Algorithmic Trading Summary Document

BTCUSDT and ETHUSDT

Team ID: 32

December 5, 2024

Contents

1 Logic Behind the Algorithms for ETH and BTC

The algorithms for BTC and ETH are based on a predictive framework utilizing historical data and statistical relationships. We made separate models for BTC and ETH on separate time frames.

1.1 Strategy for ETH:

We predicted future returns through linear regression model. We used linear regression to ensure explainability and interpretability. We created different features for 1 hr timeframe and then selected from them by plotting against future return to capture the relations between different features and the target variable i.e. future return.

We also used different statiscal tests to check the significance of the features which includes t-test, correlation matrix, etc. Further we used PCA to reduce the dimensionality of the data and then used the reduced features to predict the shifted return and finally came down to following features:

- open price
- close price
- returns square
- returns
- z-score
- high low square
- RSI
- chikou span

1.2 Strategy for BTC:

1.2.1 Strategy for Long Periods(1 day)

Initially, similar to ETH, we experimented with linear regression; however, it did not yield promising returns. Subsequently, we employed Decision Trees to generate positions (long, neutral, short), which were then utilized to create trading signals. The strategy balance generated from these signals closely matched the benchmark returns.

To enhance performance, we incorporated the Decision Tree's output as an additional feature in the linear regression model. This combination significantly reduced drawdowns, increased the Sharpe ratio, and improved overall strategy returns. Furthermore, by integrating Heikin-Ashi (HA) candles into the analysis, the strategy successfully outperformed the benchmark returns.

Statistical tests were conducted to evaluate the significance of the features, ensuring robustness in feature selection. The final features used in the linear regression model were as follows:

- Logarithm of HA close values
- Logarithm of HA open values
- Logarithm of HA high values
- Logarithm of HA low values
- Suggested positions derived from the Decision Tree

1.2.2 Strategy for Short Periods (1 Hour)

We generated trading signals by employing linear regression and feature selection techniques, experimenting with various permutations of features to identify the most impactful combinations. These signals aim to capture market dynamics effectively within the short time frame.

Ensemble of Signals

We have merged multiple signals into one final signal based on a custom averaging strategy. The intuition of this is to avoid overfitting in a specific time frame, and to combine these we used the following features:

- **Precedence Vote of Signals:** This is the case that we have multiple signals; we check that if one signal is dominant (or strong) in one direction, then one can reliably use that signal. This precedence is ensembled.
- **Average Vote of Signals:** For the cases where there is no clear preference for which signal to prefer, we use the average of signals to make the final signal.

We have also ensured that during this merging there is no bias towards short/ long trades.

2 Risk Management Practices Incorporated

Effective risk management practices were crucial to safeguarding capital and ensuring the robustness of the strategies. The following measures were implemented:

- **ATR-based Stop-Loss Mechanisms:** Stop-loss levels were dynamically set using the Average True Range (ATR) indicator. This method adapts the stop-loss to the market's volatility:
  - The stop-loss distance was calculated as a multiple of the ATR, providing a flexible threshold that adjusts to changing volatility. A larger ATR in volatile markets results in a wider stop-loss, while lower ATR in calmer markets narrows the stop-loss.
  - This approach ensured that the stop-loss levels were not arbitrary but tailored to market conditions, minimizing the likelihood of being prematurely stopped out during natural price fluctuations.
- **Hidden Markov Models (HMM):** Hidden Markov Models were employed to classify market states into high-volatility and low-volatility regimes. By identifying these states, the strategy dynamically adjusted its trading approach:
  - In high-volatility states, risk exposure was minimized to prevent large losses.
  - In low-volatility states, the strategy allowed for slightly larger position sizes, as the risk of sudden adverse price movements was reduced.
- **Drawdown Analysis:** Regular monitoring of drawdowns was conducted to evaluate the strategy's performance under adverse conditions.
  - Drawdowns were tracked to assess the strategy's risk exposure and ensure that losses were within acceptable limits.
  - We analyzed both average and maximum drawdowns to understand the strategy's risk profile and make informed decisions. We decided leverage based on the maximum drawdown such that we can maximize profit without going bankrupt.
  - Recovery metrics, such as time-to-recover (the period required to return to a previous peak), were analyzed to ensure the strategy's resilience.

- **Threshold as a function of market condition:** We designed our threshold for triggering long or short signals to adapt dynamically to market conditions. During periods of high volatility, the threshold is set higher to allow trades only above this elevated level.

These practices collectively ensured that the strategy maintained a strong risk-adjusted performance and protected the portfolio against significant losses.

3 Crucial Learning Point During Problem Statement Preparation

The most valuable insights gained during the preparation process was to understand the significant impact of correlated features on our model. During the early stage, we were only focussing on feature engineering and model selection. However, after a lot of analysis we realised that correlated features were causing multicollinearity in our model. It was crucial to identify and remove these features to improve the model's performance. To do this, we did the following:

- **Correlation Analysis:** We calculated the correlation matrix of all features.
- **Cluster Analysis:** We used hierarchical clustering to group highly correlated features above a certain threshold.
- **PCA Analysis:** On each cluster, we performed PCA to reduce the dimensionality and remove redundant features.
- **Cross Validation:** We used cross-validation to evaluate the model's performance after removing correlated features by doing hypertuning.
- **Model Selection:** We selected the model with the best performance and used it for our final strategy.

This document concisely outlines the logic, risk management practices, and critical insights behind the development of our BTCUSDT and ETHUSDT trading strategies.