Project 002: Introduction to Trading

Executive Report

ITESO, Universidad Jesuita de Guadalajara

Market Microstructure and Trading Systems

Ana Sofía Hinojosa Bale

October 7, 2025

Table of contents

1. Strategy Overview *3*

1.1 Purpose and Objectives 3

1.2 Rationale Behind the Strategy 3

2. Data Analysis and Preprocessing *4*

2.1 Data Sources and Description 4

2.2 Data Cleaning and Preparation 4

3. Methodology and Implementation *5*

3.1 Theoretical Framework 5

3.2 Model/Algorithm Selection 5

3.3 Methodology 5

3.4 Implementation 8

3.4 Tools and Technologies Used 12

4. Results and Performance Analysis *13*

4.1 Key Findings 13

4.2 Performance Evaluation 13

4.3 Interpretation of Results 14

5. Performance Metrics, Graphs, and Tables *15*

5.1 Quantitative Performance Metrics 15

5.2 Visualization of Model Results 18

6. Risk Analysis and Limitations *23*

6.1 Identified Risks 23

6.2 Limitations of the Strategy 23

7. Conclusions *24*

7.1 Summary of Insights 24

7.2 Implications for Future Work 24

7.3 Recommendations 24

8. References *25*

1. Strategy Overview

1.1 Purpose and Objectives

The project focuses on the development of trading strategies based on technical analysis indicators. To assess strategy viability, a back testing environment was implemented to simulate historical performance under realistic market conditions. Hyperparameter optimization techniques were applied to calibrate indicator parameters and improve strategy robustness. The resulting performance metrics inform recommendations for refinement.

The purpose of this project is to implement and evaluate this technical analysis-based strategy. The key objectives are as presented below:

- To implement a back testing framework capable of simulating strategy performance under historical conditions.
- To apply a selection of technical indicators and define rule-based entry and exit conditions.
- To perform hyperparameter optimization for the strategy, identifying configurations that maximize a mean Calmar ratio.
- To analyze the strategy outcomes to parameter changes and market conditions.
- To provide data-driven recommendations for strategy refinement and assess their potential for real-time application.

1.2 Rationale Behind the Strategy

The strategy uses a combination of momentum, trend, and volatility indicators to identify trading opportunities. Specifically, it uses the Relative Strength Index (RSI) to detect overbought and oversold conditions, Simple Moving Averages (SMA) to determine trend direction, and Bollinger Bands (BB) to assess price volatility.

Trades are executed when signals indicate potential price reversals, with long positions entered in oversold conditions and short positions in overbought conditions. Risk is managed through stop-loss and take-profit thresholds, while position sizing ensures appropriate capital allocation. Hyperparameters are optimized to maximize portfolio performance while maintaining balanced exposure and controlling risk.

2. Data Analysis and Preprocessing

2.1 Data Sources and Description

This project explores the design and evaluation of systematic trading strategies applied to the BTCUSDT 1-hour time series dataset. BTCUSDT refers to the trading pair between Bitcoin (BTC) and Tether (USDT). Bitcoin is a decentralized digital asset known for its volatility and global liquidity, while Tether is a stablecoin pegged to the US dollar.

The dataset contains hourly trading data for the BTCUSDT pair, sourced from CryptoDataDownload. Each record includes the following fields:

- Unix Timestamp: Millisecond-level time reference for each data point
- Date: Human-readable timestamp in UTC
- Symbol: Asset identifier (BTCUSDT)
- Open, High, Low, Close: Price levels for each hourly candle
- Volume BTC: Total Bitcoin traded during the hour
- Volume USDT: Corresponding volume in Tether

2.2 Data Cleaning and Preparation

The raw dataset was initially ordered in reverse chronological format, beginning with the most recent entry and ending with the earliest. To ensure accurate time-series analysis, the data was sorted in ascending order by date after confirming that the date field was properly formatted for computational use.

Missing values were identified and removed to prevent errors in indicator calculations and strategy execution. This step ensures consistency across all time intervals and avoids distortions in performance metrics. Additional checks were performed to validate data types and confirm the integrity of price and volume fields.

3. Methodology and Implementation

3.1 Theoretical Framework

The methodology is grounded in rule-based algorithmic trading, where technical indicators are used to generate buy and sell signals. The strategies are evaluated using historical data through back testing, and optimized using performance metrics such as the Calmar Ratio, which balances return against drawdown risk.

3.2 Model/Algorithm Selection

The strategy integrates multiple technical indicators, including:

- **Relative Strength Index (RSI)**: Measures momentum and identifies potential overbought or oversold conditions.
- **Simple Moving Average (SMA)**: Captures trend direction over a defined window.
- **Bollinger Bands (BB)**: Quantifies volatility and potential breakout zones.

Trading signals are generated based on threshold conditions applied to these indicators. The strategy includes both long and short positions, with configurable stop-loss (SL) and take-profit (TP) levels to manage risk and secure profits.

Hyperparameter optimization is performed using Optuna, a framework for automated parameter tuning. The optimization objective is to maximize the median Calmar Ratio across multiple cross-validation splits, ensuring robustness across different market segments.

3.3 Methodology

Data Preparation

Historical BTCUSDT 1-hour data is cleaned to remove missing values and sorted in chronological order. Date fields are converted to proper datetime format to ensure compatibility with time-series operations.

Indicator Calculation

Technical indicators are calculated using flexible parameters suggested during the optimization process. These indicators help identify momentum, trend direction, and volatility, key inputs for generating trading signals.

Signal Generation

Buy and sell signals are generated by applying threshold conditions to the previously calculated technical indicators. The strategy uses a combination of momentum, trend, and volatility signals to identify potential entry and exit points.

Three types of signals are considered:

- Momentum-based signals: Triggered when the momentum indicator falls below or rises above specific thresholds, indicating potential oversold or overbought conditions.
- Trend-based signals: Generated by comparing the current price to its moving average. A price above the average suggests upward momentum, while a price below suggests downward momentum.
- Volatility-based signals: Derived from the position of the current price relative to dynamic upper and lower bands. A price below the lower band may indicate a buying opportunity, while a price above the upper band may signal a potential sell.

Performance Metrics

Custom functions are implemented to compute key performance metrics, including Sharpe Ratio, Sortino Ratio, Calmar Ratio, Maximum Drawdown, and Win Rate. These metrics are used to evaluate strategy effectiveness.

$$Sharpe\ ratio = \frac{R_p - R_f}{\sigma_p}$$

$$R_p = return\ of\ portfolio$$

$$R_f = risk\ free\ rate\ (using\ 0)$$

$$\sigma_p = standard\ deviation\ of\ portfolio$$

$$Sortino\ ratio = \frac{R_p - R_f}{\sigma_d}$$

$$R_p = return\ of\ portfolio$$

$$R_f = risk\ free\ rate\ (using\ 0)$$

$$\sigma_d = standard\ deviation\ of\ negative\ returns$$

$$Calmar\ ratio = \frac{R_p - R_f}{MD}$$

$$R_p = return\ of\ portfolio$$

$$R_f = risk\ free\ rate\ (using\ 0)$$

$$MD = maximum\ drawdown$$

$$Maximum\ drawdown = \frac{P - L}{P}$$

$$P = peak\ value\ before\ largest\ drop$$

$$L = lowest\ value\ before\ new\ high$$

$$Win\ rate = \frac{w}{t}$$

$$w = number\ of\ winning\ trades$$

$$t = total\ number\ of\ trades$$

Cross-Validation

The dataset is divided into seven segments to perform walk-forward cross-validation. Each segment is treated as an independent test set to assess strategy stability over time. This will be further explained in the following section.

Optimization

The optimization process is handled using Optuna, a framework for automated hyperparameter tuning. Its goal is to identify the combination of indicator settings and trading thresholds that produce the most stable and profitable strategy performance.

During each iteration, Optuna suggests values for several hyperparameters:

- Indicator parameters:
    - RSI window
    - SMA window
    - Bollinger Band window and deviation
- Signal thresholds:
    - RSI buy and sell levels
- Trade parameters:
    - Stop-loss (SL) and take-profit (TP) percentages
    - Number of shares/contracts per trade

Each parameter set is evaluated using backtesting across the seven cross-validation splits. The optimization objective is to maximize the mean Calmar Ratio. Invalid results are penalized with a large negative value to guide the search away from unstable configurations.

3.4 Implementation

Data Preparation

Raw BTCUSDT data is cleaned using standard pandas operations. Missing values are dropped, and timestamps are converted to datetime format to support resampling and time-based analysis.

Indicator Calculation

Indicators are added to the dataset using a function that applies the selected parameters. The RSI, SMA, and Bollinger Bands are computed using the ta library, and the resulting columns are merged into the main dataset.

Signal Generation

Thresholds are applied to the indicator columns to generate binary buy and sell signals. A final signal is confirmed only when at least two of the three indicator signals are met. These signals are stored in the dataset and used for trade simulation.

Backtesting

The backtesting module simulates the execution of trades over historical data using the signals generated by the strategy. It applies stop-loss (SL) and take-profit (TP) thresholds to manage risk and ensure profits. The simulation tracks portfolio value at each time step and accounts for transaction costs using a fixed commission rate of 0.125%.

Trades are executed based on two types of signals:

- Buy signals trigger long positions, where the strategy purchases the asset expecting its price to rise.
- Sell signals trigger short positions, where the strategy sells the asset expecting its price to fall.

Each position is represented by a position object, which stores the entry price, SL and TP levels, and the number of shares traded. Long positions are closed when the price drops below the stop-loss or rises above the take-profit. Short positions are closed when the price rises above the stop-loss or falls below the take-profit. All trades are adjusted for commission costs.

At each time step, the portfolio value is calculated by summing the available cash and the market value of all open positions. This value is stored in a time series that reflects the evolution of the portfolio throughout the simulation.

Optimization

The parameters are passed to the indicator and signal-generation functions, which return a new DataFrame containing buy and sell signals. The backtesting function is applied independently to each of the seven segments defined in the cross-validation process. For each segment, the resulting portfolio values are used to compute the Calmar Ratio, which measures return relative to drawdown. The mean Calmar Ratio across all segments serves as the optimization target.

This approach ensures that the strategy is evaluated under varied market conditions and helps reduce the risk of overfitting to a specific time period. This iterative process continues until a stopping condition is met, resulting in a specific set of parameters.

Return Tables

Monthly, quarterly, and annual returns are calculated from the portfolio value time series and presented in tabular format to provide a clear view of performance across different time horizons. These tables help identify periods of strong or weak performance and support comparative analysis across market cycles.

The return series is computed by resampling the portfolio data to end-of-period values and calculating percentage changes. To ensure consistency, quarterly and annual returns are aligned with the monthly index using forward fill.

For visualization, a custom table display function is implemented using Matplotlib. It formats the return data and applies conditional cell coloring: green for positive returns and red for negative ones. If the table exceeds 25 rows, it is automatically split into two side-by-side panels to improve readability. This visual enhancement supports quick interpretation of performance trends and highlights periods of gain or loss.

Graphical Analysis

Visualization functions are used to plot portfolio value over time, return distributions, and the timing of buy and sell signals across various datasets. These graphs support interpretability and help identify strategy behavior under different market conditions.

Portfolio value plots are generated for the training, test, and validation sets. The training plot shows how the strategy performs during the optimization phase, while the test and validation plots are combined into a continuous series to illustrate performance on unseen data. To ensure visual continuity, the validation curve is shifted to begin from the final value of the test segment.

Return distributions are visualized using histograms of monthly returns. These plots help assess the skewness, dispersion, and frequency of gains and losses, offering insight into the strategy's risk profile. Additionally, rolling volatility is plotted to show how return variability evolves over time, highlighting periods of increased uncertainty or stability.

Signal timing is visualized by overlaying buy and sell markers on the price chart. This allows for inspection of how signals align with market movements and whether entries and exits occur at favorable points. Buy signals are marked with upward arrows, and sell signals with downward arrows, making it easy to evaluate.

Together, these visualizations provide a comprehensive view of strategy behavior, supporting both diagnostic analysis and presentation of results.

To illustrate the overall workflow, two diagrams were created to represent the key phases of the process. The first diagram outlines the training and optimization pipeline, including data preprocessing, signal generation, cross-validation, and hyperparameter tuning. The second diagram presents the final evaluation phase, showing how the optimized parameters are applied to the full dataset.

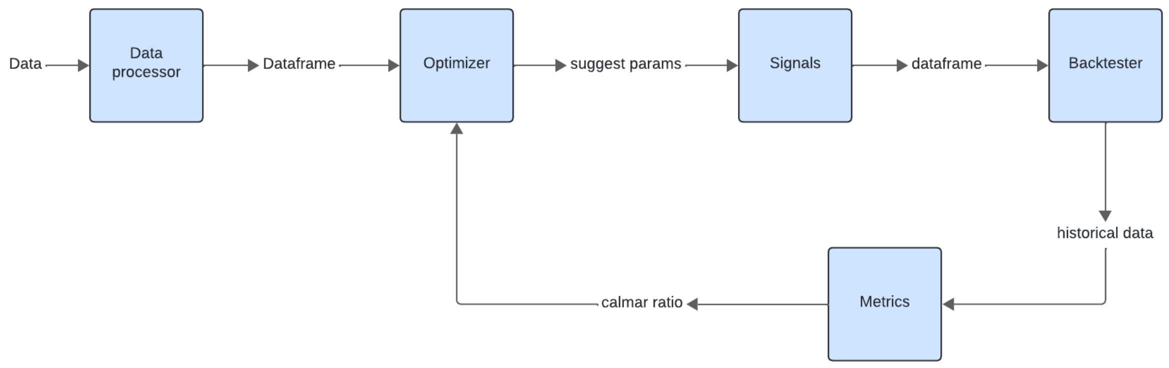


Image 1

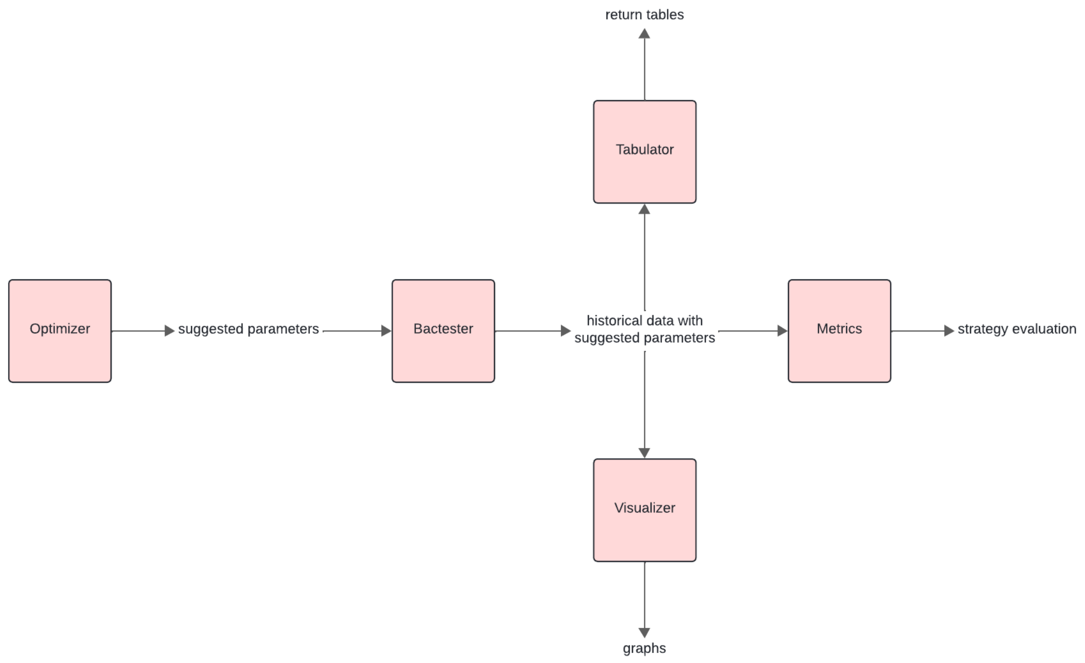


Image 2

As shown in Image 1, the dataset is first cleaned and processed, resulting in a structured DataFrame suitable for analysis. Once preprocessing is complete, the optimization phase begins.

During each iteration of the optimization loop, the program suggests a set of hyperparameters within predefined intervals. These parameters are used to compute technical indicators and generate trading signals. The signal-generation functions produce a new DataFrame that identifies the specific time points where buy or sell conditions are met.

The resulting DataFrame is then passed into the backtesting function, which simulates trades based on historical market behavior. Each trade is recorded, including the amount of capital gained or lost and any open positions that remain active. At the end of the simulation, all remaining positions are closed to finalize the portfolio value.

Once the optimization process concludes, a final evaluation phase begins. As illustrated in Image 2, the best-performing parameter set, identified through cross-validation, is re-applied to the backtesting function to simulate strategy performance across the dataset. This simulation generates a record of all executed trades, including entry and exit points, realized profits and losses, and the evolution of portfolio value over time. Final capital outcomes are calculated after all positions are closed, providing a complete view of strategy behavior.

The resulting data is then used to produce a series of visualizations and summary tables. These include monthly, quarterly, and annual return breakdowns, as well as plots showing portfolio growth, return distributions, and the timing of buy and sell signals.

It is important to note that the optimization phase (Image 1) is conducted exclusively on the training set, which comprises the first 60% of historical data. The final evaluation phase (Image 2) includes the training set, a test set (next 20%), and a validation set (final 20%). This three-part structure ensures that the strategy is evaluated on both seen and unseen data, reducing the risk of overfitting and improving confidence in the generalizability of the results. Performance metrics from each segment are compared to assess stability.

3.4 Tools and Technologies Used

The tools most frequently used throughout the project were:

- **Python**: Core programming language for data handling and strategy logic.
- **Pandas**: Used for data manipulation and time-series processing.
- **Optuna**: Framework for hyperparameter optimization.
- **Dataclasses**: Used to define structured trading positions.
- **NumPy**: Supports numerical operations and performance metric calculations.
- **Matplotlib:** Enables the creation of visualizations including portfolio value plots, return distributions, and signal timing charts.
- **Ta:** A technical analysis library used to compute indicators such as RSI, SMA, and Bollinger Bands.

4. Results and Performance Analysis

4.1 Key Findings

Parameters obtained after optimization process:

PARAMETER/METRIC	VALUE
RSI WINDOW	12
SMA WINDOW	21
BB WINDOW	12
BB DEVIATION	1.95
RSI BUY	39
RSI SELL	90
STOP LOSS	0.0994
TAKE PROFIT	0.1729
NUMBER OF SHARES	8.5075
MEAN CALMAR RATIO	1.7985

Metrics obtained after applying the strategy across different data segments:

METRIC/OUTPUT	TRAIN SET	TEST SET	VALIDATION SET
SHARPE RATIO	0.4608	0.8228	1.0640
SORTINO RATIO	0.5451	0.9981	1.2754
CALMAR RATIO	0.3223	0.9203	1.4421
MAX DRAWDOWN	-64.41%	-33.68%	-28.55%
WIN RATE	48.26%	49.55%	47.67%
FINAL CASH WHEN STARTING FROM INITIAL CAPITAL OF 1,000,000 USD	1,670,577.00 USD	1,470,923.41 USD	1,722,807.43 USD

4.2 Performance Evaluation

After running the optimization process, which resulted in a mean Calmar Ratio of 1.79852 across the seven cross-validation segments, the optimal parameter configuration was determined as follows: the Relative Strength Index (RSI) window was set to 12, the Simple Moving Average (SMA) window to 21, and the Bollinger Bands window to 12 with a deviation factor of 1.95. The RSI thresholds for signal generation were tuned to 39 for buy signals and 90 for sell signals, indicating a preference for entering trades only under strong momentum conditions. The stop-loss threshold was optimized to approximately 9.94%, while the take-profit threshold was set to nearly 17.29%.

On the training set, the strategy achieved moderate performance, with a Sharpe Ratio of 0.4608 and a Sortino Ratio of 0.5451.The Calmar Ratio was 0.3223, and the maximum drawdown reached −62.41%, indicating high volatility during the learning phase. The win rate was 48.26%, and the final cash balance grew to 1,670,577.00 from an initial capital of 1,000,000.

Performance improved on the test set, where the Sharpe Ratio rose to 0.8228 and the Sortino Ratio to 0.9981, showing stronger returns with reduced downside risk. The Calmar Ratio increased to 0.9203, and the maximum drawdown dropped to −33.68%. The win rate climbed to 49.55%, and the final cash balance reached 1,470,923.41.

Validation results confirmed the strategy's stability, with further gains in all key metrics. The Sharpe Ratio climbed to 1.0640, the Sortino Ratio to 1.2754, and the Calmar Ratio peaked at 1.4421. Maximum drawdown was the lowest across all sets at −28.55%, and the win rate remained stable at 47.67%. The final cash balance reached 1,722,807.43, indicating strong generalization and profitability on unseen data.

4.3 Interpretation of Results

The results show that the strategy performs better as it moves from training to test and validation sets, suggesting it generalizes well and avoids overfitting. While training returns were modest and drawdown was high, the test and validation sets showed steady improvement in Sharpe, Sortino, and Calmar Ratios.

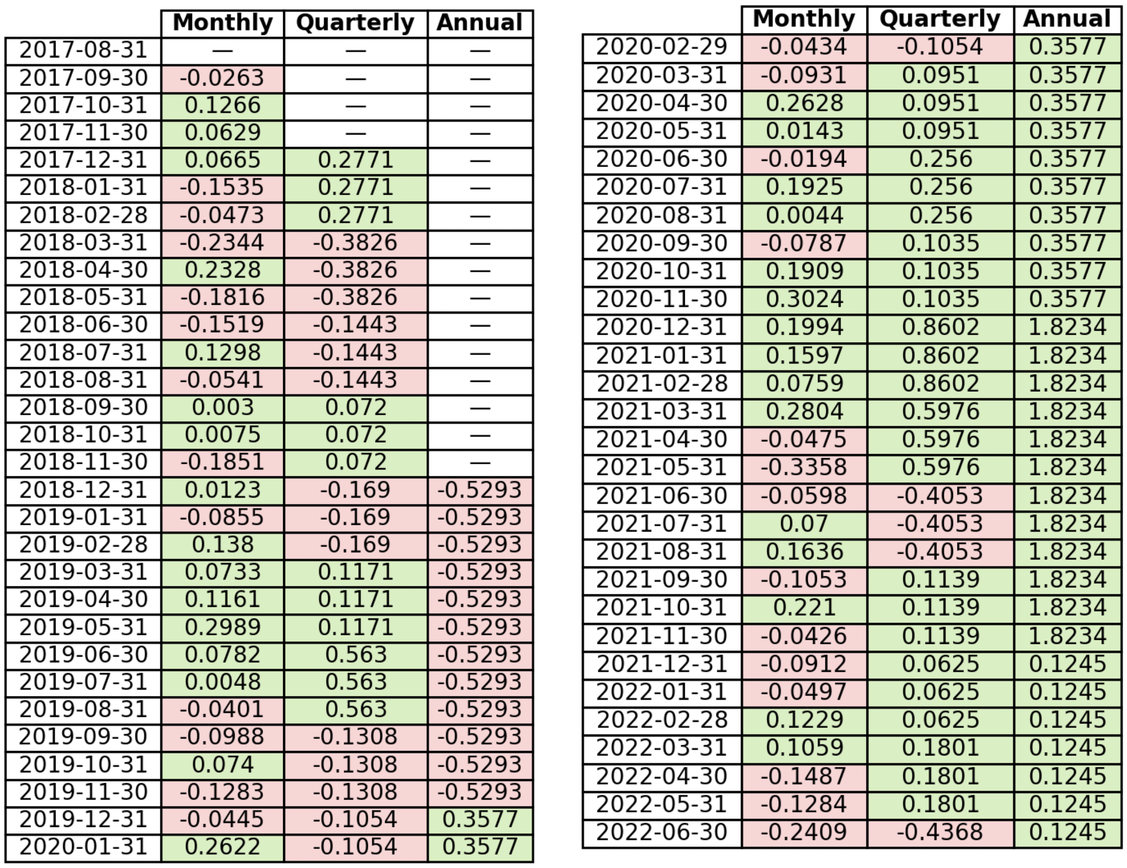
This means the chosen parameters likely reflect real market patterns, not just noise. Lower drawdowns and consistent win rates highlight the strategy's stability, even in volatile conditions.

The growing cash balances across all phases confirm strong performance, and the high Calmar Ratio on validation proves the strategy can stay profitable while managing risk. Overall, the optimization process found a reliable setup that works across different market environments.

5. Performance Metrics, Graphs, and Tables

5.1 Quantitative Performance Metrics

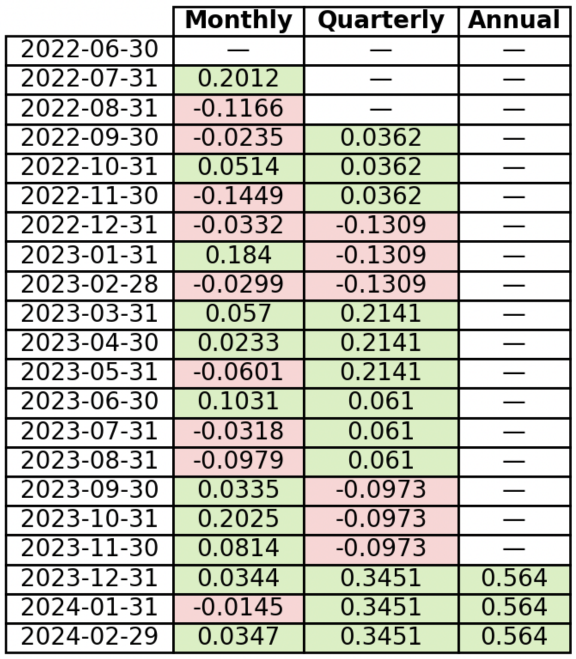
Train set returns table



	Monthly	Quarterly	Annual
2017-08-31	—	—	—
2017-09-30	-0.0263	—	—
2017-10-31	0.1266	—	—
2017-11-30	0.0629	—	—
2017-12-31	0.0665	0.2771	—
2018-01-31	-0.1535	0.2771	—
2018-02-28	-0.0473	0.2771	—
2018-03-31	-0.2344	-0.3826	—
2018-04-30	0.2328	-0.3826	—
2018-05-31	-0.1816	-0.3826	—
2018-06-30	-0.1519	-0.1443	—
2018-07-31	0.1298	-0.1443	—
2018-08-31	-0.0541	-0.1443	—
2018-09-30	0.003	0.072	—
2018-10-31	0.0075	0.072	—
2018-11-30	-0.1851	0.072	—
2018-12-31	0.0123	-0.169	-0.5293
2019-01-31	-0.0855	-0.169	-0.5293
2019-02-28	0.138	-0.169	-0.5293
2019-03-31	0.0733	0.1171	-0.5293
2019-04-30	0.1161	0.1171	-0.5293
2019-05-31	0.2989	0.1171	-0.5293
2019-06-30	0.0782	0.563	-0.5293
2019-07-31	0.0048	0.563	-0.5293
2019-08-31	-0.0401	0.563	-0.5293
2019-09-30	-0.0988	-0.1308	-0.5293
2019-10-31	0.074	-0.1308	-0.5293
2019-11-30	-0.1283	-0.1308	-0.5293
2019-12-31	-0.0445	-0.1054	0.3577
2020-01-31	0.2622	-0.1054	0.3577
2020-02-29	-0.0434	-0.1054	0.3577
2020-03-31	-0.0931	0.0951	0.3577
2020-04-30	0.2628	0.0951	0.3577
2020-05-31	0.0143	0.0951	0.3577
2020-06-30	-0.0194	0.256	0.3577
2020-07-31	0.1925	0.256	0.3577
2020-08-31	0.0044	0.256	0.3577
2020-09-30	-0.0787	0.1035	0.3577
2020-10-31	0.1909	0.1035	0.3577
2020-11-30	0.3024	0.1035	0.3577
2020-12-31	0.1994	0.8602	1.8234
2021-01-31	0.1597	0.8602	1.8234
2021-02-28	0.0759	0.8602	1.8234
2021-03-31	0.2804	0.5976	1.8234
2021-04-30	-0.0475	0.5976	1.8234
2021-05-31	-0.3358	0.5976	1.8234
2021-06-30	-0.0598	-0.4053	1.8234
2021-07-31	0.07	-0.4053	1.8234
2021-08-31	0.1636	-0.4053	1.8234
2021-09-30	-0.1053	0.1139	1.8234
2021-10-31	0.221	0.1139	1.8234
2021-11-30	-0.0426	0.1139	1.8234
2021-12-31	-0.0912	0.0625	0.1245
2022-01-31	-0.0497	0.0625	0.1245
2022-02-28	0.1229	0.0625	0.1245
2022-03-31	0.1059	0.1801	0.1245
2022-04-30	-0.1487	0.1801	0.1245
2022-05-31	-0.1284	0.1801	0.1245
2022-06-30	-0.2409	-0.4368	0.1245

Image 3

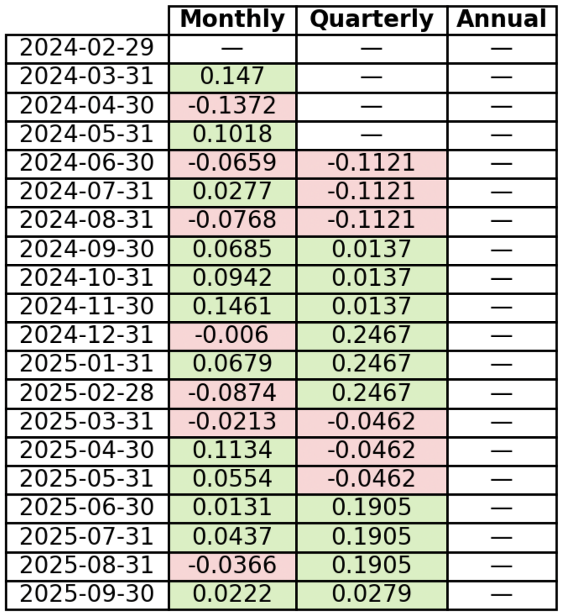
Test set returns table



	Monthly	Quarterly	Annual
2022-06-30	—	—	—
2022-07-31	0.2012	—	—
2022-08-31	-0.1166	—	—
2022-09-30	-0.0235	0.0362	—
2022-10-31	0.0514	0.0362	—
2022-11-30	-0.1449	0.0362	—
2022-12-31	-0.0332	-0.1309	—
2023-01-31	0.184	-0.1309	—
2023-02-28	-0.0299	-0.1309	—
2023-03-31	0.057	0.2141	—
2023-04-30	0.0233	0.2141	—
2023-05-31	-0.0601	0.2141	—
2023-06-30	0.1031	0.061	—
2023-07-31	-0.0318	0.061	—
2023-08-31	-0.0979	0.061	—
2023-09-30	0.0335	-0.0973	—
2023-10-31	0.2025	-0.0973	—
2023-11-30	0.0814	-0.0973	—
2023-12-31	0.0344	0.3451	0.564
2024-01-31	-0.0145	0.3451	0.564
2024-02-29	0.0347	0.3451	0.564

Image 4

Validation set returns table



	Monthly	Quarterly	Annual
2024-02-29	—	—	—
2024-03-31	0.147	—	—
2024-04-30	-0.1372	—	—
2024-05-31	0.1018	—	—
2024-06-30	-0.0659	-0.1121	—
2024-07-31	0.0277	-0.1121	—
2024-08-31	-0.0768	-0.1121	—
2024-09-30	0.0685	0.0137	—
2024-10-31	0.0942	0.0137	—
2024-11-30	0.1461	0.0137	—
2024-12-31	-0.006	0.2467	—
2025-01-31	0.0679	0.2467	—
2025-02-28	-0.0874	0.2467	—
2025-03-31	-0.0213	-0.0462	—
2025-04-30	0.1134	-0.0462	—
2025-05-31	0.0554	-0.0462	—
2025-06-30	0.0131	0.1905	—
2025-07-31	0.0437	0.1905	—
2025-08-31	-0.0366	0.1905	—
2025-09-30	0.0222	0.0279	—

Image 5

Returns were calculated as shown in Images 3, 4, and 5, on a monthly, quarterly, and annual basis. Annual returns were computed from December of one year to December of the next. Therefore, annual returns are not shown for the validation set, as it does not contain the two December endpoints required for that calculation.

Green and red highlighting colors were used to visually distinguish positive and negative returns, green indicating positive values and red indicating negative ones. As shown in Image 4, which displays the training set return table, early periods were dominated by negative returns. However, as time progressed, the strategy's performance improved, with more frequent positive outcomes appearing in later intervals.

In the case of the test set, as shown in Image 4, some negative returns were still present. However, most of these returns were positive, including a favorable annual return. This suggests that the strategy began to stabilize and generate more consistent gains outside the training phase.

Lastly, for the validation set, as illustrated in Image 5, most monthly and quarterly returns remained positive. This suggests that the strategy sustained its profitability even on unseen data, reinforcing its ability to generalize and perform consistently beyond the training and test phases.

5.2 Visualization of Model Results

Portfolio value over time

Train, test and validation sets

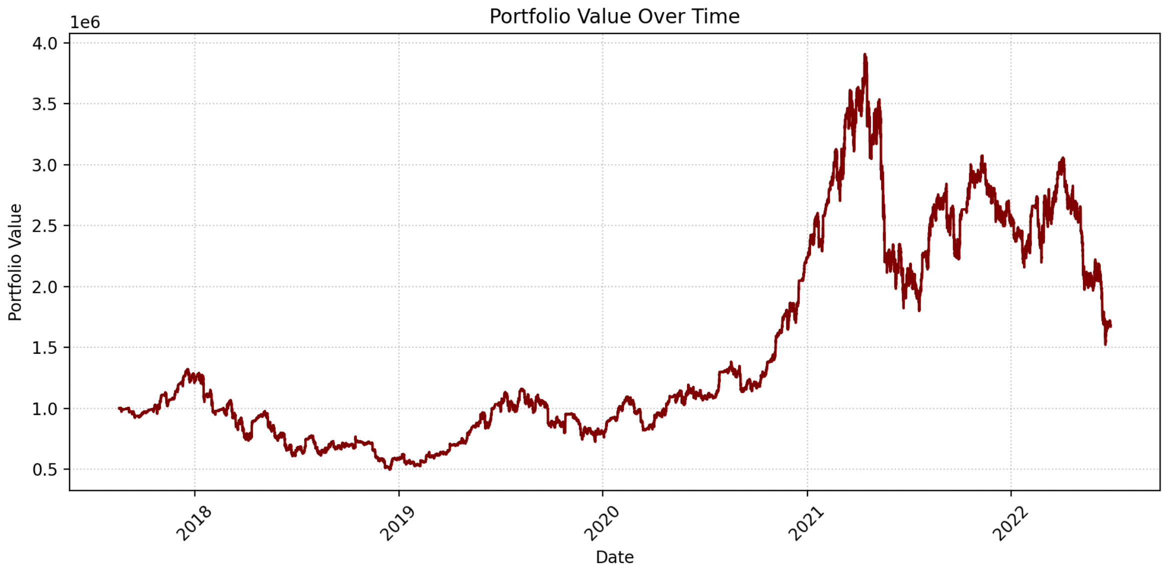


Image 6

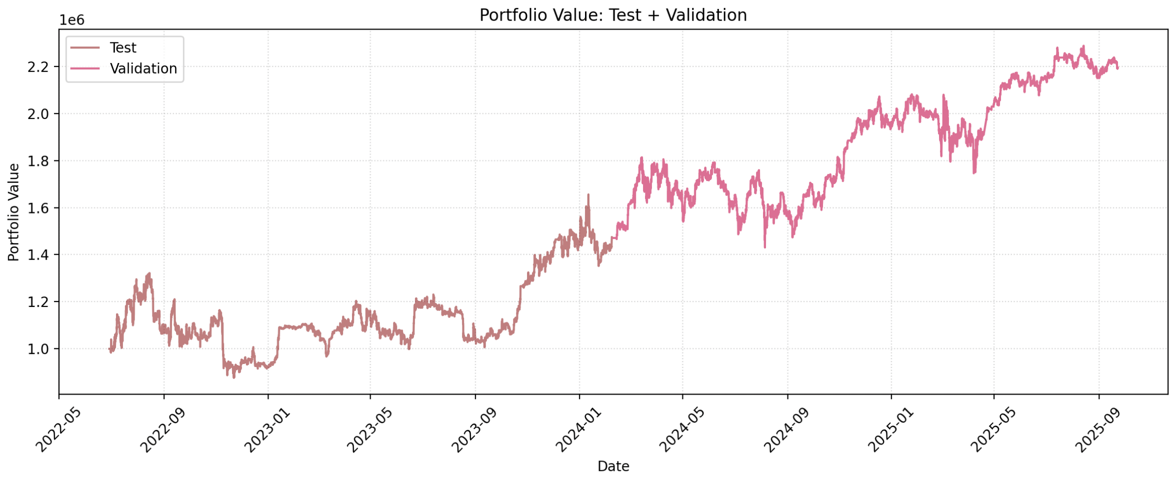


Image 7

Image 6 illustrates how the portfolio value evolves across the training set. As can be seen, the value remains relatively stable until the end of 2020, after which it begins to rise sharply, reaching its peak in early 2021. Following this high point, the portfolio experiences a gradual decline, ending slightly higher than its initial level by the close of 2022.

In contrast, Image 7 shows the portfolio value progression across the test and validation sets. To ensure continuity, the validation phase begins with the final capital from the test set. Despite some fluctuations, the overall trend is upward, with the portfolio value steadily increasing over time. This suggests that the strategy maintained its effectiveness beyond the training phase, adapting well to new data.

Monthly returns distribution

Test and validation sets

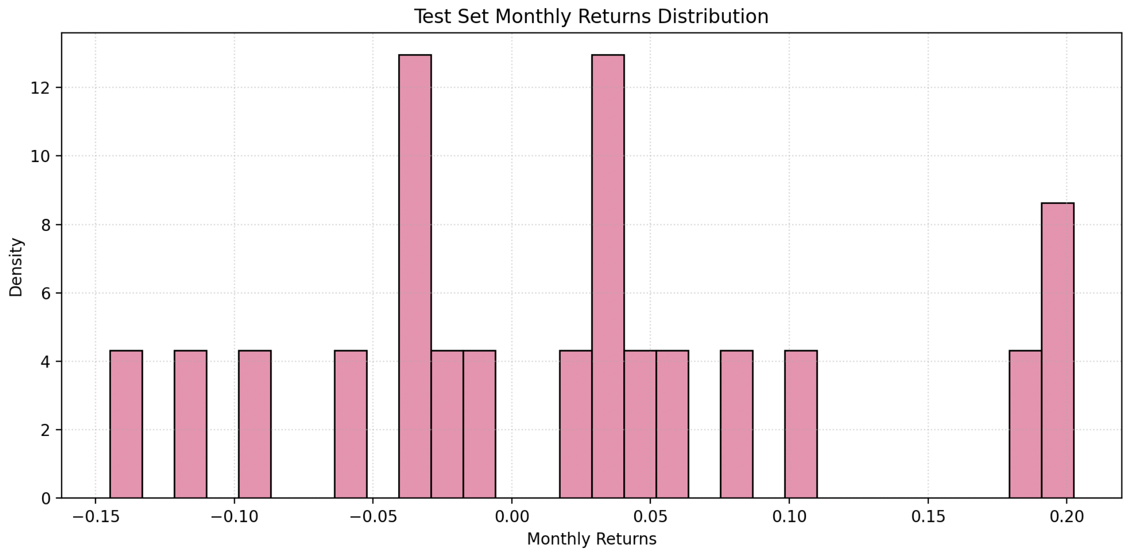


Image 8

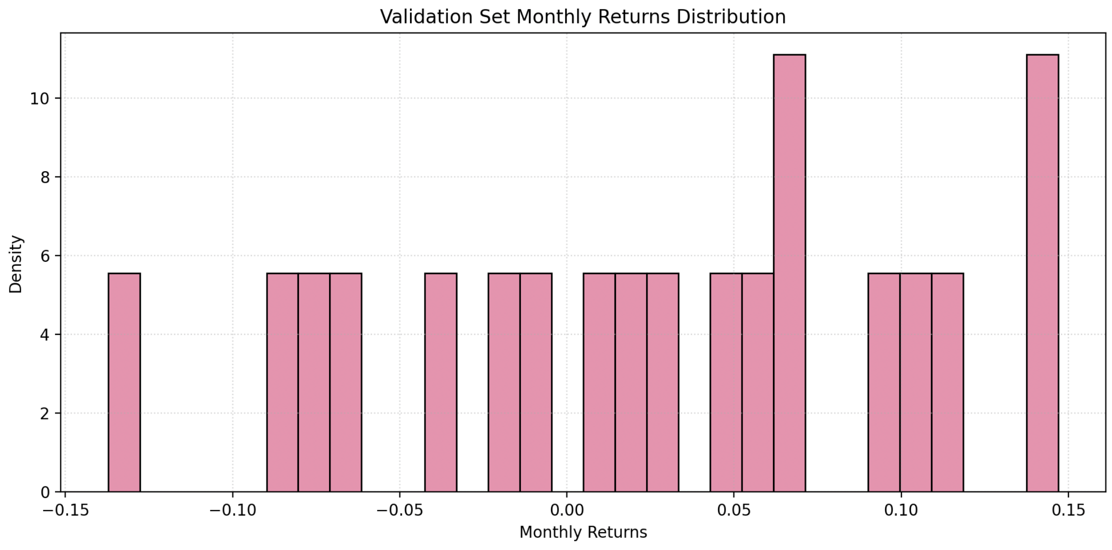


Image 9

Distributions of the monthly returns for both the test and validation sets were created to help visually identify the most frequent return intervals. As can be seen in Image 8, the histogram for the test set shows a distribution with peaks around -0.05 and 0.20, indicating that both moderate losses and strong gains occurred with notable frequency.

In contrast, the validation set, also shown in Image 9, displays a more balanced distribution, with peaks around 0.05 and 0.15, suggesting a tendency toward positive returns. These visualizations provide insight into the behavior of the strategy across different datasets and help assess its stability and performance under varying conditions.

60 period rolling volatility of hourly returns

Test and validation sets

Test set

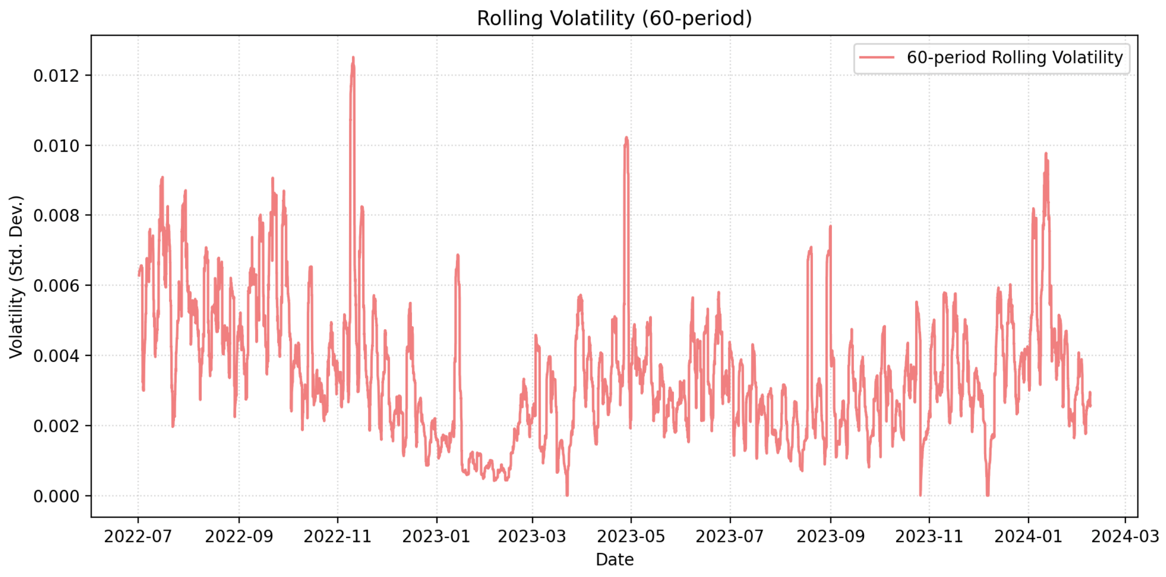


Image 10

Validation set

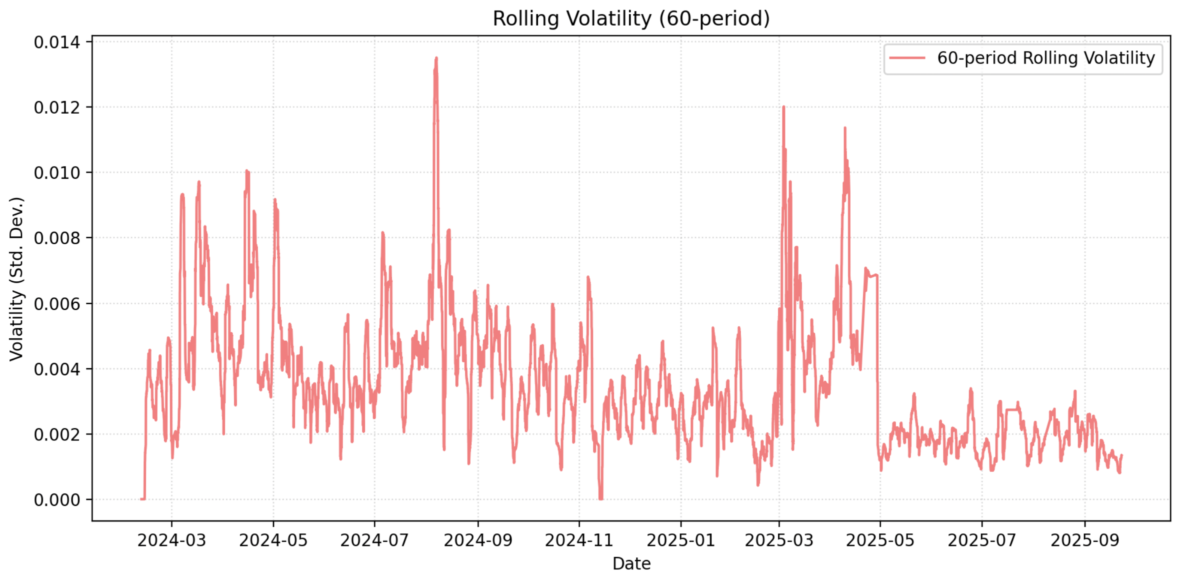


Image 11

The two previous graphs visualize how the volatility of hourly returns changes over time using a 60-period rolling window. This means that for each point on the graph, the standard deviation of returns was calculated over the previous 60 periods, providing a measure of how much returns fluctuated during that span. Higher values indicate greater uncertainty or risk, while lower values suggest more stable performance. This rolling approach helps reveal trends in market behavior and detect shifts in volatility that may not be visible in raw return data.

In the case of the test set, as seen in Image 10, elevated volatility levels persist throughout the dataset, with occasional intervals of reduced fluctuation. This pattern suggests that the strategy was exposed to sustained market uncertainty during the test period, although brief phases of relative stability did occur. In contrast, the validation set, illustrated in Image 11, initially shows higher volatility values, but over time the line trends downward, indicating a gradual decrease in volatility and a shift toward portfolio stability. The occasional spikes in the graph reflect short bursts of increased uncertainty or market movement, which may correspond to external shocks or regime changes. Overall, the declining trend in the validation set supports the interpretation that the strategy adapted well to changing conditions and maintained resilience in less volatile environments.

Buy and sell points on price chart

Test set

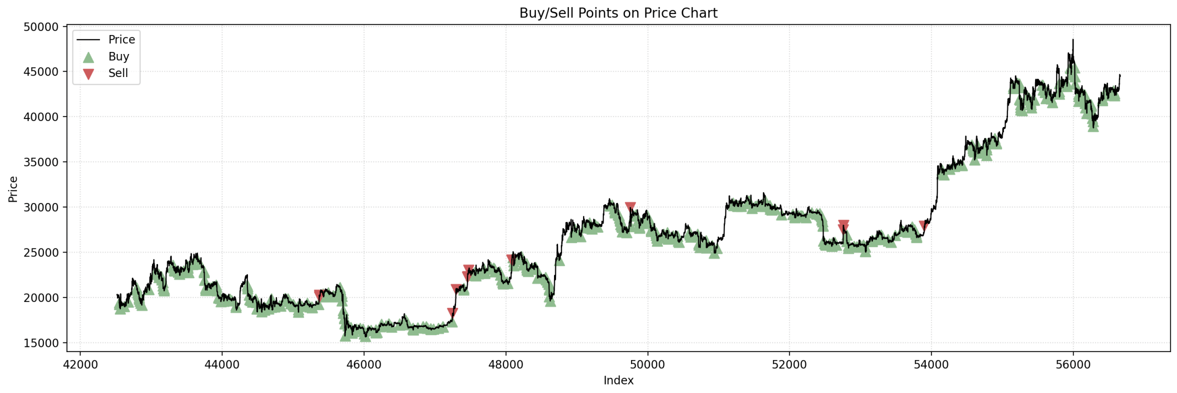


Image 12

Lastly, buy and sell points were plotted on the price chart to illustrate, at least in the test set, how the strategy operates in practice. As can be seen, the great majority of executed trades were triggered by buy signals. As observed across multiple simulations, the most favorable results often occurred in scenarios where most trades followed this pattern, suggesting that the strategy performs best when focused on identifying long opportunities.

6. Risk Analysis and Limitations

6.1 Identified Risks

The strategy, while systematic and data-driven, is subject to several risks:

- **Market Volatility**: Cryptocurrency markets are highly volatile and can experience rapid price swings that exceed predefined stop-loss or take-profit thresholds, leading to unexpected losses.
- **Overfitting**: Despite the use of walk-forward cross-validation, there remains a risk that the strategy is overly tuned to historical data and may not generalize well to future market conditions.
- **Execution Risk**: The backtesting setup assumes that trades happen exactly at the signal price. But in real trading, delays, price changes, and limited availability can cause trades to happen at worse prices, which may reduce actual profits.
- **Liquidity Constraints**: Although BTCUSDT is highly liquid, sudden drops in volume or exchange outages may prevent trades from executing as expected.
- **Parameter Sensitivity**: Small changes in hyperparameters lead to large variations in performance, indicating potential instability in the strategy's behavior.

6.2 Limitations of the Strategy

Several limitations were identified during development and testing:

- **Simplified Assumptions**: The strategy does not account for complex market dynamics such as news events, economic announcements, or relationships between different assets. It also does not use leverage, meaning all trades are made with available capital only. This limits both potential gains and risks but may not reflect how many real-world strategies operate with borrowed funds to amplify returns.
- **Fixed Trade Size**: The model uses a constant number of shares per trade, which may not reflect optimal capital allocation or risk-adjusted position sizing.
- **No Portfolio Diversification**: The strategy is applied to a single asset (BTCUSDT), limiting its ability to balance risk across multiple instruments.
- **Static Thresholds**: RSI and other indicator thresholds are fixed per optimization cycle and do not adapt dynamically to changing market regimes.

These risks and limitations should be carefully considered before deploying the strategy in a live trading environment. Future work may include dynamic position sizing, multi-asset integration, and real-time execution modeling to address these constraints.

7. Conclusions

7.1 Summary of Insights

The strategy showed strong and consistent performance across both the test and validation sets, with a predominance of positive monthly returns and a favorable annual return. Rolling volatility plots revealed a gradual decline in risk over time, especially in the validation phase, suggesting improved stability and adaptability. Trade signal analysis confirmed that most executed trades were triggered by buy signals, which aligned with the most profitable outcomes across simulations. Portfolio value charts further supported this, showing steady growth in the test and validation phases, reinforcing the strategy's ability to generalize and perform well on unseen data.

7.2 Implications for Future Work

The strategy's focus on long-side signals worked well under the conditions tested, but its effectiveness in different conditions remains uncertain. Future work could explore incorporating more indicators to improve adaptability across different types of prices. Expanding the testing framework to include additional indicators beyond Bollinger Bands, RSI, and SMA would help assess the strategy's stability and generalizability. Incorporating tools like MACD for trend confirmation, ATR for dynamic stop-loss sizing, or volume-based metrics such as On-Balance Volume (OBV) could provide complementary signals and improve trade timing.

7.3 Recommendations

To improve performance, it is recommended to refine the entry logic to better detect changes in price direction and expand the strategy beyond buy-only setups. Adding ways to trigger sell signals, even though buying produced the best results, could make the strategy more flexible in different market conditions. Using adaptive tools that respond to changes in volatility can help manage risk more effectively. Including more evaluation metrics during optimization, such as the Sortino ratio may lead to better and more stable results.

8. References

Chen, J. (2023, August 28). *Relative Strength Index (RSI): Definition and calculation*. Investopedia. https://www.investopedia.com/terms/r/rsi.asp

Chen, J. (2023, August 28). *Simple Moving Average (SMA): Definition and calculation*. Investopedia. https://www.investopedia.com/terms/s/sma.asp

Chen, J. (2023, August 28). *Bollinger Bands: Definition, calculation, and uses*. Investopedia. https://www.investopedia.com/terms/b/bollingerbands.asp

Equiti Group. (n.d.). *Key metrics for evaluating risk and return in trading*. Equiti. https://www.equiti.com/sc-en/news/trading-ideas/key-metrics-for-evaluating-risk-and-return-in-trading/

Bukosabino, I. (n.d.). *Technical Analysis Library in Python*. Read the Docs. https://technical-analysis-library-in-python.readthedocs.io/en/latest/

Matplotlib Development Team. (2021). *matplotlib.pyplot — Matplotlib 3.5.3 documentation*. https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html

Pandas Development Team. (n.d.). *pandas: Python data analysis library*. https://pandas.pydata.org/

NumPy Developers. (n.d.). *NumPy*. https://numpy.org/

Python Software Foundation. (n.d.). *dataclasses — Data Classes*. https://docs.python.org/3/library/dataclasses.html

Optuna Developers. (n.d.). *Optuna: Hyperparameter optimization framework*. https://optuna.org/