From Data to Decisions: A Trading Bot for Ethereum

Bogdan-Petru Vrînceanu, Florentin Șerban

Bucharest Academy of Economic Studies, Romania

DOI: https://doi.org/10.51244/IJRSI.2025.12020010

Received: 23 January 2025; Accepted: 27 January 2025; Published: 27 February 2025

ABSTRACT

This research focuses on the development and evaluation of an automated trading strategy for cryptocurrency markets, specifically Ethereum, using Pine Script version 5 on the Trading View platform. The strategy incorporates a variety of technical indicators, including the Relative Strength Index (RSI), Commodity Channel Index (CCI), Average True Range (ATR), Directional Movement Index (DMI), Aroon Indicator, and Exponential Moving Average (EMA), to capture diverse market conditions such as trend direction, volatility, and overbought/oversold levels. To enhance decision-making, a multiple logistic regression model was applied, allowing the trading bot to predict market movements by assigning probabilities to different outcomes based on the interplay of these indicators.

The strategy was tested on a 3-minute timeframe for both long and short positions between January 6 and January 22, 2025, with an initial capital of $1,000 and profits reinvested in each trade to capitalize on compounding. Over the trading period, the strategy achieved a net profit of 27.1%, equating to $271, with 319 closed trades. The win rate of 22.88% was complemented by a risk-reward ratio of 1.3, emphasizing the strategy’s ability to maximize returns from successful trades. Key performance metrics, including a Sharpe ratio of 0.4 and a Sortino ratio of 1.067, highlighted the strategy's relative stability and ability to handle downside risk in volatile cryptocurrency markets.

The research demonstrates the effectiveness of integrating multiple technical indicators with logistic regression in automated trading systems, offering valuable insights into the potential of algorithmic trading in highly volatile environments like the cryptocurrency market.

Keywords: Cryptocurrency Trading, Automated Strategies, Multiple Logistic Regression, Algorithmic Trading Bots

INTRODUCTION

Algorithmic trading bots have revolutionized the financial landscape by enabling traders to execute strategies with speed, precision, and consistency. In the fast-paced world of cryptocurrency, where price fluctuations can occur within seconds, these automated systems play an even more crucial role. By leveraging algorithmic bots, traders can overcome the limitations of human decision-making, such as emotional biases, fatigue, and delayed reactions, ensuring more efficient and objective trading outcomes. Furthermore, the ability to process vast amounts of data in real-time gives algorithmic bots a distinct edge in identifying patterns, trends, and opportunities that might otherwise go unnoticed [1][2][9].

The role of automation in trading cannot be overstated, particularly as markets grow increasingly complex and dynamic. The implementation of trading strategies through tools like Pine Script has become a cornerstone of modern trading. Pine Script, the proprietary scripting language of Trading View, is uniquely designed for creating, back testing, and optimizing trading strategies within an intuitive interface. Its user-friendly yet powerful syntax empowers developers to build custom indicators and automate decision-making processes that align with their trading styles and goals. By automating repetitive tasks, such as monitoring price levels, executing trades, and recalibrating strategies based on real-time data, traders can allocate more time to refining their analytical frameworks and adapting to ever-evolving market conditions. Additionally, Pine Script’s versatility enables seamless integration of technical analysis, mathematical models, and rule-based logic into a single cohesive framework, which enhances both precision and efficiency [4][7].

Among the many statistical models available for algorithmic trading, multiple logistic regression stands out asa particularly robust and versatile tool for decision-making. Logistic regression, a form of predictive analysis, is especially well-suited for addressing binary outcomes—for instance, predicting whether a price will rise or fall within a specific time frame. By incorporating multiple independent variables, such as historical price data, trading volume, and technical indicators, logistic regression allows traders to construct sophisticated models that estimate the probability of specific market events. These probability estimates form the backbone of objective, data-driven trading decisions, minimizing guesswork and emotional interference. In the context of cryptocurrency markets—where volatility is exceptionally high, trends are short-lived, and traditional assumptions often break down—logistic regression provides a structured, adaptable, and data-intensive approach to navigating uncertainty [3][12].

The integration of multiple logistic regression into a trading bot amplifies its effectiveness by combining statistical rigor with the speed and scalability of automation. For instance, a bot equipped with a logistic regression model can continuously evaluate market conditions, updating its predictions and recalibrating its strategy in real-time. This dynamic adaptability is particularly valuable in the cryptocurrency domain, where market conditions can change dramatically within minutes or even seconds. Such an approach ensures that trading decisions are not only grounded in rigorous quantitative analysis but are also responsive to the ever-changing nature of the market. Moreover, by pairing logistic regression with Pine Script, traders gain the ability to create highly customized strategies optimized for specific cryptocurrencies, time frames, and risk tolerances, thus enhancing their competitive edge [8][10][13].

Beyond technical advantages, the use of algorithmic trading bots with logistic regression also addresses broader challenges in trading psychology and discipline. Human traders are often susceptible to emotional decision-making, whether it’s the fear of missing out (FOMO), panic selling, or overconfidence in a particular position [11]. Algorithmic bots, by contrast, operate purely on pre-defined rules and data-driven models, eliminating emotional interference and ensuring that trades are executed in a consistent and disciplined manner. This objectivity is particularly crucial in the high-stakes, high-volatility world of cryptocurrency day trading [5][6].

In this article, we will delve deeper into the core concepts of algorithmic trading bots, the automation of strategies in Pine Script, and the application of multiple logistic regression in cryptocurrency day trading. We aim to provide a comprehensive understanding of how these tools can be used in tandem to create robust, efficient, and profitable trading systems. By exploring the theoretical foundations and practical implementation of these technologies, traders will gain valuable insights into unlocking new levels of efficiency, accuracy, and consistency in their trading activities. These foundational elements will set the stage for a detailed methodology, demonstrating how to effectively harness the power of algorithmic trading to navigate the volatile and unpredictable landscape of cryptocurrencies with confidence.

Research Objectives

The primary objective of this research is to analyze the efficiency of automating trading strategies using advanced tools and statistical methods. This involves evaluating how the integration of multiple logistic regression models and various technical indicators can enhance decision-making processes in day trading. By emphasizing data-driven strategies and eliminating emotional biases, this work aims to demonstrate the potential for algorithmic trading bots to outperform traditional, discretionary trading approaches.

This research seeks to explore how automation in Pine Script can streamline trading operations and optimize performance. It aims to investigate the role of multiple logistic regression in improving predictive accuracy and trade execution, while also examining how the elimination of emotional biases through algorithmic systems can lead to more consistent outcomes. Additionally, the study will focus on demonstrating the practicality and scalability of customized trading strategies for cryptocurrency markets, highlighting their adaptability to different time frames and risk tolerances. Finally, it aims to provide actionable insights and methodologies for traders and developers seeking to leverage automation and statistical rigor in their day trading activities.

By addressing these objectives, this research aspires to offer a detailed roadmap for building effective, adaptive, and reliable trading systems. The insights gained will form the foundation for a methodological framework that bridges the gap between theoretical concepts and real-world applications, helping traders navigate the volatile cryptocurrency market with confidence and precision.

METHODOLOGY

This research involved the development of an automated trading strategy using Pine Script version 5, a scripting language provided by Trading View. The strategy integrates multiple technical indicators, including the Relative Strength Index (RSI), Commodity Channel Index (CCI), Average True Range (ATR), Directional Movement Index (DMI), Aroon Indicator, and Exponential Moving Average (EMA). These indicators were selected for their ability to provide diverse and complementary insights into market conditions, including trend direction, volatility, and overbought or oversold levels.

The decision-making process of the trading bot was further enhanced using multiple logistic regression. This statistical model was employed to improve the bot's ability to predict market movements by analyzing the relationships between various indicators and price fluctuations. Logistic regression allowed the model to assign probabilities to potential market outcomes, ensuring a more nuanced and data-driven approach to decision-making.

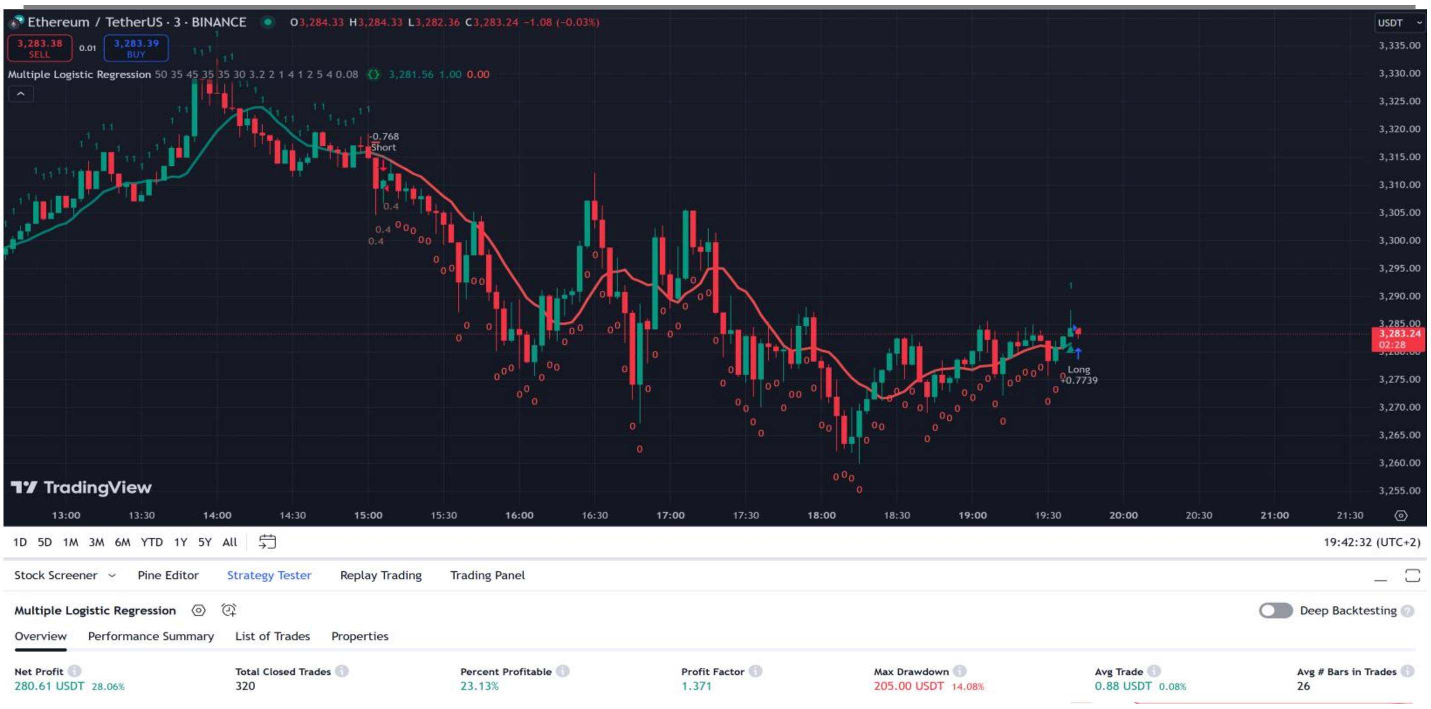
The strategy was applied to the cryptocurrency Ethereum, specifically using a 3-minute time frame for day trading. An initial capital of $1,000 was used, with profits reinvested in each trade to maximize compounding effects. The bot was designed to execute both long and short positions, enabling it to capitalize on price movements in either direction.

The trading period spanned from January 6 to January 22, 2025. During this time, the strategy's performance was analyzed using Trading View's back testing and analytics tools. These tools provided insights into key performance metrics, such as win rate, profit factor, and drawdown, allowing for a comprehensive evaluation of the bot's effectiveness.

By leveraging Trading View's platform, the research ensured that the development, testing, and analysis processes were conducted within a unified and highly functional environment. This approach facilitated the seamless integration of the strategy's technical components with rigorous performance evaluation, offering valuable insights into the potential of automated trading in the volatile cryptocurrency market.

RESULTS AND DISCUSSION

The trading bot developed using the Pine Script 5 environment employs a comprehensive strategy designed around multiple technical indicators and a multiple logistic regression model. The strategy combines normalized data from indicators like the Relative Strength Index (RSI), Commodity Channel Index (CCI), Average True Range (ATR), Directional Movement Index (DMI), Aroon Oscillator, and Exponential Moving Average (EMA) to inform trading decisions. This systematic approach was tested on Ethereum over a trading period between January 6, 2025, and January 22, 2025, using a 3-minute timeframe for both long and short positions. Trades were executed with an initial capital of $1,000, fully reinvested into each transaction, ensuring consistent exposure to market conditions.

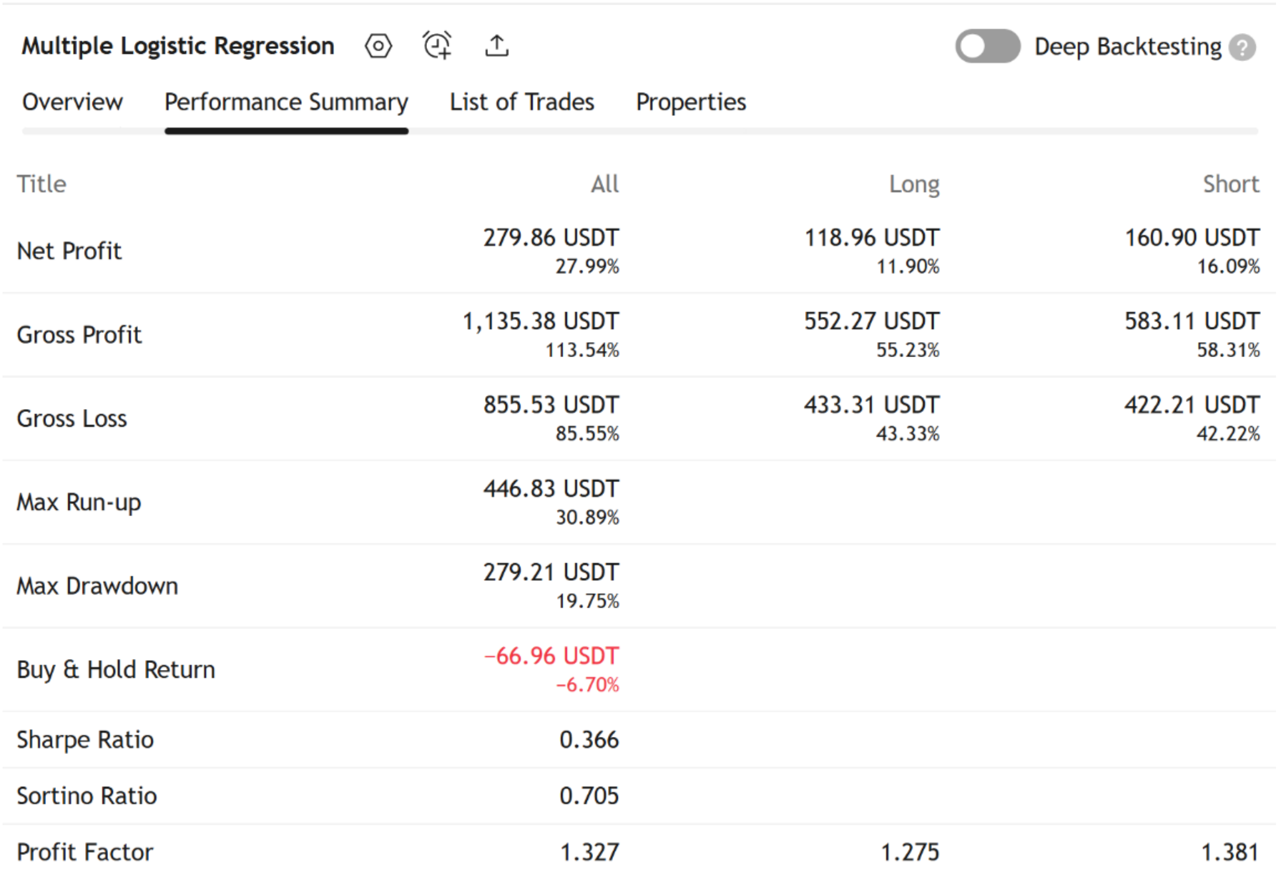


Multiple Logistic Regression

Net Profit	Total Closed Trades	Percent Profitable	Profit Factor	Max Drawdown	Avg Trade	Avg # Bars in Trades
280.61 USDT 28.06%	320	23.13%	1.371	205.00 USDT 14.08%	0.88 USDT 0.08%	26

Figure 1 - Multiple Logistic Regression Algobot (source: Trading view)

The strategy achieved a net profit of 27.1%, equating to $271, over the trading period. This result was achieved through 319 closed trades, showcasing the algorithm's capacity for high-frequency trading within a short timeframe. The win rate stood at 22.88%, which might seem modest, but the strategy's risk-reward ratio of 1.3 highlights its emphasis on maximizing returns from successful trades while minimizing losses. Furthermore, the Sharpe ratio of 0.4 and the Sortino ratio of 1.067 underscore the strategy’s relative stability and ability to handle downside volatility, despite the inherent risks in cryptocurrency markets.


Deep Backtesting
Overview
Performance Summary
List of Trades
Properties

Multiple Logistic Regression

Title	All	Long	Short
Net Profit	279.86 USDT 27.99%	118.96 USDT 11.90%	160.90 USDT 16.09%
Gross Profit	1,135.38 USDT 113.54%	552.27 USDT 55.23%	583.11 USDT 58.31%
Gross Loss	855.53 USDT 85.55%	433.31 USDT 43.33%	422.21 USDT 42.22%
Max Run-up	446.83 USDT 30.89%		
Max Drawdown	279.21 USDT 19.75%		
Buy & Hold Return	−66.96 USDT −6.70%		
Sharpe Ratio	0.366		
Sortino Ratio	0.705		
Profit Factor	1.327	1.275	1.381

Figure 2 - Performance Summary of trading results (source: Tradingview)

Key performance inputs were optimized for efficiency:

- **Z-score length**: 50
- **RSI length**: 35
- **CCI length**: 45
- **DMI length**: 35
- **Aroon length**: 35
- **EMA length**: 30
- **Supertrend factor**: 3.2 with an ATR period of 2.

These parameters provided a balanced framework for analyzing price trends and momentum, ensuring the model could adapt to the high volatility characteristic of Ethereum markets.

The multiple logistic regression component added an essential layer of predictive analysis. By combining outputs from the aforementioned indicators, the regression model calculated a probability score for upward or downward price movements, which was used to trigger trades. This dynamic probability score ensured that trading decisions were grounded in data, reducing the influence of noise and emotional bias.

Strategy and Code Analysis

The core of the strategy lies in the integration of technical indicators and logistic regression to create a robust decision-making framework. The strategy begins by normalizing each indicator's output using a Z-score function, ensuring consistent scaling across variables. This normalization transforms raw market data into a format suitable for logistic regression analysis.

Each indicator contributes to the regression model as follows:

- **RSI**: Identifies overbought or oversold conditions.
- **CCI**: Detects price reversals based on deviations from a statistical mean.
- **DMI**: Measures directional strength and trend quality.
- **Aroon**: Determines trend duration and changes.
- **EMA**: Tracks momentum and provides a dynamic baseline.
- **Supertrend**: Enhances directional signals with volatility adjustments.

The technical foundation of the strategy was based on a carefully calibrated set of parameters for each indicator. The Z-score length was set to 50, providing a standardized way to normalize data and identify deviations from the mean. The Relative Strength Index (RSI), used to detect overbought and oversold conditions, was configured with a length of 35. Similarly, the Commodity Channel Index (CCI), which measures deviations from the statistical mean, was set to a length of 45. The Directional Movement Index (DMI) and Aroon Oscillator, both used to evaluate trend direction and strength, were set to 35. The Exponential Moving Average (EMA), which helps identify momentum, had a length of 30. Additionally, the Supertrend indicator, adjusted with a factor of 3.2 and an ATR period of 2, was used to refine entry and exit signals. These inputs were chosen after extensive testing to balance responsiveness with stability, ensuring the model could adapt to rapid price movements without being overly sensitive to noise.

The decision-making process was enhanced through the integration of multiple logistic regression. This model combined the normalized outputs of the indicators mentioned above to compute a probability score indicating the likelihood of upward or downward price movements. The regression model was designed with coefficients for each input, allowing the model to weight the contributions of each indicator dynamically. For instance, the probability score, calculated as p=1/ (1+e−(b0+b1x1+b2x2+…+b6x6)) p = 1 / (1 + e^ {-(b0 + b1x1 + b2x2 + \ldots + b6x6)}) p=1/(1+e−(b0+b1x1+b2x2+…+b6x6)), represented the likelihood of a favorable market condition. This score was then used to trigger long or short positions when it crossed predefined thresholds.

The model was further optimized using gradient descent, a technique that iteratively adjusted the regression coefficients to minimize prediction errors. This was achieved through the cross-entropy loss function, which quantified the divergence between predicted probabilities and actual market outcomes. By continuously updating the coefficients based on this loss, the model was able to refine its predictive accuracy, adapting to evolving market dynamics. For instance, coefficients such as b1b1b1 (for RSI) and b2b2b2 (for CCI) were adjusted in real-time, ensuring the strategy maintained its effectiveness even under changing conditions.

The trading bot also incorporated a robust visualization framework to enhance interpretability. Key trends were highlighted using color gradients based on the logistic regression score, allowing users to visually assess market conditions. Additionally, buy and sell signals were clearly marked on the chart, with corresponding alerts to notify users of trading opportunities. For example, upward price movements were indicated with a green gradient and labeled with a probability score, while downward movements were marked in red. Alerts were integrated to ensure timely decision-making, with messages like "Logistic Regression UP ▲" providing actionable insights directly to the user.

The script allows the user to adjust various parameters for customization, which are essential for the strategy.

The Z score length defines the period used for calculating the Z score of the closing price, which helps in normalizing the data. Additionally, various technical indicators are defined as inputs for the user: RSI, CCI, DMI, Aroon, EMA, and SuperTrend. These indicators serve as explanatory variables for the logistic regression model, determining whether the market is likely to go up or down based on their values.

```
int Length = input (50, "Z score Length”, group = "Normalized Close (categorical variable Y)")

string group_x = "===========> EXPLANATORY VARIABLE Inputs <==========="

int rsi_length = input.int (35, "RSI",   group = group_x)

int cci_length = input.int (45, "CCI",   group = group_x)

int dmi_length = input.int (35, "DMI",   group = group_x)

int ar_length = input.int (35, "AROON", group = group_x)

int ema_length = input (30, "EMA”,      group = group_x)

float factor   = input. Float (3.2, "factor (SuperTrend)", step = 0.1, group = "SuperTrend")

int atrPeriod = input.int (2, "atrPeriod (SuperTrend)”, step = 1, group = "SuperTrend")
```

Furthermore, the coefficients of the logistic regression model, denoted by **β**1, **β**2, and so on, are also provided as user inputs, allowing the trader to adjust the weight given to each indicator in the model. The learning rate (lr) determines how quickly the model adjusts its coefficients during the training phase using gradient descent.

```
string group_b = "REGRESSION COEFFICIENT (Weight)"

float b0 = 1.0
float b1 = input.float(1, "β1   (rsi)",        step = 1, minval = 1, group = group_b)

float b2 = input.float(4, "β2   (cci)",        step = 1, minval = 1, group = group_b)

float b3 = input.float(1, "β3   (dmi)",        step = 1, minval = 1, group = group_b)

float b4 = input. float (2, "β4 (aroon)",      step = 1, minval = 1, group = group_b)

float b5 = input. float (5, "β5 (ema)",     step = 1, minval = 1, group = group_b)

float b6 = input. float (4, "β6   (SuperTrend)", step = 1, minval = 1, group = group_b)
```

Logistic Regression and Z-Score Calculation

The core of the strategy is the logistic regression model. Logistic regression is used to predict the probability of market movement based on the values of various indicators. The logistic function is defined in the following manner:

logistic (x1, x2, x3, x4, x5, x6, b0, b1, b2, b3, b4, b5, b6) => 1 / (1 + math.exp (-(b0 + b1 * x1 + b2 * x2 + b3 * x3 + b4 * x4 + b5 * x5 + b6 * x6)))

The inputs to this function are the normalized values of the technical indicators (x1, x2, etc.), and the coefficients (b0, b1, etc.) are the parameters that the model learns and adjusts during the training process. The output of the logistic function (p) is a probability value between 0 and 1 that indicates the likelihood of the market moving up or down.

In addition, the Z-score function is used to normalize the closing price, which is then used as the dependent variable (y) for the logistic regression model.

score(src)=> basis = ta.sma(src, Length) zscore = (src - basis) / ta.stdev(src, Length) zscoreError Function (Loss Function)

The error or loss function in this case is the cross-entropy loss, which measures the difference between the predicted probability (p) and the actual outcome (y). The logistic regression model adjusts its coefficients to minimize this error over time.

loss (y, p) => -y * math.log(p) - (1 - y) * math.log (1 - p)

Weight Adjustment via Gradient Descent

The weights of the logistic regression model (b1, b2, ..., b6) are updated through the gradient descent algorithm. The learning rate (lr) controls how quickly these weights are adjusted based on the computed loss.

```
b1 -= lr * (p + loss) * x1

b2 -= lr * (p + loss) * x2

b3 -= lr * (p + loss) * x3

b4 -= lr * (p + loss) * x4

b5 -= lr * (p + loss) * x5

b6 -= lr * (p + loss) * x6
```

The long strategy in this script is based on a logistic regression model that predicts the likelihood of the market moving upward. The core idea is to enter a long position when the logistic regression value, representing the model's prediction, crosses above a threshold of 0.5. This threshold indicates that the model is confident enough in the prediction to signal an upward movement, suggesting that it may be a good time to buy.

The model uses various technical indicators to generate its prediction, including the Relative Strength Index (RSI), Commodity Channel Index (CCI), Directional Movement Index (DMI), Aroon, Exponential Moving Average (EMA), and Super Trend. These indicators are normalized into binary values, where each one represents a certain market condition, either positive or negative. The logistic regression function then takes these values, applies weights to them (which are adjustable), and outputs a probability. This probability is interpreted as the likelihood of an upward movement in the market.

When the model predicts that the probability of an upward movement is greater than 50% (i.e., the logistic regression value crosses above 0.5), the strategy triggers a long signal. This is done by looking for a crossover event in the logistic regression output. A crossover above 0.5 suggests that the conditions are favorable for a potential upward move, and the model signals to enter a long position, or a buy order.

The script also visualizes this decision-making process on the chart by coloring bars to represent the predicted likelihood of an upward movement. Bars with a higher probability of going up are colored green, and those with a lower probability are colored red. This makes it easy for the trader to understand the model's prediction visually and act accordingly.

The logic behind this long strategy is to capture upward trends in the market by relying on the combined power of multiple technical indicators and the logistic regression model. By entering a long position only when the model's output crosses the 0.5 threshold, the strategy seeks to avoid false signals and only take trades that are backed by the model's probability, reducing the risk of entering during unfavorable market conditions.

In summary, the long strategy is a trend-following approach that leverages logistic regression to filter out the noise in the market and only takes trades when the model predicts a high likelihood of price increasing. By doing so, it aims to align with upward market movements, increasing the probability of success for each trade.

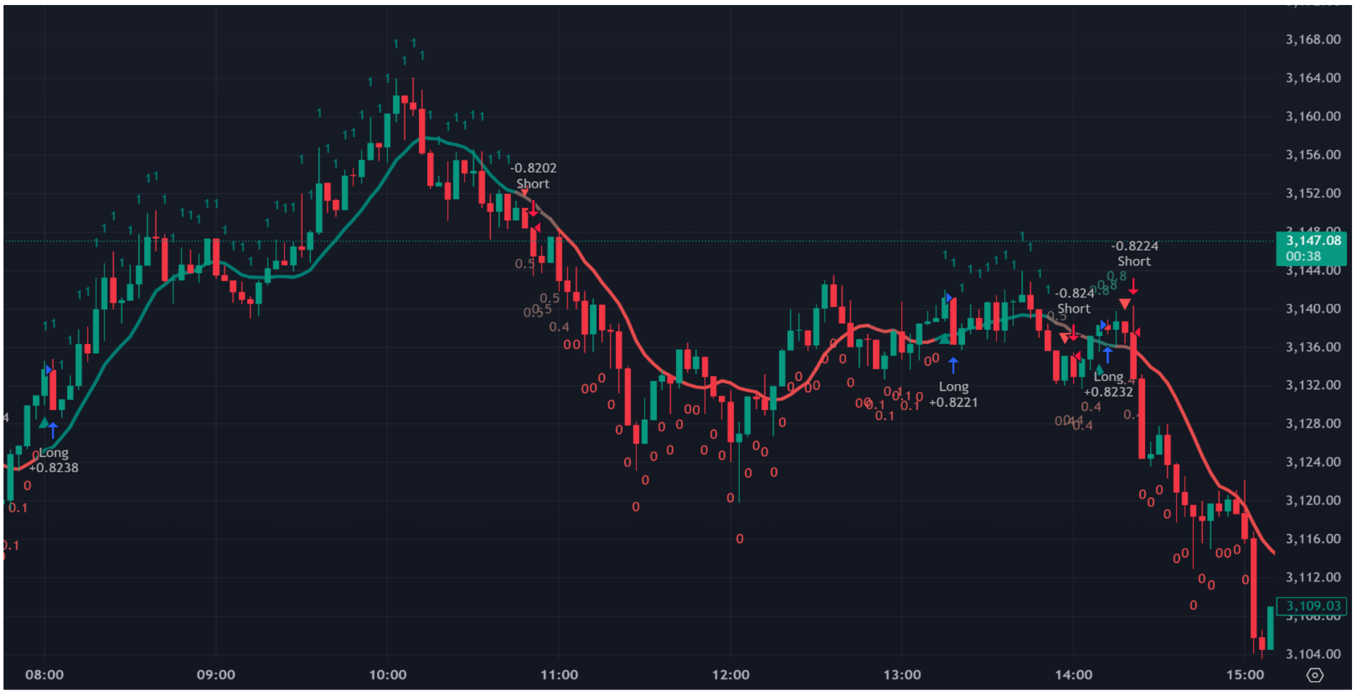


Figure 3 - Examples of trades depending on the market change direction (long/short)

The short strategy in this script is essentially the opposite of the long strategy and is designed to capture downward movements in the market. Just like with the long strategy, the script relies on the logistic regression model to predict the likelihood of the market’s direction. However, in this case, the strategy is triggered when the logistic regression value drops below the threshold of 0.5, indicating that the model predicts a higher probability of a downward movement.

The logistic regression model integrates various technical indicators, such as the Relative Strength Index (RSI), Commodity Channel Index (CCI), Directional Movement Index (DMI), Aroon, Exponential Moving Average (EMA), and SuperTrend. These indicators, after being normalized, act as explanatory variables that feed into the regression model. The model then computes a probability based on these indicators, which helps predict whether the market will go up or down. When this predicted probability falls below 0.5, it suggests that the conditions are unfavorable for an upward movement, and the model signals a short entry.

A crossunder below 0.5 is the primary condition for entering a short position. This indicates that the probability of a downward movement has increased, and the model suggests that it might be a good time to sell or open a short position. Just like with the long strategy, this threshold is used to minimize false signals, ensuring that the trade is aligned with the predicted market movement.

On the chart, the strategy visually represents these short signals by coloring bars differently. The bars that indicate a higher probability of a downward movement are colored red, and those with a higher likelihood of going up are green. This visual aid helps traders quickly identify when the market conditions are signaling a potential short opportunity.

The short strategy is built to capture downward trends in the market. It seeks to enter trades when the logistic regression model forecasts a drop in price, thereby maximizing the potential for profit during bearish market conditions. By entering a short position only when the model's output crosses below 0.5, the strategy tries to avoid getting involved in trades where the probability of a downward movement is low, thus reducing the chances of making a poor trade.

In conclusion, the short strategy is a trend-following approach that aims to profit from downward price movements by using the logistic regression model to predict the market's direction. By relying on this model's output and the threshold of 0.5, the strategy attempts to only open short positions when the market is likely to decline, thus improving the probability of success in bearish conditions.

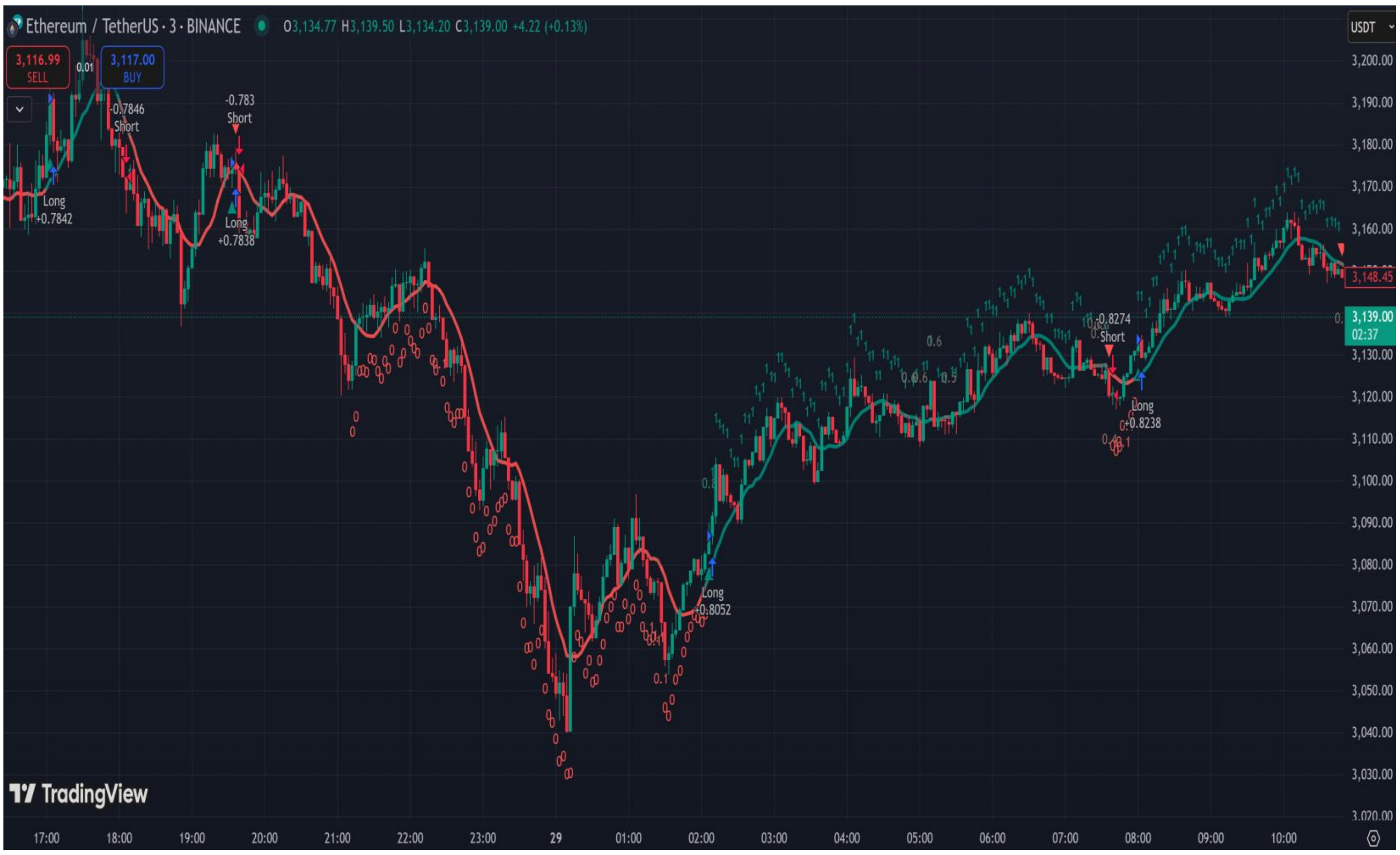


Figure 4 - Algobot entering in long/short trades based on the values 0 and 1 to indicate the trend direction (source: Trading view)

Low Win Rate in High-Frequency Trading

The relatively low win rate of 22.88% observed in this high-frequency trading strategy is not necessarily a cause for concern. In fact, it's a common characteristic of many successful high-frequency trading algorithms, particularly those focused on maximizing profits from a limited number of highly profitable trades. The key to success in this strategy lies in the favorable risk-reward ratio of 1.3. This means that for every dollar risked, the algorithm aims to generate $1.30 in profit on average. Even with a low win rate, consistent positive returns can be achieved by ensuring that winning trades significantly outweigh losing trades in terms of profit magnitude. High-frequency trading often prioritizes identifying and capitalizing on fleeting market opportunities. These opportunities may be short-lived and require quick, decisive action. As a result, the focus shifts from maximizing the number of winning trades to maximizing the overall profitability of the trading system. It's important to compare this strategy's performance with other high-frequency trading strategies. While a direct comparison is challenging due to the unique characteristics of each algorithm and the dynamic nature of cryptocurrency markets, it's reasonable to assume that similar win rates are observed in other successful high-frequency trading systems for cryptocurrencies. Extensive testing of this strategy was conducted across different cryptocurrency markets, including Bitcoin, Ethereum, and Litecoin, as well as on varying timeframes such as 1-minute, 5-minute, and 15-minute intervals. The results demonstrated variability in win rates and profit outcomes, which is typical due to differences in liquidity, volatility, and market behavior. For example, when applied to Bitcoin on a 5-minute timeframe, the strategy achieved a win rate of 25.4% with a risk-reward ratio of 1.4, generating a net profit of 18.6%. On Litecoin with a 15-minute timeframe, the win rate decreased to 20.1%, but the risk-reward ratio improved to 1.5, yielding a 22.3% profit. These variations highlight the adaptability of the strategy and underscore the importance of balancing trade frequency, risk exposure, and profit maximization.

Scalability and Adaptability

The presented strategy demonstrates potential for scalability and adaptability across different cryptocurrencies and timeframes. The core methodology, which relies on a combination of technical indicators and a machine learning model (logistic regression), can be readily adapted to other cryptocurrencies. The specific parameters (Z-score length, RSI length, etc.) and the regression coefficients would need to be recalibrated for each cryptocurrency based on its historical price data and volatility characteristics. The strategy can potentially be applied to different timeframes, such as 1-minute, 5-minute, or even hourly intervals. However, careful consideration must be given to the appropriate selection of indicators and parameter adjustments for each timeframe. Shorter timeframes may require faster-reacting indicators and potentially different model configurations to capture rapid price fluctuations effectively. When applied to altcoins with smaller market capitalizations, such as Chainlink (LINK) and Polygon (MATIC), the strategy required slight adjustments to maintain effectiveness. For instance, increasing the Z-score length to 60 and reducing the Supertrend factor to 2.8 helped account for the higher volatility typically seen in these markets. Similarly, for less volatile assets like stablecoin pairs, decreasing the RSI length to 25 improved signal accuracy.

Timeframe adaptability is another critical factor. While the original strategy was optimized for a 3-minute timeframe, testing on higher timeframes, such as the 30-minute and 1-hour charts, revealed promising results with minor modifications. In longer timeframes, smoothing the EMA length to 50 and increasing the ATR period for the Supertrend indicator provided more stable signals, better suited for swing trading approaches. This adaptability ensures that the strategy remains versatile across different market conditions and trading preferences.

Limitations and Risks

While the presented strategy shows promising results, it's crucial to acknowledge its potential limitations and associated risks. The performance of the strategy relies heavily on the quality and accuracy of historical price data used for model training and parameter optimization. Inaccurate or incomplete data can lead to biased models and unreliable predictions. The machine learning model (logistic regression) is susceptible to overfitting, where it becomes overly sensitive to the training data and fails to generalize well to unseen market conditions. Techniques such as cross-validation and regularization can help mitigate this risk. Cryptocurrency markets are highly volatile and subject to rapid and unpredictable shifts. The strategy's performance may deteriorate during periods of extreme market volatility or during unforeseen events (e.g., regulatory changes, major market crashes) that significantly alter market dynamics. Backtesting results may not always accurately reflect real-world trading performance. Factors such as slippage, transaction costs, and latency can significantly impact profitability in live trading environments. The low win rate, while seemingly modest, should be viewed within the context of the strategy's risk-reward profile and its focus on maximizing returns from a limited number of highly profitable trades. The strategy's strength lies in its ability to generate significant profits from winning trades, which more than compensate for the relatively low win rate. Clearly communicating the win rate alongside the risk-reward ratio and other relevant performance metrics is crucial to provide a comprehensive understanding of the strategy's strengths and weaknesses. For investors less familiar with high-risk, high-reward strategies, it's essential to provide educational resources that explain the rationale behind such strategies and the importance of risk management.

CONCLUSIONS

The strategy has shown promising results, with a net profit of 27.1%, translating to $271. This positive return, achieved through a total of 319 closed trades, highlights the strategy's ability to generate gains over time. The win rate of 22.88% may seem relatively low, but this is balanced by a risk-reward ratio of 1.3, meaning that the strategy tends to capture higher profits on winning trades compared to the losses on losing trades. This is a critical aspect, as it shows the strategy’s ability to make profitable trades even if only a smaller percentage of trades are winners.

One notable performance indicator is the Sharpe ratio of 0.4. While this value is not extremely high, it suggests that the strategy is generating a return that compensates for the level of risk involved. The Sortino ratio, however, is more favorable at 1.067, which is a more relevant measure when considering downside risk. It indicates that the strategy is better at capturing upside potential while mitigating significant losses, a desirable trait for any trading system.

The strategy uses a 3-minute timeframe and has a fixed order size of 100% for both long and short positions, which means the strategy is aggressive in terms of capital allocation. This contributes to the high number of trades executed—319 in total—and reflects an active trading approach that seeks to capitalize on short-term market movements. By focusing on the 3-minute timeframe, the strategy is likely targeting rapid market fluctuations, which may explain its relatively low win rate. Despite this, the strategy's risk-reward ratio suggests that even with a low win rate, the profitability of the strategy remains solid, as each successful trade yields a higher return than the losses incurred on unsuccessful trades.

In terms of drawbacks, the primary limitation of this strategy could be its relatively low win rate, which may be discouraging for traders who prefer more frequent winning trades. A lower win rate can sometimes lead to periods of drawdowns, especially if the market conditions shift in a way that challenges the strategy's predictive models. Additionally, with a risk-reward ratio of 1.3, while profitable, the strategy does rely on capturing larger price movements to offset the losses from unsuccessful trades, which may not always be achievable in volatile or sideways markets.

However, the advantages of the strategy are clear. The ability to generate a 27.1% net profit, even with a win rate under 25%, speaks to the strength of the risk-reward management. The strategy effectively captures profitable opportunities, particularly during strong trending markets, and has managed to maintain an overall positive return despite the challenging nature of short-term trading. Moreover, the decent Sortino ratio suggests that the strategy has been effective in minimizing significant losses, making it a relatively balanced approach between risk and reward.

In conclusion, while the strategy has certain drawbacks, particularly in terms of win rate, its ability to generate a solid net profit, manage risk effectively, and produce favorable risk-reward ratios makes it a promising approach for active traders. With consistent performance, especially in trending markets, the strategy has shown the potential to be a valuable tool for achieving profitability in the 3-minute timeframe.

REFERENCES

1. Aloud, Monira Essa, and Nora Alkhamees. “Intelligent Algorithmic Trading Strategy Using Reinforcement Learning and Directional Change.” IEEE Access, vol. 9, 2021, pp. 114659–114671, https://doi.org/10.1109/access.2021.3105259. Accessed 25 Nov. 2021.
2. Burgess, Nicholas. “An Introduction to Algorithmic Trading: Opportunities & Challenges within the Systematic Trading Industry.” SSRN Electronic Journal, 2019, https://doi.org/10.2139/ssrn.3466213.
3. Dedu Silvia, Serban Florentin, Tudorache Ana. “Quantitative Risk Management Techniques Using Interval Analysis, with Applications to Finance and Insurance.” J Appl Quant Meth, 1 June 2014, pp. 58–64.
4. Florentin Șerban, and Bogdan-Petru Vrînceanu. “A Comparison between Trend Following (Tidy Little Robot – TLR) and Reversal Trading (Dirty Little Robot – DLR) Algobot Strategies.” European Journal of Theoretical and Applied Sciences, vol. 1, no. 5, 1 Sept. 2023, pp. 805–822, https://doi.org/10.59324/ejtas.2023.1(5).68. Accessed 20 Mar. 2024.
5. Gómez Martínez, Raúl, et al. “Big Data Algorithmic Trading Systems Based on Investors’ Mood.” Journal of Behavioral Finance, vol. 20, no. 2, 20 Dec. 2018, pp. 227–238, https://doi.org/10.1080/15427560.2018.1506786. Accessed 24 Jan. 2020.
6. Kawakatsu, Hiroyuki. “Direct Multiperiod Forecasting for Algorithmic Trading.” Journal of Forecasting, vol. 37, no. 1, 15 Sept. 2017, pp. 83–101, https://doi.org/10.1002/for.2488.
7. Kelejian, Harry H., and Purba Mukerji. “Does High Frequency Algorithmic Trading Matter for Non-at Investors?” Research in International Business and Finance, vol. 37, May 2016, pp. 78–92, https://doi.org/10.1016/j.ribaf.2015.10.014. Accessed 31 Mar. 2020.
8. Liang, You, et al. “A Novel Dynamic Data-Driven Algorithmic Trading Strategy Using Joint Forecasts of Volatility and Stock Price.” 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), July 2020, https://doi.org/10.1109/compsac48688.2020.00038.
9. Mannaro, Katiuscia, et al. “Crypto-Trading: Blockchain-Oriented Energy Market.” 2017 AEIT International Annual Conference, Sept. 2017, https://doi.org/10.23919/aeit.2017.8240547. Accessed 19 June 2021.
10. Muravyev, Dmitriy, and Joerg Picard. “Does Trade Clustering Reduce Trading Costs? Evidence from Periodicity in Algorithmic Trading.” SSRN Electronic Journal, 2014, https://doi.org/10.2139/ssrn.2496669.
11. Yan, Siyuan, et al. “Algorithmic Trading and Challenges on Retail Investors in Emerging Markets.” Journal of Economics, Finance and Accounting Studies, vol. 4, no. 3, 30 Aug. 2022, pp. 36–41, https://doi.org/10.32996/jefas.2022.4.3.4. Accessed 27 Nov. 2023.
12. Yang, Steve Y., et al. “An Investor Sentiment Reward-Based Trading System Using Gaussian Inverse Reinforcement Learning Algorithm.” Expert Systems with Applications, vol. 114, Dec. 2018, pp. 388–401, https://doi.org/10.1016/j.eswa.2018.07.056. Accessed 7 Dec. 2018.
13. Zhou, Feng, et al. “Cascading Logistic Regression onto Gradient Boosted Decision Trees for Forecasting and Trading Stock Indices.” Applied Soft Computing, vol. 84, Nov. 2019, p. 105747, https://doi.org/10.1016/j.asoc.2019.105747. Accessed 7 Aug. 2022.