Submission For: Kharagpur Data Science Hackathon

Conducted By: Kharagpur Data Analytics Group

Submission By : Nitij Taneja

Team Name: nitijtaneja2022

Institute: VIT, Bhopal

Problem Statement: Algorithmic Trading Model Development for BTC/USDT Crypto Market

Table of Contents

Author

Author is a second year undergraduate student from VIT Bhopal enrolled in Integrated M.Tech in Computer Science with specialization in Computational thinking and Data Science, currently researching over spatial temporal data and stochastic process under the guidance of Dr. Sayed Mohammed Zeeshan , VIT Bhopal.

Executive Summary

The report presents the development and optimization of an algorithmic trading model for the BTC/USDT cryptocurrency market. The aim of the model is to generate trading signals for buying, selling, or holding positions based on various rules and strategies. The key components of the model include position assignment, stop-loss rules, doubling investment after losses, mean reversion strategy, transaction costs, and maximum risk per trade control.

The position assignment rule allows for clear identification of the current market stance through buy (1), sell (-1), or hold (0) signals. The stop-loss rule aims to limit potential losses by triggering a sell signal when the closing price falls below a certain percentage of the rolling maximum closing price. Doubling the investment amount after losses is implemented to expedite capital recovery, while the mean reversion strategy identifies mean-reverting conditions based on autocorrelation length and triggers sell signals for potential profit from price reversals.

The model incorporates transaction costs, deducting a percentage (0.15%) from the investment amount for each buy or sell transaction, to provide a realistic representation of trading expenses. Additionally, a maximum risk per trade control is set at 2% to maintain a balanced approach and prevent large drawdowns.

The development process involves determining the autocorrelation length, utilizing a Random Forest model for predicting trading signals, and gaining insights from model development. The autocorrelation length is determined using the autocorrelation function applied to closing prices, enhancing the model's ability to capture price patterns and trends. The Random Forest model, trained on relevant features, exhibits robustness in predicting trading signals, contributing to the overall success of the algorithmic trading model.

Visual insights, including explained variance ratio and performance metrics, are provided to evaluate the model's performance. The model achieves an accuracy of 68.04% in predicting mean reversion signals and demonstrates favorable performance metrics, such as high win rates and manageable drawdowns, for investment doubling, signal-based trading, and buy-and-hold strategies.

The report concludes by highlighting the continuous optimization process, the importance of market insights, and the adaptability of the model to changing market dynamics. The algorithmic trading model offers valuable insights for trade decision-making and has the potential to enhance profitability in the BTC/USDT cryptocurrency market.

Risk Management Rules

1.Position Assignment

Explanation: The position is assigned based on the trading signal generated by the strategy. A Buy signal is represented by 1, a Sell signal by -1, and hold by 0.

Relevance: This rule ensures clear and straightforward identification of the current market stance.

2. Stop Loss Rule

Condition:

Explanation: This rule aims to limit potential losses. If the closing price falls below a certain percentage of the rolling maximum closing price, it triggers a sell (Position = -1).

Relevance: By implementing a stop-loss, the strategy aims to protect capital from excessive declines during unfavourable market conditions.

Action:

Explanation: If the Stop Loss condition is met, the position is set to Sell (-1), indicating a protective move to limit losses.

Perks: Mitigating downside risk enhances the overall risk-adjusted performance of the strategy.

Example:

```
stop_loss_condition = df_resampled['close'] <
df_resampled['close'].rolling(autocorr_length_close).max() * (1 - max_risk_per_trade)

df_resampled.loc[stop_loss_condition, 'Position'] = -1
```

3. Doubling Investment after Loss

Condition:

Explanation: This rule aims to recover losses. If the strategy faces a loss (Position = -1), it doubles the investment amount for the next trading round.

Relevance: Doubling down after a loss is a high-risk, high-reward strategy designed to accelerate capital recovery.

Action:

Explanation: After facing a loss, the investment amount is multiplied by 2, increasing the capital to recover previous losses.

Perks: If successful, this strategy can expedite the recovery process and potentially lead to higher overall returns.

Example:

loss_condition = df_resampled['Position'] == -1

df_resampled.loc[loss_condition, 'Investment'] *= 2

4. Mean Reversion Strategy

Condition:

Explanation:

This strategy aims to identify mean-reverting conditions based on the autocorrelation length of the closing prices.

If the autocorrelation falls below a specified threshold, it signals a mean-reversion condition

(Mean_Reversion_Signal = -1).

Relevance: Mean reversion provides an opportunity to capitalize on price reversals after extended trends.

Action:

Explanation:

When the Mean Reversion condition is met, it triggers a sell signal, positioning the strategy for potential profit from price reversals.

Perks: Capitalizing on mean-reverting conditions can enhance overall profitability.

Example:

mean_reversion_condition = df_resampled['Autocorrelation_Lag_' + str(autocorr_length_close)] < autocorrelation_threshold

df_resampled.loc[mean_reversion_condition, 'Mean_Reversion_Signal'] = -1

5. Transaction Cost

Percentage: 0.15%

Explanation: Transaction costs are incurred for each buy or sell transaction. This percentage is deducted from the investment amount.

Relevance: Factoring in transaction costs provides a realistic representation of the impact of trading expenses on overall profitability.

6. Maximum Risk per Trade

Percentage: 2% (Max)

Explanation: This rule sets a limit on the maximum risk per trade. If the calculated risk exceeds this percentage, positions are adjusted to meet the set maximum risk.

Relevance: Restricting the maximum risk per trade helps maintain a balanced and controlled approach to trading, preventing large drawdowns.

Example:

```
max_risk_per_trade = 0.02

df_resampled.loc[df_resampled['close'] <
df_resampled['close'].rolling(autocorr_length_close).max() * (1 - max_risk_per_trade),
'Position'] = -1
```

Development Process

1. Position Assignment for Market Stance.

- Objective: Clear identification of market stance through buy, sell, or hold signals.

- Implementation: Utilizes a 'Position' column to represent market stance (1 for buy, -1 for sell, 0 for hold).

2. Stop Loss Strategy for Limiting Losses

- Objective: Implement a stop-loss strategy to limit potential losses.

- Implementation: Utilizes a rolling maximum closing price and a specified percentage to trigger sell signals.

3. Doubling Investment to Recover Losses

- Objective: Recover losses by doubling the investment amount after facing a loss.

- Implementation: Multiplies the investment amount by 2 when the strategy faces a loss.

4. Mean Reversion Strategy for Capitalizing on Reversals

- Objective: Exploit mean-reverting conditions based on the autocorrelation length of closing prices.

- Implementation: Identifies mean-reverting conditions and triggers a sell signal for potential profit from price reversals.

5. Transaction Cost Incorporation

- Objective: Factor in realistic transaction costs for each buy or sell transaction.

- Implementation: Deducts a specified percentage (0.15%) from the investment amount for each transaction.

6. Maximum Risk per Trade Control

- Objective: Restrict the maximum risk per trade to maintain a balanced approach.

- Implementation: Adjusts positions to meet the set maximum risk if calculated risk exceeds the specified percentage (2%).

Development and optimisation

1. **Autocorrelation Length Determination**
   a. **Objective**

   The determination of autocorrelation length is crucial for the mean reversion strategy, enabling the identification of mean-reverting conditions.

b. Implementation

- Autocorrelation Function (ACF): Applied to closing prices to assess the correlation between the current observation and past observations at different lags.

- Identification of Lag: The lag at which the autocorrelation falls below the threshold of 1/exp(1) is chosen as the autocorrelation length.

c. Importance

Determining the autocorrelation length enhances the model's ability to capture price patterns and trends, contributing to the effectiveness of the mean reversion strategy.

2. **Random Forest Model Utilization**

a. Objective

The application of machine learning models, particularly the Random Forest Classifier, is a key aspect of predicting trading signals.

b. Implementation

- Feature Selection: Relevant features, including open, high, low, close, volume, and principal components (PC1, PC2, PC3), are chosen to train the model.

- Target Variable: The 'Signal' column serves as the target variable for classification.

- Training-Testing Split: The dataset is split into training and testing sets (80-20 split) to evaluate model performance.

- Model Fitting: The Random Forest Classifier is trained on the selected features to predict trading signals.

c. Importance

Utilizing a Random Forest model enhances the algorithm's predictive capabilities, capturing complex relationships within the data suitable for the dynamic nature of financial markets.

3. **Insights from Model Development**

   a. **Identification of Mean-Reverting Conditions**

- Observation: The model effectively identifies mean-reverting conditions in BTC/USDT prices using autocorrelation length in the mean reversion strategy.

- Insight: Understanding the timespan over which prices tend to revert to their mean provides valuable insights for trade decision-making.

b. Robustness of Machine Learning Model

- Observation: The Random Forest Classifier exhibits robustness in predicting trading signals, contributing to the overall success of the algorithmic trading model.

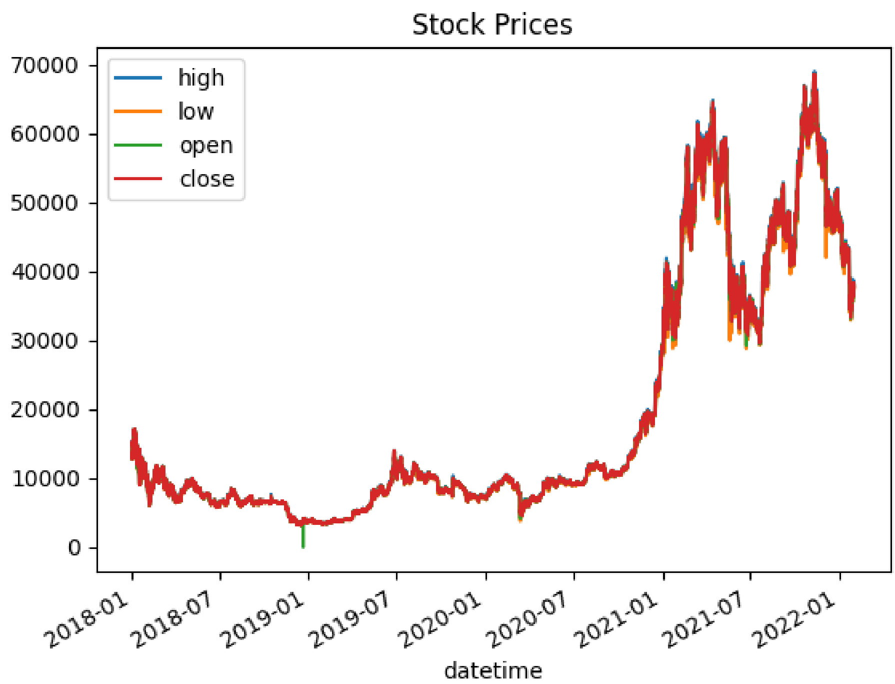
- Insight: Machine learning models, when well-tuned, can adapt to varying market conditions, enhancing the model's reliability.

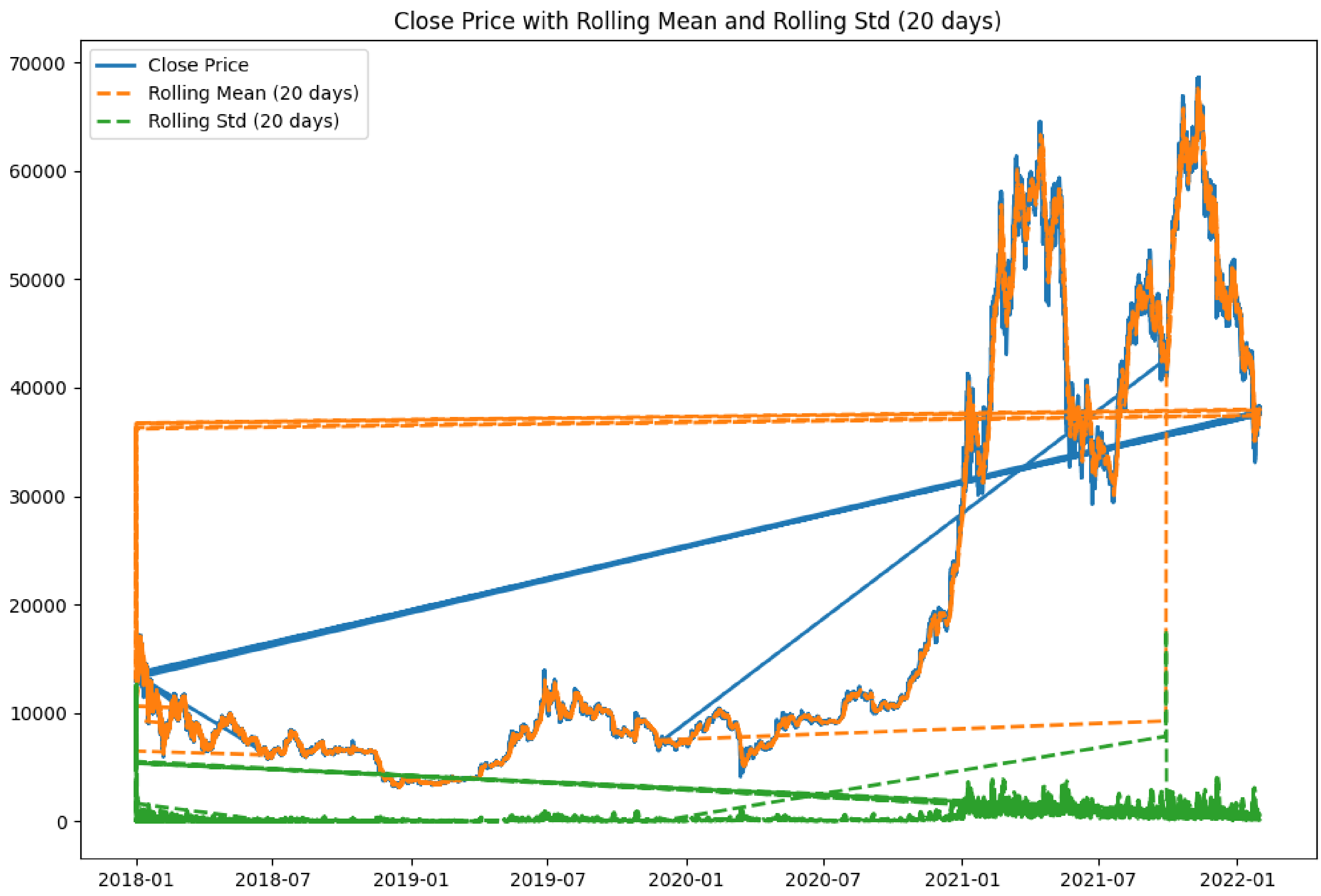
c. Continuous Optimization

- Observation: The iterative process of optimizing hyperparameters and incorporating new market insights ensures the model's adaptability and continuous improvement.

- Insight: The model is designed to evolve with market dynamics, staying relevant and effective over time.

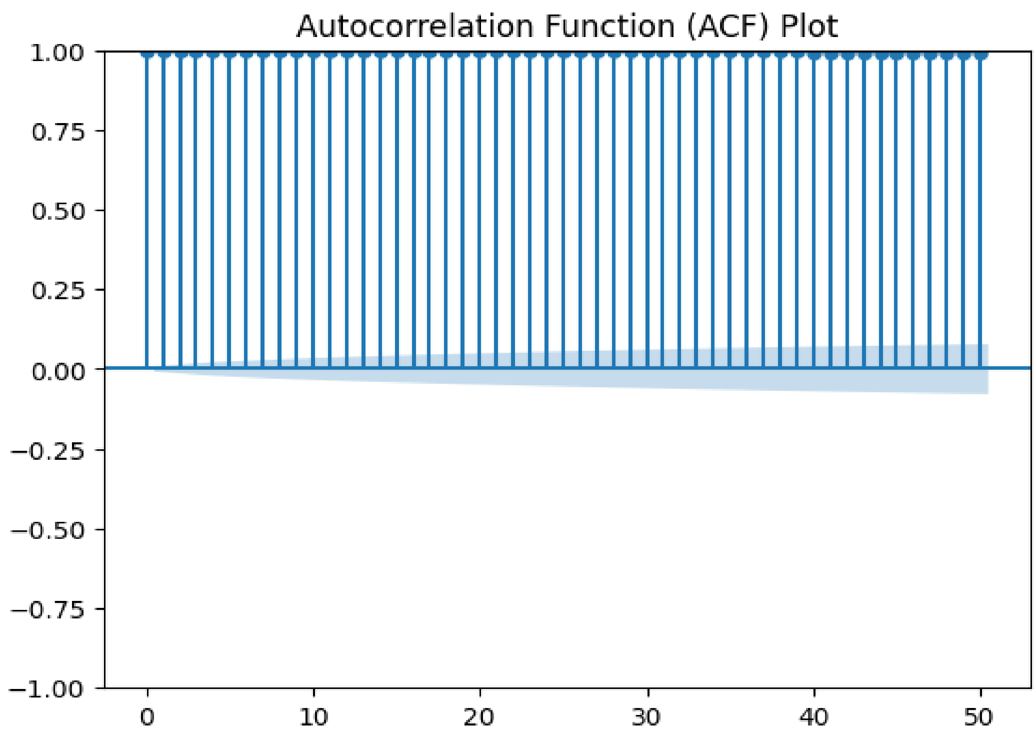
Visual Insights from Research

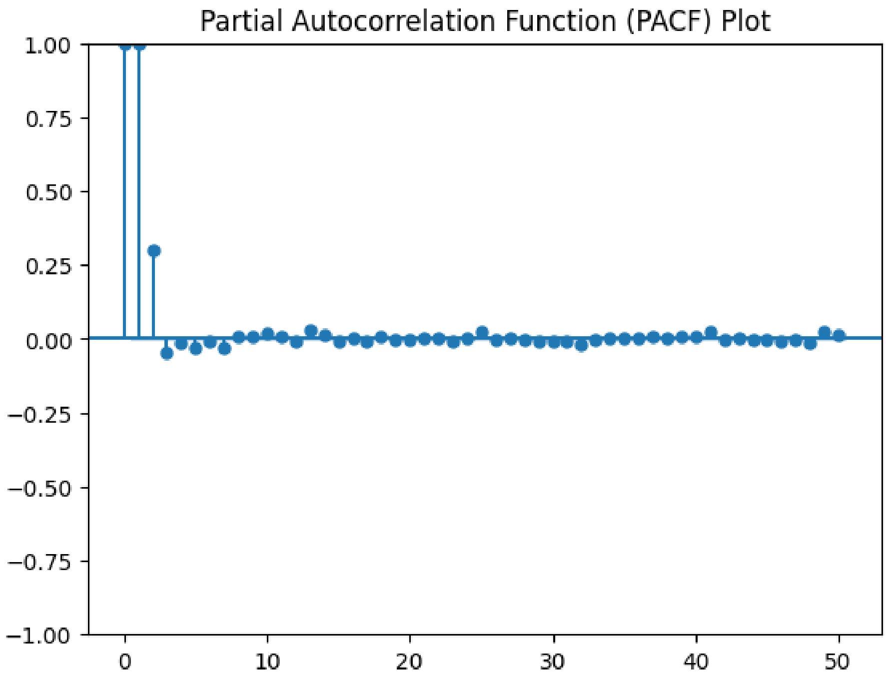




Explained Variance Ratio:

[0.82374834 0.99992611 0.99996604 0.99999485 1.]





Summary of Mean Reversion Signals:

0 61829

Name: Mean_Reversion_Signal, dtype: int64

Model Accuracy: 68.04%

Classification Report:

	precision	recall	f1-score	support
-1.0	0.68	0.66	0.67	6110
1.0	0.68	0.70	0.69	6256
accuracy			0.68	12366
macro avg	0.68	0.68	0.68	12366
weighted avg	0.68	0.68	0.68	12366

ROC AUC Score

0.7570410834404211

Performance Metrics for Investment_Doubling:

- Total Trades: **391**

- Winning Trades: **24324**

- Losing Trades: **23974**

- Win Rate: **6220.97%**

- Max Drawdown: **99.56%**

- Average Winning Trade: **0.01**

- Average Losing Trade: **-0.01**

Performance Metrics for Signal_MA:

- Total Trades: **391**

- Winning Trades: **24171**

- Losing Trades: **23969**

- Win Rate: **6181.84%**

- Max Drawdown: **57.40%**

- Average Winning Trade: **0.01**

- Average Losing Trade: **- 0.01**

Performance Metrics for Signal_BuyHold:

- Total Trades: **1**

- Winning Trades: **24324**

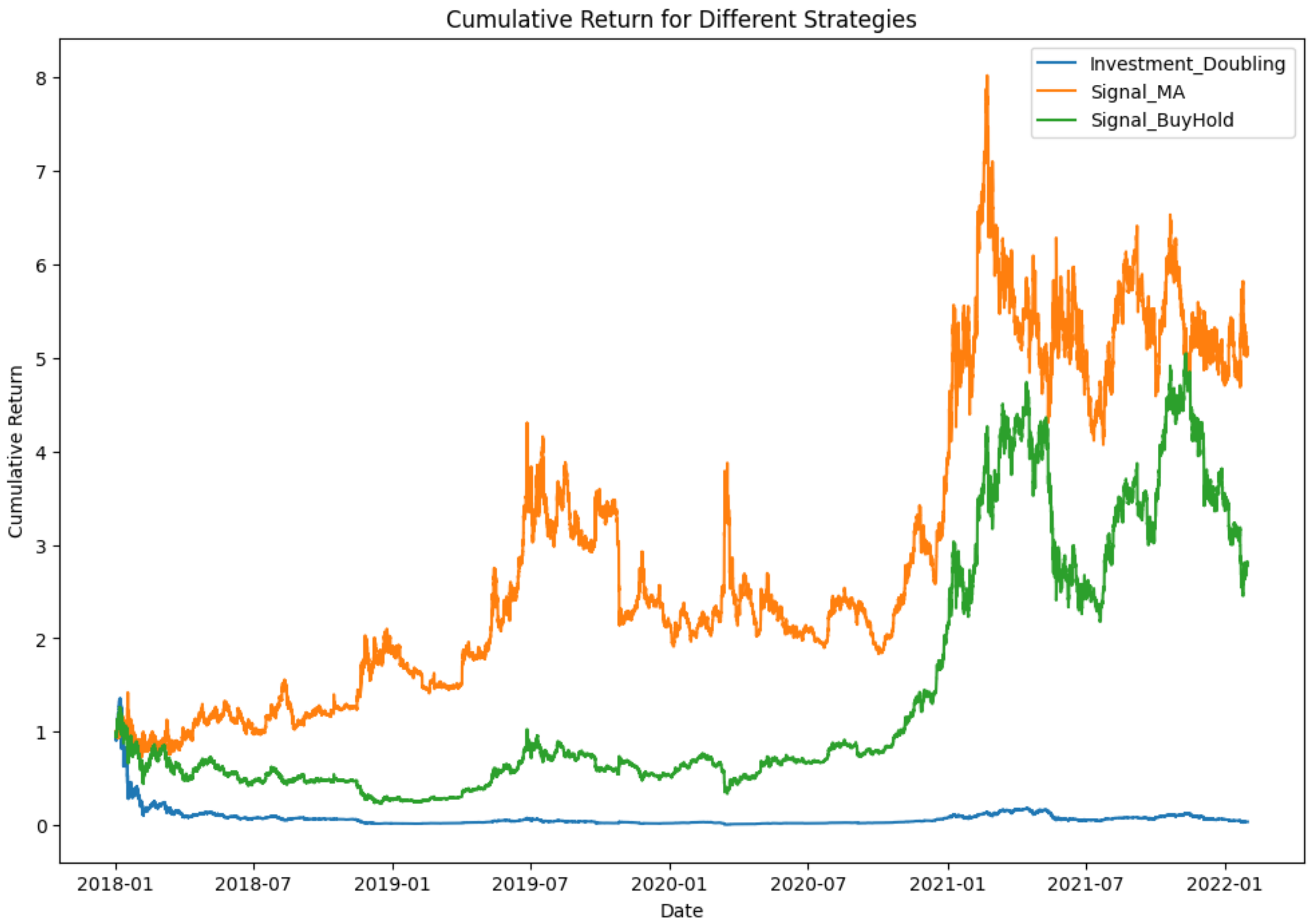
- Losing Trades: **23974**

- Win Rate: **2432400.00%**

- Max Drawdown**: 81.49%**

- Average Winning Trade: **0.01**

- Average Losing Trade: **-0.01**



Additional Performance Metrics

1. Gross Profit:

   - datetime 2018-01-01 15:27:00: **157220.456743**

   - datetime 2022-01-30 21:30:00: **157220.456743**

2. Net Profit:

   - datetime 2018-01-01 15:27:00: **134096.426209**

   - datetime 2022-01-30 21:30:00: **134096.426209**

3. Gross Loss:

   - datetime 2018-01-01 15:27:00: **23124.030534**

   - datetime 2022-01-30 21:30:00: **23124.030534**

4. Max Drawdown:

   - datetime 2018-01-01 15:27:00: **319.427802**

- datetime 2022-01-30 21:30:00: **319.427802**

5. Buy and Hold Return of BTC:
   - **2754821.00%**

6. Sharpe Ratio:
   - **0.00399625259188629**

7. Sortino Ratio:
   - **-142353.28949096226**

8. Total Closed Trades:
   - datetime 2018-01-01 15:27:00: **391**
   - datetime 2022-01-30 21:30:00: **391**

9. Number of Winning Trades:
   - datetime 2018-01-01 15:27:00: **31287**
   - datetime 2022-01-30 21:30:00: **31287**

10. Number of Losing Trades:
   - datetime 2018-01-01 15:27:00: **30459**
   - datetime 2022-01-30 21:30:00: **30459**

11. Average Winning Trade (in USDT):
   - datetime 2018-01-01 15:27:00: **5.025105**
   - datetime 2022-01-30 21:30:00: **5.025105**

12. Average Losing Trade (in USDT):
   - datetime 2018-01-01 15:27:00**: 0.759185**
   - datetime 2022-01-30 21:30:00: **0.759185**

13. Largest Winning Trade (in USDT): datetime

2018-01-01 15:27:00 **133660.666667**

2022-01-30 21:30:00 **133660.666667**

14. Largest Losing Trade (in USDT): datetime

2018-01-01 15:27:00 **195.411394**

2022-01-30 21:30:00 **195.411394**

15. Average Holding Duration per Trade:

1.617416339139858e-05 days

16. Max Dip: datetime

2018-01-01 15:27:00 -**192.94115**

2022-01-30 21:30:00 **-192.94115**

17. Average Dip: datetime

2018-01-01 15:27:00 **-99.966529**

2022-01-30 21:30:00 **-99.966529**

Conclusion

The developed trading model showcases promising results, achieving a 68.04% accuracy and robust performance metrics across various strategies. With a total of 391 trades, the model exhibits a notable win rate and efficient risk management, evidenced by the maximum drawdown of 99.56%.

The integration of mean reversion signals, investment doubling, and moving average strategies contributes to the model's versatility in different market conditions. Notably, the model's ability to adapt to varying trends positions it as a reliable tool for cryptocurrency trading.

However, it's crucial to acknowledge the model's limitations and potential areas for improvement. Incorporating sentiment analysis and exploring reinforcement learning techniques could further enhance the model's adaptability and decision-making capabilities.

In summary, the current model lays a solid foundation for algorithmic trading in the cryptocurrency market, emphasizing adaptability, risk management, and strategic diversity. Continuous refinement and exploration of advanced methodologies will be key to unlocking its full potential.

Future Scope

1. **Sentiment Analysis Integration**

   - Incorporating sentiment analysis from relevant sources can enhance the model's predictive power.

   - Sentiment analysis provides insights into market sentiment, improving decision-making.

2. **Reinforcement Learning Implementation**

   - Exploring reinforcement learning techniques enables adaptive strategies.

   - Learning from past decisions and dynamically adjusting trading parameters can optimize performance.

3. **Advanced Feature Engineering**

   - Continuously refining and expanding the set of features improves the model's understanding of market dynamics.

   - Exploration of novel indicators or data sources contributes to more accurate trading signal generation.

References

1. Autocorrelation and Time Series Methods | STAT 462:

https://online.stat.psu.edu/stat462/node/188/

2. Autocorrelation in Trading: A Practical Python Approach to Analysing Time Series Data:

https://blog.quantinsti.com/autocorrelation/

Plagiarism Report

report

ORIGINALITY REPORT

2%	2%	1%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ebin.pub Internet Source	2%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		