharp correction followed by stabilization. Figure 2 corroborates the statistical findings: trading volume in the testing set shows frequent, high-magnitude spikes, coinciding with the observed price drop, while Open Interest in the testing set shows a sharp decline, indicative of position liquidation events. This is the most critical point of our data analysis: The ability of our models to capture this specific downward trend is the decisive factor in determining their performance.

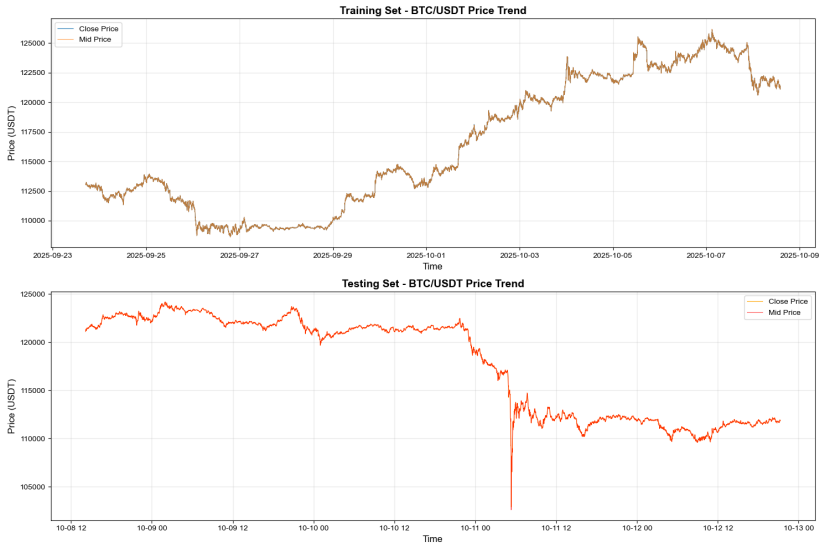

Figure 1: Price trend of BTC/USDT. The training set (top) shows an uptrend, while the testing set (bottom) captures a volatility event and subsequent recovery.

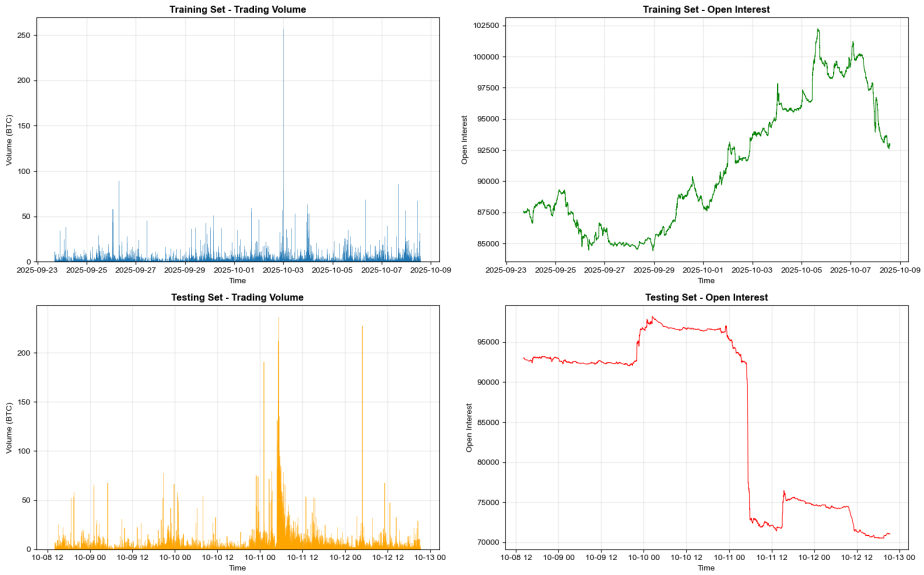

Figure 2: Trading volume and open interest. Note the dense volume spikes in the testing set (bottom left) and the sharp drop in Open Interest (bottom right), suggesting a liquidation event.

3. Feature engineering

3.1. Feature engineering methodology

To prepare the raw data described for machine learning models, we employed a multi-step feature engineering process. A key initial step in this methodology is the temporal aggregation of the 1-minute raw data. To reduce signal noise and align with a more practical trading horizon, the 1-minute K-line data (OHLCV) and order book information are resampled into 5-minute intervals. We directly use the first observation of each 5-minute window, preserving actual market prices. This 5-minute data forms the basis from which the final set of 18 features (1 target, 17 predictors) are derived. These features are designed to capture various aspects of market dynamics, including price momentum, microstructure, and technical sentiment. The primary target variable for prediction, `future_10t_return`, is defined as the 50-minute forward-looking percentage return. This is calculated using the 5-minute mid-price (P_t) at time t and time $t + 10$ (representing 50 minutes), as specified in Equation 1:

$$\overline{R_{t+10}} = (P_{t+10}/P_t) - 1 \tag{1}$$

The predictor variables include eight price momentum features, three market microstructure features, and six technical and market indicators.

3.2. Engineered feature definitions

The final dataset consists of one target variable and 17 predictor features. Their definitions, categories, and data types are summarized in Table 5.

Table 5: Definitions of engineered features. The dataset includes a target variable, price momentum, market microstructure, and technical/market indicators.

Category	Feature	Type	Description
Target	$\overline{R_{t+10}}$	float64	50-min (10 periods) forward return (Eq. 1)
Price Momentum	`delta_price_t`	float64	5-min percentage return of mid-price at t
	`delta_price_t-1`	float64	5-min percentage return of mid-price at t-1
	`delta_price_t-2`	float64	5-min percentage return of mid-price at t-2
	`delta_price_t-3`	float64	5-min percentage return of mid-price at t-3
	`delta_price_t-4`	float64	5-min percentage return of mid-price at t-4
	`delta_price_t-5`	float64	5-min percentage return of mid-price at t-5
	`delta_price_t-6`	float64	5-min percentage return of mid-price at t-6
	`delta_price_t-7`	float64	5-min percentage return of mid-price at t-7
Microstructure	`order_imbalance`	float64	Order book imbalance (Level 1)
	`spread`	float64	Best Ask - Best Bid (USDT)
	`mid_price`	float64	(Best Ask + Best Bid) / 2
Technical/Market	`macd_signal`	float64	MACD Signal Line
	`ADX`	float64	14-period Average Directional Index
	`RSI`	float64	14-period Relative Strength Index
	`open_interest`	float64	5-min aggregated open interest
	`vol`	float64	5-min aggregated trading volume
	`ATR%`	float64	14-period ATR as percentage of mid-price

3.3. Data quality and final structure

The calculation of lagged features and technical indicators (such as RSI, ADX, and MACD, which require lookback periods) introduces necessary missing values (NA) at the beginning of any time series. To create a complete dataset for modeling, these initial observations containing NA values were removed. After this cleaning step, the final engineered dataset is confirmed to have zero missing values and is ready for model implementation.

3.4. Exploratory analysis of engineered features

3.4.1. Descriptive statistics

Table 6 provides the descriptive statistics for all final 18 features based on the training dataset. The target variable `future_10t_return` is centered close to zero (Mean 0.000033). Unlike previous assumptions of a constant spread, the `spread` feature here shows variance (Std 0.101) with a maximum

value of 10.12, capturing liquidity fluctuations. The `ATR%` indicates the relative volatility, with an average of approximately 0.03%.

Table 6: Descriptive statistics of all engineered features (Training Set).

Feature	Mean	Std Dev	Min	25%	50%	75%	Max
`future_10t_return`	0.000 033	0.001 321	−0.009 020	−0.000 641	0.000 000	0.000 656	0.013 852
`delta_price_t`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 184	0.004 377
`delta_price_t-1`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`delta_price_t-2`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`delta_price_t-3`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`delta_price_t-4`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`delta_price_t-5`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`delta_price_t-6`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`delta_price_t-7`	0.000 003	0.000 395	−0.003 901	−0.000 181	0.000 000	0.000 183	0.004 377
`order_imbalance`	−0.0772	0.6419	−0.9999	−0.6897	−0.1141	0.5083	0.9999
`macd_signal`	2.709	52.234	−313.190	−22.610	0.334	25.784	352.203
`spread`	0.0120	0.1009	0.0100	0.0100	0.0100	0.0100	10.1200
`mid_price`	116 291	5499	108 620	111 720	114 301	122 016	126 188
`ADX`	21.49	11.39	3.19	13.06	18.95	27.21	90.58
`RSI`	50.29	18.58	0.00	37.51	50.16	63.08	100.00
`open_interest`	91 090	5105	84 426	86 440	89 249	95 749	102 237
`vol`	2.209	5.377	0.001	0.253	0.868	2.227	256.385
`ATR%`	0.000 311	0.000 197	0.000 000	0.000 183	0.000 270	0.000 391	0.002 094

3.4.2. Correlation analysis

A Pearson correlation analysis was conducted to understand the linear relationships between the engineered features. The results are visualized in the heatmap in Figure 3.

Two main patterns are observed from this analysis. First, consistent with prior financial literature, the linear correlation between the predictor variables and the target variable `future_10t_return` remains weak. All Pearson coefficients with the target fall below 0.1 in magnitude, suggesting that a simple linear model will likely have limited predictive power and validating the need for non-linear machine learning models. Second, a high degree of multicollinearity is visible among the price lag features (`delta_price_t` to `t-7`), which is expected. This multicollinearity suggests that dimensionality reduction techniques or feature selection may be beneficial during the modeling phase.

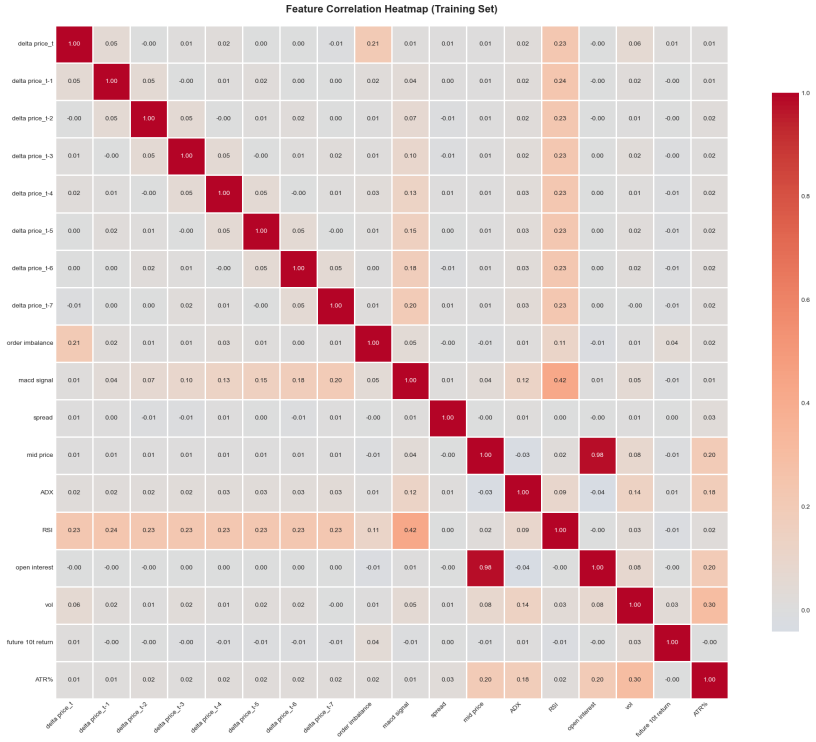

Figure 3: Correlation heatmap of engineered features. High correlation is evident within the price lag blocks. Correlations with the target variable (first row/column) are weak.

3.4.3. Pairplot analysis

To further investigate potential non-linear relationships, a pairplot was generated for all predictors and the target variable, as shown in Figure 4. This visualization includes the target and predictor variables. The plots confirm the findings from the correlation matrix: there are no obvious linear relationships between the predictors and the target. The scatter plots for `future_10t_return` show the variable is concentrated around zero, regardless of the value of the other predictors.

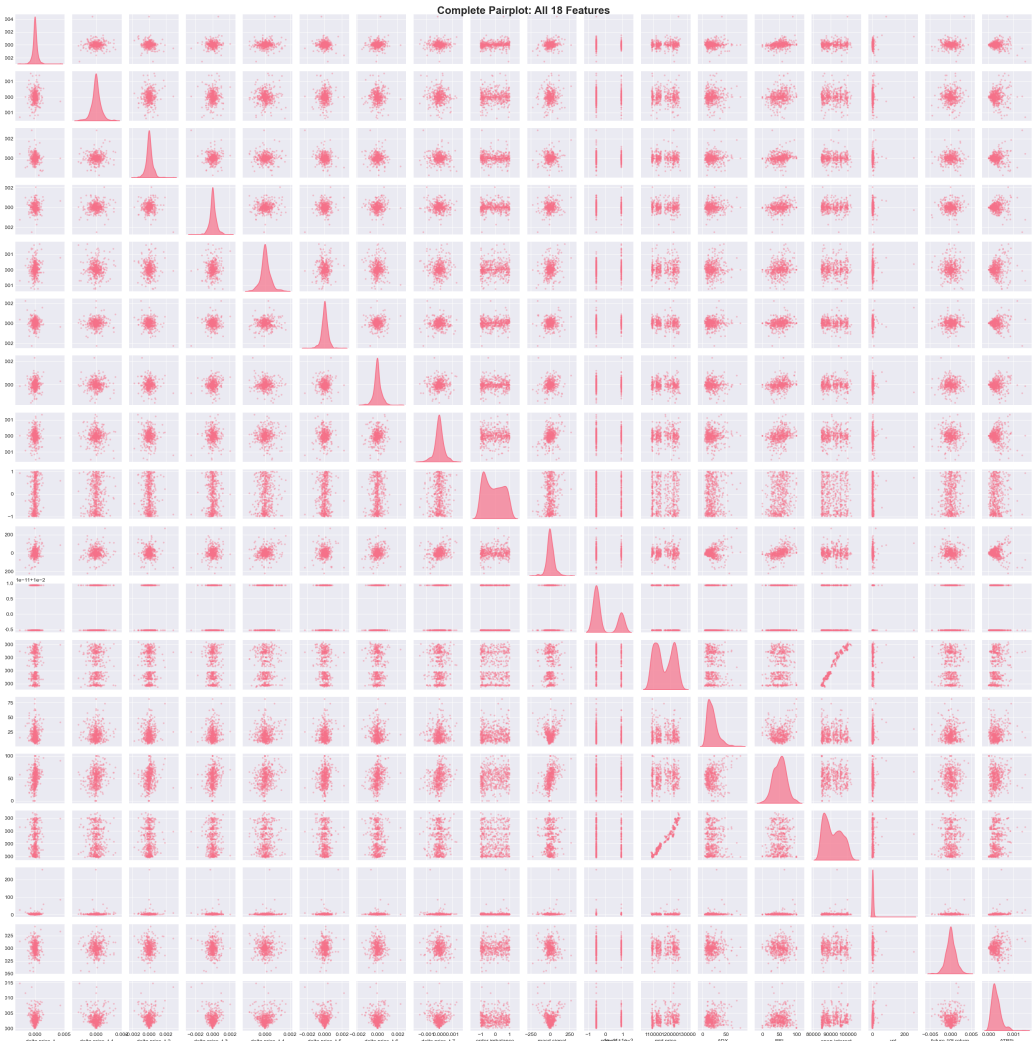

Figure 4: Pairplot of the target variable against predictor variables. The diagonal shows the distribution (KDE) of each variable.

4. Methodology

4.1. Trading strategy

4.1.1. Trading signal

Trading signals are generated based on the model's predicted return $\hat{r}_t$ and the trading fee f. A long position is initiated when the predicted return exceeds the fee:

$$\text{Long Entry:} \quad \hat{r}_t > f.$$

A short position is opened when the predicted return falls below the negative fee:

$$\text{Short Entry:} \quad \hat{r}_t < -f.$$

No trade is executed when $\hat{r}_t \in [-f,\, f]$, ensuring that positions are taken only when the expected return is sufficient to cover transaction costs.

4.1.2. Trading strategy and stop-loss/take-profit design

We apply an ATR stop-loss and take-profit strategy based on the model's predicted return signal (y_{pred}). Positions are opened when the predicted signal exceeds the transaction cost threshold: a long position is established if $y_{pred} >$ fee, and a short position is opened if $y_{pred} < -$fee. Each position remains active until the realized return surpasses either the take-profit threshold ($TP \times ATR_t$) or falls below the stop-loss threshold ($-SL \times ATR_t$), where ATR_t represents the percentage Average True Range, serving as a dynamic measure of market volatility. This allows both thresholds to adapt to changing market conditions.

To determine the optimal parameters, a grid search is conducted over $TP, SL \in [0.5, 3.0]$ with an increment of 0.1. For each pair (TP, SL), a backtest is performed on the training dataset to calculate the annualized Sharpe ratio:

$$SR = \frac{\bar{r} - r_f}{\sigma_r}$$

where $\bar{r}$ denotes the mean trade return, σ_r the standard deviation of returns, and r_f the risk-free rate. The pair (TP^*, SL^*) that maximizes SR is selected as the optimal stop-loss and take-profit combination.

Finally, the optimized parameters (TP^*, SL^*) are applied to the testing dataset to evaluate the model's out-of-sample performance. This procedure ensures that the trading thresholds are determined solely from the training period, avoiding look-ahead bias and providing a fair assessment of predictive stability in unseen market conditions.

4.2. Benchmark model

4.2.1. Model specification

The linear regression included sixteen explanatory variables, covering eight lagged price changes (ΔP_t to ΔP_{t-7}), order book imbalance, and several technical indicators such as $MACD_t$, S_t, M_t, ADX_t, RSI_t, OI_t^{open}, and σ_t.The dependent variable is the short-term log return of Bitcoin, denoted as $\overline{R_{t+10}}$

Linear regression model is specified in Equation 2:

$$\overline{R_{t+10}} = \beta_0 + \sum_{i=0}^{7} \beta_{i+1} P(t-i) + \beta_9 MACD_t + \beta_{10} S_t + \beta_{11} M_t + \beta_{12} ADX_t + \beta_{13} RSI_t + \beta_{14} OI_t^{open} + \beta_{15} \sigma_t + \beta_{16} OI_t + \epsilon_t \quad (2)$$

where
$\overline{R_{t+10}}$: the Bitcoin average return between t=1 and t=10;
ΔP_t: logarithmic change in price between t=0 and t=-1;
ΔP_{t-1}: logarithmic change in price between t=-1 and t=-2;
ΔP_{t-2}: logarithmic change in price between t=-2 and t=-3;
$MACD_t$: moving average convergence divergence signal value;
S_t: logarithmic spread, representing market liquidity;
M_t: mid price, the average of the best bid and ask prices;
ADX_t: logarithmic average directional index, representing trend strength;
RSI_t: relative strength index;
OI_t^{open}: open interest, representing the total number of open contracts;
σ_t: volatility of price;
ϵ_t: error term assumed to be independently and identically distributed with zero mean and constant variance.

4.2.2. Discussion of model results

Statistic	Value
R-squared	0.009
Adj. R-squared	0.005
F-statistic	2.362

Variable	Coef	Std. Err	t-Stat	p-Value	0.025	0.975
Constant	0.0002	4.46e-05	3.884	0.000	8.58e-05	0.000
delta price_t	-1.305e-05	5.47e-05	-0.239	0.811	-0.000	9.42e-05
delta price_t-1	-2.173e-05	5.72e-05	-0.380	0.704	-0.000	9.03e-05
delta price_t-2	3.903e-05	5.88e-05	0.664	0.507	-7.63e-05	0.000
delta price_t-3	3.908e-05	5.91e-05	0.661	0.508	-7.68e-05	0.000
delta price_t-4	0.0001	5.73e-05	1.890	0.059	-4.06e-06	0.000
delta price_t-5	5.806e-05	5.55e-05	1.046	0.295	-5.07e-05	0.000
delta price_t-6	5.124e-05	5.29e-05	0.968	0.333	-5.25e-05	0.000
delta price_t-7	6.135e-05	5.13e-05	1.195	0.232	-3.93e-05	0.000
order imbalance	8.479e-05	4.52e-05	1.877	0.061	-3.75e-06	0.000
macd signal	-5.327e-05	8.51e-05	-0.626	0.532	-0.000	0.000
log spread	0.0001	4.49e-05	3.006	0.003	4.69e-05	0.000
mid price	-0.0007	0.000	-3.091	0.002	-0.001	-0.000
log ADX	4.45e-05	4.75e-05	0.937	0.349	-4.87e-05	0.000
RSI	8.533e-05	7.21e-05	1.184	0.236	-5.59e-05	0.000
open interest	0.0007	0.000	2.846	0.004	0.000	0.001
vol	-2.358e-05	4.81e-05	-0.490	0.624	-0.000	7.08e-05

Table 7: OLS regression summary and coefficient estimates for the 10-step forward return prediction.

4.2.3. Discussion of model results

The regression results indicate that the linear model has extremely limited explanatory power for short-term Bitcoin returns. The R-squared and adjusted R-squared values are close to zero, which mean that linear regression explain nothing, and the F-statistic suggests only weak joint significance among the predictors. Most price-based features and technical indicators are statistically insignificant, reflecting the high noise and rapid fluctuations inherent in five-minute data. In contrast, microstructure variables—particularly the bid–ask spread, mid price, and to a lesser extent order imbalance—show

comparatively stronger significance, highlighting their relevance in capturing short-horizon market dynamics. Overall, the linear specification fails to extract meaningful predictive signals, reinforcing the need for nonlinear models capable of capturing more complex relationships in high-frequency cryptocurrency markets.

4.2.4. Backtest results

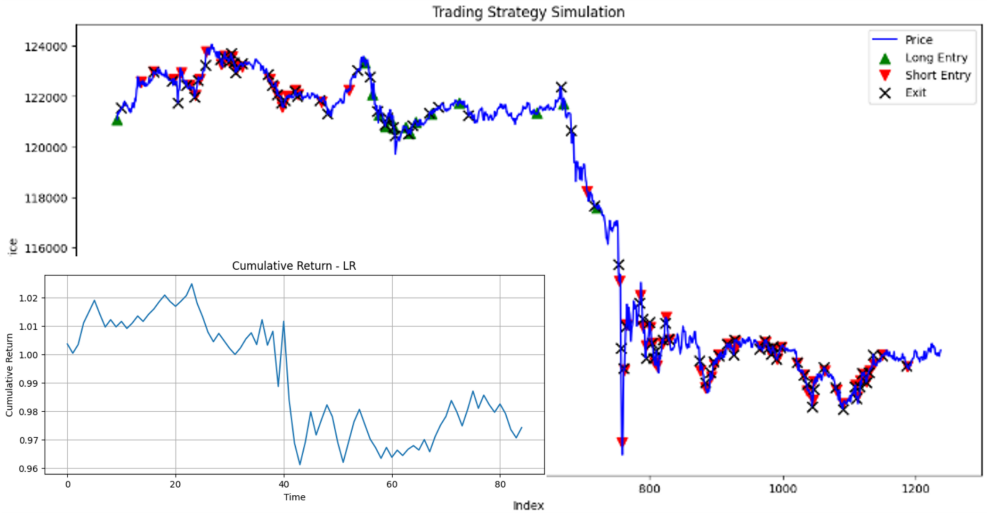

Figure 5: Trading strategy simulation and cumulative return for the linear regression model.

The figure above illustrates the performance of the trading strategy generated by the linear regression model. Most of the predicted signals fail to capture the price direction, especially toward the end, where the model repeatedly issues short signals regardless of market movement.

4.2.5. Performance measurement

Performance Measure	Value
Sharpe ratio	-0.8399537070702341
Max Drawdown	0.06197069932081422
EV per trade	0.000285
Win rate	0.517647
CAGR	-0.9999817

Table 8: Backtest performance results for the linear regression benchmark model.

The backtest results indicate that the model performs poorly during the evaluated period. The negative Sharpe ratio suggests that the strategy generates returns that are unfavorable relative to its risk. Although the maximum drawdown remains relatively small, the overall profitability is weak, as reflected by the very low expected value per trade. The win rate is slightly above 50%, but the returns of winning trades are insufficient to offset losses. This is further confirmed by the strongly negative CAGR, indicating that the strategy loses nearly all of its value over time. Overall, the model fails to deliver a viable trading performance under the given setup.

4.3. Design of research

This study adopts a systematic five-stage framework to evaluate the efficacy of machine learning models in high-frequency Bitcoin trading. As illustrated in Figure 6, the research design progresses from feature selection and model training to signal generation, risk management, and final performance evaluation. The primary contribution of this research lies in the integration of market microstructure features with an adaptive, volatility-adjusted trading strategy. Unlike traditional econometric approaches that rely on linear assumptions, this study employs non-linear machine learning algorithms to capture the complex, impulse-like behaviors of cryptocurrency markets. Furthermore, distinct from studies that focus solely on directional prediction accuracy, this framework bridges the gap between statistical forecasting and practical trading by incorporating transaction cost thresholds and dynamic risk management.

The experimental design incorporates two distinct feature selection strategies to evaluate model robustness and input sensitivity. First, we utilize the full set of 16 engineered features to establish a baseline, allowing the models to access all available market information, including price momentum, technical

indicators, and microstructure data. Second, we employ Ridge regression as a feature selection technique to identify the most significant predictors and reduce potential multicollinearity. By comparing the performance of models trained on the full feature set versus the Ridge-selected subset, we aim to determine the optimal input structure for short-term price prediction. Following this, five distinct regression models are trained on both feature sets: a linear regression benchmark, support vector regression (SVR), extreme gradient boosting (XGBoost), decision tree (DT), and random forest (RF). This holistic design allows for a rigorous assessment of whether machine learning models can generate sustainable alpha in the presence of realistic market friction and liquidity shocks.

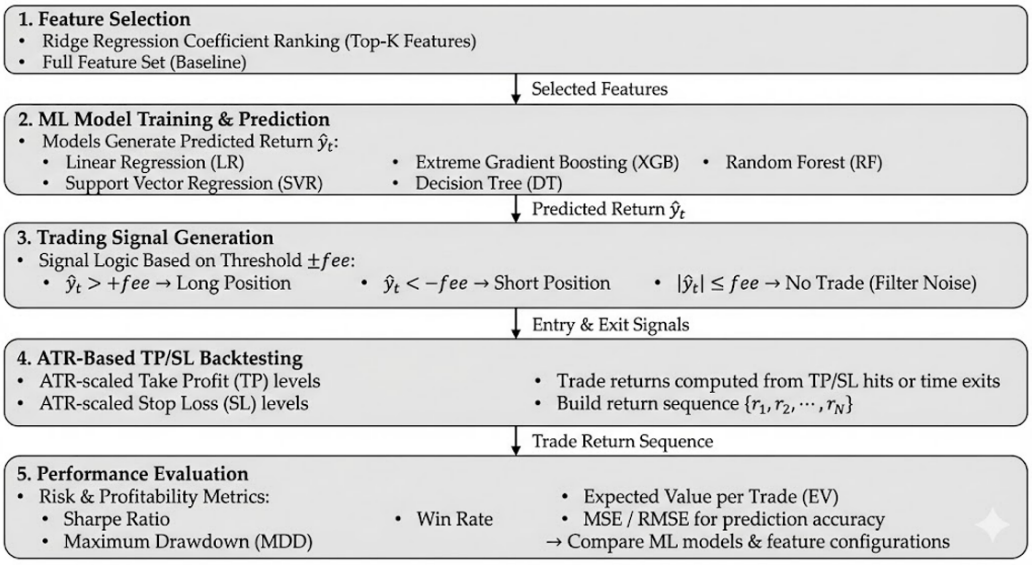

Figure 6: **Research study plan and workflow.** The process consists of five sequential stages: (1) Feature selection comparing the full feature set against Ridge regression rankings; (2) Training of five distinct regression models (LR, SVR, XGB, DT, RF) to predict short-term returns; (3) Generation of trading signals based on a transaction fee threshold to filter noise; (4) Execution of an ATR-based dynamic risk management strategy with optimized take-profit and stop-loss levels; and (5) Comprehensive performance evaluation using both statistical and financial metrics.

4.4. Performance measures

To comprehensively evaluate the effectiveness of the proposed trading framework, we employ two categories of metrics: statistical metrics for assessing the prediction accuracy and financial metrics for evaluating the profitability

and risk of the trading strategy. The primary statistical metric used in this study is the mean squared error (MSE), which quantifies the average squared difference between the predicted returns and the actual returns. As defined in Equation 3, a lower MSE indicates a model with higher predictive precision.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{pred,i} - y_{real,i})^2 \tag{3}$$

For financial performance, we focus on risk-adjusted returns and capital preservation. The Sharpe ratio, denoted in Equation 4, is the primary metric for evaluating risk-adjusted performance. It is calculated as the ratio of the annualized excess return to the annualized volatility, where a higher value indicates superior return per unit of risk.

$$SharpeRatio = \frac{R_p - R_f}{\sigma_p} \tag{4}$$

To assess downside risk, we employ the maximum drawdown (MDD), defined in Equation 5. This metric measures the largest percentage decline from a cumulative return peak to a subsequent trough, representing the worst-case scenario for an investor.

$$MDD = \min_t \left(\frac{V_t - \max_{\tau<t} V_\tau}{\max_{\tau<t} V_\tau} \right) \tag{5}$$

Additionally, the compound annual growth rate (CAGR) is used to measure the geometric progression ratio that provides a constant rate of return over the time period. Equation 6 illustrates this calculation using the ending value V_{end} and starting value V_{start} over T years.

$$CAGR = \left(\frac{V_{end}}{V_{start}} \right)^{\frac{1}{T}} - 1 \tag{6}$$

Finally, to understand the reliability of the signals, we examine the win rate and the expected value (EV) per trade. The win rate is the ratio of profitable trades to the total number of trades. The expected value, shown in Equation 7, combines the probability of winning and losing with their respective average payoffs, providing an estimate of the average profit or loss per trade.

$$EV = P(win) \times \mu(win) + P(loss) \times \mu(loss) \tag{7}$$

5. Results

5.1. Feature selection and prediction accuracy

We utilized Ridge regression to identify the most influential features among the original set. Based on the coefficient magnitude elbow plot, the top eight features were selected for the optimized models. These features include log spread, specific lagged price changes ($\Delta P_{t-4}, \Delta P_{t-7}, \Delta P_{t-5}$), order imbalance, RSI, mid-price, and the MACD signal. Notably, market microstructure features such as spread and order imbalance ranked highly, confirming their importance in short-term price discovery.

In terms of prediction accuracy, the random forest (RF) model achieved the lowest mean squared error of 4.00×10^{-6}, closely followed by the linear regression (LR) benchmark and the decision tree (DT) model. Conversely, the support vector regression (SVR) model exhibited the highest prediction error. Table 9 summarizes the trading performance metrics derived from these predictions.

5.2. Trading strategy performance

The backtesting results on the testing set (2025/10/08 to 2025/10/12) reveal significant performance disparities across models. Table 9 presents a comprehensive comparison of the benchmark OLS model against the machine learning variants, including versions with and without Ridge feature selection.

Table 9: Performance comparison of trading strategies. The table reports the Sharpe ratio, maximum drawdown (MDD), compound annual growth rate (CAGR), expected value (EV) per trade, and win rate for each model. The testing period ranges from 2025/10/08 to 2025/10/12. Note that "Ridge" indicates models using the subset of features selected by Ridge regression.

Model	Sharpe Ratio	MDD	CAGR	EV per Trade	Win Rate
LR (Benchmark)	-0.840	0.062	-1.00	-0.000285	0.518
SVR	-0.376	0.048	-0.98	-0.000109	0.492
SVR (Ridge)	2.018	0.065	5.51×10^{10}	0.000537	**0.588**
XGBoost	**2.422**	**0.037**	$\mathbf{2.87 \times 10^{12}}$	0.000608	0.534
XGBoost (Ridge)	0.343	0.094	46.16	0.000103	0.527
Decision Tree	2.321	0.038	1.73×10^{6}	**0.000816**	0.523
DT (Ridge)	-3.832	0.094	-1.00	-0.001956	0.388
Random Forest	0.795	0.050	187.10	0.000292	0.438
RF (Ridge)	0.015	0.065	-0.43	0.000006	0.475

The benchmark linear regression model failed to generate positive risk-adjusted returns, yielding a negative Sharpe ratio of -0.840. In contrast, the XGBoost model utilizing the full feature set demonstrated the most robust performance, achieving the highest Sharpe ratio of 2.422 and the lowest maximum drawdown of 0.037. This suggests that XGBoost effectively captures non-linear dependencies without overfitting in this high-frequency context. Similarly, the Decision Tree model performed well with a Sharpe ratio of 2.321 and the highest expected value per trade (0.000816).

Interestingly, the impact of feature selection varied by model. While Ridge-based feature selection significantly improved the SVR model (turning a negative Sharpe ratio to 2.018 and achieving the highest win rate of 58.8%), it deteriorated the performance of tree-based models (XGBoost and Decision Tree). This implies that tree-based algorithms may benefit from the redundancy and subtle signals present in the full feature set, whereas SVR requires dimensionality reduction to function effectively. Overall, the machine learning approaches, particularly XGBoost, significantly outperformed the traditional linear benchmark.

5.3. Prediction model comparison

5.3.1. Overall comparison

Figures 7 and 8 compare the model–predicted 10-step-ahead returns (y_{pred}) with the realized returns (y_{real}) for the testing period. Each subplot corresponds to one forecasting model (Linear Regression, SVR, XGBoost, Decision Tree, and Random Forest). Across both settings, the predicted returns remain tightly clustered around zero and generally follow the direction of the realized returns, reflecting the low signal-to-noise ratio in short-horizon return data. However, all models fail to anticipate the large negative return shock in the testing window, which appears as a sharp downward spike in the true series.

Figure 7 presents the baseline specification without feature selection. In this case, SVR, XGBoost, and Random Forest track the small fluctuations in y_{real} more closely than Linear Regression and the single Decision Tree, but none of the models capture the extreme drawdown. Figure 8 reports the results for the Ridge regularized setting. Here, the prediction curves become smoother, especially for the linear models, indicating reduced overfitting. Nonetheless, the regularized models still exhibit the same limitation around the volatility event, confirming that the unexpected shock is not learnable from the available features.

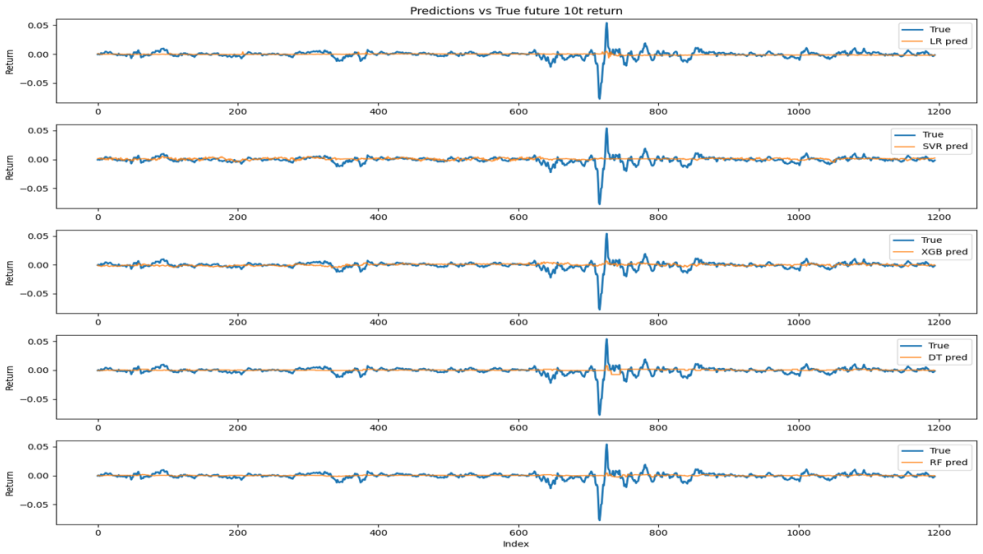

Figure 7: Predicted vs. true 10-step-ahead returns for models without feature selection.

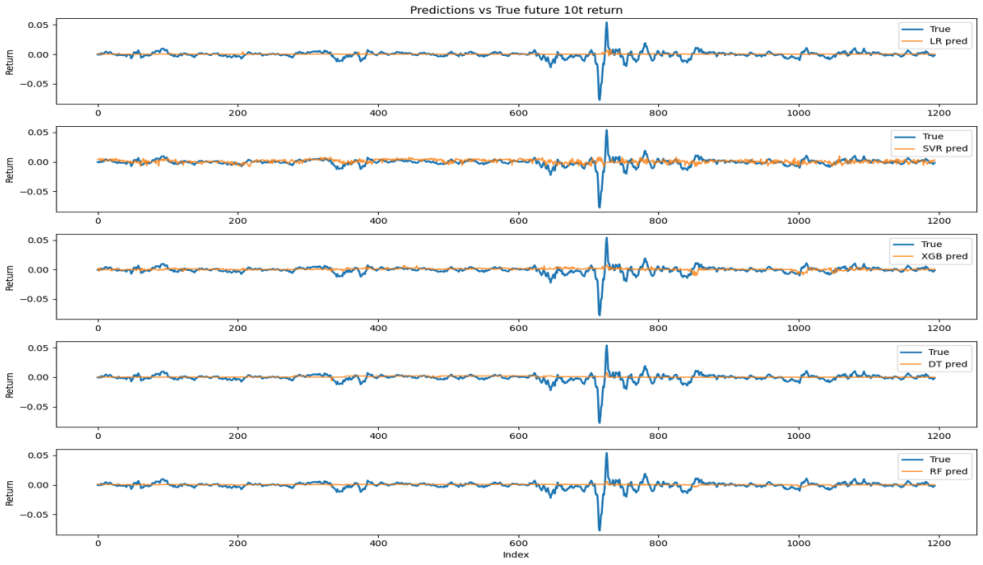

Figure 8: Predicted vs. true 10-step-ahead returns for models with Ridge regularization applied.

5.3.2. SVR and SVR–Ridge trading strategy comparison

Figures 9 and 10 present the trading strategy simulations for the baseline Support Vector Regression (SVR) model and its Ridge-regularized counterpart. Both figures overlay the model's generated long and short entry signals on the underlying price series and display the corresponding cumulative return trajectories.

The baseline SVR model demonstrates highly aggressive behavior, as indicated by the dense clustering of trade entries and exits across the testing horizon. While this sensitivity allows SVR to react quickly to minor fluctuations, it also leads to substantial instability during regime-shift events. In the highlighted downward-trending segment, SVR fails to capture the persistent decline, continuing to generate counter-directional or delayed signals that result in misaligned trades and significant performance deterioration. This behavior illustrates the model's susceptibility to overreacting to noise rather than identifying sustained market movements.

After applying Ridge regularization, the SVR model exhibits a marked improvement. The SVR–Ridge variant produces fewer spurious signals, displays smoother transitions between long and short positions, and aligns more effectively with the underlying price trend outside the volatility shock. Although the regularized model still cannot anticipate the sudden price collapse—a

limitation shared across all tested models—its overall signal discipline and directional consistency improve substantially. Among all forecasting models evaluated, SVR shows the greatest performance enhancement after regularization, and the SVR–Ridge configuration emerges as the most robust, balancing responsiveness with reduced overfitting.

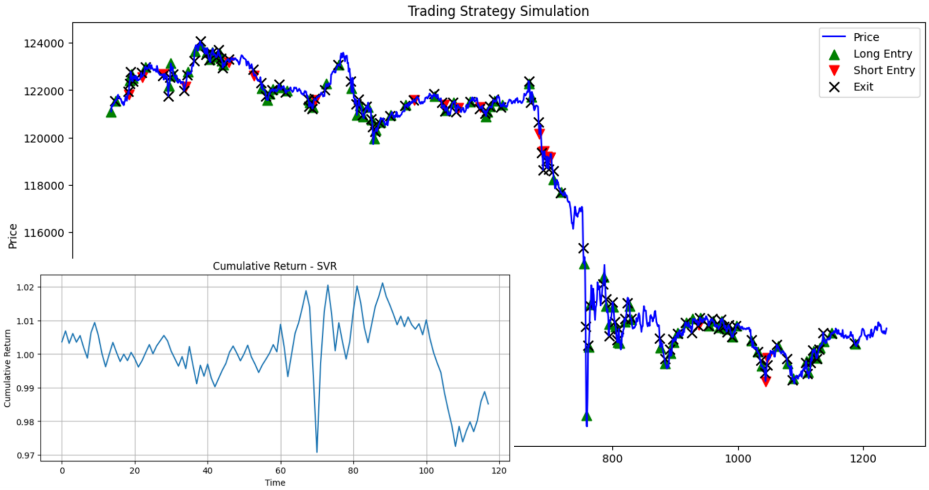

Figure 9: Trading strategy simulation for the baseline Support Vector Regression (SVR) model.
The model exhibits aggressive entry behavior and fails to capture the pronounced downward trend in the highlighted region.

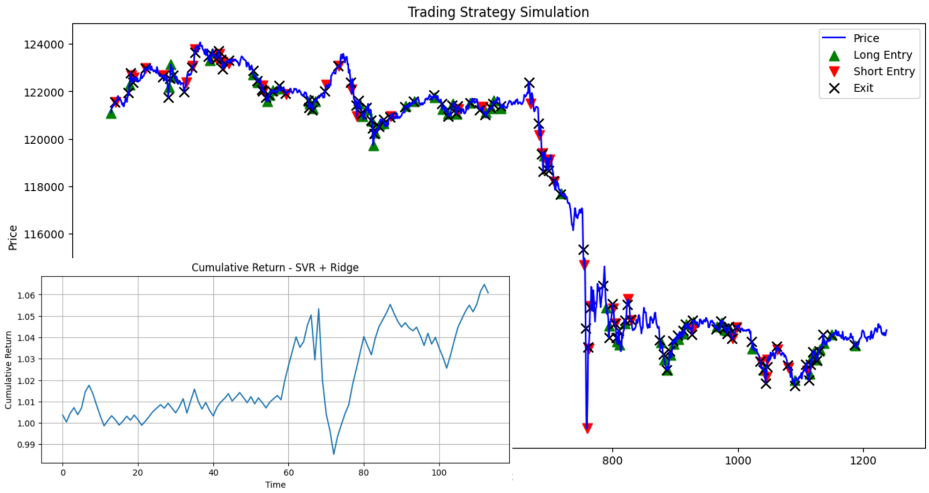

Figure 10: Trading strategy simulation for the Ridge-regularized SVR model.
The regularization reduces noise, improves signal stability, and enhances alignment with long-term trends outside the volatility shock.

5.3.3. XGBoost and XGBoost–Ridge trading strategy comparison

Figures 11 and 12 report the trading strategy simulations for the baseline XGBoost (XGB) model and its Ridge-regularized counterpart. As before, the figures overlay the model's long/short entry and exit signals on the price path and show how the strategy responds to the large drawdown region highlighted in red.

Compared with SVR–Ridge, the baseline XGB model exhibits weaker performance in the early and late segments of the testing period: its trade timing in these regions is noisier, and the resulting decisions do not align as closely with the prevailing trend. However, XGB performs remarkably well around the major volatility event. It not only identifies the onset of the sharp decrease but also closely tracks the downward trajectory and correctly anticipates the reversal point, switching direction at the bottom of the drawdown and making profitable trades after the large decline. This behavior indicates that the boosted tree ensemble is particularly effective at capturing nonlinear dynamics associated with abrupt regime changes.

In contrast, the XGB model with Ridge regularization performs poorly across most of the testing horizon. The regularization dampens the model's ability to exploit the nonlinear structure of the data, leading to delayed or misaligned trading signals both before and after the volatility event. The Ridge-regularized XGB fails to reproduce the accurate detection of the turning

point observed in the baseline specification and generates less consistent decisions throughout. Overall, while the unregularized XGB model shows localized strengths around the crash and subsequent rebound, applying Ridge regression substantially degrades its trading performance.

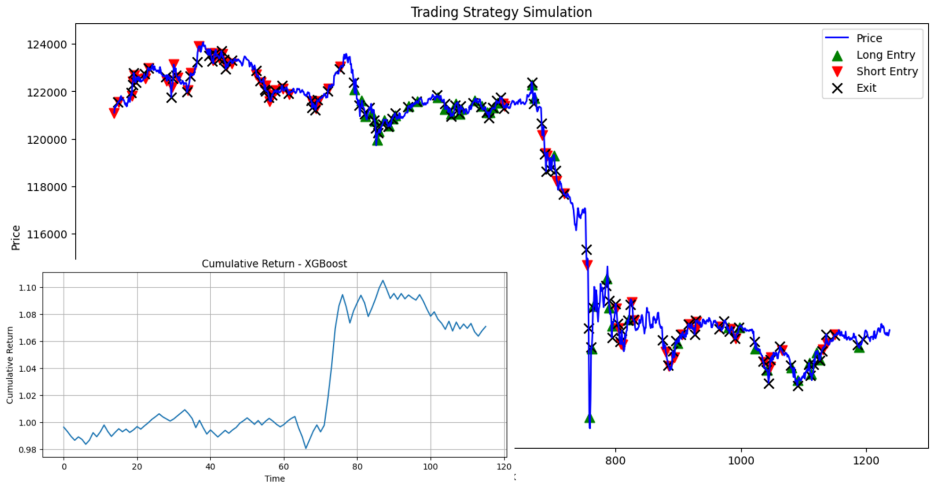

Figure 11: Trading strategy simulation for the baseline XGBoost (XGB) model. Although XGB performs relatively poorly in the early and late parts of the testing period, it successfully captures the sharp decreasing trend and accurately identifies the reversal point after the major drawdown.

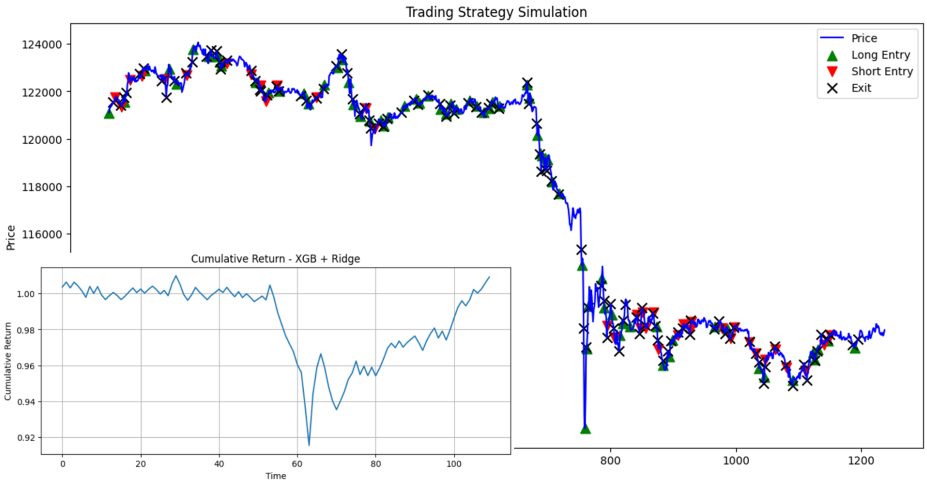

Figure 12: Trading strategy simulation for the Ridge-regularized XGBoost model. After applying Ridge, the model loses its ability to exploit nonlinear patterns, resulting in poorly timed trades and degraded performance across most of the testing horizon.

5.3.4. Decision Tree trading strategy comparison

Figures 13–14 present the trading strategy simulation results for the baseline Decision Tree (DT) model and its Ridge-regularized variant. As in the previous subsections, the figures display the model's long and short entry signals overlaid on the price trajectory and highlight the major drawdown region during the testing period.

The baseline Decision Tree model behaves relatively conservatively, generating fewer trading signals compared with SVR or XGBoost. Similar to XGBoost, DT performs poorly at the beginning and the end of the testing horizon, where its trade timing does not match the prevailing market direction. Nevertheless, the model effectively captures the pronounced decreasing segment and also identifies the subsequent upward recovery, although it produces a small misalignment in the middle of this upward trend. This behavior indicates that while the model is simple and less adaptive, it is still capable of recognizing major structural movements.

After applying Ridge regularization, the Decision Tree's performance deteriorates substantially. The regularized DT fails to capture both the decreasing and increasing phases with meaningful accuracy, resulting in poorly timed signals throughout most of the sample. This degradation is consistent with the quantitative backtest metrics: the Decision Tree exhibits the largest decline in Sharpe ratio and win rate among all evaluated models

after regularization. Interestingly, despite these sharp decreases in risk-adjusted performance, the Decision Tree still produces the highest expected value (EV) per trade. This suggests that although the model generates relatively few successful trades, the trades that do succeed tend to yield comparatively large gains, creating a unique performance profile that mixes low frequency with high payoff magnitude.

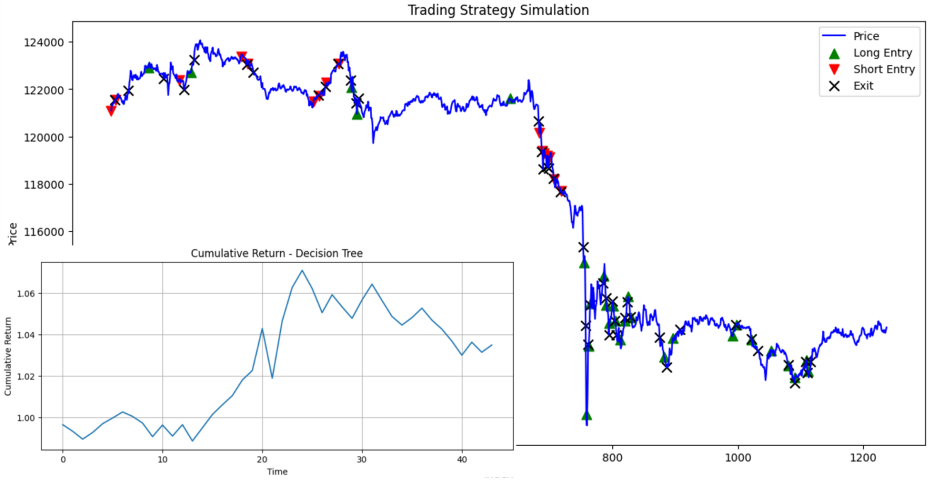

Figure 13: Trading strategy simulation for the baseline Decision Tree (DT) model. The model is conservative, performs poorly at the beginning and end of the testing period, but successfully captures both the major downward movement and much of the subsequent upward correction.

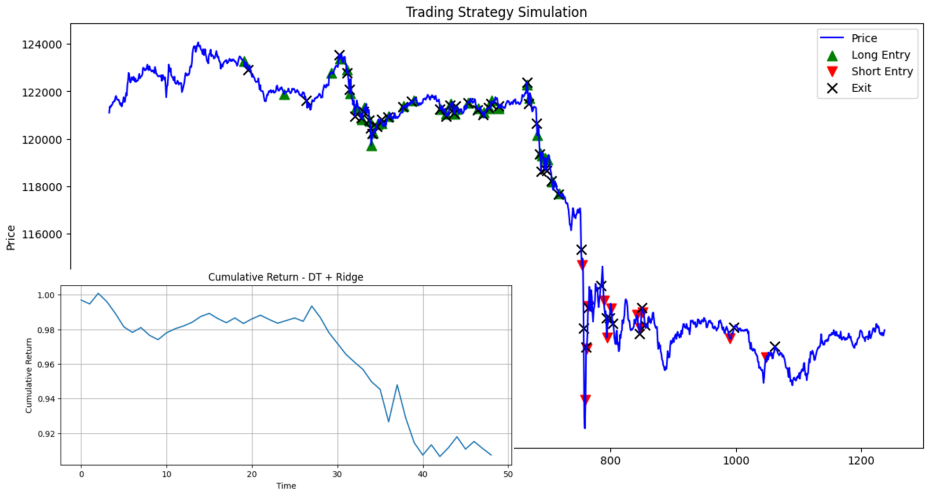

Figure 14: Trading strategy simulation for the Ridge-regularized Decision Tree model. Regularization significantly degrades performance, producing poorly timed trades and leading to large declines in Sharpe ratio and win rate, despite maintaining a relatively high expected value per trade.

5.3.5. Random Forest trading strategy comparison

Figures 15–16 present the trading strategy simulations for the baseline Random Forest (RF) model and its Ridge-regularized variant. As with the previous analyses, the figures show the model-generated long and short entries across the price series and emphasize the major drawdown region.

The baseline Random Forest model exhibits a conservative trading profile, similar to the Decision Tree. Although RF generates fewer trading signals compared with more aggressive models such as SVR or XGBoost, it successfully captures the major decreasing trend as well as the upward reversal that follows. However, its performance outside these two key segments is noticeably weak. Across much of the remaining testing horizon, its trade timing lacks precision, resulting in ineffective entries and exits that do not translate into meaningful gains. This behavior highlights the model's limited responsiveness to subtle market shifts despite its ensemble structure.

After applying Ridge regularization, the Random Forest model becomes more aggressive, producing a higher density of trading signals. However, this increase in signal frequency does not improve predictive quality. Instead, the model continues to struggle with timing accuracy and fails to reliably identify trend reversals or sustained movements. While the regularization alters the model's behavior, making it more reactive, it does not enhance its ability

to capture underlying market structure. As a result, the Ridge-regularized RF does not exhibit performance improvements and still generates imprecise signals throughout the testing period.

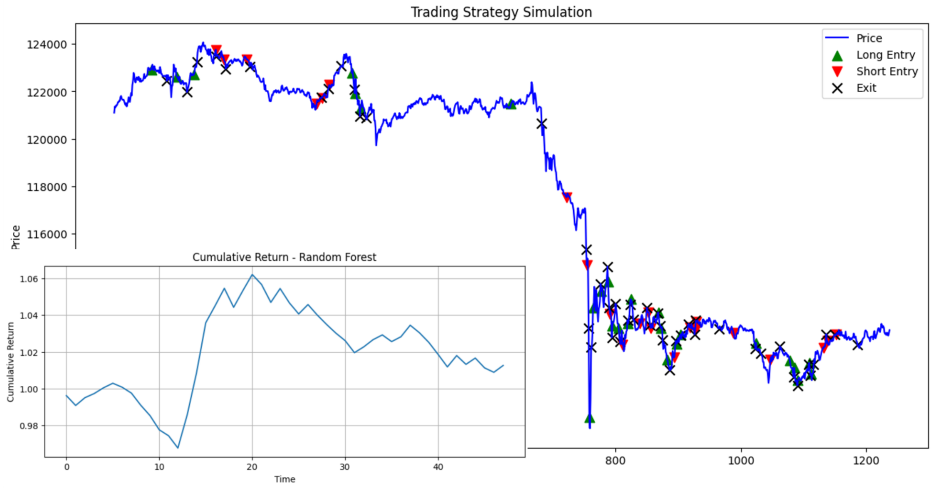

Figure 15: Trading strategy simulation for the baseline Random Forest (RF) model. RF is conservative and effectively captures both the sharp decline and subsequent increase, but performs poorly across the rest of the testing horizon due to imprecise trade timing.

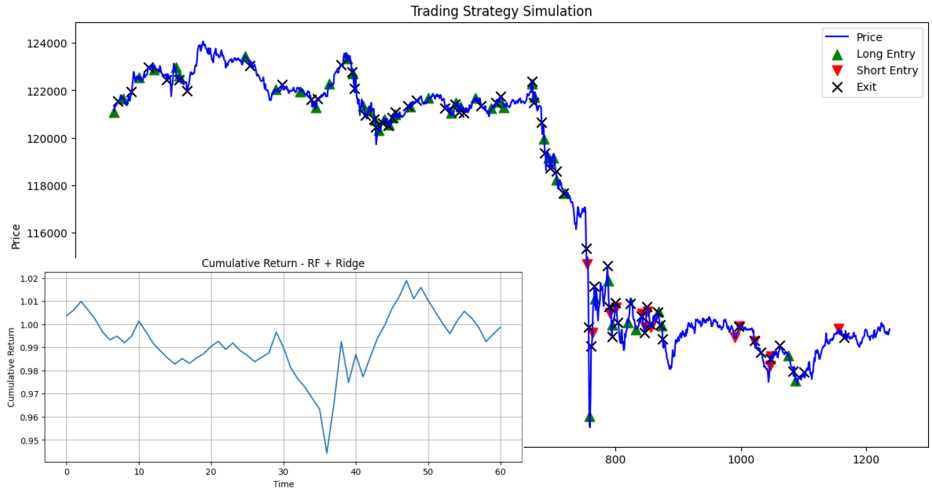

Figure 16: Trading strategy simulation for the Ridge-regularized Random Forest model. Regularization increases the model's aggressiveness by generating more frequent trading signals, but timing remains imprecise, leading to no meaningful performance improvement.

6. Conclusion

6.1. Overall model performance evaluation

Figures 17 summarize the statistical performance of all forecasting models. The empirical results reveal substantial performance heterogeneity across the evaluated approaches. Among the baseline models, XGBoost achieves the strongest risk-adjusted performance, exhibiting the highest Sharpe ratio (2.42), the lowest maximum drawdown (0.037), and one of the highest win rates. These results reflect its ability to capture nonlinear structure and respond effectively to regime-shift conditions, consistent with its successful identification of both the sharp downturn and the subsequent reversal in the trading simulations. The baseline Decision Tree also delivers competitive results, with a Sharpe ratio above 2.3 and the highest expected value (EV) per trade, although its weak performance outside major structural movements limits its practical reliability. Random Forest produces moderate yet stable results, aligned with its conservative trading behavior.

The application of Ridge regularization introduces markedly different effects across model classes. While regularization significantly improves SVR, boosting its Sharpe ratio to 2.02 and yielding the highest win rate among all regularized models, it severely degrades the performance of tree-based models. The Ridge-regularized Decision Tree shows the most dramatic deterioration, with its Sharpe ratio collapsing to –3.83 and its win rate dropping sharply. A similar—though less severe—decline is observed for the Ridge-regularized Random Forest. These outcomes highlight that introducing coefficient shrinkage disrupts the partitioning mechanisms fundamental to tree-based learners, limiting their ability to represent nonlinear relationships. By contrast, SVR benefits from the added regularization, which reduces overfitting while preserving directional predictive power.

Overall, the findings indicate that XGBoost is the strongest performer in the unregularized setting, whereas SVR becomes the most robust model once regularization is applied. The contrasting responses to Ridge regression underscore the importance of model–regularization compatibility and emphasize the necessity of adaptive model selection frameworks in financial environments characterized by abrupt market transitions and asymmetric return distributions.

Model	Sharpe Ratio	Max Drawdown	CAGR	EV_per_trade	win_rate
LR	−0.839954	0.061971	−9.999817e−01	−0.000285	0.517647
SVR	−0.376193	0.047644	−9.980736e−01	−0.000109	0.491525
XGB	2.422149	0.037379	2.865176e+12	0.000608	0.534483
DT	2.321044	0.038358	1.730686e+06	0.000816	0.522727
RF	0.794592	0.050185	1.871041e+02	0.000292	0.437500
LR_ridge	−2.526128	0.065205	−1.000000e+00	−0.001475	0.441176
SVR_ridge	2.017615	0.064518	5.508954e+10	0.000537	0.587719
XGB_ridge	0.343478	0.093718	4.616423e+01	0.000103	0.527273
DT_ridge	−3.832247	0.094308	−1.000000e+00	−0.001956	0.387755
RF_ridge	0.014803	0.064893	−4.293534e−01	0.000006	0.475410

Figure 17: Performance metrics for all forecasting models.
Red boxes highlight top-performing metrics in the baseline setting, while green boxes indicate severe performance deterioration after Ridge regularization.

Declaration

Declaration of Pedagogical Purpose

This manuscript is prepared as part of the course project for Machine Learning and FinTech (Fall 2025) at National Yang Ming Chiao Tung University, instructed by Professor Huei-Wen Teng. The work is intended solely for pedagogical purposes, including learning, discussion, and academic training. It does not represent original research submitted for journal publication, nor should it be considered as professional financial advice.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used [ChatGPT] and [Grok], developed by [OpenAI] and [xAI], respectively, in order to improve the clarity, precision, and overall quality of the writing. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

References

Boozary, P., Sheykhan, S., GhorbanTanhaei, H., 2023. Forecasting the bitcoin price using various machine learning techniques: A systematic review in data-driven marketing. Journal of Retailing and Consumer Services 71, 103208. doi:10.1016/j.jretconser.2022.103208.

Chen, Z., Li, C., Sun, W., 2020. Bitcoin price prediction using machine learning: An approach to sample dimension engineering. Journal of Computational and Applied Mathematics 365, 112395. doi:10.1016/j.cam.2019.112395.

Cocco, L., Tonelli, R., Marchesi, M., 2021. Predictions of bitcoin prices through machine learning based frameworks. PeerJ Computer Science 7, e413. doi:10.7717/peerj-cs.413.

Dimitriadou, A., Gregoriou, A., 2023. Predicting bitcoin prices using machine learning. International Review of Financial Analysis 87, 102582. doi:10.1016/j.irfa.2023.102582.

Dubey, R., Enke, D., 2021. Bitcoin price direction prediction using on-chain data and feature selection. Financial Innovation 7, 1–28. doi:10.1186/s40854-021-00231-0.

Guo, H., Zhang, D., Liu, S., Wang, L., Ding, Y., 2021. Bitcoin price forecasting: A perspective of underlying blockchain transactions. Physica A: Statistical Mechanics and its Applications 565, 125574. doi:10.1016/j.physa.2020.125574.

Huang, Z.C., Sangiorgi, I., Urquhart, A., 2022. Forecasting bitcoin volatility using machine learning techniques. Expert Systems with Applications 187, 115892. doi:10.1016/j.eswa.2021.115892.

Jaquart, P., Dann, D., Weinhardt, C., 2021. Short-term bitcoin market prediction via machine learning. Decision Support Systems 149, 113597. doi:10.1016/j.dss.2021.113597.

Nagula, P.K., Alexakis, C., 2023. A new hybrid machine learning model for predicting the bitcoin (btc-usd) price. International Review of Financial Analysis 89, 102645. doi:10.1016/j.irfa.2023.102645.

Raju, S.M., Tarif, A.M., 2023. Real-time prediction of bitcoin price using machine learning techniques and public sentiment analysis. Materials Today: Proceedings 80, 2952–2957. doi:10.1016/j.matpr.2021.11.417.